

**RANCANG BANGUN SISTEM PEMANTAUAN CUACA *REAL-TIME*  
BERBASIS *RASPBERRY PI 5* DENGAN INTEGRASI MODEL  
*MOBILENETV2*, *CUSTOM CNN*, DAN *YOLOV5* UNTUK KLASIFIKASI  
AWAN, CUACA, DAN *ESTIMASI VISIBILITY* BERBASIS CITRA  
KAMERA TERINTEGRASI *WEBSITE***

(Skripsi)

Oleh :

**Made Jaya Setiawan  
2117041004**



**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM  
UNIVERSITAS LAMPUNG  
BANDAR LAMPUNG  
2025**

## ABSTRAK

### **RANCANG BANGUN SISTEM PEMANTAUAN CUACA *REAL-TIME* BERBASIS *RASPBERRY PI 5* DENGAN INTEGRASI MODEL *MOBILENETV2*, *CUSTOM CNN*, DAN *YOLOV5* UNTUK KLASIFIKASI AWAN, CUACA, DAN *ESTIMASI VISIBILITY* BERBASIS CITRA KAMERA TERINTEGRASI *WEBSITE***

Oleh

**Made Jaya Setiawan**

Penelitian ini bertujuan untuk merancang dan membangun sistem pemantauan cuaca *real-time* berbasis *Raspberry Pi 5* dengan integrasi model *MobileNetV2*, *Custom CNN*, dan *YOLOv5*. Sistem ini memanfaatkan citra yang ditangkap oleh kamera untuk secara otomatis mengklasifikasikan jenis awan, kondisi cuaca, serta mengestimasi *visibility*. Seluruh hasil klasifikasi dan estimasi ditampilkan melalui antarmuka *website* yang terhubung ke jaringan lokal, sehingga memudahkan pengguna dalam mengakses dan memantau informasi cuaca secara langsung dari berbagai perangkat. Estimasi *visibility* dilakukan dengan menggabungkan metode deteksi objek berbasis *YOLOv5* dan pendekatan *Laplacian Variance*, yang menganalisis kontras warna, ketajaman detail, serta tingkat keburaman citra. Hasil pengujian menunjukkan bahwa akurasi deteksi objek menggunakan model *YOLOv5* dalam mendukung estimasi *visibility* mencapai 95% bila dibandingkan dengan data acuan dari BMKG. Untuk klasifikasi cuaca, model *Custom CNN* yang dikembangkan mampu mencapai akurasi sebesar 70%, menandakan efektivitasnya dalam mengenali kondisi cuaca berbasis citra secara *real-time*. Sementara itu, model *MobileNetV2* yang digunakan untuk klasifikasi awan hanya memperoleh akurasi sebesar 10%, yang diduga disebabkan oleh ketidaksesuaian antara data pelatihan dan kondisi awan di lapangan. Meski demikian, *MobileNetV2* tetap digunakan karena efisiensi komputasinya yang sesuai untuk perangkat IoT seperti *Raspberry Pi*. Dengan integrasi antara sistem pemrosesan citra dan antarmuka *website* lokal yang informatif, sistem ini mampu memberikan solusi pemantauan cuaca mandiri yang efisien, cepat, dan mudah diakses oleh pengguna dalam lingkungan jaringan terbatas.

**Kata kunci:** *Raspberry Pi 5*, *MobileNetV2*, *Custom CNN*, *YOLOv5*.

## ***ABSTRACT***

### **DESIGN AND DEVELOPMENT OF A REAL-TIME WEATHER MONITORING SYSTEM BASED ON RASPBERRY PI 5 WITH INTEGRATION OF MOBILENETV2, CUSTOM CNN, AND YOLOV5 MODELS FOR CLOUD AND WEATHER CLASSIFICATION AND VISIBILITY ESTIMATION USING CAMERA IMAGE INTEGRATED WITH WEBSITE**

**By**

**Made Jaya Setiawan**

This study aims to design and develop a real-time weather monitoring system based on the Raspberry Pi 5, integrating MobileNetV2, a Custom CNN, and YOLOv5 models. The system utilizes images captured by a camera to automatically classify cloud types, weather conditions, and estimate visibility. All classification and estimation results are displayed through a web-based interface connected to a local network, enabling users to conveniently access and monitor weather information directly from various devices. Visibility estimation is performed by combining object detection using YOLOv5 and the Laplacian Variance approach, which analyzes color contrast, detail sharpness, and image blurriness. Experimental results show that the object detection model YOLOv5 achieved an accuracy of 95% in supporting visibility estimation when compared with reference data from BMKG (Indonesian Meteorological Agency). For weather classification, the developed Custom CNN model reached an accuracy of 70%, indicating its effectiveness in recognizing weather conditions from images in real-time. Meanwhile, the MobileNetV2 model used for cloud classification achieved only 10% accuracy, presumably due to a mismatch between the training dataset and actual cloud conditions in the field. Nevertheless, MobileNetV2 is still employed due to its computational efficiency, which suits implementation on IoT devices such as the Raspberry Pi. With the integration of image processing and an informative local web interface, this system offers an efficient, fast, and accessible standalone weather monitoring solution for environments with limited network connectivity.

**Keywords:** Raspberry Pi 5, MobileNetV2, Custom CNN, YOLOv5.

**RANCANG BANGUN SISTEM PEMANTAUAN CUACA *REAL-TIME*  
BERBASIS *RASPBERRY PI 5* DENGAN INTEGRASI MODEL  
*MOBILENETV2, CUSTOM CNN, DAN YOLOV5* UNTUK KLASIFIKASI  
AWAN, CUACA, DAN *ESTIMASI VISIBILITY* BERBASIS CITRA  
KAMERA TERINTEGRASI *WEBSITE***

Oleh :

**Made Jaya Setiawan**

**Proposal Hasil**

**Sebagai Salah Satu Syarat Untuk Mancapai Gelar  
SARJANA SAINS**

**Pada**

**Jurusan Fisika  
Fakultas Matematika Dan Ilmu Pengetahuan Alam**



**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM  
UNIVERSITAS LAMPUNG  
BANDAR LAMPUNG  
2025**

Judul Skripsi

: **Rancang Bangun Sistem Pemantauan Cuaca  
Real- Time Berbasis Raspberry Pi 5 Dengan  
Integrasi Model Mobilenetv2, Custom Cnn, Dan  
Yolov5 Untuk Klasifikasi Awan, Cuaca, Dan  
Estimasi Visibility Berbasis Citra Kamera  
Terintegrasi Website**

Nama

: **Made Jaya Setiawan**

Nomer Induk Mahasiswa

: **2117041004**

Jurusan

: **Fisika**

Bidang Keahlian

: **Instrumentasi**

Fakultas

: **Matematika Dan Ilmu Pengetahuan Alam**



1. komisi Pembimbing

**Dr. Junaidi, S.Si., M.Sc.**  
NIP . 198206182008121001

**Rudi Harianto, S.Kom, M.Si**  
NIP. 198101292008011004

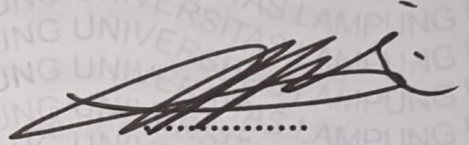
2 Ketua Jurusan Fisika

**Arif Surtono, S.Si, M.Si., M.Eng**  
NIP.197109092000121001

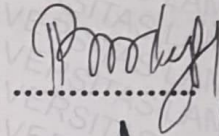
**MENGESAHKAN**

**1. Tim Penguji**

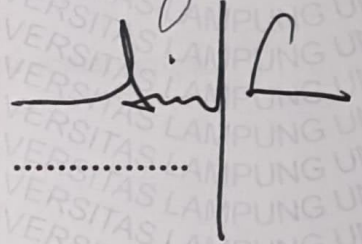
**Ketua : Dr. Junaidi, S.Si., M.Sc.** .....



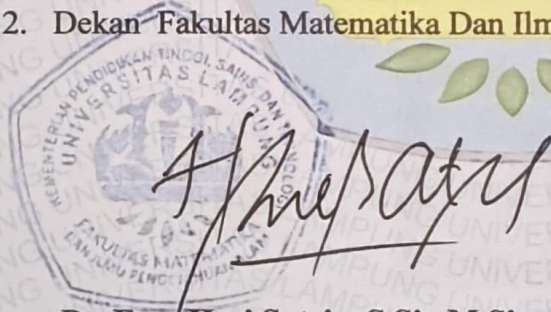
**Sekretaris : Rudi Harianto, S.Kom, M.Si** .....



**Penguji  
Bukan Pembimbing : Arif Surtono, S.Si, M.Si., M.Eng** .....



**2. Dekan Fakultas Matematika Dan Ilmu Pengetahuan Alam**



**Dr. Eng. Heri Satria, S.Si., M.Si.  
NIP. 197110012005011002.**

**Tanggal Lulus Ujian Skripsi : 2 Juli 2025**

## PERNYATAAN

Dengan ini saya menyatakan bahwa skripsi ini tidak terdapat karya yang pernah dilakukan orang lain dan sepengetahuan saya tidak ada karya atau pendapat yang ditulis atau di terbitkan oleh orang lain kecuali yang secara tertulis tercantum dalam naskah ini sebagaimana di sebutkan dalam daftar Pustaka. Selain itu, saya menyatakan pula bahwa skripsi ini dibuat oleh saya sendiri

Apabila pernyataan saya tidak benar maka saya bersedia dikenai sangsi sesuai dengan hukum yang berlaku

Bandar lampung, 11 juni 2025



**Made Jaya Setiawan**  
NPM. 2117041004

## RIWAYAT HIDUP



Penulis yang bernama Made Jaya Setiawan lahir di Rejo Binangun, Kecamatan Raman Utara, Kabupaten Lampung Timur, pada tanggal 17 April 2003. Penulis merupakan anak pertama dari dua bersaudara. Pendidikan dasar ditempuh di SD Negeri 2 Rejo Binangun dan berhasil diselesaikan pada tahun 2015. Selanjutnya, penulis melanjutkan pendidikan di SMP Negeri 1 Raman Utara dan lulus pada tahun 2018. Masa pendidikan menengah atas dilanjutkan di SMA Negeri 1 Raman Utara dan diselesaikan pada tahun 2021. Sejak kecil, penulis memiliki ketertarikan pada bidang sains dan teknologi, yang kemudian menjadi motivasi utama dalam menentukan arah pendidikan selanjutnya.

Pada tahun 2021, penulis diterima sebagai mahasiswa Program Studi Fisika di Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Lampung, melalui jalur Seleksi Nasional Masuk Perguruan Tinggi Negeri (SNMPTN). Selama masa perkuliahan, penulis aktif dalam berbagai kegiatan organisasi dan pengembangan diri. Penulis tergabung dalam Himpunan Mahasiswa Fisika (HIMAFI) di bidang Sosial Masyarakat, serta aktif di Unit Kegiatan Mahasiswa (UKM) Catur dan UKM Hindu di bidang Dana Usaha. Dalam pengembangan akademik, penulis pernah mengikuti lomba Kontes Robot Terbang Indonesia (KRTI) tingkat wilayah pada kategori Racing Plane, serta meraih prestasi dalam lomba karya tulis ilmiah tingkat fakultas. Penulis juga mengikuti program Studi Independen Bersertifikat (MSIB) di Dicoding Indonesia dengan fokus pada pengembangan front-end dan back-end, serta berpartisipasi dalam program Merdeka Belajar Kampus Merdeka (MBKM) bidang penelitian. Saat ini, penulis tengah menyelesaikan tugas akhir

dengan judul **"Rancang Bangun Sistem Pemantauan Cuaca Real-Time Berbasis Raspberry Pi 5 dengan Integrasi Model MobileNetV2, Custom CNN, dan YOLOv5 untuk Klasifikasi Awan, Cuaca, dan Estimasi Visibility Berbasis Citra Kamera Terintegrasi Website."** Skripsi ini merupakan wujud dari ketertarikan penulis pada integrasi antara teknologi Internet of Things (IoT) dan kecerdasan buatan dalam bidang pemantauan lingkungan

## **MOTTO**

Hidup Itu Adalah Sebuah Perjalanan Yang Perlu Di Perjuangkan

“Salah Satu Cara Melakukan Pekerjaan Hebat Adalah Dengan Mencintai Apa  
Yang Kamu Lakukan “  
~**Steve Jobs**~

“Bermimpilah Dalam Hidup , Jangan Hidup Dalam Mimpi”  
~**Andrea Hirata**~

“Anda Mungkin Bisa Menunda Tapi Waktu Tidak Bisa Menunggu”  
~**Benjamin Franklin**~

## **PERSEMBAHAN**

Dengan Rasa Syukur Kepada Ida Sang Hyang Widhi Wasa

Saya Pesembahkan Karya Ini Kepada:

### **Kedua Orang Tua Tercinta**

#### **Bapak Wayan Nada Sukarya dan Ibu Made Rusni**

atas segala doa, kasih sayang, pengorbanan, dan dukungan tiada henti yang selalu menjadi sumber kekuatan dan inspirasi dalam perjalanan hidup dan pendidikan penulis.

#### **Bapak dan Ibu Guru serta Dosen,**

yang telah memberikan ilmu, bimbingan, dan motivasi selama masa belajar, sehingga penulis dapat menuntaskan pendidikan dengan baik.

#### **Kakak tercinta, Wayan Putri Yanti**

yang selalu memberikan semangat, dukungan, serta menjadi contoh teladan bagi penulis dalam menjalani setiap proses dan tantangan hidup.

**Rekan Rekan,** Fisika FMIPA Unila Angkatan 2021

Almamater Tercinta

**“UNIVERSITAS LAMPUNG”**

## KATA PENGATAR

Puji syukur kehadiran Allah SWT atas segala rahmat dan karunia-Nya sehingga penulisan laporan dengan judul **“Rancang Bangun Sistem Pemantauan Cuaca Real-Time Berbasis Raspberry Pi 5 dengan Integrasi Model MobileNetV2, Custom CNN, dan YOLOv5 untuk Klasifikasi Awan, Cuaca, dan Estimasi Visibility Berbasis Citra Kamera Terintegrasi Website”** dapat diselesaikan dengan baik.

Laporan ini disusun sebagai salah satu bentuk pemenuhan kewajiban akademik. Penulis menyadari sepenuhnya bahwa laporan ini masih terdapat kekurangan. Oleh karena itu, saran dan kritik yang membangun sangat diharapkan untuk kesempurnaan laporan ini di masa yang akan datang.

Semoga laporan ini dapat memberikan manfaat bagi pembaca serta menjadi bahan referensi di kemudian hari.

Bandar Lampung, 11 Juni 2025  
Penulis

Made Jaya Setiawan  
NPM. 2117041004

## SANWACANA

Puji syukur ke hadapan Ida Sang Hyang Widhi Wasa, Tuhan Yang Maha Esa, atas segala rahmat, anugerah, dan tuntunan-Nya sehingga penulisan laporan ini dapat diselesaikan dengan baik.

Pada kesempatan ini, penulis menyampaikan rasa terima kasih dan penghargaan yang sebesar-besarnya kepada:

1. Bapak Dr. Junaidi S.Si. M.Sc. selaku Dosen Pembimbing I, yang telah dengan sabar memberikan bimbingan, arahan, koreksi, serta motivasi selama proses perencanaan, pelaksanaan penelitian, hingga penyusunan laporan ini sehingga dapat terselesaikan dengan baik.
2. Bapak Rudi Harianto, S.Kom, M.Si selaku Dosen Pembimbing II, atas segala waktu, masukan, saran, dan ilmu yang diberikan, yang sangat membantu penulis dalam memahami permasalahan dan menyelesaikan laporan ini secara sistematis.
3. Bapak Arif Surtono S.Si. M.Si. M.Eng. selaku Dosen Penguji, atas kritik, koreksi, serta saran yang membangun dalam rangka penyempurnaan hasil penelitian dan penulisan laporan ini.
4. Bapak Dr. Heri Satria, S.Si., M.Si.. selaku Dekan FMIPA Universitas Lampung, atas fasilitas, dukungan, dan kesempatan yang diberikan kepada penulis selama menempuh pendidikan.
5. Seluruh dosen Program Studi Fisika FMIPA Universitas Lampung, yang telah berbagi ilmu, pengalaman, dan pembelajaran berharga selama masa studi, yang menjadi bekal penting dalam penyelesaian laporan ini.
6. Kedua orang tua tercinta, atas doa, kasih sayang, pengorbanan, serta dukungan moral maupun material yang tiada henti mengiringi setiap langkah perjalanan penulis.

7. Anis, Inayah, Lulu, Dinda, Gusti, Ega, Armel, dan semua pihak yang telah memberikan bantuan, semangat, kebersamaan, dan dukungan dalam menyelesaikan penelitian dan penulisan laporan ini hingga tuntas.
8. Mbak Farah, Mbak Ana, Kak Adi, Ibu Damil, dan seluruh staf BMKG, saya ucapkan terima kasih yang sebesar-besarnya atas bimbingan, dukungan, dan kesempatan yang telah diberikan, yang menjadi pengalaman berharga dalam memperluas wawasan di bidang meteorologi dan pemantauan cuaca.

Penulis menyadari bahwa dalam laporan ini masih terdapat kekurangan. Oleh karena itu, saran dan kritik yang membangun sangat diharapkan untuk penyempurnaan di masa mendatang. Semoga laporan ini dapat memberikan manfaat dan menjadi kontribusi kecil dalam pengembangan ilmu pengetahuan.

Bandar Lampung, Juni 2025  
Penulis

**Made Jaya Setiawan**  
**NPM. 2117041004**

## DAFTAR ISI

Halaman

|                       |             |
|-----------------------|-------------|
| <b>SANWACANA.....</b> | <b>xiii</b> |
|-----------------------|-------------|

|                        |           |
|------------------------|-----------|
| <b>DAFTAR ISI.....</b> | <b>xv</b> |
|------------------------|-----------|

|                           |            |
|---------------------------|------------|
| <b>DAFTAR GAMBAR.....</b> | <b>xix</b> |
|---------------------------|------------|

|                           |            |
|---------------------------|------------|
| <b>DAFTAR TABEL .....</b> | <b>xxi</b> |
|---------------------------|------------|

### **I. PENDAHULUAN**

|                             |   |
|-----------------------------|---|
| 1.1 Latar Belakang.....     | 1 |
| 1.2 Rumusan Masalah .....   | 4 |
| 1.3 Tujuan Penelitian.....  | 5 |
| 1.4 Manfaat Penelitian..... | 5 |
| 1.5 Batasan Masalah.....    | 6 |

### **II. TINJAUAN PUSTAKA**

|                                                                 |    |
|-----------------------------------------------------------------|----|
| 2.1 Penelitian Terkait.....                                     | 7  |
| 2.2 Teori Dasar .....                                           | 13 |
| 2.2.1 Cuaca .....                                               | 13 |
| 2.2.2 Pengolahan Citra .....                                    | 15 |
| 2.2.3 <i>Algoritma Convolutional Neural Network (CNN)</i> ..... | 15 |
| 2.2.3.1 Convolutional layer .....                               | 15 |
| 2.2.3.2 MaxPooling Layer .....                                  | 18 |
| 2.2.3.2.1 <i>Flatten</i> .....                                  | 18 |
| 2.2.3.2.2 <i>Dropout</i> .....                                  | 19 |
| 2.2.3.2.3 <i>Fully Connected Layer</i> .....                    | 20 |
| 2.2.3.2.4 <i>Fungsi Softmax</i> .....                           | 21 |
| 2.2.4 <i>Internet of Things (IoT)</i> .....                     | 21 |
| 2.2.5 <i>Python</i> .....                                       | 22 |
| 2.2.6 <i>Hypertext Transfer Protocol (HTTP)</i> .....           | 23 |
| 2.2.7 <i>Deep Learning</i> .....                                | 24 |
| 2.2.8 <i>Edge Computing</i> .....                               | 26 |
| 2.2.9 <i>MobileNetV2</i> .....                                  | 27 |
| 2.2.10 <i>Raspberry Pi</i> .....                                | 29 |
| 2.2.11 <i>YOLO</i> .....                                        | 30 |
| 2.2.11.1 Backbone .....                                         | 31 |

|                                                             |    |
|-------------------------------------------------------------|----|
| 2.11.2 Neck.....                                            | 31 |
| 2.2.11.3 .....                                              | 32 |
| 2.2.12 <i>Visibility</i> .....                              | 33 |
| 2.2.13 <i>API (Application Programming Interface)</i> ..... | 34 |
| 2.2.14 <i>Website</i> .....                                 | 36 |
| 2.2.15 <i>OpenCV</i> .....                                  | 37 |
| 2.2.16 Flask .....                                          | 39 |
| 2.2.17 Filter Laplacian.....                                | 40 |
| 2.2.18 Awan.....                                            | 44 |

### III.METODE PENELITIAN

|                                                                |    |
|----------------------------------------------------------------|----|
| 3.1 Tempat dan Waktu Penelitian .....                          | 47 |
| 3.2 Alat dan Bahan .....                                       | 47 |
| 3.3 Prosedur Penelitian.....                                   | 48 |
| 3.3.1 Studi Literatur.....                                     | 50 |
| 3.3.2 Perancangan Sistem Deteksi.....                          | 50 |
| 3.4 Perancangan Sistem <i>Hardware</i> .....                   | 51 |
| 3.5 Rancang Sistem Dalam Bentuk 3D .....                       | 53 |
| 3.6 Komponen Alat Klasifikasi Cuaca .....                      | 54 |
| 3.5.1 <i>Raspberry Pi 5</i> .....                              | 55 |
| 3.5.2 Kamera Board Rev 1.3 .....                               | 55 |
| 3.5.3 PC .....                                                 | 56 |
| 3.5.4 Power Bank .....                                         | 56 |
| 3.7 Pengumpulan Data .....                                     | 56 |
| 3.7.1 Sumber Data .....                                        | 57 |
| 3.7.2 Preprocessing Data.....                                  | 57 |
| 3.7.3 Augmentasi Data .....                                    | 57 |
| 3.8 Arsitektur Model Cnn MobileNetV2 .....                     | 58 |
| 3.8.1 Proses Training Dan Evaluasi Model CNN MobileNetV2 ..... | 59 |
| 3.8.2 Implentasi Transfer Learning dengan MobileNetV2 .....    | 60 |
| 3.9 Arsitektur Custom CNN.....                                 | 60 |
| 3.9.1 Arsitektur Model Custom CNN .....                        | 60 |
| 3.9.2 Proses Training dan Evaluasi Model Custom CNN.....       | 62 |
| 3.9.3 Implementasi Model Custom CNN .....                      | 62 |
| 3.10 Model YoloV5.....                                         | 62 |
| 3.10.1 Proses Taining dan Evaluasi Model YoloV5 .....          | 63 |
| 3.10.2 Implementasi Model YoloV5.....                          | 64 |
| 3.11 Perancangan Software .....                                | 64 |
| 3.11.1 Arsitektur Sistem Software .....                        | 65 |
| 3.11.2 Bahasa Pemograman Dan Framewark Software .....          | 65 |
| 3.11.3 Alur Kerja Sistem Software.....                         | 66 |
| 3.11.3.1 Membangun Server .....                                | 67 |
| 3.11.3.2 Membangun Front end web .....                         | 67 |
| 3.11.4 Keamanan dan Kendala.....                               | 68 |
| 3.12 Pengujian Sistem .....                                    | 68 |
| 3.12.1 Pengujian Perangkat Keras .....                         | 69 |
| 3.12.2 Pengujian Akuisisi Data .....                           | 69 |
| 3.12.3 Pengujian Pemrosesan Data .....                         | 69 |

|                                           |    |
|-------------------------------------------|----|
| 3.12.4 Pengujian Antar Muka Website ..... | 69 |
| 3.12.5 Evaluasi Dan Analisis Hasil .....  | 70 |
| 3.14 Grafik dan Visualisasi Data .....    | 70 |
| 3.14.1 Grafik Evaluasi Model .....        | 71 |

#### IV HASIL DAN PEMBAHASAN

|                                                               |     |
|---------------------------------------------------------------|-----|
| 4.1 Realisasi Alat .....                                      | 73  |
| 4.2 Pengolahan Dataset Awan Untuk Model CNN MobileNetV2 ..... | 75  |
| 4.2.1 Preprocessing Dataset Awan.....                         | 77  |
| 4.2.1.1 Agugmentasi Dataset Awan .....                        | 77  |
| 4.2.1.2 Normalisasi Dataset Awan.....                         | 81  |
| 4.3 Pengolahan Dataset Cuaca Untuk Model Custom CNN .....     | 83  |
| 4.3.1 Proses Normalisasi Dataset Cuaca .....                  | 83  |
| 4.3.2 Proses Augmentasi Dataset Cuaca .....                   | 85  |
| 4.4 Membangun Model CNN MobileNetV2.....                      | 87  |
| 4.4.1 Membangun Arsitektur Model MobileNetV2.....             | 88  |
| 4.4.2 Training Model CNN .....                                | 90  |
| 4.4.3 Evaluasi Model CNN MobileNetV2.....                     | 92  |
| 4.5 Membangun Model Custom CNN.....                           | 94  |
| 4.5.1 Membangun Asitektur Custom CNN .....                    | 96  |
| 4.5.2 Arsitektur Model Custom CNN.....                        | 98  |
| 4.6 Model CNN YoloV5 Sebagai Estimasi Visibility.....         | 100 |
| 4.6.1 Mengunduh Model YoloV5 di Github.....                   | 101 |
| 4.6.2 Pemanggilan Model YoloV5 .....                          | 102 |
| 4.6.3 Estimasi Visibility Berdasarkan Laplacian.....          | 103 |
| 4.6.3.1 Konversi Gambar ke Grayscale.....                     | 103 |
| 4.6.3.2 Penghitungan Variansi Laplacian .....                 | 104 |
| 3.6.3.3 Klasifikasi Estimasi Visibility .....                 | 105 |
| 4.7 Implementasi Tiga Model CNN di Raspberry Pi 5 .....       | 106 |
| 4.7.1 Persiapan Model Dan Pengambilan Data .....              | 108 |
| 4.7.1.1 Persiapan Model yang Telah Di Latih .....             | 108 |
| 4.7.1.2 Pengambilan Citra Cuaca Dengan Kamera Raspberry Pi .  | 109 |
| 4.7.1.3 Pembacaan Dan Preprocessing Citra.....                | 110 |
| 4.7.1.4 Menyimpan Dan Menyiapkan Hasil.....                   | 110 |
| 4.7.2 Proses Prediksi Oleh Masing Masing Model.....           | 111 |
| 4.7.2.1 Prediksi Klasifikasi Cuaca.....                       | 111 |
| 4.7.2.2 Prediksi Klasifikasi Awan.....                        | 112 |
| 4.7.2.3 Evaluasi Dan Hasil Prediksi .....                     | 112 |
| 4.7.3 Pengiriman Data Ke Server .....                         | 113 |
| 4.8 Membangun Server Website .....                            | 114 |
| 4.8.1 Deskripsi Umum Server.....                              | 116 |
| 4.8.2 Struktur Program Server .....                           | 116 |
| 4.8.3 Pengaturan Folder Unggah .....                          | 117 |
| 4.8.4 Variabel Data Terbaru .....                             | 118 |
| 4.9 Frontend Halaman Utama Website .....                      | 119 |
| 4.9.1 Struktur dan Elemen Halaman Utama Website.....          | 121 |
| 4.9.2 Disain Tampilan Dan Respositivitas .....                | 122 |

|                                                                  |     |
|------------------------------------------------------------------|-----|
| 4.10 Frontend Tampilan Website Halaman Keterangan Awan .....     | 126 |
| 4.10.1 Struktur HTML Website Halaman Keterangan Awan .....       | 127 |
| 4.10.2 Struktur Style CSS website Halaman Keterangan Awan .....  | 129 |
| 4.10.3 Struktur JavaScript Website Halaman Keterangan Awan ..... | 130 |
| 4.11 Data Hasil Pengamatan .....                                 | 132 |
| 4.11,1 Estimasi Visibility Dengan Model YoloV5 .....             | 132 |
| 4.11.2 Hasil Klasifikasi Cuaca Dengan Model Custom CNN .....     | 136 |
| 4.11.3.1 Hasil Klasifikasi Awan dengan Model MobileNetV2.....    | 139 |
| 4.11.3.2 Ketidak Sesuaian Data Latih Dan Data BMKG.....          | 144 |

