PENERAPAN ALGORITMA GENETIKA PADA PELATIHAN JARINGAN SYARAF TIRUAN UNTUK PERAMALAN HARGA VALUTA ASING

SKRIPSI

Oleh:
KANIA PUTRI N
0710960006-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUANALAM
UNIVERSITAS BRAWIJAYA
MALANG
2012

UNIVERSITAS BRAWIJAYA

PENERAPAN ALGORITMA GENETIKA PADA PELATIHAN JARINGAN SYARAF TIRUAN UNTUK PERAMALAN HARGA VALUTA ASING

FALSAN JUDUL

Skripsi

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

Oleh :
KANIA PUTRI N
0710960006-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUANALAM
UNIVERSITAS BRAWIJAYA
MALANG

2012

UNIVERSITAS BRAWIJAYA

LEMBAR PENGESAHAN SKRIPSI

PENERAPAN ALGORITMA GENETIKA PADA PELATIHAN JARINGAN SYARAF TIRUAN UNTUK PERAMALAN HARGA VALUTA ASING

Oleh :

KANIA PUTRI N
0710960006-96

Setelah dipertahankan di depan Majelis Penguji
pada tanggal 10 Januari 2012

dan dinyatakan memenuhi syarat untuk memperoleh gelar
Sarjana Komputer dalam bidang Ilmu Komputer

Pembimbing I,	Pembimbing II,
Dian Eka Ratnawati,SSi.,M.Kom	**Dany Primanita K,ST**
NIP. 197306192002122001	**NIP. 197711162005012003**

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas Brawijaya

Dr. Abdul Rouf Alghofari, M.Sc
NIP.196709071992031001

UNIVERSITAS BRAWIJAYA

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Kania Putri Noveria
NIM : 0710960006-96
Jurusan : Matematika
Program Studi : Ilmu Komputer
Penulis skripsi berjudul : Penerapan Algoritma Genetika pada Pelatihan Jaringan Syaraf Tiruan untuk Peramalan Harga Valuta Asing

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 termaksud 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, 10 Januari 2012
Yang menyatakan,

Kania Putri Noveria
NIM. 0710960006-96

PENERAPAN ALGORITMA GENETIKA PADA PELATIHAN JARINGAN SYARAF TIRUAN UNTUK PERAMALAN HARGA VALUTA ASING

ABSTRAK

Perdagangan valuta asing dianggap sebagai bisnis alternatif karena dapat mendatangkan keuntungan. Untuk mendapatkan keuntungan yang maksimal, diperlukan teknik peramalan untuk mengetahui kapan harga valas naik atau turun. Salah satu teknik peramalan yang digunakan untuk meramalkan harga valas adalah Jaringan Syaraf Tiruan (JST). JST melakukan proses pelatihan terhadap jaringan untuk mendapatkan model jaringan dan bobot yang sesuai sehingga mampu mengenali pola pergerakan harga valas. Salah satu algoritma pelatihan JST yang dapat digunakan adalah Algoritma Genetika (AG). Algoritma pelatihan khusus seperti *backpropagation* dapat digunakan, namun biasanya digunakan pada permasalahan yang khusus saja. Sebaliknya, AG dengan sedikit modifikasi dapat digunakan untuk melatih JST dalam menyelesaikan berbagai masalah.

Pada penelitian ini menggunakan algoritma *backpropagation* yaitu tahap *feed-forward backpropagation* dan AG sebagai algoritma pelatihan JST. AG mengkodekan kromosom sebagai susunan bobot-bobot JST. Data yang digunakan adalah harga valas Pounsterling terhadap Dollar US (GBP/USD). Untuk proses pelatihan digunakan data selama 1 tahun, sedangkan proses pengujian digunakan data selama 2 bulan setelah data pelatihan.

