Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

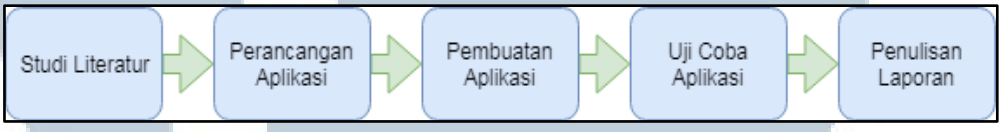
This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

BAB III

METODOLOGI DAN PERANCANGAN APLIKASI

3.1 Metodologi Penelitian

Metodologi penelitian yang digunakan dalam implementasi metode *Image Enhancement* dan algoritma *Backpropagation* pada aplikasi autentikasi sidik jari digambarkan pada Gambar 3.1. Adapun penjelasan dari tiap tahap penelitian ini adalah sebagai berikut.



Gambar 3.1 Metodologi Penelitian yang Digunakan

1. Studi Literatur

Studi literatur dilakukan sebagai pembelajaran dan pencarian ilmu yang mendukung dalam pembuatan aplikasi autentikasi sidik jari. Ilmu yang dimaksud yaitu ilmu tentang sidik jari, penangkapan gambar, pengolahan gambar, pengambilan ciri fitur gambar, dan pengenalan gambar dengan *backpropagation*.

2. Perancangan Aplikasi

Tahap awal perancangan aplikasi ini dengan menganalisis kebutuhan yang diperlukan dalam membuat aplikasi ini. Setelah itu dilanjutkan dengan pembuatan model aplikasi agar memudahkan penjelasan dari komponen aplikasi yang dibuat. Setelah model aplikasi, dilanjutkan dengan membuat *flowchart* agar proses jalannya alur aplikasi dapat dipahami dengan mudah. Selanjutnya membuat tampilan antarmuka aplikasi agar memudahkan dalam membayangkan gambaran dari aplikasi yang akan dibuat.

Implementasi Metode Image..., Bryan, FTI UMN, 2018

3. Pembuatan Aplikasi

Pembuatan aplikasi autentikasi sidik jari ini berbasis *desktop* dan menggunakan bahasa pemrograman C#.

4. Uji Coba Aplikasi

Selesai membuat aplikasi, sudah saatnya untuk mengujicobakan aplikasi yang dibuat. Data yang digunakan merupakan data *sample* yang diambil dengan *scanner*. Karena aplikasi ini dibuat untuk mengautentikasi seseorang, nilai yang ingin didapatkan berupa nilai *false acceptance rate* dan *false rejection rate*. Ketika seseorang melakukan uji coba, bagian aplikasi ini akan memunculkan nama dari seseorang tersebut, yang sebelumnya telah melakukan registrasi. Jika saat uji coba memunculkan nama seseorang yang lain dan seseorang tersebut sudah melakukan registrasi di dalam aplikasi, maka kejadian ini disebut dengan *false acceptance*. Jika saat uji coba tidak memunculkan nama seseorang dan seseorang tersebut sudah melakukan registrasi di dalam aplikasi, maka kejadian ini disebut dengan *false rejection*.

5. Penulisan Laporan

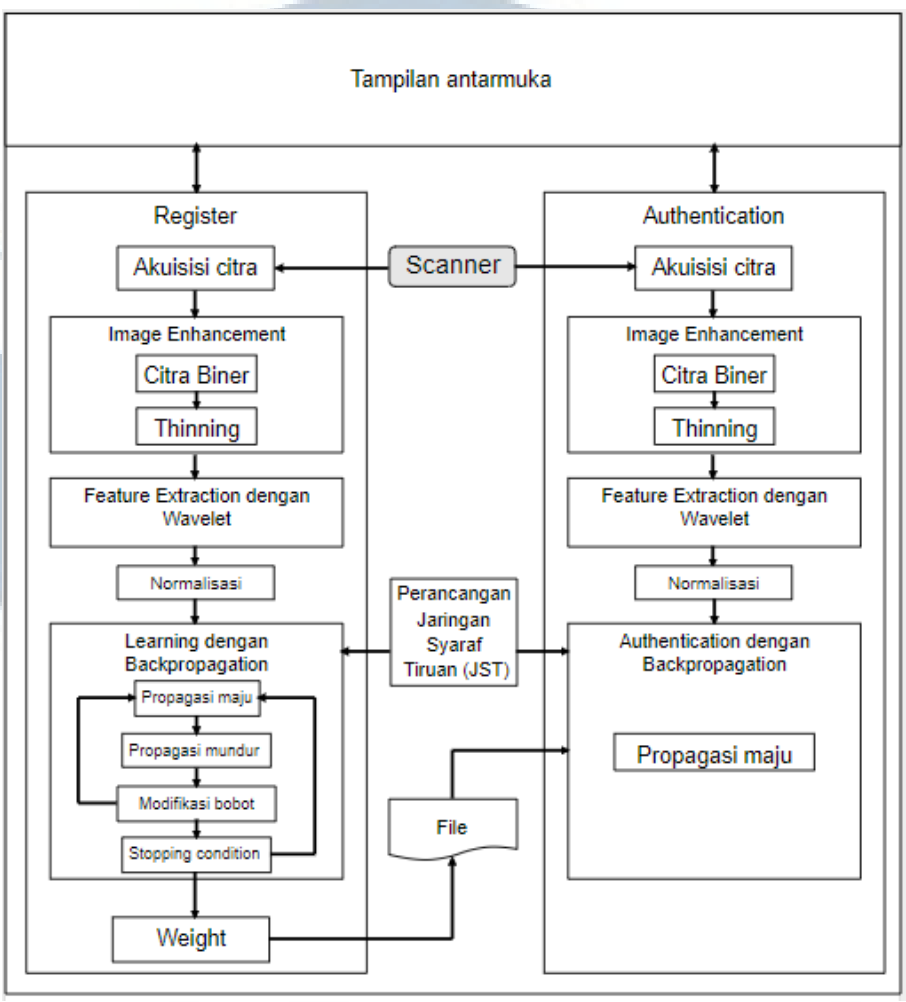
Dari hasil yang dibuat dan didapatkan, semuanya dituangkan ke dalam laporan sebagai bukti bahwa telah melakukan penelitian.

3.2 Perancangan Aplikasi

Aplikasi dirancang dengan menggunakan model aplikasi, *flowchart*, dan desain antarmuka.

UNIVERSITAS MULTIMEDIA NUSANTARA

3.2.1 Model Aplikasi



Gambar 3.2 Model Aplikasi Autentikasi Sidik Jari

Gambar 3.2 memperlihatkan model aplikasi autentikasi sidik jari yang dibuat. Proses Register dan *Authentication* ditampilkan dalam tampilan antarmuka pada menu Register dan menu *Authentication* dalam aplikasi ini. Proses registrasi secara berurutan terdiri dari akuisisi citra, *image enhancement*, *feature extraction*, normalisasi, dan *learning*. Bagian *image enhancement* terdiri dari dua tahap yaitu citra biner dan *thinning*. Bagian *learning* dengan menggunakan *backpropagation* terdiri dari tiga tahap yaitu propagasi maju, propagasi mundur, dan modifikasi bobot. Setiap *epoch*-nya akan dilakukan pemeriksaan kondisi berhenti dari

backpropagation. Jika kondisi berhenti sudah terpenuhi, maka bobot akan disimpan di *file*. Proses *authentication* hampir mirip dengan proses registrasi, secara berurutan terdiri dari akuisisi citra, *image enhancement*, *feature extraction*, normalisasi, dan *authentication*. Autentikasi dilakukan dengan menggunakan data bobot yang ada di *file*. Bagian autentikasi menggunakan satu tahap saja pada *backpropagation* yaitu propagasi maju. Penggunaan *backpropagation*, sebelumnya diperlukan perancangan jaringan syaraf tiruan terlebih dahulu. Perancangan ini bertujuan untuk mengetahui jumlah *node* yang digunakan dalam jaringan dan juga nilai-nilai dari variabel jaringan syaraf tiruan yang diperlukan.

