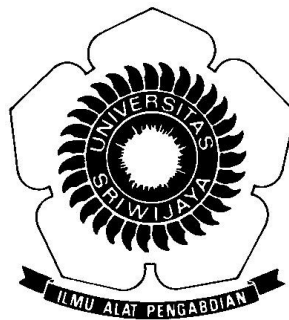


ADDITIVE ANGULAR MARGIN LOSS UNTUK PENGENALAN CITRA
WAJAH BERMASKER

Diajukan Sebagai Syarat Untuk Menyelesaikan
Pendidikan Program Strata-1 Pada
Jurusan Teknik Informatika



Oleh:

Ferza Reyaldi
NIM: 09021281924060

Jurusan Teknik Informatika
FAKULTAS ILMU KOMPUTER UNIVERSITAS SRIWIJAYA
Tahun 2022

LEMBAR PENGESAHAN SKRIPSI

***ADDITIVE ANGULAR MARGIN LOSS* UNTUK PENGENALAN
CITRA WAJAH BERMASKER**

Oleh:

Ferza Reyaldi

NIM: 09021281924060

Indralaya, 26 Desember 2022

Pembimbing I

Pembimbing II

Dr. Muhammad Fachrurrozi, M.T.
NIP 198005222008121002

Muhammad Qurhanul Rizqie, M.T., Ph.D
NIP 198712032022031006

Mengetahui,

Ketua Jurusan Teknik Informatika

Alvi Syahrini Utami, M.Kom.
NIP 197812222006042003

TANDA LULUS UJIAN KOMPREHENSIF SKRIPSI

Pada hari Rabu tanggal 4 Januari 2023 telah dilaksanakan telah dilaksanakan ujian komprehensif skripsi oleh Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Sriwijaya.

Nama : Ferza Reyaldi
NIM : 09021281924060
Judul : *Additive Angular Margin Loss* untuk Pengenalan Citra Wajah Bermasker

dan dinyatakan **LULUS**.

1. Ketua Penguji

Dian Palupi Rini, M.Kom., Ph.D.
NIP 197802232006042002

2. Penguji

Dr. Abdiansah, S.Kom., M.CS.
NIP 198410012009121005

3. Pembimbing I

Dr. Muhammad Fachrurrozi, M.T.
NIP 198005222008121002

4. Pembimbing II

Muhammad Qurhanul Rizqie, M.T., Ph.D
NIP 198712032022031006

Mengetahui,
Ketua Jurusan Teknik Informatika

Alvi Syahrini Utami, M.Kom.
NIP 197812222006042003

HALAMAN PERNYATAAN BEBAS PLAGIAT

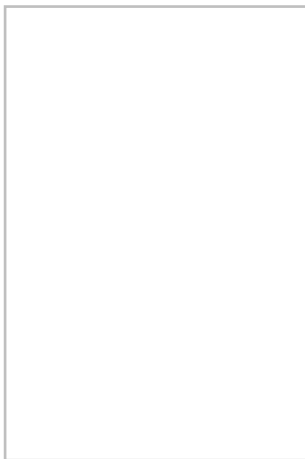
Yang bertanda tangan di bawah ini:

Nama : Ferza Reyaldi
NIM : 09021281924060
Program Studi : Teknik Informatika Reguler
Judul Skripsi : *Additive Angular Margin Loss* untuk Pengenalan Citra Wajah Bermasker

Hasil Pengecekan *iThenticate*/Turnitin: 7 %

Menyatakan bahwa laporan proyek saya merupakan hasil karya sendiri dan bukan hasil penjiplakan/plagiat. Apabila ditemukan unsur penjiplakan/plagiat dalam laporan proyek ini, maka saya bersedia menerima sanksi akademik dari Universitas Sriwijaya sesuai dengan ketentuan yang berlaku.

Demikian, pernyataan ini saya buat dengan sebenarnya dan tidak ada paksaan oleh siapa pun.



Indralaya, 26 Desember 2022

Ferza Reyaldi

NIM 09021281924060

MOTTO DAN PERSEMBAHAN

"You Only Live Once. But if you do it right, once is enough."

- Mae West

"Every journey has its final day. Don't rush."

- Zhongli

Kupersembahkan Karya Tulis ini kepada:

- Allah SWT
- Orang Tua dan Saudara-Saudaraku
- My Besto Furendo

ADDITIVE ANGULAR MARGIN LOSS FOR MASKED-FACE IMAGE RECOGNITION

FERZA REYALDI
09021281924060

ABSTRACT

The use of face masks imposed during the Covid-19 Pandemic caused partial occlusion of facial features. This resulted in a person's identity being difficult to recognize by the recognition system. This research will perform masked face recognition using Convolutional Neural Network (CNN) and additive angular margin loss. Additive angular margin loss is used to obtain highly discriminatory features for face recognition. The dataset used in this study is the Real-World Masked Face Dataset (RMFD). The use of additive angular margin loss shows an increase in the accuracy of the facial recognition model. The results of this study show that the training data accuracy is 36.33% greater and the validation data accuracy is 12.65% greater than the CNN model without using additive angular margin loss.

Keywords: *Masked Face Recognition, Convolutional Neural Network, Additive Angular Margin Loss, Real-World Masked Face Dataset*

ADDITIVE ANGULAR MARGIN LOSS UNTUK PENGENALAN CITRA WAJAH BERMASKER

FERZA REYALDI
09021281924060

ABSTRAK

Pemakaian masker wajah yang diberlakukan saat Pandemi Covid-19 menyebabkan oklusi parsial pada fitur-fitur wajah. Hal ini berakibat identitas seseorang sulit dikenali oleh sistem pengenalan. Penelitian ini akan melakukan pengenalan wajah bermasker menggunakan *Convolutional Neural Network* (CNN) dan *additive angular margin loss*. *Additive angular margin loss* digunakan untuk mendapatkan fitur yang sangat diskriminatif untuk pengenalan wajah. Dataset yang digunakan pada penelitian ini adalah *Real-World Masked Face Dataset* (RMFD). Penggunaan *additive angular margin loss* menunjukkan peningkatan akurasi terhadap model pengenalan wajah. Hasil penelitian ini mendapat nilai akurasi data latih lebih besar 36,33% dan akurasi data validasi lebih besar 12,65% dibandingkan model CNN tanpa menggunakan *additive angular margin loss*.

Kata Kunci: Pengenalan Wajah Bermasker, *Convolutional Neural Network*, *Additive Angular Margin Loss*, *Real-World Masked Face Dataset*

KATA PENGANTAR

Segala pujian dan syukur kepada Allah SWT atas karunia-Nya yang telah diberikan kepada penyusun sehingga dapat menyelesaikan penelitian skripsi berjudul “*Additive Angular Margin Loss* untuk Pengenalan Citra Wajah Bermasker” dengan baik.

Penyusun juga mengucapkan banyak terima kasih kepada pihak-pihak yang telah memberi dukungan dan membantu dalam penyelesaian skripsi ini, yaitu:

1. Orang tua dan saudara-saudara saya yang telah memberikan dukungan dan doa kepada saya.
2. Bapak Jaidan Jauhari, S.Pd., M.T. selaku Dekan Fakultas Ilmu Komputer Universitas Sriwijaya.
3. Ibu Alvi Syahrini Utami, M.Kom., selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Sriwijaya.
4. Bapak Dr. Muhammad Fachrurrozi, M.T., selaku Dosen Pembimbing I.
5. Bapak Muhammad Qurhanul Rizqie, M.T., Ph.D., selaku Dosen Pembimbing II sekaligus Dosen Pembimbing Akademik.
6. Bapak Alm. Drs. Megah Mulya, M.T., selaku Dosen Pembimbing Akademik dari Semester 1 sampai 3.
7. Seluruh Bapak dan Ibu Dosen Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Sriwijaya.
8. Seluruh Tata Usaha Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Sriwijaya.

9. Teman-teman Teknik Informatika Reguler A 2019 yang telah menemani dan membantu selama perkuliahan.
10. Teman-teman Dinas Akademik BEM Fasilkom UNSRI 2021.
11. Serta semua pihak yang telah membantu Penyusun selama penyusunan skripsi yang tidak dapat disebutkan satu-persatu.

Penyusun menyadari bahwa dalam penyusunan skripsi masih terdapat banyak kesalahan dan kekurangan yang disebabkan oleh keterbatasan pengetahuan dan pengalaman. Oleh karena itu, kritik dan saran yang bersifat membangun sangat diperlukan untuk kemajuan penelitian di masa mendatang. Semoga penelitian ini dapat memberikan manfaat bagi kita semua.

Indralaya, 26 Desember 2022

Penyusun,

Ferza Reyaldi
NIM 09021281924060

DAFTAR ISI

Halaman

| | |
|--|----------|
| HALAMAN JUDUL | i |
| LEMBAR PENGESAHAN SKRIPSI | ii |
| TANDA LULUS UJIAN KOMPREHENSIF SKRIPSI | iii |
| HALAMAN PERNYATAAN BEBAS PLAGIAT | iv |
| MOTTO DAN PERSEMBAHAN | v |
| <i>ABSTRACT</i> | vi |
| ABSTRAK..... | vii |
| KATA PENGANTAR | viii |
| DAFTAR ISI..... | x |
| DAFTAR TABEL..... | xiii |
| DAFTAR GAMBAR..... | xiv |
| DAFTAR LAMPIRAN..... | xv |
| BAB I PENDAHULUAN..... | I-1 |
| 1.1 Pendahuluan | I-1 |
| 1.2 Latar Belakang Masalah | I-1 |
| 1.3 Rumusan Masalah | I-3 |
| 1.4 Tujuan Penelitian | I-3 |
| 1.5 Manfaat Penelitian | I-4 |
| 1.6 Batasan Masalah | I-4 |
| 1.7 Sistematika Penulisan | I-5 |
| 1.8 Kesimpulan..... | I-6 |
| BAB II KAJIAN LITERATUR..... | II-1 |
| 2.1 Pendahuluan | II-1 |
| 2.2 Landasan Teori | II-1 |
| 2.2.1 Citra Digital | II-1 |
| 2.2.2 Pengenalan Wajah | II-2 |
| 2.2.3 Facial Landmark | II-3 |

| | |
|---|-----------|
| 2.2.4 Deep Learning | II-4 |
| 2.2.5 Convolutional Neural Network | II-4 |
| 2.2.7 TensorFlow | II-7 |
| 2.2.6 <i>Additive Angular Margin Loss</i> | II-8 |
| 2.2.8 Metrik Evaluasi | II-9 |
| 2.2.9 Rational Unified Process (RUP)..... | II-11 |
| 2.3 Penelitian Lain yang Relevan | II-12 |
| 2.4 Kesimpulan..... | II-14 |
| BAB III METODOLOGI PENELITIAN | III-1 |
| 3.1 Pendahuluan | III-1 |
| 3.2 Pengumpulan Data..... | III-1 |
| 3.2.1 Jenis Data | III-1 |
| 3.2.2 Sumber Data | III-1 |
| 3.2.3 Metode Pengumpulan Data | III-2 |
| 3.2.4 Sampel Data | III-2 |
| 3.3 Tahapan Penelitian | III-4 |
| 3.3.1 Akuisisi Data | III-5 |
| 3.3.2 Kerangka Kerja..... | III-5 |
| 3.3.3 Kriteria Pengujian..... | III-10 |
| 3.3.4 Format Data Pengujian | III-10 |
| 3.3.5 Alat yang Digunakan dalam Pengujian | III-11 |
| 3.3.6 Pengujian Penelitian | III-11 |
| 3.3.7 Analisis Hasil Pengujian | III-12 |
| 3.4 Metode Pengembangan Perangkat Lunak | III-12 |
| 3.4.1 Fase Insepsi | III-12 |
| 3.4.2 Fase Elaborasi..... | III-13 |
| 3.4.3 Fase Konstruksi | III-13 |
| 3.4.4 Fase Transisi..... | III-14 |
| 3.5 Kesimpulan..... | III-14 |
| BAB IV PENGEMBANGAN PERANGKAT LUNAK | IV-1 |
| 4.1 Pendahuluan | IV-1 |
| 4.2 Rational Unified Process | IV-1 |

| | |
|---|-------|
| 4.2.1 Fase Insepsi | IV-1 |
| 4.2.2 Fase Elaborasi..... | IV-7 |
| 4.2.3 Fase Kontruksi..... | IV-13 |
| 4.2.4 Fase Transisi..... | IV-16 |
| 4.3 Kesimpulan..... | IV-19 |
| BAB V HASIL DAN ANALISIS PENELITIAN | V-1 |
| 5.1 Pendahuluan | V-1 |
| 5.2 Data Hasil Penelitian | V-1 |
| 5.2.1 Konfigurasi Percobaan Parameter | V-1 |
| 5.2.2 Analisis Hasil Penelitian..... | V-2 |
| 5.4 Kesimpulan..... | V-7 |
| BAB VI KESIMPULAN DAN SARAN | VI-1 |
| 6.1 Pendahuluan | VI-1 |
| 6.2 Kesimpulan..... | VI-1 |
| 6.3 Saran | VI-2 |
| DAFTAR PUSTAKA | vii |
| LAMPIRAN..... | xxii |

DAFTAR TABEL

Halaman

| | |
|--|--------|
| Tabel III-1. Sampel Citra Wajah Bermasker..... | III-2 |
| Tabel III-2. Sampel Citra Wajah Tidak Bermasker | III-4 |
| Tabel III-3. Jumlah Data setelah Pra-Olah Data | III-7 |
| Tabel III-4. Pembagian Data | III-7 |
| Tabel III-5. Konfigurasi Parameter Augmentasi Data | III-8 |
| Tabel III-6. Tabel Evaluasi Model CNN dengan <i>Additive Angular Margin Loss</i> | III-11 |
| Tabel IV-1. Kebutuhan Fungsional Perangkat Lunak..... | IV-2 |
| Tabel IV-2. Kebutuhan Non-Fungsional Perangkat Lunak | IV-2 |
| Tabel IV-3. Definisi Aktor | IV-3 |
| Tabel IV-4. Definisi <i>Use Case</i> | IV-4 |
| Tabel IV-5. Skenario <i>Use Case</i> Masukan Gambar Wajah..... | IV-4 |
| Tabel IV-6. Skenario <i>Use Case</i> Mendeteksi dan Mengidentifikasi Citra Wajah | IV-6 |
| Tabel IV-7. Keterangan Implementasi Kelas | IV-14 |
| Tabel IV-8. Daftar File HTML | IV-15 |
| Tabel IV-9. Rencana Pengujian <i>Use Case</i> Masukan Gambar Wajah | IV-17 |
| Tabel IV-10. Rencana Pengujian <i>Use Case</i> Deteksi dan Identifikasi Wajah .. | IV-17 |
| Tabel IV-11. Pengujian <i>Use Case</i> Masukan Gambar/Video Wajah | IV-18 |
| Tabel IV-12. Pengujian <i>Use Case</i> Deteksi dan Identifikasi Wajah | IV-18 |
| Tabel V-1. Variasi Konfigurasi Parameter..... | V-1 |
| Tabel V-2. Konfigurasi Parameter Tetap | V-2 |
| Tabel V-3. Perbandingan Hasil <i>Training-Validation</i> Model | V-3 |
| Tabel V-4. Perbandingan Hasil <i>Testing</i> Model..... | V-6 |

DAFTAR GAMBAR

Halaman

| | |
|--|-------|
| Gambar II-1. Representasi Elemen Citra Digital | II-2 |
| Gambar II-2. Tengara Wajah | II-3 |
| Gambar II-3. Sampel wajah yang telah teranota tengara wajah..... | II-3 |
| Gambar II-4. Contoh proses kerja CNN | II-5 |
| Gambar II-5. Arsitektur Model ResNet-50 | II-6 |
| Gambar II-6. Arsitektur Model InceptionV3 | II-7 |
| Gambar II-7. Confusion matrix..... | II-9 |
| Gambar II-8. Fase-fase RUP | II-11 |
| Gambar III-1. Tahapan-Tahapan Penelitian..... | III-5 |
| Gambar III-2. Diagram Alir Klasifikasi Citra Wajah Bermasker Menggunakan CNN dengan Teknik Additive Angular Margin Loss | III-6 |
| Gambar III-3. Arsitektur Model ResNet-50 + ArcFace | III-9 |
| Gambar III-4. Arsitektur Model InceptionV3 + ArcFace | III-9 |
| Gambar IV-1. <i>Use Case Diagram</i> Perangkat Lunak | IV-3 |
| Gambar IV-2. Rancangan Tampilan Halaman Depan | IV-8 |
| Gambar IV-3. Rancangan Tampilan Halaman Pengenalan Wajah | IV-8 |
| Gambar IV-4. Rancangan Tampilan Halaman Pengenalan Wajah setelah Ditekan Tombol <i>Predict</i> | IV-9 |
| Gambar IV-5. <i>Activity Diagram</i> Masukan Gambar Wajah..... | IV-10 |
| Gambar IV-6. <i>Activity Diagram</i> Deteksi dan Identifikasi Citra Wajah | IV-11 |
| Gambar IV-7. <i>Sequence Diagram</i> Masukan Gambar Wajah..... | IV-12 |
| Gambar IV-8. <i>Sequence Diagram</i> Deteksi dan Identifikasi Citra Wajah | IV-12 |
| Gambar IV-9. Diagram Kelas | IV-13 |
| Gambar IV-10. Antarmuka Halaman Depan | IV-15 |
| Gambar IV-11. Antarmuka Halaman Pengenalan Wajah | IV-16 |
| Gambar IV-12. Antarmuka Halaman Pengenalan Wajah setelah Ditekan Tombol <i>Predict</i> | IV-16 |
| Gambar V-1. Grafik Akurasi dan <i>loss</i> Model 1 | V-3 |
| Gambar V-2. Grafik Akurasi dan <i>loss</i> Model 2 | V-4 |
| Gambar V-3. Grafik Akurasi dan <i>loss</i> Model 3 | V-4 |
| Gambar V-4. Grafik Akurasi dan <i>loss</i> Model 4 | V-4 |
| Gambar V-5. Diagram Batang Perbandingan Akurasi Data Uji | V-6 |

DAFTAR LAMPIRAN

| | |
|-------------------------------|------|
| Lampiran 1. Kode Program..... | xxii |
|-------------------------------|------|

BAB I

PENDAHULUAN

1.1 Pendahuluan

Bab I membahas latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, sistematika penulisan dan kesimpulan berdasarkan penelitian yang diajukan secara rinci.

1.2 Latar Belakang Masalah

Pengenalan wajah adalah salah satu solusi pengenalan biometrik yang lebih disukai karena tidak memerlukan kontak langsung dengan pengguna dan tingkat akurasi yang dicapai tinggi. Namun, wajib pemakaian masker wajah mulai diberlakukan di tempat-tempat umum saat Pandemi Covid-19 yang bertujuan untuk menjaga agar pandemi tetap terkendali (Boutros et al., 2021). Hal tersebut menyebabkan oklusi parsial pada wajah dikarenakan pemakaian aksesoris pakaian berupa masker tidak dapat dihindari (Montero et al., 2021). Masalah khusus ini menjadi tantangan utama di bidang pengenalan wajah dikarenakan fitur-fitur wajah yang tersedia berkurang (Hariri, 2022). Hal ini dikarenakan pengenalan wajah memainkan peran penting dalam kehidupan sehari-hari seperti pemeriksaan paspor, ATM, kartu kredit, verifikasi pemilih (*voter verification*), pintu pintar, investigasi kriminal atau teroris dan banyak tujuan lainnya (Ejaz et al., 2019).

