PERAMALAN NILAI TUKAR MATA UANG UNITED STATES DOLLAR TERHADAP CANADA DOLLAR MENGGUNAKAN JARINGAN SYARAF TIRUAN RESILIENT BACKPROPAGATION

SKRIPSI

Sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer
dalam bidang Ilmu Komputer

oleh:
ARIEF RACHMANSYAH
0210960011-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2009

LEMBAR PENGESAHAN SKRIPSI

PERAMALAN NILAI TUKAR MATA UANG UNITED STATES DOLLAR TERHADAP CANADA DOLLAR MENGGUNAKAN JARINGAN SYARAF TIRUAN RESILIENT BACKPROPAGATION

oleh:
ARIEF RACHMANSYAH
0210960011-96

Setelah dipertahankan di depan Majelis Penguji
pada tanggal 14 Agustus 2009
dan dinyatakan memenuhi syarat untuk memperoleh gelar
Sarjana Komputer dalam bidang Ilmu Komputer

Pembimbing I

Drs. Marji, M.T
NIP. 131 993 386

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas Brawijaya

Drs. Agus Suryanto, M.Sc
NIP. 132 126 049

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Arief Rachmansyah
NIM : 0210960011-96
Jurusan : Matematika
Program Studi : Ilmu Komputer
Penulis skripsi berjudul : Peramalan Nilai Tukar Mata Uang United States Dollar Terhadap Canada Dollar Menggunakan Jaringan Syaraf Tiruan Resilient Backpropagation

Dengan ini menyatakan bahwa :

1. **Isi dari skripsi yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam Skripsi ini.**
2. **Apabila dikemudian hari ternyata Skripsi yang saya tulis terbukti hasil jiplakan, maka saya akan bersedia menanggung segala resiko yang akan saya terima.**

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang, Agustus 2009
Yang menyatakan,

Arief Rachmansyah
NIM. 0210960011

PERAMALAN NILAI TUKAR MATA UANG UNITED STATES DOLLAR TERHADAP CANADA DOLLAR MENGGUNAKAN JARINGAN SYARAF TIRUAN RESILIENT BACKPROPAGATION

ABSTRAK

Untuk meramalkan nilai mata uang dengan jaringan syaraf tiruan diperlukan data nilai rata-rata harga penutupan selama beberapa minggu untuk meramalkan nilai rata-rata pada minggu berikutnya. Nilai rata-rata inilah yang nantinya akan dijadikan sebagai input pada jaringan syaraf tiruan. Metode Resilient Backpropagation mencari nilai bobot hanya dengan menggunakan parameter pembelajaran berupa nilai *learning rate* dan tanda dari turunan pertama fungsi *error* terhadap nilai bobot.

Sistem peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation* mampu meramalkan nilai *moving average* pada minggu yang akan datang dengan cukup baik.

UNITED STATES DOLLAR TO CANADA DOLLAR FOREIGN EXCHANGE RATES PREDICTION USING RESILIENT BACKPROPAGATION NEURAL NETWORK

ABSTRACT

.To predict foreign exchange rates using neural network, it needs average closing rates for weeks to predict average of the next incoming weeks. This average used to be the input of neural networks. Resilient backpropagation method looking for the weight using only learning parameters, which is learning rate value and the sign of first derivative of error function to weight value.

Foreign exchange rate prediction system using resilient backpropagation neural network is good enough to be able to predict moving average of next incoming weeks.

KATA PENGANTAR

Puji syukur penyusun panjatkan ke hadirat Allah SWT yang telah melimpahkan segala Rahmat, Karunia dan Hidayah-Nya sehingga Penulis dapat menyelesaikan tugas akhir dengan judul "**Peramalan nilai tukar mata uang united states dollar terhadap canada dollar menggunakan jaringan syaraf tiruan resilient backpropagation**".

Tugas akhir ini diajukan sebagai syarat untuk memperoleh gelar Sarjana Komputer di Fakultas MIPA, Jurusan Ilmu Komputer, Universitas Brawijaya Malang. Atas terselesaikannya tugas akhir ini, Penulis mengucapkan terima kasih kepada:

1. Drs. Marji, M.T, selaku Dosen Penasehat Akademik sekaligus Dosen Pembimbing penulisan tugas akhir.
2. Wayan Firdaus Mahmudy, S.Si., MT, selaku Ketua Program Studi Ilmu Komputer Jurusan Matematika FMIPA Universitas Brawijaya.
3. Dr. Agus Suryanto, M.Sc, selaku Ketua Jurusan Matematika FMIPA Universitas Brawijaya.
4. Segenap bapak dan ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada Penulis selama menempuh pendidikan di Program Studi Ilmu Komputer Jurusan Matematika FMIPA Universitas Brawijaya.
5. Segenap staf dan karyawan di Jurusan Matematika FMIPA Universitas Brawijaya yang telah banyak membantu Penulis dalam pelaksanaan penyusunan tugas akhir ini.
6. Orang tua Penulis atas segala dukungan materi dan doa restunya kepada Penulis.
7. Rekan-rekan Ilmu Komputer 2002 yang telah memberikan dukungannya dalam penyusunan tugas akhir ini.
8. Semua pihak yang telah membantu terselesaikannya laporan ini yang tidak dapat kami sebutkan satu per satu.

Penulis menyadari bahwa laporan ini tentunya tidak terlepas dari berbagai kekurangan dan kesalahan. Oleh karena itu, segala kritik dan saran yang bersifat membangun sangat Penulis harapkan dari berbagai pihak guna peningkatan kualitas penelitian serupa di masa mendatang.

Malang, Agustus 2009

Penulis

DAFTAR ISI

Halaman

DAFTAR TABEL

Halaman

DAFTAR GAMBAR

Halaman

BAB I
PENDAHULUAN

1.1. Latar Belakang Masalah

Pada era persaingan global seperti saat ini, nilai tukar mata uang menjadi salah satu faktor yang mempengaruhi perdagangan internasional dan stabilitas ekonomi pada berbagai negara (Kamruzzaman, 2006). Nilai tukar mata uang menjadi sangat penting ketika suatu perusahaan atau perseorangan membeli barang atau layanan dari produsen di luar negeri. Hal ini dikarenakan pembeli harus membayar barang atau layanan tersebut menggunakan mata uang yang digunakan oleh produsen tersebut.

Sebagai contoh jika nilai tukar rupiah lebih rendah terhadap dolar Amerika, maka perusahaan di Amerika akan lebih suka untuk membeli barang dari Indonesia dikarenakan barang di Indonesia lebih murah harganya dan hal ini akan menguntungkan exporter di Indonesia. Namun di lain pihak jika ternyata negara Indonesia mempunyai ketergantungan terhadap barang-barang dari Amerika, nilai tukar rupiah yang lebih rendah terhadap dolar akan membuat barang-barang dari Amerika menjadi lebih mahal. Dapat disimpulkan bahwa harga barang expor dan import sangat dipengaruhi oleh nilai tukar mata uang. Oleh karena nilai tukar mata uang selalu berubah tiap waktu maka beberapa metode digunakan untuk memprediksinya.

Salah satu metode yang paling banyak digunakan untuk meramalkan nilai tukar mata uang adalah metode jaringan syaraf tiruan backpropagation. Jaringan syaraf tiruan *backpropagation* adalah sistem pemroses informasi yang memiliki karakteristik mirip dengan jaringan syaraf biologi. Jaringan syaraf tiruan *backpropagation* dibentuk sebagai generalisasi model matematika dari jaringan syaraf biologi, dengan asumsi bahwa pemrosesan informasi terjadi pada banyak elemen sederhana (*neuron*), sinyal dikirimkan diantara *neuron* – *neuron* melalui penghubung – penghubung, penghubung antar *neuron* memiliki bobot yang akan memperkuat atau memperlemah sinyal, untuk menentukan *output* setiap *neuron* menggunakan fungsi aktivasi yang dikenakan pada jumlahan *input* yang diterima.

Metode *backpropagation* kemudian dikembangkan lagi oleh Redmiller dan Braun (Rojas, 1996) agar menjadi lebih efisien dalam

mendapatkan nilai bobot yang menghasilkan nilai error minimum. Metode backpropagation ini kemudian disebut dengan Resilient Backpropagation. Metode Resilient Backpropagation mencari nilai bobot hanya dengan menggunakan parameter pembelajaran berupa nilai *learning rate* dan tanda dari turunan pertama fungsi error terhadap nilai bobot.

Untuk meramalkan nilai mata uang dengan jaringan syaraf tiruan diperlukan data nilai rata-rata harga penutupan selama beberapa minggu untuk meramalkan nilai rata-rata pada minggu berikutnya. Nilai rata-rata inilah yang nantinya akan dijadikan sebagai input pada jaringan syaraf tiruan. Dari penelitian yang telah dilakukan oleh Yao (2000), metode peramalan mata uang dengan menggunakan jaringan syaraf tiruan menghasilkan nilai prediksi rata-rata lebih baik daripada metode statistika seperti ARIMA.

Data yang digunakan sebagai bahan peramalan dalam skripsi ini adalah data nilai tukar mata uang United State Dollar terhadap Canada Dollar. Nilai tukar United State Dollar terhadap Canada Dollar mempunyai nilai persaingan yang sangat ketat dalam pasar mata uang. Berdasarkan latar belakang tersebut, untuk memprediksi nilai tukar mata uang United State Dollar terhadap Canada Dollar dengan menggunkan jaringan syaraf tiruan resilient backpropagation. Oleh karena itu diambil judul **Peramalan Nilai Tukar Mata Uang United State Dollar Terhadap Canada Dollar Menggunakan Jaringan Syaraf Tiruan Resilient Backpropagation.**

1.2. Rumusan Masalah

Berdasarkan uraian pada latar belakang masalah, maka dalam skripsi ini dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana memodelkan / menentukan arsitektur jaringan syaraf tiruan yang terbaik untuk meramalkan nilai tukar mata uang asing?
2. Bagaimana hasil peramalan yang dihasilkan apakah mendekati harga yang sebenarnya atau tidak?

1.3. Batasan Masalah

Dari permasalahan di atas, berikut ini diberikan batasan masalah :

1. *Input* yang digunakan untuk pelatihan pada Jaringan Syaraf Tiruan disini adalah faktor-faktor internal saja.
2. Jaringan Syaraf Tiruan digunakan untuk peramalan nilai tukar mata uang adalah data dalam kurun waktu 11 tahun
3. Jaringan Syaraf Tiruan disini menggunakan 1 *hidden layer* saja dengan menggunakan bias.

1.4. Tujuan Penelitian

Tujuan penelitian yang ingin dicapai adalah:

1. Menentukan arsitektur jaringan syaraf tiruan *backpropagation* yang terbaik untuk peramalan nilai tukar mata uang.
2. Menentukan *input* yang paling baik digunakan untuk peramalan nilai tukar mata uang berdasarkan percobaan.
3. Membandingkan peramalan nilai tukar mata uang hasil perangkat lunak dengan nilai tukar mata uang asing hari berikutnya yang sebenarnya.

1.5. Manfaat Penelitian

Memberikan arsitektur jaringan syaraf tiruan *backpropagation* yang terbaik dengan *input* yang paling sesuai untuk peramalan nilai tukar mata uang asing, yang dapat digunakan oleh pelaku pasar mata uang untuk mengetahui penutupan nilai tukar mata uang asing untuk keesokan harinya sebagai bahan pertimbangan apakah akan menukar mata uang asing atau tidak.

BAB II
TINJAUAN PUSTAKA

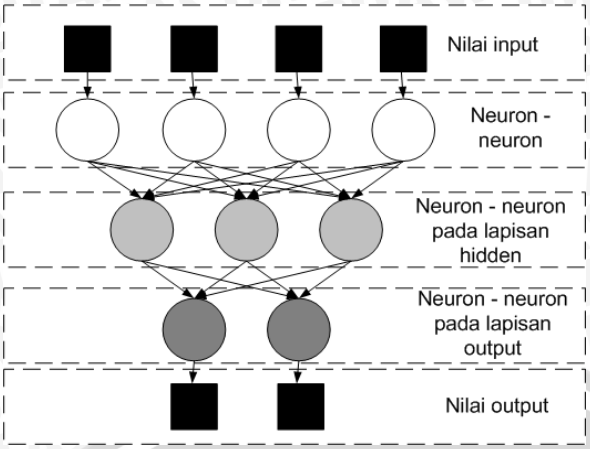
2.1. Jaringan Syaraf Tiruan

Menurut Kusumadewi (2003), jaringan syaraf tiruan ialah suatu sistem pengolah informasi yang mempunyai karakteristik menyerupai jaringan syaraf biologis tubuh manusia. Jaringan syaraf tiruan telah dikembangkan dengan menggunakan model matematis untuk meniru cara kerja jaringan syaraf biologis, dengan berdasarkan asumsi :

a. Pengolah informasi terdiri dari elemen-elemen sederhana yang disebut *neuron* atau unit.
b. Sinyal dilewatkan dari satu *neuron* ke *neuron* yang lain melalui hubungan koneksi.
c. Tiap hubungan koneksi mempunyai nilai bobot tersendiri.
d. Tiap *neuron* mempergunakan fungsi aktivasi terhadap masukan yang diterimanya untuk menentukan sinyal keluarannya.