## **V. SIMPULAN**

|                    |     |
|--------------------|-----|
| 5.1. Simpulan..... | 145 |
| 5.2 Saran .....    | 146 |

## **DAFTAR PUSTAKA**

## **LAMPIRAN**

## DAFTAR GAMBAR

| Gambar                                                                                                                                                                                              | Halaman |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| 2.1 Hasil Perbandingan Desain Graphical Unit Interface Sistem <i>Fuzzy Inference System (FIS)</i> Dengan <i>Adaptive Neuro-Fuzzy Inference System (ANFIS)</i> .....                                 | 8       |
| 2.2 Prototipe Sistem Klasifikasi Langit Cerah Dan Berawan Menggunakan <i>Metode Gray Level Co-Occurrence Matrices (GLCM)</i> Dan <i>K-Nearest Neighbor (KNN)</i> Berbasis <i>Raspberry Pi</i> ..... | 9       |
| 2.3 Diagram Long Short-Term Memory (LSTM) .....                                                                                                                                                     | 10      |
| 2.4 Arsitektur <i>Convolutional Neural Network (CNN)</i> Berbasis <i>Mobilenet</i> .....                                                                                                            | 11      |
| 2.5 Citra <i>Satelit Himawari 8</i> Dengan Metode <i>Optical Flow</i> .....                                                                                                                         | 12      |
| 2.6 Dataset Keadaan Cuaca Untuk Data Latih <i>Model Custom CNN</i> .....                                                                                                                            | 14      |
| 2.7 Arsitektur CNN .....                                                                                                                                                                            | 16      |
| 2.8 Ilustrasi Perhitungan Konvolusi .....                                                                                                                                                           | 17      |
| 2.9 Ilustrasi <i>Maxpooling</i> .....                                                                                                                                                               | 18      |
| 2.10 (a) Ilustrasi <i>Flutten Tanpa Dropout</i> , (b) Ilustrasi <i>Flutten Dengan Dropout</i> .....                                                                                                 | 20      |
| 2.11 Ilustrasi Fully Connected Layer .....                                                                                                                                                          | 21      |
| 2.12 Alur Kerja Hypertext Transfer Protocol.....                                                                                                                                                    | 24      |
| 2.13 <i>Model Long Short-Term Memory (LSTM)</i> .....                                                                                                                                               | 25      |
| 2.14 Arsitektur <i>MobileNetV2</i> .....                                                                                                                                                            | 28      |
| 2.15 <i>Raspberry Pi 5 Microcomputer Module</i> .....                                                                                                                                               | 29      |

|                                                                                                           |    |
|-----------------------------------------------------------------------------------------------------------|----|
| 2.16 Prediksi <i>Output YOLO</i> .....                                                                    | 30 |
| 2.17 Alur Kerja <i>Application Programming Interface</i> .....                                            | 35 |
| 2.18 Menunjukkan Proses Pengenalan Citra Menggunakan <i>Opencv</i> .....                                  | 38 |
| 2.19 (a) <i>Tampa Filter</i> , (b) <i>Filter Gradien-x</i> , Dan (c) <i>Filter Gradien-y</i> .....        | 41 |
| 2.20 <i>Dataset Gambar Awan Untuk Data Latih Model MobileNetV2</i> .....                                  | 45 |
| 3.1 Diagram Alir Penelitian. ....                                                                         | 49 |
| 3.2 Diagram Blok Sistem Identifikasi Citra Awan. ....                                                     | 51 |
| 3.2 Diagram Blok Sistem Identifikasi Citra Awan. ....                                                     | 52 |
| 3.4 Rancangan Sistem dalam Bentuk 3D.....                                                                 | 54 |
| 3.5 Arsitektur <i>Model CNN MobileNetV2</i> .....                                                         | 59 |
| 3.6 Arsitektur <i>Model Custom CNN</i> .....                                                              | 61 |
| 3.7 Alur Kerja <i>Model YoloV5</i> .....                                                                  | 63 |
| 3.8 Alur Kerja <i>software Website</i> . ....                                                             | 66 |
| 3.9 Rancangan Tampilan <i>Website</i> . ....                                                              | 68 |
| 3.10 Garfik Akurasi Model. ....                                                                           | 72 |
| 3.11 Garfik <i>Loss Model</i> .....                                                                       | 72 |
| 4.1 Realisasi Alat Pemantauan Cuaca.....                                                                  | 73 |
| 4.2 Pelaksanaan Pengujian Alat Pemantauan Cuaca Berbasis Kamera di<br>Lingkungan BMKG Raden Intan II..... | 74 |
| 4.3 <i>Output Augmentasi Dataset Gambar Awan sebagai Data Latih Model<br/>MobilnetV2</i> .....            | 80 |
| 4.5 Peningkatan Jumlah Data Citra Awan Akibat Proses <i>Augmentasi</i> .....                              | 81 |
| 4.7 Proses <i>Training Model Custom CNN</i> pada Dataset Citra Cuaca .....                                | 91 |
| 4.8 (a) grafik akurasi dan (b) <i>loss</i> pelatihan model <i>Custom CNN</i> . ....                       | 92 |
| 4.9 Akurasi dan <i>Loss Training Model Custom CNN</i> .....                                               | 93 |
| 4.10 Proses <i>Training Model CNN MobileNetV2</i> .....                                                   | 94 |

|                                                                                                                                                  |     |
|--------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 4.11 Akurasi dan <i>Loss Training Model Model MobileNetV2</i> .....                                                                              | 98  |
| 4.12 (a) Grafik <i>Training Dan Validation Accuracy</i> , Dan (b) Grafik<br>Training Dan <i>Validation Loss Training Model Mobilenetv2</i> ..... | 99  |
| 4.11 Menjalankan Model YOLOv5 untuk Mengukur <i>Estimasi Visibility</i> .....                                                                    | 100 |
| 4.12 Implementasi Tiga Model CNN ( <i>MobileNetV2, Custom CNN, dan</i><br><i>YOLOv5</i> ) pada <i>Raspberry Pi 5</i> .....                       | 107 |
| 4.13 Menjalankan <i>Server Website</i> .....                                                                                                     | 115 |
| 4.14 User Interface Halaman Utama Website Sistem Pemantauan Cuaca<br>Berbasis Kamera .....                                                       | 120 |
| 4.15 Tampilan Responsif Halaman Utama <i>Website</i> Sistem Klasifikasi<br>Cuaca pada Perangkat <i>Handphone</i> .....                           | 125 |
| 4.16 Antarmuka Halaman Penjelasan Keterangan Jenis Awan dalam<br>Sistem Klasifikasi Cuaca .....                                                  | 127 |
| 4.17 Analisis Perbandingan Estimasi <i>Visibility</i> Alat Pemantauan Cuaca<br>terhadap Data BMKG .....                                          | 134 |
| 4.18 Grafik Kesesuaian Data Estimasi <i>Visibility</i> Terhadap Data BMKG.....                                                                   | 135 |
| 4.19 Analisis Perbandingan Klasifikasi Cuaca Berdasarkan Data Alat<br>dan BMKG .....                                                             | 138 |
| 4.20 Grafik Kesesuaian Klasifikasi Cuaca Pada Alat Terhadap Data<br>BMKG.....                                                                    | 139 |
| 4.17 Tingkat Kecocokan Klasifikasi Awan Sistem Pemantauan dengan<br>Data BMKG. ....                                                              | 142 |
| 4.18 Grafik Kesesuaian Data Klasifikasi Awan Pada Alat Terhadap Data<br>BMKG.....                                                                | 143 |

## DAFTAR TABEL

| Tabel                                                   | Halaman |
|---------------------------------------------------------|---------|
| 3.1 Alat dan bahan penelitian.....                      | 24      |
| 3.2 Data Hasil Analisis Cuaca .....                     | 55      |
| 4.1 Asitektur Model Custom CNN .....                    | 83      |
| 4.2 Hasil Estimasi <i>Visibility (YOLOv5)</i> .....     | 133     |
| 4.3 Hasil Klasifikasi Cuaca ( <i>Custom CNN</i> ) ..... | 137     |
| 4.3 Hasil Klasifikasi Awan ( <i>MobileNetV2</i> ) ..... | 140     |

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Cuaca memiliki pengaruh yang sangat besar dalam kehidupan manusia, baik dalam aktivitas sehari-hari maupun sektor-sektor strategis seperti pertanian, penerbangan, maritim, dan mitigasi bencana (Siregar, 2020). Perubahan cuaca yang tidak terduga dapat menimbulkan dampak signifikan, seperti gagal panen akibat curah hujan yang tinggi, keterlambatan penerbangan akibat badai, atau meningkatnya risiko kecelakaan di laut akibat cuaca ekstrem (Ulya *and* Kamal, 2017). Oleh karena itu, kemampuan untuk memantau dan memprediksi cuaca secara akurat menjadi kebutuhan yang semakin mendesak (Priyahita *et al.*, 2016).

Awan memiliki peran penting dalam menentukan pola cuaca karena dapat memberikan informasi mengenai kemungkinan hujan, badai, atau perubahan atmosfer lainnya (Islamiah *et al.*, 2022). Berdasarkan karakteristiknya, awan diklasifikasikan ke dalam beberapa jenis, seperti *cirrus*, *cumulus*, dan *stratus*, yang masing-masing memiliki pengaruh yang berbeda terhadap kondisi cuaca (Sufy *et al.*, 2017). Klasifikasi awan yang akurat dapat membantu dalam memahami proses meteorologi yang sedang berlangsung, sehingga dapat digunakan sebagai dasar untuk peramalan cuaca yang lebih baik (Suaydhl, 2010).

Perkembangan teknologi kecerdasan buatan (*Artificial Intelligence/AI*), khususnya di bidang *Deep Learning*, telah membawa inovasi signifikan dalam berbagai aspek analisis data, termasuk pengolahan citra (Peramala *et al.*, 2016). Salah satu metode yang terbukti efektif dalam klasifikasi citra adalah *Convolutional Neural Network (CNN)*. CNN telah banyak digunakan dalam berbagai aplikasi,

seperti pengenalan wajah, analisis medis, dan deteksi objek pada sistem kendaraan otonom (Pranjaya *et al*, 2024). Dalam konteks pemantauan cuaca, CNN dapat diterapkan untuk mengklasifikasikan jenis awan dan kondisi cuaca secara otomatis berdasarkan citra yang diambil dari kamera atau sensor yang tersedia (Dewi and Muslikh, 2013).

Namun, salah satu tantangan dalam penerapan *Deep Learning* terletak pada kebutuhan daya komputasi yang tinggi. Model CNN konvensional, seperti *VGGNet*, *AlexNet*, dan *ResNet*, umumnya memerlukan perangkat keras dengan spesifikasi tinggi, termasuk unit pemrosesan grafis atau *server* dengan kapabilitas komputasi besar (Nugroho *et al.*, 2020). Ketergantungan ini menjadi kendala, terutama dalam implementasi di perangkat *edge* atau sistem dengan sumber daya terbatas. Untuk menjawab tantangan tersebut, para peneliti mengembangkan arsitektur CNN yang lebih ringan namun tetap mempertahankan performa klasifikasinya. Salah satu pendekatan yang cukup populer yakni *MobileNet*, yang dirancang dengan efisiensi tinggi sehingga dapat dijalankan pada perangkat dengan keterbatasan daya komputasi tanpa menurunkan akurasi secara signifikan (Akay *et al.*, 2021).

Selain itu, pemanfaatan model *Custom CNN* yang dirancang secara spesifik untuk klasifikasi cuaca, serta integrasi dengan metode *object detection* seperti *YOLO* untuk estimasi jarak pandang (*visibility*), menjadi pendekatan terbaru yang potensial dalam sistem pemantauan cuaca. *Estimasi visibility* berdasarkan deteksi objek seperti pesawat, landasan pacu, dan tower menggunakan kamera, memungkinkan pengukuran jarak pandang secara visual dan *real-time*, yang sangat relevan dalam dunia penerbangan (Thuan, 2021).

Pemanfaatan perangkat keras seperti *Raspberry Pi* juga menjadi kunci dalam pengembangan sistem berbasis *AI* yang hemat biaya. *Raspberry Pi 5*, sebagai generasi terbaru, menawarkan peningkatan performa signifikan dibanding pendahulunya dan cukup kuat untuk menjalankan model *deep learning* ringan serta model deteksi objek secara lokal. *Raspberry Pi* memiliki fleksibilitas tinggi dalam integrasi dengan berbagai sensor dan kamera, serta konsumsi daya rendah,

menjadikannya pilihan ideal untuk sistem pemantauan cuaca *portabel* (Wahab *et al.*, 2021).

Dalam penelitian ini, dikembangkan sebuah sistem pemantauan cuaca *real-time* berbasis *Raspberry Pi 5*, yang mengintegrasikan tiga model CNN dengan fungsi berbeda, *MobileNetV2* untuk klasifikasi jenis awan, *CNN custom* untuk klasifikasi kondisi cuaca, dan *YOLOv5* untuk *estimasi visibility* berdasarkan deteksi objek pada citra kamera. Seluruh sistem didukung oleh antarmuka *website* yang memungkinkan pengguna memantau informasi cuaca secara langsung dan interaktif.

Hasil dari penelitian ini diharapkan dapat memberikan kontribusi nyata bagi dunia meteorologi modern, terutama dalam pengembangan sistem pemantauan cuaca otomatis yang efisien, portabel, dan murah. Selain itu, sistem ini juga dapat diaplikasikan pada pendidikan atmosfer, pemantauan iklim, serta sebagai bagian dari sistem peringatan dini bencana berbasis data visual.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana merancang dan membangun sistem pemantauan cuaca *real-time* berbasis *Raspberry Pi 5* yang mampu melakukan klasifikasi awan, klasifikasi cuaca, dan *estimasi visibility* menggunakan citra kamera?.
2. Bagaimana performa model *MobileNetV2* dalam mengklasifikasikan jenis awan berdasarkan citra kamera secara *real-time*?.
3. Bagaimana efektivitas model *Custom CNN* dalam mengklasifikasikan kondisi cuaca berdasarkan data citra awan yang tersedia?.
4. Bagaimana akurasi model *YOLOv5* dalam mendeteksi objek dan mengestimasi *visibility* dari citra kamera?.
5. Bagaimana sistem antarmuka *website* dapat menampilkan hasil klasifikasi dan estimasi cuaca secara *real-time* dengan tampilan yang informatif dan mudah diakses?.

### 1.3 Tujuan Penelitian

Penelitian ini bertujuan untuk:

1. Merancang dan membangun sistem pemantauan cuaca *real-time* berbasis *Raspberry Pi 5* yang mampu melakukan klasifikasi awan, klasifikasi cuaca, dan *estimasi visibility* menggunakan citra dari kamera.
2. Menerapkan dan mengevaluasi model *MobileNetV2* untuk mengklasifikasikan jenis awan secara akurat dan efisien pada perangkat dengan keterbatasan daya komputasi.
3. Mengembangkan dan menguji performa model *Custom CNN* dalam mengklasifikasikan kondisi cuaca berdasarkan citra awan.
4. Mengimplementasikan model *YOLOv5* untuk mendeteksi objek pada citra kamera dan mengestimasi *visibility* berdasarkan kontras warna, ketajaman detail, serta tingkat keburaman citra sebuah objek yang terdeteksi.
5. Membangun antarmuka *website* sebagai media untuk menampilkan hasil klasifikasi awan, cuaca, dan *estimasi visibility* secara *real-time* dengan aksesibilitas yang mudah bagi pengguna.

### 1.4 Batasan Masalah

1. Sistem dirancang menggunakan *Raspberry Pi 5* sebagai perangkat utama untuk pemrosesan model *AI* dan pengambilan gambar.
2. Pengambilan citra dilakukan secara *real-time* menggunakan kamera statis yang terpasang di lokasi tertentu.
3. Model *MobileNetV2* digunakan khusus untuk klasifikasi jenis awan, sedangkan *Custom CNN* digunakan untuk klasifikasi kondisi cuaca.
4. Model *YOLOv5* digunakan untuk mendeteksi objek di lingkungan sekitar (seperti runway, pesawat, atau bangunan) dan digunakan untuk *estimasi visibility* (jarak pandang).
5. Hasil klasifikasi dan *estimasi visibility* ditampilkan melalui antarmuka *website* lokal, tanpa integrasi langsung ke sistem pemantauan cuaca resmi.

6. Pemantauan kondisi awan dan estimasi visibility dilakukan hanya pada siang hari, saat pencahayaan alami mencukupi untuk menghasilkan citra yang jelas dan akurat.

### 1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

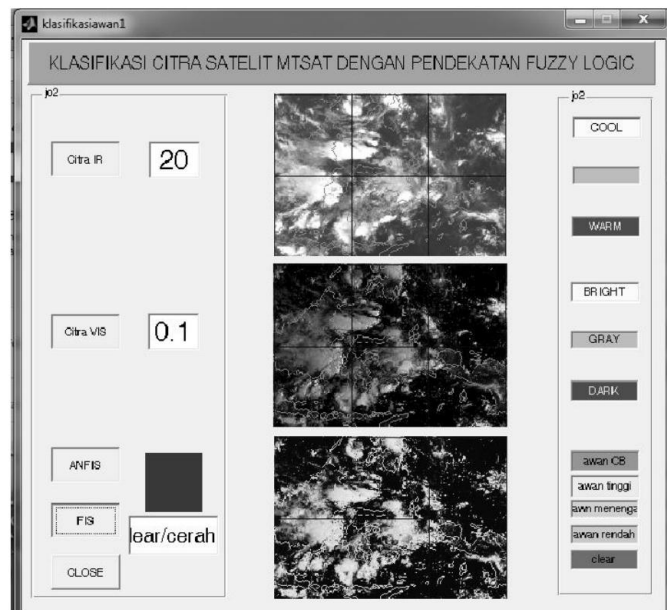
1. Memberikan solusi monitoring cuaca berbasis kamera dan *AI* yang dapat dioperasikan secara mandiri menggunakan perangkat berbiaya rendah seperti *Raspberry Pi 5*.
2. Membantu dalam klasifikasi awan dan cuaca secara otomatis menggunakan kombinasi model CNN, sehingga dapat mempercepat proses pemantauan atmosfer.
3. Menawarkan pendekatan baru dalam *estimasi visibility* (jarak pandang) melalui deteksi objek menggunakan model *YOLOv5*, yang dapat berguna dalam konteks penerbangan dan keselamatan.
4. Mendorong penerapan teknologi *deep learning* di perangkat *edge* seperti *Raspberry Pi 5*, sehingga membuka peluang pemanfaatan *AI* dalam bidang meteorologi di daerah dengan keterbatasan infrastruktur.
5. Menjadi referensi bagi penelitian selanjutnya dalam bidang pemantauan cuaca otomatis, pengolahan citra, dan penerapan *AI* di sistem *IoT* serta dapat digunakan untuk pengembangan sistem serupa di sektor lain.

## **BAB II**

### **TINJAUAN PUSTAKA**

#### **2.1 Penelitian Terkait**

Penelitian yang dilakukan oleh Nardi dan Nazori, pada 2024 membahas secara mendalam mengenai penerapan metode pengelompokan (*clustering*) dalam proses klasifikasi awan, dengan menggunakan data citra satelit yang diperoleh dari satelit MTSAT. Data yang dimanfaatkan dalam penelitian ini mencakup tingkat keabuan (*gray level*) dari kanal citra satelit *visible* dan *infrared*, di mana masing-masing kanal memberikan informasi yang berbeda namun saling melengkapi, kanal gelombang tampak (*visible*) berkaitan erat dengan nilai albedo permukaan awan, sedangkan kanal *infrared* merepresentasikan suhu dari awan yang diamati. Dengan memanfaatkan metode pengelompokan berbasis fungsi jarak, penelitian ini mampu mengelompokkan jenis-jenis awan berdasarkan karakteristik tekstur dan tingkat kecerahan yang terdapat pada citra satelit tersebut. Untuk menguji tingkat akurasi dan keandalan dari hasil klasifikasi yang diperoleh, analisis yang dilakukan kemudian divalidasi dengan membandingkannya terhadap data radar cuaca yang diperoleh dari beberapa wilayah di Indonesia, termasuk Padang, Surabaya, Manado, dan Makassar. Validasi ini bertujuan untuk memastikan bahwa pengelompokan yang dilakukan benar-benar mencerminkan kondisi atmosfer yang sebenarnya, serta dapat diandalkan dalam konteks pemantauan cuaca berbasis pengolahan citra (Nardi and Nazori, 2024).

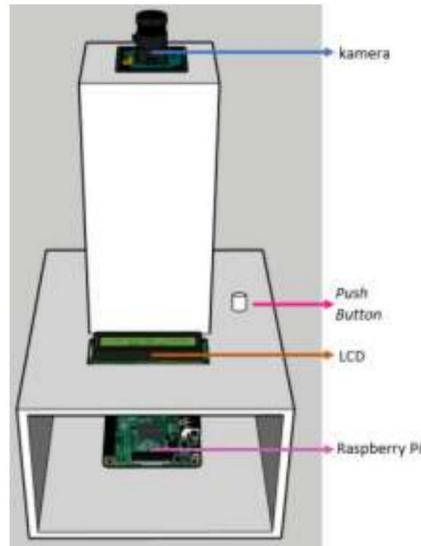


**Gambar 2.1** Hasil Perbandingan Desain Graphical Unit Interface Sistem *Fuzzy Inference System (FIS)* Dengan *Adaptive Neuro-Fuzzy Inference System (ANFIS)* (Nardi and Nazori, 2024).

Pendekatan *fuzzy logic* yang diterapkan dalam penelitian ini memungkinkan klasifikasi awan yang lebih fleksibel dibandingkan dengan metode konvensional. Dengan mempertimbangkan keanggotaan dalam kelompok berdasarkan fungsi jarak, sistem ini dapat mengadaptasi variasi karakteristik awan dalam data citra satelit. Keunggulan dari metode ini terletak pada kemampuannya dalam menangani data yang bersifat tidak pasti, seperti perubahan intensitas kecerahan dan suhu awan yang dinamis.

Penelitian lain yang dilakukan oleh Mawarni *et al*, pada 2021 mengembangkan sistem klasifikasi langit cerah dan berawan menggunakan metode *Gray Level Co-occurrence Matrices (GLCM)* dan *K-Nearest Neighbor (KNN)* berbasis Raspberry Pi. Sistem ini menggunakan citra fisheye untuk memperoleh bidang pandang yang lebih luas dibandingkan dengan citra panoramic. Ekstraksi fitur tekstur dilakukan menggunakan GLCM dengan parameter energy, contrast, correlation, homogeneity, dan dissimilarity pada empat variasi sudut ( $0^\circ$ ,  $45^\circ$ ,  $90^\circ$ , dan  $135^\circ$ ). Proses klasifikasi dilakukan menggunakan KNN dengan nilai  $k = 3, 5, 7$ , dan  $9$ , di

mana *Raspberry Pi 4* digunakan sebagai pusat pemrosesan dan hasil klasifikasi ditampilkan pada LCD (Mawarni *et al*, 2021).



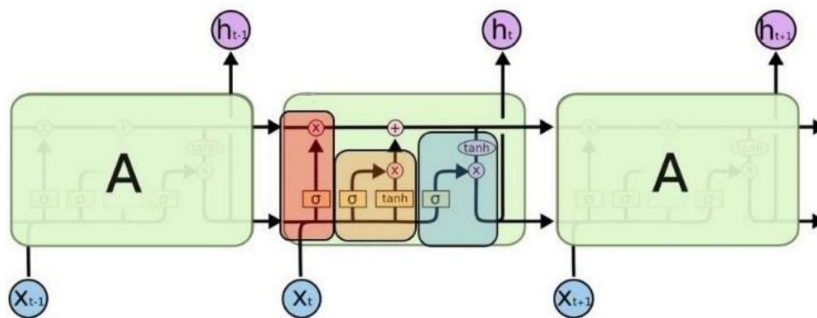
**Gambar 2.2** Prototipe Sistem Klasifikasi Langit Cerah Dan Berawan Menggunakan Metode Gray Level Co-Occurrence Matrices (GLCM) Dan K-Nearest Neighbor (KNN) Berbasis Raspberry Pi (Mawarni *et al.*, 2021).

Hasil pengujian sistem menunjukkan bahwa metode ini memiliki akurasi rata-rata sebesar 100% dengan waktu komputasi rata-rata sebesar 2,3516 detik menggunakan *k-fold* dengan 5 *fold*. Penelitian ini memberikan kontribusi penting dalam klasifikasi kondisi langit dengan memanfaatkan kombinasi teknik ekstraksi fitur tekstur dan algoritma klasifikasi berbasis pembelajaran mesin. Pendekatan ini juga relevan sebagai dasar dalam pengembangan sistem klasifikasi awan berbasis CNN *MobileNetV2* yang diimplementasikan pada *Raspberry Pi* untuk pemantauan cuaca secara *real-time*.

Penelitian yang dilakukan oleh Supriyadi, pada 2019 dari Pusat Meteorologi Maritim BMKG menggunakan metode *deep learning Long Short-Term Memory* (LSTM) untuk memprediksi parameter cuaca berdasarkan data time series. Studi ini memanfaatkan data pengukuran suhu udara, kelembaban udara, kecepatan angin, dan tekanan udara yang dikumpulkan selama bulan Januari dan Februari 2019. Data bulan Januari digunakan untuk pelatihan (*training*) dan pengujian (*test*)

model, sedangkan data bulan Februari digunakan untuk mengevaluasi hasil prediksi. Hasil penelitian menunjukkan bahwa model *LSTM* memberikan performa yang baik dalam memprediksi parameter cuaca, dengan nilai *Root Mean Square Error (RMSE)* yang lebih rendah setelah dilakukan pembaruan (*update*). *RMSE* yang diperoleh untuk parameter suhu, kelembaban, kecepatan angin, dan tekanan udara masing-masing adalah 0,576; 2,8687; 2,1963; dan 1,0647 setelah validasi (Supriyadi, 2019).

Selain itu, prediksi untuk 1 Februari 2019 menghasilkan *RMSE* masing-masing sebesar 1,0337 untuk suhu udara, 6,3413 untuk kelembaban, 2,8934 untuk kecepatan angin, dan 1,4313 untuk tekanan udara. Penelitian ini juga menemukan bahwa nilai *RMSE* parameter suhu dan kelembaban cenderung meningkat seiring waktu, sedangkan parameter kecepatan angin dan tekanan udara menunjukkan pola fluktuasi, dengan penurunan pada hari ketiga dan peningkatan secara kontinu hingga satu bulan ke depan. Hal ini menunjukkan bahwa metode *LSTM* mampu menangkap pola data *time series* dengan baik, terutama untuk parameter cuaca yang memiliki hubungan kompleks antar variabel. Hasil penelitian ini memberikan kontribusi signifikan dalam pengembangan sistem prediksi cuaca berbasis *deep learning*.

