PENERAPAN HETEROASSOCIATIVE MEMORY UNTUK PENGENALAN POLA

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika

Disusun oleh :

Nama : ZUHDAN MARWANTO

No. Mhs : 99 523 115

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2004

LEMBAR PENGESAHAN PEMBIMBING

PENERAPAN HETEROASSOCIATIVE MEMORY UNTUK PENGENALAN POLA

TUGAS AKHIR

Oleh:
Nama : ZUHDAN MARWANTO
No. Mhs : 99 523 115

Yogyakarta, Juni 2004

Pembimbing

Sri Kusumadewi, S.Si., M.T.

LEMBAR PENGESAHAN PENGUJI

PENERAPAN HETEROASSOCIATIVE MEMORY UNTUK PENGENALAN POLA

TUGAS AKHIR

Oleh :

Nama : ZUHDAN MARWANTO

No. Mhs : 99 523 115

Telah Dipertahankan di Depan Sidang Penguji sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika
Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, Juni 2004

Tim Penguji

Sri Kusumadewi, S.Si., M.T.
Ketua

Amy Fauzijah, S.T., M.T.
Anggota I

Wawan Indarto, S.T.
Anggota II

Mengetahui,
Dekan Fakultas Teknologi Industri
Universitas Islam Indonesia

Ir. H. Bachrun Sutrisno, M.Sc.

MOTTO

Katakanlah (ya Muhammad): Dialah Allah yang Maha Esa. (Al - Ikhlaas:1)

Tiadalah Aku jadikan Jin dan Manusia, melainkan supaya mereka menyembah kepada-Ku. (Az - Zariyaat : 56)

Sesungguhnya Allah tiada merubah suatu kaum, kecuali jika mereka mengubah keadaan diri mereka sendiri. (Ar – Ra'du)

HALAMAN PERSEMBAHAN

Kupersembahkan Tugas Akhir ini kepada
Ayah dan ibundaku tercinta dan
Saudaraku yang selalu memberikan
dukangan dan semangat.

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Alhamdulillah, segala puji dan syukur penulis panjatkan kehadirat Allah Swt, karena atas rahmat, taufik dan hidayah-Nya laporan tugas akhir ini dapat terselesaikan.

Laporan tugas akhir ini disusun sebagai syarat untuk mendapatkan gelar Sarjana pada Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.

Penulis menyampaikan ucapan terima kasih kepada semua pihak yang telah memberikan bantuan, untuk itu secara khusus kepada :

1. Ibu Sri Kusumadewi, S.Si., M.T. selaku Ketua Jurusan Teknik Informatika Universitas Islam Indonesia dan Dosen Pembimbing.
2. Dosen-dosen FTI khususnya Jurusan Teknik Informatika, yang telah mengajar, memberi contoh, semangat dan nasehat selama 5 tahun.
3. Orang tua dan saudara-saudara yang selalu setia mendukung.
4. Sahabat-sahabat yang selalu memberikan dorongan dan bantuan selama masa kuliah hingga saat ini: Gunawan, Adhi Sura, Encus, Rofiq, Bustomy, Risky Raharjo, Irka, Arif K, Shodiq, Udi, Rahmat K, Andri S.
5. Teman-teman seperjuangan khususnya Informatika angkatan 99 yang telah membantu selama ini.

Penulis Menyadari bahwa tugas akhir ini masih jauh dari sempurna, untuk itu sangat diharapkan kritik dan saran yang bersifat membangun untuk perbaikan dimasa mendatang. Semoga tugas akhir ini dapat bermanfaat. Amin.

Wassalamu'alaikum Wr. Wb.

Yogjakarta, 21 Juni 2004

Penulis

ABSTRAKSI

Jaringan syaraf tiruan adalah salah satu sistem pemrosesan informasi yang didesain dengan menirukan cara kerja otak manusia dalam menyelesaikan suatu masalah. Keberadaannya terbukti telah membawa dampak yang sangat besar dalam perkembangan teknologi manusia yang menuntut kemudahan, kepraktisan dan kecepatan. Jaringan syaraf tiruan telah digunakan dalam berbagai bidang seperti kedokteran, industri, hingga pengembangan robotika.

Dalam perkembangannya di bidang pengenalan pola, teknologi jaringan syaraf tiruan dalam mengatasi masalah akan digunakan untuk mengenali bentuk-bentuk pola dan pemetaan sinyal kompleks. Cukup banyak metode yang diterapkan dalam jaringan syaraf tiruan digunakan untuk mengenali bentuk-bentuk pola, salah satunya adalah jaringan syaraf *heteroassociative memory*.

Jaringan syaraf *heteroassociative memory* adalah jaringan yang bobot-bobotnya telah ditentukan sedemikian rupa sehingga jaringan tersebut dapat menyimpan kumpulan pengelompokan pola. Dalam jaringan ini algoritma pembelajaran yang digunakan adalah algoritama *hebb rule* dan *delta rule*.

Keberhasilan jaringan syaraf tiruan dalam pengenalan pola sangat tergantung pada algoritma pelatihan yang digunakan serta bentuk pola yang dilatih. Algoritma jaringan syaraf tiruan cukup handal dalam mengenali bentuk pola yang pernah dilatihkan kepadanya walaupun bentuk pola tersebut bayak memiliki kemiripan.

Kata Kunci : Jaringan Syaraf Tiruan, *Heteroassociative Memory*, *Delta Rule*, *Hebb Rule*, *learning rate*, *threshold*, Pelatihan, Pengujian.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Seiring dengan perkembangan jaman dan makin kompleksnya kebutuhan, maka manusia berusaha mencari jalan terobosan dalam bidang rekayasa teknologi untuk memudahkan dan menyempurnakan hidupnya. Perkembangan besar-besaran terjadi dalam bidang informatika dan komputer. Komputer tidak menjadi barang mewah lagi, akan tetapi sudah menjadi kebutuhan manusia di jaman ini. Kebutuhan tersebut antara lain untuk pengolahan informasi atau data dalam berbagai macam bidang kehidupan. Mulai dari bidang keuangan, militer, kedoteran, teknologi rekayasa, seni dan masih banyak bidang yang lainnya. Pemakaian teknologi komputer berkembang terus dengan munculnya berbagai perangkat lunak ataupun perangkat keras yang lebih lengkap dan canggih. Selaras dengan hal tersebut, ditemukan teknologi komputer yaitu Kecerdasan Buatan (*Artificial Inteligence*). Kecerdasan buatan adalah sebuah pembahasan yang mempelajari cara untuk memberikan komputer kemampuan berpikir layaknya manusia dengan menggunakan aturan-aturan dan pelatihan sehingga ia dapat mengambil keputusan selayaknya manusia.

Jaringam syaraf tiruan merupakan cabang dari kecerdasan buatan. Sistem ini meniru cara kerja otak manusia. Muncul pertama kali ketika para ahli melihat kesamaan antara sifat psikologi otak dengan pemrosesan dan komputer (1940).

metode pembelajaran *Hebb Rule*, dan lain sebagainya. Sampai saat ini jaringan syaraf tiruan terus diteliti dan dikembangkan serta diaplikasikan di kehidupan nyata manusia.

Dengan perkembangan aplikasi di bidang pengenalan pola, teknologi jaringan syaraf tiruan dalam mengatasi masalah akan digunakan untuk mengenali bentuk-bentuk pola dan pemetaan sinyal-sinyal kompleks. Pada proses pengenalan pola jaringan akan belajar dari pola-pola yang diberikan dan mengingatnya. Sehingga saat dimasukkan pola masukan yang mirip atau sama dengan pola yang telah dipelajari maka jaringan tersebut akan mengidentifikasikannya dan memberikan pola keluaran yang sesuai. Dalam tugas akhir ini penerapan jaringan syaraf tiruan digunakan untuk mengenali bentuk-bentuk pola karakter 2 dimensi.

Masukan pola pada jaringan syaraf tiruan bisa berupa citra, simbol atau bilangan, tetapi untuk pengolahannya selalu dialihkan menjadi sekumpulan bilangan yang disebut dengan vektor masukan. Keluaran dari jaringan syaraf tiruan berupa vektor yang dapat dikembalikan lagi menjadi citra, simbol atau bilangan. Dari hal tersebut maka penulis mencoba membangun aplikasi yang dapat mengenali bentuk-bentuk pola yang diiputkan. Aplikasi yang akan dibangun menerima masukan berupa pola karakter 2 dimensi. *Software* ini menggunakan metode *heteroassociative memory* yang merupakan salah satu metode dalam jaringan syaraf tiruan untuk menyimpan kumpulan pengelempokan pola. Dari pola yang disimpan akan dihitung nilai-nilai masukan dari bentuk karakter 2 dimensi yang dipolakan menjadi nilai dan dimasukkan dalam rumus. Hasil yang

telah diperoleh dari perhitungan dipolakan lagi menjadi pola karakter 2 dimensi.

1.2 Rumusan Masalah

Bagaimana membangun aplikasi jaringan syaraf tiruan yang dapat digunakan untuk mengenali bentuk-bentuk pola karakter 2 dimensi berdasarkan bentuk pola yang diinputkan.

1.3 Batasan Masalah

Batasan masalah yang digunakan dalam penelitian tugas akhir ini adalah :

a. Sistem yang akan dibangun adalah sebuah sistem aplikasi jaringan syaraf tiruan yang akan melakukan proses pembelajaran untuk mengenali bentuk-bentuk pola.
b. Pola sebagai masukan dan target yang akan dilakukan proses pembelajaran, disimbolkan oleh pola karakter 2 dimensi berupa matrik dengan batas minimal ukuran 5 x 3 dan batas maksimal ukuran 20 x 20.
c. Model jaringan syaraf yang digunakan adalah *heteroassociative memory* dengan fungsi aktivasi bipolar.
d. Bahasa pemrograman yang digunakan adalah Borland Delphi 7.0.

1.4 Tujuan Penelitian

Penelitian yang dilakukan bertujuan untuk mengaplikasikan algoritma jaringan syaraf tiruan dengan metode *heteroassociative memory* dan merancang suatu sistem yang dapat mengenali bentuk-bentuk pola karakter 2 dimensi.

1.5 Manfaat Penelitian

Manfaat yang diperoleh dari penelitian ini adalah :

a. Membuat jaringan syaraf tiruan yang mampu mengenali bentuk-bentuk pola yang disimbolkan oleh pola karakter 2 dimensi.
b. Penelitian diharapkan dapat memberi masukan yang cukup berarti bagi pengembangan penelitian di bidang aplikasi berbasis jaringan syaraf tiruan.

1.6 Metodologi Penelitian

1.6.1 Pengumpulan Data

Metode pengumpulan data merupakan tahap awal untuk mengumpulkan informasi sebagai bahan penelitian. Data berupa pola-pola karakter 2 dimensi yang selanjutnya digunakan sebagai pembelajaran. Data didapat dari studi literature yang berhubungan dengan sistem jaringan syaraf tiruan. Data diolah agar dapat sesuai dengan sistem yang dibuat, sehingga dapat diterapkan dalam sistem jaringan syaraf tiruan.

1.6.2 Analisis Kebutuhan

Tahap ini merupakan tahap lanjutan penelitian untuk menganalisis data yang sudah terkumpul, yaitu analisis kebutuhan masukan, keluaran, dan arsitektur model sistem jaringan syaraf tiruan.

1.6.3 Perancangan Sistem

Dalam tahap ini data bertujuan untuk membuat model sistem yang sempurna yang kemudian menentukan bentuk-bentuk penjabaran yang lebih detail dan mendekati ke tahap implementasi serta membuat antarmuka yang mudah dimengerti dan digunakan oleh user.

1.6.4 Implementasi

Setelah data dan program siap, maka data dimasukkan dalam program yaitu dalam tahap pelatihan dan pengujian. Jika penerapan sistem sudah berjalan lancar, maka sistem dapat diimplementasikan langsung untuk melakukan prediksi terhadap keluaran yang diinginkan.

1.7 Sistematika Penulisan

BAB I : PENDAHULUAN

Pada bab ini meliputi latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian serta sistematika penulisan dalam penyusunan laporan ini.

BAB II : LANDASAN TEORI

Berisi landasan teori yang berhubungan dengan penelitian ini, yakni tentang arsitektur jaringan syaraf biologi, arsitektur jaringan syaraf tiruan, fungsi aktivasi dan metode jaringan syaraf tiruan yang diaplikasikan.

BAB III : ANALISIS KEBUTUHAN PERANGKAT LUNAK

Mengemukakan analisis kebutuhan perangkat lunak yang meliputi metode analisis, analisis kebutuhan masukan, keluaran, fungsi yang dibutuhkan serta antarmuka yang diinginkan.

BAB IV : PERANCANGAN PERANGKAT LUNAK

Berisi metode perancangan perangkat lunak yang dipakai, dan juga hasil perancangannya.

BAB V : IMPLEMENTASI PERANGKAT LUNAK

Mengutarakan tentang implementasi perangkat lunak yang meliputi batasan implementasi dan implementasi perangkat lunak berupa *form* hasil perancangan yang dibangun.

BAB VI : ANALISIS KINERJA PERANGKAT LUNAK

Menjelaskan tentang analisis kinerja perangkat lunak yang memuat dokumentasi pengujian perangkat lunak terhadap kebutuhan perangkat lunak yang meliputi metode analisis, penanganan kesalahan, serta analisis keluaran.

BAB VII : PENUTUP

Bab ini menguraikan kesimpulan yang diperoleh dari proses-proses perancangan perangkat lunak yang telah dilakukan serta mengemukakan saran-saran bagi pengembangan yang akan datang.

BAB II

LANDASAN TEORI

Jaringan syaraf tiruan adalah merupakan salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut. Istilah buatan disini digunakan karena jaringan syaraf ini diimplementasikan dengan menggunakan program komputer yang mampu menyelesaikan sejumlah proses perhitungan selama proses pembelajaran [KUS03].

2.1 Jaringan Syaraf Biologi

Otak manusia tersusun atas sel-sel syaraf atau *neuron-neuron* yang membentuk jaringan. Sekumpulan *neuron* (syaraf) yang saling terhubung ini berfungsi untuk memberikan sebuah tanggapan atas sebuah rangsangan yang mengenainya. Setiap rangsangan memiliki tanggapan-tanggapan tersendiri dari sebuah *neuron* tergantung dari tingkat pengenalan *neuron* tersebut terhadap jenis rangsangan yang mengenainya.

Sebuah *neuron* memiliki 3 tipe komponen: yaitu *dendrit* (bagian yang menerima rangsang dari axon), *soma* (badan sel syaraf) dan *axon* (bagian sel yang berhubungan dengan *dendrit* sel syaraf lain dan membentuk simpul yang disebut sinapsis). *Dendrit* dapat menerima banyak sinyal dari *neuron* lain. Sinyal adalah impuls listrik yang dipancarkan menyebrangi celah sinapsis yang disebabkan proses kimia. Tindakan dari pancaran proses kimia mengubah sinyal yang datang

(secara khas, dengan penskalaan frekuensi sinyal yang diterima). Proses tersebut sama dengan sifat bobot dalam jaringan syaraf tiruan [FAU94].

Soma, atau badan sel, menjumlahkan sinyal yang datang. Ketika masukan cukup diterima, sel menjadi aktif, saat itulah sel mengirimkan sinyal melalui *axon*nya ke sel lain. Kejadian ini menimbulkan anggapan bahwa setiap sel syaraf berada dalam keadaan aktif atau tidak aktif, pada setiap satuan waktu. Sehingga pengiriman sinyal dikenali sebagai kode biner. Kenyataannya, frekuensi dari keadaan aktif bervariasi, sesuai dengan kekuatan sinyal yakni kuat atau lemah magnitudenya. Pencocokan dengan kode biner ini dilakukan untuk menentukan tahap-tahap dalam tiap waktu diskrit dan menjumlahkan semua aktivitas (sinyal diterima atau dikirim) pada tahap tertentu berdasarkan satuan waktu.

Transmisi sinyal dari *neuron* tertentu disempurnakan dengan hasil kerja energi potensial *neuron* yang disebabkan perbedaan konsentrasi ion-ion dari setiap sisi sarung pelindung *axon neuron* (sumsum otak manusia). Ion-ion kebanyakan secara langsung melibatkan zat-zat potassium, sodium dan klorida.

Beberapa fitur penting proses elemen dari jaringan syaraf tiruan yang berasal dari cara kerja jaringan syaraf biologi [FAU94]:

1. Elemen pemroses menerima beberapa sinyal.
2. Sinyal memungkinkan dimodifikasi oleh bobot pada sinapsis penerima.
3. Elemen pemroses menjumlahkan bobot *input*.
4. Dalam lingkungan yang sesuai (jumlah *input* yang sesuai), *neuron* mengirimkan *output* tunggal.

5. *Output* dari *neuron* khusus memungkinkan dipindahkan ke beberapa *neuron* lain (melalui cabang *axon*).

Beberapa fitur jaringan syaraf tiruan yang dipelajari dari *neuron* biologi:

6. Pemrosesan informasi bersifat lokal (meskipun cara berbeda dalam proses transmisi, seperti aksi beberapa hormon, memungkinkan penganjuran cara control proses yang bersifat keseluruhan).
7. Memori terdistribusi :
   a. Memori yang berjangka panjang berada dalam sinapsis *neuron* atau bobot.
   b. Memori jangka pendek merespon sinyal kiriman oleh *neuron*.
8. Kekuatan sinapsis dapat dimodifikasi oleh pengalaman.
9. *Neuron* pengirim untuk sinapsis mungkin bersifat pengeksitasi atau penghambat.

Karakteristik penting lain jaringan syaraf tiruan yang merupakan bagian dari sistem syaraf biologi adalah toleransi kesalahan/kekurangan data. Sistem syaraf biologi memiliki toleransi kesalahan dalam 2 aspek [FAU94]:

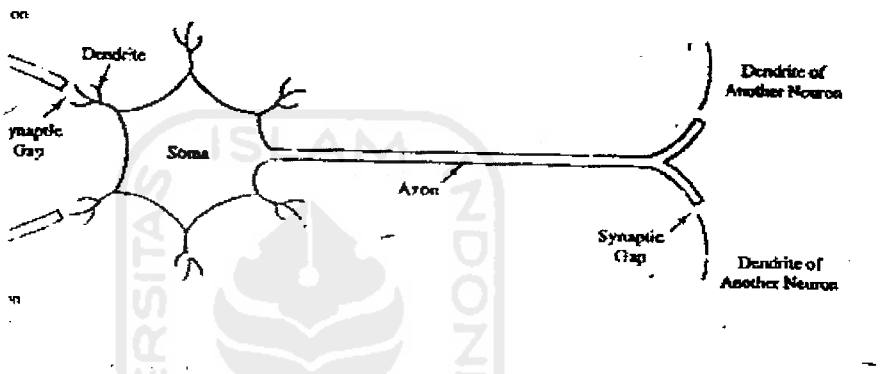
1. Dapat mengenali banyak *input* sinyal yang beberapa diantaranya berbeda dengan yang pernah dikenali sebelumnya. Sebagai contoh kemampuan kita untuk mengenali seseorang dari suatu gambaran atau mengenali seseorang setelah periode yang lama.
2. Dapat menerima kerusakan ke dalam sistem syaraf itu sendiri.