Metode-metode seperti *Principal Component Analysis* (Ejaz et al., 2019) dan *Convolutional Neural Network* (Mandal et al., 2021) diusulkan oleh peneliti untuk melakukan ekstraksi fitur pada citra wajah.

Principal Component Analysis (PCA) adalah prosedur statistik yang juga didefinisikan sebagai transformasi linier ortogonal. Algoritma ini menekankan variasi dan memunculkan pola yang kuat dalam kumpulan data. Ini digunakan untuk meminimalkan dataset besar ke dataset kecil masih berisi hampir semua informasi sebagai dataset besar. Namun, pengenalan wajah memiliki tingkat pengenalan yang buruk dikarenakan fitur dari wajah bermasker lebih sedikit menyebabkan turunnya tingkat pengenalan (Ejaz et al., 2019).

Convolutional Neural Network (CNN) merupakan arsitektur *deep learning* yang tersusun atas banyak *layer*, seperti *input layer*, *convolution layer*, *pooling layer*, dan *fully connected layer* (Xiong et al., 2017). Arsitektur CNN cocok digunakan untuk masalah klasifikasi karena CNN dapat melakukan ekstraksi dan mempelajari fitur - fitur pada data secara otomatis (Chandra et al., 2017). Namun arsitektur CNN membutuhkan daya komputasi yang cukup tinggi dikarenakan penggunaan filter berupa matriks 2 dimensi dalam ekstraksi fitur pada data (Nurmaini et al., 2020). Penelitian pengenalan wajah bermasker pernah dilakukan menggunakan model pra-latih CNN ResNet-50. Hasil penelitian menunjukkan bahwa tingkat akurasi dari model pengenalan wajah bermasker adalah 47,9%, jauh lebih rendah dibandingkan tingkat akurasi dari model pengenalan wajah tanpa masker yang mencapai 89,7% (Mandal et al., 2021).

Additive Angular Margin Loss adalah *loss function* yang memungkinkan untuk mendapatkan fitur yang sangat diskriminatif untuk pengenalan wajah. *Additive Angular Margin Loss* memiliki interpretasi geometris yang jelas karena korespondensi yang tepat dengan jarak geodesik pada *hypersphere* (Deng et al., 2018).

Pada penelitian ini, *Additive Angular Margin Loss* diimplementasikan pada arsitektur CNN yang akan dilatih untuk meningkatkan tingkat diskriminatif fitur-fitur pada citra wajah bermasker yang terbatas.

1.3 Rumusan Masalah

Rumusan masalah dari penelitian yang dilakukan adalah sebagai berikut:

1. Pengenalan citra wajah bermasker lebih sulit dibandingkan pengenalan wajah tanpa masker dikarenakan sebagian besar fitur-fitur wajah tertutup oleh masker.
2. Model identifikasi wajah bermasker pada model CNN dengan arsitektur ResNet-50 yang dilakukan (Mandal et al., 2021) memiliki akurasi yang rendah.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah, tujuan dari penelitian yang dilakukan adalah sebagai berikut:

1. Membangun model pengenalan citra wajah bermasker menggunakan algoritma CNN dan *Additive Angular Margin Loss*.

2. Mengukur dan mengetahui peningkatan akurasi model pengenalan citra wajah bermasker menggunakan algoritma CNN dan *Additive Angular Margin Loss*.

1.5 Manfaat Penelitian

Penelitian yang dijalankan diharapkan memberi manfaat sebagai berikut:

1. Penelitian ini dapat membantu dalam pengembangan model identifikasi citra wajah bermasker yang lebih baik.
2. Penelitian ini dapat menjadi rujukan dalam penelitian lebih lanjut mengenai pengenalan wajah bermasker menggunakan algoritma CNN.

1.6 Batasan Masalah

Untuk mencegah penelitian ini terlalu meluas dan tidak terarah, peneliti membatasi lingkup masalah pada penelitian ini dengan rincian sebagai berikut:

1. Dataset yang digunakan untuk melatih model CNN adalah *Real-World Masked Face Dataset* (RMFD).

1.7 Sistematika Penulisan

Sistematika penulisan skripsi ini adalah sebagai berikut:

BAB I. PENDAHULUAN

Bab ini membahas secara rinci tentang latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, sistematika penulisan, dan kesimpulan.

BAB II. KAJIAN LITERATUR

Bab ini membahas secara rinci mengenai penelitian – penelitian lain yang relevan dan landasan teori yang menjadi dasar dalam menyusun penelitian ini.

BAB III. METODOLOGI PENELITIAN

Bab ini membahas secara rinci mengenai kerangka kerja, instrumen penelitian, data yang digunakan dalam penelitian, dan perencanaan dari kegiatan – kegiatan penelitian.

BAB IV. PENGEMBANGAN PERANGKAT LUNAK

Bab ini membahas secara rinci mengenai proses pengembangan perangkat lunak yang sudah direncanakan pada BAB III, dan melakukan pengujian pada perangkat lunak yang digunakan untuk penelitian.

BAB V. HASIL DAN ANALISIS PENELITIAN

Bab ini membahas secara rinci mengenai hasil dari perangkat lunak yang digunakan pada penelitian dan melakukan analisa pada hasil tersebut.

BAB VI. KESIMPULAN DAN SARAN

Bab ini membahas secara rinci mengenai kesimpulan yang dapat ditarik dari hasil penelitian yang telah dilakukan serta saran – saran yang dapat digunakan untuk mengembangkan penelitian tersebut.

1.8 Kesimpulan

Bab I menjelaskan terkait latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, Batasan masalah dan sistematika penulisan. Berdasarkan penjelasan diatas, penelitian dalam pengenalan citra wajah bermasker menggunakan CNN dengan teknik *Additive Angular Margin Loss* diharapkan memberikan hasil yang baik sesuai hipotesis.

BAB II

KAJIAN LITERATUR

2.1 Pendahuluan

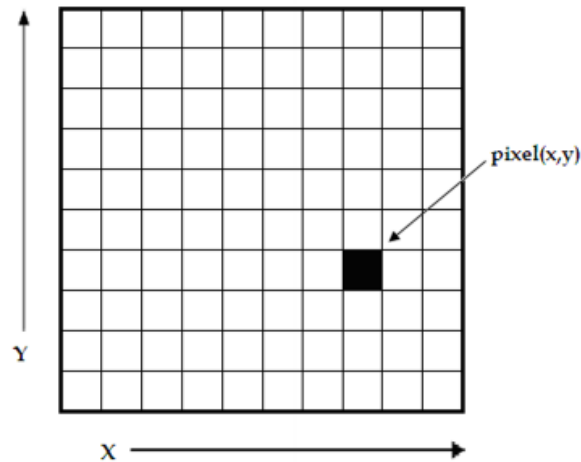
Bab ini berisi landasan teori yang berhubungan dengan masalah yang diteliti, penelitian sebelumnya yang berkaitan dengan penelitian yang diajukan dalam beberapa tahun terakhir, dan hipotesis yang diajukan.

2.2 Landasan Teori

Bab ini berisi mengenai dasar-dasar teori yang berhubungan dengan pengenalan citra wajah Bermasker menggunakan CNN dan pendekatan *Additive Angular Margin Loss*. Bab ini juga menjelaskan penelitian lain mengenai metode yang pernah digunakan dalam beberapa tahun terakhir serta membahas beberapa dasar teori yang mendukung penelitian ini.

2.2.1 Citra Digital

Citra adalah representasi visual dari suatu objek. Citra digital adalah proyeksi citra tiga dimensi ke dalam bidang dua dimensi. Citra digital direpresentasikan sebagai fungsi dua dimensi $f(x,y)$, dimana x dan y mewakili lokasi elemen citra atau piksel (*pixel*) dan memiliki intensitas nilai yang diskrit. Secara matematis, citra digital digambarkan sebagai representasi matriks dari citra dua dimensi menggunakan sejumlah elemen sel titik yang terbatas yang disebut piksel (Tyagi, 2018).



Gambar II-1. Representasi Elemen Citra Digital

2.2.2 Pengenalan Wajah

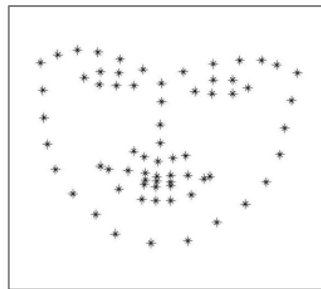
Pengenalan wajah (*face recognition*) merupakan subdivisi masalah dari pengenalan pola visual. Teknologi pengenalan wajah merupakan teknologi biometrik yang melakukan identifikasi seseorang berdasarkan fitur-fitur wajahnya (*facial features*) (Li et al., 2020). Teknologi ini banyak disukai dikarenakan pengenalan wajah tanpa memerlukan kontak langsung dengan pengguna dan memiliki tingkat akurasi tinggi (Boutros et al., 2021).

Sistem pengenalan wajah mencakup deteksi wajah (*face detection*), posisi wajah (*face position*), dan pengenalan identitas (*identity recognition*). Algoritma deteksi wajah adalah untuk mengetahui sistem koordinat semua wajah dalam satu citra. Algoritma ini melakukan pemindaian seluruh citra untuk menentukan apakah area kandidat adalah wajah. Keluaran dari deteksi wajah dapat berbentuk persegi, persegi panjang, dll. Posisi wajah adalah posisi koordinat fitur wajah dalam sistem koordinat deteksi wajah. Dibandingkan dengan deteksi wajah, waktu perhitungan algoritma penentuan posisi wajah jauh lebih singkat (Li et al., 2020).

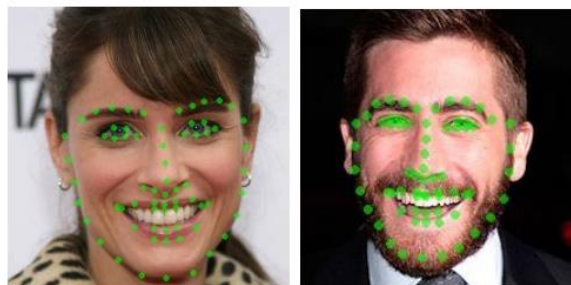
Pada penelitian ini, penulis menggunakan salah satu algoritma *deep learning* dalam membangun model klasifikasi citra wajah bermasker dengan bantuan TensorFlow sebagai pustaka utama.

2.2.3 Facial Landmark

Dalam visi komputer, wajah manusia dikenali melalui tengara wajah (*facial landmark*). Tengara wajah mendefinisikan bentuk wajah. *Facial landmark detection* biasanya dilakukan melalui proses pelokalan titik kunci wajah *fiducial* (fidusia) (Wu & Ji, 2019). Titik kunci wajah tersebut antara lain sudut mata, ujung hidung, bibir, dll (Zhang et al., 2018).



Gambar II-2. Tengara Wajah



Gambar II-3. Sampel Wajah yang Telah Teranota Tengara Wajah

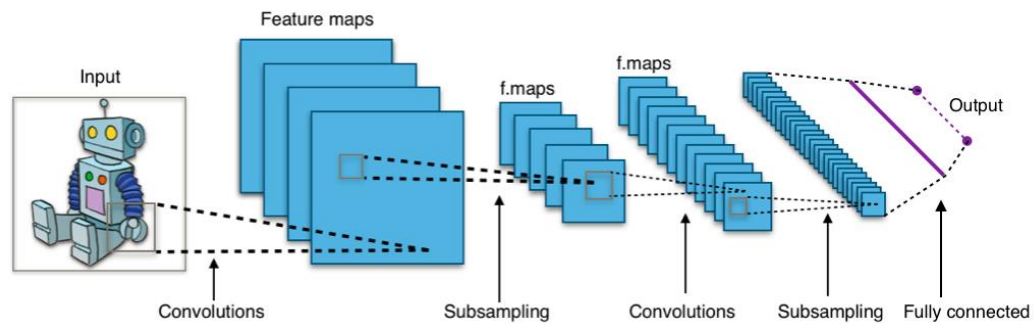
2.2.4 Deep Learning

Pembelajaran mendalam (*deep learning*) adalah cabang ilmu dari pembelajaran mesin (*machine learning*). *Deep learning* dapat mengetahui fitur-fitur data melalui pembelajaran untuk permasalahan klasifikasi secara otomatis tanpa perlu melalui tahap ekstraksi fitur secara spesifik (Li et al., 2020).

Deep learning telah menunjukkan tingkat akurasi superior pada sistem pengenalan wajah (Anwar & Raychowdhury, 2020), sehingga *deep learning* disebut algoritma *state-of-the-art* (termutakhir) dari sistem pengenalan wajah. *Deep learning* ini belajar untuk mengekstrak fitur-fitur penting dari citra wajah dan menanamkannya ke dalam vektor n-dimensi dengan jarak antar kelas yang kecil dan jarak antar kelas yang besar (Montero et al., 2021).

2.2.5 Convolutional Neural Network

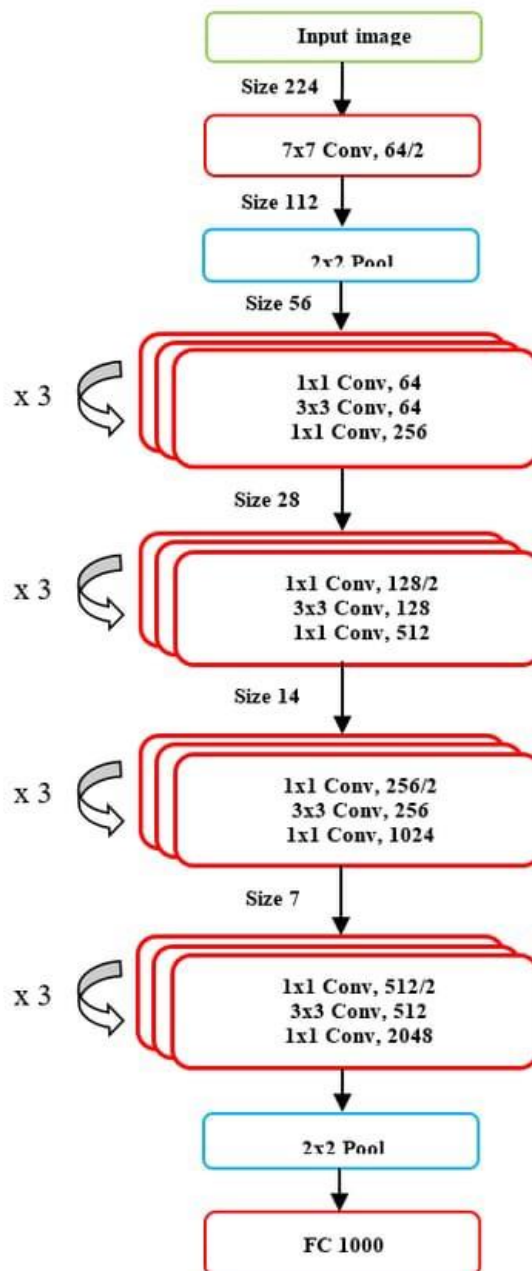
Convolutional Neural Network (CNN) adalah salah satu jenis jaringan saraf tiruan yang paling efektif. CNN telah banyak menunjukkan keunggulannya dalam berbagai aplikasi seperti klasifikasi gambar (*image classification*), pengenalan objek (*object recognition*), pengambilan objek (*object retrieval*), dan deteksi objek (*object detection*). CNN bisa terdiri atas banyak lapisan berjenjang (*cascaded layer*). Lapisan-lapisan berjenjang tersebut berfungsi untuk mengontrol tingkat pergeseran, skala, dan distorsi. Jenis lapisan-lapisan berjenjang tersebut antara lain *input layer*, *convolutional layer*, *subsampling layer*, *full-connected layer*, dan *output layer* (Alzu'bi et al., 2021). Berikut pada Gambar II-4 merupakan contoh proses kerja dari CNN.



Gambar II-4. Contoh proses kerja CNN

Pada penerapannya, terdapat banyak model pra-latih CNN yang dapat digunakan untuk menyelesaikan permasalahan klasifikasi. *AlexNet*, *AlexNetOWTbn*, *GooLeNet*, *Overfeat*, *VGG* merupakan sekian contoh arsitektur model CNN yang umum digunakan. Arsitektur-arsitektur tersebut menggunakan banyak lapisan konvolusi. Namun, hal tersebut menyebabkan muncul masalah baru, seperti sulitnya optimasi jaringan, masalah gradien menghilang, dan masalah degradasi (Mukti & Biswas, 2019).

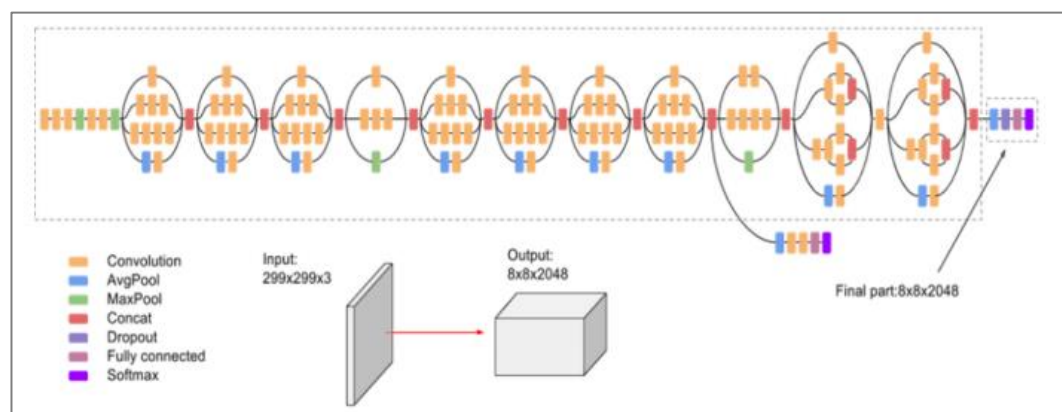
Ide baru yang dapat ditawarkan untuk permasalahan tersebut adalah *Residual Network* (ResNet). ResNet memiliki kelebihan untuk menyelesaikan tugas-tugas rumit dan juga meningkatkan akurasi deteksi. ResNet mencoba mengatasi kesulitan dalam proses pelatihan CNN yang dalam, kejenuhan dan penurunan akurasi. Dalam penelitian, penyusun menggunakan arsitektur ResNet-50. Sesuai dengan namanya, ResNet-50 menggunakan 50 lapisan residual (*residual layer*). Pada Gambar II-5 menunjukkan arsitektur model CNN ResNet-50 (Mukti & Biswas, 2019).



Gambar II-5. Arsitektur Model ResNet-50

Selain ResNet-50, Ide baru yang dapat ditawarkan adalah InceptionV3. Model InceptionV3 adalah model pra-latih yang memiliki performa superior dalam pengenalan objek jika dibandingkan dengan versi pendahulunya, GoogleNet

(InceptionV1). Model InceptionV3 mencakup tiga bagian: blok konvolusional dasar, modul Inception yang ditingkatkan, dan *classifier* (Lin et al., 2019). Arsitektur CNN model InceptionV3 terdiri dari 48 *layer* (Chowdary et al., 2020). Pada Gambar II-6 menunjukkan arsitektur model CNN InceptionV3 (Khan et al., 2020).



Gambar II-6. Arsitektur Model InceptionV3

2.2.7 TensorFlow

TensorFlow merupakan *open-source library* pengembangan model jaringan syaraf tiruan yang dikembangkan oleh Google. TensorFlow bersifat fleksibel dan *scalable* untuk komputasi numerik menggunakan grafik aliran data. Pustaka ini memungkinkan pengguna memprogram dan melatih jaringan saraf dan model pembelajaran mesin lainnya secara efisien dan men-*deploy*-nya ke tahap produksi (Pang et al., 2020).