Hasil uji coba menunjukkan bahwa AG dapat melakukan pelatihan JST dengan baik. Parameter genetika sangat berpengaruh terhadap proses pelatihan JST. Parameter genetika yang menghasilkan nilai kesalahan peramalan terkecil yaitu ukuran populasi 50, jumlah generasi maksimal 10000, probabilitas *crossover* 0.9 dan probabilitas mutasi 0.3. Bobot JST yang dihasilkan dari pelatihan AG mampu mengenali pola pergerakan harga valas dengan nilai RMSE sebesar 0.00418 dan MAPE sebesar 0.002. Hal ini menunjukkan bahwa tingkat akurasi yang dihasilkan sistem sangat baik karena sangat mendekati nilai sebenarnya.

THE GENETIC ALGORITHM IMPLEMENTATION ON ARTIFICIAL NEURAL NETWORKS TRAINING FOR FORECASTING FOREIGN EXCHANGE RATES

ABSTRACT

Foreign exchange trading has considered as profitable alternative business. In order to approach the maximum benefit, forecasting techniques has done to perceive the exchange rates rise or fall movements. One of the forecasting techniques which already used is Artificial Neural Networks (ANN). ANN performed the training process for network to obtain a network model and an appropriate weight to be able to recognize the foreign exchanges movement patterns. Genetic Algorithm (GA) is one of the ANN training that can be applied. Specialized training algorithms such as back propagation could be used, the lack of this algorithm could be used only on special cases. In contrast, a less modification GA can be used in ANN train in order to solving any various problems.

This research was using feed-forward back propagation stage and GA as train algorithm. GA codes chromosomes as ANN weights arrangement. The data that used are foreign exchange Poundsterling against US Dollar (GBP/USD). The training process was using data in 1 year period, whereas the testing process was using data for 2 months after the training data.

The trial results showed that the GA can do well the ANN training. Genetic parameters had greatly influenced the ANN training process. Genetic parameters was producing the error smallest prediction of population size was 50, maximum generation number 10000, crossover probability 0,9, and mutation probability 0.3. ANN weights resulting from training GA were able to recognize foreign currency price movement patterns with RMSE 0.00418 and MAPE for 0.0002. It is showed that accuracy rate that produces by system was good because it's approaches an exact values.

KATA PENGANTAR

Alhamdulillah, dengan mengucapkan puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi ini. Skripsi yang berjudul " ***Penerapan Algoritma Genetika pada Pelatihan Jaringan Syaraf Tiruan untuk Peramalan Harga Valuta Asing***" merupakan salah syarat memperoleh gelar Sarjana Komputer pada program studi Ilmu Komputer Jurusan Matematika Fakultas MIPA Universitas Brawijaya.

Dalam penyelesaian skripsi ini, penulis telah mendapat begitu banyak bantuan baik moral maupun materiil dari banyak pihak. Atas bantuan yang telah diberikan, penulis ingin menyampaikan penghargaan dan ucapan terima kasih yang sedalam-dalamnya kepada:

1. Dian Eka Ratnawati,SSi.,M.Kom selaku dosen pembimbing yang telah membimbing dengan bijaksana dan sabar dalam membimbing dengan baik penyusunan skripsi ini.
2. Dany Primanita K,ST., selaku dosen pembimbing sekaligus dosen pembimbing akademik, yang telah membimbing dengan bijaksana dan sabar dalam membimbing dengan baik penyusunan skripsi ini.
3. Dr. Abdul Rouf Alghofari, M.Sc, selaku Ketua Jurusan Matematika Fakultas MIPA Universitas Brawijaya.
4. Drs. Marji, M.T, selaku Ketua Program Studi Ilmu Komputer, Jurusan Matematika, MIPA Universitas Brawijaya.
5. Segenap Bapak dan Ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada penulis selama menempuh pendidikan di Program Studi Ilmu Komputer Jurusan Matematika Fakultas MIPA Universitas Brawijaya.
6. Segenap staf dan karyawan di Jurusan Matematika Fakultas MIPA Universitas Brawijaya yang telah banyak membantu penulis dalam pelaksanaan penyusunan proposal skripsi ini.
7. Papa Anang Subadi, Mama Farida Arianie, Kakak Dita dan Adek Aning, serta seluruh keluarga. Terima kasih atas semua doa, kasih sayang dan perhatian yang tulus serta dukungan yang telah diberikan.
8. Ryan Hidayat, Ariyati Novita Sari, Ilham Nur Febriansyah, Hendrawan Kuncoro, Dita Ayu, Betha, Che, Yessy, Iid, Elok, Udin, Kakah, Festri, Khuluk, Rina, Nani, Iwan, Fajar, John, Nini,