   Manusia memiliki lebih dari 100 trilyun *neuron*. Kebanyakan diantaranya berada di otak. Jika terjadi kerusakan pada salah satu bagian, maka akan

memungkinkan bagian lain akan menjalankan fungsi dari *neuron* yang rusak (FAU94).

Penelitian pertama yang menggunakan prinsip-prinsip jaringan syaraf biologi adalah penelitian oleh McCulloch-Pitts pada dekade 40-an.

Ilustrasi dari jaringan syaraf biologi terlihat pada Gambar 2.1.



Gambar 2.1 Gambar Jaringan Syaraf Biologi

2.2 Jaringan Syaraf Tiruan

2.2.1 Pengertian Jaringan Syaraf Tiruan

Jaringan syaraf tiruan merupakan sistem pemrosesan informasi yang memiliki karakteristik kemampuan yang secara umum mirip dengan jaringan syaraf biologis. Jaringan syaraf tiruan telah dikembangkan sebagai turunan model matematika dari kesadaran manusia atau syaraf biologis, karena berdasar pada asumsi bahwa [FAU94]:

1. Pemrosesan informasi terjadi pada beberapa elemen sederhana yang disebut *neuron.*

2. Sinyal lewat diantara *neuron* menciptakan jaringan koneksi.
3. Setiap koneksi penghubung memiliki bobot yang terhubung yang dalam jaringan syaraf tertentu mengalikan sinyal yang ditransmisikan.
4. Setiap *neuron* mempunyai fungsi aktivasi (biasanya non linier) pada jaringan *input*nya (jumlah dari bobot sinyal *input*) untuk menentukan sinyal o*utput*nya.

Karakteristik dari jaringan syaraf tiruan adalah [FAU94]:

1. Pola hubungan antar *neuron* (yang menjadi arsitekturnya).
2. Metode penentuan bobot dalam koneksi (disebut sebagai proses latihan, pembelajaran, atau Algoritma).
3. Fungsi aktivasi.

Jaringan syaraf terdiri dari sebagian besar unsur-unsur pengolahan sederhana yang disebut *neuron*, unit, sel, atau node. Masing-masing *neuron* terhubung dengan *neuron* yang lain dengan bobot masing-masing. Bobot merepresentasikan informasi yang digunakan oleh jaringan untuk memecahkan masalah.

2.2.2 Karakteristik Jaringan Syaraf Tiruan

Pada bagian ini, akan diterangkan 3 karakteristik jaringan syaraf tiruan.

2.2.2.2 Arsitektur Jaringan Syaraf Tiruan

Umumnya, *neuron* yang terletak pada lapisan yang sama akan memiliki keadaan yang sama. Faktor terpenting dalam menentukan kelakuan suatu *neuron*

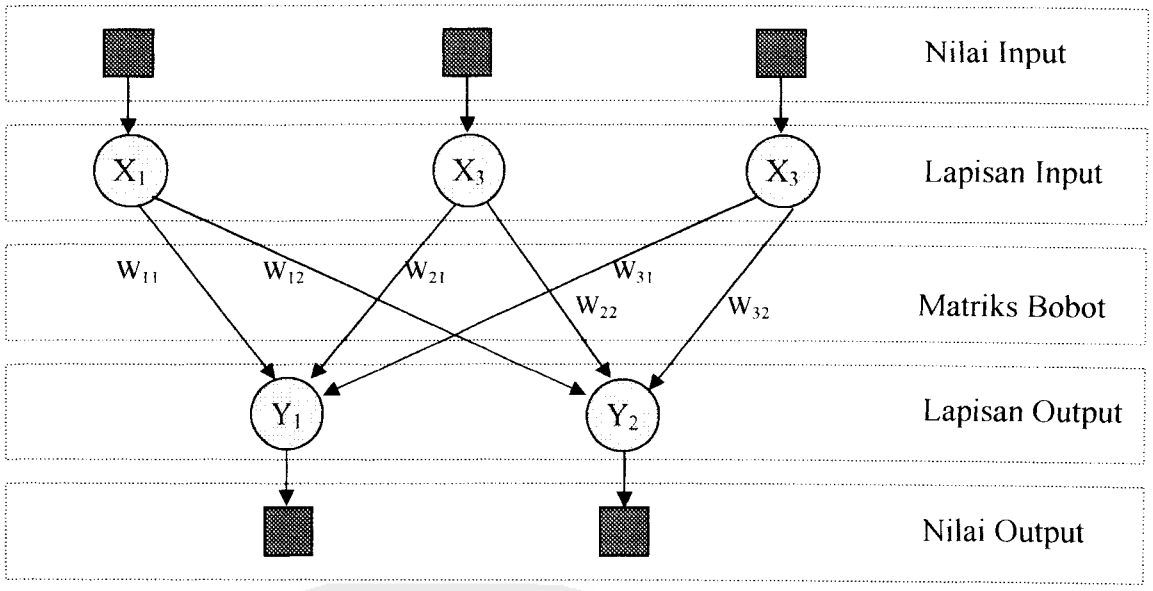
adalah fungsi aktivasi dan pola bobotnya. Pada setiap lapisan yang sama, setiap *neuron* akan memiliki fungsi aktivasi yang sama. Apabila *neuron* dalam suatu lapisan (misalkan lapisan tersembunyi) akan dihubungkan dengan *neuron* yang lain (misalkan lapisan *output*), maka setiap *neuron* pada lapisan tersebut (misalkan lapisan tersembunyi) juga harus dihubungkan dengan setiap lapisan pada lapisan lainnya (misalkan lapisan *output*). Ada beberapa arsitektur jaringan syaraf, antara lain [KUS03] :

1. Jaringan dengan lapisan tunggal (*single layer net*)

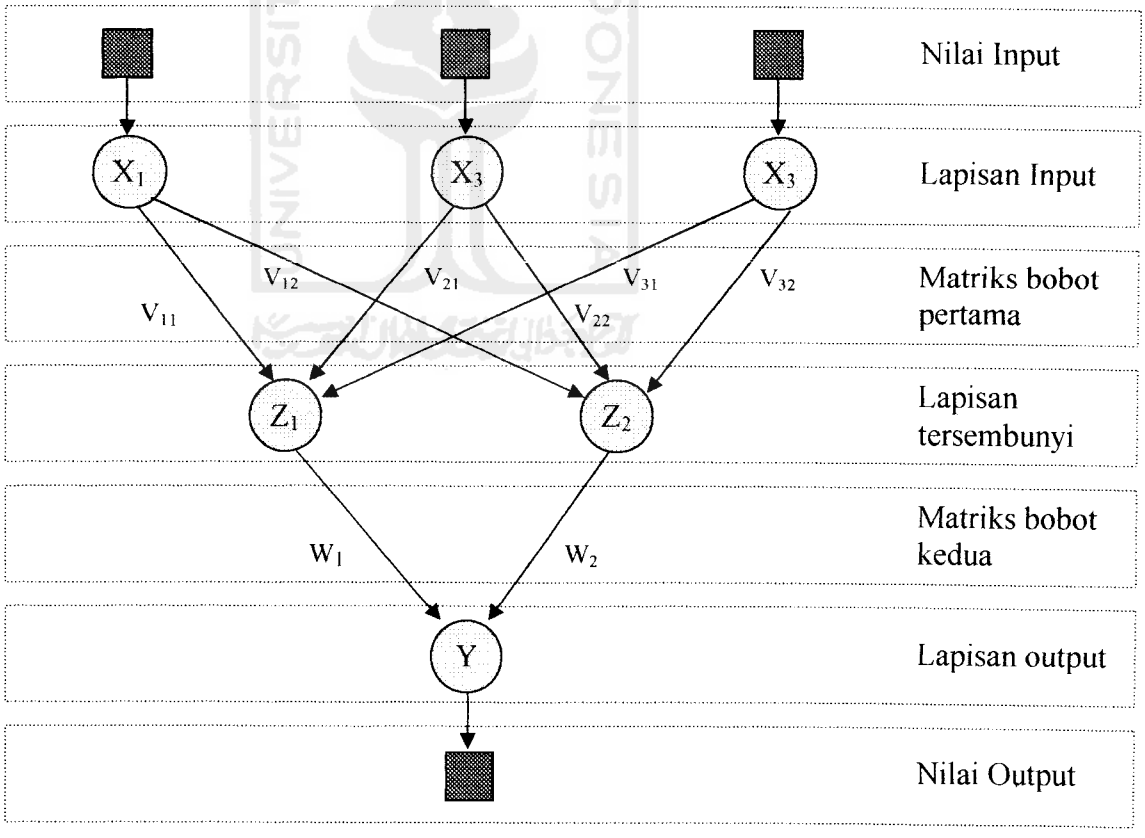
Jaringan dengan lapisan tunggal hanya memiliki satu lapisan dengan bobot-bobot terhubung. Jaringan ini hanya menerima *input* kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi (Gambar 2.2). pada Gambar 2.2 tersebut, lapisan *input* memiliki 3 *neuron*, yaitu X1, X2 dan X3. sedangkan pada lapisan *output* memiliki 2 *neuron* yaitu Y1 dan Y2. N*euron* pada kedua lapisan saling berhubungan. Seberapa besar hubungan antara 2 *neuron* ditentukan oleh bobot yang bersesuaian. Semua unit *input* akan dihubungkan dengan setiap unit *output*.

2. Jaringan dengan banyak lapisan (*multilayer net*)

Jaringan dengan banyak lapisan memiliki 1 atau lebih lapisan yang terletak diantara lapisan *input* dan lapisan *output* (memiliki 1 atau lebih lapisan tersembunyi), seperti terlihat pada Gambar 2.3 umumnya, ada lapisan bobot-bobot yang terletak antara 2 lapisan yang bersebelahan.



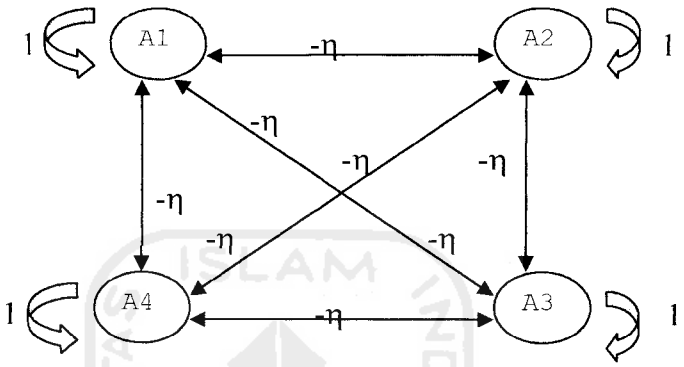
Gambar 2.2 Jaringan syaraf dengan lapisan tunggal



Gambar 2.3 Jaringan syaraf dengan banyak lapisan

3. Jaringan dengan lapisan kompetitif (*competitive layer net*)

Umumnya, hubungan antar *neuron* pada lapisan kompetitif ini tidak diperlihatkan pada diagram arsitektur. Gambar 2.4 menunjukkan salah satu contoh arsitektur jaringan dengan lapisan kompetitif yang memiliki bobot -η.



Gambar 2.4 Jaringan Syaraf Lapisan Kompetitif

2.2.2.2 Fungsi Aktivasi

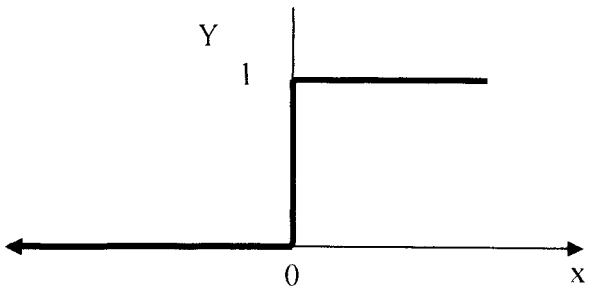
Ada beberapa fungsi aktivasi yang sering digunakan dalam jaringan syaraf tiruan, antara lain [KUS03]:

1. Fungsi Undak Biner (*Hard Limit*)

   Jaringan dengan lapisan tunggal sering menggunakan fungsi undak (*step function*) untuk mengkonversikan *input* dari suatu variabel yang bernilai kontinu ke suatu *output* biner (0 atau1). Fungsi undak biner (*hard limit*) dirumuskan sebagai berikut:

y = 0, jika x<=0

y = 1, jika x > 0



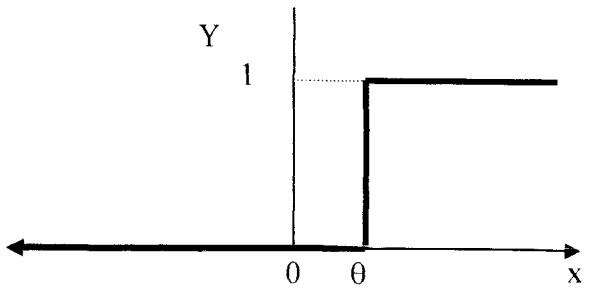
Gambar 2.5 Fungsi aktivasi: Undak Biner *(hard limit)*

2. Fungsi Undak Biner (*Threshold*)

Fungsi undak biner dengan menggunakan nilai ambang sering disebut dengan nama fungsi nilai ambang (*threshold*) atau fungsi Heaviside. Fungsi undak biner dengan nilai ambang (dengan nilai ambang θ) dirumuskan sebagai berikut:

y = 0, jika x <= θ

y = 1, jika x > θ



Gambar 2.6 Fungsi aktivasi: Undak Biner *(threshold)*

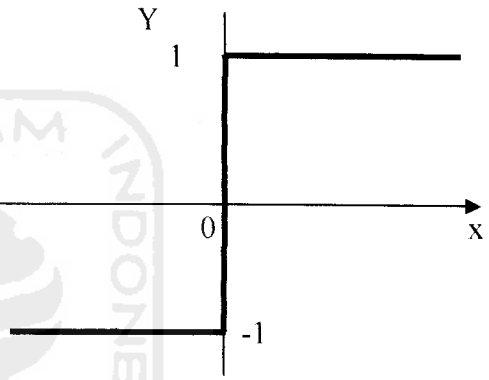
3. Fungsi Bipolar (*Symetric Hard Limit*)

Fungsi bipolar sebenarnya hampir sama dengan fungsi undak biner, hanya saja *output* yang dihasilkan berupa 1, 0 atau -1. Fungsi *Symetric Hard Limit* dirumuskan sebagai berikut:

y = 0, jika x <= 0

y = 1, jika x = 0

y = -1, jika x < 0



Gambar 2.7 Fungsi aktivasi:Bipolar *(Symetric Hard Limit)*

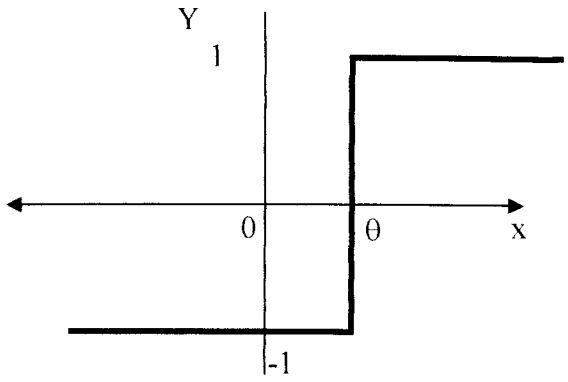
4. Fungsi Bipolar (dengan *threshold*)

Fungsi bipolar sebenarnya hampir sama dengan fungsi undak biner, dengan *threshold* hanya saja *output* yang dihasilkan berupa 1, 0 atau -1. Fungsi bipolar (dengan nilai ambang θ) dirumuskan sebagai berikut:

y = 0, jika x <= θ

y = 1, jika x = θ

y = -1, jika x < θ



Gambar 2.8 Fungsi aktivasi: Bipolar *(threshold)*

2.2.2.3 Proses Pembelajaran

Jaringan syaraf akan mencoba untuk mensimulasikan kemampuan otak manusia untuk belajar. Jaringan syaraf tiruan tersusun atas *neuron* dan *dendrit*. Tidak seperti model biologis, jaringan syaraf tiruan memiliki struktur yang tidak dapat diubah, dibangun oleh sejumlah *neuron*, dan memiliki nilai tertentu yang menunjukkan seberapa besar koneksi antara *neuron* (yang dikenal dengan nama bobot). Perubahan yang terjadi selama proses pembelajaran adalah perubahan nilai bobot. Nilai bobot akan bertambah, jika informasi yang diberikan *neuron* yang bersangkutan tersampaikan, sebaliknya jika informasi tidak disampaikan oleh suatu *neuron* ke *neuron* yang lain, maka nilai bobot yang menghubungkan keduanya akan dikurangi. Pada saat pembelajaran dilakukan pada *input* yang berbeda, maka nilai bobot akan diubah secara dinamis hingga mencapai suatu nilai yang cukup seimbang. Apabila nilai ini telah tercapai mengindikasikan

bahwa tiap-tiap *input* telah berhubungan dengan *output* yang diharapkan [KUS03].

1. Pembelajaran terawasi (*supervised learning*)

   Metode pembelajaran pada jaringan syaraf disebut terawasi jika *output* yang diharapkan telah diketahui sebelumnya. Sebagai contoh jaringan syaraf yang akan digunakan untuk mengenali pasangan pola, misalkan pada operasi AND:

Input		Target
0	0	0
0	1	0
1	0	0
1	1	1

   Pada proses pembelajaran, suatu pola *input* akan diberikan ke satu *neuron* pada lapisan *input*. Pola ini akan dirambatkan di sepanjang jaringan syaraf hingga sampai ke *neuron* pada lapisan *output*. Lapisan *output* ini akan membangkitkan pola *output* yang nantinya akan dicocokkan dengan pola *output* targetnya. Apabila terjadi perbedaan antara pola *output* hasil pembelajaran dengan pola target, maka disini akan muncul error. Apabila nilai error ini masih cukup besar, mengindikasikan bahwa masih perlu dilakukan lebih banyak pembelajaran lagi.

2. Pembelajaran tak terawasi (*unsupervised learning*)

Pada metode pembelajaran tak terawasi ini tidak memerlukan target *output*. Pada metode ini, tidak dapat ditentukan hasil yang seperti apakah yang diharapkan selama proses pembelajaran. Selama proses pembelajaran, nilai bobot disusun dalam suatu range tertentu tergantung pada nilai *input* yang diberikan. Tujuan pembelajaran ini adalah mengelompokkan unit-unit yang hampir sama dalam suatu area tertentu.

2.2.3 Proses Pelatihan Jaringan Syaraf Tiruan

Salah satu karakteristik dari jaringan syaraf tiruan adalah pada jenis proses pelatihan bobot-bobotnya. Pada sub bab ini akan diperkenalkan tentang metode pelatihan jaringan syaraf tiruan menggunakan tipe-tipe; *hebb rule, delta rule* dan *heteroassociative memory*.