Informasi (disebut dengan : *input*) akan dikirim ke *neuron* dengan bobot kedatangan tertentu. *Input* akan diproses oleh suatu fungsi perambatan yang akan menjumlahkan nilai-nilai semua bobot yang datang. Hasil penjumlahan ini kemudian dibandingkan dengan suatu nilai ambang (*threshold*) tertentu melalui fungsi aktivasi setiap *neuron*.

Pada jaringan syaraf tiruan, *neuron-neuron* dikumpulkan dalam lapisan-lapisan (*layer*). Suatu jaringan syaraf minimum tersusun atas *input layer* dan *output layer*. Dalam beberapa tipe jaringan diantara *input layer* dan *output layer* terdapat *hidden layer*. Hal ini berarti bahwa semua neuron pada *input layer* akan berhubungan ke semua *neuron* dalam *hidden layer* yang selanjutnya setiap *neuron* dalam *hidden layer* nantinya akan dihubungkan ke semua *neuron* di *output layer* (Siang, 2005). Contoh model jaringan syaraf dengan 3 lapisan dapat dilihat pada Gambar 2.1.



Gambar 2.1 Jaringan syaraf dengan 3 lapisan

2.1.1. Proses Pembelajaran

Proses pembelajaran adalah proses untuk menentukan bobot penghubung antar *neuron*. Nilai bobot bertambah jika informasi yang diberikan oleh *neuron* yang bersangkutan tersampaikan, sebaliknya jika informasi tidak disampaikan oleh *neuron* ke *neuron* yang lain, maka nilai bobot yang menghubungkan keduanya dikurangi.

Pada saat pembelajaran dilakukan pada *input* yang berbeda, nilai bobot diubah secara dinamis hingga mencapai suatu nilai yang cukup seimbang. Nilai yang cukup seimbang ini mengindikasikan bahwa tiap-tiap *input* telah berhubungan dengan *output* yang diharapkan.

Berdasarkan metode pembelajarannya jaringan syaraf tiruan dibagi menjadi (Siang, 2005) :

1. *Supervised* (pembelajaran yang terawasi)
   Pada sistem pembelajaran *supervised* terdapat pasangan data (masukan dan target keluaran) yang dipakai untuk melatih jaringan hingga memperoleh bobot yang diinginkan.
2. *Unsupervised* (pembelajaran yang tidak terawasi)
   Sistem pembelajaran *unsupervised* kebalikan dari sistem pembelajaran *supervised*, tidak ada pasangan data (masukkan dan target keluaran) yang dipakai untuk melatih, tetapi perubahan bobot jaringan berdasarkan parameter tertentu dan jaringan dimodifikasi menurut parameter tersebut.

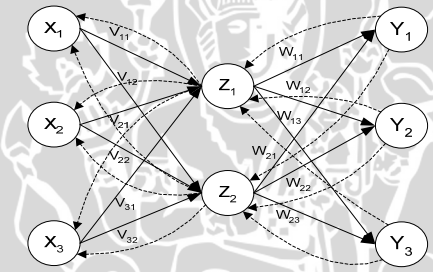
Di dalam proses pembelajaran, *learning rate* adalah sebuah konstanta yang nilainya diantara 0 sampai 1 yang berguna untuk mempercepat proses pembelajaran. Sedangkan *max epoch* adalah batas maksimum perulangan untuk melakukan proses pembelajaran.

2.1.2. Algoritma *Backpropagation*

Algoritma *backpropagation* merupakan algoritma pembelajaran yang terawasi. Algoritma *backpropagation* menggunakan nilai *error* pada *output layer* untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan nilai *error*, tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu. Pada saat perambatan maju, *neuron-neuron* diaktifkan dengan menggunakan fungsi aktivasi *sigmoid*. Fungsi *sigmoid* yang sering dipakai yaitu :

$$f(x) = \frac{1}{1+e^{-x}} \qquad (2.1)$$

Keterangan :
e : bilangan natural yaitu 2.71828.
x : hasil penjumlahan dari sinyal-sinyal *input* terbobot.
Arsitektur jaringan *backpropagation* seperti terlihat pada gambar 2.2.



Gambar 2.2 Arsitektur jaringan *backpropagation.*

Algoritma *backpropagation* (Kusumadewi, 2003) :
1. Inisialisasi bobot dengan nilai *random* yang cukup kecil.
2. Kerjakan langkah–langkah berikut selama kondisi berhenti belum terpenuhi :

Untuk tiap-tiap pasangan elemen yang akan dilakukan pembelajaran, kerjakan :

feedforward :

a. Tiap-tiap unit *input* (X_i = 1,2,3,..,n) menerima sinyal x_i dan meneruskan *sinyal* tersebut ke semua unit pada lapisan di atasnya (*hidden layer*). Dimana n adalah jumlah unit *input.*

b. Tiap-tiap unit tersembunyi (Z_j, j= 1,2,3,…,p) menjumlahkan sinyal – sinyal *input* terbobot (z_in_j) :

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \quad (2.2)$$

Keterangan :

v_{0j} : nilai bobot dari bias ke unit tersebunyi j.

v_{ij} : nilai bobot dari input i ke unit tersebunyi j.

p : jumlah unit tersembunyi.

Gunakan fungsi aktivasi untuk menghitung sinyal *output* (z_j) :

$$z_j = f(z\_in_j) \quad (2.3)$$

Dan kirimkan sinyal tersebut ke semua unit di lapisan atasnya (unit-unit *output*).

c. Tiap-tiap unit *output* (Y_k, k = 1,2,3,..,m) menjumlahkan sinyal-sinyal *input* terbobot dari unit tersembunyi (y_in_k) :

$$y\_in_k = W_{0k} + \sum_{i=1}^{p} z_i w_{jk} \quad (2.4)$$

Keterangan :

W_{0k} : nilai bobot dari bias ke unit *output.*

W_{jk} : nilai bobot dari unit tersembunyi j ke unit *output* k.

m : jumlah unit *output.*

Gunakan fungsi aktivasi untuk menghitung sinyal *output* (y_k) :

$$y_k = f(y\_in_k) \quad (2.5)$$

Dan kirimkan sinyal tersebut ke semua unit di lapisan atasnya (unit-unit *output*).

Backpropagation :

d. Tiap-tiap unit *output* (Y_k, k=1,2,3,…,m) menerima target pola yang berhubungan dengan pola *input* pembelajaran, hitung informasi *error*-nya (δ_k) :

$$\delta_k = (t_k - y_k)f'(y\_in_k) \quad (2.6)$$

Keterangan :

t_k : target pola pada *output* k.

$f'(y\_in_k)$: turunan dari fungsi aktivasi.

Kemudian hitung koreksi bobot dari unit-unit tersembunyi j ke unit-unit *output* k (Δw_{jk}) :

$\Delta w_{jk} = \alpha\ \delta_k\ z_j$ (2.7)

Keterangan :

α : nilai *learning rate.*

Hitung juga koreksi bias ke unit-unit *output* k (Δw_{0k}) :

$\Delta w_{0k} = \alpha\ \delta_k$ (2.8)

Kirimkan informasi *error* (δ_k) ke unit-unit yang ada di lapisan bawahnya.

e. Tiap-tiap unit tersebunyi (Z_j, j = 1,2,3,…,p) menjumlahkan delta *input*-nya (δ_in_j) dari unit-unit yang berada pada lapisan di atasnya :

$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk}$ (2.9)

Kalikan nilai ini dengan turunan dari fungsi aktivasi untuk menghitung informasi error :

$\delta_j = \delta\_in_j f'(z\_in_j)$ (2.10)

Kemudian hitung koreksi bobot dari unit-unit *input* i ke unit-unit tersembunyi j (Δv_{ij}) :

$\Delta v_{ij} = \alpha \delta_j x_i$ (2.11)

Hitung juga koreksi bias ke unit-unit tersembunyi j (Δv_{0j}) :

$\Delta v_{0j} = \alpha \delta_j$ (2.12)

f. Tiap-tiap unit *output* (Y_k, k = 1,2,3,…,m) memperbaiki bias dan bobotnya (j=0,1,2,…,p) :

$W_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk}$ (2.13)

Tiap–tiap unit tersebunyi (z_j, j = 1,2,3,…,p) memperbaiki bias dan bobotnya (i=0,1,2,…,n) :

$V_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij}$ (2.14)

3. Tes kondisi berhenti, kondisi berhenti terpenuhi bila proses *feedforward* dan *backpropagation* telah diulang sebanyak *max epoch* atau *error* pada unit *output* telah memenuhi batas yang ditentukan.

2.1.3. Algoritma *Backpropagation* Dengan Metode *Resilient Bacpropagation.*

Untuk mendapatkan bobot yang dapat menghasilkan nilai error yang optimal, Redmiller dan Braun (Rojas, 1996) mengusulkan metode Resilient Backpropagation. Ide dari algoritma Resilient Backpropagation adalah dengan mencari bobot yang optimal hanya dengan menggunakan parameter pembelajaran berupa nilai *learning rate* dan tanda dari turunan pertama fungsi error terhadap nilai bobot.

Metode ini terbukti dapat mempercepat proses pembelajaran dan mendapatkan nilai error yang minimum.

Proses pembelajaran pada algoritma backpropagation dapat dioptimalisasi lagi dengan menggunakan fungsi aktivasi pada persamaan 2.15. (LeCun, 1998)

$$f(x) = 1.7159 \tanh(\tfrac{2}{3}x) \qquad (2.15)$$

yang mempunyai turunan :

$$f'(x) = \frac{1.7159(\frac{2}{3})}{(\cosh(\frac{2}{3}x))^2} \qquad (2.16)$$

Algoritma Resilient backpropagation menggunakan beberapa parameter diantaranya adalah nilai minimum *learning rate* (γ_{min}) dan nilai *learning rate* maksimum (γ_{max}) agar proses pembelajaran tidak terlalu cepat atau terlalu lambat. Berikut adalah algoritma backpropagation menggunakan metode Resilient Backpropagation (Nissen, 2006) :

1. Inisialisasi tiap-tiap bobot dengan nilai acak yang cukup kecil (mendekati nol) yaitu berkisar diantara -0.05 sampai 0.05.
2. Inisialisasi nilai turunan pertama fungsi error terhadap bobot pada input ke *hidden layer* (ΔE_{ij}) dan nilai turunan pertama fungsi error terhadap bobot pada *hidden* ke *output layer* (ΔE_{jk}) sama dengan nol. Kemudian inisialisasi nilai turunan pertama fungsi error terhadap bobot pada input ke *hidden layer* sebelumnya (ΔE_{ij}lama) dan nilai turunan pertama fungsi error terhadap bobot pada *hidden* ke *output layer* sebelumnya (ΔE_{jk}lama) sama dengan nol.
3. Inisialisasi nilai *learning rate* sebelumnya pada input ke *hidden layer* (γ_{ij}lama) dan nilai *learning rate* pada *hidden* ke *output layer* (γ_{jk}lama) dengan nilai yang cukup kecil misalnya 0.1. Kemudian inisialisasi *learning rate* maksimum γ_{max} (misalnya 50) dan *learning rate* minimum $\gamma_{min.}$ (misalnya 0).
4. Untuk tiap-tiap elemen lakukan proses normalisasi input dengan persamaan :

$$x' = \frac{0.8(x - minvalue)}{maxvalue - minvalue} + 0.1 \qquad (2.17)$$

5. Untuk tiap-tiap pasangan elemen yang akan dilakukan pembelajaran, kerjakan :
   a. Proses feedforward seperti pada backpropagation standar menggunakan persamaan 2.2 sampai 2.5.
   b. Menghitung nilai informasi error (δ_k) pada tiap-tiap unit output :

$$\delta_k = f'(y_k)\frac{1}{2}(t_k - y_k) \quad (2.18)$$

   c. Menghitung nilai informasi error (δj) pada tiap-tiap unit *hidden* dengan melakukan proses backpropagation terhadap error pada output (δ_k).

$$\delta j = f'(z_j)\sum_k \delta_k w_{jk} \quad (2.19)$$

   d. Menghitung nilai turunan pertama fungsi error terhadap bobot pada input ke *hidden layer* (ΔE_{ij})

$$\Delta E_{ij} = \Delta E_{ij} + \delta_j x_i \quad (2.20)$$

   e. Menghitung nilai turunan pertama fungsi error terhadap bobot pada *hidden* ke *output layer* (ΔE_{jk})

$$\Delta E_{jk} = \Delta E_{jk} + \delta_k z_j \quad (2.21)$$

6. Menghitung nilai *learning rate* yang baru pada bobot antara input ke *hidden layer* (γ_{ij}).

   Jika ΔE_{ij}lama $\cdot \Delta E_{ij} \geq 0$ maka

$$\gamma_{ij} = \min(\gamma_{ij}\text{lama}\ u, \gamma_{max}) \quad (2.22)$$

   Jika tidak maka

$$\gamma_{ij} = \max(\gamma_{ij}\text{lama}\ d, \gamma_{min})$$
$$\Delta E_{ij} = 0$$

   Dimana *u* dan *d* adalah konstanta yang nilainya $u > 1$ dan $d < 1$. (misalnya nilai u sama dengan 1.2 dan nilai d sama dengan 0.5). Konstanta u disebut juga *learning rate increasor* sedangkan konstanta d disebut juga *learning rate decreasor.*

