**­­­­­­DETEKSI POTENSI PENYAKIT JANTUNG BERDASARKAN IRIDOLOGI**

****

**TUGAS AKHIR**

*Disusun dalam rangka memenuhi salah satu persyaratan*

*Untuk menyelesaikan program Strata-1 Departemen Teknik Informatika*

*Fakultas Teknik Universitas Hasanuddin*

*Makassar*

**Disusun Oleh:**

|  |
| --- |
| **KEVIN CHRISTIAN HALIM**  **D421 15 313** |

**DEPARTEMEN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS HASANUDDIN**

**MAKASSAR**

**2019**

# KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan yang Maha Esa karena berkat rahmat dan karunia-Nya sehingga tugas akhir yang berjudul “DETEKSI POTENSI PENYAKIT JANTUNG BERDASARKAN IRIDOLOGI” ini dapat diselesaikan sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Penulis menyadari bahwa dalam penyusunan dan penulisan laporan tugas akhir ini tidak lepas dari bantuan, bimbingan serta dukungan dari berbagai pihak, dari masa perkuliahan sampai dengan masa penyusunan tugas akhir. Oleh karena itu, penulis dengan senang hati menyampaikan terima kasih kepada:

1. Kedua Orang tua penulis, Bapak Rudy Halim dan Ibu Chieko Thevani T yang selalu memberikan dukungan, doa, dan semangat serta selalu sabar dalam mendidik penulis sejak kecil;
2. Ibu Dr. Ir. Ingrid Nurtanio, M.T. selaku pembimbing I dan Bapak Dr. Indrabayu S.T., M.T., M.Bus.Sys selaku pembimbing II yang selalu menyediakan waktu, tenaga, pikiran dan perhatian yang luar biasa untuk mengarahkan penulis dalam penyusunan tugas akhir;
3. Ibu Dr. Eng. Intan Sari Areni, S.T., M.T., dan Ibu Anugrahyani Bustamin., S.T., M.T., yang senantiasa memberikan nasehat, masukan, serta perhatian yang luar biasa kepada penulis dalam penyusunan tugas akhir;
4. Bapak Dr. Amil Ahmad Ilham, ST., M.IT., selaku Ketua Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas bimbingannya selama masa perkuliahan penulis;
5. Para sahabat, teman-teman dan kakak-kakak AIMP Research Group FT-UH yang telah memberikan begitu banyak bantuan selama penelitian, pengambilan data dan diskusi progress penyusunan tugas akhir;
6. Teman-teman Hypervisor FT-UH atas dukungan dan semangat yang diberikan selama ini;
7. Segenap Staf dan Dosen Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah membantu penulis.
8. Orang-orang berpengaruh lainnya yang tanpa sadar telah menjadi inspirasi penulis.

Akhir kata, penulis berharap semoga Tuhan berkenan membalas segala kebaikan dari semua pihak yang telah banyak membantu. Semoga Tugas Akhir ini dapat memberikan manfaat bagi pengembangan ilmu.

Makassar, September 2019

Penulis

# ABSTRAK

Iridologi merupakan metode yang digunakan untuk *screening* atau pemeriksaan kesehatan dini melalui pemeriksaan iris mata dan telah dipraktikkan selama ratusan tahun untuk memeriksa berbagai kondisi kesehatan dalam tubuh termasuk penyakit jantung. Penyakit jantung merupakan salah satu penyakit yang menjadi penyebab utama kematian di Indonesia. Hasil riset kesehatan dasar pada tahun 2007-2013 menyatakan bahwa Sulawesi Selatan dan Bangka Belitung memiliki potensi besar terhadap penyakit jantung. Oleh sebab itu, dibutuhkan metode yang cepat dalam melakukan pemeriksaan kesehatan jantung untuk mengurangi risiko kematian akibat serangan jantung. Dengan memanfaatkan iridologi dan teknologi informasi di bidang visi komputer, dapat dibuat sebuah sistem untuk memprediksi kondisi kesehatan jantung berdasarkan analisis iris mata pasien. Data yang digunakan dalam penelitian ini sebanyak 110 citra yang terdiri dari 55 data normal dan 55 data berpenyakit jantung, dan dibagi menjadi 88 data latih dan 22 data uji. Sedangkan metode yang digunakan adalah *canny edge detection* dan *Principal Component Analysis (PCA)* untuk pengambilan fitur iris mata, dan jaringan saraf tiruan *backpropagation* untuk model prediksi. Pada penelitian ini dihasilkan akurasi sebesar **95.455%** untuk percobaan parameter sigma *canny* 0.3, ukuran PC 50, dan jumlah *hidden neuron* sebanyak 40 dan 50.

**Kata Kunci:** iridologi, penyakit jantung, *canny edge detection*, *principal component analysis*, *backpropagation*

# DAFTAR ISI

[KATA PENGANTAR i](#_Toc19268482)

[ABSTRAK iii](#_Toc19268483)

[DAFTAR ISI iv](#_Toc19268484)

[DAFTAR GAMBAR vii](#_Toc19268485)

[DAFTAR TABEL ix](#_Toc19268486)

[BAB I PENDAHULUAN 1](#_Toc19268487)

[1.1. Latar Belakang 1](#_Toc19268488)

[1.2. Rumusan Masalah 2](#_Toc19268489)

[1.3. Tujuan Penelitian 3](#_Toc19268490)

[1.4. Manfaat Penelitian 3](#_Toc19268491)

[1.5. Batasan Masalah 4](#_Toc19268492)

[1.6. Sistematika Penulisan 4](#_Toc19268493)

[BAB II TINJAUAN PUSTAKA 6](#_Toc19268494)

[2.1. Iridologi 6](#_Toc19268495)

[2.2. Visi Komputer 8](#_Toc19268496)

[2.3. Dimensionality Reduction 19](#_Toc19268497)

[2.4. Principal Component Analysis (PCA) 22](#_Toc19268498)

[2.5. Jaringan Saraf Tiruan 26](#_Toc19268499)

[2.6. Algoritma Backpropagation 30](#_Toc19268500)

[BAB III METODOLOGI PENELITIAN 33](#_Toc19268501)

[3.1. Tahapan Penelitian 33](#_Toc19268502)

[3.2. Waktu dan Lokasi Penelitian 35](#_Toc19268503)

[3.3. Instrumen Penelitian 35](#_Toc19268504)

[3.4. Akuisisi Data 35](#_Toc19268505)

[3.5. Perancangan dan Implementasi Sistem 36](#_Toc19268506)

[3.5.1. Input Citra 37](#_Toc19268507)

[*3.5.2.* *Grayscaling* 40](#_Toc19268508)

[*3.5.3.* *Histogram Equalization* 41](#_Toc19268509)

[*3.5.4.* *Deblurring* 42](#_Toc19268510)

[*3.5.5.* *Pupil Masking* 43](#_Toc19268511)

[*3.5.6.* *Region of Interest (ROI) Cropping* 45](#_Toc19268512)

[*3.5.7.* *Canny Edge Detection* 49](#_Toc19268513)

[*3.5.8.* Seleksi Fitur *Principal Component Analysis (PCA)* 53](#_Toc19268514)

[*3.5.9.* Model Jaringan *Backpropagation* 59](#_Toc19268515)

[3.6. Analisis Kerja Sistem 64](#_Toc19268516)

[BAB IV HASIL DAN PEMBAHASAN 65](#_Toc19268517)

[4.1. Hasil Penelitian 65](#_Toc19268518)

[4.2. Pembahasan 71](#_Toc19268519)

[4.2.1. Pengaruh nilai sigma terhadap hasil kinerja sistem 71](#_Toc19268520)

[4.2.2. Pengaruh arsitektur jaringan terhadap hasil kinerja sistem 73](#_Toc19268521)

[4.2.3. Pengaruh nilai batas eror terhadap hasil kinerja sistem 75](#_Toc19268522)

[BAB V PENUTUP 76](#_Toc19268523)

[5.1. Kesimpulan 76](#_Toc19268524)

[5.2. Saran 77](#_Toc19268525)

[DAFTAR PUSTAKA 78](#_Toc19268526)

[LAMPIRAN 80](#_Toc19268527)

[Lampiran 1. Confusion Matrix untuk batas eror 0.1 81](#_Toc19268528)

[Lampiran 2. Confusion Matrix untuk batas eror 0.01 86](#_Toc19268529)

DAFTAR GAMBAR  
[Gambar 2.1. Diagram iridologi 8](#_Toc19268071)

[Gambar 2.2. Ekualisasi histogram 14](#_Toc19268072)

[Gambar 2.3. Citra *threshold* 17](#_Toc19268073)

[Gambar 2.4. Grafik hubungan dimensi dan performa 19](#_Toc19268074)

[Gambar 2.5. Komponen pada PCA 25](#_Toc19268075)

[Gambar 2.6. *Single layer network* 29](#_Toc19268076)

[Gambar 2.7. *Multi-layer network* 29](#_Toc19268077)

[Gambar 3.1. Tahapan penelitian 33](#_Toc19268078)

[Gambar 3.2. Garis besar sistem 36](#_Toc19268079)

[Gambar 3.3. Flowchart sistem 38](#_Toc19268080)

[Gambar 3.4. (a) Citra iris berpenyakit jantung (b) Citra subjek normal 39](#_Toc19268081)

[Gambar 3.5. Citra grayscale 41](#_Toc19268082)

[Gambar 3.6. (a) Histogram awal, (b) Histogram hasil ekualisasi 42](#_Toc19268083)

[Gambar 3.7. Contoh median filtering 43](#_Toc19268084)

[Gambar 3.8. Citra biner 44](#_Toc19268085)

[Gambar 3.9. Citra hasil masking 45](#_Toc19268086)

[Gambar 3.10. Citra kartesian 46](#_Toc19268087)

[Gambar 3.11. Citra hasil rotasi 46](#_Toc19268088)

[Gambar 3.12. (a) Target, (b) Hasil crop target 47](#_Toc19268089)

[Gambar 3.13. (a) Area Pupil, (b) Hasil crop pupil 48](#_Toc19268090)

[Gambar 3.14. (a) ROI, (b) Hasil crop ROI, (c) Hasil resize ROI 48](#_Toc19268091)

[Gambar 3.15. *Non-maximum supression* 51](#_Toc19268092)

[Gambar 3.16. Hasil deteksi tepi *canny* 53](#_Toc19268093)

[Gambar 3.17. Confusion matrix 64](#_Toc19268094)

[Gambar 4.1. Hasil deteksi tepi *canny* untuk masing-masing nilai sigma 73](#_Toc19268095)

[Gambar 4.2. Diagram hasil kinerja sistem untuk sigma canny 0.3 dan batas eror 0.01 74](#_Toc19268096)

DAFTAR TABEL  
[Tabel 3.1. Contoh dataframe latih 53](#_Toc19268194)

[Tabel 3.2. Contoh dataframe uji 54](#_Toc19268195)

[Tabel 3.3. Contoh dataframe latih hasil normalisasi 54](#_Toc19268196)

[Tabel 3.4. Contoh dataframe uji hasil normalisasi 55](#_Toc19268197)

[Tabel 3.5. Contoh dataframe latih hasil dekomposisi PCA 58](#_Toc19268198)

[Tabel 3.6. Contoh dataframe uji hasil dekomposisi PCA 58](#_Toc19268199)

[Tabel 4.1. Tabel akurasi untuk sigma canny 0.1 dan batas eror 0.1 66](#_Toc19268200)

[Tabel 4.2. Tabel akurasi untuk sigma canny 0.1 dan batas eror 0.01 66](#_Toc19268201)

[Tabel 4.3. Tabel akurasi untuk sigma canny 0.2 dan batas eror 0.1 67](#_Toc19268202)

[Tabel 4.4. Tabel akurasi untuk sigma canny 0.2 dan batas eror 0.01 67](#_Toc19268203)

[Tabel 4.5. Tabel akurasi untuk sigma canny 0.3 dan batas eror 0.1 68](#_Toc19268204)

[Tabel 4.6. Tabel akurasi untuk sigma canny 0.3 dan batas eror 0.01 69](#_Toc19268205)

[Tabel 4.7. Tabel akurasi untuk sigma canny 0.4 dan batas eror 0.1 69](#_Toc19268206)

[Tabel 4.8. Tabel akurasi untuk sigma canny 0.4 dan batas eror 0.01 70](#_Toc19268207)

[Tabel 4.9. Tabel akurasi untuk sigma canny 0.5 dan batas eror 0.1 70](#_Toc19268208)

[Tabel 4.10. Tabel akurasi untuk sigma canny 0.5 dan batas eror 0.01 71](#_Toc19268209)

[Tabel 4.11. Nilai low dan high threshold untuk gambar (3.14.(c)) 72](#_Toc19268210)

[Tabel 4.12. Hasil akurasi untuk setiap nilai batas eror 75](#_Toc19268211)

# BAB I PENDAHULUAN

## Latar Belakang

Iridologi merupakan metode untuk menunjukkan potensi gangguan kondisi kesehatan dengan memeriksa area berwarna pada mata, atau disebut juga iris. Iridologi dipopulerkan oleh Dr. Ignatz Von Peczely pada tahun 1881 dalam bukunya “Discoveries in the Field of Natural Science and Medicine”. Metode ini telah dipraktikkan dalam berbagai bentuk selama beratus-ratus tahun, iridologi modern memanfaatkan pencitraan digital dan teknik manual untuk menentukan kesehatan pasien secara keseluruhan.

Penelitian di Rusia yang melibatkan 800.000 pasien menemukan bahwa iridologi dapat menjadi salah satu bentuk diagnosa yang efektif yang berlaku pada 85% kasus. Iridologi dapat digunakan untuk mengidentifikasi penyakit yang mungkin tidak memiliki gejala, seperti hipertensi, hipertiroid, gangguan liver, masalah kantung empedu, infeksi, pelemahan sistem imun, masalah pencernaan, alergi, kesehatan limfatik, hingga kolesterol tinggi (10 Conditions That Iridology Can Detect | Better Living Wellness Clinic, 2017). Meskipun iridologi tidak dapat dengan pasti melakukan diagnosa, tetapi iridologi sangat berguna terutama dalam memahami area tubuh yang memiliki masalah atau membutuhkan perhatian khusus untuk menghindari penyakit yang lebih berbahaya.

Penyakit kardiovaskuler yang berhubungan dengan jantung dan pembuluh darah menduduki peringkat pertama sebagai penyakit yang menyebabkan kematian. Berdasarkan data WHO, 37% kematian di Indonesia disebabkan karena penyakit jantung dan pembuluh darah. Hasil riset kesehatan dasar Indonesia dari tahun 2007-2013 menunjukkan bahwa terdapat dua provinsi yang tercatat memiliki risiko tinggi terhadap penyakit jantung yaitu Sulawesi Selatan dan Bangka Belitung. (CNNIndonesia, 2018)

Sebagai salah satu penyakit dengan biaya paling besar dan banyak menyebabkan kematian, maka diperlukan pengidentifikasian potensi dini penyakit secara cepat dan tepat. Dengan menggabungkan metode iridologi dan perkembangan teknologi informasi terutama dalam bidang visi komputer dan *Machine Learning*, maka dapat dibuat sebuah sistem yang dapat dengan cepat mengidentifikasi dan memprediksi adanya potensi penyakit jantung yang diderita sehingga risiko kematian dan mahalnya biaya akibat penyakit tersebut dapat diminimalkan. Atas dasar latar belakang tersebut, maka diangkatlah topik penelitian dengan judul “Deteksi Potensi Penyakit Jantung Berdasarkan Iridologi”.

## Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan, maka rumusan masalah pada penelitian ini antara lain:

1. Bagaimana cara mengekstraksi fitur pada citra iris mata?
2. Bagaimana memprediksi potensi penyakit jantung menggunakan fitur iris mata berdasarkan iridologi?
3. Bagaimana unjuk kerja sistem yang dibuat?

## Tujuan Penelitian

Tujuan dari penelitian ini antara lain:

1. Mengekstraksi fitur dari citra iris mata sesuai dengan letak organ jantung berdasarkan iridologi.
2. Membuat sistem yang dapat memprediksi adanya potensi penyakit jantung berdasarkan fitur iris mata.
3. Mengetahui keakuratan dari sistem yang dibuat.

## Manfaat Penelitian

Penelitian ini diharapkan dapat memberi manfaat sebagai berikut:

1. Bagi masyarakat

Dengan pengembangan lebih lanjut, diharapkan dapat membantu masyarakat untuk mengetahui lebih cepat terkait adanya potensi penyakit jantung, sehingga dapat mengurangi biaya serta resiko kematian akibat keterlambatan identifikasi dan penanganan penyakit.

1. Bagi instansi kesehatan

Hasil dari sistem diharapkan mampu menjadi acuan bagi instansi kesehatan dalam menangani keluhan penyakit pasien, sehingga pasien dapat ditangani dengan lebih cepat.

1. Bagi peneliti

Sistem diharapkan mampu menjadi salah satu acuan untuk pengembangan dan penyempurnaan sistem serupa, baik dalam hal variasi penyakit maupun keakuratan sistem.

## Batasan Masalah

Batasan masalah yang ditentukan dalam sistem ini adalah:

1. Potensi penyakit yang akan diprediksi adalah penyakit jantung.
2. Data yang digunakan harus memperoleh pencahayaan yang cukup untuk menampilkan serat saraf pada iris.
3. Data yang digunakan berupa citra iris berwarna dengan resolusi 500\*500 piksel berformat .JPG.
4. Data yang digunakan telah diisolasi pada bagian iris dan pupilnya dihitamkan.
5. Masukan sistem berupa citra iris mata.
6. Keluaran sistem berupa prediksi potensi penyakit jantung.

