PERANCANGAN DAN PEMBUATAN SISTEM IDENTIFIKASI SIDIK JARI BERBASIS JARINGAN SYARAF HOPFIELD

Oleh :

Nama : Yudi Agustono

NIM : 01.41020.0003

Program : S1 (Strata Satu)

Jurusan : Sistem Komputer

SEKOLAH TINGGI

MANAJEMEN INFORMATIKA & TEKNIK KOMPUTER

SURABAYA

2006

PERANCANGAN DAN PEMBUATAN SISTEM IDENTIFIKASI SIDIK JARI BERBASIS JARINGAN SYARAF HOPFIELD

SKRIPSI

Diajukan sebagai salah satu syarat untuk menyelesaikan

Program Sarjana Komputer

Oleh:

Nama : YUDI AGUSTONO

NIM : 01.41020.0003

Program : S1 (Strata Satu)

Jurusan : Sistem Komputer

SEKOLAH TINGGI

MANAJEMEN INFORMATIKA & TEKNIK KOMPUTER

SURABAYA

2006

UNIVERSITAS
Dinamika

Ketika satu pintu tertutup, pintu lain terbuka,

namun terkadang kita melihat dan menyesali pintu tertutup tersebut terlalu lama hingga kita tidak melihat pintu lain yang telah terbuka.

*(**Alexander Graham Bell**)*

Ku persembahkan kepada

Bapak & Ibu tercinta

Kakak & Adikku yang kukasihi

Seseorang yang kukasihi dan kucintai

PERANCANGAN DAN PEMBUATAN SISTEM IDENTIFIKASI SIDIK JARI

BERBASIS JARINGAN SYARAF HOPFIELD

Disusun Oleh :

Nama : Yudi Agustono

Nim : 01.41020.0003

Surabaya, Agustus 2006

Telah diperiksa, diuji dan disetujui :

Pembimbing I	Pembimbing II
Basuki Rahmat, S.Si., MT	Harianto, S.Kom
NIDN. 0723076902	NIDN. 0722087701

Mengetahui :

Wakil Ketua Bidang Akademik

Drs. Antok Supriyanto, M.MT
NIDN. 070850500455

SEKOLAH TINGGI MANAJEMEN INFORMATIKA & TEKNIK KOMPUTER SURABAYA

UNIVERSITAS
Dinamika

ABSTRAKSI

Sistem identifikasi sidik jari merupakan sistem identifikasi biometrik yang mengidentifikasi seseorang berdasarkan karakteristik sidik jarinya. Sidik jari mempunyai pola yang berbeda tiap orang yang tidak akan berubah seumur hidup, kecuali jika terjadi kecelakaan serius. Sidik jari telah terbukti akurat, aman, mudah, dan nyaman untuk dipakai sebagai identifikasi. Metode yang umum digunakan dalam identifikasi sidik jari adalah metode *String Matching* dan jaringan syaraf *Backpropagation*.

Dalam penelitian ini metode yang digunakan adalah metode jaringan syaraf *Hopfield*. Penelitian ini bertujuan untuk mengimplementasikan jaringan syaraf tiruan ini dalam mengenali sidik jari, karena metode ini belum pernah digunakan untuk mengidentifikasi sidik jari. Dari hasil penelitian jaringan syaraf *Hopfield* mampu mengenali 100% dari sepuluh sidik jari dengan posisi sidik jari yang tegak lurus dan titik koordinat yang tidak terlalu berbeda dengan saat pendaftaran sidik jari. Namun jika posisi koordinat sidik jari mempunyai perbedaan yang jauh, jaringan syaraf *Hopfield* hanya mampu mengenali 20% dari sidik jari tersebut.

Metode ini sangat cocok dalam pengenalan pola karena *Hopfield* mampu mengenali pola yang terkena derau sekalipun. Namun dalam penelitian ini dengan kemampuan tersebut menyebabkan *Hopfield* mempunyai kelemahan dalam membedakan pola-pola sidik jari yang berbeda tapi bentuknya hampir sama.

KATA PENGANTAR

Dengan mengucapkan puji syukur Alhamdulillah kehadirat ALLAH SWT. yang telah melimpahkan berkah dan rahmat-Nya, sehingga penulis dapat menyelesaikan tugas akhir yang merupakan persyaratan dalam menyelesaikan program studi strata satu di Sekolah Tinggi Manajemen Informatika & Teknik Komputer Surabaya (STIKOM). Tugas akhir ini merupakan penelitian tentang perancangan sistem identifikasi sidik jari berbasis jaringan syaraf hopfield.

Penulisan tugas akhir ini tidak akan terselesaikan dengan baik apabila tidak mendapat dukungan, saran, masukan, ataupun kritik dari berbagai pihak. Maka dengan sepenuh hati penulis dalam kesempatan ini mengucapkan terima kasih atas bantuannya, kepada yang terhormat:

1. Bapak dan Ibu yang yang telah memberikan doa dan dorongan serta motivasi kepada penulis.
2. Bapak Basuki Rahmat, sebagai dosen pembimbing I atas segala arahan dan bimbingan selama penulis mengerjakan tugas akhir ini.
3. Bapak Harianto, sebagai dosen pembimbing II atas segala arahan dan bimbingan selama penulis mengerjakan tugas akhir ini.
4. Bapak Tjio Hok Hoo, sebagai Kaprodi Sistem Komputer atas segala arahan dan bimbingan selama penulis mengerjakan tugas akhir ini.
5. Nia, yang selalu setia mendukung dan memberi motivasi serta doa demi terselesainya tugas akhir ini.
6. Mas Dani dan Mbak Yus yang telah memberi waktu dan tempat untuk menyelesaikan tugas akhir ini.

7. Teman-teman angkatan 2001 (terutama Didik, Galih, Irwan, Iwan), yang telah memberikan semangat kepada penulis sehingga tugas akhir ini dapat diselesaikan dengan baik.
8. Dan masih banyak orang-orang yang sangat berperan dalam mewujudkan tugas akhir ini yang tidak bisa penulis sebutkan satu per satu.

Semoga ALLAH SWT. membalas ketulusan dan budi baik mereka yang telah banyak memberikan bantuan, bimbingan, ataupun nasehat-nasehat kepada penulis. Penulis menyadari bahwa masih banyak kekurangan pada penulisan tugas akhir ini. Namun penulis berharap semoga Tugas Akhir ini dapat ikut menunjang perkembangan ilmu pengetahuan, khususnya ilmu komputer.

Surabaya, Juli 2006

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR ISTILAH DAN SINGKATAN

Axon	Penghubung lebih lanjut dari body sel dan pembawa sinyal dari neuron
BMP	*Bitmap*
Dendrite	Percabangan berupa serat-serat dari bodi sel
DNA	*Deoxyribonucleic Acid*
Dpi	*Dot per inchi*
Error	Tingkat kesalahan
ID	Identitas
JPEG	*Joint Photographic Experts Group*
JST	Jaringan Syaraf Tiruan
Konvergen	kondisi dimana *output* yang sudah tidak berubah lagi untuk iterasi selanjutnya
Neuron	Unit pemroses informasi yang menjadi dasar dalam pengoperasian JST.
MM	*Milimeter*
Perceptron	Salah satu algorimta JST *Single Layer* dengan satu *output.*
Pixel	*Picture Element*
RGB	*Red-Green-Blue*
Synapsis	Akhir dari sebuah organ
USA	United State Of America
USB	*Universal Serial Bus*

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Sistem identifikasi (pengenalan) sidik jari dapat disebut juga dengan sistem identifikasi *biometrik*. Yang dimaksud dengan *biometrik* adalah suatu teknik untuk mengidentifikasi seseorang berdasarkan karakteristik yang melekat pada diri orang tersebut, misalnya identifikasi sidik jari, pencocokan wajah, identifikasi retina mata, dan yang lainnya. Dalam hal ini peneliti berusaha mengenali seseorang dengan mengenali pola sidik jarinya.

Sidik jari telah terbukti akurat, aman, mudah dan nyaman untuk dipakai sebagai identifikasi bila dibandingkan dengan sistem *biometrik* lainnya, seperti retina mata atau DNA. Hal ini dapat dilihat pada sifat yang dimiliki oleh sidik jari, antara lain :

1. *Perennial nature*, yaitu guratan-guratan pada sidik jari yang melekat pada kulit manusia seumur hidup.
2. *Immutability*, yaitu sidik jari seseorang tidak pernah berubah, kecuali mendapatkan kecelakaan yang serius.
3. *Individuality*, yaitu pola sidik jari adalah unik dan berbeda untuk setiap orang.

Dari ketiga sifat tersebut, sidik jari secara signifikan dapat digunakan sebagai sistem identifikasi yang dapat digunakan dalam aplikasi teknologi informasi seperti :

1. *Accsess System Security*, yaitu akses untuk masuk ke suatu area atau ruangan tertentu yang *restricted* (terlarang).
2. *Authentification System*, yaitu untuk akses data yang sifatnya rahasia dan terbatas (misalnya data pada perbankan, militer dan diplomatik).

Pengenalan atau identifikasi pola sidik jari ini telah banyak diteliti sebelumnya dengan bermacam-macam cara atau metode yang berbeda-beda. Ada yang menggunakan metode konvensional maupun metode kecerdasan buatan.

Dalam hal ini peneliti berusaha mengenali pola sidik jari menggunakan metode pengenalan pola yang berbasis jaringan syaraf *Hopfield*. Jaringan syaraf *Hopfield* ini dapat digunakan dengan mudah dalam menyelesaikan permasalahan pengenalan pola. Hal ini dapat dilihat dari rumus perhitungan algoritma dalam suatu matrik memori yang tidak terlalu rumit, tetapi memberikan hasil yang sesuai dan tepat. Jaringan syaraf *Hopfield* sangat cocok untuk memproses pengaturan bobot (*weight*), kemampuan pengkodean, dan lebih fleksibel untuk permasalahan yang rumit. Dengan kemampuan yang lebih, metode jaringan syaraf *Hopfield* ini diharapkan dapat mengidentifikasi sidik jari dengan baik dan tepat.

1.2 Rumusan Masalah

1. Bagaimana cara agar jaringan syaraf tiruan ini dapat mempelajari pola sidik jari yang diinputkan kepadanya?
2. Bagaimana cara agar jaringan syaraf tiruan ini dapat mengidentifikasi atau mengenali pola sidik jari yang akan diuji dengan baik?

1.3 Batasan Masalah

1. Sistem identifikasi sidik jari menggunakan metode pengenalan pola yang berbasis jaringan syaraf *Hopfield*.
2. Sidik jari yang akan diidentifikasi adalah sidik jari pada bagian ibu jari pada tangan sebelah kanan.
3. Sidik jari yang akan diidentifikasi adalah sebanyak 256 pola sidik jari dari orang yang berbeda, Artinya tiap orang diambil satu pola sidik jarinya.
4. Sidik jari yang diinputkan harus sesuai dengan ketentuan yang telah ditetapkan.
5. Sidik jari diinputkan ke alat *fingerprint scanner* yang dimana outputnya berupa citra *bit biner*.

1.4 Tujuan

Tujuan dari penelitian ini adalah untuk mengimplementasikan sistem jaringan syaraf tiruan dalam pengenalan pola sidik jari manusia.

1.5 Kontribusi

Pengenalan atau identifikasi pola sidik jari ini telah banyak diteliti sebelumnya dengan bermacam-macam cara atau metode yang berbeda-beda. Ada yang menggunakan metode konvensional maupun metode kecerdasan buatan. Pada umumnya untuk mengenali atau identifikasi sidik jari menggunakan metode konvensional yaitu : membandingkan citra sidik jari satu dengan yang lainnya dengan mencocokan guratan-guratan dari sidik jari tersebut.

Namun metode kecerdasan buatan juga telah banyak dikembangkan untuk mengidentifikasi sidik jari antara lain Pengenalan Identifikasi Diri

Menggunakan Pola Sidik Jari dengan Metode *Backpropagation Neural Networks* oleh Muhammad Wahyu (2003), Pengenalan Pola Sidik Jari dengan *Artificial Neural Network* Menggunakan Metode *Backpropagation* oleh Suhartati, F., Fahmi, A. & Indratno, W. (2003), dan Sistem Keamanan Akses Menggunakan Pola Sidik Jari Berbasis Jaringan Syaraf Tiruan oleh Elvayandri (2002).

Dalam tugas akhir ini, penelitian yang dilakukan dengan metode jaringan syaraf *Hopfield* untuk mengenali pola sidik jari. Penelitian dengan metode ini sepertinya belum pernah dilakukan sebelumnya. Sedangkan pola sidik jari didapat dari pembacaan sensor. Jadi input sidik jari hasil pembacaan sensor akan diidentifikasi oleh jaringan syaraf untuk diketahui identitas pemiliknya.

1.6 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir disusun sebagai berikut :

BAB I : Pendahuluan

Pada bab ini membahas tentang latar belakang, perumusan masalah, pembatasan masalah, tujuan, kontribusi dan sistematika penulisan.

BAB II : Landasan Teori

Pada bab ini membahas tentang teori dasar dalam pembuatan tugas akhir ini yang meliputi teori dasar biometrik, sidik jari, pengolahan citra, jaringan syaraf tiruan beserta metode algoritma *Hopfield*.

BAB III : Metode Penelitian

Pada bab ini membahas tentang perancangan dan pembuatan sistem identifikasi sidik jari yang berbasis jaringan syaraf *Hopfield*.

BAB IV : Pengujian Sistem

Pada bab ini membahas tentang pengujian sistem identifikasi sidik jari yang berbasis jaringan syaraf *Hopfield* yang meliputi pengujian proses pengenalan data uji oleh jaringan syaraf *Hopfield* yang sebelumnya dilakukan proses pembelajaran data latih yang berupa sidik jari dan selanjutnya akan diidentifikasi output dari jaringan syaraf tersebut.

BAB V : Penutup

Pada bab ini berisi kesimpulan hasil pengujian sistem secara keseluruhan dan saran-saran yang diharapkan terhadap pengembangan dari tugas akhir ini.

BAB II

LANDASAN TEORI

2.1 *Biometrik*

Biometrik adalah bidang ilmu pengetahuan yang menggunakan karakteristik individual seseorang untuk memeriksa dan menentukan identitasnya, yang hasilnya dijadikan sebagai pengganti dari nomor identitas seseorang (Machan, 2004:3). Menurut Stephen Cobb (1996) istilah *biometrik* merujuk ke arah sistem authentifikasi. Dapat didefinisikan sebagai : sebuah pengukuran karakteristik fisik dari seseorang yang digunakan sebagai identitas pengenal, atau untuk memeriksa dan menguji identitas yang dipunyai seseorang melalui alat otomatis. Teknik identifikasi *biometrik* ini terdiri berbagai macam pengenalan karakteristik individu sebagai identitas pengenal, yaitu :

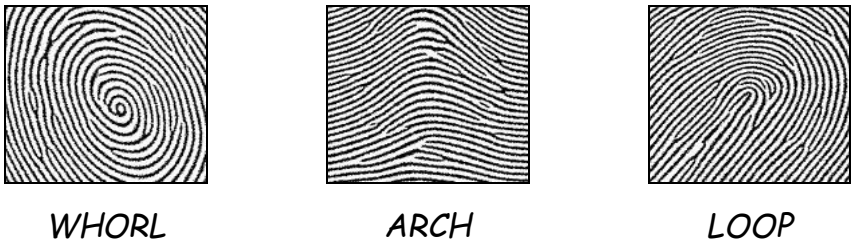
1. Identifikasi melalui pengenalan pola sidik jari
2. Identifikasi melalui pengenalan pola retina mata
3. Identifikasi melalui pengenalan suara
4. Identifikasi melalui pengenalan pola wajah
5. Identifikasi melalui pengenalan DNA, dan yang lainnya

2.2 Identifikasi Sidik Jari

Identifikasi sidik jari merupakan salah satu teknik dari *biometrik*, yaitu menemukan pola sidik jari, dan kemudian mencari karakteristik dari sidik jari tersebut.

Sistem pengenalan sidik jari meliputi salah satu dari :

1. *Classification System* : yang mana secara umum mengklasifikasikan sidik jari menjadi tiga kategori (Whorl, Arch, Loop).



Gambar 2.1 Klasifikasi Sidik Jari

2. *Matching System* : setiap sidik jari yang tidak dikenali akan diidentifikasi atau mencocokkan antara dua sidik jari, kemudian menolak atau menerima sidik jari tersebut ke dalam sistem.

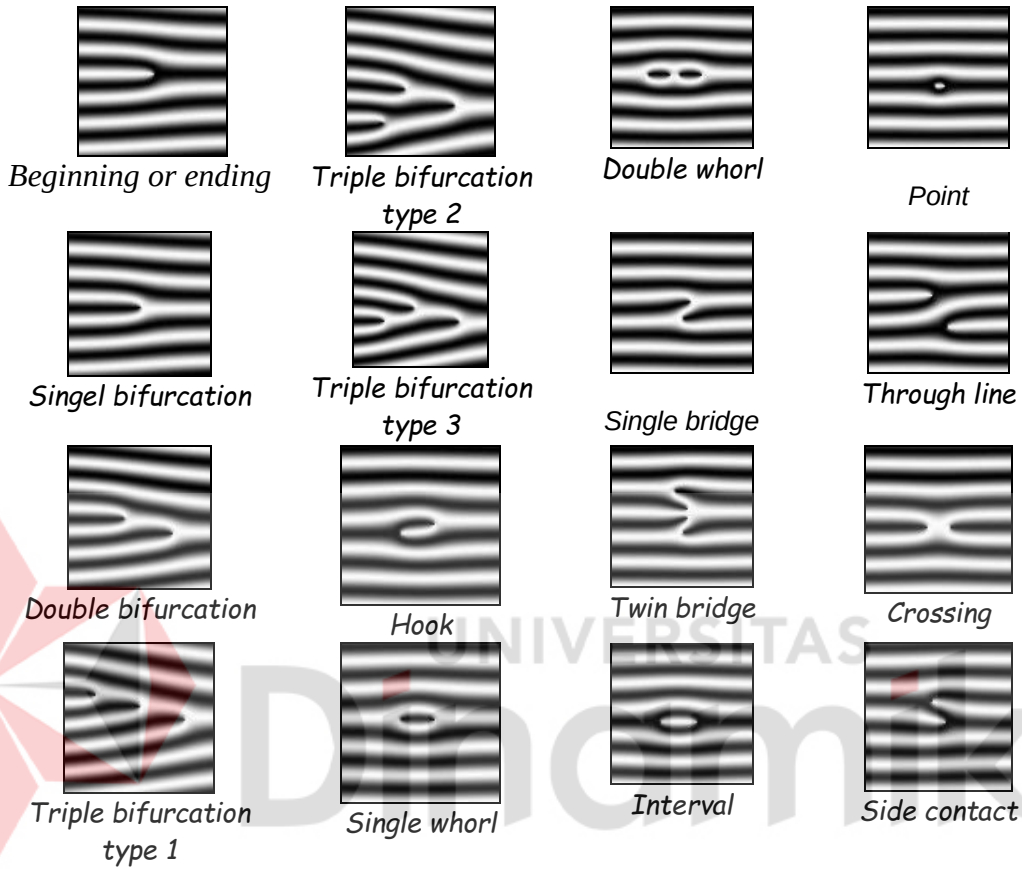
Sistem identifikasi dapat diklasifikasikan berdasarkan *feature* yang dimiliki sebuah image sidik jari ke dalam :

1. *Predominately Local* (*Minutiae Based*), mengektrak titik-titik *minutiae* dan membandingkan penempatan relatifnya dengan *template* yang sudah ada.
2. *Exclusively Global* (*Base on Henry System*), menggunakan *core* dan *delta point* untuk mengklasifikasikan sidik jari ke dalam tiga kategori di atas.

2.3 *Minutiae*

Minutiae adalah karakteristik dari guratan-guratan sidik jari. *Feature* yang digunakan adalah guratan sidik jari yang dapat diidentifikasi dengan cara menganalisa *minutiae* tersebut. Menurut Francis Galton (1822-1916) mengatakan

bahwa tidak ada dua sidik jari yang sama. Berikut *feature* guratan sidik jari dapat dilihat pada gambar berikut :



Gambar 2.2 Minutiae atau Guratan-guratan Sidik Jari (Evayandri, 2002:4)

Menurut Prabhakar, Jain dan Pankanti (2001:5) pencocokan sidik jari dapat dilakukan dengan metode *Minutiae Filtering*, yaitu dengan mencari lokasi dan tipe dari *minutiae* apakah termasuk akhiran atau percabangan pada suatu citra sidik jari yang telah dinormalisasi dengan metode-metode pengolahan citra yang ada. Setelah semua titik direkam dalam *template* citra sidik jari yang ada dalam database. Sedangkan menurut Abbas (2001:19) pencocokan sidik jari dapat dilakukan dengan menghitung nilai rata-rata *pixel* dari suatu area yang diharapkan

mengelilingi titik tengah sidik jari yang selanjutnya disebut dengan *Finger Code*, kemudian menjadikan *Finger Code* tersebut sebagai *input* pada tahap pelatihan jaringan syaraf.

2.4 Digitasi Citra

Secara harafiah citra adalah gambar pada bidang dwimatra (dua dimensi). Ditinjau dari sudut pandang matematis, citra merupakan fungsi menerus (*continue*) dari intensitas cahaya pada bidang dwimatra. Sumber cahaya menerangi obyek, obyek ditangkap oleh alat-alat optik, misalnya mata pada manusia, kamera, pemindai (*scanner*), dan sebagainya. Sehingga bayangan obyek tersebut dapat direkam (Munir, 2004:2).

Citra terbagi menjadi dua macam, yaitu : citra kontinu dan citra diskrit. Citra kontinyu dihasilkan dari sistem optik yang menerima sinyal analog, misalnya mata manusia dan kamera analog. Citra diskrit dihasilkan melalui proses digitalisasi terhadap citra kontinyu. Beberapa sistem optik dilengkapi dengan fungsi digitalisasi sehingga ia mampu menghasilkan citra diskrit, misalnya kamera digital dan scanner, citra diskrit ini dapat disebut juga citra digital.

Citra hasil proses digitalisasi ini, secara matematis berupa matrik dua dimensi. Tiap-tiap elemen dari matrik merepresentasikan nilai dari *pixel* yang bersesuaian dalam citra. misalkan sebuah citra berukuran 256 x 256 *pixel* dan direpresentasikan secara numerik dengan matrik yang terdiri dari 256 buah baris dan 256 bauh kolom. Jadi citra digital sangat sesuai direpresentasikan dalam matrik i_{nxm}, dengan n adalah jumlah baris, dan m adalah jumlah kolom (Munir, 2004:18).

$$i = \begin{bmatrix} i(1,1) & i(1,2) & \cdots & i(1,m) \\ i(2,1) & i(2,2) & \cdots & i(2,m) \\ \vdots & \vdots & & \vdots \\ i(n,1) & i(n,2) & \cdots & i(n,m) \end{bmatrix} \quad (2.1)$$

Setiap elemen dari matrik di atas, berisi nilai dari *pixel* gambar dengan indeks yang bersesuaian. Nilainya menyatakan warna atau tingkat kecerahan dari gambar. *Range* nilainya dapat bervariasi sesuai dengan format penyimpanan gambar dalam memori.

Pada gambar *Gray Scale* ada banyak kemungkinan warna yang muncul dari kombinasi warna hitam dan putih. Banyaknya kemungkinan ini tergantung pada jumlah bit yang digunakan dalam tiap-tiap *pixel*-nya. Untuk skala keabuan 4 bit, maka jumlah kemungkinan nilainya adalah 2^4 atau 16, dengan *range* nilai dari 0 sampai 2^4-1. Sedangkan untuk skala keabuan 8 bit, maka jumlah kemungkinan nilainya adalah 2^8 atau 256, dengan *range* nilai dari 0 sampai 2^8-1. Format gambar ini disebut skala keabuan karena pada umumnya warna yang dipakai adalah antara warna hitam sebagai warna minimal, dan warna putih sebagai warna maksimal, sehingga warna antaranya adalah abu-abu.

Untuk gambar biner (*Monochrome*), setiap *pixel* bernilai 0 dan 1, masing-masing merepresentasikan warna tertentu. Contoh yang paling lazim adalah warna hitam bernilai 0 dan warna putih bernilai 1. Setiap *pixel* pada gambar hanya membutuhkan 1 bit, sehingga setiap *byte* dapat menampung informasi 8 *pixel* (Ahmad, 2005:11).

2.5 Sensor U.are.U 4000 Digital Persona

Sensor U.are.U 4000 Digital Persona adalah pembaca sidik jari dengan *interface* USB (*Universal Serial Bus*) didisain untuk digunakan dengan program

aplikasi dan pengembangan alat dari perusahaan Digital Persona. Pengguna dapat dengan mudah meletakkan jari pada jendela optik (tempat untuk meletakkan jari) dan alat ini akan membacanya dan disimpan dalam bentuk citra *grayscale*. Dan kualitas hasil dari pembacaan sidik jari ini sangat bagus. Berikut spesifikasi sensor U.are.U 4000 :

a. Ukuran sensor 79 mm x 49 mm x 19 mm.
b. Ukuran area jendela optik pembaca lebarnya = 14,6 mm dan untuk panjangnya = 18,1 mm
c. Resolusi *pixel*-nya adalah 512 dpi.
d. *Output* citra adalah *grayscale* (8-bit)
e. Temperatur operasi 0 sampai 40^0c
f. *Voltase* 5 *volt* yang disuplai dari USB
g. Koneksi *pin out* dari sensor adalah *pin* 1 = *Vcc*, *pin* 2 dan 3 = *Ground*, *pin* 4 = USB D-, dan *pin* 5 = USB D+.
h. Alat ini cocok dengan USB 1.0, 1.1, dan 2.0 serta cocok dengan semua operasi sistem Microsoft.

Gambar 2.3 Sensor U.are.U 4000 Digital Persona

2.6 Binerisasi

Citra biner adalah citra yang hanya mempunyai dua derajat keabuan, yaitu hitam dan putih. *Pixel-pixel* obyek bernilai 1 dan *pixel-pixel* latar belakang bernilai 0, jadi pada ctra biner latar belakang berwarna putih dan obyek berwarna hitam (Munir, 2004:179).

Untuk mencari nilai biner dari suatu citra *grayscale* dilakukan dengan cara pengambangan secara global (*global image thresholding*). Setiap *pixel* di dalam citra dipetakan ke dua nilai yaitu 1 atau dengan fungsi pengambangan sebagai berikut :

$$f_B(i,j) = \begin{cases} 1 \cdots f_G(i,j) \leq T \\ 0 \cdots Lainnya \end{cases} \quad (2.2)$$

Keterangan :

f_B (i,j) adalah citra biner

f_G (i,j) adalah citra *grayscale*

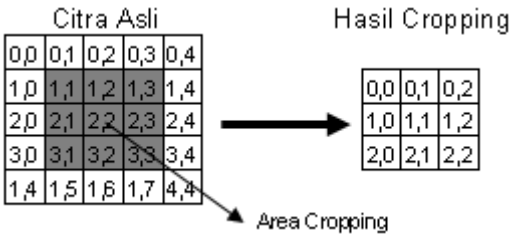
T adalah nilai ambang yang dispesifikasikan

Dengan operasi pengambangan tersebut obyek dibuat berwarna gelap (1 atau hitam), sedangkan latar belakang berwarna terang (0 atau putih). Nilai ambang T ditentukan dengan membuat histogram citra. Jika citra mengandung satu buah obyek dan latar belakang mempunyai nilai intensitas yang homogen, maka citra tersebut umumnya mempunyai histogram *bimodal* (mempunyai dua puncak atau dua maksimal lokal). Nilai T dipilih pada nilai minimum lokal yang terdapat di antara puncak.

2.7 *Cropping*

Cropping adalah proses pemotongan citra pada koordinat tertentu pada area citra. Untuk memotong bagian dari citra digunakan dua koordinat, yaitu

koordinat awal yang merupakan awal koordinat bagi citra hasil pemotongan dan koordinat akhir yang merupakan titik koordinat akhir dari citra hasil pemotongan. Sehingga akan membentuk bangun segi empat yang mana tiap-tiap *pixel* yang ada pada area koordinat tersebut akan disimpan dalam citra yang baru.



Gambar 2.4 Proses *Cropping*

Dari gambar tersebut dapat dijelaskan bahwa terjadi proses pemotongan citra. Pada awalnya ukuran *pixel* dari citra asli adalah 5x5 *pixel*, setelah dilakukan proses pemotongan pada koordinat awal (1,1) dan koordinat akhir (3,3) atau dengan lebar 3 *pixel* dan tinggi 3 *pixel* akan tebentuk citra baru dengan ukuran 3x3 *pixel*. Citra baru ini berisi nilai *pixel* dari koordinat (1,1) sampai koordinat (3,3).