Sari, Mas Anis dan sahabat-sahabat di Program Studi Ilmu Komputer Fakultas MIPA Universitas Brawijaya yang telah banyak memberikan bantuan demi kelancaran pelaksanaan penyusunan skripsi ini. Terima kasih atas senyuman, semangat, bantuan, dukungan, doa dan hari-hari kita.

9. Keluarga besar Kertorejo 35, tempat penulis bernaung selama menjalani masa perantauan untuk menuntut ilmu, terima kasih telah menjadi keluarga kedua untuk penulis.
10. Mbak Wilda, Komoth, Iam, Kiben, Ega, Adit, Rizky aka Batok, Kemal, Gee, Novan, Dewi Noona, Kiky dan teman-teman Sone Malang yang telah menjadi bagian dari aktivitas penulis. Terima kasih atas senyuman, semangat, dukungan, dan do'a yang telah diberikan kepada penulis selama menyelesaikan skripsi ini.
11. Dan semua pihak yang telah terlibat baik secara langsung maupun tidak langsung yang tidak dapat penulis sebutkan satu per satu terima kasih atas semua bantuan yang telah diberikan.

Semoga penulisan laporan tugas akhir ini bermanfaat bagi pembaca sekalian. Dengan tidak lupa sebagai manusia, penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan, dan mengandung banyak kekurangan, sehingga dengan segala kerendahan hati penulis mengharapkan kritik dan saran yang membangun dari pembaca.

Malang, Januari 2012

Penulis

DAFTAR ISI

HALAMAN JUDUL ..... **i**
LEMBAR PENGESAHAN SKRIPSI ..... **iii**
LEMBAR PERNYATAAN ..... **v**
ABSTRAK ..... **vii**
KATA PENGANTAR ..... **xi**
DAFTAR ISI ..... **xiii**
DAFTAR GAMBAR ..... **xvii**
DAFTAR TABEL ..... **xix**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Rumusan Masalah 3
1.3 Batasan Masalah 3
1.4 Tujuan 4
1.5 Manfaat 4
1.6 Sistematika Penulisan 4
BAB II TINJAUAN PUSTAKA **7**
2.1 Peramalan (*Forecasting)* 7
2.2 Pengenalan Valuta Asing 8
2.2.1 Perdagangan Valuta Asing 9
2.2.2 Teknik Peramalan Harga Valuta Asing 9
2.3 Jaringan Syaraf Tiruan 11
2.3.1 Model Umum Jaringan Syaraf Tiruan 11
2.3.2 Arsitektur Jaringan Syaraf Tiruan 14
2.3.3 Jaringan Umpan Maju (*Feed-Forward Networks*) 16
2.4 Algoritma Genetika 18
2.4.1 Pengkodean Kromosom 19