7. Perbarui nilai bobot antara input ke *hidden layer* (v_{ij})

   Jika $\Delta E_{ij} < 0$ maka

$$v_{ij} = v_{ij}\text{ (lama)} - \gamma_{ij} \quad (2.23)$$

   Jika tidak maka

$$v_{ij} = v_{ij}\text{ (lama)} + \gamma_{ij}$$

8. Perbarui nilai *learning rate* sebelumnya (γ_{ij}lama) dan nilai turunan pertama fungsi error terhadap bobot pada input ke *hidden layer* sebelumnya (ΔE_{ij} lama)

γ_{ij}lama = γ_{ij} (2.24)
ΔE_{ij} lama = ΔE_{ij}
$\Delta E_{ij} = 0$

9. Menghitung nilai *learning rate* yang baru pada bobot antara *hidden* ke *output layer* (γ_{jk}).
   Jika ΔE_{jk}lama $\cdot$ $\Delta E_{jk} \geq 0$ maka
   $\gamma_{jk} = \min(\gamma_{jk}\text{lama}\ u, \gamma_{max})$ (2.25)
   Jika tidak maka
   $\gamma_{jk} = \max(\gamma_{jk}\text{lama}\ d, \gamma_{min})$
   $\Delta E_{jk} = 0$
   Dimana *u* dan *d* adalah konstanta yang nilainya $u > 1$ dan $d < 1$. (misalnya nilai u sama dengan 1.2 dan nilai d sama dengan 0.5)
7. Perbarui nilai bobot antara *hidden* ke *output layer* (v_{jk})
   Jika $\Delta E_{jk} < 0$ maka
   $w_{jk} = w_{jk}$ (lama) - γ_{jk} (2.26)
   Jika tidak maka
   $w_{jk} = w_{jk}$ (lama) + γ_{jk}
8. Perbarui nilai *learning rate* sebelumnya (γ_{ij}lama) dan nilai turunan pertama fungsi error terhadap bobot pada *hidden* ke *output layer* sebelumnya (ΔE_{jk} lama)
   γ_{jk}lama = γ_{jk} (2.27)
   ΔE_{jk} lama = ΔE_{jk}
   $\Delta E_{jk} = 0$

2.2. Nilai Tukar Mata Uang Asing

Forex, atau *foreign exchange*, adalah nilai tukar mata uang sebuah negara terhadap mata uang negara lain. Menurut ensiklopedi Britannica (www.britannica.com), *foreign exchange* berarti pembelian atau penjualan dari mata uang suatu negara sebagai pertukaran terhadap mata uang negara lain, dengan nilai yang tergantung pada pasar. *Foreign exchange* memungkinkan transaksi internasional seperti impor ekspor dan pergerakan modal antar negara. Nilai mata uang sebuah negara terhadap negara lain ditetapkan dengan *exchange rate* (nilai tukar).

Sedangkan menurut P. Einzig (1970), *foreign exchange* adalah metode dan instrumen yang digunakan untuk menyesuaikan pembayaran hutang antara dua negara yang menggunakan sistem mata uang yang berbeda. Kemampuan finansial sebuah negara

memegang peranan penting dalam menentukan nilai tukar mata uangnya. Nilai tukar mata uang adalah harga dari mata uang lokal terhadap sebuah mata uang asing, dan nilai ini ditentukan oleh persediaan dan permintaan atas mata uang pada pasar mata uang internasional. Pembelian dan penjualan mata uang asing untuk mendapatkan keuntungan atas perubahan secara tiba tiba dari nilai tukar mata uang ini disebut sebagai *arbitrage*.

Ketika seseorang membicarakan keuntungan atau kerugian atas forex, dia membicarakan kenaikan dan penurunan nilai sebuah investasi yang disebabkan semata-mata oleh pergerakan mata uang. sebagai contoh, jika seorang investor berpikir mata uang dollar amerika lemah, dia mungkin akan membeli mata uang jerman. Account investor tersebut mungkin akan mendapatkan 3% bunga tahunan, akan tetapi kerugian atau keuntungan sesungguhnya dipengaruhi oleh nilai DM (German Mark) terhadap US$ (United State Dollar). Jika investor tersebut menabung mata uang DM selama setahun penuh, dan jika mata uang DM naik 5% terhadap dolar, maka nilai investasinya bukan hanya naik 3%, akan tetapi 5% dari investasi pokok ditambah dengan 3% bunga.

2.3. Peramalan Nilai Mata Uang Asing Menggunakan Jaringan Syaraf Tiruan

Dalam peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan, data yang digunakan adalah data rata-rata penutupan tiap minggu. Nilai rata-rata penutupan yang dipakai dalam proses peramalan mempunyai periode yang berbeda-beda, diantarnya nilai rata-rata penutupan selama seminggu, nilai rata-rata penutupan dalam dua minggu dan seterusnya, sehingga disebut sebagai nilai rata-rata bergerak atau *Moving Average*.

2.3.1. Metode *moving average*

Metode *moving average* mempunyai tiga varian yang berbeda yaitu Simple *Moving Average*, Weighted *Moving Average* dan Exponential *Moving Average*. Masing-masing merupakan metode rata-rata bergerak, hanya saja cara me-rataratakannya yang berbeda satu sama lain. Dalam skripsi ini metode *moving average* yang dipakai adalah Simple *Moving Average* (persamaan 2.28)

berdasarkan penelitian yang dilakukan oleh Yao (2000). Contoh metode simple *moving average* dapat dilihat pada Tabel 2.1.

$$SMA = \frac{PM + PM_{-1} + ... + PM_{-(x-1)}}{x} \qquad (2.28)$$

Ketarangan :
SMA : nilai Simple *moving average* selama x periode
PM : nilai rata-rata periode saat ini
PM_{-x} : nilai rata-rata pada x periode yang lalu

Tabel 2.1 Contoh metode simple *moving average*

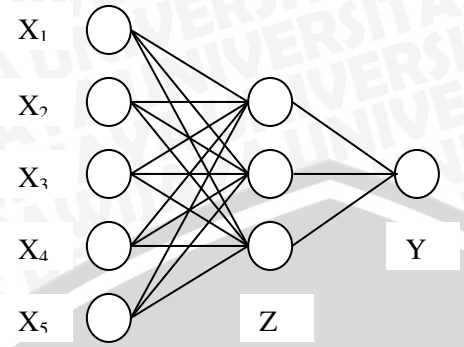
Sampel	SMA 3 periode	SMA 4 periode
23	-	-
24	-	-
25	= (23+24+25)/3 = 24	-
26	= (24+25+26)/3 = 25	(23+24+25+26)/4 = 24.5
27	= (25+26+27)/3 = 26	(24+25+26+27)/4 = 25.5
28	= (26+27+28)/3 = 27	(25+26+27+28)/4 = 26.5
29	= (27+28+29)/3 = 28	(26+27+28+29)/4 = 27.5
30	= (28+29+30)/3 = 29	(27+28+29+30)/4 = 28.5

Dalam penelitian yang telah dilakukan oleh Yao (2000) digunakan nilai rata-rata bergerak pada beberapa periode yaitu selama 5 hari, 10 hari, 20 hari, 60 hari dan 120 hari. Dalam hal ini nilai yang dirata-rata adalah nilai harga penutupan.

2.3.2. Arsitektur jaringan syaraf tiruan untuk peramalan mata uang asing

Arsitektur jaringan syaraf tiruan untuk peramalan mata uang asing terdiri atas 5 unit input dan 1 target. Perlu diingat bahwa pasar mata uang asing dibuka mulai hari senin sampai jumat (5 hari). Tiap-tiap unit input nantinya akan diisi oleh nilai *moving average* selama beberapa periode yaitu 5 hari yang lalu (minggu lalu), 10 hari yang lalu (2 minggu yang lalu), 20 hari yang lalu, 60 hari yang lalu dan 120 hari yang lalu. Sedangkan yang dijadikan sebagai target adalah

nilai *moving average* pada minggu ini (Yao, 2000) Gambar 2.3 adalah arsitektur jaringan syaraf tiruan untuk peramalan mata uang asing.



Gambar 2.3 Arsitektur jaringan syaraf tiruan untuk peramalan nilai mata uang

Keterangan :
X_1 : input pertama yaitu nilai *moving average* 5 hari yang lalu
X_2 : input kedua yaitu nilai *moving average* 10 hari yang lalu
X_3 : input ketiga yaitu nilai *moving average* 20 hari yang lalu
X_4 : input keempat yaitu nilai *moving average* 60 hari yang lalu
X_5 : input kelima yaitu nilai *moving average* 120 hari yang lalu
Z : *hidden* unit (misalnya 3 unit)
Y : target yaitu nilai *moving average* minggu ini

Menurut Kamruzzaman (2003) untuk mengetahui keberhasilan suatu arsitektur jaringan syaraf tiruan dalam proses peramalan nilai mata uang dapat diukur melalui nilai *Normalized Mean Squared Error* atau NMSE (persamaan 2.29) , nilai *Mean Absolute Error* atau MAE (persamaan 2.30) dan *Directional Symmetry* atau DS (persaamaan 2.31).

$$NMSE = \frac{1}{\sigma^2 N}\sum_k (t_k - y_k)^2 \qquad (2.29)$$

σ^2 = nilai variasi data

N = adalah jumlah data

t_k = target

y_k= nilai hasil peramalan

Nilai NMSE digunakan untuk mengukur kesalahan (selisih) rata-rata hasil peramalan dengan data asli.

$$MAE = \frac{1}{N} | t_k - y_k | \qquad (2.30)$$

Nilai MAE digunakan untuk mengukur kesalahan (selisih) mutlak antara nilai hasil peramalan dengan nilai data asli.

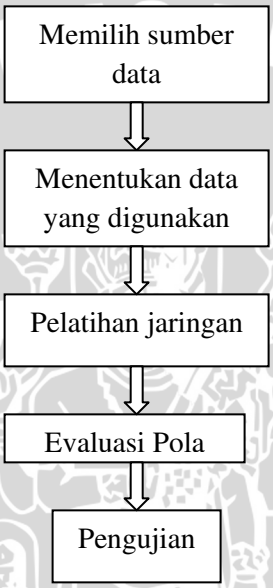
BAB III
METODOLOGI DAN PERANCANGAN

3.1 Analisa Permasalahan

Pada bab metodologi dan perancangan ini akan dibahas tentang langkah-langkah dalam proses peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation.* Langkah-langkahnya adalah sebagai berikut :

1. Memilih sumber data.
2. Menentukan data apa saja yang dipakai.
3. Pelatihan jaringan.
4. Evaluasi pola.
5. Pengujian.

Yang dapat digambarkan sebagai berikut:



Gambar 3.1. Blok diagram penelitian

3.1.1 Sumber Data

Dalam penelitian ini data diambil dari web site Bank Canada (http://www.bankofcanada.ca/en/rates/exchange-look.html), yaitu nilai tukar mata uang dollar united states terhadap mata uang dollar canada. Data nilai tukar mata uang yang digunakan diambil mulai 2 januari 1998 sampai 2 maret 2009 dan yang dipakai hanya nilai penutupan dalam satu hari.

3.1.2 Data yang Digunakan

Data yang digunakan dalam skripsi ini adalah data nilai penutupan selama 11 tahun. Data tersebut kemudian dibagi menjadi data latih dan data evaluasi. Data latih yang dipergunakan adalah sebanyak 500 pola sedangkan data evaluasi yang dipakai sebanyak 50 pola.

3.1.3 Pembelajaran

Pembelajaran dilakukan menggunakan algoritma jaringan syaraf tiruan resilient backpropagation untuk mendapatkan nilai bobot yang optimal dalam memprediksi nilai *moving average* pada minggu yang akan datang. Data yang digunakan selama pembelajaran adalah data selama 516 minggu.

Dalam proses pembelajaran pada algoritma jaringan syaraf tiruan resilient backpropagation diperlukan beberapa pola sebagai bahan pembelajaran. Pola pertama dimulai dengan mengambil data selama 125 hari yang pertama untuk kemudian dicari nilai *moving average* pada 5 hari yang lalu (minggu lalu), 10 hari yang lalu (2 minggu yang lalu), 20 hari yang lalu, 60 hari yang lalu, 120 hari yang lalu dan nilai *moving average* pada minggu ini. Pada pola yang kedua data yang digunakan adalah mulai data hari ke 5 sampai data pada hari 130 (125 hari) untuk kemudian dicari nilai *moving average* pada 5 hari yang lalu (minggu lalu), 10 hari yang lalu (2 minggu yang lalu), 20 hari yang lalu, 60 hari yang lalu, 120 hari yang lalu dan nilai *moving average* pada minggu ini. Pada setiap pola, nilai *moving average* yang dijadikan sebagai input pada jaringan syaraf tiruan adalah nilai *moving average* selama 5 hari yang lalu (minggu lalu), 10 hari yang lalu (2 minggu yang lalu), 20 hari yang lalu, 60 hari yang lalu dan 120 hari yang lalu, sedangkan yang dijadikan sebagai

target adalah nilai *moving average* pada minggu ini. Nilai *moving average* tiap periode dihitung dengan menggunakan persamaan 2.28. Demikian proses pembuatan pola tersebut berlangsung sampai mencapai 516 minggu (500 pola).