Berikut ini adalah tahap-tahap perancangan jaringan syaraf tiruan yang akan digunakan dalam aplikasi ini.

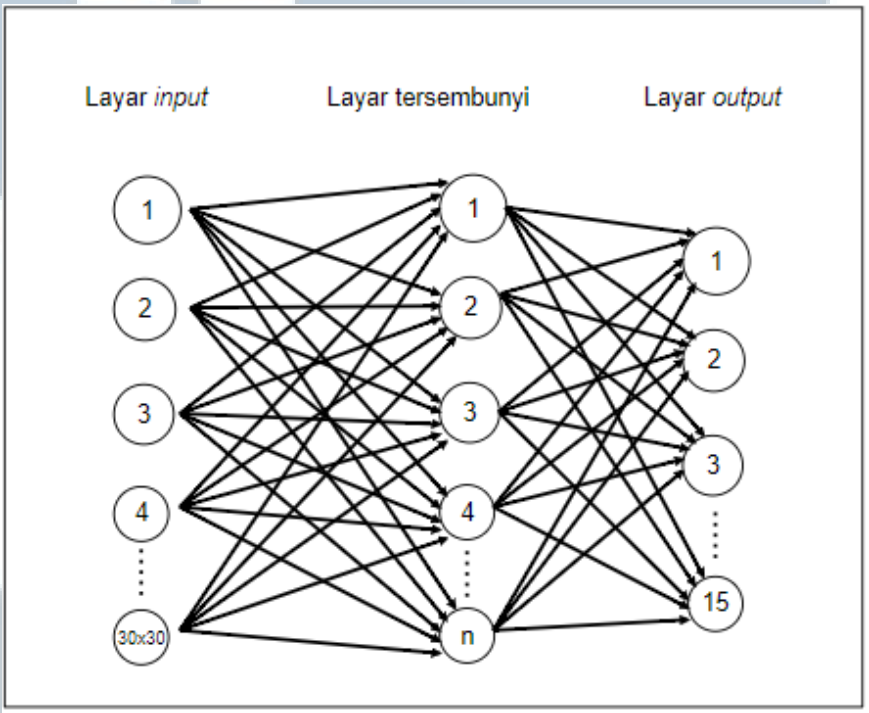
a. Jaringan syaraf tiruan *backpropagation* terdiri dari tiga buah layar seperti yang dijelaskan pada Gambar 3.3.
b. Jumlah masukkan pada layar *input* pada aplikasi ini ditetapkan dengan hasil dari ekstraksi ciri yang didapatkan dari gambar yang diinputkan adalah 30 x 30 buah.
c. Jumlah keluaran yang ada pada jaringan merupakan 15 buah untuk memetakan sebanyak 15 orang. Contoh pemetaannya dapat dilihat pada Tabel 3.1.
d. Layar tersembunyi yang dicoba berjumlah satu buah dan jumlah *node* pada layar tersembunyi akan ditentukan setelah proses pencarian arsitektur jaringan optimum. Sebagai awalan jumlah *node* pada layar tersembunyi akan ditentukan dengan cara mengakarkan hasil perkalian jumlah *node input* dengan jumlah *node output*. Jumlah *node* pada layar tersembunyi yang diuji berjumlah 120, 240, dan 480 buah.

Implementasi Metode Image..., Bryan, FTI UMN, 2018

e. Nilai *learning rate* yang akan diuji sebesar 0.01, 0.1, 0.2, 0.3, 0.4 dan *maximum error* sebesar 0.0001.

f. Data sidik jari diambil dari 15 orang dengan setiap orangnya berjumlah 20 data. Dari 20 data gambar sidik jari, 15 gambar sebagai data pelatihan dan 5 gambar sebagai data uji coba untuk setiap orangnya.

Gambar 3.3 di bawah ini merupakan gambar dari perancangan jaringan syaraf tiruan pada aplikasi autentikasi sidik jari ini.



Gambar 3.3 Perancangan Jaringan Syaraf Tiruan

Tabel 3.1 di bawah ini merupakan contoh pemetaan *output* pada perancangan jaringan syaraf tiruan.

Tabel 3.1 Contoh Pemetaan Output

Sidik jari orang ke-	**Pemetaan output**
1	100000000000000
2	010000000000000
3	001000000000000
4	000100000000000
5	000010000000000

Tabel 3.1 Contoh Pemetaan Output (Lanjutan)

Sidik jari orang ke-	Pemetaan output
6	000001000000000
7	000000100000000
8	000000010000000
9	000000001000000
10	000000000100000
11	000000000010000
12	000000000001000
13	000000000000100
14	000000000000010
15	000000000000001

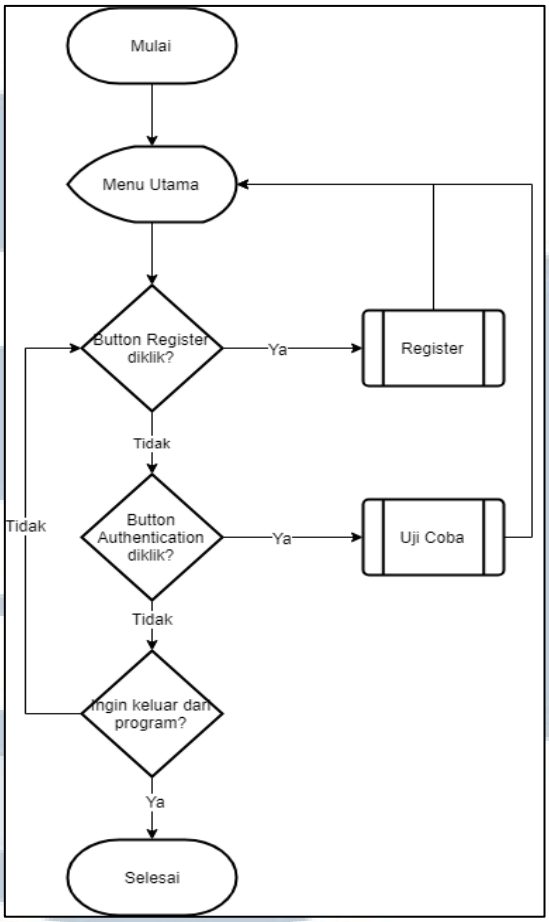
3.2.2 Flowchart

Berikut merupakan flowchart dari aplikasi autentikasi sidik jari ini.

A. Flowchart Aplikasi

Gambar 3.4 adalah *flowchart* aplikasi autentikasi sidik jari ini. Pertama kali saat aplikasi ini dijalankan, *user* akan dihadapi dengan tiga pilihan yaitu tombol Register, tombol *Authentication*, dan tombol *Exit*. Tombol Register berfungsi untuk menuju ke halaman Register. Di halaman Register, *user* bisa meregistrasikan sidik jarinya untuk diuji pada halaman *Authentication*. Tombol *Authentication* berfungsi untuk menuju ke halaman *Authentication*. Di halaman *Authentication*, *user* bisa menguji gambar sidik jari yang telah di dapatkan sebelumnya. Tombol *Exit* berfungsi untuk keluar dari aplikasi.