2.8 *Resizing*

Resizing dapat disebut juga dengan *image zooming*, yaitu pengubahan ukuran citra baik itu pengubahan membesar (*zoom in*) atau pengubahan mengecil (*zoom out*). Berikut rumus pengubahan citra (Munir, 2004:69):

$$x' = s_x \cdot x \text{ dan } y' = s_y \cdot y \quad (2.3)$$

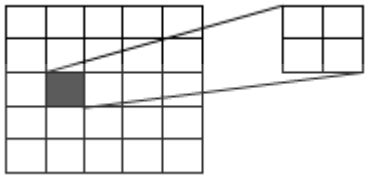
Keterangan :

s_x dan s_y adalah faktor skala masing-masing dalam arah x dan y

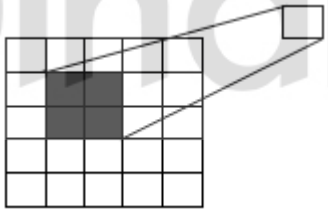
Jika citra semula adalah A dan citra hasil penskalaan adalah B, maka penskalaan citra dinyatakan sebagai berikut :

$$B[x'][y'] = B[s_x \cdot x][s_y \cdot y] = A[x][y] \quad (2.4)$$

Operasi pembesaran ukuran (*zoom in*) dengan faktor skala sama dengan 2 (yaitu : $s_x = s_y = 2$) diimplementasikan dengan menyalin setiap *pixel* akan dijadikan 4 *pixel*. Sedangkan untuk operasi pengecilan ukuran (*zoom out*) dengan faktor skala sama dengan $1/2$ (yaitu : $s_x = s_y = 1/2$) maka akan dilakukan pengambilan rata-rata dari 4 *pixel* bertetangga menjadi 1 *pixel*.



Gambar 2.5 Operasi *Zoom In*



Gambar 2.6 Operasi *Zoom Out*

2.9 Penipisan Pola

Penipisan (*Thinning*) adalah operasi pemrosesan citra biner yang dalam hal ini obyek (*region*) direduksi menjadi rangka yang menghampiri garis sumbu obyek. Tujuan penipisan adalah mengurangi bagian yang tidak perlu (*redundant*) sehingga hanya dihasilkan informasi yang penting saja. Pola hasil harus tetap

menyerupai pola yang aslinya. Penipisan pola merupakan proses iteratif yang menghilangkan *pixel-pixel* hitam pada tepi-tepi pola dirubah menjadi *pixel* putih.

Algoritma penipisan pola adalah memeriksa *pixel-pixel* di dalam jendela yang berukuran 3x3 *pixel*.

P8	P1	P2
P7	P0	P3
P6	P5	P4

Gambar 2.7 Jendela 3x3 *Pixel*

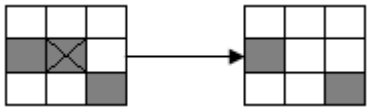
Berikut langkah-langkahnya :

1. Mula-mula diperiksa jumlah *pixel* obyek yang bernilai 1 di dalam jendela 3x3 *pixel* dan disimpan di dalam variabel N.
2. Jika N kurang dari 2 tidak ada aksi yang dilakukan karena di dalam jendela terdapat ujung lengan obyek. Jika proses ini terus dijalankan maka setiap pngecekan pada pixel berikutnya akan mengakibatkan ujung dari lengan akan terpotong terus menerus.

Gambar 2.8 Penghapusan Lengan Obyek

3. Jika N lebih besar dari 7 tidak ada aksi yang dilakukan karena dapat menyebabkan pengikisan (*erotion*) obyek.

4. Jika N lebih besar dari 2 apakah penghilangan *pixel* tengah menyebabkan obyek tidak terhubung. Ini dilakukan dengan membentuk barisan p_1, p_2, p_3,…, p_8, p_1. Jika jumlah peralihan dari $0 \rightarrow 1$ di dalam barisan tersebut sama dengan 1, berarti hanya terdapat 1 komponen terhubung dalam jendela tersebut, sedangkan jika terdapat 2 peralihan maka dilakukan penghapusan menyebabkan terputusnya titik penghubung. (Munir, 2004:192).

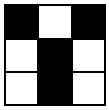
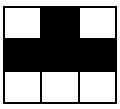


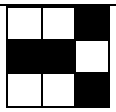
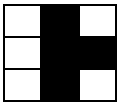
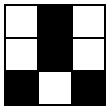
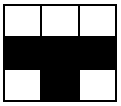
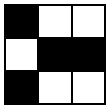
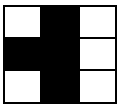
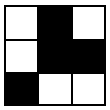
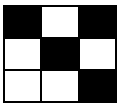
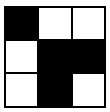
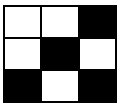
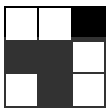
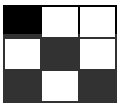
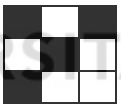
Gambar 2.9 Penghapusan Titik Penghubung

2.10 *Feature Extraction*

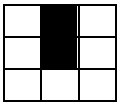
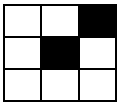
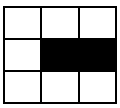
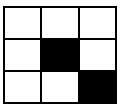
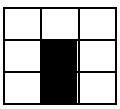
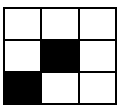
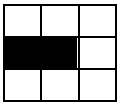
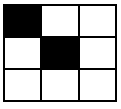
Feature Extraction adalah proses mengambil ciri-ciri yang terdapat pada obyek di dalam citra (Munir, 2004:244). Pengambilan ciri-ciri ini yang dapat digunakan untuk membedakan suatu obyek dengan obyek yang lainnya. Dalam identifikasi sidik jari yang akan diambil ciri-cirinya adalah guratan-guratan dari sidik jari tersebut (*Minutiae Extraction*). Guratan guratan yang umum digunakan dalam proses pengambilan ciri adalah guratan percabangan (*Ridge Bifurcation*) dan guratan akhiran (*Ridge Ending*). Berikut guratan-guratan dalam matrik 3x3 dan nilai binernya (Lee dan Wang, 1999:12).

Tabel 2.1 Sampel Pola Guratan *Ridge Bifurcation*

No.	Minutiae	Nilai Biner	No.	Minutiae	Nilai Biner
1		101010010	9		010111000

2		001110001	10		010011010
3		010010101	11		000111010
4		100011100	12		010110010
5		010011100	13		101010001
6		100011010	14		001010101
7		001110010	15		100010101
8		010110001	16		101010100

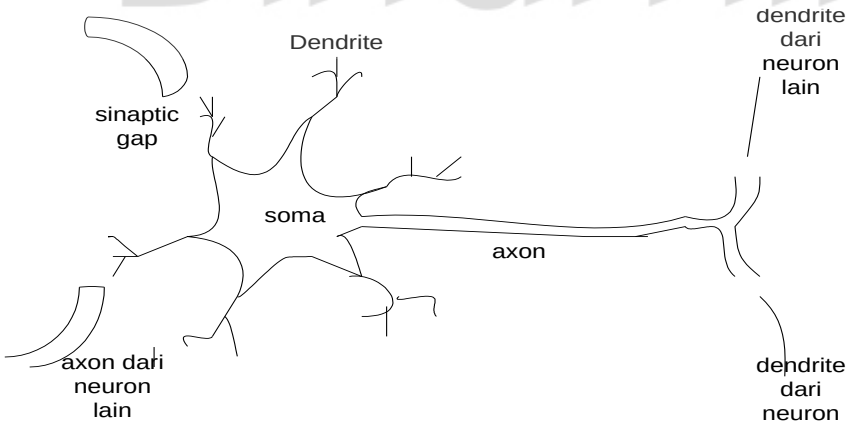
Tabel 2.2 Sampel Pola Guratan *Ridge Ending*

No.	Minutiae	Nilai Biner	No.	Minutiae	Nilai Biner
1		010010000	5		001010000
2		000011000	6		000010001
3		000010010	7		000010100
4		000110000	8		100010000

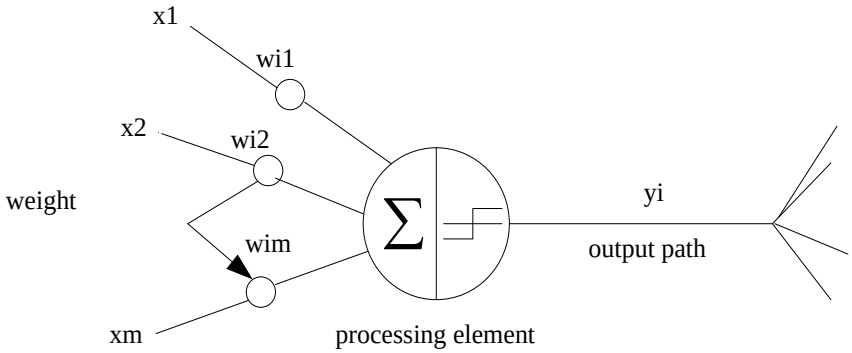
2.11 Jaringan Syaraf Tiruan (JST)

JST (Jaringan Syaraf Tiruan) atau *Artificial Neural Networks* adalah sistem komputansi dimana arsitektur dan operasi diilhami dari pengetahuan tentang sel syaraf biologi di dalam otak (Kristanto, 2004:2). Pada dasarnya JST meniru cara kerja otak makhluk hidup. Pada jaringan syaraf manusia, sebuah *neuron* memiliki : *dendrites* (sebagai *input*), tubuh *sel*, dan *axon* (sebagai *output*).

Neuron menerima *input* dari *neuron* yang lain dan akan dijumlahkan dengan menggunakan suatu model matematika dan berdasarkan arsitektur dari jaringan syaraf itu sendiri. *Axon* yang paling akhir mengenai badan *sel* atau *dendrite* serta mengenai *neuron* yang berikutnya. Transmisi dari suatu isyarat elektrik dari suatu *neuron* kepada *neuron* yang lain diakibatkan oleh *neurotransmitter*, dimana dari *neuron* pertama akan mengikat sebuah sel pada *neuron* yang kedua dan seterusnya. Mata rantai ini disebut dengan suatu *sinapsis*.



Gambar 2.10 Sistem *Neuron* Pada Jaringan Syaraf Manusia



Gambar 2.11 Sistem *Neuron* pada Jaringan Syaraf Tiruan

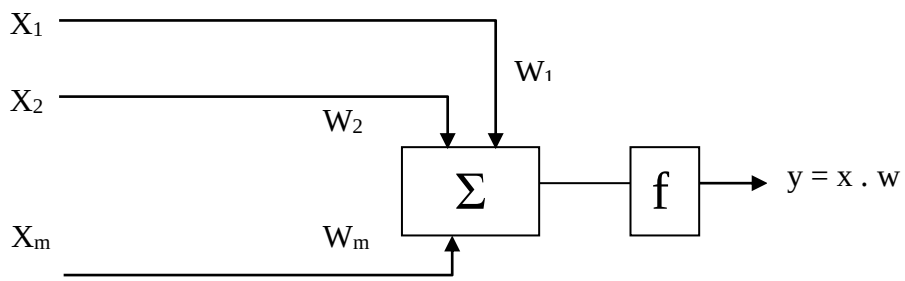
Jadi hubungan antara *neuron* satu dengan yang lain disertai dengan bobot *sinapsis* dan nilai ambang batas (*threshold*).

JST dapat digambarkan sebagai model matematis dan komputansi untuk fungsi *aproksimasi nonlinear*, klasifikasi data, *cluster* dan *regresi non parametik* atau sebagai sebuah simulasi dari koleksi model syaraf biologi. Berdasarkan kemampuan yang dimiliki, JST dapat digunakan untuk belajar dan menghasilkan aturan atau operasi dari beberapa contoh, untuk menghasilkan *output* yang sempurna dari contoh atau *input* yang dimasukkan dan membuat prediksi tentang kemungkinan *output* yang akan muncul atau menyimpan karakteristik dari *input* yang disimpan kepadanya.

2.12 Model Matematis Jaringan Syaraf Tiruan

Paradigma, model atau pola Jaringan Syaraf Tiruan adalah kondisi topologi dari interkoneksinya serta aturan yang dikenakan padanya. Pola atau model jaringan syaraf tiruan pada dasarnya meniru jaringan syaraf yang ada pada

manusia. Berikut model yang disederhanakan dari sebuah sel syaraf (*neuron*) tiruan yang merupakan dasar dari jaringan syaraf tiruan:



Gambar 2.12 Matematis JST

Dalam gambar tersebut dijelaskan bahwa *input* dalam jaringan tersebut adalah x_1, x_2,…, x_m yang beranalogi dengan tingkat rangsangan yang datang dan kumpulan nilai bobot koneksi (*weight*) w_1, w_2,…, w_m yang secara biologis memiliki analogi dengan kekuatan *sinapsis* (*synaptic strengths*) yang dimiliki *neuron* (Kristanto, 2004:21).

Sebuah *neuron* memiliki 2 buah hubungan yaitu hubungan *Inhibitory* dan hubungan *excitatory*. Hubungan *Inhibitory* memberitahukan kepada *neuron* untuk tidak mengaktifkan sinyal *input*. Hubungan *excitatory* memberitahukan kepada *neuron* untuk membangkitkan sinyal *input*. Dari gambar tersebut atas dapat dinyatakan hubungan matematis yaitu perkalian antara *input* dan nilai bobot koneksi dan hasil perkalian tersebut dijumlahkan serta disimpan dalam *neuron*. Setelah itu dimasukkan dalam fungsi *nonlinear* f atau yang disebut dengan fungsi *aktivasi*. Apabila dimasukkan dalam rumus, maka persamaannya adalah :

$$y = f\left(x_1 * w_2 + x_2 * w_2 + ... + x_m * w_m\right) \quad (2.5)$$

Atau dalam notasi vektor

$$y = f(x * w) \tag{2.6}$$

Dimana :

x = vektor *input* yang terdiri dari m anggota

w = vektor bobot yang terdiri dari m anggota

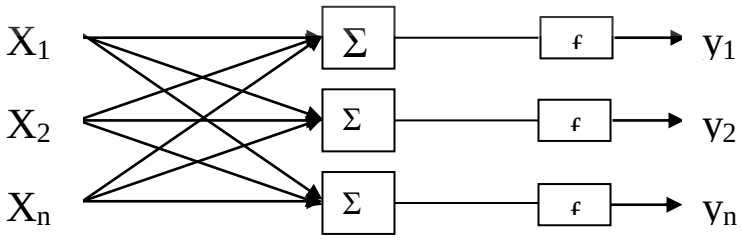
y = besaran skalar

f = fungsi *nonlinear* atau *aktivasi*

1. Model Jaringan Syaraf Tiruan Satu Lapisan

Jaringan syaraf tiruan satu lapisan pertama kali dirancang oleh Widrow dan Hoff pada tahun 1960. Walaupun jaringan syaraf tiruan satu lapisan ini sangat terbatas penggunaannya, namun konsep dan gagasannya banyak dipakai oleh beberapa pakar untuk membuat model jaringan syaraf tiruan banyak lapisan.

Dalam model jaringan syaraf tiruan satu lapisan, *sel-sel* syaraf tiruan digabungkan menjadi satu lapisan, berikut gambar model jaringan syaraf tiruan satu lapisan :

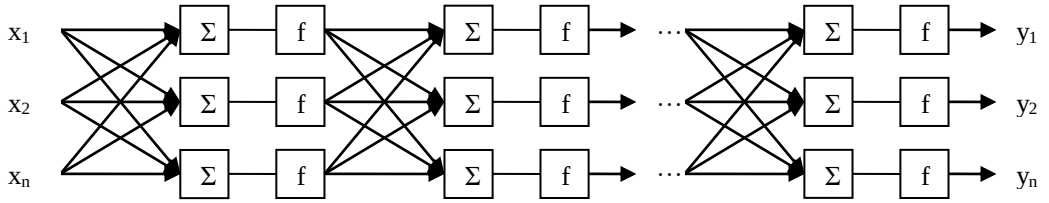


Gambar 2.13 Model Jaringan Syaraf Tiruan Satu Lapisan

Pada gambar di atas dapat dijelaskan bahwa *input* sepenuhnya dihubungkan dengan kumpulan sel syaraf dan *output*nya dapat dihitung dengan mengakumulasikan perkalian *input* ke-n dengan bobot koneksi ke-n juga.

2. Model Jaringan Syaraf Tiruan Banyak Lapisan

Dalam model ini sebenarnya merupakan model jaringan syaraf tiruan satu lapisan yang jumlahnya banyak dan prinsip kerja dari model ini sama dengan model jaringan syaraf tiruan satu lapisan. *Output* dari tiap lapisan sebelumnya merupakan *input* untuk lapisan berikutnya (Kristanto, 2004:23).



Gambar 2.14 Model Jaringan Syaraf Tiruan Banyak Lapisan

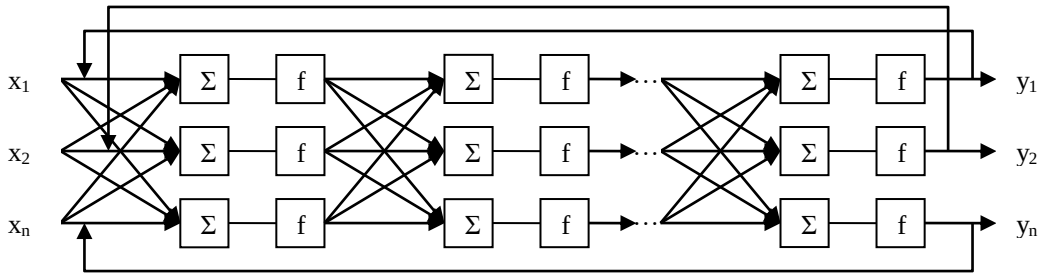
3. Model Jaringan Syaraf Tiruan dengan Umpan Balik

Tokoh yang pertama kali mencetuskan ide tentang model jaringan syaraf tiruan dengan umpan balik adalah John *Hopfield* dari *California Institute of Technology* pada tahun 1982. *Hopfield* berpendapat bahwa kumpulan *neuron* tiruan dalam jumlah yang sangat besar dapat melakukan tugas-tugas tertentu.

Hopfield juga membandingkan antara jumlah *neuron* pada binatang dengan jumlah *neuron* pada manusia. Sebagai contoh cacing dengan jumlah *neuron* diperkirakan 1000 buah dan pada manusia adalah mencapai 100 trilyun buah.

Dengan jumlah *neuron* yang sangat besar, Jaringan Syaraf Tiruan memiliki sifat yaitu *fault tolerance*. Sifat ini mengandung maksud kerusakan sedikit atau sebagian *sel-sel* dalam jaringan tidak akan mempengaruhi *output* yang akan dikeluarkan. Dengan adanya *output* yang diumpan balik ini maka *output*

yang yang dihasilkan akan mempengaruhi *input* yang akan masuk lagi ke dalam jaringan syaraf tersebut (Kristanto, 2004:24).



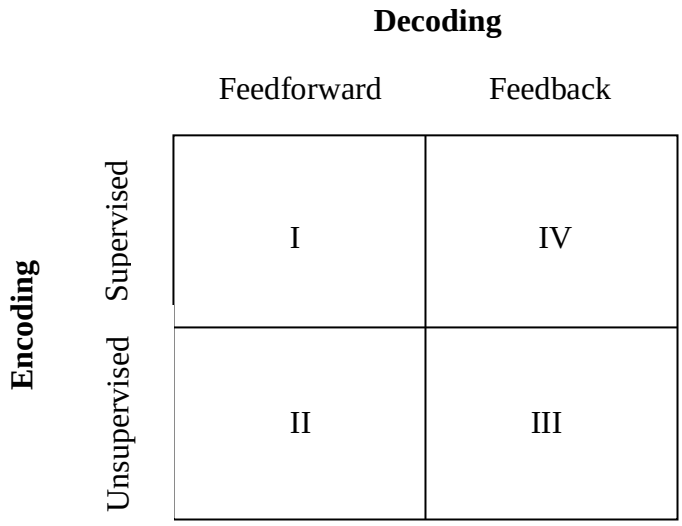
Gambar 2.15 Model Jaringan Syaraf Tiruan dengan Umpan Balik

2.13 Prinsip Dasar Pelatihan JST

Suatu karakteristik dari jaringan syaraf tiruan adalah kemampuannya untuk belajar. Cara belajar dari latihan yang diberikan padanya menunjukkan kesamaan dengan perkembangan intelektual manusia. Namun kemampuan belajar jaringan syaraf tiruan bersifat terbatas sehingga jaringan ini tidak dapat melakukan segalanya (Kristanto, 2004:27).

Proses pelatihan dan pembelajaran jaringan syaraf tiruan diilhami dari pakar yang bernama Kosko. Menurut Kosko (1990) mengklasifikasikan jaringan syaraf tiruan dengan dua definisi yaitu bagaimana jaringan syaraf tiruan ini menyimpan pengetahuan (*encode*) dan bagaimana jaringan syaraf tiruan ini menanggapi dan memproses data yang masuk (*decode*).

Berdasarkan *encode* dibedakan menjadi dua bagian yaitu *supervised* atau dibimbing dan *unsupervised* atau tidak dibimbing. Sedangkan berdasarkan *decode* dibedakan menjadi *feedforward* atau lurus dan *feedback* atau umpan balik.



Gambar 2.16 Klasifikasi Jaringan Syaraf Tiruan

Dari gambar tersebut dapat dijelaskan pembagian jaringan syaraf tiruan menjadi empat kuadran. Kuadran I adalah *supervised feedforward*. Dalam hal ini, jaringan syaraf tiruan dalam proses belajarnya yaitu harus dibimbing dalam penyimpanan pengetahuan atau datanya dan *input* yang masuk akan diteruskan serta tidak memberikan umpan balik.

Kuadran II adalah *unsupervised forward*, yang mana jaringan syaraf tiruan dalam proses belajarnya yaitu tidak harus dibimbing dalam penyimpanan pengetahuan atau datanya dan *input* yang masuk akan diteruskan serta tidak memberikan umpan balik. Kuadran III adalah *unsupervised feedback* yaitu jaringan syaraf tiruan dalam proses belajarnya tidak harus dibimbing dalam penyimpanan pengetahuan atau datanya dan *input* yang masuk akan diteruskan serta memberikan umpan balik. Dan kuadran IV adalah *supervised feedback* yaitu jaringan syaraf tiruan dalam proses belajarnya harus dibimbing dalam penyimpanan pengetahuan atau datanya dan *input* yang masuk akan diteruskan serta memberikan umpan balik.

Dalam konsep jaringan syaraf tiruan yang *supervised,* jaringan diberi masukan tertentu dan keluarnya ditentukan oleh pengajarnya. Dalam proses belajarnya, jaringan syaraf tiruan akan menyesuaikan bobot sinaptiknya. Sebaliknya pada jaringan syaraf tiruan yang *unsupervised*, secara mandiri jaringan syaraf tiruan akan mengatur keluarannya sesuai dengan aturan yang dimiliki jaringan tersebut.

Pada konsep *feedforward,* hasik *output*nya sudah dapat diketahui sebelumnya. Sedangkan pada *feedback* bersifat dinamis, artinya kondisi jaringan akan selalu berubah sampai diperoleh keseimbangan tertentu.

2.14 Penggolongan Jaringan Syaraf Tiruan

Karena setiap pakar dalam membuat model jaringan syaraf tiruan dengan konfigurasi *neuron* dan algoritma perhitungan yang berbeda-beda, maka banyak sekali nama-nama model jaringan yang muncul, antara lain : JST *Backpropagation,* JST *Kohoen,* JST *Hopfield,* dan masih banyak lagi yang lainnya.

Walaupun berbeda, model-model jaringan syaraf tiruan mempunyai kesamaan yaitu dalam hal aplikasi terhadap bidang tertentu. Adapun masalah atau bidang yang populer adalah masalah pengenalan pola. Dalam pengenalan pola jaringan syaraf tiruan dibagi menjadi dua jenis yaitu jenis *auto associative* dan *hetero associative*. Pada *auto associative output* yang dihasilkan dari sebuah jaringan syaraf tiruan sama dengan *input* yang dimasukkan padanya. Sedangkan pada *hetero associative input* berupa vektor yang akan dipetakan ke dalam *output* yang berupa vektor target (Kristanto, 2004:28).

Kemampuan *associative* jaringan syaraf tiruan adalah kemampuan untuk melakukan pemetaan *nonlinear* yang komplek dari vektor *input* yang diteruskan ke vektor *output*. Kumpulan dari pasangan vektor *input* dan vektor *output* yang diinginkan disebut pola pelatihan. Kumpulan dari pasangan vektor *input* dan vektor *output* yang diinginkan dimasukkan bersama dengan nilai bobot koneksi serta *neuron-neuron* tiruan dalam jaringan. Semuanya ini akan digunakan dalam proses belajar.

Proses belajar dimulai dengan memberikan satu vektor *input* ke dalam jaringan pada saat nilai target. Setelah vektor *input* masuk ke dalam jaringan, maka jaringan akan melakukan sebuah perhitungan sehingga menghasilkan *output* sementara. Selisih antara *output* yang diinginkan dengan *output* sementara akan dipakai untuk memperbaharui seluruh nilai bobot koneksi dengan jaringan.

Proses tersebut akan dilakukan secara berulang-ulang sehingga kesalahan yang dihasilkan mendekati angka yang kecil atau disebut dengan keadaan konvergen. Proses belajar akan selesai, apabila matrik koneksi yang diperoleh membuat suatu sistem yang dapat memberikan pola yang sempurna walaupun pola masukan yang diberikan kepada sebuah jaringan, informasinya hanya sepotong atau informasi yang kena derau.

2.15 Pembelajaran Tanpa Bimbingan (*Unsupervised learning*)

Metode ini tidak memerlukan pola *output* target untuk *output*-nya, sehingga tidak ada perbandingan antara *output* target dengan *output* jaringan. Kumpulan pola pelatihannya hanya berupa pola *input*. Pola *input* dimasukkan dalam jaringan dan jaringan sendiri yang harus *self organizing* sedemikian rupa

sehingga menghasilkan *output* yang sama dengan *input*-nya. (Laurene Fausett, 1994)

Pelatihan ini dilakukan dengan membangkitkan vektor masukan secara berurutan dan mengatur bobot jaringan menurut prosedur yang telah ditetapkan. Selama pelatihan, bobot jaringan secara berangsur-angsur menuju ke sebuah nilai yang sedemikian rupa sehingga setiap vektor masukan menghasilkan vektor keluaran yang diinginkan.

2.16 Fungsi Ativasi

Operasi dasar dari jaringan syaraf tiruan meliputi penjumlahan bobot sinyal dengan *input* yang menghasilkan *output*. Untuk menentukan bahwa *output* ini dapat digunakan dalam *neuron* dibutuhkan fungsi *aktivasi*. Fungsi *aktivasi* ini dapat berupa fungsi sigmoid yang mana nilai fungsinya terletak antar 0 dan 1 atau -1 dan 1. Fungsi *aktivasi sigmoid* yang nilainya diantara 0 dan 1 disebut dengan fungsi *biner sigmoid*. Sedangakan yang nilai fungsi *aktivasi*-nya diantara -1 dan 1 adalah fungsi *bipolar sigmoid* (Kristanto, 2004:28).

2.17 *Hebbian Learning*

Hebbian learning atau proses pembelajaran dengan metode *Hebbian* adalah metode pembelajaran yang tertua yang dikenalkan oleh Hellopt tahun 1949, dua pernyataan Hebb yang digunakan dalam *neurobiologi* (Blum, 1992:137) adalah :

1. Jika dua *neuron* pada salah satu sisi dari *sinapsis* (hubungan) ter*aktivasi* secara bersamaan (sinkron), maka kekuatan *sinapsis* yang terpilih akan ditingkatkan.

2. Jika dua *neuron* pada salah satu sisi dari *sinapsis* ter-*aktivasi* secara tidak bersamaan (asinkron), maka *sinapsis* yang terpilih akan dilemahkan atau dihilangkan.

Model pembelajaran dari Hebb, yaitu *Sinapsis weight* (bobot) adalah W_{kj} dari *neuron* k dengan sinyal pras*inapsis* (*presynaptic*) dari j akhir *sinapsis* (*postsynaptic*) yang diwakilkan dengan x_j dan y_k. pengaturan diterapkan pada bobot *sinapsis* W_{kj} pada waktu n seperti bentuk dibawah ini :

$$\Delta w_{kj}(n) = F\left(y_k(n), x_j(n)\right) \tag{2.7}$$

F (.,.) adalah fungsi dari sinyal *sinapsis*

x_j dan y_k. sering diartikan sebagai dimensi

n adalah menyatakan elemen

2.18 Aplikasi Jaringan Syaraf Tiruan

Ada beberapa aplikasi dari jaringan syaraf tiruan yang ini banyak digunakan dalam berbagai bidang, antar alain : pemrosesan sinyal, pengenalan pola, kedokteran bahkan untuk meramal suatu keadaan, dan masih banyak lagi yang lainnya (Kristanto, 2004:34).

a. Pemrosesan Sinyal

Jaringan syaraf tiruan dapat digunakan dalam pemrosesan sinyal yaitu contohnya untuk menekan gangguan suara pada *line* telepon. jaringan syaraf tiruan yang digunakan untuk kebutuhan ini adalah jaringan syaraf tiruan *Adaline*. Kebutuhan untuk membatalkan gema *adaptive* telah menjadi penekanan dengan perkembangan jaringan satelit *transkontinental* untuk sirkuit telepon jarak jauh.