2.2.3.1 Hebb Rule

Hebb rule adalah metode pembelajaran yang paling sederhana. Metode ini termasuk proses pembelajaran terawasi. Pada metode ini pembelajaran dilakukan dengan cara memperbaiki nilai bobot sedemikian rupa sehingga jika ada 2 *neuron* yang terhubung, dan keduanya pada kondisi 'hidup' (on) pada saat yang sama, maka bobot antara keduanya dinaikkan. Apabila data direpresentasikan secara bipolar, maka perbaikan bobotnya adalah:

$$w_i \text{ (baru)} = w_i \text{ (lama)} + x_i * y$$

Dengan:

w_i : bobot data *input* ke- i;

x_i : *input* data ke-i;

y : *output* data.

Misalkan digunakan pasangan vektor *input* s dan *output* sebagai pasangan vektor yang akan dilatih. Sedangkan vektor yang hendak digunakan untuk testing adalah vektor x.

Algoritma:

0. Inisialisasi semua bobot:

   $w_{ij} = 0$; dengan i = 1,2,...,n; dan j = 1,2,...,m.

1. Untuk setiap pasangan *input-output* (s-t), lakukan langkah-langkah sebagai berikut:

   a. Set *input* dengan nilai sama dengan vektor *input*:

      $x_i = s_i$; $(i = 1,2,...,n)$

   b. Set *output* dengan nilai sama dengan vektor *output*:

      $y_i = t_j$; $(j = 1,2,...m)$

   c. Perbaiki bobot:

      w_i (baru) = w_i (lama) + $x_i * y_j$

      (i = 1,2,...,n; dan j = 1,2,...,m)

      dengan catatan bahwa nilai bias selalu 1.

2.2.3.2 Delta Rule

Pada *delta rule* akan mengubah bobot yang menghubungkan antara jaringan input ke unit *output* (y_in) dengan nilai target (t). Hal ini dilakukan untuk meminimalkan *error* selama pelatihan pola. *Delta rule* untuk memperbaiki bobot ke_i (untuk setiap pola) adalah:

$$\Delta w_i = \alpha(t - y\_in)*x_i$$

dengan:

x = vekor input.

y_in = input jaringan ke unit output Y.

$$y\_in = \sum_{i=1}^{n} x_i * w_i$$

t = target (output)

Nilai w baru diperoleh dari nilai w lama ditambahkan dengan Δw,

$$\Delta w_i = w_i + \Delta w_i$$

2.2.3.3 Heteroassociative Memory

Jaringan syaraf *associative memory* adalah jaringan yang bobot-bobotnya ditentukan sedemikian rupa sehingga jaringan tersebut dapat menyimpan kumpulan pengelompokan pola. Masing-masing kelompok merupakan pasangan vektor (s(p),t(p)) dengan p = 1,2,...,P. Tiap-tiap vektor s(p) memiliki n komponen, dan tiap-tiap t(p) memiliki m komponen. Bobot-bobot tersebut dapat ditentukan dengan menggunakan *hebb rule* atau *delta rule*. Jaringan ini nanti akhirnya akan mendapatkan vektor *output* yang sesuai dengan vektor

*input*nya (x) yang merupakan salah satu vektor s(p) atau merupakan vektor lain di luar s(p).

Algoritma:

0. Inisialisasi bobot dengan menggunakan Hebb rule atau delta rule.
1. Untuk setiap vektor *input*, kerjakan:
   a. Set *input* dengan nilai sama dengan vektor *input*
   b. Hitung *input* jaringan ke unit *output*:

$$y\_in_j = \sum_i x_x * w_{ij}$$

   c. Tentukan aktivasi dari setiap unit *output*:

$$y_j = \begin{cases} 1; & y\_in_j > 0 \\ 0; & y\_in_j = 0 \\ -1; & y\_in_j < 0 \end{cases}$$

(untuk target bipolar)

BAB III

ANALISIS KEBUTUHAN PERANGKAT LUNAK

3.1 Metode Analisis

Metode analisis yang dipakai dalam pengembangan perangkat lunak ini adalah metode analisis dengan pendekatan terstruktur. Model matematis yang dibuat diimplementasikan dalam perangkat lunak dengan masukan berupa sekumpulan bilangan (vektor masukan) dan keluaran berupa pola karakter 2 dimensi.

Algoritma pelatihan dan pengujian dalam perangkat lunak ini dinyatakan dengan bagan alir (*flowchart*) yang digunakan untuk mengembangkan dan menggambarkan langkah-langkah algoritma dalam menentukan proses perhitungan pelatihan dan pengujian.

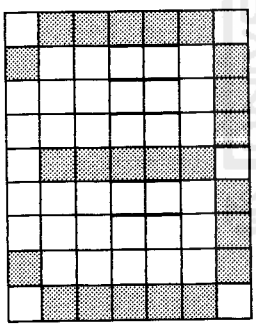
3.2 Hasil Analisis

Setelah dianalisis maka dapat diketahui yang menjadi masukan dan keluaran sistem. Sistem ini menggunakan sistem operasi Windows. Sedangkan struktur data yang akan dipakai adalah berbasis file. Semua data sistem akan disimpan dalam sebuah file berekstensi .HTR.

3.2.1 Masukan

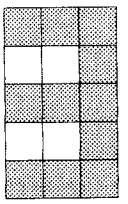
Masukan program berupa pola karakter 2 dimensi yang dialihkan menjadi sekumpulan vektor masukan. Pola tersebut direpresentasikan oleh pola karakter 2 dimensi berbentuk matrik dengan beberapa komponen. Jumlah maksimal komponen matrik yang diimplementasikan pada perangkat lunak ini adalah 400 komponen (20 x 20) dan minimal 15 komponen (5 x 3).

Sebagai contoh vektor dengan 63 komponen yang merepresentasikan pola karakter 2 dimensi dengan ukuran matrik 9x7 (Gambar3.1). Sedangkan target dengan 15 komponen yang merepresentasikan pola karekter 2 dimensi dengan ukuran matrik 5x3(Gambar 3.2).



$$\begin{pmatrix} -1 & 1 & 1 & 1 & 1 & 1 & -1 \\ 1 & -1 & -1 & -1 & -1 & -1 & 1 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 \\ -1 & 1 & 1 & 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 \\ 1 & -1 & -1 & -1 & -1 & -1 & 1 \\ -1 & 1 & 1 & 1 & 1 & 1 & -1 \end{pmatrix}$$

Gambar 3.1 Angka 3, sebagai matriks 9x7



$$\begin{pmatrix} -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & 1 & 1 \\ -1 & -1 & -1 \end{pmatrix}$$

Gambar 3.2 Angka 3, sebagai matriks 5x3

Apabila komponen vektor bernilai 1, menandakan bahwa kotak yang diwakilinya berwarna hitam, sedangkan komponen vektor berinilai -1 menandakan bahwa kotak yang diwakilinya berwarna putih.

Vektor yang bersesuaian dengan Gambar 3.1 adalah:

-1 1 1 1 1 1 -1 1 -1 -1 -1 -1 -1 1 -1 -1 -1 -1 -1 -1 1 -1 -1 -1 -1 -1 -1

1 -1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 1 -1 -1 -1 -1 -1 -1 1 1 -1 -1 -1 -1

-1 1 -1 1 1 1 1 1 -1

Vektor yang bersesuaian dengan Gambar 3.2 adalah:

1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 1 1

3.2.2 Keluaran

Hasil akhir atau keluaran dari sistem ini adalah nilai dari proses pengujian berdasarkan nilai input yang akan direpresentasikan kembali ke dalam pola karakter 2 dimensi.

3.2.3 Fungsi yang dibutuhkan

Sesuai dengan sistem yang diterapkan dalam kasus ini yaitu sistem jaringan syaraf tiruan, maka fungsi-fungsi yang dibutuhkan dalam perangkat lunak ini adalah :

1. Fungsi Input Data

   Digunakan untuk memasukan data pola karakter 2 dimensi yang akan digunakan dalam proses pelatihan dan proses pengujian.

2. Fungsi Pelatihan

   Fungsi ini digunakan untuk melatih jaringan syaraf dalam rangka mendapatkan bobot-bobot yang sama.

3. Fungsi Pengujian

   Pada fungsi ini digunakan untuk menguji sistem berdasarkan nilai bobot akhir dari proses pelatihan.

3.2.4 Antarmuka Perangkat Lunak

Antarmuka yang diinginkan adalah berbasis grafik dengan harapan akan memudahkan *user* dalam menggunakannya. Untuk pembuatannya, digunakan Bahasa Pemrograman Borland Delphi 7.0 yang memiliki kemudahan pemakaian dan kelengkapan fungsi-fungsi.

BAB IV

PERANCANGAN PERANGKAT LUNAK

4.1 Metode Perancangan

Metode perancangan yang digunakan untuk membuat perangkat lunak ini adalah metode perancangan terstuktur (*Structured Design Method)* menggunakan bagan alir sistem.

4.2 Hasil Perancangan

Hasil pada tahap perancangan berkaitan erat dengan hasil tahap analisis, karena pada tahap analisis telah ditentukan fungsi-fungsi dan metode-metode yang digunakan, serta antarmuka yang diharapkan. Dari hasil tahap analisis tersebut didapat suatu gambaran tentang sistem aplikasi jaringan syaraf tiruan untuk pengenalan pola.

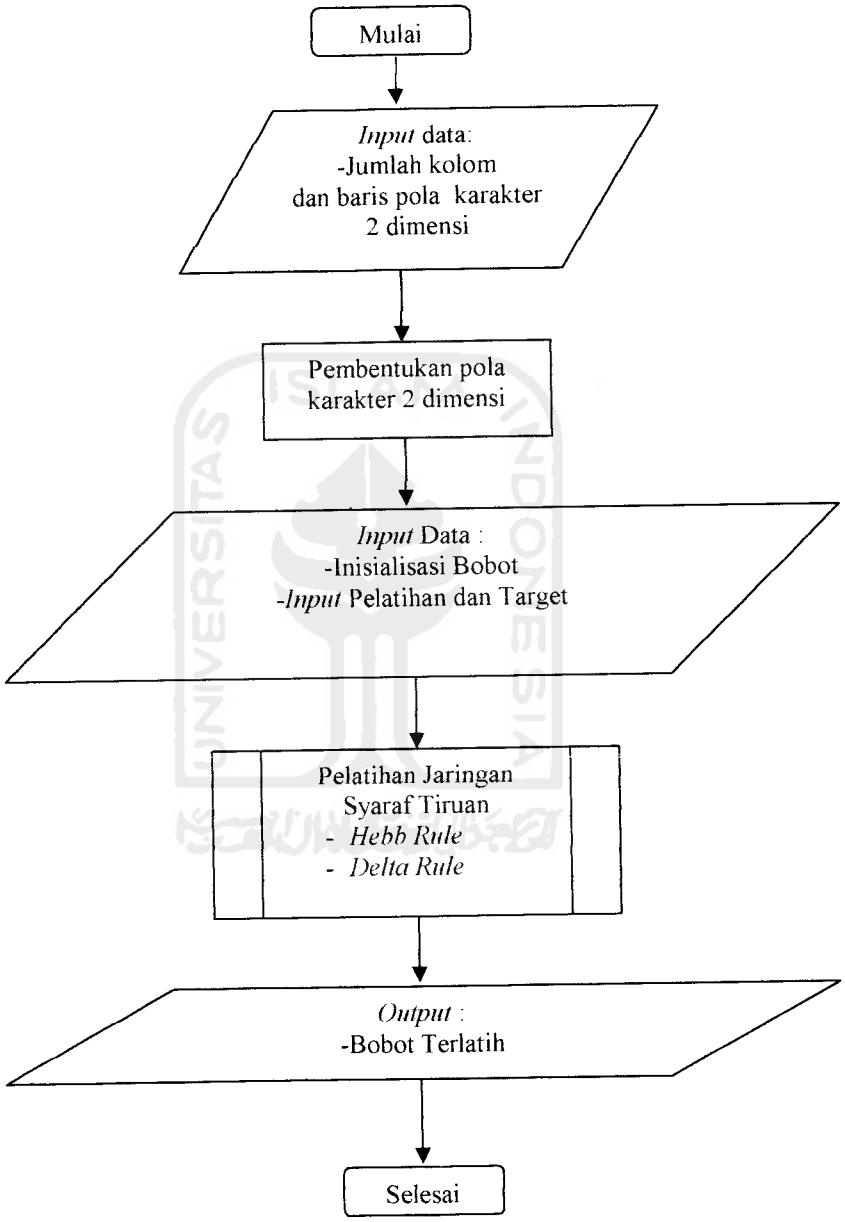
4.2.1 Perancangan Diagram Alir Sistem

Bagan alir adalah suatu bagan yang berisi simbol-simbol grafis yang manunjukkan arah aliran kegiatan dan data yang terjadi dalam sebuah program. Secara umum, bagan alir sistem (*system flowchart*) dan bagan alir program (*program flowchart*) merupakan 2 sistem yang saling berpengaruh untuk menjelaskan suatu perancangan agar perancangan tersebut mudah untuk dipahami.

Pada sub bab ini, sistem akan digambarkan sebagai sebuah diagram alir (*flow chart*) secara keseluruhan. Kemudian akan dibuat diagram alir dari proses

perhitungan pelatihan sistem dan proses perhitungan pengujian.

Gambar 4.1 merupakan gambar diagram alir sistem perangkat lunak untuk pelatihan jaringan syaraf tiruan.

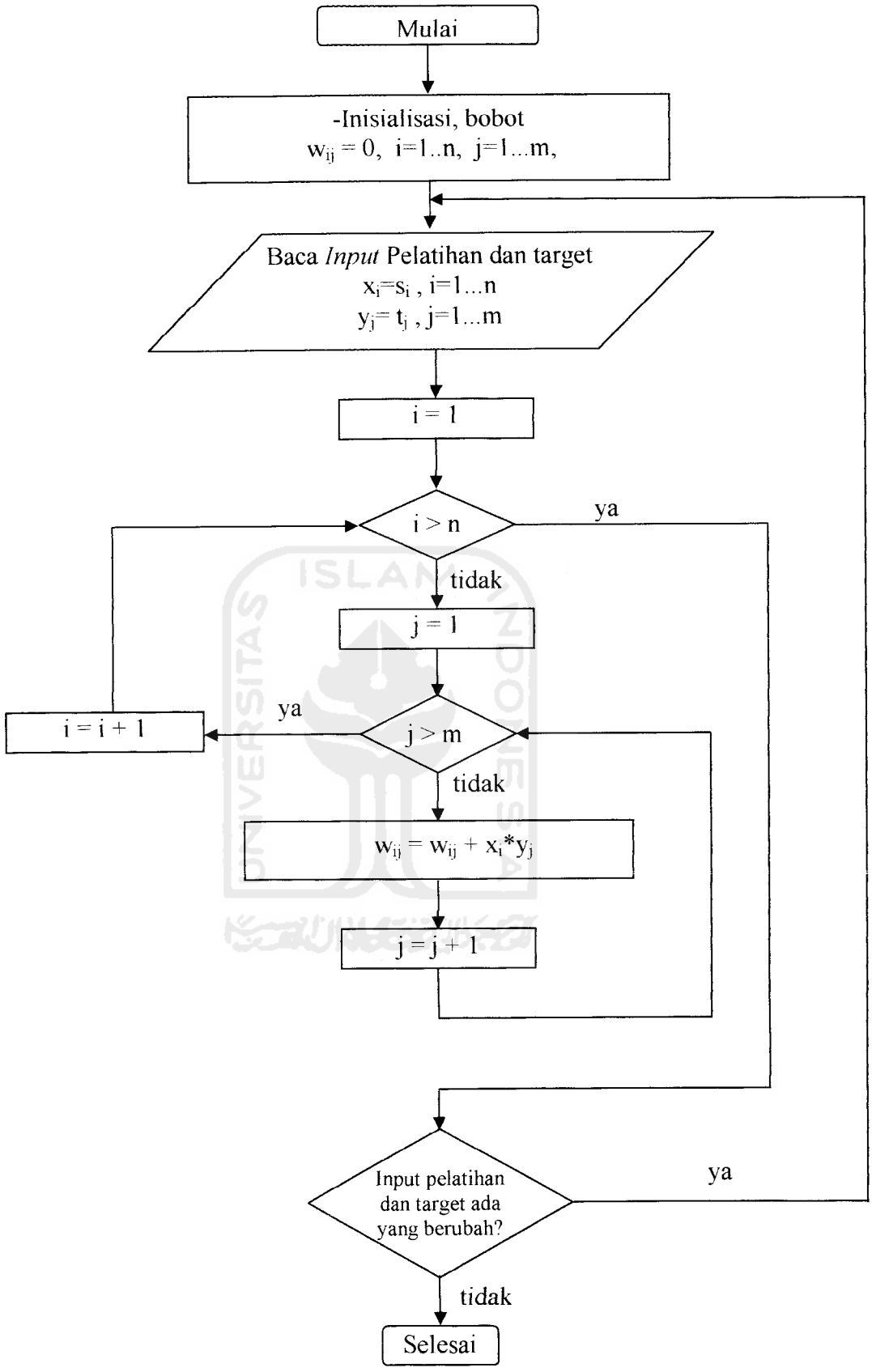


Gambar 4.1 Diagram Alir Perangkat Lunak Pelatihan Jaringan Syaraf Tiruan

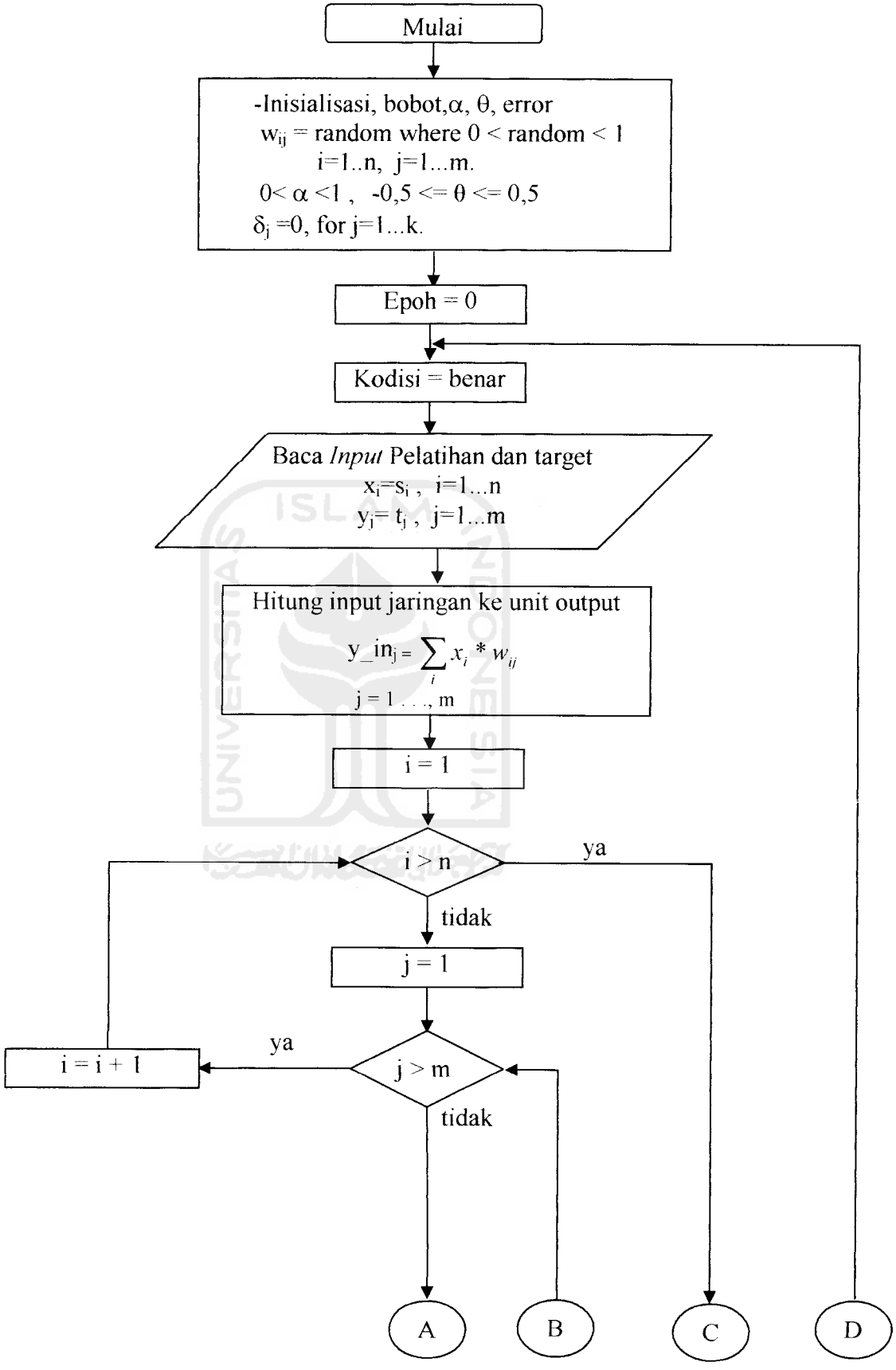
Dari diagram alir, pada proses pembentukan pola karakter 2 dimensi, perangkat lunak akan membentuk kotak-kotak yang tersusun berbentuk matrik yang merepresentasikan bentuk pola sebagi *input* dan target.

