KLASIFIKASI USIA PADA CITRA WAJAH BERBASIS *GRAY LEVEL CO-OCCURRENCE MATRIX* MENGGUNAKAN *ARTIFICIAL NEURAL NETWORK*

Oleh

RIZAL RINALDI MAPPE

T3115126

SKRIPSI

Untuk memenuhi salah satu syarat ujian
Guna memperoleh gelar Sarjana

PROGRAM SARJANA
TEKNIK INFORMATIKA
UNIVERSITAS ICHSAN GORONTALO
GORONTALO
2019

PERSETUJUAN SKRIPSI

KLASIFIKASI USIA PADA CITRA WAJAH BERBASIS *GRAY LEVEL CO-OCCURRENCE MATRIX* MENGGUNAKAN *ARTIFICIAL NEURAL NETWORK*

Oleh

RIZAL RINALDI MAPPE

T3115126

SKRIPSI

Untuk memenuhi salah satu syarat ujian
Guna memperoleh gelar Sarjana
Program Studi Teknik Informatika,
Ini telah disetujui oleh Tim Pembimbing

Gorontalo, 9 April 2019

Pembimbing I	**Pembimbing II**
SUDIRMAN S. PANNA, M.Kom	**HASTUTI DALAI, M.Kom**
NIDN. 0924038205	**NIDN. 0918038803**

PENGESAHAN SKRIPSI

KLASIFIKASI USIA PADA CITRA WAJAH BERBASIS *GRAY LEVEL CO-OCCURRENCE MATRIX* MENGGUNAKAN *ARTIFICIAL NEURAL NETWORK*

Oleh

RIZAL RINALDI MAPPE

T3115126

Diperiksa oleh Panitia Ujian Strata Satu (S1)

Universitas Ichsan Gorontalo

Gorontalo, 9 April 2019

1. Ketua Penguji
   Zohrahayaty, M.Kom ……..

2. Anggota
   Husdi, M.Kom ……..

3. Anggota
   Yulianty Lasena, M.Kom ……..

4. Anggota
   Sudirman S. Panna, M.Kom ……..

5. Anggota
   Hastuti Dalai, M.Kom ……..

PERNYATAAN SKRIPSI

Dengan ini saya menyatakan bahwa :

1. Karya tulis (Skripsi) saya ini adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik (Sarjana) baik di Universitas Ichsan Gorontalo maupun diperguruan tinggi lainnya.
2. Karya tulis (Skripsi) saya ini adalah murni gagasan, rumusan, dan penelitian saya sendiri, tanpa bantuan pihak lain, kecuali arahan dari Tim Pembimbing.
3. Dalam karya tulis (Skripsi) saya ini tidak terdapat karya atau pendapat yang telah dipublikasikan orang lain, kecuali secara tertulis dicantumkan sebagai acuan/sitasi dalam naskah dan dicantumkan pula daftar pustaka.
4. Pernyataan ini saya buat dengan sesungguhnya dan apabila dikemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya tulis ini, serta sanksi lainnya sesuai dengan norma-norma yang berlaku di Universitas Ichsan Gorontalo.

Gorontalo, 9 April 2019

Yang Membuat Pernyataan,

Rizal Rinaldi Mappe
NIM: T3115126

ABSTRACT

Age classification in this study was divided into three groups, namely children (5-11 years), adolescents (12-25 years) and adults (+26 years). This study aims to classify age based on face image using the extraction feature method, namely gray level co-occurrence matrix. This research method consists of: conversion of RGB data to grayscale, image normalization, face location detection, feature extraction and classification. In this study 90 training images were used, which are public data taken from a FG-NET dataset provider. The data consists of three groups according to the age group above and each group consists of 30 images for children, 30 images for teenagers, and 30 images for adults. The image data is processed into grayscale images which are then performed face detection. After obtaining the location of the face then crop it on the location of the face found. What follows is a feature calculation using the gray level co-occurrence matrix. The algorithm used for the classification process is an artificial neural network algorithm. The final results of the test show that the proposed method has been able to group age based on face images with the results of accuracy calculated using a confussion matrix of 73.3%.

Keywords: Age Classification, Gray Level Co-Occurrence Matrix, Artificial Neural Network

ABSTRAK

Klasifikasi usia dalam penelitian ini dibagi menjadi tiga kelompok yaitu anak-anak (5-11 tahun), remaja (12-25 tahun) dan dewasa (+26 tahun). Penelitian ini bertujuan untuk mengklasifikasi usia berdasarkan citra wajah dengan menggunakan metode fitur ekstraksi yaitu *gray level co-occurrence matrix*. Metode penelitian ini terdiri dari : konversi data *rgb* ke *grayscale*, normalisasi citra, deteksi lokasi wajah, ekstraksi fitur dan klasifikasi. Dalam penelitian ini digunakan data latih sebanyak 90 citra wajah yang merupakan data *public* yang diambil dari sebuah penyedia dataset *FG-NET*. Data tersebut terdiri dari tiga kelompok sesuai dengan kelompok umur di atas dan masing-masing kelompok terdiri dari 30 citra untuk anak-anak, 30 citra untuk remaja, dan 30 citra untuk dewasa. Data citra tersebut diolah menjadi citra grayscale yang kemudian dilakukan deteksi wajah. Setelah didapat lokasi wajah kemudian dilakukan *crop* pada bagian lokasi wajah yang ditemukan. Yang selanjutnya dilakukan perhitungan ciri menggunakan *gray level co-occurrence matrix*. Algoritma yang digunakan untuk proses klasifikasi adalah algoritma *artificial neural network*. Hasil akhir pengujian menunjukkan bahwa metode yang diusulkan telah mampu mengelompokan usia berdasarkan citra wajah dengan hasil akurasi yang digitung menggunakan *confussion matrix* sebesar 73,3 %.

Kata kunci : *Klasifikasi Usia, Gray Level Co-Occurrence Matrix, Artificial Neural Network*

KATA PENGANTAR

Alhamdulillah, penulis dapat menyelesaikan skripsi ini dengan judul **"Klasifikasi Usia Pada Citra Wajah Berbasis *Gray Level Co-Occurrence Matrix* Menggunakan *Artificial Neural Network*"**, sebagai salah satu Ujian Akhir guna memperoleh gelar Sarjana Komputer pada Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Ichsan Gorontalo.

Penulis menyadari sepenuhnya bahwa skripsi ini tidak mungkin terwujud tanpa bantuan dan dorongan dari berbagai pihak, baik bantuan moril maupun materil. Untuk itu, dengan segala keikhlasan dan kerendahan hati, penulis mengucapkan banya terimah kasih dan penghargaan yang setinggi-tingginya kepada:

1. Ibu Dr. Hj. Juriko Abdussamad, M.Si selaku Ketua Yayasan Pengembangan Ilmu Pengetahuan dan Teknologi (YPIPT) Ichsan Gorontalo;
2. Bapak Dr. Abdul Gaffar La Tjokke, M.Si selaku Rektor Universitas Ichsan Gorontalo;
3. Ibu Zohrahayaty, S.Kom, M.Kom selaku Dekan Fakultas Ilmu Komputer Universitas Ichsan Gorontalo;
4. Bapak Sudirman Melangi, M.Kom selaku Pembantu Dekan I Bidang Akademik Fakultas Ilmu Komputer Universitas Ichsan Gorontalo;
5. Ibu Irma Surya Kumala Idris, S.Kom, M.Kom selaku Pembantu Dekan II Bidang Administrasi Umum dan Keuangan Fakultas Ilmu Komputer Universitas Ichsan Gorontalo;
6. Bapak Andi Bode, M.Kom selaku Pembantu Dekan III Bidang Kemahasiswaan Fakultas Ilmu Komputer Universitas Ichsan Gorontalo;
7. Bapak Irvan Abraham Salihi, S.Kom, M.Kom selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Ichsan Gorontalo;
8. Bapak Sudirman S. Panna, M.Kom, selaku Pembimbing I;
9. Ibu Hastuti Dalai, M.Kom, selaku Pembimbing II;

10. Bapak dan Ibu Dosen Universitas Ichsan Gorontalo yang telah mendidik dan mengajarkan berbagai disiplin ilmu kepada penulis;
11. Kedua Orang Tua saya yang tercinta, atas segala kasih sayang, jerih payah dan doa restunya dalam membesarkan dan mendidik penulis;
12. Rekan-rekan seperjuangan yang telah banyak memberikan bantuan dan dukungan moril yang sangat besar kepada penulis;
13. Kepada semua pihak yang ikut membantu dalam penyelesaian proposal ini yang tak sempat penulis sebutkan satu-persatu.

Semoga Allah SWT melimpahkan balasan atas jasa-jasa kepada kami. Penulis menyadari sepenuhnya bahwa apa yang telah dicapai ini masih jauh dari kesempurnaan dan masih banyak terdapat kekurangan. Oleh karena itu, penulis sangat mengharapkan adanya kritik dan saran yang sangat konstruktif. Akhirnya penulis berharap semoga hasil yang telah dicapai ini dapat bermanfaat bagi kita semua, Amin.

Gorontalo, 2019

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi saat ini semakin canggih, banyak produk ditawarkan guna mendukung penggunaan teknologi informasi tersebut, tentunya hal tersebut menjadi suatu kemudahan bagi siapa saja yang ingin memanfaatkan teknologi untuk mendapatkan informasi secara efektif dan efisien. Namun dengan adanya kemudahan tersebut setiap informasi dapat mengalir dengan mudah kepada siapa saja, selain memiliki dampak positif juga terdapat dampak negatif yang dihasilkan terutama bagi kelompok usia anak-anak, misalnya anak-anak secara bebas dapat mengakses konten-konten khusus dewasa melalui media internet, televisi maupun perangkat *mobile*, tentunya aktivitas tersebut dapat memberikan dampak buruk terhadap anak terutama pembentukan moral anak yang tumbuh menjadi tidak baik, hal tersebut disebabkan oleh tidak tersedianya *filtering* dan otentikasi secara visual terhadap pengakses informasi yang diperuntukkan khusus kepada kelompok usia tertentu.

Di era yang serba komputer saat ini, banyak penelitian yang memanfaatkan komputer, diantaranya penelitian mengenai citra wajah manusia. Topik tersebut muncul saat diketahui bahwa citra wajah manuisa mengandung banyak informasi seperti gender, usia dan lain sebagainya[1]. Pemrosesan citra merupakan salah satu bidang *computer vision* yang digunakan untuk mengolah citra, khususnya citra wajah secara komputerisasi[2].

Pada penelitian sebelumnya yang dilakukan oleh Ranita dkk dengan judul "Deteksi kelompok usia manusia berdasarkan fitur wajah menggunakan filter gabor 2D. Berdasarkan hasil pegujian diperoleh akurasi tertinggi sistem adalah saat menggunakan 32 ciri dengan metode pengukuran kemiripan menggunakan *Euclidean Distance*, k=3, sebesar 79% dengan rata-rata akurasi sebesar 70.00% [3]. penelitian juga dilakukan oleh Muchammad Al Amin dan Dwi Juniati dengan judul "klasifikasi kelompok umur manusia berdasarkan analisis dimensi fraktal *box*

counting dari citra wajah dengan deteksi tepi Canny", penelitian tersebut mengelompokan usia menjadi empat kelompok umur yaitu kanak-kanak, remaja, dewasa dan lansia. Penelitian tersebut menghasilkan akurasi paling optimal yaitu 98,33% ketika nilai k=2. Sedangkan penelitian yang dilakukan oleh Nur Hayatin "Klasifikasi kelompok usia berdasarkan citra wajah menggunakan algoritma *neural network* dengan fitur *face anthropometry* dan kedalam kerutan" penelitian tersebut membagi data citran *input* menjadi 3 kelas yaitu anak, remaja dan tua. Hasil penelitan mampu menghasilkan akurasi sebesar 65%.

Seiring bertambahnya usia, wajah mengalami perubahan pada tekstur wajah. Metode *Gray-Level Co-Occurrence Matrix* (GLCM) merupakan salah satu metode untuk ektraksi tekstur pada citra . Ekstraksi tekstur dilakukan untuk mengambil informasi pokok dari suatu citra sebelum digunakan ke proses berikutnya. Metode *GLCM* menggunakan beberapa fitur pendekatan statistic seperi energi, entropi, kontras, dan sebagainya. *GLCM* dapat menentukan ciri dalam merepresentasikan karakteristik tekstur dari sebuah citra [4]. Menurut (Mirzapour,2013; Ella, 2008) pemilihan metode *GLCM* untuk ciri tekstur karena metode *GLCM* mampu mencapai akurasi tertinggi dibandingkan dengan metode Filter Gabor[5].

Pada penelitian ini algoritma yang digunakan untuk proses klasifikasi adalah *Artificial Neural Network* yang akan mengelompokan data citra wajah menjadi 3 kelas yaitu, kanak-kanak, remaja dan dewasa. Tujuan akhir dari penelitian ini akan dihasilkan sebuah informasi seberapa baik penggunaan algoritma *GLCM* sebagai tekstur analisis untuk mendeskripsikan fitur wajah dalam proses klasifikasi usia. Lebih lanjut hasil penelitian ini akan dijadikan sebagai bahan rujukan dalam merancang sebuah sistem otentikasi atau *filtering* dalam mengakses sebuah konten informasi untuk kategori kelompok usia tertentu yang disebut dengan sistem *automatic video filtering.*

Berdasarkan berbagai pemaparan diatas, maka peneliti tertarik untuk melakukan penelitian dengan judul: "Klasifikasi Usia pada Citra Wajah berbasis *Gray-Level Co-Occurrence Matrix* menggunakan *Artificial Neural Network*".

Diharapkan penelitian ini mampu memberikan hasil paling akurat untuk mendeskripsikan fitur wajah dalam proses klasifikasi usia.

1.2 Identifikasi Masalah

Berdasarkan latar belakang masalah di atas dapat diidentifikasikan masalah yang terkait dengan penelitian ini antara lain.

1. Anak-anak secara bebas dapat mengakses konten-konten dewasa melalui media internet, televisi maupun perangkat *mobile*.
2. Penggunaan informasi yang tidak sesuai dengan usia dapat mengakibatkan dampak yang buruk.

1.3 Rumusan Masalah

1. Bagaimana hasil penggunaan ekstraksi ciri citra wajah menggunakan metode *Gray Level Co-Occurrence Matrix* (GLCM) dalam proses klasifikasi usia?
2. Bagaimana kinerja klasifiksasi usia pada citra wajah menggunakan metode *Artificial Neural Network* ?

1.4 Maksud dan Tujuan Penelitian

1. Memperoleh informasi mengenai hasil penggunaan metode GLCM dalam mengekstraksi ciri wajah pada proses klasifikasi usia.
2. Mengetahui hasil kinerja penggunaan metode ANN dalam melakukan klasifikasi usia melalui citra wajah.

1.5 Manfaat Penelitian

1. Manfaat Teoritis

   Memberikan masukan bagi perkembangan ilmu pengetahuan dan teknologi, khususnya dalam bidang *computer vision*, yaitu berupa uji coba metode untuk klasifikasi usia pada citra wajah.
2. Manfaat Praktis

Sumbangan pemikiran, karya, bahan pertimbangan, atau solusi untuk mengurangi penyalahgunaan informasi berupa konten-konten yang diperuntukan untuk kelompok usia tertentu.

BAB II
LANDASAN TEORI

2.1 Tinjauan Studi

Beberapa penelitian berkaitan dengan klasifikasi usia pada citra wajah yang menjadi acuan dan sumber beberapa penelitian tersebut adalah:

1. Pada tahun 2017 penelitian yang dilakukan oleh Muchammad Al Amin dan Dwi Juniati jurusan matematika FMIPA Universitas Negeri Surabaya dengan judul "klasifikasi kelompok umur manusia berdasarkan analisis dimensi fraktal *box counting* dari citra wajah dengan deteksi tepi *canny*" Klasifikasi kelompok umur manusia dalam penelitian tersebut dibagi menjadi empat kelompok yaitu kanak-kanak, remaja, dewasa, dan lansia. Klasifikasi kelompok umur manusia didasarkan pada intensitas kerutan yang nampak pada citra wajah. Metode yang digunakan untuk menganalisis intensitas kerutan tersebut yaitu dimensi fraktal *box counting*. Untuk mendapatkan penampakan kerutan wajah yang lebih jelas pada citra wajah dapat digunakan deteksi tepi Canny. Dalam penelitian tersebut digunakan data 60 citra wajah individu yang diambil secara langsung dari warga desa Ngingas, Kecamatan Waru, Kabupaten Sidoarjo. Data tersebut terdiri dari empat kelompok sesuai dengan kelompok umur di depan dan masing-masing kelompok terdiri dari 15 citra. Data citra tersebut diolah menjadi citra grayscale yang kemudian dilakukan deteksi tepi Canny. Setelah didapat citra tepi wajah kemudian dilakukan penghitungan dimensi fraktal *box counting*. Nilai dimensi fraktal digunakan untuk klasifikasi. Data citra dibagi secara acak menggunakan metode *k-fold cross validation* (k=5) menjadi 5 partisi dan dilakukan 5 kali iterasi. Kemudian dilakukan klasifikasi dari tiap data citra menggunakan metode κNN (κ-*Nearest Neighbor*) dengan percobaan nilai κ=1, 2, 3, …, dan 12. Didapat nilai akurasi paling optimal yaitu 98,33% ketika nilai k=2.
2. Pada tahun 2016 penelitian yang dilakukan oleh Nur hayatin Teknik Informatika Universitas Muhammadiyah Malang dengan judul "Klasifikasi

usia berdasarkan citra wajah menggunakan algoritma *neural network* dengan fitur *face anthropometry* dan kedalam kerutan" Penelitian tersebut bertujuan untuk mengklasifikasi kelompok usia berdasarkan citra wajah dengan menggunakan fitur penting yaitu *face anthropometry* dan kerutan (*wrinkle*). Di mana fitur kerutan yang digunakan selain memperhitungkan lebar kerutan (*wrinkle density*) juga digunakan fitur kedalaman kerutan (*the dept of wrinkle*). Lokasi titik wajah diidentifikasi berdasarkan bentuk simetri wajah dan perbedaan nilai intensitas piksel. Sedangkan kerutan didapatkan dari gabungan metode deteksi tepi menggunakan operator Sobel dan histogram *equalization*. Algoritma yang digunakan untuk proses klasifikasi adalah algoritma *Neural Network* (NN) yang akan mengelompokan data citra *input* menjadi 3 kelas yaitu anak, remaja dan tua. Hasil akhir pengujian mampu mengelompokan usia berdasarkan citra wajah dengan cukup baik dengan hasil akurasi pengujian sebesar 65 % dengan *epochs* = 1000, dan *error rate* = 0.0095, sebanyak 100 kali iterasi.

3. Pada tahun 2016 penelitian yang dilakukan oleh Neneng dkk dengan judul Support Vector Machine Untuk Klasifikasi Citra Jenis Daging Berdasarkan Tekstur Menggunakan Ekstraksi Ciri Gray Level Co-Occurrence Matrices (GLCM), Penelitian ini menggunakan empat arah GLCM yakni 0°, 45°, 90°, dan 135°. Data yang digunakan dalam penelitian ini adalah citra daging kambing, daging kerbau, daging kuda, dan daging sapi dengan jarak pengambilan 20 cm, 30 cm, dan 40 cm. Penelitian ini menghasilkan tingkat pengenalan terbaik yakni 87,5% berada pada jarak pengambilan 20 cm dengan jarak piksel tetangga d=2 pada arah GLCM 135°.

2.2 Tinjauan Pustaka

2.2.1 Klasifikasi

Klasifikasi merupakan proses pengelompokan pixel pada suatu citra ke dalam sejumlah class (kelas), sehingga setiap kelas dapat menggambarkan suatu entitas dengan ciri-ciri tertentu. Klasifikasi citra merupakan suatu proses

pengelompokan seluruh pixel pada suatu citra kedalam dalam kelompok sehingga dapat diinterpretasikan sebagai suatu property yang spesifik.