Ide pembatalan gangguan *adaptive* cukutp sederhana. Pada akhir *line* jarak jauh, sinyal yang datang diaplikasikan ke dalam kedua komponen sistem

telepon (*Hybrid*) dan *filter adaptive* (tipe *Adaline* dari jaringan syaraf tiruan). Perbedaan diantara *output hybrid* dan *output Adaline* adalah tingkat kesalahan, yang digunakan untuk menyesuaikan bobot pada *Adaline*. Jaringan syaraf tiruan ini dilatih memindahkan gangguan dari sinyal *output Hybrid.*

b. Pengenalan Pola

Jaringan syaraf tiruan sangat populer digunakan untuk mengenali suatu pola tertentu. Contohnya untuk pengenalan pola huruf, tanda tangan, wajah manusia dan lainnya. Jaringan syaraf tiruan yang sering digunakan adalah jaringan syaraf tiruan metode *Backpropagation*.

c. Kedokteran

Aplikasi jaringan syaraf tiruan untuk kedokteran dikembangkan pada pertengahan tahun 1980-an oleh Anderson. Jaringan syaraf tiruan dilatih untuk menyimpan sejumlah data kedokteran yang meliputi informasi pada gejala, *diagnosis*, dan perawatan untuk hal-hal tertentu. Setelah dilatih jaringan tersebut nantinya akan diberi *input* gejala, dari *input* ini jaringan syaraf tiruan akan berusaha mengenalinya dari data yang pernah dilatih padanya. Sebagai *output* dari hasil pengenalan gejala ini bisa berupa diagnosis atau perawatan yang baik.

d. Peramalan

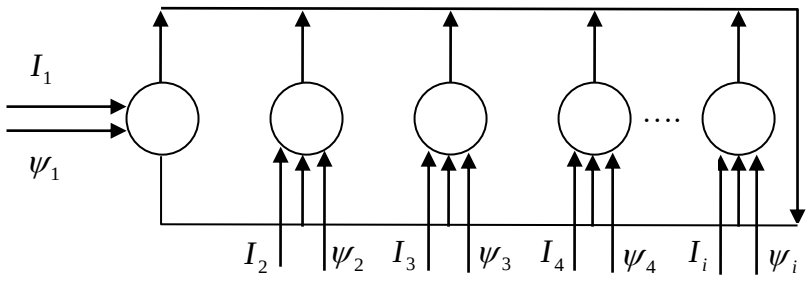
Jaringan syaraf tiruan juga dapat digunakan untuk peramalan suatu keadaan. Misalnya untuk peramalan cuaca, bursa saham atau yang lainnya. Aplikasi ini didasari dengan kemampuan dari jaringan syaraf tiruan ini untuk menyimpan suatu keadaan yang diberikan kepadanya. Dengan proses pembelajaran yang dilakukan padanya membuat jaringan syaraf tiruan dapat

memprediksikan keadaan berikutnya dengan cara menyimpulkan keadaan-keadaan sebelumnya.

2.19 Jaringan Syaraf *Hopfield*

Pada tahun 1982, John *Hopfield* dari California Institute of Technology merancang sebuah jaringan syaraf tiruan yang kemudian dikenal dengan jaringan syaraf *Hopfield*. Semua *neuron* saling berhubungan penuh, *neuron* yang satu mengeluarkan *output* dan kemudian menjadi *input* bagi semua *neuron* yang lain. Proses pengiriman dan penerimaan sinyal antar *neuron* ini secara *feedback* tertutup terus menerus sampai dicapai kondisi stabil (Kristanto, 2004:105).

Dalam model diskrit, jaringan *Hopfield* bobot sinaptiknya menggunakan vektor biner dimensi n atau $a\{0,1\}^n$. model semacam ini berisi *neuron* dan jaringan terdiri dari n(n-1) interkoneksi dua jalur. Secara matematis konsep diatas dapat disajikan dalam matrik n x n dengan diagonal utamanya benilai nol. Dengan kata lain, setiap *neuron* tidak memberi *input* kepada dirinya sendiri.



Gambar 2.17 Topologi JST *Hopfield*

Keterangan :

I_i : external *input*

ψ_i : bias

Pada gambar diatas ditunjukkan bahwa semua *neuron* saling berhubungan penuh. *Neuron* yang satu mengeluarkan *output* dan kemudian menjadi *input* bagi *neuron* yang lain. *Output* setiap simpul diumpan-balikkan ke *input* dari simpul lainnya melalui bobot koneksi W_{ij} yang tetap. Nilai W_{ij} mula-mula diinisialisasi menggunakan rumus yang diberikan *Hopfield* untuk seluruh pola-pola contoh.

Pola yang tidak dikenal dimasukkan ke jaringan satu kali saja yaitu pada waktu nol. Kemudian jaringan beriterasi hingga konvergen. Konvergen adalah kondisi dimana *output* yang sudah tidak berubah lagi untuk iterasi selanjutnya. Pola yang dinyatakan oleh simpul-simpul *output* setelah jaringan konvergen adalah *output* JST.

Kemampuan JST *Hopfield* untuk mengenali pola ditentukan dengan rumus 0.15 x n. n adalah banyaknya dimensi yang digunakan. Misalnya dimensi yang digunakan 100, maka pola yang akan dikenali sebanyak 15 buah. Dimensi adalah suatu vektor yang mewakili banyaknya baris dan kolom yang digunakan. Dari contoh diatas berarti baris yang digunakan berjumlah 100 dan kolom yang akan digunakan berjumlah 100 juga (Kristanto, 2004:107).

Berdasarkan prosedur yang diajukan oleh *Hopfield*, bobot sinaptiknya dari setiap *neuron* dibangkitkan melalui operasi logika dari vektor-vektor biner yang disimpan pada jaringan tersebut. Misalkan ada tiga 9 dimensi vektor yaitu α, β, dan γ. Ketiga vektor dimensi tersebut adalah :

α = 101 010 101

β = 110 011 001

γ = 010 010 010

Apabila setiap nilai 0 diganti dengan nilai -1, maka akan diperoleh bentuk *bipolar* dari masing-masing vektor, sehingga :

α = 1-11 1-1-1 1-11

β = 11-1 -111 -1-11

γ = -11-1 -11-1 -11-1

Untuk menghitung bobot matrik diperoleh dengan cara menghasilkan setiap vektor dengan dirinya sendiri dan kemudian dijumlahkan :

$$w = \alpha^{'}\alpha^{'} + \beta^{'}\beta^{'} + \gamma^{'}\gamma^{'}$$

Apabila diagonal utama dari matrik W ini dibuat 0, maka akan diperoleh bobot *sinaptik* dari jaringan tersebut. Nilai dari setiap elemen matrik merupakan bobot yang dikenakan pada proses pengiriman sinyal dari sumbu vertikal ke sumbu horisontal yang terkait dengan elemen tersebut.

2.20 *Discrete Hopfield Network*

Jaringan syaraf *Hopfield* dibagi menjadi dua, yaitu *Continous Hopfield* dan *Discrete Hopfield*. Untuk *Continous Hopfield* didasarkan pada model *additive*, sedangkan *Discrete Hopfield* berdasarkan model McCulloch-Pitts. Pada *Continous* diijinkan untuk terjadi looping terhadap dirinya atau self-loop, tetapi pada *discrete* tidak boleh ada *self-loop*. Jaringan *Iterative Autoassociative* hampir sama dengan jaringan yang dikembangkan oleh *Hopfield* (1982, 1984). *Hopfield* menggunakan proses pembelajaran dari *Hebb Learning*, dengan menghitung w, *weight* (bobot) dari *neuron-neuron*nya, jaringan ini terkoneksi penuh diantara *neuron* satu dengan yang lainnya (Pratomo, 2002:15).

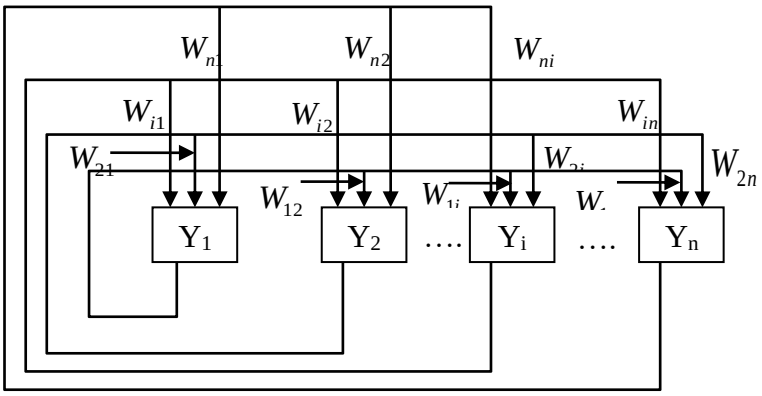
Perbedaannya dengan *Interative Associative* yaitu :

1. Hanya satu unit yang memperbarui aktivasinya pada sekali waktu, berdasar pada sinyal yang diterima dari unit lain.
2. Masing-masing unit berkelanjutan untuk menerima sinyal dari luar, untuk penambahan sinyal dari unit lain dalam jaringan.

Formula asli dari *Discrete Hopfield* ditujukan untuk kegunaan *content-addressable memory*. Kemudian *Hopfield* dengan Tank (1985) menggunakannya untuk assosiasi pola atau optimasi.

2.21 Arsiktektur JST *Hopfield*

Model atau arsitektur dari jaringan *Hopfield* ditunjukkan pada gambar dibawah ini. Dimana dalam gambar tersebut terdiri dari n buah simpul yang mengandung *nonlinearitas hard limiter* dan *input* serta *output* biner yang dapat bernilai +1 atau -1. Tiap-tiap *neuron* saling berhubungan satu dengan yang lainnya, *output* dari dari suatu *neuron* merupakan *input* bagi *neuron-neuron* yang lainnya.



Gambar2.18 Arsitektur Jaringan *Hopfield*

Dari gambar diatas, dapat dijelaskan bahwa *output* setiap simpul diumpan-balikkan ke *input* dari simpul lainnya melalui bobot koneksi W_{ij} yang tetap. Nilai W_{ij} mula-mula diinisialisasi menggunakan algoritma *Hopfield* untuk seluruh pola yang masuk. Pola yang tidak dikenal dimasukkan ke dalam jaringan sekali saja pada waktu kondisi nol. Kemudian jaringan akan beriterasi sampai *output* yang dihasilkan bersifat konvergen. Nilai konvergen ini tidak akan berubah sampai iterasi selesai pola yang dinyatakan oleh simpul-simpul *output* setelah jaringan konvergen adalah *output* JST (Kristanto, 2004:112).

Untuk mencapai kovergen jaringan harus diupdate dalam setiap iterasi. Dalam jaringan syaraf tiruan ini terdapat empat cara mengupdate jaringan (Babadi, 2004:10), yaitu:

a. *Serial-Random*

   Untuk meng-*update* kondisi *output* dari jaringan dilakukan dengan cara memilih suatu *neuron* secara acak (*random*) untuk dimasukkan dalam jaringan.

b. *Serial-Sequential*

   Untuk meng-*update output* jaringan dilakukan secara berurutan pada tiap-tiap *neuron*.

c. *Parallel-Synchronous*

   Semua *neuron* akan di-*update* secara serempak untuk mencari *output* jaringan pada suatu iterasi.

d. *Parallel-Asynchronous*

   Semua *neuron* akan di-*update* secara tidak bersamaan untuk mencari *output* jaringan pada suatu iterasi.

2.22 Algoritma JST *Hopfield*

Ada beberapa versi dari algoritma jaringan syaraf *Hopfield* (Kristanto, 2000:112). Untuk deskripsi dari algoritma *Hopfield* (1982) menggunakan *input* vektor yang berupa angka biner. Untuk menyimpan sekumpulan pola biner digunakan notasi :

$S(p) = (S_1(p),\ldots, S_i(p),\ldots, S_n(p))$ (2.8)

Sedangkan untuk menyimpan bobot matrik $W = \{W_{ij}\}$, dengan notasi :

$$W_{ij} = \sum_{p=1}^{p}[2s_i(p)-1][2s_j(p)-1] \quad \text{untuk } i \neq j \text{ dan } W_{ij} = 0 \tag{2.9}$$

Sedangkan versi dari deskripsi algoritma *Hopfield* lainnya (1984) menggunakan *input* vektor yang berupa angka *bipolar*. Untuk menyimpan sekumpulan pola *bipolar* digunakan notasi S(p), p = 1,…,P dimana :

$S(p) = (S_1(p),\ldots, S_i(p),\ldots, S_n(p))$

Sedangkan untuk menyimpan bobot matrik $W = \{W_{ij}\}$, dengan notasi :

$$W_{ij} = \sum_{p=1}^{p}[s_i(p)s_j(p)] \quad \text{untuk } i \neq j \text{ dan } W_{ij} = 0 \tag{2.10}$$

Dan algritma untuk mencari output jaringan syaraf *Hopfield* adalah sebagai berikut :

$$y\_in_i = x_i + \sum_{j=1}^{n} y_j(W_{ij}) \tag{2.11}$$

Dimana : y_in : Output yang dicari

x_i : Bias

y_j : Input dari neuron lain

W_{ij} : Bobot koneksi

Berikut penjelasan algoritma *Hopfield* dari penjelasan perhitungan bobot matriks dan perhitungan output jaringan syaraf *Hopfield* :

1. Menciptakan vektor pola yang ke-p, yang merupakan himpunan n simpul untuk pola yang ke p.

   $S(p) = (S_1(p),S_2(p),\ldots,S_i(p),\ldots,S_n(p))$

   Dengan : p adalah jumlah pola, i adalah indek dari vektor

   S adalah Vektor

2. Menciptakan matriks koneksi sebagai tempat pola.

   Inisialisasi matriks W (orde nxn) untuk menyimpan pola

$$W_{ij} = \begin{bmatrix} \sum_{p}^{10} S_i(p) * S_j(p) \ldots i \neq j \\ 0 \ldots i = j \end{bmatrix} \quad (2.12)$$

   Dengan ;

   p = jumlah pola

   W_{ij} = bobot koneksi dari simpul i ke simpul j

   $S_i(p)$ = elemen in dari pola s yang hanya memiliki nilai +1 atau -1

   $S_j(p)$ = elemen in dari pola s yang hanya memiliki nilai +1 atau -1

3. Kerjakan langkah-langkah ini selama pola belum konvergen.

   Langkah a : untuk tiap vektor x (x1, x2, ..., xn)

   Langkah b : inisialisasi pola input yang tidak diketahui

$$Y_i = X_i \quad (2.13)$$

   Dimana : i =1..n

   Langkah c : kerjakan lankah b sampai semua pola diuji.

   Langkah d :

$$y\_in_i = x_i + \sum_{j=1}^{n} y_j (W_{ij})$$

Langkah e *determine activation* (sinyal *output*)

$$y_i = \begin{Bmatrix} 1 jika \ldots y\_in_i \geq 0 \\ -1 jika \ldots y\_in_i < 0 \end{Bmatrix} \quad (2.14)$$

Keterangan : y_i = output ke-i

Langkah f : hasil dari *output* y_i berlaku untuk semua pola lain.

Langkah g : uji pola sampai konvergen (Kristanto, 2004:113).

2.23 Keadaan Palsu (*Sparious State*)

Keadaan palsu merupakan keadaan stabil yang terbentuk dari jaringan *Hopfield* yang berbeda dari memori dasarnya dan menyebabkan terjadinya kesalahan pengenalan pola.

Penyebab *Sparious State* digolongkan menjadi tiga (Pratomo, 2002:22) yaitu :

1. *Reserved Fundamental Memory* yaitu suatu pola *input* merupakan pola yang terdiri dari bit-bit yang berkebalikan dari pola dalam memori dasar.
2. *Mixture State* yaitu suatu pola *input* yang merupakan hasil dari penggabungan sejumlah pola-pola dengan urutan ganjil.
3. *Spin-Glass States* yaitu suatu pola *input* yang sama sekali berbeda dengan pola-pola memori dasar.

2.24 Kapasitas Penyimpanan

Dalam mengenali pola yang telah diberi gangguan atau derau, *Hopfield* memberikan rumus yaitu $P \approx 0.15xn$, dimana n adalah jumlah *neuron* yang ada dalam jaringan. Tetapi dalam jaringan *Hopfield* memiliki dua keterbatasan yaitu sebagai berikut :

1. Jika terlalu banyak pola yang disimpan, maka jaringan akan konvergen kepada pola yang diberikan. Untuk menghindarkan hal tersebut, *Hopfield* memberikan rumus empiris s<0.15n atau jumlah pola pelatihan harus lebih kecil dari 0.15 kali jumlah simpul dalam jaringan.
2. Jika pola-pola contoh terlalu banyak memiliki kesamaan maka jaringan menjadi tidak stabil yaitu jaringan akan konvergen dengan pola lain (Kristanto, 2004, 117).

2.25 Fungsi Energi

Fungsi energi dari jaringan *Hopfield* (Babadi, 2004:15) adalah :

$$E = -\frac{1}{2}\sum_{j}\sum_{\substack{i=1 \\ i\neq j}}^{N} w_{i,j}u_i\, u_j \tag{2.15}$$

Dari rumus tersebut di atas dapat dijabarkan sebagai berikut :

$$E = -\frac{1}{2}\sum_{i} u_i \sum_{\substack{j=1 \\ i\neq j}}^{N} w_{i,j}u_j \tag{2.16}$$

Sedangkan untuk mencari output JST pada persamaan (2.8) adalah :

$$y\_in_i = x_i + \sum_{j=1}^{N} y_j(W_{ij})$$

Jadi rumus energi setelah disederhanakan adalah :

$$E = -\frac{1}{2}\sum_{j} u_j y\_in_i \tag{2.17}$$

Keterangan : E adalah Energi

u_i adalah output jaringan syaraf

u_j adalah hasil aktivasi output

w_{ji} adalah bobot koneksi jaringan

i, j adalah indek 0…N

N adalah jumlah Neuron

2.26 *Hamming Distance*

Hamming Distance digunakan untuk menghitung dua pola yang mempunyai posisi bit atau vektor yang berbeda (Fausett, 1994:147). Dua pola yang berbeda itu, yaitu x_1 dan x_2 dengan menggunakan formula :

$$\frac{1}{N} H[x_1, x_2] \tag{2.18}$$

Dimana n adalah jumlah bit yang digunakan dan H adalah fungsi *Hamming Distance*. Maka akan didapatkan nilai yang terkecil dari perbedaan diantara pola-pola yang dibandingkan, yang berarti mempunyai kemungkinan diantaranya.

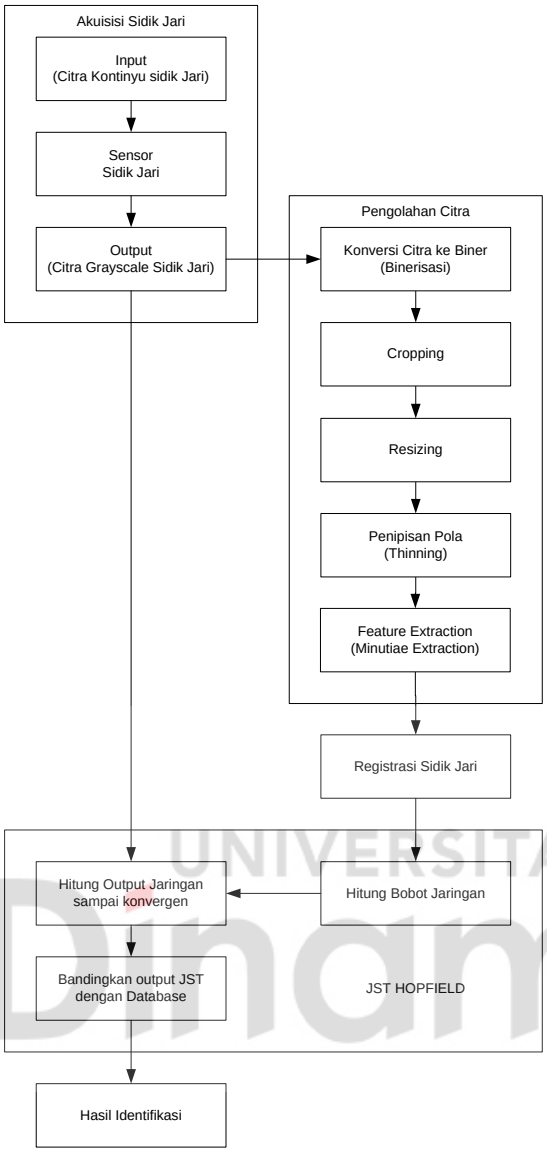
BAB III

METODE PENELITIAN

Metode penelitian yang digunakan pada pembuatan perangkat lunak (*software*) yaitu berasal dari studi kepustakaan. Dengan cara ini penulis berusaha untuk mendapatkan dan mengumpulkan data-data, informasi, konsep-konsep yang bersifat teoritis dari buku, bahan-bahan kuliah dan internet yang berkaitan dengan permasalahan yang akan diselesaikan.

Dalam bab ini akan dijelaskan tentang cara memperoleh data dan praproses data sehingga pengolahan data menjadi siap untuk digunakan sebagai masukan pada jaringan syaraf tiruan serta cara pembuatan perangkat lunak Perancangan sistem identifikasi sidik jari ini merupakan penelitian yang 100% *software*. Sehingga dalam perancangannya hanya meliputi perancangan *software* saja. Namun dalam perancangan *software* ini masih dibagi lagi menjadi 4 tahap. Masing-masing adalah tahap akuisisi citra sidik jari (digitasi citra), tahap pengolahan citra, tahap registrasi sidik jari dan tahap perancangan jaringan syaraf *Hopfield*.

Pada tahap akuisisi sidik jari dijelaskan tentang bagaimana prosedur pengambilan pola sidik jari oleh sensor, tahap pengolahan citra menjelaskan proses pengolahan citra sidik jari, tahap registrasi sidik jari menjelaskan bagaimana prosedur penyimpanan data sidik jari ke dalam *database*. Sedangkan tahap perancangan dan pembuatan jaringan syaraf *Hopfield* ini terdiri dari proses pembelajaran pola sidik jari dan proses pengenalan pola sidik jari. Berikut blok diagram dari sistem :



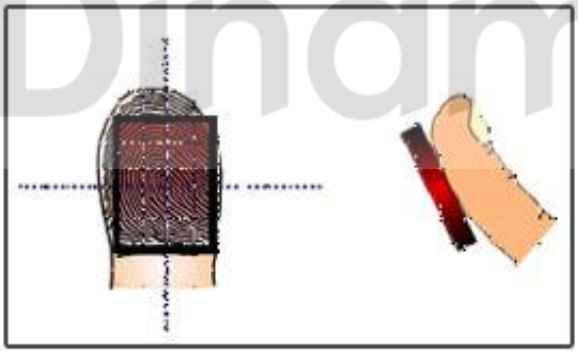
Gambar 3.1. Blok Diagram Sistem

3.1 Akuisisi Citra Sidik Jari

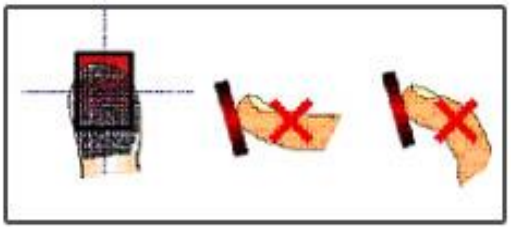
Input dari sistem ini adalah citra sidik jari yang berupa citra analog, sehingga supaya sidik jari yang akan dimasukkan pada sistem dapat diproses, dibutuhkan proses digitalisasi. Jadi citra analog sidik jari dirubah menjadi citra digital sidik jari. Dalam penelitian ini citra sidik jari yang digunakan adalah sidik jari pada bagian ibu jari.

Sedangkan sensor yang digunakan untuk membaca sidik jari dalam penelitian ini adalah sensor U.are.U 4000 Digital Persona USA. Hasil pembacaan sensor sidik jari ini adalah citra *grayscale* dengan ukuran 500x550 *pixel* dan resolusinya 512 dpi. Pada awalnya sensor yang akan digunakan adalah sensor yang dimana hasil pembacaan dari sidik jari adalah berupa citra biner, namun karena sensor yang seperti ini susah dicari dalam pasaran maka peneliti menggunakan sensor U.are.U 4000, oleh karena itu dibutuhkan proses binerisasi untuk mencari citra binernya.

Untuk mendapatkan area sidik jari yang bagus sehingga dapat mewakili daerah pada jari yang diletakkan pada sensor, dibutuhkan kecermatan dalam meletakkan jari pada kaca optik sensor. Posisi jari pada sensor harus tegak lurus dengan daerah optik sensor. Atau lebih jelasnya dapat dilihat dari gambar berikut :



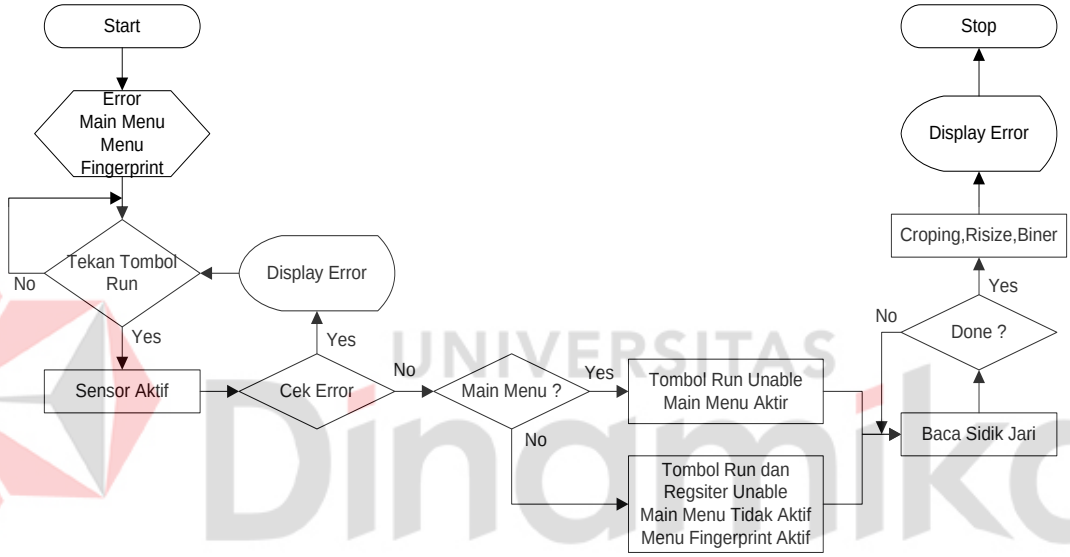
Gambar 3.2 Posisi Jari Yang Dianjurkan



Gambar 3.3 Posisi Jari Yang Tidak Dianjurkan

3.1.1 Perancangan Akuisisi Citra Sidik Jari

Untuk mengoperasikan sensor ini dibutuhkan *driver* yang harus di-*instal* terlebih dahulu, *driver* ini berfungsi sebagai *interface* antara program aplikasi dengan sensor tersebut. Setelah *driver* ter-*instal* sensor ini siap untuk digunakan. Di dalam *driver* ini tersedia berbagai komponen yang dapat digunakan untuk operasi pembacaan sidik jari.



Gambar 3.4 Diagram Alur Input Sidik Jari

Dari diagram alur tersebut diatas dapat dijelaskan bahwa pada sensor U.are.U teradapat dua proses untuk membaca sidik jari. Pertama sensor ini harus diaktifkan terlebih dahulu oleh program, yang mana keadaan ini dapat disebut juga keadaan siap untuk membaca sidik jari. Program juga akan mengecek apakah ada *error* yang dihasilkan pada saat pembacaan sensor. Setelah tidak ada *error*, sensor akan membaca sidik jari dan memberi sinyal bahwa sidik jari telah dibaca dan data sidik jari ini akan disimpan pada variabel supaya dapat diproses lebih lanjut.

Pada proses akuisisi citra sidik jari ini akan digunakan pada saat pendaftaran atau registrasi sidik jari dan saat identifikasi sidik jari. Dalam tugas akhir ini, pada saat pengaktifan sensor sidik jari akan ditandai proses mana yang sedang memanggil atau mengaktifkan sensor. Tanda ini akan menentukan proses selanjutnya apakah untuk proses pembelajaran sidik jari oleh jaringan syaraf *Hopfield* atau untuk identifikasi sidik jari oleh jaringan syaraf *Hopfield*.

3.1.2 Listing dan Analisis Akuisisi Sidik Jari

```
Dim err As DPSDKOPSLib.AIErrors
Dim serial As String = "54900066"

Me.stbStatus.Panels("lblStatus").Text = "Initializing..."
Me.picFingerprintsMenuSample.Image = Nothing
Application.DoEvents()
Me.fpxSample.SelectDevice(serial)
err = Me.fpxSample.Run()
If (err = DPSDKOPSLib.AIErrors.Er_OK) Then
    Me.stbStatus.Panels("lblStatus").Text = "Please put your finger in the
sensor..."
    If (sender Is Me.btnMainMenuRun) Then
        Me.btnMainMenuRun.Enabled = False
        Me.isMainMenuRun = True
        ElseIf (sender Is Me.btnFingerprintsMenuRun) Then
            Me.btnFingerprintsMenuRun.Enabled = False
            Me.btnRegisterFingerprint.Enabled = False
            Me.cmdRegisterFingerprint.IsEnabled = False
            Me.isMainMenuRun = False
     End If
 Else
    Me.stbStatus.Panels("lblStatus").Text = "Initialize failed"
    MessageBox.Show(Me, Me.GetAIErrorsToString(err), Me.Text,
    MessageBoxButtons.OK,
    MessageBoxIcon.Error)
 End If
```

Dari potongan program di atas dapat dijelaskan bahwa untuk memerintahkan sensor untuk membaca sidik jari dibutuhkan komponen bawaan dari alat sensor sidik jari ini. Untuk itu *driver* dari alat ini harus di-instal-kan ke dalam *operating system* komputer ini. Komponen yang dibutuhkan untuk membaca sidik jari ini adalah berupa *ActiveX* dengan nama *AxFPGetSampleX*. Variabel *fpxSample* pada program tersebut diatas dideklarasikan sebagai *AxFPGetSampleX*. Setelah variabel ini dideklarasikan maka variabel ini dapat

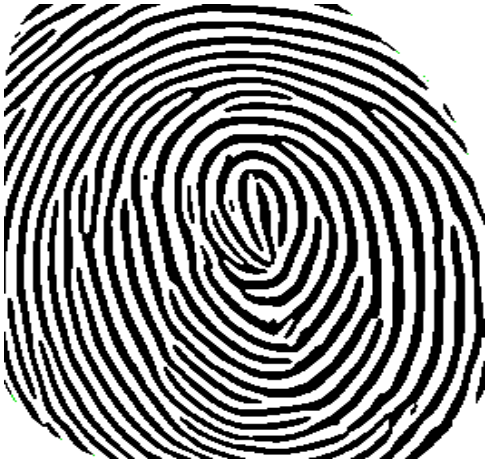
digunakan didalam program sesuai dengan perintah-perintah yang ada pada *AxFPGetSampleX*. Untuk mengoperasikan agar sensor ini dapat membaca sidik jari cukup dengan perintah *fpxSample.Run*.