TensorFlow memungkinkan membangun model dalam tiga cara, yaitu model sekuensial (*sequential model*), *functional API*, dan model pra-latih (*pre-trained model*) (Sanchez et al., 2020).

2.2.6 Additive Angular Margin Loss

Additive Angular Margin Loss atau yang disebut juga dengan ArcFace, adalah salah satu jenis *loss function* yang dapat digunakan dalam membangun model *Convolutional Neural Network* (CNN). Keuntungan-keuntungan penggunaan ArcFace dapat dirangkum atas empat hal, yaitu *engaging*, *effective*, *easy*, dan *efficient* (Deng et al., 2018).

- *Engaging*, ArcFace secara langsung mengoptimalkan jarak batas geodesik (*geodesic distance margin*).
- *Effective*, ArcFace mencapai kinerja mutakhir pada sepuluh tolok ukur pengenalan wajah termasuk set data gambar dan video skala besar.
- *Easy*, ArcFace hanya membutuhkan beberapa baris dan mudah diimplementasikan pada *deep learning* berbasis grafik komputasi.
- *Efficient*, ArcFace hanya menambahkan kompleksitas komputasi yang dapat diabaikan selama pelatihan.

Secara matematis, ArcFace dapat dirumuskan seperti rumus II-1.

$$L_3 = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s(\cos(\theta_{y_i} + m))}}{e^{s(\cos(\theta_{y_i} + m))} + \sum_{j=1, j \neq y_i}^n e^{s \cos \theta_j}} \quad (\text{II-1})$$

Keterangan:

L_3 : Additive Angular Margin Loss

N : batch size

m : angular margin penalty

y_i : kelas ke- y_i

2.2.8 Metrik Evaluasi

Evaluasi model klasifikasi dapat dilakukan menggunakan *Confusion matrix*. *Confusion matrix* digunakan sebagai alat visualisasi evaluasi model pada permasalahan *supervised learning*. Kolom matriks mewakili kelas prediksi (*predicted class*) dan baris mewakili kelas sebenarnya (*actual class*) (Novandya, 2017).

Pada Gambar II-7 menunjukkan bentuk dari *confusion matrix*.

| | | Actual Values | |
|------------------|--------------|---------------|--------------|
| | | Positive (1) | Negative (0) |
| Predicted Values | Positive (1) | TP | FP |
| | Negative (0) | FN | TN |

Gambar II-7. *Confusion Matrix*

Confusion matrix terdiri dari 4 kategori nilai, yaitu:

- *True Positive (TP)*

Model memprediksi kelas dengan benar, prediksi positif dan nilai sebenarnya positif.

- *True Negative (TN)*

Model memprediksi kelas dengan benar, prediksi negatif dan nilai sebenarnya negatif.

- *False Positive (FP)*

Model salah prediksi kelas, prediksi positif dan nilai sebenarnya negatif.

- *False Negative (FN)*

Model salah prediksi kelas, prediksi negatif dan nilai sebenarnya positif.

Perhitungan *confusion matrix* menghasilkan empat keluaran yang dapat dijadikan tolak ukur mengukur performa model, yaitu *accuracy*, *recall*, *precision*, dan *F-1 Score* (Saputro & Sari, 2020). Pada penelitian yang dilakukan, penulis menggunakan keempat metrik tersebut.

- *Accuracy*

Accuracy merupakan jumlah model benar memprediksi kelas (*true positive* dan *true negative*) dibagi dengan total semua hasil prediksi.

Secara matematis, *Accuracy* dirumuskan seperti pada rumus II-2.

$$acc = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \text{ (II-2)}$$

- *Recall*

Recall merupakan perhitungan dari kondisi ketika kelas aktualnya positif, seberapa sering model memprediksi positif. Secara matematis, *recall* dirumuskan seperti pada rumus II-3.

$$r = \frac{TP}{TP+FN} \times 100\% \text{ (II-3)}$$

- *Precision*

Precision merupakan hasil perhitungan dari ketika model memprediksi positif, seberapa sering prediksi tersebut bernilai benar. Secara matematis, *precision* dirumuskan seperti pada rumus II-4.

$$p = \frac{TP}{TP+FP} \times 100\% \text{ (II-4)}$$

- *F-1 Score*

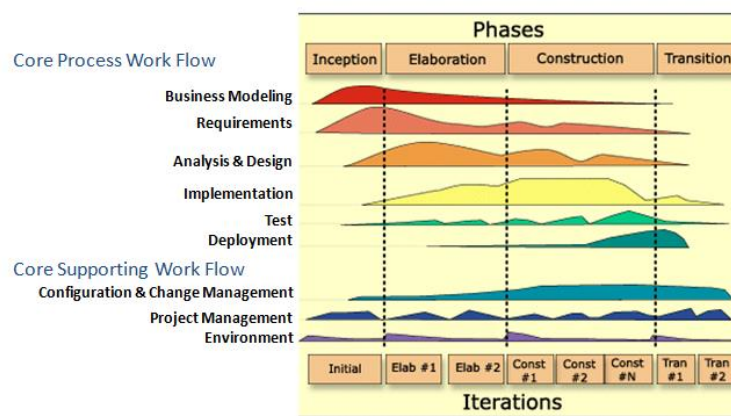
F-1 Score merupakan nilai rata-rata harmonic dari *precision* dan *recall*.

Secara matematis, *F-1 Score* dirumuskan seperti pada rumus II-5.

$$FM = 2 \times \frac{p \times r}{p+r} \times 100\% \text{ (II-5)}$$

2.2.9 Rational Unified Process (RUP)

Rational Unified Process (RUP) adalah metode pengembangan perangkat lunak yang digunakan dalam proses pengembangan sistem pengenalan citra wajah bermasker. RUP adalah metode pengembangan perangkat lunak yang bersifat *architecture-centric* dan berorientasi *use case driven*. RUP dilakukan secara berulang-ulang (*iterative*). Dalam penerapannya, setiap iterasi RUP terbagi atas 4 fase pengembangan sistem, yaitu fase inepsi, fase elaborasi, fase konstruksi, dan fase transisi (Hakim & Rizky, 2020). Visualisasi fase-fase dalam RUP dapat dilihat pada Gambar II-8.



Gambar II-8. Fase-fase RUP

Dalam setiap iterasi, serangkaian aktivitas dilaksanakan. Aktivitas-aktivitas tersebut disebut disiplin (*disciplines*). Disiplin tersebut dibagi atas dua jenis, yaitu disiplin utama dan disiplin pendukung. Disiplin utama antara lain *business modelling, requirements, analysis and design, implementation*, dan *testing and deployment*. Sedangkan disiplin pendukung adalah manajemen konfigurasi dan perubahan, manajemen proyek, serta lingkungan. Secara umum, mayoritas aktivitas hanya dilakukan pada satu fase (Shafiee et al., 2020).

2.3 Penelitian Lain yang Relevan

Dalam proses penyusunan skripsi, terdapat beberapa-beberapa referensi penelitian-penelitian sebelumnya yang digunakan penulis sebagai pendukung dan penunjang proses penelitian.

Topik pengenalan wajah bermasker pernah diteliti oleh (Li et al., 2021) dengan menggunakan pendekatan *cropping-based* yang diintegrasikan dengan *Convolutional Block Attention Module* (CBAM). Pendekatan *cropping-based* digunakan penulis bertujuan untuk mencari titik pemotongan optimal (*optimal cropping*) dari masing-masing studi kasus yang diteliti. Penulis menggunakan 4 dataset untuk membangun model pengenalan wajah, yaitu *Simulated Masked Face Recognition Dataset* (SMFRD), *Webface*, *AR*, dan *Extend Yela B*. Pengujian pendekatan diajukan menggunakan metrik akurasi yang dibagi atas 4 studi kasus berbeda, yaitu *case 1*, *case 2*, *case 3*, *case 4*. Kemudian, hasil penelitian dibandingkan dengan tingkat akurasi penelitian-penelitian terdahulu. Tingkat

akurasi dari metode yang diajukan penulis pada masing-masing studi kasus antara lain 91,5%, 86,9%, 81,4%, dan 92,6%.

Penelitian pengenalan wajah bermasker pernah dilakukan oleh (Mandal et al., 2021) menggunakan salah satu arsitektur model pra-latih CNN, yaitu ResNet-50. Penelitian dilakukan bertujuan untuk menjawab tantangan ditengah Pandemi Covid-19 untuk membangun sistem pengenalan wajah yang dapat mengenali citra wajah bermasker. Dataset yang digunakan dalam penelitian ini adalah RMFD. Pengujian hasil penelitian menggunakan metrik akurasi dengan melakukan perbandingan terhadap model pengenalan wajah bermasker dan model pengenalan wajah tidak bermasker. Hasil penelitian menunjukkan bahwa tingkat akurasi dari model pengenalan wajah bermasker adalah 47,9%, sedangkan tingkat akurasi dari model pengenalan wajah tidak bermasker adalah 89,7%.

Penelitian (Hariri, 2022) menggunakan CNN dengan paradigma *Bag-of-Features* (BoF) untuk membangun model pengenalan wajah bermasker yang efisien. Penulis melaksanakan penelitian ini bertujuan untuk mengatasi masalah tingginya penggunaan memori dan kapasitas pemrosesan. Dalam proses penelitian, 3 arsitektur model CNN pra-latih digunakan, yaitu VGG-16, AlexNet, dan ResNet-50. Dua dataset digunakan pada penelitian, antara lain *Real-world Masked Face Recognition Dataset* (RMFRD) dan *Simulated Masked Face Recognition Dataset* (SMFRD). Pengujian hasil penelitian dilakukan menggunakan metrik akurasi pada masing-masing dataset. Hasil penelitian menunjukkan bahwa tingkat akurasi dari metode yang diajukan 91,3% pada dataset RMFRD dan 88,9% pada dataset SMFRD.

2.4 Kesimpulan

Bab kajian literatur telah menjabarkan secara rinci terkait dasar-dasar teori dan penelitian-penelitian sebelumnya yang relevan untuk mendukung proses penelitian.

BAB III

METODOLOGI PENELITIAN

3.1 Pendahuluan

Bab ini menguraikan setiap tahapan penelitian yang dilakukan dalam proses pengembangan perangkat lunak, mulai dari tahap pengumpulan data hingga manajemen proyek penelitian. Tahapan penelitian ini dijadikan acuan pada setiap tahap pengembangan perangkat lunak pengenalan citra wajah bermasker menggunakan CNN dengan teknik *Additive Angular Margin Loss*.

3.2 Pengumpulan Data

3.2.1 Jenis Data

Jenis dataset yang digunakan pada penelitian model klasifikasi citra wajah bermasker adalah data sekunder. Dataset tersebut adalah *Real-World Masked Face Dataset* (RMFD) (Wang et al., 2020). RMFD terdiri dari 525 subjek (kelas) yang mencakup 90.000 citra wajah tanpa masker dan 5.000 citra wajah bermasker. Ukuran citra gambar yang tersedia pada dataset yang digunakan bervariasi.

3.2.2 Sumber Data

RMFD digunakan pada penelitian model klasifikasi citra wajah bermasker bersumber dari *National Engineering Research Center for Multimedia Software*

(NERCMS), *School of Computer Science, Wuhan University*. Dataset tersebut dapat diakses langsung secara publik melalui laman Kaggle¹.

3.2.3 Metode Pengumpulan Data


Dalam penelitian ini, data dikumpulkan dengan cara mengunduh langsung dataset dari sumber data.

3.2.4 Sampel Data





Dalam penelitian ini, data dikumpulkan dengan cara mengunduh langsung dataset dari sumber data.

Tabel III-1 dan Tabel III-2 berikut secara berurutan menunjukkan sampel gambar dari masing-masing kelas citra wajah bermasker dan citra wajah tidak bermasker dari dataset RMFD. Tabel-tabel tersebut terdiri dari 6 baris dan 4 kolom. Kolom-kolom terdiri dari No, Nama File (berisi informasi tentang direktori dan nama file gambar), Identitas Wajah (berisi informasi tentang label/kelas file gambar), dan Gambar.

Tabel III-1. Sampel Citra Wajah Bermasker

| No | Nama File | Identitas Wajah | Gambar |
|----|---------------------------|-----------------|---|
| 1 | masked/jingtian/0_0_0.jpg | Jingtian |  |

¹ <https://www.kaggle.com/datasets/muhammedalkran/masked-facerecognition>

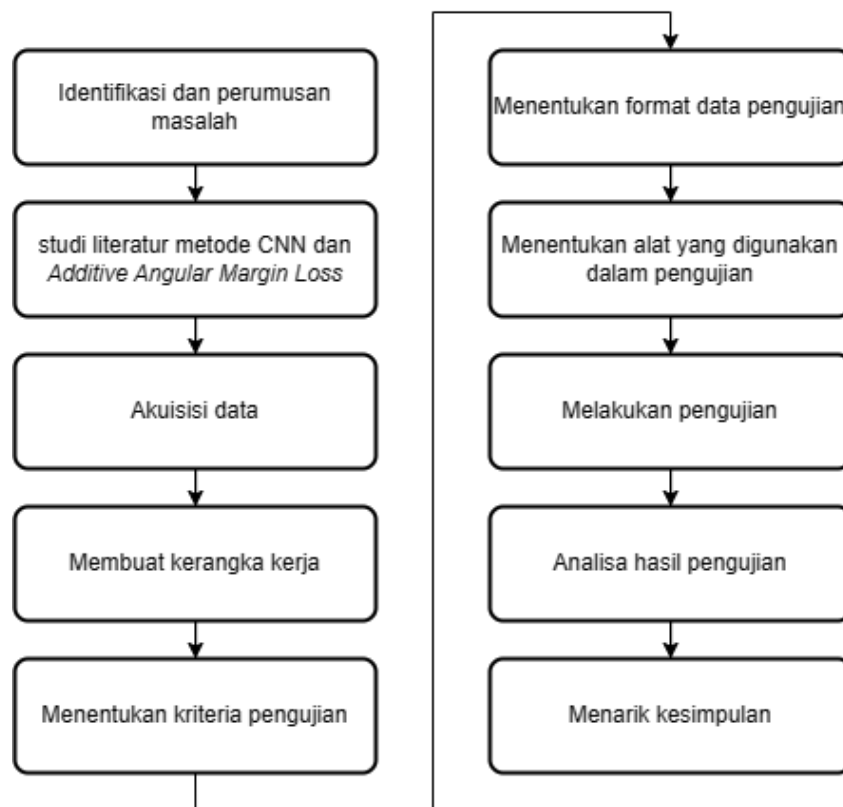
| No | Nama File | Identitas Wajah | Gambar |
|----|----------------------------------|-----------------|---|
| 2 | masked/linxinru/0_0_2.jpg | Linxinru |  |
| 3 | masked/luhan/0_0_0.jpg | Luhan |  |
| 4 | masked/masu/0_0_3.jpg | Masu |  |
| 5 | masked/zhangyixing/ 0_0_0.jpg | Zhangyixing |  |

Tabel III-2. Sampel Citra Wajah Tidak Bermasker

| No | Nama File | Identitas Wajah | Gambar |
|----|---|-----------------|---|
| 1 | unmasked/jingtian/ 0_0_jingtian_0002.jpg | Jingtian |  |
| 2 | unmasked/linxinru/ 0_0_linxinru_0002.jpg | Linxinru |  |
| 3 | unmasked/luhan/ 0_0_luhan_0012.jpg | Luhan |  |
| 4 | unmasked/masu/ 0_0_masu_0005.jpg | Masu |  |
| 5 | unmasked/zhangyixing/ 0_0_zhangyixing_0002.jpg | Zhangyixing |  |

3.3 Tahapan Penelitian

Penelitian ini dilakukan berdasarkan tahapan-tahapan penelitian. Adapun tahapan penelitian dapat dilihat pada Gambar III-1.



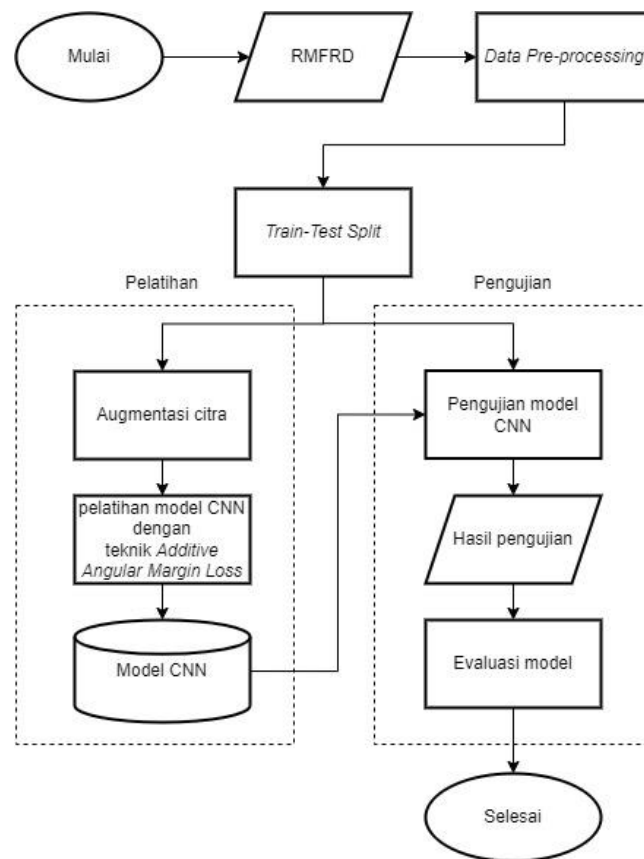
Gambar III-1. Tahapan-Tahapan Penelitian

3.3.1 Akuisisi Data

Pada tahapan akuisisi data, penulis mengambil dataset yang diperlukan sesuai dengan yang dijelaskan pada subbab 3.2. Dataset tersebut disimpan penulis di komputer dan penyimpanan awan, yaitu *Google Drive* pada akun pribadi pengguna.

3.3.2 Kerangka Kerja

Kerangka kerja penelitian model klasifikasi pengenalan citra wajah bermasker menggunakan CNN dengan teknik *Additive Angular Margin Loss* mengacu pada Gambar III-2.



Gambar III-2. Diagram Alir Klasifikasi Citra Wajah Bermasker Menggunakan CNN dengan Teknik Additive Angular Margin Loss

3.3.1.1 Data Pre-Processing

Data pre-processing merupakan suatu proses mengolah data menjadi lebih efektif dan sesuai dengan kebutuhan pengembangan model pembelajaran. Pada proses ini, dilakukan *resizing image* dan *rescale* terhadap dataset citra wajah yang digunakan pada penelitian. Hal ini dilakukan bertujuan untuk mempercepat pemrosesan citra pada tahapan selanjutnya dan mengurangi sumber daya perangkat keras yang dipakai.

Pada proses *resizing*, data-data gambar tersebut disamakan rasio resolusinya menjadi 180 x 180 menggunakan bantuan pustaka TensorFlow.

Kemudian, dari 525 kelas yang tersedia di dataset, penelitian yang dilakukan hanya menggunakan 15 kelas. Hal ini dikarenakan 15 kelas tersebut memiliki gambar minimal sebanyak 20 gambar pada data wajah bermasker dan wajah tidak bermasker. Tabel III-3 berikut menunjukkan jumlah data pada dataset setelah dilakukan pra-olah data.