Klasifikasi pertama kali diterapkan pada bidang tanaman yang mengklasifikasikan suatu spesies tertentu, seperti yang dilakukan oleh Carolus Von Linne (atau dikenal dengan nama Caralus Linnaeus) yang pertama kali mengklasifikan spesies berdasarkan karakteristik fisik. Selanjutnya dia dikenal sebagai bapak klasifikasi. Komponen-komponen utama dari proses klasifikasi antara lain:

1. Kelas, merupakan variabel tidak bebas yang merupakan label dari hasil klasifikasi, sebagai contoh adalah kelas loyalitas palanggan, kelas badai atau gempa bumi dan lain-lain.
2. Prediktor, merupakan variabel bebas suatu model berdasarkan dari karakteristik atribut data yang diklasifikasikan, misalnya merokok, minum-minuman beralkohol, tekanan darah, status perkawinan, dan sebagainya.
3. Set data pelatihan, merupakan sekumpulan data lengkap yang berisi kelas dan prediktor untuk dilatih agar model dapat mengelompokan kedalam kelas yang tepat. Contohnya adalah group pasien yang telah di-*test* serangan jantung, grup pelanggan di suatu supermarket, dan sebagainya.
4. Set data uji, berisi data-data baru yang akan dikelompokan oleh model guna mengetahui akurasi dari model yang telah dibuat.

Klasifikasi adalah proses untuk menemukan model yang menjelaskan atau membedakan konsep atau kelas data dengan tujuan untuk dapat memperkirakan kelas dari suatu objek yang kelasnya tidak diketahui[11]. Didalam klasifikasi diberikan sejumlah *record* yang dinamakan *training set,* yang terdiri dari beberapa atribut, atribut dapat berupa kontinyu ataupun kategori, salah satu atribut menunjukan atau record. Untuk mendapatkan model, kita harus kita harus melakukan analisis terhadap data latih (*training set*). Sedangkan data uji (*test set*) digunakan untuk mengetahui tingkat akurasi dari model yang telah dihasilkan, model itu sendiri bisa berupa aturan "jika-maka", ada beberapa teknik klasifikasi yang digunakan antara lain: *Decision tree, rule based, neural network, support vector machine, naïve bayes,* dan *nearest neighbor.*

Klasifikasi dapat didefinisikan secara detail sebagai suatu pekerjaan yang melakukan pelatihan/pembelajaran terhadap fungsi target f yang memetakan setiap vektor (set fitur) x kedalam satu dari sejumlah label kelas f yang tersedia. Pekerjaan pelatihan tersebut akan menghasilkan suatu model yang kemudian disimpan sebagai memori[11].

2.2.2 Usia

Usia adalah satuan waktu yang mengukur waktu keberadaan suatu benda atau makhluk, baik yang hidup maupun yang mati. Misalnya, umur manusia dikatakan lima belas tahun diukur sejak dia lahir hingga waktu umur itu dihitung. Dengan demikian, umur itu diukur dari tarikh ianya lahir sehingga tarikh semasa(masakini). Manakala usia pula diukur dari tarikh kejadian itu bermula sehinggalah tarikh semasa(masa kini).

Kategori penggolongan umur menurut Depkes RI 2009[2]:

1. Masa balita: 0-5 tahun.
2. Masa kanak-kanak 5-11 tahun.
3. Masa remaja awal 12-16 tahun.
4. Masa remaja akhir 17-25 tahun.
5. Masa dewasa awal 26-35' tahun.
6. Masa dewasa akhir 36-46 tahun.
7. Masa lansia awal 46-55 tahun.
8. Masa lansia akhir 56-65 tahun.
9. Masa manula 65-sampai atas.

2.2.3 Citra

Citra atau *image* adalah kombinasi antara titik, garis, bidang, dan warna untuk menciptakan suatu imitasi dari suatu objek, biasanya objek fisik atau manusia. Citra bisa berwujud gambar (*picture*) dua dimensi, seperti lukisan, foto, dan berwujud tiga dimensi, seperti patung. Terdapat tiga jenis citra yang umum

digunakan yakni citra berwarna, citra berskala keabuan, dan citra biner yang dijelaskan sebagai berikut:

1. Citra RGB (*color image*)

   Pada *color image* ini masing-masing piksel memiliki warna tertentu, warna tersebut adalah merah (*Red*), hijau (*Green*) dan biru (*Blue*). Jika masing-masing warna memiliki *range* 0 - 255, maka totalnya adalah 2553 = 16.581.375 (16 K) variasi warna berbeda pada gambar, dimana variasi warna ini cukup untuk gambar apapun. Karena jumlah bit yang diperlukan untuk setiap *pixel*, gambar tersebut juga disebut gambar-bit warna. *Color image* ini terdiri dari tiga matriks yang mewakili nilai-nilai merah, hijau dan biru untuk setiap pikselnya.

Gambar 2.1: Citra RGB (*color image*) [6]

2. Citra *Grayscale*

   Citra digital *black and white* (*grayscale*) setiap pikselnya mempunyai warna gradasi mulai dari putih sampai hitam. Rentang tersebut berarti bahwa setiap piksel dapat diwakili oleh 8 bit, atau 1 *byte*. Rentang warna pada *black and white* sangat cocok digunakan untuk pengolahan file gambar. Salah satu bentuk fungsinya digunakan dalam kedokteran (*X-ray*). *Black and white* sebenarnya merupakan hasil rata-rata dari *color image*.

Gambar 2.2: Citra berskala keabuan (*Grayscale*) [6]

3. Citra Biner

   Setiap piksel hanya terdiri dari warna hitam atau putih, karena hanya ada dua warna untuk setiap piksel, maka hanya perlu 1 bit per piksel (0 dan 1) atau apabila dalam 8 bit (0 dan 255), sehingga sangat efisien dalam hal penyimpanan. Gambar yang direpresentasikan dengan biner sangat cocok untuk teks (dicetak atau tulisan tangan), sidik jari (finger print), atau gambar arsitektur.

Gambar 2.3: Citra biner [6]

Citra diperoleh dari suatu pengambilan atau pengolahan data pada objek dengan menggunakan alat perekam, data yang didapatkan melalui kegiatan interpretasi citra secara manual dengan alat bantu kamera digital. Suatu citra memiliki karakteristik dan elemen-elemen dasar yaitu:

1. Karakteristik citra

   Ciri merupakan suatu tanda yang khas, yang membedakan antara satu dengan yang lain. Ciri-ciri dasar dari gambar :

a. Warna

Ciri warna suatu gambar dapat dinyatakan dalam bentuk histogram dari gambar tersebut yang dituliskan dengan : H(r,g,b), dimana H(r,g,b) adalah jumlah munculnya pasangan warna r (red), g (green) dan b (blue) tertentu.

b. Bentuk Ciri bentuk suatu gambar dapat ditentukan oleh tepi (sketsa), atau besaran moment dari suatu gambar. Pemakaian besaran moment pada ciri bentuk ini banyak digunakan orang dengan memanfaatkan nilai-nilai transformasi fourier dari gambar.

c. Teksture

Tekstur dari suatu gambar dapat ditentukan dengan menggunakan filter Gabor. Ciri tekstur ini sangat handal dalam menentukan informasi suatu gambar bila digabungkan dengan ciri warna gambar.

2. Elemen-elemen citra

Citra mengandung sejumlah elemen-elemen dasar sebagai berikut :

a. Kecerahan (brightness) : intensitas cahaya rata-rata dari suatu area yang melingkupinya.

b. Kontras (contrast) : sebaran terang (lightness) dan gelap (darkness) didalam sebuah citra. Citra dengan kontras rendah komposisi citranya sebagian besar gelap. Citra dengan kontras yang baik, komposisi gelap dan terangnya tersebar merata.

c. Kontur (contour) : keadaan yang ditimbulkan oleh perubahan intensitas pada piksel-piksel tetangga, sehingga kita dapat mendeteksi tepi objek didalam citra.

d. Warna (colour) : persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh objek. Warna-warna yang dapat ditangkap oleh mata manusia merupakan kombinasi cahaya dengan panjang berbeda. Kombinasi yang memberikan rentang warna paling lebar Red (R), green (G), blue (B)

e. Bentuk (shape) : property intrinsik dari objek tiga dimensi, dengan pengertian bahwa bentuk merupakan property intrinsik utama untuk visual

manusia. Umumnya citra yang dibentuk oleh manusia merupakan 2D, sedangkan objek yang dilihat adalah 3D.

f. Tekstur (texture) : distribusi spasial dari derajat keabuan didalam sekumpulan piksel-piksel yang bertetangga.

2.2.4 Pengolahan Citra

Pengolahan citra (*Image Processing*) merupakan proses memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau komputer. Teknik pengolahan citra dilakukan dengan mentrasformasikan citra menjadi citra lain, misalnya: pemanfaatan citra (*image compression*). Pengolahan citra merupakan proses awal (*preprocessing*) dari komputer visi. Pengelompokkan data numerik dan simbolik (termasuk citra) dilakukan secara otomatis oleh komputer agar suatu objek dalam citra dapat dikenali dan diinterpreasi. Pengenalan pola adalah tahapan selanjutnya atau analisis dari pengolahan citra. Citra ada 2 macam yaitu:

1. Citra Kontinyu. Dihasilkan dari sistem optik yang menerima sinyal analog. Contoh: mata manusia, kamera analog.
2. Citra Diskrit / Citra Digital dibentuk dari pixel-pixel yang tergabung dalam satu kesatuan yang membentuk sebuah citra yang hanya dapat dibuka dengan komputerisasi.

Dalam pengolahan citra, terdapat operasi-operasi yang secara umum dapat diklasifikasikan sebagai berikut:

1. Perbaikan kualitas citra (*image enhancement*), contohnya perbaikan kontras gelap/terang, penajaman (*sharpening*), dan perbaikan tepian objek (*edge enhancement*).
2. Restorasi citra (*image restoration*), contohnya penghilangan kesamaran (*deblurring*).
3. Pemampatan citra (*image compression*).
4. Segmentasi citra (*image segmentation*).
5. Pengorakan citra (*image analysis*), contohnya pendeteksian tepi objek (*edge enhancement*) dan ekstraksi batas (*boundary*).
6. Rekonstruksi citra (*image recronstruction*).

Adapun Manfaat Pengolahan Citra Dalam Kehidupan Sehari-hari antara lain:

1. Bidang kesehatan, digunakan untuk rontgen tubuh manusia yang berfungsi untuk mengetahui ada atau tidaknya kelainan di tubuh.
2. Bidang visual, bisa digunakan untuk pemotretan lewat satelit, GPS, foto kamera dan lain-lain.
3. Bidang hiburan, Gambar gambar kartun, yang dibuat bisa bergerak.

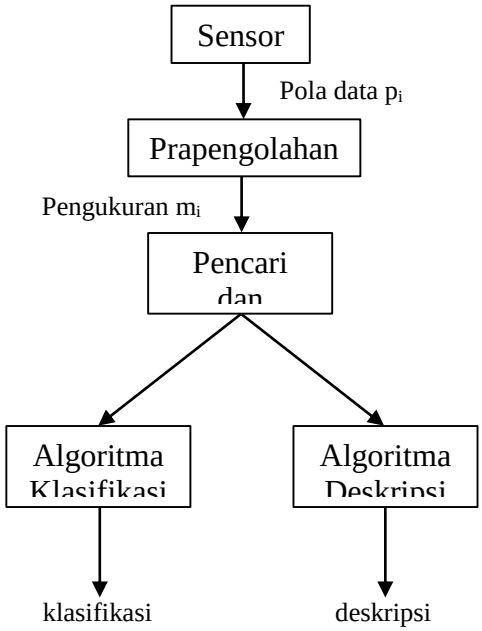
2.2.5 *Computer Vision*

Computer vision merupakan cabang dari teknik kecerdasan buatan (*artificial intelligence*) yang berhubungan dengan kegiatan simulasi visualisasi manusia. Secara fungsional, computer vision dan penglihatan manusia adalah sama, dengan tujuan menafsirkan data spasial yaitu data yang diindeks oleh lebih dari satu dimensi. Meskipun demikian, computer vision tidak dapat diharapkan untuk mereplikasi persis seperti mata manusia. Hal ini disebabkan pengetahuan tentang bagaimana sistem mata dan otak bekerja belum sepenuhnya dipahami, sehingga tidak dapat merancang sebuah sistem untuk mereplikasi secara tepat fungsi mata manusia. Yang dapat dilakukan adalah teknik computer vision yang dapat mereplikasi dan, dalam beberapa kasus bahkan memperbaiki sistem penglihatan manusia[12].

2.2.6 *Pettern Recognition* (pengenalan pola)

Pola adalah suatu entitas yang terdefinisi (mungkin secara samar) dan dapat diidentifikasi serta diberi nama. Pola bisa merupakan kumpulan hasil pengukuran atau pemantauan dan bisa dinyatakan dalam notasi vektor. Contoh : sidik jari, raut wajah, gelombang suara, tulisan tangan dan lain sebagainya. Dalam pengenalan pola data yang akan dikenali biasanya dalam bentuk citra atau gambar, akan tetapi ada pula yang berupa suara. Secara umum pengenalan pola (pattern recognition) adalah suatu ilmu untuk mengklasifikasikan atau menggambarkan sesuatu berdasarkan pengukuran kuantitatif fitur (ciri) atau sifat utama dari suatu obyek.

Pattern recognition (pengenalan pola) merupakan teknik yang bertujuan untuk mengklasifikasikan citra yang telah diolah sebelumnya berdasarkan kesamaan atau kemiripan ciri yang dimilikinya. Bagian terpenting dari teknik pengenalan pola adalah bagaimana memperoleh informasi atau ciri penting yang terdapat dalam sinyal. *Pattern recognition* merupakan bagian dalam *machine learning,* dapat pula *computer vision* yang menurut Richard O. Duda *et al*. dapat diartikan sebagai: "tindakan mengambil data mentah dan bertindak berdasarkan klasifikasi data. *Pattern recognition* melakukan pemetaan suatu data ke dalam konsep *class* atau *category* tertentu yang telah didefenisikan sebelumnya. Dengan demikian, *pattern recognition* termasuk dalam *supervised learning*. *Pattern recognition* bertujuan menentukan kategori pola berdasarkan ciri-ciri yang dimiliki oleh pola tersebut.



Gambar 2.4: Struktur sistem pengenalan pola [7]

Sensor untuk menangkap objek nyata yang selanjutnya diubah menjadi sinyal digital melalui proses digitalisasi. Prapengolahan befungsi mempersiapkan citra atau sinyal agar dapat menghasilkan ciri yang lebih baik pada tahap berikutnya. Pada tahap ini citra atau sinyal informasi ditonjolkan dan sinyal pengganggu (derau) dimimalisasi. Pencari dan seleksi ciri berfungsi menemukan

karakteristik pembeda yang mewakili sifat utama sinyal dan sekaligus mengurangi dimensi sinyal menjadi sekumpulan bilangan yang lebih sedikit tetapi representatif. Algoritma klasifikasi berfungsi untuk mengelompokkan ciri ke dalam kelas yang sesuai. Algoritma deskripsi berfungsi memberikan deskripsi pada sinyal.

Pattern recognition merupakan salah satu cabang dalam ilmu komputer yang cukup banyak diminati dan terus mengalami perkembangan yang pesat. Hal ini dapat ditunjukkan dari berbagai penelitian-penelitian yang telah dilaksanakan. Contoh Pattern recognition sangat luas, beberapa diantaranya seperti: identifikasi biometrik (face recognition, voice recognition, finger print recognition, dll), diagnosis penyakit pasien secara otomatis, complex letter recognition, line following robot navigation, wall following robot navigation, dll.

2.2.7 Ekstraksi Fitur (*feature extraction*)

Ekstraksi fitur adalah proses untuk mendapatkan ciri-ciri pembeda yang membedakan suatu sampel dari sampel lainya. Bagi sebagian besar aplikasi pengenalan pola, teknik ekstraksi fitur yang handal merupakan kunci utama dalam penyelesaian masalah pengenalan pola. Dengan adanya ekstraksi fitur, informasi penting yang berada dalam citra tersebut dapat diambil dan disimpan kedalam vektor fitur (*feature vector*). Fitur-fitur yang dapat disekstrak pada citra dapat berdasarkan elemen warna, bentu, tekstur, kecerahan, kontur dan kontras dengan menggunakan teknik ekstraksi fitur tertentu.

2.2.8 Metode *Gray-Level Co-Occurrence Matrix* (GLCM)

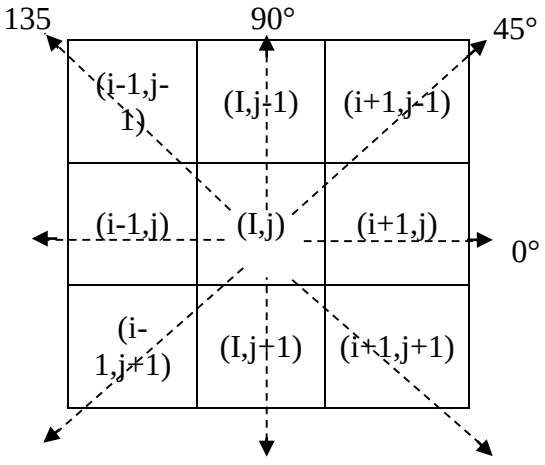
Metode GLCM merupakan salah satu metode untuk melakukan klasifikasi suatu citra. Metode GLCM termasuk dalam metode statistik dimana dalam perhitungan statistiknya menggunakan distribusi derajat keabuan (histogram) dengan mengukur tingkat kekontrasan, granularitas, dan kekasaran suatu daerah dari hubungan ketetanggaan antar piksel di dalam citra. Paradigma statistik ini penggunaannya tidak terbatas, sehingga sesuai untuk tekstur-tekstur alami yang tidak terstruktur dari sub pola dan himpunan aturan (mikrostruktur). Metode statistik terdiri dari ekstraksi ciri orde pertama dan ekstraksi ciri orde kedua.

Ekstraksi ciri orde pertama dilakukan melalui histogram citra sedangkan ekstraksi ciri statistik orde kedua dilakukan dengan matriks kookurensi, yaitu suatu matriks antara yang merepresentasikan hubungan ketetanggaan antar piksel dalam citra pada berbagai arah orientasi dan jarak spasial.

Matriks kookurensi merupakan matriks berukuran L x L (L menyatakan banyaknya tingkat keabuan) dengan elemen P(x1, x2) yang merupakan distribusi probabilitas bersama (join probability distribution) dari pasangan titik-titik dengan tingkat keabuan x1 yang berlokasi pada koordinat (j,k) dengan x2 yang berlokasi pada koordinat (m,n). Koordinat pasangan titik-titik tersebut berjarak r dengan sudut θ. Histogram tingkat kedua P(x1, x2) dihitung dengan pendekatan sebagai berikut :

$$P_{(x1,x2)} = \frac{banyaknya\ pasangan\ titik{-}titik\ dengan\ tingkat\ keabuan\ x1\ dan\ x2}{banyaknya\ titik\ pada\ daerah\ suatu\ citra}$$

GLCM adalah suatu matriks yang elemen-elemennya merupakan jumlah pasangan piksel yang memiliki tingkat kecerahan tertentu, di mana pasangan piksel itu terpisah dengan jarak d, dan dengan suatu sudut inklinasi θ. Dengan kata lain, matriks kookurensi adalah probabilitas munculnya gray level i dan j dari dua piksel yang terpisah pada jarak d dan sudut θ. Suatu piksel yang bertetangga yang memiliki jarak d diantara keduanya, dapat terletak di delapan arah yang berlainan.