UNIVERSITAS MULTIMEDIA NUSANTARA



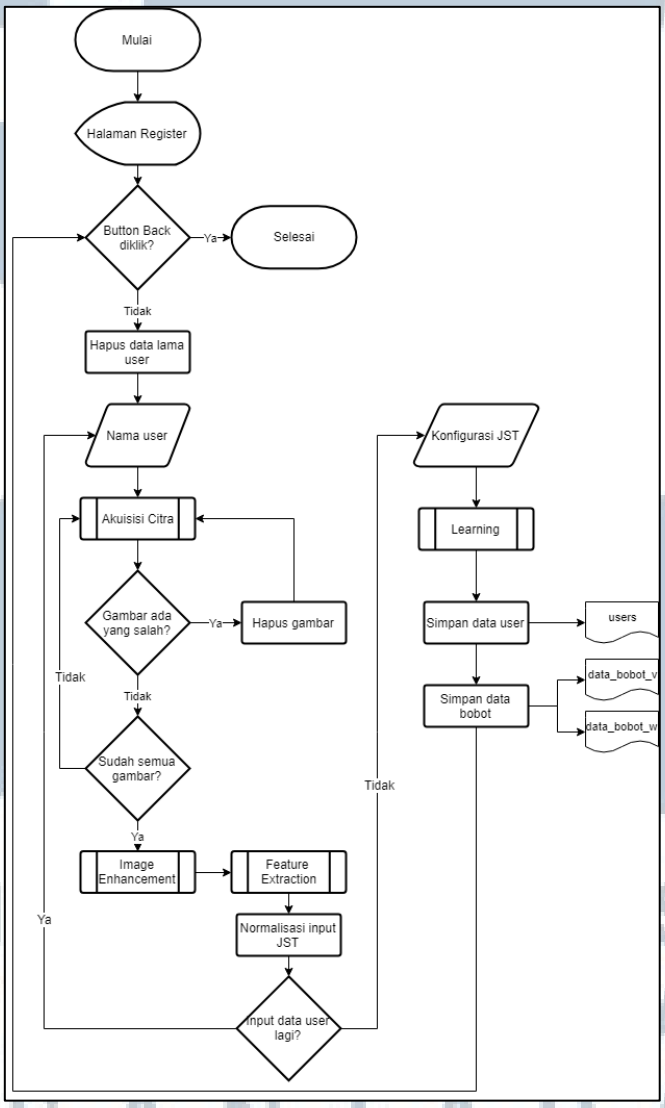
Gambar 3.4 Flowchart Aplikasi

B. Flowchart Register

Gambar 3.5 merupakan *flowchart* register. Tahap register dalam aplikasi ini hanya akan dilakukan sekali ketika selesai mendapatkan bobot optimal yang akan digunakan pada tahap uji coba. Jika ada data baru yang ingin di registrasi maka proses register diulang dari awal data kembali. Tahap pertama yaitu menghapus penampung seluruh data *user* yang pernah tersimpan di dalam *file*. Selesai menghapus, *user* akan diminta untuk memasukan namanya dan juga sidik jarinya sebagai identitas diri. Gambar yang diinputkan akan sesuai dengan jumlah data yang akan di *learning*. Ketika setiap data *user* di simpan untuk dijadikan data *learning*, aplikasi akan menampilkan gambar dari tahapan proses *image*

Implementasi Metode Image..., Bryan, FTI UMN, 2018

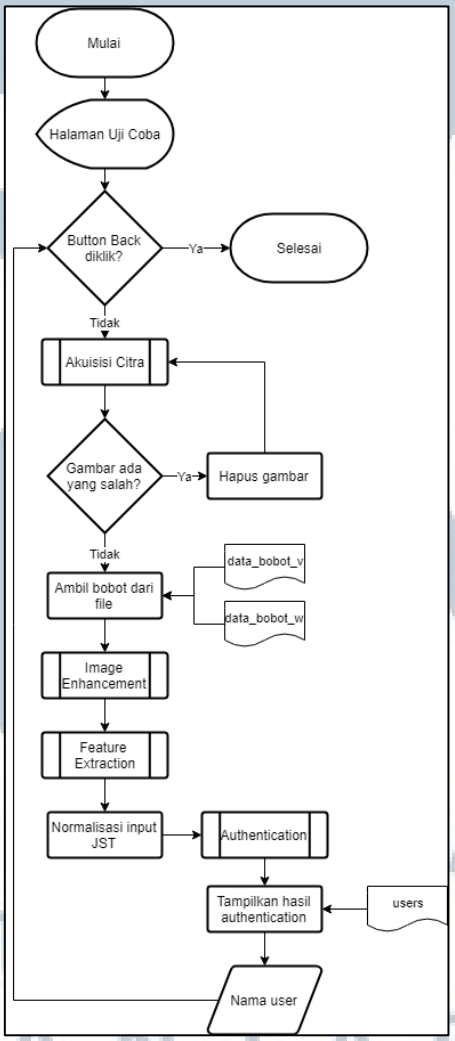
enhancement dan memberikan *output message*. Selesai memasukkan data gambar untuk di *learning*, *user* diminta untuk memberikan input yang akan digunakan sebagai konfigurasi JST berupa jumlah *node hidden*, nilai *learning rate*, nilai *max error*, dan jumlah *max epoch*. Jika sudah selesai memberikan input, aplikasi akan memulai proses *learning* dari seluruh data yang di registrasikan. Ketika proses learning selesai, aplikasi akan langsung menyimpan data *user* dan bobot optimal ke dalam *file*. Bobot optimal ini yang nantinya akan digunakan kembali pada tahap uji coba. Untuk kembali ke halaman Menu Utama, *user* bisa menekan tombol Back.



Gambar 3.5 Flowchart Register

C. Flowchart Uji Coba

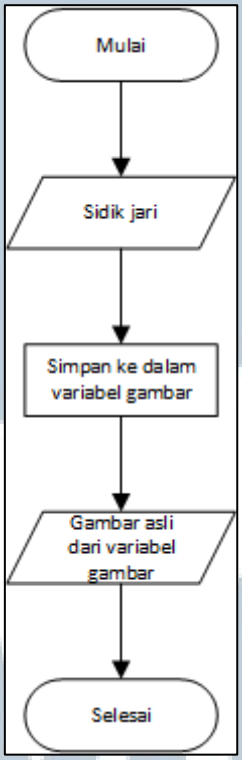
Gambar 3.6 merupakan *flowchart* uji coba. Hampir mirip dengan *flowchart* register pada Gambar 3.5, yang membedakan adalah bagian *authentication* dan menampilkan *output* berupa nama *user*. Pada bagian *authentication*, sistem berusaha untuk mengenali siapa pemilik dari sidik jari yang diinputkan ke dalam aplikasi. Informasi mengenai pemiliki sidik jari akan diambil dari *file* dan ditampilkan pada halaman *Authentication*. Uji coba bisa dilakukan berkali-kali sesuai keinginan. Untuk kembali ke halaman Menu Utama, *user* bisa menekan tombol Back.



Gambar 3.6 Flowchart Uji Coba

D. Flowchart Akuisisi Citra

Gambar 3.7 merupakan *flowchart* akuisisi citra. Ketika tahap register atau tahap uji coba dijalankan, *user* diminta untuk memberikan input berupa sidik jarinya dengan membuka file gambar sidik jari yang telah didapatkan sebelumnya. Setelah diberikan input, sistem akan menyimpan data sidik jari tersebut ke dalam variabel dan variabel tersebut ditampilkan agar terlihat gambar sidik jari yang terinput.