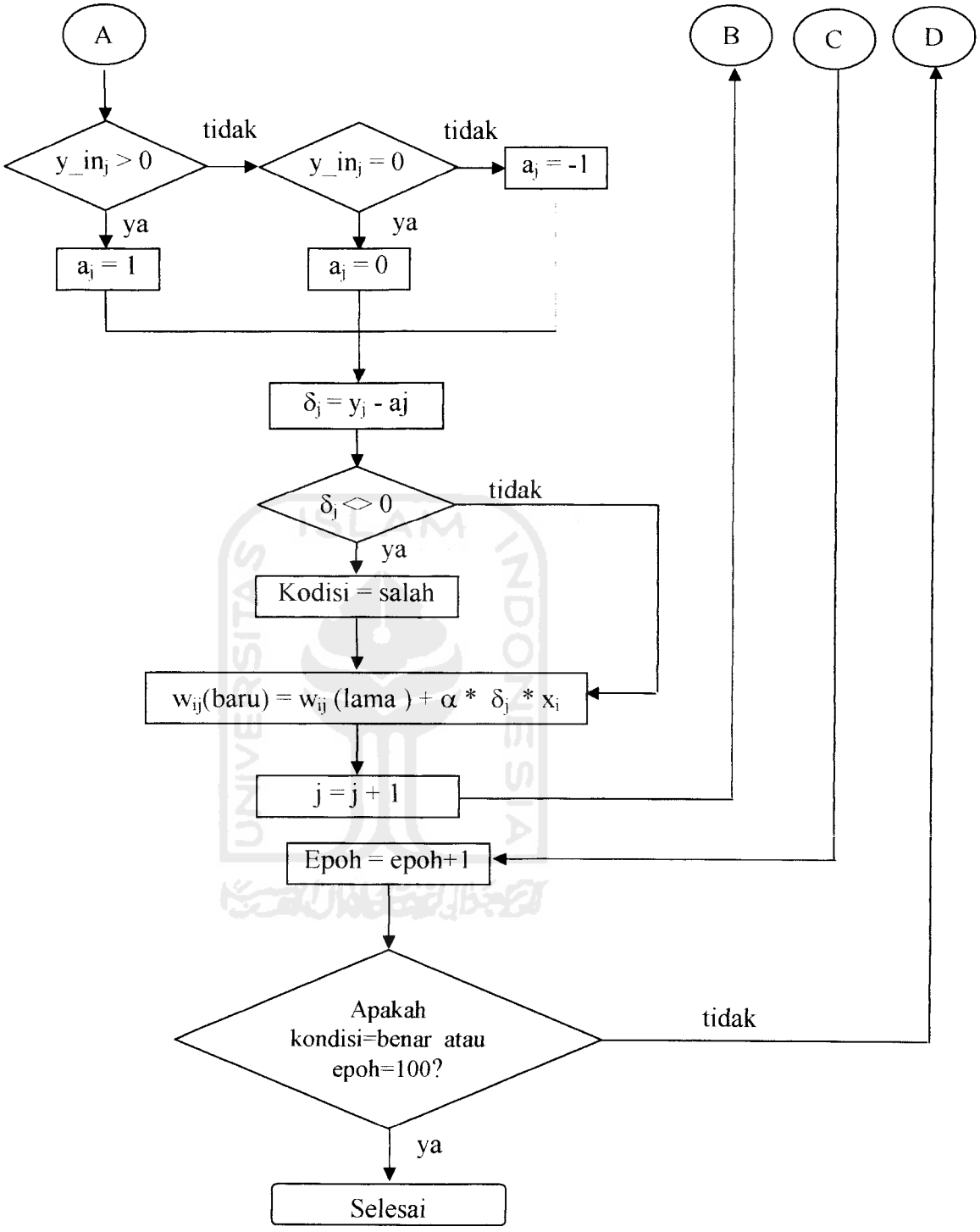
Pada tahap pelatihan jaringan syaraf tiruan, tahap pertama yang dilakukan adalah inisialisasi bobot kemudian perangkat lunak akan membaca *input* pelatihan dan target. Tahap selanjutnya jaringan akan melatih bentuk-bentuk pola yang di*input*kan dari kotak-kotak yang mewakili bentuk pola tersebut. Apabila kotak ditekan maka warna akan berubah menjadi hitam dan nilainya 1, sebaliknya apabila kotak tidak ditekan warnanya putih maka nilainya -1, yang artinya kotak tersebut tidak mewakili dari bentuk pola yang di*input*kan. Dari bentuk pola tersebut dijadikan sekumpulan bilangan yang akan dilatih. Perubahan bobot dari proses pelatihan selanjutnya dapat digunakan dalam proses pengujian.

Berikut ini bagan alir proses pelatihan dan proses pengujian:

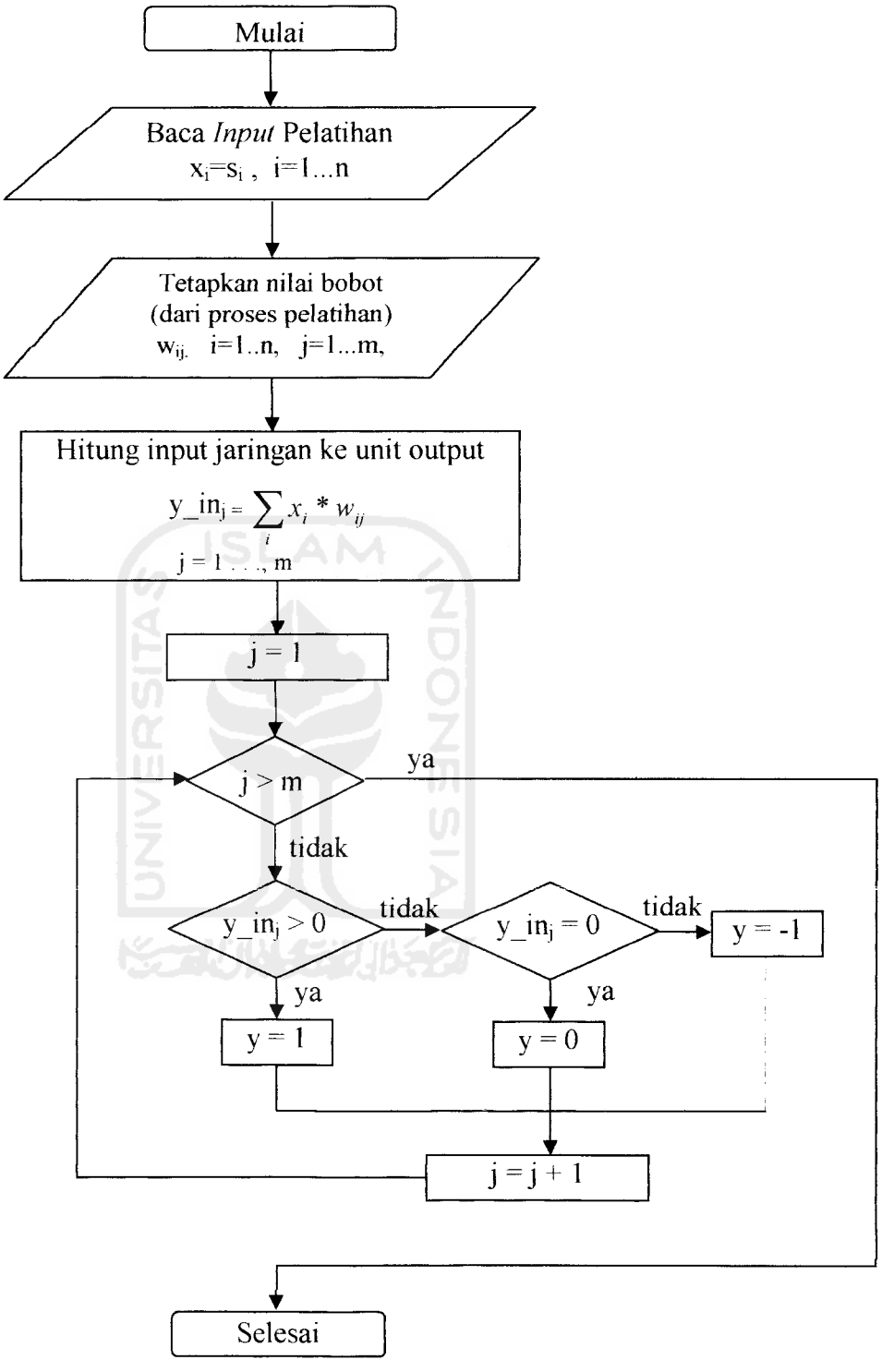


Gambar 4.2 Diagram Alir Proses Pelatihan Metode *Hebb Rule*


Mulai
-Inisialisasi, bobot,α, θ, error
w_{ij} = random where 0 < random < 1
i=1..n, j=1...m.
0< α <1 , -0,5 <= θ <= 0,5
δ_j =0, for j=1...k.
Epoh = 0
Kodisi = benar
Baca *Input* Pelatihan dan target
$x_i=s_i$, i=1...n
$y_j= t_j$, j=1...m
Hitung input jaringan ke unit output
$y\_in_j = \sum_i x_i * w_{ij}$
j = 1 . . ., m
i = 1
i > n
ya
tidak
j = 1
j > m
ya
i = i + 1
tidak
A
B
C
D



Gambar 4.3 Diagram Alir Proses Pelatihan Metode *Delta Rule*



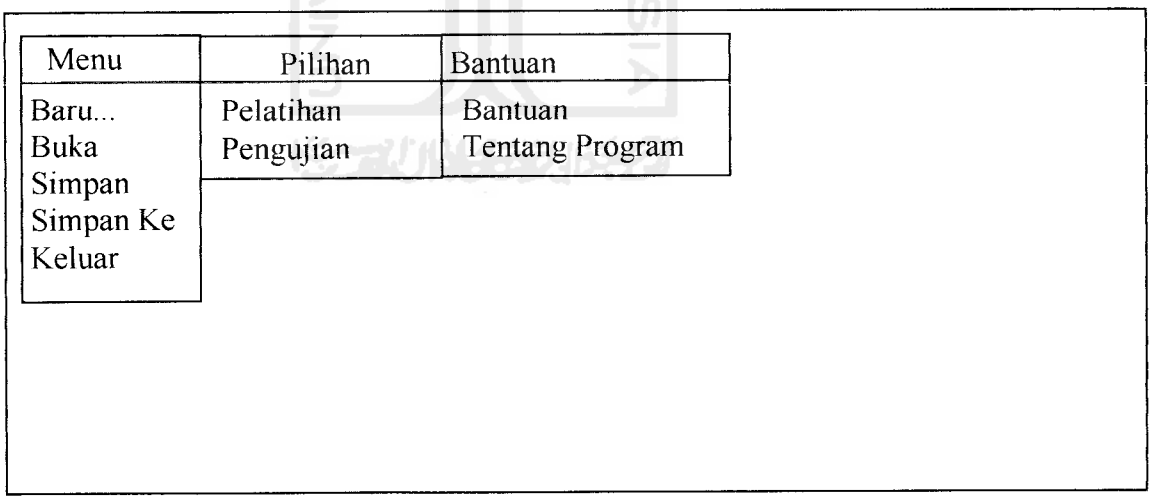
Gambar 4.4 Diagram Alir Proses Pengujian

4.2.2 Perancangan Antarmuka

Rancangan antarmuka dari perangkat lunak aplikasi jaringan syaraf tiruan ini menggunakan perancangan model grafis. Perancangan yang dibuat terdiri dari antarmuka menu utama, antarmuka ukuran pola, antarmuka pelatihan, antarmuka pengujian dan lain-lain.

4.2.2.1 Rancangan Antarmuka Menu Utama

Antarmuka ini digunakan untuk memilih menu-menu yang terdapat di dalam sistem perangkat lunak dari aplikasi jaringan syaraf tiruan. Gambar 4.5 mengilustrasikan perancangan antarmuka menu utama.

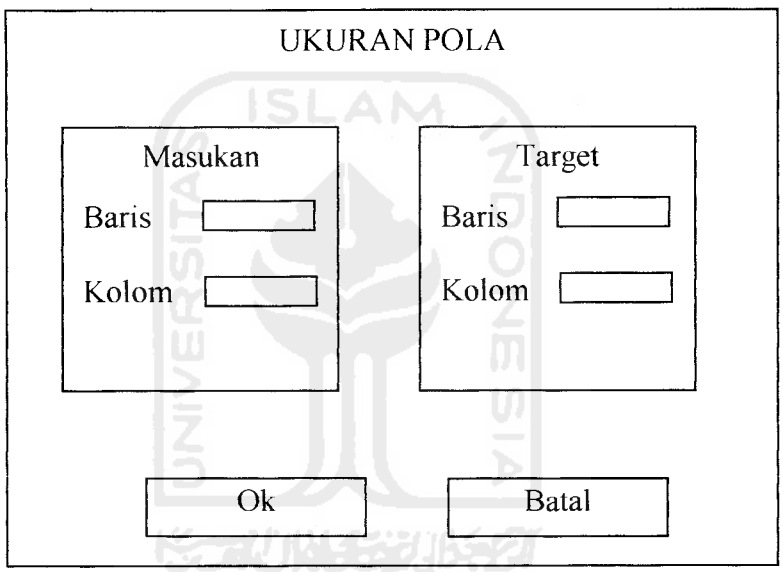


Gambar 4.5 Rancangan Antarmuka Menu Utama

4.2.2.2 Rancangan Antarmuka Ukuran Pola

Antarmuka ini digunakan untuk menentukan besar kecil pola karakter 2 dimensi sebagai *input* dan target.

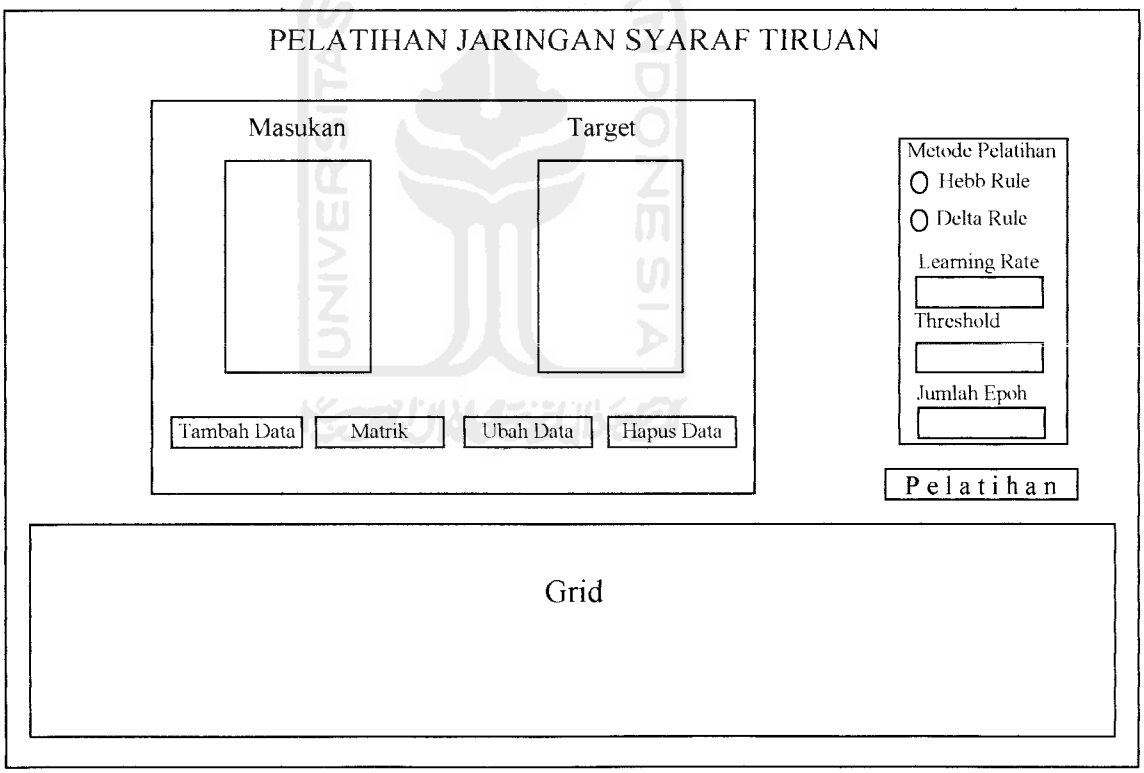
Gambar 4.6 adalah ilustrasi dari rancangan antarmuka ukuran pola.