Tabel III-3. Jumlah Data setelah Pra-Olah Data

| No | Nama | Jumlah Data |
|-------|-----------------------------|-------------|
| 1 | Citra wajah bermasker | 373 |
| 2 | Citra wajah tidak bermasker | 2906 |
| Total | | 3279 |

3.3.1.2 *Train-Test Split*

Pada proses *Train-test split*, dataset dibagi menjadi 2 bagian yaitu data latih dan data uji. Perbandingan banyak data latih dan data uji sebesar 7:3, 70% untuk data latih dan 30% untuk data uji. Data tersebut dibagi dengan bantuan salah satu fungsi dari pustaka *Scikit Learn*, yaitu `train_test_split`. 2 bagian data tersebut masing-masing secara berurutan digunakan untuk proses pelatihan model dan pengujian model. Tabel III-4 menunjukkan jumlah tiap bagian data setelah dilakukan pembagian data.

Tabel III-4. Pembagian Data

| No | Jenis Dataset | Data Latih | Data Uji |
|----|---|------------|----------|
| 1 | <i>Masked Face Dataset</i> | 257 | 116 |
| 2 | <i>Masked + Non Masked Face Dataset</i> | 2285 | 994 |

3.3.1.3 Augmentasi Citra Wajah Data Latih

Pada data latih yang telah dipisahkan dengan data uji, diterapkan augmentasi data. Hal ini dilakukan bertujuan untuk memperbanyak data latih dan meningkatkan variasi data latih. Augmentasi data yang diterapkan antara lain: *rotation*, *horizontal flipping*, dan *zooming*. Fungsi yang digunakan untuk melakukan augmentasi-augmentasi tersebut adalah `ImageDataGenerator()`, salah satu fungsi dari pustaka TensorFlow. Tabel III-5 berikut menunjukkan konfigurasi augmentasi data yang dilakukan.

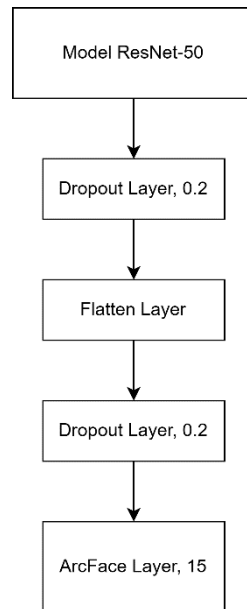
Tabel III-5. Konfigurasi Parameter Augmentasi Data

| No | Jenis Parameter | Nilai Parameter |
|----|------------------------|-----------------|
| 1 | <i>Rescale</i> | 1/255 |
| 2 | <i>Rotation range</i> | 20 |
| 3 | <i>Zoom range</i> | 0.05 |
| 4 | <i>Horizontal flip</i> | <i>True</i> |
| 5 | <i>fill mode</i> | <i>nearest</i> |

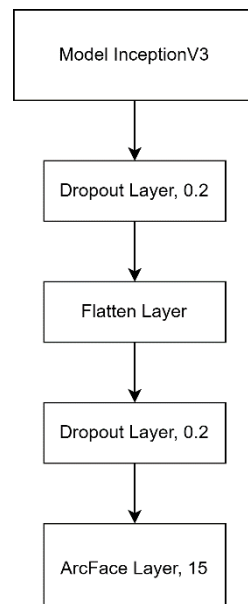
3.3.1.4 Pelatihan Model Pengenalan Wajah Bermasker

Dataset yang telah dibagi digunakan untuk proses pelatihan model klasifikasi citra wajah bermasker. Bagian dataset yang digunakan untuk pelatihan adalah data latih. Pelatihan model pengenalan wajah bermasker menggunakan CNN arsitektur ResNet-50 dan InceptionV3 dengan *Additive Angular Margin Loss*. Gambar III-3

dan III-4 menunjukkan arsitektur model ResNet-50 + ArcFace dan InceptionV3 + ArcFace.



Gambar III-3. Arsitektur Model ResNet-50 + ArcFace



Gambar III-4. Arsitektur Model InceptionV3 + ArcFace

3.3.1.5 Pengujian dan Evaluasi Model

Setelah melakukan pelatihan model, model tersebut diuji performanya menggunakan data uji. Kemudian hasil prediksi dari pengujian dibandingkan dengan hasil sebenarnya dari data uji. Perbandingan tersebut diukur menggunakan *accuracy*, *recall*, *precision*, dan *F-1 Score* yang dijadikan acuan dalam evaluasi model.

3.3.3 Kriteria Pengujian

Tahap kriteria pengujian dilakukan pengujian terhadap performa model klasifikasi citra wajah bermasker menggunakan CNN dan *Additive Angular Margin Loss*. Kriteria pengujian yang digunakan untuk melakukan evaluasi terhadap model adalah *confusion matrix* terhadap data latih dan data uji. Setelah dibuat *confusion matrix*, nilai-nilai yang diambil antara lain: *accuracy*, *recall*, *precision*, dan *F-1 Score*.

3.3.4 Format Data Pengujian

Proses pelatihan model menghasilkan nilai akurasi pelatihan, akurasi validasi, *loss*, dan *validation loss* dari *epoch* pertama sampai *epoch* terakhir. Keempat nilai tersebut dibandingkan dan divisualisasikan dalam bentuk plot garis menggunakan Matplotlib. Kemudian, model yang dihasilkan akan diuji dengan data uji. Data hasil pengujian tersebut disajikan dalam format tabel yang terdiri dari nilai evaluasi berdasarkan kriteria pengujian. Format tabel tersebut akan disajikan sesuai dengan Tabel III-6.

Tabel III-6. Tabel Evaluasi Model CNN dengan *Additive Angular Margin Loss*

| Konfigurasi Model | <i>Accuracy</i> | <i>Recall</i> | <i>Precision</i> | <i>F-1 Score</i> |
|-------------------|-----------------|---------------|------------------|------------------|
| | | | | |
| | | | | |
| | | | | |

3.3.5 Alat yang Digunakan dalam Pengujian

Perangkat keras dan perangkat lunak yang digunakan dalam penelitian adalah sebagai berikut:

- Prosesor : AMD Ryzen 7 2700U with Radeon Vega Mobile Gfx 2.20 GHz
- VGA : AMD Radeon(TM) RX Vega 10 Graphics
- Memori (RAM) : 8 GB
- Sistem Operasi : Windows 10 (64 bit)
- Bahasa Pemrograman: Python
- Pustaka : TensorFlow, scikit-learn, OpenCV, Matplotlib, Numpy
- Framework : Flask, Bootstrap
- IDE : Google Colab, Visual Studio Code.

3.3.6 Pengujian Penelitian

Pengujian pada penelitian dilakukan menggunakan arsitektur dan *hyperparameter* CNN serta *loss function* yang diajukan. Pengujian tersebut

dilakukan menggunakan data uji berupa citra wajah yang bermasker. Hasil pengujian tersebut disajikan seperti pada Tabel III-5.

3.3.7 Analisis Hasil Pengujian

Analisis hasil pengujian dilakukan dengan berdasarkan data hasil yang disajikan menggunakan Tabel III-5. Model terbaik ditentukan berdasarkan hasil analisis tersebut. Setelah model terbaik ditentukan, penulis melakukan analisis terhadap visualisasi plot garis dari akurasi dan *loss* pada data latih dan data uji. Hal ini dilakukan untuk mengetahui model yang dihasilkan *good fitting*, *overfitting* atau *underfitting*.

3.4 Metode Pengembangan Perangkat Lunak

Metode pengembangan perangkat lunak yang digunakan pada penelitian adalah *Rational Unified Process* (RUP). Langkah-langkah yang dilakukan dalam pengembangan perangkat lunak pada penelitian ini dijelaskan secara detail pada sub bab 3.4.1 sampai sub bab 3.4.4.

3.4.1 Fase Insepsi

Fase insepsi berfokus tahapan *business modelling* dan *requirements*. Pada tahap *business modelling*, penulis memahami dan mengeksplorasi terkait ruang lingkup masalah yang diteliti. Pada tahap *requirements*, penulis menyusun *user requirements* dan *functional requirements* dari perangkat lunak yang dikembangkan.

3.4.2 Fase Elaborasi

Fase elaborasi berfokus pada tahap *requirements, analysis and design*, dan *implementation*. Pada tahap *requirements*, penulis menganalisa *requirements* dan *use case diagram* sederhana yang telah dibuat. Setelah itu, penulis menentukan arsitektur perangkat lunak, desain basis data, dan desain antarmuka yang berdasarkan pada *requirements*. Kemudian pada tahap *analysis and design*, penulis membuat *activity diagram*, *sequence diagram*, dan *class diagram*. Pada tahap *implementation*, penulis mendokumentasikan arsitektur perangkat lunak, desain basis data, desain antarmuka, *activity diagram*, *sequence diagram*, dan *class diagram*.

3.4.3 Fase Konstruksi

Fase konstruksi berfokus pada tahap *analysis and design, implementation*, dan *testing and deployment*. Pada tahap *analysis and design*, penulis membangun arsitektur CNN dengan model pra-latih ResNet-50 dengan teknik *additive angular margin loss*. Kemudian dilanjutkan tahap *implementation*, penulis melakukan pra-pengolahan terhadap dataset, membagi dataset menjadi data latih dan data uji, melakukan augmentasi citra terhadap data latih. Setelah itu, penulis melakukan pembelajaran model CNN dan membangun perangkat lunak sistem pengenalan wajah. Pada tahap *testing and deployment*, penulis menguji model CNN dengan data uji dan men-deploy model yang telah dibuat ke perangkat lunak yang telah dikembangkan.

3.4.4 Fase Transisi

Pada fase transisi hanya berfokus di tahapan *testing and deployment*. Pada tahap tersebut, penulis menyusun beberapa kenario pengujian. Setelah itu, penulis melakukan pengujian berdasarkan scenario tersebut. Hasil pengujian tersebut, didokumentasikan secara rapi oleh penulis untuk keperluan keberlanjutan pengembangan perangkat lunak.

3.5 Kesimpulan

Bab ini telah menjabarkan secara rinci tahapan-tahapan yang dilakukan penulis selama penelitian. Selain itu, bab ini juga menjelaskan juga metode pengembangan perangkat lunak yang digunakan oleh penulis untuk membangun perangkat lunak sistem pengenalan citra wajah bermasker.

BAB IV

PENGEMBANGAN PERANGKAT LUNAK

4.1 Pendahuluan

Bab ini menjabarkan proses pengembangan perangkat lunak yang digunakan sebagai alat penelitian. Metode *Rational Unified Process* (RUP) digunakan dalam pengembangan perangkat lunak tersebut.

4.2 Rational Unified Process

RUP terdiri dari empat fase yaitu fase inepsi, fase elaborasi, fase transisi, dan fase konstruksi. Keempat fase tersebut dijelaskan secara rinci pada subbab 4.2.

4.2.1 Fase Inepsi

Pada tahap inepsi dilakukan beberapa aktivitas yaitu membuat pemodelan bisnis, menentukan kebutuhan pengguna, fungsional, dan non-fungsional.

4.2.1.1 Pemodelan Bisnis

Perangkat lunak yang dikembangkan dalam penelitian ini adalah sistem pengenalan citra wajah bermasker berbasis web. Data yang digunakan sebagai masukan pada perangkat lunak yaitu data gambar orang bermasker. Keluaran yang dihasilkan dari perangkat lunak ini adalah identitas orang dari citra gambar masukan.

4.2.1.2 Kebutuhan Sistem

Kebutuhan sistem pada perangkat lunak terdiri dari kebutuhan fungsional dan kebutuhan non-fungsional. Kebutuhan fungsional adalah layanan atau proses yang harus disediakan oleh perangkat lunak, sedangkan kebutuhan non-fungsional adalah kebutuhan pelengkap yang menekankan sifat perilaku perangkat lunak. Kebutuhan fungsional dan kebutuhan non-fungsional tersebut dapat dilihat pada Tabel IV-1 dan Tabel IV-2.

Tabel IV-1. Kebutuhan Fungsional Perangkat Lunak

| No | Kebutuhan Fungsional |
|----|--|
| 1 | Perangkat lunak dapat mendeteksi wajah dan mengklasifikasi atau mengenali identitas orang. |
| 2 | Perangkat lunak dapat menerima masukan citra wajah yang dimasukkan pengguna berupa gambar. |

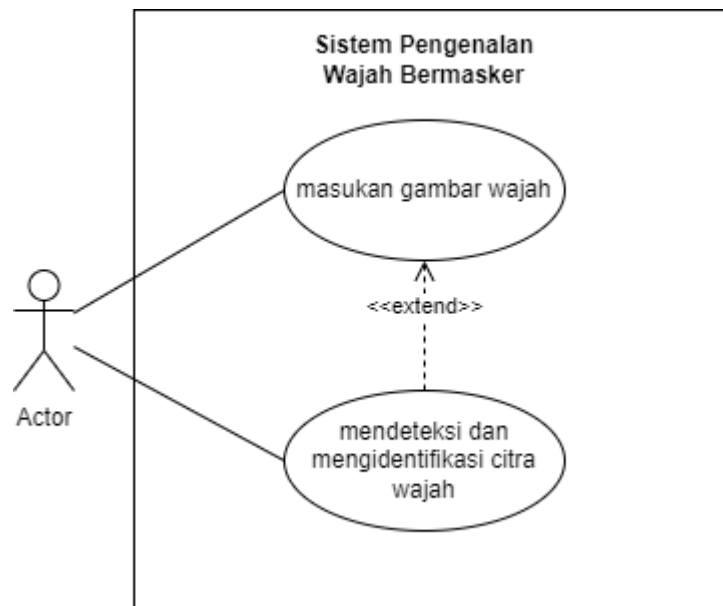
Tabel IV-2. Kebutuhan Non-Fungsional Perangkat Lunak

| No | Kebutuhan Non-Fungsional |
|----|---|
| 1 | Perangkat lunak memiliki antarmuka yang simpel dan mudah digunakan. |

4.2.1.3 Analisis dan Desain

a. Use Case Diagram

Use case diagram menggambarkan kegiatan yang dilakukan oleh aktor terhadap perangkat lunak. Use case diagram perangkat lunak sistem pengenalan citra wajah masker ditunjukkan pada Gambar IV-1.



Gambar IV-1. Use Case Diagram Perangkat Lunak

b. Definisi Aktor

Aktor pada perangkat lunak sistem pengenalan citra wajah bermasker ditunjukkan pada Tabel IV-3.

Tabel IV-3. Definisi Aktor

| No | Aktor | Deskripsi |
|----|----------|--|
| 1 | Pengguna | Orang yang dapat berinteraksi dengan perangkat lunak dan menggunakan semua fitur yang telah tersedia di dalam perangkat lunak. |

c. Definisi *Use Case*

Use Case pada perangkat lunak sistem pengenalan citra wajah bermasker didefinisikan pada Tabel IV-4.

Tabel IV-4. Definisi *Use Case*

| No | Use Case | Definisi |
|----|---|---|
| 1 | Masukan gambar wajah | Kegiatan ini digunakan untuk memuat data ke perangkat lunak dari data uji yang berupa gambar. |
| 2 | Mendeteksi dan mengidentifikasi citra wajah | Kegiatan ini digunakan untuk melakukan proses deteksi dan identifikasi citra wajah bermasker menggunakan arsitektur model ResNet-50 dengan teknik <i>additive angular margin loss</i> . |

d. Skenario *Use Case*

Skenario *use case* menguraikan urutan spesifik dari aksi aktor dan reaksi sistem berdasarkan *use case diagram* yang telah dibuat pada gambar IV-1. Berikut skenario *use case* dari perangkat lunak sistem pengenalan citra wajah bermasker yang diuraikan pada Tabel IV-5 dan Tabel IV-6.

Tabel IV-5. Skenario *Use Case* Masukan Gambar Wajah

| Identifikasi | |
|--------------|----------------------|
| Nomor | 101 |
| Nama | Masukan Gambar Wajah |

| Identifikasi | |
|---|---|
| Tujuan | Proses ini digunakan untuk mendapatkan paket |
| Deskripsi | Use case menggambarkan proses dalam mendapatkan paket dalam lingkup area tertentu |
| Aktor | Pengguna |
| Kondisi Awal | Pengguna membuka aplikasi |
| Pemicu | Menekan tombol pilih gambar. |
| Skenario Utama (Mengupload gambar) | |
| Aksi Aktor | Reaksi Sistem |
| 1. Menekan tombol pilih gambar | |
| | 2. Menampilkan folder dan file gambar di direktori penyimpanan |
| 3. Memilih gambar yang ingin diunggah | |
| 4. Menekan tombol unggah | |
| Kondisi Akhir | Gambar berhasil diunggah |

Tabel IV-6. Skenario *Use Case* Mendeteksi dan Mengidentifikasi Citra Wajah

| Identifikasi | |
|--|---|
| Nomor | 102 |
| Nama | Mendeteksi dan Mengidentifikasi Citra Wajah |
| Tujuan | Proses ini digunakan untuk mendeteksi dan mengenali wajah pengguna |
| Deskripsi | Use case menggambarkan proses deteksi dan identifikasi citra wajah pengguna yang bermasker atau tidak bermasker |
| Aktor | Pengguna |
| Kondisi Awal | Pengguna telah mengunggah gambar wajah |
| Pemicu | Menekan tombol kenali |
| Skenario Utama (Mendeteksi wajah) | |
| Aksi Aktor | Reaksi Sistem |
| 1. Menekan tombol kenali | |
| | 2. Menampilkan citra wajah yang telah dibatasi kotak dan dilengkapi nama pengguna |

| Identifikasi | |
|--|---|
| Kondisi Akhir | Citra wajah berhasil diprediksi |
| Skenario Alternatif (Tidak mendeteksi wajah) | |
| Aksi Aktor | Reaksi Sistem |
| Melakukan no 1 | |
| | 2. mengirim pesan peringatan bahwa tidak terdapat wajah di gambar yang diunggah |
| Kondisi Akhir | Citra wajah gagal diprediksi |

4.2.2 Fase Elaborasi

Setelah fase insepisi, pengembangan perangkat lunak dilanjutkan ke fase elaborasi. Pada fase ini dilakukan beberapa aktivitas yaitu membuat perangkat data perancangan antarmuka perangkat lunak, *activity diagram*, dan *sequence diagram*.