## Sistematika Penulisan

Untuk memberikan gambaran singkat mengenai isi tulisan ini, maka akan diuraikan beberapa tahapan dari penulisan secara sistematis, yaitu:

**BAB I PENDAHULUAN**

Bab ini menguraikan secara umum mengenai hal yang menyangkut latar belakang, perumusan masalah, batasan masalah, tujuan, dan manfaat penelitian.

**BAB II TINJAUAN PUSTAKA**

Bab ini berisi teori-teori terkait hal-hal yang mendasari dan berhubungan dengan penelitian, termasuk di dalamnya iridologi, visi komputer, dan metode-metode yang digunakan dalam penelitian.

**BAB III METODOLOGI PENELITIAN**

Bab ini berisi tentang perencanaan dan proses penerapan algoritma dan metode-metode dalam pengolahan data, mulai dari preprocessing hingga menghasilkan prediksi.

**BAB IV HASIL DAN PEMBAHASAN**

Bab ini berisi tentang hasil penelitian dan pembahasan terkait pengolahan data yang telah dilakukan yang disertai dengan tabel hasil penelitian.

**BAB V PENUTUP**

Bab ini berisi tentang kesimpulan yang didapatkan berdasarkan hasil penelitian yang telah dilakukan serta saran untuk pengembangan sistem yang lebih lanjut.

# BAB II TINJAUAN PUSTAKA

## Iridologi

Iridologi merupakan ilmu dan praktik yang mengungkapkan peradangan, letak radang, dan pada tahap apa manifestasi peradangan terjadi dalam tubuh. Iridologi dapat mengungkapkan keadaan tubuh, kelemahan bawaan, tingkat kesehatan, dan perubahan yang terjadi pada tubuh seseorang berdasarkan cara hidupnya.

Iridologi bekerja dengan menganalisis struktur halus dari iris mata. Iris merupakan bagian dari mata yang membawa warna. Iris merepresentasikan sebuah sistem komunikasi yang mampu menangani kuantitas informasi yang sangat besar. Informasi tersebut dapat ditunjukkan dalam karakteristik dari serat iris seseorang. Serat iris tersebut berjumlah sangat banyak hingga tidak dapat dihitung. Kombinasi dari berbagai karakteristik serat iris ini dapat menghasilkan variasi yang tidak terhingga (Jensen, 2011).

1. Iris

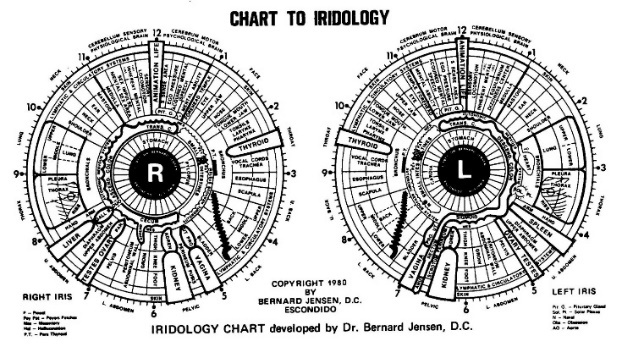
Iris mata merupakan jaringan paling kompleks dari tubuh yang dapat dilihat dari luar tubuh. Iris adalah perpanjangan dari otak, yang terhubung dengan ratusan ribu ujung saraf, pembuluh darah mikroskopis, otot, dan jaringan lainnya. Iris terhubung dengan semua organ dan jaringan tubuh melalui otak dan sistem saraf. Serabut saraf menerima impuls melalui koneksinya dengan saraf optik, serat optik, dan sumsum tulang belakang. Mereka dibentuk secara *embriologis* dari jaringan *mesoderm* dan *neuroectoderm*. Kedua sistem saraf simpatik dan para-simpatik terdapat pada iris mata.

Dengan cara ini, alam menyediakan layar miniatur untuk menunjukkan bagian tubuh terdalam dengan respons refleks saraf. Ini menunjukkan bahwa mata manusia bekerja secara dua arah, bukan hanya memungkinkan untuk membawa gambar dari luar tubuh ke dalam, tetapi juga menunjukkan gambaran tentang apa yang terjadi dalam tubuh ke luar.

1. Diagram Iridologi

Diagram yang dikembangkan oleh Dr. Bernard Jensen merupakan hasil dari penelitian dan pembelajarannya selama 50 tahun. Menghabiskan bertahun-tahun sebagai mahasiswa master iridologi, ia telah mengedepankan dan memasukkan aspek-aspek dan detail yang terbukti akurat melalui verifikasi dalam praktik klinis (Jensen, 2011).

Setiap organ diidentifikasi dan yang terpenting diuraikan sehingga dapat dengan mudah ditemukan. Diagram ini merepresentasikan susunan iris ke dalam lokasi yang sesuai seperti yang ditemukan pada tubuh yang teratur. Diagram iridologi ditunjukkan seperti pada gambar 2.1. berikut.



Gambar 2.1. Diagram iridologi

1. Organ Jantung Berdasarkan Teori Iridologi

Berdasarkan diagram iridologi, organ jantung terletak pada mata kiri pasien dengan letak spesifik di antara arah jam 2 hingga jam 3 dan terletak antara ujung pupil hingga tengah cincin iris. Ciri yang terlihat pada pasien yang berpotensi terkena gangguan jantung adalah berupa noda seperti lubang atau garis berwarna gelap/kehitaman.

## Visi Komputer

Visi komputer merupakan analisis otomatis terhadap gambar dan video yang dilakukan oleh komputer dalam rangka memperoleh pemahaman terhadap dunia. Visi komputer terinspirasi dari kemampuan sistem penglihatan manusia, ketika pertama kali diperkenalkan pada tahun 1960 dan 1970-an, visi komputer dianggap sebagai masalah yang cukup mudah dipecahkan. Namun, hal itu dianggap mudah karena manusia memiliki sistem visualnya sendiri yang membuat tugas tersebut tampak mudah. Faktanya, sistem visual manusia sangat kompleks dan bahkan diperkirakan menggunakan fungsi otak manusia sebanyak 25% hingga lebih dari 50% (Dawson-Howe, 2014).

Visi komputer terbagi atas dua bagian besar, yaitu pengolahan citra dan pengenalan pola.

1. Pengolahan Citra
   1. Citra

Citra merupakan pusat dari visi komputer, karena citra merupakan representasi dari apa yang diperoleh dari perangkat pencitraan (seperti kamera). Citra menyediakan representasi dari penampilan visual dari sebuah kejadian, yang dapat diproses untuk meningkatkan fitur tertentu yang menarik, sebelum memperoleh abstraksi informasi.

Untuk memproses sebuah citra pada komputer digital, maka citra harus:

* disampel ke dalam bentuk matriks, dan
* dikuantisasi agar setiap elemen dalam matriks diberikan nilai integer. Dengan kata lain, citra dibagi ke dalam beberapa interval angka (k) di mana k = 256.

Citra digital mengandung sejumlah elemen-elemen dasar. Elemen-elemen dasar yang paling penting diuraikan sebagai berikut (Darmawan & Dwi Saptia, 2010):

* Kecerahan (*Brightness*) merupakan intensitas cahaya rata-rata dari suatu area yang melingkupinya.
* Kontras (*Contrast*) merupakan sebaran terang (*lightness*) dan gelap (*darkness*) di dalam sebuah citra. Citra dengan kontras rendah adalah citra dengan komposisi sebagian besar terang atau sebagian besar gelap. Citra dengan kontras yang baik adalah citra yang komposisi gelap dan terangnya tersebar merata.
* Kontur (*Contour*) merupakan keadaan yang ditimbulkan oleh perubahan intensitas pada pixel-pixel tetangga, sehingga dapat dideteksi tepi objek dalam citra.
* Warna (*Color*) merupakan 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 adalah *Red* (R), *Green* (G), dan *Blue* (B).
* Bentuk (*Shape*) merupakan properti intrinsik dari objek tiga dimensi, dengan pengertian bahwa bentuk merupakan properti intrinsik utama untuk visual manusia. Umumnya citra yang dibentuk manusia merupakan 2D, sedangkan objek yang dilihat adalah 3D.
* Tekstur (*Texture*) merupakan distribusi spasial dari derajat keabuan di dalam sekumpulan pixel-pixel yang bertetangga.
  1. Perbaikan Kualitas Citra

Dalam meningkatkan mutu citra ada beberapa teknik yang digunakan antara lain *image enhancement*, berupa proses perbaikan citra dengan meningkatkan kualitas citra baik kontras maupun kecerahan. Kecerahan suatu citra sangat diperlukan agar seseorang dapat melihat dan mengamati citra tersebut dengan jelas tanpa adanya gangguan. Tingkat ketajaman dan kecerahan citra merupakan salah satu parameter yang utama dalam pemrosesan citra digital. Tetapi sering kali citra yang dimiliki mengalami kerusakan (degradasi), misalnya citra mengalami derau (*noise*), citra terlalu gelap atau terlalu terang, citra kurang tajam atau *blur*, dan sebagainya (Aripin, Ginting, & Silalahi, 2017).

* 1. Citra *Grayscale*

Sesuai dengan namanya, jenis citra ini memiliki gradasi warna hitam dan putih, yang menghasilkan efek warna abu-abu. Intensitas berkisar antara 0 sampai dengan 255. Nilai 0 menyatakan hitam dan 255 menyatakan putih. Citra RGB dengan matriks penyusun citra yang sebelumnya 3 matriks akan berubah menjadi 1 matriks saja.

Citra *grayscale* merupakan citra yang nilai intensitas pixelnya didasarkan pada derajat keabuan. Pada citra *grayscale* 8-bit, derajat warna hitam sampai dengan putih dibagi ke dalam 256 derajat keabuan di mana warna hitam sempurna direpresentasikan dengan nilai 0 dan putih sempurna dengan nilai 255. Citra RGB dapat dikonversi menjadi citra *grayscale* sehingga dihasilkan hanya satu kanal warna. Terdapat tiga metode yang digunakan untuk mengonversi citra RGB ke *grayscale*:

1. *Lightness Method*

Metode ini mengambil rata-rata pada dari intensitas warna yang paling mencolok dan paling tidak mencolok dengan persamaan (2.1) berikut:

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

1. *Average Method*

Metode rerata (average method) merupakan metode yang paling sederhana. Nilai piksel dari tiap-tiap kanal diambil dan dirata-ratakan. Metode ini dituliskan dalam persamaan (2.2) berikut:

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

1. *Luminosity Method*

Metode ini memberikan nilai beban-beban tertentu pada kanal-kanal tertentu sesuai dengan kebutuhan pengguna. Manusia lebih sensitif terhadap warna hijau dibandingkan warna-warna lainnya, sehingga warna hijau biasanya memiliki bobot yang paling besar. Adapun salah satu contoh persamaan dari metode ini adalah persamaan (2.3):

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

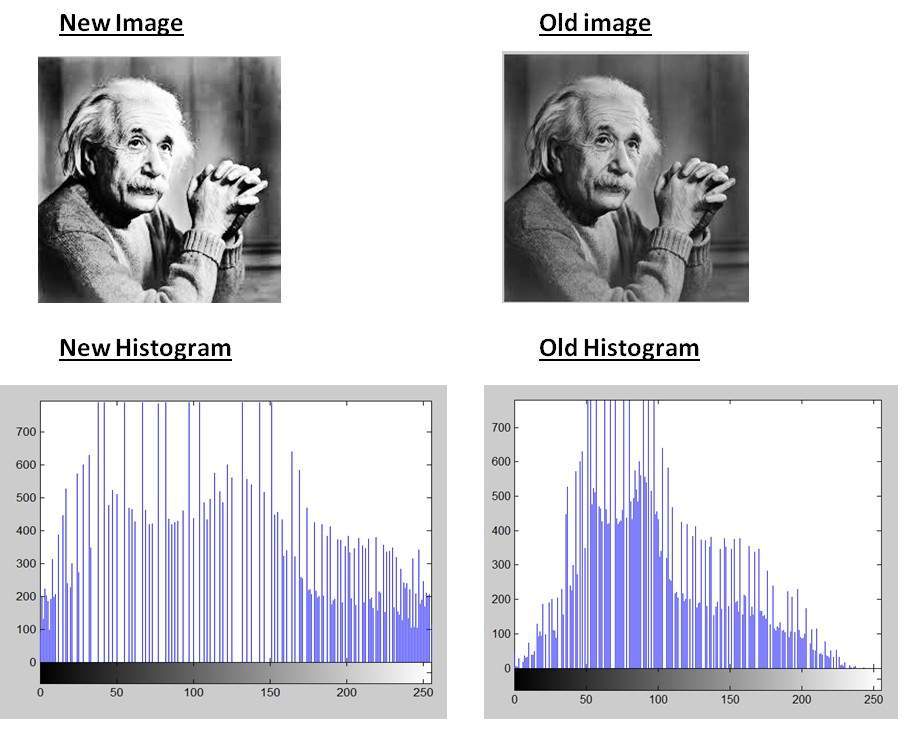
* 1. Histogram Citra

Histogram citra merupakan abstraksi dari citra di mana nilai frekuensi tiap citra (kecerahan/intensitas) ditentukan. Pada citra *grayscale* terdapat 256 intensitas skala keabuan yaitu bernilai 0 hingga 255. Nilai intensitas tersebut menunjukkan tingkat keabuan yang dimiliki masing-masing pixel dalam suatu citra. Histogram mengandung informasi mengenai citra dan informasi tersebut bersifat independen terhadap posisi dan orientasi objek dalam citra. Dalam beberapa kasus, histogram atau informasi yang diturunkan dari histogram (seperti rata-rata intensitas dan standar deviasi) bisa digunakan untuk melakukan klasifikasi. Namun, perlu diperhatikan bahwa histogram citra tidak unik dan memungkinkan banyak citra memiliki histogram yang serupa.

Citra terkadang sulit diamati oleh manusia dikarenakan keadaan citra yang terlalu gelap atau terlalu terang. Manusia dapat membedakan antara 700 hingga 900 tingkat keabuan dalam kondisi penglihatan optimal. Meskipun demikian, manusia bisa lebih mudah membedakan perbedaan intensitas yang lebih besar, jadi jika distribusi derajat keabuan dalam citra ditingkatkan, maka akan meningkatkan pemahaman observasi manusia.

Salah satu teknik untuk meningkatkan distribusi derajat keabuan dalam citra adalah dengan ekualisasi histogram (*histogram equalization*). Teknik ini berusaha mendistribusikan derajat keabuan dalam citra secara merata, sehingga menghasilkan histogram yang merata. Ekualisasi histogram bukan merupakan solusi yang baik (kecuali beberapa derajat keabuan yang umum pada citra masukan dipetakan ke beberapa skala keabuan berbeda pada citra keluaran), karena biasa menghasilkan histogram yang memiliki derajat keabuan yang tidak memiliki pasangan pixel, yang diselingi skala keabuan dengan nilai tinggi (Dawson-Howe, 2014).

Contoh citra hasil ekualisasi histogram ditunjukkan pada gambar 2.2. berikut.



Gambar 2.2. Ekualisasi histogram

* 1. *Smoothing*

Menghilangkan atau mengurangi derau pada gambar dapat dilakukan dengan berbagai metode. Metode paling umum yang digunakan untuk menghilangkan derau adalah *linear smoothing transformation* misalnya *Image Averaging* dan *Gaussian Smoothing*. Penekanan derau menggunakan *linear smoothing transformation* umumnya menghasilkan penghalusan pada tepi yang tajam. Selain transformasi linear, terdapat juga beberapa metode transformasi non-linear seperti Median Filter (Dawson-Howe, 2014).

1. *Image Averaging*

Jika terdapat beberapa citra dengan kondisi yang sama persis, maka citra-citra tersebut dapat dirata-ratakan untuk mengurangi derau. Rata-rata citra dapat diekspresikan seperti persamaan (2.4) berikut:

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

1. *Local Averaging*

Jika hanya terdapat satu citra, perhitungan rata-rata tetap dapat dilakukan namun dalam lingkungan lokal. Bentuk paling sederhana dari teknik ini adalah menghitung rata-rata suatu titik dalam citra dengan 8 titik di sekitarnya, untuk setiap titik yang ada dalam citra.

*Local Averaging* merupakan proses perhitungan rata-rata lokal di mana setiap titik pada citra diberikan beban (*weight*) yang sama. Perubahan nilai beban pada saat perhitungan rata-rata juga dimungkinkan, hal ini sering juga disebut *Gaussian Distribution* dan perhitungan rata-rata menggunakan pembebanan ini disebut juga dengan *Gaussian Smoothing*.

Pengaplikasian filter seperti ini biasanya dilakukan dengan menggunakan teknik konvolusi, di mana *mask* konvolusi (h(i,j) merepresentasikan *weights* dari filter *smoothing*) akan dikonvolusi dengan citra masukan f(i,j) untuk menghasilkan citra keluaran f’(i,j). Konvolusi ini dinotasikan seperti persamaan (2.5) berikut:

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

1. *Median Filtering*

Median filter merupakan salah satu operasi smoothing non-linear, dimana operasi ini akan mengganti nilai tiap piksel dengan nilai median dari piksel-piksel dalam suatu wilayah kecil di sekitar piksel yang akan digantikan. Teknik ini bekerja cukup baik untuk menghilangkan noise, tanpa melakukan banyak blurring tepi dan dapat dilakukan secara berulang-ulang.

* 1. *Thresholding*

Citra *grayscale* umumnya memiliki 8 bit tiap pixel. Sementara pemrosesan citra *grayscale* lebih mudah dalam beberapa hal dibandingkan citra warna, terdapat bentuk citra yang lebih sederhana, yaitu citra biner, di mana prosesnya bahkan lebih mudah. Faktanya, sebagian besar dari aplikasi praktis dari visi komputer telah dikembangkan menggunakan citra biner.