Pada saat perintah *Run* dijalankan sensor dalam keadaan siap untuk membaca sidik jari yang akan dimasukkan. Tapi sebelumnya program ini akan membaca *error* yang akan dikirim oleh driver, *error* ini mungkin berupa peringatan bahwa sensor belum dihubungkan atau *error* yang lainnya. Jika tidak ada *error* maka sensor ini siap untuk menerima masukan. Saat ada sidik jari yang terbaca oleh sensor maka sensor ini akan memberi sinyal bahwa sidik jari telah selesai dibaca. Berikut potongan programnya :

```
Private Sub fpxSample_Done(ByVal sender As System.Object, ByVal e As
AxDPSDKOPSLib. _IFPGetSampleXEvents_DoneEvent) Handles fpxSample.Done
    Dim asli As Bitmap
    Dim bmp As Bitmap
        Me.sample.Import.FromPicture(e.pSample.Picture)
```

Pada prosedur diatas menunjukkan bahwa sidik jari telah terbaca dan data sidik jari yang telah dibaca masih dalam tipe data *object*. Agar data sidik jari ini dapat diproses lebih lanjut, maka dalam potongan program tersebut diatas dideklarasikan variabel *sample* sebagai penampung. Dalam program ini *sample* dideklarasikan sebagai berikut : `Private sample As Atalasoft.ImgX6Net.ImgX`

Jadi dalam program sistem ini juga menggunakan *data link library* (*dll*) dari *Atalasoft*. *Atalasoft* ini adalah suatu *software* yang menangani berbagai macam proses pengolahan citra (*Image Processing*). Di dalam penelitian ini penggunaan *software* hanya dalam proses penampungan gambar dan proses-proses pengolahan citra sederhana, seperti konversi citra sidik jari ke biner, pemotongan gambar, dan merubah ukuran gambar.



Gambar 3.5 Hasil Pembacaan Sensor

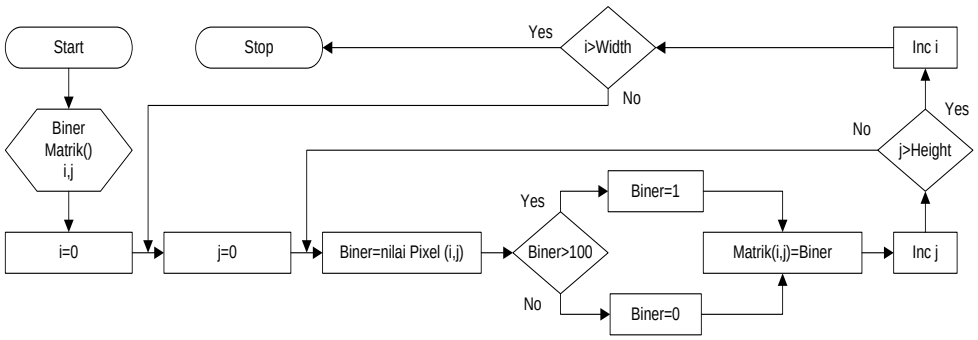
3.2 Pengolahan Citra Sidik Jari

Pada tahap pengolahan citra ini, citra hasil pembacaan sensor akan diolah untuk mendapatkan citra yang benar-benar bagus dan mengambil ciri bagian dari citra tersebut yang dapat mewakili citra secara keseluruhan. Pengolahan citra yang akan dilakukan meliputi proses binerisasi, *cropping*, *resizing*, penipisan pola (*Thinning*), dan pengambilan ciri-ciri guratan dari sidik jari (*Minutiae Extraction*). Hasil pengolahan citra ini nantinya yang akan digunakan dalam proses berikutnya. Berikut tahap-tahap pengolahan citranya :

3.2.1 Binerisasi

Citra sidik jari hasil pembacaan sensor yang berupa citra grayscale ini akan dilakukan proses binerisasi untuk mendapatkan nilai *pixel* 0 dan 1. Untuk mendapatkan citra biner maka tiap *pixel* pada citra akan dilakukan proses pengambangan secara global (*global image Thresholding*). Dalam proses ini akan ditentukan nilai ambang sama dengan 127.

A. Perancangan Proses Binerisasi



Gambar 3.6 Diagram Alur Proses Binerisasi

Dari diagram alur di atas dapat dijelaskan langkah-langkah proses binersasi sebagai berikut :

1. Pertama ambil nilai *pixel* pada koordinat awal dari citra kemudian disimpan pada variabel penampung.
2. Bandingkan nilai *pixel* ini dengan nilai ambang, jika nilai *pixel* lebih besar dari 127 maka nilai binernya sama dengan = 1, sedangkan jika nilai *pixel* lebih kecil dari 127 maka nilai binernya sama dengan = 0.
3. Simpan hasil pengambangan ini pada matrik penampung.
4. Lakukan proses ini sampai semua *pixel* pada citra telah dilakukan pemeriksaan dan telah dirubah nilainya menjadi nilai biner.

B. Listing dan Analisa Proses Binerisasi

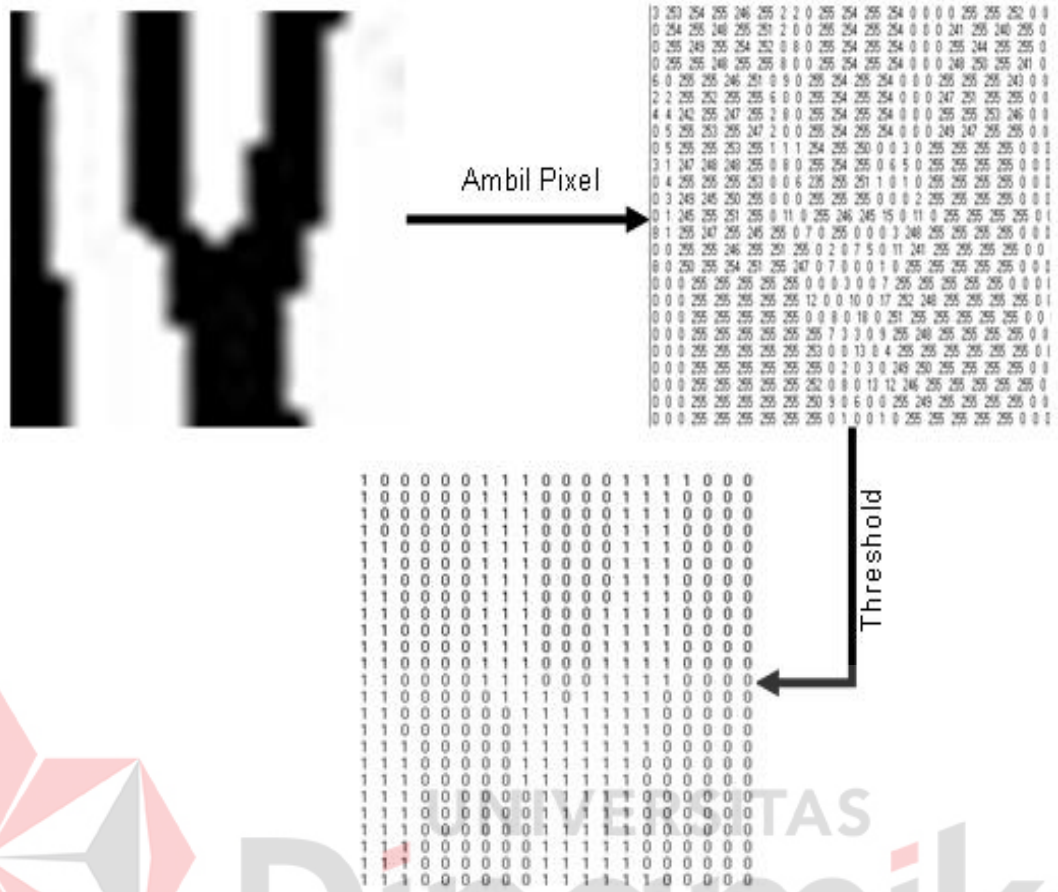
```
Private Sub GetSamplePattern1()
    Dim tem As String
    Dim bil As Integer
    For i As Integer = 0 To IMAGE_HEIGHT - 1
        For j As Integer = 0 To IMAGE_WIDTH - 1
            tem = Me.sample1.Image.GetPixel(j, i).Index.ToString()
            GetRGB (tem)
            tem = (Rred + Ggreen + Bblue)
             tem = (temp / 3)
            if  tem >127 then
               tem = 1
            else
               tem =0
            end if
             bil = CInt(tem)
```

```
            matrik(j, i) = bil
        Next
    Next
    End Sub
```

Untuk mencari nilai biner dari citra sidik jari digunakan dua buah *looping* untuk mengambil nilai *pixel* dari citra tersebut. Citra hasil pengambilan *pixel* ini bernilai antara 0 sampai dengan 255, sehingga untuk mencari nilai binernya dapat dicari dengan cara membandingkan nilai *pixel* ini dengan nilai ambang. Nilai ambang yang digunakan pada program tersebut adalah 127. Jika nilai *pixel* lebih besar dari atau sama dengan nilai 127 maka nilai binernya adalah sama dengan 0 sedangkan jika nilai *pixel*-nya lebih kecil dari 127 maka nilai binernya adalah sama dengan 1. Hasil binerisasi pada program akan disimpan pada matrik sebagai tempat penyimpanan. Berikut bagian citra sidik jari yang akan dipakai sebagai contoh hasil binerisasi

Gambar 3.7 Contoh Bagian Yang Akan Dicari Binernya

Untuk proses Binerisasi ini yang akan dicontohkan hanya *pixel* yang diberi tanda diatas, karena ukuran sebenarnya dari citra tersebut sangat besar. Berikut proses binerisasi :

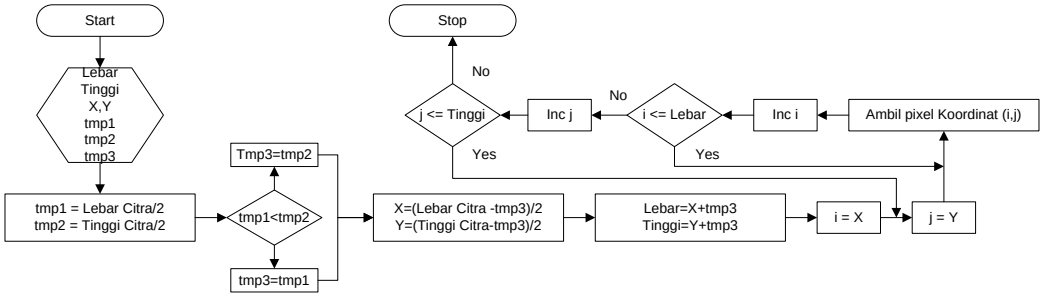


Gambar 3.8 Hasil Proses Binerisasi

3.2.2 *Cropping*

Setelah hasil pembacaan sidik jari yang berukuran 500x550 *pixel* ini dilakukan proses binerisasi maka selanjutnya akan dipotong atau dilakukan proses *cropping* pada koordinat tertentu. Area citra sidik jari yang akan dipotong adalah dibagian tengah dari citra. Pemotongan citra sidik jari ini bertujuan agar area citra sidik jari yang akan dipelajari guratan-guratannya tidak terlalu banyak. Dengan adanya proses *cropping* ini ukuran dari citra sidik jari ini akan berubah menjadi setengah dari citra asalnya.

B. Perancangan Proses *Cropping*



Gambar 3.9 Diagram Alur *Cropping*

Dari diagram alur tersebut di atas dapat dijelaskan bahwa untuk memotong citra sidik jari pada suatu koordinat tertentu dapat dijelaskan dalam langkah-langkah sebagai berikut :

1. Pertama ukuran lebar dan tinggi dari citra dibagi dengan 2, kemudian simpan pada variabel penampung tmp1 dan tmp2.
2. Bandingkan tmp1 dengan tmp2, jika tmp1 lebih kecil dari tmp2 maka tmp3 sama dengan tmp1, dan jika tmp1 lebih besar dari tmp2 maka tmp3 sama dengan tmp2.
3. Cari koordinat titik awal dari proses *cropping* dengan cara lebar citra dikurangi dengan tmp3 dan hasilnya dibagi dengan 2, untuk tinggi citra dikurangi dengan tmp3 dan hasilnya juga dibagi dengan 2..
4. Kemudian tentukan lebar dan tinggi untuk citra yang akan dipotong ini dengan menjumlahkan titik awal pemotongan dengan tmp3 dan tmp3.
5. Berikutnya ambil *pixel* pada koordinat yang telah ditentukan dan ditampung pada variabel penampung.

Berdasarkan ukuran citra hasil pembacaan sensor adalah citra 500x550, sehingga dapat dilakukan perhitungan pemotongan citra sebagai berikut :

tmp1 = lebar citra / 2 = 500 / 2 = 250

tmp2 = tinggi citra / 2 = 550 = 275

tmp1 < tmp2 ⟶ tmp3 = tmp1 = 250

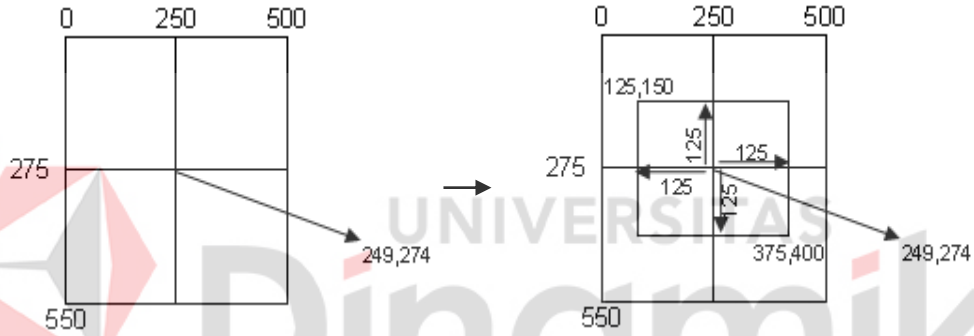
X = (lebar citra – tmp1) /2 = (500-250) / 2 = 125

Y = (tinggi citra-tmp2) / 2 = (550-250) / 2 = 150

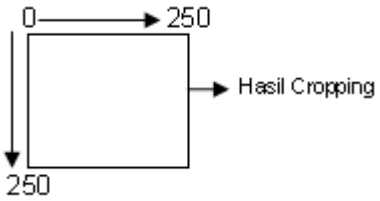
Lebar = tmp1 = 250

Tinggi = tmp = 250

Maksud dari perhitungan tersebut adalah sebagai berikut :



Gambar 3.10 Proses *Cropping*



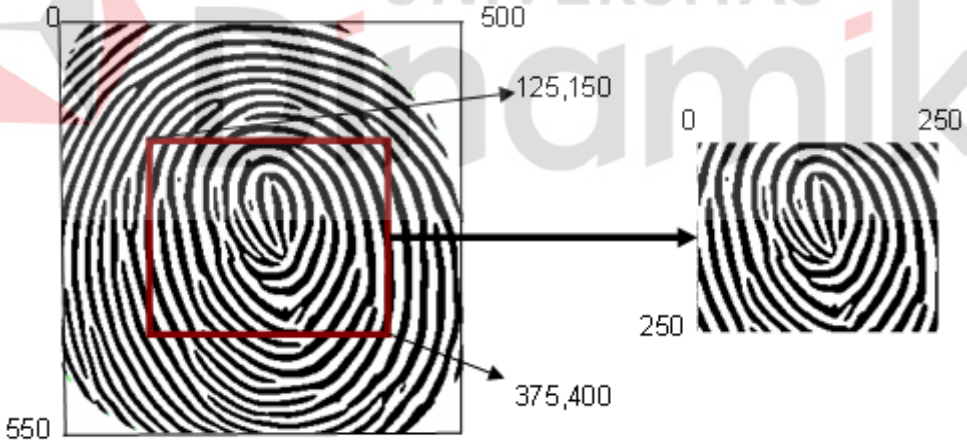
Gambar 3.11 Hasil Proses *Cropping*

Sehingga citra yang akan dipotong dimulai dari koordinat titik *pixel* (125,150) sampai koordinat akhir pada titik *pixel* (375,400). Kemudian hasil pembacaan *pixel* ini akan ditampung pada variabel untuk memudahkan mengkonversi lagi menjadi bentuk citra.

C. Listing dan Analisa Proses *Cropping*

```
Dim tinggi As Integer
Dim lebar As Integer
Dim tmp as Integer
tinggi = Me.sample.Image.Height / 2
lebar = Me.sample.Image.Width / 2
if tinggi > lebar then
    tmp = lebar
else
    tmp = tinggi
end if
Me.sample.Effects.Crop((Me.sample.Image.Width - tmp) / 2,_
(Me.sample.Image.Height - tmp) / 2, tmp, tmp)
```

Dari potongan program tersebut di atas dapat dijelaskan bahwa untuk memotong citra sidik jari dibutuhkan variabel sebagai parameter, antara lain : tinggi dan lebar berfungsi untuk menyimpan ukuran citra dan variabel tmp sebagai tempat peyimpanan jika ukuran citra pada lebar tidak sama dengan tingginya. Proses *cropping* ini menggunakan *Data Link Library* dari *Atalasoft*. Berikut hasil proses *cropping* :



Gambar 3.12 Proses *Cropping*

3.2.3 *Resizing*

Setelah proses *cropping* dilakukan, citra sidik jari ini masih akan diproses lagi untuk dirubah ukurannya menjadi lebih kecil lagi. Dalam penelitian ini digunakan citra 50x50 *pixel* dan 70x70 *pixel* yang akan dipelajari oleh sidik jari

Hopfield. Proses ini dilakukan dengan cara mengurangi skala dari citra sidik jari tersebut. Citra dengan ukuran tersebut dimaksudkan agar dalam pembelajaran jaringan syaraf *Hopfield*, bobot yang akan digunakan tidak terlalu besar.

A. Perancangan *Resizing*

Hasil pembacaan sensor tersebut di atas mempunyai ukuran yang berbeda, sehingga hasil pemotongan citra sidik jarinya menjadi 250x250 *pixel*. Pada proses pengubahan ukuran suatu citra sebaiknya menggunakan ukuran yang mempunyai perbandingan yang sama. Karena proses pengubahan ukuran ini juga disebut dengan proses penskalaan, jadi bila ukuran citra 100x100 maka jika dilakukan pengubahan ukuran dengan nilai skala $1/2$ maka hasil penskalaannya adalah citra dengan ukuran 50x50 *pixel*.

Berdasarkan ukuran citra hasil pemotongan citra adalah 250x250 *pixel* dan target ukuranya adalah 50x50 *pixel* sehingga dapat dilakukan perhitungan skala pengubahan citra sebagai berikut :

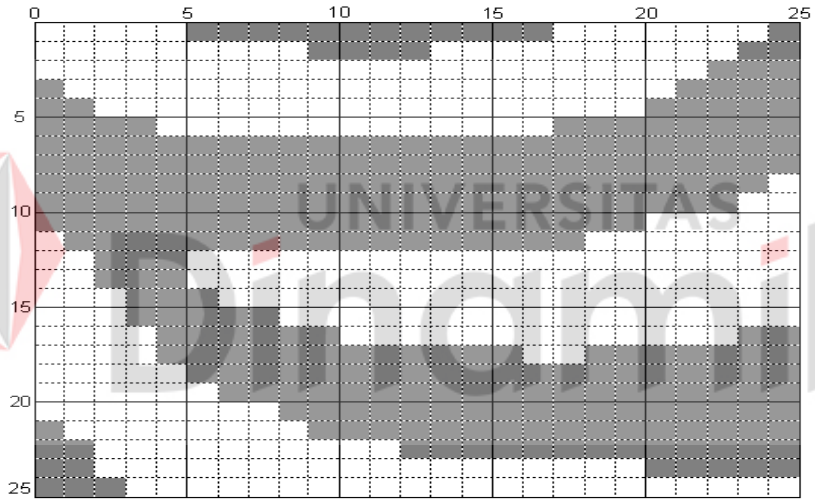
Skala = ukuran target / tinggi = 50 / 250 = 10 / 50 =1/5

Jadi skala pengubahan ukuran sidik jarinya adalah 1/5. Jadi perbandingannya adalah 1 dibanding 5, sehingga dari 5x5 *pixel* akan dirubah ukurannya menjadi 1 *pixel*.

Setelah diketahui nilai skalanya adalah 1/5, selanjutnya akan diperiksa tiap-tiap *pixel* pada citra yang akan dilakukan proses *resizing*. Citra akan diambil tiap 5x5 *pixel*-nya untuk dihitung nilai *pixel* yang bernilai 1. Jika nilai *pixel* 1 jumlahnya lebih besar atau sama dengan 50 persen maka saat diskalakan akan bernilai 1, sedangkan jika nilai *pixel* 1 jumlahnya lebih kecil dari 50 persen maka

saat diskalakan akan menjadi nilainya. Berikut contoh citra yang akan dilakukan proses *resizing* :

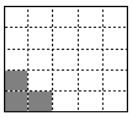
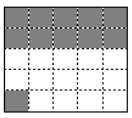
Gambar 3.13 Contoh Bagian Guratan Yang Akan Diproses *Resizing*

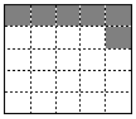
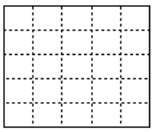
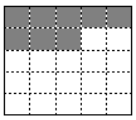
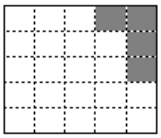
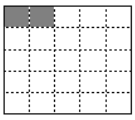
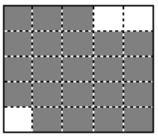
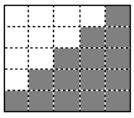
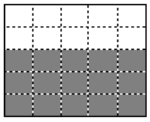
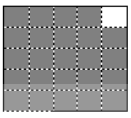
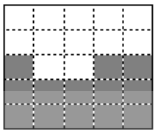
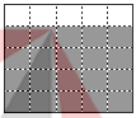
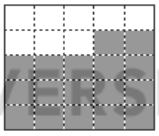
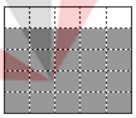
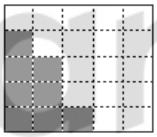
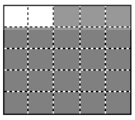
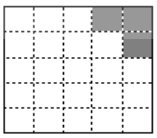
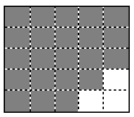
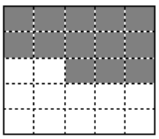
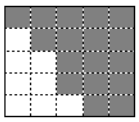
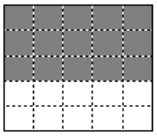
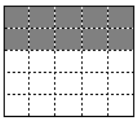
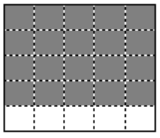
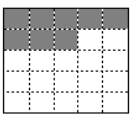


Gambar 3.14 Contoh Guratan Untuk Proses *Resizing*

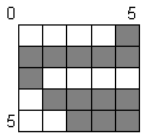
Dari citra tersebut di atas dapat dijelaskan tahap-tahap proses *resizing* sebagai berikut :

Tabel 3.1 Proses *Resizing*

No.	Guratan	Hasil Resize	No.	Guratan	Hasil Resize
1		Jumlah 1 ada 3 maka nilainya 0 Hasilnya : 	14		Jumlah 1 ada 11 maka nilainya 0 Hasilnya : 

2		Jumlah 1 ada 6 maka nilainya 0 Hasilnya :	15		Jumlah 1 ada 0 maka nilainya 0 Hasilnya :
3		Jumlah 1 ada 8 maka nilainya 0 Hasilnya :	16		Jumlah 1 ada 4 maka nilainya 0 Hasilnya :
4		Jumlah 1 ada 2 maka nilainya 0 Hasilnya :	17		Jumlah 1 ada 22 maka nilainya 1 Hasilnya :
5		Jumlah 1 ada 15 maka nilainya 1 Hasilnya :	18		Jumlah 1 ada 15 maka nilainya 1 Hasilnya :
6		Jumlah 1 ada 24 maka nilainya 1 Hasilnya :	19		Jumlah 1 ada 13 maka nilainya 1 Hasilnya :
7		Jumlah 1 ada 20 maka nilainya 1 Hasilnya :	20		Jumlah 1 ada 17 maka nilainya 1 Hasilnya :
8		Jumlah 1 ada 20 maka nilainya 1 Hasilnya :	21		Jumlah 1 ada 8 maka nilainya 0 Hasilnya :
9		Jumlah 1 ada 23 maka nilainya 1 Hasilnya :	22		Jumlah 1 ada 3 maka nilainya 0 Hasilnya :
10		Jumlah 1 ada 22 maka nilainya 1 Hasilnya :	23		Jumlah 1 ada 13 maka nilainya 1 Hasilnya :
11		Jumlah 1 ada 17 maka nilainya 1 Hasilnya :	24		Jumlah 1 ada 18 maka nilainya 1 Hasilnya :
12		Jumlah 1 ada 10 maka nilainya 0 Hasilnya :	25		Jumlah 1 ada 20 maka nilainya 1 Hasilnya :
13		Jumlah 1 ada 8 maka nilainya 0 Hasilnya :			

Setelah semua tahap proses *resizing* telah dilakukan, maka citra guratan sidik jari yang didapat adalah sebagai berikut :



Gambar 3.15 Citra Hasil Proses *Resizing*

B. Listing dan Analisa Resizing

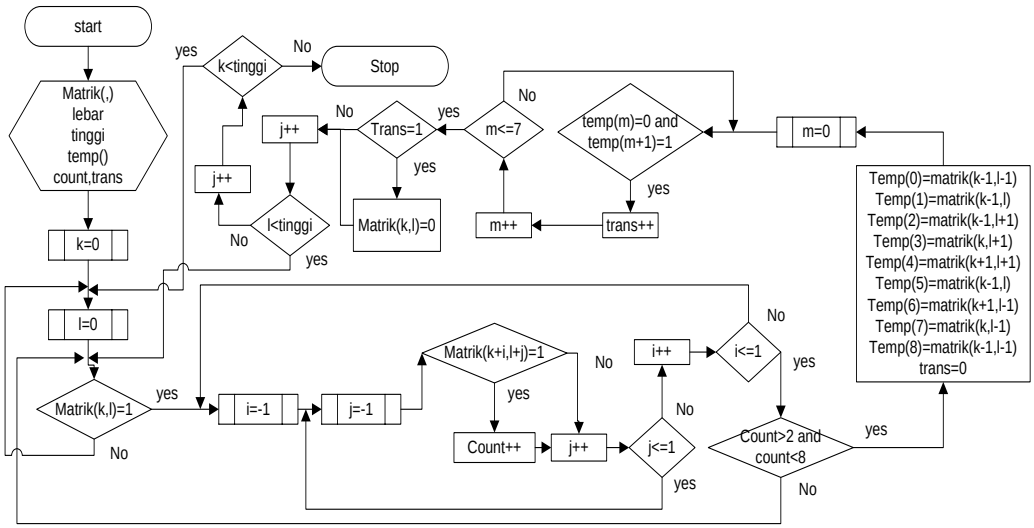
```
Dim bmp As Bitmap
Me.sample.Import.FromPicture(e.pSample.Picture)
Me.sample.Effects.Resize(IMAGE_WIDTH, IMAGE_HEIGHT,Atalasoft.ImgX6Interop._
    ImgX_ResizeMethods.ixrmGaussianFilter)
```

Dari potongan program tersebut di atas dijelaskan bahwa untuk melakukan perubahan ukuran (*Resizing*) digunakan *Data Link Library* (*dll*) dari *Atalasoft*. Dalam penggunaannya sangat sederhana hanya membutuhkan dua parameter, yaitu lebar dan tinggi citra yang akan dihasilkan. Di dalam *Data Link Library* untuk proses *resizing* ini terdapat proses perhitungan skala yang akan digunakan dalam proses. Skala didapat dari menghitung parameter lebar dan tinggi yang dimasukkan. Setelah skala didapat nilainya maka selanjutnya citra yang akan diproses akan dihitung dengan cara memeriksa tiap-tiap *pixel*-nya. Sehingga akhirnya didapat citra dengan ukuran 50x50 *pixel*.

3.2.4 Penipisan Pola

Dari citra biner ini bentuk pola sidik jari harus dilakukan proses penipisan pola. Penipisan ini dilakukan untuk menyeragamkan ukuran guratan-guratan sidik jari menjadi 1 bit saja. Sehingga tiap-tiap sidik jari mempunyai ukuran guratan-guratan sidik jari yang sama.

A. Perancangan Penipisan Pola

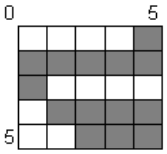


Gambar 3.16 Diagram Alur Penipisan Pola

Pada diagram alur di atas dapat dijelaskan bahwa untuk menipiskan citra dibutuhkan jendela 3x3 yang berfungsi sebagai penampung nilai yang akan diperiksa. Penipisan ini dilakukan pada *pixel* yang bernilai 1, banyaknya nilai 1 matrik dan perubahan dari $0 \rightarrow 1$ akan mempengaruhi dalam mengambil keputusan dalam penipisan.

