

**TUGAS AKHIR – IF184802**

**Pengenalan Pembuluh Darah pada Telapak Tangan menggunakan Ekstraksi Fitur Convolutional Neural Network dan Metode Klasifikasi Support Vector Machine**

**Nuzul Ristyantika Yuliana**

**NRP 05111540000078**

Dosen Pembimbing I

Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.

Dosen Pembimbing II

Dini Adni Navastara, S.Kom., M.Sc.

Departemen Informatika

Fakultas Teknologi Informasi dan Komunikasi

Institut Teknologi Sepuluh Nopember

Surabaya 2019



**TUGAS AKHIR – IF184802**

***Pengenalan Pembuluh Darah pada Telapak Tangan menggunakan Ekstraksi Fitur Convolutional Neural Network dan Metode Klasifikasi Support Vector Machine***

**Nuzul Ristyantika Yuliana**

**NRP 05111540000078**

**Dosen Pembimbing I**

**Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.**

**Dosen Pembimbing II**

**Dini Adni Navastara, S.Kom., M.Sc.**

**Departemen Informatika**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2019**

***(Halaman ini sengaja dikosongkan)***

****

**UNDERGRADUATE THESIS – IF184802**

**Palm Vein Recognition using Features Extraction Convolutional Neural Network and Classification Method Support Vector Machine**

**HENDRY WIRANTO**

**NRP 05111540000102**

**First Advisor**

**Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.**

**Second Advisor**

Dini Adni Navastara, S.Kom., M.Sc.

**Department of Informatics**

**Faculty of Information and Communication Technology**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2019**

***(Halaman ini sengaja dikosongkan)***

# LEMBAR PENGESAHAN

**PENGENALAN EKSPRESI WAJAH MENGGUNAKAN *WAVELET TRANSFORM* DAN *CONVOLUTIONAL NEURAL NETWORK***

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer

pada

Bidang Studi Komputasi Cerdas dan Visi

Program Studi S-1 Departemen Informatika

Fakultas Teknologi Informasi dan Komunikasi

Institut Teknologi Sepuluh Nopember

Oleh:

**HENDRY WIRANTO**

**NRP: 05111540000102**

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom. ........................

(NIP. 19751220 200112 2 002) (Pembimbing 1)

1. Dini Adni Navastara, S.Kom., M.Sc. .........................

(NIP. 19851017 201504 2 001) (Pembimbing 2)

**SURABAYA**

**Januari, 2019**

***(Halaman ini sengaja dikosongkan)***

**Pengenalan Pembuluh Darah pada Telapak Tangan menggunakan Ekstraksi Fitur Convolutional Neural Network dan Metode Klasifikasi Support Vector Machine**

|  |  |  |
| --- | --- | --- |
| **Nama Mahasiswa** | **:** | **Nuzul Ristyantika Yuliana** |
| **NRP** | **:** | **05111540000078** |
| **Jurusan** | **:** | **Informatika, FTIK-ITS** |
| **Dosen Pembimbing 1** | **:** | **Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.** | |
| **Dosen Pembimbing 2** | **:** | **Dini Adni Navastara, S.Kom., M.Sc.** | |

# ABSTRAK

*Machine learning telah menjadi bagian dari kehidupan sehari-hari bagi banyak orang. Salah satu pengaplikasian machine learning adalah pengenalan pembuluh darah pada telapak tangan. Pengenalan pembuluh darah telapak tangan manusia mengkategorikan gambar pembuluh darah menjadi satu dari banyak kelas kode pembuluh darah berdasarkan fitur gambar tersebut. Banyak perusahaan, badan riset dan universitas yang terus mengembangkan machine learning agar mendapat hasil yang lebih akurat dan cepat. Dari situlah lahir algoritma deep learning, yang merupakan bagian dari machine learning. Convolutional Neural Network (CNN) adalah salah satu deep neural network yang cocok digunakan untuk mengolah data yang berbentuk 2 dimensi, seperti gambar dan video.*

*Pada tugas akhir ini, penulis mengusulkan sebuah algoritma Inner Borer Tracing untuk mendapatkan ROI dari telapak tangan dan beberapa algoritma preprocessing yang meliputi Median Filter, Adaptive Histogram Equalization, Adaptive Noise Removal, Anistropic Diffusion Filter, Morfologi Closing. Tujuannya adalah untuk memperjelas pola pembuluh darah telapak tangan guna meningkatkan akurasi pengenalan pembuluh darah telapak tangan dengan metode ekstraksi fitur Convolutional Neural Network (Arsitektur AlexNet) dan metode klasifikasi Support Vector Machine. Data pelatihan dan uji coba yang digunakan diambil dari dataset “CASIA Multi-Spectral Palmprint V1.0” yang terdiri dari 600 data dengan 100 kelas yang berbeda berdasarkan kode untuk setiap individu. Metode Augmentasi Rotasi dan Scaling juga dilakukan untuk memperbanyak jumlah data pada setiap kelas setelah preprocessing gambar. Dan hasil uji coba terakhir didapatkan akurasi 89%.*

**Kata kunci:** *Convolutional Neural Network*, Dataset CASIA Multi-Spectral Palmprint V1.0, Inner Border Tracing, Support Vector Machine, Arsitektur AlexNet.

***FACIAL EXPRESSION RECOGNITION USING WAVELET TRANSFORM AND CONVOLUTIONAL NEURAL NETWORK***

|  |  |  |
| --- | --- | --- |
| **Student’s Name** | **:** | **Hendry Wiranto** |
| **Student’s ID** | **:** | **05111540000102** |
| **Department** | **:** | **Informatics, Faculty of ICT-ITS** |
| **First Advisor** | **:** | **Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.** |
| **Second Advisor** | **:** | **Dini Adni Navastara, S.Kom., M.Sc.** |

# ABSTRACT

*Machine learning has become a part of the daily life of people around the world. One of the application of machine learning is human facial expression recognition. Human facial expression recognition categorizes an image of facial expression into one of many facial expression classes based on the features extracted from the image. Many companies, researchers and universities keep improving the machine learning to get a better and faster result. And from those improvements, deep learning algorithm is born. Convolutional Neural Network (CNN) is one of the deep neural network that suitable to process 2 dimentional data like image and video.*

*In this undergraduate thesis, the images are transformed into Wavelet Domain using Wavelet Transform before being processed into the proposed network. The purpose of this method is to improve the accuracy of the human facial expression recognition using Convolutional Neural Network. The train and test data used in this thesis is taken from “Karolinska Directed Emotional Faces” (KDEF) dataset which contains human facial expression with 7 different expressions which will be the prediction labels of the human facial expression recognition. The preprocessing of the images include changing the image format to grayscale, changing the image resolution to 256x256 pixels, applying level 1 Discrete Wavelet Transform and applying data augmentation with horizontal reflection and zoom in. The final test accuracy is 89,6%.*

**Keywords: *Convolutional Neural Network, Human Facial Expression Recognition, Image Data,******Karolinska Directed Emotional Faces Dataset, Wavelet Transform.***

# KATA PENGANTAR

سْمِ للا ّبِ لا ْ للا ّبِ للا ّبِسْمِ

Segala puji syukur bagi Allah SWT yang telah melimpahkan rahmat dan anugerah-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul:

“**Pengenalan Pembuluh Darah pada Telapak Tangan menggunakan Ekstraksi Fitur Convolutional Neural Network dan Metode Klasifikasi Support Vector Machine**”

Pengerjaan tuags akhir ini menjadi suatu pengalaman yang baik bagi penulis. Penulis dapat memperoleh banyak pengalaman yang berharga dalam memperdalam dan meningkatkan keilmuan dalam bidang Informatika selama perkuliahan di Departmen Informatika ITS.

Terselesaikannya Tugas Akhir ini tidak terlepas dari bantuan dan dukungan banyak pihak, oleh karena itu melalui lembar ini penulis ingin mengucapkan terima kasih dan penghormatan kepada:

1. Allah SWT, karena atas izin-Nya lah penulis dapat menyelesaikan Tugas Akhir dengan baik.
2. Kedua orangtua penulis, dan anggota keluarga lainnya yang telah memberikan dukungan doa, moral, dan material kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
3. Dr.Eng. Chastine Fatichah, S. Kom., M. Kom. dan Dini Adni Navastara, S. Kom., M.Sc. selaku pembimbing I dan II yang telah membimbing dan memberikan motivasi, nasihat dan bimbingan dalam menyelesaikan Tugas Akhir ini.
4. Dr. Eng. Darlis Herumurti, S. Kom., M. Kom. selaku Ketua Departemen Informatika ITS dan seluruh dosen dan karyawan Departemen Informatika ITS yang telah memberikan ilmu dan pengalaman kepada penulis selama menjalani masa kuliah di Informatika ITS.
5. Admin-admin Laboratorium Komputasi Cerdas & Visi (KCV) yang telah membantu penulis dalam menyediakan fasilitas dalam penyelesaian Tugas Akhir ini.
6. Teman-teman dari grup Petualang spesialnya kepada Aditya Pratama sebagai teman terdekat penulis yang selalu memberi semangat dan dukungan moral kepada penulis.
7. Hendry Wiranto, Pradipta Baskara dan teman-teman USER TA KCV angkatan 2015 yang telah menghibur, menemani dan membantu penulis selama pengerjaan Tugas Akhir ini.
8. Rahandi Noor Pasha dan Rully Soelaiman sebagai penyemangat Adik-adik 2016 yang telah memberikan dukungan kepada penulis selama pengerjaan Tugas Akhir ini.
9. Kakak-kakak 2014 yang selalu memberikan dukungan kepada penulis selama pengerjaan Tugas Akhir ini.
10. Seluruh mahasiswa Informatika ITS angkatan 2015 yang telah menjadi teman penulis selama menjalani masa kuliah di Informatika ITS.
11. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa laporan Tugas Akhir ini masih memiliki banyak kekurangan. Oleh karena itu dengan segala kerendahan hati penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan penulis kedepannya. Selain itu, penulis berharap laporan Tugas Akhir ini dapat berguna bagi pembaca secara umum.

Surabaya, April 2019

# 

# DAFTAR ISI

LEMBAR PENGESAHAN vii

ABSTRAK ix

ABSTRACT xi

KATA PENGANTAR xiii

DAFTAR ISI xv

DAFTAR TABEL xix

DAFTAR KODE SUMBER xxi

DAFTAR GAMBAR xxiii

1 BAB I PENDAHULUAN 1

1.1 Latar Belakang 1

1.2 Rumusan Masalah 2

1.3 Batasan Permasalahan 2

1.4 Tujuan 2

1.5 Manfaat 3

1.6 Metodologi 3

1.6.1 Penyusunan Proposal Tugas Akhir 3

1.6.2 Studi Literatur 3

1.6.3 Implementasi Perangkat Lunak 3

1.6.4 Pengujian dan Evaluasi 4

1.6.5 Penyusunan Buku 4

1.7 Sistematika Penulisan Laporan 4

2 BAB II TINJAUAN PUSTAKA 7

2.1 Pengenalan Ekspresi Wajah Manusia 7

2.2 Multi Layer Perceptron 8

2.3 Convolutional Neural Network 10

2.3.1 Convolution Layer 11

2.3.2 Pooling Layer 11

2.3.3 Fully Connected Layer 12

2.3.4 ReLU Activation Function 12

2.3.5 Fungsi Softmax 13

2.3.6 Cross Entropy 13

2.3.7 Stochastic Gradient Descent 14

2.3.8 Adagrad 14

2.3.9 RMSProp 15

2.3.10 Adam 15

2.3.11 Dropout 16

2.3.12 Batch Normalization 16

2.3.13 Zero Padding 17

2.3.14 Merge Layer Concatenate 17

2.4 Wavelet Transform 18

2.5 Augmentasi Data 20

2.5.1 Refleksi Citra 20

2.5.2 Perbesaran Ukuran Gambar 20

2.6 Akurasi, Precision & Recall 21

2.7 Python 22

2.8 Keras 22

2.9 TensorFlow 22

2.10 OpenCV 22

2.11 PyWavelets 23

2.12 Numpy 23

2.13 Scikit-learn 23

2.14 Matplotlib 23

3 BAB III PERANCANGAN SISTEM 25

3.1 Perancangan Data 25

3.2 Desain Umum Sistem 26

3.2.1 Tahap Praproses Data 28

3.2.2 Tahap Pembangunan Arsitektur 31

3.2.3 Tahap Pelatihan dan Pengujian CNN 35

4 BAB IV IMPLEMENTASI 37

4.1 Lingkungan Implementasi 37

4.1.1 Perangkat Keras 37

4.1.2 Perangkat Lunak 37

4.2 Implementasi Praproses Data 37

4.2.1 Implementasi Perubahan Ukuran Gambar 38

4.2.2 Implementasi Discrete Wavelet Transform 38

4.3 Implementasi Augmentasi Data 39

4.3.1 Refleksi 39

4.3.2 Perbesar Ukuran Gambar 39

4.4 Implementasi Pembangunan Arsitektur 40

4.5 Implementasi Pelatihan dan Evaluasi CNN 44

5 BAB V UJI COBA DAN EVALUASI 47

5.1 Lingkungan Uji Coba 47

5.2 Dataset 47

5.3 Hasil Praproses 48

5.4 Skenario Uji Coba 50

5.4.1 Uji Coba Pembagian Data 51

5.4.2 Uji Coba Penggunaan Wavelet 53

5.4.3 Uji Coba Parameter CNN 55

5.4.4 Uji Coba Augmentasi Data 58

5.5 Hasil dan Evaluasi 60

6 BAB VI KESIMPULAN DAN SARAN 63

6.1 Kesimpulan 63

6.2 Saran 64

DAFTAR PUSTAKA 65

LAMPIRAN 69

L.1 Hasil Uji Coba Individu F01 69

L.2 Hasil Uji Coba Individu F09 72

L.3 Hasil Uji Coba Individu F15 76

L.4 Hasil Uji Coba Individu F18 79

L.5 Hasil Uji Coba Individu F24 83

L.6 Hasil Uji Coba Individu M21 86

L.7 Hasil Uji Coba Individu M23 90

L.8 Hasil Uji Coba Individu M24 93

L.9 Hasil Uji Coba Individu M26 97

L.10 Hasil Uji Coba Individu M30 100

BIODATA PENULIS 105

***(Halaman ini sengaja dikosongkan)***

# DAFTAR TABEL

[Tabel 2.1 *Confusion matrix* 21](#_Toc536399106)

[Tabel 3.1 Spesifikasi awal dataset 26](#_Toc536399107)

[Tabel 3.2 Arsitektur Wave Convolution Layers 34](#_Toc536399108)

[Tabel 3.3 Arsitektur Convolution Layers setelah merge 35](#_Toc536399109)

[Tabel 5.1 Spesifikasi dataset setelah proses DWT 49](#_Toc536399110)

[Tabel 5.2 Spesifikasi dataset setelah proses augmentasi data 50](#_Toc536399111)

[Tabel 5.3 Parameter awal yang digunakan dalam arsitektur 51](#_Toc536399112)

[Tabel 5.4 Perbandingan rata-rata lama waktu pelatihan dan akurasi, *precision*, *recall* arsitektur CNN pada uji coba pembagian data 53](#_Toc536399113)

[Tabel 5.5 Perbandingan lama waktu pelatihan dan akurasi, *precision*, *recall* arsitektur CNN pada uji coba penggunaan data *wavelet* 55](#_Toc536399114)

[Tabel 5.6 Perbandingan lama waktu pelatihan dan akurasi terbaik, *precision*, *recall* arsitektur CNN pada uji coba penggantian parameter CNN 57](#_Toc536399115)

[Tabel 5.7 Perbandingan lama waktu pelatihan dan akurasi, *precision*, *recall* arsitektur CNN pada uji coba augmentasi data 59](#_Toc536399116)

[Tabel 5.8 Parameter optimal yang ditetapkan 61](#_Toc536399117)

***(Halaman ini sengaja dikosongkan)***

# DAFTAR KODE SUMBER

[Kode Sumber 4.1 Fungsi perubahan ukuran gambar 38](#_Toc536399118)

[Kode Sumber 4.2 Fungsi *Discrete Wavelet Transform* 38](#_Toc536399119)

[Kode Sumber 4.3 Implementasi augmentasi data 39](#_Toc536399120)

[Kode Sumber 4.4 Fungsi pembangunan konvolusi *wavelet* 40](#_Toc536399121)

[Kode Sumber 4.5 Fungsi pembangunan arsitektur CNN 41](#_Toc536399122)

[Kode Sumber 4.6 Pemanggilan fungsi pembangunan arsitektur CNN 43](#_Toc536399123)

[Kode Sumber 4.7 Pembuatan index acak data uji 44](#_Toc536399124)

[Kode Sumber 4.8 Pembagian data latih dan data uji 45](#_Toc536399125)

[Kode Sumber 4.9 Pelatihan CNN 45](#_Toc536399126)

[Kode Sumber 4.10 Pengujian CNN 46](#_Toc536399127)

[Kode Sumber 4.11 Evaluasi *precision* dan *recall* 46](#_Toc536399128)

***(Halaman ini sengaja dikosongkan)***

# DAFTAR GAMBAR

[Gambar 2.1 Struktur umum sistem FER 7](#_Toc536399129)

[Gambar 2.2 Ilustrasi neuron manusia dalam model matematika [3] 8](#_Toc536399130)

[Gambar 2.3 Ilustrasi arsitektur *Multi Layer Perceptron* [3] 9](#_Toc536399131)

[Gambar 2.4 Contoh arsitektur *Convolutional Neural Network* [7] 10](#_Toc536399132)

[Gambar 2.5 Ilustrasi cara kerja konvolusi [8] 11](#_Toc536399133)

[Gambar 2.6 Ilustrasi cara kerja *Max Pooling* [9] 12](#_Toc536399134)

[Gambar 2.7 ReLU *Activation Function* [11] 13](#_Toc536399135)

[Gambar 2.8 Ilustrasi *neural network* dalam mengaplikasikan *Dropout* [3] 16](#_Toc536399136)

[Gambar 2.9 Ilustrasi cara kerja *Zero Padding* [13] 17](#_Toc536399137)

[Gambar 2.10 Ilustrasi cara kerja *Merge Layer Concatenate* 18](#_Toc536399138)

[Gambar 2.11 Contoh dekomposisi wavelet level 1 sampai 4 [1] 19](#_Toc536399139)

[Gambar 3.1 Contoh 7 ekspresi berbeda pada dataset KDEF 25](#_Toc536399140)

[Gambar 3.2 Contoh 5 sudut berbeda pada dataset KDEF 26](#_Toc536399141)

[Gambar 3.3 Diagram alir sistem yang dibangun 27](#_Toc536399142)

[Gambar 3.4 Diagram alir praproses data tanpa augmentasi 29](#_Toc536399143)

[Gambar 3.5 Diagram alir praproses data dengan augmentasi 31](#_Toc536399144)

[Gambar 3.6 Arsitektur CNN yang digunakan 32](#_Toc536399145)

[Gambar 3.7 Hasil penggabungan *layer* konvolusi *subband* dengan *Merge Layer Concatenate* 34](#_Toc536399146)

[Gambar 5.1 Gambar asli dan gambar setelah konversi ke *grayscale* dan *resize* 48](#_Toc536399147)

[Gambar 5.2 Hasil proses DWT 49](#_Toc536399148)

[Gambar 5.3 Gambar asli dan gambar hasil augmentasi 50](#_Toc536399149)

[Gambar 5.4 Grafik akurasi arsitektur CNN terbaik pada pembagian data berdasarkan sesi 52](#_Toc536399150)

[Gambar 5.5 Grafik akurasi arsitektur CNN terbaik pada pembagian data berdasarkan individu 52](#_Toc536399151)

[Gambar 5.6 Grafik akurasi pada arsitektur CNN dalam uji coba penggunaan data *wavelet* 54](#_Toc536399152)

[Gambar 5.7 Grafik akurasi pada arsitektur CNN dalam uji coba penggunaan data tanpa *wavelet* 54](#_Toc536399153)

[Gambar 5.8 Perbedaan akurasi pada arsitektur CNN dalam uji coba penggantian *optimizer* dan *learning rate* 56](#_Toc536399154)

[Gambar 5.9 Grafik akurasi pada arsitektur CNN dalam uji coba penggunaan data asli 58](#_Toc536399155)

[Gambar 5.10 Grafik akurasi pada arsitektur CNN dalam uji coba penggunaan data asli dan data augmentasi 59](#_Toc536399156)

[Gambar 5.11 Contoh tampilan ekspresi wajah yang mengalami salah klasifikasi terbanyak yaitu takut dan sedih 60](#_Toc536399157)

# BAB I PENDAHULUAN

## Latar Belakang

*Machine learning* telah menjadi bagian dari kehidupan sehari-hari bagi banyak orang di seluruh dunia. Penemuan dan implementasi *machine learning* memungkinkan komputer untuk belajar dan memprediksi pola yang mungkin terjadi dan dapat digunakan untuk membantu manusia melakukan kegiatan sehari-hari. Teknologi ini di zaman modern memungkinkan penyelesaian masalah lama dengan cara yang baru dan efisien. Beberapa pengaplikasian *machine learning* meliputi *fraud detection*, *image classification*, *information retrieval* dan *medical diagnosis* [1].