Citra biner adalah citra yang hanya memiliki 1 bit untuk tiap pixelnya. Citra ini dibuat dengan menggunakan *threshold* di mana *threshold* yang digunakan ditentukan dalam berbagai cara. Citra biner yang dihasilkan biasanya diproses menggunakan morfologi matematika dan region biner yang disegmentasi diekstraksi dari citra menggunakan analisis komponen yang berkaitan.

Sebuah citra biner dibentuk dari citra *grayscale* dengan cara *thresholding*. Algoritma *thresholding* untuk citra biner adalah sebagai berikut persamaan (2.6):

|  |  |
| --- | --- |
|  | (2.6) |

Operasi *thresholding* umumnya digunakan untuk memisahkan *Object of Interest* dari background. *Object of Interest* umumnya direpresentasikan dengan nilai 1 (255) namun terkadang citra biner perlu diinvers untuk memperoleh *Object of Interest*. Contoh citra hasil *thresholding* dapat dilihat pada gambar 2.3. berikut.



Gambar 2.3. Citra *threshold*

* 1. Deteksi Tepi

Faktor kunci dalam mengekstraksi ciri adalah kemampuan mendeteksi keberadaan tepi (*edge*) dari objek di dalam citra. Setelah tepi objek diketahui, langkah selanjutnya dalam analisis citra adalah segmentasi, yaitu mereduksi citra menjadi objek atau region, misalnya memisahkan objek-objek yang berbeda dengan mengekstraksi batas-batas objek (*boundary*). Langkah terakhir dari analisis citra adalah klasifikasi, yaitu memetakan segmen-segmen yang berbeda ke dalam kelas objek yang berbeda pula.

Ada tiga macam tepi yang terdapat di dalam citra digital. Ketiganya adalah:

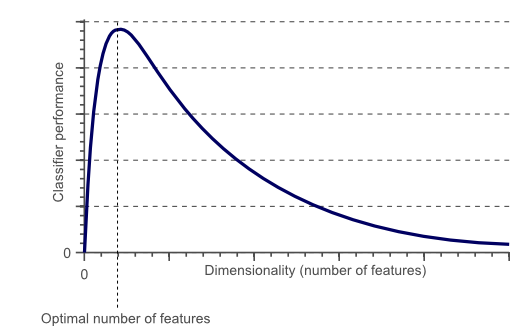
1. Tepi curam memiliki perubahan intensitas yang tajam. Arah tepi berkisar 90°.
2. Tepi landai atau disebut juga tepi lebar, yaitu tepi dengan sudut arah yang kecil. Tepi landai dapat dianggap terdiri dari sejumlah tepi-tepi lokal yang lokasinya berdekatan.
3. Tepi yang mengandung derau (*noise*) umumnya memiliki tepi yang tidak konsisten.
4. Pengenalan Pola

Pengenalan pola merupakan proses pengelompokan data numerik dan simbolik secara otomatis, yang bertujuan untuk mengidentifikasi objek pada citra. Pola adalah entitas yang terdefinisi dan dapat diidentifikasi melalui ciri-cirinya. Ciri-ciri tersebut digunakan untuk membedakan suatu pola dengan pola lainnya. Ciri yang bagus adalah ciri yang memiliki daya pembeda yang tinggi, sehingga pengelompokan pola berdasarkan ciri yang dimiliki dapat dilakukan dengan keakuratan yang tinggi. Ciri pada suatu pola diperoleh dari hasil pengukuran terhadap objek uji. Khusus pada pola yang terdapat di dalam citra, ciri-ciri yang dapat diperoleh berasal dari informasi berikut:

1. Spasial: intensitas pixel, histogram.
2. Tepi: arah, kekuatan.
3. Kontur: garis, elips, lingkaran.
4. Wilayah/bentuk: keliling, luas, pusat massa.
5. Hasil transformasi *fourier*: frekuensi.

## Dimensionality Reduction

*Dimensionality Reduction* merupakan suatu proses untuk mereduksi atau mengurangi jumlah dimensi variabel fitur dalam suatu kumpulan data dengan mencari beberapa variabel fitur utama. *Dimensionality Reduction* perlu dilakukan untuk mengurangi kompleksitas dari model pembelajaran mesin yang akan dibuat. Semakin banyak jumlah variabel fitur yang digunakan dalam model, maka jumlah sampel data yang dibutuhkan juga akan semakin banyak untuk menemukan korelasi antar fitur. Selain itu, semakin banyak jumlah fitur maka peluang terjadinya *overfitting* pada model juga akan semakin besar. Sebuah model pembelajaran mesin yang dilatih menggunakan jumlah fitur yang terlalu banyak, akan menjadi ketergantungan terhadap data yang digunakan untuk pelatihan sehingga dapat menghasilkan performa yang buruk pada saat pengujian. Hubungan antara dimensi variabel fitur dengan performa model dapat dilihat pada gambar 2.4. berikut.



Gambar 2.4. Grafik hubungan dimensi dan performa

Selain untuk mencegah *overfitting*, *dimensionality reduction* juga memberikan beberapa kelebihan, seperti:

1. Semakin sedikit data yang tidak berhubungan, maka semakin baik akurasi model yang dibuat.
2. Semakin sedikit dimensi variabel fitur, maka semakin sedikit komputasi yang dilakukan sehingga proses pelatihan dapat berlangsung lebih cepat.
3. Semakin sedikit data, maka semakin sedikit memori yang digunakan.
4. Mengurangi fitur berulang dan derau.

Metode paling sederhana yang bisa diterapkan untuk dimensionality reduction adalah seleksi fitur. Secara garis besar, terdapat dua metode pada seleksi fitur, yaitu *Linear Dimensionality Reduction* dan *Non-linear Dimensionality Reduction.*

1. ***Linear Dimensionality Reduction***

Beberapa jenis metode *Linear Dimensionality Reduction* antara lain:

1. *Principal Component Analysis (PCA)*: umumnya digunakan untuk mereduksi dimensi pada data kontinu, PCA merotasi dan memroyeksikan data sepanjang arah peningkatan varian. Fitur dengan varian maksimum dianggap sebagai komponen utama (*principal components*).
2. *Factor Analysis*: merupakan teknik yang digunakan untuk mereduksi variabel dalam jumlah besar ke dalam jumlah yang lebih kecil. Nilai-nilai data yang diamati dinyatakan sebagai fungsi dari sejumlah kemungkinan penyebab untuk menemukan nilai yang paling penting. Dengan asumsi bahwa pengamatan disebabkan oleh transformasi linear dari faktor laten dimensi yang lebih rendah dan penambahan *noise Gaussian.*
3. *Linear Discriminant Analysis (LDA)*: memroyeksikan data ke dalam cara di mana pemisahan kelas dimaksimalkan. Data-data dari kelas yang sama ditempatkan pada posisi yang berdekatan pada saat proyeksi. Sedangkan kelas yang berbeda akan ditempatkan jauh dari proyeksi kelas lain.
4. ***Non-linear Dimensionality Reduction***

Metode transformasi non-linear digunakan ketika data tidak berada pada ruang linear. Beberapa jenis metode *Non-linear Dimensionality Reduction* antara lain:

1. *Multi-dimensional scaling (MDS)*: yaitu sebuah teknik yang digunakan untuk menganalisis mirip atau tidaknya data sebagai jarak dalam ruang geometri. Memroyeksikan data ke dimensi yang lebih rendah sehingga titik data yang dekat satu sama lain (dalam hal ini jaran *Euclidean*) di dimensi yang lebih tinggi juga dekat dengan dimensi yang lebih rendah.
2. *Isometric Feature Mapping (Isomap)*: memroyeksikan data ke dalam dimensi yang lebih rendah dengan mempertahankan jarak *geodesic*. Jarak *geodesic* adalah jarak terpendek antara dua titik pada kurva.
3. *Locally Linear Embedding (LLE)*: memulihkan struktur non-linear global dari kecocokan linear. Setiap tambalan lokal dari *manifold* dapat ditulis sebagai *weighted sum* linear dari *neighbour-*nya dengan data yang cukup.
4. *Hessian Eigenmapping (HLLE)*: memroyeksikan data ke dimensi yang lebih kecil dengan mempertahankan *neighbourhood* lokal seperti LLE namun menggunakan operator *Hessian* untuk memperoleh hasil.
5. *Spectral Embedding (Laplacian Eigenmaps)*: menggunakan teknik spektral untuk mereduksi dimensi dengan memetakan masukan terdekat ke keluaran terdekat. Metode ini mempertahankan lokalitas daripada linearitas lokal.
6. *T-Distributed Stochastic Neighbor Embedding (t-SNE)*: menghitung probabilitas keterkaitan antar titik data dalam ruang dimensi tinggi dan kemudian memilih penerapan dimensi rendah yang menghasilkan distribusi yang serupa.

## Principal Component Analysis (PCA)

*Principal Component Analysis* (PCA) merupakan salah satu metode *dimensionality reduction* yang biasa digunakan untuk mereduksi dimensi dataset dalam jumlah besar, dengan menransformasikan variabel dalam dimensi besar ke dalam dimensi yang lebih kecil namun tetap mempertahankan informasi yang ada seperti pada dimensi yang lebih besar. Proses yang terjadi dalam PCA dijabarkan sebagai berikut (I Smith, 2002):

1. Melakukan standarisasi

Tahap pertama dalam PCA adalah melakukan standarisasi rentang nilai variabel data sehingga setiap variabel memiliki kontribusi yang sama terhadap analisis. Secara spesifik, standarisasi dilakukan untuk menghindari bias yang terjadi akibat perbedaan rentang nilai variabel (varians). Secara matematis, proses standarisasi dilakukan dengan mengurangkan nilai variabel dengan nilai rata-rata variabel tersebut. Persamaan (2.7) menunjukkan perhitungan pada tahap standarisasi.

|  |  |
| --- | --- |
|  | (2.7) |

1. Menghitung *covariance matrix*

Tahap selanjutnya adalah menghitung *covariance matrix* untuk mengetahui apakah ada hubungan antar variabel dalam dataset. Hal ini dilakukan karena terkadang terdapat variabel yang sangat berkaitan sehingga menyebabkan redundansi informasi.

*Covariance matrix* berukuran simetris (di mana adalah jumlah dimensi) yang memiliki entri kovarian yang terkait dengan semua pasangan yang mungkin dari variabel awal. Contoh *covariance matrix* untuk matriks data berukuran ditunjukkan oleh persamaan (2.8) berikut.

|  |  |
| --- | --- |
|  | (2.8) |

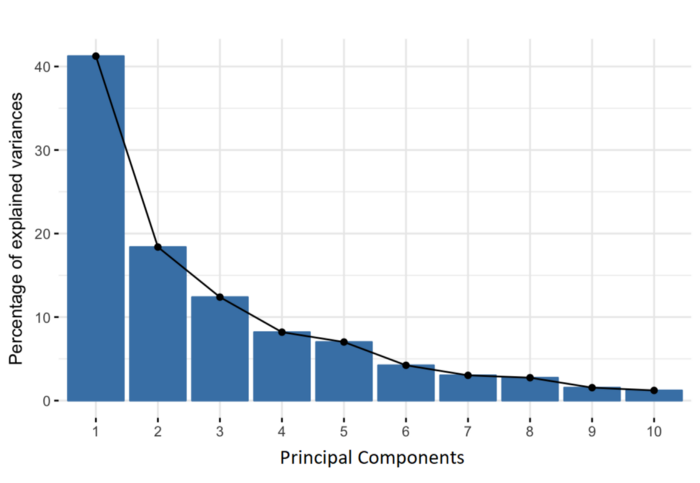
Nilai pada *covariance matrix* menandakan keterkaitan antar variabel dengan ketentuan sebagai berikut(Holland 2008):

* Bila bernilai positif, berarti pasangan variabel akan meningkat dan menurun bersama-sama (berkaitan)
* Bila bernilai negatif, berarti akan bergerak berlawanan di mana salah satu variabel meningkat dan variabel lainnya menurun (berkaitan terbalik)

1. Menghitung *eigen vector* dan *eigen value* dari *covariance matrix* untuk mengetahui komponen utama (*principal component*)

*Eigen value* dan *eigen vector* merupakan konsep aljabar linear yang perlu dihitung dari *covariance matrix* untuk mengetahui atau menentukan komponen utama (*principal components*) dari data.

*Principal components* merupakan variabel baru yang dibentuk dari kombinasi linear atau gabungan dari variabel awal. Kombinasi ini dilakukan dengan cara tertentu sehingga variabel baru tersebut tidak saling berkaitan dan sebagian besar informasi dalam variabel awal dipadatkan di dalam komponen pertama. Dengan kata lain, ide utama dari PCA adalah untuk membagi informasi ke dalam komponen-komponen utama di mana informasi akan dipadatkan pada komponen pertama dan kelebihan informasi lainya akan dipadatkan pada komponen selanjutnya, hingga menghasilkan produk komponen dengan informasi seperti pada gambar 2.5. berikut.



Gambar 2.5. Komponen pada PCA

Dengan menyusun komponen-komponen seperti pada gambar 2.5., maka dapat dilakukan pengurangan dimensi tanpa kehilangan banyak informasi dengan cara mengabaikan komponen yang mengandung sedikit informasi dan menganggap komponen lainnya sebagai variabel baru.

Penentuan komponen utama yang telah dijelaskan di atas dilakukan berdasarkan nilai *eigen vector* dari covariance matrix pada tahap sebelumnya. *Eigen vector* menyatakan arah dari sumbu di mana terdapat varians terbesar (informasi terbanyak) atau disebut juga *principal components.* Sedangkan *eigen value* merupakan koefisien dari *eigen vector*, yang menunjukkan jumlah varian yang dibawa oleh tiap *principal component.* Dengan mengurutkan *eigen vector* berdasarkan *eigen value-*nya dari besar ke kecil, dapat diperoleh urutan *principal components* dengan urutan berdasarkan informasi yang terkandung di dalamnya.

1. Menentukan vektor fitur

Pada tahap ini, dilakukan penentuan komponen yang akan digunakan. Penentuan komponen ini dapat dilakukan dengan tetap mengambil seluruh komponen yang ada, atau membuang komponen yang mengandung sedikit informasi sehingga dapat terjadi reduksi dimensi data.

Setiap nilai *eigen vector* dari komponen yang akan digunakan kemudian dimasukkan ke dalam kolom pada sebuah matriks yang disebut dengan vektor fitur.

1. Menyusun kembali data sepanjang sumbu komponen utama

Tahap terakhir dari PCA adalah menyusun kembali data fitur menggunakan vektor fitur yang telah dibentuk pada tahap sebelumnya. Penyusunan data ini dilakukan dengan mengorientasikan data dari sumbu awal ke dalam sumbu yang direpresentasikan oleh *principal components*. Hal ini dapat dilakukan dengan cara mengalikan *transpose* data awal dengan *transpose* dari vektor fitur yang diperoleh seperti yang dituliskan dalam persamaan (2.9) berikut:

|  |  |
| --- | --- |
|  | (2.9) |

## Jaringan Saraf Tiruan

Jaringan saraf tiruan merupakan teknik pembelajaran mesin yang menyimulasikan mekanisme pembelajaran pada organisme biologis. Sistem saraf manusia mengandung sel, yang disebut sebagai neuron. Mekanisme biologis ini disimulasikan dalam jaringan saraf tiruan yang berisi unit perhitungan yang juga disebut neuron. Unit komputasi (neuron) tersebut terhubung satu sama lain melalui bobot, yang berperan seperti koneksi sinaptik dalam organisme biologis. Setiap masukan ke neuron diskalakan dengan bobot, yang mempengaruhi nilai fungsi yang dihitung pada unit tersebut.

Jaringan saraf tiruan menghitung fungsi masukan dengan menyebarkan nilai yang dihitung dari neuron masukan ke neuron keluaran dan menggunakan bobot sebagai parameter perantara. Pembelajaran terjadi dengan mengubah bobot yang menghubungkan neuron. Sama seperti rangsangan eksternal yang diperlukan untuk belajar dalam organisme biologis, rangsangan eksternal dalam jaringan saraf tiruan berupa data pelatihan yang berisi pasangan masukan dan keluaran dari fungsi yang harus dipelajari. Pasangan data pelatihan ini dimasukkan ke dalam jaringan saraf tiruan dengan menggunakan representasi masukan untuk membuat prediksi tentang label keluaran. Data pelatihan memberikan umpan balik terhadap nilai bobot dalam jaringan saraf tiruan berdasarkan pada seberapa baik keluaran yang dihasilkan untuk masukan tertentu. (Aggarwal, 2018)

Haykin menyatakan bahwa “Jaringan saraf adalah prosesor terdistribusi paralel besar-besaran yang memiliki kecenderungan alami untuk menyimpan pengetahuan pengalaman dan membuatnya tersedia untuk digunakan. Ini menyerupai otak dalam dua hal: 1. Pengetahuan diperoleh oleh jaringan melalui proses pembelajaran; 2. Kekuatan interkoneksi antar neuron, yang dikenal sebagai bobot sinaptik atau bobot, digunakan untuk menyimpan pengetahuan.” (Samarasinghe, 2007)

Berdasarkan cara belajarnya, jaringan saraf tiruan terbagi atas dua, yaitu *Supervised Learning* dan *Unsupervised Learning*. (Duda, Hart, & Stork, 2000)

1. Supervised Learning

Pada supervised learning, seorang “guru” menyediakan kategori label atau cost untuk setiap pola dalam set pelatihan, dan mesin berusaha untuk mengurangi cost pada pola-pola tersebut.