4.2.2.1 Pemodelan Bisnis

a. Perancangan Data

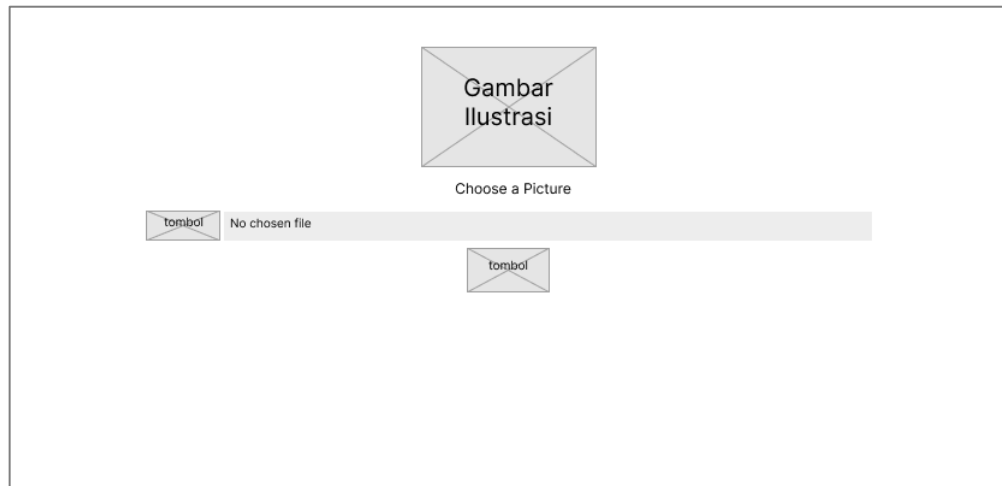
Perangkat lunak yang dibangun memiliki kemampuan untuk mengidentifikasi identitas dari citra wajah yang diunggah. Data yang akan melalui proses pengenalan adalah citra wajah yang termasuk dalam bagian data uji pada dataset.

b. Perancangan Antarmuka

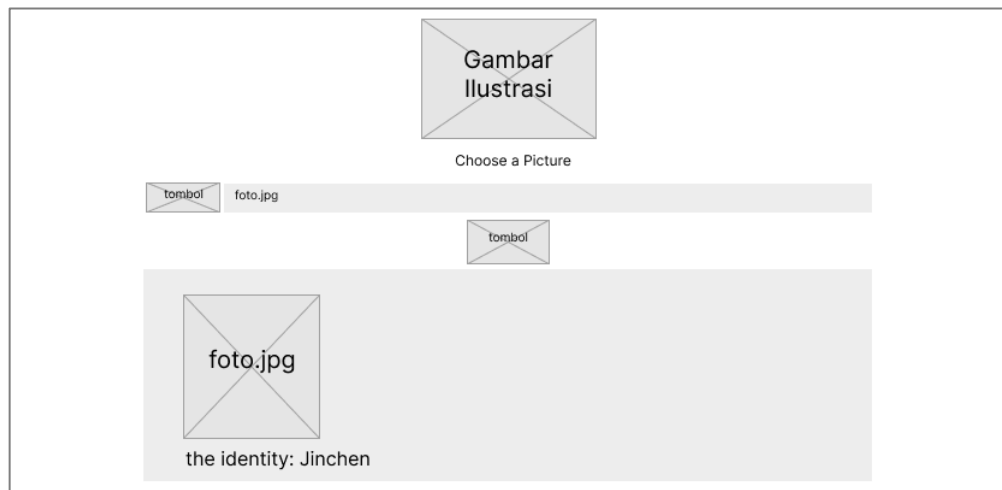
Perancangan antarmuka dilakukan bertujuan memberikan tampilan perangkat lunak yang sesuai dengan kebutuhan pengguna. Gambar IV-2 sampai dengan Gambar IV-4 merupakan rancangan antarmuka perangkat lunak yang dibangun.



Gambar IV-2. Rancangan Tampilan Halaman Depan



Gambar IV-3. Rancangan Tampilan Halaman Pengenalan Wajah



Gambar IV-4. Rancangan Tampilan Halaman Pengenalan Wajah setelah Ditekan Tombol *Predict*

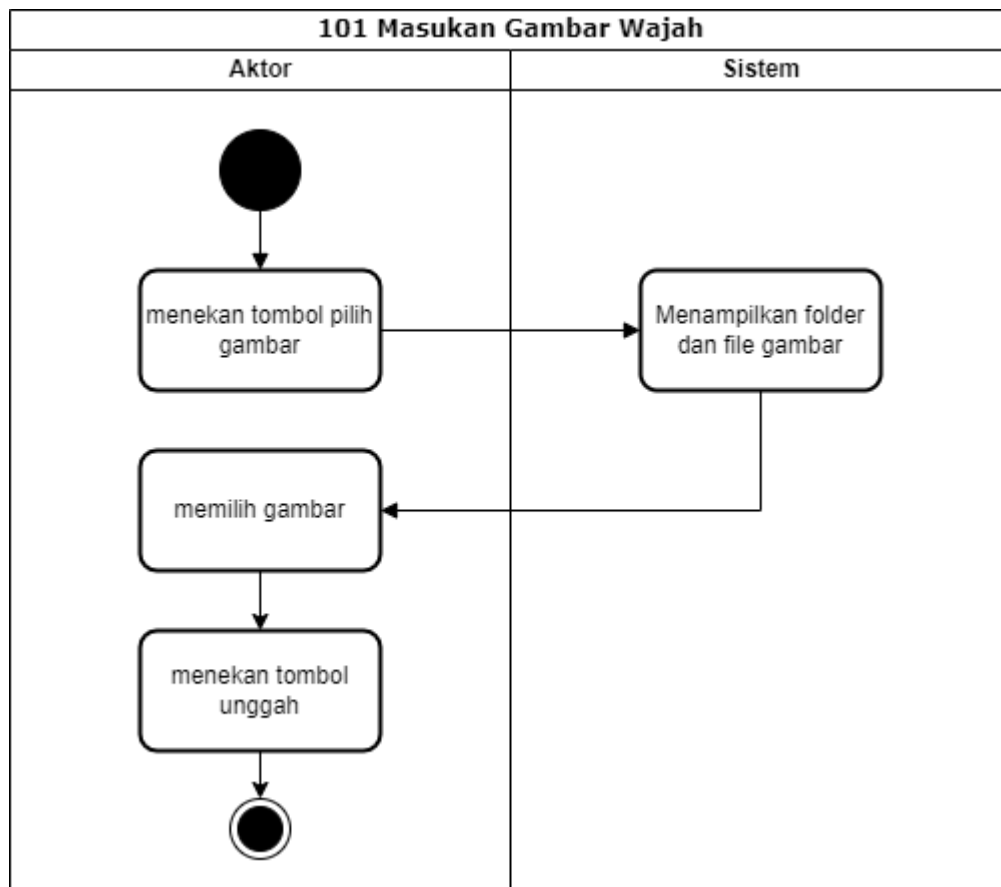
4.2.2.2 Kebutuhan Sistem

Perangkat keras, perangkat lunak, dan Bahasa pemrograman diperlukan dalam pengembangan perangkat lunak pada penelitian ini. Bahasa pemrograman yang digunakan yaitu Python, sedangkan perangkat lunak yang digunakan adalah Google Colab dan Figma.

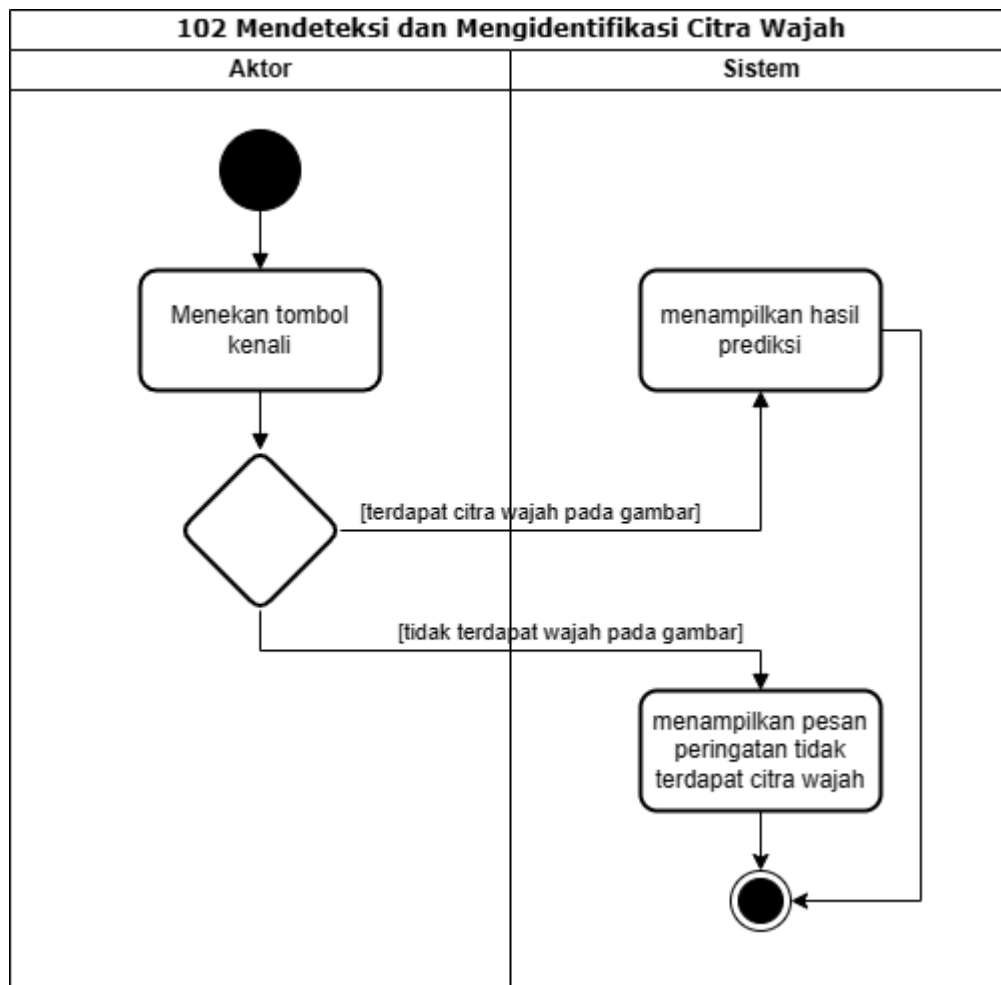
4.2.2.3 Analisis dan Desain

a. Activity Diagram

Activity diagram menggambarkan urutan aktivitas proses yang berjalan dalam sistem berdasarkan *use case* dan skenario *use case* yang telah dibuat. *Activity diagram* dari perangkat lunak sistem pengenalan wajah bermasker dapat dilihat pada Gambar IV-5 dan Gambar IV-6.



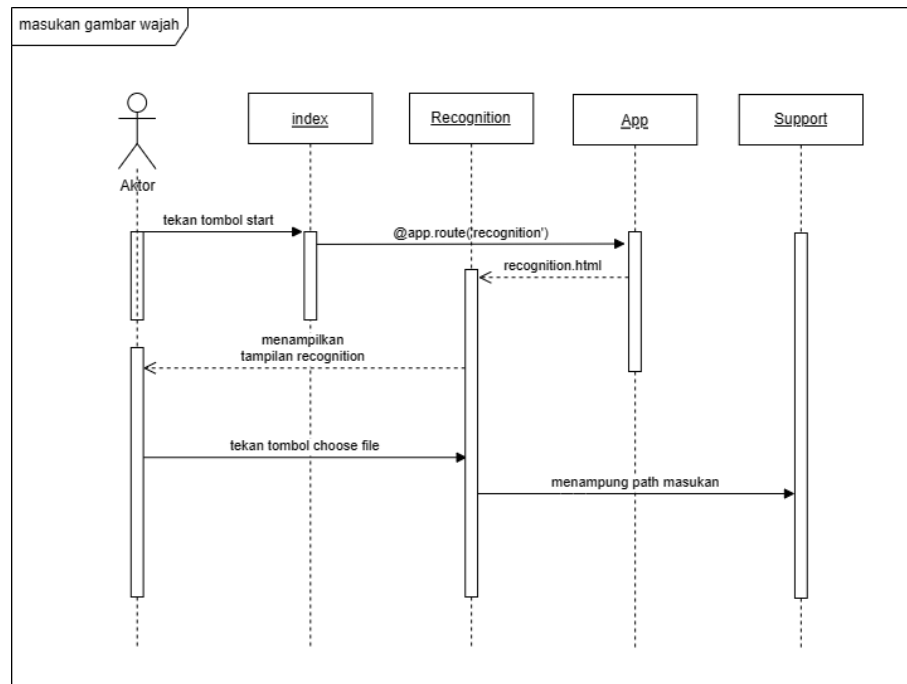
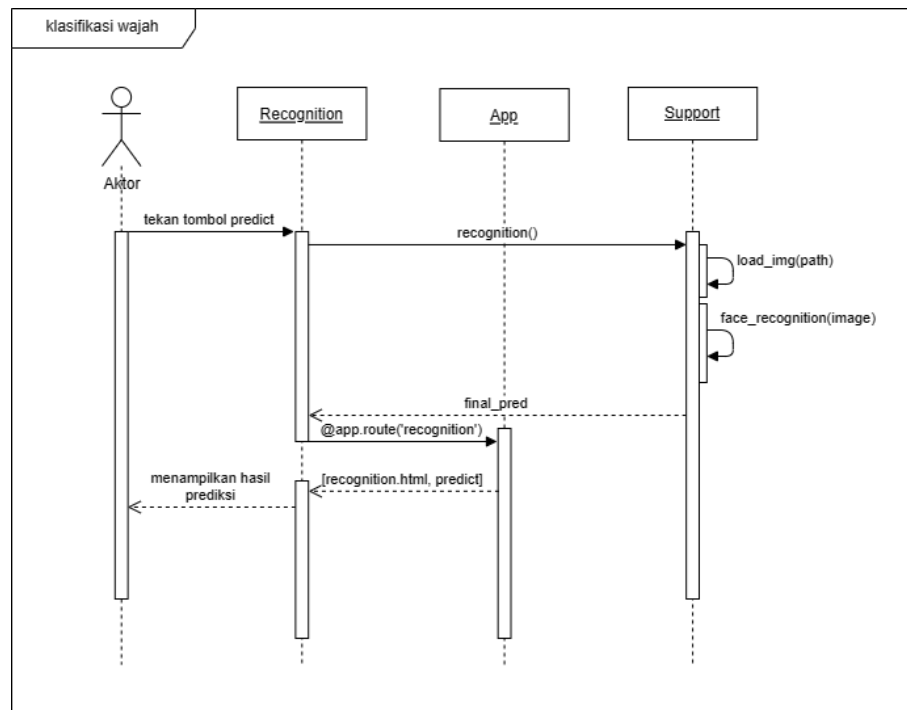
Gambar IV-5. *Activity Diagram* Masukan Gambar Wajah



Gambar IV-6. *Activity Diagram* Deteksi dan Identifikasi Citra Wajah

b. Sequence Diagram

Sequence diagram menggambarkan alur interaksi antar objek di dalam sistem. Berdasarkan *use case* pada Gambar IV-1, terdapat dua *sequence diagram* yaitu masukan gambar wajah pada Gambar IV-7 dan deteksi dan identifikasi citra wajah pada Gambar IV-8.

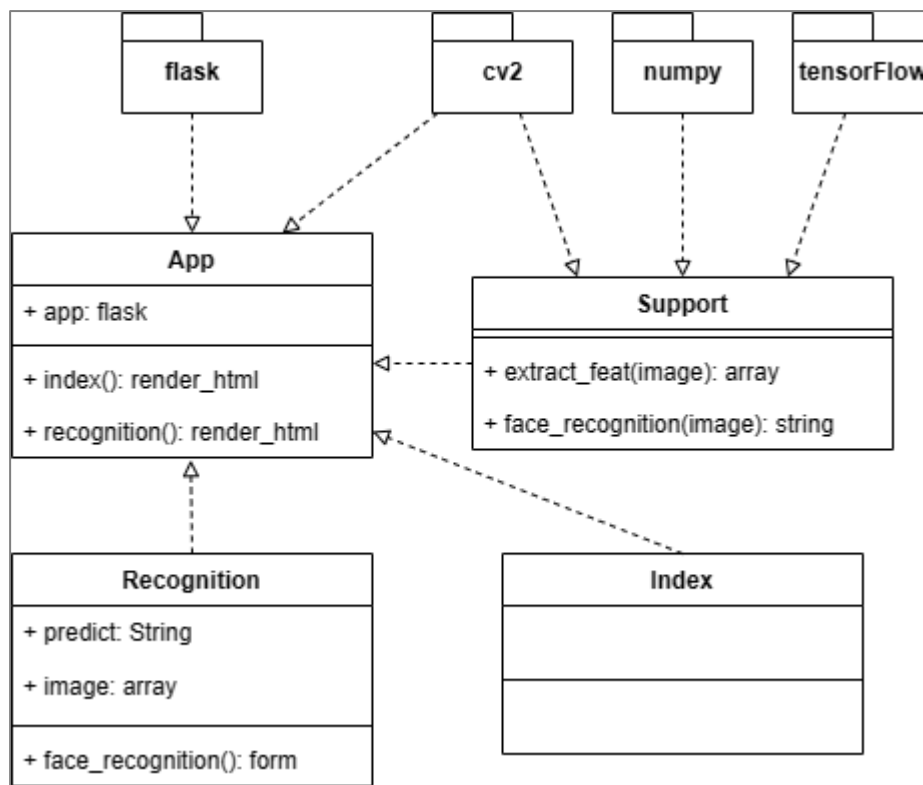
Gambar IV-7. *Sequence Diagram* Masukan Gambar WajahGambar IV-8. *Sequence Diagram* Deteksi dan Identifikasi Citra Wajah

4.2.3 Fase Kontruksi

Pada fase ini dilakukan beberapa aktivitas yang berkait dengan pembuatan perangkat lunak, yaitu merancang pemodelan *class diagram*, mengimplementasikan kelas-kelas tersebut pada perangkat lunak, dan mengimplementasikan antarmuka yang telah dirancang.

4.2.3.1 Kebutuhan Sistem

Pada tahap ini dijelaskan kelas yang dibangun pada perangkat lunak dan hubungannya satu sama lain melalui pemodelan diagram kelas. Diagram kelas perangkat lunak pada penelitian ini ditunjukkan oleh Gambar IV-9 berikut.



Gambar IV-9. Diagram Kelas

4.2.3.2 Implementasi

Pada fase konstruksi tahap implementasi, rancangan yang telah dibuat diimplementasikan menjadi sebuah perangkat lunak. Dalam subbab ini, implementasi dibahas dalam dua bagian, yaitu implementasi kelas dan implementasi antarmuka. Alat pendukung dalam proses implementasi yaitu bahasa pemrograman Python dan pustaka Flask.

a. Implementasi Kelas

Kelas yang telah dirancang sebelumnya diimplementasikan menggunakan bahasa pemrograman Python. Tabel IV-7 berikut menunjukkan hasil implementasi dari kelas-kelas tersebut.

Tabel IV-7. Keterangan Implementasi Kelas

| No | Nama Kelas | Nama File | Keterangan |
|----|------------|------------|--|
| 1 | Router | app.py | Kelas yang mengatur <i>routing</i> saat pengguna ingin membuka laman tertentu. |
| 2 | Support | support.py | Kelas yang melakukan proses deteksi dan identifikasi wajah dari gambar yang dimasukkan pengguna. |

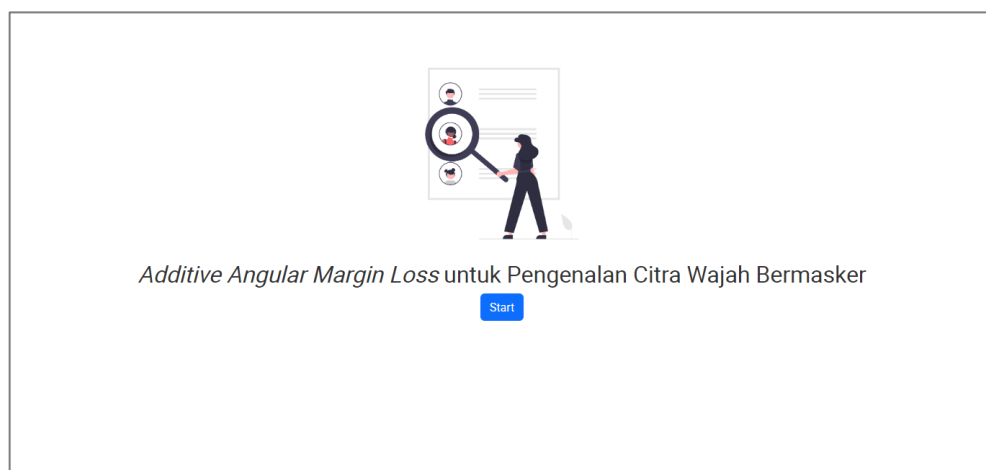
b. Implementasi Antarmuka

Pada tahap ini, antarmuka yang telah dirancang pada fase elaborasi diimplementasikan ke perangkat lunak. Implementasi antarmuka tersebut menggunakan *flask*. Tabel IV-8 adalah daftar file HTML sebagai bentuk implementasi antarmuka perangkat lunak.