Salah satu pengaplikasian *machine learning* yang populer adalah *image classification*. *Image classification* mengkategorikan piksel-piksel di dalam suatu gambar menjadi satu dari banyak kelas gambar berdasarkan fitur yang berhasil diekstrak dari gambar tersebut [1]. Banyak bidang menggunakan *image classification* untuk meningkatkan kualitas produk, seperti bidang bisnis, finansial, kesehatan, riset, teknologi dan lain-lain. Seiring dengan berkembangnya teknologi, banyak perusahaan, badan riset dan universitas yang terus mengembangkan *machine learning* agar mendapat hasil yang lebih akurat, efisien dan cepat. Dari situlah lahir algoritma *deep learning*, yang merupakan bagian dari *machine learning*.

*Convolutional Neural Network* (CNN) adalah salah satu *deep neural network* yang cocok digunakan untuk mengolah data yang berbentuk 2 dimensi, seperti gambar dan video. Normalnya, CNN diterapkan untuk mengklasifikasikan gambar yang belum diolah (*raw image*). Di dalam tugas akhir ini, penulis mengusulkan sebuah algoritma *Convolutional Neural Network* (CNN) untuk ekstraksi fitur dan *Multi-class Support Vector Machine* untuk klasifikasi gambar. Selain itu sebelum di lakukan tahapan Ekstraksi Fitur dan Klasifikasi maka terlebih dahulu dilakukan tahapan *Preprocessing* dengan tujuan mengambil daerah telapak tangan dan meningkatkan intensitas pola dari pembuluh darah. Tujuannya adalah untuk meningkatkan ketepatan dan akurasi dari *Image Classification.* Data pelatihan dan uji coba yang digunakan diambil dari dataset “CASIA Multi-Spectral Palmprint V1.0” yang terdiri dari 600 data dengan 100 kelas yang berbeda berdasarkan kode untuk setiap individu. Metode *Augmentasi Rotasi* dan *Scaling* juga dilakukan untuk memperbanyak jumlah data pada setiap kelas setelah preprocessing gambar.

## Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

1. Bagaimana memperbanyak data gambar pembuluh darah telapak tangan menggunakan metode *Augmentasi* sebelum diimplementasikan ke *Convolutional Neural Network*?

2. Bagaimana mengimplementasikan *Convolutional Neural Network* untuk Ekstraksi Fitur pada data gambar pembuluh darah telapak tangan?

3. Bagaimana mengimplementasikan *Multi-Class Suuport Vector Machine* untuk klasifikasi data gambar pembuluh darah telapak tangan pada setiap individu?

4. Bagaimana mengevaluasi kinerja model klasifikasi *Multi Class Support Vector Machine* yang telah diimplementasikan?

## Batasan Permasalahan

Permasalahan yang dibahas pada Tugas Akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Data gambar diambil dari dataset “CASIA Multi-Spectral Palmprint V1.0”.

2. Implementasi program menggunakan bahasa pemrograman *Python 3* dan *Matlab*.

3. Pengenalan pembuluh darah telapak tangan dengan jumlah 6 data pada setiap individu.

## Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah untuk membangun sebuah sistem klasifikasi gambar dengan menggunakan *Convolutional Neural Network* untuk Ekstraksi Fitur dan metode klasifikasi *Multi Class Support Vector Machine* yang dapat mengenali pembuluh darah pada telapak tangan untuk setiap individu.

## Manfaat

Tugas akhir ini diharapkan dapat membantu menambah kemampuan yang ada pada pengenalan pembuluh darah telapak tangan, guna membantu dalam perkembangan sistem keamanan berdasarkan anggota tubuh manusia yang disebut dengan keamanan Biometrik.

## Metodologi

Pembuatan Tugas Akhir ini dilakukan dengan menggunakan metodologi sebagai berikut:

### Penyusunan Proposal Tugas Akhir

Tahapan awal dari Tugas Akhir ini adalah penyusunan Proposal Tugas Akhir yang berisi pendahuluan, deskripsi dan gagasan metode-metode yang dibuat dalam Tugas Akhir ini. Pendahuluan ini terdiri dari latar belakang diajukannya Tugas Akhir, rumusan masalah dan batasan masalah yang ditetapkan, serta manfaat dari hasil pembuatan Tugas Akhir ini. Selain itu, dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan Tugas Akhir. Terdapat pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan Tugas Akhir.

### Studi Literatur

Pada tahap ini dilakukan pencarian literatur berupa jurnal yang digunakan sebagai referensi untuk pengerjaan tugas akhir ini. Literatur yang dipelajari pada pengerjaan tugas akhir ini berasal dari jurnal ilmiah yang diambil dari berbagai sumber di internet, beserta berbagai literatur online tambahan terkait *Inner Border Tracing, Median Filter, Adaptive Histogram Equalization, Adaptive Noise Removal, Anistropic Diffusion Filter, Morfologi Closing, Convolutional Neural Network*, *TensorFlow*, *Keras* dan *Multi Class Support Vector Machine*.

### Implementasi Perangkat Lunak

Pada tahap ini akan dilaksanakan implementasi metode dan algoritma yang telah direncanakan. Implementasi sistem menggunakan *Python 3* dan *Matlab* sebagaibahasa pemrograman, *TensorFlow* dan *Keras* sebagai *framework*, serta *library* pendukung lainya.

### Pengujian dan Evaluasi

Tahap pengujian dan evaluasi dilakukan menggunakan dataset “CASIA Multi-Spectral Palmprint V1.0”. “CASIA Multi-Spectral Palmprint V1.0” untuk mengetahui hasil dan performa arsitektur yang telah dibangun. Evaluasi dilakukan dengan metode pengukuran akurasi*, precision,* dan *recall*.

### Penyusunan Buku

Pada tahap ini dilakukan penyusunan buku yang menjelaskan seluruh konsep, teori dasar dari metode yang digunakan, implementasi, serta hasil yang telah dikerjakan sebagai dokumentasi dari pelaksanaan Tugas Akhir.

## Sistematika Penulisan Laporan

Sistematika penulisan laporan Tugas Akhir adalah sebagai berikut:

**Bab I Pendahuluan**

Bab ini berisikan penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan dari pembuatan Tugas Akhir.

**Bab II Tinjauan Pustaka**

Bab ini berisi kajian teori dari metode dan algoritma yang digunakan dalam penyusunan Tugas Akhir ini. Secara garis besar, bab ini berisi tentang *Convolutional Neural Network*, *Multi Class Support Vector Machine* dan *library* yang digunakan.

**Bab III Perancangan Sistem**

Bab ini berisi pembahasan mengenai perancangan dari *Convolutional Neural Network* yang digunakan untuk Ekstraksi Fitur dan *Multi Class Support Vector Machine* yang digunakan untuk pengenalan ekspresi wajah manusia pada data gambar.

**Bab IV Implementasi**

Bab ini membahas implementasi dari perancangan yang telah dibuat pada bab sebelumnya. Penjelasan berupa kode yang digunakan untuk proses implementasi.

**Bab V Uji Coba Dan Evaluasi**

Bab ini membahas tahapan uji coba, kemudian hasil uji coba dievaluasi terhadap kinerja dari sistem yang dibangun.

**Bab VI Kesimpulan dan Saran**

Bab ini merupakan bab yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan, masalah-masalah yang dialami pada proses dan tertulis saat pengerjaan Tugas Akhir, dan saran untuk pengembangan solusi ke depannya.

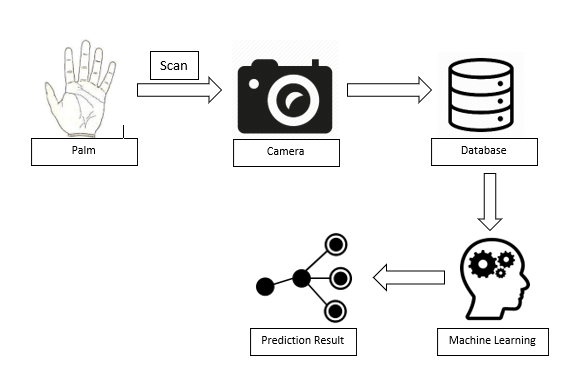
***(Halaman ini sengaja dikosongkan)***

# BAB II TINJAUAN PUSTAKA

Bab ini membahas mengenai teori-teori dasar yang digunakan dalam Tugas Akhir. Teori-teori tersebut diantaranya adalah Convolutional Neural Network dan Multi Class Support Vector Machine, dan beberapa teori lain yang mendukung pembuatan Tugas Akhir. Penjelasan ini bertujuan untuk memberikan gambaran umum dan diharapkan dapat mendukung sistem yang dibangun.

## Pengenalan Palm Vein

Pengenalan pembuluh darah telapak tangan atau *Palm Vein Recognition* (PVR) adalah kemampuan untuk menafsirkan pola pembuluh darah pada telapak tangan manusia, seperti yang terlihat pada Gambar 2.1. Pengenalan pembuluh darah manusia merupakan bidang penting dalam bidang *Computer Vision* dan penerapannya dapat digunakan di berbagai hal seperti untuk keamanan biometrik[2].



Gambar 2.1 Struktur umum sistem

## Regoin of Interest (ROI)

*Region of interest* (ROI) merupakan bagian dari sebuah gambar yang dibutuhkan untuk identifikasi dengan tujuan tertentu [2]. Konsep dari ROI telah banyak digunakan diberbagai bidang. Berikut adalah contoh ROI berdasarkan dimensinya [3]:

* Dataset 1D: berupa waktu atau interval frekuensi dalam bentuk gelombang.
* Dataset 2D: berupa batas-batas suatu objek pada gambar.
* Dataset 3D: kontur atau permukaan dari sebuah objek volume (atau biasanya dikenal sebagai *Volume of Interest* (VOI)).
* Dataset 4D: outline dari sebuah objek atau interval waktu padab volume waktu.

Macam-macam bentuk ROI 2D yaitu *circle, ellipse, rectangle*, dan *polygonal*. Pengambilan ROI dalam tugas akhir ini dilakukan menggunakan algoritma *Inner Border Tracing* yang akan menghasilkan fokusan citra pada pembuluh darah telapak tangan.

## Median Filter

Median filter merupakan teknik peningkatan kualitas citra dalam hal penghilangan noise. Metode ini termasuk dalam *non-linear filtering* [4]. Tujuan dari median filter ini adalah mereduksi noise yang terdapat pada citra dengan blurring sehingga akan memperhalus gambar atau gambar akan terlihat lebih smooth.

Median filter bekerja dengan mengganti nilai suatu pixel pada citra asal dengan nilai median dari piksel tersebut dan lingkungan tetangganya. Nilai median dicari dengan melakukan pengurutan nilai piksel dari nilai terkecil hingga terbesar dari titik-titik pada matriks yang sudah ditentukan, lalu dicari dan ditentukan nilai tengahnya. Berikut adalah bentuk persamaan dari Median Filter

(1)

Dimana X adalah elemen data dan n adalah ukuran himpunanv data. Apabila n merupakan bilangan genap, maka nilai Median Filter merupakan setengah dari penjumlahan elemen data pada urutan dan elemen data pada urutan . Sedangkan apabila n merupakan bilangan ganjil, maka nilai Median Filter merupakan elemen data pada urutan .



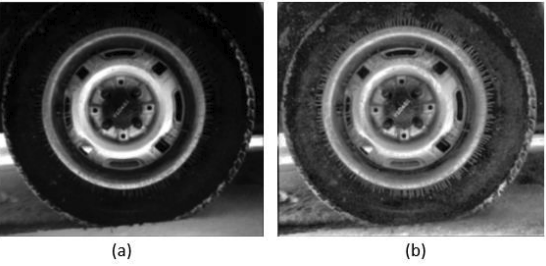
Gambar 1. Contoh penggunaan Median Filter (a) citra awal; (b) citra hasil Median Filter [4].

## Adaptive Histogram Equalization

*Adaptive Histogram Equalization* (AHE) merupakan salah satu teknik untuk meningkatkan kontras pada citra. Metode ini cocok untuk memperbaiki kontras lokal dan meningkatkan definisi tepi pada setiap wilayah citra.

AHE ini dapat memperbaiki citra dengan kontras lebih terang atau lebih gelap dengan cara mengubah setiap piksel dengan fungsi transformasi.

Variasi dari AHE adalah *Contrast Limited Adaptive Histogram Equalization* (CLAHE). CLAHE beroperasi pada wilayah/region kecil dalam gambar, yang disebut tiles [5]. Adaptive histogram equalization menghitung fungsi pengubahan kontras setiap tiles secara individual/satu per satu. Setiap kontras tiles akan ditingkatkan sehingga histogram dari wilayah output akan sesuai dengan histogram yang ditentukan oleh nilai distribusi. Lalu setiap neighboring tiles akan digabung menggunakan interpolasi bilinear untuk menghilangkan batas-batas yang diinduksi secara artifisial. Kontras pada daerah homogen dapat dibatasi untuk menghindari amplifikasi noise yang mungkin terdapat pada gambar [2].

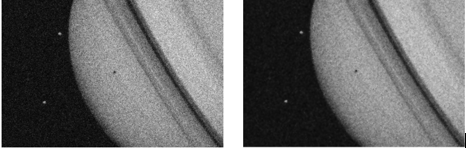


Gambar 2. Contoh penggunaan Adaptive Histogram Equalization (a) citra awal; (b) citra hasil Adaptive Histogram Equalization [5].

## Adaptive Noise Removal

*Adaptive Noise Removal* merupakan teknik filtering yang digunakan untuk mengurangi noise dengan mengganti nilai pixel tetangga dengan nilai baru yang didapat nilai ketetanggaannya. Filter ini bertujuan untuk memperbaiki citra yang awalnya terdapat titik-titik noise pada citra, lalu noise tersebut menjadi berkurang dan citra terlihat lebih jelas. Algoritma dari teknik filter *Adaptive Noise Removal* yaitu [6]:

* Membentuk graf pixel tetangga dengan cara menentukan piksel-piksel tetangga mana saja yang akan mempengaruhi perubahan nilai untuk piksel aktual.
* Memberi bobot nilai berdasarkan perbedaan jarak antar piksel yang difilter dengan piksel-piksel tetangga. Bobot piksel-piksel tetangga dihitung dengan fungsi Gaussian.
* Mengganti nilai piksel dengan rata-rata nilai.
* Filter ini akan memilih piksel tetangga yang relevan. Ukuran dan bentuk piksel tetangga akan mempengaruhi perubahan citra.



Gambar 3. Citra awal dan citra hasil Adaptive Noise Removal [7].

## Anistropic Diffusion Filter

*Anisotropic Diffusion Filter* merupakan teknik untuk mengurangi noise pada gambar tanpa menghilangkan bagian penting dari konten gambar, seperti garis, tepi, atau bagian lain yang menginterpretasikan gambar [8].



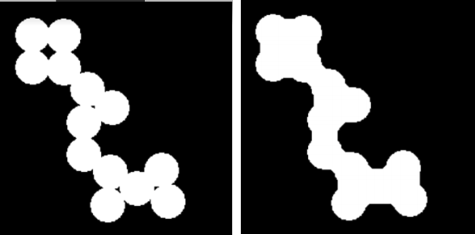
Gambar 4. Citra awal sebelum ADF dan citra hasil ADF [8].

## Morfologi Closing

Operasi morfologi memiliki peranan yang penting dalam pengolahan citra digital. Prinsip dari operasi morfologi adalah mengekstrak komponen-komponen citra yang berguna dalam representasi dan deskripsi bentuk seperti ekstrak boundary dari suatu region.

Pada metode operasi morfologi sering digunakan structuring element [9]. Structuring element merupakan kumpulan pola spesifik berupa koordinat angka diskrit. Representasi koordinat angka yang biasa digunakan adalah koordinat cartesian.