Gambar 4.6 Rancangan Antarmuka Ukuran Pola

4.2.2.3 Rancangan Antarmuka Pelatihan

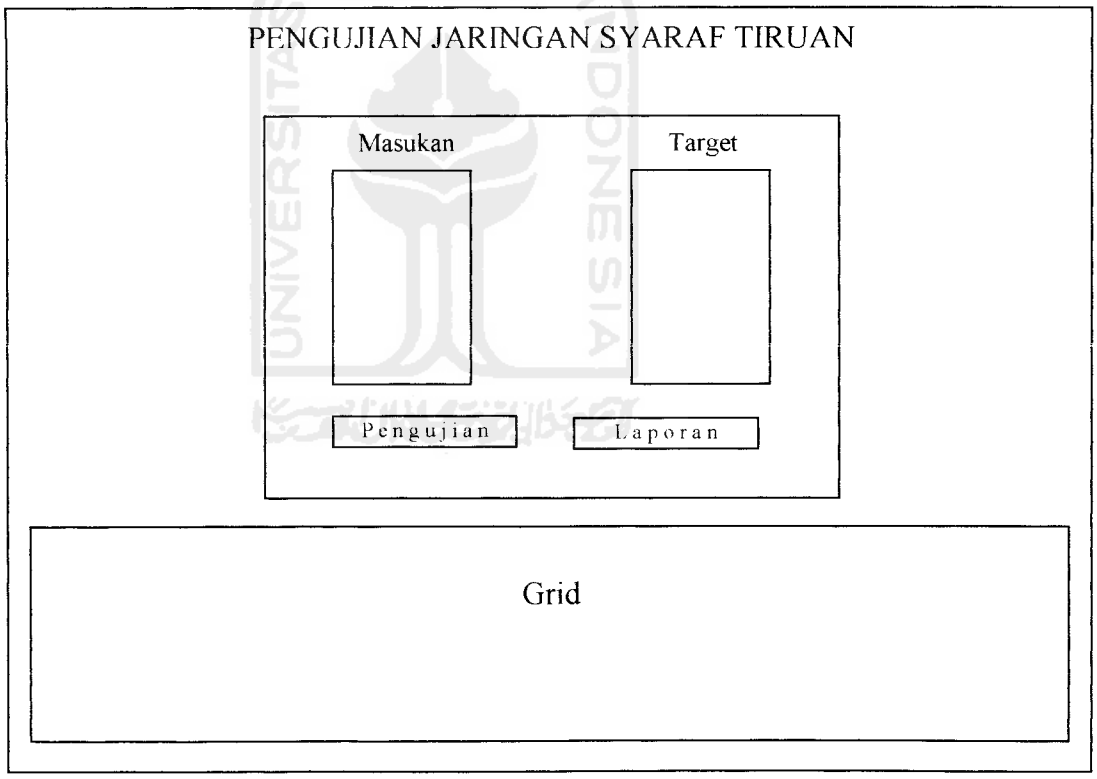
Pada antarmuka ini digunakan untuk memasukkan data pola karakter 2 dimensi beserta targetnya serta untuk melatih nilai bobot pola tersebut yang dibentuk oleh *user*. Selain itu metode dari proses pelatihan dan beberapa parameter pelatihan seperti *learning rate* dan nilai ambang (*threshold)* disertakan disini. Pada antarmuka ini juga terdapat tabel yang menampilkan data-data pola yang disimpan. Gambar 4.7 adalah ilustrasi dari rancangan antarmuka pelatihan.



Gambar 4.7 Rancangan Antarmuka Pelatihan

4.2.2.4 Rancangan Antarmuka Pengujian

Pada antarmuka pengujian digunakan untuk memasukkan pola karakter 2 dimensi sebagai *input* (masukan) yang akan diuji dalam sistem jaringan syaraf tiruan berdasarkan bobot yang dilatih dari proses pelatihan. Pada antarmuka ini juga menampilkan hasil atau target dari proses pengujian pola tersebut. Gambar 4.8 adalah ilustrasi dari rancangan antarmuka pengujian.

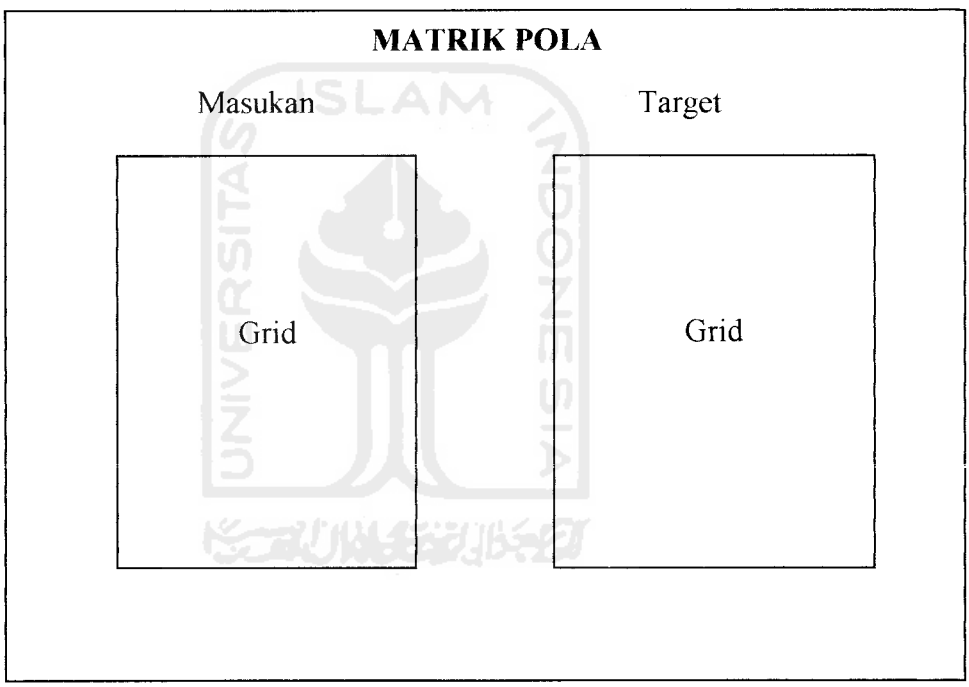


Gambar 4.8 Rancangan Antarmuka Pengujian

4.2.2.5 Rancangan Antarmuka Matrik Pola

Rancangan antarmuka matrik pola digunakan untuk menampilkan nilai dari masukan dan target pola karakter 2 dimensi yang direpresentasikan sebagai matrik. Nilai yang mewakili pola tersebut bernilai 1 atau -1.

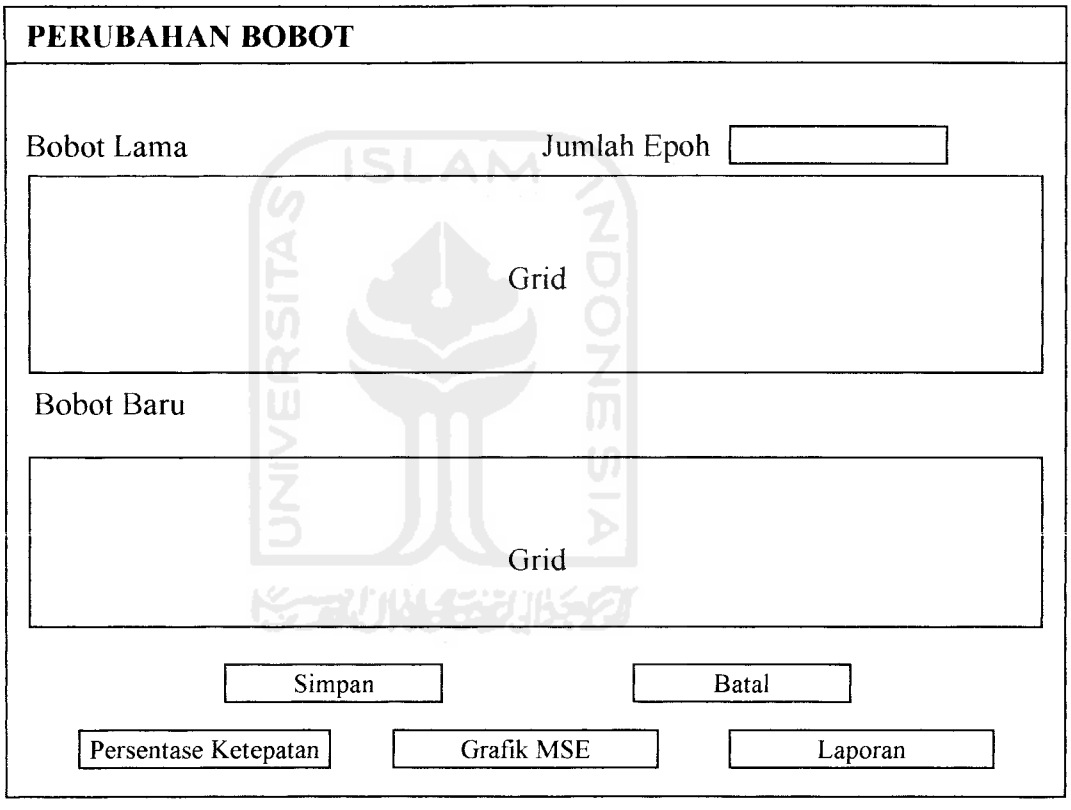
Gambar 4.9 adalah ilustrasi dari rancangan antarmuka matrik pola.



Gambar 4.9 Rancangan Antarmuka Matrik Pola

4.2.2.6 Rancangan Antarmuka Perubahan Bobot

Pada rancangan antarmuka perubahan bobot menampilkan nilai-nilai bobot sebelum dilakukan proses pelatihan dan setelah dilakukan proses pelatihan. Gambar 4.10 adalah ilustrasi rancangan antarmuka perubahan bobot.

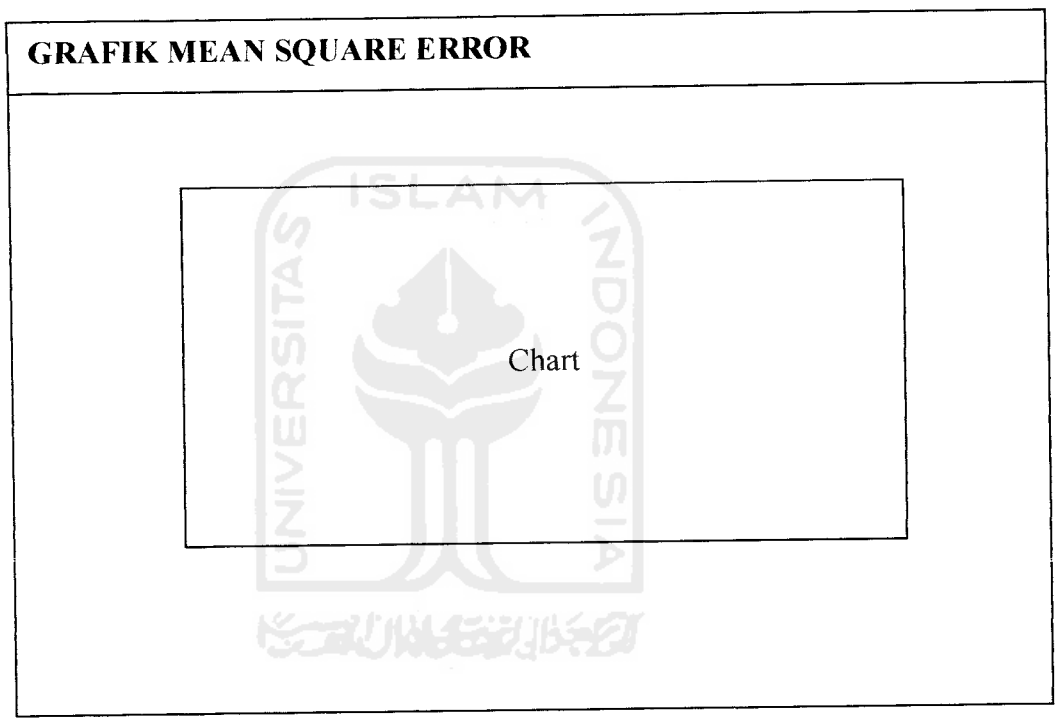


Gambar 4.10 Rancangan Antarmuka Perubahan Bobot

4.2.2.7 Rancangan Antarmuka Grafik *Mean Square Error*

Pada rancangan antarmuka ini menampilkan grafik dari nilai MSE (*Mean Square Error*) pada setiap *epoch*.

Gambar 4.11 adalah gambar rancangan antarmuka grafik *mean square error*.

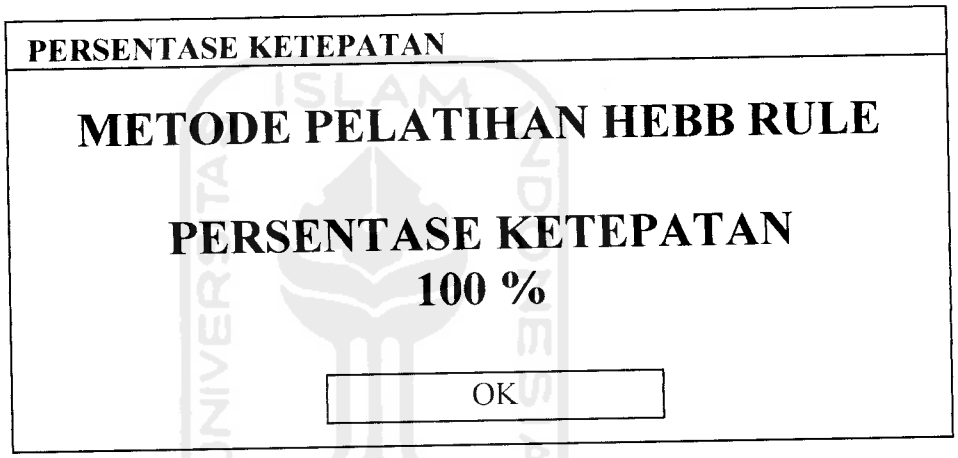


Gambar 4.11 Rancangan Antarmuka Grafik *Mean Square Error*.

4.2.2.8 Rancangan Antarmuka Persentase Ketepatan

Form ini digunakan untuk menampilkan persentase ketepatan pada setiap data pola karakter 2 dimensi yang dilatih, baik pada metode *hebb rule* maupun metode *delta rule*

Gambar 4.12 adalah gambar rancangan antarmuka persentase ketepatan

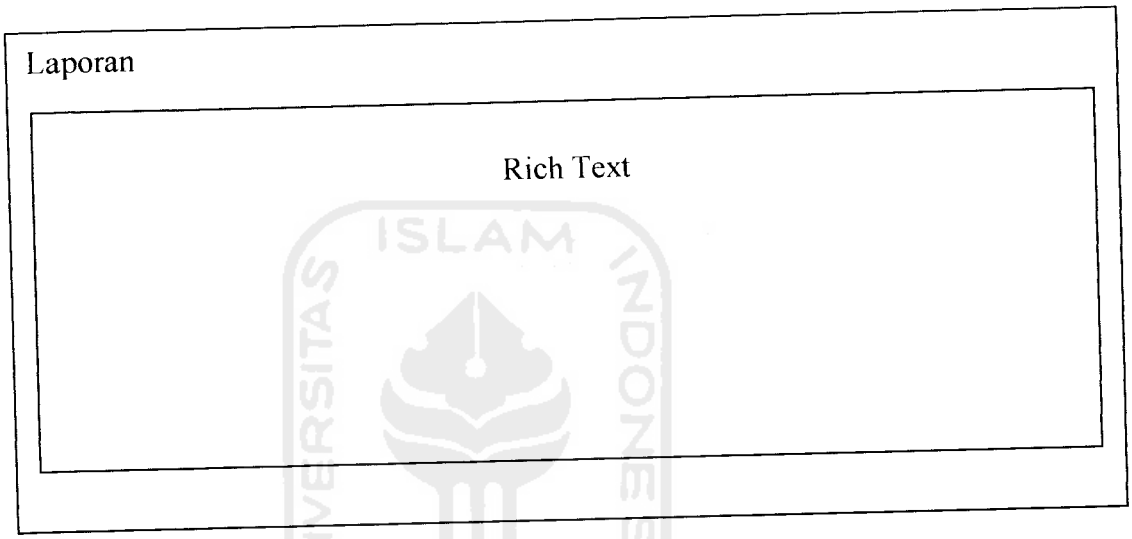


Gambar 4.12 Rancangan Antarmuka Persentase Ketepatan

4.2.2.9 Rancangan Antarmuka Laporan

Form Laporan digunakan untuk melihat perhitungan yang terjadi pada saat pelatihan dan pengujian.

Gambar 4.13 adalah gambar rancangan antarmuka laporan.



Gambar 4.13 Rancangan Antarmuka Laporan

4.2.2.10 Rancangan Antarmuka Bantuan

Form ini berisi informasi mengenai aplikasi jaringan syaraf tiruan untuk pengenalan pola secara praktis.

Bab V

Implementasi Perangkat Lunak

5.1 Batasan Implememtasi

Pada Bagian ini akan dijelaskan apa saja yang menjadi batasan implementasi perangkat lunak, antara lain bahasa pemrograman yang dipakai, lingkungan pengembangan, asumsi-asumsi yang ditemui dan dibuat selama pengembangan perangkat lunak.

5.1.1 Pemilihan Bahasa Pemrograman

Sistem perangkat lunak yang akan dibuat adalah sebuah perangkat lunak yang berisi banyak perhitungan dan peraturan logika. Maka dibutuhkan sebuah bahasa pemrograman yang terstruktur, handal, praktis, mudah digunakan dan juga mendukung batasan sistem operasi secara umum. Borland Delphi 7.0 adalah salah satu bahasa pemrograman yang memiliki semua persyaratan yang dibutuhkan dalam proses pembuatan perangkat lunak ini. Sehingga penggunaan bahasa pemrograman ini dirasakan sangat tepat.

5.1.2 Asumsi-asumsi yang dipakai

Berikut ini adalah beberapa asumsi yang ada dalam sistem ini :

1. Pola karakter 2 dimensi direpresentasikan ke dalam bentuk matrik m x n. Setiap perubahan besar pola karakter akan membuat bobot-bobot yang

akan diinisialisasi ulang. Batas maksimal pola karakter 2 dimensi adalah 20 x 20 dan minimal 5 x 3.

2. Metode pelatihan yang digunakan adalah metode *hebb rule* dan *delta rule*.
3. Pada metode pelatihan *delta rule* nilai *learning rate* dan *threshold* digunakan, dengan batasan nilai *learning rate* ($0 < \alpha < 1$) dan nilai *threshold* ($-0,5 <= \theta <= 0,5$). Batas maksimal *epoch* dari setiap pelatihan ditetapkan 100 *epoch*.
4. Pada metode pelatihan *delta rule* sistem akan menghentikan pelatihan jika nilai *error* adalah 0, meskipun jumlah iterasi yang ditentukan belum selesai.
5. Batas maksimal data yang akan dilatih adalah 250 data pola karakter 2 dimensi.