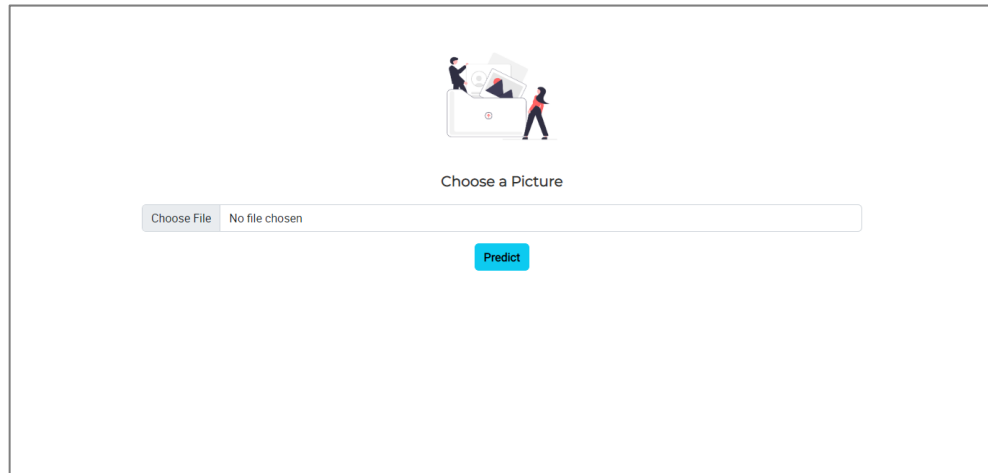
Tabel IV-8. Daftar File HTML

| No | Nama File | Keterangan |
|----|------------------|---|
| 1 | base.html | Sebagai kerangka tampilan dari setiap halaman perangkat lunak. |
| 2 | index.html | Sebagai tampilan dari halaman depan. |
| 3 | recognition.html | Sebagai tampilan dari halaman unggah gambar wajah dan identifikasi wajah. |

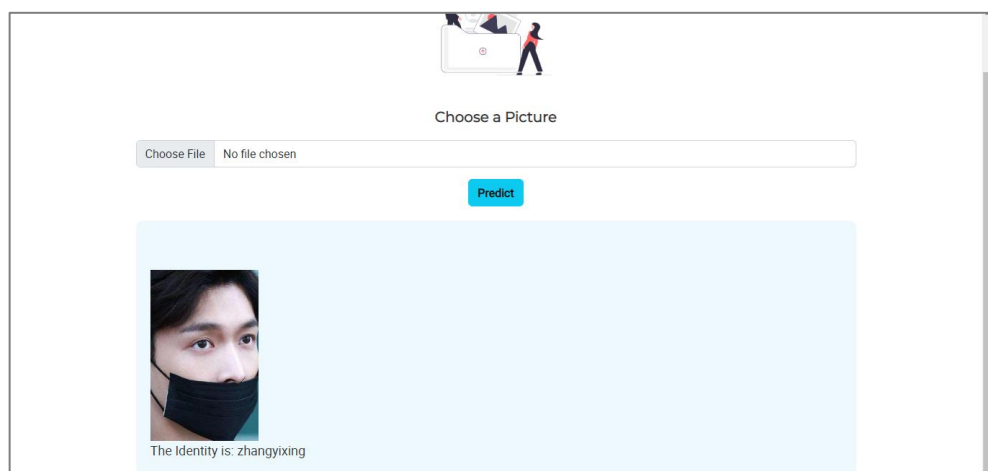
Gambar IV-10 sampai Gambar IV-12 berikut menunjukkan tampilan antarmuka dari file HTML pada Tabel IV-14 ketika dijalankan.



Gambar IV-10. Antarmuka Halaman Depan



Gambar IV-11. Antarmuka Halaman Pengenalan Wajah



Gambar IV-12. Antarmuka Halaman Pengenalan Wajah setelah Ditekan Tombol *Predict*

4.2.4 Fase Transisi

Fase transisi merupakan fase terakhir dari pengembangan perangkat lunak dengan metode RUP. Pada fase ini, beberapa aktivitas dilakukan seperti menyusun scenario pengujian, melakukan pengujian, dan mendokumentasikan hasil pengujian.

4.2.4.1 Pemodelan Bisnis

Pada tahap pemodelan bisnis, perangkat lunak yang telah dibangun diuji dengan menggunakan *Black Box Testing*. Pengujian diawali dengan pembuatan rencana pengujian.

4.2.4.2 Analisis dan Desain

Tahap analisis dan desain membahas tentang rencana pengujian. Rencana pengujian disusun berdasarkan dari diagram *use case* yang ada pada Gambar IV-1. Rencana pengujian untuk *use case* masukan gambar wajah ditunjukkan oleh Tabel IV-9. Kemudian, rencana pengujian untuk *use case* deteksi dan identifikasi wajah ditunjukkan oleh Tabel IV-10.

Tabel IV-9. Rencana Pengujian *Use Case* Masukan Gambar Wajah

| No | ID | Pengujian | Tingkat Pengujian |
|----|----------|--|-------------------|
| 1 | UC-101-1 | Mengunggah gambar wajah dari data uji. | Pengujian unit |

Tabel IV-10. Rencana Pengujian *Use Case* Deteksi dan Identifikasi Wajah

| No | ID | Pengujian | Tingkat Pengujian |
|----|----------|--|-------------------|
| 1 | UC-102-1 | Melakukan proses deteksi dan identifikasi wajah terhadap gambar/video yang diunggah. | Pengujian unit |

4.2.4.3 Implementasi

Pada tahap implementasi, dibahas hasil implementasi dari rencana pengujian pada Tabel IV-9 dan Tabel IV-10 terhadap perangkat lunak yang dibangun. Pengujian untuk *use case* masukan gambar wajah ditunjukkan oleh Tabel IV-11. Kemudian, pengujian untuk *use case* deteksi dan identifikasi wajah ditunjukkan oleh Tabel IV-12.

Tabel IV-11. Pengujian *Use Case* Masukan Gambar/Video Wajah

| ID | Prosedur Pengujian | Keluaran yang Diharapkan | Keluaran yang Didapatkan | Kesimpulan |
|----------|---|--|--|------------|
| UC-101-1 | Tekan tombol “pilih gambar”, kemudian pilih gambar. | Menampilkan gambar yang telah diunggah | Menampilkan gambar yang telah diunggah | Terpenuhi |

Tabel IV-12. Pengujian *Use Case* Deteksi dan Identifikasi Wajah

| ID | Prosedur Pengujian | Keluaran yang Diharapkan | Keluaran yang Didapatkan | Kesimpulan |
|----------|-----------------------|--|--|------------|
| UC-102-1 | Tekan tombol “kenali” | Menampilkan identitas dari citra wajah yang diunggah | Menampilkan identitas dari citra wajah yang diunggah | Terpenuhi |

4.3 Kesimpulan

Metode RUP digunakan untuk pengembangan perangkat lunak pada penelitian ini. RUP terdiri dari 4 fase, yaitu fase insepasi, fase elaborasi, fase konstruksi, dan fase transisi. Fase insepasi melakukan analisis data dan rancangan perangkat lunak. Fase elaborasi membuat rancangan antarmuka perangkat lunak dan *sequence diagram*. Fase konstruksi mengimplementasi rancangan yang telah dibuat pada fase sebelumnya. Terakhir, fase transisi melakukan pengujian terhadap perangkat lunak yang telah dibangun.

BAB V

HASIL DAN ANALISIS PENELITIAN

5.1 Pendahuluan

Bab ini membahas hasil penelitian pengenalan citra wajah bermasker menggunakan CNN dengan pendekatan *additive angular margin loss* yang telah dikembangkan berdasarkan parameter terbaik. Performa dari model CNN yang dilatih dan diuji diukur menggunakan metrik pengukuran berupa *confussion matrix*, *accuracy*, *precision*, *recall*, *F-1 Score*.

5.2 Data Hasil Penelitian

5.2.1 Konfigurasi Percobaan Parameter

Percobaan penelitian dilakukan dalam 4 skenario. Skenario-skenario tersebut memiliki konfigurasi-konfigurasi yang bervariasi pada model ataupun dataset. Selain itu, terdapat juga konfigurasi-konfigurasi yang sama pada 4 skenario tersebut. Konfigurasi-konfigurasi yang digunakan mengacu pada Tabel V-1 dan Tabel V-2.

Tabel V-1 berikut menunjukkan skenario-skenario dan variasi konfigurasi parameter yang digunakan pada masing-masing skenario.

Tabel V-1. Variasi Konfigurasi Parameter

| No | Model | Arsitektur | Dataset |
|----|---------|-----------------------|----------------------------|
| 1 | Model 1 | ResNet-50 + ArcFace | <i>Masked Face Dataset</i> |
| 2 | Model 2 | InceptionV3 + ArcFace | <i>Masked Face Dataset</i> |

| No | Model | Arsitektur | Dataset |
|----|---------|-----------------------|---|
| 3 | Model 3 | ResNet-50 + ArcFace | <i>Masked and Non Masked Face Dataset</i> |
| 4 | Model 4 | InceptionV3 + ArcFace | <i>Masked and Non Masked Face Dataset</i> |

Tabel V-2 berikut menunjukkan konfigurasi parameter tetap yang dijadikan sebagai acuan. Konfigurasi tersebut digunakan berdasarkan penelitian sebelumnya (Mandal et al., 2021).

Tabel V-2. Konfigurasi Parameter Tetap

| No | Jenis Parameter | Nilai Parameter |
|----|----------------------|---------------------------------|
| 1 | <i>Optimizer</i> | Adam |
| 2 | <i>Learning rate</i> | 0.0016 |
| 3 | <i>Batch Size</i> | 32 |
| 4 | <i>Loss</i> | <i>Categorical Crossentropy</i> |
| 5 | <i>Epoch</i> | 25 |

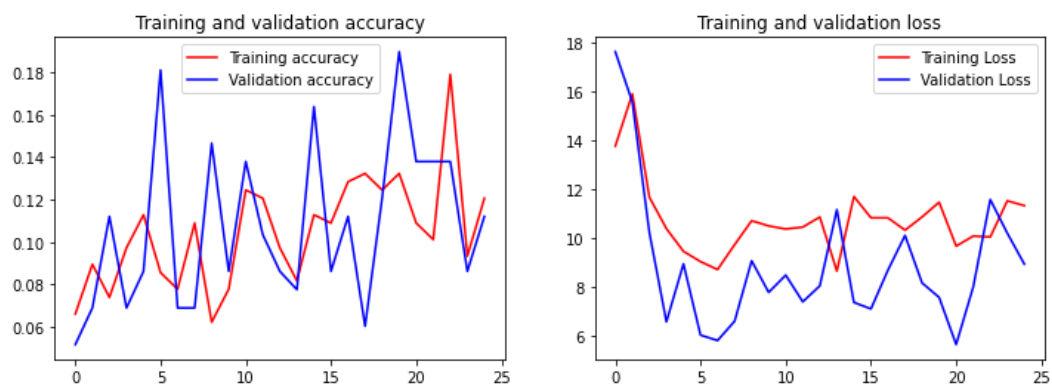
5.2.2 Analisis Hasil Penelitian

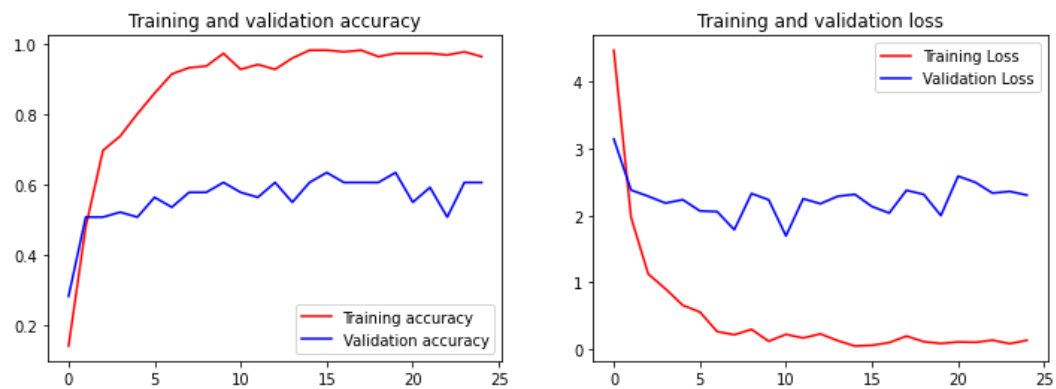
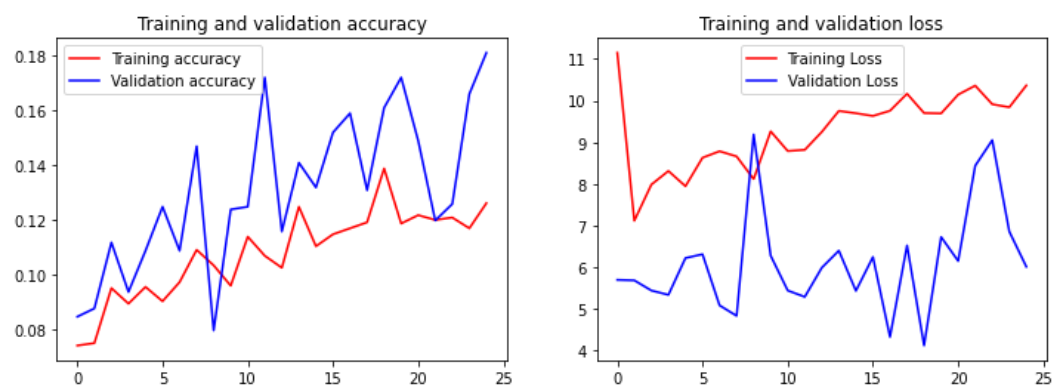
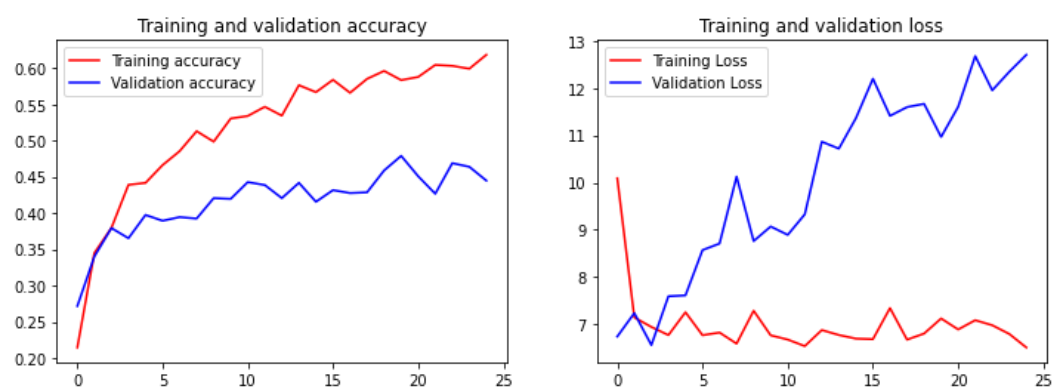
Setiap variasi konfigurasi model CNN memiliki performa pelatihan dan validasi yang berbeda-beda dalam klasifikasi citra wajah bermasker. Tabel V-3 berikut menunjukkan perbandingan performa model berdasarkan data latih dan data validasi.

Tabel V-3. Perbandingan Hasil *Training-Validation* Model

| No | Konfigurasi Model | <i>Train_Acc</i> | <i>Val_Acc</i> | <i>Train_Loss</i> | <i>Val_Loss</i> |
|----|---|------------------|----------------|-------------------|-----------------|
| 0 | ResNet-50 (Mandal et al., 2021) | 60,05% | 47,91% | 1,5005 | 2,4092 |
| 1 | Model 1: ResNet-50 + ArcFace | 12,06% | 11,21% | 11,3264 | 8,9414 |
| 2 | Model 2: InceptionV3 + ArcFace | 96,38% | 60,56% | 0,1357 | 2,3069 |
| 3 | Model 3: ResNet-50 + ArcFace | 12,60% | 18,11% | 10,3623 | 6,0141 |
| 4 | Model 4: InceptionV3 + ArcFace | 61,84% | 44,47% | 6,5041 | 12,7073 |

Gambar V-1 sampai Gambar V-4 berikut menunjukkan grafik nilai akurasi dan *loss* dari masing-masing skenario percobaan terhadap jumlah *epoch*.

Gambar V-1. Grafik Akurasi dan *loss* Model 1

Gambar V-2. Grafik Akurasi dan *loss* Model 2Gambar V-3. Grafik Akurasi dan *loss* Model 3Gambar V-4. Grafik Akurasi dan *loss* Model 4

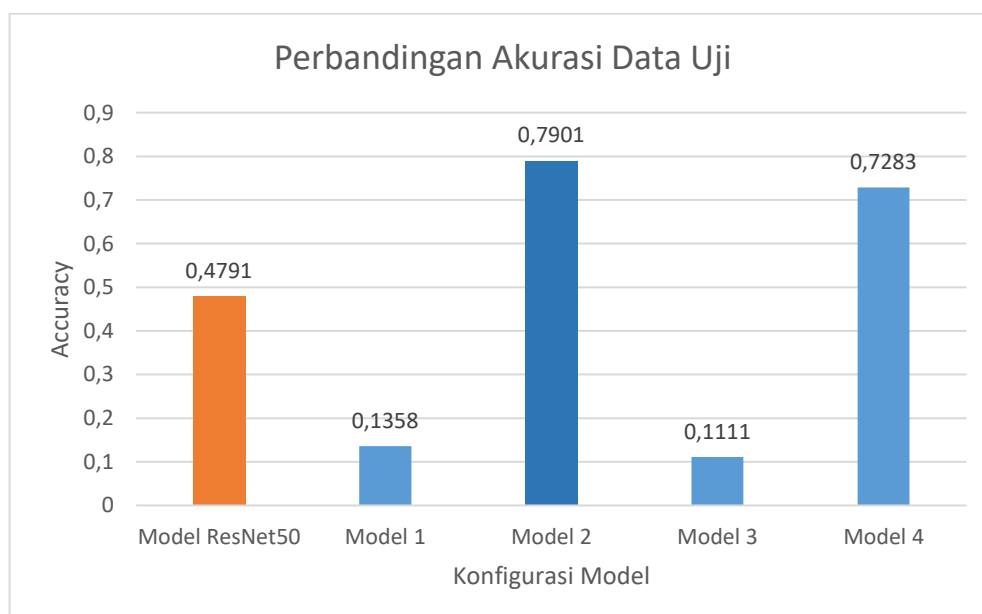
Berdasarkan Tabel V-3, Model 2 memiliki nilai akurasi dan *loss* terbaik. Model 2, skenario yang menggunakan arsitektur CNN model InceptionV3 dengan ArcFace, memiliki akurasi dan *loss* yang juga lebih baik dibandingkan model acuan, Model ResNet-50 (Mandal et al., 2021). Hal ini dikarenakan penggunaan ArcFace mampu memperbesar jarak batas antar kelas, sehingga fitur model dari setiap kelas menghasilkan akurasi klasifikasi yang tinggi. Model 2 memiliki akurasi data latih lebih baik 36,33% dibandingkan model acuan dan memiliki akurasi data validasi lebih baik 12,65% dibandingkan model acuan. Kemudian, *loss* yang didapatkan Model 2 terhadap data latih dan data validasi juga lebih baik dibandingkan model acuan. Nilai *loss* Model 2 terhadap data latih dan data validasi tersebut masing-masing adalah 0,1357 dan 2,3069.