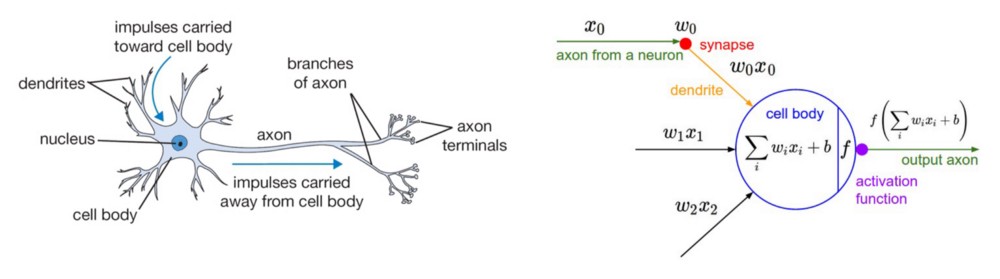
Operasi morfologi closing digunakan untuk menghaluskan bagian dari kontur dari sebuah objek dan menyatukan dua objek yang terputus. Closing juga digunakan untuk menghilangkan lubang kecil pada objek dengan mengisi celah yang terdapat dalam kontur. Pada operasi morfologi closing dilakukan dilasi terlebih dahulu, lalu dilakukan erosi [10].

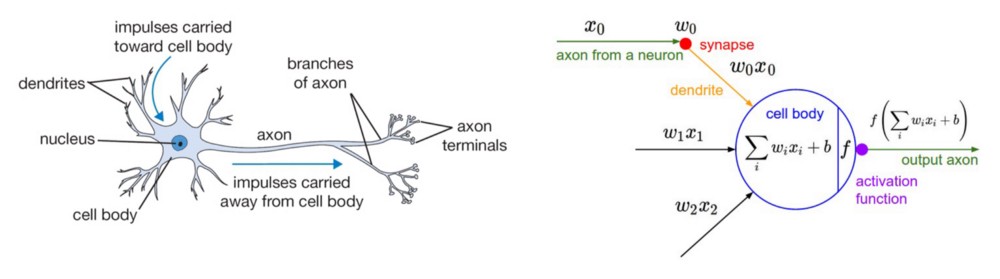


Gambar 5. Citra Awal dan citra hasil Morfologi Closing [15].

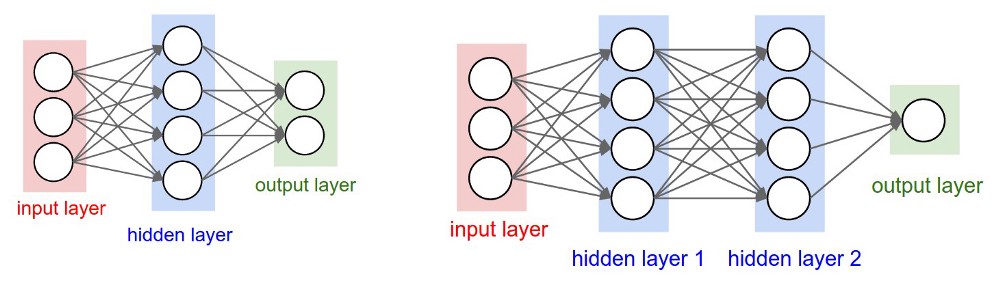
## Multi Layer Perceptron

*Multi Layer Perceptron* (MLP) atau dikenal dengan *Fully Connected Layer* pada *Convolutional Neural Network* adalah model yang terinspirasi dari bagaimana neuron dalam otak manusia bekerja. Tiap neuron pada otak manusia saling berhubungan dan informasi mengalir dari setiap neuron tersebut. Tiap neuron menerima *input* dan melakukan operasi dot dengan sebuah bobotatau nilai bobot dan menambahkan nilai bias. Hasil dari operasi ini akan dijadikan parameter dari fungsi aktivasi yang akan dijadikan *output* dari neuron tersebut [3]. Ilustrasi dapat dilihat pada Gambar 2.2. MLP terdiri dari *Input Layer*, *Hidden Layer*, *Output Layer*. *Input Layer* menerima masukan (tanpa melalui fungsi aktivasi), kemudian nilai diberikan ke *Hidden Layer*, dimana akan dilakukan perhitungan hasil fungsi aktivasi untuk tiap-tiap neuron, lalu hasilnya diberikan ke *Output Layer*. Ilustrasi arsitektur *Multi Layer Perceptron* terlihat pada Gambar 2.3.





Gambar 2.2 Ilustrasi neuron manusia dalam model matematika [3]



Gambar 2.3 Ilustrasi arsitektur *Multi Layer Perceptron* [3]

Pada tahap pelatihan nilai bobot dan bias pada tiap neuron akan diperbarui terus menerus hingga *output* yang dihasilkan sesuai yang diharapkan. Pada tiap iterasi akan dilakukan proses evaluasi yang biasanya digunakan untuk menentukan kapan harus menghentikan proses pelatihan. Proses pelatihan MLP terdiri dari *Forward* dan *Backward Pass.*

Tahap *Forward Pass* adalah proses dimana data masukan dibawa melewati tiap neuron sampai kepada *Output Layer* yang nantinya akan dihitung nilai *error*. Persamaan (2.1) adalah operasi dot *x* (*input*) dengan *w* (bobot) dan ditambah dengan *b* (bias)yang kemudian menggunakan fungsi aktivasi, misalnya ReLU pada Persamaan (2.2).

|  |  |
| --- | --- |
|  | (2.1) |
|  | (2.2) |

Tahap *Backward Pass* bertujuan untuk menyesuaikan kembali tiap bobot dan bias berdasarkan *error* yang didapat pada saat *forward pass*. Proses ini disebut sebagai propagasi balik (*backpropagation*) yaitu tahap pelatihan yang mengubah *weight* neuron-neuron di dalam MLP. Propagasi balik memanfaatkan sebuah *loss function* untuk menghitung *error* dari nilai prediksi dengan nilai sebenarnya. Selanjutnya, digunakan algoritma pengoptimalan untuk memperbarui bobot dan bias dengan tujuan menurunkan nilai *error*, hal ini dilakukan secara iteratif sampai *epoch* tertentu.

## Convolutional Neural Network

*Convolutional Neural Network* (CNN) adalah salah satu algoritma dari *deep learning* yang merupakan pengembangan dari *Multi Layer Percepton* (MLP) yang dirancang untuk mengolah data dalam bentuk dua dimensi, misalnya gambar atau suara. CNN sering digunakan untuk mengenali citra benda atau pemandangan, melakukan deteksi dan segmentasi objek [4].

Penelitian awal yang mendasari penemuan ini dilakukan oleh Hubel dan Wiesel [5] yang melakukan penelitian visual korteks pada indera penglihatan kucing. Penelitian ini sangat berguna dalam sistem pemrosesan visual yang pernah ada. Hingga banyak penelitian yang terinspirasi dari cara kerjanya dan menghasilkan model-model baru. Arsitektur dari CNN dibagi menjadi 2 bagian besar, *Feature Learning* / *Extraction Layer* dan *Classification Layer* [6], seperti yang dipaparkan pada Gambar 2.4.



Gambar 2.4 Contoh arsitektur *Convolutional Neural Network* [7]

*Feature Learning* / *Extraction Layer* adalah bagian dimana terjadi proses penerjemahan dari sebuah citra menjadi *features. Features* ini berupa angka-angka yang merepresentasikan citra tersebut, yaitu berupa *feature map*. Proses ini terdiri dari *Convolutional Layer* dan *Pooling Layer*.

*Classification Layer* adalah dimana *feature map* yang dihasilkan dari *convolutional layers* masih berbentuk array multidimensi, sehingga harus dilakukan pengubahan *feature map* menjadi sebuah *feature vector* agar bisa digunakan sebagai masukan dari *fully connected layer*. *Fully connected layer* yang dimaksud disini adalah *Multi Layer Perceptron* (MLP) yang memiliki beberapa *hidden layer*, *activation function*, *output layer* dan *loss function*.

### Convolution Layer

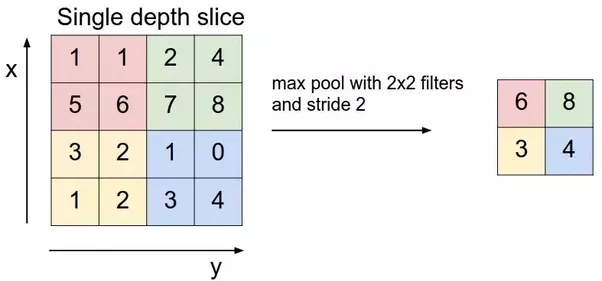
*Convolution Layer* melakukan operasi konvolusi pada output dari lapisan sebelumnya. Konvolusi adalah istilah matematis yang artinya mengaplikasikan sebuah fungsi pada *output* fungsi lain secara berulang. Tujuan dilakukannya konvolusi pada data citra adalah untuk mengekstrak fitur dari citra masukan [4]. Ilustrasi cara kerja konvolusi bisa dilihat pada Gambar 2.5, dimana I adalah citra, K adalah *filter* atau *kernel* yang digunakan, I \* K adalah hasil operasi konvolusi.



Gambar 2.5 Ilustrasi cara kerja konvolusi [8]

### Pooling Layer

Fungsi dari *Pooling Layer* adalah mereduksi ukuran dari data. Terdapat beberapa tipe *Pooling Layer* diantaranya yaitu *max*, *average*, *sum* dan lainnya. Metode *Pooling* dalam CNN yang biasa digunakan adalah *Max Pooling* & *Average Pooling*. *Max Pooling* membagi *output* dari *Convolution Layer* menjadi beberapa matrikskecil lalu mengambil nilai maksimal dari tiap matriksuntuk menyusun matriks citra yang telah direduksi, sedangkan *Average Pooling* akan memilih nilai rata-ratanya. Proses tersebut memas-tikan fitur yang didapatkan akan sama meskipun obyek citra mengalami translasi. Ilustrasi cara kerja *Max Pooling* bisa dilihat pada Gambar 2.6.



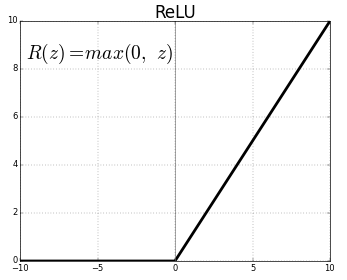
Gambar 2.6 Ilustrasi cara kerja *Max Pooling* [9]

### Fully Connected Layer

*Fully Connected Layer* dalam penerapannya sama dengan *Multi Layer Perceptron* (MLP)yang bertujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear. *Feature map* dari *Convolution Layer* perlu ditransformasi menjadi data satu dimensi terlebih dahulu yang disebut *feature vector* sebelum dapat dimasukkan ke dalam sebuah *Fully Connected Layer*. Karena hal tersebut menyebabkan data kehilangan informasi spasialnya dan tidak reversibel, *Fully Connected Layer* diimplementasikan di akhir jaringan [10].

### ReLU Activation Function

Fungsi aktivasi befungsi untuk menentukan apakah neuron tersebut harus aktif atau tidak berdasarkan nilai masukan. Salah satu contoh fungsi aktivasi adalah ReLU (Rectified Linear Unit) dimana fungsi ini melakukan *thresholding* dengan nilai nol terhadap nilai masukan, dimana seluruh nilai yang kurang dari nol akan dijadikan nol, seperti pada Gambar 2.7.



Gambar 2.7 ReLU *Activation Function* [11]

### Fungsi Softmax

Fungsi *softmax* biasa digunakan dalam klasifikasi banyak kelas. *Softmax* memberikan nilai probabilitas untuk setiap label kelas, dimana jumlah seluruh probabilitas adalah 1. *Softmax* pada dasarnya adalah probabilitas eksponensial yang dinormalisasi dari nilai masukan sejumlah kelas pada model klasifikasi seperti pada Persamaan (2.3).

|  |  |
| --- | --- |
|  | (2.3) |

Dimana y adalah nilai masukan. Operasi akan menghasilkan nilai probabilitas. Label dari data masukan akan ditentukan berdasarkan kelas dengan nilai probabilitas tertinggi.

### Cross Entropy

*Loss function* merupakan fungsi yang menggambarkan kerugian yang dihasilkan oleh model. *Loss function* dikatakan baik, ketika menghasilkan *error* yang diharapkan paling rendah. Pada permasalahan klasifikasi banyak kelas, *cross entropy* adalah *loss function* yang biasa digunakan. *Cross entropy* akan menghitung *error* antara nilai prediksi *S* dengan nilai sebenarnya *T*, seperti pada Persamaan (2.4). Selanjutnya, nilai *error* akhir diambil dari rata-rata hasil *cross entropy*, seperti pada Persamaan (2.4) dan (2.5).

|  |  |
| --- | --- |
|  | (2.4) |
|  | (2.5) |

### Adam

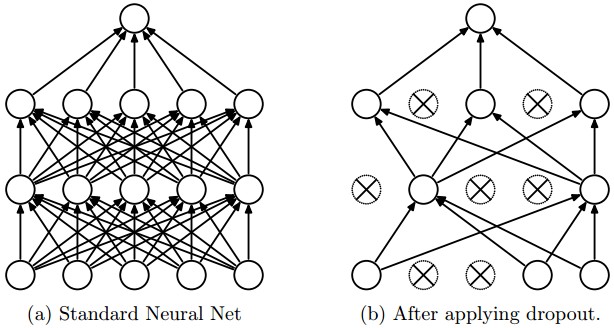
Adam(*Adaptive Moment Estimation)* juga adalah algoritma pengoptimalan yang dapat digunakan. Hampir sama dengan Adagrad, Adam memiliki sebuah *learning rate* untuk setiap parameter dan secara terpisah beradaptasi saat proses pelatihan. Adam memperbarui nilai setiap parameter seperti RMSProp [13]. Perbedaannya Adam menggunakan gradien yang telah diperhalus dan semakin mengecil seperti yang dapat dilihat pada Persamaan (2.12). Lalu gradien tersebut akan digunakan untuk memperbarui parameter, seperti yang dapat dilihat pada Persamaan (2.13) dan (2.14).

|  |  |
| --- | --- |
|  | (2.6) |
|  | (2.7) |
|  | (2.8) |

### Dropout

*Dropout* merupakan proses mencegah terjadinya *overfitting* dan juga mempercepat proses learning. *Dropout* mengacu kepada menghilangkan neuron yang berupa *hidden layer* mapun *visible layer* di dalam jaringan [14]. Dengan menghilangkan suatu neuron, berarti menghilangkannya sementara dari jaringan yang ada. Neuron yang akan dihilangkan akan dipilih secara acak.

Pada Gambar 2.8, (a) neuron tetap utuh pada *neural network* yang belum memakai *Dropout*, dan (b) *neural network* yang sebagian dari neuronnya tidak digunakan setelah diaplikasikan Dropout.



Gambar 2.8 Ilustrasi *neural network* dalam mengaplikasikan *Dropout* [3]

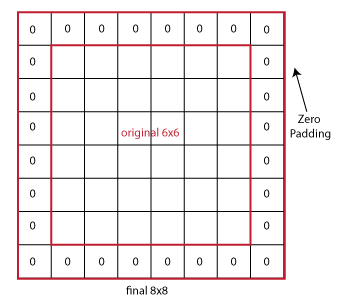
### Batch Normalization

*Batch Normalization* adalah teknik melakukan normalisasi terhadap *batch* atau kumpulan data masukan, seperti yang terlihat pada Persamaan (2.15). Dimana *x* adalah nilai masukan, *μb* adalah *mean* dari *batch*, *σb* adalah standar deviasi dari *batch*. Normalisasi dilakukan agar data memiliki *mean* mendekati 0 dan standar deviasi mendekati 1.

|  |  |
| --- | --- |
|  | (2.15) |

### Zero Padding

*Padding* atau *Zero Padding* adalah teknik penambahan nilai 0 di sisi-sisi data masukan. Teknik ini digunakan untuk memanipulasi dimensi *output* dari *Convolution Layer*. Dengan menggunakan *padding*, dimensi *output* dapat tetap sama seperti dimensi *input* atau setidaknya tidak berkurang secara signifikan. Sehingga kita bisa memanfaatkan *Convolution Layer* lebih dalam dan lebih banyak *features* yang berhasil didapatkan. Ilustrasi cara kerja *Zero Padding* bisa dilihat pada Gambar 2.9.



Gambar 2.9 Ilustrasi cara kerja *Zero Padding* [13]

## Augmentasi Data

Dalam mendapatkan performa yang optimal, *deep learning* seperti *Convolutional Neural Network* membutuhkan data yang lebih banyak dibandingkan dengan algoritma yang lain, untuk itu kita perlu melakukan augmentasi data. Augmentasi data adalah sebuah teknik menambah data dengan cara memanipulasi data yang telah ada dengan pengaturan keragaman tertentu. Untuk data berupa citra, kita bisa lakukan operasi seperti translasi, refleksi, perbesaran ukuran, dan lain-lain.

### Refleksi Citra

Refleksi citra atau pencerminan terdapat 2 macam, yaitu horizontal dan vertikal. Refleksi horizontal adalah pencerminan pada sumbu-Y (koordinat kartesian) dari citra *A* menjadi citra *B* seperti pada Persamaan (2.19) [16].

|  |  |
| --- | --- |
| *B[x][y] = A[-x][y]* | (2.19) |

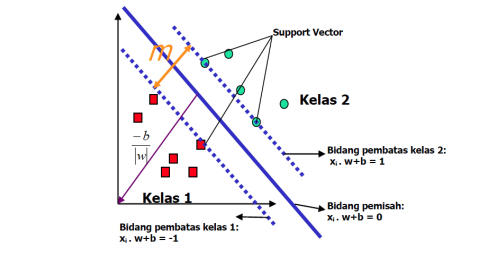
### Perbesaran Ukuran Gambar

Perbesaran ukuran gambar adalah proses memperbesar ukuran gambar secara acak untuk memperoleh gambar yang lebih variatif. Gambar diperbesar pada salah satu titiknya untuk dicari keragaman lain dari gambar.

## Support Vector Machine

*Support Vector Machine* (SVM) [17] merupakan teknik untukmelakukan prediksi, baik dalam kasus klasifikasi maupun regresi. SVM merupakan metode yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dengan tujuan menemukan hyperplane (bidang pemisah) *classifier* yang optimal yang dapat memisahkan dua set data dari dua kelas berbeda. Bidang pemisah atau hyperplane terbaik tidak hanya dapat memisahkan data tetapi juga memiliki margin paling besar. *Hyperplane* pemisah terbaik antara dua kelas dapat ditemukan dengan mengukur margin dari hyperplane tersebut. Margin adalah jarak antara hyperplane dengan pola (*support vector*) terdekat dari masing-masing kelas.

Contoh bidang pemisah (hyperplane) yang dapat memisahkan dua buah kelas dapat dilihat pada Gambar 11.

Gambar 11. Contoh bidang pemisah (hyperplane) [17].