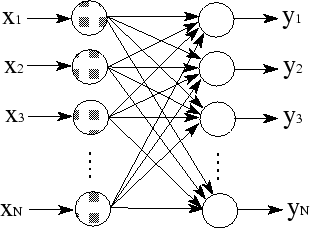
1. Unsupervised Learning

Pada unsupervised learning atau clustering, tidak ada “guru” spesifik, dan sistem akan membentuk cluster atau pengelompokan secara natural berdasarkan pola masukan. “Natural” selalu diartikan secara eksplisit maupun implisit dalam sistem pengelompokan itu sendiri, dan bila diberikan sebuah set pola tertentu, setiap algoritma clustering akan menghasilkan pengelompokan yang berbeda pula.

Sedangkan berdasarkan arsitekturnya, jaringan saraf tiruan terbagi atas dua, yaitu *Single Layer Network* dan *Multi Layer Network.*

1. Single Layer Network

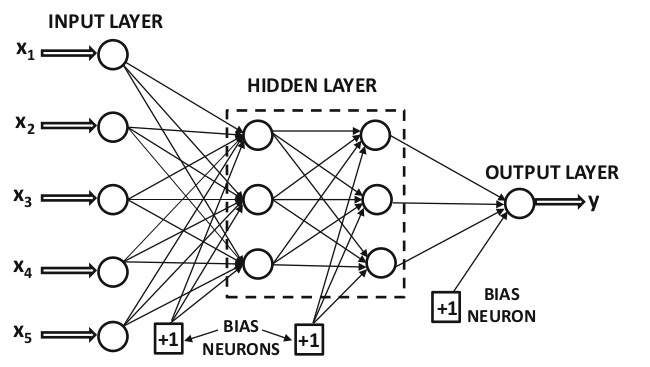
Arsitektur jaringan ini hanya terdiri dari satu *input layer* dan satu *output layer*. Setiap neuron yang terdapat di dalam *input layer* selalu terhubung dengan setiap neuron yang terdapat pada *output layer*. Jaringan ini hanya menerima masukan kemudian secara langsung akan mengolahnya menjadi suatu keluaran. Gambar 2.6. menunjukkan arsitektur dari *single layer network*.



Gambar 2.6. *Single layer network*

1. Multi Layer Network

Arsitektur jaringan ini terdiri dari lebih dari satu lapisan komputasional. Berbeda dari *single layer network* yang hanya terdiri dari *input layer* dan keluaran, pada *multi layer network* terdapat lapisan komputasi tambahan di antara *input layer* dan *output layer*. Arsitektur dari jaringan saraf *multi layer* secara spesifik dinyatakan sebagai *feed forward network*, karena nilai pada neuron pada suatu lapisan berturut-turut dimasukkan ke neuron pada lapisan berikutnya, mulai dari *input layer* hingga pada *output layer*. Gambar 2.7. menunjukkan arsitektur *multi layer network.*



Gambar 2.7. *Multi-layer network*

## Algoritma Backpropagation

Algoritma *Backpropagation* merupakan salah satu algoritma yang sering diterapkan dalam jaringan saraf tiruan. Pada umumnya, algoritma ini memiliki arsitektur seperti *multi layer network* yaitu terdiri dari *input layer*, *hidden layer*, dan *output layer*. Setiap lapisan memiliki neuron yang berfungsi untuk menampung dan memproses suatu nilai. Pada algoritma ini juga dikenal sebuah neuron yang terletak pada *hidden layer*, yaitu neuron bias yang berfungsi untuk membantu mempercepat proses pelatihan algoritma.

Proses algoritma *backpropagation* terdiri atas dua tahap, yaitu tahap *feed forward* dan tahap *backpropagation*. Tahap pertama adalah *feed forward*, yaitu tahap perhitungan yang berjalan dari *input layer* ke arah *output layer*, dan menghasilkan suatu nilai dari *output layer*. Setelah nilai keluaran diperoleh, kemudian dilakukan perhitungan eror antara nilai keluaran dengan keluaran yang diharapkan.Tahap selanjutnya adalah *backpropagation* yaitu tahap perhitungan mundur mulai dari *output layer* hingga *input layer*. Pada tahap ini, algoritma akan memanfaatkan nilai eror yang dihasilkan setelah tahap *feed forward* untuk memperbaiki bobot-bobot guna memperkecil nilai eror pada tahap *feed forward* selanjutnya. Seluruh proses ini akan berjalan pada setiap data yang akan menjadi masukan, dan akan berulang hingga kondisi tertentu terpenuhi.

Terdapat tiga parameter yang dapat mempengaruhi perhitungan pada algoritma *backpropagation*, yaitu laju pembelajaran, jumlah iterasi, dan batas eror minimum. Laju pembelajaran menentukan seberapa cepat suatu jaringan mengendalikan proses penyesuaian bobot. Rentang nilai laju pembelajaran adalah antara 0 hingga 1. Semakin kecil nilainya, maka konvergensi jaringan menjadi semakin lambat, sehingga iterasi yang dibutuhkan untuk melatih suatu jaringan menjadi lebih banyak, begitu pula sebaliknya. Jumlah iterasi merupakan jumlah yang menentukan seberapa banyak algoritma harus melakukan perhitungan untuk seluruh data masukan. Semakin banyak jumlah iterasi, maka semakin lama waktu yang diperlukan untuk proses pelatihan jaringan. Batas eror minimum adalah batas yang ditentukan untuk menghentikan perhitungan algoritma dengan membandingkan nilai eror perhitungan dengan batas eror minimum. Semakin kecil batas eror minimum, maka semakin banyak iterasi yang diperlukan untuk mencapai batas (Kholis, 2015).

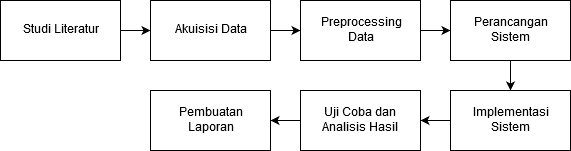
Secara garis besar, proses dalam algoritma *backpropagation* dijabarkan sebagai berikut:

1. Menentukan arsitektur jaringan, seperti jumlah *hidden layer*, jumlah neuron pada *hidden layer*, laju pembelajaran, batas eror minimum, dan jumlah iterasi.
2. Menentukan bobot awal untuk setiap neuron yang berhubungan. Penentuan bobot awal umumnya dilakukan secara acak menggunakan nilai kecil dalam rentang 0 hingga 1.
3. Melakukan perhitungan maju (*feed forward*). Untuk menghitung nilai neuron pada suatu lapisan, perhitungan dilakukan dengan mengambil total dari semua neuron pada lapisan sebelumnya yang dikalikan dengan bobot yang menghubungkannya dengan neuron yang ingin diketahui nilainya. Setelah itu, nilai total tersebut kemudian dimasukkan ke dalam suatu fungsi aktivasi untuk menentukan nilai neuron yang dicari. Salah satu fungsi aktivasi yang biasa digunakan adalah fungsi aktivasi sigmoid.
4. Melakukan perhitungan tahap ke-3 hingga memperoleh nilai pada *output layer*.
5. Menghitung eror dari nilai keluaran dengan mengurangkan nilai keluaran dengan nilai yang diharapkan.
6. Menghitung nilai turunan fungsi aktivasi menggunakan nilai eror.
7. Menghitung nilai eror neuron pada lapisan sebelumnya menggunakan nilai hasil turunan fungsi aktivasi.
8. Mengubah nilai bobot yang menghubungkan neuron dengan menjumlahkan nilai bobot lama dengan hasil kali nilai neuron dengan laju pembelajaran.
9. Melakukan langkah 5-8 untuk seluruh neuron pada setiap lapisan sebelum *output layer*.
10. Melakukan langkah 3-9 untuk seluruh baris data yang ada pada dataset.

# BAB III METODOLOGI PENELITIAN

## Tahapan Penelitian

Sistem yang diusulkan terdiri dari proses segmentasi *region of interest* (ROI), seleksi fitur menggunakan *principal component analysis* (PCA) dan algoritma jaringan saraf tiruan *backpropagation* untuk melakukan klasifikasi gangguan jantung. Adapun tahapan-tahapan yang akan dilakukan pada penelitian ini dapat dilihat pada gambar 3.1.



Gambar 3.1. Tahapan penelitian

Berdasarkan diagram pada gambar 3.1., tahapan penelitian secara garis besar dapat dijabarkan sebagai berikut.

1. Studi literatur merupakan tahap awal dalam penelitian ini. Tahap ini dilakukan untuk mengumpulkan penelitian terkait metode-metode yang digunakan baik dalam ekstraksi fitur, maupun klasifikasi. Pada tahap ini juga dilakukan pengumpulan landasan teori yang mendasari penelitian yang akan dilakukan.
2. Akuisisi data dilakukan dengan mengirimkan permohonan permintaan data kepada salah satu klinik yang melayani pemeriksaan iridologi. Data yang diperoleh berupa citra mata pasien yang didiagnosis menderita gangguan jantung berdasarkan hasil pemeriksaan iridologi dan citra mata pasien normal.
3. *Preprocessing* data yang dilakukan berupa menganalisis data yang layak digunakan, menyortir data dan melakukan labeling, mengisolasi area iris, dan melakukan penghitaman pada area pupil untuk mempermudah proses pengolahan selanjutnya.
4. Perancangan sistem dilakukan untuk mencari metode-metode yang akan digunakan dalam penelitian termasuk di dalamnya metode untuk melakukan segmentasi, mengekstraksi fitur, hingga melakukan klasifikasi. Dalam penelitian ini, metode yang digunakan untuk mengekstraksi fitur adalah deteksi tepi *canny* dan *principal component analysis* (PCA), dan untuk klasifikasi digunakan algoritma jaringan saraf tiruan *backpropagation*. Pada tahap ini juga dilakukan pembuatan *flowchart* terkait alur kerja sistem.
5. Implementasi sistem dilakukan sesuai dengan *flowchart* yang telah dibuat pada tahap sebelumnya. Pada penelitian ini, sistem dibuat dengan menggunakan bahasa pemrograman *python* dan *library OpenCV*.
6. Uji coba sistem dilakukan untuk mengetahui seberapa akurat sistem yang dibuat.
7. Tahap akhir dalam penelitian ini adalah melakukan penulisan laporan penelitian dalam bentuk skripsi sebagai bahan publikasi.

## Waktu dan Lokasi Penelitian

Penelitian ini dilakukan selama 12 bulan yang dimulai sejak disetujuinya proposal penelitian ini pada akhir bulan September 2018 hingga proses pelaporan hasil penelitian ini pada bulan September 2019. Penelitian ini dilakukan di Laboratorium Kecerdasan Buatan Departemen Teknik Informatika, Fakultas Teknik, Universitas Hasanuddin.

## Instrumen Penelitian

Instrumen yang digunakan pada penelitian ini adalah:

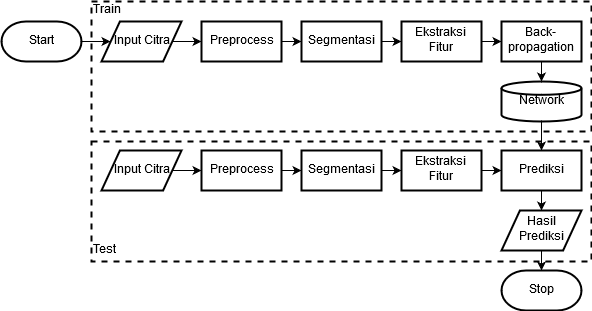
1. Software
   1. Sistem operasi: Windows 10 - 64bit
   2. Aplikasi pengolah citra: GIMP
   3. Bahasa pemrograman: Python 3.7
   4. OpenCV 4
2. Hardware
   1. Laptop Dell Inspiron 3443, RAM 8GB, Processor Intel(R) Core (TM) i7-5500U CPU @ 2.40GHz

## Akuisisi Data

Pada penelitian ini, data diperoleh dari salah satu klinik yang menyediakan pemeriksaan iridologi. Terdapat dua jenis data yang diperoleh, yang pertama adalah data citra mata para pasien yang telah diperiksa menggunakan iridologi dan terindikasi potensi gangguan jantung berjumlah 145 citra, dan yang kedua adalah data citra mata pasien yang tidak terindikasi gangguan jantung (normal) berjumlah 55 citra, sehingga total seluruh data berjumlah 200 citra. Semua data yang diperoleh dan digunakan merupakan citra mata sebelah kiri. Hal ini dikarenakan posisi yang menandakan gangguan jantung berdasarkan iridologi terletak di iris mata sebelah kiri.

Berdasarkan keterangan dari dokter pengelola klinik, pengambilan data dilakukan dengan menggunakan peralatan berupa kamera digital *Canon PowerShot* dengan *focal length* yang digunakan sebesar 4 mm, penyanggah kepala, serta sebuah lampu meja LED sebagai tambahan pencahayaan. Peletakan kamera berjarak kurang lebih 15 cm dari dudukan kepala, sementara posisi lampu diletakkan di samping kiri kamera.

## Perancangan dan Implementasi Sistem



Gambar 3.2. Garis besar sistem

Sistem yang dibuat secara garis besar terbagi atas dua bagian, yaitu proses pelatihan dan proses pengujian yang dapat dilihat pada gambar 3.2. Kedua proses tersebut akan melewati proses *preprocessing* dan segmentasi untuk selanjutnya diekstrak fiturnya berupa hasil deteksi tepi *canny* yang disederhanakan menggunakan *Principal Component Analysis* (PCA). Dalam proses pelatihan, fitur yang diekstrak akan digunakan untuk membuat model PCA dengan proses *fitting*, yang kemudian dimasukkan ke dalam Jaringan Saraf Tiruan *Backpropagation* untuk memperoleh model pelatihan *network* yang akan digunakan untuk proses pengujian. Sedangkan pada proses pengujian, fitur yang diekstrak akan ditransformasikan sesuai dengan model PCA yang telah dibuat pada proses pelatihan. Hasil PCA ini kemudian digunakan sebagai bahan yang akan diprediksi oleh model JST yang telah dibuat pada proses pelatihan. Adapun *flowchart* sistem secara keseluruhan dapat dilihat pada gambar 3.3.

Berdasarkan *flowchart* sistem pada gambar 3.3., maka perancangan sistem dapat diuraikan sebagai berikut:

### Input Citra

Langkah pertama adalah mempersiapkan data citra iris mata yang akan menjadi masukan. Sebelum dimasukkan ke dalam sistem, data sebelumnya harus melalui *preprocess* terlebih dahulu secara manual. *Preprocess* yang dilakukan berupa mengisolasi bagian iris dari citra mata utuh dan menghitamkan bagian pupil dari mata. Hasil *preprocess* yang akan digunakan dalam sistem memiliki keterangan sebagai berikut:

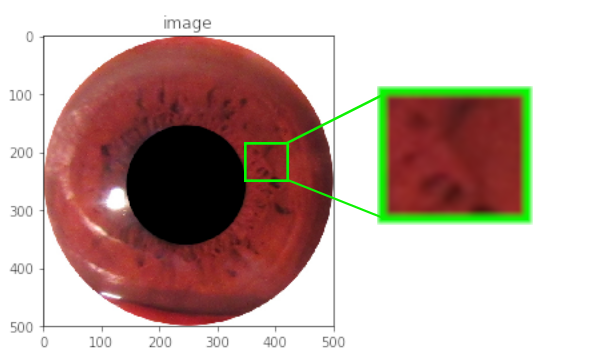
1. Dimensi : 500 × 500 pixel
2. Ekstensi : .JPG



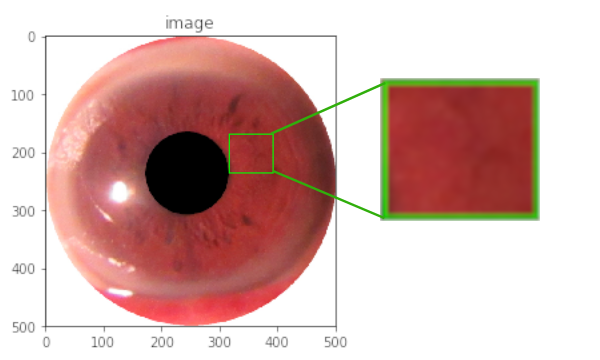
Gambar 3.3. Flowchart sistem

Data yang digunakan dalam pembuatan sistem berjumlah 110 citra yang terdiri dari 55 citra ‘normal’ dan 55 citra ‘berpotensi’. Data citra iris kemudian dibagi ke dalam dua bagian, yaitu data latih dan data uji dengan pembagian masing-masing sebesar 80%:20%. Untuk data latih, terdapat 44 citra ‘normal’ dan 44 citra ‘berpotensi’, sedangkan data uji terdiri dari 11 citra ‘normal’ dan 11 citra ‘berpotensi’.

Contoh data yang akan digunakan dalam sistem ditunjukkan pada gambar 3.4.