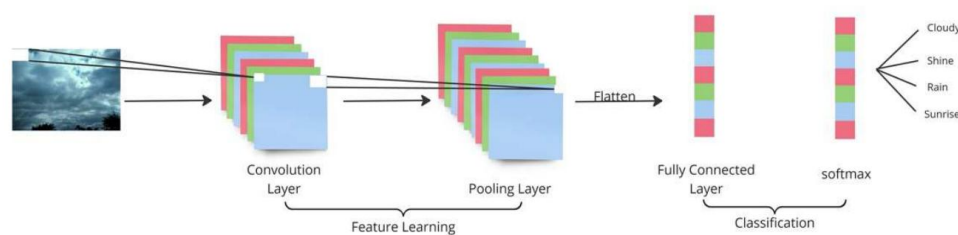


**Gambar 2.3** Diagram *Long Short-Term Memory (LSTM)* (Supriyadi, 2019).

**Gambar 2.3** menunjukkan mekanisme dasar perhitungan pada *Long Short-Term Memory (LSTM)*, sebuah *metode deep learning* yang dirancang untuk data *time series*. Perhitungan dengan metode ini umumnya melibatkan lebih dari satu *layer*, yang menyebabkan proses pengolahan data menjadi lebih lambat seiring bertambahnya jumlah *layer*. Oleh karena itu, *deep learning* ideal dijalankan pada

*Graphics Processing Unit (GPU)*, yang memiliki jumlah *core* lebih banyak dibandingkan dengan *Central Processing Unit (CPU)*. *LSTM* sendiri pertama kali dikembangkan oleh *Hochreiter* dan *Schmidhuber* pada tahun 1997. Setiap modul *LSTM* terdiri dari berbagai operasi komputasi, seperti operasi penambahan, perkalian, penggabungan, duplikasi vektor, serta fungsi matematika seperti *tanh* dan *sigmoid*.

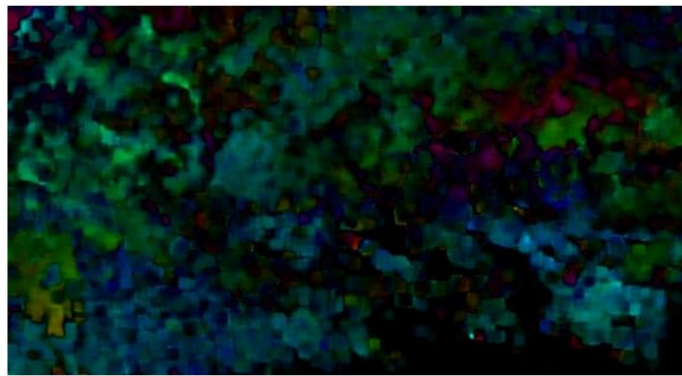
Penelitian yang dilakukan oleh *Aditya et al. (2023)* menyajikan solusi untuk klasifikasi kondisi cuaca dengan memanfaatkan arsitektur *Convolutional Neural Network (CNN)* berbasis *MobileNet*. Tujuannya adalah menghasilkan sistem klasifikasi cuaca yang cepat dan akurat, mengingat pentingnya analisis cuaca yang efisien bagi berbagai sektor seperti pertanian, penerbangan, dan akuakultur. Dataset yang digunakan terdiri dari empat kelas cuaca—berawan, hujan, cerah, dan cerah berawan—yang diperoleh dari platform *Kaggle*. Dataset ini mencakup 1.125 citra digital yang dibagi menjadi 80% data latih dan 20% data uji (*Aditya et al., 2023*).



**Gambar 2.4** Arsitektur *Convolutional Neural Network (CNN)* Berbasis *MobileNet* (*Aditya et al., 2023*).

Penelitian tersebut mengevaluasi berbagai parameter, termasuk ukuran citra (*image size*), *optimizer*, nilai *epoch*, *batch size*, dan *learning rate*. Melalui lima skenario pengujian, parameter optimal yang ditemukan adalah ukuran citra sebesar 224×224 *pixel*, *optimizer Adamax*, *epoch 50*, *batch size 16*, dan *learning rate 0,0001*. Kombinasi parameter ini menghasilkan akurasi sebesar 95,56% dan nilai loss sebesar 0,194, menunjukkan kinerja yang sangat baik dalam klasifikasi cuaca. Studi ini menunjukkan potensi *MobileNet* sebagai model yang efisien dan akurat untuk aplikasi pengklasifikasian kondisi atmosfer menggunakan data citra.

Penelitian yang dilakukan oleh Islah berjudul *Monitoring Data Citra Satelit Himawari 8 dengan Metode Optical Flow* membahas pemantauan fenomena atmosfer *overshooting tops* (OT) menggunakan data citra satelit *Himawari-8* berbasis kecerdasan buatan. OT adalah fenomena di mana puncak awan menembus lapisan *troposfer* hingga mencapai stratosfer bawah, yang dapat menyebabkan cuaca ekstrem seperti angin kencang di permukaan bumi. Penelitian ini menggunakan metodologi yang terdiri dari perumusan masalah dan kajian literatur, pemodelan monitoring OT, pengumpulan data melalui wawancara dan dataset citra satelit, serta implementasi model yang telah dirancang (Islah *et al.*, 2023).



**Gambar 2.5** Citra Satelit Himawari 8 Dengan Metode *Optical Flow* (Islah *et al.*, 2023).

Dalam implementasinya, penelitian ini mengadopsi algoritma *Lucas-Kanade* untuk menghitung *optical flow* pada piksel yang terdeteksi sebagai *corner* serta algoritma *Gunnar Farneback* untuk mendeteksi *dense optical flow*. Pendekatan ini bertujuan untuk memonitor keberadaan OT secara lebih akurat menggunakan citra satelit. Melalui wawancara dengan ahli klimatologi dan pihak BMKG, peneliti berhasil mengumpulkan dataset yang mencakup citra dengan OT dan tanpa OT untuk melatih dan menguji model. Studi ini memberikan kontribusi penting dalam pemantauan cuaca ekstrem di wilayah tropis dengan memanfaatkan teknologi citra satelit dan metode *optical flow*.

## 2.2 Tiori Dasar

## 2.1 Cuaca

Cuaca merupakan kondisi udara yang diamati dalam periode yang relatif singkat atau pendek pada teritori yang terbatas (Siregar, 2020). Unsur-unsur yang terkandung dalam cuaca dan iklim antara lain temperatur udara, tekanan udara, kelembapan udara, angin, awan, hujan dan lain-lain. Perubahan cuaca terjadi karena interaksi antara faktor-faktor tersebut, yang dipengaruhi oleh energi matahari, rotasi bumi, dan karakteristik geografis suatu wilayah. Oleh karena itu, cuaca dapat sangat bervariasi dari satu tempat ke tempat lain, bahkan dalam periode waktu yang sama (Luthfiarta *et al.*, 2020).

Perbedaan utama antara cuaca dan iklim terletak pada skala waktu dan ruang. Cuaca bersifat sementara dan lokal, sedangkan iklim menggambarkan pola cuaca rata-rata di suatu wilayah selama periode waktu yang lebih panjang, biasanya 30 tahun atau lebih. Studi tentang cuaca disebut meteorologi, yang berfokus pada analisis data atmosfer untuk memahami pola, mengidentifikasi anomali, dan membuat prakiraan cuaca. Pemahaman tentang cuaca penting tidak hanya untuk aktivitas harian seperti berpakaian atau merencanakan perjalanan, tetapi juga untuk sektor yang lebih luas seperti pertanian, penerbangan, dan pengelolaan bencana (Handayani, 2010).

Dalam konteks ilmiah dan teknologi, pemantauan dan prediksi cuaca telah berkembang pesat berkat kemajuan dalam alat-alat pengamatan seperti satelit, radar, dan stasiun cuaca otomatis. Data yang dikumpulkan oleh alat-alat ini dianalisis menggunakan model matematika dan *algoritma* komputer untuk menghasilkan prakiraan yang lebih akurat. Selain itu, penggunaan metode berbasis kecerdasan buatan, seperti algoritma *deep learning* dan *machine learning*, semakin meningkatkan kemampuan dalam mengolah data cuaca secara cepat dan efisien, memberikan manfaat yang signifikan bagi berbagai sektor kehidupan.



**Gambar 2.6** *Dataset Keadaan Cuaca Untuk Data Latih Model Custom CNN* (Handayani, 2010).

**Gambar 2.6**, menampilkan data visual mengenai berbagai kondisi cuaca yang dapat terjadi di lingkungan sekitar. Gambar tersebut terdiri dari empat panel yang masing-masing mewakili satu jenis cuaca, yaitu berawan, berkabut, cerah, dan hujan. Pada bagian kiri atas terlihat gambar langit yang tertutup awan tebal berwarna gelap dengan cahaya matahari yang tersembunyi di balik awan, menunjukkan kondisi cuaca berawan. Di sebelah kanannya, terlihat jalan tol dengan banyak kendaraan dan suasana yang tampak buram karena tertutup kabut, yang menggambarkan kondisi berkabut. Pada gambar kiri bawah, terlihat pemandangan laut dengan langit cerah berwarna biru dan matahari bersinar terang, mewakili kondisi cuaca cerah. Terakhir, pada gambar kanan bawah, tampak sebuah mobil sport yang sedang berada di jalan basah dengan butiran hujan yang turun dari langit, menggambarkan cuaca hujan. Keempat gambar ini memberikan ilustrasi yang jelas dan informatif mengenai variasi kondisi cuaca yang dapat diamati secara visual, sehingga dapat membantu dalam pemahaman karakteristik masing-masing jenis cuaca.

### 2.2.2 Pengolahan Citra

Pengolahan citra digital (*Digital Image Processing*) adalah cabang ilmu yang mempelajari berbagai teknik untuk memproses dan memanipulasi citra. Citra yang dimaksud mencakup gambar diam (foto) maupun gambar bergerak yang dapat berasal dari perangkat seperti *webcam*. Istilah "digital" merujuk pada pengolahan citra yang dilakukan secara digital menggunakan *computer* citra adalah fungsi kontinu yang merepresentasikan intensitas cahaya dalam bidang dua dimensi. Agar dapat diproses oleh komputer, citra ini harus diubah menjadi bentuk numerik dengan nilai-nilai diskrit, yang dikenal sebagai proses digitalisasi citra (Kusumanto and Tompunu, 2011).

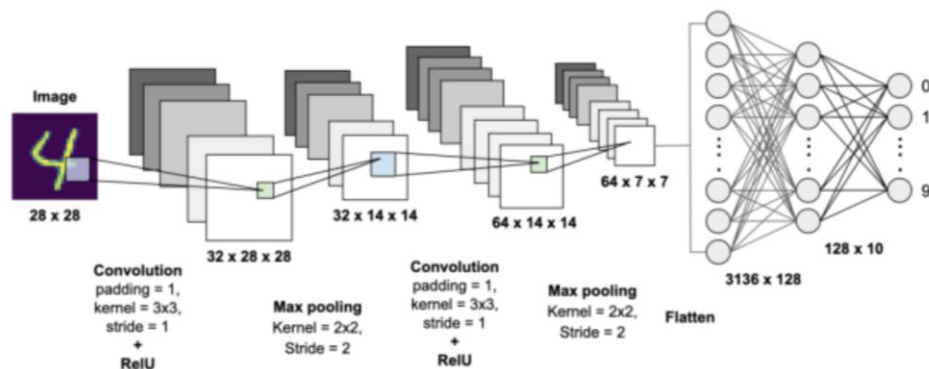
Digitalisasi citra melibatkan pengambilan gambar nyata dan merepresentasikannya dalam bentuk *numerik* menggunakan nilai diskrit. Representasi ini memungkinkan komputer untuk mengolah citra menggunakan algoritma pengolahan citra, yang mencakup teknik seperti perbaikan kualitas, deteksi tepi, segmentasi, dan ekstraksi fitur. Citra digital banyak digunakan dalam berbagai bidang, seperti fotografi, pengolahan gambar medis, pengawasan keamanan, dan kecerdasan buatan. Dalam aplikasi ini, pengolahan citra digital memungkinkan analisis dan interpretasi data visual secara efisien (Ririh *et al.*, 2020).

Citra digital didefinisikan sebagai fungsi dengan dua variabel,  $f(x, y)$  di mana  $x$  dan  $y$  merepresentasikan koordinat spasial, sementara nilai  $f(x, y)$  menunjukkan intensitas citra pada titik tersebut. Konsep ini digambarkan dalam ilustrasi pada Gambar 2.3. Teknologi yang mendasari pembentukan dan tampilan warna dalam citra digital didasarkan pada prinsip bahwa warna dapat dihasilkan dari kombinasi tiga warna utama, yaitu merah, hijau, dan biru (*Red, Green, Blue* - RGB) (Gazali *et al.*, 2012).

### 2.2.3 Algoritma Convolutional Neural Network (CNN)

*Convolutional Neural Network* merupakan salah satu jenis *neural network* yang biasanya digunakan untuk pengolahan data image. CNN digunakan untuk melakukan klasifikasi data yang berlabel dengan menggunakan metode *supervised*

*learning* (Suwitono and Kaunang, 2022). Pada *supervised learning* target yang diharapkan dari *input* yang diterima jaringan telah diketahui lebih lanjut. CNN adalah salah satu jenis arsitektur jaringan saraf tiruan yang dirancang khusus untuk mengolah data berbentuk citra atau gambar. CNN mampu menangkap fitur-fitur penting dari data visual seperti tepi, tekstur, pola, atau objek, yang kemudian digunakan untuk analisis lebih lanjut, seperti klasifikasi, segmentasi, atau deteksi objek. Kemampuan ini membuat CNN menjadi metode yang sangat efektif dalam berbagai aplikasi, termasuk pengenalan wajah, kendaraan otonom, dan analisis citra medis (Wulandari *et al.*, 2020).

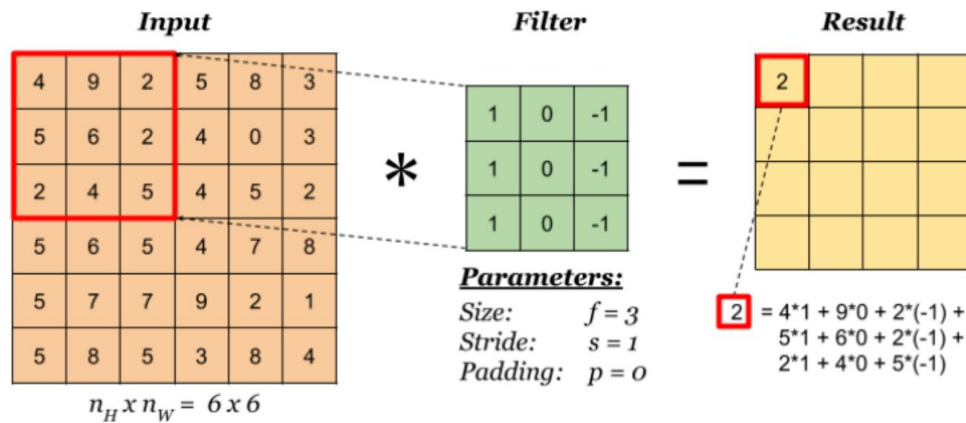


**Gambar 2.6** Arsitektur CNN (Sholawati *et al.*, 2022).

**Gambar 2.6.** merupakan ilustrasi arsitektur metode CNN yang terdiri atas tiga bagian, antara lain *input*, *feature extraction* dan *classification*. Pada *layer feature extraction* terdapat beberapa *layer* antara lain:

### 2.2.3.1 Convolutional layer

*Convolutional layer* merupakan proses ekstraksi fitur pada citra dengan menggunakan *filter* atau *kernel*. Proses konvolusi dilakukan dengan menggeser filter hingga mencakup seluruh bagian citra. Jumlah pergeseran *filter* disebut *stride*. Apabila pada proses konvolusi ingin mengekstraksi semua *pixel* pada citra, maka dapat dilakukan dengan cara menambah pixel bernilai 0 pada setiap sisi *matrix*. Setiap pergeseran *filter* akan dilakukan perkalian matriks antara matriks input dengan matriks filter, hasil dari proses tersebut berupa matriks 2 dimensi



**Gambar 2.7** Ilustrasi Perhitungan Konvolusi (Sholawati *et al.*, 2022).

Untuk menghitung dimensi dari *feature map* kita bisa gunakan, Persamaan 2.1 Berikut.

$$f_{out} = \frac{W - N + 2P}{s} + 1 \dots\dots\dots(2.1)$$

(Sholawati *et al.*, 2022)

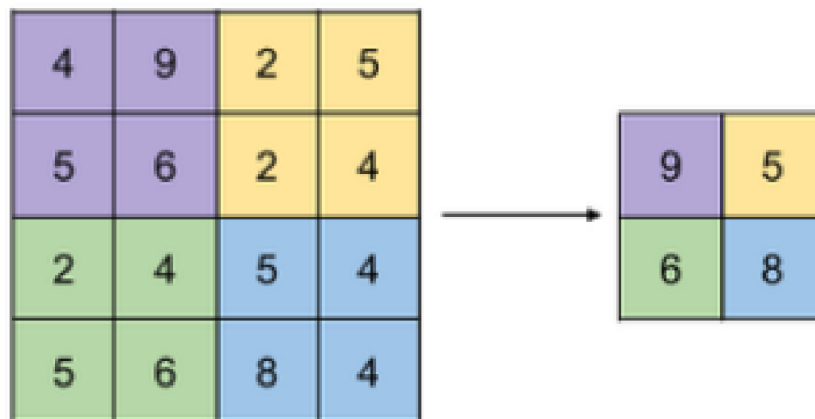
Persamaan 2.1, menjelaskan proses konvolusi pada jaringan saraf tiruan, terdapat beberapa parameter penting yang memengaruhi hasil keluaran (*feature map*), yaitu ukuran *input* (W), ukuran *filter* atau *kernel* (N), jumlah *padding* (P), dan nilai *stride* (S). Ukuran *input* (W) menunjukkan dimensi awal data yang akan diproses, biasanya berupa citra dua dimensi. Ukuran *filter* (N) adalah besarnya matriks yang digunakan untuk mendeteksi pola atau fitur dalam *input*. *Padding* (P) adalah jumlah piksel tambahan yang disisipkan di tepi input untuk menjaga ukuran *output* atau mencegah kehilangan informasi pada bagian tepi. Sementara itu, *stride* (S) menunjukkan seberapa jauh *filter* digeser setiap kali proses konvolusi dilakukan. Keempat parameter ini saling berpengaruh dalam menentukan ukuran akhir dari *feature map* yang dihasilkan, sehingga pemilihannya harus disesuaikan dengan kebutuhan arsitektur model.

Hasil *feature map* pada proses konvolusi akan dinormalisasi menggunakan fungsi aktivasi *ReLU*, dengan Persamaan 2.2.

$$x = \max(0, x) \dots\dots\dots(2.2)$$

### 2.2.3.2 MaxPooling Layer

Proses *pooling* merupakan salah satu tahapan penting dalam jaringan saraf konvolusional yang berfungsi untuk mengurangi dimensi dari matriks hasil konvolusi tanpa kehilangan informasi penting. Dengan mengurangi dimensi ini, pooling membantu mengurangi jumlah parameter dan kompleksitas komputasi, sehingga membuat proses pelatihan dan inferensi menjadi lebih cepat dan efisien. Selain itu, *pooling* juga berperan dalam menjaga invarian terhadap translasi kecil, artinya model tetap dapat mengenali fitur meskipun terjadi sedikit pergeseran pada posisi objek dalam citra. Tujuan utama dari *pooling* adalah untuk mempertahankan fitur dominan dari hasil konvolusi sambil menyaring informasi yang kurang relevan, sehingga mempercepat proses komputasi dan mencegah *overfitting*.

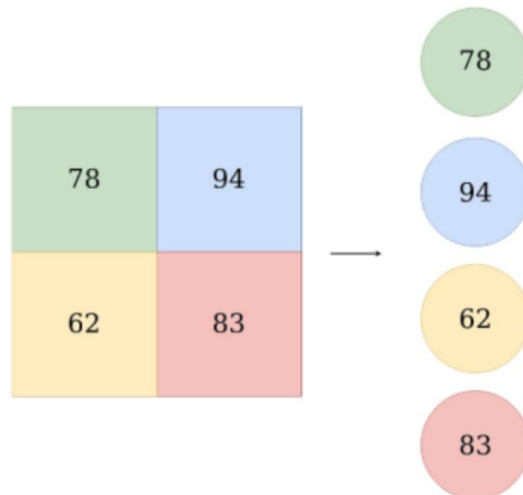


**Ganbar 2.8** Ilustrasi *Maxpooling* (Sholawati *et al.*, 2022).

#### 2.2.3.2.1 Flatten

Proses ini dikenal sebagai *flattening*, yaitu langkah dalam jaringan saraf konvolusional yang bertujuan untuk mengubah matriks dua dimensi hasil dari *feature learning* menjadi vektor satu dimensi. *Flattening* dilakukan sebelum data masuk ke lapisan *fully connected (dense layer)*, karena lapisan ini membutuhkan *input* dalam bentuk satu dimensi. Dengan mengubah matriks fitur menjadi vektor, seluruh informasi yang telah diekstraksi oleh lapisan konvolusi dan *pooling* dapat

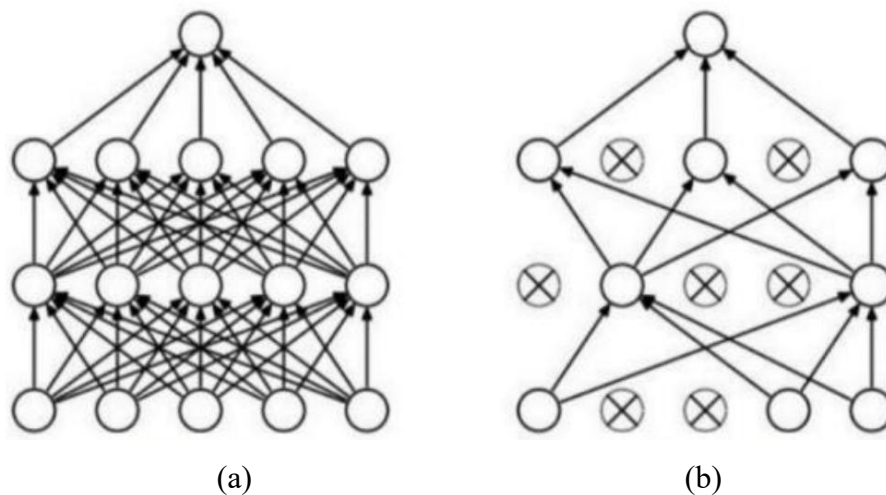
dihubungkan langsung dengan *neuron-neuron* di lapisan akhir untuk proses klasifikasi atau prediksi. Proses ini penting karena menjadi jembatan antara bagian ekstraksi fitur dan bagian pengambilan keputusan dalam arsitektur CNN.



**Ganbar 2.8** Ilustrasi *Flatten* (Sholawati *et al.*, 2022).

#### 2.2.3.2.2 *Dropout*

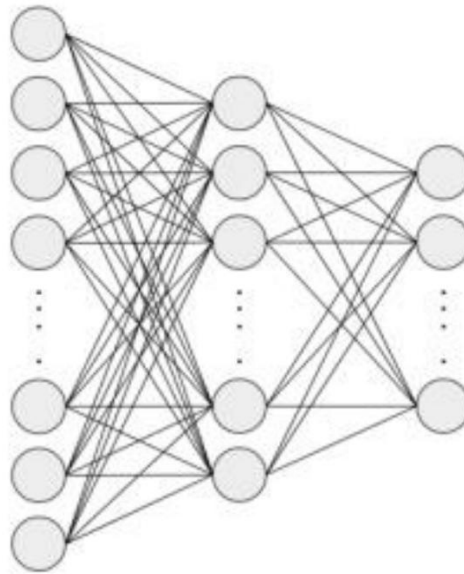
Proses ini dikenal sebagai *dropout*, yaitu teknik regularisasi dalam jaringan saraf tiruan yang dilakukan dengan menghilangkan secara sementara sejumlah *neuron* secara acak selama proses pelatihan. Tujuannya adalah untuk mencegah *overfitting*, yaitu kondisi ketika model terlalu baik dalam menghafal data latih namun gagal melakukan generalisasi terhadap data baru. Dengan mengabaikan neuron tertentu beserta koneksinya pada setiap iterasi, dropout mendorong jaringan untuk tidak bergantung pada *neuron* tertentu saja dan memaksa model untuk belajar pola yang lebih umum dan kuat. Meskipun selama pelatihan sebagian neuron dinonaktifkan, saat proses inferensi (pengujian) semua *neuron* tetap diaktifkan, sehingga hasil prediksi tetap optimal.



**Ganbar 2.8** (a) Ilustrasi *Flatten* Tanpa *Dropout*, (b) Ilustrasi *Flatten* Dengan *Dropout* (Sholawati *et al.*, 2022).

#### 2.2.3.2.3 *Fully Connected Layer*

Proses ini merujuk pada *fully connected layer* atau *dense layer*, yaitu tahap dalam jaringan saraf tiruan di mana semua *neuron* dari data hasil feature learning yang telah diflatten disatukan ke dalam satu vektor dan dihubungkan sepenuhnya dengan neuron-neuron di lapisan berikutnya. Setelah proses *flattening* mengubah output dua dimensi menjadi vektor satu dimensi, *fully connected layer* mengambil vektor ini dan memprosesnya lebih lanjut untuk menghasilkan prediksi akhir. Setiap neuron dalam lapisan ini memiliki koneksi ke seluruh neuron pada lapisan sebelumnya, sehingga memungkinkan jaringan untuk melakukan kombinasi dan pemetaan fitur secara kompleks. Tahap ini merupakan inti dari proses klasifikasi atau pengambilan keputusan dalam arsitektur CNN.



**Ganbar 2.9** Ilustrasi *Fully Connected Layer* (Sholawati et al., 2022).

#### 2.2.3.2.4 Fungsi *Softmax*

Menghitung nilai probabilitas dari tiap kelas target dengan semua kelas target. Rentang nilai probabilitas dari hasil perhitungan *softmax* yaitu antara 0 hingga 1. Rumus fungsi *softmax* ditunjukkan pada Persamaan 2.3.

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \dots\dots\dots (2.3)$$

Persamaan 2.3, menjelaskan  $P(y_i)$  adalah probabilitas bahwa *input*  $x$  diklasifikasikan ke dalam kelas ke- $i$ , yang dihitung menggunakan fungsi *softmax*. Nilai  $z_i$  merupakan *logit*, yaitu *output* dari model sebelum diterapkan fungsi *softmax*, untuk kelas ( $K$ ) ke- $i$  menyatakan jumlah total kelas yang tersedia dalam klasifikasi bilangan *Euler* ( $e$ ), yaitu konstanta matematika dengan nilai sekitar 2,718, yang digunakan sebagai basis dalam perhitungan eksponensial pada fungsi *softmax*.

#### 2.2.4 *Internet of Things* (IoT)

*Internet of Things* (IoT) adalah konsep di mana semua benda di dunia dapat saling terhubung dan berkomunikasi melalui jaringan *internet*, membentuk satu kesatuan

sistem yang terintegrasi. IoT bekerja dengan mengandalkan tiga elemen utama: benda fisik yang dilengkapi modul IoT, perangkat penghubung ke *internet* seperti modem atau *router nirkabel*, serta *cloud* sebagai pusat data untuk menyimpan aplikasi dan basis data. Dengan ketiga elemen ini, IoT menciptakan jaringan yang memungkinkan berbagai perangkat berfungsi secara otomatis dan terkoordinasi (Sari *et al.*, 2024).