Pertama dihitung *pixel* yang bernilai 1 pada jendela 3x3, kemudian hasil perhitungan ini akan diperiksa jumlahnya. Jika jumlah *pixel* yang bernilai 1 lebih besar dari 2 dan lebih kecil dari 8 maka nilai *pixel* yang terdapat pada jendela tersebut akan disimpan untuk diperiksa. Pemeriksaan ini dilakukan dengan cara menghitung jumlah peralihan nilai *pixel* dari 0 ke 1. Jika jumlah peralihan adalah sama dengan 1 maka nilai *pixel* yang ada pada tengah akan dihapus, sedangkan jika lebih dari 1 tidak dilakukan penghapusan.

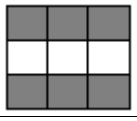
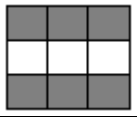
Gambar 3.17 Bagian Guratan Yang Akan Dilakukan Penipisan Pola

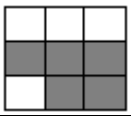
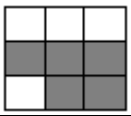
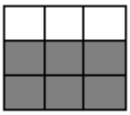


Gambar 3.18 Contoh Guratan Sidik Jari Yang Diproses Penipisan Pola

Dari gambar tersebut di atas dapat dijelaskan proses penipisan pola sebagai berikut :

Tabel 3.2 Hasil Analisa Tiap Jendela 3x3 Pada Penipisan Pola

Pergeseran Pixel	Guratan	Penipisan	Keterangan
Baris 1-3 Kolom 1-3			Tidak Berubah Ada 2 peralihan dari 0 ke 1
Baris 2-4 Kolom 1-3			Tidak Berubah Ada 2 peralihan dari 0 ke 1
Baris 3-5 Kolom 1-3			Tidak Berubah Ada 2 peralihan dari 0 ke 1
Baris 1-3 Kolom 2-4			Titik Tengah = 0
Baris 2-4 Kolom 2-4			Titik Tengah = 0
Baris 3-5 Kolom 2-4			Titik Tengah = 0

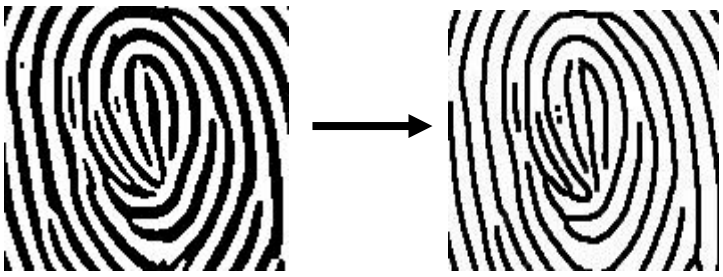
Baris 1-3 Kolom 3-5			Tidak Berubah Ada 2 peralihan dari 0 ke 1
Baris 2-4 Kolom 3-5			Tidak Berubah Ada 2 peralihan dari 0 ke 1
Baris 3-5 Kolom 3-5			Ada 1 Peralihan dari 0 ke 1 Angka 1 ada 6

B. Listing dan Analisa Penipisan Pola

```
Dim k, l, i, j, m, count As Integer
Dim tam(9), tam1(9) As Integer
Dim trans As Integer = 0
For k = 1 To IMAGE_WIDTH - 1
    For l = 1 To IMAGE_HEIGHT - 1
        If matrik(k, l) = 1 Then
             count = 0
            For i = -1 To 1
                For j = -1 To 1
                    If matrik(k + i, j + l) = 1 Then
                         count += 1
                    End If
                Next
            Next
            If count > 2 And count < 8 Then
                tam(0) = matrik(k - 1, l) 'matrik(k - 1, l - 1)
                tam(1) = matrik(k - 1, l + 1) 'matrik(k - 1, l)
                tam(2) = matrik(k, l + 1) 'matrik(k - 1, l + 1)
                tam(3) = matrik(k + 1, l + 1) 'matrik(k, l + 1)
                tam(4) = matrik(k + 1, l) 'matrik(k + 1, l + 1)
                tam(5) = matrik(k + 1, l - 1) 'matrik(k + 1, l)
                tam(6) = matrik(k, l - 1) 'matrik(k + 1, l - 1)
                tam(7) = matrik(k - 1, l - 1) 'matrik(k, l - 1)
                tam(8) = matrik(k - 1, l) 'matrik(k - 1, l - 1)
                trans = 0
                For m = 0 To 7
                    If tam(m) = 0 And tam(m + 1) = 1 Then
                        trans += 1
                    End If
                Next
                If trans = 1 Then
                     matrik(k, l) = 0
                  End If
            End If
        End If
    Next
Next
```

Pada potongan program tersebut dapat dijelaskan bahwa untuk memeriksa *pixel* yang akan dilakukan proses penipisan adalah dengan menggunakan dua *looping* untuk mengakses data *pixel*. *Looping* ini diawali dari koordinat *pixel* (1,1) sehingga pada baris dan kolom pertama dapat diambil *pixel*nya untuk diproses. Kemudian dihitung *pixel* yang bernilai 1 dan dicari

jumlah *pixel* yang lebih besar dari 2 dan lebih kecil dari 8 untuk disimpan. Selanjutnya dihitung perubahan dari 0 ke 1, jika hanya terjadi satu kali perubahan maka akan dilakukan penghapusan *pixel* yang dituju tersebut jika lebih dari dua tidak dilakukan penghapusan.

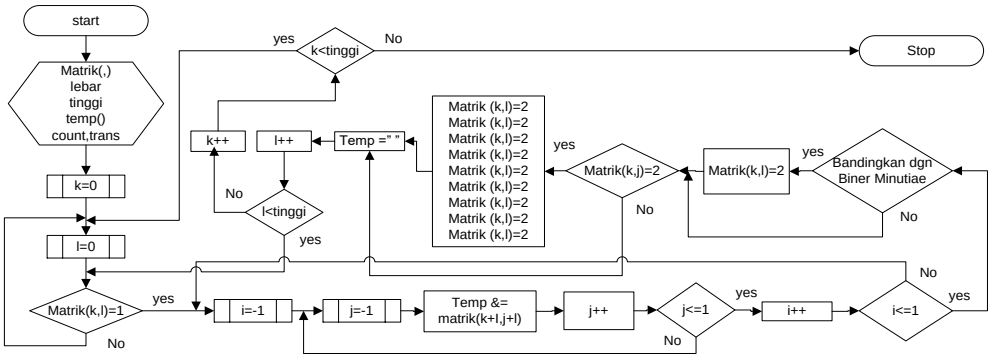


Gambar 3.19 Hasil Penipisan Pola Sidik Jari

3.2.5 Feature Extraction

Setelah pola biner sidik jari ditipiskan, selanjutnya adalah proses pengambilan ciri-ciri guratan sidik jari. Pengambilan ciri (*feature extraction*) guratan sidik jari yang akan diambil adalah guratan yang berbentuk percabangan (*ridge bifurcation*) dan guratan garis dengan titik akhir (*ridge endings*).

A. Perancangan *Feature Extraction*

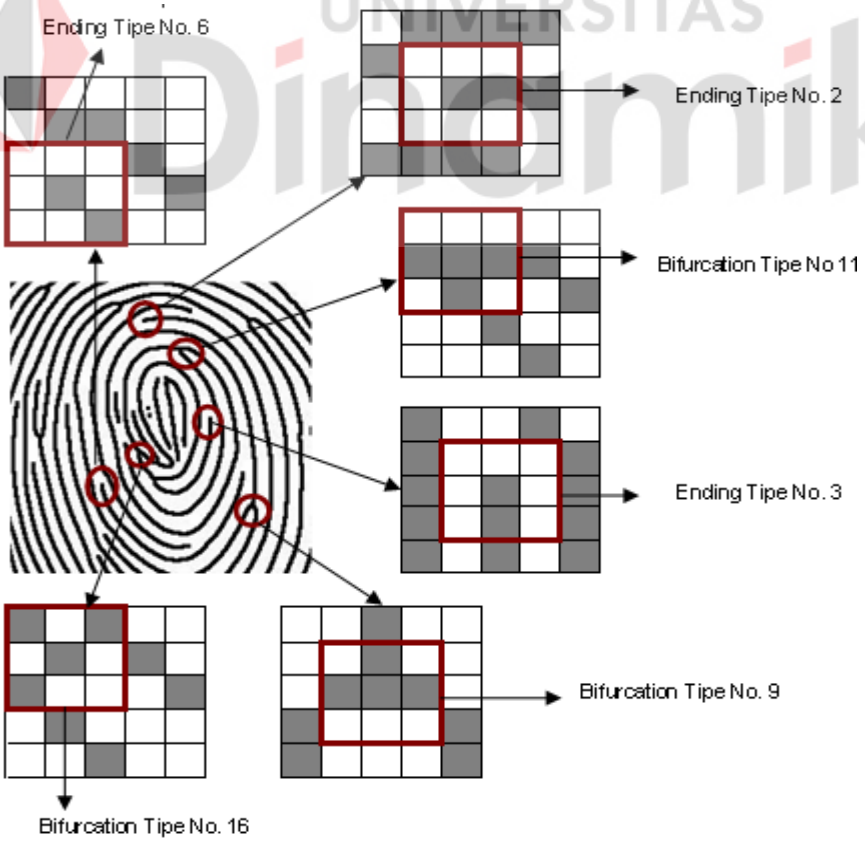


Gambar 3.20 Diagram Alur *Minutiae Extraction*

Untuk mengekstraksi guratan-guratan sidik jari juga membutuhkan jendela 3x3 untuk menampung nilai-nilai biner yang akan diperiksa. Dengan bantuan jendela 3x3 ini dapat memudahkan dalam menampung nilai biner yang akan diperiksa dengan guratan-guratan percabangan dan garis dengan titik akhir.

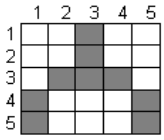
Jika pada saat pembandingan terdapat nilai yang sama dengan pembanding, maka nilai binernya akan dirubah nilainya menjadi 2, nilai 2 ini hanya sebagai kondisi bahwa terdapat guratan-guratan yang diinginkan di sini. Untuk berikutnya nilai 2 ini akan dikonversikan menjadi warna hitam pada pola sidik jari. Sehingga terbentuk titik hitam pada tiap-tiap percabangan dan garis dengan titik akhir tersebut.

Berikut contoh proses pengambilan ciri sidik jari selengkapnya :



Gambar 3.21 Contoh *Minutiae* Yang Akan Diambil

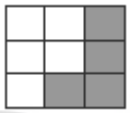
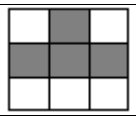
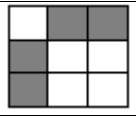
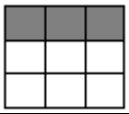
Dari gambar di atas dapat di ambil contoh sebagai berikut :



Gambar 3.22 *Pixel* Yang Akan Diperiksa

Untuk mengambil ciri tersebut di atas citra akan diteliti tiap *pixel*-nya pada matrik 3x3 dan akan disimpan pada variabel *Minutiae* dengan tipe data *string*. Berikut contoh Proses Pemeriksaan ciri-cirinya :

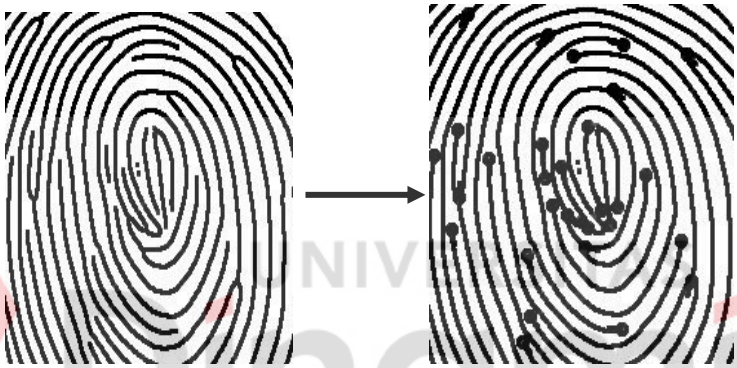
Tabel 3.3 Hasil Analisa Tiap Jendela 3x3 Pada *Minutiae Extraction*

Pergeseran Pixel	Guratan	Nilai Biner	Keterangan
Baris 1-3 Kolom 1-3		001001011	Tidak Ada Yang Cocok
Baris 2-4 Kolom 1-3		010010111	Tidak Ada Yang Cocok
Baris 3-5 Kolom 1-3		100100110	Tidak Ada Yang Cocok
Baris 1-3 Kolom 2-4		001011100	Tidak Ada Yang Cocok
Baris 2-4 Kolom 2-4		010111000	Cocok Dengan Ending Tipe No. 9
Baris 3-5 Kolom 2-4		100110001	Tidak Ada Yang Cocok
Baris 1-3 Kolom 3-5		011100100	Tidak Ada Yang Cocok
Baris 2-4 Kolom 3-5		111000000	Tidak Ada Yang Cocok
Baris 3-5 Kolom 3-5		110001001	Tidak Ada Yang Cocok

B. Listing dan Analisa *Feature Extraction*

```
Dim k, l, i, j As Integer
Dim tam(9), tam1(9) As Integer
Dim trans As Integer = 0
Dim temp As String = ""
    For k = 1 To IMAGE_WIDTH - 3
        For l = 1 To IMAGE_HEIGHT - 3
            For i = -1 To 1
                For j = -1 To 1
                    temp &= matrik(k + i, j + l)
                Next
            Next
        If matrik(k, l) = 1 Then
            If  String.Equals(temp,  "101010010")  =  True  Or  String.Equals(temp,
            "001110001")  =  True  Or  String.Equals(temp,  "010010101")  =  True  Or
            String.Equals(temp, "100011100 ") = True Then
                matrik(k, l) = 8
            ElseIf String.Equals(temp, "111010010") = True Or String.Equals(temp,
            "001111001")  =  True  Or  String.Equals(temp,  "010010111")  =  True  Or
            String.Equals(temp, "100111100 ") = True Then
                matrik(k, l) = 8
            ElseIf String.Equals(temp, "101111010") = True Or String.Equals(temp,
            "011110011")  =  True  Or  String.Equals(temp,  "010111101")  =  True  Or
            String.Equals(temp, "110011110 ") = True Then
                matrik(k, l) = 8
            ElseIf String.Equals(temp, "010011100") = True Or String.Equals(temp,
            "100011010")  =  True  Or  String.Equals(temp,  "001110010")  =  True  Or
            String.Equals(temp, "010110001") = True Then
                matrik(k, l) = 8
            ElseIf String.Equals(temp, "010111000") = True Or String.Equals(temp,
            "010011010")  =  True  Or  String.Equals(temp,  "000111010")  =  True  Or
            String.Equals(temp, "010110010") = True Then
                matrik(k, l) = 8
            ElseIf String.Equals(temp, "010101000") = True Or String.Equals(temp,
            "010001010")  =  True  Or  String.Equals(temp,  "000101010")  =  True  Or
            String.Equals(temp, "010100010") = True Then
                matrik(k, l) = 8
            ElseIf String.Equals(temp, "101010001") = True Or String.Equals(temp,
            "100010101")  =  True  Or  String.Equals(temp,  "101010100")  =  True  Or
            String.Equals(temp, "001010101") = True Then
                matrik(k, l) = 8
            ElseIf String.Equals(temp, "100010000") = True Or String.Equals(temp,
            "010010000")  =  True  Or  String.Equals(temp,  "001010000")  =  True  Or
            String.Equals(temp, "000011000") = True Then
                matrik(k, l) = 8
            ElseIf String.Equals(temp, "000010001") = True Or String.Equals(temp,
            "000010010")  =  True  Or  String.Equals(temp,  "000010100")  =  True  Or
            String.Equals(temp, "000110000") = True Then
                matrik(k, l) = 8
            End If
        End If
        If matrik(k, l) = 8 Then
            matrik(k, l) = 2
            matrik(k, l + 1) = 2
            matrik(k - 1, l - 1) = 2
            matrik(k - 1, l) = 2
            matrik(k - 1, l + 1) = 2
            matrik(k, l - 1) = 2
            matrik(k + 1, l + 1) = 2
            matrik(k + 1, l - 1) = 2
            matrik(k + 1, l) = 2
         End If
         temp = ""
        Next
    Next
End Sub
```

Sama halnya dengan penipisan, pada potongan program mengekstraksi titik guratan sidik jari dibutuhkan dua *looping* untuk mengakses *pixel* sidik jari dan menyimpan *pixel* yang diawali dari koordinat (1,1) sampai akhir dari koordinat *pixel*. Kemudian memeriksa atau membandingakan dengan biner guratan percabangan dan garis dengan titik akhir yang dipakai acuan. Jika terjadi kesamaan *pixel* yang akan diperiksa dengan *pixel* acuan maka akan diberi tanda dengan memberi nilai 1 pada sekitar *pixel* tersebut.



Gambar 3.23 Hasil *Minutiae Extraction*

3.3 Registrasi Sidik Jari

3.3.1 Perancangan Registrasi Sidik Jari

Untuk menyimpan data-data sidik jari, peneliti menggunakan fasilitas *database* dari Microsoft Access 2003. Penggunaan Microsoft Access 2003 ini sebagai tempat penyimpanan dikarenakan teknik-teknik yang digunakan dalam prosesnya tidak terlalu susah dan rumit. Dan sesuai dengan judul dan tujuan dari penelitian, sistem ini hanya mengimplementasikan jaringan syaraf *Hopfield* untuk pengenalan pola sidik jari, selain itu data yang akan disimpan dalam *database*

hanya data sidik jari beserta identitasnya. Jadi tidak terlalu membutuhkan penyimpanan data yang canggih dengan keamanan data yang baik.

Proses registrasi sidik jari ini dapat disebutkan juga dengan proses administrasi sidik jari. Dimana pada proses ini terjadi proses pendaftaran sidik jari beserta identitasnya. Namun pada proses ini tidak menutup kemungkinan terjadi proses penghapusan sidik jari ataupun merubah profil dari sidik jari tersebut. Proses ini berhubungan dengan *database* yang digunakan dalam program ini.

Segala sesuatu yang terjadi pada proses registrasi sidik jari sangat berpengaruh pada proses berikutnya, yaitu pada proses pembelajaran sidik jari yang kemudian digunakan untuk proses pengenalan sidik jari. Dengan ada perubahan pada registrasi sidik jari ini akan membutuhkan pembelajaran sidik jari ulang oleh jaringan syaraf *Hopfield*. Karena *user* atau sidik jari yang telah dirubah tidak akan dipakai lagi dalam proses perhitungan atau pembelajaran. Sehingga pembelajaran yang baru ini dapat digunakan oleh proses pengenalan sidik jari dengan benar dan tepat. Berikut segala sesuatu yang terjadi pada proses registrasi sidik jari :

A. Tabel Profil *User*

Tabel profil *user* ini tersimpan nomor ID dari *user*, nama *user*, Alamat *user*, nomor telepon *user*. Dan terdapat tiga proses yang mungkin akan terjadi, yaitu penambahan *user*, perubahan identitas *user*, dan penghapusan *user* tertentu. Ketiga proses tersebut mempengaruhi kelas *user*, yang dimana kelas *user* merupakan kelas yang berhubungan langsung dengan *database user*. Aktivitas pada tabel *user* ini juga berpengaruh pada tabel sidik jari. Karena jika ada perubahan pada tabel *user* maka tabel sidik jari juga harus disesuaikan.

***1. Add User* (Penambahan *User*)**

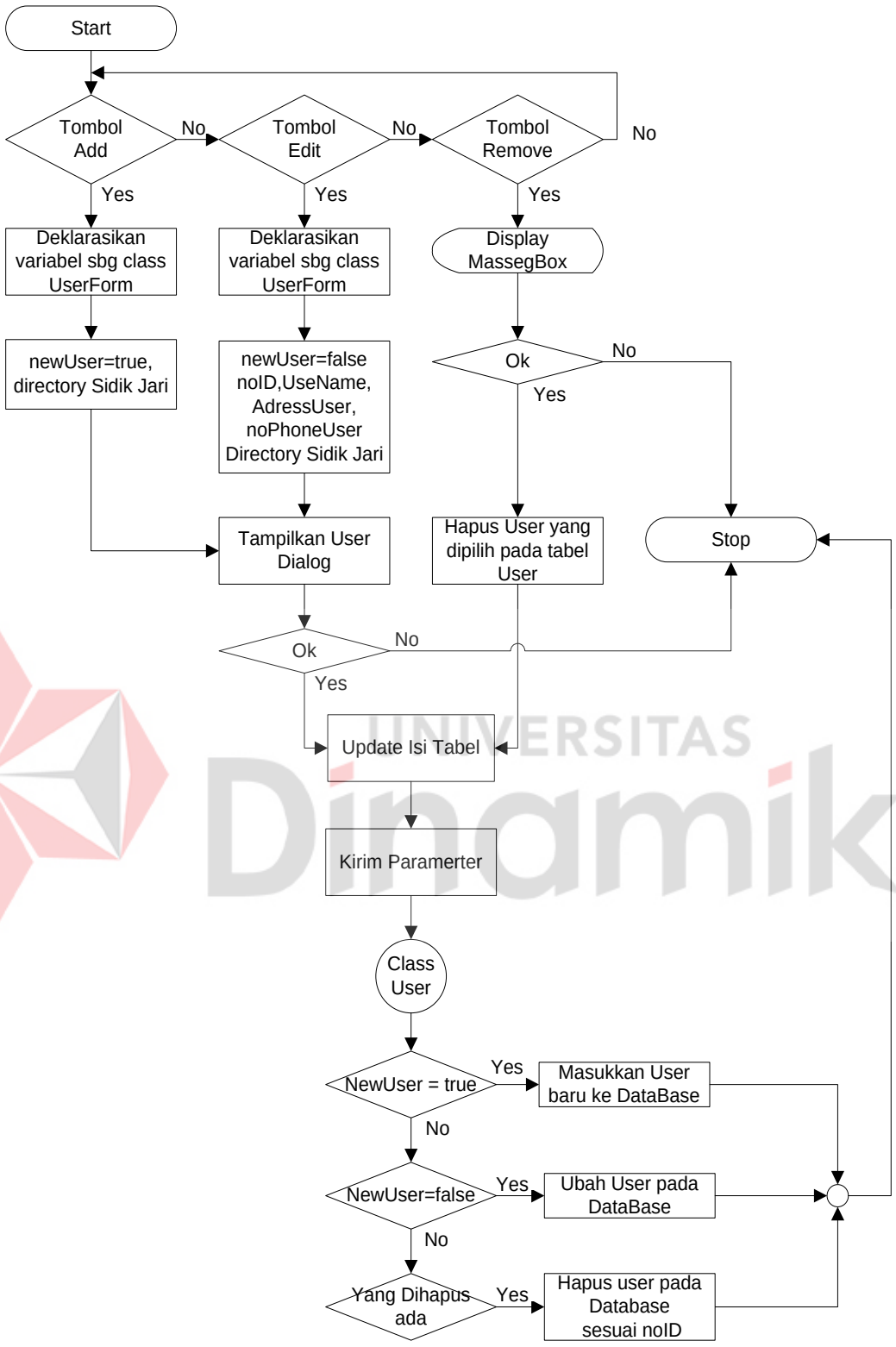
Tombol *Add User* berfungsi untuk menambah *user* (anggota) ke dalam sistem ini. Proses ini akan memberi sinyal bahwa akan dilakukan penambahan anggota dengan menampakkan suatu *box dialog* untuk mengisi data-data yang akan dimasukkan. Dalam proses ini yang akan dimasukkan ke dalam *database* adalah nomor identitas, nama anggota, alamat anggota, nomor telepon anggota, dan data pola sidik jari dalam format file disimpan dalam suatu *directory*.

***2. Edit User* (Perubahan *User*)**

Tombol *Edit User* berfungsi untuk merubah *user* (anggota) yang ada dalam sistem ini. Proses ini berawal dari tombol *Edit* ditekan dan kemudian akan dibuat variabel sebagai kelas *User*. Proses ini akan memberi sinyal bahwa akan dilakukan perubahan anggota. Dalam proses ini, yang akan menggantikan anggota yang dipilih untuk dirubah keanggotaanya dengan menampilkan *box dialog* yang berisi anggota yang akan dirubah. Jadi nomor identitas, nama anggota, alamat anggota, dan nomor telepon anggota yang ada dalam tabel *user* dan *database* akan digantikan dengan data yang baru sesuai dengan data yang dirubah. Begitu juga dengan data pola sidik jari akan diganti sesuai dengan data yang dirubah

***3. Remove User* (Penghapusan *User*)**

Tombol *Remove User* berfungsi untuk menghapus *user* (anggota) dari dalam sistem ini. Proses ini berawal dari tombol *Remove* ditekan dan kemudian akan menghapus anggota yang telah dipilih. Jadi anggota yang ada pada tabel *user* akan dihapus begitu juga data yang ada di dalam *database* akan dihapus pula.



Gambar 3.24 Diagram Alur Proses Registrasi Sidik Jari Pada Tabel *User*

B. Tabel Sidik Jari

Tabel sidik jari ini tersimpan nomor ID dari sidik jari, data sidik jari dalam biner, nomor ID *user*. Sama halnya dengan tabel *user* dalam tabel ini juga terdapat proses regristrasi (penambahan) sidik jari, perubahan sidik jari, dan penghapusan sidik jari tertentu. Ketiga proses tersebut juga mempengaruhi kelas *fingerprints*, yang dimana kelas ini merupakan kelas yang berhubungan langsung dengan *database fingerprints*. Aktivitas pada tabel sidik jari ini juga berpengaruh pada tabel *user*. Karena jika ada perubahan pada tabel sidik jari maka tabel *user* harus disesuaikan.

1. *Register* Sidik Jari (Pendaftaran Sidik Jari)

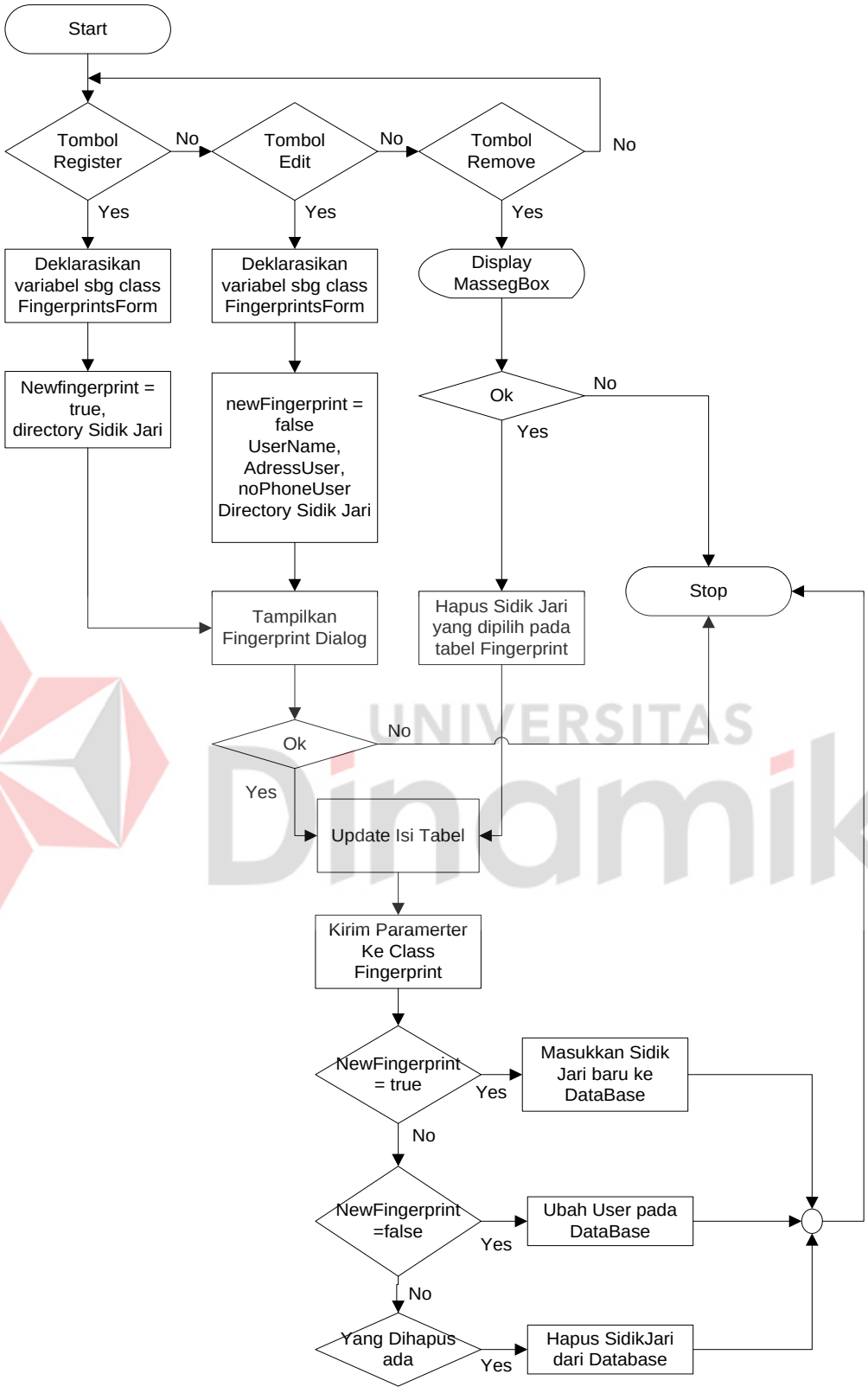
Tombol register ini berfungsi untuk mendaftarkan sidik jari berserta identitasnya yang mana telah diisi pada tabel *user*. Proses ini berhubungan dengan proses penambahan dan perubahan pada tabel *user*. Jadi setelah ada penambahan atau perubahan anggota maka sidik jari harus didaftarkan.