(a)



(b)

Gambar 3.4. (a) Citra iris berpenyakit jantung (b) Citra subjek normal

### *Grayscaling*

Proses pertama yang diterapkan pada gambar masukan adalah *grayscaling*, yaitu mengubah ruang warna gambar dari RGB menjadi *grayscale* (keabuan). Perubahan ruang warna ini dilakukan untuk mengurangi dimensi gambar yang akan diolah sehingga proses komputasi menjadi lebih ringan. Dalam *grayscaling*, untuk memperoleh nilai masing-masing pixel digunakan persamaan (3.1) sebagai berikut:

|  |  |
| --- | --- |
|  | (3.1) |

Di mana:

R = nilai pixel pada kanal Red (Merah)

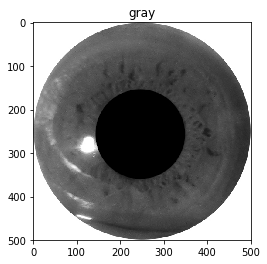
G = nilai pixel pada kanal Green (Hijau)

B = nilai pixel pada kanal Blue (Biru)

Gray = nilai pixel keabuan

Sebagai contoh, misalkan pada sebuah citra warna terdapat matriks, di mana pixelnya terdiri dari tiga kanal (*Red*, *Green*, dan *Blue*) bernilai [153, 63, 65]. Maka perhitungan untuk memperoleh nilai pixel keabuan dari pixel tersebut adalah sebagai berikut:

Adapun contoh hasil konversi citra RGB ke *grayscale* dapat dilihat pada gambar 3.5.



Gambar 3.5. Citra grayscale

### *Histogram Equalization*

Langkah selanjutnya setelah proses *grayscaling* adalah memperbaiki kontras citra. Perbaikan kontras citra dilakukan dengan ekualisasi histogram, yaitu meratakan penyebaran nilai-nilai keabuan yang terdapat dalam citra. Pada sistem yang dibuat, metode ekualisasi histogram yang digunakan adalah *Contrast Limited Adaptive Histogram Equalization* (CLAHE), yaitu metode ekualisasi histogram yang didasari oleh *Adaptive Histogram Equalization* (AHE) yang membatasi nilai peningkatan kontras yang dilakukan. CLAHE memproses citra dalam region-region kecil yang disebut *tiles* dengan menghitung nilai fungsi transformasi untuk setiap *tiles* secara individual. *Tiles* yang bertetangga kemudian digabungkan dengan menggunakan interpolasi bilinear untuk menghilangkan batasan antar *tiles* yang terjadi. Pada pengaplikasiannya, CLAHE memerlukan dua parameter yaitu *clipLimit* (batas kontras) dan ukuran *tiles*. Perubahan histogram menggunakan CLAHE ditunjukkan pada gambar 3.6.(a) dan 3.6.(b) berikut.

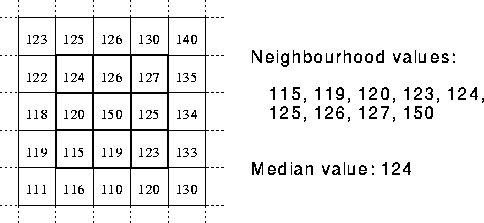
|  |  |
| --- | --- |
| C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\4EE9BDDF.tmp  (a) | C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\AA6BE245.tmp  (b) |
| Gambar 3.6. (a) Histogram awal, (b) Histogram hasil ekualisasi | |

### 

### *Deblurring*

Setelah citra hasil ekualisasi histogram diperoleh, langkah selanjutnya adalah mengurangi *noise* menggunakan *median filtering*. *Median filtering* mempertimbangkan setiap pixel dalam citra secara bergantian dan melihat nilai pixel tetangganya untuk menentukan apakah nilai pixel tersebut dapat menjadi representasi dari pixel-pixel sekitarnya. Nilai median mula-mula dihitung dengan mengurutkan semua nilai pixel yang bertetangga ke dalam urutan numerik dan kemudian mengganti nilai pixel tengah dengan nilai mediannya. Langkah-langkah yang dilakukan dalam *median filtering* antara lain:

1. Menentukan ukuran *neighborhood* filter. Misalnya 3x3 atau 5x5.
2. Memeriksa setiap nilai pixel dalam cakupan ukuran *neighborhood* yang sudah ditentukan.
3. Mengurutkan semua nilai pixel dalam neighborhood.
4. Menentukan nilai median dari urutan pixel. Sebagai contoh dapat dilihat pada gambar 3.7.



Gambar 3.7. Contoh median filtering

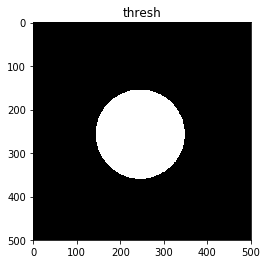
1. Mengganti nilai yang berada di tengah kelompok *neighborhood*. Pada gambar 3.7. nilai yang berada di tengah kernel (150) akan diubah menjadi nilai median *neighborhood* yaitu 124.
2. Menggeser kernel median filter sebanyak satu pixel ke kanan dan mengulangi langkah 2-6, hingga semua *pixel* pada citra telah diproses.

### *Pupil Masking*

Setelah citra *blur* diperoleh, kemudian dilakukan *masking* region pupil dari citra. Proses ini dijelaskan dalam langkah-langkah berikut:

1. Konversi citra *grayscale* ke biner

Konversi citra *grayscale* ke biner dilakukan dengan menggunakan *invers threshold*. Pada proses *threshold* yang dilakukan, digunakan nilai *threshold* sebesar 10. Nilai tersebut digunakan karena wilayah pupil pada citra mata umumnya berwarna gelap mendekati hitam (bernilai 0), sehingga bila nilai pixel pada citra lebih kecil atau sama dengan nilai *threshold*, maka nilai pixel akan diubah menjadi 1. Hasil dari proses *threshold* ini dapat dilihat pada gambar 3.8.



Gambar 3.8. Citra biner

1. Mencari kontur terbesar

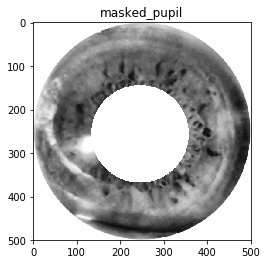
Setelah diperoleh citra biner dengan pupil berwarna putih, langkah selanjutnya adalah mencari kontur terbesar dari citra. Hal ini dilakukan untuk mengatasi masalah yang timbul bila terdapat objek lain selain pupil yang terdeteksi. Objek lain yang mungkin terdeteksi pada saat *thresholding* adalah serat iris yang berwarna cukup gelap. Namun serat iris tersebut umumnya berukuran kecil, sehingga pencarian kontur terbesar dilakukan untuk memastikan bahwa yang diseleksi adalah region pupil dari citra.

1. Mengestimasi lingkaran pupil

Setelah kontur terbesar (pupil) diketahui, kemudian dilakukan estimasi bentuk lingkaran terdekat dari pupil. Estimasi ini dilakukan untuk memperoleh nilai *center* dan *radius* yang paling dekat dengan pupil, untuk kemudian digunakan untuk proses *masking*.

1. Menerapkan masking pada region pupil

Setelah nilai *center* dan *radius* pupil diperoleh, kemudian proses *masking* dapat dilakukan. Pada sistem yang dibuat, proses *masking* dilakukan dengan membuat lingkaran dengan parameter *center* dan *radius* pupil yang telah diperoleh. Lingkaran yang dibuat diberi warna putih dan akan memenuhi wilayah pupil dari citra mata. Hasil dari proses *masking* ini dapat dilihat pada gambar 3.9.



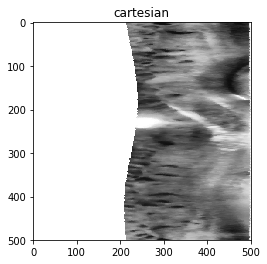
Gambar 3.9. Citra hasil masking

### *Region of Interest (ROI) Cropping*

Setelah tahap *preprocessing* selesai, langkah selanjutnya adalah menyegmentasi *region of interest* (ROI) dari citra iris. Pada penelitian ini, ROI yang akan diambil adalah bagian iris pada arah jam 2 hingga jam 3, dimulai dari ujung pupil hingga tengah cincin iris. Proses segmentasi ROI yang dilakukan terdiri dari beberapa langkah, antara lain:

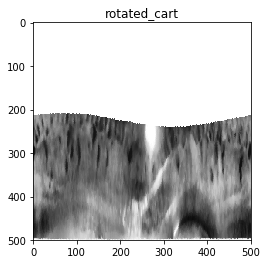
1. Transformasi citra polar iris ke kartesian

Langkah pertama dalam segmentasi ROI adalah mengubah bentuk citra lingkaran iris dari polar menjadi bentuk kartesian seperti pada gambar 3.10. Transformasi ini menggunakan parameter berupa pusat dari iris dan radius iris yang diperoleh dengan mengambil pusat gambar (x, y) sebagai pusat iris, dan jarak dari ujung hingga ke pusat gambar untuk radiusnya. Dalam penelitian ini, diperoleh nilai pusat (250.250) dan nilai radius sebesar 250 pixel.



Gambar 3.10. Citra kartesian

Setelah itu, citra kemudian dirotasi sebesar -90 derajat untuk mempermudah mengambil target region yang diinginkan. Hasil rotasi citra dapat dilihat pada gambar 3.11.



Gambar 3.11. Citra hasil rotasi

1. Mencari target region

Langkah selanjutnya adalah menentukan target area yang akan digunakan sebagai ROI. Pada gambar 3.12.(a) letak organ jantung akan berada pada satu per dua belas bagian pertama dari keseluruhan gambar. Target tersebut kemudian dipotong dari gambar seperti pada gambar 3.12.(b).

|  |  |
| --- | --- |
| C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\DDC40ED9.tmp  (a) | C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\9A552F4F.tmp  (b) |
| Gambar 3.12. (a) Target, (b) Hasil crop target | |

1. Menghilangkan region pupil

Langkah selanjutnya adalah memotong area pupil (area besar berwarna putih) untuk mempermudah dalam melakukan spesifikasi ROI yang akan diambil fiturnya. Proses pemotongan ini dilakukan dengan mencari kontur terbesar dari gambar kemudian menentukan ukuran box yang mewakili ukuran pupil yang akan dibuang. Hasil penentuan pupil dan pemotongannya dapat dilihat pada gambar 3.13.(a) dan 3.13.(b) berikut.

|  |  |
| --- | --- |
| C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\14B6C287.tmp  (a) | C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\8ADB9351.tmp  (b) |
| Gambar 3.13. (a) Area Pupil, (b) Hasil crop pupil | |

1. Memotong ROI

Tahap terakhir dari segmentasi adalah menentukan ROI spesifik yang akan digunakan untuk mencari fitur. Proses ini dilakukan dengan mengambil potongan citra sebesar 50% dari panjang citra, atau dengan kata lain pemotongan dari titik (0,0) hingga (x, y/2). Setelah citra dipotong, kemudian ROI di-*resize* ukurannya menjadi 50x50 pixel. Hasil penentuan ROI dapat dilihat pada gambar 3.14. berikut.

|  |  |  |
| --- | --- | --- |
| C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\930552D.tmp  (a) | C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\1C75CCC3.tmp  (b) | C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\6F1BA6C9.tmp  (c) |
| Gambar 3.14. (a) ROI, (b) Hasil crop ROI, (c) Hasil resize ROI | | |

### *Canny Edge Detection*

Setelah ROI jantung diperoleh dari citra iris, selanjutnya dilakukan penerapan deteksi tepi Canny untuk menentukan tepian dari target fitur yang akan diambil dari ROI. Pada proses ini, digunakan *double thresholding* (penentuan *low threshold* dan *high threshold*) untuk menghilangkan derau dan mempertegas bagian yang termasuk tepian dan yang bukan tepian.

Pada penelitian ini, penentuan nilai *low threshold* dan *high threshold* dibuat lebih dinamis mengikuti citra masukan dengan melakukan langkah-langkah berikut:

1. Mencari nilai median dari citra
2. Menentukan nilai operator konstanta (sigma)
3. Menghitung *low* dan high *threshold* dengan persamaan (3.2).

|  |  |
| --- | --- |
|  | (3.2) |

Oleh karena nilai *threshold* dibuat dinamis mengikuti citra masukan, maka tidak perlu dilakukan percobaan terhadap nilai *low* dan *high* *threshold*, yang perlu dilakukan hanya mengubah satu nilai sigma sebagai operator konstanta. Pada penelitian ini, dilakukan percobaan dengan menggunakan nilai sigma sebesar 0.1, 0.2, 0.3, 0.4, dan 0.5 untuk menghasilkan citra tepian. Masing-masing citra tepian tersebut kemudian akan digunakan pada proses selanjutnya hingga menghasilkan prediksi. Hasil dari sistem kemudian akan dibandingkan satu sama lain untuk tiap nilai sigma.

Adapun proses yang terjadi dalam proses deteksi tepi Canny dijelaskan sebagai berikut:

1. Mengurangi derau menggunakan *Gaussian blur*.

Deteksi tepi umumnya sangat sensitif terhadap derau, oleh karena itu pengurangan derau perlu dilakukan untuk mengurangi kesalahan deteksi tepi. Metode pengurangan derau yang digunakan dalam deteksi tepi canny adalah *Gaussian blur* dengan ukuran kernel 5×5 dan nilai standar deviasi (σ) sebesar 1,4. Citra asal kemudian dikonvolusi dengan kernel *Gaussian* untuk memperoleh citra dengan derau yang lebih sedikit.

1. Menentukan gradien intensitas

Langkah selanjutnya adalah menentukan gradien intensitas tepi. Untuk menghitung gradien intensitas tepi, digunakan filter Sobel yang dapat menegaskan perubahan intensitas pada kedua arah horizontal (x) dan vertikal (y). Filter Sobel yang digunakan untuk masing-masing arah ditunjukkan oleh persamaan (3.3) berikut.

|  |  |
| --- | --- |
|  | (3.3) |

Hasil dari proses filter Sobel adalah turunan pertama dari arah horizontal dan arah vertikal . Setelah turunan pertama diperoleh, kemudian nilai *magnitude* dan *slope* θ dari gradien dihitung dengan persamaan (3.4) berikut.

|  |  |
| --- | --- |
|  | (3.4) |

1. Melakukan *Non-Maximum Supression*

Setelah memperoleh nilai gradien *magnitude* dan arah tepian, kemudian dilakukan pemeriksaan terhadap keseluruhan gambar untuk menghilangkan pixel yang tidak diinginkan yang mungkin tidak mengandung tepi. Proses ini dilakukan dengan memeriksa apakah nilai suatu pixel merupakan *local maximum* dari pixel tetangganya dalam suatu arah gradien.



Gambar 3.15. *Non-maximum supression*

Pada gambar 3.15. titik A berada pada tepian (dalam arah vertikal). Arah gradien tegak lurus terhadap tepian. Titik B dan C berada pada arah gradien. Sehingga titik A diperiksa terhadap titik B dan C apakah titik A merupakan *local maximum*. Jika ya, maka titik A dianggap sebagai bagian dari tepi. Namun jika tidak, titik A akan ditekan (diubah menjadi nol).

1. *Double thresholding*

Proses ini bertujuan untuk mengidentifikasi jenis pixel, yaitu kuat, lemah, dan non-relevan. Pixel kuat adalah pixel yang memiliki intensitas tinggi dan diyakini sebagai tepian. Pixel lemah adalah pixel yang memiliki nilai intensitas yang tidak cukup tinggi untuk dikategorikan sebagai pixel kuat, namun tidak cukup kecil untuk dikategorikan sebagai pixel non-relevan. Pixel non-relevan adalah pixel yang intensitasnya sangat rendah dan diyakini sebagai bukan tepian.

*Double threshold* terdiri atas dua yaitu *high threshold* dan *low threshold*. *High threshold* digunakan untuk mendeteksi pixel kuat (yang berintensitas lebih dari *high threshold*), sedangkan *low threshold* digunakan untuk mendeteksi pixek non-relevan (yang berintensitas di bawah *low threshold*). Pixel yang berada di antara *high* dan *low threshold* dianggap sebagai pixek lemah, dan untuk menentukan apakah termasuk tepi atau bukan maka dilakukan mekanisme *Hysteresis*.

1. *Hysteresis thresholding*

*Hysteresis thresholding* berfungsi untuk mengubah pixel lemah menjadi pixel kuat. Hal ini dilakukan dengan cara memeriksa nilai pixel sekitar pixel lemah. Pixel lemah akan diubah menjadi pixel kuat jika dan hanya jika terdapat satu atau lebih pixel kuat pada pixel tetangganya.

Contoh hasil deteksi tepi canny pada ROI dapat dilihat pada gambar 3.16. berikut.



Gambar 3.16. Hasil deteksi tepi *canny*

### Seleksi Fitur *Principal Component Analysis (PCA)*

Setelah seluruh citra pada data latih dan data uji telah diproses menggunakan deteksi tepi *canny*, kemudian dilakukan pembentukan dataframe. Pembentukan dataframe dilakukan dengan mengubah array tiap citra pada masing-masing dataset ke dalam bentuk satu dimensi, dan digabungkan ke dalam masing-masing dataframe (latih dan uji), serta diberi label ‘0’ untuk normal dan ‘1’ untuk gangguan jantung. Contoh dataframe latih awal dan dataframe uji awal dapat dilihat pada tabel 3.1. dan tabel 3.2.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Tabel 3.1. Contoh dataframe latih | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | … | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | ... | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **255** | **255** | **255** | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Tabel 3.2. Contoh dataframe uji | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | … | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | **255** | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | ... | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | **255** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Sebelum diproses menggunakan PCA, perlu dilakukan normalisasi terhadap dataframe untuk menyederhanakan perhitungan. Normalisasi pada penelitian ini menggunakan *MinMaxScaler*, yaitu normalisasi dengan mengubah nilai dalam dataframe ke dalam rentang 0 hingga 1. Contoh dataframe hasil normalisasi untuk data latih dan data uji dapat dilihat pada tabel 3.3. dan 3.4.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Tabel 3.3. Contoh dataframe latih hasil normalisasi | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | … | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | ... | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | **1** | **1** | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Tabel 3.4. Contoh dataframe uji hasil normalisasi | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | … | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | ... | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Setelah normalisasi, langkah selanjutnya adalah melakukan dekomposisi PCA. Proses ini dilakukan untuk kedua dataframe latih dan dataframe uji. Langkah-langkah pada dekomposisi PCA dapat dituliskan sebagai berikut:

1. Melakukan standarisasi.
2. Menghitung *covariance matrix.*
3. Menghitung *eigen vector* dan *eigen value* dari *covariance matrix.*
4. Menentukan vektor fitur.
5. Menyusun kembali data sepanjang sumbu komponen utama.