Gambar 2.5: Hubungan ketetanggaan antar piksel sebagai fungsi orientasi dan jarak spasial [7]

Setelah diperoleh matriks kookurensi tersebut dilakukan pengukuran nilai tekstur didasarkan pada persamaan Harralick yang didefenisikan sebagai berikut :

1. Energy

   Mengukur tentang keseragaman atau sering disebut angular second moment, Energi akan bernilai tinggi ketika nilai piksel mirip dengan piksel yang lain, sebaliknya akan bernilai kecil menandakan nilai dari GLCM normalisasi adalah heterogen.

$$\text{Energy} = \sum_i \sum_j (Pn)^2(i,j)$$

2. Entropy

   Entropi dapat menunjukkan ketidakteraturan aras keabuan dalam ukuran bentuk citra, Nilai entropinya akan tinggi jika elemen – elemen pada GLCM mempunyai nilai yang relatif sama. Nilai rendah jika elemen-elemen GLCM dekat dengan nilai 0 atau 1.

$$\text{Entropy} = \sum_i \sum_j P_n(i,j) log\ P_n(i,j)$$

3. Contrast

   Contrast pada fitur GLCM menunjukkan ukuran penyebaran (momen inersia) elemen-elemen matriks citra. Jika letaknya jauh dari diagonal utama, nilai kekontrasan besar.

$$\text{Contrast} = \sum_i \sum_j P_n(i-j)^2$$

4. Correlation

   Korelasi menyatakan ukuran ketergantungan linear derajat keabuan citra sehingga dapat memberikan petunjuk adanya struktur linear dalam citra.

$$\text{Correlation} = \sum_i \sum_j \frac{(i\ x\ j)P_n(i,j) - (\mu_i\ x\ \mu_j)}{\sigma_i \sigma_j}$$

5. Dissimilarity

Mengukur ketidakmiripan pada suatu tekstur, akan bernilai besar apabila bentuk tekstur acak dan bernilai kecil jika bentuk tekstur seragam.

$$\text{Dissimilarity} = \sum_i \sum_j |i - j| P_n(i,j)$$

2.2.9 Metode *Artificial Neural Network* (ANN)

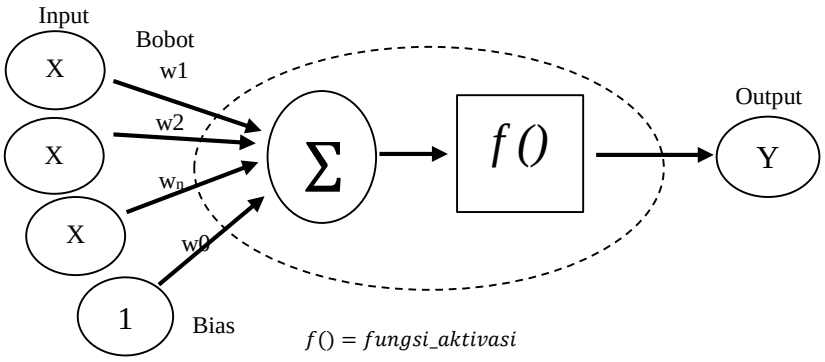
Artificial Neural Network atau jaringan saraf tirual (JST) merupakan salah satu representasi tiruan dari otak manuasia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut. Otak manusia berisi berjuta-juta sel syaraf (*neuron*) yang bertugas untuk memproses informasi. Masing-masing sel saling berinteraksi sehingga mendukung kemampuan kerja otak. Setiap sel syaraf akan memiliki satu inti sel yang bertugas untuk melakukan pemrosesan informasi.

Seperti halnya otak manusia, jaringan syaraf juga terdiri dari beberapa neuron, yang ada hubungan antara neuron-neuron tersebut. Neuron-neuron akan mentransformasikan informasi yang diterima melalui sambungan keluarnya menuju neuron-neuron yang lain. Dalam jaringan syaraf hubungan ini dikenal dengan nama bobot. Informasi tersebut disimpan pada suatu nilai tertentu pada bobot tersebut.

ANN *(Artificial Neural Network)* terdiri dari sejumlah satuan masukan (input) dan keluaran (output) yang terkoneksi, dan pada setiap koneksinya terdapat bobot *(weight)* tersendiri yang dapat diubah-ubah untuk mendapatkan hasil prediksi sesuai yang diinginkan. Lapisan-lapisan pada ANN adalah sebagai berikut:

a. *Input Layer* (Lapisan Masukan): merupakan lapisan yang menghubungkan sumber data ke jaringan pemrosesan. Dalam artian, setiap masukan akan merepresentasikan variabel-variabel bebas yang berpengaruh terhadap keluaran (output)
b. *Hidden Layer* (Lapisan Tersembunyi): merupakan lapisan perambat variabel-variabel input untuk mendapatkan hasil output yang lebih mendekati keinginan. Suatu ANN *Multi Layer* dapat memiliki satu atau lebih *hidden layer*.

c. *Ouput Layer* (Lapisan Keluaran): merupakan hasil keluaran dari pemrosesan data ANN. Keluaran yang didapatkan bergantung pada bobot, jumlah lapisan tersembunyi (*hidden layer*), dan fungsi aktivasi yang ditetapkan.

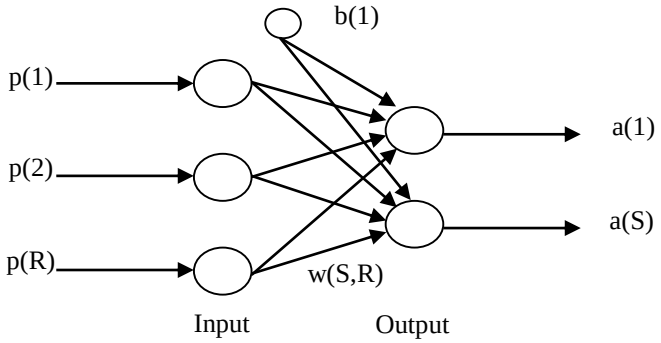


Gambar 2.6: Struktur neuron jaringan syaraf tiruan [8]

Pada umumnya metode *Artificial Neural Network* terdapat tiga jenis yang sering digunakan, yaitu: Single-Layer Neural Network dan Multilayer Neural Network.

1. Single-Layer Neural Network

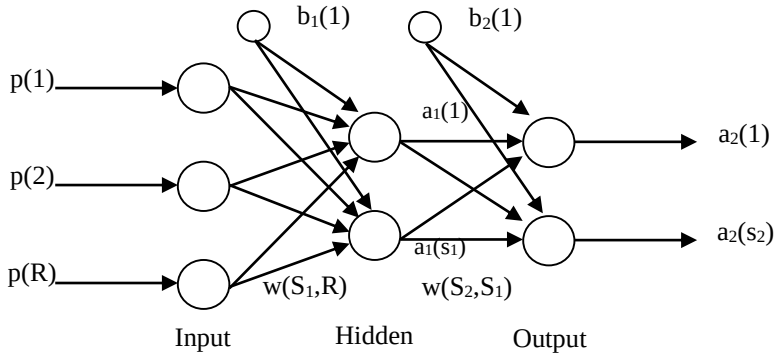
   Pada *Single-Layer Neural Network*, input layer terhubung langsung ke output layer. Kelemahan dari jenis ini adalah hanya bisa digunakan pada kasus sederhana.



Gambar 2.7: Single-Layer Neural Network [8]

2. Multilayer Neural Network

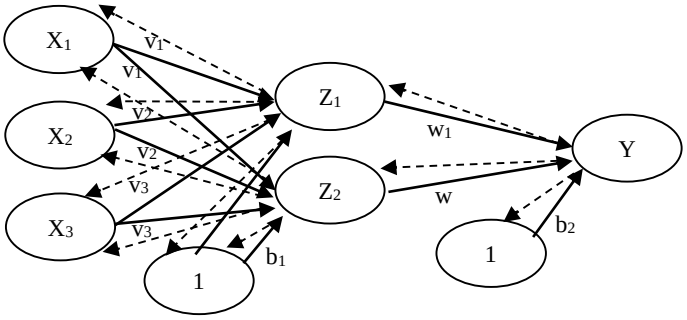
Pada *Multilayer Neural Network*, terdapat hidden layer yang terletak diantara input layer dan output layer.



Gambar 2.8: Multilayer Perceptron Neural Network [8]

3. Feed-forward Backpropagtion

Algoritma *Feed-forward Backpropagation* disebut sebagai propagasi balik karena ketika jaringan diberikan pola masukan sebagai pola pelatihan maka pola tersebut menuju ke unit-unit pada lapisan tersembunyi untuk diteruskan ke unit lapisan keluaran. Selanjutnya, unit-unit lapisan keluaran memberikan tanggapan yang disebut sebagai keluaran jaringan, saat keluaran jaringan tidak sama dengan keluaran yang diharapkan maka keluaran akan menyebar mundur (*backward*) pada lapisan tersembunyi diteruskan ke unit pada lapisan masukan.



Gambar 2.9: Feed-forward Backpropagation (Ardana PDH) [8]

2.2.10 Penerapan Metode *Artificial Neural Network* (ANN)

Salah satu metode pelatihan terawasi pada *neural network* adalah metode *backpropagation*, di mana ciri dari metode ini adalah meminimalkan *error* pada

output yang dihasilkan oleh jaringan. Pada gambar di bawah ini, *unit input* dilambangkan X, *hidden unit* dilambangkan Z, dan *unit output* dilambangkan Y. Bobot antara X dan Z dilambangkan dengan v sedangkan bobot antara Z dan Y dilambangkan dengan w[13].

Penerapan *backpropagation network* terdiri dari 2 tahap:

1. Tahap pelatihan, di mana pada tahap ini diberikan sejumlah data pelatihan dan target;
2. Tahap pengujian atau evaluasi, dilakukan setelah selesai tahap pelatihan.

Pada intinya, pelatihan dengan metode *backpropagation* terdiri dari tiga langkah, yaitu[13]:

1. Data dimasukkan ke *input* jaringan (*feedforward*);
2. Perhitungan dan propagasi balik dari *error* yang bersangkutan;
3. Pembaharuan (*adjustment*) bobot dan bias.

Saat umpan maju (*feedforward*), setiap unit *input* (Xi) akan menerima sinyal *input* dan akan menyebarkan sinyal tersebut pada tiap *hidden unit* (Zj), setiap *hidden unit* kemudian akan menghitung aktivasinya dan mengirim sinyal (zj) ke tiap unit *output*. Kemudian setiap unit *output* (Yk) juga akan menghitung aktivasinya (yk) untuk menghasilkan respons terhadap *input* yang diberikan jaringan. Saat proses pelatihan (*training*), setiap unit *output* membandingkan aktivasinya (yk) dengan nilai target (tk) untuk menentukan besarnya *error*. Berdasarkan *error* ini dihitung faktor k, di mana faktor ini digunakan untuk mendistribusikan *error* dari *output* ke *layer* sebelumnya. Dengan cara yang sama, faktor j juga dihitung pada *hidden* unit Zj, di mana faktor ini digunakan untuk memperbaharui bobot antara *hidden layer* dan *input layer*. Setelah semua faktor ditentukan, bobot untuk semua *layer* diperbaharui. Notasi yang digunakan dalam algoritma pelatihan[13]:

x = Data *training input* x = (x1,…,xi,…,xn)
t = Data *training* untuk *target output* t = (t1,…,tk,…,tm)
α = *Learning rate* yaitu parameter untuk mengontrol perubahan bobot selama pelatihan.
Xi = Unit *input* ke-i
Zj = *Hidden* unit ke-j

Yk = Unit *output* ke-k

v0j = Bias untuk *hidden* unit ke-j

vij = Bobot antara unit *input* ke-i dengan *hidden* unit ke-j

w0k = Bias untuk unit *output* ke-k

Wjk = Bobot antara *hidden* unit ke-j dengan unit *output* ke-k

δk = Faktor koreksi *error* untuk bobot wjk

δj = Faktor koreksi *error* untuk bobot vij

m = Momentum

Tahap-tahap pelatihan[13]:

1. Tahap 0: Inisialisasi bobot dan bias. Baik bobot maupun bias dapat diset dengan sembarang angka (acak) dan biasanya angka di sekitar 0 dan 1 atau -1 (bias positif atau negatif).
2. Tahap 1: Jika *stopping condition* masih belum terpenuhi, jalankan tahap 2-9
3. Tahap 2: Untuk setiap data *training*, lakukan tahap 3-8
4. Tahap 3: Setiap unit *input* (Xi,i=1,…,n) menerima sinyal *input* xi dan menyebarkan sinyal tersebut pada seluruh unit pada *hidden layer*. Perlu diketahui bahwa *input* xi yang dipakai di sini adalah *input* training data yang sudah diskalakan.
5. Tahap 4: Setiap *hidden unit* (Zj,j=1,…,p) akan menjumlahkan sinyal-sinyal *input* yang sudah berbobot, termasuk biasnya

   $$z_{in_j} = V_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

   dan memakai fungsi aktivasi yang telah ditentukan untuk menghitung sinyal *output* dari *hidden unit* yang bersangkutan,

   $$z_j = f\left(z_{in_j}\right)$$

   lalu mengirim sinyal *output* ini ke seluruh *unit* pada *unit output.*
6. Tahap 5: Setiap *unit output* (Yk,k=1,…,m) akan menjumlahkan sinyal-sinyal *input* yang sudah berbobot termasuk biasnya,

$$y_{in_k} = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

dan memakai fungsi aktivasi yang telah ditentukan untuk menghitung sinyal *output* dari *unit output* yang bersangkutan

$$z_j = f\left(z_{in_j}\right)$$

7. Tahap 6: Propagasi balik *error* (*backpropagation of error*). Setiap *unit output* (Yk,k=1,…,m) menerima suatu *target* (*output* yang diharapkan) yang akan dibandingkan dengan *output* yang dihasilkan.

$$\delta_k = (t_k - y_k)\, f'(y_{in_k})$$

Faktor k ini digunakan untuk menghitung koreksi *error* (wjk) yang nantinya akan dipakai untuk memperbaharui wjk, di mana:

$$\Delta w_{jk} = \alpha \delta_k z_j$$

Selain itu juga dihitung koreksi bias w0k yang nantinya akan dipakai untuk memperbaharui w0k, di mana:

$$\Delta w_{0k} = \alpha \delta_k$$

Faktor δ_k ini kemudian dikirimkan ke *layer* di depannya.

8. Tahap 7: Setiap *hidden unit* (Zj,j=1,…,p) menjumlah *input delta* (yang dikirim dari *layer* pada tahap 6) yang sudah berbobot.

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk}$$

Kemudian hasilnya dikalikan dengan turunan dari fungsi aktivasi yang digunakan jaringan untuk menghasilkan faktor koreksi *error* j, di mana:

$$\delta_j = \delta_{in_j} f'\left(z_{in_j}\right)$$

Faktor j ini digunakan untuk menghitung koreksi *error* (vij) yang nantinya akan dipakai untuk memperbaharui vij, di mana:

$$\Delta V_{ij} = \alpha \delta_j x_i$$

Selain itu juga dihitung koreksi bias v0j yang nantinya akan dipakai untuk memperbaharui v0j, di mana:

$$\Delta V_{0j} = \alpha \delta_j$$

9. Tahap 8: Pembaharuan bobot dan bias: Setiap *unit output* (Yk,k=1,…,m) akan memperbaharui bias dan bobotnya dengan setiap *hidden unit*.

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk}$$

10. Demikian pula untuk setiap *hidden unit* akan memperbaharui bias dan bobotnya dengan setiap *unit input.*

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij}$$

11. Tahap 9: Memeriksa *stopping condition Jika stop condition* telah terpenuhi, maka pelatihan jaringan dapat dihentikan. Untuk menentukan stopping condition terdapat dua cara yang biasa dipakai, yaitu: (1) Membatasi iterasi yang ingin dilakukan. Misalnya jaringan akan dilatih sampai iterasi yang ke-500. Yang dimaksud dengan satu iterasi adalah perulangan tahap 3 sampai tahap 8 untuk semua *training data* yang ada; (2) Membatasi *error*.

    Misalnya menentukan besar antara output yang dikehendaki dan output yang dihasilkan oleh jaringan. Jika terdapat sebanyak m *training* data, maka untuk menghitung *Mean Square Error* digunakan persamaan berikut:

$$MSE = 0.5\,\{(tk1 - yk1)2 + (tk2 - yk2)2 + \cdots + (tkm - ykm)2\}$$

Setelah pelatihan selesai, *backpropagation network* (BPN) dianggap telah pintar sehingga apabila jaringan diberi input tertentu, jaringan akan menghasilkan output seperti yang diharapkan. Cara mendapatkan output tersebut adalah dengan mengimplementasikan metode *backpropagation* yang sama seperti proses pelatihan, tetapi hanya pada bagian umpan majunya saja. Notasi yang digunakan dalam algoritma pengujian:

Xi = Unit input ke-i

Zj = Hidden unit ke-j

Yk = Unit output ke-k

v0j = Bias untuk hidden unit ke-j

vij = Bobot antara unit input ke-i dengan hidden unit ke-j

w0k = Bias untuk unit output ke-k

Wjk = Bobot antara hidden unit ke-j dengan unit output ke-k

Tahap-tahap pengujian pada model ANN, sbb:

1. Tahap 0: Inisialisasi bobot sesuai dengan bobot yang telah dihasilkan pada proses pelatihan di atas;
2. Tahap 1: Untuk setiap *input*, lakukan tahap 2-4;
3. Tahap 2: Untuk setiap *input* i=1,…,n skalakan bilangan dalam *range* fungsi aktivasi seperti yang dilakukan pada proses pelatihan di atas
4. Tahap 3: untuk j=1,…,p:

$$z_{in_j} = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

$$z_j = f\left(z_{in_j}\right)$$

5. Tahap 4: Untuk k=1,…,m:

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

$$y_k = f(y\_in_k)$$

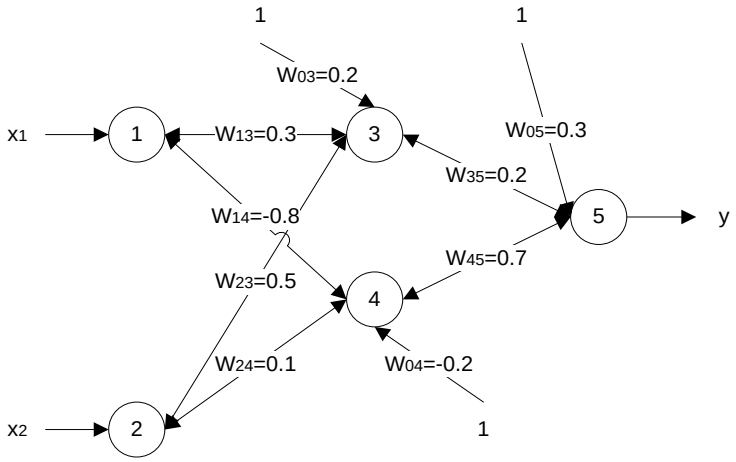
Variabel y_k adalah *output* yang masih dalam skala menurut *range* fungsi aktivasi. Untuk mendapatkan nilai *output* yang sesungguhnya, y_k harus dikembalikan seperti semula.

Contoh:

jaringan terdiri dari 2 unit input (X1 dan X7) dan 1 output yang diambil dari dataset untuk data ke 2, 16, 41, dan 62.

Tabel 2.1: Contoh manual ANN [13]

No.	X1	X7	y
2.	2	3	0
16.	2	6	1
41.	1	4	1
62.	1	2	0



Gambar 2.10: Kondisi bobot saat inisialisasi [13]

Kondisi bobot saat inisialisasi seperti ditunjukkan pada gambar di atas dengan jumlah neuron pada layer tersembunyi = 2.

Laju pembelajaran ($\boldsymbol{\eta}$) = 0.1;

Fungsi aktivasi yang digunakan adalah Sigmoid Biner;

Target *error* = 0.0001 dengan kriteria SSE;

Momentum yang digunakan = 0.95;

Maksimum jumlah iterasi pelatihan = 1,000 kali.

Maka:

1. Nilai pada neuron di hidden layer 3 dan 4:

$$\boldsymbol{v_j(p) = \sum_{i=1}^{n} x_i(p) w_{ij}(p)}$$

$$\boldsymbol{y_j(p) = \frac{1}{1 + e^{-v_j(p)}}}$$

(P) adalah iterasi ke-i.