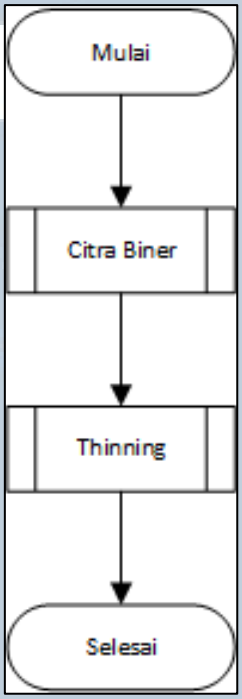
Gambar 3.7 Flowchart Akuisisi Citra

E. Flowchart Image Enhancement

Gambar 3.8 merupakan *flowchart image enhancement*. Pada tahap ini, gambar asli akan diperbaiki agar memiliki kualitas gambar yang lebih baik. Tahapannya terbagi menjadi dua, yaitu citra biner dan *thinning*. Citra biner

Implementasi Metode Image..., Bryan, FTI UMN, 2018

bertujuan untuk mengubah warna sehinga didapatkan dua warna yaitu hitam dan putih. *Thinning* bertujuan untuk mengurangi garis-garis yang tidak diperlukan.

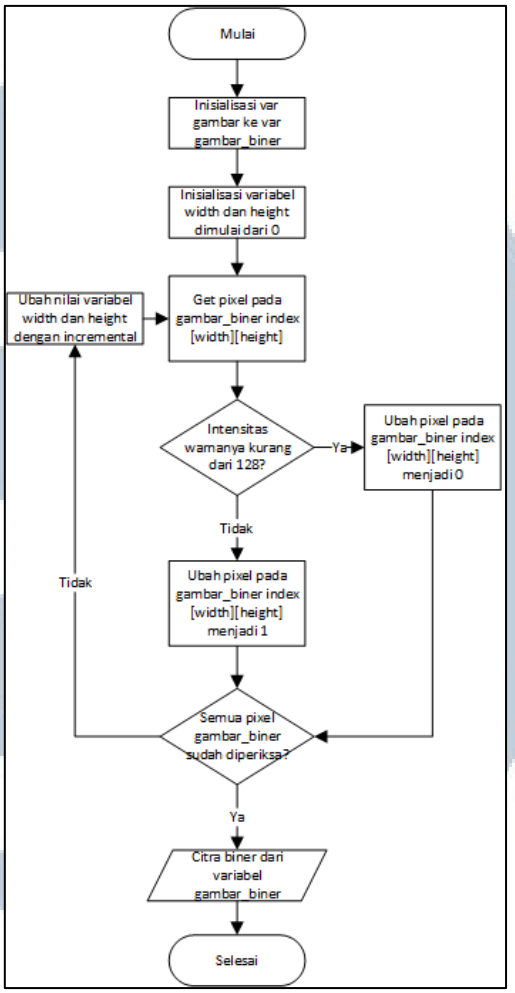


Gambar 3.8 Flowchart Image Enhancement

F. Flowchart Citra Biner

Gambar 3.9 merupakan *flowchart* citra biner. Pada tahap ini, sistem akan mengubah warna gambar asli menjadi dua warna yaitu hitam dan putih. Caranya dengan memeriksa seluruh *pixel* dari gambar asli. Setiap *pixel* akan diperiksa intensitas warnanya. Jika intensitas warnanya kurang dari 128, maka *pixel* tersebut akan diubah menjadi warna hitam (nilainya 0). Jika intensitas warnanya lebih dari atau sama dengan 128, maka *pixel* tersebut akan diubah menjadi warna putih (nilainya 1). Proses pemeriksaan akan terus berlanjut hingga semua *pixel* berhasil diperiksa. Kemudian hasil gambar dari tahap ini akan ditampilkan.

Implementasi Metode Image..., Bryan, FTI UMN, 2018



Gambar 3.9 Flowchart Citra Biner

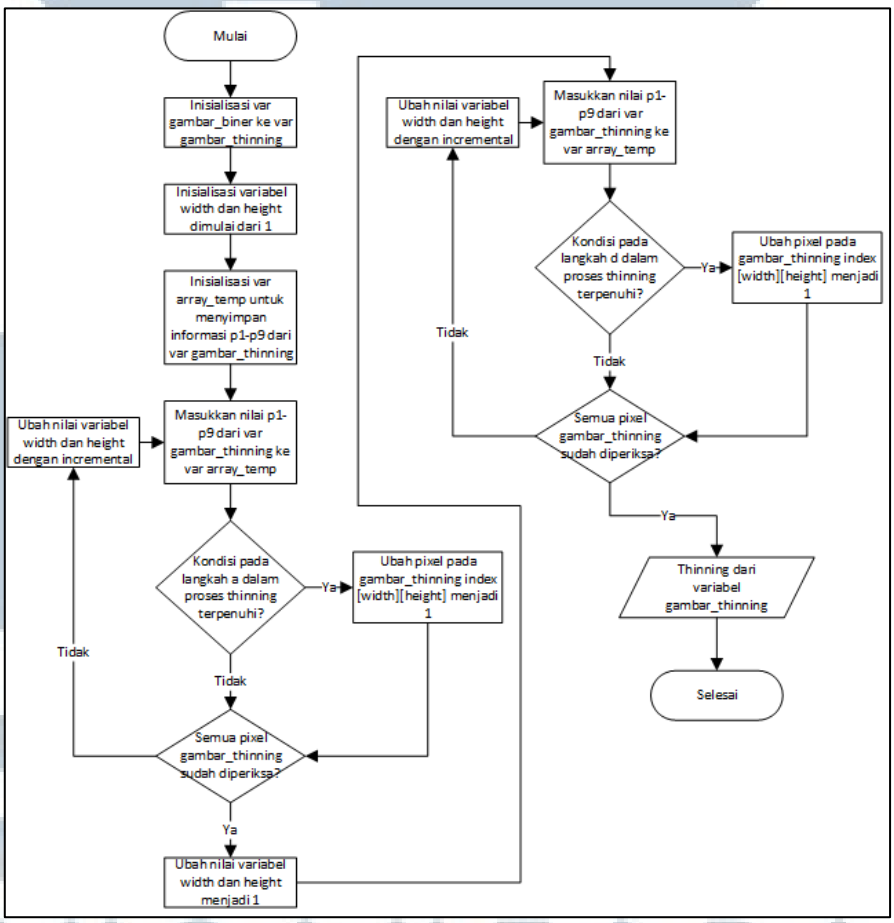
G. Flowchart Thinning

Gambar 3.10 merupakan *flowchart thinning*. Gambar yang telah diolah pada tahap citra biner akan diteruskan ke tahap ini. Dalam tahap ini, sistem akan menghapus sebagian titik yang tidak diperlukan. Caranya dengan memeriksa seluruh *pixel* bagian dalam gambar, kecuali *pixel* bagian luar gambar atau sisi gambar. Tahap pemeriksaan *pixel* ini terjadi dua kali.

Tahap pertama, *pixel* yang diperiksa dan sekeliling dari *pixel* yang diperiksa akan dimasukan ke dalam variabel terlebih dahulu (pixel[i][j] = p1, pixel[i-1][j] = p2, pixel[i-1][j+1] = p3, pixel[i][j+1] = p4, pixel[i+1][j+1] = p5, pixel[i+1][j] = p6,

pixel[i+1][j-1] = p7, pixel[i][j-1] = p8, pixel[i-1][j-1] = p9). Dari variabel tersebut akan dicocokkan dengan kondisi yang terdapat pada Subbab 2.3 bagian *thinning* langkah a. Jika kondisinya terpenuhi maka *pixel* yang diperiksa (variabel p1) akan dirubah menjadi 1 (warna putih). Jika kondisinya tidak terpenuhi maka sistem akan meneruskan ke *pixel* selanjutnya untuk diperiksa. Setelah seluruh *pixel* bagian dalam gambar telah diperiksa, selanjutnya akan dilanjutkan ke tahap kedua.