Proses yang terjadi dalam dekomposisi PCA ditunjukkan dalam contoh berikut:

1. Misalkan sebuah matriks berukuran 3×2 sebagai berikut:
2. Melakukan Standarisasi dengan menggunakan persamaan (2.7) untuk tiap kolom matriks.
3. Menghitung nilai *covariance matrix* dari matriks yang telah distandarisasi. Perhitungan *covariance* dapat dilakukan dengan persamaan (3.5) berikut.

|  |  |
| --- | --- |
|  | (3.5) |

Sedangkan untuk *covariance matrix* dapat dilakukan menggunakan persamaan (3.6) berikut.

|  |  |
| --- | --- |
|  | (3.6) |

Sehingga diperoleh *covariance matrix* sebagai berikut.

1. Mencari nilai *eigen vector* dan *eigen value* dari *covariance matrix.*
2. Menentukan vektor fitur yang akan digunakan berdasarkan *eigen vektor dan eigen value*. *Eigen vector* dengan *eigen value* kecil dapat diabaikan, sehingga vektor fitur hanya terdiri dari *eigen vector yang memiliki eigen value* besar.
3. Menghitung nilai matriks baru menggunakan persamaan (2.9).

Contoh hasil dekomposisi PCA untuk dataframe latih dan uji untuk ukuran PC 50 dapat dilihat pada tabel 3.5. dan 3.6. berikut.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Tabel 3.5. Contoh dataframe latih hasil dekomposisi PCA | | | | | | | | | | | | -0,166 | -2,069 | 0,437 | -0,559 | 0,964 | … | 0,228 | -1,391 | 1,633 | -0,209 | -0,885 | | -2,563 | -0,856 | 2,341 | 0,688 | -0,334 | -0,929 | 1,095 | 0,693 | -1,871 | 0,422 | | -0,507 | -3,135 | 1,256 | 0,167 | -0,839 | -0,627 | 4,827 | 3,552 | 0,349 | 0,612 | | -2,695 | 0,503 | 0,399 | 0,368 | 0,231 | -0,513 | -0,417 | -0,024 | -0,643 | 0,441 | | -1,489 | -0,714 | 0,023 | -0,204 | 0,380 | -0,529 | 1,985 | -3,030 | 2,266 | -2,863 | | -1,341 | -0,194 | 0,683 | 0,060 | -0,350 | 0,086 | 0,153 | 2,231 | -0,589 | -0,315 | | 2,281 | 0,527 | -4,267 | 1,100 | -1,391 | -0,727 | -0,445 | -1,220 | -0,506 | 0,662 | | ... | | | | | | | | | | | | 1,406 | 0,750 | -1,707 | 0,025 | 1,184 | ... | 1,845 | -0,518 | -0,072 | 0,764 | 1,097 | | 5,792 | -2,243 | -0,537 | -9,478 | -5,790 | -0,002 | -0,584 | 0,876 | -0,932 | -0,764 | | 0,898 | -0,169 | -0,881 | -0,294 | 1,562 | 0,526 | -0,171 | -0,791 | -0,805 | 0,860 | | 4,963 | 1,077 | -3,958 | -1,320 | 6,956 | -0,946 | -0,203 | -1,152 | -0,674 | -0,212 | | -0,052 | 0,750 | -1,683 | 0,093 | 0,524 | -0,427 | -0,081 | -0,957 | 0,201 | -0,482 | | -1,117 | -1,546 | 0,086 | 0,300 | 0,397 | -1,042 | 2,354 | 3,202 | 0,119 | 0,946 | | -1,728 | -0,822 | 0,965 | -1,594 | -0,133 | -0,503 | -1,829 | -2,099 | -1,529 | 2,070 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Tabel 3.6. Contoh dataframe uji hasil dekomposisi PCA | | | | | | | | | | | | -1,655 | -0,074 | -0,038 | 0,174 | -0,371 | … | 0,098 | -0,044 | -0,596 | 0,153 | -0,334 | | -0,962 | -0,987 | 0,397 | -0,004 | -0,230 | -0,327 | -0,388 | 0,051 | -0,279 | -0,436 | | -0,806 | 0,966 | -0,841 | 0,072 | 0,547 | -0,515 | 0,470 | -0,155 | 0,140 | -0,041 | | -1,918 | 0,105 | 1,452 | 0,517 | -0,571 | -0,113 | 0,009 | -0,460 | -0,006 | -0,396 | | -2,568 | 0,126 | 0,388 | 0,267 | 0,017 | 0,156 | -0,411 | 0,230 | 0,361 | 0,295 | | -1,492 | 0,069 | 0,793 | 0,037 | 0,013 | -0,588 | 0,378 | 0,028 | -0,021 | -0,387 | | -1,128 | 0,447 | 0,028 | 0,144 | -0,055 | -0,224 | 0,098 | -0,598 | -0,003 | -0,255 | | ... | | | | | | | | | | | | 0,108 | -1,378 | 0,586 | -0,522 | 0,588 | ... | -0,536 | -0,080 | 0,014 | 0,065 | 0,189 | | 5,054 | -0,325 | -0,479 | -0,247 | -0,446 | -0,534 | -0,427 | -0,082 | 0,231 | 0,481 | | 3,439 | 0,899 | -0,236 | 0,455 | 0,565 | 0,821 | -0,239 | -0,162 | -0,035 | -0,086 | | 3,739 | -0,022 | -0,257 | -0,950 | 0,047 | -0,243 | 0,368 | -0,082 | 0,438 | -0,008 | | 0,775 | 0,144 | -0,390 | -0,430 | 0,394 | -0,035 | -0,016 | -0,512 | 0,060 | 0,668 | | 0,525 | -0,765 | -0,348 | 0,294 | -0,420 | -0,033 | 0,777 | -0,184 | 0,927 | -0,033 | | 2,476 | -0,180 | -0,920 | 0,573 | 0,855 | 0,053 | 0,149 | 0,872 | 0,854 | -0,080 | |

### Model Jaringan *Backpropagation*

Pada tahap ini, dilakukan pembuatan model jaringan *backpropagation* yang berfungsi untuk melakukan prediksi kelas data, apakah termasuk ‘normal’ atau ‘berpotensi’. Pada penelitian ini, kelas data dikodekan dengan ‘0’ sebagai kelas ‘normal’ dan ‘1’ sebagai kelas ’berpotensi’. Pembuatan model jaringan *backpropagation* dibagi menjadi dua tahap, yaitu tahap pelatihan dan tahap pengujian.

1. Tahap Pelatihan

Tahap pelatihan bertujuan untuk memperoleh arsitektur dan nilai bobot yang tepat untuk digunakan dalam perhitungan sehingga dapat menghasilkan prediksi. Dalam algoritma *backpropagation*, terdapat dua proses yang dilakukan pada saat pelatihan yaitu *feed forward* dan *backpropagation.* Proses *feed forward* merupakan proses yang akan menghasilkan nilai keluaran, sedangkan *backpropagation* merupakan proses yang akan memperbaiki nilai bobot berdasarkan eror keluaran dari *feed forward*. Langkah-langkah proses pelatihan algoritma *backpropagation* dapat dijabarkan sebagai berikut:

1. Menentukan arsitektur jaringan

Langkah pertama dalam proses pelatihan algoritma *backpropagation* adalah menentukan arsitektur jaringan. Arsitektur jaringan yang perlu didefinisikan di awal antara lain:

* + 1. Jumlah neuron pada *input layer*
    2. Jumlah *hidden layer*
    3. Jumlah neuron pada *hidden layer*
    4. Jumlah neuron pada *output layer*

1. Menginisialisasi bobot awal

Langkah selanjutnya adalah membangkitkan nilai bobot awal untuk setiap jalur neuron yang ada pada jaringan dan ditambah dengan satu nilai bias untuk masing-masing *hidden layer* yang ada. Penentuan nilai bobot awal dilakukan secara acak dalam rentang 0 hingga 1.

1. Menghitung nilai neuron dan nilai aktivasi neuron

Setelah memperoleh nilai bobot awal, kemudian dilakukan perhitungan nilai neuron dan nilai aktivasi neuron untuk setiap neuron pada lapisan setelah *input layer* (*input layer*) atau disebut juga *feed forward*. Untuk *hidden layer* pertama, nilai input digunakan untuk menghitung nilai neuron dan aktivasinya. Sedangkan untuk *hidden layer* berikutnya hingga *output layer*, nilai aktivasi neuron sebelumnya yang akan menjadi masukan bagi neuron di lapisan tersebut. Pada penelitian ini, digunakan fungsi aktivasi *sigmoid*. Untuk menghitung nilai neuron dan nilai aktivasinya, digunakan persamaan (3.7) dan (3.8) berikut.

|  |  |
| --- | --- |
|  | (3.7) |
|  | (3.8) |

1. Menghitung eror dan delta eror pada *output layer*

Setelah memperoleh nilai pada *output layer*, kemudian dilakukan perhitungan eror. Perhitungan eror dilakukan dengan mengurangkan nilai keluaran yang diharapkan dengan nilai keluaran sistem, seperti yang ditunjukkan pada persamaan (3.9) berikut.

|  |  |
| --- | --- |
|  | (3.9) |

Sedangkan untuk delta eror dihitung dengan mengalikan nilai eror dengan turunan fungsi aktivasi yang digunakan (dalam hal ini sigmoid) seperti pada persamaan (3.10) berikut.

|  |  |
| --- | --- |
|  | (3.10) |

Nilai eror dan delta eror dihitung untuk setiap kelas yang ada pada *output layer*.

1. Menghitung nilai eror dan delta eror neuron pada *hidden layer*

Setelah memperoleh nilai dan pada *output layer*, kemudian dapat dilakukan perhitungan nilai eror pada neuron di *hidden layer*. Perhitungan eror dan delta eror dilakukan dengan persamaan (3.11) dan (3.12) berikut.

|  |  |
| --- | --- |
|  | (3.11) |
|  | (3.12) |

1. Menghitung nilai eror pada setiap bobot

Untuk menghitung besar eror bobot yang menghubungkan *hidden layer* dengan *output layer*, digunakan persamaan (3.13) berikut.

|  |  |
| --- | --- |
|  | (3.13) |

Sedangkan untuk eror pada bobot yang menghubungkan *input layer* dengan *output layer* dituliskan dalam persamaan (3.14) berikut.

|  |  |
| --- | --- |
|  | (3.14) |

1. Menghitung nilai bobot baru

Setelah memperoleh nilai eror bobot yang menghubungkan setiap neuron, kemudian dilakukan perhitungan nilai bobot baru. Perhitungan bobot baru ini dapat dilakukan dengan persamaan (3.15) berikut.

|  |  |
| --- | --- |
|  | (3.15) |

1. Mengulang langkah c-f untuk semua baris data selanjutnya dalam dataset. Setelah seluruh baris data dalam dataset diproses, maka dinyatakan sebagai satu *epoch*.
2. Mengulang langkah a-i hingga eror keluaran mencapai batas eror yang ditentukan, atau mencapai batas *epoch* maksimum.
3. Tahap Pengujian

Pada tahap pengujian, dilakukan uji coba terhadap model yang telah dibuat pada tahap pelatihan. Proses yang terjadi pada saat pengujian sama dengan proses yang terjadi pada langkah *feed forward*, yaitu terjadi perhitungan nilai pada setiap neuron dalam arsitektur jaringan hingga menghasilkan nilai keluaran. Namun yang membedakan dari *feed forward* pada tahap pelatihan adalah nilai bobot yang digunakan pada saat pengujian merupakan nilai bobot akhir yang diperoleh pada saat pelatihan. Indeks dari nilai keluaran terbesar kemudian diambil dan dinyatakan sebagai hasil prediksi kelas data.

Sebagai contoh, misalnya untuk prediksi dua kelas diperoleh nilai keluaran sebagai berikut:

|  |  |
| --- | --- |
| Kelas | Nilai Keluaran |
| 0 | 0.56 |
| 1 | 0,97 |

Maka hasil yang keluar sebagai prediksi adalah nilai keluaran terbesar, yaitu 0,97 yang berindeks pada kelas 1. Sehingga prediksi dari contoh di atas adalah 1.

## Analisis Kerja Sistem

Analisis kerja sistem prediksi potensi gangguan jantung dilakukan dengan menghitung nilai akurasi berdasarkan ketepatan prediksi yang dilakukan oleh sistem. Perhitungan nilai akurasi dilakukan menggunakan bantuan *confusion matrix* seperti pada gambar 3.17. berikut.



Gambar 3.17. Confusion matrix

*Confusion matrix* terdiri atas empat bagian, yaitu:

1. True Positif : yaitu jumlah prediksi benar dari data positif.
2. False Positif : yaitu jumlah prediksi salah dari data positif.
3. False Negatif : yaitu jumlah prediksi salah dari data negatif.
4. True Negatif : yaitu jumlah prediksi benar dari data negatif.

Untuk menghitung nilai akurasi sistem, dapat dilakukan dengan persamaan (3.16) berikut.

|  |  |
| --- | --- |
|  | (3.16) |

# BAB IV HASIL DAN PEMBAHASAN

## Hasil Penelitian

Pada bab ini, disajikan hasil kinerja sistem prediksi potensi gangguan jantung menggunakan metode jaringan saraf tiruan *backpropagation*. Pada penelitian ini, *confusion matrix* digunakan untuk membantu dalam melakukan perhitungan akurasi sistem. Tabel *confusion matrix* untuk masing-masing percobaan batas eror 0.1 dan 0.01 dapat dilihat pada lampiran 1 dan 2. Adapun jumlah data yang digunakan sebanyak 110 citra, yang dibagi menjadi 88 data latih dan 22 data uji.

Pada penelitian ini, dilakukan percobaan terhadap empat variabel yang dapat mempengaruhi kinerja sistem dalam menghasilkan prediksi. Keempat variabel tersebut antara lain:

1. Nilai *sigma* pada deteksi tepi *canny* sebesar **0.1**, **0.2**, **0.3**, **0.4**, dan **0.5**.
2. Target ukuran *principal component (PC)* sebesar **10**, **20**, **30**, **40**, dan **50**.
3. Nilai batas eror pada proses *backpropagation* sebesar **0.1** dan **0.01**.
4. Jumlah *hidden neuron* pada proses *backpropagation* sebanyak **10**, **20**, **30**, **40**, dan **50**.

Sedangkan variabel tetap yang digunakan antara lain:

1. Nilai *learning rate* pada model *backpropagation* sebesar **0.1**.
2. Jumlah *hidden layer* pada model *backpropagation* sebanyak **1**.
3. Jumlah iterasi maksimal sebanyak **20.000** iterasi.
4. Nilai bobot awal yang sama untuk setiap percobaan.

Adapun hasil kinerja sistem untuk masing-masing percobaan ditunjukkan sebagai berikut.