SVM memiliki prinsip dasar *linier classifier* namun SVM telah dikembangkan agar dapat bekerja pada problem non-linear dengan memasukkan konsep kernel pada ruang kerja berdimensi tinggi. Menurut Satosa (2007) hyperplane klasifikasi linier SVM dinotasikan:

(7)

sehingga menurut Vapnik dan Cortes (1995) diperoleh persamaan berikut

(8)

Dengan, adalah normal bidang, *b* adalah posisi bidang relatif terhadap pusat koordinat, adalah himpunan data training, dan = label kelas dari .

Dalam mencari hyperplane terbaik dapat digunakan metode *Quadratic Programming* *(QP)* *Problem*. Solusi untuk mengoptimasi penemuan Vapnik (1995) yaitu menggunakan fungsi *Lagrange* sebagai berikut:

(9)

Dengan pengganda fungsi Lagrange dan

Lalu terdapat penyempurnaan-penyempurnaan lain hingga didapatkan persamaan dual problem berikut oleh Hestie et al (2001).

(10)

Jadi persoalan pencarian bidang pemisah terbaik dapat dirumuskan pada persamaan (11).

(11)

Dengan batasan, dan

Dengan demikian, diperoleh nilai yang akan digunakan untuk menemukan *w*. Data yang memiliki nilai adalah *support vector* sedangkan sisanya memiliki nilai . Maka fungsi keputusan yang dihasilkan hanya dipengaruhi oleh support vector.

Setelah solusi permasalahan *quadratic programming* ditemukan (nilai ), maka kelas dari data pengujian *x* dapat ditentukan berdasarkan nilai dari fungsi keputusan pada persamaan (12).

(12)

adalah *support vector*, *ns* adalah jumlah support vector, dan adalah data yang akan diklasifikasikan.

Selain itu, SVM dapat bekerja pada data non-linier dengan menggunakan pendekatan kernel. Fungsi kernel yaitu untuk memetakan dimensi awal (dimensi yang lebih rendah) himpunan data ke dimensi baru (dimensi yang relatif lebih tinggi). Macam-macam fungsi kernel diantaranya:

* + - 1. *Kernel Linear*

(13)

* + - 1. *Kernel Gaussian Radial Basic Function* (RBF)

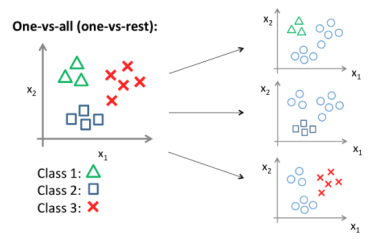
(14)

* + - 1. *Kernel Polynomial*

(15)

dan adalah pasangan dua data training. Parameter *γ, c, d* *> 0* merupakan konstanta. Fungsi kernel yang harus digunakan untuk substitusi dot product di *feature space* bergantung pada data karena fungsi kernel ini akan menentukan fitur baru yang akan dicari hyperplane-nya.

Seiring berjalannya waktu, pengembangan SVM dilakukan sehingga bisa mengklasifikasi data yang memiliki lebih dari dua kelas. Metode ini disebut *Multi Class SVM*. Proses pemodelan pada *Multi Class SVM* membutuhkan data pelatihan yang lebih banyak dibandingkan dengan dua kelas. Tahapan metode *Multi-Class* ini yaitu membuat model atau melakukan pemisahan setiap kelas dengan kelas lain yang dilakukan sebanyak jumlah kelas. Kemudian untuk memprediksi kelas dari data masukan, maka dibandingkan terhadap setiap model yang sudah dibuat. Ilustrasi *Multi-Class SVM* dapat dilihat pada Gambar 12.



Gambar 12. Ilustrasi Multi-Class SVM [17].

## Akurasi, Precision & Recall

Ketika membangun sebuah model klasifikasi, pertanyaan yang muncul adalah bagaimana mengetahui seberapa baik model tersebut. Mengevaluasi model klasifikasi dilakukan dengan mencari tahu seberapa baik hasil prediksi dari model tersebut. *Recall* di Persamaan (2.20), *precision* di Persamaan (2.21), dan akurasi di Persamaan (2.22) adalah metode pengukuran yang biasa digunakan dalam mengevaluasi model, penjelasan variabel ada pada *confusion matrix* pada Tabel 2.1.

Tabel 2.1 *Confusion matrix*

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Kelas Prediksi** | |
| Benar | Salah |
| **Kelas sebenarnya** | Benar | *True Positive* (TP) | *False Negative* (FN) |
| Salah | *False Positive* (FP) | *True Negative* (TN) |

Keterangan :

1. *True Positive* (TP) pada gambar dikenali ekspresi manusia, pada keluaran program mengenali ekspresi tersebut.
2. *True Negative* (TN) pada gambar tidak dikenali ekspresi manusia, pada keluaran program tidak mengenali ekspresi tersebut.
3. *False Positive* (FP) pada gambar dikenali ekspresi manusia, pada keluaran program tidak mengenali ekspresi tersebut.
4. *False Negative* (FN) pada gambar tidak dikenali ekspresi manusia, pada keluaran program mengenali ekspresi tersebut.

|  |  |
| --- | --- |
| *Recall = TP / (TP + FN)* | (2.20) |
| *Precision = TP / (TP + FP)* | (2.21) |
| *Akurasi* *= (TP + TN) / (TP + FP + TN + FN)* | (2.22) |

## Matlab

MATLAB adalah bahasa pemrograman tinggi, tertutup, dan case sensitive dalam lingkungan komputasi numerik yang dikembangkan oleh MathWorks. Salah satu kelebihannya yang paling populer adalah kemampuan membuat grafik dengan dukungan kustomisasi terbaik. MATLAB mempunyai banyak tools yang dapat membantu berbagai disiplin ilmu. Ini merupakan salah satu penyebab industri menggunakan MATLAB. Selain itu MATLAB mempunyai banyak library yang sangat membantu untuk menyelesaikan permasalahan matematika seperti membuat simulasi fungsi, pemodelan matematika dan perancangan GUI.

## Python

Python adalah bahasa pemograman yang populer. *Python* sering dimanfaatkan dalam pengembangan web, perangkat lunak, penelitian, dan *system scripting*. Python dapat digunakan untuk menangani data besar dan melakukan operasi matematika yang kompleks. Python bekerja di berbagai *platform* seperti Windows, Mac, Linux, Raspberry Pi, dan lain-lain. Python dirancang untuk mudah dibaca, yaitu memiliki sintaks yang sederhana dan menggunakan bahasa inggris [17].

## Keras

Keras adalah *high-level neural networks* API, yang ditulis dalam bahasa pemograman Pythondan mampu berjalan di atas TensorFlow dan Theano. Keras dikembangkan dalam rangka memungkinkan eksperiman dilakukan dengan cepat. Keras dapat berjalan baik di CPU dan GPU. Keras berisi banyak implementasi *neural network* yang umum digunakan, fungsi aktivasi, *optimizer*, dan *tool* lain yang memudahkan dalam pengolahan citra dan data teks [18].

## TensorFlow

TensorFlowadalah *library open source* untuk pembuatan program yang membutuhkan komputasi numerik berkinerja tinggi. TensorFlow dikembangakan oleh tim Google Brain. TensorFlow menyediakan fungsi-fungsi *machine learning* dan *deep learning*, dan dapat dijalankan dalam CPU atau GPU [19].

## OpenCV

OpenCV (*Open Source Computer Vision*) adalah *library* yang dimanfaatkan dalam pengolahan citra dinamis secara *real-time*. OpenCV dapat digunakan dalam berbagai bahasa pemograman seperti Python, C++, Java, atau MATLAB. OpenCV memiliki fitur seperti *Feature & Object Detection,* *Motion Analysis and Object Tracking, Image Filtering, Image Processing*, dan lain-lain [20].

## Numpy

Numpy adalah *library* *Python* yang mendukung pengolahan data pada *array* dan matriks multidimensi yang besar. Numpy menyediakan kumpulan fungsi matematika, seperti aljabar linear, transformasi Fourier, pembuatan angka acak, dan lain-lain. *Numpy* bersifat *open source* sehingga banyak dimanfaatkan dalam pengolahan data peneilitian [22].

## Scikit-learn

Scikit-learn adalah *open source* *machine learning library* untuk bahasa pemogramanPython*.* Scikit-learn menyediakan fitur seperti *classification*, *regression*, *clustering*, termasuk juga didalamnya algoritma *support vector machines*, *random forest*, *gradient boosting*, dan lain-lain [23].

## Scipy

## Scipy adalah *library Python* yang mendukung dalam pembuatan operasi Aljabar Linear, Integrasi, interpolasi, optimasi dan Statistika.

## Matplotlib

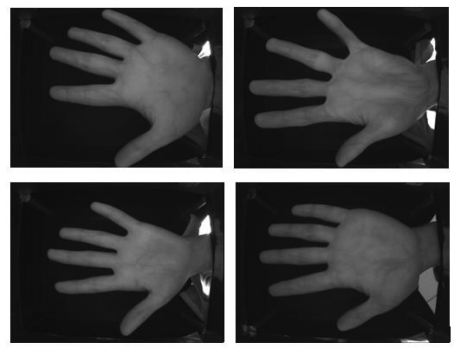
Matplotlib adalah *library* *Python* yang mendukung pembuatan grafik dua dimensi dalam berbagai format dan dari berbagai jenis data. Matplotlib bersifat *open source* dan banyak digunakan untuk pengolahan data dalam penelitian. Matplotlib dapat membuat plot, histogram, spektrum daya, diagram batang, diagram kesalahan, plot pencar, dan lain-lain [24].

# BAB III PERANCANGAN SISTEM

Bab ini menjelaskan mengenai perancangan data dan sistem pengenalan pembuluh darah telapak tangan menggunakan *Convolutional Neural Network*. Bab ini juga akan menjelaskan gambaran umum sistem dalam bentuk diagram alir*.*

## Perancangan Data

Data yang digunakan sebagai masukan awal dari sistem pengenalan ekspresi wajah manusia menggunakan *Convolutional Neural Network* adalah data “CASIA Multi-Spectral Palmprint V1.0”. *CASIA Multi-Spectral Palmprint V1.0*adalah dataset yang terdiri dari 600 citra *palm vein* yang diambil dari 100 orang, masing-masing memiliki enam citra.. Tiap kelas pada database *palm vein* diambil dalam dua sesi dengan interval lebih dari satu bulan. Masing-masing sesi terdiri dari tiga citra. Contoh dataset *palm vein* bisa dilihat pada Gambar 3.1.



Gambar 3.1 Contoh citra Palm Vein

Dari 600 gambar dataset CASIA ini, akan dilakukan proses Augmentasi untuk menambah jumlah citra. Teknik Augmentasi yang digunakan adalah Scaling dan Rotasi dimana terjadi penambahan jumlah data menjadi 3600 data dengan 100 kelas (individu). Data tersebut dibagi menjadi 2 jenis data yaitu data latih dan data uji. Data latih diproses menggunakan *Convolutional Neural Network* untuk ekstraksi fitur dan metode *Multi Class Support Vector Machine* untuk membangun model, kemudian model digunakan dalam pengenalan terhadap data uji dan kinerja model diukur dengan akurasi, *precision*, dan *recall* pengujian. Spesifikasi lengkap dataset dapat dilihat pada Tabel 3.1.

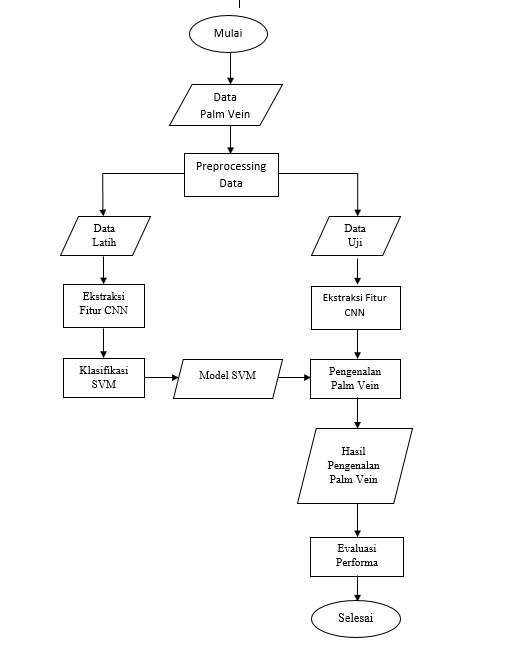
|  |  |
| --- | --- |
| **Keterangan** | **Spesifikasi** |
| Ukuran resolusi asli | 768 x 576 *pixels* |
| Ekstensi | .JPG |
| Jumlah gambar | 600 |
| Jumlah orang | 100 |
| Jumlah sesi foto | 2 kali |
| Jumlah kelas | 100 kelas |
| Jumlah gambar per kelas | 6 |
| Interval waktu | 1 bulan |
| Ukuran file | 17 - 18 KB |
| Kanal warna | 3 (RGB) |

Tabel 3.1 Spesifikasi awal Dataset

## Desain Umum Sistem

Sistem pengenalan pembuluh darah pada telapak tanga yang dibangun memiliki proses utama diantaranya preprocessing data, pelatihan dan pengujian *Convolutional Neural Network* dan *Multi Class Support Vector Machine*. Diagram alir dari sistem ditunjukan pada Gambar 3.2.

Akan dilakukan *preprocessing* data terlebih dahulu untuk mendapatkan daerah telapak tangan dan pola pembuluh darah. Selain itu juga dilakukan proses *Augmentasi* data untuk memperbanyak jumlah data dalam setiap kelas /individu. Pada setiap data dilakukan preprocessing data menggunakan beberapa metode preprocessing citra meliputi, rotasi lalu diubah menjadi *grayscale* dan metode *presprocessing*.



Gambar 3.2 Diagram alir sistem yang dibangun

Proses pelatihan adalah proses pembuatan model klasifikasi pola pembuluh darah pada telapak tangan manusia. Data latih yang telah dilakukan preprocessing akan diekstraksi fiturnya yakni melalui proses konvolusi, *pooling*, dan fungsi-fungsi aktivasi. Selanjutnya, hasil dari proses pelatihan tersebut akan menjadi model untuk proses klasifikasi. Data uji juga akan dilakukan preprocessing data dahulu sebelum diekstraksi fiturnya untuk menjadi *input* dalam proses pengujian.

Proses pengujian memanfaatkan metode *Multi Class Support Vector Machine* lalu akan didapatkan hasil prediksi berupa label kelas. Selanjutnya hasil prediksi tersebut akan dibandingkan dengan label kelas sebenarnya, maka dapat dievaluasi nilai akurasi, *precision*, dan *recall* dari model tersebut.

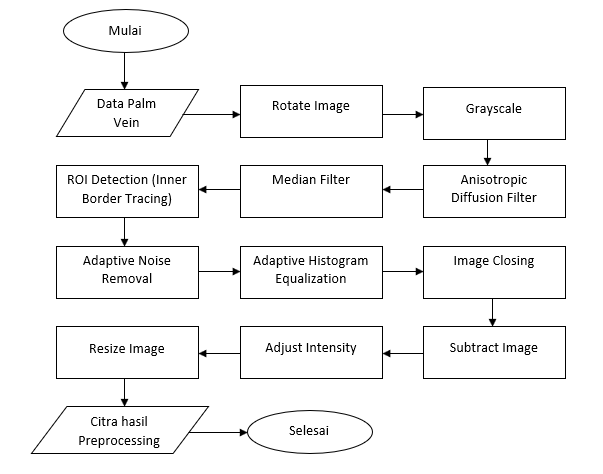
### Tahap Preprocessing Data

Pada tugas akhir ini, preprocessing data yang dilakukan adalah rotasi lalu melakukan perubahan format gambar menjadi *grayscale*, perubahan ukuran gambar, lalu melakukan beberapa metode preprocessing pada gambar untuk mendapatkan pola pembuluh darah. Diagram alir preprocessing data dapat dilihat pada Gambar 3.3.

Setiap data akan di rotasi sebesar lalu diubah menjadi format *grayscale*. Lalu dilakukan tahapan preprocessing meliputi deteksi ROI, perbaikan kualitas citra, dan penonjolan pembuluh darah pada telapak tangan. Untuk setiap tahapan preprocessing disertai dengan metode-metode preprocessing diantaranya:

#### ROI Detection

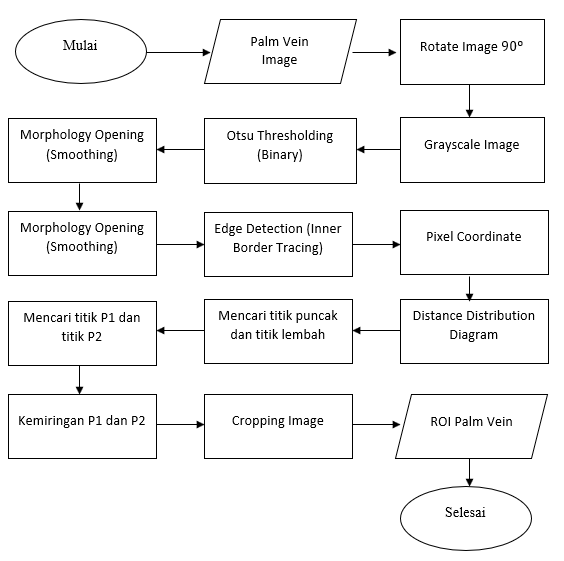
Pengambilan ROI pada tahapan ini menggunakan algoritma *Inner Border Tracing* untuk mendapatkan bagian utama dari telapak tangan. Diagram alir proses pengambilan ROI dari citra Palm Vein ditunjukkan pada Gambar 3.4.



Gambar 3.3 Diagram alir preprocessing data tanpa augmentasi

Algoritma inner border tracing menggunakan 8-connectivity

yang berarti akan mencari piksel tetangga dari 8 arah. Inisialisasi untuk mencari posisi piksel tetangga pada 8-connectivity adalah 7. Untuk mendapatkan nilai arah dari piksel tetangga, apabila nilai dari arah pencarian sebelumnya adalah bilangan genap maka menggunakan rumus (posisi piksel + 7) mod8. Sedangkan apabila nilai dari pencarian sebelumnya adalah bilangan ganjil, maka menggunakan rumus (posisi piksel + 6) mod 8. Lalu, untuk mencari nilai arah baru hanya perlu mengikuti arah berlawanan dengan jarum jam. Setelah mencari koordinat piksel, dilanjutkan menghitung jarak setiap pixel border menuju piksel bagian tengah bawah citra menggunakan diagram distribusi jarak.



