

***FINE-TUNING MODEL FACENET BERBASIS PYTORCH UNTUK FACE
RECOGNITION PADA SISTEM ABSENSI DIGITAL DI LABORATORIUM
TEKNIK KOMPUTER UNIVERSITAS LAMPUNG***

(Skripsi)

Oleh

BUDI CAHYONO

2115061123



**FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG**

2025

***FINE-TUNING MODEL FACENET BERBASIS PYTORCH UNTUK FACE
RECOGNITION PADA SISTEM ABSENSI DIGITAL DI LABORATORIUM
TEKNIK KOMPUTER UNIVERSITAS LAMPUNG***

**Oleh
BUDI CAHYONO**

Skripsi

**Sebagai Salah Satu Syarat untuk Mencapai Gelar
SARJANA TEKNIK**

Pada

**Jurusan Teknik Elektro
Fakultas Teknik Universitas Lampung**



**FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG**

2025

ABSTRAK

FINE-TUNING MODEL FACENET BERBASIS PYTORCH UNTUK FACE RECOGNITION PADA SISTEM ABSENSI DIGITAL DI LABORATORIUM TEKNIK KOMPUTER UNIVERSITAS LAMPUNG

Oleh

Budi Cahyono

Pengelolaan administrasi laboratorium komputer di Universitas Lampung masih dilakukan secara manual, sehingga berisiko terhadap kehilangan data dan pencatatan yang kurang efisien. Kondisi ini menimbulkan kebutuhan akan sistem digital yang mampu meningkatkan akurasi dan transparansi proses absensi. Penelitian ini bertujuan untuk mengembangkan sistem absensi berbasis face recognition dengan pendekatan fine-tuning pada model FaceNet menggunakan PyTorch. Dataset penelitian terdiri dari 2.700 citra wajah dari 30 kelas mahasiswa yang telah melalui tahap preprocessing berupa deteksi, cropping, dan resize. Penggunaan triplet loss pada model FaceNet dengan arsitektur InceptionResNetV1 menghasilkan embedding wajah yang lebih representatif. Evaluasi dilakukan menggunakan cosine similarity untuk penentuan threshold, serta metrik performa berupa akurasi, confusion matrix, ROC-AUC, dan visualisasi t-SNE. Hasil pengujian menunjukkan threshold optimal sebesar 0,70. Model fine-tuned mencapai akurasi 99,66%, lebih tinggi dibandingkan model pre-trained yang hanya 98,99%. Pada uji coba langsung menggunakan kamera terhadap lima mahasiswa, model fine-tuned berhasil mengenali wajah dengan akurasi 100%, sedangkan model pre-trained hanya mencapai 86,67%. Visualisasi t-SNE juga memperlihatkan bahwa kluster wajah pada model fine-tuned lebih jelas dan terpisah. Hasil penelitian ini menunjukkan bahwa penerapan triplet loss pada model FaceNet dapat meningkatkan keakuratan dan keandalan sistem face recognition, sehingga layak digunakan untuk mendukung digitalisasi absensi di Laboratorium Teknik Komputer Universitas Lampung.

Kata Kunci: Absensi Digital, Face Recognition, FaceNet, PyTorch, Fine-Tuning

ABSTRACT

FINE-TUNING MODEL FACENET BERBASIS PYTORCH UNTUK FACE RECOGNITION PADA SISTEM ABSENSI DIGITAL DI LABORATORIUM TEKNIK KOMPUTER UNIVERSITAS LAMPUNG

By

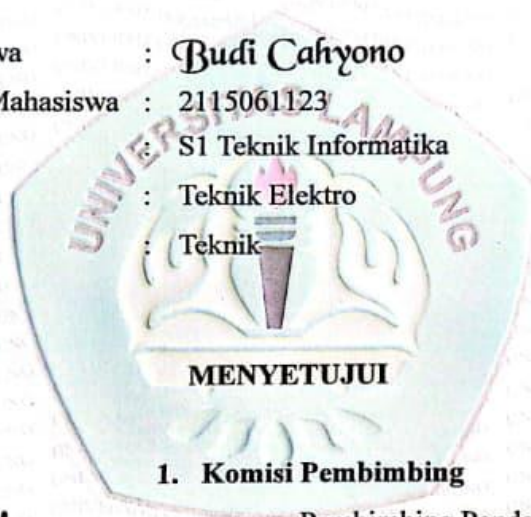
Budi Cahyono

The administrative management of the computer laboratory at the University of Lampung is still conducted manually, making it prone to data loss and inefficient record-keeping. This situation highlights the need for a digital system that can enhance the accuracy and transparency of the attendance process. This study aims to develop a face recognition-based attendance system using a fine-tuning approach on the FaceNet model implemented with PyTorch. The dataset consists of 2,700 facial images from 30 student classes that have undergone preprocessing stages, including detection, cropping, and resizing. The application of triplet loss on the FaceNet model with the InceptionResNetV1 architecture produces more representative facial embeddings. The evaluation was performed using cosine similarity to determine the optimal threshold, along with performance metrics such as accuracy, confusion matrix, ROC-AUC, and t-SNE visualization. The experimental results show that the optimal threshold is 0.70. The fine-tuned model achieved an accuracy of 99.66%, which is higher than the pre-trained model's 98.99%. In real-time testing using a camera with five students, the fine-tuned model successfully recognized faces with 100% accuracy, while the pre-trained model achieved only 86.67%. The t-SNE visualization also demonstrates that the facial clusters in the fine-tuned model are clearer and more distinct. These results indicate that applying triplet loss to the FaceNet model improves the accuracy and reliability of the face recognition system, making it suitable for supporting the digitalization of attendance in the Computer Laboratory at the University of Lampung.

Keywords: Digital Attendance, Face Recognition, FaceNet, PyTorch, Fine-Tuning

Judul Skripsi : *FINE-TUNING MODEL FACENET BERBASIS
PYTORCH UNTUK FACE RECOGNITION
PADA SISTEM ABSENSI DIGITAL DI
LABORATORIUM TEKNIK KOMPUTER
UNIVERSITAS LAMPUNG*

Nama Mahasiswa : **Budi Cahyono**
Nomor Pokok Mahasiswa : 2115061123
Program Studi : S1 Teknik Informatika
Jurusan : Teknik Elektro
Fakultas : Teknik



1. Komisi Pembimbing

Pembimbing Utama

Yessi Mulyani, S.T., M.T.
NIP. 197312262000122001

Pembimbing Pendamping

Puput Budi Wintoro, S. Kom, M.T.I.
NIP. 198410312019031004

2. Mengetahui

Ketua Jurusan
Teknik Elektro

Herlinawati, S.T., M.T.
NIP. 197103141999032001

Ketua Program Studi
Teknik Informatika

Yessi Mulyani, S.T., M.T.
NIP. 197312262000122001

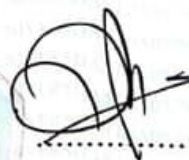
MENGESAHKAN

1. Tim Penguji

Ketua : Yessi Mulyani S.T., M.T.



Sekretaris : Puput Budi Wintoro, S. Kom, M.T.I.



Penguji : Ir. M. Komarudin S.T., M.T.



2. Dekan Fakultas Teknik



Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc. ✓

NIP. 197509282001121002

Tanggal Lulus Ujian Skripsi : 28 Juli 2025

SURAT PERNYATAAN

Saya yang bertanda tangan dibawah ini, menyatakan dengan sebenarnya bahwa skripsi saya yang berjudul “ *Fine-Tuning Model Facenet Berbasis Pytorch Untuk Face Recognition Pada Sistem Absensi Digital Di Laboratorium Teknik Komputer Universitas Lampung*” merupakan hasil karya saya sendiri. Semua hasil yang tertuang dalam skripsi ini telah mengikuti kaidah [enulisdan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti bahwa skripsi saya ini merupakan salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan ketentuan hukum atau akademik yang berlaku.

Bandar Lampung, 28 Juli 2025



Budi Cahyono

2115061123

RIWAYAT HIDUP



Penulis dilahirkan di Lampung Timur pada tanggal 1 Mei 2003, sebagai anak kelima dari pasangan Bapak Nur Hadi dan Ibu Ida Yati. Penulis memulai pendidikan di PAUD Permata Bunda pada tahun 2007, kemudian melanjutkan ke SD Negeri 1 Tritunggal dan lulus pada tahun 2015. Pendidikan menengah pertama diselesaikan di SMP Negeri 2 Tanjung Sari pada tahun 2018, dan pendidikan menengah atas di SMA Negeri 1 Tanjung Sari yang diselesaikan pada tahun 2021. Kemudian penulis melanjutkan pendidikan tinggi di Universitas Lampung, Program Studi S1 Teknik Informatika, Jurusan Teknik Elektro, Fakultas Teknik, melalui jalur beasiswa PMPAP (Program Perluasan Akses Pendidikan). Selama menjalani masa perkuliahan, penulis aktif berpartisipasi dalam berbagai kegiatan, antara lain:

1. Berperan sebagai Asisten Praktikum sekaligus Koordinator di Laboratorium Teknik Komputer Universitas Lampung pada tahun 2023 hingga 2025.
2. Menjadi Facilitator pada program Google Cloud Arcade Facilitator yang diselenggarakan oleh Google Indonesia pada tahun 2024 dan 2025.
3. Menjadi Facilitator FGA pada program Elevate with Microsoft, Digital Talent Scholarship (KomDigi) yang diadakan oleh Dicoding pada tahun 2025.
4. Berperan sebagai Front-End Developer dalam program Virtual Internship yang diselenggarakan oleh PT Sarana Digital Inovatif pada tahun 2024.
5. Berperan sebagai Graphic Designer dalam program Virtual Internship yang diselenggarakan oleh PT Moslem Tour and Travel, dan PT Renjana Sinergi Indonesia pada tahun 2024.
6. Mengikuti program Studi Independen Bersertifikat (MSIB) dari Kementerian Pendidikan dan Kebudayaan, Bangkit Academy 2023 by Google, Tokopedia, Gojek & Traveloka sebagai Machine Learning Cohort. Dan Infinite Learning (PT Kinema Systrans Multimedia) pada tahun 2024 sebagai peserta program IBM Advanced AI.

MOTTO

Mari tumbuh bersama, saling menguatkan dan mengingatkan, melengkapi ketidaksempurnaan, menjaga saat satu di antara kita dihadap rasa takut. Karena mungkin, kehidupan akan berjalan sangat panjang, atau bisa jadi Tuhan memberi batas waktunya tak lebih sampai hari ini. Mari mengukir sejarah pada detik-detik yang tersisa, memberi tahu semesta bahwa kita pernah ada di sini, menjalankan tugas dari Yang Maha Kuasa, untuk menjadi sebaik-baiknya manusia.

Mari hidup dengan lebih baik.

(Budi Cahyono)

PERSEMBAHAN

Alhamdulillahirabbil‘alamin

Segala puji bagi Allah SWT yang telah melimpahkan rahmat dan karunia-Nya sehingga skripsi ini telah diselesaikan dengan baik. Shalawat serta salam semoga senantiasa tercurah kepada Nabi Muhammad SAW, teladan sepanjang masa.

Bismillahirrahmanirrahim Kupersembahkan Skripsi ini kepada

Kedua orangtua tercinta Bapak Nur Hadi dan Ibu Ida Yati, yang tak pernah lelah melangitkan doa-doa terbaik sepanjang hidup mereka demi kesuksesan anaknya.

Terimakasih atas segalanya dan semoga Allah senantiasa memberikan perlindungan di dunia dan akhirat, aamiin.

Seluruh dosen dan segenap civitas akademik serta rekan rekan seperjuangan prodi Teknik Informatika Universitas Lampung, atas motivasi, diskusi, persaingan dan kebersamaan yang menjadikan perjalanan akademik ini penuh makna.

Dan kepada seluruh rakyat Indonesia, semoga karya kecil ini dapat menjadi kontribusi nyata bagi bangsa yang kita cintai.

SANWACANA

Alhamdulillahirabbil'alam, Penulis mengucapkan puji syukur atas kehadiran Allah SWT, Tuhan Yang Maha Esa, atas segala rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi berjudul "Fine-Tuning Model FaceNet Berbasis PyTorch untuk Face Recognition pada Sistem Absensi Digital di Laboratorium Teknik Komputer Universitas Lampung." Skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik pada Program Studi S1 Teknik Informatika, Fakultas Teknik, Universitas Lampung. Penulis menyadari bahwa penyelesaian skripsi ini tidak terlepas dari dukungan bimbingan, dan motivasi dari berbagai pihak. Oleh karena itu, dengan tulus hati, penulis mengucapkan terimakasih kepada:

1. Allah SWT, Tuhan Yang Maha Esa, atas segala kekuatan, kesehatan, dan kemudahan yang telah diberikan sehingga penulis dapat menyelesaikan pendidikan ini dengan baik.
2. Ayahanda Nur Hadi dan Ibunda Ida Yati tercinta, terima kasih atas doa, kasih sayang, pengorbanan, dan dukungan yang tiada henti. Semoga segala usaha dan doa Ayah dan Bunda dibalas oleh Allah SWT dengan keberkahan dan kebahagiaan dunia akhirat.
3. Kakak-kakak tersayang: Mas Toni Wibowo, Mbak Endang Suratmiati, Mbak Indah Murniasih, dan Mbak Sri Utami, terima kasih atas doa, dukungan, serta semangat yang selalu diberikan. Semoga keluarga kita semakin kuat dan senantiasa dalam lindungan Allah SWT.
4. Ibu Novitha Sari, S.Psi., terima kasih atas bimbingan, perhatian, dan motivasi yang telah menjadi titik awal penulis menempuh pendidikan tinggi. Semoga Ibu bersama keluarga: Mas Rian, Akbar senantiasa diberi kesehatan, kebahagiaan, dan keberkahan.
5. Ibu Yessi Mulyani, S.T., M.T., selaku Ketua Program Studi Teknik Informatika sekaligus Dosen Pembimbing I, terima kasih atas segala arahan, bimbingan, dan kesabaran yang telah diberikan. Semoga Allah membalas dengan kesehatan, umur panjang, dan kebahagiaan.
6. Bapak Puput Budi Wintoro, S.Kom., M.T.I., selaku Dosen Pembimbing II, atas bimbingan, motivasi, dan arahan yang sangat membantu penulis dalam penyusunan skripsi ini. Semoga Bapak selalu sehat dan diberi keberkahan.

7. Bapak Ir. M. Komarudin S.T., M.T. selaku Dosen Penguji, terimakasih atas masukan, kritik, dan saran yang membangun. Semoga bapak sehat, panjang umur, selalu tersenyum seperti biasanya dan bahagia selalu.
8. Bapak Rio Ariesta Pradipta, S.Kom., M.T.I., selaku Dosen Pembimbing Akademik, atas bimbingan akademik selama masa studi. Semoga bapak sehat, panjang umur dan bahagia selalu.
9. Ibu Ir.Titin Yulianti S.T., M.Eng, selaku Kepala Laboratorium Teknik Komputer, trimakasih atas dukungan dan fasilitas yang diberikan serta ilmu dan pengetahuan yang telah diberikan kepada penulis. Trimakasih karna sudah menjadi orang baik. Semoga ibu sehat dan bahagia selalu.
10. Valen Vebriana, di akhir perjalanan ini, aku sempat menulis beberapa pesan untukmu, tak terlalu rumit hanya satu sapa sederhana, lalu kulanjut dengan ungkapan rindu. Tapi kamu tahu, kalimat yang paling aku kagumi di tulisan ini? Dengan mencintaimu, aku merasakan tumbuh menjadi lebih kuat, lebih baik, dan aku akan terus melakukannya, dan bertemu denganmu, adalah salah satu hal yang paling aku sukai.
11. Levian Dandra dan Nur Annisa Septriza, rekan satu tim skripsi, semoga kita selalu sehat, sukses, dan semakin hebat.
12. Seluruh Asisten Laboratorium Teknik Komputer angkatan 2021, atas kerja keras dan kebersamaan yang luar biasa. Terutama kepada Teguh Karya Rizki, Rizki Pangestu dan Yos Sianipar, trimakasih sudah membantu penulis untuk tetap bertahan hidup, menjalani hidup dan menjadi diri sendiri sebagai seorang mahasiswa di lab komputer, sukses untuk semuanya.
13. Uyunin Khumairoh, Anjar Asri Wijayanti, Rina Afriyani, Yoger Danuarta, Rico Ramadhani, Okta Surya Arif, Abdul Rahman Wahid, dan Ajeng Ayu Hiemas Praditha, sahabat terbaik selama perjalanan perkuliahan. Semoga kalian selalu bahagia, semoga selalu bertemu orang orang baik. Semoga kebahagiaan dan kebaikan senantiasa menyertai langkah kalian.
14. Seluruh teman-teman PSTI 2021, terima kasih atas kebersamaan, dukungan, dan perjalanan yang telah dilalui bersama. Semoga kita semua memperoleh kesuksesan, pekerjaan yang layak, serta kehidupan yang lebih baik.

Akhir kata, penulis berharap karya ini dapat memberikan manfaat dan kontribusi positif bagi pembaca, khususnya dalam pengembangan sistem absensi digital berbasis pengenalan wajah.

Bandar Lampung, 28 Juli 2025
Penulis

Budi Cahyono

DAFTAR ISI

	Halaman
ABSTRAK	iii
ABSTRACT	iv
DAFTAR ISI	iv
DAFTAR GAMBAR	viii
DAFTAR TABEL.....	x
I. PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Tujuan Penelitian.....	3
1.4 Manfaat Penelitian	3
1.5 Batasan Masalah.....	3
1.6 Sistematika Penulisan Skripsi / Tugas Akhir	4
II. TINJAUAN PUSTAKA	6
2.1. <i>Python</i>	6
2.2 <i>Artificial Intelligence</i>	7
2.3 <i>Machine Learning</i>	8
2.4 <i>Deep Learning</i>	8
2.5 Pengolahan Citra Digital	9
2.5.1 Citra analog	12
2.5.2 Citra digital	12
2.5.3 Jenis Jenis Citra.....	13
2.6 Jaringan Syaraf Tiruan	14

2.6.1 Komponen jaringan syaraf tiruan.....	15
2.6.2 Jenis jenis jaringan syaraf tiruan	16
2.6.3 Proses belajar jaringan syaraf tiruan	17
2.7 <i>Convolutional Neural Network</i>	19
2.7.1 <i>Convolutional</i>	19
2.7.2 <i>Rectified Linear Unit (ReLu)</i>	21
2.7.3 <i>Pooling</i>	22
2.7.4 <i>Flatten</i>	22
2.7.5 <i>Fully Connected Layer</i>	22
2.7.6 <i>Softmax</i>	22
2.8 <i>Face Recognition</i>	22
2.8.1 <i>Holistic Learning</i> (1991–2000).....	24
2.8.2 <i>Local Handcrafted Features</i> (2000–2010).....	24
2.8.3 <i>Shallow Learning</i> (2010–2014).....	24
2.8.4 <i>Deep Learning</i> (2014–sekarang).....	25
2.9 <i>FaceNet</i>	30
2.9.1 <i>Arsitektur FaceNet</i>	30
2.9.2 <i>Embedding Vektor</i>	31
2.9.3 <i>Metode FaceNet</i>	32
2.9.4 <i>Evaluasi metrik dalam FaceNet</i>	34
2.10 <i>Fine-Tuning</i>	40
2.10.1 <i>Strategi fine-tuning</i>	41
2.10.2 <i>Hyperparameter dalam fine-tuning</i>	42
2.11 <i>Google Colab</i>	45
2.12 <i>Cursor</i>	46
2.13 <i>Github</i>	46
2.14 <i>Pseudocode</i>	47
2.15 <i>Penelitian Terdahulu</i>	48
III. METODOLOGI PENELITIAN	57
3.1 Waktu dan Tempat.....	57

3.2 Jadwal penelitian.....	57
3.3 Alat dan bahan penelitian	58
3.4 Tahapan Penelitian	59
3.4.1 Dataset.....	60
3.4.2 <i>Preprocessing</i>	62
3.4.3 <i>Fine-tuning FaceNet</i>	64
3.4.4 Evaluasi	68
IV. HASIL DAN PEMBAHASAN	69
4.1 Dataset.....	69
4.2 <i>Preprocessing</i>	69
4.2.1 <i>Preprocessing</i> manual	69
4.2.2 Deteksi wajah	70
4.2.3 <i>Cropping</i>	72
4.2.4 <i>Resize</i>	74
4.2.5 Pembagian dataset.....	75
4.3 <i>Fine-tuning FaceNet</i>	77
4.3.1 Proses inisialisasi model <i>pre-trained</i>	77
4.3.2 Modifikasi layer untuk <i>fine-tuning</i>	77
4.3.3 <i>Hyperparameter</i>	78
4.3.4 <i>Triplet loss implemetation</i>	80
4.3.5 Proses <i>training</i>	81
4.3.6 Hasil	83
4.4 Evaluasi Model <i>fine-tuning</i>	86
4.4.1 Evaluasi Metrik <i>Embedding</i>	86
4.4.2 Evaluasi Metrik Performa	92
4.5 Evaluasi Model <i>Pre-Train</i>	98
4.5.1 Evaluasi Metrik <i>Embedding</i>	98
4.5.2 Evaluasi Metrik Performa	101
4.6 Analisis perbandingan <i>model pre-trained vs fine-tuning</i>	105
4.6.1 Model <i>pre-trained</i>	106

4.6.2 Model fine-tuning.....	108
4.6.3 Perbandingan Model	109
4.7 Uji Coba Objek Lain	110
4.8 Standar kelayakan Face Recognition	113
V. KESIMPULAN.....	116
5.1 Kesimpulan	116
5.2 Saran.....	116
DAFTAR PUSTAKA	117

DAFTAR GAMBAR

Gambar 2. 1 Logo <i>Python</i>	7
Gambar 2. 2 Ilustrasi neuron biologis dan neuron buatan[15].....	14
Gambar 2. 3 Single Layer Neural Network	16
Gambar 2. 4 Multi Layer Neural Network.....	16
Gambar 2. 5 Arsitektur CNN[16].....	19
Gambar 2. 6 Kernal	20
Gambar 2. 7 Pergeseran stride	20
Gambar 2. 8 Padding lapisan piksel.....	21
Gambar 2. 9 Grafik <i>ReLU</i>	21
Gambar 2. 10 Tonggak sejarah <i>Face Recognition</i> [18]	23
Gambar 2. 11 Logo <i>OpenCV</i>	26
Gambar 2. 12 Logo <i>pytorch</i>	26
Gambar 2. 13 Arsitektur MTCNN[23]	27
Gambar 2. 14 <i>Tradisional learning</i> dan <i>Transfer Learning</i> [21].....	29
Gambar 2. 15 Metode FaceNet[25].....	32
Gambar 2. 16 Fungsi <i>Triplet Loss FaceNet</i>	34
Gambar 2. 17 Logo <i>Google Colab</i>	45
Gambar 2. 18 Logo <i>Cursor Code Editor</i>	46