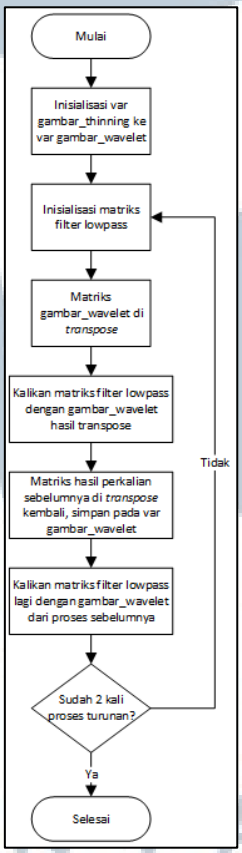
Tahap kedua hampir mirip dengan tahap pertama, bedanya pada bagian kondisi menggunakan kondisi yang terdapat pada subbab 2.3 bagian *thinning* langkah d. Proses pemeriksaan kembali dari awal *pixel* bagian dalam gambar. Ketika selesai, sistem akan menampilkan gambar hasil *thinning*.



Gambar 3.10 Flowchart Thinning

H. Flowchart Feature Extraction

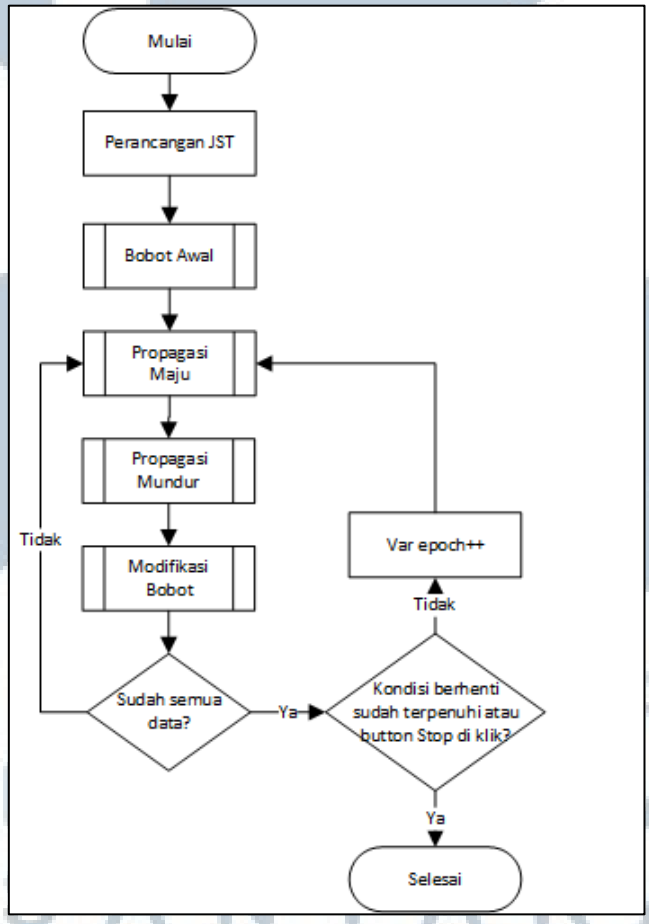
Gambar 3.11 merupakan *flowchart feature extraction* dengan menggunakan *wavelet*. Pada tahap ini, gambar yang telah diperbaiki akan diambil ciri gambarnya. Caranya dengan mengalikan matriks filter dengan matriks gambar. Sistem akan menginisialisasikan matriks filternya. Kemudian gambar akan ditransposekan terlebih dahulu untuk memudahkan proses perkalian. Kalikan matriks filter *lowpass* dengan matriks gambar. Hasil perkaliannya akan menghasilkan matriks yang baru. Hasil matriks tersebut ditransposekan kembali untuk memudahkan proses perkalian. Lakukan perkalian lagi dengan mengalikan matriks filter *lowpass* dengan matriks baru. Proses dilakukan sebanyak 2 kali turunan *wavelet*. Hasil yang didapat dari proses ini merupakan angka dari filter *sub-band* LL yang akan digunakan sebagai input untuk proses selanjutnya.



Gambar 3.11 Flowchart Feature Extraction

I. Flowchart Learning

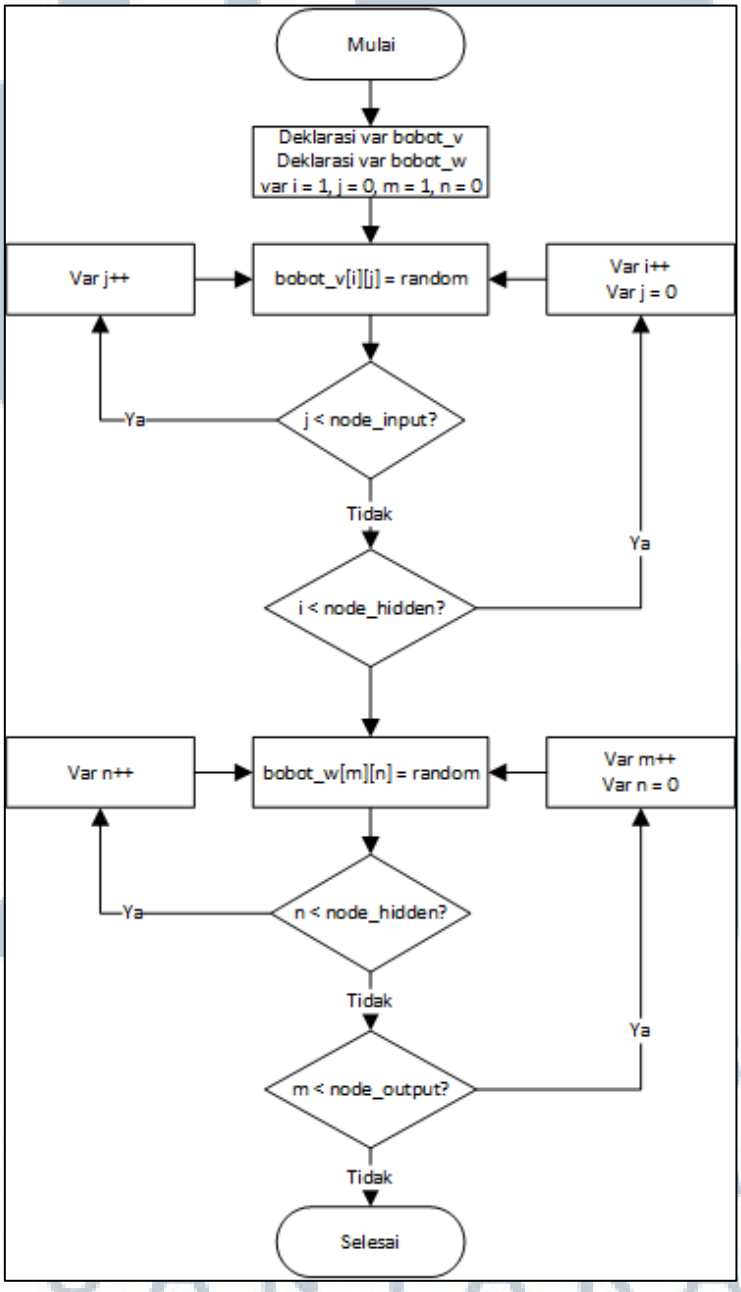
Gambar 3.12 merupakan *flowchart learning*. Pada tahap ini, sistem akan berusaha untuk mencari bobot optimal yang didapat dari input yang diberikan. Caranya dengan melakukan algoritma *backpropagation*. Pertama-tama diinisialisasikan terlebih dahulu perancangan jaringan syaraf tiruannya untuk mendapatkan beberapa variabel yang akan digunakan. Kemudian dengan penentuan bobot awal secara acak, lalu masuk ke tahap propagasi maju. Selesai tahap propagasi maju, tahapan akan dilanjutkan ke tahap propagasi mundur, modifikasi bobot, dan akan berulang kembali ke tahap propagasi maju sampai seluruh data *learning* dipelajari. Setiap seluruh data learning, dilakukan pencocokan kondisi berhenti. Jika kondisi masih belum sesuai, maka iterasi *epoch* akan dilanjutkan.