Setelah didapatkan semua pola, kemudian dilanjutkan dengan proses pembelajaran dengan algoritma jaringan syaraf tiruan resilient backpropagation seperti yang dijelaskan pada bab 2.1.3. Pada akhir setiap epoch dalam proses pembelajaran diukur nilai kesalahan yang terjadi yaitu nilai Normalized Mean Squared Error atau NMSE (persamaan 2.29) , nilai Mean Absolute Error atau MAE (persamaan 2.30).

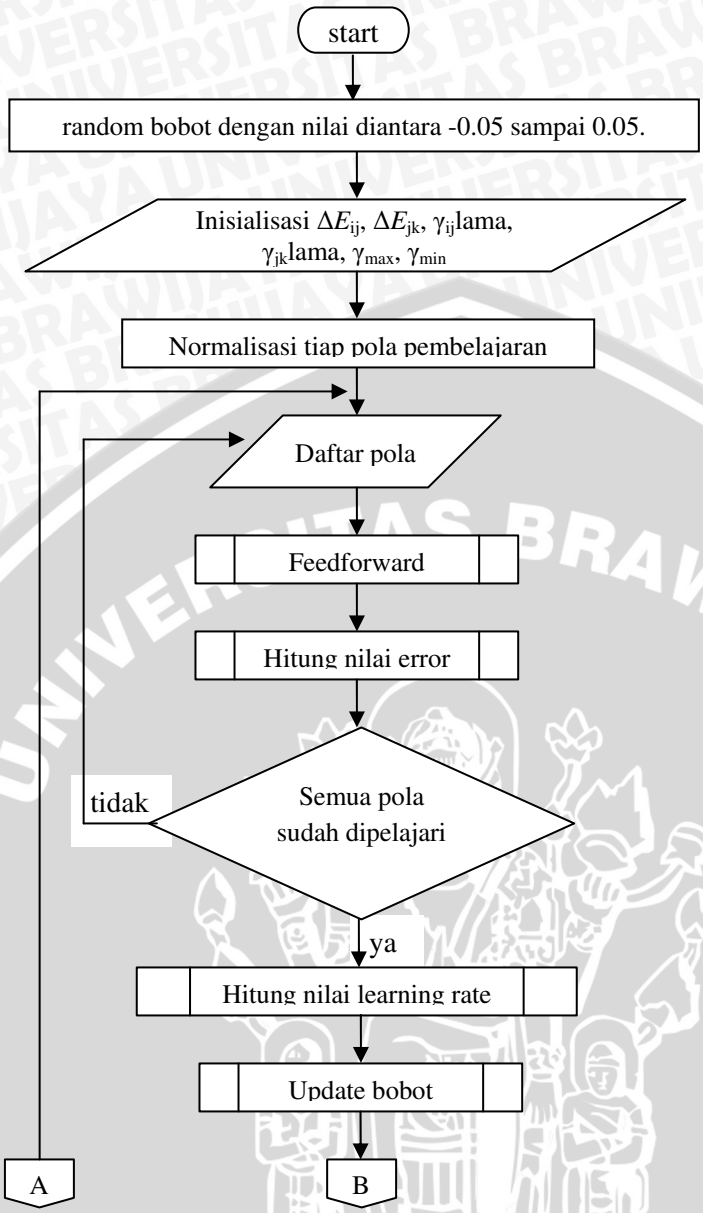
Proses pembelajaran dilakukan selama beberapa epoch dan dicari jumlah unit pada *hidden layer* yang dapat meminimalkan nilai error pada persamaan 2.29 sampai 2.30. Setelah didapatkan jumlah unit pada *hidden layer* yang optimal kemudian dilajutkan dengan proses pengujian.

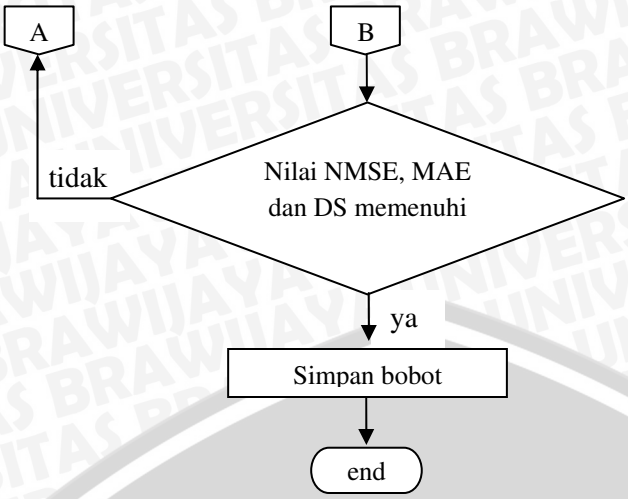
3.1.5 Pengujian

Langkah pengujian ini dilakukan untuk menguji arsitektur jaringan yaitu jumlah unit pada *hidden layer* serta nilai bobot yang telah dihasilkan pada tahap pembelajaran. Proses pengujian dilakukan terhadap pola yang belum pernah dilakukan proses pembelajaran sebelumnya yaitu sebanyak 66 minggu (50 pola). Dari proses pengujian didapatkan hasil keakuratan arsitektur jaringan syaraf tiruan dalam memprediksi nilai *moving average* pada minggu yang akan datang.

3.2 *Flowchart* Proses Pembelajaran Jaringan Syaraf Tiruan Resilient *Backpropagation.*

Pada gambar 3.2 adalah *flowchart* proses pembelajaran dengan metode jaringan syaraf tiruan resilient *backpropagation* yang telah dijelaskan pada subbab 2.1.3. Pada *flowchart* ini terdapat 4 *predifined process* yang akan dijabarkan satu persatu.

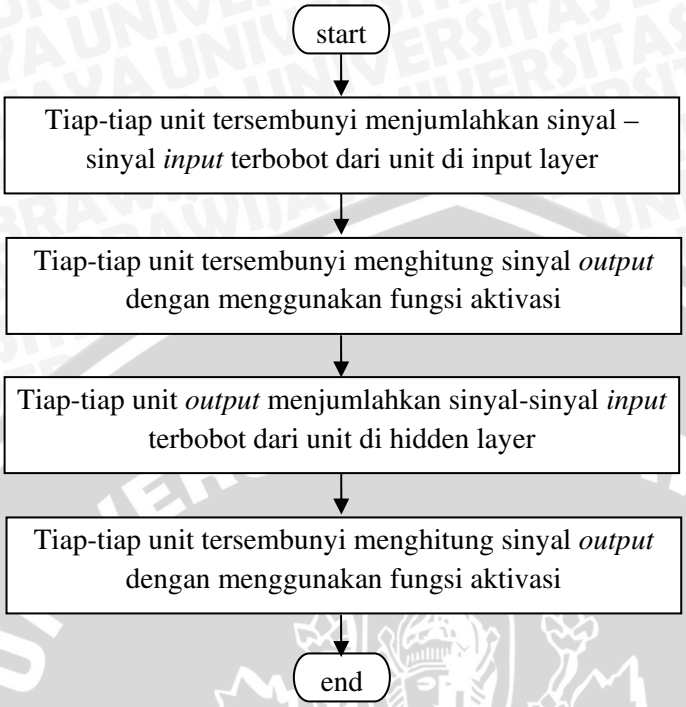

start
random bobot dengan nilai diantara -0.05 sampai 0.05.
Inisialisasi ΔE_{ij}, ΔE_{jk}, γ_{ij}lama, γ_{jk}lama, γ_{max}, γ_{min}
Normalisasi tiap pola pembelajaran
Daftar pola
Feedforward
Hitung nilai error
Semua pola sudah dipelajari
tidak
ya
Hitung nilai learning rate
Update bobot
A
B



Gambar 3.2 *Flowchart* algoritma JST *backpropagation*

Masukan dari proses pembelajaran pada gambar 3.2 adalah daftar pola yang diperoleh dari proses pembuatan pola dan telah dijelaskan pada subbab 3.13. Proses pembelajaran dilakukan sampai nilai NMSE dan MAE telah memenuhi. Hasil dari proses pembelajaran adalah nilai bobot yang dapat menghasilkan nilai NMSE dan MAE yang minimum. Nilai bobot tersebut akan digunakan pada proses pengujian.

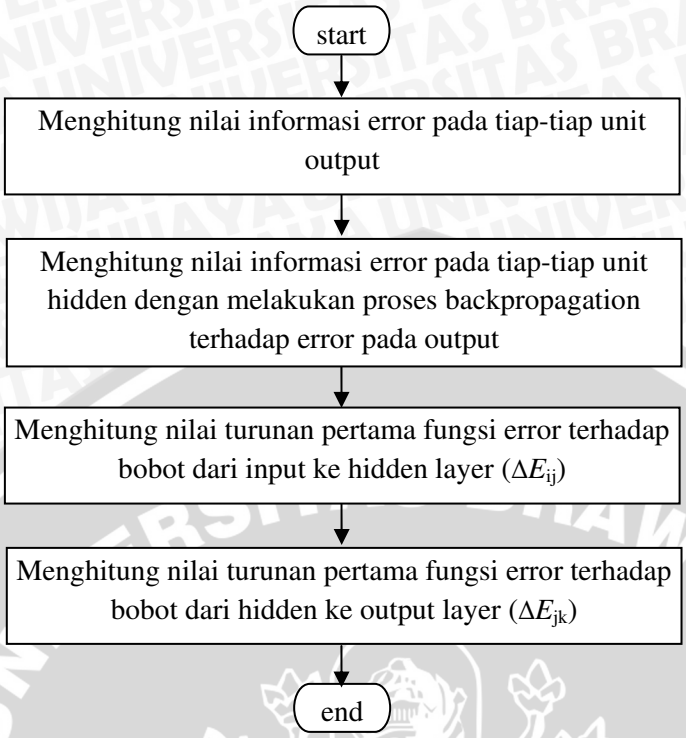
3.2.2 Blok Diagram Proses *Feedforward*



Gambar 3.3 Blok diagram proses *feedforward*

Proses feedforward pada gambar 3.3 digunakan untuk menghitung nilai output dari tiap-tiap unit di setiap *layer*. Persamaan yang digunakan dalam proses *feedforward* adalah persamaan 2.2 sampai 2.5. Untuk mengoptimalkan proses pembelajaran, fungsi aktivasi yang dipakai dalam skripsi ini adalah fungsi aktivasi pada persamaan 2.15 yang mempunyai turunan pertama pada persamaan 2.16.

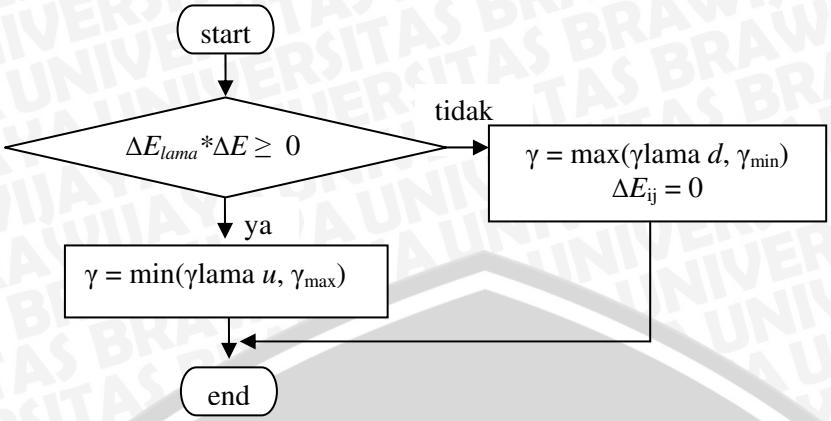
3.2.3 Blok Diagram Proses Hitung Error



Gambar 3.4 Blok diagram proses hitung *error*

Proses hitung error pada gambar 3.4 dilakukan dengan menggunakan persamaan pada.2.18 sampai 2.21. Proses hitung error digunakan untuk mendapatkan nilai turunan pertama fungsi error terhadap bobot pada tiap-tiap unit di setiap *layer*. Nilai turunan pertama tersebut kemudian digunakan untuk menentukan nilai *learning rate*.

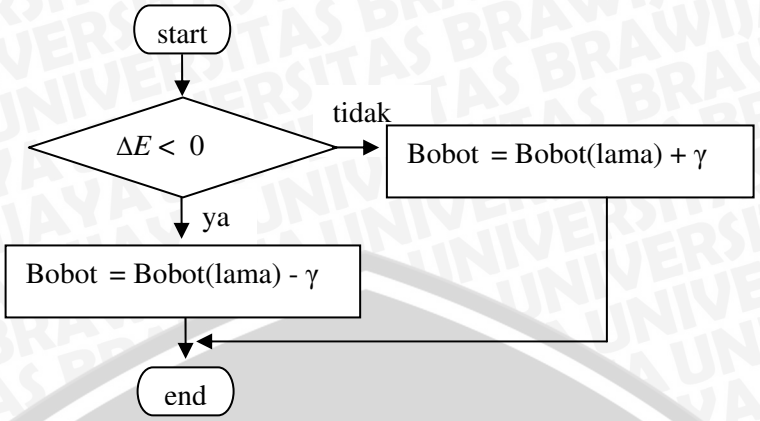
3.2.4 *Flowchart* Proses Hitung *Learning rate*



Gambar 3.5 *flowchart* proses hitung *learning rate*

Proses hitung *learning rate* (γ) pada gambar 3.5 dilakukan dengan menggunakan persamaan pada.2.22 dan 2.25. Proses hitung *learning rate* digunakan untuk menentukan besar kecil nilai *learning rate* sesuai dengan nilai turunan pertama fungsi error terhadap bobot pada epoch yang sekarang dengan epoch sebelumnya.