2.4.2 Fungsi *fitness* 20
2.4.3 Operator Genetika 21
2.4.4 Parameter Genetika 23
2.5 Pelatihan JST Menggunakan Algoritma Genetika 24
2.6 Normalisasi dan Denormalisasi 27
2.7 Evaluasi Hasil Peramalan 27
BAB III METODOLOGI DAN PERANCANGAN 29
3.1 Analisa Sistem 30
3.1.1 Deskripsi Sistem 30
3.1.2 Deskripsi Data 30
3.2 Perancangan Sistem 30
3.2.1 Normalisasi Data 32
3.2.2 Penentuan Arsitektur JST 33
3.2.3 Pelatihan JST menggunakan Algoritma Genetika 34
3.3 Rancangan Antarmuka 48
3.4 Perhitungan Manual 50
3.4.1 Normalisasi Data 50
3.4.2 Pembangkitan Populasi Awal 52
3.4.3 Perhitungan *Fitness* 53
3.4.4 Seleksi *Parent* dengan *Roulette Wheel* 56
3.4.5 *Crossover* 57
3.4.6 Mutasi 58
3.4.7 Pembentukan Populasi baru 59
3.4.8 Proses Peramalan 60
3.5 Perancangan Desain Data 67
3.6 Perancangan Uji Coba 68
BAB IV IMPLEMENTASI DAN PEMBAHASAN 71
4.1 Lingkungan Implementasi 71

4.1.1 Lingkungan Perangkat Keras 71
4.1.2 Lingkungan Perangkat Lunak 71
4.2 Implementasi Program 71
4.2.1 Implementasi tipe data `valuta` 71
4.2.2 Implementasi normalisasi data 72
4.2.3 Implementasi inisialisasi populasi awal 73
4.2.4 Implementasi prosedur hitung *fitness* 73
4.2.5 Implementasi *select parent* 75
4.2.6 Implementasi operasi *crossover* 76
4.2.7 Implementasi operasi mutasi 77
4.2.8 Implementasi seleksi populasi baru 78
4.2.9 Implementasi pengujian *output* jaringan 79
4.3 Implementasi Antar Muka 82
4.4 Sistematika Pengujian 85
4.5 Uji Coba dan Analisis Hasil 86
4.5.1 Pengaruh Probabilitas *Crossover* dan Probabilitas Mutasi terhadap Nilai *Fitness* 86
4.5.2 Pengaruh jumlah generasi terhadap nilai *fitness* 90
4.5.3 Pengaruh ukuran populasi terhadap nilai *fitness* 92
4.5.4 Perbandingan harga saham aktual dengan *output* jaringan . 94
BAB V KESIMPULAN DAN SARAN 99
5.1 Kesimpulan 99
5.2 Saran 100
LAMPIRAN 101
DAFTAR PUSTAKA 111

DAFTAR GAMBAR

Gambar 2.1 Struktur neuron jaringan syaraf tiruan 12
Gambar 2.2 Taksonomi JST beserta nama-nama algoritma pembelajaran yang termasuk ke dalamnya 15
Gambar 2.3 Jaringan *feed-forward* dengan 4 *input* dan 2 *output*..... 16
Gambar 2.4 Diagram algoritma genetika secara umum..... 19
Gambar 2.5 Contoh individu dengan *real-number encoding*..... 19
Gambar 2.6 Contoh individu dengan *discrete decimal encoding* 20
Gambar 2.7 Contoh individu dengan *binary encoding* 20
Gambar 2.8 Contoh seleksi orang tua menggunakan metode *roulette wheel*..... 21
Gambar 2.9 Contoh *crossover* metode *one-cut-point crossover*..... 22
Gambar 2.10 Contoh mutasi dengan menambah dan mengurangi konstanta (a) dan penggantian gen dengan bilangan acak baru (b)..... 22
Gambar 2.11 Diagram alir pelatihan neural network dengan algoritma genetika..... 25
Gambar 2.12 Skema pengkodean kromosom untuk pelatihan jaringan syaraf tiruan 26
Gambar 3.1 Tahapan penelitian 29
Gambar 3.2 *Flowchart* proses peramalan harga valas secara umum 31
Gambar 3.3 *Flowchart* normalisasi data 32
Gambar 3.4 Arsitektur jaringan *feed-forward*..... 34
Gambar 3.5 *Flowchart* Algoritma Genetika 35
Gambar 3.6 Arsitektur jaringan yang digunakan 36
Gambar 3.7 Representasi Kromosom 36
Gambar 3.8 *Flowchart* proses hitung *fitness*..... 38
Gambar 3.9 *Flowchart* proses *feedforward* pada perhitungan nilai *fitness*..... 39
Gambar 3.10 *Flowchart* metode *Roulette Wheel* 42
Gambar 3.11 Contoh proses *crossover* 43
Gambar 3.12 *Flowchart* proses *one-cut-point crossover* 44
Gambar 3.13 Contoh proses mutasi 45
Gambar 3.14 *Flowchart* proses mutasi 46
Gambar 3.15 *Flowchart* proses pembentukan populasi baru..... 47
Gambar 3.16 Rancangan Antarmuka untuk Proses Pelatihan..... 48
Gambar 3.17 Rancangan Antarmuka untuk Proses Pengujian..... 49