$v_3(1) = x1w_{13} + x2w_{23} + 1w_{03} = 2 * 0.3 + 3 * 0.5 + 1 * 0.2 = 2.3$

$y_3(1) = \frac{1}{1+e^{-2.3}} = \frac{1}{1+2.71828^{-2.3}} = 0.908877$

$v_4(1) = x1w_{14} + x2w_{24} + 1w_{04} = 2 * (-0.8) + 3 * 0.1 + 1 * (-0.2) = 1.7$

$y_4(1) = \frac{1}{1+e^{-1.7}} = 0.845535$

2. Nilai pada neuron di output layer:

$$\boldsymbol{v_k(p) = \sum_{j=1}^{m} x_j(p) w_{jk}(p)}$$

$$\boldsymbol{y_k(p) = \frac{1}{1 + e^{-v_k(p)}}}$$

$\mathrm{v}_5(1) = \mathrm{Y3}(1)\mathrm{w}_{35} + \mathrm{Y4}(1)\mathrm{w}_{45} + 1\mathrm{w}_{05} = 0.908877 * 0.2 + 0.845535 * 0.7 + 1 * 0.3 = 2.054411496$

$\mathrm{y}_5(1) = \frac{1}{1+\mathrm{e}^{-2.054411496}} = 0.886392478$

3. Hitung gradient error untuk neuron pada output layer:

$$\boldsymbol{e_k(p) = y_{dk}(p) - y_k(p)}$$

$$\boldsymbol{\delta_k(p) = y_k(p) * [1 - y_k(p)] * e_k(P)}$$

Untuk data pertama, target/class nilai yang diharapkan adalah $\boldsymbol{y_d = 0}$, sedangkan keluaran yang didaptkan $\boldsymbol{y_5(1) = 0.886392478}$. Maka hitung *error* pada iterasi pertama untuk data pertama pada neuron 5 di output layer:

$e_5^1(1) = y_d - y_5(1) = 0 - 0.886392478 = \boldsymbol{-0.886392478}$

$\delta_5(1) = y_5(1) * \left(1 - y_5(1)\right) * e_5^1(1) = 0.886392478 * (1 - 0.886392478) * (-0.886392478) = -0.089260478$

4. Hitung koreki bobot untuk output layer, $\Delta w_{35}, \Delta w_{45}, dan\ \Delta w_{05}$:

$$\boldsymbol{\Delta w_{jk}(p) = \eta . y_j(p) * \delta_k(p)}$$

$\Delta w_{35} = \eta * y_3(1) * \delta_5(1) = 0.1 * 0.91 * (-0.089) = -0.008112679$

$\Delta w_{45} = \eta * y_4(1) * \delta_5(1) = 0.1 * 0.85 * (-0.089) = -0.007547282$

$\Delta w_{05} = \eta * 1 * \delta_5(1) = 0.1 * 1 * (-0.089) = -0.0089266048$

5. Hitung *gradient error* pada hidden layer $\delta_3(1)\ dan\ \delta_4(1)$:

$$\boldsymbol{\delta_j(p) = y_j(p) * [\left(1 - y_j(p)\right] + \sum_{k=1}^{1} \delta_k(p) * w_{jk}(p)}$$

$\delta_3(1) = y_3(1) * \left(1 - y_3(1)\right) * \sum_{k=1}^{1} \delta_k(1).w_{3k}(1) = y_3(1) * \left(1 - y_3(1)\right) * \delta_5(1).w_{35}(1) = 0.91 * (1 - 0.91) * (-0.089) * 0.2 = -0.001478505$

$\delta_4(1) = y_4(1) * \left(1 - y_4(1)\right) * \sum_{k=1}^{1} \delta_k(1).w_{4k}(1) = y_4(1) * \left(1 - y_4(1)\right) * \delta_5(1).w_{45}(1) == 0.85 * (1 - 0.85) * (-0.089) * 0.7 = -0.008160558$

6. Hitung koreksi bobot untuk hidden layer

$$\Delta w_{13}, \Delta w_{23}, \Delta w_{03}, \Delta w_{14}, \Delta w_{24}, dan\ \Delta w_{04}$$

$$\Delta \boldsymbol{w_{ij}(p)} = \boldsymbol{\eta} * \boldsymbol{x_i(p)} * \boldsymbol{\delta_j(p)}$$

$\Delta w_{13} = \eta * x1 * \delta_3(1) = 0.1 * 2 * (-0.0015) = -0.0003$

$\Delta w_{23} = \eta * x2 * \delta_3(1) = 0.1 * 3 * (-0.0015) = -0.00044$

$\Delta w_{03} = \eta * 1 * \delta_3(1) = 0.1 * 1 * (-0.0015) = -0.00015$

$\Delta w_{14} = \eta * x1 * \delta_4(1) = 0.1 * 2 * (-0.0082) = -0.00163$

$\Delta w_{24} = \eta * x2 * \delta_4(1) = 0.1 * 3 * (-0.0082) = -0.00245$

$\Delta w_{04} = \eta * 1 * \delta_4(1) = 0.1 * 1 * (-0.0082) = -0.00082$

7. Perbaharui bobot untuk neuron pada hidden layer w_{35}, w_{45}, w_{05}:

$$\boldsymbol{w_{ij}(p+1)} = \boldsymbol{w_{ij}(p)} + \Delta \boldsymbol{w_{ij}(p)}$$

$w_{35}(2) = w_{35}(1) + \Delta w_{35} = 0.2 + (-0.008112679) = 0.191887321$

$w_{45}(2) = w_{45}(1) + \Delta w_{45} = 0.7 + (-0.007547282) = 0.692452718$

$w_{05}(2) = w_{05}(1) + \Delta w_{05} = 0.3 + (-0.0089266048) = 0.291073952$

8. Perbaharui bobot pada layer tersembunyi, $w_{13}, w_{23}, w_{03}, w_{14}, w_{24}, w_{04}$:

$w_{13}(2) = w_{13}(1) + \Delta w_{13} = 0.3 + (-0.0003) = 0.299704$

$w_{23}(2) = w_{23}(1) + \Delta w_{23} = 0.5 + (-0.00044) = 0.499556$

$w_{03}(2) = w_{03}(1) + \Delta w_{03} = 0.2 + (-0.00015) = 0.199852$

$w_{14}(2) = w_{14}(1) + \Delta w_{14} = 0.8 + (-0.00163) = -0.798368$

$w_{24}(2) = w_{24}(1) + \Delta w_{24} = 0.1 + (-0.00245) = 0.097552$

$w_{04}(2) = w_{04}(1) + \Delta w_{04} = (-0.2) + (-0.00082) = -0.20082$

9. Naikkan 1 langkah iterasi (***p***), kembali ke langkah 2 dan ulangi proses tersebut sampai ***kriteria error*** tercapai.

10. Dengan demikian, didapatkan bobot akhir pada iterasi pertama untuk data pertama[2, 3]:

$w_{13} = 0.299704;$

$w_{23} = 0.499556\ ;$

$w_{03} = 0.199852;$

$w_{14} = -0.798368;$

$w_{24} = 0.097552;$

$w_{04} = -0.20082;$

$w_{35} = 0.191887321;$

$w_{45} = 0.692452718;$

$w_{05} = 0.291073952$

Error yang didapatkan dadalah $= -\mathbf{0.886392478}$. Selanjutnya proses di atas dulangi untuk data ke-dua [2, 6], data ke-tiga [1, 4], dan data ke-empat [1, 2] sehingga iterasi pertama selesai dilakukan. Pada setiap data yang diproses dalam satu iterasi, error keluaran disimpan untuk dihitung sebagai kriteria error, seperti SSE, MSE, dsb. Apanila hasil MSE < error yang didapatkan maka iterasi berhenti, sebaliknya dilakukan perambatan terus hingga batas perulangan/epoch.

2.2.11 *Confusion Matrix*

Confusion matrix merupakan suatu metode yang biasanya digunakan untuk melakukan perhitungan akurasi. Confusion matrix memberikan keputusan yang diperoleh penilaian *performance* klasifikasi berdasarkan objek dengan benar atau salah (*gurunescu*). *Confusion matrix* berisi informasi *actual* dan prediksi pada sistem klasifikasi[14].

Tabel 2.2: Tabel *Confusion Matrix*[14]

	Actual	*Actual*
	Positive	Negative
Predicted positive	*TP*	*FP*
Predicted negative	*FN*	*TN*

Recall	=	TP/(TP+FN)
Precision	=	TP/(TP+FP)
True positive rate	=	TP/(TP+FN)
False positive rate	=	FP/(FP+TN)

Keterangan:

- True positive (TP) = proporsi positif dalam dataset yang diklasifikasi positif.
- True negative (TN) = proporsi negativ dalam dataset yang diklasifikasi negativ.

- False positive (FP) = proporsi negativ dalam dataset yang diklasifikasi positif.
- False negative (FP) = proporsi negativ dalam dataset yang diklasifikasi.

2.2.12 *Unified Modeling Language* (UML)

Unified Modeling Language (UML) merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung[15]. *Unified Modeling Language* sering juga digunakan untuk menggambarkan, menentukan, desain, dan dokumen yang sudah ada atau proses bisnis baru, struktur dan perilaku artifak dari sistem perangkat lunak.

Unified Modeling Language (UML) adalah salah satu standar bahasa untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. UML digunakan untuk mendefinisikan sebuah sistem perangkat lunak secara detail. Artifak dalam sistem, untuk mendokumentasikan dan membangun bahasa yang dicetak biru dan ditulis dalam *Unified Modeling Language*, UML dapat digunakan dalam berbagai cara untuk mendukung metodologi pengembangan perangkat lunak.

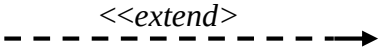
UML menyediakan beberapa notasi dan artifak standar yang bisa digunakan sebagai alat komunikasi bagi para pelaku dalam proses analisis dan desain. Artifak di dalam UML didefinisikan sebagai informasi dalam berbagai bentuk yang digunakan atau dihasilkan dalam proses pengembangan perangkat lunak. UML terdiri dari 13 macam diagram yang dikelompokan dalam 3 kategori yaitu[15]:

1. *Structure Diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2. *Behavior Diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sistem.
3. *Interaction Diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem yang lain.

2.2.12.1 *Use Case Diagram*

Use Case Diagram menggambarkan suatu unit fungsionalitas yang di sediakan. *Use Case Diagram* merupakan pemodelan untuk melakukan (*behavior*) sistem informasi yang dibuat[15]. *Use Case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem infromasi yang akan dibuat. Tujuan *Use Case Diagram* adalah membantu tim pengembang alam meng-visualisasikan kebutuhan fungsional dari suatu sistem, yang meliputi '*actor*' dengan proses-proses penting. Berikut ini adalah simbol-simbol yang digunakan pada *Use Case Diagram*:

Tabel 2.3: Simbol-Simbol *Use Case Diagram*

Simbol	Deskripsi
Use Case  *Nama Use Case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan dengan unit atau aktor biasanya dinyatakan dengan menggunakan kata kerja diawal *frase* nama *Use Case.*
Aktor/*actor*  *Nama aktor*	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi *actor* belum tentu merupakan orang.
Asosiasi/*asosiation* 	Komunikasi antar aktor dan *use case* dan berpartisipasi pada *use case* atau memiliki interaksi dengan aktor.
Ekstensi/*extend* *<<extend>>* 	*Relasiuse case* tambahan kesebuah *use case* dimana *use case* yang ditambahkan dapat berdiri walau tanpa *use case* tambahan itu.

Include <<*include* 	Mewakili fungsionalitas satu *use case* dengan *use case* lainya, serta arah panah digambar dari *use case* besar menuju ke *case* fungsionalnya.
Generalisasi/*generalization* 	Mewakili *use case* spesifik ke *use case* yang lebih umum, dan arah panah digambar dari *use case* spesifik menuju *use case* dasar.

Sumber: Noviani Mahmud (*Skripsi*, FIKOM 2017)

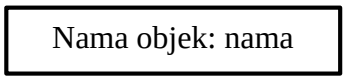
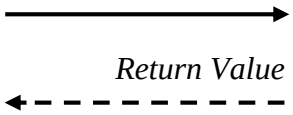
2.2.12.2 *Sequence Diagram*

Sequence Diagram menggambarkan suatu aliran detil untuk masing-masing *use case* secara spesifik atau bahkan hanya sebagian dari use case spesifik. *Sequence Diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *massage* yang dikirimkan dan diterima antar objek.

Sequence Diagram memiliki data dimensi. Dimensi vertikal menunjukan urutan pesan-pesan atau panggilan pada saat adanya suatu kejadian. Dimensi horizontal melanjutkan instansi objek dimana pesan-pesan dikirim. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefisinian *use case* yang memiliki proses sendiri. Berikut adalah simbol-simbol yang ada pada *Sequence Diagram:*

Tabel 2.4: Simbol-Simbol *Sequence Diagram*

Simbol	Deskripsi
Aktor/*actor*  *Nama aktor*	Aktor adalah orang atau pengguna sistem yang berada di luar sistem. Aktor berpartisifasi secara berurutan dengan mengirim dan menerima pesan, aktor ditempatkan di bagian atas diagram dan digambarkan dalam bentuk tongkat tang membentuk orang dan apabila aktor bukanlah manusia

	maka lambang aktor menjadi "<<*actor*>>"
Object 	Berpartisifasi secara berurutan dengan mengirim dan menerima pesan, dan ditempatkan dibagian atas diagram.
Linelife 	Menunjukan kehidupan objek dalam urutan dan *linelife* berisi "X" pada titik dimana objek tidak lagi berinteraksi.
Execution occurrence 	Merupakan persegi panjang sempit yang ditempatkan di atas sebuah garis hidup dan menunjukan suatu objek mengirim atau menerima data.
message 	Menyampaikan informasi dari suatu objek ke satu sama lain dan *message* diberi label sesuai dengan pesan yang dikirim dan panah yang solid, sedangkan *reutrn value* sebuah label dengan nilai yang dikembalikan dan ditampilkan sebagai panah putus-putus.

Sumber: Noviani Mahmud (*Skripsi*, FIKOM 2017)

2.2.12.3 *Activity Diagram*

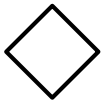
Activity Diagram menggambarkan *workflow* (aliran kerja) prosedural antara dua atau lebih *object class* ketika melakukan pemrosesan suatu aktivitas. *Activity diagram* dapat digunakan untuk menggambarkan proses bisnis pada tingkat yang lebih tinggi pada level unit bisnis, atau menggambarkan aksi *Class Internal* pada

tingkat rendah. Diagram *activity* juga banyak digunakan untuk mendefinisikan hal-hal sebagai berikut[15]:

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas memerlukan sebuah pengujian yang perlu didefinisikan ujinya.
4. Merancang menu yang ditampilkan pada perangkat lunak.

Berikut ini adalah simbol-simbol yang ada pada *activity diagram* memiliki 6 simbol:

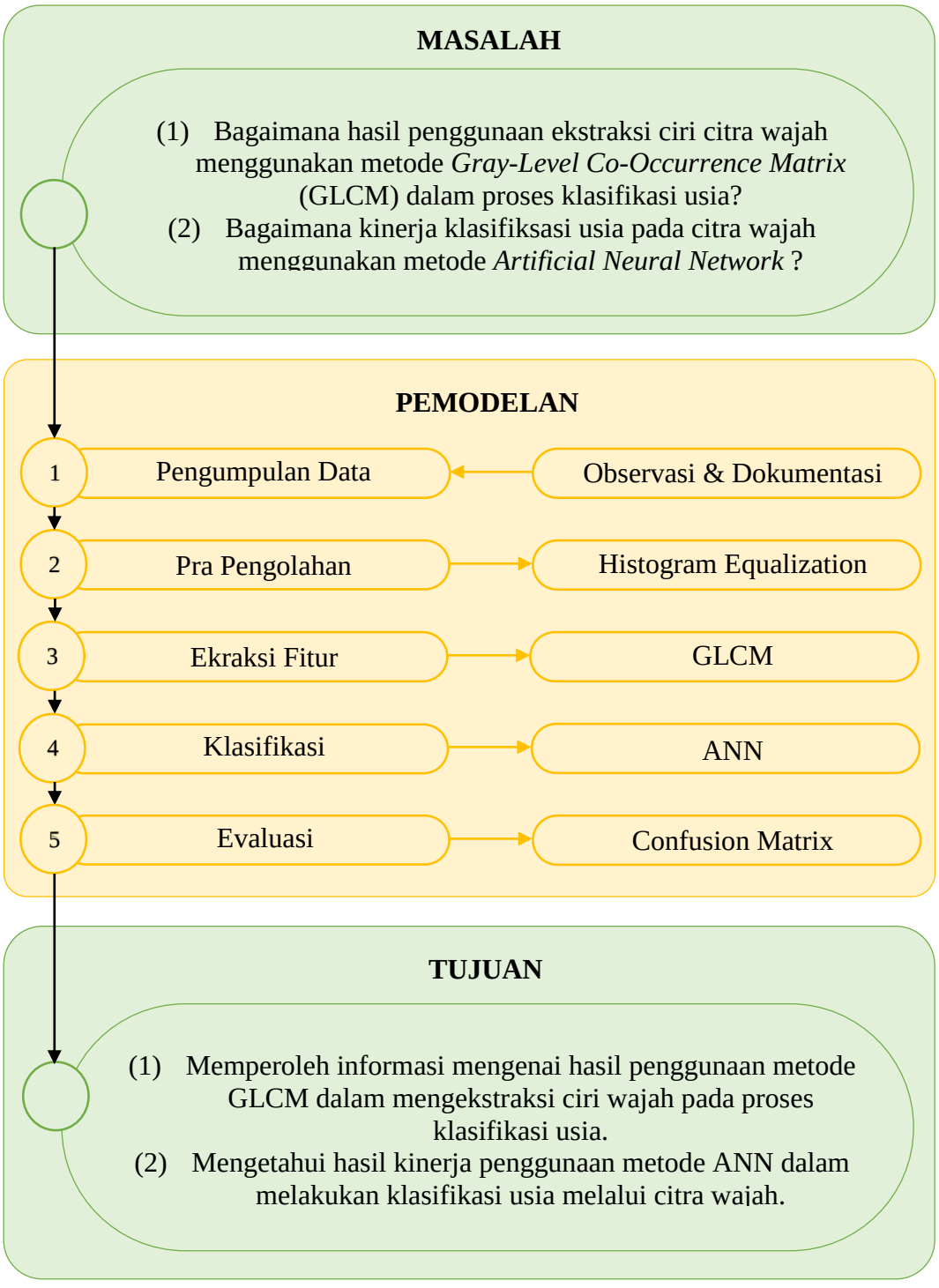
Tabel 2.5: Simbol-Simbol *Activity Digram*

Simbol	**Deskripsi**
Status awal 	Mengambarkan awal dari serangkaian tindakan atau kegiatan.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali kata kerja.
Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan/*join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Digunakan untuk menghentikan semua aliran kontrol dan arus objek dalam satu kegiatan.

Swimlance Nama swimlance	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Sumber: Noviani Mahmud (*Skripsi*, FIKOM 2017)

2.1 Kerangka Pikir

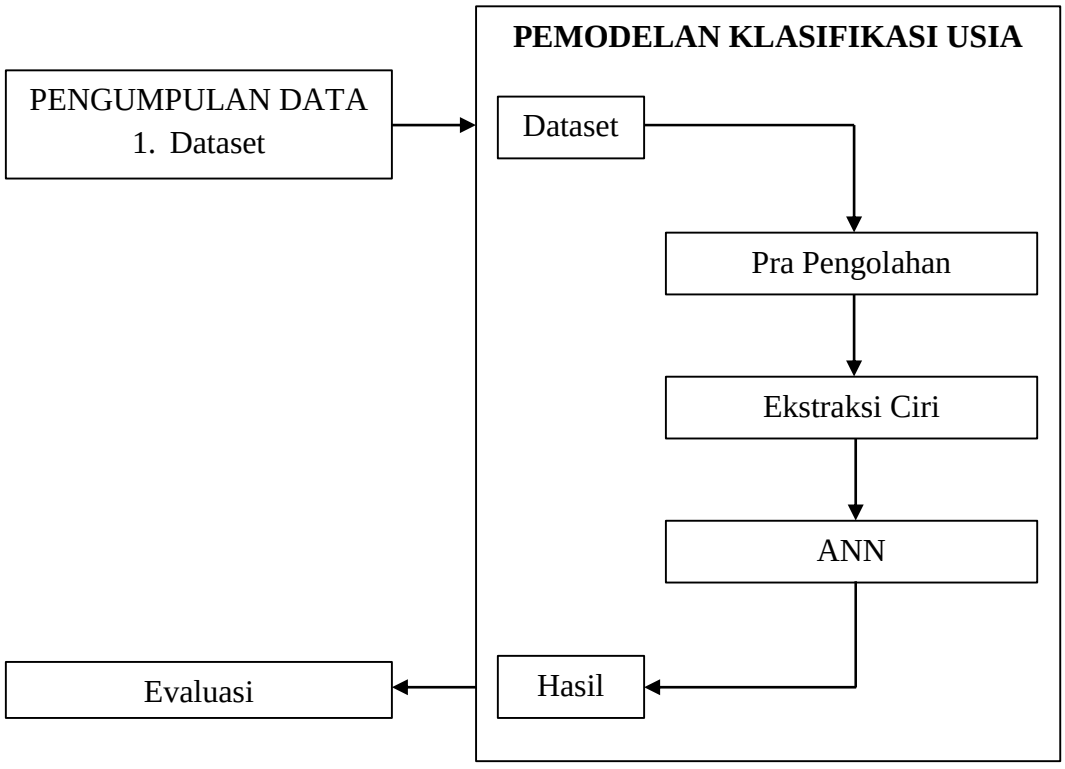


BAB III
METODE PENELITIAN

3.1 Jenis, Metode, Subjek, Objek, Waktu dan Lokasi Penelitian

Penelitian ini merupakan metode penelitian *eksperimen* . dengan demikian jenis penelitian ini adalah penelitian *eksperimental*.