Gambar 2. 19 Logo <i>Github</i>	46
Gambar 3. 1 Tahapan Fine-Tuning FaceNet.....	59
Gambar 4. 1 Format Dataset	69
Gambar 4. 2 <i>preview</i> deteksi wajah menggunakan MTCNN.....	71
Gambar 4. 3 <i>preview bounding box</i> wajah.....	73
Gambar 4. 4 <i>preview resize</i> ukuran piksel	74
Gambar 4. 5 Distribusi dataset	75
Gambar 4. 6 Contoh hasil <i>embedding</i> wajah	84
Gambar 4. 7 Grafik <i>training</i> dan <i>validation loss</i>	86
Gambar 4. 8 Cosine similarity model <i>fine-tuning</i>	89
Gambar 4. 9 t-SNE model <i>fine-tuning</i>	91
Gambar 4. 10 visualisasi ROC-AUC	95
Gambar 4. 11 Confussion matrix	96
Gambar 4. 12 <i>Real Case Camera</i>	97
Gambar 4. 13 <i>Cosine similarity</i> model <i>pre-trained</i>	99
Gambar 4. 14 t-SNE model <i>pre-trained</i>	100
Gambar 4. 15 <i>confussion matrix</i> menggunakan model <i>pre-trained</i>	104
Gambar 4. 16 Roc auc – curve model <i>pre-trained</i>	105
Gambar 4. 17 hasil pengujian model <i>pre-trained</i>	107
Gambar 4. 18 hasil pengujian model <i>fine-tuning</i>	108
Gambar 4. 19 hasil pengujian gambar kucing.....	111
Gambar 4. 20 hasil pengujian gambar lemari	112
Gambar 4. 21 hasil pengujian kue muffin.....	113

DAFTAR TABEL

Tabel 2. 1 Arsitektur FaceNet dan performanya	31
Tabel 2. 2 Metrik performa FaceNet.....	38
Tabel 3. 1 Jadwal Penelitian.....	57
Tabel 3. 2. Alat penelitian	58
Tabel 3. 3. Bahan penelitian.....	58
Tabel 3. 4 kriteria dataset.....	60
Tabel 4. 1 Nilai loss	85
Tabel 4. 2 Contoh perhitungan <i>threshold</i>	88
Tabel 4. 3 Evaluasi data Test-set dengan model fine-tuning	92
Tabel 4. 4 Evaluasi data uji dengan model <i>pre-train</i>	101
Tabel 4. 5 perbandingan akurasi data test-set	105
Tabel 4. 6 Standar ISO/IEC 29794-5	114

I. PENDAHULUAN

1.1 Latar Belakang

Kecerdasan buatan atau *Artificial Intelligence* (AI) adalah teknologi yang memiliki kemampuan untuk memecahkan masalah layaknya manusia. Dalam praktiknya, AI meniru kecerdasan manusia melalui berbagai cara, seperti pengenalan gambar, pemrosesan bahasa alami, dan pengambilan keputusan berbasis data. Kecerdasan buatan saat ini lebih mengarah ke sistem berbasis agen cerdas, yang dapat bekerja secara mandiri dan berinteraksi dalam lingkungan yang kompleks. Kecerdasan buatan modern mampu belajar, beradaptasi, berkomunikasi, dan mengelola ketidakpastian lebih baik daripada sistem sebelumnya[1]. Dalam bidang pengenalan wajah, *Deep Learning* yang merupakan salah satu sub-bidang dari Artificial Intelligence, yang berperan sangat penting. *Deep Learning* memanfaatkan jaringan saraf tiruan dalam atau *Deep Neural Network* untuk menganalisis dan mengenali pola dalam data wajah[2].

Laboratorium komputer merupakan salah satu fasilitas penting dalam menunjang proses belajar-mengajar di Universitas Lampung, khususnya bagi mahasiswa Teknik Informatika. Dalam praktik mengajarnya kegiatan yang dilakukan yaitu Praktikum, Penelitian (Program Kreativitas Mahasiswa, Pengabdian Masyarakat, Skripsi, dan lain lain). Laboratorium komputer terdiri dari kepala laboratorium, teknisi dan juga asistant laboratorium yang tertulis resmi dalam sebuah surat keputusan, serta memiliki tata tertib pelaksanaan praktikum dan *Standard Operating Procedure* (SOP) untuk penggunaan laboratorium. Namun, saat ini masih terdapat beberapa kendala dalam pengelolaan dan administrasi laboratorium. Salah satu permasalahan utama adalah sistem absensi mahasiswa dan informasi laboratorium yang masih bersifat manual, baik untuk kegiatan praktikum, piket asisten, maupun kegiatan lainnya. Ketergantungan pada media sosial seperti

Instagram atau platform komunikasi lainnya sering kali kurang optimal dalam menjangkau semua pihak terkait dalam menyampaikan informasi Laboratorium Teknik Komputer Universitas Lampung.

Absensi manual yang dilakukan melalui buku sering kali rentan terhadap kehilangan, pencatatan yang tidak akurat, pemalsuan kehadiran, dan membutuhkan waktu yang lebih lama dalam pengelolaan data[3]. Hal ini juga berlaku di Laboratorium Teknik Komputer Universitas Lampung, dimana pendataan mahasiswa untuk keluar-masuk laboratorium masih dilakukan secara manual. Hal ini menyulitkan pihak laboratorium dalam memantau aktivitas mahasiswa secara kondusif dalam mendata aktivitas di lab tersebut. Informasi keberadaan asisten laboratorium dan jadwal piket juga sering kali tidak terkoordinasi dengan baik, sehingga menyebabkan keterlambatan dan kurangnya informasi yang akurat bagi mahasiswa yang membutuhkan bantuan asisten.

Meskipun Universitas Lampung telah memiliki sistem akademik unila (Siakadu) dan *virtual class* (Vclass) yang salah satunya digunakan untuk absensi, Laboratorium Teknik Komputer tidak menggunakan siakadu dan vclass dikarenakan perbedaan aturan pelaksanaan kegiatan di praktikum, yang salah satunya adalah jadwal dan kegiatan praktikum yang selalu berubah ubah. Hasil dari kuesioner yang disebarkan kepada mahasiswa aktif teknik informatika unila tahun 2021 – 2024 memperkuat bukti permasalahan yang terjadi di laboratorium teknik komputer terkait kegiatan praktikum, dimana 92,6% partisipan mengungkapkan bahwa mereka mengalami kesulitan ketika akan melakukan praktikum karena waktu yang lama untuk mengisi absensi.

Penerapan *Face Recognition* dapat menjadi solusi digitalisasi absensi di Laboratorium Teknik Komputer Universitas Lampung. Teknologi ini memungkinkan deteksi dan verifikasi wajah mahasiswa secara real-time, menggantikan sistem manual yang rentan manipulasi. Manfaatnya meliputi pengurangan penggunaan kertas, efisiensi rekapitulasi, dan pengelolaan laboratorium yang lebih terstruktur. Dengan demikian, sistem ini diharapkan meningkatkan kemudahan akses data presensi.

1.2 Rumusan Masalah

Berdasarkan latar belakang, rumusan masalah yang mendasari penelitian ini adalah:

1. Bagaimana membangun model *face recognition* melalui *fine-tuning* untuk mengatasi kendala pada sistem absensi manual di Laboratorium Teknik Komputer Universitas Lampung dengan pengujian dilakukan pada periode Juni 2025?
2. Bagaimana performa model *FaceNet* hasil *fine-tuning* dalam mengenali wajah mahasiswa, dibandingkan dengan model *pre-trained* berdasarkan metrik evaluasi embedding dan metrik evaluasi performa terhadap karakteristik dataset?

1.3 Tujuan Penelitian

Tujuan penelitian ini adalah:

1. Mengembangkan model *face recognition* dengan melakukan *fine-tuning* model *FaceNet* untuk mengenali wajah mahasiswa praktikan dan asisten Laboratorium Teknik Komputer Universitas Lampung.
2. Membandingkan performa model *FaceNet pre-trained* dengan *fine-tuning* dalam proses pengenalan wajah pada dataset wajah Laboratorium Teknik Komputer Universitas Lampung.

1.4 Manfaat Penelitian

Adapun manfaat dalam penelitian ini adalah :

1. Menyediakan solusi digital yang dapat mempermudah proses rekapitulasi data absensi di Laboratorium Teknik Komputer Universitas Lampung, sehingga mengurangi risiko kesalahan pencatatan yang sering terjadi pada sistem manual.
2. Meningkatkan transparansi dalam pengelolaan kegiatan laboratorium, khususnya dalam pencatatan kehadiran praktikan dan asisten, serta meminimalkan kemungkinan kehilangan data absensi.

1.5 Batasan Masalah

Dalam penelitian ini, pembatasan masalah meliputi hal-hal sebagai berikut:

1. Penelitian ini difokuskan pada proses *fine-tuning* model *FaceNet* berbasis *PyTorch* menggunakan dataset Laboratorium Teknik Komputer Universitas Lampung. Perbandingan akurasi hanya dilakukan pada model InceptionResNetV1 tanpa melibatkan model *face recognition pre-trained* lainnya dengan menggunakan 5 data mahasiswa.
2. Penelitian ini tidak membahas aspek keamanan sistem absensi, user interface, ataupun implementasi dalam lingkungan produksi secara menyeluruh.

1.6 Sistematika Penulisan Skripsi / Tugas Akhir

Sistematika penulisan skripsi / tugas akhir ini terdiri dari 5 (lima) bab sebagai berikut:

BAB I : PENDAHULUAN

Bab ini membahas mengenai latar belakang penelitian, perumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA

Bagian bab ini memaparkan teori-teori yang terkait dengan penelitian serta sebagai penunjang penelitian.

BAB III : METODE PENELITIAN

Bagian bab ini membahas mengenai waktu dan tempat penelitian, jadwal penelitian, alat penelitian, serta metode penelitian yang digunakan dalam pengembangan Aplikasi

BAB IV : HASIL DAN PEMBAHASAN

Bagian bab ini membahas mengenai hasil dan pembahasan sesuai dengan tahapan-tahapan penelitian.

BAB V : KESIMPULAN DAN SARAN

Bagian bab ini membahas mengenai kesimpulan berdasarkan dari hasil penelitian, serta saran untuk penelitian lebih lanjut.

- DAFTAR : Bab ini memuat daftar sumber kutipan teori yang dijadikan
PUSTAKA acuan dalam penulisan laporan.
- LAMPIRAN : Bagian ini mencakup dokumen dokumen pendukung
penelitian.

II. TINJAUAN PUSTAKA

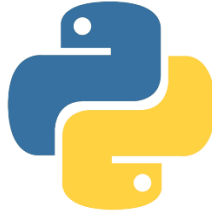
2.1. *Python*

Python adalah bahasa pemrograman populer yang bersifat open source dan digunakan secara luas dalam berbagai domain pengembangan perangkat lunak. Bahasa ini dirancang untuk mudah dibaca dan dipelajari, dengan sintaksis yang sederhana namun powerful. *Python* pertama kali dikembangkan oleh Guido van Rossum pada tahun 1991. Nama *Python* diambil dari acara komedi *Monty Python's Flying Circus*, yang menjadi salah satu inspirasi Guido[4]. *Python* dirancang sebagai bahasa pemrograman yang mudah dipahami, dengan fokus pada keterbacaan kode dan produktivitas pengembang.

Python merupakan salah satu bahasa pemrograman yang banyak digunakan di berbagai bidang karena memiliki sejumlah keunggulan. Sebagai bahasa tingkat tinggi, *Python* menyederhanakan sintaksis pemrograman sehingga lebih mudah dipahami dan mendekati bahasa manusia. Selain itu, *Python* bersifat multi-paradigma, mendukung gaya pemrograman prosedural, berorientasi objek (*object-oriented*), serta fungsional. Fitur *dynamic typing* memungkinkan pengguna mendefinisikan variabel tanpa harus menyatakan tipe datanya secara eksplisit. *Python* juga bersifat portabel, artinya kode program yang ditulis dapat dijalankan di berbagai sistem operasi tanpa perlu banyak penyesuaian[4].

Penggunaan *Python* sangat luas di berbagai bidang. Dalam pengembangan web, framework seperti *Django* dan *Flask* memungkinkan pembuatan aplikasi web secara efisien. Di bidang ilmu data dan kecerdasan buatan (*Artificial Intelligence*), *Python* menjadi bahasa utama berkat ekosistem pustaka seperti *NumPy*, *Pandas*, *TensorFlow*, *Pytorch* dan *Scikit-Learn* yang sangat mendukung proses analisis data dan pelatihan model *machine learning* dan *deep learning*.

Seiring waktu, *Python* terus mengalami perkembangan, baik dari sisi fitur maupun performa. Versi terbaru *Python* 3.x membawa banyak peningkatan dibandingkan versi sebelumnya. Adapun logo *Python* dapat dilihat pada gambar 2.1.



Gambar 2. 1 Logo *Python*

2.2 *Artificial Intelligence*

Artificial Intelligence adalah bidang ilmu komputer yang berfokus pada pengembangan sistem yang dapat melakukan tugas yang biasanya memerlukan kecerdasan manusia[5]. *Artificial Intelligence* pertama kali diperkenalkan oleh Alan Turing pada tahun 1950 melalui konsep Turing Machine dalam makalahnya berjudul “*Computing Machinery and Intelligence*”[6]. Makalah ini memperkenalkan Turing Test-set, sebuah metode untuk menilai apakah sebuah mesin dapat menunjukkan kecerdasan yang setara dengan manusia melalui percakapan tertulis. AI secara resmi diperkenalkan sebagai bidang studi pada Konferensi Dartmouth tahun 1956, yang diorganisir oleh John McCarthy dan rekan-rekannya di Dartmouth College. Konferensi ini juga memperkenalkan istilah “*Artificial Intelligence*”, yang kemudian menjadi disiplin ilmu tersendiri dalam dunia komputasi.

Terdapat empat komponen utama dalam AI yaitu *searching*, *reasoning*, *planning*, dan *learning*[7]. *Searching* merupakan proses dalam menemukan solusi menggunakan beberapa jalur kemungkinan dengan tujuan mencari solusi paling optimal menggunakan kombinasi dan jalur terbaik, seperti algoritma *Breadth-First Search* (BFS) dan *Depth-First Search* (DFS). *Reasoning* merupakan kemampuan AI untuk menarik kesimpulan dari data yang berupa fakta, proses ini mencakup penalaran logika. *Planning* merupakan kemampuan dalam merumuskan langkah langkah untuk mencapai tujuan tertentu, dengan membuat serangkaian tindakan yang efektif serta lebih efisien. *Learning* merupakan kemampuan AI dalam

menyelesaikan masalah melalui proses menganalisis data dari pengalaman yang telah terjadi sebelumnya tanpa dilakukannya program secara eksplisit.

2.3 Machine Learning

Machine Learning (ML) atau Mesin Pembelajaran adalah cabang dari AI yang fokus belajar dari data (*learn from data*), yaitu fokus pada pengembangan sistem yang mampu belajar secara “mandiri” tanpa harus berulang kali diprogram manusia[7]. Untuk bisa mengaplikasikan teknik-teknik *Machine Learning* maka harus ada data. Tanpa data maka algoritma *Machine Learning* tidak dapat bekerja. Data yang ada biasanya dibagi menjadi tiga, yaitu data *training validation*, dan data *Test-seting*. Data *training* digunakan untuk melatih algoritma, Data *validation* Digunakan untuk menguji performa model selama proses training, membantu dalam *hyperparameter tuning*. Sedangkan data *Test-set* digunakan untuk mengetahui performa algoritma yang telah dilatih sebelumnya ketika menemukan data baru yang belum pernah dilihat[8].

2.4 Deep Learning

Deep Learning adalah sub-bidang dari *Machine Learning* yang yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi *non-linear* yang ditata berlapis-lapis dan mendalam. *Deep Learning* sangat baik untuk diterapkan pada *supervised learning*, *unsupervised learning* dan *semi-supervised learning* maupun untuk *reinforcement learning* dalam berbagai aplikasi seperti pengenalan citra, suara, performa teks, dan sebagainya[7]. *Deep Learning* pada dasarnya menggunakan jaringan saraf tiruan (*neural networks*) dengan banyak lapisan (*deep architectures*) untuk memodelkan data yang kompleks. Lapisan-lapisan ini memungkinkan komputer untuk mengekstrak fitur secara otomatis dari data tanpa perlu pemrograman eksplisit[9]. Arsitektur *Deep Learning* yang terkenal untuk memproses data gambar atau video adalah *Convolutional Neural Network* (CNN).

2.5 Pengolahan Citra Digital

Citra adalah representasi visual dari dunia nyata atau objek dalam bentuk yang dapat dipahami oleh komputer[10]. Piksel adalah unit dasar dari citra, setiap piksel memiliki koordinat spasial (baris dan kolom) yang menentukan posisinya dalam citra. Setiap piksel memiliki nilai numerik yang menggambarkan tingkat kecerahan (*grayscale*) atau kombinasi warna (citra berwarna). Resolusi mengacu pada jumlah piksel dalam citra. Citra dengan resolusi tinggi memiliki lebih banyak detail karena memiliki lebih banyak piksel. Citra dengan resolusi rendah cenderung lebih kasar. Format citra mengacu pada cara data citra disimpan dalam memori komputer, seperti JPEG, PNG, atau BMP[10].

Pengolahan citra digital (*digital image processing*) adalah bidang ilmu yang memanfaatkan algoritma komputer untuk memanipulasi, menganalisis, dan mengekstraksi informasi dari citra digital[11]. Proses ini bertujuan untuk meningkatkan kualitas visual citra atau memperoleh data bermakna untuk keperluan diagnostik dan pengambilan keputusan. Tujuan utama dari pengolahan citra digital adalah memproses dan memanipulasi citra digital untuk berbagai keperluan[10]. Pengolahan citra ini melibatkan serangkaian operasi yang bertujuan untuk meningkatkan kualitas citra, mengekstraksi informasi penting, dan membuat citra lebih mudah dipahami atau digunakan dalam berbagai aplikasi. Salah satu tahap awal dalam pengolahan citra digital adalah :

- A. Peningkatan kualitas citra. Pada tahap ini, dilakukan berbagai metode seperti penajaman kontras untuk mempertegas perbedaan antara area terang dan gelap, peningkatan kejelasan agar detail citra terlihat lebih tajam, serta reduksi noise yang berfungsi mengurangi gangguan acak atau bintik-bintik yang dapat mengganggu tampilan citra[10].
- B. Tahap selanjutnya adalah ekstraksi fitur, yaitu proses untuk mengambil karakteristik penting dari citra. Beberapa fitur yang umum diekstraksi meliputi tepi (*edge*), tekstur, dan bentuk objek. Deteksi tepi dilakukan untuk menemukan batas antar objek dalam citra, ekstraksi tekstur untuk mengidentifikasi pola permukaan, serta ekstraksi bentuk yang berfungsi dalam mengenali struktur atau siluet objek di dalam gambar[10].

- C. Setelah fitur diekstraksi, proses segmentasi citra dilakukan untuk memisahkan atau mengelompokkan bagian-bagian dalam citra berdasarkan karakteristik tertentu. Contohnya seperti segmentasi berdasarkan warna, pemisahan objek dari latar belakang, hingga pembagian wilayah homogen dalam citra agar dapat dianalisis secara lebih spesifik[10].
- D. Untuk keperluan efisiensi penyimpanan dan transmisi, dilakukan kompresi citra. Kompresi ini bertujuan untuk mengurangi ukuran file citra. Dalam prosesnya, terdapat dua jenis pendekatan: kompresi tanpa kehilangan (*lossless*) yang menjaga semua informasi tetap utuh, dan kompresi dengan kehilangan (*lossy*) yang mengurangi ukuran file dengan mengorbankan sebagian data citra[10].
- E. Tahap terakhir dalam proses pengolahan citra adalah pengenalan pola, yaitu proses mengidentifikasi objek atau pola tertentu dalam citra. Contohnya meliputi pengenalan objek umum, pengenalan karakter atau teks dalam citra (seperti tulisan tangan atau cetakan), serta pengenalan wajah untuk identifikasi individu[10].

Pengolahan citra digital memiliki beragam aplikasi yang sangat relevan di berbagai bidang. Kemampuan untuk memproses dan menganalisis citra digital membuka pintu bagi banyak inovasi dan perbaikan dalam berbagai industri dan disiplin ilmu. Berikut adalah beberapa aplikasi utama dari pengolahan citra digital

- A. Di bidang kedokteran, pengolahan citra digital telah menjadi salah satu elemen penting dalam sistem pendukung diagnosis dan pemantauan pasien. Melalui citra medis seperti MRI (*Magnetic Resonance Imaging*) dan CT scan (*Computed Tomography*), dokter dapat mendeteksi adanya kelainan jaringan, tumor, atau gangguan lainnya dengan lebih cepat dan akurat. Selain itu, citra digital juga digunakan dalam proses pemantauan perkembangan penyakit secara berkala sehingga memudahkan dalam evaluasi efektivitas pengobatan. Dalam prosedur bedah, sistem pencitraan *real-time* seperti endoskopi juga mengandalkan pengolahan citra untuk memberikan visualisasi internal tubuh yang membantu tenaga medis selama proses operasi berlangsung[10].
- B. Pada sektor industri manufaktur, teknologi pengolahan citra digunakan untuk meningkatkan efisiensi dan menjaga kualitas produk. Sistem kamera dan

algoritma pengolahan citra dapat secara otomatis mengidentifikasi cacat produksi seperti retak, goresan, atau bentuk yang tidak sesuai standar, sehingga proses kontrol kualitas menjadi lebih cepat dan objektif. Selain itu, proses pemantauan jalannya lini produksi pun dapat diotomatisasi untuk memastikan setiap tahapan berjalan sesuai parameter yang telah ditentukan. Tidak hanya itu, pengolahan citra juga berperan dalam pengembangan desain produk, mulai dari tahap *prototyping* hingga simulasi visual produk sebelum diproduksi massal[10].