Proses kerja IoT menggunakan argumen pemrograman yang memungkinkan mesin untuk saling berinteraksi tanpa memerlukan intervensi manusia dan tanpa dibatasi oleh jarak. *Internet* menjadi penghubung utama dalam interaksi antar mesin, sedangkan peran manusia terbatas pada mengatur dan memantau operasi perangkat secara langsung. Dengan pendekatan ini, IoT memungkinkan terciptanya sistem yang lebih efisien dan otomatis dalam berbagai aspek kehidupan sehari-hari.

*Raspberry Pi 5*, termasuk *Raspberry Pi 5*, berperan sebagai modul *mikrokontroler* dengan kemampuan komunikasi *Wi-Fi* yang memungkinkan perangkat terhubung dan berinteraksi dalam jaringan internet. Dengan dukungan pemrograman melalui platform seperti Arduino IDE dan *MicroPython*, modul ini mempermudah pengembangan berbagai solusi teknis dan kreatif. Dalam konteks *Internet of Things* (IoT), ESP sering digunakan sebagai jembatan antara perangkat keras seperti sensor dan aktuator dengan *server cloud*, memungkinkan pengumpulan, pengolahan, dan pengiriman data secara real-time. IoT menghubungkan perangkat fisik dalam sebuah jaringan yang saling berkomunikasi untuk berbagi informasi dan melaksanakan tugas otomatis, seperti sistem rumah pintar, kontrol mesin industri, atau pemantauan lingkungan. ESP, dengan fitur konsumsi daya rendah, performa tinggi, serta fleksibilitas integrasi, menjadi pilihan utama dalam pengembangan aplikasi IoT yang tidak hanya cerdas tetapi juga efisien dan skalabel untuk berbagai kebutuhan.

### 2.2.5 Python

*Python* merupakan sebuah bahasa pemrograman tingkat tinggi yang dibuat oleh Guido Van Rossum dan dirilis pada tahun 1991 *Python* juga merupakan bahasa yang sangat populer belakangan ini. Selain itu *python* juga merupakan bahasa

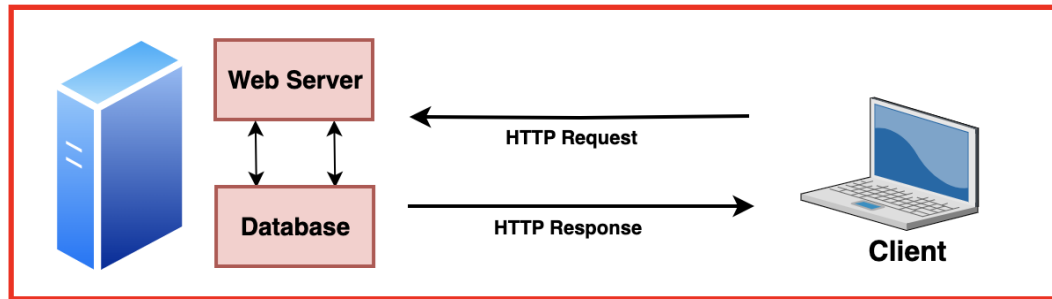
pemrograman yang multi fungsi salah satunya pada bidang *Machine Learning* dan *Deep Learning* (Alfarizi *et al.*, 2023).

*Python* dapat digunakan untuk menghasilkan visualisasi grafik fungsi dua peubah dengan memanfaatkan modul-modul yang disediakan dalam bahasa *Python*, seperti *Matplotlib* dan *NumPy*. *Matplotlib* berfungsi untuk menciptakan visualisasi data dalam berbagai bentuk, termasuk *scatter plot*, *line chart*, *bar chart*, dan *pie chart*. Modul ini juga memberikan kemampuan untuk melakukan modifikasi pada tampilan grafik, seperti penambahan judul dan lainnya. Sementara itu, *NumPy* merupakan modul yang sangat berguna dalam analisis data dan pemodelan matematika. *NumPy* memiliki peran utama dalam melakukan operasi pada data numerik dalam bentuk *array*. Contoh operasi yang dapat dilakukan oleh *NumPy* melibatkan operasi *aritmatika*, *trigonometri*, fungsi matematika, serta statistik. Dengan menggunakan *NumPy*, pemrosesan data *numerik* dapat dilakukan dengan lebih efisien dalam lingkungan bahasa pemrograman *Python* (Surbakti *et al.*, 2024).

Kemampuan *Python* untuk mendukung pengembangan berbasis *platform* yang berbeda juga menjadi salah satu kekuatannya. Selain digunakan dalam aplikasi berbasis AI, *Python* juga diadopsi dalam bidang pengembangan perangkat IoT, simulasi sains, hingga otomatisasi. Bahasa ini mampu beradaptasi dengan teknologi masa depan berkat pembaruan yang konsisten dan fleksibilitasnya untuk diintegrasikan dengan berbagai perangkat keras dan sistem. Kombinasi efisiensi, keandalan, dan keberagaman fungsionalitas menjadikan *Python* tidak hanya relevan tetapi juga sangat vital dalam ekosistem teknologi saat ini.

#### **2.2.6 Hypertext Transfer Protocol (HTTP)**

*Hypertext Transfer Protocol* (HTTP) adalah protokol lapisan aplikasi yang digunakan sebagai dasar komunikasi data di *World Wide Web*. Protokol ini dirancang untuk mentransfer informasi seperti teks, gambar, video, dan data lainnya antara *server* dan *klien* melalui jaringan berbasis IP. HTTP berfungsi sebagai protokol permintaan dan tanggapan, *klien* (*browser* atau perangkat lain) mengirimkan permintaan ke *server*, dan *server* memberikan tanggapan berupa data yang diminta (Gigih *et al.*, 2017).



**Gambar 10** Alur Kerja *Hypertext Transfer Protocol* (Gigih et al., 2017).

Protokol HTTP memiliki karakteristik sederhana, efisien, dan dapat diimplementasikan pada berbagai *platform*. Sebagai protokol *stateless*, setiap transaksi HTTP bersifat *independen*, artinya server tidak menyimpan informasi tentang *klien* setelah permintaan selesai. Karakteristik ini memungkinkan pengelolaan sumber daya server menjadi lebih ringan, meskipun terkadang memerlukan mekanisme tambahan seperti *cookie* untuk menjaga konteks interaksi pengguna.

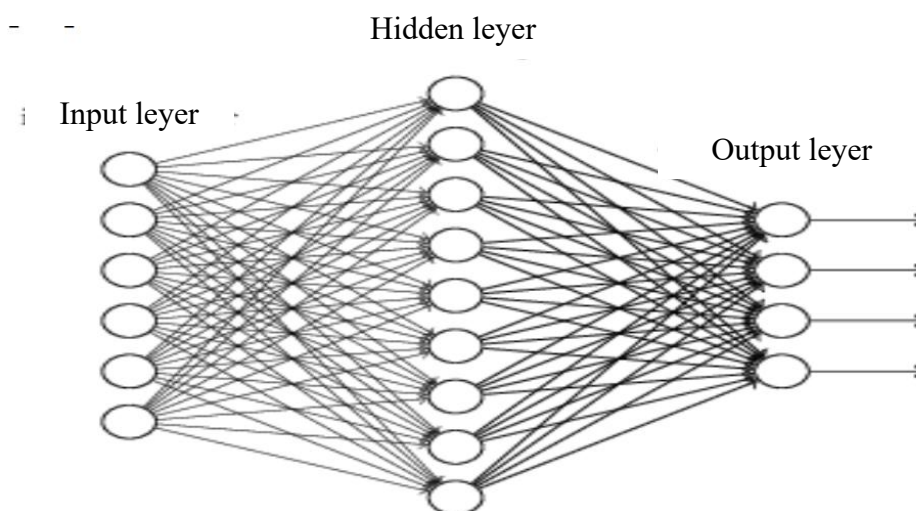
Dalam skripsi ini, HTTP digunakan sebagai protokol utama untuk mengirimkan data citra awan dari *Raspberry pi 5* ke server. HTTP dipilih karena kemudahannya dalam implementasi, kompatibilitasnya dengan berbagai perangkat, serta dukungannya terhadap format data yang fleksibel, seperti JSON atau XML, yang mempermudah integrasi dengan *model deep learning* di *server*. Selain itu, protokol HTTP memiliki dukungan luas di komunitas pengembang, sehingga memudahkan *troubleshooting* dan pengembangan sistem.

### 2.2.7 Deep Learning

*Deep Learning* adalah cabang dari pembelajaran mesin yang berfokus pada model berbasis jaringan saraf tiruan (*artificial neural networks*) dengan banyak lapisan (*deep neural networks*). Teknologi ini meniru cara kerja otak manusia dalam memproses data, mengenali pola, dan membuat keputusan. Salah satu keunggulan utama *deep learning* adalah kemampuannya untuk secara otomatis mengekstraksi fitur dari data mentah tanpa memerlukan intervensi manual, sehingga sangat cocok digunakan untuk pengolahan data yang kompleks, seperti pengenalan gambar,

pemrosesan bahasa alami, dan prediksi data *time series*. Model *deep learning* dilatih menggunakan algoritma *backpropagation*, yang menyesuaikan bobot dalam jaringan berdasarkan error yang dihasilkan dari prediksi sebelumnya (Nugroho *et al.*, 2020).

Dalam konteks penelitian cuaca, *deep learning* dapat diterapkan untuk memprediksi parameter cuaca seperti suhu, kelembaban, dan tekanan udara. Model seperti *Long Short-Term Memory* (LSTM), yang merupakan jenis jaringan saraf rekursif (*recurrent neural network*), sering digunakan untuk data *time series* karena kemampuannya dalam mengingat informasi dari data sebelumnya. Penggunaan *deep learning* pada prediksi cuaca memungkinkan analisis yang lebih akurat dibandingkan metode tradisional karena model dapat menangkap pola kompleks dalam data historis. Fleksibilitas dan skalabilitas *deep learning* menjadikannya alat yang penting dalam memanfaatkan data besar (*big data*) untuk menghasilkan prediksi yang lebih presisi.



**Gambar 2.10** Model *Long Short-Term Memory* (LSTM) (Nugroho *et al.*, 2020).

**Gambar 2.10** menggambarkan arsitektur jaringan saraf tiruan (*Artificial Neural Network*) dengan lapisan yang sepenuhnya terhubung, atau yang dikenal sebagai *fully connected layer*. Dalam jaringan ini, setiap *neuron* pada satu lapisan terhubung ke semua *neuron* pada lapisan berikutnya melalui bobot (*weights*). Lapisan *input* menerima data mentah yang kemudian diproses melalui satu atau lebih lapisan tersembunyi (*hidden layers*) sebelum menghasilkan *output* di lapisan akhir (*output*

*layer*). Proses ini memungkinkan jaringan untuk mempelajari hubungan kompleks antara fitur *input* dan target *output*. Lapisan tersembunyi bertugas mengekstraksi fitur dari data, sementara lapisan output menghasilkan prediksi berdasarkan informasi yang telah diproses.

*Fully connected layer* merupakan komponen penting dalam *deep learning* karena fleksibilitasnya dalam memodelkan hubungan *non-linear* yang kompleks. Bobot dan bias dalam setiap koneksi diperbarui selama proses pelatihan menggunakan *algoritma backpropagation* untuk meminimalkan *error* antara *output* prediksi dan target sebenarnya. Namun, arsitektur ini juga memiliki kelemahan, seperti kebutuhan komputasi yang tinggi dan risiko *overfitting* jika jumlah parameter terlalu besar. Oleh karena itu, *fully connected layer* sering digunakan dalam kombinasi dengan lapisan lain, seperti *convolutional layers* atau *pooling layers*, untuk meningkatkan efisiensi dan akurasi model.

### 2.2.8 Edge Computing

*Edge computing* adalah paradigma komputasi terdistribusi yang memindahkan proses pemrosesan data lebih dekat ke sumbernya, seperti perangkat IoT atau sensor. Berbeda dengan komputasi tradisional yang mengandalkan *cloud* pusat, *edge computing* memanfaatkan perangkat lokal atau *server* di dekat lokasi data untuk mengurangi latensi dan meningkatkan efisiensi. Pendekatan ini sangat cocok untuk aplikasi yang membutuhkan respons *real-time*, seperti prediksi cuaca, sistem pengawasan, dan analisis data sensor. Selain itu, *edge computing* membantu mengurangi beban pengiriman data ke *cloud*, sehingga dapat menghemat *bandwidth* dan menurunkan biaya operasional (Wulandari *et al.*, 2023).

*Platform edge computing* biasanya mendukung integrasi dengan perangkat keras dan perangkat lunak lain untuk menciptakan solusi yang dapat diskalakan dan dapat diandalkan. Contohnya, *edge computing* sering digunakan bersama dengan teknologi *machine learning* untuk melakukan analisis data secara langsung di perangkat *edge*. Hal ini memungkinkan model prediksi untuk berjalan secara lokal tanpa bergantung sepenuhnya pada koneksi *internet*. Dalam konteks penelitian ini, *platform edge computing* dapat membantu mengolah data dari sensor cuaca secara

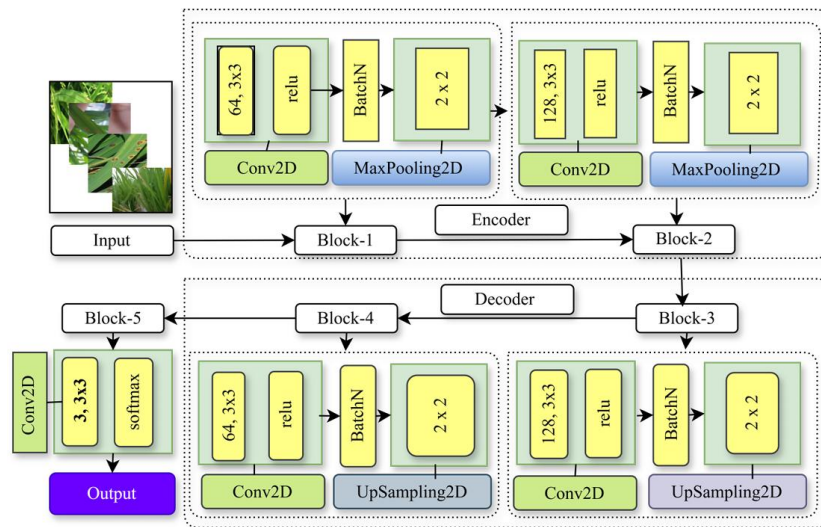
langsung di perangkat sebelum dikirim ke *server* melalui protokol HTTP, sehingga mempercepat proses pengambilan keputusan dan meningkatkan efisiensi sistem prediksi cuaca (Naufal, 2021).

*Platform Edge Computing* menggambarkan skema implementasi *edge computing* untuk mendukung pengolahan data secara lokal sebelum mengirimkannya ke *cloud*. *Platform* ini terdiri dari beberapa komponen utama, yaitu perangkat pengguna (laptop dan *smartphone*), jaringan *Open Radio Access Network* (RAN), *server edge computing*, dan *cloud*. Sistem ini menunjukkan bagaimana data diproses di berbagai tingkatan, mulai dari perangkat pengguna hingga *multi-access edge computing*, sebelum dihubungkan ke aplikasi *cloud*.

Pada tahap pertama, perangkat pengguna terhubung ke jaringan *Open RAN*, yang berfungsi sebagai penghubung awal untuk mengirimkan data ke *platform edge computing*. *Platform edge computing* dapat berupa *server*, PC lokal, atau kombinasi keduanya, yang bertugas melakukan pengolahan data awal, seperti analisis atau deteksi berbasis AI (contohnya deteksi masker wajah). Data yang telah diolah kemudian diteruskan ke *cloud* untuk penyimpanan lebih lanjut atau pengolahan tambahan, jika diperlukan. Dengan cara ini, *platform edge computing* mengurangi latensi, mengoptimalkan penggunaan *bandwidth*, dan meningkatkan responsivitas sistem dalam aplikasi *real-time*.

### 2.2.9 MobileNetV2

*MobileNetV2* adalah arsitektur jaringan konvolusional yang dioptimalkan untuk perangkat dengan daya komputasi terbatas, seperti *Raspberry Pi*. Model ini merupakan pengembangan dari *MobileNetV1* dengan memperkenalkan *inverted residuals* dan *linear bottleneck*, yang memungkinkan efisiensi tinggi dalam pemrosesan gambar dengan tetap mempertahankan akurasi yang baik (Nafisa *et al.*, 2023).



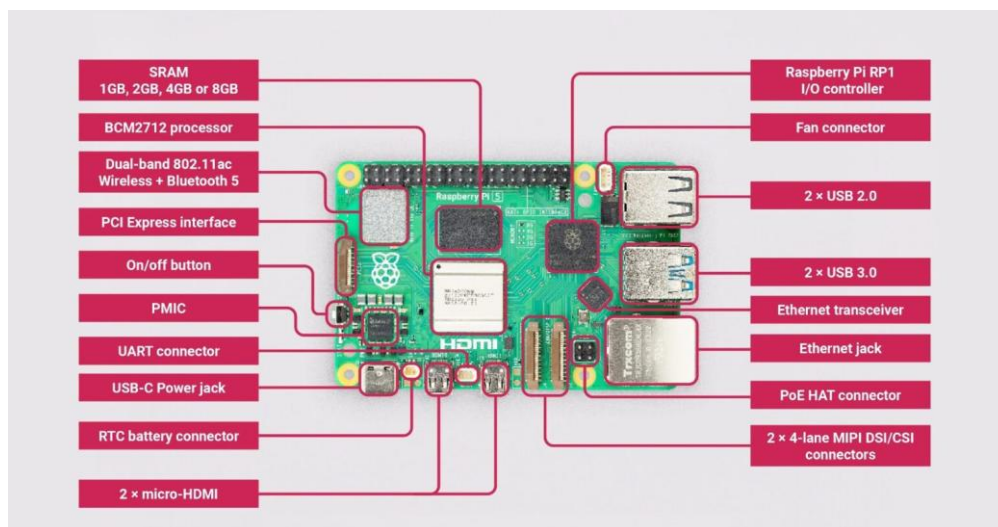
**Gambar 2.11** Arsitektur *MobileNetV2* (Dutta *et al.*, 2024).

Arsitektur *MobileNetV2* menggunakan *depthwise separable convolution* untuk mengurangi jumlah parameter dan operasi komputasi yang diperlukan dibandingkan dengan konvolusi standar. Selain itu, *MobileNetV2* memanfaatkan *residual connections* dengan struktur *bottleneck* yang memungkinkan aliran informasi tetap efisien melalui lapisan-lapisan dalam jaringan. Pendekatan ini mengurangi kebutuhan daya komputasi tanpa mengorbankan performa model dalam tugas klasifikasi gambar (Dutta *et al.*, 2024).

Dalam penelitian ini, *MobileNetV2* digunakan sebagai model utama untuk sistem klasifikasi awan. Model ini dilatih menggunakan *dataset* citra awan yang dikumpulkan dari berbagai kondisi cuaca untuk memastikan keakuratan prediksi. Implementasi pada *Raspberry Pi* memungkinkan sistem untuk melakukan klasifikasi awan secara *real-time* dengan efisiensi tinggi, menjadikannya solusi yang ideal untuk pemantauan cuaca berbasis IoT. Dengan keunggulannya dalam efisiensi komputasi dan akurasi tinggi, *MobileNetV2* menjadi pilihan yang tepat dalam mendukung pengembangan sistem klasifikasi awan berbasis CNN untuk pemantauan cuaca *real-time*.

### 2.2.10 Raspberry Pi 5

*Raspberry Pi 5* adalah komputer single-board yang banyak digunakan dalam berbagai aplikasi, termasuk *Internet of Things (IoT)*, robotika, dan pemrosesan citra. Perangkat ini memiliki ukuran kecil, konsumsi daya rendah, serta dukungan terhadap berbagai sistem operasi berbasis *Linux*. Kemampuannya dalam menangani tugas komputasi ringan hingga menengah menjadikannya pilihan populer untuk proyek-proyek berbasis sistem tertanam dan kecerdasan buatan (Sałuch *et al.*, 2018)



**Gambar 2.8** *Raspberry Pi 5 Microcomputer Module* (Sałuch *et al.*, 2018).

Dalam penelitian ini, *Raspberry Pi 5* digunakan sebagai pusat pemrosesan utama untuk mengimplementasikan model klasifikasi awan berbasis *MobileNetV2*. Perangkat ini mampu menjalankan *model deep learning* ringan secara *real-time* dengan efisiensi daya yang optimal. Dengan adanya unit pemrosesan grafis terintegrasi, *Raspberry Pi 5* dapat melakukan inferensi model dengan waktu respon yang cukup cepat, memungkinkan sistem untuk memberikan hasil klasifikasi awan secara instan.

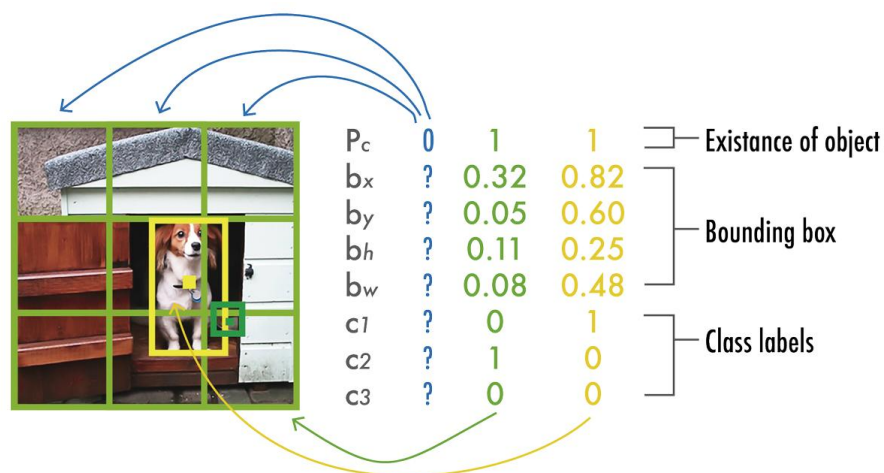
Selain itu, *Raspberry Pi 5* memiliki antarmuka yang mendukung berbagai sensor dan modul kamera, sehingga dapat dengan mudah diintegrasikan untuk akuisisi citra langit. Kemampuan konektivitasnya yang mencakup *Wi-Fi* dan *Bluetooth* juga memungkinkan perangkat ini untuk mengirim hasil klasifikasi ke *server* atau

aplikasi monitoring cuaca. Hal ini membuat *Raspberry Pi 5* menjadi solusi yang fleksibel dan mudah diakses dalam sistem pemantauan cuaca berbasis *IoT*.

Keunggulan utama *Raspberry Pi 5* dalam penelitian ini adalah efisiensi daya, fleksibilitas pemrograman, dan kemampuannya dalam menangani model machine learning sederhana. Dengan biaya yang relatif rendah dibandingkan komputer konvensional, *Raspberry Pi 5* menjadi alternatif yang efektif dalam mengembangkan sistem klasifikasi awan berbasis CNN untuk pemantauan cuaca *real-time*, tanpa perlu ketergantungan pada infrastruktur komputasi yang lebih kompleks.

### 2.2.11 YOLO

*You Only Look Once (YOLO)* adalah algoritma deteksi objek berbasis *deep learning* yang mampu mendeteksi dan mengenali objek dalam sebuah gambar hanya dengan satu kali proses pemindaian (*single pass*) melalui jaringan saraf. Berbeda dengan pendekatan tradisional yang memerlukan beberapa tahap, YOLO mengubah deteksi objek menjadi masalah regresi langsung dari piksel gambar ke koordinat *bounding box* dan label kelas. Algoritma ini membagi gambar menjadi *grid*, lalu untuk setiap *grid* memprediksi posisi, ukuran, *confidence score*, dan kelas objek secara simultan. Karena efisiensinya yang tinggi, YOLO sangat cocok untuk aplikasi *real-time* seperti sistem pengawasan, kendaraan otonom, dan robotika (Raspati, 2024).



**Gambar 2.9** Prediksi *Output* YOLO (Terven *et al.*, 2023).

**Gambar 2.9**, menggambarkan model yolo yang disederhanakan dengan *grid* (kotak pembagi gambar) berukuran 3x3, memiliki tiga kelas, dan menghasilkan satu prediksi kelas untuk setiap elemen *grid*, sehingga menghasilkan sebuah vektor dengan delapan nilai (Terven *et al.*, 2023).

### 2.211.1 Backbone

*Backbone* adalah komponen utama dalam arsitektur YOLO yang berfungsi untuk mengekstraksi fitur visual dari gambar *input* melalui serangkaian lapisan konvolusi. Pada tahap ini, gambar diubah menjadi representasi fitur yang lebih sederhana namun informatif, yang mencakup elemen-elemen seperti tepi, bentuk, dan tekstur. Proses ini dilakukan secara bertahap, di mana lapisan awal mendeteksi fitur dasar, sedangkan lapisan yang lebih dalam mengenali pola yang lebih kompleks. Hasil ekstraksi fitur dari backbone kemudian digunakan oleh bagian selanjutnya dalam jaringan untuk melakukan deteksi objek secara akurat dengan Persamaan 2.4.

$$f_{out}(x, y) = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} w(i, j) \cdot f_{in}(x + i, y + j) \dots\dots\dots (2.4)$$

(Terven *et al.*, 2023).

Persamaan 2.4, menjelaskan proses operasi konvolusi dalam jaringan saraf konvolusional, yaitu dengan mengalikan area lokal dari citra *input*  $f_{in}(x + i, y + j)$  dengan bobot *kernel*  $w(i, j)$  lalu menjumlahkan hasilnya untuk menghasilkan nilai *output*  $f_{out}(x, y)$  pada posisi tertentu. Operasi ini dilakukan dalam jangkauan ukuran kernel  $k \times k$  dan diterapkan secara bergeser ke seluruh gambar. Hasilnya adalah peta fitur (*feature map*) yang menyoroti pola-pola penting seperti tepi, sudut, atau tekstur dalam gambar. Dengan konvolusi ini, jaringan dapat mengekstraksi informasi visual secara bertahap dari fitur sederhana hingga kompleks.

### 2.2.11.2 Neck

*Neck* dalam arsitektur YOLO adalah bagian yang menghubungkan *backbone* dan *head*, berfungsi untuk menggabungkan dan memperkuat fitur dari berbagai level atau skala. Setelah fitur diekstraksi oleh *backbone*, *neck* memprosesnya lebih lanjut agar informasi dari fitur dangkal (detail lokal) dan fitur dalam (informasi semantik)

dapat digabungkan secara efektif. Proses ini umumnya dilakukan melalui operasi seperti *upsampling*, *concatenation*, dan konvolusi, sehingga menghasilkan representasi fitur yang lebih kaya dan informatif. Dengan demikian, *neck* membantu model mendeteksi objek dengan ukuran dan posisi yang bervariasi secara lebih akurat dengan Persamaan 2.5.

$$f_{up}(x, y) = (1 - \alpha)(1 - \beta)f(i, j) + \alpha(1 - \beta)f(i + 1, j) + (1 - \alpha)\beta f(i, j + 1) + \alpha\beta f(i + 1, j + 1) \dots (2.5)$$

(Terven *et al.*, 2023).