Subjek penelitian ini adalah klasifikasi pada citra wajah. Penelitian ini mulai dari Desember 2018 s/d Maret 2019.

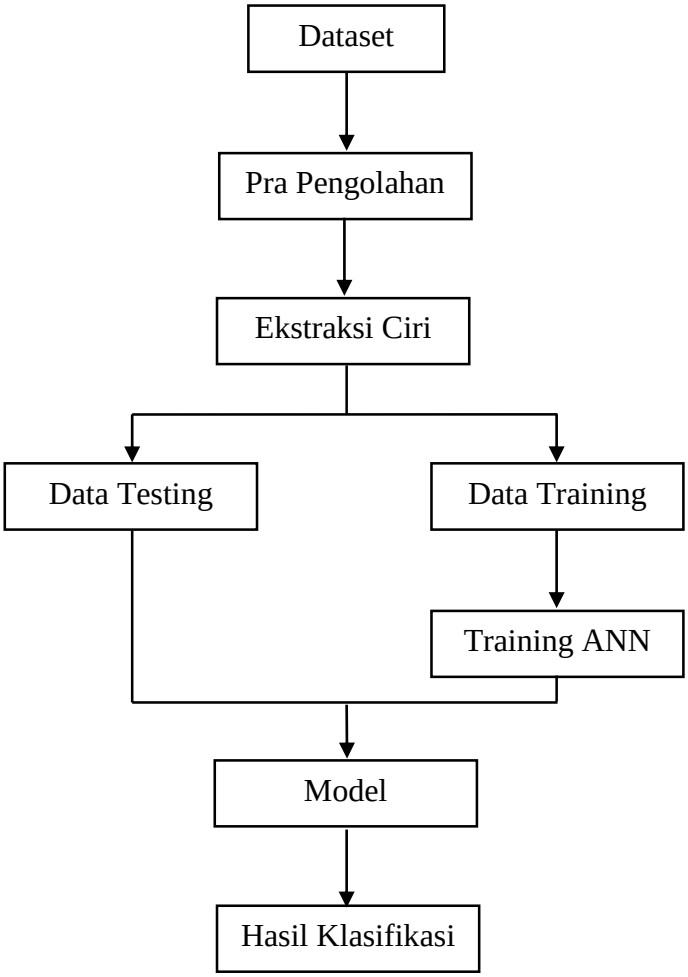


Gambar 3.1: Desain Penelitian

3.2 Pengumpulan Data

Data primer penelitian ini adalah citra wajah yang merupakan data *public* yang diambil dari sebuah penyedia dataset citra wajah yaitu *The face and gesture recognition network* (FG-NET) dengan jumlah data 1.002 file dalam format JPG dengan rentang usia 0 – 69 tahun.

3.3 Pemodelan



Gambar 3.2: Pemodelan

3.3.1 Pra Pengolahan Data

Sebelum data diolah, terlebih dahulu dilakukan proses *Histogram Equalization*, hal ini dilakukan karena *Histogram Equalization* adalah sebuah proses yang mengubah distribusi nilai derajat keabuan pada sebuah citra sehingga menjadi seragam (uniform). Tujuan dari *histogram equalization* adalah untuk memperoleh penyebaran histogram yang merata sehingga setiap derajat keabuan memiliki jumlah piksel yang relatif sama.

3.3.2 Ekstraksi Ciri

Ekstraksi ciri berfungsi sebagai pendeteksi ciri dari suatu citra. Ciri yang dapat digunakan untuk membedakan objek satu dengan objek lainya, diantaranya adalah ciri bentuk, ukuran, ciri geometri, ciri tekstur, dan warna. Pada penelitian ini digunakan ekstraksi ciri tekstur menggunakan *Gray Level Co-Occurrence Matrix* (GLCM). Masing-masing objek diekstrak cirinya berdasarkan parameter-parameter tertentu dan dikelompokan pada kelas tertentu. Nilai dari parameter-parameter tersebut kemudian dijadikan sebagai data masukan dalam proses identifikasi klasifikasi.

3.3.3 Data Training

Data training ini merupakan kumpulan data yang telah terekstraksi cirinya yang selanjutnya akan dilatih menggunakan algoritma *artificial neural network backpropagation.* Algoritma ini akan menentukan/mencari bobot yang terbaik.

3.3.4 Training ANN

Proses training ANN dilakukan dengan menjadikan data training sebagai input untuk menemukan model yang tepat. Arsitektur dari algoritma ANN akan dilakukan secara eksperimen untuk menentukan komposisi jumlah layer dan neuron yang tepat untuk mendapatkan hasil kinerja terbaik.

3.3.5 Model Klasifikasi

Model klasifikasi merupakan tahapan yang terdiri dari *input, hidden layer,* dan *output* dengan kombinasi bobot terbaik yang dihasilkan dari proses pelatihan data menggunakan metode ANN. Model ini akan digunakan untuk mengklasifikasikan data *testing* berdasarkan *output* yang dihasilkan.

3.3.6 Data Testing

Data testing merupakan data yang telah terekstraksi cirinya yang digunakan untuk menguji data yang telah ditraining.

3.3.7 Hasil Klasifikasi

Hasil klasifikasi merupakan hasil yang didapatkan dari proses klasifikasi yang dilakukan model.

3.4 Evaluasi

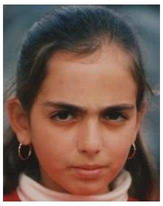
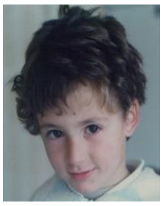
Proses evaluasi dalam penelitian ini memiliki tujuan untuk mengetahui kinerja dari metode tekstur analisis yang digunakan. Proses evaluasi dilakukan pada seluruh data *image testing* kemudian target *output* yang dihasilkan akan dipetakan ke dalam *confussion matrix* untuk kemudian menghitung nilai akurasinya.

BAB IV
HASIL PENELITIAN

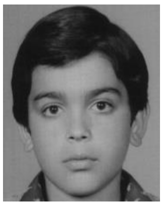
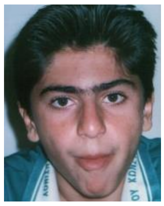
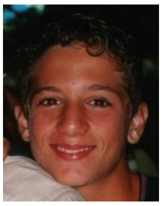
4.1 Hasil Pengumpulan Data

Data yang digunakan dalam penelitan ini merupakan data citra *RGB* yang merupakan data *public* dari penyedia dataset *FG-NET*, masing-masing data terdiri data *training* sebanyak, 30 citra anak-anak, 30 citra remaja, 30 citra dewasa. Sedangkan data *testing* sebanyak 5 citra anak-anak, 5 citra remaja, dan 5 citra dewasa, jumlah keseluruhan data sebanyak 105 citra dalam format *JPG* dengan rentang usia 0 – 69 tahun.

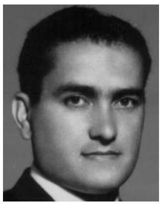
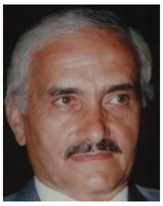
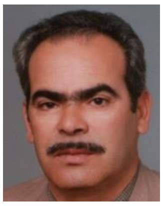
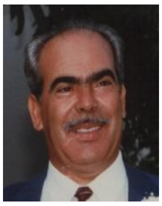
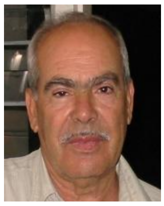
Tabel 4.1: Hasil Pengumpulan Data (Anak-Anak)

Anak-Anak				
				
				
				
				

Tabel 4.2: Hasil Pengumpulan Data (Remaja)

Remaja				
				
				
				
				
				
				

Tabel 4.3: Hasil Pengumpulan Data (Dewasa)

Dewasa				
				
				
				
				
				
				

.

4.2 Hasil Pemodelan

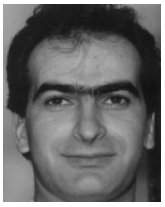
4.2.1 Pra Pengolahan

Pada tahap ini citra wajah akan dikonversi agar diperoleh data citra wajah sesuai kebutuhan. Tahap ini berfungsi untuk menormalisasi citra wajah dari permasalahan luminasi yang teralalu gelap atau terlalu terang sehingga dapat meningkatkan performa dari sistem deteksi usia. Pra-pengolahan dalam penelitian ini terdiri dari:

1. *Grayscale*

Data *training* dan *testing* yang awalnya berbentuk citra dari *RGB* (*red, green, blue*) di konversi menjadi citra *grayscale*, perubahan ini dilakukan karena citra *grayscale* memiliki persamaan yang sederhana dan mampu mengurangi kebutuhan *memory* dimana nilai warna putih diwakili dengan angka 255 dan nilai warna hitam diwakili dengan angka 0. Berikut hasil perubahan citra *RGB* ke *grayscale*.

Tabel 4.4: Hasil Konversi data citra dari *RGB* ke *Grayscale*

Anak-Anak	Remaja	Dewasa
		
		
		

2. *Histogram Equalization*

Setelah citra *RGB* dikonversi kecitra *Grayscle* maka prapengolahan selanjutnya adalah normalilasi citra dengan *Histogram Equalization*. Berikut hasil normalisasi citra *Grayscale* dengan *Histogram Equalization*.

Tabel 4.5: Hasil Normalisasi citra menggunakan *Histogram Equalization*

Anak-Anak	Remaja	Dewasa
		
		
		
		
		

4.2.2 Ekstraksi Ciri

Setelah diperolah hasil dari pra-pengolahan citra wajah, tahapan selanjutnya adalah ekstraksi ciri dari objek di dalam citra wajah baik citra *training* maupun citra *testing*. Metode yang digunakan yakni *Gray Leve Co-Occurrence Matrix.*

Berikut *matrix* hasil prapengolahan dari salah satu *image testing* dengan ukuran 106 x 106 *pixels*. Untuk memudahkan perhitungan, diambil *matrix* dengan ukuran 3 x 3 *pixels* dari ukuran *matrix* sebenarnya.

	0	1	2
0	11	16	16
1	18	11	19
2	19	14	19

Gambar 4.1: *Matrix* GLCM

i/j	11	14	16	18	19
11	0	0	1	0	1
14	0	0	0	0	1
16	0	0	1	0	0
18	1	0	0	0	0
19	0	1	0	0	0

Gambar 4.2: *Matrix GLCM* Jarak 1 dan sudut 0^o

i/j	11	14	16	18	19
11	0	0	0,167	0	0,167
14	0	0	0	0	0,167
16	0	0	0,167	0	0
18	0,167	0	0	0	0
19	0	0,167	0	0	0

Gambar 4.3: *Matrix* Ternormalisasi

1. Menghitung nilai *Contrast*

   Contrast = $((11\text{-}18)^2 * 0{,}167) + ((14\text{-}19)^2 * 0{,}167) + ((16\text{-}11)^2 * 0{,}167) +$
   $((16\text{-}16)^2 * 0{,}167) + ((19\text{-}11)^2 * 0{,}167) + ((19\text{-}14)^2 * 0{,}167)$

   Contrast = $8.183+4.175+4.175+0+10.688+4.175$

   *Contras*t = $31{,}396$

2. Menghitung nilai *Correlation*

   $\mu i = (11 * 0.167) + (14 * 0.167) + (16 * 0.167) + (16 * 0.167) +$
   $(19 * 0.167) + (19 * 0.167)$

   $\mu i = 1{,}837 + 2{,}338 + 2{,}672 + 2{,}672 + 3{,}173 + 3{,}173$

   $\mu i = 15{,}865$

   $\mu j = (11 * 0.167) + (11 * 0.167) + (14 * 0.167) + (16 * 0.167) +$
   $(18 * 0.167) + (19 * 0.167)$

   $\mu j = 1{,}837 + 1{,}837 + 2{,}338 + 2{,}672 + 3{,}006 + 3{,}173$

   $\mu j = 14{,}863$

$$\sigma i = \sqrt{\begin{matrix}((0{,}167(11 - 15{,}865)^2) + ((0{,}167(14 - 15{,}865)^2) + \\ ((0{,}167(16 - 15{,}865)^2) + \ ((0{,}167(16 - 15{,}865)^2) + \\ ((0{,}167(19 - 15{,}865)^2) + \ ((0{,}167(19 - 15{,}865)^2)\end{matrix}}$$

$$\sigma i = \sqrt{\begin{matrix}((0{,}167 * (-4{,}865)^2) + ((0{,}167 * (-1{,}865)^2) + \\ ((0{,}167 * (0{,}135)^2) + \ ((0{,}167 * (0{,}135)^2) + \\ ((0{,}167 * (3.135)^2) + \ ((0{,}167 * (3.135)^2)\end{matrix}}$$

$$\sigma i = \sqrt{\begin{matrix}3{,}952 + 0{,}580 + 0{,}003 \ + \\ 0{,}003 + 1{,}641 \ + \ 1{,}641\end{matrix}}$$

$\sigma i = 2{,}8$

$$\sigma j = \sqrt{\begin{matrix}((0{,}167(11-14{,}863)^2)+((0{,}167(11-14{,}863)^2)+\\((0{,}167(14-14{,}863)^2)+\ ((0{,}167(16-14{,}863)^2)+\\((0{,}167(18-14{,}863)^2)+\ ((0{,}167(19-14{,}863)^2)\end{matrix}}$$

$$\sigma j = \sqrt{\begin{matrix}2{,}492+2{,}492+\ 0{,}124+\\0{,}216+\ 1{,}643\end{matrix}}$$

$$\sigma j = 2{,}6$$

Correlation = ((11 -15,865)*(18 - 14,863)*(0,167) / (2,8 ∗ 2,6)) +
((14- 15,865)*(19- 14,863)*(0,167) / (2,8 ∗ 2,6)) +
((16- 15,865)*(11- 14,863)*(0,167) / (2,8 ∗ 2,6)) +
((16- 15,865)*(16- 14,863)*(0,167) / (2,8 ∗ 2,6)) +
((19- 15,865)*(11- 14,863)*(0,167) / (2,8 ∗ 2,6)) +
((19- 15,865)*(14- 14,863)*(0,167) / (2,8 ∗ 2,6))

Correlation = -0.8765

3. Menghitung nilai *Energy*

Energy = $(0{,}167)^2+(0{,}167)^2+(0{,}167)^2+(0{,}167)^2+(0{,}167)^2+(0{,}167)^2$
$= 0.167$

4. Menghitung nilai *Entropy*

Entropy = (0,167*log 0,167)+(0,167*log 0,167)+(0,167*log 0,167)+
(0,167*log 0,167)+(0,176*log 0,167) +(0,176*log 0,167)
= -0.779

5. Menghitung nilai *Dissimilarity*

Dissimilarity = |11-18|*0,167+|14-19|*0,167+|16-11|*0,167+|16-16|*
0,167+|19-11|*0,167+|19-14|*0,167
= 5,01

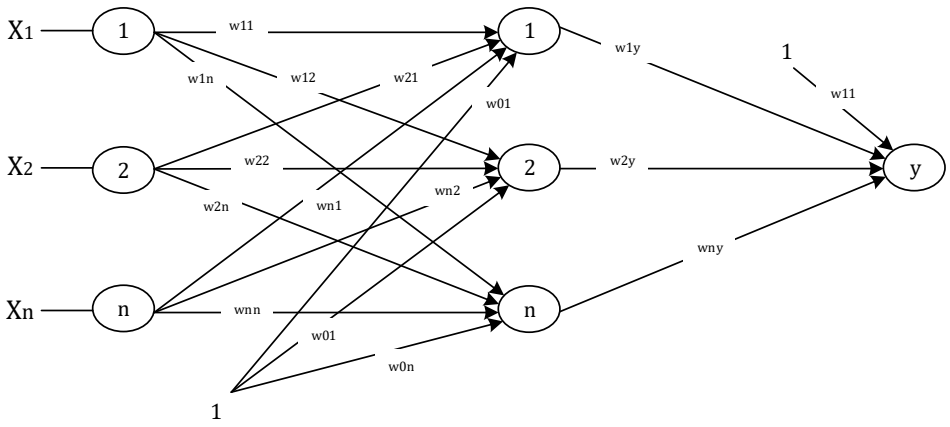
Berikut hasil ekstraksi ciri yang dihasilkan oleh fitur GLCM dari salah satu *image testing* :

Tabel 4.6: Ekstraksi Ciri GLCM dengan Jarak piksel 1

		
0°	*Contrast*	3424290
	Dissimilarity	131672
	Energy	259.37231926325524
	Correlation	0.9999999999452385
	Entropy	7.87770518843328
45°	*Contrast*	5547314
	Dissimilarity	163070
	Energy	241.16384471972577
	Correlation	0.9999999999088534
	Entropy	8.002181685355117
90°	*Contrast*	3505443
	Dissimilarity	126181
	Energy	267.4210163767986
	Correlation	0.9999999999434425
	Entropy	7.816567291759037
135°	*Contrast*	3424290
	Dissimilarity	131672
	Energy	259.37231926325524
	Correlation	0.9999999999452385
	Entropy	7.87770518843328

4.2.3 *Training Artificial Neural Network (*ANN)

Pada penelitian ini metode klasifikasi yang digunakan adalah *Artificial Neural Network*. Pada proses *training* ini berguna untuk mencari bobot terbaik untuk setiap neuron. Berikut arsitektur *Artificial Neural Network* :



Gambar 4.4: Arsitektur *Artificial Neural Network*

Berikut salah satu hasil fitur ekstraksi dengan orientasi arah sudut 0° dan jarak piksel = 1, yang akan digunakan untuk perhitungan klasifikasi menggunakan metode *Artificial Neural Network*:

Tabel 4.7: Dataset perhitungan ANN

Inputan	Nilai
X1	3424290
X2	131672
X3	259.37
X4	0,99
X5	7.87
Y	0

Tahap *training* menggunakan algoritma ANN:

- Inisialisasi parameter yang diperlukan:
- Laju pembelajaran($\boldsymbol{\eta}$) = 0.1
- Fungsi aktivasi = *Logistic*
- Target *error* = 0.0001
- Maksimum iterasi = 1000 kali

– Bobot awal :

Tabel 4.8: Nilai bobot awal

W11	**W12**	**W21**	**W22**	**W31**
0.3	0.5	0.5	0.1	0.2
W32	**W41**	**W42**	**W51**	**W52**
0.3	0.5	0.2	0.5	0.4
W01	**W02**	**W1y**	**W2y**	**W0y**
-0.2	-0.3	0.2	0.3	-0.3