Gambar 3.12 Flowchart Learning

J. Flowchart Bobot Awal

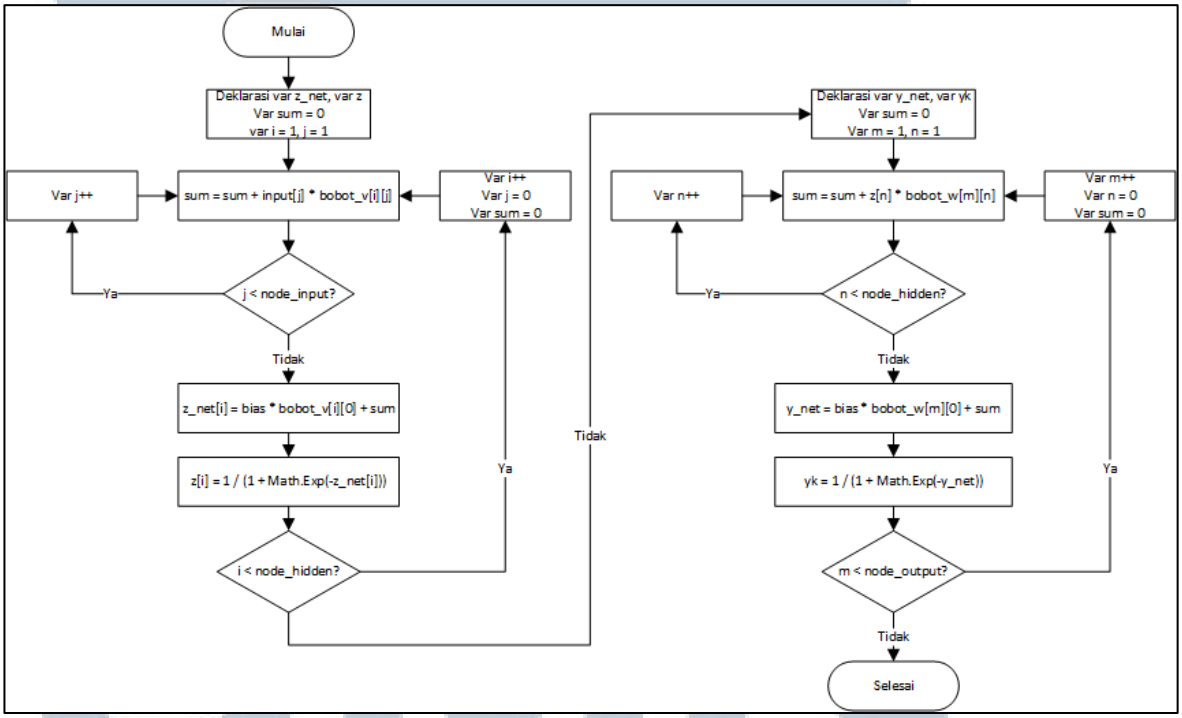
Gambar 3.13 merupakan *flowchart* penentuan bobot awal. Tahap ini bertujuan untuk menentukan bobot yang akan digunakan dalam algoritma *backpropagation*. Caranya dengan menentukan bilangan acak antara -0.5 sampai 0.5 untuk setiap bobot dari *input layer* menuju *hidden layer* (bobot_v) dan dari *hidden layer* menuju *output layer* (bobot_w).



Gambar 3.13 Flowchart Bobot Awal

K. Flowchart Propagasi Maju

Gambar 3.14 merupakan *flowchart* propagasi maju. Tahap ini merupakan fase pertama dalam algoritma *backpropagation*. Proses di dalam tahap ini bertujuan untuk mendapatkan nilai *output* dari input dan bobot yang sudah diberikan. Pertama-tama dapatkan hasil penjumlahan seluruh input yang mengarah ke setiap unit *hidden layer*. Lakukan perulangan hingga semua unit *hidden layer* mendapatkan nilainya. Setelah itu dapatkan nilai *output* dari semua unit *hidden layer* yang mengarah ke unit *output layer*.