Persamaan 2.5, menjelaskan proses *bilinear interpolation* yang digunakan dalam operasi *upsampling*, yaitu memperbesar ukuran citra atau fitur map dengan memperkirakan nilai piksel baru berdasarkan piksel-piksel tetangganya. Nilai piksel hasil upsampling di posisi  $(x, y)$ , dilambangkan sebagai  $f_{up}(x, y)$ , dihitung sebagai kombinasi linier dari empat piksel terdekat:  $f(i, j)$ ,  $f(i + 1, j)$ ,  $f(i, j + 1)$ , dan  $f(i + 1, j + 1)$ , dengan bobot yang ditentukan oleh parameter interpolasi  $\alpha$  dan  $\beta$ , yang menunjukkan jarak relatif dari titik  $(x, y)$  ke posisi piksel-piksel tetangganya. Metode ini menghasilkan transisi nilai piksel yang lebih halus dibanding *nearest-neighbor*, sehingga menjaga kontinuitas visual dalam proses pembesaran fitur.

### 2.2.11.3 Head

*Head* adalah bagian terakhir dalam arsitektur YOLO yang berfungsi untuk menghasilkan prediksi akhir dari deteksi objek. Setelah fitur diekstraksi dan diproses oleh *backbone* dan *neck*, *head* menggunakan informasi tersebut untuk menentukan lokasi objek dalam gambar melalui *bounding box*, menilai tingkat kepercayaan keberadaan objek, serta mengklasifikasikan jenis objek yang terdeteksi. Dengan demikian, *head* menggabungkan semua data yang telah dipelajari jaringan untuk memberikan hasil deteksi yang lengkap dan siap digunakan dalam aplikasi nyata dengan Persamaan 2.6.

$$S \times S \times (B \times (5 + C)) \dots (2.6)$$

(Terven *et al.*, 2023).

Persamaan 2.6, menggambarkan bentuk output tensor pada bagian head YOLO. Di sini, gambar dibagi menjadi *grid* berukuran  $S \times S$ , di mana setiap sel *grid* bertanggung jawab untuk memprediksi objek yang berada di wilayahnya. Setiap sel memprediksi  $B$  *bounding box*, dan untuk setiap *bounding box*, terdapat 5 nilai yang mencakup koordinat posisi dan ukuran kotak serta skor kepercayaan (*confidence score*) yang menunjukkan seberapa yakin model terhadap keberadaan objek tersebut. Selain itu, ada  $C$  nilai tambahan yang mewakili probabilitas klasifikasi objek ke dalam masing-masing kelas yang mungkin. Dengan struktur ini, YOLO dapat secara simultan mendeteksi banyak objek sekaligus dan mengklasifikasikannya dalam satu langkah prediksi.

### 2.2.12 Visibility

*Visibility* atau jarak pandang adalah istilah dalam meteorologi yang merujuk pada seberapa jauh seseorang dapat melihat suatu objek secara jelas dengan mata telanjang dalam kondisi cahaya normal. Jarak pandang ini sangat dipengaruhi oleh kondisi atmosfer, seperti kabut, hujan, salju, asap, debu, atau partikel lain yang melayang di udara. *Visibility* menjadi parameter penting dalam berbagai sektor, terutama dalam dunia penerbangan, pelayaran, transportasi darat, serta pengamatan cuaca, karena keterbatasan jarak pandang dapat mengganggu keselamatan dan kelancaran operasional (Kurniawan, 2014).

Dalam konteks penerbangan, *visibility* menjadi salah satu faktor krusial yang diperhitungkan dalam prosedur lepas landas dan pendaratan pesawat. Organisasi penerbangan seperti ICAO (*International Civil Aviation Organization*) menetapkan *standar visibility* minimum untuk operasi bandara demi menjaga keselamatan. Kondisi *low visibility*, seperti saat terjadi kabut tebal atau hujan deras, dapat menyebabkan penundaan penerbangan atau bahkan pembatalan, karena pilot membutuhkan panduan visual yang jelas untuk mengoperasikan pesawat dengan aman. Oleh karena itu, sistem pengukuran *visibility* yang akurat dan *real-time* sangat dibutuhkan di setiap bandara (Zaman, 2023).

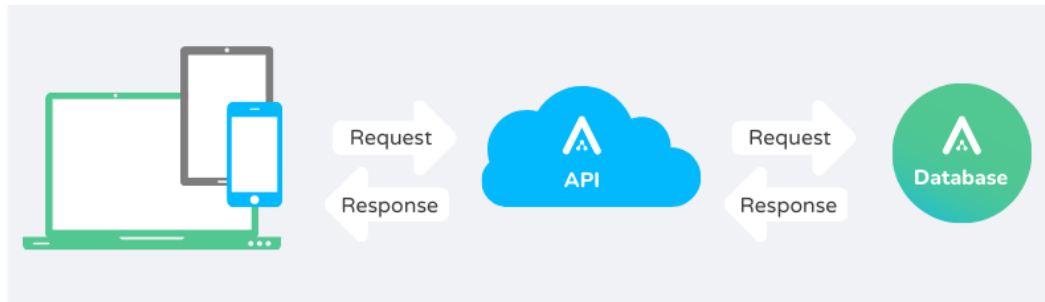
Secara teknis, *visibility* biasanya diukur menggunakan alat sensor seperti *transmissometer*, *forward scatter meter*, atau *ceilometer* yang mampu mendeteksi

dan menghitung jarak pandang berdasarkan pantulan atau hamburan cahaya di atmosfer. Namun, alat-alat ini umumnya mahal, membutuhkan perawatan *intensif*, dan tidak selalu tersedia di semua lokasi, terutama di wilayah dengan infrastruktur terbatas. Seiring perkembangan teknologi, pendekatan alternatif seperti pemrosesan citra digital dan algoritma visi komputer mulai dikembangkan untuk memperkirakan *visibility* secara lebih ekonomis dan fleksibel.

Pemanfaatan citra awan sebagai indikator visual untuk memperkirakan *visibility* menjadi pendekatan baru yang menjanjikan. Dengan menggunakan kamera yang menangkap kondisi langit dan algoritma deteksi berbasis CNN seperti *YOLOv5*, sistem dapat mengidentifikasi pola awan dan kondisi atmosfer yang berpotensi memengaruhi jarak pandang. Pendekatan ini tidak hanya mengurangi biaya peralatan, tetapi juga memungkinkan sistem untuk diintegrasikan pada perangkat edge seperti *Raspberry Pi*, sehingga dapat bekerja secara mandiri dan *real-time* di lapangan. Metode ini diharapkan menjadi solusi efisien dalam mendukung sistem pemantauan cuaca, khususnya di sektor penerbangan.

### **2.2.13 Application Programming Interface (API)**

*Application Programming Interface* (API) merupakan antarmuka yang memungkinkan dua atau lebih aplikasi perangkat lunak untuk berkomunikasi satu sama lain. API menyediakan serangkaian definisi, protokol, dan alat untuk membangun dan mengintegrasikan aplikasi perangkat lunak. Dalam konteks yang lebih luas, API bertindak sebagai jembatan yang memungkinkan suatu sistem untuk mengakses atau memanfaatkan fungsi dari sistem lain tanpa perlu mengetahui detail teknis dari bagaimana sistem tersebut bekerja. Hal ini memungkinkan pengembang untuk membangun aplikasi yang kompleks dengan lebih efisien, karena mereka dapat memanfaatkan layanan dan fungsi yang telah tersedia (Triawan, 2021).



**Gambar 10** Alur Kerja *Application Programming Interface* (Triawan, 2021).

API dapat berupa antarmuka untuk sistem operasi, pustaka perangkat lunak, layanan *web*, atau bahkan perangkat keras. Salah satu contoh paling umum adalah *RESTful API* yang menggunakan protokol HTTP untuk bertukar data dalam format seperti JSON atau XML. REST API banyak digunakan dalam pengembangan aplikasi *web* dan *mobile* karena kesederhanaannya serta kompatibilitasnya dengan berbagai platform. Dengan memanfaatkan API, pengembang dapat mengakses berbagai layanan seperti pemrosesan data, autentikasi pengguna, *integrasi database*, hingga layanan *machine learning* tanpa harus membangun semuanya dari awal (Hermanto, 2014).

Dalam sistem pemantauan cuaca atau klasifikasi citra berbasis IoT seperti pada penelitian ini, API berperan penting dalam menghubungkan perangkat keras seperti *Raspberry Pi* dengan *server* atau *platform* lain untuk mengirim dan menerima data. Misalnya, API dapat digunakan untuk mengirim hasil klasifikasi awan dari YOLOv5 ke *server web* untuk ditampilkan secara *real-time*, atau untuk mengambil data cuaca aktual dari sumber eksternal seperti *OpenWeatherMap* untuk dibandingkan dengan hasil klasifikasi lokal. Hal ini memungkinkan sistem berjalan secara terintegrasi dan dinamis dengan sumber data yang lebih luas.

Penggunaan API juga sangat membantu dalam pengembangan sistem berskala besar atau yang melibatkan banyak komponen. Dengan mendefinisikan API secara baik, setiap komponen dalam sistem dapat dikembangkan dan diuji secara terpisah, meningkatkan skalabilitas dan kemudahan pemeliharaan sistem. Dalam penelitian ini, API menjadi salah satu elemen penting dalam mendukung konektivitas antara sistem klasifikasi awan berbasis CNN dengan antarmuka pengguna, basis data,

maupun layanan eksternal lainnya. Pendekatan ini menjadikan sistem lebih fleksibel, mudah dikembangkan, dan siap untuk diimplementasikan dalam skala yang lebih luas.

#### 2.2.14 Website

*Website* merupakan sekumpulan halaman yang saling terhubung dan dapat diakses melalui jaringan *internet* menggunakan perangkat seperti komputer, tablet, atau *smartphone*. *Website* berperan sebagai antarmuka digital yang menyajikan berbagai informasi kepada pengguna secara interaktif. Dalam konteks sistem informasi, *website* dapat berfungsi sebagai media visualisasi data, pengolahan input pengguna, serta sebagai jembatan antara perangkat keras (*hardware*) dan pengguna akhir. *Website modern* dibangun dengan menggunakan berbagai teknologi seperti *HTML*, *CSS*, *JavaScript*, serta *framework backend* seperti *Flask*, *Django*, atau *Node.js* yang memfasilitasi komunikasi antara frontend dan server (Riyanto, 2015).

Perkembangan teknologi website telah mendorong *adopsi website* sebagai platform utama dalam berbagai sistem berbasis *Internet of Things (IoT)*, termasuk dalam pemantauan lingkungan secara *real-time*. *Website* memungkinkan data yang dikumpulkan oleh perangkat seperti *Raspberry Pi* dapat dikirimkan dan ditampilkan kepada pengguna dalam bentuk grafik, tabel, atau informasi teks secara langsung. Dalam implementasinya, komunikasi antara perangkat IoT dan server web dilakukan menggunakan protokol *HTTP* atau *MQTT*, dan data dapat disimpan sementara di server lokal atau dikirimkan ke layanan *cloud*. *Website* juga dapat memperkuat aspek *user experience (UX)* dengan menyediakan visualisasi interaktif yang mudah dipahami oleh pengguna (Bakri, 2016).

Dalam sistem pemantauan cuaca berbasis *Raspberry Pi* yang dikembangkan, website berperan sebagai dashboard utama yang menampilkan hasil klasifikasi cuaca, awan, dan estimasi *visibility* yang diperoleh dari model *deep learning* seperti *MobileNetV2*, *Custom CNN*, dan *YOLOv5*. *Raspberry Pi* bertugas melakukan proses akuisisi citra, klasifikasi, dan perhitungan *visibility*, kemudian hasilnya dikirimkan ke server melalui *request HTTP POST*. *Website* yang dibangun menggunakan *framework Flask* akan menerima data tersebut, menyimpannya, dan

menampilkannya dalam antarmuka yang dapat diakses melalui *browser*. Fitur ini memungkinkan pengguna untuk memantau kondisi cuaca secara *real-time* tanpa perlu berinteraksi langsung dengan perangkat fisik di lapangan.

Pemanfaatan *website* dalam sistem ini tidak hanya meningkatkan kemudahan akses data, tetapi juga memberikan nilai tambah dalam hal dokumentasi dan rekam jejak informasi. Data yang ditampilkan pada *website* dapat disimpan dalam basis data untuk analisis lebih lanjut atau untuk keperluan historis. Dengan demikian, integrasi *website* dalam sistem pemantauan cuaca berbasis *Raspberry Pi* tidak hanya berfungsi sebagai sarana visualisasi, tetapi juga sebagai komponen penting dalam mendukung transparansi, efisiensi, dan efektivitas sistem secara keseluruhan.

### 2.2.15 OpenCV

*Open Source Computer Vision Library (OpenCV)* adalah pustaka *open-source* yang dirancang untuk memproses gambar dan video secara *real-time*. Pustaka ini ditulis dalam bahasa pemrograman C/C++, tetapi juga menyediakan antarmuka untuk *Python*, *Java*, dan beberapa bahasa lainnya. *OpenCV* menyediakan berbagai fungsi untuk pemrosesan *citra digital*, seperti deteksi tepi, pelacakan objek, segmentasi, transformasi spasial, hingga pemrosesan warna dan morfologi gambar. Karena sifatnya yang efisien dan fleksibel, *OpenCV* menjadi salah satu pilihan utama dalam pengembangan aplikasi *computer vision*, baik untuk penelitian akademik maupun *industry* (Deshpande, 2020).



**Gambar 2.11** Menunjukkan Proses Pengenalan Citra Menggunakan *Opencv* (Deshpande, 2020).

Dalam implementasinya, *OpenCV* dapat digunakan untuk mempersiapkan gambar sebelum diproses oleh *model machine learning*. Tahapan preprocessing seperti pengubahan ukuran gambar, konversi ke skala abu-abu, normalisasi piksel, dan penghapusan *noise* sering kali diperlukan agar model klasifikasi dapat bekerja secara optimal. Selain itu, *OpenCV* juga menyediakan fungsi untuk anotasi visual, seperti menambahkan teks dan *bounding box* pada citra hasil klasifikasi atau deteksi objek. Hal ini sangat berguna dalam sistem monitoring berbasis citra karena memudahkan pengguna dalam menginterpretasikan hasil keluaran sistem secara visual.

Pada sistem pemantauan cuaca berbasis *Raspberry Pi*, *OpenCV* berperan penting dalam proses pengolahan citra dari kamera sebelum dikirim ke model klasifikasi seperti *MobileNetV2* dan *Custom CNN*. Gambar yang ditangkap oleh kamera akan diproses menggunakan *OpenCV* untuk diubah ukurannya sesuai kebutuhan model, serta dinormalisasi agar sesuai dengan format input. Selain itu, *OpenCV* juga digunakan untuk menambahkan informasi hasil klasifikasi cuaca, jenis awan, dan *visibility* secara langsung ke citra menggunakan teks, sehingga informasi yang dihasilkan dapat dilihat secara langsung oleh pengguna melalui antarmuka *website*.

Lebih lanjut, *OpenCV* juga dimanfaatkan dalam *estimasi visibility* dengan pendekatan variasi Laplacian. Metode ini menggunakan *OpenCV* untuk menghitung tingkat ketajaman gambar berdasarkan variansi hasil operasi *Laplacian*, yang merepresentasikan seberapa tajam atau buram suatu citra. Semakin tinggi nilai *variansi Laplacian*, maka semakin jelas gambar tersebut, yang diinterpretasikan sebagai *visibility* yang baik. Sebaliknya, nilai rendah menunjukkan kabut atau gangguan visual lain. Oleh karena itu, penggunaan *OpenCV* dalam sistem ini bukan hanya sebatas *preprocessing*, melainkan menjadi bagian integral dalam keseluruhan proses klasifikasi dan estimasi berbasis citra kamera.

### 2.2.15 Flask

*Flask* merupakan salah satu *micro web framework* yang ringan dan fleksibel yang ditulis dalam bahasa *Python*. *Flask* dirancang dengan filosofi kesederhanaan dan minimalisme, memungkinkan pengembang untuk membangun *aplikasi website* secara cepat dan efisien tanpa perlu mengikuti struktur yang kaku. Meski bersifat “*micro*”, *Flask* memiliki kapabilitas yang kuat untuk membangun aplikasi *web* dinamis dengan dukungan berbagai ekstensi, seperti pengelolaan *database*, autentikasi, dan pengiriman *file*. Kemudahan dalam integrasi dengan berbagai pustaka *Python* lainnya menjadikan *Flask* populer dalam pengembangan aplikasi berbasis *machine learning* dan *Internet of Things (IoT)* (Shafira, 2024).

Dalam konteks sistem pemantauan cuaca *real-time* berbasis *Raspberry Pi*, *Flask* digunakan sebagai *backend* dari aplikasi *website* yang berfungsi menerima data klasifikasi cuaca, awan, dan *estimasi visibility* yang dikirimkan dari perangkat *Raspberry Pi*. *Flask* menangani permintaan *HTTP POST* yang dikirim oleh *Raspberry Pi*, menyimpan informasi yang diterima, dan kemudian menyajikannya dalam bentuk halaman *web* yang dapat diakses oleh pengguna. Peran *Flask* di sini sangat penting karena menjadi jembatan antara proses inferensi di sisi *edge device* dan penyajian informasi kepada pengguna melalui *browser*.

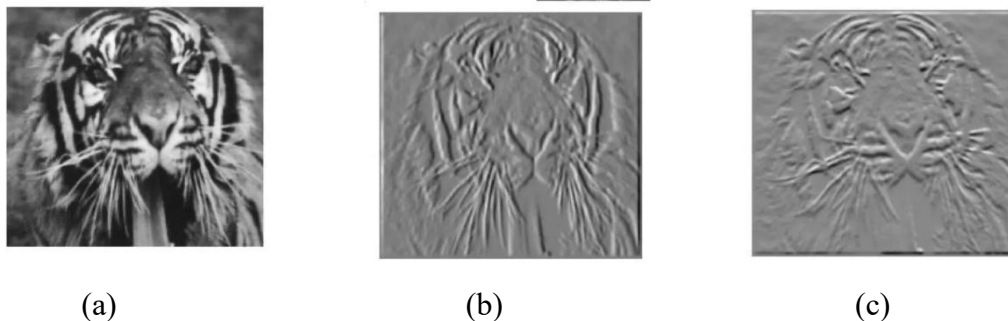
Salah satu keunggulan *Flask* adalah kemampuannya dalam menangani *file upload*, yang digunakan untuk menerima citra yang telah dianotasi oleh sistem klasifikasi di *Raspberry Pi*. Gambar tersebut kemudian disimpan di server dan ditampilkan dalam halaman antarmuka *web* bersamaan dengan data hasil klasifikasi cuaca, jenis awan, serta *estimasi visibility*. Hal ini memungkinkan pengguna untuk memantau kondisi cuaca terkini secara visual dan tekstual melalui satu platform terpadu. Dengan demikian, *Flask* mempermudah proses integrasi antara kecerdasan buatan di sisi *edge* dan penyajian data secara *real-time* di sisi pengguna.

Selain itu, *Flask* juga menyediakan fleksibilitas dalam pengembangan tampilan antarmuka melalui integrasi dengan *HTML*, *CSS*, dan *JavaScript*. Dalam proyek ini, *Flask* digunakan bersama *template engine* seperti *Jinja2* untuk menampilkan data klasifikasi secara dinamis. *Flask* juga memungkinkan pembaruan halaman secara

periodik dengan menggunakan metode pemanggilan ulang data secara otomatis (*auto-refresh*), sehingga informasi yang disajikan selalu *up-to-date* tanpa perlu interaksi manual dari pengguna. Dengan kemampuannya yang efisien, *Flask* menjadi solusi ideal untuk membangun sistem monitoring cuaca *real-time* berbasis *website* yang ringan namun fungsional.

### 2.2.16 Filter Laplacian

*Filter Laplacian* digunakan dalam pengolahan citra untuk mendeteksi area dengan perubahan intensitas yang tajam, terutama pada batas atau tepi objek. Berbasis prinsip turunan kedua, *filter* ini mampu mengidentifikasi lokasi perubahan mendadak dalam nilai piksel, menjadikannya sangat efektif dalam menyoroti kontur dan struktur penting dalam sebuah gambar. Tidak seperti *filter* deteksi tepi lain yang hanya mempertimbangkan satu arah (horizontal atau vertikal), *Laplacian* bekerja secara isotropik, yaitu memperhatikan perubahan dari berbagai arah sekaligus. Hal ini memungkinkan *filter* ini menangkap tepi dengan lebih lengkap dan menyeluruh. Namun, karena sensitivitasnya yang tinggi terhadap noise atau gangguan kecil dalam citra, penggunaan *filter Laplacian* sering didahului oleh proses *smoothing*, seperti *Gaussian blur*, untuk menekan efek *noise*. Dalam praktiknya, *filter* ini banyak digunakan dalam berbagai aplikasi, mulai dari segmentasi objek, pengenalan pola, hingga sistem navigasi visual, karena kemampuannya dalam memperkuat detail dan struktur visual yang penting dalam analisis citra digital (Paris *et al*, 2015).



**Gambar 2.12** (a) Tanpa Filter, (b) Filter Gradien-x, Dan (c) Filter Gradien-y (Paris *et al*, 2015).

**Gambar 2.12**, memperlihatkan tiga kondisi citra yang digunakan dalam proses deteksi tepi, yaitu (a) citra asli tanpa *filter* sebagai acuan, (b) hasil penerapan *filter gradien-x* yang menonjolkan perubahan intensitas piksel secara horizontal untuk mendeteksi tepi vertikal, dan (c) hasil filter *gradien-y* yang menyoroti perubahan intensitas secara vertikal untuk mendeteksi tepi horizontal pada citra. Ketiga gambar ini menunjukkan bagaimana masing-masing *filter* bekerja dalam mengekstraksi fitur arah tertentu dari citra. Berikut tahapan-tahapan perhitungan tersebut:

### 1 Rumus Operator *Laplacian*

Operator *Laplacian* merupakan operator diferensial orde dua yang digunakan untuk mendeteksi perubahan tajam (tepi) dalam citra. Secara matematis, operator Laplacian terhadap fungsi intensitas citra  $L(x, y)$  didefinisikan sebagai Persamaan 2.7 dan Persamaan 2.8:

$$L(x, y) = \nabla^2 I(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \dots\dots\dots (2.7)$$

$$L(x, y) = [I(x + 1, y) + I(x - 1, y) + I(x, y + 1) + I(x, y - 1)] - 4I(x, y) \dots\dots\dots (2.8)$$

Persamaan 2.7 dan Persamaan 2.8, menghitung turunan kedua dari intensitas piksel terhadap arah horizontal (x) dan vertikal (y). Hasil dari operasi ini adalah matriks dua dimensi (2D) dari nilai-nilai *Laplacian* di  $L(x, y)$  seluruh titik gambar.

Operator ini sangat sensitif terhadap perbedaan nilai intensitas, sehingga sangat baik digunakan untuk mengidentifikasi area perubahan intensitas yang tajam, yaitu tepi objek.

### 2 Statistik Rata-Rata *Nilai Laplacian (Mean)*

Setelah nilai Laplacian dihitung untuk seluruh piksel dalam citra, langkah selanjutnya adalah menghitung rata-rata (*mean*) dari nilai-nilai tersebut. Rata-rata

ini melambangkan tingkat ketajaman umum pada seluruh citra dalam Persamaan 2.9:

$$\mu = \frac{1}{NM} \sum_{x=1}^M \sum_{y=1}^N L(x, y) \dots\dots\dots (2.9)$$

Persamaan 2.9, menjelaskan komponen-komponen yang digunakan dalam perhitungan statistik nilai *Laplacian* pada citra digital.  $L(x, y)$  merupakan nilai *Laplacian* di titik koordinat  $(x, y)$ , yang mencerminkan tingkat perubahan intensitas piksel pada area tersebut dan digunakan untuk menandai keberadaan tepi atau kontur dalam gambar. Ukuran citra dinyatakan dalam  $M \times N$  yang menunjukkan jumlah total piksel berdasarkan panjang dan lebar gambar. Sementara itu,  $\mu$  adalah nilai rata-rata dari seluruh  $L(x, y)$  yang ada dalam citra, yang merepresentasikan tingkat ketajaman umum atau intensitas tepi secara menyeluruh. Rata-rata ini menjadi acuan dalam perhitungan variansi, untuk mengetahui sejauh mana nilai-nilai *Laplacian* tersebar atau menyimpang dari nilai pusatnya, yang pada akhirnya digunakan untuk menilai tingkat visibilitas atau kejernihan citra.

### 3 Statistik Variansi Nilai *Laplacian*

Setelah diketahui rata-rata  $\mu$ , tahap akhir adalah menghitung variansi dari nilai-nilai *Laplacian*. Variansi ini menyatakan seberapa besar penyebaran nilai ketajaman (*edge*) terhadap nilai rata-ratanya. Semakin tinggi variansi, semakin banyak perubahan tajam dalam citra, yang berarti visibilitas tinggi.

Rumus umum variansi untuk suatu himpunan nilai  $X_i$ , dalam Persamaan 2.10:

$$Var(X) = \frac{1}{N} \sum_{i=1}^n (X_i - \bar{X})^2 \dots\dots\dots (2.10)$$

Persamaan 2.10, digunakan untuk menghitung variansi, yaitu ukuran statistik yang menunjukkan seberapa besar penyebaran atau keragaman suatu kumpulan data terhadap nilai rata-ratanya. Dalam rumus ini,  $X_i$  adalah nilai individual ke- $i$  dalam himpunan data,  $\bar{X}$  adalah nilai rata-rata (*mean*) dari seluruh data, dan  $N$  adalah jumlah total data. Proses perhitungan dilakukan dengan mengurangi setiap nilai  $X_i$  dari rata-rata  $\bar{X}$ , mengkuadratkan hasil selisih tersebut agar semua nilai menjadi

positif, lalu menjumlahkannya dan membaginya dengan jumlah data N. Hasil akhir dari *variansi* menunjukkan seberapa besar nilai-nilai dalam himpunan tersebut tersebar dari pusatnya. Semakin besar nilai variansi, semakin besar penyimpangan data dari rata-rata; sebaliknya, jika variansi kecil, maka data cenderung berdekatan atau homogen di sekitar rata-ratanya.