- C. Dalam bidang kehutanan dan lingkungan, pengolahan citra digital memiliki peran vital terutama dalam pengawasan dan pelestarian lingkungan hidup. Melalui citra satelit atau *drone*, kondisi hutan dapat dipantau secara berkala untuk mendeteksi perubahan tutupan lahan, deforestasi, hingga potensi kebakaran hutan. Informasi ini sangat penting untuk memberikan respon cepat terhadap kerusakan lingkungan. Selain itu, pengolahan citra juga digunakan dalam penelitian ekologi untuk mengamati perkembangan vegetasi, distribusi spesies, serta analisis habitat satwa liar secara lebih komprehensif[10].
- D. Di bidang robotika, pengolahan citra merupakan komponen utama dalam sistem visi komputer (*computer vision*) yang memungkinkan robot memahami dan merespons lingkungan sekitarnya. Robot industri, misalnya, menggunakan kamera dan sistem pengenalan objek untuk mengambil, memindahkan, atau merakit komponen secara presisi. Pada kendaraan otonom seperti mobil tanpa pengemudi, pengolahan citra digunakan untuk mendeteksi jalur jalan, rambu lalu lintas, pejalan kaki, serta kendaraan lain di sekitar, sehingga kendaraan dapat melakukan navigasi secara mandiri dengan aman[10].
- E. Sistem keamanan dan pengawasan juga menjadi salah satu bidang yang sangat diuntungkan dengan hadirnya teknologi pengolahan citra digital. Kamera pengawas (CCTV) yang dipasangkan dengan algoritma deteksi pergerakan atau perilaku mencurigakan dapat digunakan untuk memantau area secara real-time dan otomatis. Selain itu, teknologi pengenalan wajah yang mengandalkan citra digital banyak digunakan untuk sistem keamanan akses, baik di institusi pemerintah, perkantoran, maupun ruang publik, untuk mengidentifikasi identitas individu secara cepat dan akurat[10].

- F. Dalam industri hiburan, terutama perfilman dan permainan video, pengolahan citra digunakan untuk menciptakan efek visual yang kompleks dan realistis. Teknologi ini memungkinkan pembuatan dunia virtual, animasi, hingga manipulasi adegan melalui teknik seperti green screen dan CGI (*computer-generated imagery*). Selain itu, perangkat lunak pengeditan gambar dan video juga mengandalkan pengolahan citra untuk meningkatkan kualitas visual, memperbaiki warna, serta menerapkan filter atau efek artistik sesuai kebutuhan estetika[10].
- G. Terakhir, pada bidang astronomi, pengolahan citra memiliki peran penting dalam analisis data dari teleskop dan citra langit. Objek-objek luar angkasa seperti bintang, planet, galaksi, dan nebula diamati melalui citra digital beresolusi tinggi, kemudian diproses untuk menghilangkan noise dan menonjolkan fitur penting. Melalui teknologi ini, para astronom dapat mengidentifikasi struktur dan fenomena alam semesta yang sebelumnya tidak terlihat dengan jelas, sehingga pengolahan citra turut berkontribusi dalam memperluas pemahaman manusia tentang jagat raya[10].

Berdasarkan bentuk sinyal penyusunnya, citra dapat digolongkan menjadi dua jenis yaitu citra analog dan citra digital.

2.5.1 Citra analog

Citra analog merupakan representasi visual dari informasi dunia nyata dalam bentuk sinyal kontinu, baik dalam domain spasial maupun waktu. Dijelaskan bahwa sumber-sumber citra dari dunia nyata (seperti cahaya yang dipantulkan oleh objek) ditangkap oleh sensor fisik dan mulanya masih berada dalam bentuk analog, misalnya berupa tegangan listrik atau arus yang berubah secara kontinu. Citra analog ini belum dapat diproses secara langsung oleh komputer dan perlu dikonversi terlebih dahulu menjadi citra digital melalui proses *sampling* dan *quantization*. [12]

2.5.2 Citra digital

Citra digital adalah representasi diskrit dari citra analog yang telah melalui proses digitalisasi, yaitu *sampling* (mengubah dimensi spasial menjadi diskrit) dan

quantization (mengubah intensitas menjadi nilai diskrit). Setelah dikonversi, citra digital dapat disimpan, dimanipulasi, dan dianalisis oleh komputer. Citra digital tersusun dari piksel-piksel yang memiliki koordinat (x, y) dan nilai intensitas tertentu[12]. Hasil dari sensor digital akan membentuk array dua dimensi berupa piksel yang dapat dimanipulasi lebih lanjut dengan berbagai teknik pengolahan citra. Gambar digital sendiri didefinisikan sebagai fungsi dua dimensi $f(x, y)$, di mana x dan y adalah koordinat spasial, dan nilai dari f pada pasangan koordinat tersebut disebut sebagai intensitas atau tingkat keabuan (*gray level*)[11]. Pada umumnya, berdasarkan kombinasi warna pada piksel, citra dibagi menjadi tiga jenis yaitu citra RGB, citra grayscale, dan citra biner.

2.5.3 Jenis Jenis Citra

A. Citra RGB (*Red, Green, Blue*)

Citra berwarna mengandung informasi kecerahan dan warna. Representasi paling umum untuk citra berwarna adalah mode RGB (Merah, Hijau, Biru). Setiap piksel dalam citra RGB direpresentasikan oleh tiga nilai, masing-masing mewakili tingkat merah, hijau, dan biru dalam rentang 0 hingga 255[10]. Setiap piksel pada citra RGB, memiliki intensitas warna yang merupakan kombinasi dari tiga nilai intensitas pada kanal R, G, dan B. Banyaknya kombinasi warna piksel yang mungkin pada citra RGB truecolor 24-bit adalah sebanyak $256 \times 256 \times 256 = 16.777.216$.

B. Citra Grayscale

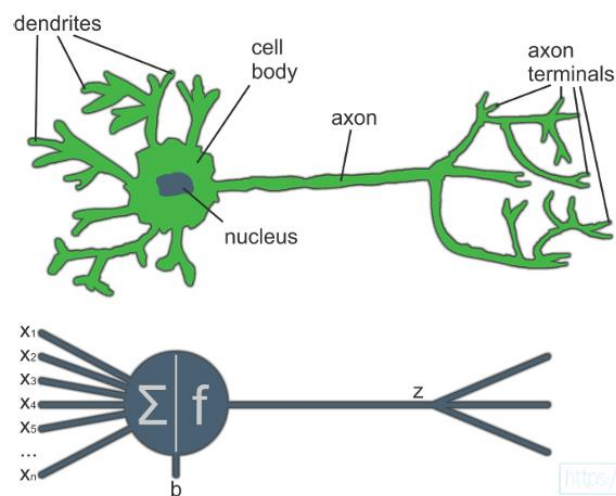
Citra grayscale adalah citra yang hanya memiliki tingkat kecerahan sebagai informasi. Setiap piksel dalam citra grayscale direpresentasikan oleh satu angka, biasanya dalam rentang 0 hingga 255. Nilai 0 mewakili warna hitam, sementara nilai 255 mewakili warna putih[10]. Citra grayscale merupakan citra yang nilai intensitas pikselnya didasarkan pada derajat keabuan. Pada citra grayscale 8-bit, derajat warna hitam sampai dengan putih dibagi ke dalam 256 derajat keabuan di mana warna hitam sempurna direpresentasikan dengan nilai 0 dan putih sempurna dengan nilai 255. Citra RGB dapat dikonversi menjadi citra grayscale sehingga dihasilkan hanya satu kanal warna.

C. Citra biner

Citra biner adalah citra yang pikselnya memiliki kedalaman bit sebesar 1 bit sehingga hanya memiliki dua nilai intensitas warna yaitu 0 (hitam) dan 1 (putih). Citra *grayscale* dapat dikonversi menjadi citra biner melalui proses *thresholding*. Dalam proses *thresholding*, dibutuhkan suatu nilai *threshold* sebagai nilai pembatas konversi. Nilai intensitas piksel yang lebih besar atau sama dengan nilai *threshold* akan dikonversi menjadi 1. Sedangkan nilai intensitas piksel yang kurang dari nilai *threshold* akan dikonversi menjadi 0. Misalnya nilai *threshold* yang digunakan adalah 128, maka piksel yang mempunyai intensitas kurang dari 128 akan diubah menjadi 0 (hitam) dan yang lebih dari atau sama dengan 128 akan diubah menjadi 1 (putih)[10].

2.6 Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan atau *Artificial Neural Network* adalah salah satu cabang dari *Machine Learning* dan menjadi cikal bakal dari *deep learning*. Jaringan syaraf tiruan merupakan model komputasi yang terinspirasi dari cara kerja otak manusia dalam memproses informasi atau perhitungan pada komputer[13]. Jaringan syaraf tiruan terdiri dari neuron-neuron buatan yang saling terhubung untuk melakukan tugas seperti performa, regresi, dan pengenalan pola. Struktur Jaringan Syaraf Tiruan terdiri dari lapisan *input*, *hidden* dan *output*[14].



Gambar 2. 2 Ilustrasi neuron biologis dan neuron buatan[15]

Seperti otak manusia, fungsi dari jaringan ditentukan oleh hubungan antar neuron. Hubungan antar neuron ini disebut bobot (*weight*). Untuk mendapatkan fungsi tertentu dapat dilakukan dengan pelatihan (*training*) dengan menyesuaikan nilai bobot dari masing-masing neuron. Satu sel syaraf terdiri dari tiga bagian, yaitu fungsi penjumlah (*summing function*), fungsi aktivasi (*activation function*), dan keluaran (*output*). Pada umumnya, JST dilatih agar input mengarah ke target output yang spesifik. Jadi jaringan dilatih terus menerus hingga mencapai kondisi di mana input sesuai dengan target yang telah ditentukan. Pelatihan di mana setiap input diasosiasikan dengan target yang telah ditentukan disebut pelatihan terarah atau supervised learning[8].

2.6.1 Komponen jaringan syaraf tiruan

Setiap JST terdiri dari beberapa komponen utama:

- A. Neuron (unit pemroses): Simulasi unit otak yang menerima input dan menghasilkan output.
- B. Bobot (*Weight*): Nilai yang menunjukkan kekuatan koneksi antar neuron.
- C. *Bias*: Nilai tambahan untuk mengatur ambang aktivasi.
- D. Fungsi Aktivasi: Mengubah sinyal masukan menjadi keluaran non-linear.

Rumus perhitungan neuron :

$$y = f \left(\sum_{i=1}^n w_i x_i + b \right) \quad (1)$$

Dimana

y = output neuron

x_i = input ke-i ($i = 1, 2, \dots, n$)

w_i = bobot (*weight*) untuk input ke-i

b = bias

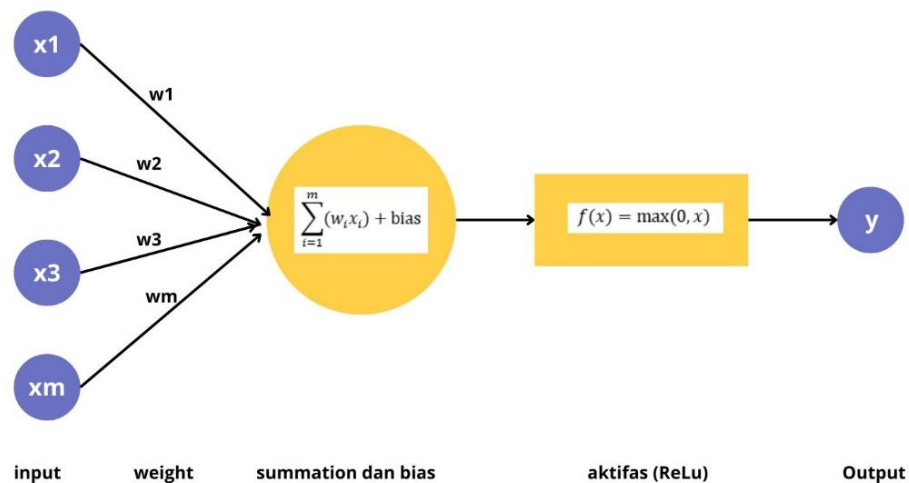
f = fungsi aktivasi

n = jumlah input

2.6.2 Jenis jenis jaringan syaraf tiruan

A. *Single Layer Neural Network*

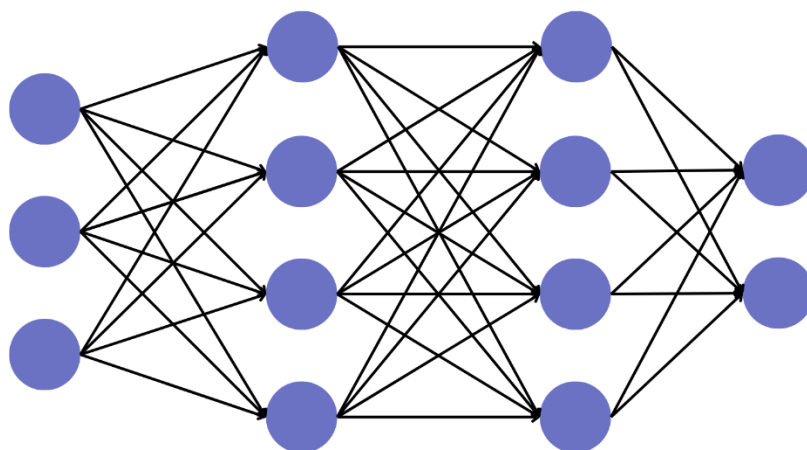
Single Layer Neural Network adalah jenis jaringan saraf tiruan yang hanya memiliki satu lapisan neuron antara *input* dan *output*. Model ini digunakan untuk performa linier sederhana, seperti pemisahan data dengan garis lurus.



Gambar 2. 3 Single Layer Neural Network

B. *Multi Layer Neural Network*

Multi Layer Neural Network adalah jaringan saraf tiruan yang memiliki satu atau lebih *hidden layer* di antara *input* dan *output*. Model ini digunakan untuk masalah yang lebih kompleks, seperti pengenalan pola atau prediksi berbasis data.



Gambar 2. 4 Multi Layer Neural Network

2.6.3 Proses belajar jaringan syaraf tiruan

Backpropagation merupakan algoritma pembelajaran terarah (*supervised learning*) yang umum digunakan dalam pelatihan jaringan syaraf tiruan, terutama pada arsitektur *Multilayer Neural Network*. Algoritma ini bekerja dengan menghitung selisih antara output prediksi dan target (*error*), kemudian menyebarkan informasi kesalahan tersebut ke belakang (*backward*) melalui jaringan untuk memperbarui bobot-bobot yang terlibat. Tujuan utama dari proses ini adalah untuk meminimalkan kesalahan prediksi (*loss*), sehingga jaringan dapat menghasilkan output yang semakin akurat seiring waktu.

A. *Forward propagation*

Input dikalikan dengan bobot, ditambahkan bias, lalu dilewatkan ke fungsi aktivasi. Fungsi aktivasi merupakan komponen penting dalam jaringan syaraf tiruan (JST) yang berperan mengubah input linear menjadi output *non-linear*. Pada fungsi *linear*, setiap perubahan input akan selalu menghasilkan perubahan output secara proporsional. Pada fungsi *non-linear*, input bisa berubah, tapi output belum tentu berubah. Dengan menggunakan fungsi aktivasi, jaringan dapat mempelajari hubungan yang kompleks dan tidak linear dalam data. Fungsi aktivasi menentukan output dari sebuah neuron berdasarkan input yang diterima. Berikut adalah fungsi aktivasi yang umum digunakan dalam jaringan syaraf tiruan.

1. Fungsi *Sigmoid* (*Logistic*)

$$f(x) = \frac{1}{1 + e^{-x}} \quad (2)$$

Fungsi *sigmoid* memiliki karakteristik berupa range output adalah rentang (0, 1), dan cocok untuk performa *biner*.

2. Fungsi *Tanh* (*Hyperbolic Tangent*)

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (3)$$

Fungsi *Tanh* memiliki karakteristik berupa *range output* adalah (-1, 1), bersifat *zero centered* dan konvergensi lebih cepat dibanding *sigmoid*.

3. Fungsi ReLU (*Rectified Linear Unit*)

$$z = \sum_{i=1}^n x_i w_i + b \quad (4)$$

Fungsi ReLU memiliki karakteristik berupa output berada di rentang $(0, \infty)$

4. *Softmax*

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (5)$$

Fungsi *softmax* memiliki karakteristik mengubah output menjadi nilai probabilitas (jumlahnya = 1)

B. Loss Function

Digunakan untuk mengukur seberapa besar kesalahan output JST.

1. *Mean Squared Error* (MSE)

$$E = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (6)$$

2. *Cross Entropy*

$$L = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \quad (7)$$

C. Backpropagation dan optimasi

1. *Gradient Descent*

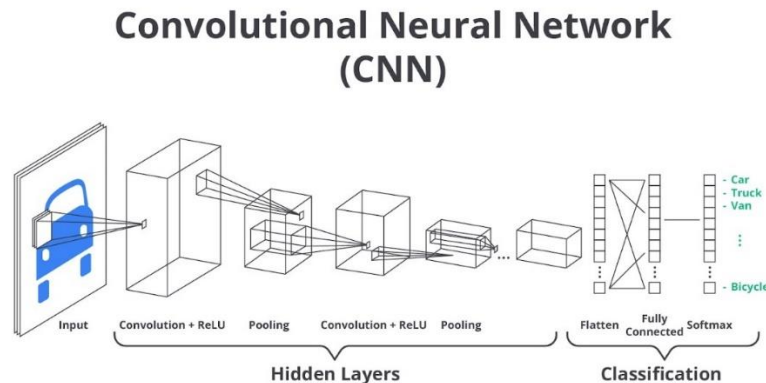
$$w = w - \alpha \cdot \frac{\partial E}{\partial w} \quad (8)$$

2. Pembaruan bobot

$$\Delta w = -\eta \cdot \frac{\partial E}{\partial w} \quad (9)$$

2.7 Convolutional Neural Network

Convolutional Neural Network (CNN) adalah salah satu jenis jaringan saraf tiruan yang dirancang khusus untuk memproses data berbentuk gambar dan video. CNN telah menjadi standar dalam berbagai aplikasi computer vision, seperti performa gambar, deteksi objek, pengenalan wajah, dan segmentasi gambar. Keunggulan CNN terletak pada kemampuannya mengurangi kompleksitas pemrosesan citra dengan menggunakan lapisan konvolusi dan *pooling*, yang memungkinkan model untuk belajar secara otomatis dari data tanpa memerlukan ekstraksi fitur manual menjadi dasar bagi berbagai model *deep learning* modern, termasuk *FaceNet*, *MTCNN*, dan *ResNet*, yang digunakan dalam pengenalan wajah dan analisis citra. Dengan kemampuannya dalam menangkap pola visual yang kompleks, CNN telah mengubah banyak bidang, seperti biometrik, kesehatan, kendaraan otonom, dan sistem keamanan[16].



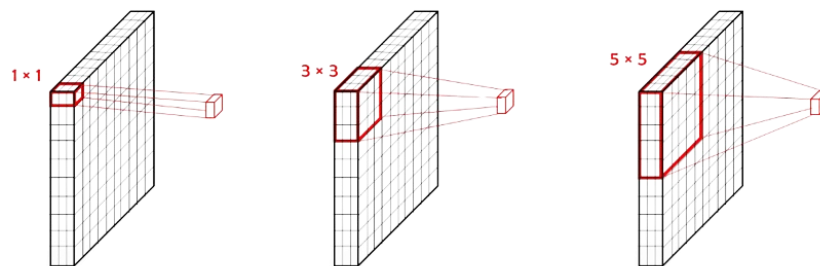
Gambar 2. 5 Arsitektur CNN[16]

2.7.1 Convolutional

Lapisan konvolusi adalah inti dari CNN. Ini bertujuan untuk mengekstrak fitur-fitur penting dari gambar input. *Convolutional* (konvolusi) adalah operasi matematis

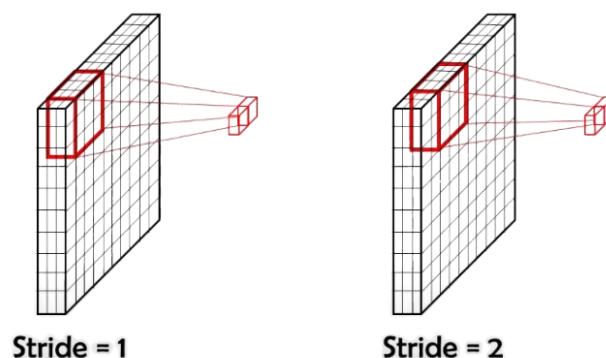
yang digunakan dalam CNN untuk mengekstrak fitur dari data input, seperti gambar. Operasi ini melibatkan penggunaan filter (atau *kernel*) yang digeser melintasi gambar untuk menghasilkan peta fitur (*feature map*). Konvolusi adalah perkalian dari dua matriks, yaitu matriks gambar dan matriks kernel kemudian di jumlahkan / operasi *dot product*.

1. *Filter/Kernel* adalah sebuah matriks kecil yang digunakan untuk mendeteksi pola seperti tepi, tekstur, atau bentuk tertentu dalam gambar.



Gambar 2. 6 Kernal

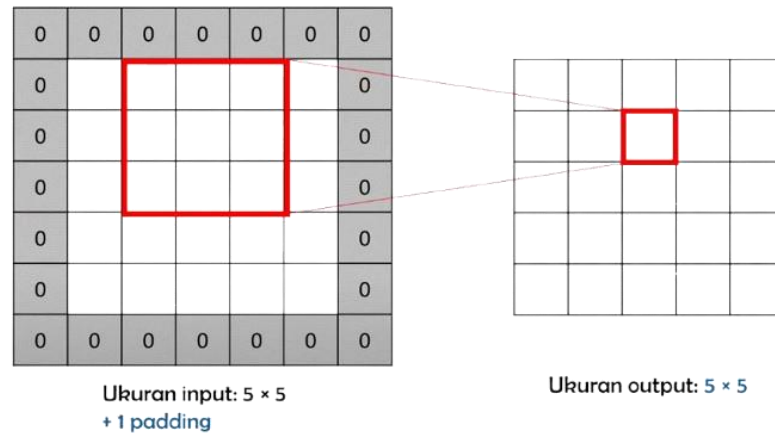
2. *Stride* merupakan jarak pergerakan *filter* melintasi gambar. *Stride* yang lebih besar akan mengurangi ukuran *output*. *Stride* menentukan seberapa banyak *filter* bergerak melintasi gambar. Jika *stride* adalah 1, *filter* bergerak satu piksel pada satu waktu. Jika *stride* adalah 2, filter bergerak dua piksel, dan seterusnya. *Stride* mempengaruhi ukuran output dari operasi konvolusi.



Gambar 2. 7 Pergeseran stride

3. *Padding* adalah proses menambahkan lapisan piksel nol di sekitar gambar input. Ini dilakukan untuk mengontrol ukuran *output* setelah konvolusi. *Padding* dapat

membantu mempertahankan ukuran spasial gambar atau untuk memastikan bahwa *filter* dapat diterapkan ke tepi gambar.