2. *Edit* Sidik Jari (Perubahan Sidik Jari)

Tombol e*dit* ini berfungsi hanya untuk merubah data pemilik dari sidik jari yang ada dalam *database* berdasarkan data yang dipilih untuk dirubah. Sedangkan untuk merubah data sidik jari melalui tabel *user*.

3. *Remove* Sidik Jari (Penghapusan Sidik Jari)

Tombol *remove* ini berfungsi hanya untuk menghapus sidik jari yang terdaftar pada *database*. Jadi yang dihapus dalam proses ini bukan identitas dari sidik jari ini.



Gambar 3.25 Diagaram Alur Proses Registrasi Sidik Jari Pada Tabel *Fingerprint*

3.3.2 Listing dan Analisa Registrasi Sidik Jari

A. Pembuatan Tabel *User*

Pada tabel *user* terdapat tiga proses sesuai dengan perancangan diatas, yaitu :

1. Add User

```
Dim f As FingerprintForm

f = New FingerprintForm(True, -1, Me.GetSamplePattern, "",Me.photoDirectory)
If (f.ShowForm()) Then
    Me.tblFingerprints.Fill(Me.dstFingerprintRecognitionSystem.Fingerprints)
    Me.picFingerprintsMenuSample.Image = Nothing
    Me.btnRegisterFingerprint.Enabled = False
    Me.cmdRegisterFingerprint.IsEnabled = False
    Me.isValidWeights = False
End if
```

2. Edit User

```
Dim row As Janus.Windows.GridEX.GridEXRow
Dim f As FingerprintForm
row = Me.grxFingerprints.SelectedItems(0).GetRow()
f = New FingerprintForm(False,
Convert.ToInt32(row.Cells("clmFingerprintID").Value), _
row.Cells("clmPattern").Value.ToString(), row.Cells("clmUserID").Value.ToString(),
_Me.photoDirectory)
f.ShowDialog()
Me.tblFingerprints.Fill(Me.dstFingerprintRecognitionSystem.Fingerprints)
```

3. Remove User

```
Dim count As Integer
count = Me.grxFingerprints.SelectedItems.Count
If (MessageBox.Show(Me, String.Format("Are you sure want to delete this {0} " + _
   "item(s)?", count), Me.Text, MessageBoxButtons.YesNo,
   MessageBoxIcon.Exclamation) = _ Windows.Forms.DialogResult.Yes) Then
   Dim tbl As FingerprintRecognitionSystemDataSetTableAdapters.
   _FingerprintsTableAdapter
   tbl = New FingerprintRecognitionSystemDataSetTableAdapters.
   _FingerprintsTableAdapter()
   For i As Integer = 0 To count - 1

        tbl.DeleteByFingerprintID(CType(Me.grxFingerprints.SelectedItems(i).GetRow(
        )
        _.Cells("clmFingerprintID").Value, Integer))
   Next
   Me.tblFingerprints.Fill(Me.dstFingerprintRecognitionSystem.Fingerprints)
   Me.isValidWeights = False
End If
```

Dari ketiga potongan program tersebut diatas dapat dijelaskan bahwa setiap terjadi penambahan *user* (*Add User*), perubahan *user* (*Edit User*), atau penghapusan *user* (*Remove User*) akan dilakukan inisialisasi kelas *user* baru,

dimana kelas *user* ini yang berhubungan langsung dengan *database*. Dalam kelas *user* ini akan difilter apakah penambahan, perubahan, atau penghapusan *user*.

4. Program Kelas User

```
Dim tbl As FingerprintRecognitionSystemDataSetTableAdapters.UsersTableAdapter
tbl = New FingerprintRecognitionSystemDataSetTableAdapters.UsersTableAdapter()
Try
        If (Me.newUser) Then
         tbl.Insert(Me.txtUserID.Text, Me.txtUserName.Text, Me.txtAddress.Text, _
         Me.txtPhone.Text)
        Else
         tbl.Update(Me.txtUserID.Text, Me.txtUserName.Text, Me.txtAddress.Text,
         Me.txtPhone.Text, Me.oldUserID, Me.oldUserName, Me.oldAddress, _
             Me.oldPhone)
         End If
         If (File.Exists(Me.photoDirectory + "\" + Me.oldUserID)) Then
             File.Delete(Me.photoDirectory + "\" + Me.oldUserID)
         End If
         Me.picPhoto.Image.Save(Me.photoDirectory + "\" + Me.txtUserID.Text)
         Me.Close()
Catch ex As Exception
         MessageBox.Show(Me, ex.Message, Me.Text, MessageBoxButtons.OK, _
         MessageBoxIcon.Error)
End Try
```

Potongan program inti dari kelas *user* tersebut diatas menjelaskan bahwa sinyal yang diberikan oleh program kelas utama akan difilter berdasarkan tanda yang diberikan. Jika *newUser = true* maka ini merupakan penambahan *user* baru, sedangkan jika *newUser = false* maka ini merupakan perubahan *user*. Untuk penghapusan *user* dilakukan jika nomor ID yang ditunjuk benar-benar ada dalam *database*, jika nomor IDnya ada maka data dalam *database* akan dihapus sesuai dengan nomor ID tersebut.

B. Pembuatan Tabel Sidik Jari

1. *Register* Sidik Jari

```
Dim f As FingerprintForm
f = New FingerprintForm(True, -1, Me.GetSamplePattern, "", Me.photoDirectory)
    If (f.ShowForm()) Then
        Me.tblFingerprints.Fill(Me.dstFingerprintRecognitionSystem.Fingerprints)
        Me.picFingerprintsMenuSample.Image = Nothing
        Me.btnRegisterFingerprint.Enabled = False
        Me.cmdRegisterFingerprint.IsEnabled = False
        Me.isValidWeights = False
    End If
```

2. *Edit* Pemilik Sidik Jari

```
Dim row As Janus.Windows.GridEX.GridEXRow
Dim f As FingerprintForm
row = Me.grxFingerprints.SelectedItems(0).GetRow()
f = New FingerprintForm(False,
Convert.ToInt32(row.Cells("clmFingerprintID").Value), _
     row.Cells("clmPattern").Value.ToString(),row.Cells("clmUserID")._
     Value.ToString(),  _ Me.photoDirectory)
f.ShowDialog()
Me.tblFingerprints.Fill(Me.dstFingerprintRecognitionSystem.Fingerprints)
```

3. *Remove* Sidik Jari

```
Dim count As Integer
count = Me.grxFingerprints.SelectedItems.Count
If (MessageBox.Show(Me, String.Format("Are you sure want to delete this {0} " + _
 "item(s)?", count), Me.Text, MessageBoxButtons.YesNo, MessageBoxIcon.Exclamation)
= _
 Windows.Forms.DialogResult.Yes) Then
 Dim tbl As
FingerprintRecognitionSystemDataSetTableAdapters.FingerprintsTableAdapter
tbl = New
FingerprintRecognitionSystemDataSetTableAdapters.FingerprintsTableAdapter()
    For i As Integer = 0 To count - 1
        tbl.DeleteByFingerprintID(CType(Me.grxFingerprints.SelectedItems(i).GetRow(
) _
       .Cells("clmFingerprintID").Value, Integer))
       Next
       Me.tblFingerprints.Fill(Me.dstFingerprintRecognitionSystem.Fingerprints)
       Me.isValidWeights = False
End If
```

Ketiga potongan program tersebut diatas dapat dijelaskan bahwa setiap terjadi penambahan *user* (*Register fingerprint*), perubahan *user* (*Edit Owner fingerprint*), atau penghapusan *user* (*Remove fingerprint*) akan dilakukan inisialisasi kelas *fingerprint* baru, dimana kelas *fingerprint* ini yang berhubungan langsung dengan *database*. Dalam kelas *user* ini akan difilter apakah penambahan, perubahan, atau penghapusan *fingerprint.*

4. Potongan Program Inti Kelas *Fingerprint*

```
Dim tbl As
FingerprintRecognitionSystemDataSetTableAdapters.FingerprintsTableAdapter
tbl = NewFingerprintRecognitionSystemDataSetTableAdapters._
FingerprintsTableAdapter()
Try
    If (Me.newFingerprint) Then
        tbl.Insert(Me.pattern, Me.cboUserID.Text)
    Else
        tbl.Update(Me.pattern, Me.cboUserID.Text, Me.fingerprintID, Me.oldUserID)
    End If
    Me.isOK = True
    Me.Close()
```

```
Catch ex As Exception
MessageBox.Show(Me, ex.Message, Me.Text, MessageBoxButtons.OK,
MessageBoxIcon.Error)
End Try
```

Potongan program inti dari kelas *fingerprint* tersebut diatas menjelaskan bahwa sinyal yang diberikan oleh program kelas utama akan difilter berdasarkan tanda yang diberikan. Jika *newFingerprint = true* maka ini merupakan penambahan *fingerprint* baru, sedangkan jika *newFingerprint = false* maka ini merupakan perubahan pemilik dari *fingerprint* tersebut. Untuk penghapusan *fingerprint* sesuai dengan nomor ID *fingerprint* tersebut.

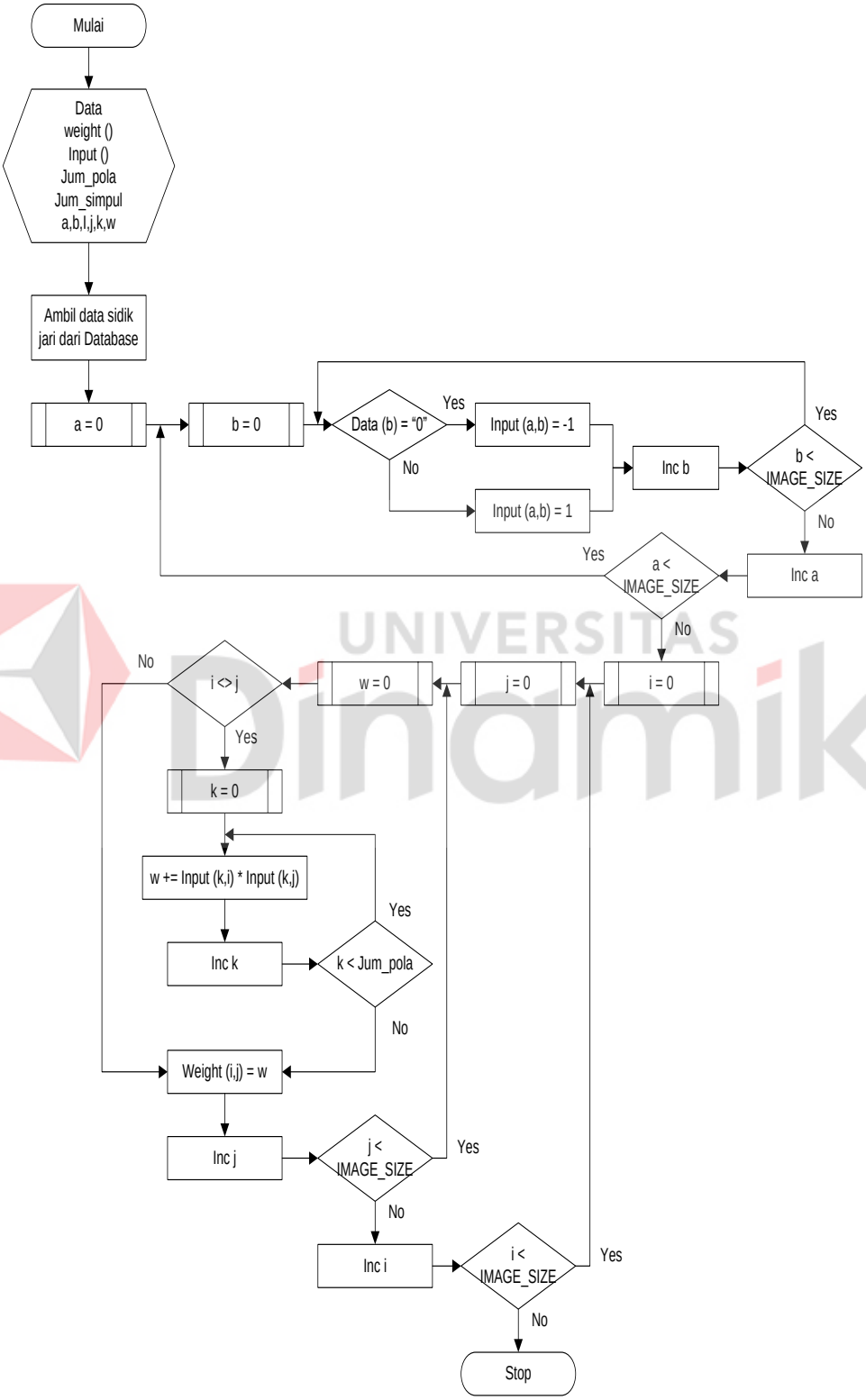
3.4 Jaringan Syaraf *Hopfield*

Metode pembelajaran jaringan syaraf tiruan yang digunakan pada penelitian ini adalah jaringan syaraf tiruan dengan metode *Hopfield* atau jaringan syaraf *Hopfield*. Jaringan syaraf *Hopfield* ini dapat digunakan dengan mudah dalam menyelesaikan permasalahan pengenalan pola. Dalam penelitian ini digunakan citra dengan ukuran 50x50 dan 70x70 *pixel*. Dengan menggunakan ukuran tersebut di atas, bila citra dengan ukuran 50x50 *pixel* maka bobot koneksi yang diperoleh adalah matrik 2500x2500, sedangkan untuk ukuran 70x70 *pixel* maka bobot koneksi yang diperoleh adalah matrik 4900x4900.

Pada jaringan syaraf *Hopfield* semua *neuron* terhubung satu sama lainnya, ini dapat dilihat dari *output* dari suatu *neuron* merupakan *input* bagi *neuron-neuron* yang lainnya. Dan antara *neuron* yang satu dengan yang lainnya dihubungkan oleh bobot koneksi yang tersimpan dari tiap-tiap *neuron*.

3.4.1 Perancangan jaringan syaraf *Hopfield*

A. Pembelajaran Sidik Jari



Gambar 3.26 Diagram Alur Perhitungan Bobot Koneksi

Proses pembelajaran merupakan tahapan yang harus dilakukan agar jaringan syaraf tiruan dapat mengenali pola-pola yang hendak diidentifikasikan. Proses ini diawali dengan pengambilan data-data dari *database* untuk dilakukan perhitungan matrik bobot koneksi dari jaringan syaraf *Hopfield*. Hasil perhitungan bobot koneksi ini tidak disimpan pada *database*, karena matrik dari bobot koneksi ini terlalu besar dan membutuhkan waktu yang lama untuk memasukkan ke dalam *database*. Jadi perhitungan bobot koneksi ini dilakukan pada saat akan dilakukan pengenalan pola sidik jari.

Pada gambar diagram alur tersebut diatas bahwa untuk mencari bobot koneksi, pola-pola sidik jari yang telah disimpan dalam *database* diambil untuk disimpan di *array* penampung. *Array* ini akan dipelajari oleh jaringan syaraf *Hopfield* untuk dicari bobot koneksinya. Antara pola sidik jari satu dengan yang lainnya yang tertampung dalam array ini akan dihitung dengan algoritma jaringan syaraf *hopfield*. Dalam penelitian ini peneliti menggunakan angka bipolar untuk *input* vektornya, dengan kata lain *input* vektor hanya bernilai 1 atau -1. Sehingga pola sidik jari yang nilai binernya nol akan dirubah menjadi -1 sedangakan nilai vektor 1 tidak dirubah. Untuk menghitung bobot matriknya diperoleh dengan cara mengalikan vektor dengan dirinya sendiri pada indek satu dengan indek berikutnya dari *pixel* pola tersebut dan dijumlahkan dengan perkalian indek pada vektor pola sidik jari yang lainnya. Namun pada perkalian pada satu indek di dalam satu vektor pola tidak diijinkan, dan bobot pada indek tersebut diisi dengan nilai nol. Berikut cara perhitungannya :

Misalkan : n = indek baris dan m = indek kolom (Untuk Bobot)

k = indek baris dari data pola sidik jari

Bobot (1,1) = 0

Bobot (1,2) = *input* (pola ke-1, 1) * *input* (pola ke-1, 2) + *input* (pola ke-2, 1) * *input* (pola ke-2, 2) +…+ *input* (pola ke-k, m-1) * *input* (pola ke-k, m)

Bobot (1,3) = *input* (pola ke-1, 1) * *input* (pola ke-1, 3) + *input* (pola ke-2, 1) * *input* (pola ke-2, 3) +…+ *input* (pola ke-k, m-1) * *input* (pola ke-k, m)

Dan perhitungan bobot berkutnya sama, sampai :

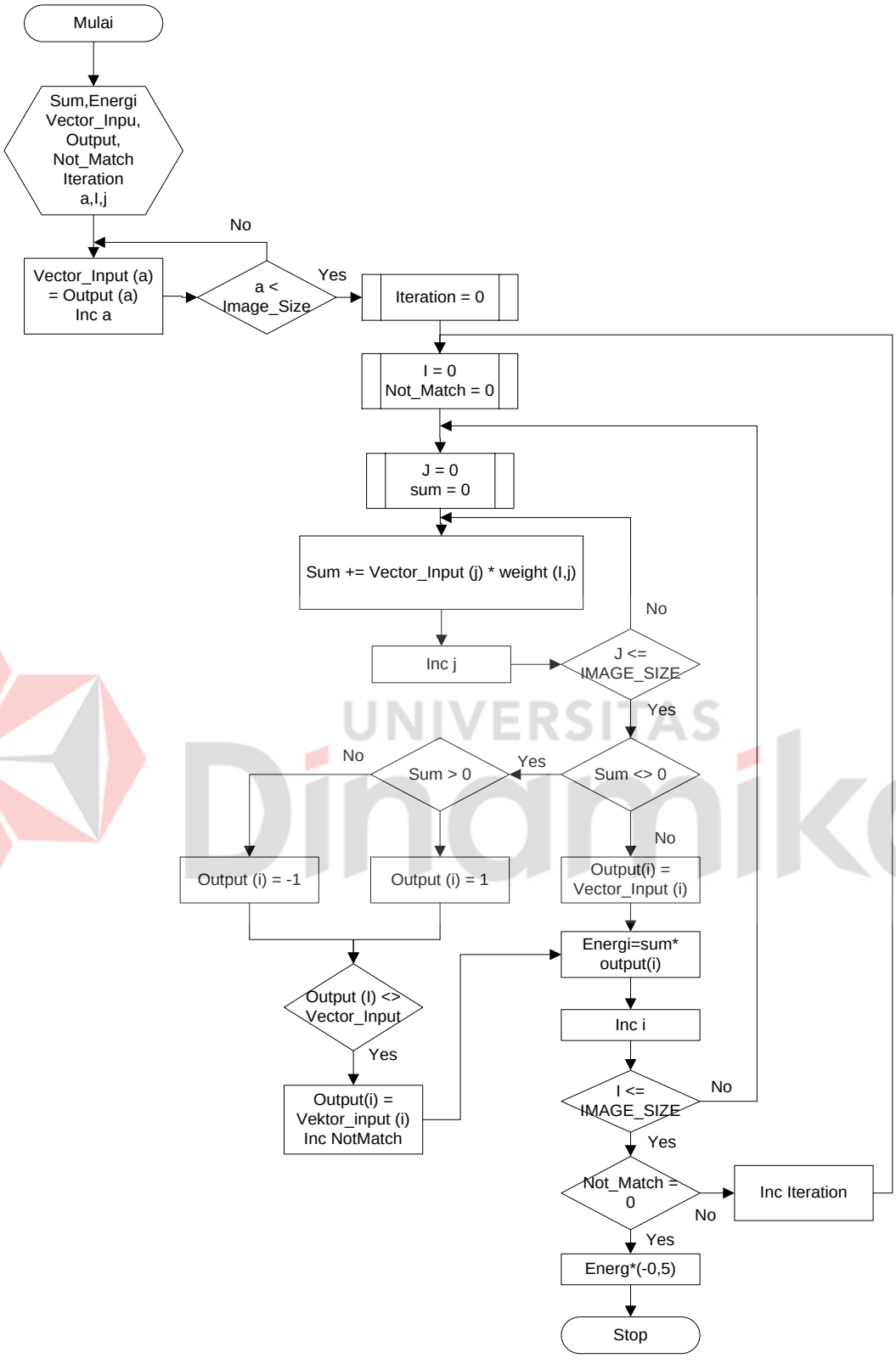
Bobot (n-1, m) = *input* (pola ke-1, n-1) * *input* (pola ke-1, m) + *input* (pola ke-2, n-1) * *input* (pola ke-2, m) +...+ *input* (pola ke-k, n-1) * *input* (pola ke-k, m)

Bobot (n, m) = 0

B. Pengenalan jaringan syaraf *Hopfield*

Setelah data-data sidik jari telah terkumpul dan disimpan di *database*, dan juga tahap pembelajaran jaringan syaraf *Hopfield* telah dilakukan. Tahap berikutnya adalah tahap pengenalan jaringan syaraf *Hopfield* terhadap data sidik jari yang dimasukkan. Tahap ini berawal dari proses akuisisi sidik jari melalui sensor, kemudian citra digital sidik jari dalam format biner ini akan disimpan dalam suatu vektor atau matrik satu dimensi.

Namun sebelumnya data biner dari vektor ini terlebih dahulu akan diproses dalam menentukan nilai akitivasi yang akan digunakan nantinya dalam jaringan syaraf. Untuk itu nilai biner dari input akan dikonversikan dulu ke nilai biner *sigmoid* atau merubah angka biner yang bernilai 0 dengan nilai angka -1, sedangkan untuk nilai biner 1 tidak akan dirubah nilainya.



Gambar 3.27 Diagram Alur Pengenalan JST

Sesuai dengan diagram alur tersebut diatas dapat dijelaskan dalam langkah-langkah sebagai berikut :

Langkah 1 : Deklarasikan variabel *input* vektor yang berfungsi sebagai penampung data-data masukan untuk *neuron-neuron* pada jaringan syaraf. Dan variabel *output* sebagai keluaran dari jaringan syaraf atau dapat disebut dengan aktivasi.

Langkah 2 : Pertama *output* atau aktivasi diberi nilai sesuai dengan data biner sidik jari yang akan diidentifikasi. Kemudian *input* vektor juga diberi nilai yang sama dengan *output*. Karena data *input* dari suatu *neuron* adalah *output* dari *neuron* yang lainnya.

Langkah 3 : Pilih salah satu *neuron* untuk dihitung nilai aktivasinya. Dalam penelitian ini pemilihan *neuron* dilakukan dengan cara berurutan dari indek yang paling kecil sampai yang paling besar.

Langkah 4 : Kalikan vektor *input* dengan bobot koneksi sesuai dengan *neuron* yang dipilih. Misalkan *neuron* yang dipilih adalah *neuron* yang pertama maka vektor *input*-nya adalah *output* dari *neuron* kedua, *output neuron* ketiga dan *output neuron* berikutnya sampai *neuron* yang terakhir. Sedangkan bobot koneksinya adalah bobot pada baris pertama kolom pertama sampai kolom terakhir dari matrik bobot koneksi. Jadi perkaliannya dapat ditulis sebagai berikut :

Output neuron (ke-1) = vektor *input* (ke-1) * bobot (baris ke-1,kolom ke-1) + vektor *input* (ke-2) * bobot (baris ke-1,kolom ke-2)+…+vektor *input* (terakhir) * bobot (baris ke-1, kolom terakhir).

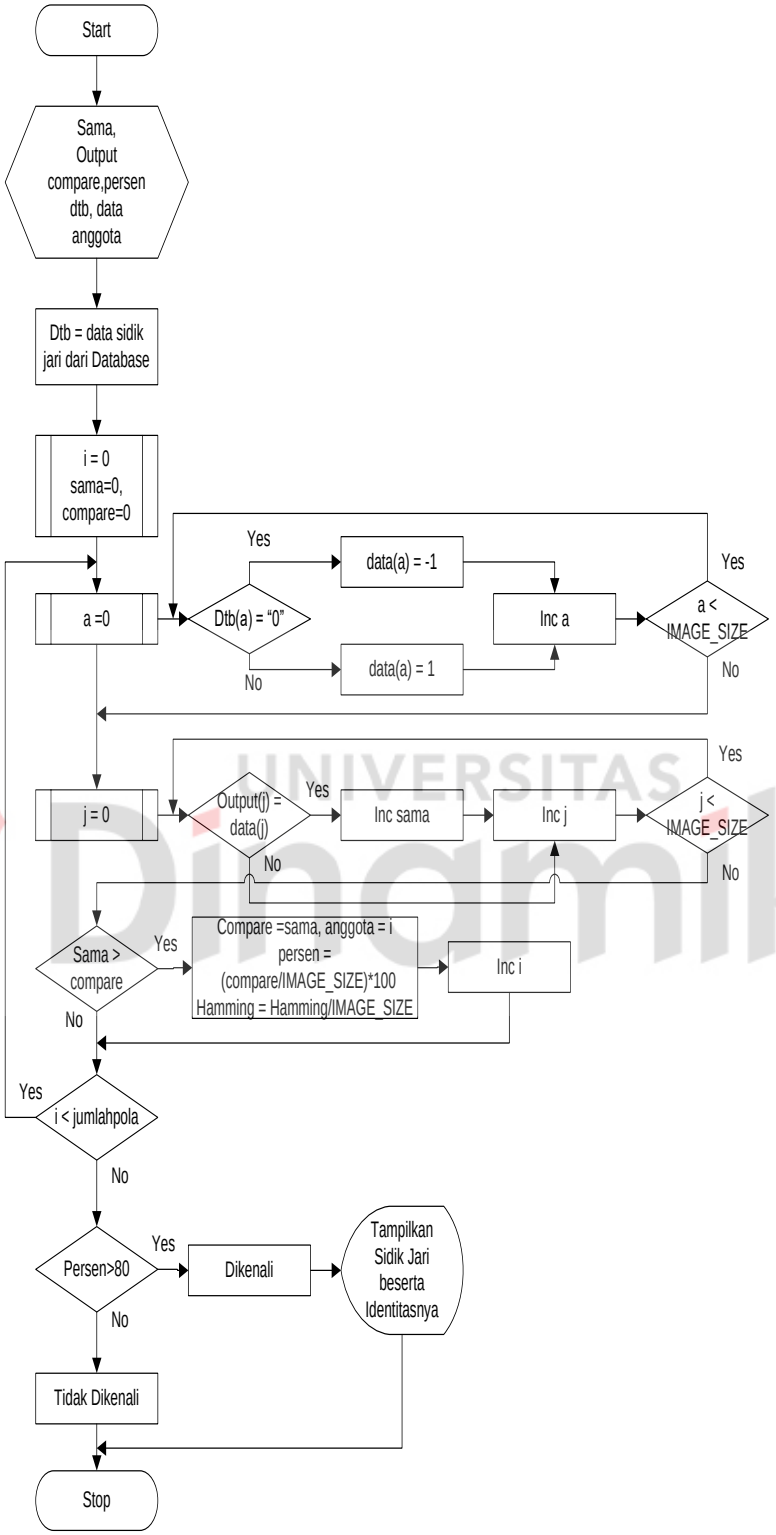
Untuk *neuron* kedua dan *neuron* berikutnya adalah sama perhitungannya.

Langkah 5 : Setelah *output* dari suatu *neuron* telah dihitung, kemudian nilai *output* ini akan dibandingkan dengan nilai *threshold*. Dalam penelitian ini apabila nilai *output* lebih besar dari nol maka aktivasi bernilai satu, dan nilai *output* yang lebih kecil dari nol nilai aktivasinya bernilai -1, sedangkan nilai *output* yang sama dengan nol maka nilai aktivasinya tidak akan berubah sesuai dengan vektor *output* sebelumnya.

Langkah 6 : Bandingkan aktivasi dengan vektor *input*, apakah nilai aktivasi konvergen dengan vektor *input*. Konvergen terjadi apabila nilai aktivasi sama dengan vektor *input*, jika nilai aktivasi tidak sama dengan vektor *output* maka nilai vektor *input* disamakan dengan nilai aktivasi. Jadi jika masih ada *neuron* yang masih belum konvergen akan dilakukan perhitungan kembali (iterasi selanjutnya) sampai kondisi konvergen tercapai. Hitung energi dengan cara menjumlahkan hasil perkalian antara *output neuron* dengan nilai aktivasi

Langkah 7 : Ulangi langkah 3 sampai konvergen

Setelah proses pengenalan oleh jaringan syaraf *Hopfield* mencapai konvergen, maka selanjutnya adalah membandingkan *output* hasil pengenalan sidik jari dengan data sidik jari (*fundamental memory*) yang ada dalam *database*. Data sidik jari yang ada dalam *database* diambil kemudian nilai biner yang bernilai nol diganti dengan -1 sedangkan yang bernilai 1 tidak ada perubahan.