3.2.4 *Flowchart* Proses Hitung Bobot



Gambar 3.6 *flowchart* proses hitung bobot

Proses hitung bobot pada gambar 3.6 dilakukan dengan menggunakan persamaan pada.2.23 dan 2.26. Proses hitung bobot dilakukan untuk mendapatkan bobot dengan nilai error yang minimum. Besarnya perubahan nilai bobot dipengaruhi oleh besar *learning rate* (γ) sedangkan arah perubahan bobotnya dipengaruhi oleh nilai turunan pertama fungsi error terhadap bobot (Δ*E*).

3.3 Rancangan Tabel

Tabel 3.1 Tabel TPembelajaran

Field	Type	Keterangan
LearningRate	Text	Nilai *learning rate*
*Hidden*Unit	Integer	Jumlah unit pada *hidden layer*
MaxEpoch	Integer	Jumlah perulangan
TargetError	Double	*Error* yang minimum

Tabel TPembelajaran (Tabel 3.1) digunakan untuk menyimpan parameter-parameter pembelajaran. Tabel TPembelajaran diperlukan pada saat proses pembelajaran.

Tabel 3.2 Tabel TBobot

Field	Type	Keterangan
IDBobot	Integer (*primary key*)	Contoh: 1
ID*Layer*	Integer	Contoh: 1
DariNeuron	Integer	Contoh: 1
KeNeuron	Integer	Contoh: 2
Bobot	Double	Contoh: 0.56

Tabel TBobot (Tabel 3.2) digunakan untuk menyimpan nilai bobot pada tiap *layer* yang dihasilkan setelah proses pembelajaran. Nilai bobot yang disimpan adalah nilai bobot yang dapat menghasilkan nilai error minimum.

Tabel 3.3 Tabel TPenutupan

Field	Type	Keterangan
ID	Integer (*primary key*)	Contoh: 1
Tgl	DateTime	Contoh: 1/3/2004
NilaiPenutupan	Double	Contoh: 100.5

Tabel TPenutupan (Tabel 3.3) digunakan untuk menyimpan nilai penutupan nila tukar mata uang dollar united states terhadap mata uang dollar canada. Data dalam tabel TPenutupan akan dipakai pada saat proses pembelajaran dan pengujian.

Tabel 3.4 Tabel TPola

Field	Type	Keterangan
ID	Integer (*primary key*)	Contoh: 1
nextweekdate	DateTime	Contoh: 1/3/2004
Last120date	DateTime	Contoh: 1/3/2004
T	Double	Contoh: 0,001287
X1	Double	Contoh: 0,001287
X2	Double	Contoh: 0,001287
X3	Double	Contoh: 0,001287
X4	Double	Contoh: 0,001287

X5	Double	Contoh: 0,001287

Tabel TPola (Tabel 3.4) digunakan untuk menyimpan nilai *moving average* setiap pola yang dihasilkan dari penghitungan pola. Data dalam tabel TPola akan dipakai pada saat proses pembelajaran dan pengujian.

Tabel 3.5 Tabel TNormalizedpola

Field	Type	Keterangan
ID	Integer (*primary key*)	Contoh: 1
counter	Integer	Contoh: 1
startdate	DateTime	Contoh: 1/3/2004
nextweekdate	DateTime	Contoh: 1/3/2004
Last120date	DateTime	Contoh: 1/3/2004
T	Double	Contoh: 0,001287
X1	Double	Contoh: 0,001287
X2	Double	Contoh: 0,001287
X3	Double	Contoh: 0,001287
X4	Double	Contoh: 0,001287
X5	Double	Contoh: 0,001287

Tabel TNormalizedpola (Tabel 3.5) digunakan untuk menyimpan nilai *moving average* setiap pola dari tabel TPola yang telah dinormalisasi. Data dalam tabel TNormalizedpola akan dipakai pada saat proses pembelajaran dan pengujian.

Tabel 3.5 Tabel TNormalizedpola_nolearn

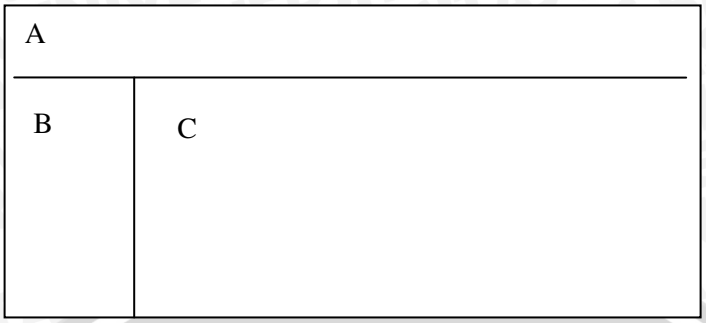
Field	Type	Keterangan
ID	Integer (*primary key*)	Contoh: 1
counter	Integer	Contoh: 1
startdate	DateTime	Contoh: 1/3/2004
nextweekdate	DateTime	Contoh: 1/3/2004
Last120date	DateTime	Contoh: 1/3/2004
T	Double	Contoh: 0,001287
X1	Double	Contoh: 0,001287

X2	Double	Contoh: 0,001287
X3	Double	Contoh: 0,001287
X4	Double	Contoh: 0,001287
X5	Double	Contoh: 0,001287

Tabel TNormalizedpola_nolearn (Tabel 3.5) digunakan untuk menyimpan nilai *moving average* setiap pola dari tabel TPola yang telah dinormalisasi. Data dalam tabel TNormalizedpola akan dipakai pada proses pengujian.

3.4 Rancangan *Interface*

***Form* utama**



Gambar 3.7 rancangan *form* utama

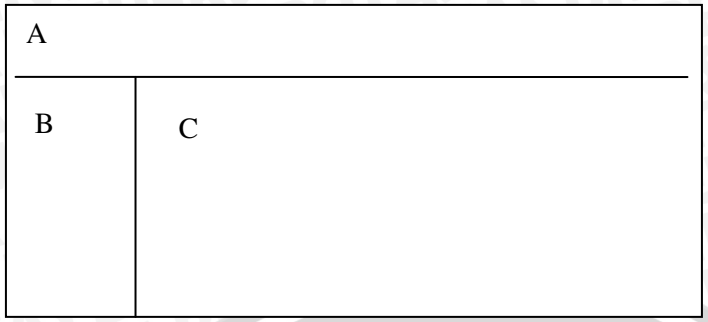
Pada *form* utama ini terdiri dari beberapa bagian antara lain:

A: Untuk Judul

B : Berisi menu yang ada, ada 2 menu yaitu menu training dan pengujian.

C : Untuk menampilkan informasi berdasarkan menu yang telah dipilih oleh user.

***Form training*/ pelatihan**



Gambar 3.8 Rancangan *form training*

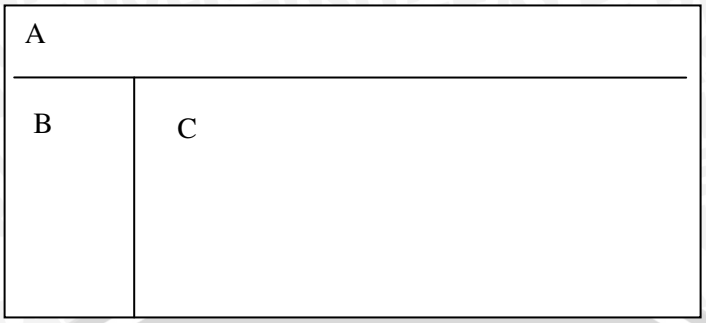
Pada *form* pelatihan terdiri dari 3 bagian yaitu:

A : Untuk judul

B : Navigasi menu

C : Menampilkan html form untuk memasukkan data yang diperlukan pada tahap pelatihan. Data-data tersebut antara lain jumlah *neuron hidden layer, neuron input layer, learning rate,* koefisien momentum, batas error yang diharapkan dan jumlah *epoch*.

***Form* pengujian**



Gambar 3.9 Rancangan *form* Peramalan

Pada *form pelatihan* ini terdiri dari 3 bagian yaitu:

A : Untuk judul

B : Navigasi menu

C : Menampilkan html form untuk melakukan peramalan nilai tukar mata uang dolar amerika yang akan diramalkan.

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

Pada bab ini akan dibahas mengenai implementasi dari perancangan pada bab 3. Dari implementasi tersebut kemudian dibahas mengenai hasil evaluasi dari perangkat lunak yang dihasilkan.

4.1. Lingkungan Implementasi

Lingkungan implementasi meliputi lingkungan perangkat keras serta lingkungan perangkat lunak.

4.1.1. Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan sistem peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation* ini adalah:

1. *Processor* Intel(R) Centrino Duo(R) T5500 1.66 GHz.
2. RAM 512 MB.
3. *Harddisk* dengan kapasitas 80 GB.
4. *Monitor*.
5. *Keyboard*.
6. *Mouse*.

4.1.2. Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan sistem peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation* ini adalah :

1. Sistem Operasi *Microsoft Windows XP Professional Service Pack 2*.
2. *PHP 5*.
3. *MySQL 5*.

4.2. Implementasi Program

Berdasarkan perancangan perangkat lunak pada bab 3 maka pada subbab ini akan dibahas mengenai implementasi dari perancangan tersebut.

4.2.1 Pembuatan Pola

Dalam perangkat lunak ini terdapat proses pembelajaran yang membutuhkan pola pembelajaran berupa nilai moving average pada 5 hari, 10 hari, 20 hari, 60 hari dan 120 hari yang lalu serta *moving average* pada minggu ini. Proses pencarian nilai *moving average* tiap periode dilakukan melalui *code* getpola.php (Gambar 4.1).

```
<?php
include "conf.php";
include "fungsi.php";

  if(!isset($_SESSION[counter]))
     $_SESSION[counter] = 0;

  if(is_null($_SESSION[counter]))
     $_SESSION[counter] = 0;

  $i = $_GET[countx];

   $startd = mktime(0, 0, 0, 1, 7, 2008);
   $startdate = $startd - ($i*7*24*60*60);

  //target ... next 1 week (1 week 7 days)
  $nextweek = $startdate + (7*24*60*60);
  //5 hari yang lalu (1 week 7 days)
  $last5days = $startdate - (7*24*60*60);
  //10 hari yang lalu (2 week 14 days)
  $last10days = $startdate - (14*24*60*60);
  //20 hari yang lalu (4 week 28 days)
  $last20days = $startdate - (28*24*60*60);
  //60 hari yang lalu (12 week 84 days)
  $last60days = $startdate - (84*24*60*60);
  //120 hari yang lalu (24 week 168 days)
  $last120days = $startdate - (168*24*60*60);

  $sqlt = "select avg(close) as avrate from tpenutupan where tgl
  >= '" . date("Y-m-d", $startdate) . "' and tgl < '" . date("Y-
  m-d", $nextweek) . "' ";
  $sql5 = "select avg(close) as avrate from tpenutupan where tgl
  >= '" . date("Y-m-d", $last5days) . "' and tgl < '" . date("Y-
  m-d", $startdate) . "' ";
  $sql10 = "select avg(close) as avrate from tpenutupan where
  tgl >= '" . date("Y-m-d", $last10days) . "' and tgl < '" .
  date("Y-m-d", $startdate) . "' ";
  $sql20 = "select avg(close) as avrate from tpenutupan where
  tgl >= '" . date("Y-m-d", $last20days) . "' and tgl < '" .
  date("Y-m-d", $startdate) . "' ";
  $sql60 = "select avg(close) as avrate from tpenutupan where
  tgl >= '" . date("Y-m-d", $last60days) . "' and tgl < '" .
  date("Y-m-d", $startdate) . "' ";
```

```
   $sql120 = "select avg(close) as avrate from tpenutupan where
   tgl >= '" . date("Y-m-d", $last120days) . "' and tgl < '" .
   date("Y-m-d", $startdate) . "' ";

   $valt = single_q(mysql_query($sql5));
   $val5 = single_q(mysql_query($sql5));
   $val10 = single_q(mysql_query($sql10));
   $val20 = single_q(mysql_query($sql20));
   $val60 = single_q(mysql_query($sql60));
   $val120 = single_q(mysql_query($sql120));

$sqlcek = "select count(id) as c from tpola where id = $i";
$valcek = single_q(mysql_query($sqlcek));
if ($valcek >= 1)
    $sqlin = "update tpola set `startdate` = '" . date("Y-m-d",
    $startdate) . "',`x1` = $val5,`x2` = $val10, `x3`=
    $val20,`x4`= $val60,`x5`= $val120,`t`= $valt where `id` =
    $i";
else
    $sqlin = "insert into tpola(`id`,`startdate`,`x1`, `x2`,
    `x3`, `x4`, `x5`, `t`) values($i, '" . date("Y-m-d",
    $startdate) . "', $val5, $val10, $val20, $val60, $val120,
    $valt) ";

$result = mysql_query($sqlin);
if (!$result) {
    die('Invalid query: ' . mysql_error());
}

    $startdate = date("d/m/y", $startdate);
    $val5 = number_format($val5, 6, '.', '') ;
    $val10 = number_format($val10, 6, '.', '') ;
    $val20 = number_format($val20, 6, '.', '') ;
    $val60 = number_format($val60, 6, '.', '') ;
    $val120 = number_format($val120, 6, '.', '') ;
    $valt = number_format($valt, 6, '.', '') ;

    $polaarray =
"$i,$startdate,$val5,$val10,$val20,$val60,$val120,$valt";
    echo $polaarray;
?>
```

Gambar 4.1 *Code* getpola.php untuk mencari nilai *moving average*

Sebelum dilakukan proses pembelajaran, setiap pola yang terdapat pada tabel tpola perlu dinormalisasi terlebih dahulu. Proses normalisasi pola dilakukan pada *code* normpola.php (Gambar 4.2).