1. Nilai pada neuron di *hidden layer* 1 dan 2 :

   (P) adalah iterasi ke=i

$$v_1(1) = x1.w_{11} + x2.w_{21} + x3.w_{31} + x4.w_{41} + x5.w_{51} + 1.w_{01}$$
$$= 3424290 * 0.3 + 131672 * 0.5 + 259.37 * 0.2 + 0{,}99 * 0.5 + 7.87 * 0.5 + 1 * (-0.2)$$
$$= 1027287 + 65836 + 51{,}874 + 0{,}495 + 3{,}935 + (-0{,}2)$$
$$= 1093179{,}104$$
$$y_1(1) = \frac{1}{1+e^{-1093179,104}} = \frac{1}{1+2.71828^{-1093179,104}} = 1$$
$$v_2(1) = x1.w_{12} + x2.w_{22} + x3.w_{32} + x4.w_{42} + x5.w_{52} + 1.w_{02}$$
$$= 3424290 * 0.5 + 131672 * 0.1 + 259.37 * 0.3 + 0{,}99 * 0.2 + 7.87 * 0.4 + 1 * (-0.3)$$
$$= 1725393.057$$
$$y_2(1) = \frac{1}{1+e^{-1725393.057}} = 1$$

2. Nilai pada neuron di output layer:

$$v_y(1) = y_1(1)w_{1y} + y_2(1)w_{2y} + 1w_{0y}$$
$$= 1 * 0.2 + 1 * 0.3 + 1 * (-0.3)$$
$$= 0{,}2$$
$$y_y(1) = \frac{1}{1+e^{-0.2}} = 0.81$$

3. Hitung gradient error untuk neuron pada output layer:

Untuk data pertama, target/class nilai yang diharapkan adalah $y_d = 0$, sedangkan keluaran yang didaptkan $y_y(1) = 0.81$. Maka hitung *error* pada iterasi pertama untuk data pertama pada neuron Y di output layer:

$$e_5^1(1) = y_d - y_y(1) = 0 - 0.81$$
$$= -0.81$$
$$\delta_y(1) = y_y(1) * \left(1 - y_y(1)\right) * e_5^1(1)$$
$$= 0.81 * (1 - 0.81\ ) * (-0.81\ ) = -0.12$$

4. Hitung koreksi bobot untuk output layer:

$$\Delta w_{1y} = \eta * y_1(1) * \delta_y(1) = 0.1 * 1 * (-0.12)$$
$$= -0.012$$
$$\Delta w_{2y} = \eta * y_2(1) * \delta_y(1) = 0.1 * 1 * (-0.12)$$
$$= -0.012$$
$$\Delta w_{0y} = \eta * 1 * \delta_y(1) = 0.1 * 1 * (-0.12)$$
$$= -0.012$$

5. Hitung *gradient error* pada hidden layer $\delta_1(1)\ dan\ \delta_2(1)$:

$$\delta_1(1) = y_1(1) * \left(1 - y_1(1)\right) * \delta_y(1).w_{1y}(1)$$
$$= 1 * (1 - 1) * (-0.12)\ * 0.2$$
$$= 0$$
$$\delta_2(1) = y_2(1) * \left(1 - y_2(1)\right) * \delta_y(1).w_{2y}(1)$$
$$= 1 * (1 - 1) * (-0.12) * 0.3$$
$$= 0$$

6. Hitung koreksi bobot untuk hidden layer

$$\Delta w_{11} = \eta * x1 * \delta_1(1) = 0.1 * 3424290 * (0)$$
$$= 0$$
$$\Delta w_{21} = \eta * x2 * \delta_1(1) = 0.1 * 131672 * (0)$$
$$= 0$$

$$\Delta w_{31} = \eta * x3 * \delta_1(1) = 0.1 * 259.37 * (0)$$
$$= 0$$
$$\Delta w_{41} = \eta * x4 * \delta_1(1) = 0.1 * 0{,}99 * (0)$$
$$= 0$$
$$\Delta w_{51} = \eta * x5 * \delta_1(1) = 0.1 * 7.87 * (0)$$
$$= 0$$
$$\Delta w_{01} = \eta * 1 * \delta_1(1) = 0.1 * 1 * (0)$$
$$= 0$$
$$\Delta w_{12} = \eta * x1 * \delta_2(1) = 0.1 * 3424290 * (0)$$
$$= 0$$
$$\Delta w_{22} = \eta * x2 * \delta_2(1) = 0.1 * 131672 * (0)$$
$$= 0$$
$$\Delta w_{32} = \eta * x3 * \delta_2(1) = 0.1 * 259.37 * (0)$$
$$= 0$$
$$\Delta w_{42} = \eta * x4 * \delta_2(1) = 0.1 * 0{,}99 * (0)$$
$$= 0$$
$$\Delta w_{52} = \eta * x5 * \delta_2(1) = 0.1 * 7.87 * (0)$$
$$= 0$$
$$\Delta w_{02} = \eta * 1 * \delta_2(1) = 0.1 * 1 * (0)$$
$$= 0$$

7. Perbaharui bobot untuk neuron pada hidden layer $w_{1y}w_{2y}w_{0y}$:

   $w_{1y}(2) = w_{1y}(1) + \Delta w_{1y} = 0.2 + (-0.012) = 0.18$

   $w_{2y}(2) = w_{2y}(1) + \Delta w_{2y} = 0.3 + (-0.012) = 0.288$

   $w_{0y}(2) = w_{0y}(1) + \Delta w_{0y} = -0.3 + (-0.012) = -0.312$

8. Perbaharui bobot pada layer tersembunyi, $w_{11}, w_{21}, w_{31}, w_{41}, w_{51}w_{01}$, dan $w_{12}, w_{22}, w_{32}, w_{42}, w_{52}w_{02}$

   $w_{11}(2) = w_{11}(1) + \Delta w_{11} = 0.3 + 0 = 0.3$

$w_{21}(2) = w_{21}(1) + \Delta w_{21} = 0.5 + 0 = 0.5$
$w_{31}(2) = w_{31}(1) + \Delta w_{31} = 0.2 + 0 \ = 0.2$
$w_{41}(2) = w_{41}(1) + \Delta w_{41} = 0.5 + 0 = 0.5$
$w_{51}(2) = w_{51}(1) + \Delta w_{51} = 0.5 + 0 = 0.5$
$w_{01}(2) = w_{01}(1) + \Delta w_{01} = (-0.2) + 0 = -0.2$
$w_{12}(2) = w_{12}(1) + \Delta w_{12} = 0.5 + 0 = 0.5$
$w_{22}(2) = w_{22}(1) + \Delta w_{22} = 0.1 + 0 = 0.1$
$w_{32}(2) = w_{32}(1) + \Delta w_{32} = 0.3 + 0 = 0.3$
$w_{42}(2) = w_{42}(1) + \Delta w_{42} = 0.2 + 0 \ = 0.2$
$w_{52}(2) = w_{52}(1) + \Delta w_{52} = 0.4 + 0 = 0.4$
$w_{02}(2) = w_{02}(1) + \Delta w_{02} = (-0.3) + 0 \ = -0.3$

9. Naikkan 1 langkah iterasi (***p***), kembali ke langkah 2 dan ulangi proses tersebut sampai ***kriteria error*** tercapai.

10. Dengan demikian, didapatkan bobot akhir pada iterasi pertama :

$w_{11} = 0.3$
$w_{21} = 0.5$
$w_{31} = 0.2$
$w_{41} = 0.5$
$w_{51} = 0.5$
$w_{01} = -0.2$
$w_{12} = 0.5$
$w_{22} = 0.1$
$w_{32} = 0.3$
$w_{42} = 0.2$
$w_{52} = 0.4$
$w_{02} = -0.3$
$w_{1y} = 0.18$
$w_{2y} = 0.288$

$w_{0y} = -0.312$

Error yang didapatkan dadalah =$-\mathbf{0},\mathbf{81}$. Selanjutnya proses di atas diulangi untuk data selanjutnya sehingga iterasi pertama selesai dilakukan.

Berikut ini hasil pengujian data *testing* menggunakan fitur ekstraksi *Gray Leve Co-Occurrence Matrix* dan metode klasifikasi *Artificial Neural Network*:

Tabel 4.9: Hasil *Testing* anak-anak

NO	Image	GLCM (Arah,Jarak)	ANN (Hidden Layer, Max Iterasi)	Klasifikasi	Aktual
1		0°, 1	32,24, 1000	Anak-anak	Anak-anak
2		0°, 1	32,24, 1000	Anak-anak	Anak-anak
3		0°, 1	32,24, 1000	Anak-anak	Anak-anak
4		0°, 1	32,24, 1000	Anak-anak	Anak-anak
5		0°,1	32,24, 1000	Dewasa	Anak-anak

Tabel 4.10: Hasil *Testing* remaja

NO	Image	GLCM (Arah, Jarak)	ANN (Hidden Layer, Max Iterasi)	Klasifikasi	Aktual
6		0°,1	32,24, 1000	Remaja	Remaja
7		0°,1	32,24, 1000	Remaja	Remaja
8		0°,1	32,24, 1000	Dewasa	Remaja
9		0°,1	32,24, 1000	Anak-anak	Remaja
10		0°,1	32,24, 1000	Anak-anak	Remaja

Tabel 4.11: Hasil *Testing* dewasa

NO	Image	GLCM (Arah,Jarak)	ANN (Hidden Layer, Max Iterasi)	Klasifikasi	Aktual
11		0°,1	32,24, 1000	Dewasa	Dewasa
12		0°,1	32,24, 1000	Dewasa	Dewasa

13		0°,1	32,24, 1000	Dewasa	Dewasa
14		0°,1	32,24, 1000	Dewasa	Dewasa
15		0°,1	32,24, 1000	Dewasa	Dewasa

4.2.4 Evaluasi Model

Penelitian ini menggunakan confusion matrix sebagai metode untuk mengukur seberapa besar akurasi dari sebuah algoritma untuk melakukan klasifikasi. Berikut merupakan tabel *confusion matrix* :

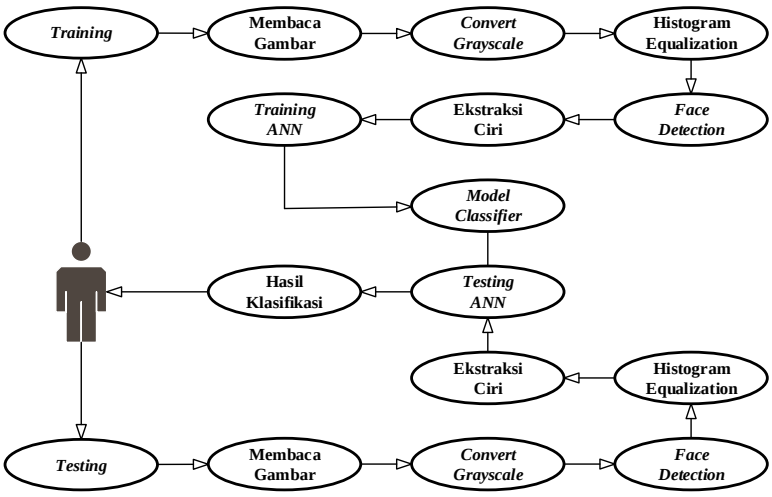
Tabel 4.12: Hasil *Confusion Matrix*

		Prediksi		
		Anak-anak	Remaja	Dewasa
Aktual	Anak-anak	4	0	1
	Remaja	2	2	1
	Dewasa	0	0	5

Akurasi = $\frac{11}{15} * 100\%$ = **73,3%**

4.3 Hasil Pengembangan Sistem

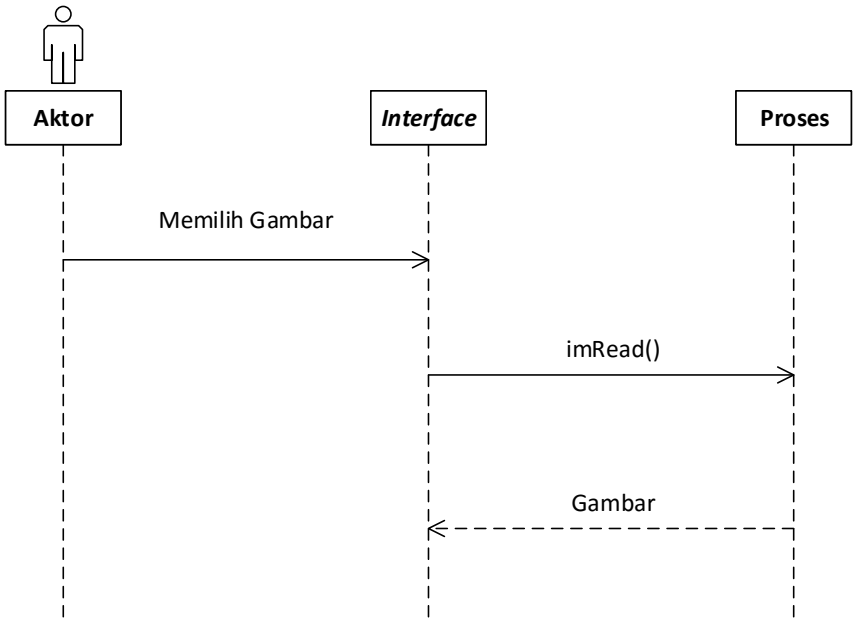
4.3.1 *Use Case Diagram*



Gambar 4.5: *Use Case Diagram*

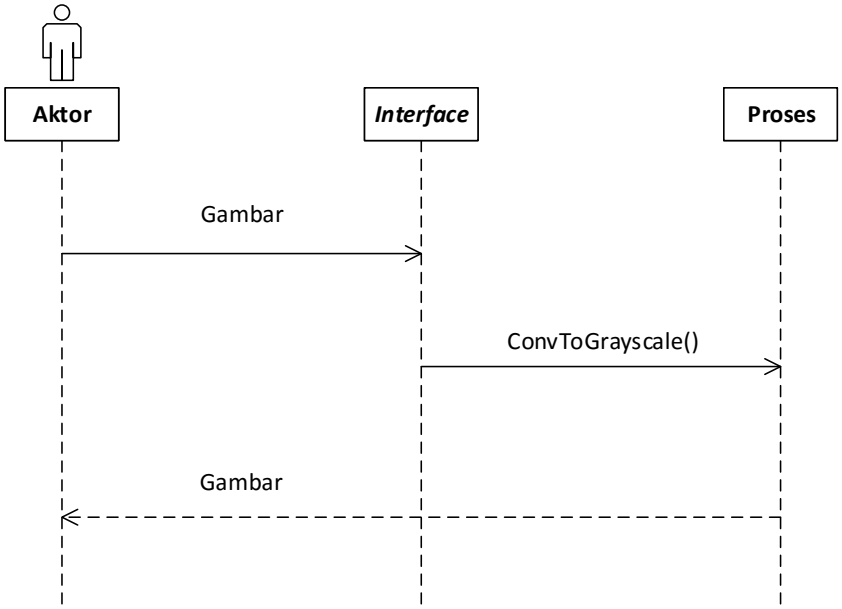
4.3.2 *Sequence Diagram*

1. Membaca Gambar



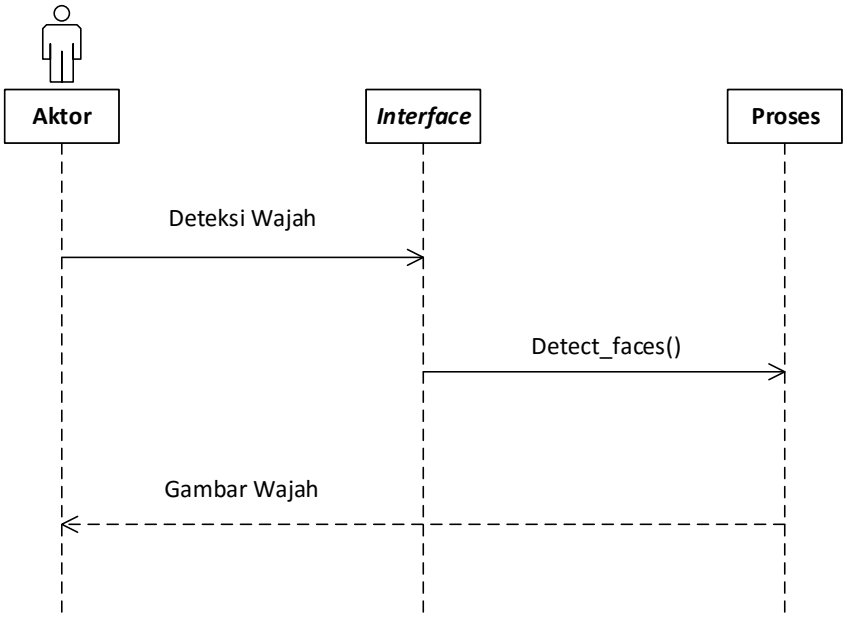
Gambar 4.6: *Sequence Diagram* Membaca Gambar

2. *Convert Grayscale*



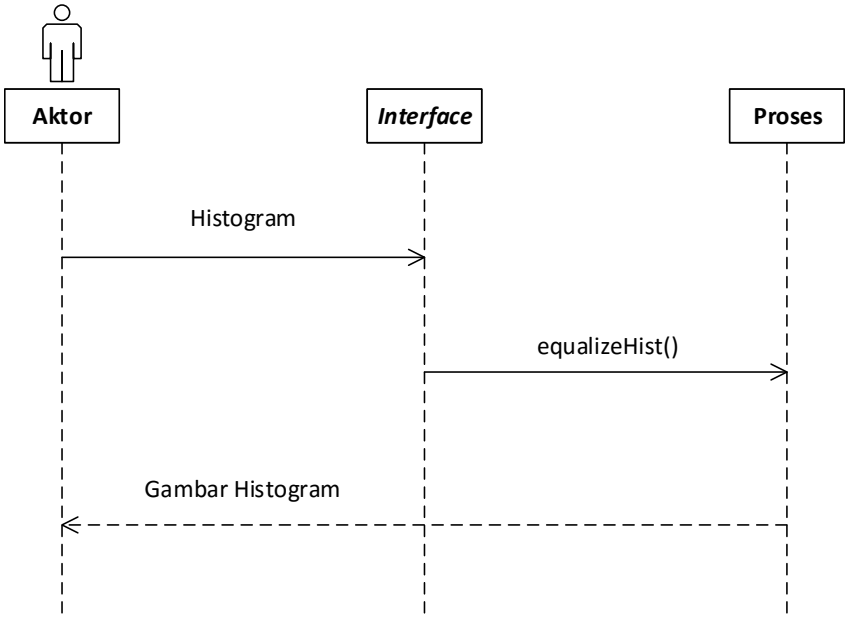
Gambar 4.7: *Sequence Diagram Convert Grayscale*

3. Deteksi Wajah



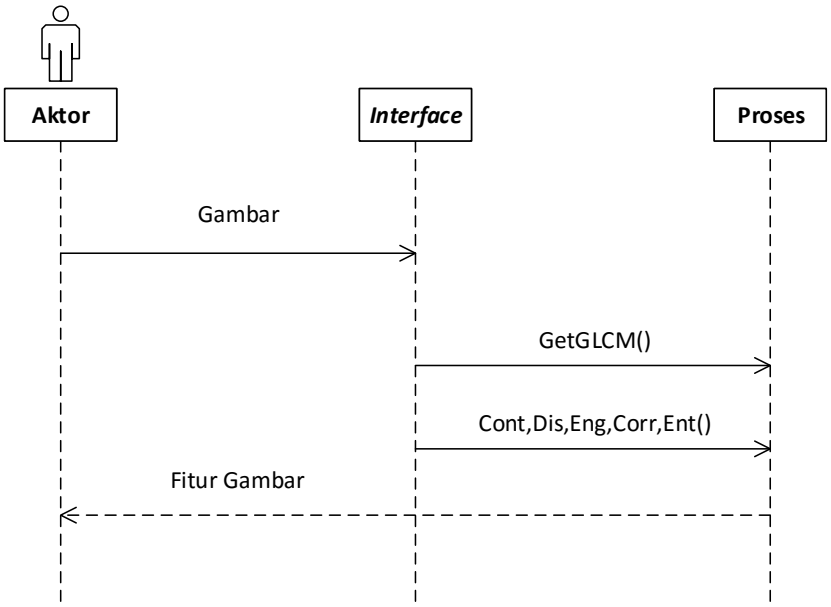
Gambar 4.8: *Sequence Diagram* Deteksi Wajah