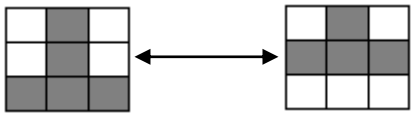


Gambar 3.28 Diagram Alur Identifikasi *Output* JST

Nilai *Hamming Distance* ini dicari dengan cara menghitung jumlah persamaan antara tiap-tiap output JST dengan tiap-tiap *pixel* yang ada dalam *database*. Pembandingan output JST ini dilakukan untuk semua data sidik jari yang ada dalam *database*. Berikut cara perhitungannya :

$$\text{Nilai } \textit{Hamming Distance} = \frac{\textit{jumalah perbedaan}}{\textit{jumlah neuron}}$$

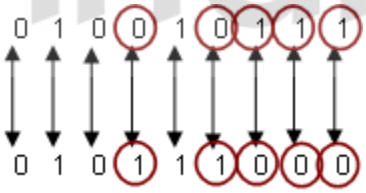
Bila dimisalkan :



CitraOutput JST Citra dari *Database*

Gambar 3.29 Contoh Citra Untuk Perhitungan *Hamming Distance*

Citra Output JST dengan nilai biner (01001011) dan citra dari *database* dengan nilai biner (010111000). Sehingga dapat dianalisa sebagai berikut :



Gambar 3.30 Pembandingan Antar Nilai Biner

Dari gambar tersebut di atas dapat dijelaskan bahwa terdapat jumlah perbedaan sebanyak 5. Sehingga dapat dihitung nilai *Hamming Distance*-nya sebagai berikut :

$$\textit{Hamming Distance (HD)} = \frac{5}{9} = 0{,}5555 \text{ atau HD} = 0{,}56$$

$$\text{Persen Kesamaan} = (1 - HD) * 100\% = (1 - 0{,}56) * 100\% = 44\%$$

Dari sekian banyak nilai *Hamming Distance* antar pola sidik jari yang dibandingkan dengan output JST , dicari nilai yang mendekati nilai 0 maka output jaringan tersebut dapat dikenali atau hampir sama dengan data yang ada pada *database*. Nilai *Hamming Distance* ini digunakan untuk mencari berapa persen nilai kesamaan dari kedua pola. Sehingga dengan Mengetahui nilai persen kesamaan tersebut dapat dijadikan acuan bahwa output jaringan tersebut teridentifikasi atau tidak. Dalam penelitian ini digunakan nilai persen kesamaan di atas 80% yang dapat dianggap dikenali. Di bawah nilai itu sidik jari yang dinputkan dianggap tidak dapat dikenali.

3.4.2 Listing dan Analisa Jaringan Syaraf *Hopfield*

A. Pembelajaran Jaringan Syaraf *Hopfield*

```
Dim dtb As FingerprintRecognitionSystemDataSet.FingerprintsDataTable

dtb = New FingerprintRecognitionSystemDataSetTableAdapters._
FingerprintsTableAdapter().GetData()
Dim input(dtb.Count, IMAGE_SIZE) As Integer

Me.stbStatus.Panels("prbStatus").ProgressBarMaxValue = dtb.Count
For i As Integer = 0 To dtb.Count - 1
    Me.stbStatus.Panels("prbStatus").ProgressBarValue = i + 1
    Me.stbStatus.Panels("lblStatus").Text = _
    String.Format("Retrieving fingerprints from database ({0}) to ({1})...", (i +
    1).ToString(), IMAGE_SIZE.ToString())
    Application.DoEvents()
    For j As Integer = 0 To IMAGE_SIZE - 1
        If (dtb(i).Pattern(j).ToString() = "0") Then
            input(i, j) = -1
        Else
            input(i, j) = 1
        End If
    Next
Next
Me.stbStatus.Panels("prbStatus").ProgressBarValue =
Me.stbStatus.Panels("prbStatus").ProgressBarMinValue
Application.DoEvents()
```

Program tersebut diatas menjelaskan cara pengambilan data sidik jari dalam *database*. Pertama didefinisikan terlebih dahulu variabel penampung data sidik jari dari *database*. Dari program diatas dtb sebagai varibelnya, dan variabel matrik *input* sebagai matrik penyimpan sidik jari. Dibutuhkan dua *looping* untuk penyimpanan data sidik jari ke dalam *array* atau matrik *input*. *Looping* pertama

digunakan untuk mengakses baris dalam *database* dan *looping* kedua untuk mengakses kolom dalam *database*. Dalam penelitian ini *input* yang nilainya nol akan dirubah menjadi -1 dan yang nilainya 1 tidak berubah. Proses ini dilakukan sampai data sidik jari telah tersimpan semua dalam matrik *input*.

```
Me.stbStatus.Panels("prbStatus").ProgressBarMaxValue = IMAGE_SIZE
For i As Integer = 0 To IMAGE_SIZE - 1
    Me.stbStatus.Panels("prbStatus").ProgressBarValue = i + 1
    Me.stbStatus.Panels("lblStatus").Text = String.Format("Calculating weights And
    Insert to Variabel ({0}) to ({1})...", (i + 1).ToString(),
    IMAGE_SIZE.ToString())
    Application.DoEvents()
    For j As Integer = 0 To IMAGE_SIZE - 1
        Dim w As Integer
        w = 0
        If (i <> j) Then
            For k As Integer = 0 To dtb.Count - 1
                w += input(k, i) * input(k, j)
            Next
        End If
        weights(i, j) = w
    Next
Next
Me.stbStatus.Panels("prbStatus").ProgressBarValue =
Me.stbStatus.Panels("prbStatus")._ ProgressBarMinValue
Me.stbStatus.Panels("lblStatus").Text = "Done"
Application.DoEvents()
Me.isValidWeights = True
```

Untuk mencari bobot koneksi dari jaringan dibutuhkan tiga *looping*. *Looping* pertama digunakan untuk menentukan baris matrik bobot yang akan diisi dengan nilai perhitungan bobot, *looping* kedua digunakan untuk menentukan kolom pada baris yang akan diisi perhitungan bobot. Sedangkan *looping* yang terakhir digunakan untuk mengakses data sidik jari. Pada awalnya nilai bobot dibuat nol, pada *looping* kedua jika baris tidak sama dengan kolom maka nilai w dihitung berdasarkan perkalian *input* sidik jari pertama pada koordinat tertentu dikalikan dengan *input* sidik jari pertama pada koordinat berikutnya. Hasil perhitungan dijumlahkan dengan hasil perhitungan sidik jari berikutnya. Porses ini dilakukan sampai *looping* ketiga selesai atau sidik jari yang terakhir. Sedangkan untuk koordinat yang terbentuk dari baris dan kolom yang sama diberi nilai nol.

B. Pengenalan Jaringan Syaraf *Hopfield*

Pada tahap pengenalan sidik jari oleh jaringan syaraf *Hopfield* ini, pertama kali *input* dan *output* dari *neuron* pada jaringan dibuat sama nilainya. Ini merupakan nilai awal dari jaringan. Jadi nilai yang masuk pada *neuron* adalah sama dengan nilai yang keluar dari *neuron*.

```
Dim energi As Integer
Dim sum As Integer
Dim Not_Match As Integer
Dim Vector_Input(IMAGE_SIZE - 1)
For a As Integer = 0 To IMAGE_SIZE - 1
    Vector_Input(a) = output(a)
Next
For Iteration = 0 To 50
    Not_Match = 0
    For i As Integer = 0 To IMAGE_SIZE - 1
        sum = 0
        Me.stbStatus.Panels("lblStatus").Text = _String.Format("hitung
        Hopfield_Network : Iteration ({0}), Node ({1})...", Iteration.ToString(),
        i.ToString())
        Application.DoEvents()
        For j As Integer = 0 To IMAGE_SIZE - 1
            sum += Vector_Input(j) * weights(i, j)
        Next
        If sum <> 0 Then
            If sum > 0 Then
                output(i) = 1
            End If
            If sum < 0 Then
                output(i) = -1
            End If
            If (output(i) <> Vector_Input(i)) Then
                Vector_Input(i) = output(i)
                Not_Match += 1
            End If

       End If
       energi += sum * output(i)
    Next
    energi = energi * (-0.5)
    energie = energi
    If Not_Match = 0 Then
        waktu.Stop()
        Exit For
    End If
Next Iteration
```

Dalam potongan program tersebut diatas terdapat tiga *looping*. *Looping* yang pertama berfungsi untuk menghitung iterasi dalam proses perhitungan pengenalan sampai konvergen. Batas *looping* pada iterasi ini adalah 50 kali, namun *looping* ini dapat berhenti jika kondisi konvergen telah terpenuhi. Untuk *looping* yang kedua berfungsi untuk menentukan koordinat baris yang akan

dihitung dan *looping* yang ketiga berfungsi untuk menentukan koordinat kolom yang akan dihitung.

Untuk mengupdate nilai keluaran suatu *neuron* dilakukan secara berurutan dari *neuron* urutan yang paling kecil samapi *neuron* urutan paling besar. Ini dilakukan untuk mempermudah dalam perhitungan keluaran dari *neuron*. Variabel sum digunakan sebagai penampung hasil jumlah total perkalian antara *neuron* yang dicari dikalikan dengan bobot koneksinya. Kemudian hasil perhitungan ini akan dibandingkan dengan nilai *threshold*. Jika sum lebih besar dari nol maka *neuron* ini akan bernilai 1 atau aktif. Sedangkan jika sum lebih kecil dari nol maka akan bernilai sama dengan -1 atau tidak aktif. Setelah itu hasil pembandingan ini akan dibandingkan lagi dengan keluaran awal *neuron* tersebut, jika nilainya sama maka ini disebut dengan konvergen namun jika tidak sama, *neuron* ini akan dihitung lagi sampai tercapai konvergen.

```
For i As Integer = 0 To dtb.Count - 1
        Me.stbStatus.Panels("prbStatus").ProgressBarValue = (i + 1).ToString()
        Me.stbStatus.Panels("lblStatus").Text = String.Format("Recognizing pattern_
        ({0}) to ({1})...", (i + 1).ToString(), dtb.Count.ToString())
        Application.DoEvents()
        data = Me.ConvertToBipolar(dtb(i).Pattern, IMAGE_SIZE)
        sama = 0
        For j As Integer = 0 To IMAGE_SIZE - 1
                If (output(j) = data(j)) Then
                    sama += 1
                End If
        Next
        If (sama > compare) Then
                compare = sama
               Haming = Haming / IMAGE_SIZE
               persen = (sama / IMAGE_SIZE) * 100
               anggota = i
               benar = data
         End If
Next
If (Hamming >= 80) Then
      Me.kenal.Text = "Dikenali"
      dtb2=NewFingerprintRecognitionSystemDataSetTableAdapters.
      _UsersTableAdapter().GetDataByUserID(dtb(anggota).UserID)
      userID = dtb2(0).UserID
      Me.NoId.Text() = userID
      userName = dtb2(0).UserName
      address = dtb2(0).Address
      phone = dtb2(0).Phone
      gmbr = Me.ConvertToBitmap2(benar, IMAGE_HEIGHT, IMAGE_WIDTH)
     Me.sample.DotNet.ImportBitmap(gmbr)
     Me.PictureBox4.Image = Me.sample.DotNet.ExportBitmap()
     MsgBox("sama " & compare)
Else
```

```
      dtb2=NewFingerprintRecognitionSystemDataSetTableAdapters.UsersTableAdapter()
      .GetDataByUserID(dtb(anggota).UserID)
      userID = dtb2(0).UserID
      Me.NoId.Text() = userID
      Me.kenal.Text = "Tidak Dikenali"
End If
```

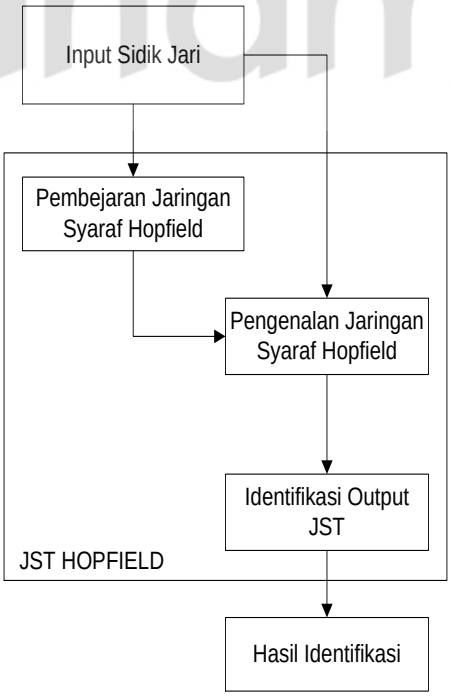
Setelah keadaan konvergen telah tercapai maka proses selanjutnya adalah membandingkan hasil perhitungan oleh jaringan syaraf *Hopfield* dengan data *fundamental memory* (*stable state*) yang ada dalam *database*. Hasil perhitungan adalah matrik satu dimensi atau disebut dengan vektor *output*. Pada program diatas vektor *output* dibandingkan satu per satu dengan data sidik jari yang ada dalam *database*. Kesamaan dengan nilai diatas 80 % yang dapat dipakai sebagai acuan bahwa sidik jari itu teridentifikasi.

BAB IV

PENGUJIAN DAN EVALUASI SISTEM

4.1 Prosedur pengujian

Pada bab pengujian dan evaluasi sistem membahas tentang pengujian dan evaluasi terhadap sistem yang telah dirancang dan dibuat. Sesuai dengan perancangan sistem, pada bab ini terdapat dua proses pengujian. Pengujian pertama adalah proses pengujian pengambilan sidik jari (akuisisi sidik jari), selanjutnya adalah proses pengujian jaringan syaraf *Hopfield*, yang sebelumnya dilakukan pembelajaran sidik jari oleh jaringan syaraf *Hopfield*. Pengujian ini dilakukan menggunakan komputer dengan spesifikasi sebagai berikut : Komputer Intel Pentium 4 2,8 Ghz dengan *memory* 256 MB dan VGA *on board* 32 MB dan untuk akusisi sidik jari menggunakan sensor U.are.U 4000 Digital Persona.



Gambar 4.1 Blok Diagram Prosedur Pengujian

4.2 Hasil Pengujian

4.2.1 Jaringan Syaraf *Hopfield*

Sebelum proses pengujian jaringan syaraf *Hopfield* dilakukan, mula-mula sidik jari yang disimpan dalam *database* akan dipelajari oleh jaringan syaraf *Hopfield*. Data sidik jari yang dipelajari akan disimpan dalam bobot koneksi. Bobot koneksi ini merupakan hasil pembelajaran yang akan berguna untuk mengenali suatu sidik jari yang akan diujikan.

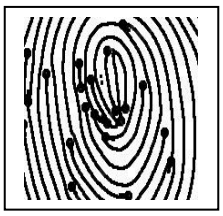
Pada pengujian jaringan syaraf *Hopfield* ini data yang akan diujikan adalah citra sidik jari yang telah dibaca oleh sensor dalam format biner dengan ukuran 50x50 dan 70x70 *pixel*. Sidik Jari yang akan dipelajari akan disimpan dalam *fundamental memory* (*database*) sebagai keadaan stabil (*stable state*). Sebelum dimasukkan ke dalam jaringan syaraf, matrik biner sidik jari ini akan diubah nilainya, yaitu biner yang bernilai nol akan dikonversi menjadi -1 dan biner 1 tidak ada perubahan. Kemudian matrik sidik jari ini akan dipelajari untuk dicari bobot koneksi antara sekian banyak *neuron* dalam jaringan.

Pengujian jaringan syaraf *Hopfield* dilakukan dengan beberapa kali pengujian dengan jumlah data sidik jari yang berbeda-beda. Selain itu sidik jari yang akan diujikan dalam sistem ini adalah sidik jari dengan hasil pembacaan sensor dengan posisi yang tidak dianjurkan (sidik jari dengan perbedaan titik koordinat yang berbeda). Dengan cara ini diharapkan dapat mengetahui kemampuan dari jaringan syaraf *Hopfield* untuk mengenali beberapa sidik jari. Berdasarkan teori jaringan syaraf *Hopfield* mampu mengenali 0,15xN keadaan stabil, dimana N adalah jumlah total *neuron*.

A. Hasil Pengujian Pertama

Pada pengujian pertama ini akan digunakan data adalah sidik jari dengan jumlah 5 buah 50x50 *pixel* dengan perincian sebagai berikut :

1. No. ID : 0141020001
   Nama : Adi
   Alamat : Keputih 45 Surabaya
   No. Telepon : (031) 5927480



2. No. ID : 0141020002
   Nama : Nia
   Alamat : Jl.Tamsing 2/46 Surabaya
   No. Telepon : (031) 3541490

3. No. ID : 0141020003
   Nama : Yudi
   Alamat : Jl.Tamsing 3/37i Surabaya
   No. Telepon : (031) 3535585

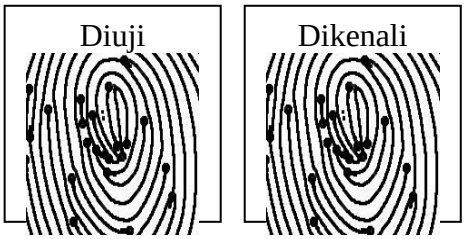
4. No. ID : 0141020004
   Nama : Didik
   Alamat : Jl.Gubeng 4/57 Surabaya
   No. Telepon : (031) 5020495

5. No. ID : 0141020005
   Nama : Irwan
   Alamat : Jl. Siwalankerto 1/21 Surabaya
   No. Telepon : (031) 8436588

Dari lima sidik jari tersebut di atas akan dipelajari oleh jaringan syaraf *Hopfield* untuk mendapatkan bobot koneksi dari jaringan. Setelah lima sidik jari tersebut dipelajari maka selanjutnya adalah dilakukan pengujian terhadap jaringan syaraf tersebut. Berikut pengujiaanya :

1. Pengujian sidik jari yang sama dengan sidik jari yang dipelajari

a. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -7769900

3) Waktu : 16 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0

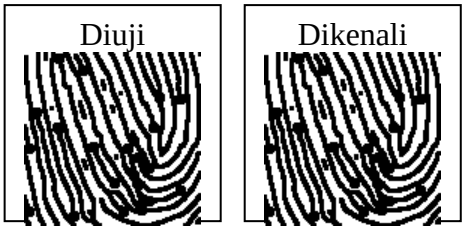
No. ID. : 0141020002 *Hamming Distance* : 0.6497

No. ID. : 0141020003 *Hamming Distance* : 0.5708

No. ID. : 0141020004 *Hamming Distance* : 0.5133

No. ID. : 0141020005 *Hamming Distance* : 0.6500

b. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -6484300

3) Waktu : 16 detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.4919

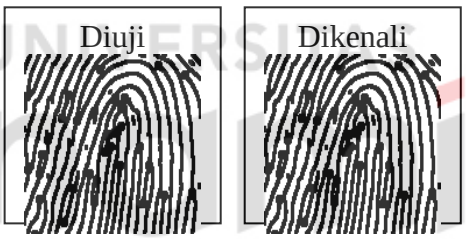
No. ID. : 0141020002 *Hamming Distance* : 0

No. ID. : 0141020003 *Hamming Distance* : 0,5133

No. ID. : 0141020004 *Hamming Distance* : 0.4869

No. ID. : 0141020005 *Hamming Distance* : 0.4900

c. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -6645534

3) Waktu : 16 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.4919

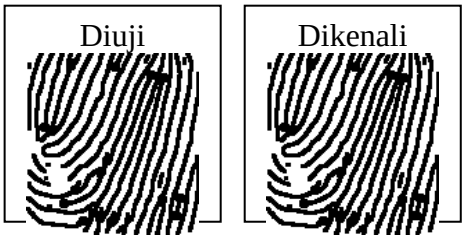
No. ID. : 0141020002 *Hamming Distance* : 0.5211

No. ID. : 0141020003 *Hamming Distance* : 0.5708

No. ID. : 0141020004 *Hamming Distance* : 0

No. ID. : 0141020005 *Hamming Distance* : 0.5347

d. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -7107774

3) Waktu : 16 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.5211

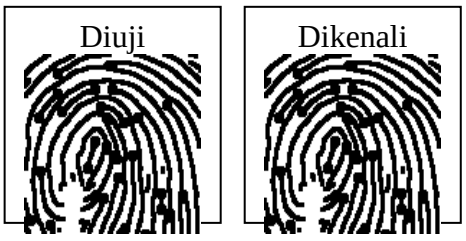
No. ID. : 0141020002 *Hamming Distance* : 0.5391

No. ID. : 0141020003 *Hamming Distance* : 0.6497

No. ID. : 0141020004 *Hamming Distance* : 0.4869

No. ID. : 0141020005 *Hamming Distance* : 0

e. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -7127804

3) Waktu : 16 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.5347

No. ID. : 0141020002 *Hamming Distance* : 0.5391

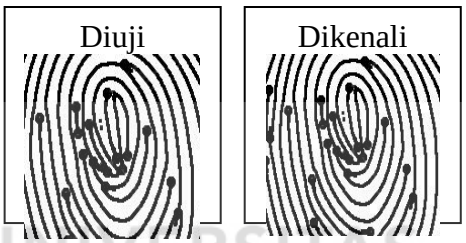
No. ID. : 0141020003 *Hamming Distance* : 0

No. ID. : 0141020004 *Hamming Distance* : 0.49

No. ID. : 0141020005 *Hamming Distance* : 0.65

2. Pengujian sidik jari yang berasal dari pembacaan sensor

a. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 2

2) Energi : -5041529

3) Waktu : 16 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0

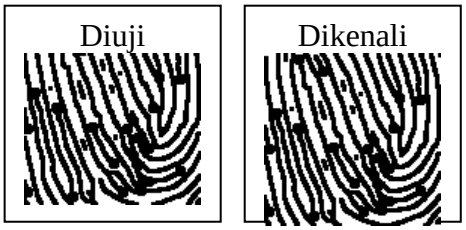
No. ID. : 0141020002 *Hamming Distance* : 0.4876

No. ID. : 0141020003 *Hamming Distance* : 0.4612

No. ID. : 0141020004 *Hamming Distance* : 0.4952

No. ID. : 0141020005 *Hamming Distance* : 0.4492

b. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 2

2) Energi : -4904037

3) Waktu : 16 detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.5124

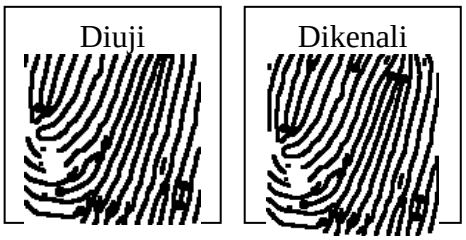
No. ID. : 0141020002 *Hamming Distance* : 0

No. ID. : 0141020003 *Hamming Distance* : 0,4888

No. ID. : 0141020004 *Hamming Distance* : 0,5079

No. ID. : 0141020005 *Hamming Distance* : 0,5112

c. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 2

2) Energi : -4106530

3) Waktu : 16 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.668

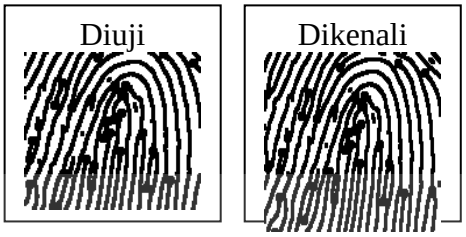
No. ID. : 0141020002 *Hamming Distance* : 0.5036

No. ID. : 0141020003 *Hamming Distance* : 0.6372

No. ID. : 0141020004 *Hamming Distance* : 0.6256

No. ID. : 0141020005 *Hamming Distance* : 0.1172

d. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 2

2) Energi : -4779308

3) Waktu : 16 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.5046

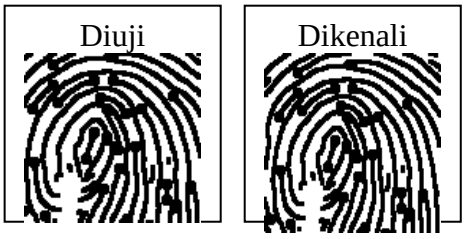
No. ID. : 0141020002 *Hamming Distance* : 0.5076

No. ID. : 0141020003 *Hamming Distance* : 0.4996

No. ID. : 0141020004 *Hamming Distance* : 0

No. ID. : 0141020005 *Hamming Distance* : 0.5084

e. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 3

2) Energi : -7890461

3) Waktu : 24 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.4612

No. ID. : 0141020002 *Hamming Distance* : 0.5112

No. ID. : 0141020003 *Hamming Distance* : 0

No. ID. : 0141020004 *Hamming Distance* : 0.5004

No. ID. : 0141020005 *Hamming Distance* : 0.48

B. Hasil Pengujian Kedua

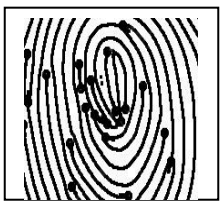
Pada pengujian kedua ini akan digunakan data adalah sidik jari dengan jumlah 5 buah 70x70 *pixel* dengan perincian sebagai berikut :

1. No. ID : 0141020001

   Nama : Adi

   Alamat : Keputih 45 Surabaya

   No. Telepon : (031) 5927480



2. No. ID : 0141020002

   Nama : Nia

   Alamat : Jl.Tamsing 2/46 Surabaya

   No. Telepon : (031) 3541490

3. No. ID : 0141020003

   Nama : Yudi

   Alamat : Jl.Tamsing 3/37i Surabaya

   No. Telepon : (031) 3535585

4. No. ID : 0141020004

   Nama : Didik

   Alamat : Jl.Gubeng 4/57 Surabaya

   No. Telepon : (031) 5020495

5. No. ID : 0141020005

   Nama : Irwan

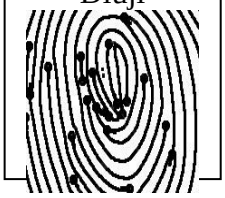
   Alamat : Jl. Siwalankerto 1/21 Surabaya

   No. Telepon : (031) 8436588

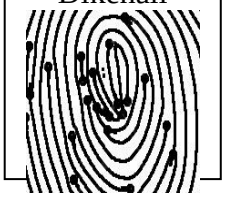
1. Pengujian sidik jari yang sama dengan sidik jari yang dipelajari

a. Sidik jari yang akan diuji adalah sebagai berikut :

Diuji



Dikenali



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -12068380

3) Waktu : 20 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0

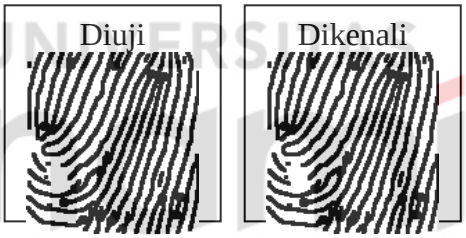
No. ID. : 0141020002 *Hamming Distance* : 0.5002

No. ID. : 0141020003 *Hamming Distance* : 0.4786

No. ID. : 0141020004 *Hamming Distance* : 0.4704

No. ID. : 0141020005 *Hamming Distance* : 0.5086

b. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -12020714

3) Waktu : 19 detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.5002

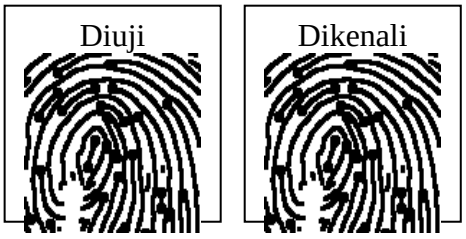
No. ID. : 0141020002 *Hamming Distance* : 0

No. ID. : 0141020003 *Hamming Distance* : 0,4886

No. ID. : 0141020004 *Hamming Distance* : 0.5159

No. ID. : 0141020005 *Hamming Distance* : 0.4859

c. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -12084922

3) Waktu : 19 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.4786

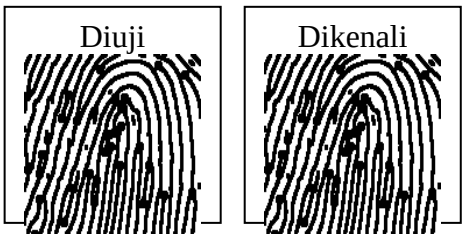
No. ID. : 0141020002 *Hamming Distance* : 0.4806

No. ID. : 0141020003 *Hamming Distance* : 0

No. ID. : 0141020004 *Hamming Distance* : 0.4702

No. ID. : 0141020005 *Hamming Distance* : 0.4789

d. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -12203842

3) Waktu : 19 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.4704

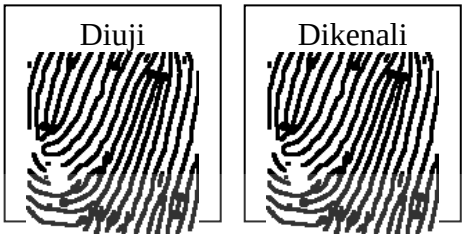
No. ID. : 0141020002 *Hamming Distance* : 0.5159

No. ID. : 0141020003 *Hamming Distance* : 0.4702

No. ID. : 0141020004 *Hamming Distance* : 0

No. ID. : 0141020005 *Hamming Distance* : 0.4512

e. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 1

2) Energi : -12141260

3) Waktu : 19 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.5085

No. ID. : 0141020002 *Hamming Distance* : 0.4859

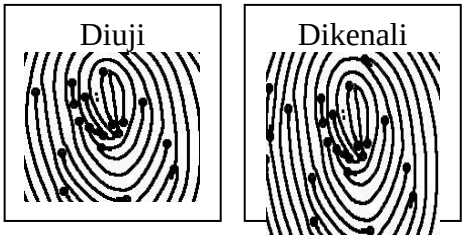
No. ID. : 0141020003 *Hamming Distance* : 0.4789