Jika diterapkan pada peta *Laplacian*, maka variansi nilai *Laplacian*, dalam Persamaan 2.11:

$$Var(L) = \frac{1}{MN} \sum_{x=1}^M \sum_{y=1}^N (L(x,y) - \mu)^2 \dots\dots\dots (2.11)$$

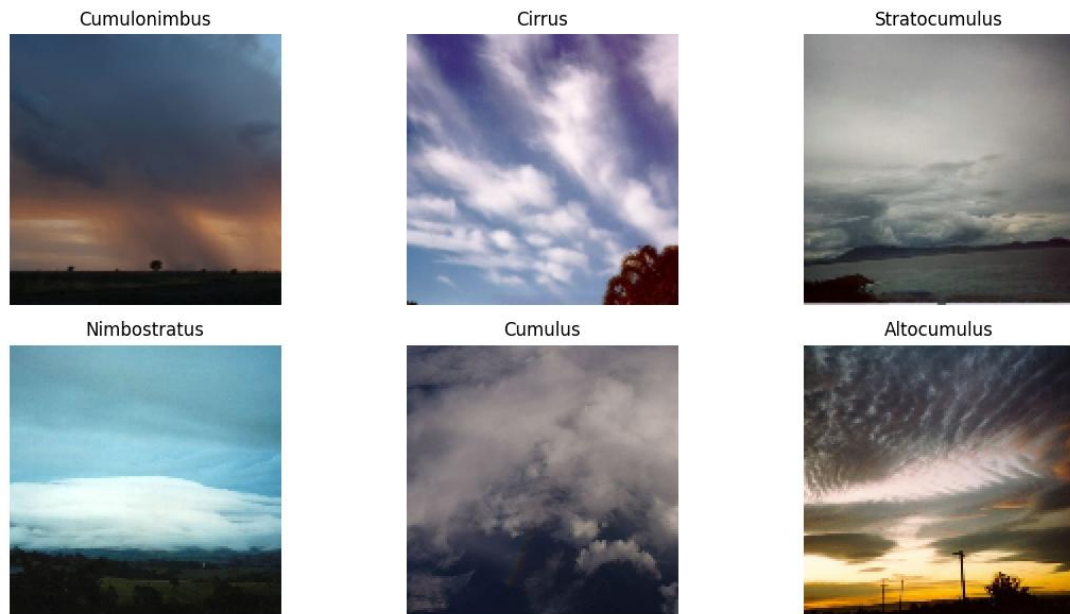
Persamaan 2.11 digunakan untuk menghitung variansi dari nilai-nilai *Laplacian* pada seluruh piksel citra berukuran M×N. Dalam rumus ini, L(x,y) adalah nilai *Laplacian* pada titik koordinat (x,y), yang mencerminkan perubahan intensitas atau ketajaman pada area tersebut, sedangkan  $\mu$  adalah rata-rata dari seluruh nilai *Laplacian* dalam citra. Proses perhitungan dilakukan dengan mengurangi setiap nilai *Laplacian* dari rata-ratanya, mengkuadratkan hasil selisih tersebut untuk menghilangkan nilai negatif, lalu menjumlahkan seluruh hasil kuadrat dan membaginya dengan jumlah total piksel dalam citra, yaitu M×N. Hasil dari variansi ini menunjukkan tingkat penyebaran nilai ketajaman dalam gambar. Jika nilai variansi tinggi, berarti banyak variasi atau perbedaan tajam antar piksel, yang menandakan citra memiliki visibilitas atau ketajaman tinggi. Sebaliknya, jika variansi rendah, maka citra cenderung datar atau kabur karena sedikit perubahan intensitas yang terdeteksi.

Variansi dari operator *Laplacian* dihitung untuk mengukur tingkat ketajaman citra atau kejelasan detail dalam gambar. Dalam pengolahan citra digital, operator *Laplacian* digunakan untuk mendeteksi tepi (*edges*) dengan cara menghitung turunan kedua dari intensitas piksel. Tepi merupakan perubahan nilai intensitas yang signifikan dan sering dikaitkan dengan batas objek dalam gambar. Ketajaman citra sangat berkaitan dengan jumlah dan kekuatan tepi yang terdeteksi.

## 2.17 Awan

Awan terbentuk dari butiran air atau kristal es yang berada di atmosfer akibat proses kondensasi uap air. Ketika udara lembap naik ke lapisan atmosfer yang lebih tinggi, suhu menurun dan uap air di dalamnya mencapai titik jenuh, lalu berubah menjadi partikel cair atau es. Proses ini sangat dipengaruhi oleh kondisi fisik atmosfer seperti suhu, tekanan udara, dan kelembaban relatif. Keberadaan awan menjadi salah satu indikator utama dalam pengamatan cuaca karena dapat mencerminkan dinamika atmosfer yang sedang berlangsung. Jenis dan bentuk awan yang tampak di langit sering kali menjadi petunjuk awal terhadap kemungkinan terjadinya hujan, badai, atau bahkan kondisi cerah yang berkepanjangan (Christiani, 2018).

Klasifikasi awan secara umum dilakukan berdasarkan ketinggian dan bentuknya. Menurut Organisasi Meteorologi Dunia, *World Meteorological Organization* (WMO), awan diklasifikasikan ke dalam tiga kelompok ketinggian: tinggi, menengah, dan rendah. Awan pada lapisan tinggi biasanya tersusun atas kristal es dan tampak halus atau berserat, sedangkan awan pada lapisan menengah dan rendah lebih banyak mengandung tetesan air dan tampak lebih padat. Dari sisi bentuk, awan dapat dikenali sebagai *cirrus* yang tipis dan serat seperti, *stratus* yang membentuk lapisan luas, atau *cumulus* yang menggumpal tebal seperti kapas. Variasi ini menjadi dasar dalam proses klasifikasi visual awan. Seiring kemajuan teknologi, pengenalan bentuk awan tidak lagi terbatas pada pengamatan manual, melainkan dapat dilakukan secara otomatis menggunakan metode berbasis pengolahan citra dan kecerdasan buatan, seperti *Convolutional Neural Network* (CNN). Pemanfaatan klasifikasi awan secara otomatis sangat mendukung sistem pemantauan cuaca *real-time*, terutama dalam rangka mitigasi cuaca ekstrem dan peningkatan akurasi prediksi (Kristanto *et al.*, 2017).



**Gambar 2.13** *Dataset* Gambar Awan Untuk Data Latih Model *MobileNetV2* (Christiani, 2018).

**Gambar 2.13** memperlihatkan contoh dataset gambar awan yang digunakan sebagai data latih untuk model klasifikasi *MobileNetV2*. Terdapat enam jenis awan yang ditampilkan, masing-masing merepresentasikan kelas berbeda yang dikenali berdasarkan ciri visualnya. *Cumulonimbus* (*Cb*) tampak sebagai awan besar yang menjulang tinggi dengan warna gelap, sering dikaitkan dengan hujan lebat dan badai petir. *Cirrus* (*Ci*) terlihat sebagai awan tinggi yang tipis dan berserat, biasanya muncul dalam kondisi cuaca cerah pada ketinggian lebih dari 6.000 meter. *Stratocumulus* (*Sc*) tampak sebagai lapisan awan rendah yang tebal dan bergelombang, menutupi sebagian besar langit dengan warna abu-abu. *Nimbostratus* (*Ns*) merupakan awan yang sangat tebal dan gelap, menutupi langit secara merata dan sering menghasilkan hujan terus-menerus. *Cumulus* (*Cu*) ditandai dengan bentuk gumpalan putih tebal seperti kapas dengan dasar datar, yang umumnya muncul pada cuaca cerah. Sementara itu, *Altocumulus* (*Ac*) terlihat sebagai awan menengah yang tersusun dalam gumpalan-gumpalan kecil, biasanya tampak di pagi atau sore hari dan dapat menjadi indikator perubahan cuaca. Keenam jenis awan ini digunakan untuk melatih model *MobileNetV2* dalam mengenali karakteristik visual dan pola tekstur khas dari masing-masing kelas, sehingga

mendukung sistem klasifikasi awan otomatis dalam aplikasi pemantauan cuaca *real-time*.

### BAB III METODE PENELITIAN

#### 3.1. Tempat dan Waktu Penelitian

Penelitian ini akan dilaksanakan di dua lokasi utama, yaitu Stasiun Meteorologi Raden Intan II, yang merupakan salah satu fasilitas meteorologi di Lampung untuk pengumpulan data cuaca dan pengujian perangkat *IoT*, serta di Jurusan Fisika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Lampung, sebagai pusat untuk pengolahan data, pengembangan *model deep learning*, dan implementasi sistem. Penelitian ini direncanakan berlangsung mulai Desember 2024 hingga Mei 2025.

#### 3.2 Alat dan Bahan

Alat dan bahan yang di gunakan pada penelitian ini dapat dilihat pada **tebel 3.1**

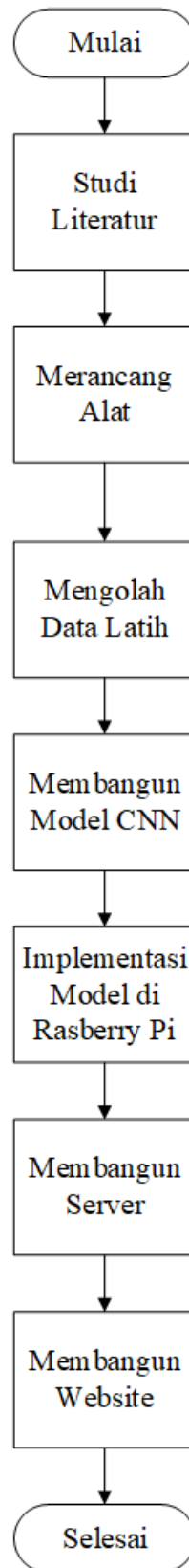
**Tabel 3.1** Alat dan bahan penelitian

| No | Nama Alat                         | Fungsinya                                                                                                                                                                                                                                       |
|----|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1  | <i>Raspberry Pi 5</i>             | Berfungsi sebagai unit pemrosesan utama dalam sistem klasifikasi awan. Perangkat ini akan menjalankan model CNN ( <i>MobileNetV2</i> ) untuk mengolah citra awan dan mengirimkan hasil klasifikasi ke sistem pemantauan berbasis <i>website</i> |
| 2  | Modul Kamera <i>Board Rev 1.3</i> | Digunakan untuk mengambil gambar awan secara <i>real-time</i> . Kamera ini terhubung ke <i>Raspberry Pi</i> dan menangkap citra yang akan digunakan dalam proses klasifikasi awan.                                                              |

| No | Nama Alat                     | Fungsinya                                                                                                                                                                                                                                                                                                                                                       |
|----|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 3  | <i>Power Bank</i>             | Berfungsi sebagai sumber daya utama bagi <i>Raspberry Pi 5</i> , memastikan perangkat mendapatkan tegangan dan arus listrik yang stabil untuk menjalankan sistem secara optimal.                                                                                                                                                                                |
| 4  | Micro SD Card                 | Berfungsi sebagai media penyimpanan utama untuk sistem operasi <i>Raspberry Pi</i> serta program pendukung, termasuk model CNN dan dataset yang digunakan dalam penelitian ini.                                                                                                                                                                                 |
| 5  | <i>Personal Computer (PC)</i> | PC ( <i>Personal Computer</i> ) digunakan sebagai antarmuka utama untuk pemrograman, pemantauan data secara <i>real-time</i> , serta pengolahan dan analisis hasil pengukuran dari sensor yang terhubung ke <i>Raspberry Pi 5</i> , sekaligus berperan sebagai pusat kendali dalam proses pengembangan dan evaluasi sistem klasifikasi awan berbasis <i>CNN</i> |
| 6  | <i>Cooler</i>                 | <i>Cooler</i> berfungsi untuk menjaga suhu <i>Raspberry Pi</i> tetap stabil selama proses inferensi model agar mencegah <i>overheating</i> dan memastikan kinerja sistem tetap optimal.                                                                                                                                                                         |

### 3.3 Prosedur Penelitian

Prosedur penelitian secara umum dapat diperlihatkan dalam diagram alir penelitian pada **gambar 3.1**



**Gambar 3.1** Diagram Alir Penelitian.

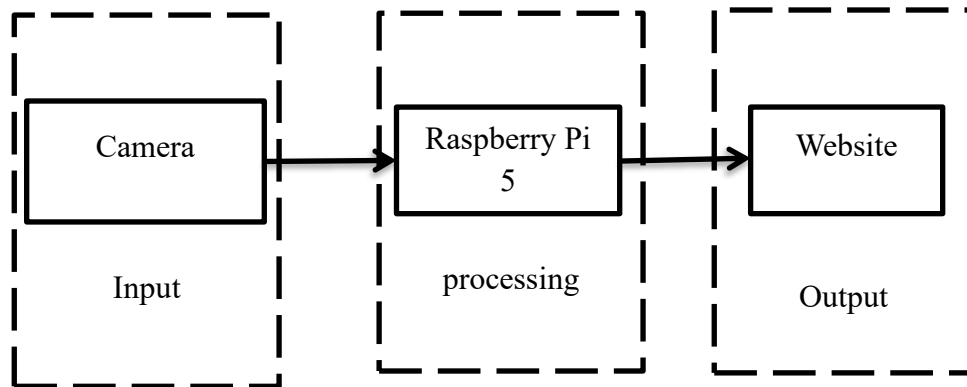
Untuk mencapai tujuan penelitian, prosedur penelitian dijelaskan lebih rinci sebagai berikut.

### 3.3.1 Studi Literatur

Studi literatur dilakukan untuk mengkaji teori dan penelitian terdahulu yang berkaitan dengan klasifikasi awan, penerapan *Convolutional Neural Network* (CNN), serta implementasi sistem berbasis *Raspberry Pi* dan *website* sebagai media pemantauan. Kajian ini mencakup pemahaman mengenai jenis-jenis awan dan pengaruhnya terhadap perubahan cuaca, serta metode klasifikasi yang telah digunakan, baik secara konvensional maupun berbasis kecerdasan buatan. Selain itu, penelitian ini meninjau konsep dasar CNN dalam pengolahan citra, khususnya penerapan *MobileNetV2* sebagai model yang efisien untuk perangkat dengan keterbatasan daya komputasi. Studi juga dilakukan terhadap penggunaan *Raspberry Pi* sebagai platform pemrosesan model AI, termasuk kelebihan dan tantangan dalam implementasinya. Dengan melakukan studi literatur ini, diharapkan penelitian dapat mengembangkan sistem klasifikasi awan yang lebih akurat, efisien, dan dapat digunakan secara luas oleh masyarakat dan peneliti.

### 3.3.2 Perancangan Sistem Deteksi

Sistem identifikasi citra awan terdiri dari tiga bagian utama: *input*, proses, dan *output*. Bagian input menggunakan Kamera *Board Rev 1.3*, yang berfungsi untuk menangkap citra awan secara *real-time*. Kamera ini memiliki resolusi 5 MP dan terhubung ke *Raspberry Pi 5* sebagai pusat pemrosesan data. Citra yang diambil akan dikirimkan ke *Raspberry Pi* untuk diproses menggunakan algoritma klasifikasi berbasis *Convolutional Neural Network*, yang bertujuan untuk mengenali jenis awan berdasarkan karakteristik visualnya. Gambar 3.2 menunjukkan alur kerja sistem ini.



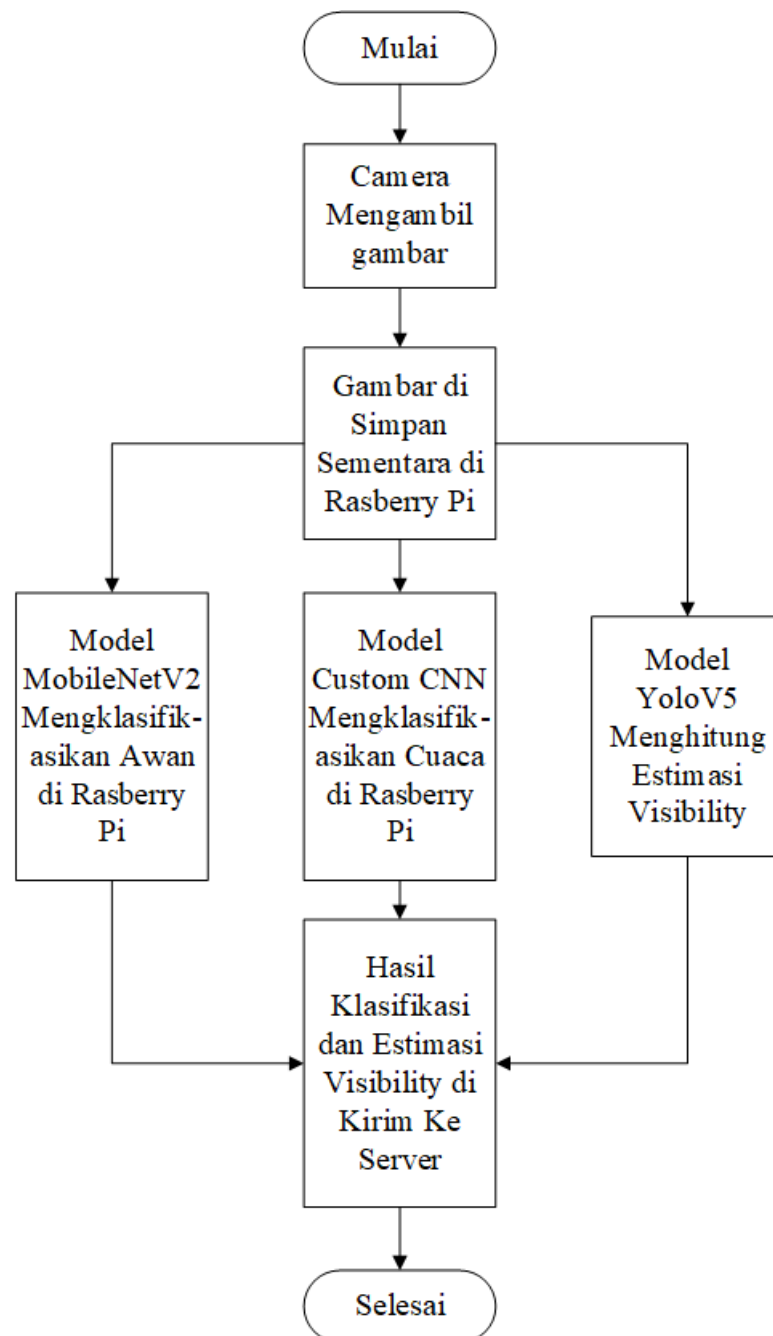
**Gambar 3.2** Diagram *Blok* Sistem Identifikasi Citra Awan.

Setelah proses klasifikasi selesai, hasilnya akan ditampilkan melalui antarmuka *website* sebagai bagian dari sistem output. *Website* ini memungkinkan pengguna untuk melihat hasil identifikasi awan secara langsung dan mendapatkan informasi terkait kondisi cuaca berdasarkan analisis yang telah dilakukan. Dengan adanya sistem berbasis *website*, pengguna dapat mengakses data identifikasi awan dari berbagai perangkat yang terhubung ke *internet*, sehingga sistem ini dapat digunakan secara luas oleh masyarakat dan peneliti untuk keperluan pemantauan cuaca dan penelitian *atmosfer*.

### 3.4 Perancangan Sistem Pada *Raspberry Pi 5*

Sistem pada penelitian ini dirancang untuk melakukan klasifikasi jenis awan secara otomatis berdasarkan citra langit yang diambil secara *real-time* menggunakan kamera, dengan tujuan menghasilkan estimasi cuaca yang lebih cepat dan akurat. Sistem tidak hanya bergantung pada model klasifikasi citra, namun juga dilengkapi dengan sensor lingkungan seperti suhu, kelembapan, tekanan udara, dan pencahayaan untuk memperkaya analisis cuaca secara menyeluruh. Model klasifikasi citra awan dikembangkan menggunakan arsitektur *Custom Convolutional Neural Network (CNN)*, serta memanfaatkan pendekatan transfer learning dengan *MobileNetV2* sebagai backbone utama. *MobileNetV2* dipilih karena kemampuannya dalam mengekstraksi fitur visual kompleks secara efisien, sangat cocok untuk implementasi pada sistem berbasis *Internet of Things (IoT)* dengan keterbatasan komputasi. Seluruh proses akuisisi dan pemrosesan data

dijalankan oleh *Raspberry Pi 5* yang berperan sebagai pusat kendali sistem, menangani akuisisi data sensor, inferensi model CNN, hingga pengambilan keputusan secara lokal tanpa ketergantungan pada server eksternal. Dengan demikian, sistem mampu merespons perubahan cuaca secara *real-time*, menjaga akurasi *estimasi visibility*, serta memberikan solusi pemantauan cuaca yang tangguh, efisien, dan fleksibel untuk berbagai kondisi lapangan.



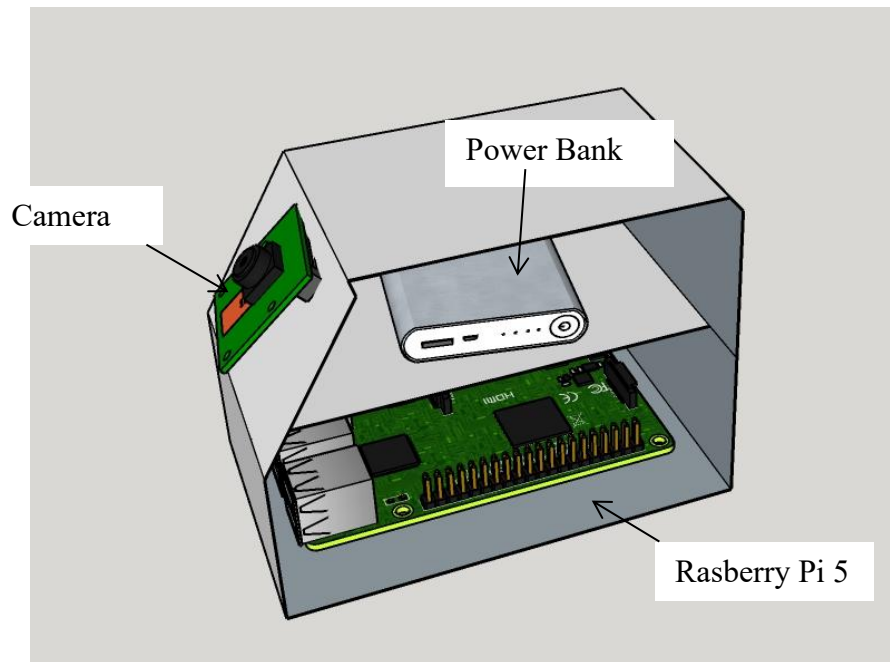
**Gambar 3.3** Diagram Skematik Sistem Pada *Raspberry Pi 5*.

**Gambar 3.3** menunjukkan diagram skematik sistem yang dikembangkan. Kamera menangkap gambar awan yang kemudian diproses oleh *Raspberry Pi* menggunakan model CNN. ditampilkan melalui antarmuka web sebagai tambahan informasi kondisi cuaca.

Integrasi ini memungkinkan sistem tidak hanya melakukan klasifikasi awan secara *real-time*, tetapi juga memberikan konteks kondisi atmosfer saat pengambilan gambar. Hal ini meningkatkan akurasi interpretasi cuaca serta mendukung pengambilan keputusan dalam berbagai aplikasi, seperti pendidikan, pertanian, atau mitigasi bencana berbasis data cuaca.

### 3.5 Rancangan Sistem dalam Bentuk 3D

Sistem yang dirancang dalam penelitian ini mengintegrasikan *Raspberry Pi 5* sebagai pusat pemrosesan utama, yang mengelola alur kerja perangkat mulai dari pengambilan data hingga pemrosesan dan pengiriman hasil. *Raspberry Pi 5* dipilih karena performa komputasinya yang tinggi dan dukungan terhadap konektivitas jaringan stabil melalui *kabel LAN* atau *Wi-Fi*, memungkinkan pemantauan jarak jauh. Kamera *Raspberry Pi Camera Module v1.3* dipasang di bagian atas perangkat untuk menangkap citra awan secara optimal dan terhubung langsung melalui *port CSI* pada *Raspberry Pi 5*. Citra yang diambil kemudian diproses menggunakan algoritma *Convolutional Neural Network (CNN)* untuk mengklasifikasikan jenis awan dan memprediksi cuaca. Dengan desain kompak ini, sistem dapat memberikan solusi pemantauan cuaca *real-time* yang efisien, yang dapat diterapkan di wilayah dengan keterbatasan infrastruktur meteorologi.



**Gambar 3.4** Rancangan Sistem dalam Bentuk 3D.

**Gambar 3.4** menunjukkan bentuk keseluruhan perangkat yang dirancang dalam model tiga dimensi. Kotak pelindung digunakan sebagai wadah utama untuk menempatkan seluruh komponen elektronik, memberikan perlindungan terhadap faktor eksternal seperti debu, air, dan perubahan suhu. Kotak ini dirancang dengan material tahan cuaca dan dilengkapi lubang ventilasi untuk menjaga sirkulasi udara yang baik di dalam perangkat. Posisi kamera diletakkan pada bagian atas kotak pelindung dengan bidang pandang terbuka ke langit, sehingga dapat menangkap citra secara optimal. Penempatan setiap komponen diatur dengan cermat agar semua elemen sistem dapat bekerja secara sinergis tanpa saling mengganggu, baik secara fungsional maupun struktural. Dengan rancangan ini, sistem tidak hanya optimal dari sisi teknis dan fungsional, tetapi juga praktis dan tahan terhadap kondisi lingkungan, memungkinkan operasional jangka panjang dalam pengamatan cuaca berbasis citra secara *real-time*.

### 3.6 Komponen Alat Klasifikasi Cuaca

Rangkaian alat yang digunakan dalam penelitian ini terdiri dari beberapa komponen utama, yaitu *Raspberry Pi 5*, modul kamera, dan *power bank*. *Raspberry Pi 5*

berfungsi sebagai pusat kendali sistem yang mengelola seluruh proses pengambilan data, pengolahan citra, dan *inferensi model Convolutional Neural Network (CNN)* secara lokal. Perangkat ini dipilih karena kemampuannya dalam menangani beban komputasi tinggi serta kompatibilitasnya dengan berbagai modul kamera, sehingga memungkinkan proses klasifikasi citra awan dilakukan secara cepat dan efisien tanpa ketergantungan pada *server* eksternal. Modul kamera yang terhubung langsung ke *Raspberry Pi 5* digunakan untuk menangkap citra awan dalam berbagai kondisi cuaca secara *real-time*, dengan kualitas gambar yang dihasilkan menjadi faktor penting dalam mendukung tingkat akurasi proses klasifikasi. Untuk menunjang mobilitas dan keberlanjutan operasional sistem di lapangan, digunakan *power bank* sebagai sumber daya utama. Dengan penggunaan *power bank*, sistem dapat beroperasi secara portabel di berbagai lokasi tanpa bergantung pada sumber listrik konvensional, sehingga memperluas fleksibilitas dalam proses pengumpulan data. Kombinasi ketiga komponen ini membentuk sistem monitoring cuaca berbasis IoT yang sederhana namun efektif, dengan kemampuan melakukan akuisisi, pengolahan, dan klasifikasi citra awan secara *real-time* dalam berbagai kondisi lingkungan.

### **3.6.1 Raspberry Pi 5**

*Raspberry Pi 5* berfungsi sebagai unit pemrosesan utama dalam sistem ini, dilengkapi dengan prosesor *quad-core ARM Cortex-A76* yang memiliki kecepatan hingga *2.4 GHz* dan *RAM 4 GB*, memungkinkan pemrosesan data yang cepat dan efisien. *Raspberry Pi 5* mengelola pemrosesan citra dari kamera serta data dari sensor suhu, kelembapan, dan tekanan udara. Dengan kemampuannya dalam menjalankan model deep learning seperti *Custom CNN*, *MobileNetV2*, dan *YOLOv5* secara lokal, *Raspberry Pi 5* mendukung pengolahan citra cuaca secara *real-time*, menjadikannya pusat kendali yang efisien untuk sistem monitoring cuaca ini.

### **3.6.2 Kamera Board Rev 1.3**

*Kamera Board Rev 1.3* adalah perangkat yang digunakan untuk menangkap citra awan yang menjadi input utama dalam sistem klasifikasi cuaca. Memiliki resolusi *5 megapiksel*, kamera ini mampu menangkap gambar dengan kualitas baik

meskipun dalam kondisi pencahayaan yang bervariasi. Terhubung langsung ke *Raspberry Pi 5* melalui *port Camera Serial Interface (CSI)*, kamera ini memastikan transfer data citra cepat dan stabil, serta mendukung integrasi dengan pustaka pengolahan citra seperti *OpenCV* untuk mempermudah praproses citra secara *real-time* dalam klasifikasi dan *estimasi visibility*.