5.2 Implementasi

Berikut ini akan dijelaskan implementasi dari sistem perangkat lunak untuk pengenalan pola.

5.2.1 Implementasi Fungsi dan Prosedur

Pada tahap pemrograman perangkat lunak ini diperlukan prosedur-prosedur yang digunakan untuk melakukan proses perhitungan. Berikut ini adalah beberapa prosedur yang digunakan pada sistem perangkat lunak ini.

5.2.1.1 Pembentukan Pola Karakter 2 Dimensi

Prosedur ini digunakan untuk membentuk pola karakter 2 dimensi berbentuk kotak-kotak atau tombol yang tersusun dalam bentuk matrik sebagai representasi pola masukan dan target yang akan dilakukan pelatihan. Kotak-kotak tersebut merupakan object (Ttombol) yang diturunkan dari kelas Tpanel dalam delphi. Kumpulan bentuk pola karakter yang akan dilakukan proses pelatihan dapat disimpan dalam sebuah data yang berekstensi .HTR. Setiap data tersebut memuat nilai bobot, nilai input dan target, metode pelatihan yang digunakan, nilai *learning rate* dan *threshold.*

5.2.1.1.1 Pembentukan Objek Ttombol

```
constructor TTombol.create(AOwner:TComponent);
begin
 inherited create(AOwner);
 Width:=15;
 Height:=15;
 Color:=clActiveCaption;
 BevelWidth:=3;
 BevelInner:=bvNone;
 BevelOuter:=bvRaised;
 OnMouseDown:=tekan;
 OnMouseUp:=lepas;
end;

procedure TTombol.Tekan(Sender: TObject; Button:
TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  if Button<>mbleft then exit;
  if Self.Color=clblack then
     Self.Color:=clActiveCaption
  else Self.Color:=clBlack;
  Self.Visible:=False;
end;
```

```
procedure TTombol.Lepas(Sender: TObject; Button:
TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  if Button<>mbleft then exit;
  self.Visible:=true;
end;
```

5.2.1.1.2 Deklarasi Variabel

```
data = record
 bobot,bobot1 : record
   nilai : array[1..400,1..400] of Real;
   klm,brs : integer;
 end;
 metode : string[18];
 alfa,teta : Real;
 nama : string[255];
 jmldata : integer;
 polainp,polatrg : record
   klm,brs: integer;
 end;
 input,target : array[1..250] of record
   nilai : array[1..20,1..20] of real;
 end;
end;

public
  nodat,wbrs(jumlah baris bobot),
  wklm(jumlah kolom bobot),maxepoh:integer;
  ip,tr,sml: array[1..400]of real;
  hasil:array[1..20,1..20]of Real;
  err: array[1..250] of record
    nilai: array[1..400] of real;
  end;
  tbl1(input),tbl2(target):record
    kotak:array[1..20,1..20] of ttombol;
    atas,kiri: integer;
  end;
    data1:data;
    data2:file of data;
```

5.2.1.1.3 Prosedur Pembetukan Pola Karakter 2 Dimensi

```
for i:=1 to polainp.brs do
  begin
     tbl1.atas:=tbl1.atas+15;
     l:=tbl1.kiri;
        for j:=1 to polainp.klm do
           begin
             l:=l+15;
             tbl1.kotak[i,j]:=Ttombol.Create(Self);
             tbl1.kotak[i,j].Parent := (letak tombol);
             tbl1.kotak[i,j].Left   := l;
             tbl1.kotak[i,j].Top    := tbl1.atas;
           end;
  end;

  for i:=1 to polatrg.brs do
  begin
     tbl2.atas:=tbl2.atas+15;
     m:=tbl2.kiri;
        for j:=1 to polatrg.klm do
           begin
             m:=m+15;
             tbl2.kotak[i,j]:=Ttombol.create(Self);
             tbl2.kotak[i,j].Parent := (letak tombol);
             tbl2.kotak[i,j].Left   := m;
             tbl2.kotak[i,j].Top    := tbl2.atas;
           end;
  end;
```

5.2.1.2 Prosedur Input dan Target

Prosedur ini digunakan untuk memasukkan nilai *input* (masukan) dan nilai target pada proses pelatihan. Pola karakter yang mewakili *input* atau target bernilai 1 atau -1. Apabila kotak atau tombol yang mewakili pola karakter berwarna hitam maka nilainya 1, sedangkan berwarna biru nilainya -1.

5.2.1.2.1 Prosedur Pengisian Nilai *Input* dan Target

```
data.jmldata:=jmldata+1;
for i:=1 to polainp.brs do
     for j:=1 to polainp.klm do
       begin
         if tbl1.kotak[i,j].Color=clBlack then
           input[jmldata].nilai[i,j]:=1
         else input[jmldata].nilai[i,j]:=-1;
       end;
        for m:=1 to polatrg.brs do
           for n:=1 to polatrg.klm do
             begin
                if tbl2.kotak[m,n].Color=clBlack then
                   target[jmldata].nilai[m,n]:=1
                else target[jmldata].nilai[m,n]:=-1;
             end;
```

5.2.1.3 Heteroassociative Memory

Pada *Heteroassociative Memory* algoritma pelatihan yang digunakan adalah *Hebb Rule* dan *Delta Rule*.

5.2.1.3.1 Algoritma Pelatihan *Hebb Rule*

```
for a:=1 to (jumlah data) do
 begin
  k:=0;
  for i:=1 to polainp.brs do
   for j:=1 to polainp.klm do
   begin
     k:=k+1;
     l:=0;
     for m:=1 to polatrg.brs do
      for n:=1 to polatrg.klm do
      begin
        l:=l+1;
        bobot1.nilai[k,l]:=bobot1.nilai[k,l]+
        input[a].nilai[i,j]*target[a].nilai[m,n];
      end;
   end;
 end;
```

5.2.1.3.2 Algoritma Pelatihan *Delta Rule*

```
begin
 repeat
  enol:=true;
  epoh:=epoh+1;
  for h:=1 to (jumlah data) do
    begin
     k:=0;
     for i:=1 to polainp.brs do
      for j:=1 to polainp.klm do
       begin
        k:=k+1;
        ip[k]:=input[h].nilai[i,j];
       end;
     k:=0;
     for i:=1 to polatrg.brs do
      for j:=1 to polatrg.klm do
       begin
        k:=k+1;
        tr[k]:=target[h].nilai[i,j];
       end;
     for i:=1 to wklm do
       begin
         hsl:=0;
         for j:=1 to wbrs do
           begin
             hsl:=hsl+ip[j]*bobot1.nilai[j,i];
           end;
          if hsl>teta then hsl:=1
          else if hsl=teta then hsl:=0
            else hsl:=-1;
          err[h].nilai[i]:=form3.tr[i]-hsl;
          if err[h].nilai[i]<>0 then  enol:=false;
       end;

     for i:=1 to wklm do
      for j:=1 to wbrs do
       begin
        bobot1.nilai[j,i]:=bobot1.nilai[j,i]+
        alfa*ip[j]*err[h].nilai[i];
       end;

 until enol=true or epoh=100;
end;
```

5.2.1.3.3 Algoritma Proses Pengujian

```
g:=0;
for i:=1 to polainp.brs do
 for j:=1 to polainp.klm do
  begin
   g:=g+1;
   if tbl1.kotak[i,j].Color=clBlack then
   ip[g]:=1 else ip[g]:=-1;
  end;

for i:=1 to wklm do
 begin
  sml[i]:=0;
  for j:=1 to wbrs do
   begin
    sml[i]:=sml[i]+(ip[j]*bobot.nilai[j,i]);
   end;
end;

h:=0;
for i:=1 to polatrg.brs do
 for j:=1 to polatrg.klm do
  begin
   h:=h+1;
   if sml[h]>0 then tbl2.kotak[i,j].Color:=clBlack
     else if sml[h]=0 then
       tbl2.kotak[i,j].Color:=clLime
         else tbl2.kotak[i,j].Color:=clYellow;
  end;
```

5.2.2 Implementasi Antarmuka

Berikut ini adalah gambaran antarmuka-antarmuka yang terdapat pada sistem perangkat lunak untuk pengenalan pola.

5.2.2.1 Antarmuka Menu Utama

Form utama berfungsi sebagai tempat menu dasar diletakkan. Terdapat beberapa menu dasar yang masing-masing mempunyai fungi sebagai berikut :

1. Tombol Baru

   Untuk menciptakan sebuah pola karakter 2 dimensi yang baru.

2. Tombol Buka

   Untuk membuka sebuah data tentang pola karakter ke dalam antarmuka.

3. Tombol Simpan

   Untuk menyimpan semua data dalam perangkat lunak.

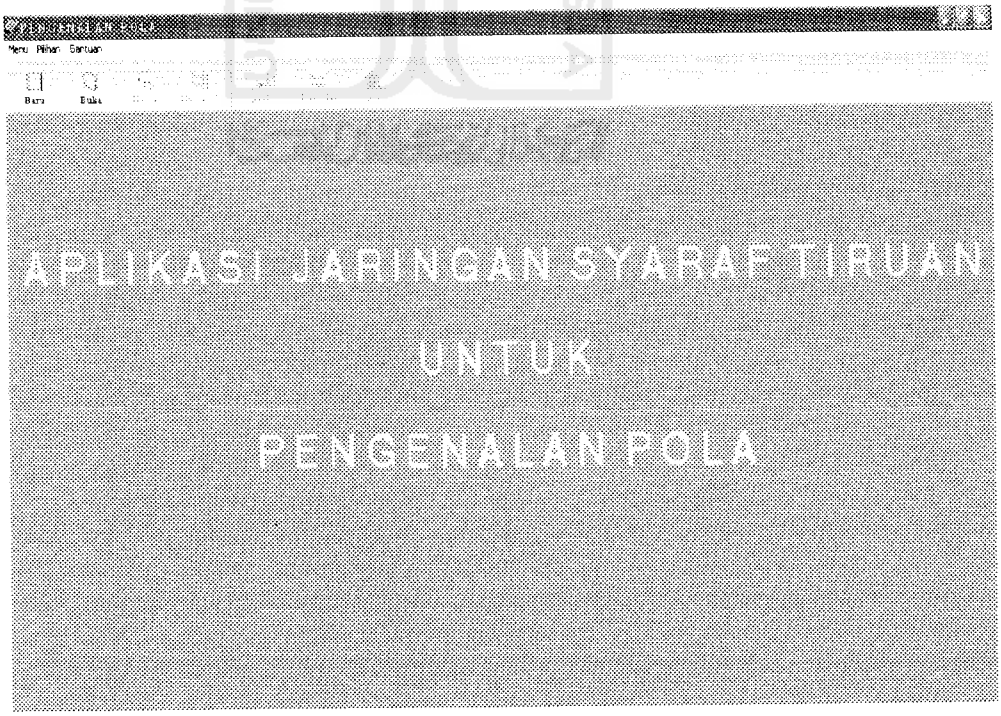
4. Tombol Simpan Ke

   Untuk menyimpan sebuah data dengan nama yang lain atau menyimpan sebuah data yang telah dirubah sebagiannya dengan nama yang lain.

5. Tombol Keluar

   Untuk keluar dari program ini.

Gambar 5.1 adalah gambar dari antarmuka menu utama :



Gambar 5.1 Antarmuka Menu Utama

5.2.2.2 Antarmuka Ukuran Pola

Antarmuka ini digunakan untuk mengisikan nilai baris dan kolom pola karakter 2 dimensi, sebagai masukan dan target yang direpresentasikan dalam bentuk matrik

Gambar 5.2 adalah gambar dari antarmuka ukuran pola :

Gambar 5.2 Antarmuka Ukuran Pola

5.2.2.3 Antarmuka Pelatihan

Antarmuka ini digunakan untuk mengisi data pola karakter, merubah data, menghapus data dan melakukan proses pelatihan.

Terdapat beberapa komponen yang digunakan dalam antarmuka ini:

1. Tombol Tambah Data

   Untuk menambahkan data pola karakter yang akan dilakukan proses pelatihan dan pengujian.

2. Tombol Matrik

   Untuk menampilkan *form* matrik.

3. Tombol Ubah Data

   Digunakan untuk merubah data pola karakter yang telah dimasukkan.

4. Tombol Hapus Data

   Digunakan untuk menghapus data pola karakter 2 dimensi.

5. Tombol Pelatihan

   Digunakan untuk melakukan proses pelatihan.

6. Kotak Masukan

   Tempat pola karakter 2 dimensi sebagai masukan.

7. Kotak Target

   Tempat pola karakter 2 dimensi sebagai targetnya.

8. *Radio Button*

   Digunakan untuk memilih metode pelatihan yang digunakan.

9. *Edit Learning Rate*

   Digunakan untuk mengisikan nilai *learning rate*.

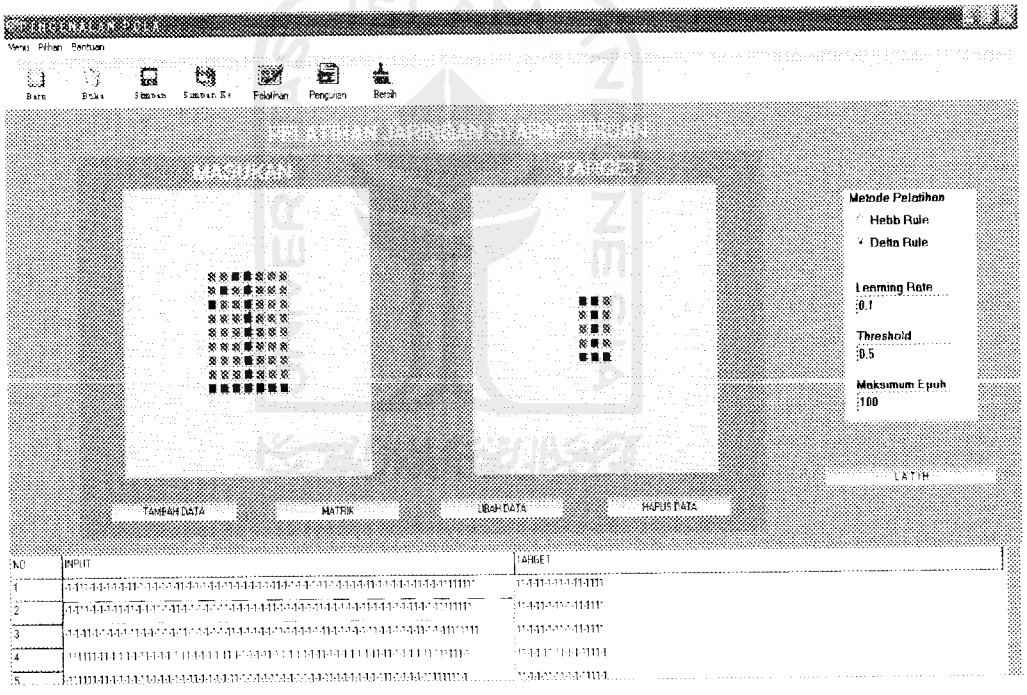
10. Edit Threshold

Digunakan untuk mengisikan nilai *threshold*.

11. StringGrid

Digunakan untuk menampilkan data-data yang telah dimasukkan.

Gambar 5.3 adalah gambar dari antarmuka pelatihan.



Gambar 5.3 Antarmuka Pelatihan

5.2.2.4 Antarmuka Pengujian

Antarmuka ini digunakan untuk melakukan proses pengujian pada jaringan syaraf tiruan. Terdapat beberapa komponen pada antarmuka ini :

1. Kotak Masukan

   Tempat pola karakter yang akan diuji.

2. Kotak Hasil

   Menampilkan hasil dari proses pengujian.

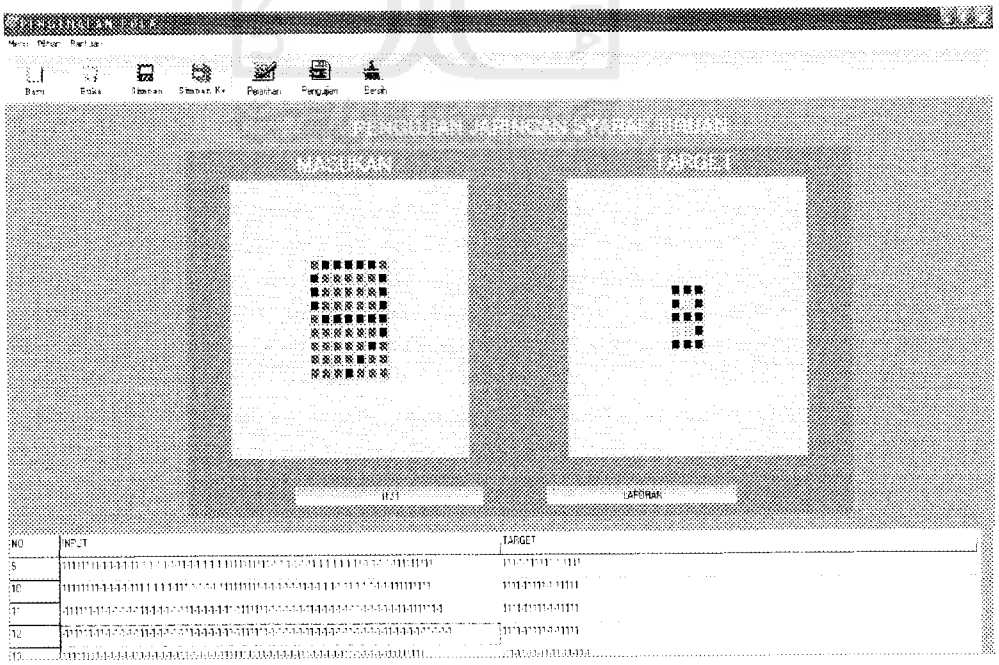
3. Tombol Uji

   Digunakan untuk melakukan proses pengujian.

4. Tombol Laporan

   Digunakan untuk melihat laporan.

Gambar 5.4 adalah gambar dari antarmuka pengujian.