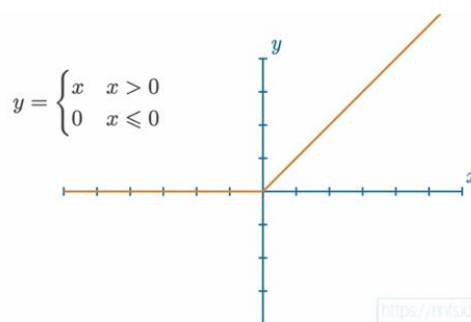


Gambar 2. 8 Padding lapisan piksel

2.7.2 Rectified Linear Unit (ReLu)

ReLU adalah fungsi aktivasi modern yang digunakan untuk memperkenalkan *non-linearitas* ke dalam jaringan. *ReLU* meningkatkan sifat nonlinier dari fungsi keputusan dan jaringan secara keseluruhan tanpa memengaruhi bidang reseptif lapisan konvolusi dan dapat diimplementasikan hanya dengan melakukan ambang batas matriks aktivasi[17]. *ReLU* membantu dalam mempercepat pelatihan dan menghindari masalah *vanishing gradient* yang sering terjadi pada fungsi aktivasi lain seperti *sigmoid* atau *tanh*. Fungsi ini didefinisikan sebagai :

$$\text{ReLU}(x) = \max(0, x) \quad (10)$$



Gambar 2. 9 Grafik *ReLU*

2.7.3 Pooling

Lapisan *pooling* digunakan untuk mengurangi dimensi *spasial* dari peta fitur, sehingga mengurangi jumlah parameter dan komputasi dalam jaringan. Ini juga membantu dalam mengendalikan *overfitting*. *Overfitting* adalah kondisi di mana model terlalu menyesuaikan diri dengan data pelatihan, sehingga performanya sangat baik pada data pelatihan, tetapi buruk pada data baru. Lapisan *pooling* terdiri dari 2 lapisan yaitu *max pooling* digunakan untuk mengambil nilai maksimum dari setiap wilayah yang dicakup oleh filter, dan *average pooling* yang digunakan untuk mengambil nilai rata-rata dari setiap wilayah yang dicakup oleh filter.

2.7.4 Flatten

Setelah beberapa lapisan konvolusi dan *pooling*, output yang dihasilkan adalah matriks multidimensi. Lapisan *flatten* mengubah matriks ini menjadi vektor satu dimensi yang dapat digunakan sebagai input untuk lapisan *fully connected*.

2.7.5 Fully Connected Layer

Lapisan ini mirip dengan lapisan dalam jaringan saraf tradisional. Setiap *neuron* dalam lapisan ini terhubung ke semua *neuron* di lapisan sebelumnya. Ini digunakan untuk menggabungkan fitur-fitur yang diekstrak oleh lapisan konvolusi dan pooling untuk melakukan performa.

2.7.6 Softmax

Softmax adalah fungsi aktivasi yang digunakan pada lapisan output untuk masalah performa multi-kelas. Ini mengubah skor (*logits*) menjadi probabilitas yang jumlahnya sama dengan 1. Fungsi *softmax* didefinisikan sebagai:

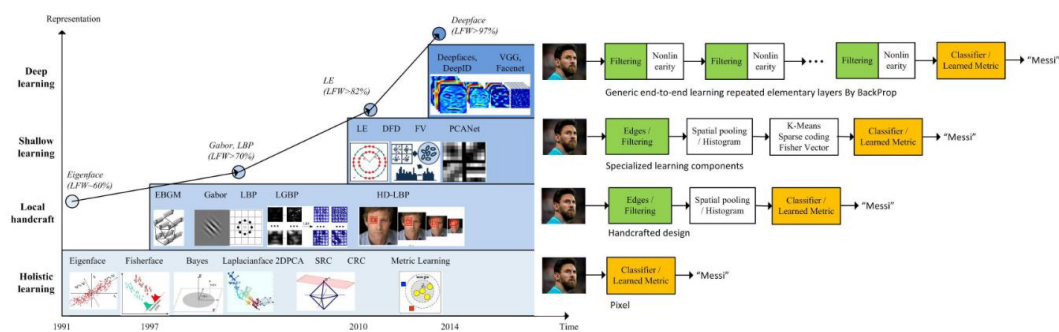
$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \quad (11)$$

2.8 Face Recognition

Face Recognition atau pengenalan wajah adalah teknologi berbasis kecerdasan buatan yang memungkinkan sistem untuk mengidentifikasi atau memverifikasi

seseorang berdasarkan fitur wajahnya. *Face Recognition* bekerja dengan cara mendeteksi, menganalisis, dan membandingkan fitur wajah dari citra atau video dengan database wajah yang telah tersimpan. *Face Recognition* pertama kali dikembangkan oleh Woodrow Bledsoe pada tahun 1964 – 1966 menggunakan teknik semi-manual untuk membandingkan koordinat wajah. Tahun 2014 adalah awal kemajuan *Face Recognition* dari metode klasik ke modern menggunakan *deep learning*.

Penelitian mengenai pengenalan wajah sudah sejak lama dilakukan, pada tahun 1990-an ditemukannya metode pengenalan wajah yang dinamakan *Eigenfaces*. *Eigenfaces* Menggunakan *Principal Component Analysis* (PCA) untuk merepresentasikan wajah dalam bentuk vektor fitur utama. Selanjutnya di tahun 1997 ada metode gabor filter, metode ini Menggunakan filter *Gabor* untuk mengekstrak fitur tekstur wajah dalam berbagai skala dan orientasi. Kemudian pada tahun 2010 terdapat metode *local descriptor*, yaitu Menggunakan teknik *Local Binary Pattern* (LBP), *Scale-Invariant Feature Transform* (SIFT), atau *Histogram of Oriented Gradients* (HOG) untuk ekstraksi fitur wajah. Pada tahun 2014 Menggunakan *Convolutional Neural Network* (CNN) dan arsitektur Deep Learning lainnya untuk mengenali wajah secara otomatis.



Gambar 2. 10 Tonggak sejarah *Face Recognition*[18]

Teknologi pengenalan wajah (*face recognition*) telah mengalami perkembangan pesat sejak awal 1990-an. Gambar 2.10 di atas menunjukkan tahapan penting dalam evolusi representasi wajah untuk pengenalan wajah, yang dibagi ke dalam empat fase utama berdasarkan tingkat kedalaman representasi dan metode yang

digunakan, yaitu: *holistic learning*, *local handcrafted features*, *shallow learning*, dan *deep learning*.

2.8.1 Holistic Learning (1991–2000)

Pada fase awal, teknologi face recognition didasarkan pada pendekatan holistik, di mana citra wajah secara keseluruhan diolah tanpa mengekstraksi fitur lokal secara spesifik. Salah satu metode paling terkenal dalam tahap ini adalah *Eigenface*, yang menggunakan *Principal Component Analysis* (PCA) untuk mereduksi dimensi dan mengekstrak fitur utama dari wajah. Metode lain yang berkembang pada masa ini adalah Fisherface, Bayes classifier, 2DPCA, SRC (*Sparse Representation-based Classification*), dan CRC (*Collaborative Representation-based Classification*). Semua metode ini bergantung pada representasi linear dari citra, dan menghasilkan akurasi pengenalan wajah di kisaran 60% pada dataset *Labeled Faces in the Wild* (LFW). Ciri-ciri utama yaitu menggunakan seluruh citra wajah sebagai input, minim pemrosesan fitur dan akurasi rendah dan rentan terhadap perubahan pencahayaan dan ekspresi[18].

2.8.2 Local Handcrafted Features (2000–2010)

Seiring berkembangnya teknologi, muncul pendekatan baru yang berfokus pada fitur lokal dengan desain manual atau *handcrafted*. Teknik ini bertujuan mengekstraksi bagian-bagian penting dari wajah seperti mata, hidung, dan mulut, yang kemudian diolah menjadi fitur numerik. Metode yang dominan dalam periode ini antara lain: *Gabor* filter untuk mendeteksi orientasi dan frekuensi tekstur wajah. LBP (*Local Binary Pattern*) untuk menangkap pola tekstur mikro pada citra. LGBP (*Local Gabor Binary Pattern*) dan HD-LBP (*High-Dimensional LBP*) Penggunaan fitur lokal ini memberikan peningkatan akurasi signifikan, bahkan mencapai lebih dari 70% pada LFW, karena mampu menangkap informasi yang lebih relevan dan tahan terhadap variasi kecil pada wajah[18].

2.8.3 Shallow Learning (2010–2014)

Memasuki dekade 2010-an, muncul pendekatan baru yang mulai memanfaatkan kemampuan mesin dalam belajar representasi fitur secara otomatis, meskipun

belum sepenuhnya dalam. Pendekatan ini dikenal sebagai shallow learning karena biasanya hanya menggunakan satu atau dua lapisan pembelajaran. Metode yang berkembang pada periode ini mencakup LE (Local Encoder), DFD (*Discriminative Feature Descriptor*), FV (*Fisher Vector*) dan PCANet. Sistem ini masih menggunakan desain modular (fitur performa), tetapi sudah mulai melibatkan algoritma pembelajaran untuk menghasilkan fitur yang lebih optimal dibanding metode manual. Akurasi meningkat menjadi lebih dari 82% pada dataset LFW. Perbedaan dengan tahap sebelumnya yaitu fitur tidak sepenuhnya dibuat tangan, tapi dipelajari dari data dan terdapat komponen seperti *k-means*, *sparse coding*, atau representasi statistik[18].

2.8.4 Deep Learning (2014–sekarang)

Tahun 2014 menjadi titik balik utama dalam sejarah *face recognition* dengan diperkenalkannya pendekatan *deep learning*. Model seperti DeepFace dari Facebook dan DeepID dari Chinese Academy of Sciences menunjukkan performa luar biasa, dengan akurasi lebih dari 97% pada LFW, bahkan meningkat hingga 99.80% hanya dalam waktu tiga tahun. Dalam sistem deep learning, seluruh pipeline pengenalan wajah dibuat *end-to-end*. Artinya, mulai dari pemrosesan citra mentah hingga performa dilakukan oleh jaringan saraf tiruan dalam beberapa lapisan (*deep neural networks*), seperti Convolutional Neural Network (CNN)[18]. Contoh *framework* yang digunakan DeepFace, DeepID, VGG-Face, FaceNet. Proses pembelajaran dilakukan dengan algoritma backpropagation, di mana bobot antar neuron diperbarui berdasarkan error yang dihasilkan dari proses performa. Ciri-ciri utama yaitu tidak lagi membutuhkan desain fitur manual, mampu belajar fitur *non-linear* dan kompleks, skema pelatihan *end-to-end* (filter, aktivasi, performa) dan terakhir adalah sangat kuat terhadap *noise*, rotasi, ekspresi, dan pencahayaan. Beberapa *framework* yang digunakan untuk membangun model *face recognition* antara lain adalah sebagai berikut :

A. OpenCV



Gambar 2. 11 Logo *OpenCV*

Open Source Computer Vision Library (*OpenCv*) adalah sebuah *library open-source* yang sangat populer untuk pemrosesan gambar dan visi komputer. *Library* ini menyediakan berbagai fungsi dan alat untuk membantu pengembang dalam membangun aplikasi yang melibatkan analisis gambar, video, dan visi komputer[19].

B. Pytorch



Gambar 2. 12 Logo *pytorch*

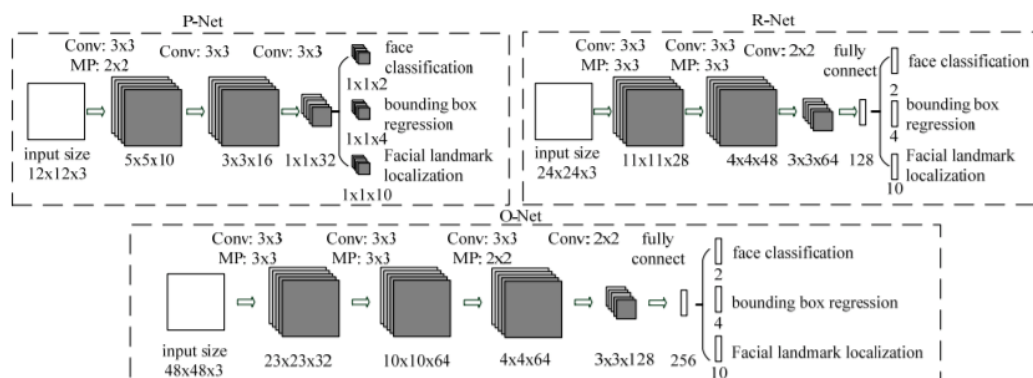
PyTorch adalah *library open-source* untuk komputasi numerik dan pengembangan model *machine learning*, khususnya *deep learning*, yang dikembangkan oleh Facebook's AI Research Lab (FAIR). *PyTorch* dirancang dengan pendekatan *dynamic computation graph* (*eager execution*), yang membuat proses debugging dan eksperimen menjadi lebih intuitif dan fleksibel dibandingkan framework lain seperti *TensorFlow*. Keunggulan *PyTorch* terletak pada kemudahan penggunaannya, integrasi yang baik dengan ekosistem *Python*, serta dukungan komunitas yang luas. *PyTorch* menyediakan API yang powerful untuk membangun dan melatih model *neural network*, serta sangat cocok untuk keperluan riset dan pengembangan model seperti *fine-tuning*, *transfer learning*, dan eksperimen pada berbagai arsitektur jaringan saraf. Selain itu, *PyTorch* juga mendukung GPU

acceleration, sehingga memungkinkan pelatihan model dilakukan dengan lebih cepat dan efisien. [20].

C. MTCNN

Multi-task Cascaded Convolutional Networks (MTCNN) adalah metode *Deep Learning* yang digunakan untuk deteksi wajah dan *face alignment* (penyesuaian wajah) sebelum gambar diproses lebih lanjut dalam sistem *Face Recognition* seperti *FaceNet*. MTCNN bekerja dengan pendekatan tiga tahap bertingkat, yaitu *Proposal Network* (P-Net), *Refine Network* (R-Net), dan *Output Network* (O-Net). MTCNN dirancang untuk menyelesaikan beberapa tugas secara bersamaan yaitu, *face detection*, *facial landmark detection* dan *bounding box regression*. Keunggulan MTCNN dibanding metode lain adalah kemampuannya menangani wajah dalam berbagai sudut, pencahayaan, dan ekspresi dengan lebih baik dibanding metode konvensional[21].

MTCNN digunakan untuk mendeteksi wajah sebelum masuk ke model performa. Metode ini terbukti lebih akurat dibandingkan teknik deteksi wajah tradisional, karena mampu menyesuaikan posisi wajah (*alignment*) secara otomatis dan memberikan *bounding box* yang lebih presisi. Penggunaan MTCNN dalam *Face Recognition* sangat penting karena memastikan bahwa wajah yang diekstrak memiliki skala, rotasi, dan pencahayaan yang optimal, sehingga model pengenalan wajah seperti *FaceNet* dapat menghasilkan *embedding* yang lebih konsisten dan akurat[22].



Gambar 2. 13 Arsitektur MTCNN[23]

1. *The Proposal Network (P-Net)*

Jaringan Proposal ini digunakan untuk memperoleh jendela kandidat dan vektor regresi kotak pembatasnya (*Bounding box*). Regresi kotak pembatas merupakan teknik populer untuk memprediksi pelokalan kotak saat tujuannya adalah mendeteksi objek dari beberapa kelas yang telah ditetapkan sebelumnya, dalam hal ini wajah. Setelah memperoleh vektor kotak pembatas, beberapa penyempurnaan dilakukan untuk menggabungkan daerah yang tumpang tindih. Hasil akhir dari tahap ini adalah semua jendela kandidat setelah penyempurnaan untuk memperkecil volume kandidat[23].

2. *The Refine Network (R-Net)*

Semua kandidat dari P-Net dimasukkan ke dalam *Refine Network*. R-Net selanjutnya mengurangi jumlah kandidat, melakukan kalibrasi dengan regresi kotak pembatas, dan menggunakan penekanan *non-maksimum* (NMS) untuk menggabungkan kandidat yang tumpang tindih. Output R-Net, baik inputnya berupa wajah atau bukan, berupa vektor 4 elemen yang merupakan kotak pembatas untuk wajah, dan vektor 10 elemen untuk lokalisasi landmark wajah[23].

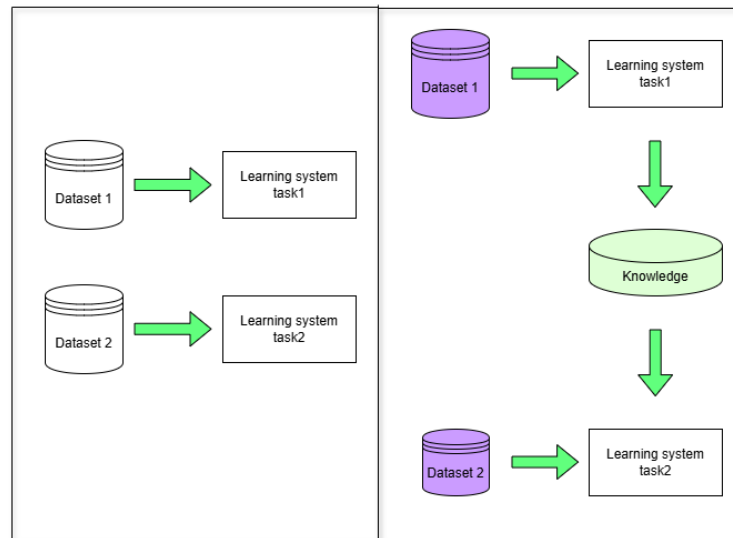
3. *The Output Network (O-Net)*

Tahap ini serupa dengan R-Net, tetapi Jaringan Keluaran ini bertujuan untuk mendeskripsikan wajah secara lebih rinci dan mengeluarkan lima posisi landmark wajah untuk mata, hidung, dan mulut[23].

D. *Transfer Learning*

Dalam membangun sebuah model, terdapat dua jenis pembelajaran utama, yaitu *Tradisional Learning* dan *Transfer Learning*. *Traditional Learning* merupakan metode pembelajaran yang bersifat terisolasi (*isolated learning*) dan hanya dilakukan dalam satu tahap (*single task learning*)[21]. Model yang dilatih tidak menyimpan atau mengakumulasi pengetahuan dari hasil pembelajaran sebelumnya. Oleh karena itu, setiap kali terdapat data baru, model harus dilatih ulang dari awal menggunakan seluruh dataset, baik data lama maupun data baru. Pendekatan ini menyebabkan proses pelatihan menjadi lebih lambat dan membutuhkan sumber

daya komputasi yang lebih besar karena model tidak dapat memperbarui pengetahuannya secara langsung dari pembelajaran sebelumnya.



Gambar 2. 14 *Tradisional learning* dan *Transfer Learning*[21]

Sedangkan teknik *Transfer learning* adalah teknik dalam *Deep Learning* yang memungkinkan model yang telah dilatih pada suatu tugas dengan dataset besar digunakan kembali untuk tugas lain yang memiliki dataset lebih kecil. Teknik ini bertujuan untuk mengurangi waktu dan sumber daya yang dibutuhkan dalam pelatihan model dari awal, dengan tetap mempertahankan fitur yang telah dipelajari sebelumnya. Selain itu, *Transfer Learning* memungkinkan model untuk memperbarui pengetahuannya secara bertahap dengan penambahan data dalam ukuran yang lebih kecil dan fleksibel sesuai dengan kapasitas komputasi yang tersedia. Keunggulan lainnya adalah kemampuannya dalam mengekstraksi fitur secara otomatis dan lebih mendetail, sehingga meningkatkan akurasi model. Dengan pendekatan ini, *Transfer Learning* juga lebih efektif dalam menangani *Big Data* karena memungkinkan pemanfaatan dataset dalam jumlah besar tanpa harus melakukan pelatihan dari awal setiap kali terjadi perubahan data[8].

Modern ini, *Face Recognition* tidak hanya digunakan untuk mengenali manusia, teknologi ini sudah berkembang sangat pesat dan digunakan dalam berbagai sektor yang salah satunya adalah pada sektor perternakan. Salah satu inovasi terbaru adalah penggunaan biometric *facial recognition* untuk identifikasi dan pemantauan

sapi perah. Teknologi ini memungkinkan peternak untuk mengenali setiap sapi secara individual, sehingga mempermudah pemantauan kesehatan, perilaku, dan produktivitas ternak secara otomatis dan lebih akurat[24].

2.9 FaceNet

FaceNet merupakan model *Deep Learning* untuk *Face Recognition*, *verification*, dan *clustering*. *FaceNet* adalah model yang dikembangkan oleh Google, pertama kali diperkenalkan dalam paper "*FaceNet: A Unified Embedding for Face Recognition and Clustering*" oleh Schroff, Kalenichenko, dan Philbin pada tahun 2015[25]. Output utama dari *FaceNet* untuk gambar wajah tertentu adalah embedding berdimensi tinggi. Embedding adalah representasi numerik (vektor) dari data, seperti teks, gambar, atau video, yang menangkap "keterkaitan" atau kesamaan antara item data tersebut. Dalam konteks gambar wajah, embedding adalah vektor angka yang secara unik merepresentasikan fitur-fitur wajah seseorang. *FaceNet* memetakan gambar wajah ke dalam ruang embedding berdimensi tetap sehingga wajah yang mirip akan memiliki embedding vektor yang saling dekat. Dalam implementasinya, kemiripan antar wajah dapat dihitung menggunakan *Cosine Similarity* (arah vektor) maupun *Euclidean Distance* (jarak antar vektor). Model ini dilatih menggunakan *Triplet Loss* agar embedding dari wajah yang sama menjadi lebih dekat dibandingkan dengan wajah orang lain[25].

2.9.1 Arsitektur FaceNet

FaceNet menggunakan *Convolutional Neural Network* (CNN) sebagai backbone, seperti *Inception-ResNet*. CNN bertugas mengekstraksi fitur penting dari citra wajah. Namun, yang membuat *FaceNet* unik bukan hanya CNN-nya, melainkan cara ia melatih model untuk mempelajari jarak antar wajah menggunakan konsep *Triplet Loss*. Berikut adalah beberapa arsitektur dari *FaceNet* beserta performa dari setiap backbone yang digunakan[25].

Tabel 2. 1 Arsitektur FaceNet dan performanya

Arsitektur	Backbone CNN	Uuran Input	Pre-trained Dari	Parameter (Juta)	Akurasi (LFW)
NN1	Zeiler & Fergus-style CNN	220×220	CASIA-WebFace	~140M	98.87
NN2	GoogLeNet/Inception v1	224×224	CASIA-WebFace	~7M	99.30
NN3	Inception-v2	160×160	CASIA-WebFace	~11M	99.50
NN4	Inception-ResNet-v1	160×160	VGGFace2	~24M	99.65
NN5	Inception-ResNet-v2	299×299	MS-Celeb-1M	~55M	99.76
ArcFace	ResNet-100	112×112	MS1M	~65M	99.83

Arsitektur yang akan peneliti gunakan adalah arsitektur NN4 dengan *Backbone CNN Inception-ResNet-v1* yang menggunakan dataset *VGGface2* untuk pelatihan dan menggunakan dataset evaluasi LFW yang menghasilkan akurasi 99.65%. Pemilihan arsitektur ini didasarkan pada kinerjanya yang terbukti unggul dalam berbagai tugas pengenalan citra, termasuk pengenalan wajah. *Inception-ResNet-v1* merupakan kombinasi inovatif dari modul Inception yang efisien dan koneksi residual dari *ResNet*, yang memungkinkan pelatihan jaringan yang sangat dalam dengan performa yang tinggi.