Gambar 4.1 Antar muka utama..... 82
Gambar 4.2 Tampilan proses pelatihan 83
Gambar 4.3 Tampilan kromosom pada proses pelatihan..... 84
Gambar 4.4 Tampilan proses pengujian 85
Gambar 4.5 Grafik perbandingan nilai *fitness* terhadap perunahan Pc dan Pm 89
Gambar 4.6 Grafik perbandingan nilai fitness terhadap jumlah generasi 91
Gambar 4.7 Grafik perbandingan nilai *fitness* terhadap ukuran populasi 93
Gambar 4.8 Grafik Perbandingan harga valas aktual terhadap 96

DAFTAR TABEL

Tabel 3.1 Data awal valas sebelum dilakukan normalisasi 50
Tabel 3.2 Hasil Normalisasi data valas 52
Tabel 3.3 Hasil perhitungan *feedforward* kromosom ke-1 55
Tabel 3.4 Hasil *fitness* tiap kromosom 56
Tabel 3.5 Hasil *fitness* tiap kromosom setelah proses reproduksi 59
Tabel 3.6 Hasil pengurutan individu berdasarkan *fitness* 60
Tabel 3.7 Individu generasi ke-1 60
Tabel 3.8 Data Uji 61
Tabel 3.9 Data uji yang sudah dinormalisasi 62
Tabel 3.10 Hasil perhitungan *feedforward* data uji 64
Tabel 3.11 Hasil *output* yang telah di-denormalisasi 65
Tabel 3.12 *Error* peramalan harga *close* valas 66
Tabel 3.13 Tipe Data yang Digunakan 67
Tabel 3.14 Percobaan Pengaruh Perubahan Probabilitas *Crossover* dan Probabilitas Mutasi terhadap Nilai *Fitness* 68
Tabel 3.15 Percobaan Pengaruh Jumlah Generasi terhadap Nilai *Fitness* 69
Tabel 3.16 Percobaan Pengaruh Ukuran Populasi terhadap Nilai *Fitness* 69
Tabel 3.17 Hasil peramalan terhadap data aktual harga valuta asing .. 69

UNIVERSITAS BRAWIJAYA

UNIVERSITAS BRAWIJAYA

UNIVERSITAS BRAWIJAYA

BAB I
PENDAHULUAN

1.1 Latar Belakang

Valuta asing atau yang disingkat dengan kata "valas" secara bebas dapat diartikan sebagai mata uang yang dikeluarkan dan digunakan sebagai alat pembayaran yang sah di negara lain. (Berlianta,2005). Dalam praktek sehari-hari pertukaran valuta asing dilakukan dalam bentuk transaksi jual-beli valuta atau transaksi valuta asing. Transaksi valuta asing dapat diartikan sebagai kesepakatan atau perjanjian antara dua pihak untuk mempertukarkan (jual/beli) mata uang yang dimilikinya (Berlianta,2005). Transaksi ini dilakukan di bursa valuta asing, yaitu tempat atau lembaga yang memperdagangkan mata uang asing. Beberapa mata uang yang sering diperdagangkan adalah U.S Dollar (USD), Great Britain Pound (GBP), Japanese Yen (JPY), dan Euro (EUR).