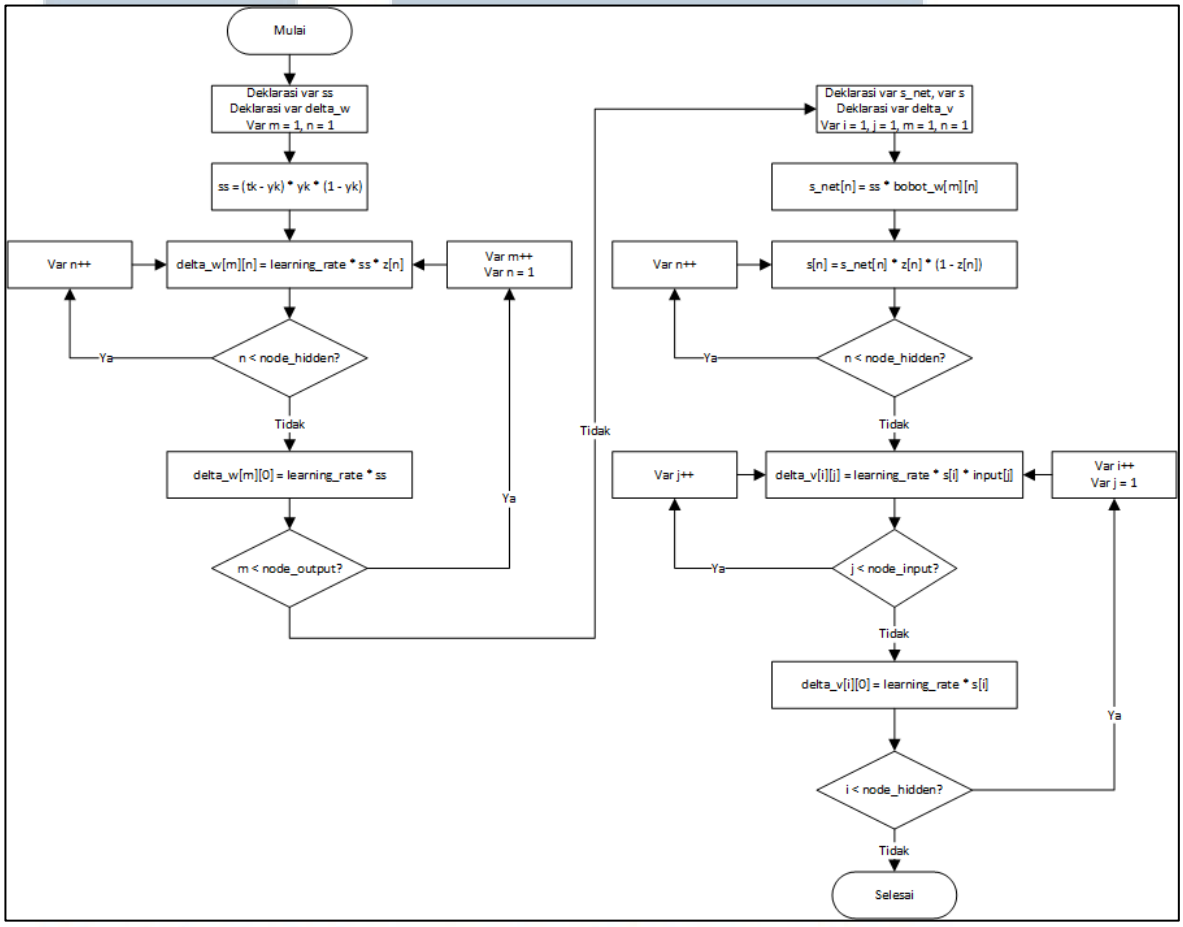


Gambar 3.14 Flowchart Propagasi Maju

L. Flowchart Propagasi Mundur

Gambar 3.15 merupakan *flowchart* propagasi mundur. Tahap ini merupakan fase kedua dalam algoritma *backpropagation*. Proses di dalam tahap ini bertujuan untuk mencari nilai kesalahan yang terjadi dari tahap propagasi maju. Nilai kesalahan tersebut akan digunakan untuk fase ketiga *backpropagation* yaitu

modifikasi bobot. Pertama-tama dapatkan nilai kesalahan dari unit *output layer* yang terhubung dengan unit *hidden layer*. Nilai kesalahan tersebut digunakan untuk mendapatkan nilai delta (perubahan bobot) pada *hidden layer*. Setelah mendapatkan nilai delta pada *hidden layer*, sistem akan mencari juga nilai kesalahan dari unit *hidden layer* yang terhubung dengan unit *input layer*. Nilai kesalahan yang dihasilkan dari unit *hidden layer* ke unit *input layer* akan digunakan untuk mendapatkan nilai delta pada *input layer*.

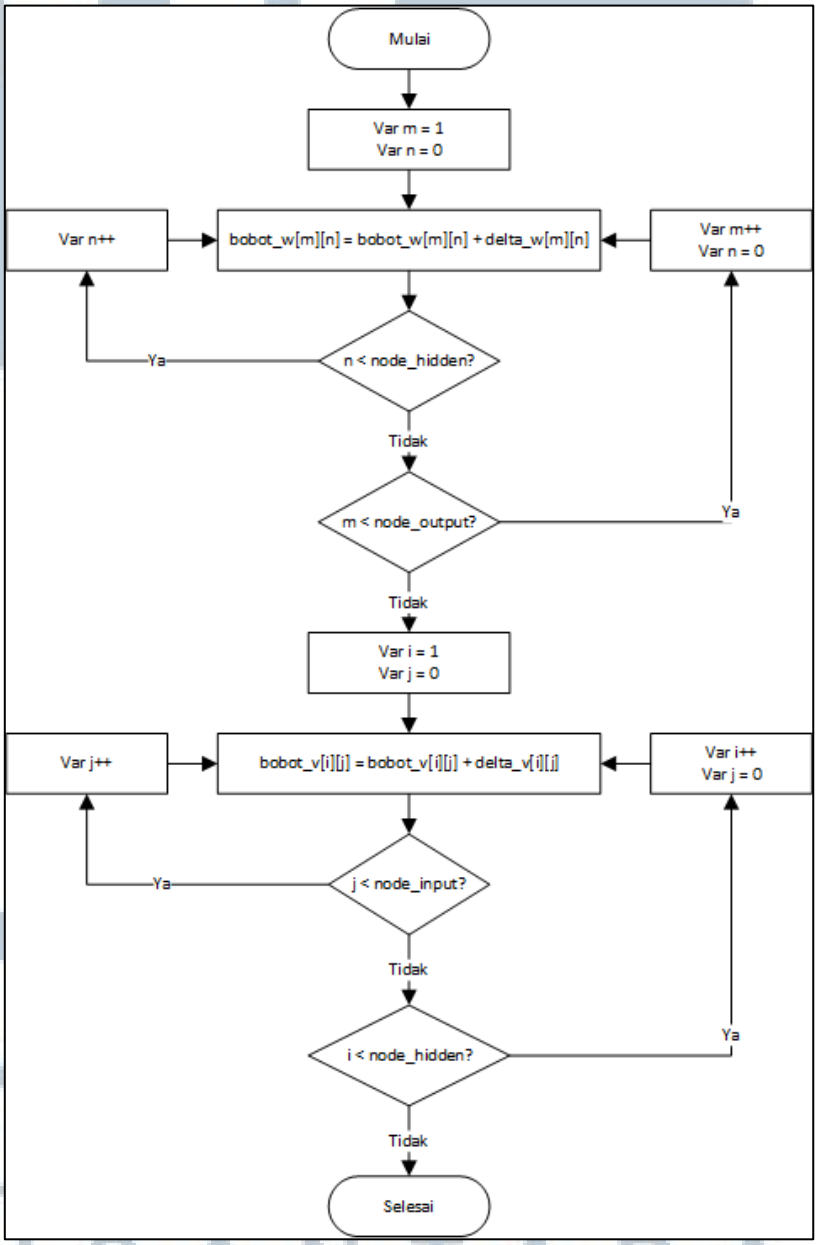


Gambar 3.15 Flowchart Propagasi Mundur

M. Flowchart Modifikasi Bobot

Gambar 3.16 merupakan *flowchart* modifikasi bobot. Tahap ini merupakan fase ketiga dari algoritma *backpropagation*. Proses di dalam tahap ini bertujuan

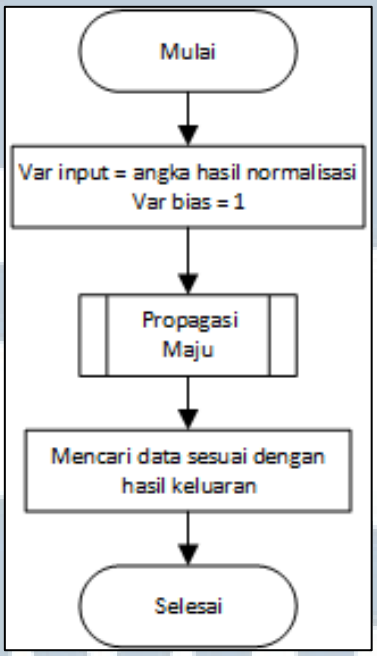
untuk merubah nilai bobot awal menjadi nilai bobot baru yang akan digunakan lagi pada fase pertama algoritma *backpropagation* yaitu tahap propagasi maju. Caranya dengan menambahkan nilai bobot lama dengan nilai delta yang telah didapatkan tahap propagasi mundur. Nilai bobot baru didapatkan mulai dari bobot pada *hidden layer* (variabel bobot_w) dan kemudian dilanjutkan pada *input layer* (variabel bobot_v).



Gambar 3.16 Flowchart Modifikasi Bobot

N. Flowchart Authentication

Gambar 3.17 merupakan *flowchart authentication*. Tahap ini bertujuan untuk mengautentikasi pemilik dari sidik jari yang diinput. Yang digunakan dalam tahapan ini hanya fase pertama dari algoritma *backpropagation* yaitu tahap propagasi maju saja. Nilai bobot diambil dari *file* karena langsung menguji dengan bobot optimalnya saja. Selesai tahap propagasi maju, *output* yang dihasilkan akan dicari dengan data yang ada di *file* dan nama dari kepemilikian sidik jari tersebut akan ditampilkan.