2.9.2 Embedding Vektor

FaceNet tidak menghasilkan label performa, melainkan mengubah setiap wajah menjadi sebuah vektor (misalnya, panjang 128 / 512 dimensi). Proses perbandingan dilakukan dengan menghitung jarak antar vektor menggunakan rumus:

$$d(x_1, x_2) = |f(x_1) - f(x_2)|_2 \quad (12)$$

Dimana Semakin kecil nilai d , semakin besar kemungkinan bahwa dua wajah berasal dari orang yang sama.

2.9.3 Metode *FaceNet*

FaceNet menggunakan *Deep Convolutional Neural Network* (DCNN) yang memetakan wajah seseorang ke dalam *euclidean space*, yaitu ruang geometris tempat setiap wajah direpresentasikan sebagai *embedding* berupa vektor berdimensi tetap. Jarak antar *embedding* ini mencerminkan tingkat kemiripan antar wajah: semakin kecil jaraknya, semakin mirip wajah tersebut, dan sebaliknya[25]. DCNN terdiri dari beberapa lapisan konvolusi, *pooling*, dan *fully connected* yang dirancang untuk mengekstraksi fitur hierarkis dari gambar wajah, dan memetakannya ke ruang *embedding*..



Gambar 2. 15 Metode FaceNet[25]

A. *Batch*

Batch merupakan sekumpulan data yang dikirim ke model neural network dalam satu proses pelatihan sebelum pembaruan bobot dilakukan. Dalam konteks *FaceNet*, satu batch terdiri dari sejumlah gambar wajah yang dipilih secara bersamaan untuk satu iterasi pelatihan. Penggunaan *batch* membuat pelatihan model lebih efisien dan stabil, dibanding memproses satu per satu gambar wajah[25].

B. *Deep Architecture*

FaceNet menggunakan varian dari arsitektur *Inception Network*, khususnya *InceptionResNetV1*, untuk menghasilkan representasi fitur wajah yang efisien dan akurat. Model ini dirancang agar dapat menerima input gambar wajah berukuran 160×160 piksel, sesuai dengan standar yang dibutuhkan untuk menghasilkan *embedding* wajah berdimensi 128-D secara optimal. Arsitektur *FaceNet* yang berbasis *Inception* umumnya terdiri dari tiga bagian utama:

1. *Feature Extraction (Convolutional Layers)*, Layer awal ini terdiri dari beberapa blok Inception yang bertugas mengekstraksi fitur penting dari wajah dalam gambar.
2. *Bottleneck Layer (Fully Connected Layer)*, Setelah fitur diekstraksi, layer ini mengubahnya menjadi vektor representasi yang padat (biasanya disebut *embedding*).
3. *Embedding Layer (Output Layer)*, Layer akhir menghasilkan embedding wajah berdimensi 128-D yang dapat digunakan untuk verifikasi, identifikasi, dan *clustering* wajah.

Inception dipilih karena kemampuannya dalam menangkap informasi multi-skala dengan efisien, yang sangat penting dalam mengenali wajah dengan berbagai pose, pencahayaan, dan ekspresi.

C. L2 normalization

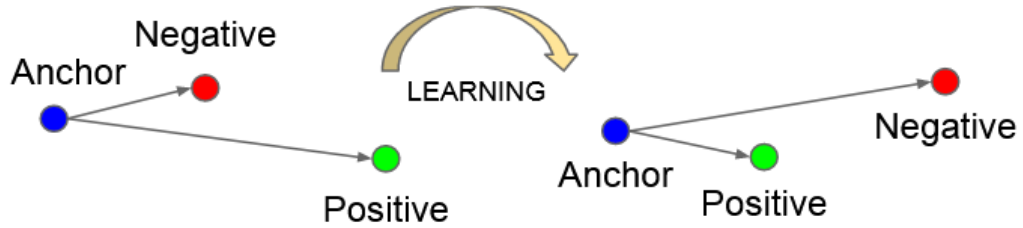
L2 Normalization adalah teknik normalisasi vektor yang membuat panjang vektor menjadi 1 (unit vektor) tanpa mengubah arah vektor. Didalam *FaceNet*, *L2 Normalization* digunakan untuk menstandarkan *embedding* wajah sehingga memiliki skala yang sama, yaitu panjang vektor selalu 1. Ini mempermudah perbandingan antar *embedding* karena hanya mempertimbangkan arah vektor, bukan skalanya[25]. Berikut adalah rumus dalam menormalisasikan vektor menggunakan *L2 Normalization*. Rumus *L2 Normalization*:

$$\hat{x} = \frac{x}{|x|_2} = \frac{x}{\sqrt{\sum_{i=1}^n x_i^2}} \quad (13)$$

D. Triplet Loss

Triplet Loss adalah fungsi *loss* yang digunakan untuk melatih model *FaceNet*, *Triplet Loss* digunakan karena *Face Recognition* merupakan pengenalan, bukan sebuah performa. Tujuannya agar bisa membedakan wajah yang berbeda dan mengenali wajah yang sama dengan lebih baik[25]. *Triplet Loss* bekerja dengan tiga gambar dalam satu *batch* (*triplet*) yaitu *Anchor* (A) sebuah gambar wajah utama (foto pertama seseorang). *Positive* (P) sebuah gambar wajah orang yang sama

dengan *Anchor* (tetapi diambil dari kondisi berbeda, misalnya sudut yang berbeda). *Negative* (N) sebuah gambar wajah orang lain yang berbeda dengan *Anchor*.



Gambar 2. 16 Fungsi *Triplet Loss FaceNet*

Triplet Loss digunakan dengan tujuan untuk membuat *embedding* (vektor fitur) dari wajah yang sama (A & P) lebih dekat satu sama lain, atau wajah yang berbeda (A & N) lebih jauh satu sama lain dengan rumus :

$$L = \sum_{i=1}^N [|f(x_i^a) - f(x_i^p)|_2^2 - |f(x_i^a) - f(x_i^n)|_2^2 + \alpha]_+ \quad (14)$$

2.9.4 Evaluasi metrik dalam *FaceNet*

Evaluasi model *FaceNet* bertujuan untuk menilai seberapa baik model bekerja dalam mengenali wajah. Proses evaluasi ini mencakup pengecekan kemungkinan *overfitting*, yaitu ketika model terlalu menghafal data training tetapi berkinerja buruk pada data baru, serta *underfitting*, di mana model terlalu sederhana sehingga gagal mengenali pola dalam data. Untuk mengevaluasi performa sistem pengenalan wajah berbasis *FaceNet*, terdapat dua jenis metrik evaluasi yang digunakan, yaitu metrik evaluasi utama dan metrik evaluasi tambahan. Evaluasi dilakukan untuk mengetahui seberapa baik sistem mengenali dan membedakan wajah yang sama maupun berbeda.

A. Metrik berbasis *embedding*

Dalam sistem pengenalan wajah seperti *FaceNet*, hasil dari proses ekstraksi fitur adalah vektor *embedding* berdimensi tetap (umumnya 128 dimensi). Untuk membandingkan dua wajah, dilakukan perhitungan jarak atau kesamaan antara dua

vektor embedding tersebut. Dua metrik yang umum digunakan adalah *Cosine Similarity* dan *Euclidean Distance*.

1. *Cosine similarity*

Cosine Similarity digunakan untuk mengukur seberapa besar sudut antara dua vektor embedding. Nilai *cosine similarity* berkisar dari -1 hingga 1, di mana Nilai 1.0 menunjukkan vektor identik (arah sama wajah kemungkinan sama). Nilai 0 menunjukkan vektor tegak lurus (tidak berkaitan). Nilai -1.0 menunjukkan arah yang berlawanan[26].

Rumus *cosine similarity* :

$$\text{CosineSimilarity}(x, y) = \frac{x \cdot y}{|x| \cdot |y|} \quad (14)$$

Keterangan :

- x, y = vektor *embedding* dua gambar wajah
- $x \cdot y$ = dot product
- $\|x\|$ dan $\|y\|$ = norma (panjang) masing-masing vektor

Hasil dari *cosine similarity* adalah semakin tinggi nilainya, semakin besar kemungkinan kedua embedding berasal dari orang yang sama.

2. *Euclidean Distance*

Euclidean Distance mengukur jarak lurus antara dua titik di ruang *embedding*. Nilai jarak yang semakin kecil menunjukkan dua wajah yang semakin mirip. Rumus *euclidean distance* :

$$\text{EuclideanDistance}(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (15)$$

Keterangan :

- x, y = *embedding* wajah
- n = jumlah dimensi (biasanya 128)

Hasil dari *eulidean similarity* adalah jika jarak mendekati 0.0 = embedding sangat mirip dan kemungkinan wajah sama. Namun jika jarak besar = embedding berbeda jauh dan kemungkinan wajah berbeda.

3. t-SNE Visualisasi

t-SNE (*t-distributed Stochastic Neighbor Embedding*) adalah algoritma reduksi dimensi *non-linear* yang digunakan untuk memproyeksikan vektor berdimensi tinggi (misal 128/512 dimensi hasil *FaceNet*) ke dalam 2D atau 3D agar bisa divisualisasikan secara mudah. Visualisasi t-SNE membantu untuk Melihat apakah embedding wajah dari orang yang sama membentuk cluster yang kompak, t-SNE juga digunakan untuk menilai apakah embedding antar orang yang berbeda terpisah jelas, selanjutnya t-SNE digunakan untuk mengevaluasi secara kualitatif apakah model *FaceNet* telah berhasil melakukan representasi fitur wajah dengan baik. Dengan demikian, t-SNE berperan dalam evaluasi kualitas representasi embedding, bukan akurasi performa.

B. Metrik Evaluasi Performa

Dalam sistem verifikasi atau identifikasi wajah menggunakan model seperti FaceNet, performa dilakukan berdasarkan jarak antar embedding. Untuk menilai performa sistem, digunakan beberapa metrik performa seperti *accuracy*, *precision*, *recall*, *F1-score*, *FAR/FRR*, dan *ROC-AUC*. Evaluasi ini penting terutama saat sistem digunakan dalam dunia nyata, seperti untuk verifikasi identitas, absensi otomatis, atau akses keamanan.

1. Accuracy

Mengukur seberapa banyak prediksi yang benar secara keseluruhan.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (16)$$

Keterangan :

- *True Positive* (TP): Jumlah kasus positif yang diprediksi dengan benar sebagai positif.

- *True Negative* (TN): Jumlah kasus negatif yang diprediksi dengan benar sebagai negatif.
- *False Positive* (FP): Jumlah kasus negatif yang salah diprediksi sebagai positif (Type I error).
- *False Negative* (FN): Jumlah kasus positif yang salah diprediksi sebagai negatif (Type II error).

Semakin tinggi, semakin akurat prediksi secara umum. Namun tidak ideal untuk dataset yang tidak seimbang.

2. *Precision*

Mengukur berapa banyak dari prediksi positif yang benar-benar benar.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (17)$$

Precision berguna ketika biaya salah mengenali orang (*False Positive*) mahal, misalnya dalam sistem keamanan.

3. *Recal*

Mengukur seberapa banyak data positif yang berhasil dikenali.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (18)$$

Recall penting dalam kasus di mana kehilangan deteksi (*False Negative*) berbahaya, seperti pada sistem absensi mahasiswa.

4. *F1-Score*

Harmonik rata-rata antara precision dan recall.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (19)$$

F1-Score cocok digunakan jika precision dan recall sama pentingnya, apalagi saat dataset tidak seimbang.

5. *FPR (False Positive Rate)*

Persentase orang yang seharusnya ditolak tetapi diterima. FAR terlalu tinggi akan menyebabkan sistem tidak aman.

$$FPR = \frac{FP}{FP + TN} \quad (20)$$

6. FNR (*False Negative Rate*)

Persentase orang yang seharusnya diterima tetapi ditolak. FRR terlalu tinggi akan menyebabkan pengguna sah kesulitan mengakses.

$$FNR = \frac{FN}{TP + FN} \quad (21)$$

7. ROC (*Receiver Operation Characteristic*) Curve dan AUC (*Area Under Cover*)

ROC (*Receiver Operating Characteristic*) curve memvisualisasikan trade-off antara *true positive rate* dan *false positive rate* pada berbagai *threshold*. AUC (*Area Under the Curve*): luas area di bawah kurva ROC, nilainya antara 0–1. Nilai AUC (1.0) model sempurna, jika (0.5) model acak.

$$AUC = \int_0^1 TPR(x) dx \quad (22)$$

Tabel 2. 2 Metrik performa FaceNet

Metrik	Fungsi
<i>Accuracy</i>	Proporsi prediksi yang benar
<i>Precision</i>	Proporsi prediksi positif yang benar-benar positif
<i>Recall</i>	Proporsi data positif yang berhasil dikenali
<i>F1-Score</i>	Keseimbangan antara precision dan recal
FAR	Tingkat salah terima wajah yang seharusnya ditolak
FRR	Tingkat salah tolak wajah yang seharusnya diterima
ROC-AUC	Luas di bawah kurva ROC, evaluasi keseluruhan saat threshold diubah

8. *Confussion Matrix*

Confusion Matrix (matriks kebingungan) adalah visualisasi yang digunakan untuk mengevaluasi performa performa. Matriks ini menggambarkan perbandingan antara kelas sebenarnya (ground truth) dan kelas yang diprediksi oleh model. Dalam konteks pengenalan wajah, *Confusion Matrix* digunakan untuk menunjukkan berapa banyak wajah dari setiap individu yang dikenali dengan benar (diagonal utama). Mengungkapkan kesalahan prediksi antar kelas (nilai di luar diagonal), dan berguna untuk menganalisis kelas mana yang paling sering tertukar dan menjadi dasar perbaikan model.

9. *P-Value*

P-value adalah salah satu konsep kunci dalam pengujian hipotesis. *P-value* menyajikan tingkat signifikansi hasil pengujian, atau probabilitas mendapatkan hasil seperti yang diamati atau lebih ekstrem jika hipotesis nol benar. Semakin kecil *p-value*, semakin kuat bukti yang dimiliki untuk menolak hipotesis nol. Tingkat signifikansi, yang sering dilambangkan sebagai α (α), adalah ambang batas yang digunakan untuk menentukan kapan kita dapat menolak hipotesis nol. Jika *p-value* kurang dari α , kita dapat menolak hipotesis nol. Tingkat signifikansi ini adalah keputusan yang harus diambil oleh peneliti dan sering kali diatur pada nilai umum seperti 0.05. Selain itu, statistik uji, seperti uji t atau uji Z, digunakan untuk mengukur seberapa jauh hasil sampel kita dari yang diperkirakan oleh hipotesis nol. Statistik uji ini memberikan dasar untuk menghitung *p-value* dan membuat keputusan inferensial yang sesuai[27].

P-value (probability value) adalah probabilitas untuk memperoleh hasil pengujian yang setidaknya sama ekstremnya dengan hasil yang diamati, dengan asumsi bahwa hipotesis nol (H_0) benar. Secara sederhana, *p-value* mengukur seberapa kuat bukti statistik yang tersedia untuk mendukung penolakan terhadap H_0 . Nilai *p-value* berada dalam rentang 0 hingga 1. Semakin kecil *p-value*, semakin kuat bukti untuk menolak H_0 . Sebaliknya, semakin besar *p-value*, semakin lemah bukti untuk menolak H_0 . Terdapat banyak uji *P-value*, salah satunya yang cocok untuk *face recognition* adalah *z-score* dua proporsi. Berikut adalah rumusnya :

$$z = \frac{p_1 - p_2}{\sqrt{p(1-p)\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}} \quad (23)$$

Keterangan :

p_1 = Proporsi keberhasilan model pertama (misalnya: *pre-trained*)

p_2 = Proporsi keberhasilan model kedua (misalnya: *fine-tuning*)

n_1 = Jumlah sampel pada model pertama

n_2 = Jumlah sampel pada model kedua

x_1 = Jumlah prediksi benar oleh model pertama

x_2 = Jumlah prediksi benar oleh model kedua

p = Proporsi gabungan:

2.10 Fine-Tuning

Fine-tuning adalah proses penyesuaian lebih lanjut terhadap model *deep learning* yang sebelumnya telah dilatih pada dataset berskala besar. Proses ini dilakukan agar model dapat bekerja lebih efektif dan akurat pada data yang lebih kecil dan spesifik domain, seperti data wajah mahasiswa dalam konteks sistem absensi otomatis di laboratorium[28]. Dalam dunia *deep learning*, *fine-tuning* merupakan bagian dari strategi *transfer learning*, yaitu pendekatan yang memungkinkan penggunaan kembali pengetahuan (parameter model) dari suatu domain sumber (source domain) ke domain target yang baru (target domain) dengan meminimalkan kebutuhan pelatihan dari awal (*training from scratch*). Dalam konteks *face recognition*, tujuan dari *fine-tuning* adalah untuk menyesuaikan model agar lebih optimal dalam seperti pengenalan wajah, performa gambar medis, deteksi objek, dan lain sebagainya. Pada dasarnya, model *pre-trained* seperti *FaceNet* telah dilatih pada dataset berskala besar seperti *VGGFace2*, sehingga sudah memahami banyak fitur umum dari wajah manusia. *Fine-tuning* terbukti dapat meningkatkan akurasi secara signifikan dibandingkan hanya menggunakan model *pre-trained* tanpa pelatihan ulang[29]. Pada umumnya, alur *fine-tuning* adalah menggunakan model *pre-trained*, membekukan beberapa lapisan awal (*freezing*), mengaktifkan lapisan akhir

(*unfreezing*) dan kemudian melakukan pengaturan *hyperparameter* yang akan digunakan.

2.10.1 Strategi *fine-tuning*

Dalam proses *fine-tuning*, terdapat beberapa strategi pelatihan ulang yang dapat disesuaikan berdasarkan ukuran dan karakteristik dataset baru, serta tujuan dari sistem yang dikembangkan. Pemilihan strategi ini penting untuk menghindari *overfitting* (model terlalu menghafal data) atau *underfitting* (model tidak mampu mempelajari pola).

A. Full *fine-tuning*

Strategi ini dilakukan dengan membuka semua layer dalam model *pre-trained* dan memperbolehkan semua bobot diperbarui selama pelatihan ulang. Strategi ini cocok ketika dataset baru besar dan representatif, tugas yang ingin diselesaikan cukup berbeda dari dataset awal model *pre-trained* dan diperlukan penyesuaian penuh terhadap domain data baru. Kelebihan dari full *fine-tuning* adalah memberikan fleksibilitas maksimum untuk menyesuaikan model ke data baru, sedangkan kekurangannya adalah membutuhkan waktu pelatihan lebih lama dan risiko *overfitting* jika data tidak cukup besar[28].

B. Partial *fine-tuning*

Pada strategi ini, hanya beberapa layer terakhir (*top layers*) dari model *pre-trained* yang dilatih ulang, sementara layer awal dibekukan (*frozen*). Ini umum digunakan ketika dataset baru terbatas jumlahnya, ingin mempertahankan fitur-fitur umum yang sudah dipelajari dari dataset besar sebelumnya (misalnya tepi, sudut, bentuk dasar wajah), dan tugas yang ingin diselesaikan masih serupa dengan tugas pada model *pre-trained*. Kelebihan dari metode ini adalah mengurangi risiko *overfitting* dan mempercepat waktu pelatihan, serta kekurangannya adalah adaptasi terhadap data baru mungkin kurang maksimal jika layer bawah tidak ikut dilatih[28].

C. Feature Extractor

Model *pre-trained* hanya digunakan sebagai pengekstrak fitur (*embedding*)[28]. Embedding yang dihasilkan oleh FaceNet bersifat diskriminatif, artinya wajah-wajah yang berasal dari individu yang sama akan berada dekat dalam ruang vektor, sedangkan wajah dari individu yang berbeda akan berjauhan. Setelah proses ekstraksi embedding selesai, dilakukan proses pengujian (*testing*) dengan membandingkan embedding wajah yang diuji terhadap embedding wajah yang telah tersimpan di basis data (*gallery*). Proses ini tidak menggunakan model performa eksternal seperti SVM atau k-NN, melainkan dilakukan dengan menghitung jarak antar embedding secara langsung, umumnya menggunakan *euclidean distance*, atau *cosine similarity*. Jika jarak antar embedding berada di bawah ambang batas (*threshold*) tertentu, maka sistem menganggap kedua wajah sama dan beda tergantung pada nilai *threshold* yang didapatkan.

2.10.2 *Hyperparameter dalam fine-tuning*

Hyperparameter merupakan parameter eksternal yang ditentukan sebelum proses pelatihan model dan tidak dipelajari secara langsung dari data. Dalam konteks pembelajaran mendalam (*deep learning*), khususnya pada proses *fine-tuning*, *hyperparameter* memainkan peran krusial dalam mengatur perilaku dan performa pelatihan model. Penyesuaian *hyperparameter* yang tepat dapat mempercepat konvergensi, meningkatkan akurasi, dan menghindari permasalahan seperti *overfitting* maupun *underfitting*. Adapun *hyperparameter* yang umum digunakan dalam *fine-tuning* antara lain sebagai berikut:

A. *Learning Rate*

Learning rate adalah besaran atau langkah yang digunakan untuk memperbarui bobot model pada setiap iterasi proses pelatihan. Nilai *learning rate* yang terlalu besar dapat menyebabkan model melompati nilai minimum dari fungsi loss, sehingga pelatihan tidak *konvergen*. Sebaliknya, nilai yang terlalu kecil membuat proses pelatihan sangat lambat dan berisiko terjebak pada *local minima*. Pemilihan *learning rate* yang optimal sangat bergantung pada jenis model, kompleksitas data, dan strategi *fine-tuning* yang digunakan. Untuk *fine-tuning* pada model pra-latih,

biasanya digunakan nilai learning rate yang lebih kecil dibandingkan pelatihan dari awal (misalnya antara 0.00001 hingga 0.001). Menggunakan rumus :

$$w_{t+1} = w_t - \alpha \cdot \nabla L(w_t) \quad (24)$$

B. Batch Size

Batch size adalah jumlah sampel data yang digunakan dalam satu iterasi pembaruan bobot. Penggunaan batch size memengaruhi stabilitas gradien, kecepatan pelatihan, dan konsumsi memori. *Batch* kecil (misal 16 atau 32): menghasilkan pembaruan model yang lebih sering, memperkenalkan variasi gradien yang lebih tinggi, serta mempercepat generalisasi, tetapi dapat menyebabkan pelatihan tidak stabil. *Batch* besar (64 atau lebih): menghasilkan pembaruan yang lebih stabil, tetapi membutuhkan lebih banyak memori dan berisiko *overfitting*. Pemilihan *batch size* ideal juga bergantung pada kapasitas GPU dan ukuran dataset.