Perdagangan valuta asing (*Forex Trading*) mulai berkembang pada era 1970-an dan dianggap menjadi salah satu bisnis alternatif karena dapat mendatangkan keuntungan bagi pelakunya, baik perusahaan atau investor pribadi. *Trader* (pelaku pasar valuta asing) harus mengetahui kapan valas tersebut mengalami penurunan harga atau kenaikan harga agar mendapatkan keuntungan yang maksimal dan tidak mengalami kerugian. Oleh karena itu, peramalan mengenai harga valas diperlukan dalam kegiatan transaksi jual beli valas. Peramalan harga valas dapat dilakukan salah satunya melalui analisis teknikal.

Analisis teknikal dapat diartikan sebagai suatu metode peramalan pergerakan harga dengan menggunakan informasi yang terkandung pada pergerakan harga di masa lalu (Berlianta, 2005). Menurut seorang analis pasar, Martin J. Pring, analisis teknikal bertujuan untuk mencari dan mengidentifikasi kecenderungan (*trend*) berdasarkan perubahan prilaku para pelaku pasar dan investor dalam menyingkapi berbagai peristiwa yang mempengaruhi pasar (Darmawan, 2007). Peristiwa ini biasanya mempengaruhi pola harga *open*, *low*, *high* dan *close.* Para analis pasar memiliki tanggapan bahwa perilaku para pelaku pasar dalam menanggapi peristiwa ekonomi yang terjadi di masa lalu sama dengan perilaku masa kini,

sehingga terjadi kesamaan pola data atau grafik. Kesamaan pola data ini dapat dimanfaatkan untuk meramalkan pergerakan harga di masa yang akan datang.

Teknik peramalan harga valuta asing yang saat ini sering digunakan adalah *Artificial Neural Network* (ANN) (Kamruzzaman, 2004). ANN atau disebut juga jaringan syaraf tiruan (JST) adalah sistem komputasi dimana arsitektur dan operasi diilhami dari pengetahuan sel syaraf biologi di dalam otak (Kristanto,2004). Jaringan syaraf tiruan merupakan jaringan dari banyak unit pemroses kecil (yang disebut neuron) yang masing-masing melakukan proses sederhana, ketika digabungkan akan menghasilkan perilaku yang kompleks yang dapat digunakan sebagai alat untuk memodelkan hubungan yang kompleks antara masukan dan keluaran pada sebuah sistem untuk menemukan pola-pola pada data (Hermawan, 2006).

Komputasi pada JST sangat terikat dengan struktur jaringan, yakni jumlah neuron dan koneksi-koneksi yang ada. Agar JST dapat mengerjakan suatu komputasi yang diinginkan untuk penyelesaian masalah, maka jumlah neuron dan bobot jaringan harus ditentukan secara tepat. Proses ini dikenal sebagai *training* (pelatihan). Salah satu metode yang mungkin digunakan untuk pelatihan JST adalah algoritma genetika. Beberapa algoritma pelatihan khusus dapat digunakan, misalnya *backpropagation* untuk pelatihan *Feed Forward Neural Network* (FFNN), namun algoritma khusus tersebut biasanya digunakan pada permasalahan yang khusus juga. Sebaliknya, dengan menggunakan algoritma genetika dengan sedikit modifikasi dapat digunakan untuk melatih JST dalam menyelesaikan berbagai masalah (Suyanto, 2005).

Algoritma genetika adalah algoritma pencarian heuristik yang didasarkan atas mekanisme seleksi dan genetika alami. Konsep dasar yang mengilhami timbulnya algoritma genetika adalah teori evolusi, yang dikemukakan oleh Charles Darwin. Algoritma genetika dimulai dengan membentuk sejumlah alternative solusi yang disebut sebagai populasi. Setiap solusi pada algoritma genetika diwakili oleh satu individu atau satu kromosom (Kusumadewi, 2003).