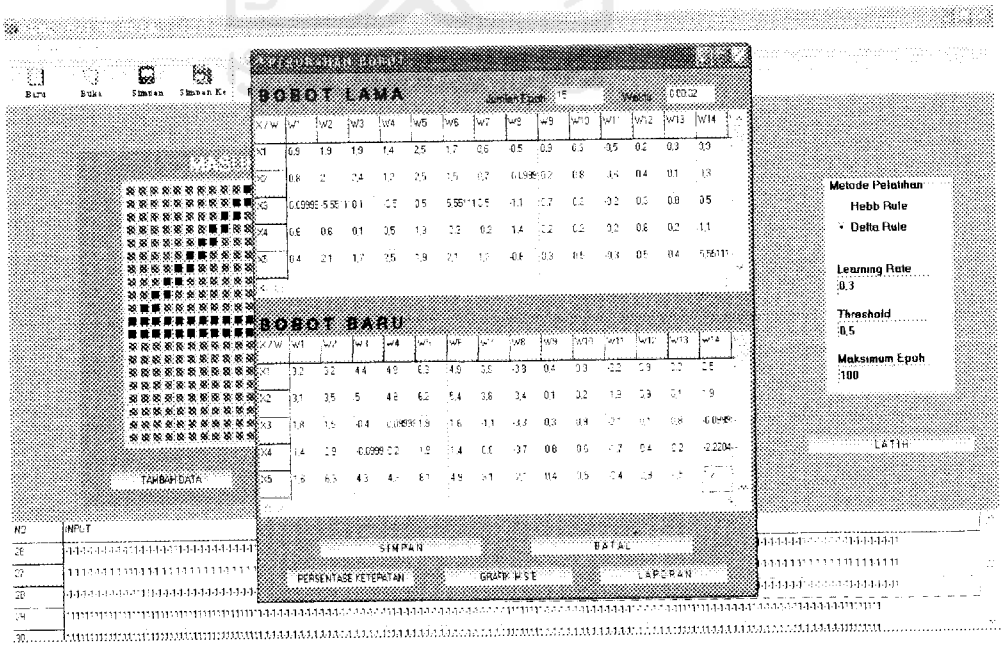
Gambar 5.4 Antarmuka Pengujian

5.2.2.5 Antarmuka Perubahan Bobot

Antarmuka ini untuk menampilkan perubahan bobot yang terjadi setelah dilakukan proses pelatihan. Komponen dalam antarmuka ini adalah:

1. Dua buah S*tringgrid* untuk melihat nilai bobot sebelum dan sesudah pelatihan.
2. Tombol simpan untuk menyimpan perubahan nilai bobot.
3. Tombol batal untuk membatalkan perubahan nilai bobot.
4. Tombol laporan untuk melihat laporan.
5. Komponen *edit1* menunjukkan jumlah *epoch* pada pelatihan metode *delta rule* dan *edit2* menunjukkan waktu yang diperlukan untuk melakukan proses pelatihan.
6. Tombol persentase ketepatan untuk menampilkan *form* persentase ketepatan.
7. Tombol Grafik MSE untuk menampilkan *form* Grafik *Mean Square Error*.

Gambar 5.4 adalah gambar dari antarmuka perubahan bobot.

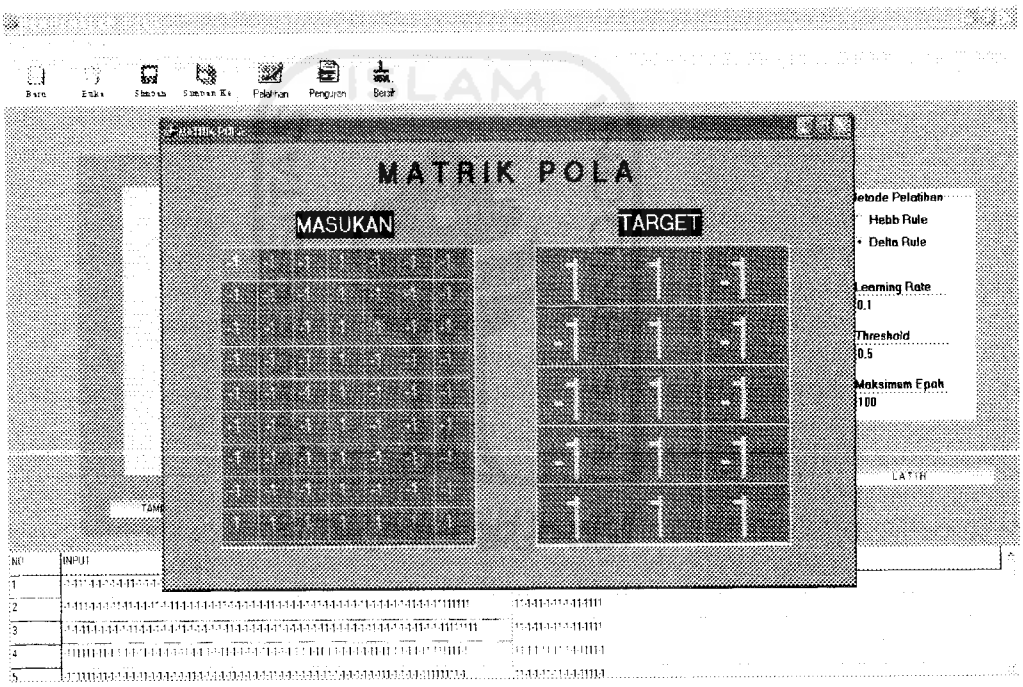


Gambar 5.5 Antarmuka Perubahan Bobot

5.2.2.6 Antarmuka Matrik Pola

Antarmuka ini digunakan untuk menampilkan nilai masukan dan target dari pola karakter 2 dimensi dalam bentuk matrik.

Gambar 5.6 adalah gambar dari antarmuka matrik pola.

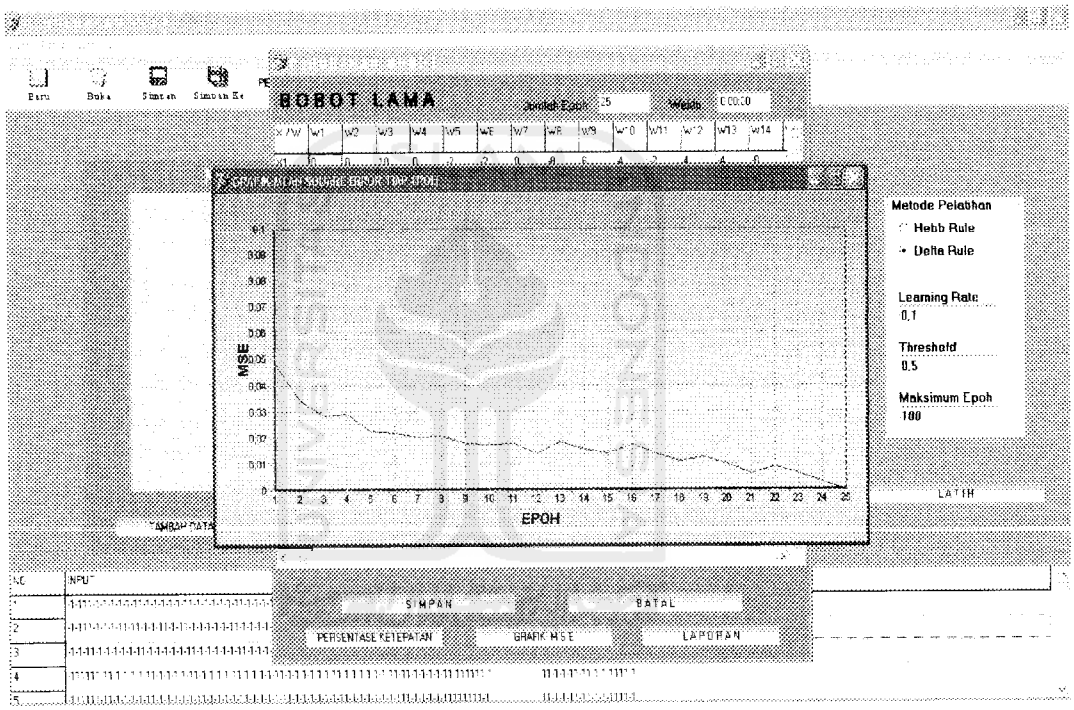


Gambar 5.6 Antarmuka Matrik Pola

5.2.2.7 Antarmuka Grafik Mean Square Error

Pada antarmuka ini menampilkan grafik *mean square error* pada setiap *epoch* pelatihan pola karakter 2 dimensi dengan menggunakan metode pelatihan *delta rule*.

Gambar 5.7 adalah gambar dari antarmuka grafik *mean square error*.

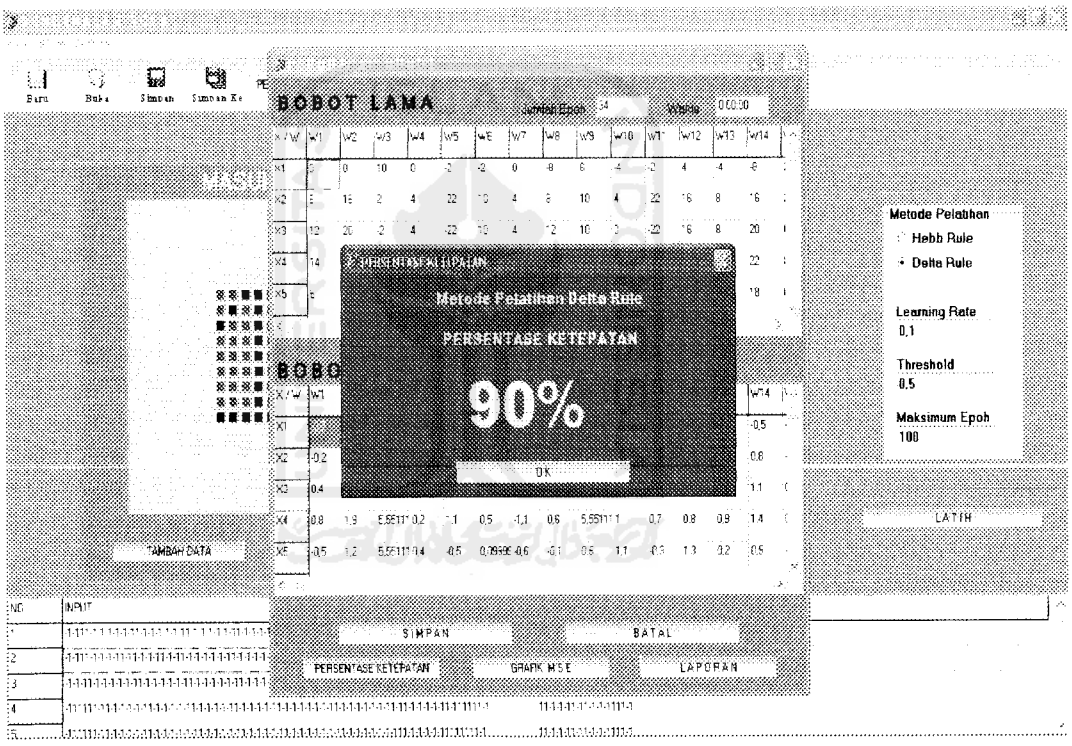


Gambar 5.7 Antarmuka Grafik *Mean Square Error*

5.2.2.8 Antarmuka Persentase Ketepatan

Pada antarmuka ini menampilkan persentase ketepatan output terhadap target yang diinginkan, baik dari proses pelatihan metode *hebb rule* maupun *delta rule*.

Gambar 5.8 adalah gambar dari antarmuka persentase ketepatan.

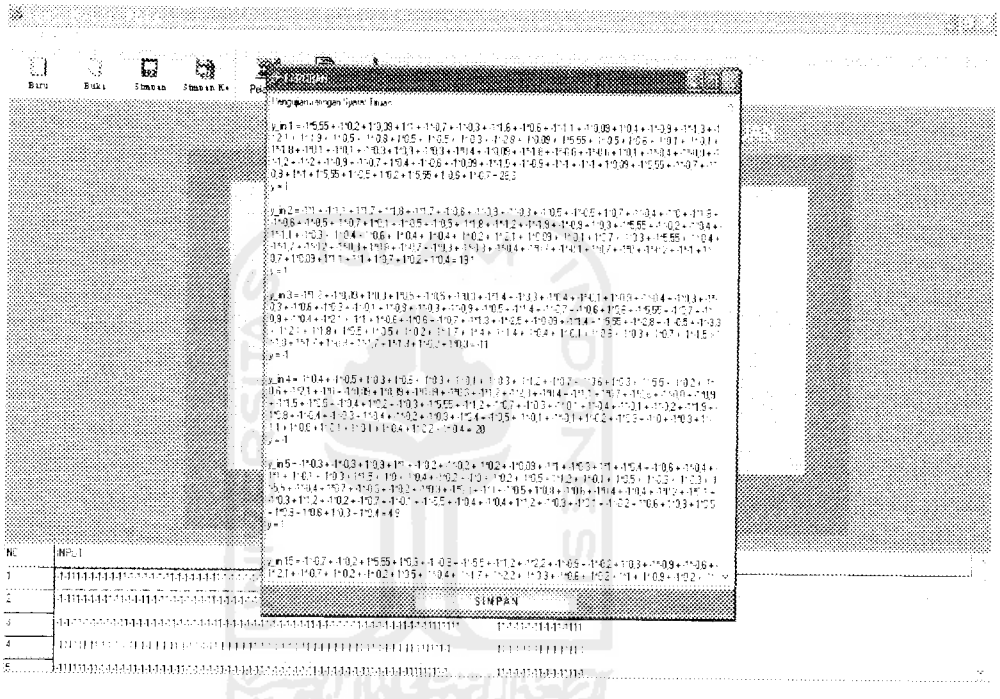


Gambar 5.8 Antarmuka Persentase Ketepatan

5.2.2.9 Antarmuka Laporan

Antarmuka ini digunakan untuk melihat proses perhitungan yang terjadi baik pada proses pelatihan maupun pada proses pengujian. Terdapat komponen memo yang digunakan untuk mencatat proses-proses diatas.

Gambar 5.9 adalah gambar dari antarmuka laporan.



Gambar 5.9 Antarmuka Laporan

BAB VI

ANALISIS KINERJA PERANGKAT LUNAK

6.1 Metode Analisis

Pada Bab ini akan dijabarkan mengenai pengujian kinerja atas perangkat lunak untuk pengenalan pola karakter 2 dimensi. Pengujian dilakukan dengan menjalankan sistem dengan 2 cara pemasukan, yakni secara normal ataupun dengan gangguan (tidak normal). Akan dijelaskan secara bertahap mengenai proses-proses yang ada dalam perangkat lunak ini.

6.1.1 Pengujian Pada Antarmuka Ukuran Pola

Langkah pertama dalam perangkat lunak ini adalah pembentukan pola karakter 2 dimensi dengan menentukan ukuran pola karakter. Dalam hal ini yang harus ditentukan adalah nilai kolom dan baris dari pola karakter masukan dan target.

a. Prosedur Normal

Pengisian nilai kolom dan baris pada masukan dan target merupakan penentu dari ukuran pola karakter 2 dimensi. Jumlah kotak-kotak sebagai representasi dari pola karakter memiliki jumlah baris dan kolom seperti nilai kolom dan baris yang diinputkan.

b. Prosedur Tidak Normal

Kesalahan pada bagian ini dikarenakan oleh pengisian jumlah kolom dan baris yang salah, baik itu pada nilai masukan maupun nilai target. Hal ini disebabkan jumlah kolom dan baris yang tidak diisi, atau baris bernilai kurang dari 5 atau kolom kurang dari 3 atau jumlah baris atau kolom melebihi 20. Ketika terjadi kesalahan ini, maka sistem akan memperingatkan *user* dan akan mengunci *user* pada parameter tempat kesalahan terjadi. Gambar 6.1 memperlihatkan kesalahan tersebut.

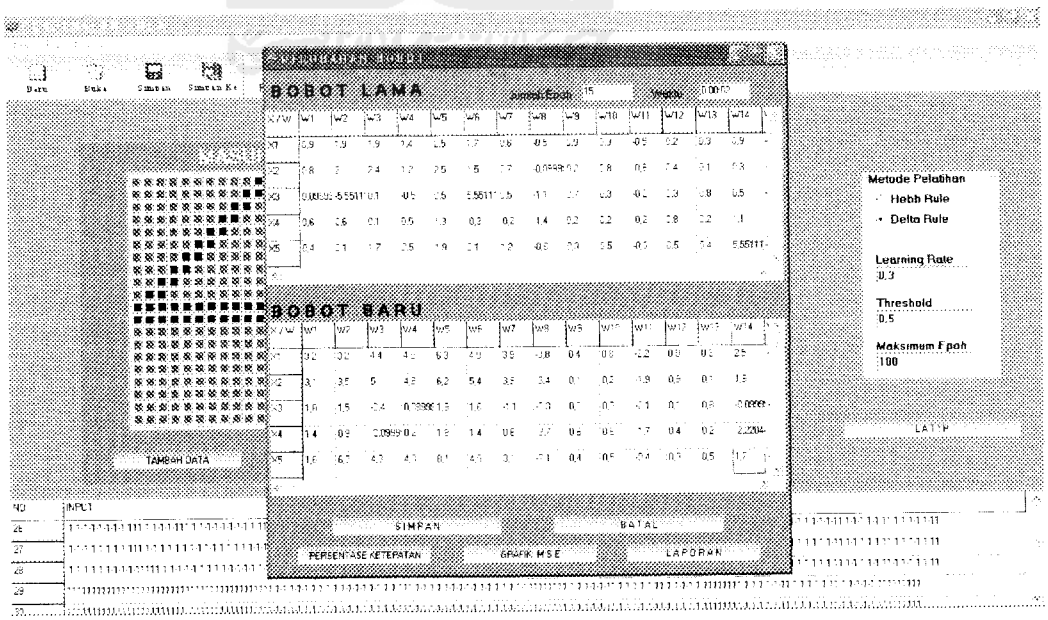
Gambar 6.1 Pesan Kesalahan pada Antarmuka Ukuran Pola

6.1.2 Pengujian Antarmuka Pelatihan

Pada Antarmuka ini, *user* akan memasukkan data-data pola masukan dan target berupa nilai-nilai biner. Selain itu *user* harus mengisi parameter-parameter tertentu yang dibutuhkan oleh metode pelatihan, seperti *learning rate*, *threshold*, dan maksimal *epoch*.