Gambar 3.4 Diagram alir preprocessing ROI detection

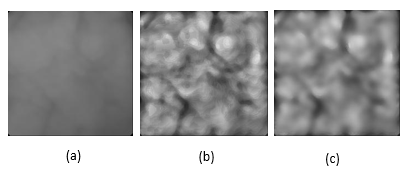
Dari sini, list dari border yang dibentuk akan disimpan. List piksel inilah yang nantinya akan digunakan dalam pengambilan dua titik pada ROI yaitu titik P1 dan P2. P1 adalah lembah yang berada disebelah jari kelingking, dan P2 adalah titik tembah yang berada diantara jari tengah dan jari telunjuk. ROI akan didapatkan dari titik P1 dan P2 yang berbentuk persegi. Dimana Panjang sisinya sama seperti jarak antara P1 dan P2. Hasil dari ROI detection ditujukkan pada Gambar 3.5.



Gambar 3.5 Hasil preprocessing ROI detection

#### Perbaikan Citra

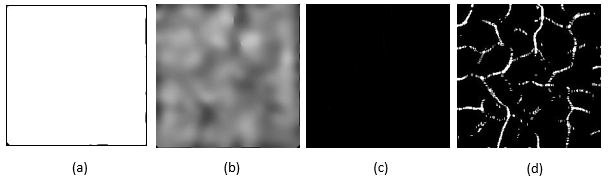
Setelah mendapatkan ROI palm vein, selanjutnya adalah melakukan perbaikan citra agar dapat diolah dan digunakan pada tahap selanjutnya. Tahap perbaikan citra ini seperti penghilangan noise dan enhancement menggunakan median filter, adaptive histogram equalization, dan adaptive noise removal. Hasil perbaikan citra ditunjukkan Gambar 3.6.



Gambar 3.6 Hasil Perbaikan Citra (a) Median Filter (b) Adaptive Histogram Equalization (c) Noise Removal

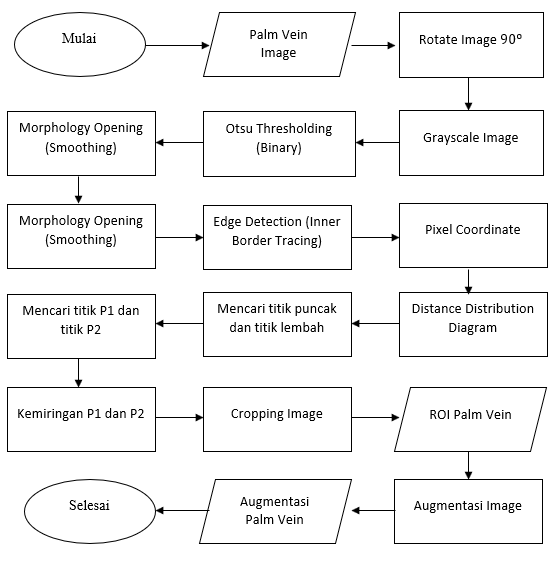
#### Penonjolan Pembuluh Darah Telapak Tangan

Setelah citra diperbaiki, selanjutnya adalah proses penonjolan pembuluh darah telapak tangan. Pada tahap ini dilakukan pemisahan objek pembuluh darah dengan telapak tangan yaitu dengan anisotropic diffusion filter, morfologi closing, serta substract image hasil dari anisotropic diffusion filter dengan citra hasil morfologi closing. Dan dilakukan peningkatan intensitas pada gambar menggunakan metode Adjust Intensity pada citra hasil substract. Hasil penonjolan pembuluh darah telapak tangan ditunjukkan pada Gambar 3.7.



Gambar 3.7 Hasil penonjolan pembuluh darah (a) ADF (b) Closing (c) Substract (d) Adjust Intensity

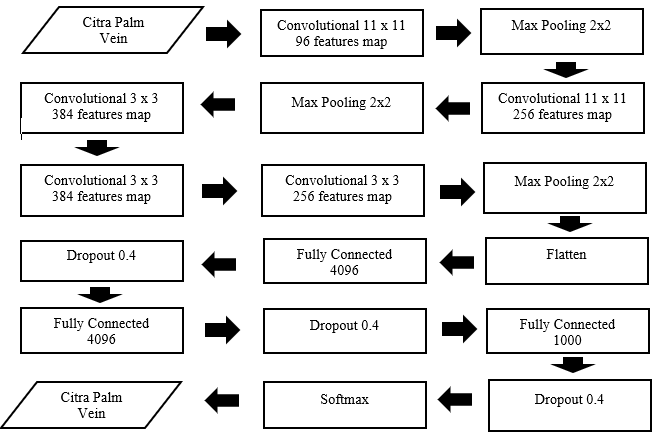
Selain praproses yang telah dijelaskan di atas, akan dilakukan juga proses augmentasi data untuk memperbanyak data latih yang dimasukkan ke arsitektur CNN. Proses augmentasi data dilakukan untuk memperbanyak varian data. Banyaknya varian data diharapkan dapat meningkatkan kualitas data dan model yang dibangun dapat memiliki hasil yang lebih baik. Augmentasi data yang dilakukan adalah Scaling dan Rotasi Image sebesar . Hasil dari Augmentasi data untuk setiap gambar yaitu sebanyak 5 gambar. Sehingga jumlah total data hasil Augmentasi sebanyak 600 x 5 yaitu 3600 data gambar dengan 100 kelas. Untuk proses Augmentasi data menggunakan Image Generator yang telah disediakan pada library keras. Diagram alir praproses data dengan augmentasi dapat dilihat pada Gambar 3.8.



Gambar 3.8 Diagram alir preprocessing data dengan augmentasi

### Tahap Ekstraksi Fitur

Sebelum memasukki tahap pembangunan model maka akan dilakukan tahap ekstraksi fitur terlebih dahulu yang bertujuan untuk mengambil fitur penting dari suatu gambar yang nantinya akan digunakan sebagai atribut informasi dari suatu pada saat dilakukan pembangunna model. CNN (*Convolutional Neural Network*) yang digunakan terdiri dari *convolutional layer*, *max pooling layer* dan *fully connected layer*. Arsitektur CNN yang digunakan pada tugas akhir ini adalah *Alex-Net* yang terdiri dari 8 layer. Arsitektur CNN *Alex-Net* dapat dilihat pada Gambar 3.9.



Gambar 3.9 Arsitektur Alex-Net

Berikut detail arsitektur CNN Alex-Net:

1. Input untuk arsitektur CNN adalah citra hasil Augmentasi.
2. Setiap imput gambar akan melewati 5 *Convolution Layer dan* 3 *Max Pooling Layer.*
3. Pada setiap *Convolution Layer* memiliki jumlah filter dan ukuran yang berbeda-beda. Sedangkan ukuran untuk setiap *Max Pooling Layer* sama yaitu sebesar 2 x 2.
4. Setelah melalui *Convolution Layer* dan *Max Pooling Layer* maka citra akan melalui *Flatten* yaitu mengubah matrix menjadi vector.
5. Hasil dari *Flatten* yang berupa vector akan dijadikan sebagai atribut atau fitur dari setiap data gambar yang akan digunakan untuk pembentukkan model menggunakan algoritma *Multi Class Support Vector Machine.*
6. *Dropout layer* memiliki nilai probabilits 0.4 secara keseluruhan.
7. CNN dilatih menggunakan *Adam Optimizer* dengan *learning rate* 0,0001.
8. Data yang digunakan memiliki 3 kanal warna (*RGB*)

Proses ekstraksi fitur ini dilakukan menggunakan metode FCN (*Fully Convolutional Layer*) *Alex-Net*. Dimana nilai vector yang di dapatkan dari Ekstraksi Fitur dengan 5 *Convolution Layer*, 3 *Max Pooling Layer* dan *Flatten* akan dijadikan sebagai input atribut data untuk tahap pembentukkan model. Detail layers dan parameter dapat dilihat pada Tabel 3.2.

|  |  |  |  |
| --- | --- | --- | --- |
| ***Layer*** | ***Input*** | ***Output*** | **Spesifikasi** |
| *Input Layer* | (224*, 224*, 3) | | - |
| *Convolution Layer* 1 | (224, 224, 3) | (56, 56, 96) | *Filter*: 11 x 11 – 96  *Stride*: 4 x 4  Fungsi: ReLU |
| *Max Pooling Layer* 1 | (56, 56, 96) | (28, 28, 96) | *Kernel*: 2 x 2  *Stride*: 2 x 2 |
| *Convolution Layer* 2 | (56, 56, 96) | (26, 26, 256) | *Filter*: 11 x 11 – 256  *Stride*: 1 x 1  Fungsi: ReLU |
| *Max Pooling Layer* 2 | (26, 26, 256) | (13, 13, 256) | *Kernel*: 2 x 2  *Stride*: 2 x 2 |
| *Convolution Layer* 3 | (13, 13, 256) | (11, 11, 384) | *Filter*: 3 x 3 – 384  S*tride*: 1 x 1  Fungsi: ReLU |
| *Comvolutin Layer 4* | (11, 11, 384) | (9, 9, 384) | *Filter*: 3 x 3 – 384  S*tride*: 1 x 1  Fungsi: ReLU |
| *Convolution Layer 5* | (9, 9, 384) | (7. 7, 256) | *Filter*: 3 x 3 – 256  S*tride*: 1 x 1  Fungsi: ReLU |
| *Max Pooling Layer* 3 | (7. 7, 256) | (3, 3, 256) | *Filter*: 2 x 2  *Stride*: 2 x 2 |
| *Flatten* | (3, 3, 256) | (2304) | - |

Tabel 3.2 Arsitektur Alex-Net

### Tahap Membagi Data

Setelah melakukan preprocessing terhadap data dan ekstraksi fitur. Data akan dibagi menjadi 2, yaitu data latih dan data uji. Jumlah data asli sebanyak 600 pada 100 individu, maka setiap individu memiliki 6 data. Sehingga untuk pembagian data uji dan data latih menjadi 1:5. Dikarenakan jumlah data tergolong sedikit dan ditakutkan akan memperngaruhi dalam pembentukkan model, maka dilakukan proses augmentasi citra untuk mengembangkan jumlah data menjadi 3600 data pada 100 kelas. Sehingga jumlah data uji sebanyak 600 data dan data latih sebnayak 3000 data dengan perbandingan 1:5.

### Tahap Pembentukan Model

Setelah melakukan tahap Ekstraksi Fitur yang menghasilkan sebuah nilai vector untuk setiap data, maka tahap selanjutnya adalah tahapan pembentukan model klasifikasi dengan menggunakan metode Multi Class Support Vector Machine. Pada tahap pembangunan model ini, data yang digunakan untuk membangun model adalah data latih sebanyak 3000 data pada 100 individu / kelas. Detail parameter dari metode Multi Class Support Vector Machine dapat dilihat pada Tabel 3.3:

|  |  |
| --- | --- |
| **Variabel Parameter** | **Value** |
| C | 1.0 |
| Kernel | Linear |
| Gamma | 0.001 |
| Tolerance | 0.001 |
| Maximum Iteration | -1 |

Tabel 3.3 Multi-Class SVM

### Tahap Pelatihan dan Pengujian CNN

Proses pelatihan memanfaatkan data latih untuk membangun model SVM. Sedangkan untuk pengujian model yang telah terbentuk menggunakan data uji yang berjumlah 600 data pada 100 individu/kelas. Dan hasil dari pengujian model akan di evaluasi menggunakan variabel evaluasi *precision, recall, f1-score* dan *accuracy.*

# BAB IV IMPLEMENTASI

Bab ini menjelaskan mengenai implementasi perangkat lunak dari rancangan sistem yang telah dibahas pada Bab 3 meliputi kode program dalam perangkat lunak. Selain itu, implementasi dari tiap proses, parameter masukan, keluaran, dan beberapa keterangan yang berhubungan dengan program juga dijelaskan.

## Lingkungan Implementasi

Dalam mengimplementasikan aplikasi pengenalan ekspresi manusia diperlukan beberapa perangkat pendukung sebagai berikut.

### Perangkat Keras

Implementasi tugas akhir ini menggunakan desktop *personal computer* (PC) HP-580-011d. Sistem operasi yang digunakan adalah Windows 10 64-bit. PC yang digunakan memiliki spesifikasi Intel Core i7-7700 dengan kecepatan 3,6 GHz, *Random Access Memory* (RAM) sebesar 16 GB, dan mempunyai *Graphics Processing Unit* (GPU) yaitu NVIDIA GeForce GTX 1060 sebesar 3 GB.

### Perangkat Lunak

PC dari sisi perangkat lunak memiliki spesifikasi antara lain menggunakan bahasa pemograman Python 3.6 dilengkapi dengan *library* antara lain OpenCV, Tensorflow, Keras, Numpy, Matplotlib dan Scikit-learn. Selain itu juga menggunakan bahasa pemrogaman *Matlab* untuk *preprocessing.*

## Implementasi Preprocessing Data

Pada subbab ini akan dijabarkan implementasi pada tahap preprocessing data, yaitu ROI Detection, perbaikan citra dan penonjolan pembuluh darah telapak tangan.

### Implementasi ROI *Detection*

Proses ROI *Detection* dibagi menjadi beberapa tahapan diantaranya:

* + - 1. **Perbaikan Citra**

Proses perbaikan citra dengan menggunakan metode Otsu’ thresholding dan Morfologi opening pada citra input. Dimana otsu thresholding akan mengubah gambar menjadi binary dan morfologi opening akan menghilangkan noise.

1. #otsu's thresholding
2. blur = cv2.GaussianBlur(img\_gray, (5,5),0)
3. ret3, img\_thresh = cv2.threshold(blur, 0, 255, cv2.THRESH\_BINARY+cv2.THRESH\_OTSU)
4. #morphological opening
5. sz = 20
6. kernel = cv2.getStructuringElement (cv2.MORPH\_ELLIPSE, (2\*sz-1, 2\*sz-1))
7. opening = cv2.morphologyEx(img\_thresh, cv2.MORPH\_OPEN, kernel)

Kode Sumber 4.1 Fungsi Noise Removal

* + - 1. **Inner Border Tracing**

1. column = np.argwhere(np.any(img==255, axis = 0))[0]
2. c = column[0]
4. row\_columnX = img[:,c:c+1]
5. row = np.argwhere(np.any(row\_columnX==255, axis = 1))[0]
6. r = row[0]
8. # new Matrix for save new image
9. ctrImg = np.zeros(img.shape)
10. height, width = img.shape
11. ctrImg[r,c] = 255
12. coor = [r,c]
13. startCoor = [r,c]
15. # Half Wrist index
16. maksCoorR, maksCoorC = img.shape
17. coorHalfWrist = [int(maksCoorR), int(maksCoorC/2)]
18. newCoor = [0,0]
19. coorDis = []
20. direction = []
21. dir = 7
22. indikator = 1
23. count = 0
24. counts = 0
25. while indikator == 1:
26. counts += 1
27. if dir % 2 == 0:
28. startDir = (dir+7) % 8
29. else:
30. startDir = (dir+6) % 8
31. currentDir = startDir
32. for i in range(8):
33. if currentDir == 8:
34. currentDir = 0
35. if currentDir == 0:
36. posisi = [0, 1]
37. elif currentDir == 1:
38. posisi = [-1, 1]
39. elif currentDir == 2:
40. posisi = [-1, 0]
41. elif currentDir == 3:
42. posisi = [-1, -1]
43. elif currentDir == 4:
44. posisi = [0, -1]
45. elif currentDir == 5:
46. posisi = [1, -1]
47. elif currentDir == 6:
48. posisi = [1, 0]
49. elif currentDir == 7:
50. posisi = [1, 1]
51. dir = currentDir
52. newCoor[0] = coor[0] + posisi[0]
53. newCoor[1] = coor[1] + posisi[1]
54. if newCoor[0] >= 0 and newCoor[1] >= 0 and newCoor[0] < maksCoorR-1 and newCoor[1] < maksCoorC-1:
55. if opening[newCoor[0], newCoor[1]] == 255:
56. ctrImg[newCoor[0], newCoor[1]] = 255
57. coor[0] = newCoor[0]
58. coor[1] = newCoor[1]
59. a = math.pow((newCoor[0] - coorHalfWrist[0] + 1),2)
60. b = math.pow((newCoor[1] - coorHalfWrist[1] + 1),2)
61. c = a + b
62. distances = math.sqrt(c)
63. coorDis.append([])
64. direction.append([])
65. coorDis[count].append(newCoor[0])
66. coorDis[count].append(newCoor[1])
67. coorDis[count].append(distances)
68. direction[count].append(dire)
69. count += 1
70. break
71. else:
72. currentDir += 1
73. else:
74. currentDir += 1
75. if startCoor == coor:
76. indikator = 0

Kode Sumber 4.2 Edge Detection

Pada tahap ini dilakukan metode Edge Detection untuk mendapatkan daerah dari telapak tangan menggunakan algoritma Inner Border Tracing. Untuk impelementasi ditunjukkan pada Kode Sumber 4.2. Pertama-tama baris ke-1 hingga ke-6 dilakukan proses pencarian posisi dari koordinat pixel pertama yang akan digunakan sebagai posisi awal dalam melakukan pencarian pixel tetangga. Setelah itu pada baris ke-9 hingga ke-13 dibuatlah sebuah matrix 2 dimensi baru dengan yang nantinya akan digunakan untuk menyimpan nilai pixel dari citra hasil edge detection. Pada baris ke-16 dan ke-17 didapatkan sebuah titik koordinat yang nantinya akan digunakan untuk titik pusat dalam menentukan jarak antar pixel koordinat.

Pada baris ke-19 hingga ke-78 adalah tahapan inner border tracing dengan 8-connetivity dan arah yang berlawan dengan jarum jam. Metode ini meliputi pengecekan nilai pixel pada tiap koordinat untuk mengetahui daerah batas dari telapak tangan dengan menggunakan arah dari variable dir. Pada baris ke-29 hingga baris ke-53 adalah tahapan untuk mendapatkan nilai arah dari pixel tetangga. Apabila nilai dari arah pencarian sebelumnya adalah bilangan genap maka menggunakan rumus (posisi pixel + 7) mod 8. Sedangkan apabila nilai dari pencarian sebelumnya adalah bilangan ganjil, maka menggunakan rumus (posisi pixel + 6) mod 8. Selain itu pada baris ke-61 hingga baris ke-64 dilakukan perhitungan untuk jarak dari setiap koordinat pixel dari tepi telapak tangan.