Berdasarkan grafik akurasi dan *loss* pada Gambar V-1 sampai dengan Gambar V-4, Gambar V-2 yang merupakan kurva akurasi dan *loss* Model 2, menunjukkan kurva akurasi dan *loss* terbaik dibandingkan grafik model-model lainnya. Gambar V-1 dan Gambar V-3 menunjukkan model tersebut *underfitting*, dikarenakan menunjukkan nilai akurasi dan *loss* yang tidak memiliki peningkatan yang signifikan dan/atau stabil. Kemudian, Gambar V-4 menunjukkan model 4 *overfitting*. Hal ini ditunjukkan pada kurva *loss* pada data validasi semakin meningkat pada setiap *epoch*-nya.

Kemudian, Tabel V-4 menunjukkan perbandingan performa model pada data uji. Metrik evaluasi yang dijadikan tolak ukur antara lain *accuracy*, *recall*, *precision*, dan *F-1 score*.

Tabel V-4. Perbandingan Hasil *Testing* Model

| No | Konfigurasi Model | <i>Accuracy</i> | <i>Recall</i> | <i>Precision</i> | <i>F-1 Score</i> |
|----|---|-----------------|---------------|------------------|------------------|
| 0 | ResNet-50 (Mandal et al., 2021) | 0,4791 | 0,4719 | 0,4613 | 0,4473 |
| 1 | Model 1: ResNet-50 + ArcFace | 0,1358 | 0,6471 | 0,1467 | 0,2392 |
| 2 | Model 2: InceptionV3 + ArcFace | 0,7901 | 0,9142 | 0,8533 | 0,8827 |
| 3 | Model 3: ResNet-50 + ArcFace | 0,1111 | 0,4500 | 0,1323 | 0,2044 |
| 4 | Model 4: InceptionV3 + ArcFace | 0,7283 | 0,8428 | 0,8676 | 0,8550 |



Gambar V-5. Diagram Batang Perbandingan Akurasi Data Uji

Berdasarkan data yang ditunjukkan pada Tabel V-4 dan Gambar V-5, model 2 memiliki nilai yang lebih baik juga dibandingkan dengan model (Mandal et al., 2021) berdasarkan metrik evaluasi yang digunakan. Model 2 memiliki *test accuracy* sebesar 0,7901, lebih baik 0,3110 dibandingkan model acuan. Kemudian, *recall* Model 2 sebesar 0,9142, lebih baik 0,4423 dibandingkan model acuan. Model 2 memiliki *precision* sebesar 0,8533, lebih baik 0,3920 dibanding model acuan. Pada pengukuran menggunakan metrik evaluasi *f-1 score*, Model 2 memiliki nilai sebesar 0,8827, juga lebih baik dibandingkan model acuan, selisih nilainya mencapai 0,4354. Oleh sebab itu, maka Model 2 terbukti memiliki kinerja yang lebih baik secara keseluruhan dibandingkan dengan model acuan berdasarkan metrik evaluasi.

5.4 Kesimpulan

Terdapat 4 model CNN yang dibangun berdasarkan 4 skenario percobaan dengan konfigurasi berbeda. Hasil dari perbandingan performa model menunjukkan Model 2 adalah model terbaik. Akurasi Model 2 dari data latih dan data validasi masing-masing mencapai 96,38% dan 60,56%. Kemudian, *loss* yang didapatkan terhadap data latih dan data validasi masing-masing adalah 0,1357 dan 2,3069. Selain itu, Model 2 yang dihasilkan juga memiliki kinerja yang baik dibandingkan model acuan berdasarkan metrik evaluasi berupa *accuracy* sebesar 79,01%, *recall* sebesar 91,42%, *precision* sebesar 85,33%, dan *f-1 score* sebesar 88,27%. Oleh karena itu, penggunaan ArcFace terbukti mampu mendapatkan fitur yang diskriminatif dari dataset yang perbedaan fitur antar kelasnya tidak signifikan.

BAB VI

KESIMPULAN DAN SARAN

6.1 Pendahuluan

Bab ini membahas mengenai kesimpulan berdasarkan hasil dan analisis penelitian yang dilakukan dan saran yang dapat dijadikan sebagian acuan atau rujukan untuk penelitian selanjutnya.

6.2 Kesimpulan

Kesimpulan yang diperoleh setelah dilakukan analisis pada hasil penelitian pada bab sebelumnya adalah sebagai berikut:

1. Perangkat lunak sistem pengenalan citra wajah bermasker menggunakan berbagai model CNN, yaitu InceptionV3 dan ResNet-50 dengan *additive angular margin loss* berhasil dikembangkan dan dapat mengklasifikasikan identitas citra wajah bermasker.
2. Model terbaik yang dihasilkan yaitu Model InceptionV3 menggunakan *additive angular margin loss* (ArcFace). ArcFace mampu memperbesar jarak batas antar kelas, sehingga fitur model dari setiap kelas menghasilkan akurasi klasifikasi yang tinggi. Model tersebut memiliki akurasi lebih besar 31,10% dibandingkan model penelitian sebelumnya tanpa menggunakan ArcFace (Mandal et al., 2021).

6.3 Saran

Berikut saran untuk penelitian selanjutnya adalah sebagai berikut:

1. Gunakan dataset yang lebih banyak agar model CNN dapat mempelajari fitur lebih baik.
2. Diperlukan studi yang lebih luas pada konfigurasi *layer* sehingga mendapat model CNN yang *fine-tuning* atau model dengan kurva akurasi dan *loss* yang lebih baik.

DAFTAR PUSTAKA

- Alzu'bi, A., Albalas, F., Al-Hadhrami, T., Younis, L. B., & Bashayreh, A. (2021). Masked face recognition using deep learning: A review. In *Electronics (Switzerland)* (Vol. 10, Issue 21). MDPI. <https://doi.org/10.3390/electronics10212666>
- Anwar, A., & Raychowdhury, A. (2020). *Masked Face Recognition for Secure Authentication*. <http://arxiv.org/abs/2008.11104>
- Boutros, F., Damer, N., Kirchbuchner, F., & Kuijper, A. (2021). *Self-restrained Triplet Loss for Accurate Masked Face Recognition*. <http://arxiv.org/abs/2103.01716>
- Chandra, B. S., Sastry, C. S., Jana, S., & Patidar, S. (2017). Atrial fibrillation detection using convolutional neural networks. *Computing in Cardiology*, 44, 1–4. <https://doi.org/10.22489/CinC.2017.163-226>
- Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2018). *ArcFace: Additive Angular Margin Loss for Deep Face Recognition*. <https://github.com/>
- Ejaz, M. S., Islam, M. R., Sifatullah, M., & Sarker, A. (2019, May 1). Implementation of Principal Component Analysis on Masked and Non-masked Face Recognition. *1st International Conference on Advances in Science, Engineering and Robotics Technology 2019, ICASERT 2019*. <https://doi.org/10.1109/ICASERT.2019.8934543>
- Hakim, Z., & Rizky, R. (2020). Analisis Perancangan Sistem Informasi Pembuatan Paspor Di Kantor Imigrasi Bumi Serpong Damai Tangerang Banten Menggunakan Metode Rational Unified Process. *Jutis (Jurnal Teknik Informatika)*, 6(2), 103–112. <http://ejournal.unis.ac.id/index.php/jutis/article/view/135>
- Hariri, W. (2022). Efficient masked face recognition method during the COVID-19 pandemic. *Signal, Image and Video Processing*, 16(3), 605–612. <https://doi.org/10.1007/s11760-021-02050-w>
- Jignesh Chowdary, G., Pun, N. S., Sonbhadra, S. K., & Agarwal, S. (2020). Face Mask Detection Using Transfer Learning of InceptionV3. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12581 LNCS, 81–90. https://doi.org/10.1007/978-3-030-66665-1_6
- Khan, H. A., Jue, W., Mushtaq, M., & Mushtaq, M. U. (2020). Brain tumor classification in MRI image using convolutional neural network. *Mathematical Biosciences and Engineering*, 17(5), 6203–6216. <https://doi.org/10.3934/MBE.2020328>
- Li, L., Mu, X., Li, S., & Peng, H. (2020). A Review of Face Recognition Technology. *IEEE Access*, 8, 139110–139120. <https://doi.org/10.1109/ACCESS.2020.3011028>

- Li, Y., Guo, K., Lu, Y., & Liu, L. (2021). Cropping and attention based approach for masked face recognition. *Applied Intelligence*, 51(5), 3012–3025. <https://doi.org/10.1007/s10489-020-02100-9>
- Lin, C., Li, L., Luo, W., Wang, K. C. P., & Guo, J. (2019). Transfer learning based traffic sign recognition using inception-v3 model. *Periodica Polytechnica Transportation Engineering*, 47(3), 242–250. <https://doi.org/10.3311/PPtr.11480>
- Mandal, B., Okeukwu, A., & Theis, Y. (2021). *Masked Face Recognition using ResNet-50*. <http://arxiv.org/abs/2104.08997>
- Montero, D., Nieto, M., Leskovsky, P., & Aginako, N. (2021). *Boosting Masked Face Recognition with Multi-Task ArcFace*. <http://arxiv.org/abs/2104.09874>
- Mukti, I. Z., & Biswas, D. (2019). Transfer Learning Based Plant Diseases Detection Using ResNet50. *2019 4th International Conference on Electrical Information and Communication Technology, EICT 2019, December*, 1–6. <https://doi.org/10.1109/EICT48899.2019.9068805>
- Novandya, A. (2017). Penerapan Algoritma Klasifikasi Data Mining C4.5 pada Dataset Cuaca Wilayah Bekasi. *KNiST*, 368–372.
- Nurmaini, S., Tondas, A. E., Darmawahyuni, A., Rachmatullah, M. N., Umi Partan, R., Firdaus, F., Tutuko, B., Pratiwi, F., Juliano, A. H., & Khoirani, R. (2020). Robust detection of atrial fibrillation from short-term electrocardiogram using convolutional neural networks. *Future Generation Computer Systems*, 113, 304–317. <https://doi.org/10.1016/j.future.2020.07.021>
- Pang, B., Nijkamp, E., & Wu, Y. N. (2020). Deep Learning With TensorFlow: A Review. *Journal of Educational and Behavioral Statistics*, 45(2), 227–248. <https://doi.org/10.3102/1076998619872761>
- Sanchez, S. A., Romero, H. J., & Morales, A. D. (2020). A review: Comparison of performance metrics of pretrained models for object detection using the TensorFlow framework. *IOP Conference Series: Materials Science and Engineering*, 844(1). <https://doi.org/10.1088/1757-899X/844/1/012024>
- Saputro, I. W., & Sari, B. W. (2020). Uji Performa Algoritma Naïve Bayes untuk Prediksi Masa Studi Mahasiswa. *Creative Information Technology Journal*, 6(1), 1. <https://doi.org/10.24076/citec.2019v6i1.178>
- Shafiee, S., Wautelet, Y., Hvam, L., Sandrin, E., & Forza, C. (2020). Scrum versus Rational Unified Process in facing the main challenges of product configuration systems development. *Journal of Systems and Software*, 170, 110732. <https://doi.org/10.1016/j.jss.2020.110732>
- Tyagi, V. (2018). Understanding Digital Image Processing. *Understanding Digital Image Processing, November*. <https://doi.org/10.1201/9781315123905>
- Wang, Z., Wang, G., Huang, B., Xiong, Z., Hong, Q., Wu, H., Yi, P., Jiang, K., Wang, N., Pei, Y., Chen, H., Miao, Y., Huang, Z., & Liang, J. (2020). *Masked Face Recognition Dataset and Application*. <http://arxiv.org/abs/2003.09093>

- Wu, Y., & Ji, Q. (2019). Facial Landmark Detection: A Literature Survey. *International Journal of Computer Vision*, 127(2), 115–142. <https://doi.org/10.1007/s11263-018-1097-z>
- Xiong, Z., Stiles, M. K., & Zhao, J. (2017). Robust ECG signal classification for detection of atrial fibrillation using a novel neural network. *Computing in Cardiology*, 44(Figure 1), 1–4. <https://doi.org/10.22489/CinC.2017.066-138>
- Zhang, H., Li, Q., Sun, Z., & Liu, Y. (2018). Combining Data-Driven and Model-Driven Methods for Robust Facial Landmark Detection. *IEEE Transactions on Information Forensics and Security*, 13(10), 2409–2422. <https://doi.org/10.1109/TIFS.2018.2800901>

LAMPIRAN

Lampiran 1. Kode Program

```
app.py
from flask import Flask, render_template, Response, request
from support import Support
import cv2 as cv

class App:

    def __init__(self, name):
        self.app = Flask(name)

        @self.app.route('/')
        def index():
            return self.__index()

        @self.app.route('/recognition', methods=['POST',
'GET'])
        def recognition():
            return self.__recognition()

    def __index(self):
        return render_template('index.html')

    def __recognition(self):
        predict = None
        if request.method == 'POST':
            img = request.files['face']
            img.save('static/uploaded.jpg', 0)

            image = cv.imread('static/uploaded.jpg',
cv.IMREAD_COLOR)

            predict = Support().face_recognition(image)
            return render_template('recognition.html',
predict=predict)

            return render_template('recognition.html',
predict=predict)

    def run(self):
        self.app.run(debug=True)

def main():
    app = App(__name__)
    app.run()
```

```
if __name__ == '__main__':  
    main()
```

support.py

```
import cv2 as cv  
import numpy as np  
import dlib  
from tensorflow.keras.models import load_model  
  
class Support(object):  
  
    def extract_feat(self, image):  
        detector = dlib.get_frontal_face_detector()  
        predictor =  
dlib.shape_predictor('model/shape_predictor_68_face_landmarks  
.dat')  
        shape = 0  
        rects = detector(image, 1)  
  
        for rect in rects:  
            shape = predictor(image, rect)  
            shape_np = np.zeros((68, 2), dtype="int")  
            for i in range(0, 68):  
                shape_np[i] = (shape.part(i).x,  
shape.part(i).y)  
            shape = shape_np  
  
            for i, (x, y) in enumerate(shape):  
                cv.circle(image, (x, y), 1, (0, 0, 255), -1)  
        return image  
  
    def face_recognition(self, image):  
        IMG_SIZE = 180  
        img_feat = self.extract_feat(image)  
        fix_img = cv.resize(img_feat, (IMG_SIZE, IMG_SIZE),  
interpolation=cv.INTER_AREA)  
        fix_img = fix_img / 255  
        fix_img = fix_img.reshape(-1, IMG_SIZE, IMG_SIZE, 3)  
        model =  
load_model('model/inceptionV3_model_config1.h5')  
        pred = model.predict(fix_img)  
        label_map = ['duhaitao', 'houminghao', 'jingtian',  
'linxinru', 'luhan', 'masu', 'matianyu', 'wuyifan',  
'xuezhiquan', 'xuweizhou', 'yangyang', 'yuanshanshan',  
'zhangruoyun', 'zhangyixing', 'zhangyuxi']  
        pred = np.argmax(pred)  
        final_pred = label_map[pred]  
        return final_pred
```

base.html

```
<!doctype html>  
<html lang="en">  
    <head>  
        <meta charset="utf-8">
```

```

<meta name="viewport" content="width=device-width,
initial-scale=1">
<title>Masked-Face Recognition App</title>

<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/b
ootstrap.min.css" rel="stylesheet">
  <link rel="stylesheet" href="{{ url_for('static',
filename='css/style.css') }}">

  <script src="https://kit.fontawesome.com/1d7a98a84b.js"
crossorigin="anonymous"></script>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/boo
tstrap.bundle.min.js"></script>

</head>
<body>

  <div>
    <div class="text-center">
      {% block heading %}

      {% endblock %}
    </div>
  </div>

  <div class="container">
    {% block content %}

    {% endblock %}
  </div>

  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquer
y.min.js"></script>
  <script
src="https://unpkg.com/aos@next/dist/aos.js"></script>
  <script src="{{url_for('static',
filename='js/script.js')}}" "></script>

</body>
</html>

```

index.html

```

{% extends 'base.html' %}

{% block heading %}
  
{% endblock %}

{% block content %}

```

```
<h2 class="text-center"><i>Additive Angular Margin Loss</i>
untuk Pengenalan Citra Wajah Bermasker</h2>
<div class="text-center"><a class="btn btn-primary" href="{{
url_for('recognition') }}">Start</a></div>
{% endblock %}
```

recognition.html

```
{% extends 'base.html' %}

{% block heading %}
    
{% endblock %}

{% block content %}
    <div class="container">
        <div>
            <h1 class="title-two">Choose a Picture</h1>
            <form action="" method="POST"
enctype="multipart/form-data">
                <input class="form-control" type="file"
name="face" accept="images/*" required />
                <div class="text-center"><button
type="submit" class="btn btn-info mt-3 fw-
bold">Predict</button></div>
            </form>
        </div>

        <div id="menu">
            {% if predict != None %}
                <div>
                    <div class="box-content primary">
                        
                        <br> The Identity is: {{ predict }}
                    </div>
                </div>
            {% elif predict == None %}
            {% endif %}
        </div>
    </div>
{% endblock %}
```

style.css

```
:root {
    --blue1: #EDF9FC;
    --blue2: #D1F4FA;
    --blue3: #53CDE2;
    --black: #333333;
    --white: #FFFFFF;
    --success: #99FF99;
    --fail: #990000;
```

```

--font1: "roboto";
--font2: "montserrat";
}

/*Font*/
@font-face {
  font-family: 'roboto';
  src: url('font/roboto.woff') format('woff');
  font-weight: normal;
  font-style: normal;
}

@font-face {
  font-family: 'montserrat';
  src: url('font/montserrat.woff') format('woff');
  font-weight: normal;
  font-style: normal;
}

@media (min-width: 577px){
  body {
    margin: 0;
    padding: 0;
    font-family: var(--font1);
    font-size: 16px;
    color: var(--black);
  }

  .head{
    background-image: linear-gradient(180deg, var(--
blue2), var(--blue3));
  }

  .head img{
    width: 300px !important;
    margin-bottom: 48px;
  }

  .content{
    position: relative;
    top: -20px;
    background-color: var(--white);
    border-top-left-radius: 25px;
    border-top-right-radius: 25px;
    overflow: auto;
  }

  .title-one, .title-two{
    font-family: var(--font2);
    font-weight: bold;
    text-align: center;
    margin-top: 30px;
  }
}

```

```

.title-two{
    font-size: 20px;
}

.box-content{
    background-color: var(--blue1);
    border-radius: 10px;
    padding: 20px;
    text-align: justify;
    margin: 20px auto;
    font-size: 18px;
    width: 1000px;
}

.padding-content{
    padding: 90px !important;
}

.padding-content2{
    padding: 50px !important;
}

.box-content.important{
    background-color: var(--blue2);
}

.box-content.success{
    background-color: var(--success);
    font-weight:bold;
}

.box-content.fail{
    background-color: var(--fail);
    font-weight:bold;
    color: var(--white);
}

.box-content img{
    width: 50px;
}

.box-content.icon-tips{
    width:auto !important;
}

.box-content.icon-tips img{
    width:100px !important;
}

.item-number{
    font-size: 100px;
    text-align: center;
    color: var(--blue3);
}