Setiap pola yang telah dinormalisasi kemudian dimasukkan ke dalam tabel tnormalizedpola.

```
<?php

include "conf.php";
include "fungsi.php";

$i = $_GET[countx];

$sqlx = "select * from tpola where id = $i ";
$valx = mysql_query($sqlx);

while($rc = mysql_fetch_assoc($valx))
{
        $x1 = $rc[x1];
        $x2 = $rc[x2];
        $x3 = $rc[x3];
        $x4 = $rc[x4];
        $x5 = $rc[x5];
        $t = $rc[t];
        $startdate = strtotime($rc[nextweekdate]);
}

$newx1 = normalize($x1);
$newx2 = normalize($x2);
$newx3 = normalize($x3);
$newx4 = normalize($x4);
$newx5 = normalize($x5);
$newt = normalize($t);

$sqlcek = "select count(id) as c from tnormalizedpola where id =
$i";
$valcek = single_q(mysql_query($sqlcek));



if ($valcek >= 1)

     $sqlin = "update tnormalizedpola set `startdate` = '" .
     date("Y-m-d", $startdate) . "',`x1` = $newx1,`x2` = $newx2,
     `x3`= $newx3,`x4`= $newx4,`x5`= $newx5,`t`= $newt where
     `id` = $i";
```

```
else

    $sqlin = "insert into
    tnormalizedpola(`counter`,`startdate`,`x1`,`x2`,`x3`,`x4`,`
    x5`,`t`) values($i, '" . date("Y-m-d", $startdate) . "',
    $newx1, $newx2, $newx3, $newx4, $newx5, $newt) ";

$result = mysql_query($sqlin);

if (!$result) {
     die('Invalid query: ' . mysql_error());
}

$startdate = date("d/m/y", $startdate);
$val1 = number_format($newx1, 6, '.', '') ;
$val2 = number_format($newx2, 6, '.', '') ;
$val3 = number_format($newx3, 6, '.', '') ;
$val4 = number_format($newx4, 6, '.', '') ;
$val5 = number_format($newx5, 6, '.', '') ;
$valt = number_format($newt, 6, '.', '') ;

$polaarray =
"$i,$startdate,$val1,$val2,$val3,$val4,$val5,$valt";

echo $polaarray;
?>
```

Gambar 4.2 *code* normpola.php untuk normalisasi pola

4.2.2 Implementasi Proses Pembelajaran Jaringan Syaraf Tiruan

Resilient Backpropagation

Setelah semua pola telah ternormalisasi kemudian dilanjutkan dengan proses pembelajaran. Proses pembelajaran dilakukan pada fungsi *Learning* (Gambar 4.4).

```
<?php
session_start();

include "conf.php";
include "fungsi.php";
```

```
/*************************************************
        Total Error
lakukan penghitungan total error
*************************************************/

$nMSE = 0;
$m_dNMSE = 0;
$m_dMAE = 0;

$sqlx2 = "select * from tnormalizedpola_nolearn where 1 order by
id asc limit 0,100";
$valx2 = mysql_query($sqlx2);

while($rc2 = mysql_fetch_assoc($valx2))
{
  $x_1 = $rc2[x1];
  $x_2 = $rc2[x2];
  $x_3 = $rc2[x3];
  $x_4 = $rc2[x4];
  $x_5 = $rc2[x5];
  $t = $rc2[t];

  $input['x'][] = $x_1;
  $input['x'][] = $x_2;
  $input['x'][] = $x_3;
  $input['x'][] = $x_4;
  $input['x'][] = $x_5;
  $input['t'] = $t;

  $hasil2 = feedfwd($input);

  //hitung output error

  for($i=0;$i<$_SESSION['neuron']['output'];$i++)
  {
     //error pada tiap output

     $m_dNMSE += Pow($input['t'] - $hasil2['y'][$i], 2);
     $m_dMAE += Abs($input['t'] - $hasil2['y'][$i]);
     $nMSE++;
  }

}
$m_dNMSE /= $nMSE;
$m_dMAE /= $nMSE;

if($m_dMinNMSE>$m_dNMSE){
  $m_dMinNMSE = $m_dNMSE;

  if($m_dMAE<$m_dMinMAE){
     $m_dMinMAE = $m_dMAE;
     $_SESSION['bobot']['w_BestWeight']                    =
$_SESSION['bobot']['w_Weight'] ;
     $_SESSION['bobot']['v_BestWeight']                    =
```

```
$_SESSION['bobot']['v_Weight'] ;
  }
}

 if($m_dNMSE <  $_GET[targetError]){
    $learningStatus += $_GET[countx] . " Convergen Best NMSE : "
. $m_dMinNMSE . " MAE : " . $m_dMinMAE ;
   }

$polaarray = "$learningStatus $_GET[countx],$m_dNMSE,$m_dMAE";


/***************************************************
        end of Total Error
***************************************************/

$i = $_GET[countx];

//get pola
$sqlx = "select * from tnormalizedpola order by id asc ";
$valx = mysql_query($sqlx);

while($rc = mysql_fetch_assoc($valx))
{
  $x1 = $rc[x1];
  $x2 = $rc[x2];
  $x3 = $rc[x3];
  $x4 = $rc[x4];
  $x5 = $rc[x5];
  $t = $rc[t];

  $input['x'][] = $x1;
  $input['x'][] = $x2;
  $input['x'][] = $x3;
  $input['x'][] = $x4;
  $input['x'][] = $x5;

  $hasil = feedfwd($input);

  $m_error['output'] = CalculateMSE($input);
  $m_error['hidden'] = BackpropagateMSE($input);
  UpdateSlope($input);
}


/***************************************************
         update weight
***************************************************/
for($j=0;$j<$_SESSION['neuron']['hidden'];$j++)
{
  for($i=0;$i<$_SESSION['neuron']['input'];$i++)
  {
    $dPrevStep                                          =
Max($_SESSION['bobot']['v_PrevDeltaWeight'][$i][$j],0.0001);
```

```
    $dSlope = $_SESSION['bobot']['v_ErrorWeight'][$i][$j];
    $dPrevSlope                                                    =
$_SESSION['bobot']['v_PrevErrorWeight'][$i][$j];
    $dX = $dPrevSlope * $dSlope;
    $dNextStep = 0;

    if ($dX >= 0.0)
    {
       $dNextStep    =    Min($dPrevStep    *    $_GET['inc_LR'],
$_GET['max_LR']);
    } else {
       $dNextStep    =    Max($dPrevStep    *    $_GET['dec_LR'],
$_GET['min_LR']);
       $dSlope = 0;
    }

    if($dSlope<0)
    {
       $_SESSION['bobot']['v_Weight'][$i][$j] -= $dNextStep;

    } else {
       $_SESSION['bobot']['v_Weight'][$i][$j] += $dNextStep;

    }

    $_SESSION['bobot']['v_PrevDeltaWeight'][$i][$j]                =
$dNextStep;
    $_SESSION['bobot']['v_PrevErrorWeight'][$i][$j] = $dSlope;
    $_SESSION['bobot']['v_ErrorWeight'][$i][$j] = 0;
  }
}

//lakukan perubahan bobot untuk output layer

for($i=0;$i<$_SESSION['neuron']['output'];$i++)
{
  for($j=0;$j<$_SESSION['neuron']['hidden'];$j++)
  {

    $dPrevStep                                                     =
Max($_SESSION['bobot']['w_PrevDeltaWeight'][$j][$i],0.0001);
    $dSlope = $_SESSION['bobot']['w_ErrorWeight'][$j][$i];
    $dPrevSlope                                                    =
$_SESSION['bobot']['w_PrevErrorWeight'][$j][$i];
    $dX = $dPrevSlope * $dSlope;
    $dNextStep = 0;

    if ($dX >= 0.0)
    {
       $dNextStep    =    Min($dPrevStep    *    $_GET['inc_LR'],
$_GET['max_LR']);
    } else {
       $dNextStep    =    Max($dPrevStep    *    $_GET['dec_LR'],
$_GET['min_LR']);
```

```
        $dSlope = 0;
    }

    if($dSlope<0)
    {
        $_SESSION['bobot']['w_Weight'][$i][$j] -= $dNextStep;

    } else {
        $_SESSION['bobot']['w_Weight'][$i][$j] += $dNextStep;

    }

    $_SESSION['bobot']['w_PrevDeltaWeight'][$j][$i]          =
$dNextStep;
    $_SESSION['bobot']['w_PrevErrorWeight'][$j][$i] = $dSlope;
    $_SESSION['bobot']['w_ErrorWeight'][$j][$i] = 0.0;

  }
}
/*************************************************
        end of update weight
*************************************************/




echo $polaarray;

?>
```

Gambar 4.3. *Source code* fungsi *Learning*

Proses pembelajaran pada fungsi *Learning* (Gambar 4.3) terdiri atas beberapa proses yaitu proses *FeedForward* (Gambar 4.4), proses *CalculateMSE* (Gambar 4.5), proses *BackpropagateMSE* (Gambar 4.8), proses *UpdateSlope* (Gambar 4.9) dan proses *UpdateWeight* (Gambar 4.10).

```
function feedfwd($input)
{
   /*
   besaran input (x) dan output (y) telah ditentukan sebelumnya,
   dimana x adalah 5 dan y adalah 1.
   */

   for($i=0;$i<$_SESSION['neuron']['hidden'];$i++)
   {
      $neuronsum = 1;
      for($j=0;$j<$_SESSION['neuron']['input'];$j++)
```

```
      {
         $neuronsum += $_SESSION['bobot']['v_Weight'][$j][$i] *
$input['x'][$j];
      }
      $hasil['z_in'][$i] = $neuronsum;
      $hasil['z'][$i] = activation($neuronsum);
   }

   //operasi pada output layer

   for($i=0;$i<$_SESSION['neuron']['output'];$i++)
   {
      $neuronsumsum = 1;
      for($j=0;$j<$_SESSION['neuron']['hidden'];$j++)
      {
         $neuronsumsum += $_SESSION['bobot']['w_Weight'][$j][$i]
* $z[$j];
      }
      $hasil['y_in'][$i] = $neuronsum;
      $hasil['y'][$i] = activation($neuronsum);
   }
   return $hasil;
}
```

Gambar 4.4 *Source code* fungsi *Feedforward*

Fungsi *Feedforward* (Gambar 4.6) bertjuan melakukan proses *feedforward* dengan menggunakan persamaan 2.2 sampai persamaan 2.5. Dalam proses *feedforward* tersebut fungsi aktivasi yang dipakai adalah fungsi aktivasi pada persamaan 2.15.

```
function CalculateMSE($input)
{
  global $hasil;
  for($i=0;$i<$_SESSION['neuron']['output'];$i++)
  {
    //error pada tiap output
    $selisih = $input['y'][$i] - $hasil['y'][$i];

    //jika fungsi aktivasi simetrik maka dibagi 2
    $selisih /= 2;
    $m_vOutputError[$i] = ActivationDerivative($hasil['y'][$i]) *
$selisih;
  }
  return $m_vOutputError;
}
```

Gambar 4.5 *Source code* fungsi *CalculateMSE*

Setelah proses *feedforward* kemudian dilanjutkan dengan proses menghitung nilai informasi error (δ_k) pada tiap-tiap unit *output* menggunakan persamaan 2.18. Proses penghitungan nilai informasi *error* dilakukan pada fungsi *CalculateMSE* (Gambar 4.5).

```
function BackpropagateMSE($input)
{
   global $hasil,$m_error;
   for($j=0;$j<$_SESSION['neuron']['hidden'];$j++)
   {
     $m_vHiddenError[$j] = 0;
     for($i=0;$i<$_SESSION['neuron']['output'];$i++)
     {
       $m_vHiddenError[$j] += $m_error['output'][$i] *
$_SESSION['bobot']['w_Weight'][$j][$i];
     }
     $m_vHiddenError[$j] *=
ActivationDerivative($hasil['z'][$i]);
   }
   return $m_vHiddenError;
}
```

Gambar 4.6 *Source code* fungsi *BackpropagateMSE*

Setelah menghitung nilai informasi *error* (δ_k) pada tiap-tiap unit *output* kemudian dilanjutkan dengan proses menghitung nilai informasi *error* (δj) pada tiap-tiap unit *hidden* dengan melakukan proses *backpropagation* terhadap *error* pada *output* (δ_k) menggunakan persamaan 2.19, proses tersebut dilakukan pada fungsi *BackpropagateMSE* (Gambar 4.6).

```
function UpdateSlope($input)
{
  global $hasil,  $m_error;

  for($j=0;$j<$_SESSION['neuron']['hidden'];$j++)
  {
    for($i=0;$i<$_SESSION['neuron']['input'];$i++)
    {
      $_SESSION['bobot']['v_ErrorWeight'][$i][$j] +=
$m_error['hidden'][$j] * $input[$x][$i];
    }
  }

  for($i=0;$i<$_SESSION['neuron']['output'];$i++)
  {
    for($j=0;$j<$_SESSION['neuron']['hidden'];$j++)
```

```
    {
      $_SESSION['bobot']['w_ErrorWeight'][$j][$i] +=
$m_error['output'][$i] * $hasil['z'][$j];
    }
  }
}
```