* + - 1. **Diagram Distribusi Jarak**

1. sumbuX = []
2. sumbuX = np.arange(len(coorDis))
3. sumbuX = np.array(sumbuX)
4. # minumum Distance
5. distances = []
6. for i in range(len(coorDis)):
7. distances.append(coorDis[i][2])
8. minDistance = np.amin(distances)
9. #index pixel with minimum distance
10. indexDistance = np.argwhere(coorDis == minDistance)
11. for i in indexDistance:
12. if i[1] == 2:
13. r = i[0]
14. break
15. c = 0
16. x, y, d = coorDis[r]
17. # sequence
18. newCoorDis = []
19. for i in range(len(coorDis) - r):
20. newCoorDis.append(coorDis[i+r])
21. for i in range(r):
22. newCoorDis.append(coorDis[i])
23. newCoorDis = np.array(newCoorDis)
24. height, width = ctrImg.shape
25. image = copy.deepcopy(ctrImg)
27. whiteDis = float(570)
28. r = []
29. c = []
30. for i in range(len(newCoorDis)):
31. if newCoorDis[i][0] > whiteDis:
32. newCoorDis[i][2] = 0
34. distancess = []
35. for i in range(len(newCoorDis)):
36. distancess.append(newCoorDis[i][2])
37. distancess = np.array(distancess)
38. s = 50
39. output = fpeak(sumbuX, distancess, s)
40. for i in range(len(output)):
41. if i == len(output) - 1:
42. output[i].append(0)
43. break
44. if output[i][1] == output[i+1][1]:
45. output[i].append(1)
46. else:
47. output[i].append(0)
48. improveOut = []
49. for i in output:
50. if i[2] == float(0):
51. improveOut.append(i)
52. # sorted(improveOut, reverse=True)
53. output = improveOut

Kode Sumber 4.3 Distance Distribution Diagram

1. def getPeak(Data, i, s):
2. if (i-s) < 1:
3. top = 0
4. else:
5. top = i - s - 1
6. y = Data[1]
8. if (i+s) > len(y):
9. bottom = len(y) - 1
10. else:
11. bottom = i + s - 1
12. p = []
13. if y[i] == max(y[top:bottom]) or y[i] == min(y[top:bottom]):
14. p = [Data[0][i], Data[1][i]]
15. else:
16. p.append(float('nan'))
17. p.append(float('nan'))
18. return p
19. def fpeak(sumbuX, distancess, s):
20. rx = len(sumbuX)
21. ry = len(distancess)
22. if s == 0:
23. s = 1
24. Data = np.vstack((sumbuX, distancess))
25. output = []
26. for i in range(rx):
27. isP= getPeak(Data, i, s)
28. if np.isnan(isP).sum() == 0:
29. output.append(isP)
30. return output

Kode Sumber 4.4 Fungsi *fpeak*

Pada tahap ini dilakukan perhitungan untuk mendapatkan posisi dari titik puncak (ujung jari) dan titik lembah (pada sela-sela jari) yang nantinya akan diambil 2 bagian dari titik lembah untuk dijadikan sebagai nilai P1 dan P2. Untuk impelementasi ditunjukkan pada Kode Sumber 4.3. Pada baris ke-5 hingga baris ke-19 adalah tahapan untuk mendapatkan koordinat pertama dari jarak terkecil. Pada baris ke-43 adalah tahapan pemanggilan fungsi *fpeak* yang mana berfungsi untuk mencari titik puncak dan titik lembah dari telapak tangan. Untuk implementasi fungsi *fpeak* ditunjukkan pada Kode Sumber 4.4. Dan pada akhirnya jumlah titik puncak yang didapatkan sebanyak 5 dan titik lembah sebanyak 6.

* + - 1. **Perbaikan Citra**

1. pixelP1 = output[0][0]
2. pixelP2 = output[6][0]
3. coorP1 = [newCoorDis[int(pixelP1)][0], newCoorDis[int(pixelP1)][1]]
4. coorP2 = [newCoorDis[int(pixelP2)][0], newCoorDis[int(pixelP2)][1]]
5. # kemiringan p1 dam p2
6. temp = (coorP2[1] - coorP1[1]) / (coorP2[0] - coorP1[0])
7. deg = math.atan(temp) / math.pi \* 180
8. newImg = misc.imrotate(img\_rotate,90 - deg,interp='bilinear')
9. temp = np.zeros\_like(img)
10. height, width = img.shape
11. temp[int(coorP1[0])][int(coorP1[1])] = 255
12. temp[int(coorP2[0])][int(coorP2[1])] = 255
13. tempRotate = misc.imrotate(temp,90 - deg,interp='bilinear')
14. tempIndex = np.argwhere(tempRotate>0)
15. newCoorP1 = [tempIndex[0][0], tempIndex[0][1]]
16. newCoorP2 = [tempIndex[len(tempIndex)-1][0], tempIndex[len(tempIndex)-1][1]]
17. a = math.pow((newCoorP1[0]-newCoorP2[0]),2)
18. b = math.pow((newCoorP1[1]-newCoorP2[1]),2)
19. distance = math.sqrt(a+b)
20. distance = int(distance)
21. image = copy.deepcopy(newImg)
22. # ROI output
23. fiturI = newImg[int(newCoorP1[0]):int(newCoorP1[0]+distance), int(newCoorP1[1]):int(newCoorP1[1]+distance)]

Kode Sumber 4.5 Region of Interest

Pada tahap ini dilakukan penentuan nilai P1 dan P2 yang nantinya akan digunakan sebagai ROI dari telapak tangan. Untuk impelementasi ditunjukkan pada Kode Sumber 4.5. Pada baris ke- hingga baris ke-5 adalah pengambilan koordinat pixel untuk nilai P1 dan P2. Pada baris ke-7 hingga baris ke-18 adalah tahapan untuk mendapatkan nilai derajad kemiringan dari P1 dan P2 yang nantinya akan digunakan untuk merotasi gambar agar membantu dalam penyempurnaan *cropping* gambar.

Pada baris ke-20 hingga baris ke-23 adalah tahapan untuk mendapatkan nilai P1 dan P2 yang baru dari gambar yang telah dirotasi. Pada baris ke-25 hingga baris ke-28 adalah tahapan untuk mendapatkan jarak dari P1 dan P2 yang nantinya akan digunakan sebagai nilai lebar dari hasil *cropping.* Pada baris ke-32 hingga baris ke-36 adalah tahapan untuk cropping.

### Implementasi Discrete Wavelet Transform

Proses perubahan gambar dari *raw image* ke *Discrete Wavelet Transform* (DWT)diimplementasikan pada Kode Sumber 4.2. Proses dilakukan pada gambar dengan memanfaatkan fungsi *wavedec2* dari PyWavelets. Dimana parameter yang digunakan adalah *resized\_img* sebagai input gambar yang akan dilakukan proses DWT, *wavelet* sebagai jenis DWT yang akan digunakan yaitu *haar wavelet*, dan *level* sebagai kedalaman *wavelet* yang digunakan.

* 1. wp = pywt.wavedec2(data = resized\_img, wavelet = 'haar', level = 1)

Kode Sumber 4.2 Fungsi *Discrete Wavelet Transform*

## Implementasi Augmentasi Data

Pada subbab ini akan dijelaskan proses augmentasi data yaitu refleksi dan memperbesar ukuran gambar.

### Refleksi

Refleksi yang dilakukan disini adalah refleksi horizontal. Refleksi diimplementasikan pada Kode Sumber 4.3. Proses refleksi horizontal dilakukan dengan menggunakan fungsi *ImageDataGenerator* dari library Keras. Parameter *horizontal\_flip=True* menandakan refleksi horizontal akan diimplementasikan pada data gambar. Lalu setelah objek *ImageDataGenerator* dibuat, akan dipanggil fungsi *flow* untuk menerapkan pengaturan augmentasi yang telah dibuat.

### Perbesar Ukuran Gambar

Perbesaran ukuran gambar dilakukan untuk mencari keragaman yang lebih variatif dari data. Perbesaran ukuran gambar diimplementasikan pada Kode Sumber 4.3. Proses perbesaran ukuran gambar dilakukan dengan menggunakan fungsi *ImageDataGenerator* dari library Keras. Parameter *zoom\_range=*0.1 menandakan perbesaran ukuran gambar sebesar 10% akan diimplementasikan pada data gambar secara acak. Lalu setelah objek *ImageDataGenerator* dibuat, akan dipanggil fungsi *flow* untuk menerapkan pengaturan perbesaran ukuran gambar yang telah dibuat.

* 1. datagen = ImageDataGenerator(horizontal\_flip=True, zoom\_range=0.1, fill\_mode='nearest')
  2. datagen.flow(image, save\_to\_dir=save\_dir, save\_prefix=fileName, save\_format='JPG')

Kode Sumber 4.3 Implementasi augmentasi data

## Implementasi Pembangunan Arsitektur

Pada subbab ini akan dijabarkan implementasi fungsi-fungsi pada tahap pembangunan model. Arsitektur CNN dimulai dengan mengimplementasi desain *convolution layer* keempat *subband.* Implementasi tersebut tercantum pada Kode Sumber 4.4 dengan fungsi *wave\_convolution\_layer* dengan layer-layer sesuai dengan spesifikasi yang dijelaskan pada Bab 3. Fungsi ini akan mengkonvolusi semua *subband* yang menjadi masukan model. Lalu, implementasi dilanjutkan dengan menggabungkan 4 hasil konvolusi *subband* tersebut dengan sebuah *Concatenation Layer* yang tercantum pada Kode Sumber 4.5. Lalu pada tahap terakhir, hasil C*oncatenation Layer* akan dikonvolusi lagi dengan 2 *convolutional layer* sesuai yang dijelaskan pada Bab 3 sebelum akhirnya akan dimasukkan ke fungsi aktivasi *Softmax*. Pembangunan arsitektur memanfaatkan Keras dan Tensorflow.

1. **def** wave\_convolution\_layer(inputs, size):
2. conv\_layer = Conv2D(32, (5,5), strides=(1,1), activation='relu', kernel\_initializer='he\_normal')(inputs)
3. conv\_layer = BatchNormalization()(conv\_layer)
4. conv\_layer = MaxPooling2D((2, 2))(conv\_layer)
5. conv\_layer = Conv2D(32, (5,5), strides=(1,1), activation='relu', kernel\_initializer='he\_normal')(conv\_layer)
6. conv\_layer = BatchNormalization()(conv\_layer)
7. conv\_layer = MaxPooling2D((2, 2))(conv\_layer)
8. conv\_layer = Conv2D(32, (5,5), strides=(1,1), activation='relu', kernel\_initializer='he\_normal')(conv\_layer)
9. conv\_layer = BatchNormalization()(conv\_layer)
10. conv\_layer = MaxPooling2D((2, 2))(conv\_layer)
11. **return** conv\_layer

Kode Sumber 4.4 Fungsi pembangunan konvolusi *wavelet*

Pada baris 1, parameter *inputs* adalah *input layer* dan parameter *size* adalah dimensi data masukan yakni 128x128 (*image length, image width)*. Baris 2 adalah inisialisasi arsitektur yang menggabung *input layer* tersebut dengan *layer* yang lain. Baris 2-10 masing-masing adalah penambahan *layer* yang dibutuhkan.

1. **def** model\_build():
2. inputs1 = Input(shape=(127, 127, 1))
3. conv\_layer1 = wave\_convolution\_layer(inputs1, 127)
4. inputs2 = Input(shape=(128, 128, 1))
5. conv\_layer2 = wave\_convolution\_layer(inputs2, 128)
6. inputs3 = Input(shape=(129, 129, 1))
7. conv\_layer3 = wave\_convolution\_layer(inputs3, 129)
8. inputs4 = Input(shape=(130, 130, 1))
9. conv\_layer4 = wave\_convolution\_layer(inputs4, 130)
10. concat = keras.layers.Concatenate()([conv\_layer1, conv\_layer2, conv\_layer3, conv\_layer4])
11. conv\_layer = Dropout(0.4)(concat)
12. conv\_layer = Conv2D(64, (5,5), strides=(1,1), activation='relu', kernel\_initializer='he\_normal')(conv\_layer)
13. print(conv\_layer.\_keras\_shape)
14. conv\_layer = BatchNormalization()(conv\_layer)
15. conv\_layer = Activation('relu')(conv\_layer)
16. conv\_layer = MaxPooling2D((2,2))(conv\_layer)
17. conv\_layer = Dropout(0.2)(conv\_layer)
18. conv\_layer = Conv2D(7, (4,4), strides=(1,1), activation='relu', kernel\_initializer='he\_normal')(conv\_layer)
19. flatten = Flatten()(conv\_layer)
20. outputs = Dense(7, activation='softmax')(flatten)
21. model = Model(inputs=[inputs1, inputs2, inputs3, inputs4], outputs=outputs)
22. **return** model

Kode Sumber 4.5 Fungsi pembangunan arsitektur CNN

Pada baris 2-9 dilakukan inisialisasi *input layer* untuk setiap *subband* dengan menggunakan fungsi *wave\_convolution\_layer* yang telah disebutkan pada Kode Program 4.4. Baris 10 adalah penggabungan 4 *convolution layer* setiap *subband* dengan menggunakan *Concatenation Layer*. Baris 11-21 masing-masing adalah penambahan *layer* yang dibutuhkan.

Arsitektur yang dibangun terdiri dari 6 macam *layer* antara lain Convolution2D, MaxPooling2D, BatchNormalization, Flatten, Dropout, dan Dense.

Convolution2D adalah implementasi *Convolution Layer* dalam bentuk 2 dimensi. Convolution2D menggunakan 6 parameter. Berikut penjelasan parameter-parameter tersebut:

1. Parameter *filters* adalah jumlah *filter* atau *kernel*.
2. Parameter *kernel\_size* adalah ukuran *filter* atau *kernel*.
3. Parameter *strides* adalah ukuran *stride* yang digunakan.
4. Parameter *acitivation* adalah fungsi aktivasi apa yang digunakan, terdapat banyak pilihan fungsi aktivasi yang disediakan oleh Keras, salah satunya adalah ‘relu’ yakni fungsi ReLU.
5. Parameter *kernel\_initializer* untuk menentukan weights awal matrix.
6. Parameter *padding* menentukan apakah *output* diaplikasikan *Zero Padding*. Jika *padding* bernilai ‘*valid*’ maka *output* tidak diaplikaskan *Zero Padding* dan jika bernilai ‘*same’* maka bentuk keluaran akan diaplikasikan *Zero Padding* untuk menjaga keluaran sama dengan *input*. Nilai default *padding* jika tidak dicantumkan adalah *‘valid’.*

MaxPooling2D adalah implementasi *Max Pooling Layer* dalam bentuk 2 dimensi. MaxPooling2D menggunakan 2 parameter. Berikut penjelasan parameter-parameter tersebut:

1. Parameter *pool\_size* adalah ukuran *kernel* *Pooling Layer*.
2. Parameter *strides* adalah ukuran *stride* yang digunakan. Nilai default *strides* jika tidak dicantumkan adalah *‘None’* yang berarti ukuran *stride* akan disamakan dengan *pool\_size*.

Flatten biasa digunakan oleh arsitektur CNN pada Keras ketika *Convolution* dan *Max Pooling Layer* akan memasuki *Fully Connected Layer*. Hal ini karena Flatten memiliki fungsi untuk merubah keluaran sebelumnya yang berupa *feature map* menjadi *feature vector*, sehingga dapat dijadikan *input* pada *Fully Connected Layer*.

Dropout adalah implementasi teknik *dropout* yang berfungsi membuat beberapa neuron yang dipilih secara acak untuk tidak dipakai selama proses pelatihan. Hal ini dilakukan untuk mengurangi *overfitting*. Dropout memiliki parameter *rate* yakni nilai probabilitas yang dipakai dalam menentukan secara acak neuron.

Dense adalah implementasi *Fully Connected Layer*. Dense menggunakan 2 parameter. Berikut penjelasan parameter-parameter tersebut:

1. Parameter *units* adalah jumlah neuron.
2. Parameter *activation* adalah fungsi aktivasi apa yang digunakan. Terdapat banyak pilihan fungsi aktivasi yang disediakan oleh Keras, ada ‘relu’ atau fungsi ReLU dan ‘softmax’ atau fungsi *softmax* yang biasa digunakan pada akhir arsitektur untuk mengklasifikasikan label kelas.

Pemanggilan fungsi pembangunan CNN diimplementasikan pada Kode Sumber 4.6.

1. model = model\_build ()

Kode Sumber 4.6 Pemanggilan fungsi pembangunan arsitektur CNN

## Implementasi Pelatihan dan Evaluasi CNN

Setelah pembangunan arsitektur *Convolutional Neural Network*, dilakukan proses pelatihan dan pengujian. Data akan dibagi menjadi 2, yaitu data latih dan data uji dengan jumlah data latih sebanyak 3900 gambar dan jumlah data uji sebanyak 1000 gambar. Dikarenakan data gambar setiap orang akan muncul 2 kali dalam dataset, maka pembagian dilakukan dengan cara memasukkan semua data gambar sesi pertama (sesi A) ke data pelatihan, lalu data gambar sesi kedua (sesi B) akan diambil secara acak sebanyak 1000 gambar untuk dimasukkan ke data uji dan sisanya akan masuk ke data latih. Pembagian ini dilakukan untuk memastikan semua orang yang ada di dataset pernah masuk ke model setidaknya 1 kali sehingga data terbagi dengan seimbang.

Pembagian data dilakukan dengan cara menggunakan seluruh gambar dari sesi A yaitu gambar dari index 0 sampai 2449. Lalu sesi B akan diambil datanya secara acak yaitu antara index 2450 sampai 4899 seperti yang diimplementasikan pada Kode Sumber 4.7.

1. **def** random\_index(start,end):
2. x = [i **for** i **in** range(start,end)]
3. random.shuffle(x)
4. **return** x
6. random\_index = random\_index(2450,4900)

Kode Sumber 4.7 Pembuatan index acak data uji

Lalu setelah index acak sesi B dibuat, akan diambil seluruh gambar dari sesi A dan 1450 gambar dari sesi B berdasarkan index acak tersebut seperti yang diimplemtasikan pada Kode Sumber 4.8.

1. **for** i **in** range(2450):
2. train\_x.append(image[i][0])
3. train\_y.append(image[i][1])
5. **for** i **in** range(len(random\_index)):
6. **if** i < 1450:
7. train\_x.append(image[random\_index[i]][0])
8. train\_y.append(image[random\_index[i]][1])
9. **else**:
10. test\_x.append(image[random\_index[i]][0])
11. test\_y.append(image[random\_index[i]][1])

Kode Sumber 4.8 Pembagian data latih dan data uji

Proses pelatihan memanfaatkan data latih untuk membangun model CNN. Pelatihan menggunakan *loss function* berupa *Cross Entropy* dan *optimizer* berupa Adam *Optimizer* dengan *learning rate* yang telah ditentukan sebelumnya. Hal ini diimplementasikan pada Kode Sumber 4.9 di baris 1 dan 2. Proses pelatihan akan dijalankan pada *batch size* 39 dan jumlah *epoch* 500. Dalam setiap akhir *epoch* terdapat proses pengujian model terhadap data uji dan didapatkan nilai akurasi*, precision* dan *recall*. Pada baris 3 dilakukan proses pelatihan dengan memanggil fungsi *fit* dari Keras. Dimana *batch\_size* adalah ukuran *batch*, *epochs* adalah jumlah *epoch* yang akan dijalankan, *validation\_data*adalah data uji dan label kelasnya.