C. Epochs

Epoch adalah jumlah siklus penuh di mana model memproses seluruh data pelatihan satu kali. Penentuan jumlah *epoch* berkaitan langsung dengan potensi model dalam mempelajari data. Jumlah *epoch* yang terlalu sedikit menyebabkan model tidak sempat belajar dengan baik (*underfitting*), sementara jumlah yang terlalu banyak meningkatkan risiko *overfitting*. Oleh karena itu, penggunaan teknik seperti *early stopping* sering dipadukan untuk menghentikan pelatihan ketika performa validasi mulai menurun.

D. Early Stopping dan Patience

Early stopping merupakan teknik regulasi yang menghentikan pelatihan ketika performa validasi tidak menunjukkan peningkatan setelah sejumlah *epoch* tertentu. *Parameter patience* menunjukkan jumlah *epoch* berturut-turut tanpa peningkatan sebelum pelatihan dihentikan. Teknik ini membantu mencegah *overfitting*, meningkatkan efisiensi waktu pelatihan, dan sangat berguna dalam kondisi di mana model berisiko kehilangan generalisasi setelah titik tertentu.

E. Margin (Khusus untuk *Triplet Loss*)

Pada model pengenalan wajah seperti *FaceNet* yang menggunakan *triplet loss*, margin adalah nilai batas minimal antara jarak anchor-positive dan anchor-negative. Margin ini bertujuan memastikan bahwa *embedding* wajah dari individu berbeda memiliki jarak yang cukup jauh di ruang vektor, sehingga meningkatkan ketajaman diskriminasi fitur wajah. Pemilihan margin yang terlalu kecil menyebabkan model tidak belajar membedakan wajah dengan baik, sedangkan margin terlalu besar dapat membuat pembelajaran sulit konvergen.

F. Optimizer

Optimizer adalah algoritma yang digunakan untuk memperbarui bobot berdasarkan hasil gradien dari fungsi loss. Pemilihan optimizer memengaruhi kecepatan dan kestabilan pelatihan diantaranya adalah SGD (*Stochastic Gradient Descent*) Stabil dan banyak digunakan, tetapi konvergensi lambat. Kemudian ada Adam (*Adaptive Moment Estimation*) yang memiliki kecepatan tinggi dan adaptif terhadap perubahan gradien. Dan RMSprop Cocok untuk data sekuensial dan masalah dengan gradien fluktuatif. Adam adalah pilihan umum dalam *fine-tuning* karena performanya yang konsisten baik di berbagai domain.

G. Regularization (*Dropout* dan *Weight Decay*)

Dropout digunakan untuk mengurangi overfitting dengan menonaktifkan neuron secara acak selama pelatihan. Sementara *weight decay* (*L2 Regularization*) digunakan untuk menambahkan penalti terhadap bobot besar untuk menghindari kompleksitas model yang berlebihan. Kedua teknik ini membantu *fine-tuning* tetap generalisasi terhadap data baru dan tidak hanya menyesuaikan terhadap data pelatihan.

Rumus *Weight Decay* :

$$L_{\text{total}} = L + \lambda \sum_i w_i^2 \quad (25)$$

H. Loss Function

Fungsi *loss* merupakan ukuran kesalahan yang digunakan model untuk mengevaluasi prediksinya. Dalam pengenalan wajah, fungsi loss seperti *triplet loss*, *contrastive loss*, atau *cross-entropy loss* sering digunakan tergantung pada arsitektur dan pendekatan yang digunakan. Pemilihan fungsi loss yang sesuai penting untuk memastikan bahwa representasi fitur wajah yang dihasilkan benar-benar diskriminatif.

2.11 Google Colab



Gambar 2. 17 Logo Google Colab

Google Colaboratory, atau yang lebih dikenal dengan *Google Colab*, adalah layanan gratis berbasis cloud yang disediakan oleh *Google* untuk menjalankan dan berbagi kode *Python* secara interaktif melalui lingkungan *Jupyter Notebook*[30]. Platform ini banyak digunakan dalam dunia akademik dan industri karena memungkinkan pengguna untuk menulis dan menjalankan kode *Python* secara langsung dari browser, tanpa memerlukan konfigurasi lokal yang rumit. *Google Colab* menjadi sangat populer dalam pengembangan proyek *machine learning*, *data analysis*, hingga pembelajaran *python* karena dukungan GPU/TPU secara gratis, integrasi langsung dengan *google drive*, serta kemudahan kolaborasi secara daring antar pengguna. *Google colab* berjalan di atas infrastruktur *Google Cloud Platform* (GCP) dan mengadopsi komputasi berbasis container untuk menyediakan lingkungan runtime yang ringan dan fleksibel. Proyek ini dikembangkan berdasarkan teknologi open-source Jupyter dan disesuaikan dengan ekosistem *Google*, seperti penyimpanan *Drive*, *Google Sheets*, dan layanan *BigQuery*[30]. Setiap notebook yang dijalankan oleh pengguna akan di-*containerized* dalam lingkungan terisolasi (*sandbox*), dengan akses terbatas namun cukup untuk melakukan eksperimen data secara efisien. Pengguna dapat memilih runtime berbasis CPU, GPU, atau TPU tergantung kebutuhan komputasi.

2.12 *Cursor*



Gambar 2. 18 Logo *Cursor Code Editor*

Cursor adalah sebuah *code editor* modern yang berbasis pada *Visual Studio Code* (VS Code) namun telah ditingkatkan dengan integrasi kecerdasan buatan (AI) untuk membantu proses penulisan kode secara lebih efisien[31]. Berbasis pada kerangka kerja *Visual Studio Code* (VSCode), *Cursor AI* dirancang untuk membantu programmer dalam menulis, memahami, memperbaiki, dan mendokumentasikan kode secara otomatis melalui dukungan teknologi *Large Language Models* (LLM), khususnya GPT-4 dari OpenAI. *Cursor AI* bekerja layaknya asisten coding yang mampu berinteraksi dalam bahasa alami, menjawab pertanyaan seputar kode, menjelaskan fungsi dari potongan program, dan bahkan melakukan *refactoring* sesuai instruksi pengguna. Keunggulan utama *Cursor AI* terletak pada kemampuannya memahami konteks proyek secara menyeluruh, sehingga interaksi antara pengguna dan sistem terasa lebih alami, cepat, dan akurat.

2.13 *GitHub*



Gambar 2. 19 Logo *GitHub*

GitHub adalah platform pengelolaan versi (version control) berbasis Git yang umum digunakan dalam pengembangan perangkat lunak. Secara sederhana, *GitHub*

menyediakan layanan untuk menyimpan, mengelola, dan berkolaborasi pada kode sumber proyek perangkat lunak. Platform ini memungkinkan kolaborasi dalam tim, pelacakan perubahan kode (*version history*), serta penyimpanan kode secara daring (*online repository*)[32]. Pada tahun 2008 oleh Tom Preston-Werner, Chris Wanstrath, dan PJ Hyett, Tom Preston Werner pertama kali mengusulkan gagasan pembuatan sebuah situs web untuk berbagi repositori Git pada 18 Oktober 2007, saat menghadiri pertemuan programmer Ruby bersama Chris Wanstrath. Chris, seorang rekan programmer Ruby, tertarik dengan konsep tersebut dan setuju untuk bekerja sama dalam pengembangan proyek ini. Proses pengembangan awal dilakukan selama tiga bulan, di mana Tom fokus pada antarmuka pengguna dan pengembangan komponen akses repositori *Git* menggunakan bahasa Ruby. Sementara itu, Chris membangun aplikasi menggunakan kerangka kerja Rails. Secara keseluruhan, *GitHub* telah menjadi alat penting dalam proses pengembangan perangkat lunak modern karena fitur-fitur kolaborasi, manajemen kode, serta ekosistemnya yang luas untuk inovasi dan pembelajaran[32].

2.14 Pseudocode

Pseudocode adalah cara penulisan algoritma dalam bentuk kode tiruan, bukan bahasa pemrograman sebenarnya, melainkan deskripsi logika dengan gaya penulisan yang mudah dipahami manusia dan tidak terikat aturan sintaks bahasa pemrograman tertentu[33]. *Pseudocode* biasanya digunakan untuk membantu merancang logika solusi sebelum diterjemahkan ke dalam bahasa pemrograman formal. Penulisan pseudocode yang efektif dan konsisten merupakan aspek penting dalam perancangan algoritma karena berfungsi sebagai penghubung antara desain logika sistem dan implementasi program nyata. Meskipun pseudocode tidak memiliki sintaks formal yang baku seperti halnya bahasa pemrograman tingkat tinggi, tetap diperlukan pedoman atau standar tertentu agar pseudocode dapat dipahami dengan mudah oleh berbagai pihak, baik oleh pengembang sistem, analis, dosen, maupun mahasiswa. Secara umum, pseudocode menggunakan struktur kendali yang menyerupai bentuk perintah dalam bahasa pemrograman prosedural, seperti *IF*, *ELSE*, *FOR*, *WHILE*, *DO*, *BEGIN*, dan *END*[33]. Struktur ini membantu pembaca memahami alur logika tanpa harus memperhatikan aturan sintaks yang

rumit. Bahasa yang digunakan pun merupakan gabungan antara bahasa natural (misalnya Bahasa Indonesia atau Bahasa Inggris) dengan notasi pemrograman yang ringkas dan universal.

2.15 Penelitian Terdahulu

1. Penelitian ini dilakukan oleh Florian Schroff, Dmitry Kalenichenko, dan James Philbin pada tahun 2015 dengan judul "*FaceNet: A Unified Embedding for Face Recognition and Clustering*". Penelitian ini membahas pengembangan sistem FaceNet yang secara langsung memetakan gambar wajah ke dalam ruang Euclidean yang kompak. Sistem ini memungkinkan pengenalan wajah, verifikasi, dan pengelompokan wajah dengan lebih efisien dibandingkan metode sebelumnya. Penelitian ini memiliki tujuan untuk meningkatkan akurasi dan efisiensi sistem pengenalan wajah dengan menghasilkan embedding 128 dimensi yang kompak dan dapat digunakan untuk berbagai tugas pengenalan wajah. Dengan cara melatih deep convolutional neural network menggunakan triplet loss untuk memisahkan embedding wajah dengan identitas yang berbeda dan mendekatkan embedding wajah dengan identitas yang sama. Dataset yang digunakan mencakup jutaan gambar wajah dengan berbagai kondisi pencahayaan dan sudut pandang. Metode yang digunakan adalah Deep Learning dengan triplet loss function, yang terdiri dari tiga gambar dalam satu grup (anchor, positive, negative). Algoritma pelatihan melibatkan pemilihan triplet yang paling sulit (hard triplet mining) untuk meningkatkan konvergensi model. Model CNN yang digunakan didasarkan pada Inception dan Zeiler & Fergus architectures, serta dioptimalkan menggunakan Stochastic Gradient Descent (SGD) dan AdaGrad. Hasil dari penelitian ini adalah FaceNet mencapai akurasi 99.63% pada dataset Labeled Faces in the Wild (LFW) dan 95.12% pada YouTube Faces DB, yang merupakan peningkatan 30% dibandingkan model sebelumnya. Selain itu, FaceNet hanya membutuhkan 128-byte per wajah, menjadikannya sangat efisien untuk implementasi skala besar[25].
2. Penelitian ini dilakukan oleh Ferry Cahyono pada tahun 2020 dengan judul "*Pengenalan Wajah Menggunakan Model FaceNet untuk Presensi Pegawai*". Penelitian ini membahas metode terbaik untuk diterapkan dalam sistem presensi

pegawai berbasis pengenalan wajah, dengan membandingkan performa model FaceNet dan OpenFace. Penelitian ini memiliki tujuan untuk mengembangkan sistem presensi pegawai berbasis pengenalan wajah dengan akurasi tinggi, menggunakan model Deep Learning yang mampu melakukan identifikasi wajah dengan baik. Dengan cara mengambil dataset gambar wajah pegawai, melakukan preprocessing (deteksi, cropping, dan resizing), mengekstraksi fitur wajah menjadi embedding 128 dimensi menggunakan FaceNet, dan mengperformakan hasilnya menggunakan Support Vector Machine (SVM). Metode yang digunakan adalah fine-tuning model FaceNet dan OpenFace, menggunakan k-fold cross validation ($k=5$ dan $k=10$) untuk validasi model, serta membandingkan performa kedua model dalam hal akurasi dan efisiensi sumber daya. Hasil dari penelitian ini adalah model FaceNet menunjukkan akurasi 100%, sedangkan OpenFace memiliki akurasi 93.3%. FaceNet memiliki performa yang lebih baik dalam memisahkan wajah berdasarkan embedding, meskipun OpenFace lebih cepat dalam proses embedding. Penggunaan Support Vector Machine (SVM) sebagai performa juga terbukti efektif dalam mengenali wajah pegawai dengan akurasi yang tinggi.[34]. Penelitian ini memiliki kaitannya dengan penelitian saya karena menggunakan model *FaceNet* serta metode perbandingan *Euclidean Distance* untuk proses pengenalan wajah.

3. Penelitian ini dilakukan oleh Haseeb Younis, Muhammad Asad Arshed, Fawad ul Hassan, Maryam Khurshid, dan Hadia Ghassan pada tahun 2022 dengan judul "*Tomato Disease Classification Using Fine-Tuned Convolutional Neural Network*". Penelitian ini membahas pengembangan model *Deep Learning* untuk mengperformakan penyakit pada tanaman tomat menggunakan teknik *fine-tuning* pada model *MobileNet*. Penelitian ini memiliki tujuan untuk meningkatkan akurasi deteksi penyakit tanaman tomat dengan memanfaatkan teknik *transfer learning* pada model *Deep Learning* yang sudah dilatih sebelumnya. Dengan cara mengumpulkan dataset berisi gambar tanaman tomat yang sehat dan yang terkena penyakit, melakukan preprocessing, lalu melatih model *MobileNet* yang telah dituning ulang (*fine-tuned*) dengan data tersebut. Metode yang digunakan adalah transfer learning dengan model *MobileNet*, di mana model awal dilatih ulang menggunakan dataset baru dengan 10 kelas

penyakit tanaman tomat. Teknik data augmentation juga diterapkan untuk meningkatkan generalisasi model. Hasil dari penelitian ini adalah model yang dihasilkan mencapai akurasi 98,93%, yang lebih tinggi dibandingkan model baseline tanpa *fine-tuning*. Model ini lebih ringan dibandingkan model lain yang digunakan dalam penelitian sebelumnya, sehingga dapat digunakan dalam aplikasi berbasis mobile untuk mendeteksi penyakit tanaman secara real-time[35].

4. Penelitian ini dilakukan oleh A.K. Sharadhi, Vybhavi Gururaj, Sahana P. Shankar, M.S. Supriya, dan Neha Sanjay Chogule pada tahun 2022 dengan judul "*Face Mask Recogniser Using Image Processing and Computer Vision Approach*". Penelitian ini membahas pengembangan sistem deteksi masker wajah berbasis Deep Learning untuk memastikan kepatuhan penggunaan masker, terutama dalam konteks pandemi COVID-19. Penelitian ini memiliki tujuan untuk menciptakan model yang dapat mengenali wajah dengan atau tanpa masker secara real-time, bahkan saat wajah berada pada berbagai sudut pandang, guna membantu pengawasan di tempat umum. Dengan cara menggunakan dataset dari Kaggle yang terdiri dari 7.553 gambar (3.725 dengan masker dan 3.828 tanpa masker), melakukan preprocessing, dan membangun model performa berbasis Deep Learning menggunakan MobileNetV2. Metode yang digunakan adalah Convolutional Neural Network (CNN) dengan arsitektur MobileNetV2, yang dilatih menggunakan TensorFlow dan Keras. Dataset dibagi menjadi 80% data pelatihan dan 20% data pengujian. Model dievaluasi menggunakan metrik precision, recall, dan F1-score. Hasil dari penelitian ini adalah model yang dikembangkan memiliki akurasi 98%, dengan precision 0.97 untuk wajah bermasker dan 1.00 untuk wajah tanpa masker. Model ini juga mampu mendeteksi masker dari sudut samping, memberikan keunggulan dibandingkan metode sebelumnya[22].
5. Penelitian ini dilakukan oleh Rita Goel, Irfan Mehmood, dan Hassan Ugail pada tahun 2021 dengan judul "*A Study of Deep Learning-Based Face Recognition Models for Sibling Identification*". Penelitian ini membahas tantangan dalam mengenali saudara kandung menggunakan model Deep Learning untuk Face Recognition. Penelitian ini mengevaluasi berbagai model Deep Learning,

seperti FaceNet, VGGFace, VGG16, dan VGG19, dalam mengidentifikasi wajah saudara kandung dengan menggunakan berbagai metrik kesamaan. Penelitian ini memiliki tujuan untuk mengevaluasi keandalan model Deep Learning dalam membedakan wajah saudara kandung dengan tingkat kemiripan tinggi serta menentukan model terbaik untuk tugas ini. Dengan cara menggunakan dataset SiblingDB, yang berisi gambar wajah saudara kandung dengan berbagai bagian wajah yang dipotong (full-frontal face, mata, hidung, dan dahi). Penelitian ini menghitung embedding wajah menggunakan model Deep Learning dan membandingkan hasilnya dengan lima metrik kesamaan: cosine similarity, Euclidean distance, structured similarity, Manhattan distance, dan Minkowski distance. Metode yang digunakan adalah eksperimen dengan berbagai model Deep Learning untuk mengekstraksi fitur wajah dan membandingkan performa mereka berdasarkan akurasi, precision, dan tingkat kesalahan performa. Hasil dari penelitian ini adalah model VGGFace memiliki performa terbaik dalam membandingkan wajah secara keseluruhan dan mata dengan akurasi di atas 95%. Namun, akurasi model ini menurun secara signifikan saat membandingkan hidung, di mana FaceNet memberikan hasil terbaik. Model VGG16 dan VGG19 tidak optimal untuk membandingkan mata tetapi lebih baik dalam membandingkan dahi[36].

6. Penelitian ini dilakukan oleh Joseph A. Mensah, Justice K. Appati, Elijah K.A Boateng, Eric Ocran, dan Louis Asiedu pada tahun 2024 dengan judul "*FaceNet recognition algorithm subject to multiple constraints: Assessment of the performance*". Penelitian ini membahas evaluasi kinerja algoritma FaceNet dalam kondisi keterbatasan ganda, yaitu adanya oklusi (occlusion) dan ekspresi wajah yang bervariasi. Tujuan utama penelitian ini adalah menilai bagaimana teknik multiple imputation dapat digunakan untuk mengatasi keterbatasan tersebut dan meningkatkan performa FaceNet. Penelitian dilakukan dengan cara menggunakan dataset gambar wajah dari Japanese Female Facial Expression (JAFPE) dan AU-Coded Cohn-Kanade (CKFE). Gambar uji dibuat dengan menambahkan oklusi sintesis sebesar 30% dan 40% pada gambar wajah dengan ekspresi berbeda. Untuk mengatasi masalah oklusi, tiga metode multiple imputation digunakan, yaitu Multivariate Imputation by Chained Equations

(MICE), MissForest, dan Regularized Expectation Maximization (RegEM). Metode yang digunakan dalam penelitian ini adalah deep learning dengan FaceNet, yang dilatih menggunakan augmentation schemes untuk meningkatkan variasi data latih. Hasil pengenalan wajah dibandingkan menggunakan tiga metode performa: Support Vector Machines (SVM), Euclidean Distance (EUC), dan City Block (CB) distance. Hasil dari penelitian ini adalah: FaceNet mengalami penurunan akurasi signifikan saat menghadapi gambar dengan oklusi dan ekspresi berbeda (akurasi di bawah 40%). Metode imputation (MICE, MissForest, dan RegEM) meningkatkan akurasi pengenalan, dengan MissForest memberikan hasil terbaik (83.8% pada 30% oklusi dan 79.52% pada 40% oklusi). Augmentation Scheme IV (rotasi, flipping horizontal, shearing, brightness adjustment, stretching) memberikan hasil terbaik dengan kombinasi MissForest + City Block Distance/SVM. FaceNet mengungguli metode PCA-based dalam kondisi oklusi tinggi[37]. Penelitian ini memiliki kaitan dengan penelitian saya karena menggunakan FaceNet untuk pengenalan wajah, dan menggunakan evaluasi metrxs yang sama yaitu euclidean distance.

7. Penelitian ini dilakukan oleh John K. Essel, Joseph A. Mensah, Eric Ocran, dan Louis Asiedu pada tahun 2024 dengan judul "*On the search for efficient face recognition algorithm subject to multiple environmental constraints*". Penelitian ini membahas pencarian algoritma pengenalan wajah yang efisien dalam menghadapi berbagai kendala lingkungan, terutama oklusi (occlusion) dan ekspresi wajah yang bervariasi. Fokus utama penelitian ini adalah membandingkan kinerja metode berbasis Principal Component Analysis (PCA) yang ditingkatkan dengan FaceNet, dalam kondisi keterbatasan tersebut. Penelitian dilakukan dengan cara menggunakan dataset gambar wajah dari Japanese Female Facial Expression (JAFPE) dan AU-Coded Cohn-Kanade (CKFE). Gambar uji dibuat dengan oklusi sintesis sebesar 30% dan 40%. Untuk mengatasi masalah oklusi, dua metode multiple imputation digunakan, yaitu MissForest dan Multiple Imputation by Chained Equations (MICE). Metode yang digunakan dalam penelitian ini adalah perbandingan antara metode deep learning (FaceNet) dan metode klasik PCA yang ditingkatkan. PCA

ditingkatkan dengan kombinasi Histogram Equalization (HE), Discrete Wavelet Transform (DWT), dan Gamma Correction (GC) untuk mengurangi kompleksitas komputasi dan mempercepat waktu eksekusi. Hasil pengenalan wajah dibandingkan menggunakan tiga metode performa: Support Vector Machines (SVM), Euclidean Distance (EUC), dan City Block (CB) distance. Hasil dari penelitian ini adalah: FaceNet dan PCA yang ditingkatkan memiliki akurasi yang sebanding pada 30% oklusi, tetapi PCA yang ditingkatkan mengungguli FaceNet pada 40% oklusi. MissForest memberikan hasil terbaik dibandingkan dengan MICE dalam menangani gambar wajah yang teroklusi. Metode HE-GC-DWT-PCA/SVD (versi PCA yang ditingkatkan) memiliki akurasi lebih tinggi daripada FaceNet pada tingkat oklusi 40%, dengan rata-rata 83.33% dibandingkan dengan 79.52% untuk FaceNet. HE-GC-DWT-PCA/SVD juga memiliki waktu komputasi lebih cepat (0.2 detik per gambar) dibandingkan FaceNet (0.98 detik per gambar)[38]. Penelitian ini memiliki kaitan dengan penelitian saya karena menggunakan FaceNet untuk pengenalan wajah, serta membahas pengaruh oklusi dan cara mengatasinya, yang juga relevan dengan proyek skripsi saya tentang sistem absensi berbasis face recognition.