Gambar 4.7 *Source code* fungsi *UpdateSlope*

Setelah proses menghitung nilai informasi *error* (δj) pada tiap-tiap unit *hidden* dengan melakukan proses *backpropagation* terhadap *error* pada *output* (δ_k) kemudian dilanjutkan dengan menghitung nilai turunan pertama fungsi error terhadap bobot pada input ke *hidden layer* (ΔE_{ij}) menggunakan persamaan 2.20 dan menghitung nilai turunan pertama fungsi *error* terhadap bobot pada hidden ke *output layer* (ΔE_{jk}) menggunakan persamaan 2.21. Proses tersebut dilakukan pada fungsi *UpdateSlope* (Gambar 4.7).

```
/************************************************
        update weight
************************************************/

for($j=0;$j<$_SESSION['neuron']['hidden'];$j++)
{
  for($i=0;$i<$_SESSION['neuron']['input'];$i++)
  {
    $dPrevStep                                          =
Max($_SESSION['bobot']['v_PrevDeltaWeight'][$i][$j],0.0001);
    $dSlope = $_SESSION['bobot']['v_ErrorWeight'][$i][$j];
    $dPrevSlope                                         =
$_SESSION['bobot']['v_PrevErrorWeight'][$i][$j];
    $dX = $dPrevSlope * $dSlope;
    $dNextStep = 0;

    if ($dX >= 0.0)
    {
       $dNextStep    =    Min($dPrevStep    *    $_GET['inc_LR'],
$_GET['max_LR']);
    } else {
       $dNextStep    =    Max($dPrevStep    *    $_GET['dec_LR'],
$_GET['min_LR']);
       $dSlope = 0;
    }
```

```
    if($dSlope<0)
    {
       $_SESSION['bobot']['v_Weight'][$i][$j] -= $dNextStep;

    } else {
       $_SESSION['bobot']['v_Weight'][$i][$j] += $dNextStep;

    }

    $_SESSION['bobot']['v_PrevDeltaWeight'][$i][$j]          =
$dNextStep;
    $_SESSION['bobot']['v_PrevErrorWeight'][$i][$j] = $dSlope;
    $_SESSION['bobot']['v_ErrorWeight'][$i][$j] = 0;
  }
}

//lakukan perubahan bobot untuk output layer

for($i=0;$i<$_SESSION['neuron']['output'];$i++)
{
  for($j=0;$j<$_SESSION['neuron']['hidden'];$j++)
  {

    $dPrevStep                                              =
Max($_SESSION['bobot']['w_PrevDeltaWeight'][$j][$i],0.0001);
    $dSlope = $_SESSION['bobot']['w_ErrorWeight'][$j][$i];
    $dPrevSlope                                             =
$_SESSION['bobot']['w_PrevErrorWeight'][$j][$i];
    $dX = $dPrevSlope * $dSlope;
    $dNextStep = 0;

    if ($dX >= 0.0)
    {
       $dNextStep    =    Min($dPrevStep    *    $_GET['inc_LR'],
$_GET['max_LR']);
    } else {
       $dNextStep    =    Max($dPrevStep    *    $_GET['dec_LR'],
$_GET['min_LR']);
       $dSlope = 0;
    }

    if($dSlope<0)
    {
       $_SESSION['bobot']['w_Weight'][$i][$j] -= $dNextStep;

    } else {
       $_SESSION['bobot']['w_Weight'][$i][$j] += $dNextStep;

    }
```

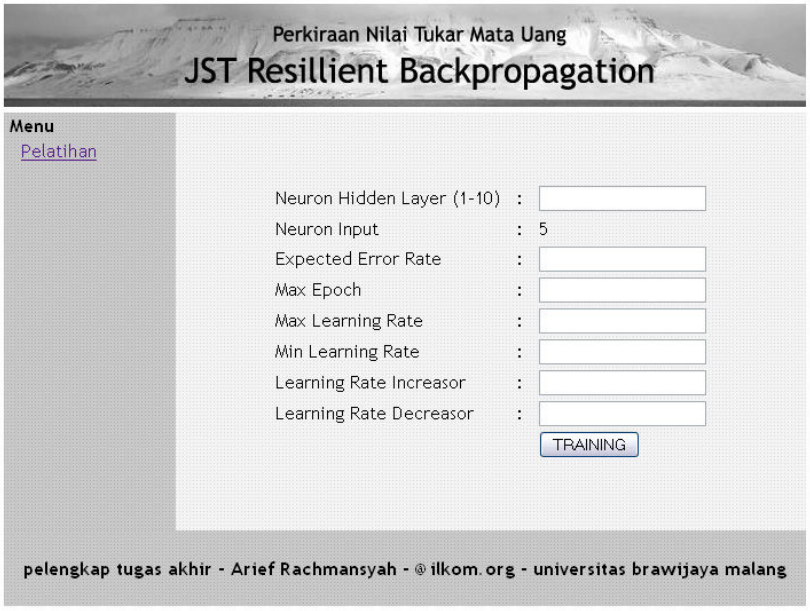
```
    $_SESSION['bobot']['w_PrevDeltaWeight'][$j][$i]           =
$dNextStep;
    $_SESSION['bobot']['w_PrevErrorWeight'][$j][$i] = $dSlope;
    $_SESSION['bobot']['w_ErrorWeight'][$j][$i] = 0.0;

  }
}
/************************************************
        end of update weight
************************************************/
```

Gambar 4.8 *Source code* fungsi UpdateWeight

Setelah proses menghitung nilai turunan pertama fungsi *error* terhadap bobot pada input ke *hidden layer* dan nilai turunan pertama fungsi *error* terhadap bobot pada *hidden* ke *output layer*, kemudian dilanjutkan dengan proses menghitung nilai *learning rate* yang baru dan proses mengubah nilai bobot pada tiap layer. Proses tersebut dilakukan oleh fungsi *UpdateWeight* (Gambar 4.8).

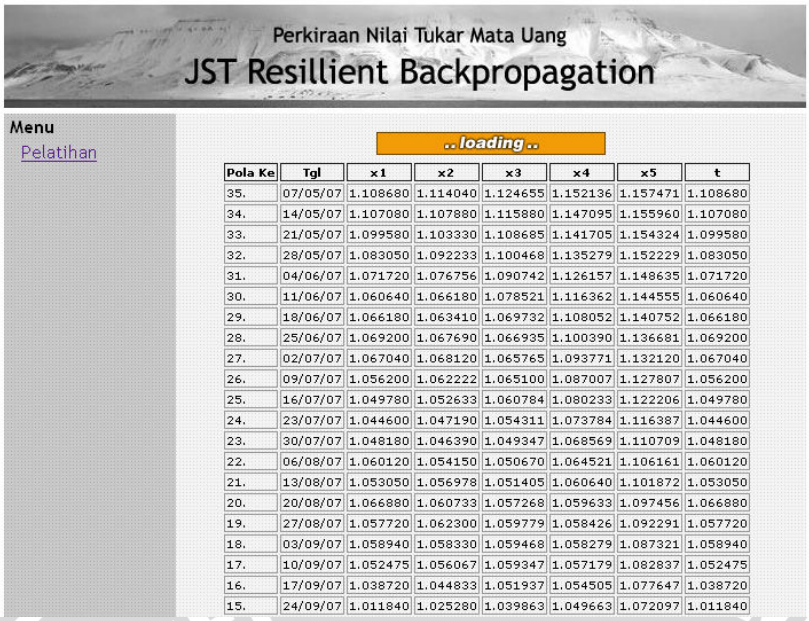
4.3 Implementasi Antarmuka

Berdasarkan rancangan antarmuka pada sub bab 3.3 maka dihasilkan antarmuka yang ditunjukkan pada Gambar 4.11.



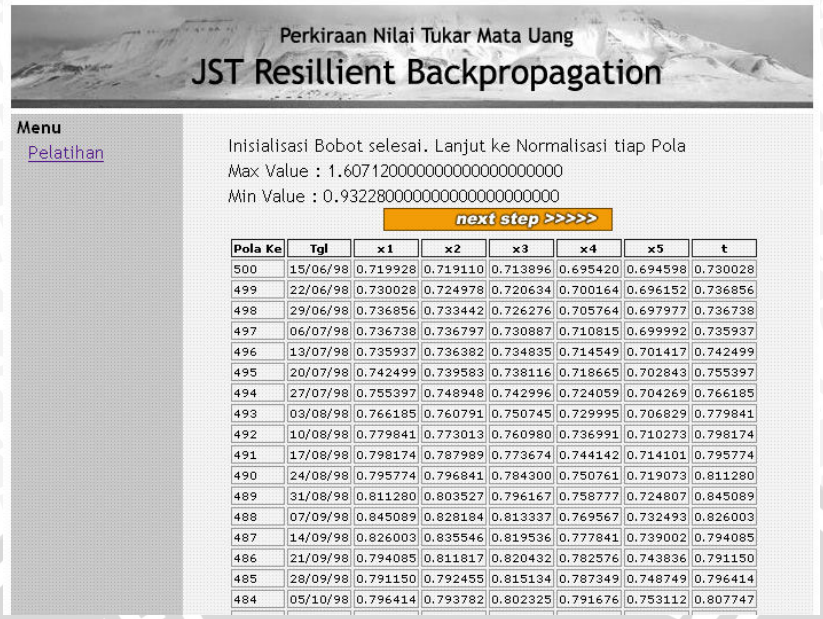
Gambar 4.9
Antarmuka form pelatihan sistem peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation*

Pada antarmuka (Gambar 4.9) menampilkan form isian dengan beberapa masukan yaitu jumlah unit di *hidden layer*, nilai minimum MSE yang diinginkan, epoch maximum, learning rate maksimum, learning rate minimum, nilai penurun learning rate dan nilai penaik learning rate.



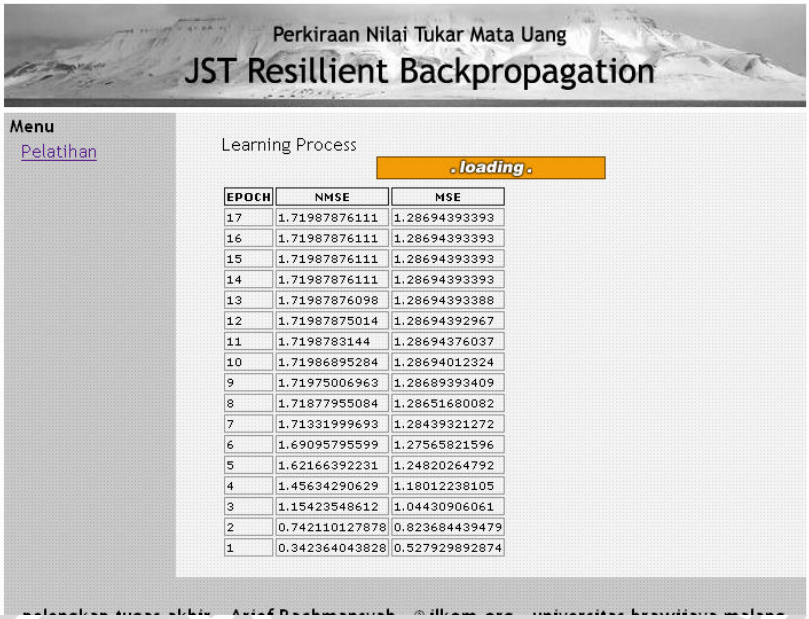
Pola Ke	Tgl	x1	x2	x3	x4	x5	t
35.	07/05/07	1.108680	1.114040	1.124655	1.152136	1.157471	1.108680
34.	14/05/07	1.107080	1.107880	1.115880	1.147095	1.155960	1.107080
33.	21/05/07	1.099580	1.103330	1.108685	1.141705	1.154324	1.099580
32.	28/05/07	1.083050	1.092233	1.100468	1.135279	1.152229	1.083050
31.	04/06/07	1.071720	1.076756	1.090742	1.126157	1.148635	1.071720
30.	11/06/07	1.060640	1.066180	1.078521	1.116362	1.144555	1.060640
29.	18/06/07	1.066180	1.063410	1.069732	1.108052	1.140752	1.066180
28.	25/06/07	1.069200	1.067690	1.066935	1.100390	1.136681	1.069200
27.	02/07/07	1.067040	1.068120	1.065765	1.093771	1.132120	1.067040
26.	09/07/07	1.056200	1.062222	1.065100	1.087007	1.127807	1.056200
25.	16/07/07	1.049780	1.052633	1.060784	1.080233	1.122206	1.049780
24.	23/07/07	1.044600	1.047190	1.054311	1.073784	1.116387	1.044600
23.	30/07/07	1.048180	1.046390	1.049347	1.068569	1.110709	1.048180
22.	06/08/07	1.060120	1.054150	1.050670	1.064521	1.106161	1.060120
21.	13/08/07	1.053050	1.056978	1.051405	1.060640	1.101872	1.053050
20.	20/08/07	1.066880	1.060733	1.057268	1.059633	1.097456	1.066880
19.	27/08/07	1.057720	1.062300	1.059779	1.058426	1.092291	1.057720
18.	03/09/07	1.058940	1.058330	1.059468	1.058279	1.087321	1.058940
17.	10/09/07	1.052475	1.056067	1.059347	1.057179	1.082837	1.052475
16.	17/09/07	1.038720	1.044833	1.051937	1.054505	1.077647	1.038720
15.	24/09/07	1.011840	1.025280	1.039863	1.049663	1.072097	1.011840