1. adam = Adam(lr=0.0001)
2. model.compile(optimizer=adam, loss= 'categorical\_crossentropy', metrics=['accuracy'])
3. model.fit(train\_x, train\_y, batch\_size=39, epochs= 500, verbose=1, validation\_data=(test\_x, test\_y))

Kode Sumber 4.9 Pelatihan CNN

Setelah proses pelatihan selesai, model akan diuji dengan menggunakan data uji. Hal ini diimplementasikan pada Kode Sumber 4.10. Fungsi *evaluate* akan mengevaluasi model dengan data uji dan labelnya.

1. **print**(model.evaluate(test\_x, test\_y))

Kode Sumber 4.10 Pengujian CNN

Pada fungsi *evaluate,* didapatkan nilai akurasi model terhadap data uji. Namun library Keras ini tidak memiliki fitur untuk menghitung *precision* dan *recall*. Sehingga perlu dilakukan cara alternatif untuk melakukan evaluasi terhadap model agar mendapatkan nilai *precision* dan *recall*. Dengan memanfaatkan fungsi *predict\_classes* dari *library* Keras terhadap *test\_x* (data uji). Hal ini diimplementasikan pada Kode Sumber 4.11. Pada baris 7, fungsi *predict\_classes* digunakan untuk mendapatkan nilai prediksi untuk masing-masing kelas dan disimpan pada variabel *pred\_y*. Pada baris 1-5, dibuat sebuah fungsi *return\_to\_label* untuk mengembalikan bentuk data prediksi dari Keras yang masih berupa tipe kelas ketegorikal menjadi tipe kelas yang numerik. Lalu pada baris 9-10, fungsi *return\_to\_label* digunakan untuk mengembalikan *pred\_y* dan *test\_y* ke tipe kelas numerik. Baris 12 memanfaatkan fungsi *classification\_report* dari *library* Scikit-learn untuk mengevaluasi nilai prediksi (*pred*) terhadap nilai sesungguhnya (*test*) untuk masing-masing kelas sehingga didapatkan nilai *precision* dan *recall*.

1. **def** return\_to\_label(y):
2. label = []
3. **for** i **in** range(len(y)):
4. label.append(np.argmax(y[i]))
5. **return** label
6. pred\_y = model.predict(test\_x)
7. test = return\_to\_label(test\_y)
8. pred = return\_to\_label(pred\_y)
9. **print**(classification\_report(test, pred))

Kode Sumber 4.11 Evaluasi *precision* dan *recall*

# BAB V UJI COBA DAN EVALUASI

Bab ini akan membahas mengenai hasil uji coba sistem yang telah dirancang dan dibuat. Uji coba dilakukan untuk mengetahui kinerja sistem dengan lingkungan uji coba yang telah ditentukan.

## Lingkungan Uji Coba

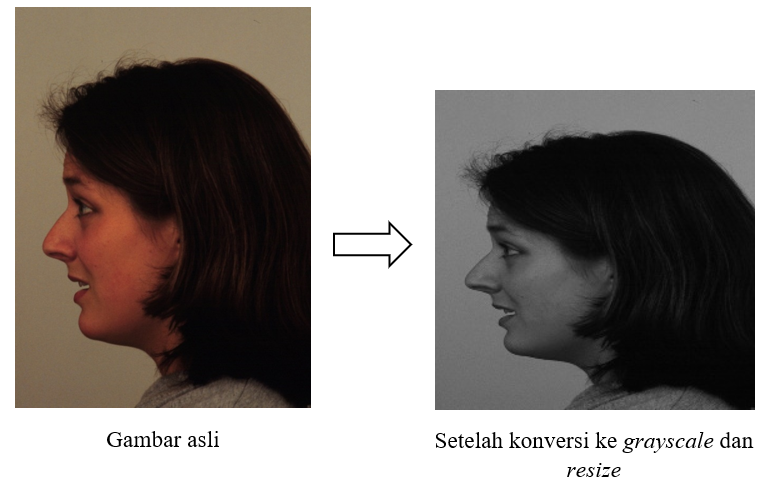
Lingkungan uji coba pada tugas akhir ini adalah sebuah desktop *personal computer* (PC) HP-580-011d. Sistem operasi yang digunakan adalah Windows 10 64-bit. PC yang digunakan memiliki spesifikasi perangkat keras Intel Core i7-7700 dengan kecepatan 3,6 GHz, *Random Access Memory* (RAM) sebesar 16 GB, dan mempunyai *Graphics Processing Unit* (GPU) yaitu NVIDIA GeForce GTX 1060 sebesar 3 GB. Pada sisi perangkat lunak, uji coba pada tugas akhir ini dilakukan dengan menggunakan bahasa pemograman Python 3.6 dilengkapi dengan *library* antara lain Keras, Tensorflow, PyWavelets, OpenCV, Numpy, Matplotlib dan Scikit-learn.

## Dataset

Pada tugas akhir ini, data yang digunakan adalah data Karolinska Directed Emotional Faces (KDEF). Dataset KDEF mengambil gambar setiap individu sebanyak 2 kali. Pengambilan sesi pertama (sesi A) dan kedua (sesi B) diambil dengan interval waktu 1 jam. Dikarenakan data gambar setiap individu akan muncul 2 kali dalam dataset, maka pembagian dilakukan dengan cara memasukkan semua data gambar sesi A ke data pelatihan, lalu data sesi B akan diambil secara acak sebanyak 1000 gambar untuk dimasukkan ke data uji dan sisanya akan masuk ke data latih. Pembagian ini dilakukan untuk memastikan semua individu yang ada di dataset pernah masuk ke model setidaknya 1 kali sehingga data terbagi dengan seimbang.

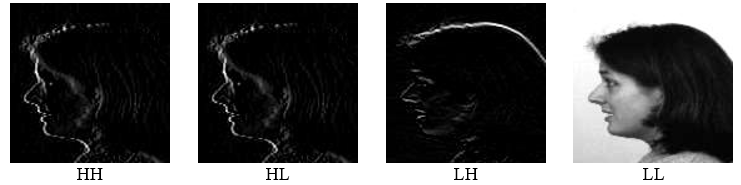
## Hasil Praproses

Sebelum memasuki proses pelatihan dengan arsitektur CNN yang telah dirancang sebelumnya, akan dilakukan praproses berupa perubahan data menjadi format *grayscale* dan *resize* data menjadi ukuran 256x256. Hasil konversi *grayscale* dan *resize* dapat dilihat pada Gambar 5.1.



Gambar 5.1 Gambar asli dan gambar setelah konversi ke *grayscale* dan *resize*

Setelah itu akan dilakukan proses *Discrete Wavelet Transform* (DWT) terhadap data. DWT akan menghasilkan 4 *subband* yang nantinya akan menjadi input dari arsitektur CNN. Hasil proses DWT dapat dilihat pada Gambar 5.2. Spesifikasi dataset setelah proses DWT data dapat dilihat pada Tabel 5.1.

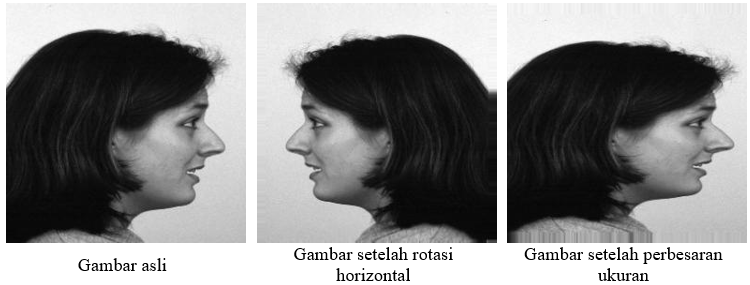


Gambar 5.2 Hasil proses DWT

Tabel 5.1 Spesifikasi dataset setelah proses DWT

|  |  |
| --- | --- |
| **Keterangan** | **Spesifikasi** |
| Ukuran resolusi | 128 x 128 |
| Ekstensi | .JPG |
| Jumlah gambar | 4900 |
| Jumlah orang | 35 Pria, 35 Wanita |
| Jumlah sesi foto | 2 kali |
| Jumlah kelas | 7 kelas |
| Jumlah gambar per kelas | 700 |
| Jumlah sudut pandang per kelas | 5 sudut pandang |
| Ukuran file | 2-3 KB |
| Kanal warna | 1 (*grayscale*) |

Setelah itu akan dilakukan proses augmentasi data dengan melakukan refleksi horizontal pada gambar dan perbesaran ukuran gambar. Augmentasi akan menghasilkan 2 gambar baru yang nantinya akan digabung dengan gambar asli untuk dijadikan data latih dari arsitektur CNN. Hasil augmentasi data dapat dilihat pada Gambar 5.3. Spesifikasi dataset setelah proses augmentasi data dapat dilihat pada Tabel 5.2.



Gambar 5.3 Gambar asli dan gambar hasil augmentasi

Tabel 5.2 Spesifikasi dataset setelah proses augmentasi data

|  |  |
| --- | --- |
| **Keterangan** | **Spesifikasi** |
| Ukuran resolusi | 128 x 128 |
| Ekstensi | .JPG |
| Jumlah gambar | 11700 |
| Jumlah orang | 35 Pria, 35 Wanita |
| Jumlah sesi foto | 2 kali |
| Jumlah kelas | 7 kelas |
| Jumlah gambar per kelas | 2100 |
| Jumlah sudut pandang per kelas | 5 sudut pandang |
| Ukuran file | 2-3 KB |
| Kanal warna | 1 (*grayscale*) |

## Skenario Uji Coba

Proses uji coba berguna untuk menemukan parameter yang menghasilkan performa model yang paling optimal. Parameter yang tepat akan memberikan hasil yang lebih baik pada saat proses uji coba. Hasil terbaik dari suatu skenario uji coba akan digunakan untuk skenario uji coba berikutnya. Ada 4 macam skenario uji coba dan semuanya akan dicoba pada arsitektur CNN yang telah dirancang. Skenario uji coba yang akan dilakukan yaitu:

1. Uji Coba Pembagian Data
2. Uji Coba Penggunaan *Wavelet*
3. Uji Coba Parameter CNN
4. Uji Coba Augmentasi Data

Tabel 5.3 berisi parameter-parameter awal arsitektur CNN yang digunakan dan dapat berubah di setiap uji coba yang dilakukan. Pada setiap skenario uji coba akan ditetapkan nilai parameter yang dapat meningkatkan kinerja arsitektur.

Tabel 5.3 Parameter awal yang digunakan dalam arsitektur

|  |  |
| --- | --- |
| **Keterangan** | **Parameter** |
| Jumlah *epoch* | 500 |
| Ukuran *batch* | 39 |
| *Optimizer* | Adam *Optimizer* |
| *Loss function* | *Categorical Cross Entropy* |
| *Learning rate* | 0,0001 |

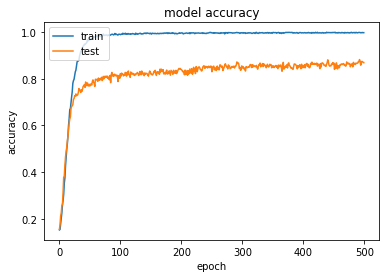
### Uji Coba Pembagian Data

Uji coba pertama adalah uji coba pembagian data. Uji coba ini dilakukan untuk menguji kemampuan model saat wajah seseorang belum pernah memasuki model sebelumnya. Akan ada 2 jenis pembagian data, yaitu:

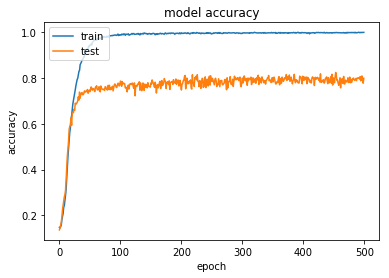
* 1. Pembagian data dengan membagi data berdasarkan individu yang menjadi model dataset. Dari 35 pria dan 35 wanita yang menjadi model dataset, akan diambil 29 individu dari setiap jenis kelamin secara acak untuk dijadikan data latih dan sisanya akan dijadikan data uji. Dengan demikian, individu data uji tidak pernah melatih model.
  2. Pembagian data dengan memasukkan semua dataset sesi A yang mana berarti semua individu pernah masuk ke dalam model lalu mengambil secara acak dataset sesi B untuk digunakan sebagai data uji. Dataset sesi B yang tidak digunakan sebagai data uji akan dijadikan data latih.

Pengambilan data secara acak pada kedua jenis pembagian data akan dilakukan sebanyak 5 kali. Rata-rata hasil akan dijadikan perbandingan untuk menentukan pembagian data yang paling baik. Hasil pembagian data yang paling baik akan dibawa ke skenario berikutnya untuk dijadikan parameter masukan.

Uji coba pembagian data masing-masing menghasilkan grafik akurasi pengujian yang dapat dilihat pada Gambar 5.4 dan Gambar 5.5. Perbandingan nilai akurasi, *precision*, *recall* pengujian serta lama waktu pelatihandapat dilihat pada Tabel 5.4.



Gambar 5.4 Grafik akurasi arsitektur CNN terbaik pada pembagian data berdasarkan sesi



Gambar 5.5 Grafik akurasi arsitektur CNN terbaik pada pembagian data berdasarkan individu

Tabel 5.4 Perbandingan rata-rata lama waktu pelatihan dan akurasi, *precision*, *recall* arsitektur CNN pada uji coba pembagian data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Pembagian Data** | **Lama waktu pelatihan** | **Akurasi** | **Kelas** | **Data uji** | ***Precision*** | ***Recall*** |
| Berdasarkan Sesi | 8901 detik | 84,68% | *Afraid* | 121 | 77% | 66% |
| *Angry* | 143 | 89% | 83% |
| *Disgust* | 155 | 85% | 83% |
| *Happy* | 147 | 94% | 97% |
| *Neutral* | 135 | 84% | 95% |
| *Sad* | 149 | 77% | 75% |
| *Surprised* | 150 | 84% | 92% |
| Berdasarkan Individu | 8917 detik | 74,74% | *Afraid* | 140 | 63% | 55% |
| *Angry* | 140 | 79% | 67% |
| *Disgust* | 140 | 82% | 75% |
| *Happy* | 140 | 91% | 92% |
| *Neutral* | 140 | 66% | 85% |
| *Sad* | 140 | 68% | 63% |
| *Surprised* | 140 | 77% | 84% |

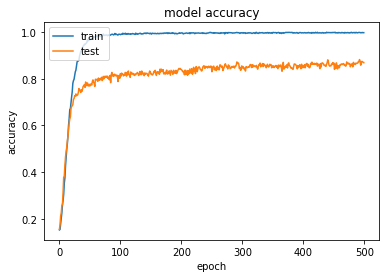
### Uji Coba Penggunaan Wavelet

Uji coba terhadap penggunaan proses *Discrete Wavelet Transform* (DWT) dilakukan untuk mengetahui apakah penggunaan DWT ini berhasil menambah performa model CNN yang dirancang. Uji coba akan dilakukan dengan 2 macam data, yaitu:

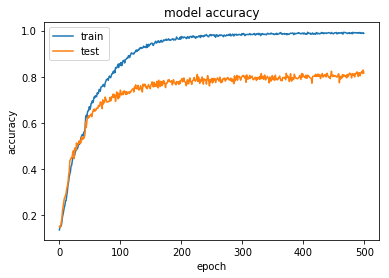
1. Data tanpa proses DWT
2. Data dengan proses DWT

Pada uji coba pembagian datadiperoleh hasil optimal pada pembagian berdasarkan sesi. Pembagian data tersebutkemudian digunakan untuk mengetahui hasil yang paling optimal untuk skenario uji coba penggunaan *wavelet*.

Uji coba penggunaan *wavelet* masing-masing menghasilkan grafik akurasi pengujian yang dapat dilihat pada Gambar 5.6 dan Gambar 5.7. Perbandingan nilai akurasi, *precision*, *recall* pengujian serta lama waktu pelatihandapat dilihat pada Tabel 5.5.



Gambar 5.6 Grafik akurasi pada arsitektur CNN dalam uji coba penggunaan data *wavelet*



Gambar 5.7 Grafik akurasi pada arsitektur CNN dalam uji coba penggunaan data tanpa *wavelet*

Tabel 5.5 Perbandingan lama waktu pelatihan dan akurasi, *precision*, *recall* arsitektur CNN pada uji coba penggunaan data *wavelet*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Wavelet** | **Lama waktu pelatihan** | **Akurasi** | **Kelas** | **Data uji** | ***Precision*** | ***Recall*** |
| Tanpa *Wavelet* | 2500 detik | 81,6% | *Afraid* | 121 | 77% | 60% |
| *Angry* | 143 | 87% | 76% |
| *Disgust* | 155 | 82% | 85% |
| *Happy* | 147 | 92% | 95% |
| *Neutral* | 135 | 77% | 88% |
| *Sad* | 149 | 71% | 76% |
| *Surprised* | 150 | 85% | 88% |
| Dengan *Wavelet* | 8806 detik | 86,8% | *Afraid* | 121 | 83% | 70% |
| *Angry* | 143 | 91% | 81% |
| *Disgust* | 155 | 88% | 85% |
| *Happy* | 147 | 98% | 99% |
| *Neutral* | 135 | 85% | 93% |
| *Sad* | 149 | 78% | 83% |
| *Surprised* | 150 | 85% | 95% |

### Uji Coba Parameter CNN

Uji coba penggantian parameter CNN digunakan untuk mengetahui parameter mana saja yang menghasilkan performa model terbaik. Parameter CNN yang akan dicoba untuk divariasikan antara lain:

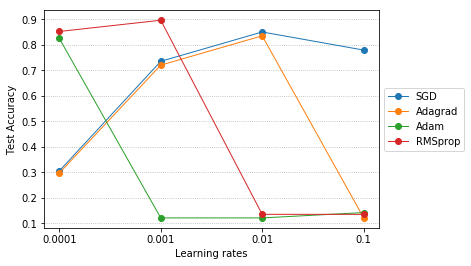
* 1. *Optimizer*
  2. *Learning rate*

Pada uji coba penggunaan *wavelet* diperoleh hasil optimal pada penggunaan *wavelet*. Data yang telah diproses menggunakan *Discrete Wavelet Transform* (DWT) tersebutkemudian digunakan untuk mengetahui hasil yang paling optimal untuk skenario uji coba penggantian parameter CNN.

*Optimizer* yang telah digunakan untuk uji coba sebelumnya adalah Adam *Optimizer* dengan *learning rate* 0,0001. Skenario uji coba penggantian parameter akan dilakukan dengan 4 penggantian *optimizer* antara lain dengan menggunakan SGD, Adagrad, Adam dan RMSprop.