4. *Histogram Equalization*



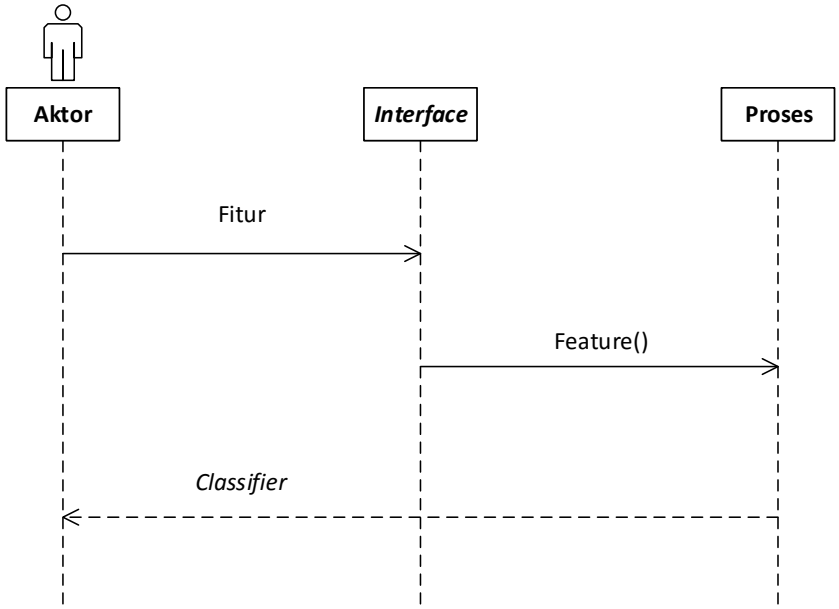
Gambar 4.9: *Sequence Diagram Histogram Equalization*

5. Ekstraksi Ciri



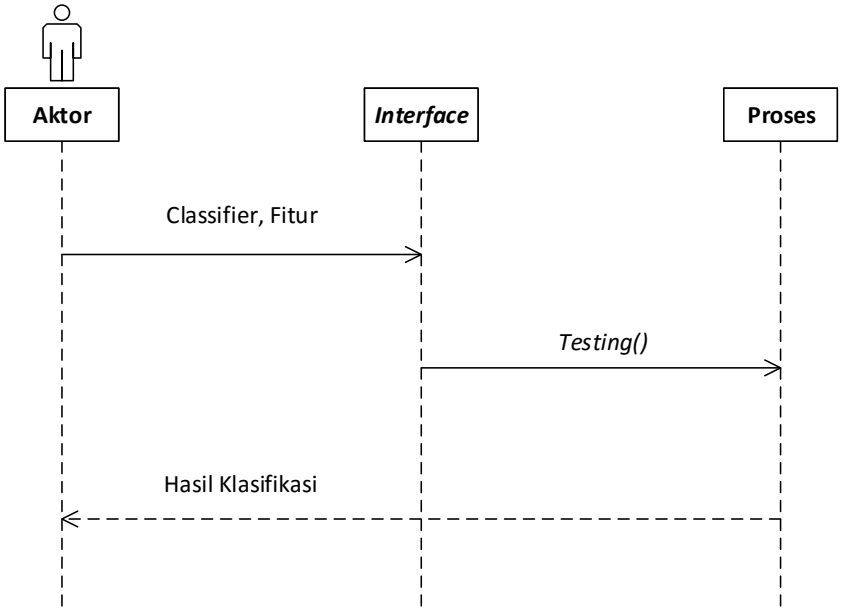
Gambar 4.10: *Sequence Diagram* Ekstraksi Ciri

6. *Training ANN*



Gambar 4.11: *Sequence Diagram Training ANN*

7. *Testing ANN*



Gambar 4.12: *Sequence Diagram Testing ANN*

BAB V
PEMBAHASAN

5.1 Pembahasan Model

Pengumpulan data yang diambil dari penyedi dataset *FG-NET* dengan jumlag data sebanyak 1.002 *image,* beberapa *image* tidak bisa diolah karena faktor *blur*, objek wajah yang miring dan tidak bisa dideteksi wajah, sehingga peneliti mengambil data sebanyak 105 dari 1.002 *image* yang dipilih berdasarkan hasil prapengolahan sistem. 105 *image* yang dipakai dalam penelitian ini terdiri dari 90 data *training* dan 15 data *testing* yang merupakan data *image RGB* dengan format *JPG.* Masing-masing data terdiri dari tiga kategori usia, untuk data *training* terdiri dari 30 *image* anak-anak dengan rentang usia 0 – 10 tahun, 30 *image* remaja dengan rentang usia 12 – 18 tahun dan 30 *image* dewasa dengan rentang usia 30 – 69 tahun. Sedangkan untuk data *testing* terdiri dari 5 *image* anak-anak dengan rentang usia 3 – 9 tahun, 5 *image* remaja dengan rentang usia 12 – 24 tahun dan 5 *image* dewasa dengan rentang usia 29 – 61 tahun.

Sebelum pengolahan data dilakukan terlebih dahulu peneliti melakukan prapengolahan data yaitu mengkonversi data *training* dan *testing* dari *image RGB* ke *Grayscale.* Hal ini dilakukan karena inputan dari metode *Gray Level Co-Occurrence Matrix* merupakan inputan *image grayscale.* Jadi, seluruh data *image RGB* harus dikonversi ke *grayscale*. Setelah didapatkan hasil konversi citra, langkah berikut yang dilakukan yaitu menormalisasi citra menggunakan *Histogram Equalization*. Pra pengolahan selanjutnya yaitu memisahkan background dan wajah menggunakan *Viola Jones*.

Setelah proses prapengolahan selesai, proses selanjutnya yaitu pengolahan data, seluruh dataset (*training* dan *testing*) akan diekstraksi ciri menggunakan metode *Gray Level Co-Occurrence Matrix*. Ciri yang akan dihasilkan dari ekstraksi ciri yaitu *contrast, energy, dissimilarity, correlation* dan *entropy.* Hasil ekstraksi ciri inilah yang akan menjadi data inputan pada proses *training* menggunakan metode *Artificial Neural Network,* inputannya yaitu x1 = *contrast,* x2 = *dissimilarity,* x3 = *energy,* x4 = *correlation,* dan x5 = *entropy*.

Beberapa pelatihan data dilakukan untuk mengukur tingkat akurasi tertinggi. Pelatihan data yang pertama: menggunakan orientasi arah sudut 0°, jarak piksel = 1, fungsi aktivasi = *logistic*, hidden layer 1 = 32 *neuron*, hidden layer 2 = 24 *neuron* dan iterasi maksimum = 1000x, mampu menghasilkan tingkat akurasi yang dihitung menggunakan *confussion matrix* adalah sebesar 73,3%.

Pada pelatihan data yang kedua menggunakan orientasi arah sudut 45°, jarak piksel = 1, fungsi aktivasi = *logistic*, hidden layer 1 = 32 *neuron*, hidden layer 2 = 24 *neuron* dan iterasi maksimum = 1000x, mampu menghasilkan tingkat akurasi sebesar 46,6%. Pada pelatihan data yang ketiga: menggunakan orientasi arah sudut 0°, jarak piksel = 2, fungsi aktivasi = *logistic*, hidden layer 1 = 32 *neuron*, hidden layer 2 = 24 *neuron* dan iterasi maksimum = 1000x, mampu menghasilkan tingkat akurasi yang dihitung menggunakan *confussion matrix* adalah sebesar 40%. Pada pelatihan data yang keempat menggunakan orientasi arah sudut 45°, jarak piksel = 2, fungsi aktivasi = *logistic*, hidden layer 1 = 32 *neuron*, hidden layer 2 = 24 *neuron* dan iterasi maksimum = 1000x, mampu menghasilkan tingkat akurasi sebesar 46,6%.

Sedangkan pada pelatihan data selanjutnya, dengan menggunakan berbagai kombinasi nilai *Gray Level Co-Occurrence Matrix* dan *artificial neural network*, menghasilkan akurasi dibawah 50%.

Untuk lebih jelasnya dapat dilihat pada tabel berikut:

GLCM		***ANN***				**Akurasi**
Sudut	**Jarak**	**Aktivasi**	**HD 1**	**HD 2**	**Iterasi**	
0°	1	*Logistic*	32	24	1000x	73,3%
45°	1	*Logistic*	32	24	1000x	46,6%
90°	2	*Logistic*	32	24	1000x	40%
135°	2	*logistic*	32	24	1000x	46,6%

Tabel 5.1: Pengujian Data

5.2 Pembahasan Sistem

Gambar 5.1: Tampilan Sistem

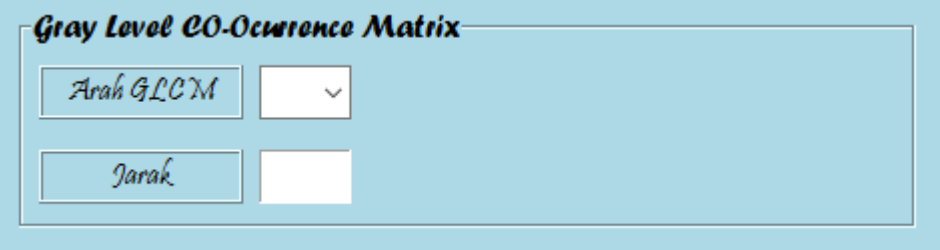
Sistem klasifikasi usia menggunakan satu *form* yang terdiri dari beberapa bagian:

1. Input Gambar

   Input gambar terdiri dari *image training* dan *image testing*.

Gambar 5.2: Input Gambar

2. *Input* nilai *Gray Level Co-Occurrence Matrix* (GLCM), terdiri dari:
   - Orientasi sudut arah: 0°, 45°, 90°, dan 135°.
   - Jarak ketetanggan: 1, 2, 3 dan seterusnya.



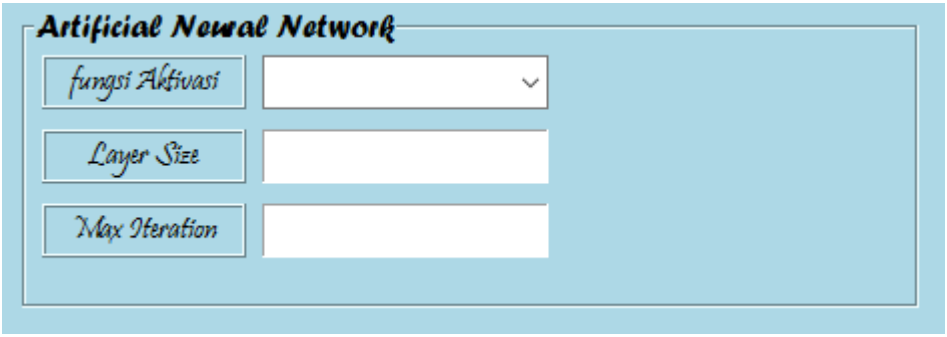
Gambar 5.3: *Input* nilai (GLCM)

3. *Input* nilai *Artificial Neural Network* (ANN)
   - fungsi aktivasi: *logictic, tanh* dan *relu*
   - jumlah *neuron* pada *hidden layer*

     Jika menggunakan *hidden layer* lebih dari 1 *hidden layer* dapat dibatasi dengan tanda koma (,).

     Contoh: 32,24:

     32 adalah jumlah *neuron* pada *hidden layer* pertama, sedangkan 24 adalah jumlah *neuron* pada *hidden layer* kedua dan seterusnya.



Gambar 5.4: *Input* nilai (ANN)

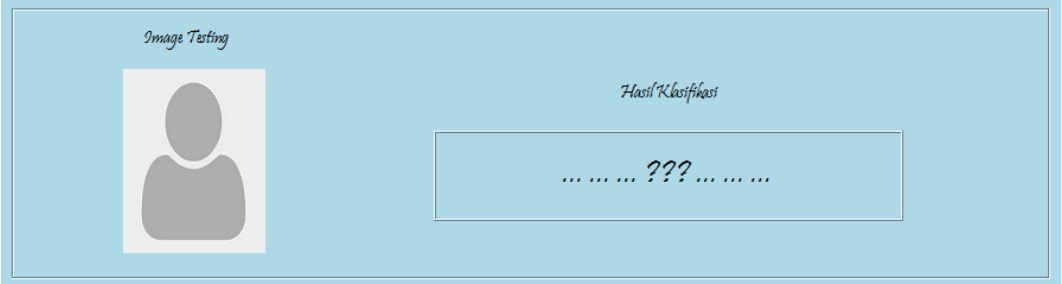
4. Prapengolahan

   Pada prapengolahan akan menampilkan *output* yang berupa hasil konversi *image* dari *RGB* ke *Grayscale*, hasil normalisasi citra menggunakan *Histogram Equalization*, deteksi wajah menggunakan *Viola Jones*, dan hasil *crop image* dari deteksi wajah.

Gambar 5.5: Prapengolahan

5. Hasil Klasifikasi

   Hasil klasifikasi menampilkan kategori usia (anak-anak, remaja dan dewasa) dan akurasi klasifikasi.



Gambar 5.6: Hasil Klasifikasi

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil penelitian dan pembahasan yang telah diuraikan diatas maka dapat ditarik suatu kesimpulan sebagai berikut:

1. Hasil penggunaan metode *Gray Level Co-Occurrence Matrix* (GLCM) dalam mengekstraksi ciri wajah menggunakan metode *artificial neural network* mampu menghasilkan tingkat akurasi yang dihitung menggunakan *confusion matrix* yaitu sebesar 73,3%.
2. Hasil kinerja metode *artificial neural network* dalam melakukan klasifikasi usia pada citra wajah sudah cukup baik, hal ini dilihat dari proses klasifikasi menggunakan inputan hasil parameter dari fitur ekstraksi yang mampu mengklasifikasi usia sesuai kelas yang telah ditentukan.

6.2 Saran

Setelah melakukan Penelitian, ada beberapa saran yang perlu di perhatikan untuk mencapai tujuan yang di harapkan, yaitu sebagai berikut:

1. Perlu diadakan penelitian lebih lanjut, dengan penggunaan metode lainnya terutama pada metode ekstraksi ciri sebagai perbandingan untuk mendapatkan hasil terbaik.
2. Untuk peneltian selanjutnya bisa menggunakan data real dengan kualitas gambar yang bagus.

DAFTAR PUSTAKA

[1] Muchammad Al Amin dan Dwi Juniati, "Klasifikasi Kelompok Umur Manusia Berdasarkan Analisis Dimensi *Fraktal Box Counting* Dari Citra Wajah Dengan Deteksi Tepi *Canny*," Volume 2 No.6 Tahun 2017.

[2] Nurhayatin, "Klasifikasi Kelompok Usia Berdasarkan Citra Wajah Menggunakan Algoritma *Neural Network* Dengan Fitur *Face Anthropometry* Dan Kedalam Kerutan, 2016.

[3] Ranita, Achmad Rizal dan Ratri Dwi Atmaja, "Deteksi Kelompok Usia Manusia Berdasarkan Fitur Wajah Menggunakan *Filter Gabor 2d*," 2012.

[4] Maghfirah Ramadhani, Drs. Suprayogi, M.T[2], Hertiana Bethaningtyas Dyah K.,S.T,M.T[3], "Klasifikasi Jenis Jerawat Berdasarkan Tekstur Dengan Menggunakan Metode Glcm," *e-Proceeding of Engineering*: Vol.5, No.1 Maret 2018.

[5] Elvia Budianita, Jasril, Lestari Handayani, "Implementasi Pengolahan Citra Dan Klasifikasi K-Nearest Neighbour Untuk Membangun Aplikasi Pembeda Daging Sapi Dan Babi," *Jurnal Sains, Teknologi dan Industri*, Vol. 12, No. 2, Juni 2015, pp.242 – 247, Juni 2015.

[6] RD. Kusumanto, Alan Novi Tompunu, "Pengolahan Citra Digital Untuk Mendeteksi Obyek Menggunakan Pengolahan Warna Model Normalisasi Rgb," *Seminar Nasional Teknologi Informasi & Komunikasi Terapan 2011*.

[7] Toni Wijanarko Adi Putra, Kusworo Adi, R. Rizal Isnanto, "Pengenalan Wajah Dengan Matriks Kookurensi Aras Keabuan Dan Jaringan Syaraf Tiruan Probabilistik," *Jurnal Sistem Informasi Bisnis*, juli 2013.

[8] Edo Satria Noviando, Edy Erviando, Indra Yasri, "Studi Penerapan ANN (*Artificial Neural Network*) Untuk Menghilangkan Harmonisa Pada Gedung Pusat Komputer," *Jom FTEKNIK* Volume 3 No. 2 Oktober 2016.

[9] Iwan K. Hadihardaja, Sugeng Sutikno, "Pemodelan Curah Hujan-Limpasan Menggunakan *Artificial Neural Network* (ANN) dengan Metode *Backpropagation,"* Vol. 12 No. 4 Oktober 2005.

[10] Neneng, Kusworo Adi dan R. Rizal Isnanto, "*Support Vector Machine* Untuk Klasifikasi Citra Jenis Daging Berdasarkan Tekstur Menggunakan Ekstraksi Ciri *Gray Level Co-Occurrence Matrix* (GLCM)," 2016.

[11] Ririn, "Klasifikasi Resiko Kredit Motor Bekas Menggunakan Metode *Naïve Bayes," Skripsi*. Fakultas Ilmu Komputer, Universitas Ichsan Gorontalo, 2018.

[12] Rudiati Evi Masithoh Dkk, "Pengembangan *Computer Vision System* Sederhana Untuk Menentukan Kualitas Tomat," *AGRITECH,* Vol. 31, No. 2, Mei 2011.

[13] Maghfirah Maulani Katalani, "Aplikasi Pengolahan Citra Mengklasifikasi Kematangan Buah Jeruk Keprok Menggunakan Metode *Artificial Neural Network* Berdasarkan Tekstur*," Skripsi*. Fakultas Ilmu Komputer, Universitas Ichsan Gorontalo, 2017.

[14] Intan, "Klasifikasi Kelayakan Pembelian Mobil Menggunakan Algoritma C4.5," *Skripsi,* Fakultas Ilmu Komputer, Universitas Ichsan Gorontalo, 2018.

[15] Noviani Mahmud, "Pemetaan Sebaran Alumni Dengan Menggunakan Algoritma K-Means*," Skripsi*. Fakultas Ilmu Komputer, Universitas Ichsan Gorontalo, 2017.