8. Penelitian ini dilakukan oleh Lingling Yang, Xingshi Xu, Jizheng Zhao, dan Huaibo Song pada tahun 2024 dengan judul "*Fusion of RetinaFace and improved FaceNet for individual cow identification in natural scenes*". Penelitian ini membahas pengenalan wajah sapi menggunakan kombinasi RetinaFace dan FaceNet yang telah ditingkatkan untuk mencapai identifikasi sapi yang akurat, non-kontak, dan presisi tinggi di lingkungan pertanian. Tujuan utama penelitian ini adalah mengembangkan sistem identifikasi sapi berbasis deep learning yang mampu menangani variasi postur, pola wajah, kondisi pencahayaan, dan oklusi. Penelitian dilakukan dengan cara menggunakan dataset gambar wajah sapi dari dua peternakan berbeda. Dataset berisi 2.376 gambar wajah dari 110 sapi, mencakup kondisi postur yang berubah, oklusi, pencahayaan rendah, serta pola warna hitam-putih, hitam polos, dan putih polos. RetinaFace digunakan untuk mendeteksi wajah sapi dan menandai fitur kunci, sedangkan FaceNet yang telah ditingkatkan digunakan untuk pengenalan

individu berdasarkan embedding vektor wajah. Metode yang digunakan dalam penelitian ini adalah penggabungan RetinaFace dengan FaceNet yang telah ditingkatkan menggunakan MobileNet sebagai backbone. Beberapa peningkatan utama dalam metode ini meliputi: RetinaFace menggunakan MobileNet-enhanced backbone untuk mengurangi jumlah parameter dan meningkatkan kecepatan deteksi. FaceNet ditingkatkan dengan MobileNetV1 dan penggunaan Cross Entropy Loss bersama Triplet Loss, yang mempercepat konvergensi dan meningkatkan stabilitas model. Hasil dari penelitian ini adalah: RetinaFace menunjukkan tingkat kesalahan deteksi negatif rendah dalam berbagai kondisi: 2.67% (oklusi), 0.66% (tanpa oklusi), 2.67% (cahaya rendah), dan 3.33% (cahaya terang). FaceNet yang telah ditingkatkan mencapai akurasi 99.50% pada data latih dan 83.60% pada data uji, lebih baik dibandingkan metode sebelumnya. Model ini melampaui YOLOX dalam pengenalan sapi, dengan peningkatan akurasi sebesar 2.67% (oklusi), 0.40% (tanpa oklusi), 2.00% (wajah ke atas), 3.34% (menunduk), dan 5.71% (warna putih polos). Model dapat digunakan dalam skenario dunia nyata, termasuk deteksi wajah sapi dalam video dengan resolusi 1920×1080 piksel[39]. Penelitian ini memiliki kaitan dengan penelitian saya karena menggunakan FaceNet untuk pengenalan wajah, serta membahas pengaruh oklusi, variasi postur, dan cara mengatasinya, yang juga relevan dengan proyek skripsi saya tentang sistem absensi berbasis face recognition.

9. Penelitian ini dilakukan oleh Khairul Shaleh Zebua, Iman Herwidiana Kartowisastro, dan Gede Putra Kusuma pada tahun 2023 dengan judul "*Low Resolution Face Recognition Using Combination of GPEN Super Resolution and FaceNet*". Penelitian ini membahas pengenalan wajah dengan resolusi rendah (Low-Resolution Face Recognition, LRFR) menggunakan kombinasi metode super resolusi GPEN dan FaceNet. Tantangan utama dalam LRFR adalah kehilangan detail penting pada gambar resolusi rendah (LR), yang menyebabkan penurunan akurasi dalam identifikasi wajah. Tujuan utama penelitian ini adalah meningkatkan akurasi pengenalan wajah dari gambar beresolusi rendah dengan metode super resolusi GPEN sebelum dikenali oleh FaceNet. Penelitian dilakukan dengan cara menggunakan dataset Labelled

Faces in the Wild (LFW). Dataset ini diseleksi hingga menyisakan 423 kelas (individu) yang memiliki minimal 5 gambar wajah. Data kemudian diperbesar dengan metode augmentasi flipping horizontal. Semua gambar wajah kemudian diproses menggunakan MTCNN untuk mendeteksi dan memotong area wajah, lalu gambar resolusi tinggi (HR) 64×64 piksel diubah menjadi gambar resolusi rendah (LR) 16×16 piksel sebelum diolah oleh model. Metode yang digunakan dalam penelitian ini adalah kombinasi GPEN (GAN Prior Embedded Network) untuk super resolusi dan FaceNet untuk pengenalan wajah. Beberapa langkah yang digunakan: Bicubic interpolation digunakan sebagai baseline untuk membandingkan hasil GPEN. GPEN meningkatkan kualitas gambar sebelum diolah oleh FaceNet. FaceNet melakukan ekstraksi fitur wajah menggunakan metode triplet loss. Support Vector Machine (SVM) digunakan sebagai model performa untuk menilai performa FaceNet. Hasil dari penelitian ini adalah: FaceNet mencapai akurasi 98.4% pada gambar resolusi tinggi (64×64 piksel), tetapi akurasi turun drastis menjadi 66.2% pada gambar resolusi rendah (16×16 piksel). Upscaling gambar menggunakan GPEN meningkatkan akurasi pengenalan hingga 82.8%, lebih baik dibandingkan metode bicubic interpolation yang hanya mencapai 80.1%. Dengan augmentasi data, akurasi pengujian meningkat menjadi 69%, lebih baik daripada tanpa augmentasi. GPEN terbukti lebih unggul dibandingkan bicubic interpolation dalam meningkatkan kualitas gambar LR untuk pengenalan wajah[40]. Penelitian ini memiliki kaitan dengan penelitian saya karena menggunakan FaceNet untuk pengenalan wajah, serta membahas cara meningkatkan akurasi dengan mengatasi masalah resolusi rendah yaitu dengan cara augmentasi data, yang juga relevan dengan proyek skripsi saya tentang sistem absensi berbasis face recognition.

10. Penelitian ini dilakukan oleh Thai-Viet Dang dan Hoai-Linh Tran pada tahun 2023 dengan judul "*A Secured, Multilevel Face Recognition based on Head Pose Estimation, MTCNN and FaceNet*". Penelitian ini membahas sistem pengenalan wajah berlapis yang menggabungkan Head Pose Estimation, MTCNN, dan FaceNet untuk meningkatkan akurasi dan keamanan dalam identifikasi pengguna. Tujuan utama penelitian ini adalah mengurangi

kemungkinan pemalsuan gambar dan meningkatkan efisiensi sistem pengenalan wajah dengan pendekatan berbasis deep learning. Penelitian dilakukan dengan cara menggunakan gambar wajah pengguna yang diambil dalam lima pose berbeda (lurus, menoleh ke kiri, kanan, atas, dan bawah). Sistem pertama-tama memverifikasi apakah semua pose sudah diinput dengan benar menggunakan algoritma Head Pose Estimation. Setelah verifikasi, gambar dengan wajah lurus diproses menggunakan MTCNN untuk mendeteksi dan mengekstrak wajah, yang kemudian diinput ke dalam FaceNet untuk identifikasi akhir. Metode yang digunakan dalam penelitian ini adalah gabungan Head Pose Estimation, MTCNN, dan FaceNet, dengan langkah-langkah utama sebagai berikut: Head Pose Estimation digunakan untuk memastikan bahwa pengguna benar-benar memberikan lima pose wajah yang berbeda, sehingga mengurangi kemungkinan pemalsuan gambar atau penggunaan foto statis. MTCNN digunakan untuk mendeteksi wajah dan mengekstrak area wajah yang akan diinput ke dalam FaceNet. FaceNet digunakan untuk ekstraksi fitur wajah dan identifikasi individu dengan metode embedding 128 dimensi. Proses performa dilakukan menggunakan K-Nearest Neighbors (KNN) untuk mencocokkan fitur wajah dengan database pengguna. Hasil dari penelitian ini adalah: Sistem yang diusulkan mencapai akurasi antara 92% hingga 95%, lebih tinggi dibandingkan metode pengenalan wajah tanpa Head Pose Estimation (sekitar 85%). Preprocessing dengan Head Pose Estimation meningkatkan akurasi deteksi wajah oleh MTCNN hingga 100% pada dataset pengujian. Sistem dapat berjalan secara efisien pada perangkat embedded seperti Jetson Nano, dengan kecepatan pemrosesan 21-25 frame per detik (fps). Metode ini lebih aman dibandingkan sistem pengenalan wajah tradisional karena mempersulit pemalsuan dengan gambar statis atau masker [41]. Penelitian ini memiliki kaitan dengan penelitian saya karena menggunakan FaceNet untuk pengenalan wajah, serta membahas cara meningkatkan akurasi dan keamanan sistem dengan Head Pose Estimation, yang juga relevan dengan proyek skripsi saya tentang sistem absensi berbasis face recognition.

III. METODOLOGI PENELITIAN

3.1 Waktu dan Tempat

Penelitian dilakukan di laboratorium Teknik Komputer Unila dengan peralatan simulasi. Waktu penelitian dimulai dari bulan January 2025 sampai dengan bulan Juni 2025.

3.2 Jadwal penelitian

Jadwal penelitian adalah sebagai berikut :

Tabel 3. 1 Jadwal Penelitian

No	Aktivitas	Januari 2025	Februari 2025	Maret 2025	April 2025	Mei 2025	Juni 2025
1.	Penentuan Kebutuhan						
2.	Penentuan User						
3.	Perancangan						
4.	Work on Wait(WOW)						
5.	Work in Progress (WIP)						
6.	Testing						
7.	Done						
8.	Pembuatan Laporan						

3.3 Alat dan bahan penelitian

Alat dan bahan yang di gunakan dalam penelitian ini adalah sebagai berikut :

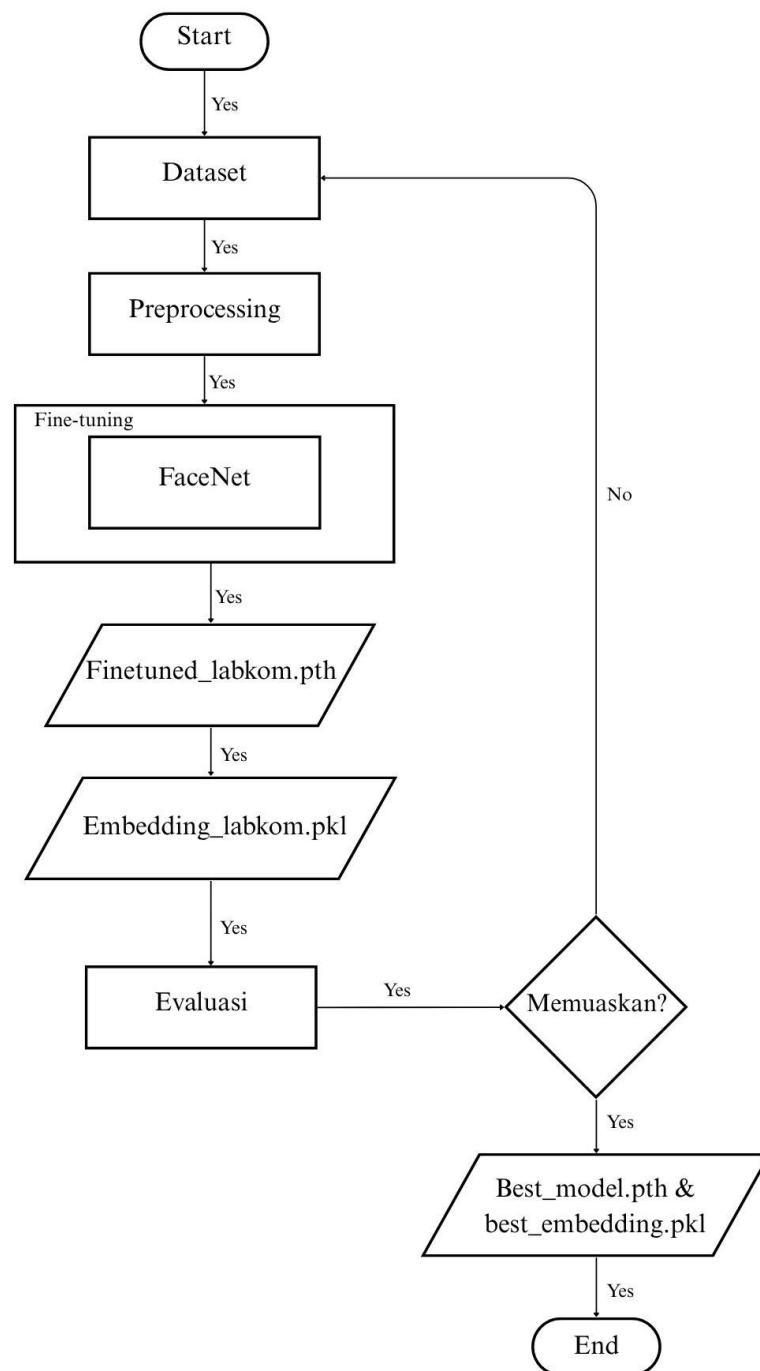
Tabel 3. 2. Alat penelitian

No	Nama Alat	Spesifikasi	Deskripsi
1.	Komputer (AIO) ACER Veritonz4	Intel Core i7, dengan Ram 8, OS Windows 11 pro	Perangkat keras yang digunakan dalam keseluruhan penelitian, seperti data analisis, data preprocessing, data modelling, dan data evaluation hingga penyusunan laporan.
2.	Google Collab	Website	Perangkat lunak yang digunakan untuk melakukan tahap preprocessing data dan ujicoba fine-tuning
3.	Cursor	1.0 update June 4, 2025	Perangkat lunak yang untuk melakukan tahap implementasi code fine-tuning hingga evaluasi dan Test-seting.
4.	Python	3.10.7 (venv)	Bahasa pemrograman utama dalam melakukan pengembangan model
5.	Pytorch	2.2.2	PyTorch adalah fondasi utama yang menjalankan seluruh sistem FaceNet kamu.
6.	Github	Website	Perangkat lunak yang digunakan untuk manajemen proyek untuk dan dokumentasi code dengan tim.
7.	Handphone	RealmeC11	Perangkat keras yang digunakan untuk pengumpulan dataset

Tabel 3. 3. Bahan penelitian

No	Nama bahan	Spesifikasi	Deskripsi
1.	Dataset_Labkom2025	Foto asisten laboratorium teknik komputer dan mahasiswa praktikan,	Data utama yang di gunakan untuk fine-tuning dan evaluasi model
2.	Model pre-trained	vggface2-features.pth	Model utama yang akan digunakan untuk melakukan fine-tuning

3.4 Tahapan Penelitian



Gambar 3. 1 Tahapan *fine-tuning* FaceNet

3.4.1 Dataset

Dataset yang digunakan terdiri dari 90 foto wajah untuk masing-masing dari 30 mahasiswa yang telah memberikan persetujuan. Pengambilan gambar dilakukan dengan kamera ponsel realme C11 di berbagai lokasi (*hybrid*). Foto disimpan dalam format JPG lalu dikelompokkan ke dalam subfolder berdasarkan nama responden.

A. Teknik Pengumpulan Data

Teknik pengumpulan data dilakukan melalui sesi pemotretan yang difasilitasi di Laboratorium Teknik Komputer menggunakan perangkat ponsel Realme C11. Namun, mengingat keterbatasan waktu dan mobilitas dari beberapa responden, peneliti memberikan opsi kepada responden untuk melakukan sesi foto secara mandiri di lokasi pilihan masing-masing. Setelah proses pengambilan gambar selesai, responden diminta untuk mengelompokkan hasil foto mereka ke dalam satu folder yang diberi nama sesuai dengan nama lengkap masing-masing. Setelah foto siap dikumpulkan, responden diminta mengirimkan dalam format ZIP melalui *Google Form* atau *Google Drive* yang telah disediakan oleh peneliti.

B. Kriteria Dataset

Tabel 3. 4 kriteria dataset

No	Kriteria	Deskripsi	Jumlah
1.	Aksesoris Wajah	1. Kepala Miring Ke Kanan, 2. Kepala Miring Ke Kiri, 3. Kepala Miring Ke Atas, 4. Kepala Miring Kebawah, 5. Kepala Miring Senyum, 6. Ekspresi Bingung, 7. Ekspresi Marah, 8. Ekspresi Diam, 9. Jarak Jauh, 10. Jarak Dekat	10
2.	Ekspresi Biasa	1. Ekspresi Biasa Miring Ke Kanan, 2. Ekspresi Biasa Miring Kekiri, 3. Ekspresi Biasa Miring Keatas, 4. Ekspresi Biasa Miring Kebawah, 5. Dari Sudut Atas, 6. Dari Sudut Bawah, 7. Dari Sudut Samping Kanan,	10

		8. Dari Sudut Samping Kiri 9. Jarak Dekat 10. Jarak Jauh (+ 1 Meter)	
3.	Ekspresi Bingung	1. Ekspresi Bingung Miring Ke Kanan, 2. Ekspresi Bingung Miring Kekiri, 3. Ekspresi Bingung Miring Keatas, Ekspresi Bingung Miring Kebawah, 4. Dari Sudut Atas, 5. Dari Sudut Bawah, 6. Dari Sudut Samping Kanan, 7. Dari Sudut Samping Kiri 8. Jarak Dekat 9. Jarak Jauh (+ 1 Meter)	10
4.	Ekspresi Diam	1. Ekspresi Diam Miring Ke Kanan, 2. Ekspresi Diam Miring Kekiri, 3. Ekspresi Diam Miring Keatas, 4. Ekspresi Diam Miring Kebawah, 5. Dari Sudut Atas, 6. Dari Sudut Bawah, 7. Dari Sudut Samping Kanan, 8. Dari Sudut Samping Kiri 9. Jarak Dekat 10. Jarak Jauh (+ 1 Meter)	10
5.	Ekspresi Marah	1. Ekspresi Marah Miring Ke Kanan, 2. Ekspresi Marah Miring Kekiri, 3. Ekspresi Marah Miring Keatas, 4. Ekspresi Marah Miring Kebawah, 5. Dari Sudut Atas, 6. Dari Sudut Bawah, 7. Dari Sudut Samping Kanan, 8. Dari Sudut Samping Kiri 9. Jarak Dekat 10. Jarak Jauh (+ 1 Meter)	10
6.	Ekspresi Senyum	1. Ekspresi Senyum Miring Ke Kanan, 2. Ekspresi Senyum Miring Kekiri, 3. Ekspresi Senyum Miring Keatas, 4. Ekspresi Senyum Miring Kebawah, 5. Dari Sudut Atas, 6. Dari Sudut Bawah, 7. Dari Sudut Samping Kanan, 8. Dari Sudut Samping Kiri 9. Jarak Dekat 10. Jarak Jauh (+ 1 Meter)	10
7.	Pencahayaan Normal	1. Pencahayaan Normal Miring Ke Kanan, 2. Pencahayaan Normal Miring Kekiri, 3. Pencahayaan Normal Miring Keatas,	10

		4. Pencahayaan Normal Miring Kebawah, 5. Dari Sudut Atas, 6. Dari Sudut Bawah, 7. Dari Sudut Samping Kanan, 8. Dari Sudut Samping Kiri 9. Jarak Dekat 10. Jarak Jauh (+ 1 Meter)	
8.	Pencahayaan Terang	1. Pencahayaan Terang Miring Ke Kanan, 2. Pencahayaan Terang Miring Kekiri, 3. Pencahayaan Terang Miring Keatas, 4. Pencahayaan Terang Miring Kebawah, 5. Dari Sudut Atas, 6. Dari Sudut Bawah, 7. Dari Sudut Samping Kanan, 8. Dari Sudut Samping Kiri 9. Jarak Dekat 10. Jarak Jauh (+ 1 Meter)	10
9.	Pencahayaan Gelap	1. Pencahayaan Gelap Miring Ke Kanan, 2. Pencahayaan Gelap Miring Kekiri, 3. Pencahayaan Gelap Miring Keatas, 4. Pencahayaan Gelap Miring Kebawah, 5. Dari Sudut Atas, 6. Dari Sudut Bawah, 7. Dari Sudut Samping Kanan, 8. Dari Sudut Samping Kiri 9. Jarak Dekat 10. Jarak Jauh (+ 1 Meter)	10
Total Keseluruhan			90

3.4.2 Preprocessing

Proses *preprocessing* merupakan tahapan penting sebelum melakukan *fine-tuning* model *FaceNet* berbasis *PyTorch*. Tahap ini bertujuan untuk memastikan bahwa seluruh data gambar dalam kondisi bersih, terstandarisasi, dan layak digunakan dalam proses pelatihan model. Berikut merupakan tahapan-tahapan preprocessing yang dilakukan dalam penelitian ini.

A. Preprocessing manual

Langkah awal dalam preprocessing adalah mengorganisasi seluruh dataset yang telah dikirimkan oleh responden ke dalam folder berdasarkan label nama masing-masing mahasiswa. Dataset yang diterima berasal dari berbagai format, lokasi dan perangkat yang berbeda, sehingga perlu dilakukan tindakan lebih lanjut untuk memastikan data valid terlebih dahulu. *Preprocessing* manual dilakukan untuk beberapa aspek seperti tipe data yang digunakan dan format dataset.

B. Face Detection

Setelah memastikan seluruh gambar berada dalam kondisi yang sesuai dan memiliki format JPG, tahap berikutnya adalah melakukan deteksi wajah menggunakan pustaka OpenCV dan MTCNN (*Multi-task Cascaded Convolutional Networks*). Deteksi wajah ini bertujuan untuk mengidentifikasi koordinat posisi wajah dalam gambar dan mengabaikan area non-wajah yang tidak relevan terhadap proses pelatihan.

C. Cropping

Gambar wajah yang telah terdeteksi kemudian dipotong (*cropped*) untuk mengambil hanya bagian wajah. Hal ini dilakukan agar model hanya mempelajari fitur-fitur penting dari wajah dan tidak terganggu oleh latar belakang atau elemen lainnya.

D. Resize

Wajah yang telah melalui proses *cropping* selanjutnya diubah ukurannya menjadi 160×160 piksel, sesuai dengan spesifikasi input yang dibutuhkan oleh arsitektur FaceNet. Ukuran ini merupakan standar yang direkomendasikan agar *embedding* yang dihasilkan optimal.