### **3.6.3 Personal Computer (PC)**

PC digunakan dalam pengembangan dan pengujian sistem, terutama untuk pemrograman, konfigurasi perangkat keras, serta pemantauan data sensor secara real-time. Selain itu, PC juga digunakan untuk pelatihan model klasifikasi berbasis CNN dan visualisasi hasil prediksi. Dengan kemampuan komputasi yang lebih tinggi dibandingkan *Raspberry Pi*, PC mendukung analisis data yang lebih mendalam, serta memungkinkan pengembangan dan evaluasi sistem dengan lebih efisien dalam berbagai skenario.

### **3.6.4 Power Bank**

Power bank dalam sistem ini menyediakan sumber daya portabel yang stabil bagi *Raspberry Pi 5* dan kamera, memungkinkan perangkat untuk beroperasi secara mandiri di luar ruangan tanpa ketergantungan pada sumber listrik utama. Dengan kapasitas yang mencukupi, *power bank* memastikan bahwa sistem dapat berfungsi dalam waktu lama, terutama selama proses pengambilan citra secara kontinu untuk monitoring cuaca, menjadikannya solusi daya yang andal untuk sistem portabel ini.

## **3.7 Pengumpulan Data**

Dalam penelitian ini, pengumpulan data dilakukan melalui beberapa tahap, yaitu sumber *dataset*, *preprocessing* data, dan teknik *augmentasi* data. Tahapan ini bertujuan untuk memastikan bahwa data yang digunakan dalam model memiliki kualitas yang baik dan sesuai untuk proses klasifikasi citra awan. *Dataset* dalam penelitian ini telah dikategorikan ke dalam enam jenis awan, yaitu *Cumulonimbus*, *Cirrus*, *Stratocumulus*, *Nimbostratus*, *Cumulus*, dan *Alto cumulus*.

### **3.7.1 Sumber Dataset**

Dataset yang digunakan dalam penelitian ini diperoleh dari *Kaggle*, yang menyediakan berbagai kumpulan data citra awan dengan label yang sudah ditentukan. *Dataset* ini berisi berbagai jenis awan yang diambil dari berbagai kondisi cuaca dan lokasi yang berbeda. Data yang diperoleh digunakan sebagai data latih dan data uji dalam pengembangan model klasifikasi awan.

### **3.7.2 Preprocessing Data**

Sebelum digunakan dalam pelatihan model, dataset perlu melalui proses preprocessing untuk meningkatkan kualitas data. Langkah-langkah preprocessing yang dilakukan dalam penelitian ini meliputi:

1. *Resize Citra* – Semua gambar dalam dataset diubah ukurannya agar memiliki dimensi yang seragam sesuai dengan kebutuhan model.
2. *Konversi Warna* – Gambar diubah menjadi format yang sesuai, seperti *grayscale* atau RGB, tergantung pada arsitektur model yang digunakan.
3. *Normalisasi* – Nilai piksel gambar dinormalisasi agar berada dalam rentang tertentu (misalnya 0 hingga 1) untuk mempercepat proses pelatihan model.
4. *Penghapusan Noise* – Teknik filtering digunakan untuk mengurangi noise yang dapat mengganggu proses klasifikasi.

### **3.7.3 Augmentasi Data**

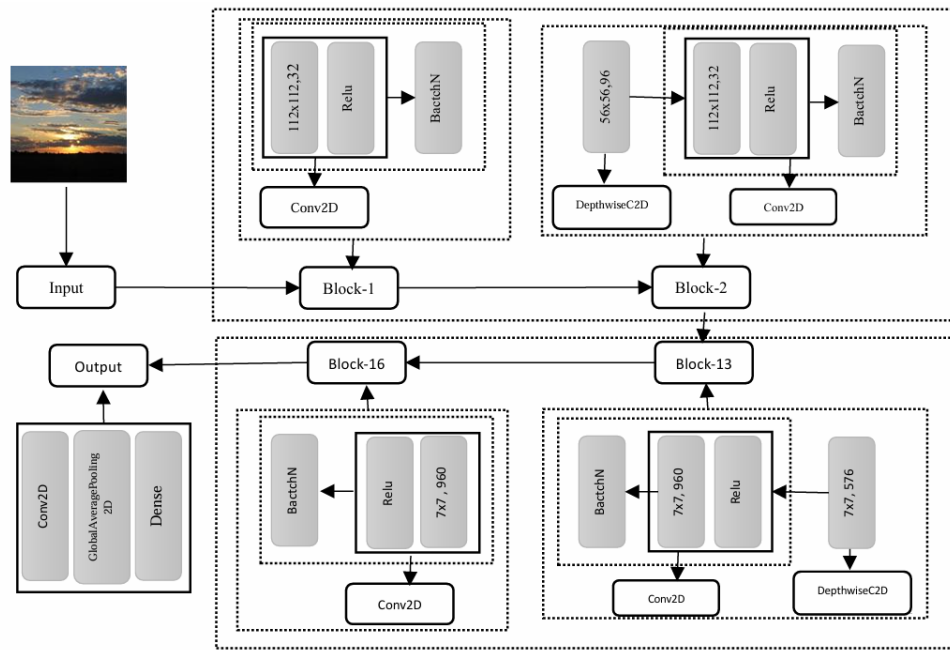
Untuk meningkatkan jumlah dan variasi data dalam dataset, dilakukan proses augmentasi data. Teknik augmentasi yang digunakan meliputi:

1. *Rotasi dan Flip* – Gambar diputar atau dibalik untuk meningkatkan variasi sudut pandang.
2. *Brightness Adjustment* – Tingkat kecerahan gambar diubah untuk mensimulasikan berbagai kondisi pencahayaan.
3. *Zooming dan Cropping* – Beberapa bagian gambar diperbesar atau dipotong untuk menciptakan variasi tambahan.

Dengan menerapkan *preprocessing* dan *augmentasi* data, model dapat dilatih dengan dataset yang lebih berkualitas dan memiliki kemampuan generalisasi yang lebih baik terhadap data baru.

### 3.8 Arsitektur Model CNN MobileNetV2

Model *Convolutional Neural Network (CNN)* yang digunakan dalam penelitian ini adalah *MobileNetV2*. Arsitektur ini dikenal karena efisiensinya dalam hal komputasi dan kemampuannya dalam mengekstraksi fitur visual secara efektif. *MobileNetV2* menerima citra awan berformat RGB dengan resolusi  $224 \times 224$  piksel sebagai *input*. Sebelum masuk ke model, citra terlebih dahulu dinormalisasi ke rentang  $[0,1]$  untuk memastikan proses pelatihan berjalan stabil dan akurat. Struktur utama *MobileNetV2* terdiri dari *depthwise separable convolution* dan *inverted residual blocks*. *Depthwise separable convolution* memisahkan proses konvolusi *spasial* dan *kanal*, sehingga mengurangi jumlah parameter secara signifikan. *Inverted residual blocks* menjaga informasi penting dalam citra tanpa meningkatkan kompleksitas model, sehingga tetap ringan namun efektif. Lapisan *fully connected* berfungsi untuk memetakan hasil ekstraksi fitur menjadi enam kelas awan yang ditentukan. Akhirnya, *output layer* menggunakan fungsi aktivasi *Softmax* untuk mengklasifikasikan citra ke dalam kategori: *Cumulonimbus*, *Cirrus*, *Stratocumulus*, *Nimbostratus*, *Cumulus*, dan *Alto cumulus*.



**Gambar 3.5** Arsitektur Model CNN MobileNetV2.

### 3.8.1 Proses *Training* dan *Evaluasi* Model CNN MobileNetV2

Proses pelatihan model diawali dengan tahap *preprocessing*, yaitu mengubah ukuran citra menjadi  $224 \times 224$  piksel dan melakukan normalisasi dengan membagi nilai RGB dengan 255. Untuk memperkaya variasi data dan mengurangi risiko *overfitting*, dilakukan augmentasi seperti rotasi, *flipping horizontal*, dan penyesuaian kecerahan. Model dilatih menggunakan algoritma optimasi Adam dengan penyesuaian *learning rate* untuk mencapai konvergensi optimal. Fungsi *loss* yang digunakan adalah *categorical crossentropy*, sesuai dengan jenis klasifikasi multi-kelas. *Dataset* dibagi menjadi 80% untuk data latih dan 20% untuk data validasi guna memantau performa model secara berkala selama proses pelatihan. Setelah pelatihan selesai, model diuji menggunakan metrik evaluasi seperti akurasi, *precision*, *recall*, dan *F1-score*. Selain itu, digunakan juga *confusion matrix* untuk menganalisis klasifikasi yang salah serta melihat kategori mana yang sering tertukar antar kelas.

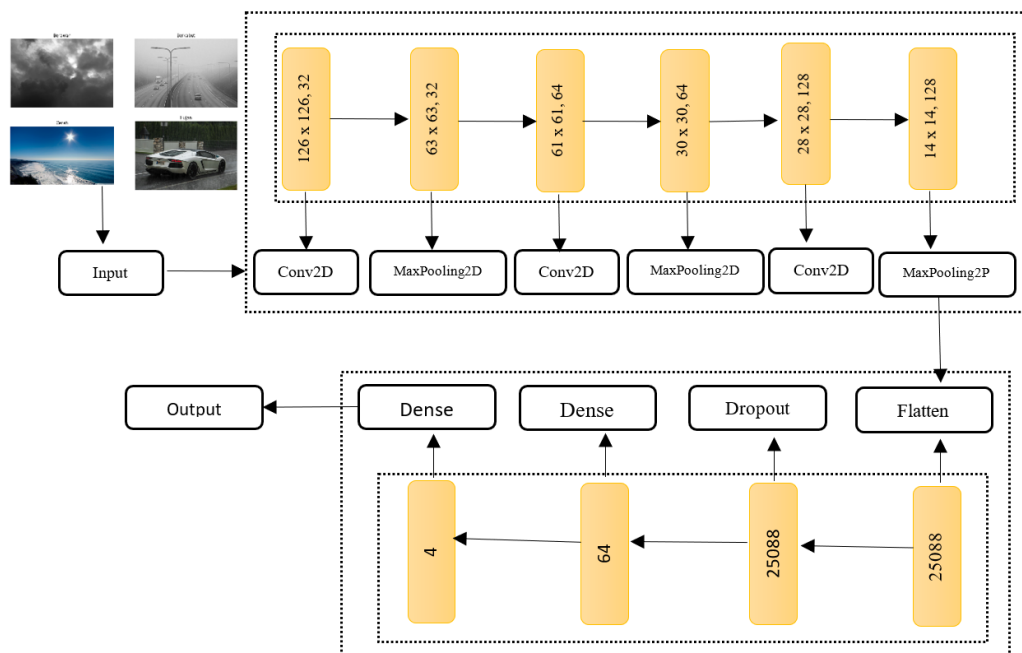
### 3.8.2 Implementasi *Transfer Learning* dengan *MobileNetV2*

Implementasi *transfer learning* dilakukan dengan memanfaatkan *MobileNetV2* yang sudah dilatih sebelumnya menggunakan dataset *ImageNet*. Lapisan akhir dari model tersebut dihapus agar dapat disesuaikan dengan kebutuhan klasifikasi enam kategori awan. Kemudian ditambahkan *Global Average Pooling* untuk mereduksi dimensi hasil ekstraksi fitur tanpa kehilangan informasi penting. Selanjutnya, ditambahkan *Dense Layer* dengan aktivasi *ReLU* untuk meningkatkan kapabilitas klasifikasi, serta *output layer* dengan fungsi aktivasi *Softmax* yang memiliki enam *neuron* sesuai jumlah kelas awan. Dengan pendekatan *transfer learning* ini, model dapat mempelajari fitur spesifik citra awan dengan lebih efisien dan membutuhkan lebih sedikit data pelatihan dibandingkan jika menggunakan CNN yang dilatih dari awal (*from scratch*). Hal ini memungkinkan model memperoleh akurasi tinggi meskipun dataset citra awan terbatas.

## 3.9 Arsitektur *Custom CNN*

### 3.9.1 Arsitektur *Model Custom CNN*

Custom CNN yang digunakan dalam penelitian ini dirancang secara manual untuk mengklasifikasikan citra cuaca menjadi empat kategori utama, yaitu cerah, berawan, hujan, dan berkabut. Tujuan utama dari desain model ini adalah menciptakan arsitektur yang ringan namun tetap mampu menghasilkan performa klasifikasi yang baik, sehingga dapat diterapkan pada perangkat dengan keterbatasan daya komputasi seperti *Raspberry Pi*.



**Gambar 3.6** Arsitektur *Model Custom CNN*.

Model menerima *input* berupa citra cuaca berwarna (RGB) berukuran  $128 \times 128$  piksel. Ukuran ini dipilih untuk menjaga keseimbangan antara detail visual yang cukup dan efisiensi komputasi. Lapisan awal dari model terdiri dari beberapa lapisan *konvolusi* dengan kernel  $3 \times 3$ , yang berfungsi untuk mengekstraksi fitur spasial penting dari gambar. Setiap lapisan konvolusi diikuti oleh fungsi aktivasi *ReLU* untuk menambahkan *non-linearitas* dan mempercepat proses konvergensi. Setelah itu, *MaxPooling*  $2 \times 2$  digunakan untuk mereduksi dimensi spasial sekaligus mempertahankan fitur penting dari hasil ekstraksi.

Setelah melewati beberapa tahap ekstraksi fitur, hasil dari lapisan konvolusi diratakan menggunakan proses *flatten*, kemudian diteruskan ke satu atau dua *Dense Layer*. *Dense Layer* ini bertugas untuk menginterpretasi fitur dan pola visual yang telah diolah sebelumnya. Di akhir jaringan, terdapat lapisan *output* dengan fungsi aktivasi *Softmax* yang terdiri dari empat *neuron*, masing-masing mewakili kategori cuaca cerah, berawan, hujan, dan berkabut.

### 3.9.2 Proses *Training* dan Evaluasi *Model Custom CNN*

Sebelum digunakan untuk pelatihan, data citra cuaca melalui tahap *praprosesing* seperti *resize* ke ukuran  $128 \times 128$  piksel dan normalisasi nilai piksel ke skala  $[0,1]$  untuk kestabilan pembelajaran. Augmentasi data juga dilakukan dengan rotasi, flipping horizontal, dan penyesuaian pencahayaan untuk meningkatkan keragaman dataset dan mengurangi risiko *overfitting*. Model dilatih menggunakan *algoritma Adam Optimizer* yang adaptif dan cepat dalam pembaruan bobot, serta fungsi *categorical crossentropy* untuk tugas klasifikasi multi-kelas. *Dataset* dibagi menjadi 80% untuk pelatihan dan 20% untuk validasi guna memantau performa model. Model dilatih hingga mencapai konvergensi tanpa *overfitting*. Evaluasi kinerja model dilakukan menggunakan metrik akurasi, *precision*, *recall*, dan *F1-score*, serta *confusion matrix* untuk menganalisis distribusi prediksi dan kelas yang sering tertukar.

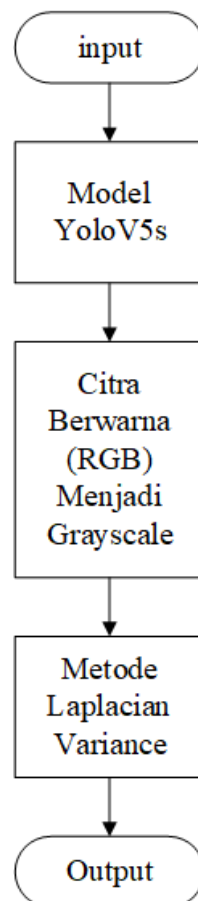
### 3.9.3 Implementasi *Model Custom CNN*

Model *Custom CNN* ini dirancang agar dapat diimplementasikan secara lokal pada sistem klasifikasi cuaca berbasis citra yang terhubung dengan kamera. Karena arsitekturnya ringan, model ini sangat cocok dijalankan di perangkat *edge computing* seperti *Raspberry Pi*, yang memiliki keterbatasan sumber daya. Model ini memungkinkan sistem untuk secara *real-time* mengenali kondisi cuaca dari gambar yang ditangkap kamera dan memberikan informasi prediksi berupa kondisi cerah, berawan, hujan, atau berkabut. Dengan pendekatan ini, sistem klasifikasi cuaca dapat digunakan untuk berbagai keperluan monitoring di area pertanian, transportasi, maupun sistem peringatan dini.

## 3.10. Model YOLOv5

*You Only Look Once version 5 (YOLOv5)* merupakan model deteksi objek berbasis *deep learning* yang dikembangkan dengan pendekatan satu tahap (*one-stage detector*). Model ini terkenal karena efisiensinya dalam mendeteksi objek secara *real-time*, bahkan pada perangkat dengan sumber daya terbatas. Dalam penelitian ini, YOLOv5 digunakan untuk mendeteksi objek di kawasan bandara seperti

pesawat, bangunan, kendaraan, dan objek lain yang terlihat dari gambar kamera. Hasil deteksi ini digunakan untuk membantu mengukur jarak pandang atau *visibility*, berdasarkan seberapa jauh dan seberapa jelas objek dapat dikenali dari citra.



**Gambar 3.7** Alur Kerja Model YoloV5.

### 3.10.1 Implementasi Pendekatan *Laplacian* pada Model YOLOv5

Untuk meningkatkan akurasi estimasi visibilitas, pendekatan *Laplacian Variance* diintegrasikan dengan model YOLOv5. Setelah YOLOv5 berhasil mendeteksi objek-objek relevan seperti awan atau kabut pada citra *real-time* yang diambil dari kamera, citra tersebut kemudian diolah lebih lanjut. Tahap pertama dari proses ini adalah konversi citra berwarna (RGB) menjadi *grayscale*. Konversi ini penting untuk efisiensi komputasi karena analisis ketajaman berbasis *Laplacian Variance* hanya memerlukan informasi intensitas terang-gelap, bukan warna, sehingga

meminimalkan beban pemrosesan pada perangkat *edge computing* seperti *Raspberry Pi*.

Setelah citra diubah menjadi *grayscale*, metode *Laplacian Variance* diterapkan untuk menghitung tingkat ketajaman citra. Nilai variansi *Laplacian* yang dihasilkan menjadi indikator numerik dari kejernihan gambar; semakin tinggi nilainya, semakin tajam gambar tersebut, yang berbanding lurus dengan tingkat visibilitas. Nilai ini kemudian digunakan untuk mengklasifikasikan visibilitas ke dalam kategori tertentu, melengkapi informasi deteksi objek dari YOLOv5. Dengan demikian, sistem tidak hanya mengetahui keberadaan objek-objek atmosfer, tetapi juga dapat memberikan estimasi kondisi jarak pandang berdasarkan kualitas visual citra yang terdeteksi.

### **3.10.2 Implementasi Model YOLOv5**

Setelah pelatihan, model *YOLOv5* diintegrasikan ke dalam sistem pemantauan visual berbasis kamera. Sistem ini digunakan untuk menghitung jarak pandang berdasarkan jumlah dan kejernihan objek yang terdeteksi. Semakin sedikit objek yang berhasil dideteksi dari gambar kamera, terutama objek yang jauh dari posisi kamera, maka semakin rendah nilai *visibility* yang dihitung. Sebaliknya, jika banyak objek terdeteksi dengan jelas, maka *visibility* dikategorikan tinggi.

Model ini memungkinkan sistem untuk memberikan estimasi *visibility* secara otomatis dan *real-time*, yang sangat penting dalam operasional bandara, khususnya untuk mendukung keputusan terkait aktivitas penerbangan saat kondisi cuaca berkabut atau hujan lebat.

### **3.11 Perancangan Software Website**

Perancangan perangkat lunak dalam sistem ini bertujuan untuk mengintegrasikan seluruh komponen perangkat keras dengan *model machine learning*, sehingga sistem dapat berjalan secara otomatis dalam proses akuisisi data, klasifikasi cuaca, *estimasi visibility*, serta pemantauan *real-time* melalui antarmuka *website*. Desain perangkat lunak dibuat secara modular agar setiap komponen dapat dikembangkan secara fleksibel sesuai kebutuhan, tanpa harus merombak keseluruhan sistem.

### 3.11.1 Sistem *Software Website*

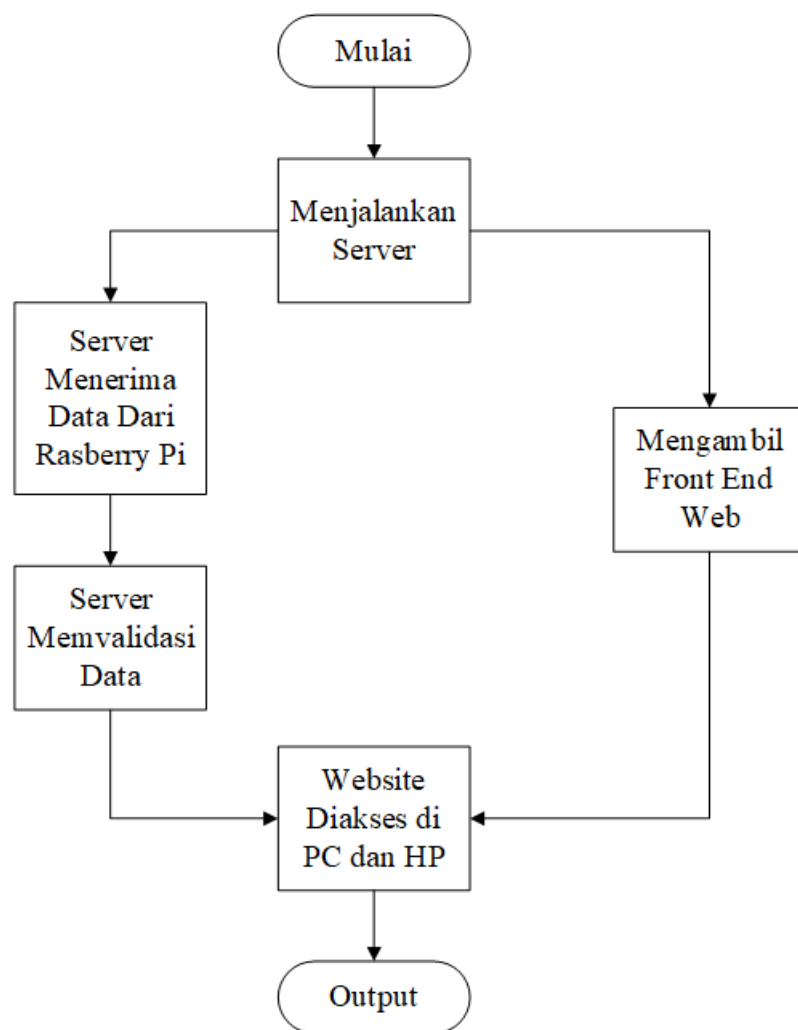
Perangkat lunak dikembangkan dengan struktur modular yang terdiri atas beberapa komponen inti yang saling terintegrasi. Modul akuisisi data bertugas mengatur pengambilan citra dari kamera yang terhubung ke *Raspberry Pi 5*, di mana citra awan yang diperoleh selanjutnya diproses melalui tahap *preprocessing* seperti pengubahan ukuran (*resize*) dan normalisasi untuk menyesuaikan dengan format *input model deep learning*. Setelah tahapan ini, citra dikirimkan ke model *MobileNetV2* yang bertugas mengidentifikasi jenis awan berdasarkan karakteristik visual yang terdeteksi. Secara paralel, model *Convolutional Neural Network (CNN)* digunakan untuk mengklasifikasikan kondisi cuaca menjadi kategori cerah, berawan, hujan, atau berkabut. Selain itu, *YOLOv5* diimplementasikan untuk mendeteksi objek di area pengamatan, yang hasilnya digunakan dalam proses *estimasi visibility* dengan menghitung jarak pandang berdasarkan keberadaan dan posisi objek dalam citra. Seluruh data hasil inferensi kemudian dikompilasi dan dikirimkan ke *server*, di mana data tersebut disimpan dalam *database* dan ditampilkan secara *real-time* melalui antarmuka *website* yang dirancang agar pengguna dapat memantau kondisi cuaca secara langsung dan akurat.

### 3.11.2 Bahasa Pemrograman dan *Framework Software*

Perangkat lunak ini menggunakan bahasa *Python* sebagai dasar untuk menghubungkan sensor, mengatur akuisisi data, serta menjalankan *model deep learning*. *Backend* dikembangkan dengan menggunakan *framework Flask* atau *FastAPI* guna menangani komunikasi data antara perangkat keras dan *web server*. *TensorFlow* atau *PyTorch* digunakan untuk keperluan pelatihan dan inferensi model CNN maupun *YOLOv5*. Untuk tampilan antarmuka pengguna, digunakan teknologi berbasis web seperti *HTML*, *CSS*, dan *JavaScript*, yang dapat dipadukan dengan *framework front-end* seperti *React* untuk menyajikan informasi secara interaktif. Data hasil klasifikasi dan sensor disimpan menggunakan basis data *SQLite* atau *MySQL*, tergantung pada kebutuhan skalabilitas sistem.

### 3.11.3 Alur Kerja Sistem *Software*

Alur kerja sistem dimulai dari *Raspberry Pi 5* yang mengambil citra kamera. Citra diproses secara lokal menggunakan CNN untuk klasifikasi awan dan YOLOv5 untuk estimasi *visibility*, sementara data sensor juga diolah. Hasilnya dikirim ke API server untuk disimpan dan disajikan melalui API. *Frontend web* mengambil data ini dan menampilkannya dalam dashboard berupa grafik, tabel, serta hasil klasifikasi cuaca dan *visibility* secara real-time. Visualisasi alur kerja ditampilkan pada **Gambar 3.5**.



**Gambar 3.5** Alur Kerja *software Website*.

### 3.11.3.1 Membangun *Server*

*Server* memainkan peran penting dalam pemrosesan data sisi server dan menjadi penghubung antara perangkat lokal seperti *Raspberry Pi* dengan antarmuka

pengguna di *frontend*. Fungsi utamanya adalah menerima data dari *Raspberry Pi*, yang mencakup hasil klasifikasi cuaca, *estimasi visibility*, dan data sensor lingkungan secara *real-time*, untuk kemudian disimpan dalam *database* agar terorganisir dan mudah diakses untuk analisis lebih lanjut. *server* juga menyediakan *API* yang memungkinkan *frontend* menampilkan data secara interaktif dan informatif. Pengembangannya menggunakan *framework* seperti *Flask* atau *FastAPI*, yang dikenal karena efisiensinya dalam menangani permintaan *HTTP* dan respons cepat, mendukung pengalaman pengguna yang lancar dan *real-time*. Selain itu, *Server* dilengkapi dengan fitur validasi data untuk memastikan hanya data akurat yang diproses, sistem *logging* kesalahan untuk pemantauan berkelanjutan dan pemecahan masalah, serta sistem notifikasi yang memberikan peringatan jika terjadi kegagalan data, gangguan komunikasi, atau kendala lain. Dengan demikian, *Server* tidak hanya berfungsi sebagai jembatan antara perangkat dan antarmuka pengguna, tetapi juga menjamin stabilitas, keandalan, dan keamanan sistem secara menyeluruh.