No. ID. : 0141020004 *Hamming Distance* : 0.4512

No. ID. : 0141020005 *Hamming Distance* : 0

2. Pengujian sidik jari yang berasal dari pembacaan sensor

a. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 3

2) Energi : -30245906

3) Waktu : 56 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0

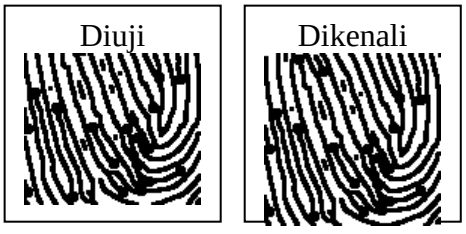
No. ID. : 0141020002 *Hamming Distance* : 0.4998

No. ID. : 0141020003 *Hamming Distance* : 0.5214

No. ID. : 0141020004 *Hamming Distance* : 0.5296

No. ID. : 0141020005 *Hamming Distance* : 0.4914

b. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 2

2) Energi : -15466912

3) Waktu : 40 detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.4998

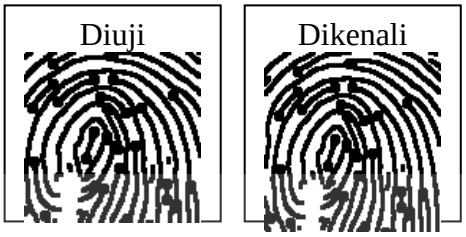
No. ID. : 0141020002 *Hamming Distance* : 0

No. ID. : 0141020003 *Hamming Distance* : 0,5114

No. ID. : 0141020004 *Hamming Distance* : 0,4841

No. ID. : 0141020005 *Hamming Distance* : 0,5191

c. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 3

2) Energi : -30211545

3) Waktu : 60 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.5411

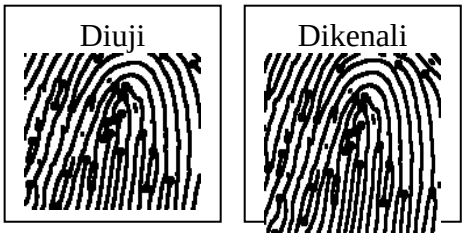
No. ID. : 0141020002 *Hamming Distance* : 0.4898

No. ID. : 0141020003 *Hamming Distance* : 0

No. ID. : 0141020004 *Hamming Distance* : 0.5387

No. ID. : 0141020005 *Hamming Distance* : 0.4586

d. sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 3

2) Energi : -31204656

3) Waktu : 60 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.5296

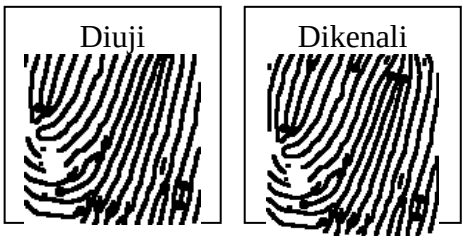
No. ID. : 0141020002 *Hamming Distance* : 0.4841

No. ID. : 0141020003 *Hamming Distance* : 0.5298

No. ID. : 0141020004 *Hamming Distance* : 0

No. ID. : 0141020005 *Hamming Distance* : 0.5488

e. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 3

2) Energi : -32014588

3) Waktu : 60 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 *Hamming Distance* : 0.4914

No. ID. : 0141020002 *Hamming Distance* : 0.5141

No. ID. : 0141020003 *Hamming Distance* : 0.5210

No. ID. : 0141020004 *Hamming Distance* : 0.5488

No. ID. : 0141020005 *Hamming Distance* : 0

C. Hasil Pengujian Ketiga

Pada pengujian ketiga ini akan digunakan data adalah sidik jari dengan jumlah 10 buah 70x70 *pixel* dengan perincian sebagai berikut :

1. No. ID : 0141020001

   Nama : Adi

   Alamat : Keputih 45 Surabaya

   No. Telepon : (031) 5927480

2. No. ID : 0141020002

   Nama : Nia

   Alamat : Jl.Tamsing 2/46 Surabaya

   No. Telepon : (031) 3541490

3. No. ID : 0141020003

   Nama : Yudi

   Alamat : Jl.Tamsing 3/37i Surabaya

   No. Telepon : (031) 3535585

4. No. ID : 0141020004
   Nama : Didik
   Alamat : Jl.Gubeng 4/57 Surabaya
   No. Telepon : (031) 5020495

5. No. ID : 0141020005
   Nama : Irwan
   Alamat : Jl. Siwalankerto 1/21 Surabaya
   No. Telepon : (031) 8436588

6. No. ID : 0141020006
   Nama : Galih
   Alamat : Jl. Gedangan 34 Surabaya
   No. Telepon : (031) 584-5689

7. No. ID : 0141020007
   Nama : Roni
   Alamat : Jl. Kepatihan 90 Surabaya
   No. Telepon : (031) 354-5879

8. No. ID : 0141020008
   Nama : Miga
   Alamat : Jl. Kemayoran 10 Surabaya
   No. Telepon : (031) 353-6548

9. No. ID : 0141020009
   Nama : Heru
   Alamat : Jl. Kebalen 14 Surabaya
   No. Telepon : (031) 354-8976

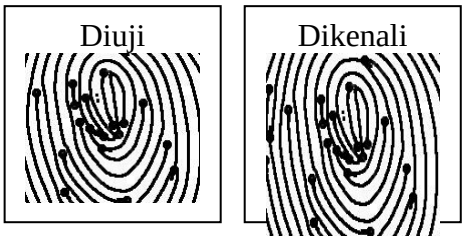
10. No. ID : 0141020010

    Nama : Yogi

    Alamat : Sidotopo 45 Surabaya

    No. Telepon : (031) 365-7895

a. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 6

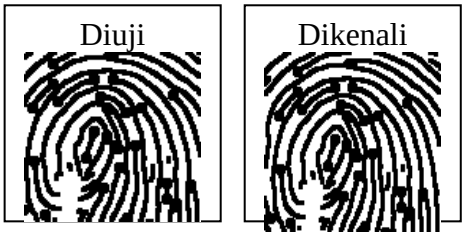
2) Energi : -57822252

3) Waktu : 122 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0	No. ID. : 0141020006 HD: 0.5077
No. ID. : 0141020002 HD: 0.5228	No. ID. : 0141020007 HD: 0.4961
No. ID. : 0141020003 HD: 0.5163	No. ID. : 0141020008 HD: 0.5108
No. ID. : 0141020004 HD: 0.5143	No. ID. : 0141020009 HD: 0.5147
No. ID. : 0141020005 HD: 0.5220	No. ID. : 0141020010 HD: 0.5022

b. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 4

2) Energi : -40498970

3) Waktu : 81 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.5163 No. ID. : 0141020006 HD: 0.5104

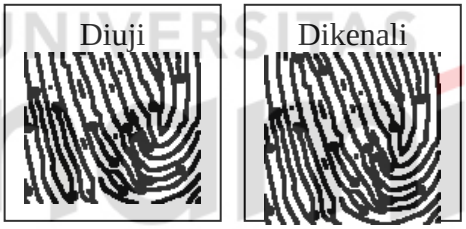
No. ID. : 0141020002 HD: 0.4988 No. ID. : 0141020007 HD: 0.4912

No. ID. : 0141020003 HD: 0 No. ID. : 0141020008 HD: 0.5006

No. ID. : 0141020004 HD: 0.5367 No. ID. : 0141020009 HD: 0.5184

No. ID. : 0141020005 HD: 0.5065 No. ID. : 0141020010 HD: 0.4945

c. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 3

2) Energi : -30590006

3) Waktu : 61 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.5229 No. ID. : 0141020006 HD: 0.5124

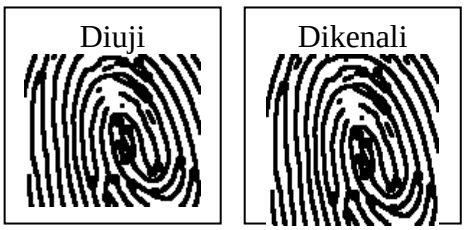
No. ID. : 0141020002 HD: 0 No. ID. : 0141020007 HD: 0.5169

No. ID. : 0141020003 HD: 0.4988 No. ID. : 0141020008 HD: 0.5071

No. ID. : 0141020004 HD: 0.4984 No. ID. : 0141020009 HD: 0.5102

No. ID. : 0141020005 HD: 0.5020 No. ID. : 0141020010 HD: 0.4973

d. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 7

2) Energi : -67508686

3) Waktu : 141 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.5027 No. ID. : 0141020006 HD: 0

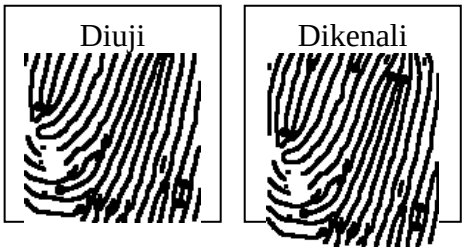
No. ID. : 0141020002 HD: 0.5124 No. ID. : 0141020007 HD: 0.5194

No. ID. : 0141020003 HD: 0.5104 No. ID. : 0141020008 HD: 0.5084

No. ID. : 0141020004 HD: 0.5141 No. ID. : 0141020009 HD: 0.4900

No. ID. : 0141020005 HD: 0.5051 No. ID. : 0141020010 HD: 0.5237

e. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 5

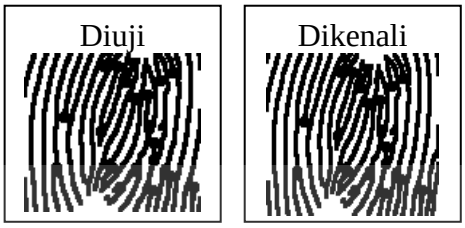
2) Energi : -50403412

3) Waktu : 101 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.5220	No. ID. : 0141020006 HD: 0.5051
No. ID. : 0141020002 HD: 0.5020	No. ID. : 0141020007 HD: 0.5161
No. ID. : 0141020003 HD: 0.5065	No. ID. : 0141020008 HD: 0.5308
No. ID. : 0141020004 HD: 0.5400	No. ID. : 0141020009 HD: 0.4951
No. ID. : 0141020005 HD: 0	No. ID. : 0141020010 HD: 0.5014

f. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 7

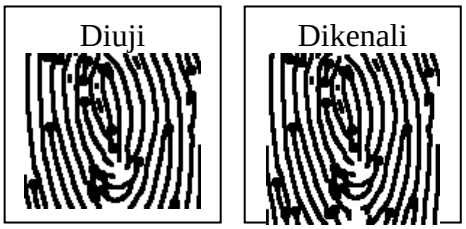
2) Energi : -68251928

3) Waktu : 143 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.5143	No. ID. : 0141020006 HD: 0.5141
No. ID. : 0141020002 HD: 0.4984	No. ID. : 0141020007 HD: 0.5157
No. ID. : 0141020003 HD: 0.5367	No. ID. : 0141020008 HD: 0.5189
No. ID. : 0141020004 HD: 0	No. ID. : 0141020009 HD: 0.4996
No. ID. : 0141020005 HD: 0.5400	No. ID. : 0141020010 HD: 0.5063

g. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 6

2) Energi : -52164636

3) Waktu : 118 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.6289 No. ID. : 0141020006 HD: 0.6496

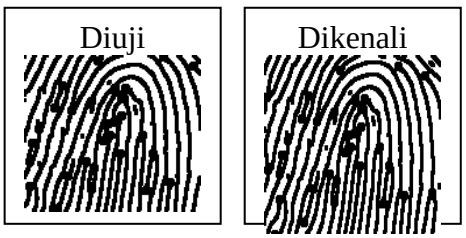
No. ID. : 0141020002 HD: 0.6375 No. ID. : 0141020007 HD: 0.1835

No. ID. : 0141020003 HD: 0.5065 No. ID. : 0141020008 HD: 0.5080

No. ID. : 0141020004 HD: 0.6379 No. ID. : 0141020009 HD: 0.5169

No. ID. : 0141020005 HD: 0.6391 No. ID. : 0141020010 HD: 0.5055

h. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 3

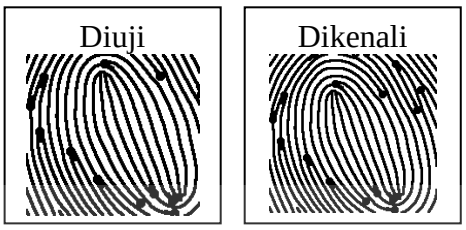
2) Energi : -28842190

3) Waktu : 61 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.4891	No. ID. : 0141020006 HD: 0.4918
No. ID. : 0141020002 HD: 0.4928	No. ID. : 0141020007 HD: 0.5253
No. ID. : 0141020003 HD: 0.4994	No. ID. : 0141020008 HD: 0
No. ID. : 0141020004 HD: 0.4810	No. ID. : 0141020009 HD: 0.4979
No. ID. : 0141020005 HD: 0.4692	No. ID. : 0141020010 HD: 0.4734

i. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 4

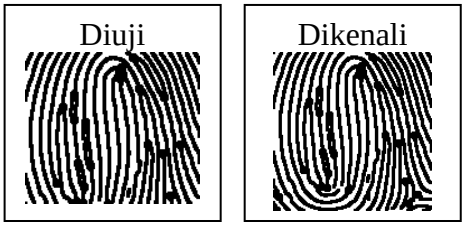
2) Energi : -41254625

3) Waktu : 81 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.5163	No. ID. : 0141020006 HD: 0.5124
No. ID. : 0141020002 HD: 0.4328	No. ID. : 0141020007 HD: 0.4181
No. ID. : 0141020003 HD: 0.5421	No. ID. : 0141020008 HD: 0.5112
No. ID. : 0141020004 HD: 0.5010	No. ID. : 0141020009 HD: 0.1745
No. ID. : 0141020005 HD: 0.5256	No. ID. : 0141020010 HD: 0.5953

j. Sidik jari yang akan diuji adalah sebagai berikut :



Berikut keterangan hasil pengenalan oleh jaringan :

1) Iterasi : 3

2) Energi : -30924784

3) Waktu : 64 Detik

4) *Hamming Distance* :

No. ID. : 0141020001 HD: 0.5108 No. ID. : 0141020006 HD: 0.4833

No. ID. : 0141020002 HD: 0.5006 No. ID. : 0141020007 HD: 0.5089

No. ID. : 0141020003 HD: 0.5145 No. ID. : 0141020008 HD: 0.5265

No. ID. : 0141020004 HD: 0.4982 No. ID. : 0141020009 HD: 0.5098

No. ID. : 0141020005 HD: 0.4896 No. ID. : 0141020010 HD: 0

4.3 Analisa

Dari pengujian tersebut di atas dapat dianalisa bahwa jaringan syaraf *Hopfield* dalam mengidentifikasi sidik jari dipengaruhi beberapa faktor yaitu : jumlah data yang akan dikenali, kemiripan data satu dengan data yang lain, besar pixel dari sidik jari, luas daerah yang akan dipelajari dan dikenali, posisi saat pengambilan sidik jari oleh sensor. Pengujian tersebut di atas hanya menampilkan pengujian pola yang tidak terlalu banyak. Untuk pengujian dengan pola yang banyak tetap dilakukan, tetapi tidak dituliskan dalam bab ini. Namun hasil analisa akan diterangkan secara terperinci. Untuk pengujian yang akan di analisa adalah

pengujian terhadap sidik jari dengan ukuran 50x50 dan 70x70 *pixel*. Dan juga pengujian dengan menggunakan sidik jari dengan posisi sidik jari yang tidak mewakili hampir seluruh area sidik jarinya.

Berikut tabel hasil analisa dengan sidik jari ukuran 50x50 *pixel* :

Tabel 4.1 Hasil Pengujian Pertama

No	Jml Pola	Pixel	Pola	Analisa					
				Iterasi	Energi	Waktu	Hamming Distance *	Persen	Status
1	5	50x50	1 **	1	-7769900	16 detik	HD = 0	100%	Dikenali
2	5	50x50	2 **	1	-6484300	16 detik	HD = 0	100%	Dikenali
3	5	50x50	3 **	1	-7127804	16 detik	HD = 0	100%	Dikenali
4	5	50x50	4 **	1	-6645534	16 detik	HD = 0	100%	Dikenali
5	5	50x50	5 **	1	-7107774	16 detik	HD = 0	100%	Dikenali
6	5	50x50	1	2	-5041529	16 detik	HD = 0	100%	Dikenali
7	5	50x50	2	2	-4904037	16 detik	HD = 0	100%	Dikenali
8	5	50x50	3	3	-7890461	24 detiki	HD = 0	100%	Dikenali
9	5	50x50	4	2	-4779308	16 detik	HD = 0	100%	Dikenali
10	5	50x50	5	2	-4106536	16 detik	HD = 0	100%	Dikenali

Keterangan :
* : Nilai yang paling besar
** : Sidik jari yang sama dengan dengan sidik jari yang dipelajari

Tabel 4.2 Hasil Pengujian Kedua

No.	Jml Pola	Pixel	Pola	Analisa					
				Iterasi	Energi	Waktu	Hamming Distance *	Persen	Status
1	5	70x70	1 **	1	-12068380	20 detik	HD = 0	100%	Dikenali
2	5	70x70	2 **	1	-12020714	19 detik	HD = 0	100%	Dikenali
3	5	70x70	3 **	1	-12084922	19 detik	HD = 0	100%	Dikenali
4	5	70x70	4 **	1	-12203842	19 detik	HD = 0	100%	Dikenali
5	5	70x70	5 **	1	-12141260	19 detik	HD = 0	100%	Dikenali
6	5	70x70	1	3	-30245906	56 detik	HD = 0	100%	Dikenali
7	5	70x70	2	2	-15466912	40 detik	HD = 0	100%	Dikenali
8	5	70x70	3	3	-30211545	60 detik	HD = 0	100%	Dikenali
9	5	70x70	4	2	-31204656	60 detik	HD = 0	100%	Dikenali
10	5	70x70	5	2	-32014588	60 detik	HD = 0	100%	Dikenali

Keterangan :
* : Nilai yang paling besar
** : Sidik jari yang sama dengan dengan sidik jari yang dipelajari

Pada pengujian menggunakan lima buah sidik jari, dalam mengenali diri sendiri dapat dikenali secara keseluruhan, ini dapat dilihat dari tabel pengujian dengan besar *pixel* berbeda tersebut. Dari tabel tersebut dapat dilihat bahwa dengan menggunakan *pixel* 50x50 iterasi dan waktu yang dibutuhkan adalah lebih kecil daripada menggunakan *pixel* 70x70. Selain itu dengan menggunakan *pixel* lebih kecil juga mempengaruhi besar energi, semakin besar *pixel*-nya maka semakin besar energinya.

Sedangkan untuk persen kesamaannya tidak ada perbedaan, pada pengujian ukuran sidik jari 70x70 *pixel*, semua nilai *Hamming Distance*-nya juga sama dengan 0. Sehingga hasil identifikasi dari pengujian tersebut didapat kesamaan yaitu sidik jari yang dimasukkan dapat dikenali semuanya, artinya dari lima sidik jari tersebut persen kesamaannya adalah 100%.

Tabel 4.3 Hasil Pengujian Ketiga

No.	Jml Pola	Pixel	Pola	Analisa					
				Iterasi	Energi	Waktu	Hamming Distance *	Persen	Status
1	10	70x70	1	6	-57822252	122 detik	HD : 0	100%	Dikenali
2	10	70x70	2	3	-40498970	61 detik	HD : 0	100%	Dikenali
3	10	70x70	3	4	-30590006	81 detik	HD : 0	100%	Dikenali
4	10	70x70	4	7	-67508686	143 detik	HD : 0	100%	Dikenali
5	10	70x70	5	5	-50403412	101 detik	HD : 0	100%	Dikenali
6	10	70x70	6	7	-68251928	141 detik	HD : 0	100%	Dikenali
7	10	70x70	7	6	-52164636	118 detik	HD : 0.1835	81%	Dikenali
8	10	70x70	8	3	-28842190	61 detik	HD : 0	100%	Dikenali
9	10	70x70	9	3	-30794308	62 detik	HD : 0.1745	83%	Dikenali
10	10	70x70	10	3	-30924784	64 detik	HD : 0	100%	Dikenali

Keterangan :
* : Nilai yang paling besar

Untuk pengujian dengan pola 50x50 *pixel* tidak dituliskan karena hasilnya hampir sama dengan tabel tersebut di atas. Sedangkan untuk pengujian dengan pola yang sama dengan pola pembelajaran tidak dituliskan juga karena sudah terbukti dapat dikenali dengan sempurna pada pengujian 5 buah sidik jari.

Dari analisa pengujian ketiga tersebut dapa diambil kesimpulan jumlah pola yang semakin banyak yaitu dengan menambahkan 5 buah sidik jari, pola tetap bisa dikenali 100% dari sepuluh sidik jari tersebut. Walupun pada percobaan yang ke-7 dan ke-9 terjadi penurunan persen kesamaan, tetapi masih bisa dianggap dapat dikenali karena masih di atas 80%.

Namun dengan penambahan sidik jari yang akan dipelajari maka akan mengakibatkan proses pembelajaran dan pengenalan sidik jari semakin lambat. Ini diakibatkan oleh jumlah perhitungan *output* jaringan *Hopfield* semakin besar, terjadi penumpukan bit data pada matrik bobot koneksi. Energi yang dibutuhkan untuk proses pengenalan juga semakin besar, sedangkan untuk nilai *Hamming Distance* terjadi peningkatan, walau hanya terjadi pada pola yang ke-7 dan ke-9.

Tabel 4.4 Hasil Pengujian Sidik Jari keempat

No.	Jml Pola	Pixel	Pola	Analisa					
				Iterasi	Energi	Waktu	Hamming Distance *	Persen	Status
1	5	50x50	1	2	-3905726	15 detik	HD : 0.2316	77%	Tidak Dikenali
2	5	50x50	2	3	-6151601	25 detik	HD : 0.244	76%	Tidak Dikenali
3	5	50x50	3	2	-3420491	14 detik	HD : 0.254	75%	Tidak Dikenali
4	5	50x50	4	3	-7328988	35 detik	HD : 0	100%	Dikenali
5	5	50x50	5	2	-3580255	14 detik	HD : 0.246	75%	Tidak Dikenali

Keterangan :
* : Nilai yang paling besar

Dari tabel pengujian tersebut di atas dapat dianalisa bahwa hasil pengujian dengan cara memasukkan sidik jari dengan posisi sidik jari tidak pada ketentuannya atau posisi jarinya seperti contoh posisi sidik jari yang tidak dianjurkan, sehingga area sidik jari tidak terbaca dengan secara keseluruhan oleh sensor. Akibatnya hanya satu sidik jari yang dapat dikenali dari lima percobaan. Sedangkan empat sidik jari yang lainnya tidak dapat dikenali, ini dapat dilihat dari nilai persen kesamaannya yang kurang dari 80%. Jadi hasil analisa di atas menyatakan bahwa sistem hanya dapat dikenali 20% dari lima sidik jari tersebut.

Dalam penelitian ini mencoba untuk mengidentifikasi sidik jari sebanyak 256 buah. Setelah dilakukan penelitian hasilnya tidak terlalu baik, karena dengan sekian banyak data biner sidik jari membuat pembelajaran jaringan syaraf *Hopfield* semakin berat. Dengan data biner yang sekian banyak tidak menutup kemungkinan terdapat pola guratan-guratan sidik jari yang nilai sama pada koordinat tertentu. Keadaan ini akan menimbulkan proses perhitungan *output* jaringan tidak mencapai konvergen dengan cepat artinya terjadi sekian kali iterasi dan hasilnya identifikasi kurang maksimal. Ini dapat dilihat pada saat dilakukan tes untuk memasukkan pola yang sama dengan pola pembeljaran ternyata hasilnya masih ada pola yang tidak dikenali.

Dalam pembelajaran pola sidik jari, dari sekian banyak sidik jari akan dihitung bobot koneksinya dengan cara menjumlahkan hasil perkalian tiap-tiap *pixel* pada pola-pola sidik jari. Jadi hasil pencarian bobot koneksi ini merupakan hasil kombinasi dari sekian banyak nilai biner pola sidik jari. Jika terdapat nilai pada koordinat tertentu yang sama akan mengakibatkan penghilangan ciri yang mutlak pada pola sidik jari.

Ada kemungkinan nilai biner sidik jari yang mirip akan dianggap sama.

pada iterasi pertama, kedua, ketiga dan berikutnya ada kemungkinan perubahan secara mutlak dari pola tersebut dan akan condong pada pola yang lain yang dianggap pola yang paling dominan dalam jaringan.

BAB V

PENUTUP

5.1 Kesimpulan

1. Waktu yang dibutuhkan untuk pembelajaran dan pengenalan sidik jari oleh jaringan syaraf *Hopfield* dipengaruhi oleh ukuran dan jumlah sidik jari, artinya jika semakin besar ukuran dan jumlah sidik jari semakin banyak maka waktu yang dibutuhkan untuk proses pembelajaran dan pengenalan akan semakin lama.
2. Jaringan syaraf *Hopfield* dalam pembelajaran dan pengenalan sidik jari membutuhkan spesifikasi komputer yang bagus, karena jaringan syaraf ini berhubungan dengan perhitungan matrik yang sangat besar sehingga membutuhkan *memory* komputer yang besar dan cepat.
3. Area sidik jari yang digunakan untuk dipelajari sangat berpengaruh, karena pengambilan sidik jari untuk proses pembelajaran dan pengambilan sidik jari untuk proses pengenalan mempunyai perbedaan titik koordinat (posisi jari pada sensor). Jadi semakin luas area sidik jari yang akan diambil maka semakin besar pula perbedaan titik koordinat antara sidik jari yang sama.
4. Jaringan syaraf *Hopfield* ini mampu untuk mengenali 100% dari sepuluh sidik jari dengan syarat sidik jari yang akan diujikan posisi sidik jari dan titik koordinatnya tidak jauh berbeda dengan sidik jari saat pembelajaran.
5. Hasil pembelajaran dari jaringan syaraf *Hopfield* ini adalah merupakan kombinasi dari sekian banyak pola biner yang dihitung dengan rumus penjumlahan dari hasil perkalian koordinat dari sumbu x dan y dari pola-pola

sidik jari tersebut. Selain itu bobot koneksi dari jaringan syaraf *Hopfield* ini adalah tetap tidak ada perubahan. Tidak ada pengecekan error, bobot untuk sekian banyak pola hanya terwakili oleh satu bobot yang sama. Maka hanya tergantung pada satu matrik bobot yang sama, sehingga jika ada kemiripan polanya jaringan ini akan tidak dapat membedakan dengan terperinci.

5.2 Saran

Dalam pembelajaran dan pengenalan sidik jari, jaringan syaraf *Hopfield* membutuhkan waktu yang lama. Metode lain yang mungkin lebih baik adalah menggunakan metode konvensional, yaitu dengan cara membandingkan guratan-guratan sidik jari yang telah ditandai (*Minutiae Matching*) atau menggunakan cara jaringan syaraf metode lain yang mungkin lebih baik dan efisien.

DAFTAR PUSTAKA

Abbas, M. 2001. *Neural Networks in Pattern Calssification*. Egypt : Zagazig University Banha Branch

Achmad, B. & Firdausy, K. 2005. *Teknik Pengolahan Citra Digital Menggunakan Delphi.* Yogyakarta: Ardi Publishing.

Babadi, B. *Neural Networks Architecture.* (http://www.sis.ipm.ac.ir/seminars/weekly%20seminars/course/Neural%20modeling/Babadi11.ppt, diakses 30 Agustus 2005)

Blum A. 1992. *Neural Network in C++*. New York : John Wiley & Sons Inc.

Elvayandi. 2002. *Sistem Keamanan Akses Menggunakan Pola Sidik Jari Berbasis Jaringan Saraf Tiruan.* Bandung : ITB

Fausett, L. 1994. *Fundamentals of Neural Networks: architectures, algorithms, and applications.* New Jersey : Prentice Hall International Inc.

Jain, A., Pankanti, S. & Prabhakar, S. *Learning Fingerprints Minutiae Location and Type. (*http://biometrics.cse.msu.edu/publications.html, diakses 24 April 2006

Kristanto, A. 2004. *Jaringan Syaraf Tiruan (Konsep Dasar, Algritma dan Aplikasi)*. Yogyakarta : Gava Media

Machan, L. *Fp Bio-metrics Road Show.* (http://www.fptech.com/Products/Demos/fpbio-01.pdf, diakses 30 Agustus 2005)

Munir, R. 2004. *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Bandung : Informatika.

Pitas, I. 1993. *Digital Image Processing Algorithms.* New York : Prentice Hall

Pratomo, D. 2002. *Rancang bangun perangkat lunak pengenalan wajah dengan menggunakan Hopfield Network.* Surabaya : STIKOM

Suhartati, F., Fahmi, A. & Indratno, W. 2003. *Pengenalan Pola Sidik Jari dengan Artificial Neural Network Menggunakan Metode Backpropagation.* Malang : UNIBRAW

Wahyu, M. 2003. *Pengenalan identitas Diri Menggunakan Pola Sidik Jari dengan Metode Backpropagation Neural Networks*. Surabaya : STIKOM

August, M., Gerald, W. & Markus, S. *Application of Hopfield Networks*. (http://student.cosy.sbg.ac.at/~amayer/ projects/hopfield/Hopfield.pdf, diakses 24 April 2005)

Nalwan, Agustinus. 1997. *Pengolahan Gambar Secara Digital*. Jakarta: Elex Media Komputindo.

Siang, Jong Jek. 2005. *Jaringan Syaraf Tiruan dan Pemrogramannya Menggunakan Matlab*. Yogyakarta: Andi Offset.

Schalkoff, R., J. 1992. *Pattern Recognition : Statistical, Structural and Neural Approaches*. US : John Wiley & Sons Inc.