*Learning rate* juga akan dicoba untuk divariasikan dengan harapan mendapatkan performa model yang lebih baik dari sebelumnya. Akan dicoba beberapa variasi *learning rate* yaitu 0.1, 0.01, 0.001 dan 0.0001.

Uji coba penggantian *optimizer* dan *learning rate* pada arsitektur CNN menghasilkan grafik akurasi pengujian yang dapat dilihat pada Gambar 5.8 dan perbandingan nilai akurasi, *precision*, *recall* pengujian serta lama waktu pelatihandapat dilihat pada Tabel 5.6.



Gambar 5.8 Perbedaan akurasi pada arsitektur CNN dalam uji coba penggantian *optimizer* dan *learning rate*

Tabel 5.6 Perbandingan lama waktu pelatihan dan akurasi terbaik, *precision*, *recall* arsitektur CNN pada uji coba penggantian parameter CNN

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Optimizer** | **Learning Rate** | **Lama waktu pelatihan** | **Akurasi** | **Kelas** | **Data uji** | ***Precision*** | ***Recall*** |
| SGD | 0,01 | 8685 detik | 85% | *Afraid* | 121 | 77% | 67% |
| *Angry* | 143 | 87% | 86% |
| *Disgust* | 155 | 89% | 80% |
| *Happy* | 147 | 95% | 97% |
| *Neutral* | 135 | 82% | 96% |
| *Sad* | 149 | 79% | 77% |
| *Surprised* | 150 | 85% | 92% |
| Adagrad | 0,01 | 8817 detik | 83,5% | *Afraid* | 121 | 71% | 70% |
| *Angry* | 143 | 93% | 73% |
| *Disgust* | 155 | 91% | 83% |
| *Happy* | 147 | 93% | 98% |
| *Neutral* | 135 | 77% | 95% |
| *Sad* | 149 | 73% | 74% |
| *Surprised* | 150 | 87% | 91% |
| Adam | 0,0001 | 8806 detik | 86,8% | *Afraid* | 121 | 83% | 70% |
| *Angry* | 143 | 91% | 81% |
| *Disgust* | 155 | 88% | 85% |
| *Happy* | 147 | 98% | 99% |
| *Neutral* | 135 | 85% | 93% |
| *Sad* | 149 | 78% | 83% |
| *Surprised* | 150 | 85% | 95% |
| RMSprop | 0,001 | 8846 detik | 89,6% | *Afraid* | 121 | 76% | 85% |
| *Angry* | 143 | 91% | 85% |
| *Disgust* | 155 | 89% | 88% |
| *Happy* | 147 | 97% | 99% |
| *Neutral* | 135 | 97% | 96% |
| *Sad* | 149 | 83% | 85% |
| *Surprised* | 150 | 94% | 89% |

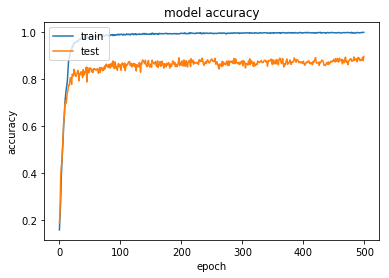
### Uji Coba Augmentasi Data

Uji coba terhadap penggunaan data augmentasi dilakukan untuk menambah akurasi model. Augmentasi data juga diharapkan dapat mengurangi *overfit* karena augmentasi data memvariasikan data dengan refleksi horizontal dan perbesaran ukuran gambar. Skenario uji coba penggunaan data augmentasi dilakukan pada 2 kelompok data yaitu:

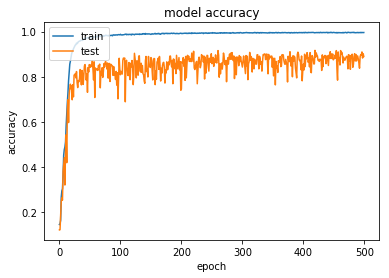
1. Data asli saja
2. Data asli & data augmentasi refleksi horizontal dan perbesaran ukuran gambar

Pada uji coba penggantian parameter CNN diperoleh hasil optimal pada penggunaan *optimizer* RMSpropdengan *learning rate* 0,001. Parameter yang telah dicoba tersebutkemudian digunakan untuk mengetahui hasil yang paling optimal untuk skenario uji coba augmentasi data.

Uji coba penggunaan augmentasi data masing-masing menghasilkan grafik akurasi pengujian yang dapat dilihat pada Gambar 5.9 dan Gambar 5.10. Perbandingan nilai akurasi, *precision*, *recall* pengujian serta lama waktu pelatihandapat dilihat pada Tabel 5.7.



Gambar 5.9 Grafik akurasi pada arsitektur CNN dalam uji coba penggunaan data asli



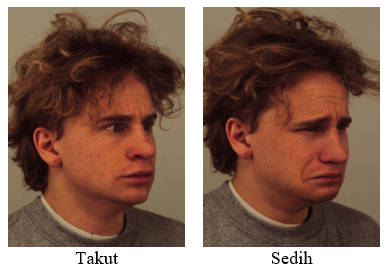
Gambar 5.10 Grafik akurasi pada arsitektur CNN dalam uji coba penggunaan data asli dan data augmentasi

Tabel 5.7 Perbandingan lama waktu pelatihan dan akurasi, *precision*, *recall* arsitektur CNN pada uji coba augmentasi data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data** | **Lama waktu pelatihan** | **Akurasi** | **Kelas** | **Data uji** | ***Precision*** | ***Recall*** |
| Data asli | 8846 detik | 89,6% | *Afraid* | 121 | 76% | 85% |
| *Angry* | 143 | 91% | 85% |
| *Disgust* | 155 | 89% | 88% |
| *Happy* | 147 | 97% | 99% |
| *Neutral* | 135 | 97% | 96% |
| *Sad* | 149 | 83% | 85% |
| *Surprised* | 150 | 94% | 89% |
| Data asli dan augmentasi | 25025 detik | 89% | *Afraid* | 121 | 75% | 83% |
| *Angry* | 143 | 98% | 87% |
| *Disgust* | 155 | 86% | 90% |
| *Happy* | 147 | 94% | 99% |
| *Neutral* | 135 | 93% | 93% |
| *Sad* | 149 | 83% | 87% |
| *Surprised* | 150 | 95% | 83% |

## Hasil dan Evaluasi

Pada uji coba pembagian data, dapat diketahui bahwa pembagian data berdasarkan sesi pengambilan gambar dapat menghasilkan akurasi yang lebih tinggi dibandingkan dengan pembagian data berdasarkan individu yang difoto yaitu sebesar 84,68%. Hal ini dapat terjadi karena pembagian data berdasarkan sesi membagi data secara lebih merata sehingga semua individu yang menjadi model dataset dan semua sudut pandang kelas ekspresi wajah pernah dimasukkan ke model. Kelas ekspresi wajah yang mendapatkan nilai *precision* dan *recall* tertinggi adalah kelas ekspresi senang. Hal ini dapat terjadi karena secara visual, ekspresi senang tidak mirip dengan ekspresi kelas yang lain sehingga dapat dibedakan dengan baik. Kelas ekspresi yang mengalami salah klasifikasi terbanyak adalah pada kelas ekspresi takut dan sedih, terlihat dari nilai *presicion* dan *recall* yang paling rendah diantara kelas ekspresi yang lain. Hal ini disebabkan karena secara visual, kedua kelas ini terlihat mirip dan lebih sulit untuk dibedakan, seperti yang dapat dilihat pada Gambar 5.11.



Gambar 5.11 Contoh tampilan ekspresi wajah yang mengalami salah klasifikasi terbanyak yaitu takut dan sedih

Pada uji coba penggunaan *wavelet*, diperoleh akurasi yang lebih baik pada data yang menggunakan *wavelet* dibanding dengan data asli tanpa *wavelet* yaitu 86,8%. Lama waktu pelatihan berbeda cukup signifikan yaitu 2500 detik untuk data tanpa *wavelet* dan 8806 detik untuk data dengan *wavelet*. Hal ini terjadi karena *wavelet* menggunakan 4 gambar yaitu gambar *subband* HH, HL, LH, dan LL sebagai input model yang dibangun sedangkan data asli hanya menggunakan 1 gambar sebagai input.

Pada uji coba penggantian parameter CNN, diperoleh hasil akurasi yang paling baik pada penggunaan RMSprop *optimizer* dengan *learning rate* sebesar 0,001 yaitu sebesar 89,6%. RMSProp *optimizer* tepat digunakan untuk arsitektur CNN yang telah dibangun karena menggunakan *adaptive learning rate* yang cenderung menghasilkan performa yang lebih baik dibandingkan dengan *optimizer* yang menggunakan *learning rate* statis seperti SGD.

Pada uji coba augmentasi data, akurasi yang hampir sama didapatkan untuk kedua jenis data yaitu 89,6% untuk data asli dan 89% untuk data asli dan augmentasi. Penambahan variasi data menggunakan augmentasi menghasilkan akurasi yang tidak jauh berbeda dibandingkan dengan data tanpa augmentasi walaupun lama waktu pelatihan bertambah secara signifikan yaitu sekitar 3 kali lipat. Hal ini terjadi karena data gambar asli telah mewakili semua kelas yang ada dengan baik sehingga tidak diperlukan data augmentasi lagi.

Dari keempat hasil uji coba yang telah dilakukan, pada Tabel 5.8 ditetapkan parameter optimal dari seluruh uji coba tersebut.

Tabel 5.8 Parameter optimal yang ditetapkan

|  |  |
| --- | --- |
| **Keterangan** | **Parameter optimal** |
| Pembagian data | Berdasarkan sesi |
| Penggunaan *wavelet* | Menggunakan DWT |
| *Optimizer* dan *Learning rate* | RMSprop dengan *learning rate* 0.001 |
| Augmentasi data | Data asli |

***(Halaman ini sengaja dikosongkan)***

# BAB VI KESIMPULAN DAN SARAN

Bab ini membahas tentang kesimpulan yang didasari oleh hasil uji coba yang telah dilakukan pada bab sebelumnya. Kesimpulan nantinya sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, juga terdapat saran yang ditujukan untuk pengembangan penelitian lebih lanjut di masa depan.

## Kesimpulan

Dalam pengerjaan Tugas Akhir ini setelah melalui tahap perancangan aplikasi, implementasi metode, serta uji coba, diperoleh kesimpulan sebagai berikut:

1. Pembagian data berdasarkan sesi menghasilkan akurasi yang lebih baik dibanding pembagian data berdasarkan individu. Akurasi model bertambah karena model pernah dilatih dengan data wajah semua individu yang menjadi model dataset. Maka dipilihlah pembagian data berdasarkan sesi yang menghasilkan rata-rata akurasi sebesar 84,68%.
2. Data yang menggunakan *Discrete Wavelet Transform* (DWT) pada tahap praproses menghasilkan akurasi yang lebih baik dibandingkan dengan data yang tidak menggunakan proses DWT yaitu sebesar 86,8%.
3. Augmentasi data berupa refleksi gambar secara horizontal dan perbesaran ukuran gambar kurang efektif untuk menambah akurasi model. Hal ini terbukti dengan akurasi model yang dilatih dengan data asli yaitu sebesar 89,6% yang tidak jauh berbeda dengan akurasi model yang dilatih dengan data yang telah dilakukan proses augmentasi yaitu sebesar 89%.
4. Sistem pengenalan ekspresi wajah manusia telah berhasil diimplementasikan dengan akurasi tertinggi 89,6% yang didapatkan dari uji coba menggunakan pembagian data berdasarkan sesi, praproses *Discrete Wavelet Transform* (DWT), RMSProp *optimizer* dengan *learning rate* 0,001 dan tanpa augmentasi data.

## Saran

Saran yang diberikan untuk pengembangan sistem pengenalan ekspresi wajah manusia menggunakan *Convolutional Neural Network* pada data gambar, yaitu:

* + - 1. Pengembangan sistem yang dapat mengenali lebih banyak ekspresi wajah manusia.
      2. Pengembangan sistem yang dapat mengenali banyak ekspresi wajah manusia dalam suatu gambar.
      3. Pengembangan sistem dengan arsitektur yang menghasilkan performa yang lebih baik.
      4. Pengembangan sistem yang dapat mengenali ekspresi wajah manusia secara *real-time.*
      5. Melakukan eksplorasi parameter selain *optimizer* dan *learning rate* yang dapat menambah performa arsitektur seperti *activation function*, jumlah dan ukuran *filter* konvolusi, ukuran *max pooling layer* dan level *Discrete Wavelet Transform* (DWT).

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | T. William dan R. Li, “An Ensemble of Convolutional Neural Networks Using Wavelets for Image Classification,” *Journal of Software Engineering and Applications,* no. 11, pp. 69-88, 2018. |
| [2] | K. Pooja, J. Kumari dan R. Rajesh, “Facial expression recognition: A survey,” *Procedia Computer Science,* vol. 58, pp. 486-491, 2015. |
| [3] | A. Karpathy, “Convolutional Neural Networks for Visual Recognition,” Stanford University, [Online]. Available: http://cs231n.github.io/. [Diakses 30 November 2018]. |
| [4] | S. Fadillah, “Penerapan Pengolahan Citra menggunakan Metode Deep Learning untuk Mendeteksi Kecacatan Permukaan Buah Manggis,” Yogyakarta, 2017. |
| [5] | D. Hubel dan T. Wiesel, “Receptive fields and functional architecture of monkey striate cortex,” *Journal of Physiology,* vol. 195, p. 215–243, 1968. |
| [6] | M. Zufar dan B. Setiyono, “Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time,” *Jurnal Sains dan Seni ITS,* vol. 5, pp. 2337-3520, 2016. |
| [7] | “Convolutional Neural Network,” MathWorks, [Online]. Available: https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html. [Diakses 29 November 2018]. |
| [8] | “Deep learning for complete beginners: convolutional neural networks with keras,” Cambridgespark, 20 March 2017. [Online]. Available: https://cambridgespark.com/content/tutorials/convolutional-neural-networks-with-keras/index.html. [Diakses 29 November 2018]. |
| [9] | “An Intuitive Explanation of Convolutional Neural Networks,” Ujjwalkarn, 11 August 2016. [Online]. Available: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets. [Diakses 29 November 2018]. |
| [10] | I. W. Suartika, A. Y. Wijaya dan R. Soelaiman, “Klasifikasi Citra Menggunakan Convolutional pada Caltech 101,” *JURNAL TEKNIK ITS,* vol. 5, 2016. |
| [11] | S. Sena, “Pengenalan Deep Learning Neural Network,” 28 October 207. [Online]. Available: https://medium.com/@samuelsena/pengenalan-deep-learning-8fbb7d8028ac. [Diakses 30 November 2018]. |
| [12] | G. Hinton, *Neural Networks for Machine Learning.* |
| [13] | S. Ruder, “Ruder.io,” 19 January 2016. [Online]. Available: http://ruder.io/optimizing-gradient-descent/index.html#rmsprop. [Diakses 23 December 2018]. |
| [14] | A. Budhiraja, “Dropout in (Deep) Machine Learning,” [Online]. Available: https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5. [Diakses 11 12 2018]. |
| [15] | “Introduction to Convolution Neural Networks,” 4 April 2016. [Online]. Available: https://www.analyticsvidhya.com/blog/2016/04/deep-learning-computer-vision-introduction-convolution-neural-networks/. [Diakses 29 November 2018]. |
| [16] | R. Munir, Pengolahan Citra Digital dengan Pendekatan Algoritmik, Bandung: Informatika ITB. |
| [17] | “About Python,” Python, [Online]. Available: https://www.python.org/about/. [Diakses 30 November 2018]. |
| [18] | “Keras: The Python Deep Learning library,” Keras, [Online]. Available: https://keras.io/. [Diakses 30 November 2018]. |
| [19] | “TensorFlow,” TensorFlow, [Online]. Available: https://www.tensorflow.org/. [Diakses 30 November 2018]. |
| [20] | “OpenCV,” [Online]. Available: https://opencv.org/. [Diakses 30 November 2018]. |
| [21] | “PyWavelets,” PyWavelets, [Online]. Available: https://pywavelets.readthedocs.io/en/latest/. [Diakses 30 November 2018]. |
| [22] | “NumPy,” NumPy, [Online]. Available: http://www.numpy.org/. [Diakses 30 November 2018]. |
| [23] | “Scikit-learn,” Scikit-learn, [Online]. Available: http://scikit-learn.org/stable/index.html. [Diakses 30 November 2018]. |
| [24] | “Matplotlib,” Matplotlib, [Online]. Available: https://matplotlib.org/index.html. [Diakses 30 November 2018]. |

***(Halaman ini sengaja dikosongkan)***

# LAMPIRAN

* 1. Hasil Uji Coba Individu F01

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Marah |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu F09

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Marah |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu F15

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Marah |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu F18

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu F24

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Marah |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu M21

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Marah |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu M23

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Marah |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu M24

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu M26

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Marah |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Jijik |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

* 1. Hasil Uji Coba Individu M30

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Takut | Takut |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 2 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Marah | Marah |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 3 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Jijik | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 4 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Senang | Senang |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 5 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Netral | Netral |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Sedih | Sedih |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |
| 7 | **Citra Asli** | **Hasil resize dan grayscale** | **Kelas Aktual** | **Kelas Prediksi** |
|  |  | Kaget | Kaget |
| **Hasil Wavelet** | | | |
| **HH** | **HL** | **LH** | **LL** |
|  |  |  |  |

***(Halaman ini sengaja dikosongkan)***

# BIODATA PENULIS

Hendry Wiranto, lahir di Ponorogo pada tanggal 24 Oktober 1996. Penulis menempuh pendidikan mulai dari TK Santa Melania Ponorogo (2002-2003), SD Katolik Santa Maria Ponorogo (2003-2009), SMP Katolik Slamet Riyadi Ponorogo (2009-2012), SMA Katolik St. Louis 1 Surabaya (2012-2015), dan sekarang sedang menjalani pendidikan S1 Informatika di ITS. Penulis aktif dalam organisasi dan kepanitiaan Himpunan Mahasiswa Teknik Computer (HMTC) dan Schematics. Diantaranya adalah menjadi staff Departemen Kewirausahaan HMTC ITS 2016-2017, staff ahli Departemen Kewirausahaan HMTC ITS 2017-2018, staff Departemen Website dan Kesekretariatan Schematics ITS 2016 dan staff ahli Departemen Website dan Kesekretariatan Schematics ITS 2017. Penulis juga merupakan salah satu penerima beasiswa dari Bank Indonesia pada tahun 2017. Komunikasi dengan penulis dapat melalui telepon: +6285334449875 dan *email*: **hendrywiranto24@gmail.com**.