1. Sigma *canny* **0.1**.
   1. Untuk batas eror **0.1**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.1. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.1. Tabel akurasi untuk sigma canny 0.1 dan batas eror 0.1 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 59.091 | 63.636 | 68.182 | 81.818 | 77.273 |
| n\_h20 | 77.273 | 72.727 | 77.273 | 77.273 | 72.727 |
| n\_h30 | 72.727 | 68.182 | 77.273 | 81.818 | 72.727 |
| n\_h40 | 77.273 | 68.182 | 77.273 | 81.818 | 72.727 |
| n\_h50 | 72.727 | 63.636 | 77.273 | 81.818 | 72.727 |

* 1. Untuk batas eror **0.01**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.2. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.2. Tabel akurasi untuk sigma canny 0.1 dan batas eror 0.01 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 54.545 | 63.636 | 68.182 | 81.818 | 77.273 |
| n\_h20 | 77.273 | 72.727 | 77.273 | 81.818 | 72.727 |
| n\_h30 | 72.727 | 68.182 | 77.273 | 81.818 | 72.727 |
| n\_h40 | 77.273 | 68.182 | 77.273 | 81.818 | 72.727 |
| n\_h50 | 72.727 | 63.636 | 77.273 | 81.818 | 72.727 |

1. Sigma *canny* **0.2**.
   1. Untuk batas eror **0.1**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.3. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.3. Tabel akurasi untuk sigma canny 0.2 dan batas eror 0.1 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 77.273 | 72.727 | 77.273 | 72.727 | 72.727 |
| n\_h20 | 77.273 | 68.182 | 81.818 | 72.727 | 81.818 |
| n\_h30 | 63.636 | 68.182 | 81.818 | 68.182 | 77.273 |
| n\_h40 | 72.727 | 72.727 | 81.818 | 68.182 | 77.273 |
| n\_h50 | 72.727 | 63.636 | 81.818 | 72.727 | 81.818 |

* 1. Untuk batas eror **0.01**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.4. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.4. Tabel akurasi untuk sigma canny 0.2 dan batas eror 0.01 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 72.727 | 72.727 | 77.273 | 72.727 | 72.727 |
| n\_h20 | 77.273 | 63.636 | 86.364 | 72.727 | 81.818 |
| n\_h30 | 63.636 | 72.727 | 81.818 | 72.727 | 77.273 |
| n\_h40 | 68.182 | 68.182 | 81.818 | 72.727 | 77.273 |
| n\_h50 | 77.273 | 63.636 | 81.818 | 72.727 | 77.273 |

1. Sigma *canny* **0.3**.
   1. Untuk batas eror **0.1**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.5. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.5. Tabel akurasi untuk sigma canny 0.3 dan batas eror 0.1 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 77.273 | 72.727 | 72.727 | 72.727 | 81.818 |
| n\_h20 | 68.182 | 68.182 | 68.182 | 81.818 | 86.364 |
| n\_h30 | 68.182 | 68.182 | 59.091 | 81.818 | 81.818 |
| n\_h40 | 72.727 | 72.727 | 63.636 | 81.818 | 90.909 |
| n\_h50 | 72.727 | 77.273 | 63.636 | 86.364 | 90.909 |

* 1. Untuk batas eror **0.01**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.6. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.6. Tabel akurasi untuk sigma canny 0.3 dan batas eror 0.01 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 72.727 | 81.818 | 72.727 | 72.727 | 77.273 |
| n\_h20 | 68.182 | 68.182 | 63.636 | 86.364 | 81.818 |
| n\_h30 | 63.636 | 68.182 | 59.091 | 81.818 | 77.273 |
| n\_h40 | 72.727 | 72.727 | 63.636 | 86.364 | 95.455 |
| n\_h50 | 68.182 | 77.273 | 59.091 | 86.364 | 95.455 |

1. Sigma *canny* **0.4**.
   1. Untuk batas eror **0.1**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.7. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.7. Tabel akurasi untuk sigma canny 0.4 dan batas eror 0.1 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 68.182 | 81.818 | 72.727 | 63.636 | 68.182 |
| n\_h20 | 72.727 | 77.273 | 68.182 | 72.727 | 77.273 |
| n\_h30 | 77.273 | 68.182 | 72.727 | 68.182 | 77.273 |
| n\_h40 | 63.636 | 68.182 | 68.182 | 72.727 | 77.273 |
| n\_h50 | 77.273 | 72.727 | 72.727 | 68.182 | 81.818 |

* 1. Untuk batas eror **0.01**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.8. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.8. Tabel akurasi untuk sigma canny 0.4 dan batas eror 0.01 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 68.182 | 81.818 | 72.727 | 68.182 | 63.636 |
| n\_h20 | 72.727 | 72.727 | 68.182 | 72.727 | 77.273 |
| n\_h30 | 77.273 | 68.182 | 72.727 | 72.727 | 77.273 |
| n\_h40 | 59.091 | 68.182 | 68.182 | 72.727 | 72.727 |
| n\_h50 | 77.273 | 72.727 | 72.727 | 68.182 | 77.273 |

1. Sigma *canny* **0.5**.
   1. Untuk batas eror **0.1**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.9. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.9. Tabel akurasi untuk sigma canny 0.5 dan batas eror 0.1 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 72.727 | 81.818 | 77.273 | 77.273 | 81.818 |
| n\_h20 | 72.727 | 86.364 | 81.818 | 81.818 | 81.818 |
| n\_h30 | 81.818 | 72.727 | 81.818 | 86.364 | 81.818 |
| n\_h40 | 76.190 | 72.727 | 81.818 | 90.909 | 77.273 |
| n\_h50 | 68.182 | 81.818 | 81.818 | 86.364 | 77.273 |

* 1. Untuk batas eror **0.01**, diperoleh hasil seperti yang ditunjukkan pada tabel 4.10. berikut.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabel 4.10. Tabel akurasi untuk sigma canny 0.5 dan batas eror 0.01 | | | | | |
|  | pc10 | pc20 | pc30 | pc40 | pc50 |
| n\_h10 | 68.182 | 86.364 | 77.273 | 77.273 | 81.818 |
| n\_h20 | 77.273 | 86.364 | 86.364 | 81.818 | 77.273 |
| n\_h30 | 81.818 | 72.727 | 81.818 | 86.364 | 77.273 |
| n\_h40 | 72.727 | 77.273 | 81.818 | 90.909 | 77.273 |
| n\_h50 | 68.182 | 81.818 | 77.273 | 86.364 | 77.273 |

Berdasarkan tabel yang telah dipaparkan, dapat dilihat bahwa akurasi terbaik sebesar **95.455%** diperoleh pada nilai sigma *canny* 0.3, ukuran PC sebesar 50, jumlah hidden neuron 40 dan 50, serta batas eror sebesar 0.01.

## Pembahasan

### Pengaruh nilai sigma terhadap hasil kinerja sistem

Seperti yang telah dijelaskan pada bab 3, bahwa nilai sigma akan mempengaruhi nilai *low* dan *high threshold* yang digunakan untuk menentukan tepi pada citra. Hasil deteksi tepi ini yang nantinya akan digunakan sebagai PC dalam PCA dan dimasukkan ke dalam model *backpropagation*.

Pada penelitian ini, digunakan lima nilai sigma untuk melihat perbedaan yang dihasilkan pada hasil kinerja sistem, yaitu 0.1, 0.2, 0.3, 0.4, dan 0.5. Sebagai contoh, menggunakan gambar (3.14.(c)) maka nilai *low* dan *high threshold* untuk masing-masing nilai sigma dapat ditunjukkan pada tabel 4.11. berikut.

Tabel 4.11. Nilai low dan high threshold untuk gambar (3.14.(c))

|  |  |  |  |
| --- | --- | --- | --- |
| Median | Sigma | Low Threshold | High Threshold |
| 117 | 0.1 | 105 | 128 |
| 0.2 | 93 | 140 |
| 0.3 | 81 | 152 |
| 0.4 | 70 | 163 |
| 0.5 | 58 | 175 |

Berdasarkan pada tabel 4.11. di atas, dapat dilihat bahwa semakin besar nilai sigma yang digunakan, maka semakin besar rentang nilai antara *low* dan *high threshold*. Hal ini menyebabkan pixel yang dinyatakan sebagai tepi kuat dan bukan tepi akan semakin sedikit, sedangkan pixel yang dinyatakan sebagai tepi lemah akan semakin banyak. Begitu pula sebaliknya, bila nilai sigma kecil, maka jumlah tepi kuat dan bukan tepi akan semakin banyak, namun jumlah tepi lemah akan semakin sedikit. Kedua hal ini bisa menyebabkan bertambahnya tepi palsu yang terdeteksi sehingga akan mempengaruhi hasil kinerja sistem melalui fitur yang diambil oleh PCA. Oleh karena itu, untuk menentukan nilai sigma terbaik harus dilakukan proses *trial and error.*

Contoh citra hasil deteksi tepi untuk masing-masing nilai sigma dapat dilihat pada gambar 4.1. berikut.

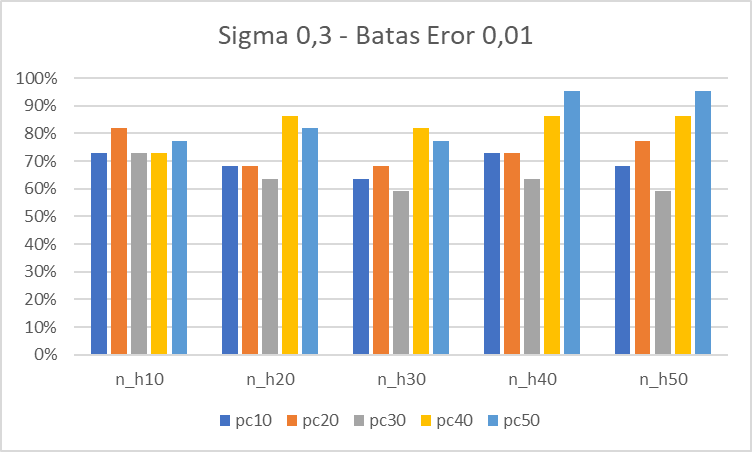
|  |  |  |
| --- | --- | --- |
| C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\868B5910.tmp | C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\D198F19E.tmp | C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\6175AF5C.tmp |
| C:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\41EF39CA.tmpC:\Users\chali\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\49FB2468.tmp | | |
| Gambar 4.1. Hasil deteksi tepi *canny* untuk masing-masing nilai sigma | | |

### 

### Pengaruh arsitektur jaringan terhadap hasil kinerja sistem

Pada penelitian ini, percobaan terhadap arsitektur jaringan yang dapat mempengaruhi hasil dari sistem adalah jumlah neuron pada *input layer* berupa ukuran PC dari fitur, jumlah *hidden layer*, dan jumlah *hidden neuron* pada *hidden layer.* Percobaan yang dilakukan pada penelitian ini hanya berfokus pada ukuran PC sebagai neuron pada *input layer* dan jumlah *hidden neuron* pada *hidden layer.*

Ukuran PC dan jumlah *hidden neuron* yang digunakan dalam percobaan yaitu bernilai 10, 20, 30, 40, dan 50 untuk masing-masing variabel. Sebagai contoh, hasil percobaan terhadap kelima nilai yang digunakan untuk sigma *canny* 0.3 dan batas eror 0.01 dapat dilihat pada gambar 4.2. berikut.



Gambar 4.2. Diagram hasil kinerja sistem untuk sigma canny 0.3 dan batas eror 0.01

Secara umum, semakin kecil ukuran PC maka jumlah fitur yang mengandung informasi akan semakin sedikit, sebaliknya semakin besar ukuran PC maka jumlah fitur juga akan semakin banyak dan memungkinkan masuknya *noise* dari citra sebagai fitur. Sedangkan untuk jumlah *hidden neuron*, semakin banyak jumlah neuron maka semakin banyak pula nilai bobot yang dapat mempengaruhi hasil keluaran sistem.

Pada contoh percobaan yang dilakukan pada gambar 4.2. di atas, dapat dilihat bahwa untuk tiap ukuran PC memiliki perubahan akurasi yang berbeda-beda di setiap jumlah *hidden neuron*. Misalnya untuk ukuran PC 30, perubahan akurasi dari jumlah *hidden neuron* 10 hingga 50 cenderung menurun, sebaliknya untuk ukuran PC 50, perubahan akurasi yang terjadi cenderung meningkat seiring meningkatnya jumlah *hidden neuron.* Sehingga dapat disimpulkan bahwa diperlukan kombinasi antara ukuran PC sebagai neuron input dan jumlah *hidden neuron* untuk memperoleh akurasi yang baik, di mana untuk memperoleh kombinasi tersebut diperlukan proses *trial and error*.

### Pengaruh nilai batas eror terhadap hasil kinerja sistem

Nilai batas eror yang digunakan dalam penelitian ini antara lain 0.1 dan 0.01. Secara umum, nilai batas eror dapat mempengaruhi akurasi dari sistem. Semakin kecil nilai batas eror yang ditentukan pada saat proses pelatihan, maka semakin besar kemungkinan sistem untuk memprediksi dengan benar. Namun nilai batas eror yang terlalu kecil juga tidak dapat menjamin kualitas kinerja sistem, karena bila nilai batas eror yang ditentukan terlalu kecil, maka memungkinkan terjadinya *overfit* pada model yang dilatih. Sebaliknya bila terlalu besar, maka model tidak dapat mengenali fitur yang terdapat pada saat pelatihan.

Peningkatan akurasi sistem dapat ditunjukkan oleh contoh percobaan pada sigma *canny* 0.3, PC 50, dan *hidden neuron* 50, seperti pada tabel 4.12 berikut.

Tabel 4.12. Hasil akurasi untuk setiap nilai batas eror

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Batas Error | TN | FP | FN | TP | Akurasi |
| 0.1 | 10 | 1 | 1 | 10 | 90.909% |
| 0.01 | **11** | **0** | **1** | **10** | **95.455%** |

# BAB V PENUTUP

## Kesimpulan

Berdasarkan hasil analisis yang dilakukan terhadap pengujian deteksi potensi penyakit jantung berdasarkan iridologi, maka dapat disimpulkan beberapa poin sebagai berikut:

1. Proses pengambilan fitur dari citra iris mata merupakan proses penting dalam menghasilkan akurasi sistem yang maksimal. Penelitian ini menggunakan metode deteksi tepi *canny* dan dekomposisi PCA untuk memperoleh fitur penting dari region jantung pada citra iris mata. Untuk deteksi tepi *canny* digunakan nilai sigma 0.1, 0.2, 0.3, 0.4, dan 0.5. Sedangkan ukuran PC yang digunakan 10, 20, 30, 40, dan 50.
2. Model prediksi dibuat menggunakan metode jaringan saraf tiruan dengan model *backpropagation,* di mana bobot awal bernilai sama untuk setiap percobaan. Parameter tetap yang digunakan dalam model ini yaitu nilai *learning rate* sebesar 0.1, jumlah *hidden layer* sebanyak 1, dan jumlah *epoch* sebanyak 20.000. Sedangkan parameter dengan nilai berubah antara lain batas eror sebesar 0.1 dan 0.01, serta jumlah *hidden neuron* pada *hidden layer* sebanyak 10, 20, 30, 40, dan 50.
3. Hasil akurasi terbaik diperoleh pada penggunaan parameter *canny* 0.3, ukuran PC 50, batas eror 0.01, dan jumlah *hidden neuron* sebanyak 40 dan 50 dengan hasil prediksi 11 *True Negative*, 0 *False Positive*, 1 *False Negative*, dan 10 *True Positive*. Sehingga nilai akurasi yang dihasilkan berdasarkan rincian tersebut adalah sebesar **95.455%**.

## Saran

Sehubungan dengan selesainya penulisan skripsi ini, penulis bermaksud menyampaikan beberapa saran kepada pembaca, antara lain:

1. Proses segmentasi iris dari citra mata dapat dibuat otomatis untuk mempermudah dalam penggunaan sistem. Dengan segmentasi otomatis ini, sistem nantinya juga dapat dibuat ke dalam bentuk aplikasi *mobile* untuk mempermudah penggunaan bagi masyarakat.
2. Sistem ini dapat dikembangkan lebih lanjut terkait variasi potensi penyakit yang dapat dideteksi oleh sistem dan meningkatkan akurasi dari sistem yang sudah ada.
3. Terkait metode yang digunakan, pembaca dapat mencoba menggunakan beberapa metode lain baik untuk pengambilan fitur maupun pembuatan model prediksi yang memungkinkan untuk meningkatkan akurasi dari sistem.

# DAFTAR PUSTAKA

*10 Conditions That Iridology Can Detect | Better Living Wellness Clinic*. (2017, Februari 20). Diambil kembali dari Better Living: https://www.betterliving.co/2017/02/20/10-conditions-that-iridology-can-detect/

Aggarwal, C. C. (2018). *Neural Networks and Deep Learning: A Textbook.* Springer International Publishing.

Anraeni, S., Nurtanio, I., & Indrabayu. (2015). Detection of Kidney Condition Using Hidden Markov Models Based on Singular Value Decomposition.

Aripin, S., Ginting, G. L., & Silalahi, N. (2017). Penerapan Metode Retinex untuk Meningkatkan Kecerahan Citra pada Hasil Screenshot.

Bansal, A., Agarwal, R., & Rk, S. (2018). Iris Images Based Pre-diagnostic Tool to Predict Obstructive Lung Diseases.

CNNIndonesia. (2018, April 26). *Penderita Meningkat, Penyakit Kardiovaskular Patut Diwaspadai*. Diambil kembali dari CNN Indonesia: https://www.cnnindonesia.com/gaya-hidup/20180426113745-255-293712/penderita-meningkat-penyakit-kardiovaskular-patut-diwaspadai

Darmawan, A., & Dwi Saptia, A. (2010). Perbandingan Metode Pergeseran Rata-Rata, Pergeseran Logaritma, dan Alpha Blending Dalam Proses Metamorfosis dari Dua Gambar Dijital.

Dawson-Howe, K. (2014). *A Practical Introduction to Computer Vision with OpenCV.* Wiley.

Dewi, A. K., Novianty, A., & Purboyo, T. W. (2016). Stomach Disorder Detection Through the Iris Image Using Backpropagation Neural Network.

Duda, R. O., Hart, P. E., & Stork, D. G. (2000). *Pattern Classification, 2nd Edition.* Wiley-Interscience.

I Smith, L. (2002). *A Tutorial on Principal Components Analysis.*

Jensen, B. (2011). *Iridology Simplified: An Introduction to the Science of Iridology and Its Relation to Nutrition.* Book Pub Co.

Kholis, I. (2015). Analisis Variasi Parameter Backpropagation Artificial Neural Network Terhadap Pengenalan Pola Data Iris.

Permatasari, L. I., Novianty, A., & Purboyo, T. W. (2016). Deteksi Gangguan Organ Jantung Menggunakan Komputerisasi Iridologi Dengan Metode Klasifikasi SVM.

Samarasinghe, S. (2007). *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition.* Taylor & Francis Group, LLC.

Wicaksono, E. S., Santoso, I., Zahra, A. A., & Isnanto, R. R. (2017). Identifikasi Kerusakan Saraf Autonomik Melalui Citra Iris Mata Menggunakan Ekstraksi Ciri Analisis Komponen Utama (PCA) dan Jaringan Saraf Tiruan Perambatan Balik.