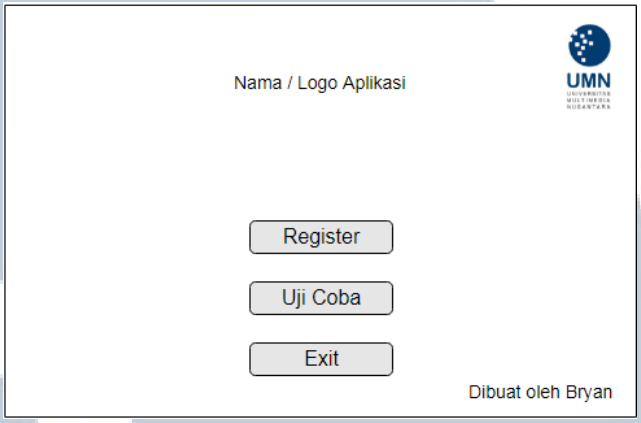
Gambar 3.17 Flowchart Authentication

3.2.3 Desain Antarmuka

Dalam mengimplementasikan aplikasi ini, dilakukan desain antarmuka terlebih dahulu. Gambar 3.18 merupakan gambar dari rancangan halaman utama yang akan dibuat. Pada halaman ini terdapat tiga tombol yaitu Register, Uji Coba, dan *Exit*. Tombol Register akan mengarahkan *user* ke halaman Register untuk registrasi sidik jarinya. Tombol Uji Coba akan mengarah ke halaman Uji Coba

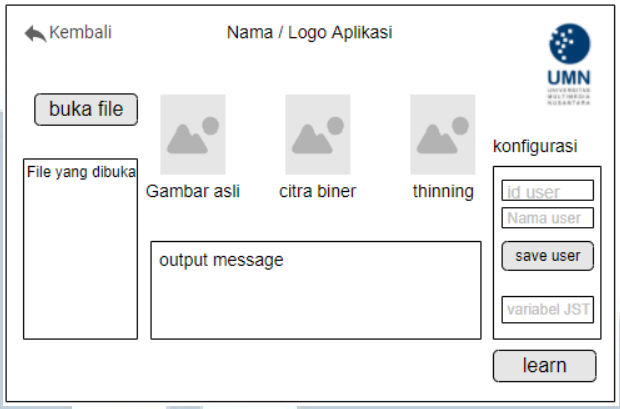
Implementasi Metode Image..., Bryan, FTI UMN, 2018

untuk melakukan proses autentikasi. Tombol *Exit* berfungsi untuk keluar dari aplikasi.



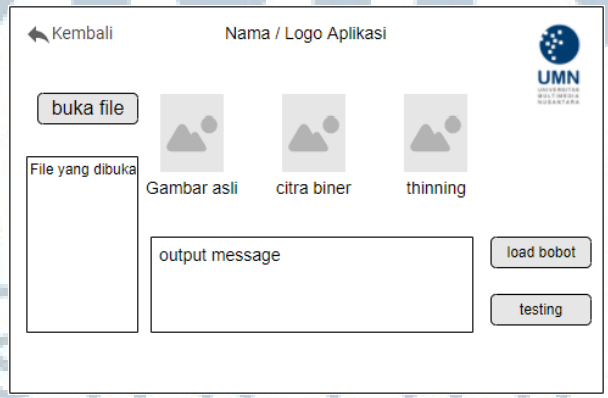
Gambar 3.18 Rancangan Halaman Utama

Gambar 3.19 merupakan gambar dari rancangan halaman Register yang akan digunakan oleh *user* untuk meregistrasikan sidik jarinya ke dalam sistem. Pada halaman ini terdapat konfigurasi untuk melakukan registrasi, potongan gambar yang dihasilkan dari proses-proses registrasi, dan juga *output message* dari proses registrasi. *User* yang ingin registrasi harus mengisi namanya terlebih dahulu sebagai data diri pada bagian konfigurasi, lalu tekan tombol buka *file*. *User* diharuskan untuk memilih *file* gambar sidik jari yang telah didapatkan sebelumnya dengan *scanner*. Selesai memilih, *user* harus menekan tombol *save user* untuk menyimpan data *user* pada *file*. Saat tombol *save user* ditekan, proses dari *image enhancement* akan ditampilkan dan juga akan mengeluarkan *output message* berupa proses telah berhasil. Proses penyimpanan data *user* ini berlanjut hingga semua data *user* yang ingin dilatih telah tersimpan. Setelah itu, *user* bisa mengubah beberapa variabel pada konfigurasi JST untuk proses *learning* yang diinginkan. Klik tombol *learn* untuk memulai proses *learning* dan tunggu hingga selesai.



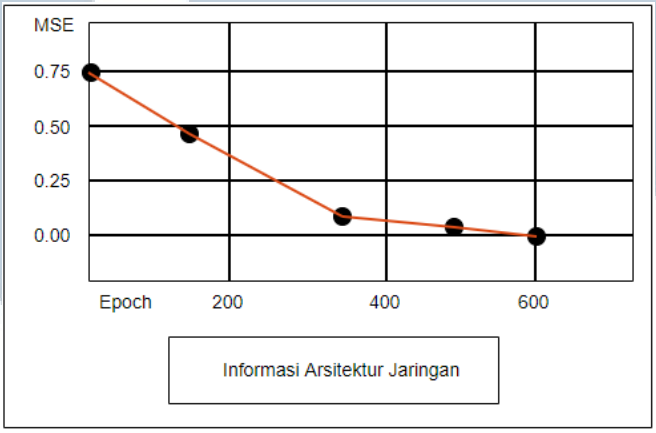
Gambar 3.19 Rancangan Halaman Register

Gambar 3.20 merupakan gambar dari rancangan halaman uji coba yang akan digunakan oleh *user* untuk mengautentikasi sidik jarinya. Pada halaman ini terdapat potongan gambar yang dihasilkan dari proses-proses uji coba, tombol untuk mengambil data bobot optimal yang ada di *database*, dan *output message* yang muncul dari proses autentikasi. Ketika *user* ingin melakukan uji coba, *user* diharuskan untuk mengambil data bobot terlebih dahulu. Setelah itu, *user* diharuskan untuk membuka sebuah *file* gambar sidik jari yang telah didapatkan sebelumnya dari *scanner*. Lalu tekan tombol *testing* untuk memulai melakukan pengujian. Proses *image enhancement* dari gambar yang diinputkan akan ditampilkan dan juga hasil dari autentikasinya akan ditampilkan di bagian *output message*.



Gambar 3.20 Rancangan Halaman Uji Coba

Gambar 3.21 merupakan gambar dari rancangan diagram *learning* yang akan ditampilkan setelah proses *learning* selesai. Pada halaman ini akan terdapat grafik *line chart* berdasarkan *epoch* yang dihasilkan dari proses *learning*. Diagram ini ditampilkan setiap 100 *epoch*. Selain itu ditampilkan juga informasi tentang arsitektur jaringan yang digunakan dalam proses *learning* seperti jumlah *node input*, jumlah *node hidden*, jumlah *node output*, *learning rate* yang digunakan, batas *error*, dan *epoch* yang didapatkan setelah proses *learning* selesai.



Gambar 3.21 Rancangan Halaman Diagram Learning

UNIVERSITAS MULTIMEDIA NUSANTARA