### 3.11.3.2 Membangun *Front end Web*

*Front end* adalah komponen yang berfungsi sebagai antarmuka antara sistem dan pengguna, memungkinkan interaksi langsung melalui aplikasi berbasis *web*. Antarmuka ini dirancang untuk menampilkan hasil klasifikasi cuaca dan estimasi *visibility* secara informatif dan mudah dipahami melalui visualisasi seperti grafik, tabel, dan indikator status kondisi atmosfer yang diperbarui secara *real-time*. Semua informasi yang disajikan berasal dari *API Server*, sehingga data yang ditampilkan selalu sinkron dan akurat. Dalam pengembangan antarmuka ini digunakan teknologi dan *framework* seperti *React*, *HTML*, *CSS*, dan *JavaScript*, yang dipilih untuk memenuhi kebutuhan fungsionalitas sistem serta memastikan antarmuka yang responsif, ringan, dan mudah diakses dari berbagai perangkat, baik *desktop* maupun *mobile*. Dengan desain yang adaptif terhadap berbagai ukuran layar, *front end* tidak hanya menjadi media penyaji data cuaca, tetapi juga berfungsi sebagai alat interaktif yang memberikan pengalaman pengguna yang efisien dan efektif dalam memantau kondisi atmosfer secara *real-time*. Tampilan *website* Ditampilkan pada **gambar 3.6**

Weather Prediction

Cloud Coverage

Location Data

Historical Data

| Date       | Time  | Location        | Prediction | Cloud Coverage (%) |
|------------|-------|-----------------|------------|--------------------|
| 2024-12-01 | 14:30 | -5.123, 105.456 | Cloudy     | 75                 |

**Gambar 3.6** Rancangan Tampilan *Website*.

### 3.11.4 Keamanan dan Keandalan

Untuk menjaga keamanan dan keandalan sistem, diterapkan validasi *input*, pencatatan *log error*, serta *backup* data berkala guna mencegah kesalahan dan kehilangan informasi. Sistem juga dilengkapi notifikasi otomatis jika terjadi gangguan, sehingga pengguna dapat segera melakukan perbaikan demi menjaga kestabilan operasional dan kontinuitas pemantauan data.

### 3.12 Pengujian Sistem

Pengujian sistem dilakukan untuk memastikan bahwa seluruh komponen perangkat keras dan perangkat lunak dalam sistem klasifikasi cuaca dan *estimasi visibility* berbasis citra awan bekerja sesuai dengan rancangan. Tahapan pengujian mencakup pengujian perangkat keras, akuisisi data, pemrosesan data, antarmuka *web*, serta evaluasi keseluruhan kinerja sistem.

#### 3.12.1 Pengujian Perangkat Keras

Pengujian perangkat keras dilakukan untuk memastikan seluruh komponen sistem berfungsi optimal di lapangan. Modul kamera diuji untuk menilai kualitas citra awan, sementara kestabilan koneksi dengan *Raspberry Pi 5* diperiksa agar akuisisi citra berjalan lancar. *Raspberry Pi 5* juga diuji kemampuannya dalam menjalankan model deep learning secara lokal dengan respons cepat. Selain itu, konsumsi daya dipantau untuk memastikan sistem dapat beroperasi stabil dalam waktu lama

menggunakan *power bank*. Hasil pengujian menunjukkan bahwa perangkat keras bekerja andal untuk akuisisi dan pemrosesan data cuaca *real-time*.

### 3.12.2 Pengujian Akuisisi Data

Pengujian akuisisi data dilakukan untuk memastikan proses pengambilan citra berjalan lancar dan menghasilkan gambar awan yang tajam, kontras, dan cukup terang. *Kamera Board Rev 1.3* diuji dalam berbagai kondisi pencahayaan untuk menilai konsistensinya, serta dievaluasi dari segi kecepatan dan kestabilan pengambilan citra. Hasil pengujian memastikan bahwa citra yang dihasilkan cukup akurat dan layak digunakan dalam proses klasifikasi berbasis model *deep learning*.

### 3.12.3 Pengujian Pemrosesan Data

Pengujian pemrosesan data dilakukan untuk memastikan bahwa setiap data yang diterima diproses sesuai dengan standar sistem. Citra awan diuji agar sesuai dengan ukuran *input model* ( $224 \times 224$  piksel) dan diproses menggunakan model CNN berbasis *MobileNetV2* untuk klasifikasi awan. Pengujian ini juga mencakup evaluasi akurasi model terhadap data uji. Selain itu, dilakukan verifikasi bahwa data dari sensor diproses tanpa kehilangan informasi dan tidak mengandung nilai *error* atau kosong.

### 3.12.4 Pengujian Antarmuka Website

Pengujian dilakukan untuk memastikan antarmuka *web* mampu menampilkan hasil klasifikasi citra secara *real-time* dengan koneksi stabil dan respons cepat. Evaluasi mencakup pengujian konektivitas *Raspberry Pi 5* dengan *web server*, tampilan *dashboard*, serta kompatibilitas pada berbagai perangkat dan kondisi jaringan. Hasil pengujian menunjukkan bahwa antarmuka *web* bekerja dengan baik, responsif, dan mudah diakses lintas *platform*.

### 3.12.5 Evaluasi dan Analisis Hasil

Setelah pengujian perangkat keras, perangkat lunak, dan antarmuka *web* selesai, dilakukan evaluasi menyeluruh terhadap kinerja sistem. Hasil klasifikasi citra awan

dianalisis menggunakan *confusion matrix* untuk mengukur akurasi *model CNN* dan *MobileNetV2* dalam mengenali jenis awan serta kesalahan klasifikasinya. Evaluasi juga mencakup kecepatan pemrosesan, kestabilan sistem, akurasi pengambilan citra, dan konsumsi daya untuk memastikan sistem dapat berjalan stabil dengan *power bank*. Seluruh hasil pengujian disajikan dalam bentuk tabel dan grafik guna mempermudah analisis dan menjadi dasar penyempurnaan sistem ke depan.

### 3.13 Data Hasil Analisis Cuaca

**Tabel 3.2** Data Hasil Analisi Cuaca

| No   | Waktu | Jenis<br>Awan | Cuaca | Visibility | Status | Gambar |
|------|-------|---------------|-------|------------|--------|--------|
| 1    |       |               |       |            |        |        |
| 2    |       |               |       |            |        |        |
| 3    |       |               |       |            |        |        |
| .... |       |               |       |            |        |        |
| 33   |       |               |       |            |        |        |

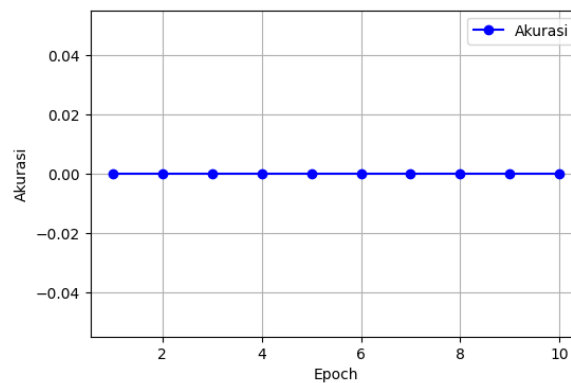
**Tabel 3.2** menyajikan data hasil analisis cuaca yang diperoleh dari sistem pemantauan berbasis *Raspberry Pi*, yang mencakup parameter utama seperti suhu, kelembapan, tekanan udara, jenis awan, dan kondisi cuaca pada berbagai tanggal pengambilan sampel. Kolom nomor dan tanggal menunjukkan urutan pencatatan data serta waktu pengambilan, yang penting untuk melihat tren perubahan cuaca. Perangkat digunakan untuk mengukur suhu dan kelembapan, sedangkan modul lain mencatat tekanan udara, yang semuanya berkontribusi dalam menentukan pola atmosfer di lokasi pengamatan. Selain itu, sistem klasifikasi berbasis CNN dengan *MobileNetV2* mengidentifikasi jenis awan dari citra yang diambil oleh Kamera Board Rev 1.3, yang kemudian dikorelasikan dengan parameter cuaca lainnya untuk memperkirakan kondisi cuaca secara keseluruhan. Dengan analisis ini, dapat diketahui hubungan antara karakteristik awan dan perubahan suhu, kelembapan, serta tekanan udara, sehingga membantu dalam memahami pola cuaca berdasarkan data yang dikumpulkan secara *real-time*.

### 3.13 Grafik dan Visualisasi Data

Visualisasi data digunakan untuk memahami pola dan *tren* dalam hasil pengujian sistem. Pada bagian ini, dua jenis grafik utama dibuat, yaitu grafik evaluasi model dan grafik parameter cuaca terhadap waktu.

#### 3.13.1 Grafik Evaluasi Model

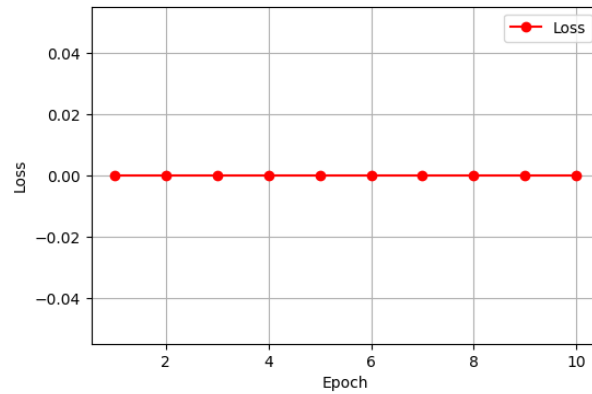
Evaluasi model dilakukan dengan mengamati perubahan akurasi selama proses pelatihan. Akurasi menunjukkan sejauh mana model dapat mengklasifikasikan jenis awan dengan benar berdasarkan data latih dan validasi. Grafik akurasi digunakan untuk melihat peningkatan performa model seiring dengan bertambahnya *epoch*. Jika akurasi pada data latih meningkat secara stabil dan akurasi validasi mengikuti pola serupa tanpa perbedaan yang signifikan, maka model dianggap telah belajar dengan baik. Namun, jika terdapat kesenjangan yang besar antara akurasi latih dan validasi, maka kemungkinan terjadi *overfitting*, yang perlu diatasi dengan teknik regulasi seperti *dropout* atau peningkatan jumlah data.



**Gambar 3.5** Grafik Akurasi Model.

Selain akurasi, *loss* atau nilai kesalahan model juga dianalisis untuk memahami sejauh mana prediksi model menyimpang dari nilai sebenarnya. *Loss* yang tinggi menunjukkan bahwa model masih melakukan kesalahan yang besar dalam prediksi, sedangkan *loss* yang rendah menandakan bahwa model semakin memahami pola dalam data. Grafik *loss* membantu dalam mengidentifikasi apakah model mengalami *underfitting* atau *overfitting*. Jika *loss* latih terus menurun tetapi *loss*

validasi tetap tinggi atau meningkat, maka model mengalami *overfitting*. Sebaliknya, jika *loss* latih dan validasi tetap tinggi, model mungkin masih *underfitting* dan memerlukan arsitektur yang lebih kompleks atau data tambahan untuk pelatihan.



**Gambar 3.6** Garfik *Loss* Model.

## BAB V SIMPULAN DAN SARAN

### 5.1 Simpulan

Berdasarkan hasil penelitian dan pembahasan dapat di simpulkan sebagai berikut

1. Sistem pemantauan cuaca *real-time* berhasil dirancang dan dibangun menggunakan *Raspberry Pi 5*. Sistem ini mampu melakukan klasifikasi awan, klasifikasi cuaca, serta *estimasi visibility* secara otomatis berdasarkan citra yang ditangkap oleh kamera.
2. Melalui integrasi deteksi objek YOLOv5 dengan pendekatan *Laplacian Variance*, estimasi *visibility* telah berhasil diimplementasikan. Metode ini mampu menganalisis kontras warna, ketajaman detail, dan tingkat keburaman pada citra. Saat dibandingkan dengan data BMKG, hasil estimasi menunjukkan akurasi sebesar 95% dan dapat dijalankan secara *real-time*.
3. Model *Custom Convolutional Neural Network* (CNN) yang dikembangkan berhasil mengklasifikasikan kondisi cuaca. Saat dibandingkan dengan data BMKG, model ini menunjukkan akurasi sebesar 70%, menegaskan efektivitasnya dalam sistem pemantauan cuaca berbasis citra *real-time*.
4. Dalam mengklasifikasikan jenis awan secara *real-time*, Model *MobileNetV2* menunjukkan akurasi sebesar 10% saat dibandingkan dengan data BMKG. Akurasi yang rendah ini diduga kuat disebabkan oleh ketidaksesuaian karakteristik data latih yang digunakan dengan kondisi data awan di lapangan. Meskipun demikian, model ini terbukti mampu beroperasi secara efisien pada perangkat dengan keterbatasan daya komputasi seperti *Raspberry Pi*, menjadikannya sesuai untuk aplikasi sistem *Internet of Things* (IoT).

informatif dan mudah diakses oleh pengguna, baik melalui perangkat lokal maupun jaringan *internet*.

5. Antarmuka *website* untuk menampilkan hasil klasifikasi awan, kondisi cuaca, dan estimasi *Visibility* secara *real-time* telah berhasil dikembangkan. Antarmuka ini dirancang agar informatif dan mudah diakses oleh pengguna, baik melalui jaringan lokal maupun *internet*, meningkatkan kemudahan pemantauan data cuaca.

## 5.2 Saran

Pengembangan sistem pemantauan cuaca ini masih memiliki beberapa keterbatasan yang perlu diperbaiki pada tahap selanjutnya. Salah satunya adalah antarmuka *website* yang belum terintegrasi dengan sistem basis data, sehingga pencatatan dan pelacakan historis hasil klasifikasi belum dapat dilakukan secara otomatis. Selain itu, *website* juga belum *dihosting* secara *online*, yang membatasi aksesibilitas pengguna hanya pada jaringan lokal. Oleh karena itu, disarankan agar pengembangan selanjutnya mencakup integrasi dengan basis data untuk penyimpanan data klasifikasi dan estimasi *visibility*, serta *hosting website* agar dapat diakses secara luas dan mendukung implementasi *real-time*. Selain itu, disarankan pula untuk menambahkan dua modul kamera, di mana satu kamera diarahkan secara vertikal untuk klasifikasi awan dan cuaca, sedangkan kamera lainnya diarahkan secara horizontal untuk estimasi *visibility*, guna meningkatkan akurasi dan cakupan pemantauan sistem. Selanjutnya, sangat penting untuk melakukan pencarian dan pengumpulan data latih awan yang lebih representatif dan sesuai dengan klasifikasi yang digunakan oleh BMKG, agar *model deep learning* dapat beradaptasi lebih baik dengan kondisi nyata di lapangan dan meningkatkan akurasi klasifikasi serta validitas hasil pemantauan.

## DAFTAR PUSTAKA

- Nugroho, P., Fenriana, I., and Arijanto, R. (2020). Implementasi Deep Learning Menggunakan Convolutional Neural Network ( Cnn ) Pada Ekspresi Manusia. *Jurnal Algor*, 2(1), 14–30.
- Akay, M., Du, Y., Sershen, C. L., Wu, M., Chen, T. Y., Assassi, S., Mohan, C., and Akay, Y. M. (2021). Deep Learning Classification Of Systemic Sclerosis Skin Using The Mobilenetv2 Model. *Journal Of Engineering In Medicine And Biology*, 2(6), 104–110.
- Arfan, A., and Elta, L. (2020). Perbandingan Algoritma Long Short-Term Memory Dengan Svr Pada Prediksi Harga Saham Di Indonesia. *Jurnal Pengkajian Dan Penerapan Teknik Informatika*, 13(1), 33–43.
- Aryani, D., Wahyudin, M., and Fazri, M. (2015). Prototype Robot Cerdas Pemotong Rumput Berbasis Raspberry Pi B+ Menggunakan Web Browser. *Journal Of Scientific Modeling*, 1, 2461–1417.
- Cahyo Santoso, B., Natasya, Y., Willian, S., and Alfando, F. (2020). Tinjauan Pustaka Sistematis Terhadap Basis Data Mongodb. *Jurnal Sistem Informasi Dan Teknik Komputer*, 8(1), 23–43.
- Dewa Agung Aditya, I. S., Saidah, S., and Usman, K. (2023). *Optimalisasi Arsitektur Mobilenet Untuk Pengklasifikasian Cuaca Mobilenet Architecture Optimization For Weather Classification*, 10(2), 61–72.
- Dewi, C., Kartikasari, D. P., and Mursityo, Y. T. (2014). Prediksi Cuaca Pada Data Time Series Menggunakan Adaptive Neuro Fuzzy Inference System (Anfis). In *Jurnal Teknologi Informasi Dan Ilmu Komputer (Jtiik)*, 2(3), 24–30.
- Wahab, M. N., Nazir, A., Ren, A. T. Z., Noor, M. H. M., Akbar, M. F., and Mohamed, A. S. A. (2021). Efficientnet-Lite And Hybrid Cnn-Knn Implementation For Facial Expression Recognition On Raspberry Pi. *Ieee Access*, 9(5), 134–156.

- Gazali, W., Soeparno, H., Ohliati, and Jenny. (2012). *Penerapan Metode Konvolusi Dalam Pengolahan Citra Digital*. In *Jitter-Jurnal Ilmiah Teknologi Dan Komputer*, 3(4), 14–23.
- Gigih, M., Suharsono, A., and Bhawiyuga, A., (2017). Implementasi Metode Store And Forward Pada Hypertext Transfer Protocol (Http). In *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-Ptiik*, 4(1), 52–63.
- Dewi, C., and Muslikh, M. (2013). Perbandingan Akurasi Backpropagation Neural Network Dan Anfis Untuk Memprediksi Cuaca. *Journal Of Scientific Modeling and Computation*, 1(3), 7–13.
- Handayani, A. S. (2010). *Analisis Daerah Endemik Bencana Akibat Cuaca Ekstrem Di Sumatera Utara*, 4(6), 27–31.
- Ihsan N C. (2021). Klasifikasidata Radar Menggunakan Algoritma Convolutional Neural Network(Cnn). *Journal Of Computer And Information Tecnology*, 5(2), 23–33.
- Kusumanto, R. D., and Tompunu, A. N. (2011). Pengolahan Citra Digital Untuk Mendeteksi Obyek Menggunakan Pengolahan Warna Model Normalisasi Rgb. In *Seminar Nasional Teknologi Informasi and Komunikasi Terapan*, 2(5), 67–77.
- Lattifia, T., Wira Buana, P., Kadek, N., and Rusjyanthi, D. (2022). Model Prediksi Cuaca Menggunakan Metode Lstm. In *Jitter-Jurnal Ilmiah Teknologi Dan Komputer*, 5(1), 34–41.
- Luthfiarta, A., Febriyanto, A., Lestiawan, H., and Wicaksono, W. (2020). Analisa Prakiraan Cuaca Dengan Parameter Suhu, Kelembaban, Tekanan Udara, Dan Kecepatan Angin Menggunakan Regresi Linear Berganda. *Joins (Journal Of Information System)*, 5(1), 10–17.
- Mawarni, M., Fitriyah, H., and Maulana, R. (2021). *Sistem Klasifikasi Langit Cerah Dan Berawan Menggunakan Gray Level Co-Occurrence Matrix Dan K-Nearest Neighbor Berbasis Raspberry Pi*, UNP press, Paalembang
- Mohajerani, S. (2019). Cloud-Net: Algoritma Deteksi Awan End-To-End Untuk Citra Landsat 8, *Symposium Geosains Internasional Dan Penginderaan Jauh Ieee*. 4(6), 3127–3136.
- Islamiah, M., Triyunita, N., and Suwondo, N., (2022). Pengembangan Perangkat Pengukuran Variabel Cuaca Menggunakan Sensor Besaran

Fisika Berbasis Internet Of Things. *Jurnal Pendidikan Mipa*, 12(3), 739–746.

Islah, A., Patombongi, A., and Kendari, S., (2023). Monitoring Data Citra Satelit Himawari 8 Dengan Metode Optical Flow. *Jurnal Sistem Informasi Dan Teknik Komputer*, 8(1). 14–23.

Nafisa, A., Purba, N., A., E., Harahap, F., and Putri, N., (2023). *Implementasi Algoritma Convolutional Neural Network Arsitektur Model Mobilenetv2 Dalam Klasifikasi Penyakit Tumor Otak Glioma, Pituitary Dan Meningioma*, PT. Elex Media Komputindo, Jakarta.

Nardi, and Nazori, A. (2024). *Otomasi Klasifikasi Awan Citra Satelit Mtsat Dengan Pendekatan Fuzzy Logic*, Ahli Media, Malang

Naufal, M. F. (2021). Analisis Perbandingan Algoritma Svm, Knn, Dan Cnn Untuk Klasifikasi Citra Cuaca. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, 4(1), 24–29.

Novandya, A., and Oktria, I. (2017). Penerapan Algoritma Klasifikasi Data Mining C4.5 Pada Dataset Cuaca Wilayah Bekasi. In *Jurnal Media Infotama*, 7(2), 13–23.

Pagan. (2018). Komparasi Kinerja Panel Surya Jenis Monokristal Dan Polykristal Studi Kasus Cuaca Banda Aceh. In *Jurnal Online Teknik Elektro*, 2(1), 29–35.

Peramala, P. S., Puspita, E. S., and Yulianti, L. (2016). Perancangan Sistem Peramalan Cuaca Berbasis Logika Fuzzy. In *Jurnal Media Infotama*, 12(1), 34–44.

Ramadhan, R., Sulistiani, H., Rahmanto, Y., Sesanti, A., and Pasaribu, B. N., (2022). Implementasi Esp32 Untuk Pengukuran Daya Tahan Otot Tes Push Up. *Jurnal Teknik Dan Sistem Komputer (Jtikom)*, 3(2), 22–33.

Alfarizi, R., S., M., Al-Farish, M., Taufiqurrahman, M., Ardiansah, G., and Elgar, M. (2023). Penggunaan Python Sebagai Bahasa Pemrograman Untuk Machine Learning Dan Deep Learning. In *Karimah Tauhid*, 2(1), 25–33.

Ririh, R., K., Laili, N., Wicaksono, A., and Tsurayya, S. (2020). Studi Komparasi Dan Analisis Swot Pada Implementasi Kecerdasan Buatan (Artificial Intelligence) Di Indonesia. In *Jurnal Teknik Industri*, 2(4), 35–33.

- Saluch, M., Tokarski, D., Grudniewski, T., Chodyka, M., Nitychoruk, J., Woliński, P., Jaworska, B., and Adamczewski, G. (2018). Raspberry Pi 3b + Microcomputer As A Central Control Unit In Intelligent Building Automation Management Systems. *Matec Web Of Conferences*, 1(3), 21–28.
- Sari, I. P., Novita, A., Khowarizmi, A.K., Ramadhani, F., and Satria, A. (2024). Pemanfaatan Internet Of Things (Iot) Pada Bidang Pertanian Menggunakan Arduino Uno R3. *Blend Sains Jurnal Teknik*, 2(4), 337–343.
- Siregar, A. M. (2020). Klasifikasi Untuk Prediksi Cuaca Menggunakan Ensemble Learning. *Jurnal Pengkajian Dan Penerapan Teknik Informatika*, 13(2), 138–147.
- Suaydhl. (2010). Variasi Musiman Berbagai Jenis Awan Di Indonesia. In *Journal Of Scientific Modeling* 3(2), 90–115.
- Sufy, A., Magdalena, R., and Nugraha, R., (2017). Purwarupa Sistem Klasifikasi Jenis Awan Dari Citra Panoramik Pantai Menggunakan Logika Fuzzy Prototype Of Cloud Feature Classification System On Sea's Panoramic Image Using Fuzzy Logic. In *Jurnal Teknologi Informasi Dan Ilmu Komputer*, 4(1), 24–29.
- Supriyadi, E. (2019). Prediksi Parameter Cuaca Menggunakan Deep Learning Long-Short Term Memory (Lstm) Weather Parameters Prediction Using Deep Learning Long-Short Term Memory (Lstm), *Jurnal Online Teknik Elektro*, 2(1), 29–35.
- Surbakti, N. M., Angelyca, A., Talia, A., Perangin, C., B., Nainggolan, D., O., Friskauly, N., D., and Tumorang. R., B., (2024). Penggunaan Bahasa Pemrograman Python Dalam Pembelajaran Kalkulus Fungsi Dua Variabel. *Algoritma : Jurnal Matematika, Ilmu Pengetahuan Alam, Kebumihan Dan Angkasa*, 2(3), 98–107.
- Suwitono, Y. A., and Kaunang, F. J. (2022). Implementasi Algoritma Convolutional Neural Network (Cnn) Untuk Klasifikasi Daun Dengan Metode Data Mining Semma Menggunakan Keras. *Jurnal Komtika (Komputasi Dan Informatika)*, 6(2), 109–121.
- Tilasefana, R. A., and Putra, R. E. (2023). Penerapan Metode Deep Learning Menggunakan Algoritma Cnn Dengan Arsitektur Vgg Net Untuk Pengenalan Cuaca. *Journal Of Informatics And Computer Science*, 5(1). 92–104.

- Ulya, F., and Kamal, M. (2017). Rancang Bangun Sistem Monitoring Cuaca Dengan Tampilan Thingspeak. *Jurnal Tektro*, 1(6), 8–34.
- Priyahita, W., F., Sugianti, N., Aliah, H., Fisika, J., Sains Dan Teknologi, F., Universitas Islam Negeri Sunan Gunung Djati, I., Meteorologi, B., & Dan Geofisika Bandung, K. (2016). Analisis Taman Alat Cuaca Kota Bandung Dan Sumedang Menggunakan Satelit Terra Berbasis Python. *Journal Of Physics*, 3(2), 33–46.
- Wulandari, A., Mayanti, A., and Hidayatullah, A. (2023). Analisis Multi-Access Edge Computing Menggunakan Jaringan Open Ran Pada Politeknik Negeri Jakarta. *Seminar Nasional Terapan Riset Inovatif (Sentrinov) Ke-9*, 3(4), 34-55.
- Putra, B., Pamungkas, G., Nugroho, B., Anggraeny, F. (2021). Deteksi Dan Menghitung Manusia Menggunakan Yolo-Cnn. In *Jurnal Informatika dan Sistem Informasi (JIFoSI)*, 2(5), 45–52.
- Dutta, M., Sujan, M. R., Mojumdar, M. U., Chakraborty, N. R., Marouf, A. Rokne, J. G., and Alhajj, R. (2024). Rice Leaf Disease Classification—A Comparative Approach Using Convolutional Neural Network (CNN), Cascading Autoencoder with Attention Residual U-Net (CAAR-U-Net), and MobileNet-V2 Architectures. *Technologies*, 12(11), 134–145.
- Sholawati, M., Auliasari, K., and Ariwibisono, F. X. (2022). Pengembangan Aplikasi Pengenalan Bahasa Isyarat Abjad Sibi Menggunakan Metode Convolutional Neural Network (Cnn). In *Jurnal Mahasiswa Teknik Informatika*, 6(1), 45-78.
- Terven, J., Esparza, D. M., and González, J. A. (2023). A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. In *Machine Learning and Knowledge Extraction*. Multidisciplinary Digital Publishing Institute. Jakarta.
- Paris, S., Hasinoff, S. W., and Kautz, J. (2015). Local Laplacian filters: Edge-aware image processing with a Laplacian pyramid. *Communications of the ACM*, 58(3), 81–91.
- Christiani, L. (2018). Peluang dan Tantangan Penerapan Cloud Computing (Komputasi Awan) Sebagai Solusi Automasi Kerjasama Antar Perpustakaan. *ANUVA*, 2(1), 43–53.
- Kristanto, Y., & Muhammad, F. R. (2017). Pendugaan Karakteristik Awan Berdasarkan Data Spektral Citra Satelit Resolusi Spasial Menengah

Landsat 8 Oli/Tirs (Studi Kasus: Provinsi Dki Jakarta). *Article in Jurnal Meteorologi Klimatologi Dan Geofisika*, 4(1), 36–48.