a. Prosedur Normal

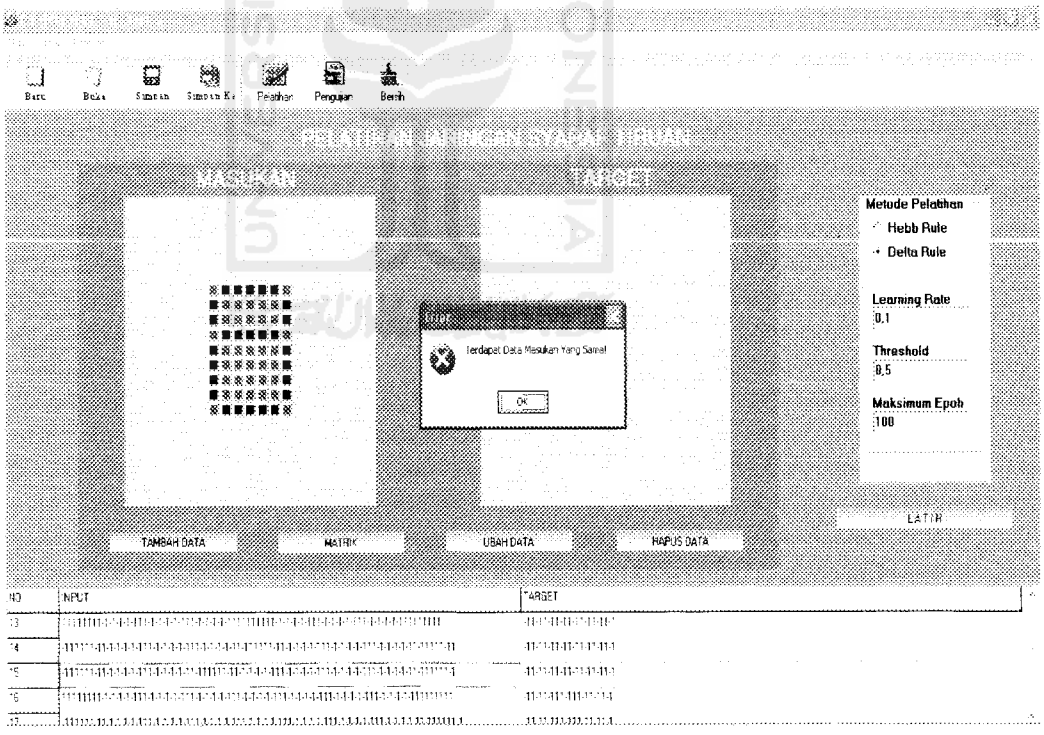
Pemasukkan input secara benar dan lengkap membuat sistem perangkat lunak melatih bobot-bobot dari data pola karakter 2 dimensi dengan menggunakan parameter-parameter pada pelatihan. Sistem memiliki batasan maximum iterasi sebesar 100. Perubahan akan diperlihatkan oleh antarmuka perubahan bobot. Jika *user* setuju akan pelatihan itu, maka *user* menekan tombol Simpan Perubahan. Hasil dari perubahan ini dapat dilihat pada antarmuka perubahan bobot



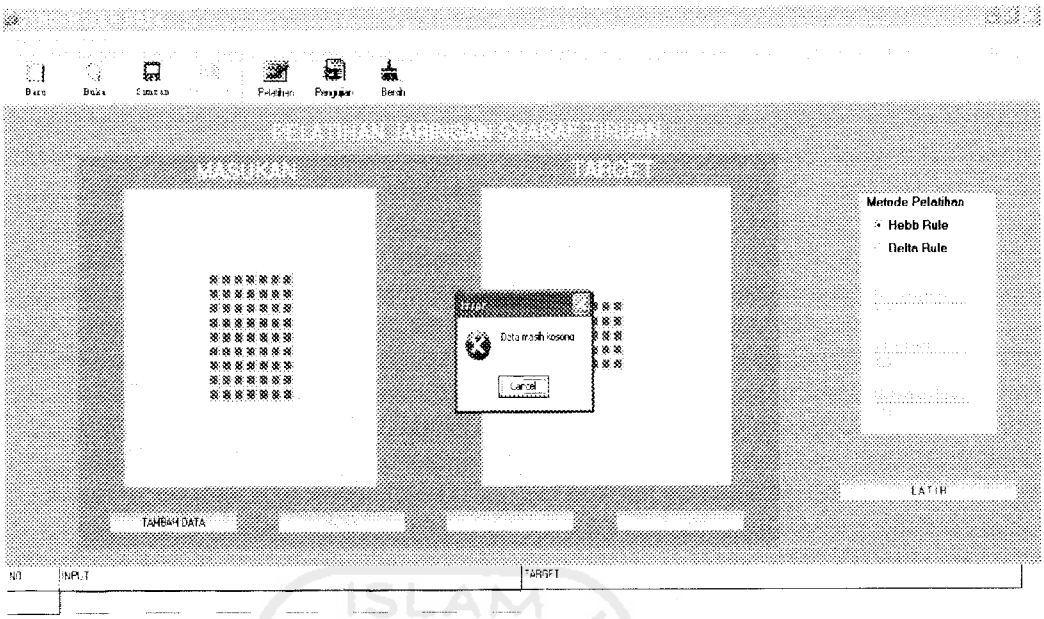
Gambar 6.2 Antarmuka Perubahan Bobot

b. Prosedur Tidak Normal

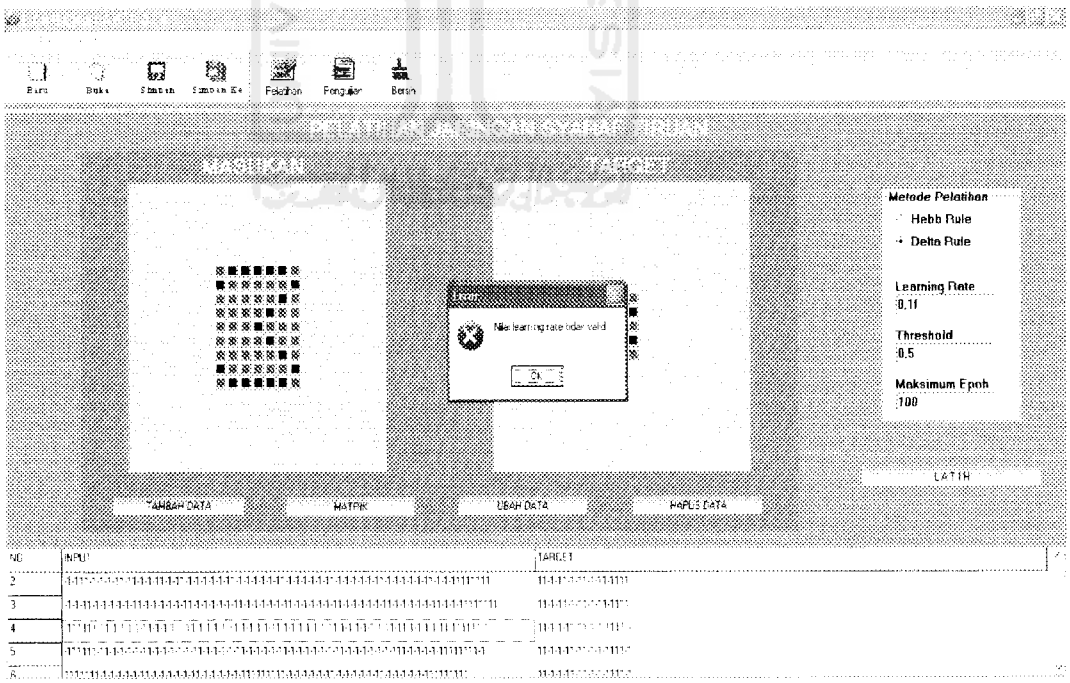
Kesalahan akan terjadi jika *user* menambahkan data lebih dari 1 data masukan yang memiliki nilai sama, tidak memasukan parameter yang lengkap seperti belum mengisikan data pola karakter, atau salah dalam mengisi nilai-nilai parameter yang dibutuhkan untuk pelatihan misalnya salah dalam mengisikan nilai *learning rate*, *threshold* dan *epoch*. Sistem akan memberikan pesan peringatan jika terjadi kesalahan pengisian pada *user*. Pesan kesalahan yang muncul untuk kesalahan jenis ini terlihat pada Gambar 6.3, Gambar 6.4, Gambar 6.5, Gamber 6.6 dan Gamber 6.7.



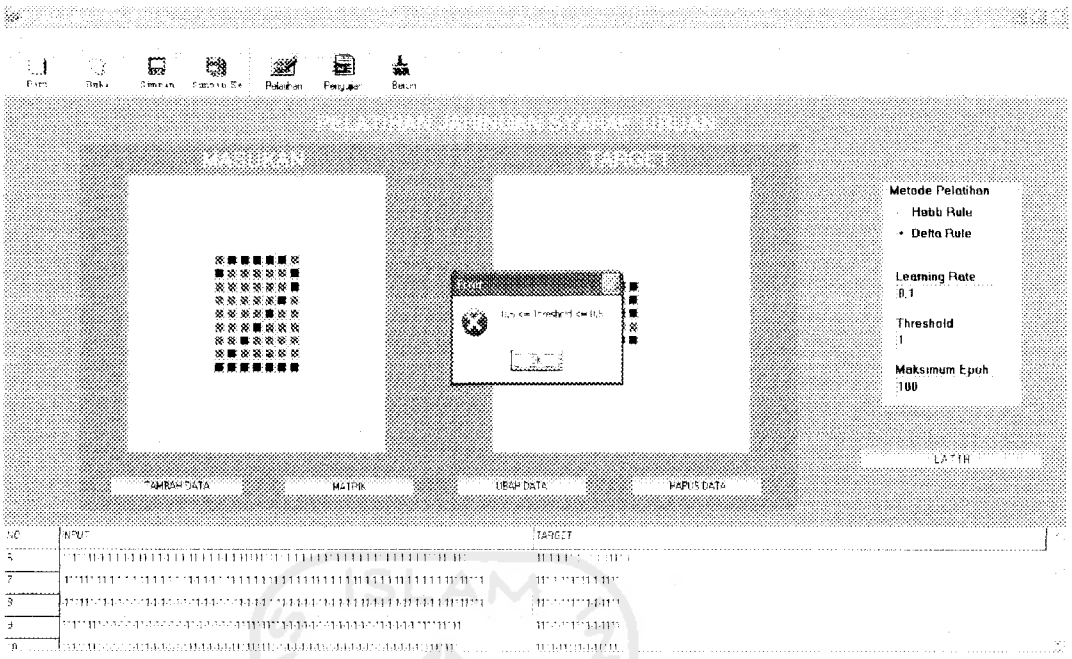
Gambar 6.3 Pesan Kesalahan Menambahkan Data Masukan yang Memiliki Nilai Sama



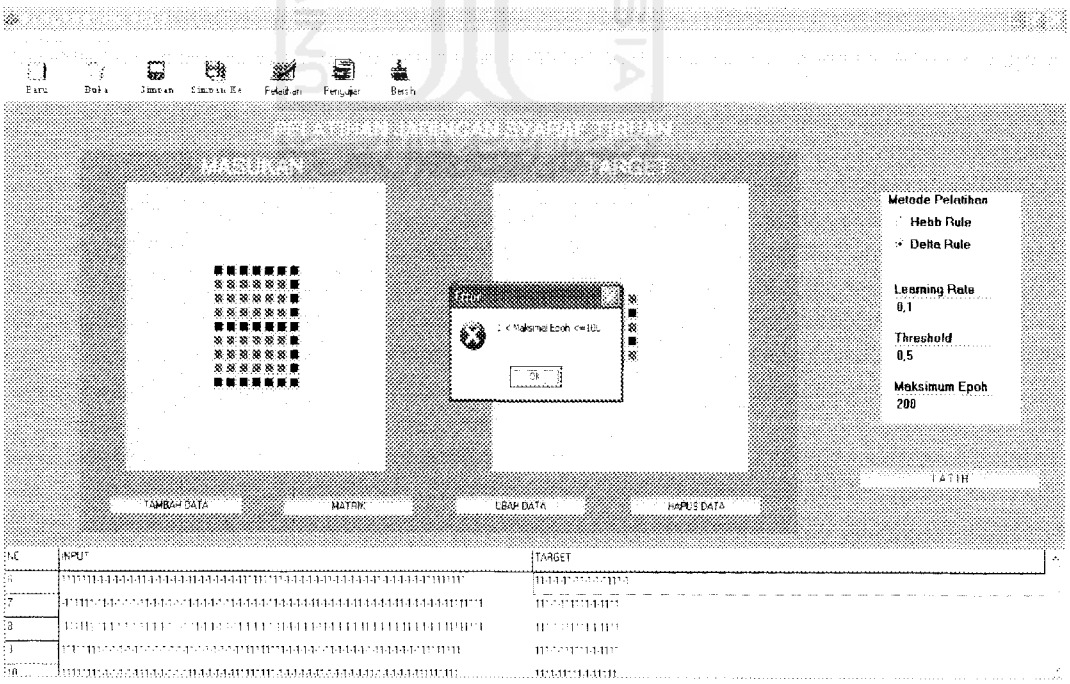
Gambar 6.4 Pesan Kesalahan pada Masukan Data Pola Karakter 2 Dimensi



Gambar 6.5 Pesan Kesalahan pada Nilai *Leraning Rate* yang Tidak Valid



Gambar 6.6 Pesan Kesalahan pada Nilai *Threshold* yang Melebihi Batas



Gambar 6.7 Pesan Kesalahan pada Nilai *Epoch* yang Melebihi Batas Maksimal

6.2 Analisis Hasil Pelatihan dan Pengujian

Pada sub bab ini akan dianalisis hasil dari proses perhitungan pelatihan dan proses perhitungan pengujian atau pengenalan pola karakter 2 dimensi. Parameter yang dianalisis meliputi metode pelatihan yang digunakan, jumlah pola karakter 2 dimensi yang dilatih, nilai *learning rate* yang digunakan, jumlah iterasi, dan ketepatan nilai *output*.

Tabel 6.1 adalah tabel perbandingan proses pelatihan jaringan syaraf tiruan menggunakan metode *delta rule*. Nilai *threshold* yang digunakan pada metode *delta rule* adalah 0,5.

Tabel 6.1 Perbandingan Proses Pelatihan Metode *Delta Rule*

Ukuran Pola Karakter 2 Dimensi		Jumlah Pola Karakter 2 Dimensi	Learning Rate	Jumlah Iterasi
Masukan	Target			
9 x 7	5 x 3	30	0,1	25
9 x 7	5 x 3	30	0,2	34
9 x 7	5 x 3	30	0,3	29
15 x 10	5 x 3	70	0,1	20
15 x 10	5 x 3	70	0,2	26
15 x 10	5 x 3	70	0,3	29
20 x 20	9 x 7	70	0,1	25
20 x 20	9 x 7	70	0,2	21
20 x 20	9 x 7	70	0,3	23

Tabel 6.2 Persentase Ketepatan Proses Pelatihan Metode *Delta Rule*

Ukuran Pola Karakter 2 Dimensi		Jumlah Pola Karakter 2 Dimensi	Learning Rate	Persentase Ketepatan
Masukan	Target			
9 x 7	5 x 3	30	0,1	93,3 %
9 x 7	5 x 3	30	0,2	96,6 %
9 x 7	5 x 3	30	0,3	100 %
15 x 10	5 x 3	70	0,1	95,7 %
15 x 10	5 x 3	70	0,2	98,5 %
15 x 10	5 x 3	70	0,3	100 %
20 x 20	9 x 7	70	0,1	98,5 %
20 x 20	9 x 7	70	0,2	98,5 %
20 x 20	9 x 7	70	0,3	100 %

Pada Tabel 6.2 menunjukkan hasil pelatihan dari jaringan syaraf tiruan yang terbentuk. Pelatihan yang dilakukan oleh jaringan syaraf tiruan mencapai hasil 100 % bilamana *output* pelatihan sesuai dengan target yang diinginkan. Semakin tinggi nilai *learning rate* maka semakin tinggi tingkat ketelitian yang dikerjakan oleh sistem.

Tabel 6.3 Persentase Ketepatan Proses Pelatihan Metode *Hebb Rule*

Ukuran Pola Karakter 2 Dimensi		Jumlah Pola Karakter 2 Dimensi	Persentase Ketepatan
Masukan	Target		
9 x 7	5 x 3	3	100 %
9 x 7	5 x 3	5	10 %
9 x 7	5 x 3	30	10 %
14 x 9	5 x 3	3	100 %
14 x 9	5 x 3	5	20 %
14 x 9	5 x 3	70	11,4 %
20 x 20	5 x 3	3	100 %
20 x 20	5 x 3	10	9,9 %
20 x 20	5 x 3	70	1,42 %

Tabel 6.3 menunjukkan tabel persentase pelatihan jaringan syaraf tiruan dengan menggunakan metode pelatihan *hebb rule*. Pada metode pelatihan ini menunjukkan bahwa semakin banyak data yang dilatih maka semakin sedikit persentase ketepatan yang diperoleh, hal ini sebabkan pada pelatihan *hebb rule* hanya menjumlahkan nilai bobot dari pola *input* dan target, tidak meminimalisir kesalahan terhadap pola target yang diharapkan.

BAB VII

PENUTUP

7.1 Kesimpulan

Kesimpulan mengenai Sistem Jaringan Syaraf Tiruan untuk pengenalan pola karakter 2 dimensi ini adalah :

1. Sistem Jaringan Syaraf Tiruan khususnya algoritma *hebb rule* dan *delta rule* dapat digunakan untuk mengenali pola-pola karakter 2 dimensi.
2. Proses pelatihan dan pengujian dengan menggunakan metode *delta rule* hasilnya lebih akurat dibandingkan dengan menggunakan metode *hebb rule*.
3. Pada proses pelatihan menggunakan metode *delta rule*, semakin tinggi nilai *learning rate* yang digunakan maka semakin tinggi tingkat ketelitian jaringan syaraf tiruan. Sedangkan pada metode *hebb rule*, semakin banyak pola yang dilatih maka semakin kecil keakuratan jaringan syaraf tiruan. Hal ini disebabkan pada pelatihan metode *hebb rule* hanya menaikkan nilai bobot pada setiap neuron yang terhubung yakni pola *input* dan target, tidak mengkoreksi terhadap nilai *output* yang diharapkan, sehingga sistem ini hanya dapat mengenali jumlah pola yang sedikit. Sedangkan pada metode *delta rule* mengubah nilai bobot yang menghubungkan antara jaringan *input* ke unit *output* dengan nilai target untuk meminimalkan nilai *error* selama proses pelatihan.

7.2 Saran

Ada beberapa saran pengembangan yang perlu dilakukan :

1. Sistem ini masih dibatasi oleh *range* besar pola karakter 2 dimensi, jumlah data pola yang dilatih, dan juga batasan iterasi. Untuk pengembangannya agar sistem memiliki *range* yang lebih luas, serta sistem dapat mengenali bentuk-bentuk pola yang lain seperti pola dalam bentuk citra ataupun bentuk yang lain.
2. Pemrograman yang lebih *user friendly* dan penggunaan sisi multimedia sebagai bantuan.

DAFTAR PUSTAKA

[FAU94] Fausset, Laurene. *Fundamental of Neural Network: Architecture, Algorithm, and Application.* New Jersey: Prentice Hall, 1994.

[GUN03] Gunawan. Skripsi: Perangkat Lunak Untuk Simulasi Jaringan Syaraf Tiruan. Yogyakarta: Fakultas Teknologi Industri, Jurusan Teknik Informatika, Universitas Isalam Indonesia, 2003.

[KAD01] Kadir, Abdul. Dasar Pemrograman Delphi 5.0. Yogyakarta: Andi Yogyakarta, 2001.

[KUS03] Kusumadewi, Sri. *Aritificial Intelligence(*Teknik dan Aplikasinya). Yogyakarta: Penerbit Graha Ilmu, 2003.

[NUG02] Nugroho, Widodo. Tip dan Trik Pemrograman Delphi. Jakarta: Penerbit PT Elex Media Komputindo, 2002.

[PRA01] Pranata, Anthony. Pemrograman Borland Delphi Edisi 3. Yogyakarta: Andi Yogyakarta, 2001.