E. Pembagian dataset

Dataset hasil *preprocessing* terdiri dari 30 kelas, masing-masing mewakili satu mahasiswa. Total gambar yang berhasil disiapkan untuk pelatihan berjumlah 2.704 gambar dari jumlah awal 3.000 gambar. Pengurangan jumlah ini disebabkan oleh adanya gambar yang tidak lolos proses seleksi dan *preprocessing*. Kemudian

dataset hasil preprocessing dibagi menjadi 3 kategori data dengan rasio pembagian 70.20.10. pembagian ini didasarkan untuk masuk tahap *fine-tuning*, dikarenakan hanya mengupdate informasi tanpa melatih dari awal.

1. Data Pelatihan (*Training Set*)

Data pelatihan digunakan untuk melatih model dalam proses *fine-tuning*. Model akan mempelajari pola-pola penting dari wajah pada tahap ini. Dataset ini juga digunakan dalam proses pembentukan *triplet* (*anchor*, *positive*, *negative*) yang dibutuhkan dalam penerapan *triplet loss*.

2. Data Validasi (*Validation Set*)

Data validasi digunakan untuk mengevaluasi performa model selama proses pelatihan, terutama untuk mendeteksi *overfitting* dan mengatur mekanisme *early stopping*. Model tidak diperbolehkan belajar dari data ini secara langsung.

3. Data Pengujian (*Test Set*)

Data pengujian digunakan untuk melakukan evaluasi akhir terhadap model setelah proses pelatihan selesai. Tujuannya adalah mengukur kemampuan model dalam mengenali wajah secara objektif pada data yang benar-benar belum pernah dilihat sebelumnya.

3.4.3 *Fine-tuning FaceNet*

Dalam pengembangan sistem absensi digital ini, digunakan arsitektur *FaceNet* yang berbasis pada model *Inception-ResNet-v1*, yang diimplementasikan melalui *library FaceNet_pytorch*. Pemilihan arsitektur ini didasarkan pada keunggulannya dalam menghasilkan embedding wajah yang kuat (robust) dan konsisten dalam berbagai kondisi pencahayaan, pose, maupun ekspresi.[25]

A. Modifikasi layer untuk *fine-tuning*

Lapisan *block8* merupakan bagian akhir dari *stack convolutional* pada arsitektur *FaceNet* berbasis *Inception-ResNet*. Lapisan ini menangani fitur wajah tingkat

tinggi yang penting dalam tugas identifikasi. Sementara itu, lapisan *last_linear* adalah *fully connected* layer terakhir yang berfungsi memproyeksikan fitur wajah ke dalam *embedding vector*. *Fine-tuning* pada dua lapisan ini memungkinkan model menyesuaikan representasi akhir dengan distribusi data baru.

B. Hyperparameter

Hyperparameter merupakan parameter konfigurasi yang menentukan perilaku proses *training* model. Pemilihan *hyperparameter* yang tepat sangat krusial untuk keberhasilan *fine-tuning* model *FaceNet*. *Hyperparameter* dikonfigurasi secara khusus untuk mengoptimalkan proses *fine-tuning* pada dataset wajah mahasiswa di Laboratorium Teknik Komputer Universitas Lampung. Diantara konfigurasi yang dilakukan adalah sebagai berikut :

1. Batch size:

Batch size menunjukkan berapa banyak data wajah yang diproses sekaligus dalam satu kali pelatihan (*iterasi*). Nilai 16 dipilih untuk menjaga keseimbangan antara kebutuhan memori dan stabilitas pelatihan. Jumlah ini cukup kecil agar bisa dijalankan di perangkat dengan sumber daya terbatas (seperti laptop atau GPU standar), namun masih cukup besar untuk menghasilkan pembelajaran yang stabil. *Batch* yang terlalu kecil dapat membuat pelatihan lebih *fluktuatif*, sedangkan batch terlalu besar bisa membuat pelatihan lebih lambat dan memakan banyak memori.

2. Epochs

Epoch adalah istilah yang digunakan untuk menyatakan berapa kali seluruh dataset dilalui oleh model dalam proses pelatihan. *Epoch* merupakan satu siklus penuh di mana seluruh dataset digunakan untuk melatih model satu kali. Dalam penelitian ini, model dilatih selama 10 epoch. Karena model *FaceNet* telah melalui proses pre-training pada dataset skala besar, pelatihan ulang dalam jumlah *epoch* yang terbatas dinilai sudah cukup untuk menyesuaikan representasi fitur terhadap data baru tanpa mengalami *overfitting*. Penggunaan epoch yang terlalu banyak dapat menyebabkan model terlalu menyesuaikan diri dengan data pelatihan, sementara jumlah *epoch* yang terlalu sedikit dapat menyebabkan model tidak belajar secara optimal.

3. *Learning rate*

Learning rate merupakan *parameter* yang menentukan besar langkah pembaruan bobot model pada setiap iterasi pelatihan. Nilai 0.0001 digunakan untuk menjaga stabilitas proses pembelajaran, terutama pada tahap *fine-tuning* model *pre-trained*. Nilai ini memungkinkan perubahan bobot dilakukan secara bertahap dan terkendali. *Learning rate* yang terlalu tinggi dapat menyebabkan model tidak konvergen, sedangkan *learning rate* yang terlalu rendah membuat proses pelatihan berlangsung sangat lambat.

4. *Margin untuk triplet loss*

Margin adalah batas minimum yang diharapkan antara jarak *embedding* wajah yang mirip dan yang tidak mirip. Nilai ini digunakan dalam fungsi *loss* yang disebut Triplet Loss, yaitu fungsi yang mendorong model agar wajah dari orang yang sama memiliki representasi (*embedding*) yang dekat, dan wajah dari orang yang berbeda memiliki representasi yang jauh. Nilai 1.0 dipilih sebagai *margin* karena sudah terbukti cukup efektif dalam berbagai penelitian face recognition sebelumnya, termasuk dalam arsitektur *FaceNet*. *Margin* ini mendorong perbedaan *embedding* wajah yang tidak sama agar setidaknya berjarak 1 satuan dari wajah yang sama. Jika *margin* terlalu kecil, model kesulitan membedakan wajah yang mirip. Jika terlalu besar, model menjadi sulit untuk mencapai target dan tidak konvergen.

5. *Early stopping (Patience)*

Early stopping merupakan teknik untuk menghentikan proses pelatihan secara otomatis jika tidak terjadi peningkatan performa pada data validasi dalam jumlah iterasi tertentu. Dalam penelitian ini, digunakan nilai *patience* = 5, artinya pelatihan akan dihentikan jika tidak terdapat peningkatan akurasi atau penurunan *loss* selama lima epoch berturut-turut. Strategi ini digunakan untuk mencegah *overfitting* dan menghemat waktu pelatihan. *Early stopping* menjaga efisiensi dan mencegah model menjadi terlalu “ngotot” belajar dari data yang akhirnya malah menurunkan performa.

6. *Triplet loss implemetation*

Triplet Loss menggunakan *framework PyTorch*. Fungsi ini bekerja dengan tiga input utama: *anchor*, *positive*, dan *negative*. *Anchor* dan *positive* merupakan gambar wajah dari individu yang sama, sedangkan *negative* berasal dari individu yang berbeda. Tujuan utama dari *Triplet Loss* adalah meminimalkan jarak antara *anchor* dan *positive* ($D(a, p)$) sekaligus memaksimalkan jarak antara *anchor* dan *negative* ($D(a, n)$) hingga selisihnya melebihi margin tertentu. Dengan cara ini, model belajar membedakan identitas wajah dengan representasi *embedding* yang optimal.

7. *Proses training*

Proses pelatihan melibatkan beberapa *epoch* dengan data dibagi menjadi *train* dan *validation set*. Pelatihan menggunakan *Triplet Loss* untuk memperbarui bobot model, fokus pada perbedaan *embedding* antar identitas. Model dioptimasi dengan *optimizer Adam*, yang adaptif dan efisien. Performanya terus dievaluasi melalui *loss* (*Triplet Loss*) dan metrik lain pada *validation set*. Pelatihan berlanjut hingga model konvergen atau terjadi *early stopping* (saat performa *validation* memburuk, menandakan *overfitting*).

Hasil pelatihan disimpan sebagai model *fine-tuning* dalam format *.pth*, yang merupakan format standar *PyTorch* untuk menyimpan *state dictionary* (arsitektur dan bobot model yang teroptimasi). Model *.pth* ini kemudian digunakan untuk *inferensi*, yaitu menghasilkan vektor *embedding* dari citra *Test-set*. *Embedding* ini adalah representasi numerik fitur wajah; wajah serupa memiliki *embedding* yang berdekatan.

Embedding dari *Test-set* disimpan dalam dua format:

1. *.pkl (Python Pickle)*: Ideal untuk menyimpan objek *Python* (termasuk array *NumPy*) secara efisien, cocok untuk pemuatan cepat ke memori dan perhitungan metrik kinerja (FAR, FNR, ROC).
2. *.json (JavaScript Object Notation)*: Format teks yang mudah dibaca dan dipertukarkan, berguna untuk pencatatan, *debugging*, atau integrasi lintas sistem, meskipun kurang efisien untuk operasi numerik langsung.

3.4.4 Evaluasi

Evaluasi dilakukan setelah proses pelatihan (*training*) model selesai, dengan menggunakan data test-set yang sebelumnya telah dipisahkan dari data pelatihan. Evaluasi ini bertujuan untuk mengukur performa dan keakuratan model pengenalan wajah (*Face Recognition*) setelah proses pembelajaran selesai, serta untuk memastikan bahwa model dapat mengenali wajah dengan presisi tinggi, baik pada data yang sudah dikenal (*registered*) maupun wajah yang tidak dikenal (*unknown*). Tujuannya adalah memastikan bahwa model dapat mengenali wajah dengan benar dan menghindari kesalahan seperti menerima wajah yang salah (*False Acceptance Rate* - FAR) atau menolak wajah yang benar (*False Rejection Rate* - FRR). Metode evaluasi dilakukan dengan pendekatan berbasis *embedding* dan berbasis performa. Evaluasi ini mencakup beberapa metrik sebagai berikut:

A. Evaluasi Jarak *Embedding*

Pada metode ini, evaluasi dilakukan dengan menganalisis kemiripan antar *embedding* wajah. Model *FaceNet* menghasilkan vektor representasi (*embedding*) berdimensi tinggi dari citra wajah. Kemiripan antara dua *embedding* digunakan untuk menentukan apakah dua gambar wajah berasal dari individu yang sama atau tidak. Metode kemiripan yang digunakan dalam penelitian ini adalah *Cosine Similarity*, Semakin besar nilai *Cosine Similarity*, maka semakin mirip wajah tersebut menurut representasi dari model.

B. Evaluasi performa

Setelah *embedding* wajah diekstraksi menggunakan model *FaceNet* dan dilakukan penentuan threshold optimal berdasarkan cosine similarity, langkah selanjutnya adalah melakukan evaluasi performa terhadap data uji (test set). Tujuannya adalah untuk mengukur seberapa baik model dalam mengperformakan identitas wajah dengan benar. Evaluasi performa dilakukan dengan membandingkan prediksi nama dari sistem dengan label sebenarnya yang diketahui dari dataset. Beberapa metrik evaluasi yang digunakan *Accuracy*, *Precision*, *Recall*, *F1-Score*, FAR (*False Acceptance Rate*), FRR (*False Rejection Rate*), ROC-AUC.

V. KESIMPULAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah saya lakukan, berikut adalah kesimpulan dari penelitian

1. Model *face recognition* berhasil dikembangkan menggunakan teknik *partial fine-tuning* dengan melatih layer *block8* dan *last_linear* menggunakan *Framework Pytorch* dan berhasil mengenali wajah asisten laboratorium teknik komputer.
2. Perbandingan performa menunjukkan bahwa model FaceNet *fine-tuning* menghasilkan akurasi dan ketahanan yang lebih baik dengan performa 99.66% pada data testing, sedangkan model FaceNet *pre-trained* hanya menghasilkan performa sebesar 98.99% pada data test. Keunggulan *fine-tuning* dibuktikan dengan testing *real case* dimana model *fine-tuning* berhasil mengenali semua wajah 100%, dan model *pre-trained* hanya 86,67%. Hasil ini menunjukan proses *fine-tuning* mampu membuat model lebih paham dengan wajah dataset Laboratorium Teknik Komputer Universitas Lampung.

5.2 Saran

Saran yang dapat saya sampaikan dalam penelitian dikemudian hari adalah

1. Pengumpulan dataset sebaiknya menggunakan perangkat kamera yang konsisten agar citra wajah memiliki pencahayaan dan resolusi seragam. Variasi kualitas gambar dapat menimbulkan masalah seperti duplikasi, data tidak sesuai ketentuan, atau format file tidak konsisten, yang pada akhirnya memengaruhi akurasi model. Oleh karena itu, proses pengambilan data perlu direncanakan dan dikontrol dengan baik.

DAFTAR PUSTAKA

- [1] D. T. Pham And P. T. N. Pham, “Artificial Intelligence In Engineering”.
- [2] A. Katsumata, “Deep Learning And Artificial Intelligence In Dental Diagnostic Imaging,” *Jpn. Dent. Sci. Rev.*, Vol. 59, Pp. 329–333, Dec. 2023, Doi: 10.1016/J.Jdsr.2023.09.004.
- [3] V. The And Y. P. Santosa, “Implementation Of Face Recognition Attendance System For Pt. Sumber Kurnia Alam With Mtcnn And Facenet Algorithm,” *Proxies J. Inform.*, Vol. 6, No. 2, Pp. 184–191, Aug. 2024, Doi: 10.24167/Proxies.V6i2.12461.
- [4] M. Lutz, *Learning Python*, 4th Ed. Beijing ; Sebastopol: O’reilly, 2009.
- [5] S. J. R. Peter Norvig, *Artificial Intelligence A Modern Approach*. New Jersey: Upper Saddle River, 2010.
- [6] A. M. Turing, *Computing Machinery And Intelligence*. 1950.
- [7] I. Cholissodin, “Buku Ajar Ai, Machine Learning & Deep Learning,” *Mach. Learn.*, 2020.
- [8] Z. A. Fikriya, M. I. Irawan, And S. Soetrisno., “Implementasi Extreme Learning Machine Untuk Pengenalan Objek Citra Digital,” *J. Sains Dan Seni Its*, Vol. 6, No. 1, Pp. 12–17, Mar. 2017, Doi: 10.12962/J23373520.V6i1.21754.
- [9] S. Abut, H. Okut, And K. J. Kallail, “Paradigm Shift From Artificial Neural Networks (Anns) To Deep Convolutional Neural Networks (Dcnns) In The Field Of Medical Image Processing,” *Expert Syst. Appl.*, Vol. 244, P. 122983, Jun. 2024, Doi: 10.1016/J.Eswa.2023.122983.
- [10] Universitas Muhammadiyah Sidoarjo And R. Dijaya, *Buku Ajar Pengolahan Citra Digital*. Umsida Press, 2023. Doi: 10.21070/2023/978-623-464-075-5.
- [11] R. C. Gonzalez And R. E. Woods, *Digital Image Processing*, Fourth Edition, Global Edition. New York: Pearson, 2017.
- [12] R. C. Gonzalez And R. E. Woods, *Digital Image Processing*. New York, Ny: Pearson, 2018.
- [13] J. Singh And R. Banerjee, “A Study On Single And Multi-Layer Perceptron Neural Network,” In *2019 3rd International Conference On Computing Methodologies And Communication (Iccmc)*, Erode, India: Ieee, Mar. 2019, Pp. 35–40. Doi: 10.1109/Iccmc.2019.8819775.
- [14] H. Kinsley And D. Kukiela, “Neural Networks From Scratch In Python”.
- [15] H. Kinsley And D. Kukiela, “Neural Networks From Scratch In Python”.
- [16] K. S. Tan, C. K. Lam, W. C. Tan, H. S. Ooi, And Z. H. Lim, “A Review Of Image Processing And Quantification Analysis For Solid Oxide Fuel Cell,” *Energy Ai*, Vol. 16, P. 100354, May 2024, Doi: 10.1016/J.Egyai.2024.100354.

- [17] G. Guo And N. Zhang, “A Survey On Deep Learning Based Face Recognition,” *Comput. Vis. Image Underst.*, Vol. 189, P. 102805, Dec. 2019, Doi: 10.1016/J.Cviu.2019.102805.
- [18] M. Wang And W. Deng, “Deep Face Recognition: A Survey,” *Neurocomputing*, Vol. 429, Pp. 215–244, Mar. 2021, Doi: 10.1016/J.Neucom.2020.10.081.
- [19] “Opencv (Open Source Computer Vision Library).” [Online]. Available: <https://opencv.org/about/>
- [20] L. C. Ngugi, M. Abelwahab, And M. Abo-Zahhad, “Recent Advances In Image Processing Techniques For Automated Leaf Pest And Disease Recognition – A Review,” *Inf. Process. Agric.*, Vol. 8, No. 1, Pp. 27–51, Mar. 2021, Doi: 10.1016/J.Inpa.2020.04.004.
- [21] S. Afroze, Md. R. Hossain, And M. M. Hoque, “Deepfocus: A Visual Focus Of Attention Detection Framework Using Deep Learning In Multi-Object Scenarios,” *J. King Saud Univ. - Comput. Inf. Sci.*, Vol. 34, No. 10, Pp. 10109–10124, Nov. 2022, Doi: 10.1016/J.Jksuci.2022.10.009.
- [22] A. K. Sharadhi, V. Gururaj, S. P. Shankar, M. S. Supriya, And N. S. Chogule, “Face Mask Recogniser Using Image Processing And Computer Vision Approach,” *Glob. Transit. Proc.*, Vol. 3, No. 1, Pp. 67–73, Jun. 2022, Doi: 10.1016/J.Gltp.2022.04.016.
- [23] Rosa Gradilla, “Multi-Task Cascaded Convolutional Networks (Mtcnn) For Face Detection And Facial Landmark Alignment,” *Medium*, Jul. 28, 2020.
- [24] S. Mahato And S. Neethirajan, “Integrating Artificial Intelligence In Dairy Farm Management – Biometric Facial Recognition For Cows,” *Inf. Process. Agric.*, P. S2214317324000696, Oct. 2024, Doi: 10.1016/J.Inpa.2024.10.001.
- [25] F. Schroff, D. Kalenichenko, And J. Philbin, “Facenet: A Unified Embedding For Face Recognition And Clustering,” In *2015 Ieee Conference On Computer Vision And Pattern Recognition (Cvpr)*, Boston, Ma, Usa: Ieee, Jun. 2015, Pp. 815–823. Doi: 10.1109/Cvpr.2015.7298682.
- [26] S. Lumbansiantar, S. Dwiasnati, And N. S. Fatonah, “Penerapan Metode Cosine Similarity Dalam Mendeteksi Plagiarisme Pada Jurnal,” *Format J. Ilm. Tek. Inform.*, Vol. 12, No. 2, P. 142, Jul. 2023, Doi: 10.22441/Format.2023.V12.I2.007.
- [27] S. Caria Ningsih, Se., Msi., Ph.D M. Pd. And Dr. F. G. Dra. Andi Reni Syamsuddin, M.Si., Ph.D. S. P. M. Si., *Panduan Praktis Untuk Analisis Data*. Pt Media Penerbit Indonesia, 2024.
- [28] J. Bian, Y. Peng, L. Wang, Y. Huang, And J. Xu, “A Survey On Parameter-Efficient Fine-Tuning For Foundation Models In Federated Learning,” Apr. 29, 2025, *Arxiv*: Arxiv:2504.21099. Doi: 10.48550/Arxiv.2504.21099.
- [29] L. Colantonio, L. Equeter, H. Giovannelli, P. Dehombreux, S. Mahmoudi, And F. Ducobu, “Image Processing With Deep-Learning And Transfer Learning For Cutting Tool Degradation Monitoring,” *Procedia Cirp*, Vol. 132, Pp. 50–55, 2025, Doi: 10.1016/J.Procir.2025.01.009.
- [30] A. D. Snowberger And K. You, “Leveraging Github Classroom And Google Colab For Short-Term Student Machine Learning Team Projects,” *J. Pract. Eng. Educ.*, Vol. 17, No. 1, Pp. 49–61, Feb. 2025, Doi: 10.14702/Jpee.2025.049.

- [31] A. Haque *Et Al.*, “Sok: Exploring Hallucinations And Security Risks In Ai-Assisted Software Development With Insights For Llm Deployment,” Jan. 31, 2025, *Arxiv*: Arxiv:2502.18468. Doi: 10.48550/Arxiv.2502.18468.
- [32] H. Cipta And M. Kom, “Pengantar Git Dan Github Dasar”.
- [33] “Buku Algoritma Dan Pemrograman.”
- [34] F. Cahyono, “Pengenaln Wajah Menggunakan Model Facenet Untuk Presensi Pegawai”.
- [35] H. Younis, M. A. Arshed, F. U. Hassan, M. Khurshid, And H. Ghassan, “Tomato Disease Classification Using Fine-Tuned Convolutional Neural Network,” *Int. J. Innov. Sci. Technol.*, Vol. 4, No. 1, Pp. 123–134, Feb. 2022, Doi: 10.33411/Ijst/2022040109.
- [36] R. Goel, I. Mehmood, And H. Ugail, “A Study Of Deep Learning-Based Face Recognition Models For Sibling Identification,” *Sensors*, Vol. 21, No. 15, P. 5068, Jul. 2021, Doi: 10.3390/S21155068.
- [37] J. A. Mensah, J. K. Appati, E. K. A. Boateng, E. Ocran, And L. Asiedu, “Facenet Recognition Algorithm Subject To Multiple Constraints: Assessment Of The Performance,” *Sci. Afr.*, Vol. 23, P. E02007, Mar. 2024, Doi: 10.1016/J.Sciaf.2023.E02007.
- [38] J. K. Essel, J. A. Mensah, E. Ocran, And L. Asiedu, “On The Search For Efficient Face Recognition Algorithm Subject To Multiple Environmental Constraints,” *Heliyon*, Vol. 10, No. 7, P. E28568, Apr. 2024, Doi: 10.1016/J.Heliyon.2024.E28568.
- [39] L. Yang, X. Xu, J. Zhao, And H. Song, “Fusion Of Retinaface And Improved Facenet For Individual Cow Identification In Natural Scenes,” *Inf. Process. Agric.*, Vol. 11, No. 4, Pp. 512–523, Dec. 2024, Doi: 10.1016/J.Inpa.2023.09.001.
- [40] K. S. Zebua, I. H. Kartowisastro, And G. Putra, “Low Resolution Face Recognition Using Combination Of Gpen Super Resolution And Facenet,” . *Vol.*, No. 12, 2023.
- [41] T.-V. Dang And H.-L. Tran, “A Secured, Multilevel Face Recognition Based On Head Pose Estimation, Mtcnn And Facenet,” *J. Robot. Control Jrc*, Vol. 4, No. 4, Pp. 431–437, Jun. 2023, Doi: 10.18196/Jrc.V4i4.18780.
- [42] Christoph Busch, *Iso/Iec 29794-5:2024 — Biometric Sample Quality*. European Association For Biometrics, 2024.