Gambar 4.10
Antarmuka pengambilan pola data pada sistem peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation*



Pola Ke	Tgl	x1	x2	x3	x4	x5	t
500	15/06/98	0.719928	0.719110	0.713896	0.695420	0.694598	0.730028
499	22/06/98	0.730028	0.724978	0.720634	0.700164	0.696152	0.736856
498	29/06/98	0.736856	0.733442	0.726276	0.705764	0.697977	0.736738
497	06/07/98	0.736738	0.736797	0.730887	0.710815	0.699992	0.735937
496	13/07/98	0.735937	0.736382	0.734835	0.714549	0.701417	0.742499
495	20/07/98	0.742499	0.739583	0.738116	0.718665	0.702843	0.755397
494	27/07/98	0.755397	0.748948	0.742996	0.724059	0.704269	0.766185
493	03/08/98	0.766185	0.760791	0.750745	0.729995	0.706829	0.779841
492	10/08/98	0.779841	0.773013	0.760980	0.736991	0.710273	0.798174
491	17/08/98	0.798174	0.787989	0.773674	0.744142	0.714101	0.795774
490	24/08/98	0.795774	0.796841	0.784300	0.750761	0.719073	0.811280
489	31/08/98	0.811280	0.803527	0.796167	0.758777	0.724807	0.845089
488	07/09/98	0.845089	0.828184	0.813337	0.769567	0.732493	0.826003
487	14/09/98	0.826003	0.835546	0.819536	0.777841	0.739002	0.794085
486	21/09/98	0.794085	0.811817	0.820432	0.782576	0.743836	0.791150
485	28/09/98	0.791150	0.792455	0.815134	0.787349	0.748749	0.796414
484	05/10/98	0.796414	0.793782	0.802325	0.791676	0.753112	0.807747

Gambar 4.11
Antarmuka normalisasi pola data pada sistem peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation*



EPOCH	NMSE	MSE
17	1.71987876111	1.28694393393
16	1.71987876111	1.28694393393
15	1.71987876111	1.28694393393
14	1.71987876111	1.28694393393
13	1.71987876098	1.28694393388
12	1.71987875014	1.28694392967
11	1.7198783144	1.28694376037
10	1.71986895284	1.28694012324
9	1.71975006963	1.28689393409
8	1.71877955084	1.28651680082
7	1.71331999693	1.28439321272
6	1.69095795599	1.27565821596
5	1.62166392231	1.24820264792
4	1.45634290629	1.18012238105
3	1.15423548612	1.04430906061
2	0.742110127878	0.823684439479
1	0.342364043828	0.527929892874

Gambar 4.12
Antarmuka proses *learning* pada sistem peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation*

4.4 Analisa Hasil

Dari uji coba terhadap perangkat lunak didapatkan hasil seperti table 4.1. Data pada table 4.1 diperoleh dari proses pembelajaran yang dilakukan menggunakan nilai *epoch* maksimum sebesar 2000 dan nilai minimum MSE sebesar 0.000001.

Tabel 4.1 Hasil uji coba proses pembelajaran

Hidden Neuron	NMSE	MAE
2	0,000471082178894998	0,0176711819470595
3	0,000445118968626224	0,0171691847054371
4	0,000461359414718611	0,017506303069287
5	0,000443297683478594	0,0171285326913774
6	0,000424635201468902	0,0167657835989877
7	0,000423792134642356	0,0167148434777906
8	0,00042062193769636	0,0166540719090822
9	0,000418038766027225	0,0165961141527163
10	0,000420415243364657	0,0166563160514452
11	0,00041175047480093	0,0164808484304246
12	0,000426004216017417	0,0167955528462755
13	0,000409702690443097	0,0164247521899531
14	0,000427912444130356	0,0168523946978328
15	0,000427041130016002	0,0168066867882103
16	0,000406514903936872	0,0163331743426233
17	0,000425818182535505	0,0168227947932949
18	0,000402439862211044	0,0162211116634791
19	0,000407963012809256	0,0163179029188442
20	0,000398715215130621	0,016167483212024
21	0,000400276941396402	0,016159879668508
22	0,000397069914883796	0,0161106326606694
23	0,000400245731952689	0,016175676980933
24	0,000395749813042967	0,0160590391794389
25	0,000409067197668422	0,0162634111708403
26	0,000416537466125079	0,0163282047684738
27	0,000495369489148641	0,0183586464725697
28	0,000410308304774373	0,0162422900677828
29	0,000401795259154556	0,016220873807508

30	0,000410953525579572	0,0163560839759399

Dari tabel 4.1 dapat disimpulkan bahwa nilai NMSE yang minimum untuk proses pembelajaran adalah 0,000395749813042967 dan nilai MAE yang minimum adalah 0,0160590391794389. Nilai minimum NMSE dan MAE tercapai pada jumlah *neuron* di *hidden layer* sebesar 24.

Setelah dilakukan proses pembelajaran kemudian dilanjutkan dengan proses peramalan nilai *moving average* untuk minggu yang akan dating. Hasil uji coba peramalan nilai *moving average* untuk minggu yang akan datang dapat dilihat pada tabel 4.2.

Tabel 4.2 Hasil uji coba proses peramalan

Hidden Neuron	NMSE	MAE
2	0,000734814827740349	0,0241007370102806
3	0,00066810699520006	0,0231324339684694
4	0,000685917094095858	0,0234825754169713
5	0,000704098962656435	0,0235551031864501
6	0,000652573712284537	0,0230183238830505
7	0,000656244624583223	0,0230284345148796
8	0,000662119807264174	0,0228359855979653
9	0,000613600492613026	0,0222247473328334
10	0,000664785748894024	0,0231350827793923
11	0,000686491409396077	0,0234002026075421
12	0,000660713693283725	0,0230610180339068
13	0,00068551005121369	0,0233273302865532
14	0,00067430240533505	0,0234024611808108
15	0,000644899053717083	0,0227960932935759
16	0,000658188727546903	0,0229899941381575
17	0,000647177505135863	0,0228616467883145
18	0,000609804217987888	0,0219431001214484
19	0,000640828204032474	0,022579887123914
20	0,000625012359613578	0,0223967462747598
21	0,000555880930120499	0,0209274860815991
22	0,000564950194676017	0,0211731892618037
23	0,000543931364031175	0,02074544138254

24	0,000605630754903105	0,0220278936861451
25	0,000539861339374656	0,0205462213179038
26	0,000514817821556841	0,019885820362511
27	0,000765643027150498	0,0250962873143367
28	0,000538485844679818	0,0204721202897465
29	0,00063031654604208	0,0224623988057455
30	0,000561611169314192	0,0209843917419214

Dari tabel 4.2 dapat disimpulkan bahwa nilai NMSE yang minimum untuk proses peramalan adalah 0,000514817821556841 dan nilai MAE yang minimum adalah 0,019885820362511. Nilai minimum NMSE dan MAE tercapai pada jumlah *neuron* di *hidden layer* sebesar 26.

Setelah melakukan kedua percobaan, yaitu pembelajaran dan peramalan kemudian dilanjutkan dengan menghitung nilai rata-rata NMSE dan MAE antara proses pembelajaran dan peramalan. Dari nilai rata-rata tersebut akan didapat jumlah *neuron* pada *hidden layer* yang terbaik untuk proses pembelajaran dan peramalan. Nilai rata-rata NMSE dan MAE untuk proses pembelajaran dan peramalan dapat dilihat pada tabel 4.3.

Tabel 4.3 Nilai rata-rata NMSE dan MAE untuk proses pembelajaran dan peramalan.

Hidden Neuron	**NMSE**	**MAE**
2	0.00060294850331767400	0.02088595947867010000
3	0.00055661298191314200	0.02015080933695320000
4	0.00057363825440723500	0.02049443924312920000
5	0.00057369832306751400	0.02034181793891380000
6	0.00053860445687671900	0.01989205374101910000
7	0.00054001837961279000	0.01987163899633510000
8	0.00054137087248026700	0.01974502875352370000
9	0.00051581962932012500	0.01941043074277480000
10	0.00054260049612934000	0.01989569941541880000
11	0.00054912094209850400	0.01994052551898330000
12	0.00054335895465057100	0.01992828544009110000
13	0.00054760637082839400	0.01987604123825320000

14	0.00055110742473270300	0.02012742793932180000
15	0.00053597009186654200	0.01980139004089310000
16	0.00053235181574188700	0.01966158424039040000
17	0.00053649784383568400	0.01984222079080470000
18	0.00050612204009946600	0.01908210589246380000
19	0.00052439560842086500	0.01944889502137910000
20	0.00051186378737210000	0.01928211474339190000
21	0.00047807893575845000	0.01854368287505350000
22	0.00048101005477990600	0.01864191096123650000
23	0.00047208854799193200	0.01846055918173650000
24	0.00050069028397303600	0.01904346643279200000
25	0.00047446426852153900	0.01840481624437200000
26	0.00046567764384096000	0.01810701256549240000
27	0.00063050625814957000	0.02172746689345320000
28	0.00047439707472709500	0.01835720517876470000
29	0.00051605590259831800	0.01934163630662680000
30	0.00048628234744688200	0.01867023785893060000

Dari tabel 4.3 dapat disimpulkan bahwa nilai rata-rata NMSE yang minimum untuk proses pembelajaran dan peramalan adalah 0.00046567764384096000 dan nilai MAE yang minimum adalah 0.01810701256549240000. Nilai minimum NMSE dan MAE tercapai pada jumlah neuron di *hidden layer* sebesar 26.

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

1. Sistem peramalan nilai tukar mata uang menggunakan jaringan syaraf tiruan *resilient backpropagation* mampu meramalkan nilai moving average pada minggu yang akan datang dengan cukup baik.
2. Nilai rata-rata NMSE yang minimum untuk proses pembelajaran dan peramalan adalah 0.00046567764384096000. Sedangkan nilai rata-rata MAE yang minimum untuk proses pembelajaran dan peramalan adalah 0.01810701256549240000. Nilai minimum NMSE dan MAE tercapai pada jumlah neuron di hidden layer sebesar 26.
3. Pembelajaran dilakukan menggunakan metode *resilient backpropagation* dengan jumlah unit pada *input layer* sebanyak 5 unit, jumlah unit pada *hidden layer* sebanyak 26 unit, jumlah unit pada *output layer* sebanyak 1 unit, nilai maksimum *learning rate* sebesar 50, nilai minimum *learning rate* sebesar 0, nilai *learning rate increasor* sebesar 1.2, nilai *learning rate decreasor* sebesar 0.5 dan *max epoch* sebesar 2000. Data yang digunakan sebanyak 11 tahun, untuk proses pembelajaran sebanyak 500 pola. Sedangkan data evaluasi yang dipakai adalah nilai penutupan sebanyak 50 pola.

5.2. Saran

Aplikasi yang dibangun masih belum sempurna. Hal yang dapat bermanfaat penelitian ini adalah pengembangan aplikasi peramalan nilai tukar mata uang dengan menggunakan jaringan syaraf tiruan resilient backpropagation yang dapat digunakan pada bermacam – macam nilai tukar mata uang.

DAFTAR PUSTAKA

Kamruzzaman, Joarder., Rezaul Begg dan Ruhul Sarker. 2006. *Artificial Neural Networks in Finance and Manufacturing.*Idea Group Publishing.

Kamruzzaman, Joarder. Ruhul A. Sarker. 2003. Comparing ANN Based Models with ARIMA for Prediction of Forex Rates. ASOR BULLETIN, Volume 22 Number 2, June 2003.

Kusumadewi, Sri. 2003. *Artificial intelligence (teknik dan aplikasinya).* Graha Ilmu. Jakarta.

LeCun, Yann., Leon Bottou., Genevieve B. Orr. dan Klaus-Robert Müller. 1998. *Efficient backprop.* IEEE volume 86 no 11.

Nissen, Steffen. 2006. *Fast Artificial Neural Network Library (fann).* http://leenissen.dk/fann/html/index.html.

O'Neill, Mike. 2006. *Neural network for recognition of handwritten digits.* http://www.codeproject.com/library/NeuralNetRecognition/NeuralNetRecognition.asp. Diakses pada tanggal 10 Desember 2006.

Rojas, Ra´ul. 1996. *Neural Networks : A Systematic Introduction.* Springer. Berlin.

Siang, JJ. 2005. *Jaringan syaraf tiruan dan pemrogramannya menggunakan MATLAB*. Penerbit ANDI Yogyakarta.

Thwien, Multilayer Perceptron from ANN for PHP5, http://ann.thwien.de/index.php/Multilayer_perceptron , Diakses pada tanggal 13 Juli 2009.

Wikipedia, Neural Network,

http://en.wikipedia.org/wiki/Neural_networks , Diakses pada tanggal 10 Juli 2009.

Yao, Jingtao., Chew Lim Tan. 2000. *A case study on using neural networks to perform technical forecasting of forex.*