# LAMPIRAN

## Lampiran 1. Confusion Matrix untuk batas eror 0.1

**Sigma 0.1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 7 | 4 | 5 | 6 | 59,091 |  | 10 | 6 | 5 | 3 | 8 | 63,636 |
| 20 | 7 | 4 | 1 | 10 | 77,273 |  | 20 | 8 | 3 | 3 | 8 | 72,727 |
| 30 | 7 | 4 | 2 | 9 | 72,727 |  | 30 | 7 | 4 | 3 | 8 | 68,182 |
| 40 | 7 | 4 | 1 | 10 | 77,273 |  | 40 | 7 | 4 | 3 | 8 | 68,182 |
| 50 | 7 | 4 | 2 | 9 | 72,727 |  | 50 | 6 | 5 | 3 | 8 | 63,636 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 8 | 3 | 4 | 7 | 68,182 |  | 10 | 8 | 3 | 1 | 10 | 81,818 |
| 20 | 9 | 2 | 3 | 8 | 77,273 |  | 20 | 9 | 2 | 3 | 8 | 77,273 |
| 30 | 9 | 2 | 3 | 8 | 77,273 |  | 30 | 9 | 2 | 2 | 9 | 81,818 |
| 40 | 9 | 2 | 3 | 8 | 77,273 |  | 40 | 9 | 2 | 2 | 9 | 81,818 |
| 50 | 9 | 2 | 3 | 8 | 77,273 |  | 50 | 9 | 2 | 2 | 9 | 81,818 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |
| 20 | 7 | 4 | 2 | 9 | 72,727 |  |  |  |  |  |  |  |
| 30 | 8 | 3 | 3 | 8 | 72,727 |  |  |  |  |  |  |  |
| 40 | 8 | 3 | 3 | 8 | 72,727 |  |  |  |  |  |  |  |
| 50 | 8 | 3 | 3 | 8 | 72,727 |  |  |  |  |  |  |  |

**Sigma 0.2**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 8 | 3 | 2 | 9 | 77,273 |  | 10 | 8 | 3 | 3 | 8 | 72,727 |
| 20 | 8 | 3 | 2 | 9 | 77,273 |  | 20 | 7 | 4 | 3 | 8 | 68,182 |
| 30 | 7 | 4 | 4 | 7 | 63,636 |  | 30 | 8 | 3 | 4 | 7 | 68,182 |
| 40 | 8 | 3 | 3 | 8 | 72,727 |  | 40 | 8 | 3 | 3 | 8 | 72,727 |
| 50 | 8 | 3 | 3 | 8 | 72,727 |  | 50 | 6 | 5 | 3 | 8 | 63,636 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 7 | 4 | 1 | 10 | 77,273 |  | 10 | 8 | 3 | 3 | 8 | 72,727 |
| 20 | 9 | 2 | 2 | 9 | 81,818 |  | 20 | 7 | 4 | 2 | 9 | 72,727 |
| 30 | 8 | 3 | 1 | 10 | 81,818 |  | 30 | 7 | 4 | 3 | 8 | 68,182 |
| 40 | 8 | 3 | 1 | 10 | 81,818 |  | 40 | 6 | 5 | 2 | 9 | 68,182 |
| 50 | 8 | 3 | 1 | 10 | 81,818 |  | 50 | 7 | 4 | 2 | 9 | 72,727 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 9 | 2 | 4 | 7 | 72,727 |  |  |  |  |  |  |  |
| 20 | 9 | 2 | 2 | 9 | 81,818 |  |  |  |  |  |  |  |
| 30 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |
| 40 | 8 | 3 | 2 | 9 | 77,273 |  |  |  |  |  |  |  |
| 50 | 9 | 2 | 2 | 9 | 81,818 |  |  |  |  |  |  |  |

**Sigma 0.3**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 8 | 3 | 2 | 9 | 77,273 |  | 10 | 8 | 3 | 3 | 8 | 72,727 |
| 20 | 6 | 5 | 2 | 9 | 68,182 |  | 20 | 8 | 3 | 4 | 7 | 68,182 |
| 30 | 6 | 5 | 2 | 9 | 68,182 |  | 30 | 9 | 2 | 5 | 6 | 68,182 |
| 40 | 7 | 4 | 2 | 9 | 72,727 |  | 40 | 8 | 3 | 3 | 8 | 72,727 |
| 50 | 7 | 4 | 2 | 9 | 72,727 |  | 50 | 9 | 2 | 3 | 8 | 77,273 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 6 | 5 | 1 | 10 | 72,727 |  | 10 | 8 | 3 | 3 | 8 | 72,727 |
| 20 | 7 | 4 | 3 | 8 | 68,182 |  | 20 | 9 | 2 | 2 | 9 | 81,818 |
| 30 | 5 | 6 | 3 | 8 | 59,091 |  | 30 | 9 | 2 | 2 | 9 | 81,818 |
| 40 | 7 | 4 | 4 | 7 | 63,636 |  | 40 | 8 | 3 | 1 | 10 | 81,818 |
| 50 | 6 | 5 | 3 | 8 | 63,636 |  | 50 | 10 | 1 | 2 | 9 | 86,364 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 9 | 2 | 2 | 9 | 81,818 |  |  |  |  |  |  |  |
| 20 | 9 | 2 | 1 | 10 | 86,364 |  |  |  |  |  |  |  |
| 30 | 9 | 2 | 2 | 9 | 81,818 |  |  |  |  |  |  |  |
| 40 | 10 | 1 | 1 | 10 | 90,909 |  |  |  |  |  |  |  |
| 50 | 10 | 1 | 1 | 10 | 90,909 |  |  |  |  |  |  |  |

**Sigma 0.4**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 7 | 4 | 3 | 8 | 68,182 |  | 10 | 8 | 3 | 1 | 10 | 81,818 |
| 20 | 8 | 3 | 3 | 8 | 72,727 |  | 20 | 8 | 3 | 2 | 9 | 77,273 |
| 30 | 9 | 2 | 3 | 8 | 77,273 |  | 30 | 6 | 5 | 2 | 9 | 68,182 |
| 40 | 7 | 4 | 4 | 7 | 63,636 |  | 40 | 6 | 5 | 2 | 9 | 68,182 |
| 50 | 7 | 4 | 1 | 10 | 77,273 |  | 50 | 7 | 4 | 2 | 9 | 72,727 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 9 | 2 | 4 | 7 | 72,727 |  | 10 | 8 | 3 | 5 | 6 | 63,636 |
| 20 | 8 | 3 | 4 | 7 | 68,182 |  | 20 | 9 | 2 | 4 | 7 | 72,727 |
| 30 | 9 | 2 | 4 | 7 | 72,727 |  | 30 | 9 | 2 | 5 | 6 | 68,182 |
| 40 | 8 | 3 | 4 | 7 | 68,182 |  | 40 | 9 | 2 | 4 | 7 | 72,727 |
| 50 | 9 | 2 | 4 | 7 | 72,727 |  | 50 | 9 | 2 | 5 | 6 | 68,182 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 9 | 2 | 5 | 6 | 68,182 |  |  |  |  |  |  |  |
| 20 | 11 | 0 | 5 | 6 | 77,273 |  |  |  |  |  |  |  |
| 30 | 10 | 1 | 4 | 7 | 77,273 |  |  |  |  |  |  |  |
| 40 | 10 | 1 | 4 | 7 | 77,273 |  |  |  |  |  |  |  |
| 50 | 11 | 0 | 4 | 7 | 81,818 |  |  |  |  |  |  |  |

**Sigma 0.5**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 7 | 4 | 2 | 9 | 72,727 |  | 10 | 9 | 2 | 2 | 9 | 81,818 |
| 20 | 7 | 4 | 2 | 9 | 72,727 |  | 20 | 10 | 1 | 2 | 9 | 86,364 |
| 30 | 9 | 2 | 2 | 9 | 81,818 |  | 30 | 9 | 2 | 4 | 7 | 72,727 |
| 40 | 8 | 3 | 2 | 8 | 76,190 |  | 40 | 8 | 3 | 3 | 8 | 72,727 |
| 50 | 7 | 4 | 3 | 8 | 68,182 |  | 50 | 9 | 2 | 2 | 9 | 81,818 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 8 | 3 | 2 | 9 | 77,273 |  | 10 | 9 | 2 | 3 | 8 | 77,273 |
| 20 | 8 | 3 | 1 | 10 | 81,818 |  | 20 | 9 | 2 | 2 | 9 | 81,818 |
| 30 | 8 | 3 | 1 | 10 | 81,818 |  | 30 | 9 | 2 | 1 | 10 | 86,364 |
| 40 | 8 | 3 | 1 | 10 | 81,818 |  | 40 | 9 | 2 | 0 | 11 | 90,909 |
| 50 | 8 | 3 | 1 | 10 | 81,818 |  | 50 | 9 | 2 | 1 | 10 | 86,364 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 9 | 2 | 2 | 9 | 81,818 |  |  |  |  |  |  |  |
| 20 | 9 | 2 | 2 | 9 | 81,818 |  |  |  |  |  |  |  |
| 30 | 10 | 1 | 3 | 8 | 81,818 |  |  |  |  |  |  |  |
| 40 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |
| 50 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |

## Lampiran 2. Confusion Matrix untuk batas eror 0.01

**Sigma 0.1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 6 | 5 | 5 | 6 | 54,545 |  | 10 | 6 | 5 | 3 | 8 | 63,636 |
| 20 | 7 | 4 | 1 | 10 | 77,273 |  | 20 | 8 | 3 | 3 | 8 | 72,727 |
| 30 | 7 | 4 | 2 | 9 | 72,727 |  | 30 | 7 | 4 | 3 | 8 | 68,182 |
| 40 | 7 | 4 | 1 | 10 | 77,273 |  | 40 | 7 | 4 | 3 | 8 | 68,182 |
| 50 | 7 | 4 | 2 | 9 | 72,727 |  | 50 | 6 | 5 | 3 | 8 | 63,636 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 8 | 3 | 4 | 7 | 68,182 |  | 10 | 8 | 3 | 1 | 10 | 81,818 |
| 20 | 9 | 2 | 3 | 8 | 77,273 |  | 20 | 9 | 2 | 2 | 9 | 81,818 |
| 30 | 9 | 2 | 3 | 8 | 77,273 |  | 30 | 9 | 2 | 2 | 9 | 81,818 |
| 40 | 9 | 2 | 3 | 8 | 77,273 |  | 40 | 9 | 2 | 2 | 9 | 81,818 |
| 50 | 9 | 2 | 3 | 8 | 77,273 |  | 50 | 9 | 2 | 2 | 9 | 81,818 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |
| 20 | 7 | 4 | 2 | 9 | 72,727 |  |  |  |  |  |  |  |
| 30 | 8 | 3 | 3 | 8 | 72,727 |  |  |  |  |  |  |  |
| 40 | 8 | 3 | 3 | 8 | 72,727 |  |  |  |  |  |  |  |
| 50 | 8 | 3 | 3 | 8 | 72,727 |  |  |  |  |  |  |  |

**Sigma 0.2**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 8 | 3 | 3 | 8 | 72,727 |  | 10 | 8 | 3 | 3 | 8 | 72,727 |
| 20 | 8 | 3 | 2 | 9 | 77,273 |  | 20 | 6 | 5 | 3 | 8 | 63,636 |
| 30 | 7 | 4 | 4 | 7 | 63,636 |  | 30 | 8 | 3 | 3 | 8 | 72,727 |
| 40 | 7 | 4 | 3 | 8 | 68,182 |  | 40 | 7 | 4 | 3 | 8 | 68,182 |
| 50 | 8 | 3 | 2 | 9 | 77,273 |  | 50 | 6 | 5 | 3 | 8 | 63,636 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 7 | 4 | 1 | 10 | 77,273 |  | 10 | 8 | 3 | 3 | 8 | 72,727 |
| 20 | 10 | 1 | 2 | 9 | 86,364 |  | 20 | 7 | 4 | 2 | 9 | 72,727 |
| 30 | 8 | 3 | 1 | 10 | 81,818 |  | 30 | 8 | 3 | 3 | 8 | 72,727 |
| 40 | 8 | 3 | 1 | 10 | 81,818 |  | 40 | 7 | 4 | 2 | 9 | 72,727 |
| 50 | 8 | 3 | 1 | 10 | 81,818 |  | 50 | 7 | 4 | 2 | 9 | 72,727 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 9 | 2 | 4 | 7 | 72,727 |  |  |  |  |  |  |  |
| 20 | 9 | 2 | 2 | 9 | 81,818 |  |  |  |  |  |  |  |
| 30 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |
| 40 | 8 | 3 | 2 | 9 | 77,273 |  |  |  |  |  |  |  |
| 50 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |

**Sigma 0.3**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 8 | 3 | 3 | 8 | 72,727 |  | 10 | 8 | 3 | 1 | 10 | 81,818 |
| 20 | 6 | 5 | 2 | 9 | 68,182 |  | 20 | 8 | 3 | 4 | 7 | 68,182 |
| 30 | 5 | 6 | 2 | 9 | 63,636 |  | 30 | 9 | 2 | 5 | 6 | 68,182 |
| 40 | 7 | 4 | 2 | 9 | 72,727 |  | 40 | 8 | 3 | 3 | 8 | 72,727 |
| 50 | 6 | 5 | 2 | 9 | 68,182 |  | 50 | 9 | 2 | 3 | 8 | 77,273 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 6 | 5 | 1 | 10 | 72,727 |  | 10 | 8 | 3 | 3 | 8 | 72,727 |
| 20 | 6 | 5 | 3 | 8 | 63,636 |  | 20 | 10 | 1 | 2 | 9 | 86,364 |
| 30 | 5 | 6 | 3 | 8 | 59,091 |  | 30 | 9 | 2 | 2 | 9 | 81,818 |
| 40 | 7 | 4 | 4 | 7 | 63,636 |  | 40 | 9 | 2 | 1 | 10 | 86,364 |
| 50 | 5 | 6 | 3 | 8 | 59,091 |  | 50 | 10 | 1 | 2 | 9 | 86,364 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 8 | 3 | 2 | 9 | 77,273 |  |  |  |  |  |  |  |
| 20 | 8 | 3 | 1 | 10 | 81,818 |  |  |  |  |  |  |  |
| 30 | 8 | 3 | 2 | 9 | 77,273 |  |  |  |  |  |  |  |
| 40 | 11 | 0 | 1 | 10 | 95,455 |  |  |  |  |  |  |  |
| 50 | 11 | 0 | 1 | 10 | 95,455 |  |  |  |  |  |  |  |

**Sigma 0.4**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 7 | 4 | 3 | 8 | 68,182 |  | 10 | 8 | 3 | 1 | 10 | 81,818 |
| 20 | 8 | 3 | 3 | 8 | 72,727 |  | 20 | 8 | 3 | 3 | 8 | 72,727 |
| 30 | 9 | 2 | 3 | 8 | 77,273 |  | 30 | 6 | 5 | 2 | 9 | 68,182 |
| 40 | 7 | 4 | 5 | 6 | 59,091 |  | 40 | 6 | 5 | 2 | 9 | 68,182 |
| 50 | 7 | 4 | 1 | 10 | 77,273 |  | 50 | 7 | 4 | 2 | 9 | 72,727 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 9 | 2 | 4 | 7 | 72,727 |  | 10 | 8 | 3 | 4 | 7 | 68,182 |
| 20 | 8 | 3 | 4 | 7 | 68,182 |  | 20 | 9 | 2 | 4 | 7 | 72,727 |
| 30 | 9 | 2 | 4 | 7 | 72,727 |  | 30 | 10 | 1 | 5 | 6 | 72,727 |
| 40 | 8 | 3 | 4 | 7 | 68,182 |  | 40 | 9 | 2 | 4 | 7 | 72,727 |
| 50 | 9 | 2 | 4 | 7 | 72,727 |  | 50 | 9 | 2 | 5 | 6 | 68,182 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 8 | 3 | 5 | 6 | 63,636 |  |  |  |  |  |  |  |
| 20 | 11 | 0 | 5 | 6 | 77,273 |  |  |  |  |  |  |  |
| 30 | 10 | 1 | 4 | 7 | 77,273 |  |  |  |  |  |  |  |
| 40 | 9 | 2 | 4 | 7 | 72,727 |  |  |  |  |  |  |  |
| 50 | 10 | 1 | 4 | 7 | 77,273 |  |  |  |  |  |  |  |

**Sigma 0.5**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCA 10 | | | | | |  | PCA 20 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 6 | 5 | 2 | 9 | 68,182 |  | 10 | 10 | 1 | 2 | 9 | 86,364 |
| 20 | 8 | 3 | 2 | 9 | 77,273 |  | 20 | 10 | 1 | 2 | 9 | 86,364 |
| 30 | 9 | 2 | 2 | 9 | 81,818 |  | 30 | 9 | 2 | 4 | 7 | 72,727 |
| 40 | 7 | 4 | 2 | 9 | 72,727 |  | 40 | 9 | 2 | 3 | 8 | 77,273 |
| 50 | 7 | 4 | 3 | 8 | 68,182 |  | 50 | 9 | 2 | 2 | 9 | 81,818 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 30 | | | | | |  | PCA 40 | | | | | |
| n\_h | TN | FP | FN | TP | Akurasi |  | n\_h | TN | FP | FN | TP | Akurasi |
| 10 | 8 | 3 | 2 | 9 | 77,273 |  | 10 | 9 | 2 | 3 | 8 | 77,273 |
| 20 | 8 | 3 | 0 | 11 | 86,364 |  | 20 | 9 | 2 | 2 | 9 | 81,818 |
| 30 | 8 | 3 | 1 | 10 | 81,818 |  | 30 | 9 | 2 | 1 | 10 | 86,364 |
| 40 | 8 | 3 | 1 | 10 | 81,818 |  | 40 | 9 | 2 | 0 | 11 | 90,909 |
| 50 | 8 | 3 | 2 | 9 | 77,273 |  | 50 | 9 | 2 | 1 | 10 | 86,364 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| PCA 50 | | | | | |  |  |  |  |  |  |  |
| n\_h | TN | FP | FN | TP | Akurasi |  |  |  |  |  |  |  |
| 10 | 9 | 2 | 2 | 9 | 81,818 |  |  |  |  |  |  |  |
| 20 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |
| 30 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |
| 40 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |
| 50 | 9 | 2 | 3 | 8 | 77,273 |  |  |  |  |  |  |  |