KODE PROGRAM

1. PEMBUATAN FORM

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""

import tkinter as tk
from tkinter import filedialog, ttk
from PIL import Image, ImageTk
import cv2
import proses as ip
from ast import literal_eval
from sklearn.preprocessing import OrdinalEncoder



class Main(tk.Frame):
    def __init__(self,parent):
        tk.Frame.__init__(self)
        self.parent = parent

        #frame judul
        self.MainFrame = tk.Frame(self.parent, width=1347, height=150,
background="red")
        self.MainFrame.grid(row=0, column=0, columnspan=3, padx=10, pady=15,
sticky="NSEW")
        self.MainFrame.grid_propagate(0)
        #frame input gambar
        self.MainFrame1 = tk.LabelFrame(self.parent, height=107,
background="lightblue", text="Input Gambar", font="forte", fg="black")
        self.MainFrame1.grid(row=1, column=0, padx=10, sticky="NSEW")
        self.MainFrame1.grid_propagate(0)
        #glcm
```

```
        self.MainFrame2 = tk.LabelFrame(self.parent, height=110,
background="lightblue", text="Gray Level CO-Ocurrence Matrix", font="forte",
fg="black")
        self.MainFrame2.grid(row=2, column=0, padx=10, pady=10,
sticky="NSEW")
        self.MainFrame2.grid_propagate(0)
        #ann
        self.MainFrame3 = tk.LabelFrame(self.parent, height=150,
background="lightblue", text="Artificial Neural Network", font="forte",
fg="black")
        self.MainFrame3.grid(row=3, column=0, rowspan=2,padx=10,
sticky="NSEW")
        self.MainFrame3.grid_propagate(0)
        #proses
        self.MainFrame4 = tk.Frame(self.parent, height=70,
background="lightblue")
        self.MainFrame4.grid(row=6, column=0, padx=10, pady=10,
sticky="NSEW")
        self.MainFrame4.grid_propagate(0)
        #hasil 1
        self.MainFrame5 = tk.LabelFrame(self.parent, background="lightblue",
font="forte", fg="black")
        self.MainFrame5.grid(row=1, column=1, rowspan=2, columnspan=2,
padx=10, pady=6, sticky="NSEW")
        self.MainFrame5.grid_propagate(0)
        #hasil 2
        self.MainFrame7 = tk.LabelFrame(self.parent, background="lightblue",
font="forte", fg="black")
        self.MainFrame7.grid(row=3, column=1, rowspan=4, columnspan=2,
padx=10, pady=6, sticky="NSEW")
        self.MainFrame7.grid_propagate(0)
        #kaki
        self.MainFrame6 = tk.LabelFrame(self.parent, height=30, width=1000,
background="lightblue")
        self.MainFrame6.grid(row=7, column=0, columnspan=3, padx=10,
sticky="NSEW")
        self.MainFrame6.grid_propagate(0)

        self.form()
```

```
    def form(self):

        #frame
        #logo
        self.PathTestingImg = 'E:/logo.png'
        self.imT = Image.open(self.PathTestingImg)
        self.imT = ImageTk.PhotoImage(self.imT)
        self.label1 = tk.Label(self.MainFrame, bg="spring green2", image=self.imT)
        self.label1.pack(side="top")

        #frame 1
        #label data training
        self.label1 = tk.Label(self.MainFrame1, text="Image Training",
font="pristina", relief="ridge", width=12, background="lightblue")
        self.label1.grid(row=3, column=0, sticky="NSEW", padx=8, pady=5)
        #textbox data training
        self.nilai = tk.StringVar(self.MainFrame1, value="")
        self.tbdatatraining = tk.Entry(self.MainFrame1, textvariable = self.nilai,
width=42)
        self.tbdatatraining.grid(row=3, column=1, sticky="NSEW", pady=5)
        #button browse training
        self.path = tk.Button(self.MainFrame1, text="Browse", font="pristina",
width=6, background="lightblue",
command=lambda:self.nilai.set(filedialog.askdirectory()))
        self.path.grid(row=3, column=5, padx=5)
        #label data testing
        self.label2 = tk.Label(self.MainFrame1, text="Image Testing",
font="pristina", relief="ridge", background="lightblue")
        self.label2.grid(row=4, column=0, sticky="NSEW", padx=8, pady=10)
        #textbox data testing
        self.imagetest = tk.StringVar(self.MainFrame1, value="")
        self.tbdatatesting = tk.Entry(self.MainFrame1, textvariable = self.imagetest)
        self.tbdatatesting.grid(row=4, column=1, sticky="NSEW", pady=10)
        #button browse tetsting
        self.path1 = tk.Button(self.MainFrame1, text="Browse",
font="pristina",background="lightblue",
command=lambda:self.imagetest.set(filedialog.askopenfilename()))
        self.path1.grid(row=4, column=5, sticky="NSEW", padx=5, pady=5)

        #frame 2
```

```
#       v = tk.IntVar(self.MainFrame2)
        self.label = tk.Label(self.MainFrame2, text="Arah GLCM", width=12,
font="pristina", relief="ridge", background="lightblue")
        self.label.grid(row=5, column=0, padx=8 , pady=10)

        #combobox arah glcm
        arah = tk.StringVar(self.MainFrame2, value="")
        self.cbarahglcm = ttk.Combobox(self.MainFrame2,
values=("0","45","90","135"), textvariable = arah, width=1)
        self.cbarahglcm.grid(row=5, column=1, columnspan=4, sticky="NSEW",
pady=10)

        #jarak
        self.label1 = tk.Label(self.MainFrame2, text="Jarak", font="pristina",
relief="ridge", width=12, background="lightblue")
        self.label1.grid(row=6, column=0, sticky="NSEW", padx=8, pady=5)
        #textbox jarak
        self.nilaijarak = tk.StringVar(self.MainFrame2, value="")
        self.tbjarak = tk.Entry(self.MainFrame2, textvariable = self.nilaijarak,
width=7)
        self.tbjarak.grid(row=6, column=1, columnspan=4, sticky="NSEW",
pady=5)

        #frame 3
        #klasifikasi ANN
        #label fungsi aktivasi
        self.label = tk.Label(self.MainFrame3, text="fungsi Aktivasi",
font="pristina", relief="ridge", width=12, background="lightblue")
        self.label.grid(row=0, column=0, sticky="NSEW", padx=8, pady=5)
        #combobox fungsi aktivasi
        nilai2 = tk.StringVar(self.MainFrame3, value="")
        self.cbfungsiaktivasi = ttk.Combobox(self.MainFrame3,
values=("identity","logistic","tanh","relu"), textvariable = nilai2, width=20)
        self.cbfungsiaktivasi.grid(row=0, column=1, sticky="NSEW", pady=5)
        #label layer size
        self.label1 = tk.Label(self.MainFrame3, text="Layer Size", font="pristina",
width=12, relief="ridge", background="lightblue")
        self.label1.grid(row=1, column=0, padx=5, pady=5)
        #textbox layer size
        nilai3 = tk.StringVar(self.MainFrame3, value="")
```

```
        self.tblayersize = tk.Entry(self.MainFrame3, textvariable = nilai3)
        self.tblayersize.grid(row=1, column=1, sticky="NSEW", pady=5)
        #label iterasi
        self.label2 = tk.Label(self.MainFrame3, text="Max Iteration",
font="pristina", width=12, relief="ridge", background="lightblue")
        self.label2.grid(row=2, column=0, padx=5, pady=5)
        #textbox max iterasi
        nilai4 = tk.StringVar(self.MainFrame3, value="")
        self.tbmaxiterasi = tk.Entry(self.MainFrame3, textvariable = nilai4)
        self.tbmaxiterasi.grid(row=2, column=1, sticky="NSEW", pady=5)

        #frame 4
        #button proses
        self.path = tk.Button(self.MainFrame4, text="Proses", font="pristina",
height=1, width=50, background="lightblue", command=self.proses)
        self.path.grid(row=5, column=0, padx=17, pady=9)

        #frame 5
        #output
        #label hasil grayscale
        self.label4 = tk.Label(self.MainFrame5, text="Grayscale", font="pristina",
background="lightblue")
        self.label4.grid(row=0, column=0, padx=30, pady=10)
        #image hasil histogram equalization
        self.noimage1 = 'E:/noimage1.jpg'
        self.noimggray = Image.open(self.noimage1)
        self.noimggray = ImageTk.PhotoImage(self.noimggray)
        self.imghasilgray = tk.Label(self.MainFrame5, bg="lightblue",
image=self.noimggray)
        self.imghasilgray.grid(row=1, column=0, padx=45)
        #label hasil histogram equalization
        self.label4 = tk.Label(self.MainFrame5, text="Histogram Equalization",
font="pristina", background="lightblue")
        self.label4.grid(row=0, column=2, padx=30, pady=10)
        #image hasil histogram equalization
        self.noimage2 = 'E:/noimage1.jpg'
        self.noimghe = Image.open(self.noimage2)
        self.noimghe = ImageTk.PhotoImage(self.noimghe)
        self.imghasilhe = tk.Label(self.MainFrame5, bg="lightblue",
image=self.noimghe)
```

```
        self.imghasilhe.grid(row=1, column=2, padx=45)
#         self.imagehe = tk.Label(self.MainFrame5, background="lightblue")
#         self.imagehe.grid(row=1, column=1, padx=20)
        #label hasil viola jones
        self.label6 = tk.Label(self.MainFrame5, text="Viola Jones", font="pristina",
background="lightblue")
        self.label6.grid(row=0, column=3, padx=30, pady=5)
        #image hasil viola jones
        self.noimage3 = 'E:/noimage1.jpg'
        self.noimgvj = Image.open(self.noimage3)
        self.noimgvj = ImageTk.PhotoImage(self.noimgvj)
        self.imghasilvj = tk.Label(self.MainFrame5, bg="lightblue",
image=self.noimgvj)
        self.imghasilvj.grid(row=1, column=3, padx=45)
#         self.imagevj = tk.Label(self.MainFrame5, width=16,height=7,
background="lightblue")
#         self.imagevj.grid(row=1, column=2, padx=20)
        #label crop
        self.label7 = tk.Label(self.MainFrame5, text="Crop Hasil Viola Jones",
font="pristina", background="lightblue")
        self.label7.grid(row=0, column=4, padx=30, pady=5)
        #image crop
        self.noimage4 = 'E:/noimage1.jpg'
        self.noimgcrop = Image.open(self.noimage4)
        self.noimgcrop = ImageTk.PhotoImage(self.noimgcrop)
        self.imghasilcrop = tk.Label(self.MainFrame5, bg="lightblue",
image=self.noimgcrop)
        self.imghasilcrop.grid(row=1, column=4, padx=45)
#         self.imageglcm = tk.Label(self.MainFrame5, width=16,height=7,
background="lightblue")
#         self.imageglcm.grid(row=1, column=3, padx=20)
        #label output
        self.label = tk.Label(self.MainFrame7, text="Image Testing", font="pristina",
background="lightblue")
        self.label.grid(row=0, column=0, padx=100, pady=10)
        #image output
        self.noimage = 'E:/noimage1.jpg'
        self.noimghasil = Image.open(self.noimage)
        self.noimghasil = ImageTk.PhotoImage(self.noimghasil)
```

```
        self.imghasil = tk.Label(self.MainFrame7, bg="lightblue",
image=self.noimghasil)
        self.imghasil.grid(row=1, column=0, rowspan=3,columnspan=2, padx=90)
#         self.labelhe.pack(side="top")
#         self.imagetesting = tk.Label(self.MainFrame7)
#         self.imagetesting.grid(row=0, column=0, padx=80, pady=30)
        #label output
#         self.label = tk.Label(self.MainFrame7, background="lightblue")
#         self.label.grid(row=0, column=3, padx=80)
        self.label1 = tk.Label(self.MainFrame7, text="Hasil Klasifikasi Dan
Akurasi", font="pristina", background="lightblue")
        self.label1.grid(row=0, column=3, padx=80)
        #label keterangan hasil output
        self.labelket = tk.Label(self.MainFrame7, text="... ... ... ??? ... ... ...",
relief="ridge", width=30,height=2, font="pristina 20", background="lightblue")
        self.labelket.grid(row=1, column=3, padx=50)
        #label akurasi
        self.labelakurasi = tk.Label(self.MainFrame7, text="Akurasi", relief="ridge",
width=30,height=2, font="pristina 20", background="lightblue")
        self.labelakurasi.grid(row=2, column=3, padx=50)

        #frame 6
        #label kaki
        self.label = tk.Label(self.MainFrame6, text="R I Z A L  R I N A L D I  M A
P P E  |  T 3 1 1 5 1 2 6  |  F A K U L T A S  I L M U  K O M P U T E R  |  T
E K N I K  I N F O R M A T I K A  |  2 0 1 9", font="mistral")
        self.label.grid(row=2, column=2, sticky="NSEW", padx=260, pady=1)
        self.label.config(background="lightblue")
        self.MainFrame6.grid_propagate(0)


    def proses(self):
        #tampil image testing
        x = self.imagetest.get()
        img = Image.open(x)
        img = img.resize((120, 155), Image.ANTIALIAS)
        img = ImageTk.PhotoImage(img)
        self.imghasil.configure(image=img)
        self.imghasil.image = img
```

```
        a = ip.imageprocessing()
        p = self.tbdatatraining.get()
        path = a.pathlist(p)
        #print(path)
        img = a.loadimage(path)
        #print(img)
        #PRAPENGOLAHAN
        j = 1
        imgcrop = []
        pra = []
        for i in img:
            #g = cv2.cvtColor(i,cv2.COLOR_BGR2GRAY)
            pra_train, imgGray, hasil_ekual, deteksi_wajah = a.prapengolahan(i)
            imgcrop.append(pra_train)
#             cv2.imwrite(str(j)+".jpg",pra_train)
            pra.append(pra_train)
            j = j + 1


        arah = self.cbarahglcm.get()
        jarak_glcm = self.nilaijarak.get()
#         print(jarak_glcm)
        fitur_train = a.fiturekstraksi(pra, arah, jarak_glcm)
        enc = OrdinalEncoder()
#         enc1 = OrdinalEncoder()
        enc.fit(fitur_train)
        fitur_trains = enc.transform(fitur_train)
#         print(fitur_train)
        label = a.buildlabel(p)


        #TESTING
        t = self.tbdatatesting.get()
        #print(t)
        imgtest = a.loadimage([t])
        #print(imgtest)
        pra_test, imgGray,hasil_ekual, deteksi_wajah = a.prapengolahan(imgtest[0])
#         cv2.imwrite("1.jpg",pra_test)
#         cv2.imwrite("2.jpg",imgGray)
#         cv2.imwrite("3.jpg",hasil_ekual)
```

```
#       cv2.imwrite("4.jpg",deteksi_wajah)
#       print(pra_test)
        fitur_test = a.fiturekstraksi([pra_test], arah, jarak_glcm)
        enc.fit(fitur_test)
        enc.transform(fitur_test)
        fitur_tests = enc.transform(fitur_test)
#       print(fitur_test)
        faktivasi = self.cbfungsiaktivasi.get()
        lsize = literal_eval(self.tblayersize.get())
        maxiterasi  = self.tbmaxiterasi.get()
#       print(faktivasi)
#       print(lsize)
#       print(maxiterasi)
        i, akurasi = a.klasifikasi(fitur_trains, fitur_tests, label, faktivasi, lsize,
maxiterasi)
        self.labelket.configure(text = i[0])
#       self.labelakurasi.configure(text = akurasi[0])
#       print(i)
        akurat = akurasi * 100,"%"
        self.labelakurasi.configure(text = akurat)
        #gray
        imgGRAYS = Image.fromarray(imgGray)
        imgGRAYS = imgGRAYS.resize((120, 155), Image.ANTIALIAS)
        imgGRAYS = ImageTk.PhotoImage(imgGRAYS)

        self.imghasilgray.configure(image = imgGRAYS)
        self.imghasilgray.image = imgGRAYS
        #he
        imgHE = Image.fromarray(hasil_ekual)
        imgHE = imgHE.resize((120, 155), Image.ANTIALIAS)
        imgHE = ImageTk.PhotoImage(imgHE)

        self.imghasilhe.configure(image = imgHE)
        self.imghasilhe.image = imgHE
        #retangle
        imgfd = Image.fromarray(deteksi_wajah)
        imgfd = imgfd.resize((120, 155), Image.ANTIALIAS)
        imgfd = ImageTk.PhotoImage(imgfd)

        self.imghasilvj.configure(image = imgfd)
```

```
        self.imghasilvj.image = imgfd
        #vj
        imgv = Image.fromarray(pra_test)
        imgv = imgv.resize((120, 155), Image.ANTIALIAS)
        imgv = ImageTk.PhotoImage(imgv)

        self.imghasilcrop.configure(image = imgv)
        self.imghasilcrop.image = imgv

#       akurasi = Image.fromarray(akurasi)
#       akurasi = akurasi.resize((120, 155), Image.ANTIALIAS)
#       akurasi = ImageTk.PhotoImage(imgv)
#
#       self.imghasilcrop.configure(image = imgv)
#       self.imghasilcrop.image = imgv

def main():
    root = tk.Tk()
    root.geometry("1024x520")
    root.configure(background='lightblue')
    root.state("zoomed")
    app = Main(root)
    app.parent.title("Klasifikasi Usia Pada Citra Wajah")
    root.mainloop()

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

2. KODE PROGRAM IMAGE PROCESSING

```
# -*- coding: utf-8 -*-
"""
Created on Thu Mar  7 12:07:42 2019

@author: Notebook
"""

import cv2
#import numpy as np
from imutils import paths
```

```
from skimage import feature
from sklearn.neural_network import MLPClassifier as ann
import os
from scipy.stats import entropy

import warnings

with warnings.catch_warnings():
    warnings.filterwarnings("ignore", category=DeprecationWarning)

class imageprocessing():
#    def __init__(self, pathtrain, pathtest):
#        self.pathtrain = pathtrain
#        self.pathtest = pathtest

        #membuat matrix
    def pathlist(self, path):
        matrixpath = []
        for pathimage in paths.list_images(path):
            matrixpath.append(pathimage)
        return matrixpath

        #membaca gambar
    def loadimage(self, imtrain):
        kumpulangambar = []
        hasil_pathlist = imtrain
        for image in hasil_pathlist:
            img = cv2.imread(image)
            kumpulangambar.append(img)
        return kumpulangambar

        #histogram eaualization
    def prapengolahan(self, image):
        imgGray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        resize_img = cv2.resize(imgGray,(120,155))
        hasil_ekual = cv2.equalizeHist(resize_img)
        img_crop = hasil_ekual.copy()
        deteksi_wajah = hasil_ekual.copy()
```

```
        #viola jones
        CASCADE_PATH = "haarcascade_frontalface_default.xml"
        face_cascade = cv2.CascadeClassifier(CASCADE_PATH)
        faces = face_cascade.detectMultiScale(hasil_ekual, 1.1, 3, minSize=(100,
100))
#         img_crop = hasil_ekual.copy()
        if (faces is None):
#             print('Tidak Ada Wajah Terdeteksi')
            return 0
        for (x, y, w, h) in faces:
            cv2.rectangle(deteksi_wajah,(x,y),(x+w,y+h),(0,0,255),2)
            r = max(w, h) / 2
            centerx = x + w / 2
            centery = y + h / 2
            nx = int(centerx - r)
            ny = int(centery - r)
            nr = int(r * 2)

            img_crop = img_crop[ny:ny+nr, nx:nx+nr]

        return img_crop,imgGray,hasil_ekual, deteksi_wajah

    def fiturekstraksi(self,imset, arah, jarak_glcm):
#         print(jarak_glcm)
        imfitur = []
        for image in imset:
            #matrixglcm = feature.graycomatrix(image, [10], [45], 255)
            matrixglcm = feature.greycomatrix(image, [int(jarak_glcm)], [int(arah)],
256)
            cont = feature.greycoprops(matrixglcm, 'contrast')[0, 0]
            dis = feature.greycoprops(matrixglcm, 'dissimilarity')[0, 0]
#             homo = feature.greycoprops(matrixglcm, 'homogeneity')[0, 0]
#             asm = feature.greycoprops(matrixglcm, 'ASM')[0, 0]
            energi = feature.greycoprops(matrixglcm, 'energy')[0, 0]
            corr = feature.greycoprops(matrixglcm, 'correlation')[0, 0]
            ent = entropy(matrixglcm.flatten())
#             imfitur.extend([[cont, dis, homo, asm, energi, corr, ent]])
            imfitur.extend([[cont, dis, energi, corr, ent]])
        return imfitur
```

```
    def buildlabel(self, pathtrain):
        label = []
        pathnya = self.pathlist(pathtrain)
        for image in pathnya:
            label.append(os.path.split(os.path.dirname(image))[-1])
        return label

    def klasifikasi(self, fiturtraining, fiturtesting, label, faktivasi, lsize, maxiterasi):
#         impath = self.pathlist(pathtrain)
#         hasil_prapengolahan = []
#         kumpulangambar = self.loadimage(impath)
#         for img in kumpulangambar:
#             hasil_prapengolahan.append(self.prapengolahan(img))
#         imfitur = self.fiturekstraksi(hasil_prapengolahan, 1, 0)
#         label = self.buildlabel(pathtrain)

        klasifikasi = ann(activation = faktivasi, hidden_layer_sizes = (lsize),
max_iter = int(maxiterasi))

        klasifikasi.fit(fiturtraining, label)
        klasifikasihasil = klasifikasi.predict(fiturtesting)
        akurasi = klasifikasi.score(fiturtraining, label)
#         print(akurasi)
        return klasifikasihasil, akurasi
        #hasiltraining = klasifikasi.coefs_
        #klasifikasi.score
```

RIWAYAT HIDUP PENELITI

Nama	: RIZAL RINALDI MAPPE
Tempat, Tgl Lahir	: Hutabohu, 13 maret 1996
Pekerjaan	: Mahasiswa
Email	: rizalrinaldymappe@gmail.com

Riwayat Pendidikan :

1. Lulusan Sekolah Dasar Negeri II Hutabohu Tahun 2008
2. Lulusan Sekolah Menengah Pertama Negeri 4 Limboto Barat Tahun 2011
3. Lulusan Sekolah Madrasah Aliya Nuruttaqwa Limboto Tahun 2014