```

```

.icon-detection{
    width: 100px !important;
}

#menu img{
    width: 150px !important;
}

form{
    width: 1000px;
    margin: 20px auto;
}

.img-tips{
    height: 112.27px !important;
    line-height: 112.27px !important;
}
}

@media (max-width: 576px){
    body {
        margin: 0;
        padding: 0;
        font-family: 'roboto';
        font-size: 14px;
        color: var(--black);
    }

    /* NAVBAR */
    .navbar {
        background-color: var(--blue2);
        margin: 0 auto;
        border-top-left-radius: 15px;
        border-top-right-radius: 15px;
        padding: 15px 0;
    }

    .navbar .nav-item .nav-link {
        color: var(--white);
        font-size: 25px;
        background-color: var(--blue3);
        border-radius: 10px;
        width: 50px;
        margin: 0 auto !important;
    }

    .main-menu{
        position: absolute;
        top: -20px;
        font-size: 30px !important;
    }
}

```

```

.navbar .nav-item .nav-link:hover,
.navbar .nav-item .nav-link.active {
  color: var(--blue3);
  background-color: var(--blue1);
}

.head{
  background-size: cover;
  height: 220px;
}

.content{
  position: relative;
  top: -20px;
  height: 460px;
  background-color: var(--white);
  border-top-left-radius: 25px;
  border-top-right-radius: 25px;
  overflow: auto;
}

.title-one{
  font-family: var(--font2);
  font-weight: bold;
  text-align: center;
  margin-top: 20px;
}

.title-two{
  font-family: var(--font2);
  font-weight: bold;
  text-align: left;
  margin-top: 20px;
}

.box-content{
  background-color: var(--blue1);
  border-radius: 10px;
  padding: 10px;
  text-align: justify;
  margin: 20px 0;
}

.box-content.important{
  background-color: var(--blue2);
}

.box-content.success{
  background-color: var(--success);
  font-weight: bold;
}

.box-content.fail{
  background-color: var(--fail);
}

```



```

        font-weight:bold;
        color: var(--white);
    }

    .item-number{
        font-size: 60px;
        text-align: center;
        color: var(--blue3);
    }
}

```

data_preparation.ipynb

```

# -*- coding: utf-8 -*-
"""data-preparation.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/14OaWjonDlW6W-
    CNhFaRyxVyc0FzGDr69
"""

import os
import sys
import glob
import shutil

# Commented out IPython magic to ensure Python compatibility.
from sklearn.model_selection import train_test_split
import zipfile
import numpy as np
from google.colab import files
from keras.preprocessing import image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
# %matplotlib inline

"""## **Data Preparation**"""

base_dir = '/content/drive/MyDrive/Thesis/Datasets/'
data_dir = os.path.join(base_dir, 'AFDB_masked_face_dataset')
filtered_data_dir = os.path.join(base_dir,
    '20_masked_face_dataset')

# os.mkdir(filtered_data_dir)

face_dict = dict()
for root_dir, cur_dir, files in os.walk(data_dir):
    if (len(files) >= 20):
        name = root_dir.replace(data_dir + '/', '')
        face_dict[name] = len(files)
        temp_dir = os.path.join(data_dir, name)
        face_dir = os.path.join(filtered_data_dir, name)

```

```

    # os.mkdir(face_dir)
    # for f in files:
    #     shutil.copy(os.path.join(temp_dir, f),
    os.path.join(face_dir, f))
    print(face_dir, len(files))

shutil.rmtree('/content/drive/MyDrive/Thesis/Datasets/20_mask
ed_face_dataset/yiyangqianxi')

#deleted this data due to the person isn't included in non-
masked face dataset

del face_dict['yiyangqianxi']
print(len(face_dict))

print(face_dict)

"""## **Split Data**"""

data_dir = os.path.join(base_dir, 'data7030')
train_dir = os.path.join(data_dir, 'train')
test_dir = os.path.join(data_dir, 'test')

os.mkdir(data_dir)
os.mkdir(train_dir)
os.mkdir(test_dir)

print(filtered_data_dir)

for root_dir, cur_dir, files in os.walk(filtered_data_dir):
    if(root_dir == filtered_data_dir):
        continue
    name = root_dir.replace(filtered_data_dir + '/', '')

    tr_name = os.path.join(train_dir, name)
    te_name = os.path.join(test_dir, name)
    os.mkdir(tr_name)
    os.mkdir(te_name)

    train_face_dir, test_face_dir =
train_test_split(os.listdir(os.path.join(filtered_data_dir,
name)), test_size = 0.30)
    for tr in train_face_dir:
        shutil.copy(os.path.join(filtered_data_dir, name, tr),
os.path.join(tr_name, tr))
    for te in test_face_dir:
        shutil.copy(os.path.join(filtered_data_dir, name, te),
os.path.join(te_name, te))

for root_dir, cur_dir, files in os.walk(data_dir):
    print(root_dir, len(files))

"""## **Split Data**
Non-masked face dataset

```

```

"""

data_dir = os.path.join(base_dir, 'ndata7030')
train_dir = os.path.join(data_dir, 'train')
test_dir = os.path.join(data_dir, 'test')

os.mkdir(data_dir)
os.mkdir(train_dir)
os.mkdir(test_dir)

filtered_data2_dir = os.path.join(base_dir,
'20_face_dataset')

for root_dir, cur_dir, files in os.walk(filtered_data2_dir):
    if(root_dir == filtered_data2_dir):
        continue
    name = root_dir.replace(filtered_data2_dir + '/', '')

    tr_name = os.path.join(train_dir, name)
    te_name = os.path.join(test_dir, name)
    os.mkdir(tr_name)
    os.mkdir(te_name)

    train_face_dir, test_face_dir =
train_test_split(os.listdir(os.path.join(filtered_data2_dir,
name)), test_size = 0.30)
    for tr in train_face_dir:
        shutil.copy(os.path.join(filtered_data2_dir, name, tr),
os.path.join(tr_name, tr))
    for te in test_face_dir:
        shutil.copy(os.path.join(filtered_data2_dir, name, te),
os.path.join(te_name, te))

"""## **Split Data (Hybrid)**
Masked + Non-masked face dataset
"""

data_dir = os.path.join(base_dir, 'hybdata7030')
train_dir = os.path.join(data_dir, 'train')
test_dir = os.path.join(data_dir, 'test')

os.mkdir(data_dir)
os.mkdir(train_dir)
os.mkdir(test_dir)

filtered_data2_dir = os.path.join(base_dir,
'20_face_dataset')

print(filtered_data_dir)
print(filtered_data2_dir)

for root_dir, cur_dir, files in os.walk(filtered_data_dir):
    if(root_dir == filtered_data_dir):

```

```

        continue
    name = root_dir.replace(filtered_data_dir + '/', '')

    tr_name = os.path.join(train_dir, name)
    te_name = os.path.join(test_dir, name)
    os.mkdir(tr_name)
    os.mkdir(te_name)

    train_face_dir, test_face_dir =
train_test_split(os.listdir(os.path.join(filtered_data_dir,
name)), test_size = 0.30)
    for tr in train_face_dir:
        shutil.copy(os.path.join(filtered_data_dir, name, tr),
os.path.join(tr_name, tr))
    for te in test_face_dir:
        shutil.copy(os.path.join(filtered_data_dir, name, te),
os.path.join(te_name, te))

for root_dir, cur_dir, files in os.walk(filtered_data2_dir):
    if(root_dir == filtered_data2_dir):
        continue
    name = root_dir.replace(filtered_data2_dir + '/', '')

    tr_name = os.path.join(train_dir, name)
    te_name = os.path.join(test_dir, name)
    # os.mkdir(tr_name)
    # os.mkdir(te_name)

    train_face_dir, test_face_dir =
train_test_split(os.listdir(os.path.join(filtered_data2_dir,
name)), test_size = 0.30)
    for tr in train_face_dir:
        shutil.copy(os.path.join(filtered_data2_dir, name, tr),
os.path.join(tr_name, tr))
    for te in test_face_dir:
        shutil.copy(os.path.join(filtered_data2_dir, name, te),
os.path.join(te_name, te))

for root_dir, cur_dir, files in os.walk(filtered_data_dir):
    print(root_dir, len(files))

for root_dir, cur_dir, files in os.walk(filtered_data2_dir):
    print(root_dir, len(files))

for root_dir, cur_dir, files in os.walk(data_dir):
    print(root_dir, len(files))

```

models.ipynb

```

# -*- coding: utf-8 -*-
"""models-mdata-aaml.ipynb

```

Automatically generated by Colaboratory.

```

Original file is located at
https://colab.research.google.com/drive/1i6e8aALCWD0tTXWtiyTl1d4i-t8B68HJ
"""

import os

import tensorflow as tf
from tensorflow.keras.models import Sequential

from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.applications.inception_v3 import InceptionV3

base_dir = '/content/drive/MyDrive/Thesis/Datasets/'
data_dir = os.path.join(base_dir, 'data8020')
train_dir = os.path.join(data_dir, 'train')
test_dir = os.path.join(data_dir, 'test')

train_datagen =
tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./25
5,

rotation_range=20,

zoom_range=0.05,

horizontal_flip=True,

fill_mode="nearest",

validation_split=0.25)

train_generator =
train_datagen.flow_from_directory(train_dir,

target_size=(180, 180),

batch_size=32,

class_mode='categorical',

subset='training',

shuffle=True,

seed=42)

val_generator = train_datagen.flow_from_directory(train_dir,

target_size=(180, 180),

batch_size=32,

```

```

class_mode='categorical',

subset='validation',

shuffle=True,

seed=42)

import numpy as np
train_generator.reset()
x1=np.concatenate([train_generator.next()[0] for i in
range(train_generator.__len__())])
y1=np.concatenate([train_generator.next()[1] for i in
range(train_generator.__len__())])
print(x1.shape)
print(y1.shape)

val_generator.reset()
x2=np.concatenate([val_generator.next()[0] for i in
range(val_generator.__len__())])
y2=np.concatenate([val_generator.next()[1] for i in
range(val_generator.__len__())])
print(x2.shape)
print(y2.shape)

"""##**Additive Angular Margin Loss**"""

import tensorflow.keras.layers as L
import tensorflow.keras.backend as K
from tensorflow.keras import regularizers

__all__ = [
    "ArcFace"
]

"""
ArcFace: Additive Angular Margin Loss for Deep Face
Recognition (CVPR '19)
Jiankang Deng, Jia Guo, Niannan Xue, Stefanos Zafeiriou
https://arxiv.org/abs/1801.07698
Implementation inspired by repo:
https://github.com/4uiiurz1/keras-arcface
additive angular margin loss:

$$L = -\log(e^{s(\cos(\theta_{y_i, i}) + m)} / (e^{s(\cos(\theta_{y_i, i}) + m)} + \sum(e^{s(\cos(\theta_{j, i}) + m)})))$$


$$W = W^* / ||W^*||$$


$$x = x^* / ||x^*||$$


$$\cos(\theta_{j, i}) = W_j.T * x_i$$

"""

class ArcFace(L.Layer):
    def __init__(self, num_classes=10, scale=30.0,

```

```

margin=0.50, **kwargs):
    super(ArcFace, self).__init__(**kwargs)
    self.num_classes = num_classes
    self.scale = scale
    self.margin = margin

    def build(self, input_shape):
        self.W = self.add_weight(name='W',
                                shape=(input_shape[0][-1],
self.num_classes),
                                initializer='glorot_uniform',
                                trainable=True)

    def call(self, inputs):
        # get embeddings and one hot labels from inputs
        embeddings, onehot_labels = inputs
        # normalize final W layer
        W = tf.nn.l2_normalize(self.W, axis=0)
        # get logits from multiplying embeddings (batch_size,
embedding_size) and W (embedding_size, num_classes)
        logits = tf.matmul(embeddings, W)
        # clip logits to prevent zero division when backward
        theta = tf.acos(K.clip(logits, -1.0 + K.epsilon(),
1.0 - K.epsilon()))
        # subtract margin from logits
        target_logits = tf.cos(theta + self.margin)
        # get cross entropy
        logits = logits * (1 - onehot_labels) + target_logits
* onehot_labels
        # apply scaling
        logits = logits * self.scale
        # get class probability distribution
        predictions = tf.nn.softmax(logits)
        return predictions

    def compute_output_shape(self, input_shape):
        return (None, self.num_classes)

    def get_config(self):
        config = super().get_config().copy()
        config.update({
            "num_classes": self.num_classes,
            "scale": self.scale,
            "margin": self.margin})
        return config

"""## **ResNet50**"""

pretrained_model = ResNet50(include_top=False,
                             input_shape=(180,180,3),
                             classes=15,
                             weights='imagenet')

```

```

for layer in pretrained_model.layers:
    layer.trainable=False

resnet50_output = pretrained_model.layers[-1].output

label = tf.keras.layers.Input(shape=(15,))
x = tf.keras.layers.Dropout(0.4)(resnet50_output)
x = tf.keras.layers.Flatten()(x)
x = tf.keras.layers.Dropout(0.4)(x)
x = ArcFace(15)([x, label])
# Create the complete model by using the Model class

resnet50_model =
tf.keras.Model(inputs=[pretrained_model.input, label],
outputs=x)

model_path = '/content/drive/MyDrive/Thesis/models'

file_name = 'config4.h5'

checkpoint_path = os.path.join(model_path, 'checkpoint',
file_name)
print(checkpoint_path)

call_back =
tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,
monitor='val_accuracy',
                                verbose=1,

save_freq='epoch',

save_best_only=True,

save_weights_only=False,
                                mode='max')

resnet50_model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.002),
                        loss='categorical_crossentropy',
                        metrics=['accuracy'],)

resnet50_model.summary()

resnet50_history = resnet50_model.fit(
    [x1,y1],
    y1,
    epochs = 25,
    validation_data=([x2,y2], y2),
    verbose = 1,
    callbacks = call_back)

"""## **InceptionV3**"""

```



```

!wget --no-check-certificate \
    https://storage.googleapis.com/mledu-
datasets/inception_v3_weights_tf_dim_ordering_tf_kernels_noto
p.h5 \
    -O
/tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5

local_weights_file =
'/tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h
5'

def create_pre_trained_model(local_weights_file):

    pre_trained_model = InceptionV3(input_shape = (180, 180,
3),
                                include_top = False,
                                weights = None)

    pre_trained_model.load_weights(local_weights_file)

    # Make all the layers in the pre-trained model non-
trainable
    for layer in pre_trained_model.layers:
        layer.trainable = False

    return pre_trained_model

pre_trained_model =
create_pre_trained_model(local_weights_file)

pre_trained_model.summary()

def output_of_last_layer(pre_trained_model):

    last_desired_layer = pre_trained_model.get_layer('mixed7')
    print('last layer output shape: ',
last_desired_layer.output_shape)
    last_output = last_desired_layer.output
    print('last layer output: ', last_output)

    return last_output

last_output = output_of_last_layer(pre_trained_model)

def create_final_model(pre_trained_model, last_output):

    label = tf.keras.layers.Input(shape=(15,))
    x = tf.keras.layers.Dropout(0.4)(last_output)
    x = tf.keras.layers.Flatten()(x)
    x = tf.keras.layers.Dropout(0.4)(x)
    x = ArcFace(15)([x, label])

    # Create the complete model by using the Model class
    model = tf.keras.Model(inputs=[pre_trained_model.input,

```

```

label], outputs=x)

    # Compile the model

model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.002),
               loss='categorical_crossentropy',
               metrics=['accuracy'])

    return model

file_name = 'config4.h5'

checkpoint_path = os.path.join(model_path, 'checkpoint',
                                file_name)
print(checkpoint_path)

call_back =
tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,
monitor='val_accuracy',
                                   verbose=1,

save_freq='epoch',

save_best_only=True,

save_weights_only=False,
                                   mode='max')

inceptionV3_model = create_final_model(pre_trained_model,
last_output)

inceptionV3_history = inceptionV3_model.fit(
    [x1,y1],
    y1,
    epochs = 25,
    validation_data=([x2,y2], y2),
    verbose = 1,
    callbacks = call_back)

"""## **Visualization**"""

# Commented out IPython magic to ensure Python compatibility.
from google.colab import files
from keras.preprocessing import image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
# %matplotlib inline

class History:
    def __init__(self, history):
        self.history = history

```

```

def visualize(self):

    acc = self.history.history['accuracy']
    val_acc = self.history.history['val_accuracy']
    loss = self.history.history['loss']
    val_loss = self.history.history['val_loss']

    epochs = range(len(acc))

    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))

    ax1.plot(epochs, acc, 'r', label='Training accuracy')
    ax1.plot(epochs, val_acc, 'b', label='Validation
accuracy')
    ax1.set_title('Training and validation accuracy')
    ax1.legend()

    ax2.plot(epochs, loss, 'r', label='Training Loss')
    ax2.plot(epochs, val_loss, 'b', label='Validation Loss')
    ax2.set_title('Training and validation loss')
    ax2.legend()

    fig.show()

resnet50_vis = History(resnet50_history)
resnet50_vis.visualize()

inceptionV3_vis = History(inceptionV3_history)
inceptionV3_vis.visualize()

inceptionV3_model.save('/content/drive/MyDrive/Thesis/models/
inceptionV3_model_config2.h5')

resnet50_model.save('/content/drive/MyDrive/Thesis/models/res
net50_model_config2.h5')

inceptionV3_test1 =
tf.keras.models.load_model('/content/drive/MyDrive/Thesis/mod
els/inceptionV3_model_config2.h5',
custom_objects={'ArcFace':ArcFace})
resnet50_test1 =
tf.keras.models.load_model('/content/drive/MyDrive/Thesis/mod
els/resnet50_model_config2.h5',
custom_objects={'ArcFace':ArcFace})

test_datagen =
tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./25
5,

rotation_range=15,

horizontal_flip=True,

```

```

validation_split=0.25)

test_generator = train_datagen.flow_from_directory(test_dir,
target_size=(180, 180),
batch_size=32,
class_mode='categorical')

from sklearn.metrics import classification_report,
multilabel_confusion_matrix
import numpy as np

Y_pred = inceptionV3_test1.predict([x2, y2])
y_pred = np.argmax(Y_pred, axis=1)
print('Confusion Matrix')
print(multilabel_confusion_matrix(y2, y_pred))
print('Classification Report')
target_names = ['duhaitao', 'houminghao', 'jingtian',
'linxinru', 'luhan', 'masu', 'matianyu', 'wuyifan',
'xuezhiquan', 'xuweizhou', 'yangyang', 'yuanshanshan',
'zhangruoyun', 'zhangyixing', 'zhangyuxi']
print(classification_report(y2, y_pred,
target_names=target_names))

from sklearn.metrics import classification_report,
multilabel_confusion_matrix
import numpy as np

Y_pred = resnet50_test1.predict(test_generator,
test_generator.n//test_generator.batch_size)
y_pred = np.argmax(Y_pred, axis=1)
print('Confusion Matrix')
print(multilabel_confusion_matrix(test_generator.classes,
y_pred))
print('Classification Report')
target_names = ['duhaitao', 'houminghao', 'jingtian',
'linxinru', 'luhan', 'masu', 'matianyu', 'wuyifan',
'xuezhiquan', 'xuweizhou', 'yangyang', 'yuanshanshan',
'zhangruoyun', 'zhangyixing', 'zhangyuxi']
print(classification_report(test_generator.classes, y_pred,
target_names=target_names))

```