Penelitian sebelumnya yang menggunakan algoritma genetika pada ANN yaitu "Prediksi Harga Saham Menggunakan Algoritma Genetika Berbasis Jaringan *Backpropagation"* yang dilakukan oleh Asif Ullah Khan, dkk (2008). Pada penelitian ini

algoritma genetika digunakan untuk mendapatkan bobot yang akan digunakan pada jaringan *Backpropagation.* Hasil ramalan yang diperoleh adalah 98% akurat dibandingkan dengan harga saham aktual.

Dari uraian diatas maka penulis melakukan penelitian menggunakan algoritma genetika untuk melatih jaringan syaraf tiruan untuk meramalkan harga valuta asing, sehingga judul yang diambil adalah "**Penerapan Algoritma Genetika pada Pelatihan Jaringan Syarat Tiruan Untuk Peramalan Harga Valuta Asing**". Algoritma genetika diharapkan mampu mendapatkan bobot jaringan syaraf tiruan sehingga dapat menghasilkan ramalan harga valuta asing yang baik.

1.2 Rumusan Masalah

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

1. Bagaimana menerapkan algoritma genetika pada pelatihan jaringan syaraf tiruan untuk meramalkan harga valuta asing.
2. Berapa tingkat kesalahan ramalan yang dihasilkan oleh jaringan syaraf tiruan dengan pelatihan menggunakan algoritma genetika.
3. Parameter algoritma genetika apakah yang mempengaruhi hasil peramalan harga valuta asing.

1.3 Batasan Masalah

Batasan masalah yang digunakan pada penelitian ini adalah sebagai berikut :

1. Data yang digunakan adalah data harga valuta asing yaitu USD terhadap GBP, mulai Januari 2010 sampai Februari 2011 yang diambil dari www.instaforex.com
2. Penelitian ini menggunakan parameter yang berasal dari data-data teknis, yaitu harga *open, low*, *high* dan *close* dan tidak memperhatikan faktor fundamental seperti tingkat inflasi, suku bunga dan faktor ekonomi lainnya.
3. Jaringan syaraf tiruan yang digunakan pada penelitian ini adalah *feed forward neural network* (FFNN) untuk menghitung *output* peramalan selama proses pelatihan dengan algoritma genetika.

4. Ramalan harga valas yang dihasilkan tidak dibandingkan dengan metode peramalan lainnya.

1.4 Tujuan

Tujuan dari penelitian ini adalah :

1. Menerapkan algoritma genetika pada pelatihan jaringan syaraf tiruan untuk meramalkan harga valuta asing.
2. Menghitung tingkat kesalahan dari ramalan harga valas yang dihasilkan oleh jaringan syaraf tiruan dengan menerapkan algoritma genetika.
3. Menentukan parameter-parameter algoritma genetika yang mempengaruhi hasil ramalan harga valuta asing GBP/USD.

1.5 Manfaat

Adapun manfaat dari penelitian ini adalah menghasilkan sistem peramalan yang dapat memberikan ramalan harga valuta asing USD terhadap GBP sehingga dapat membantu perusahaan maupun investor individual untuk memberikan pertimbangan apakah akan menjual atau membeli valuta asing yang dimiliki.

1.6 Sistematika Penulisan

Pembuatan skripsi ini disusun berdasarkan sistematika penulisan sebagai berikut:

1. BAB I PENDAHULUAN
   Berisi latar belakang masalah penelitian, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.
2. BAB II TINJAUAN PUSTAKA
   Menguraikan teori – teori yang erat hubunganya dengan valuta asing, algoritma genetika, dan jaringan syaraf tiruan untuk melakukan peramalan.
3. BAB III METODOLOGI PERANCANGAN
   Pada bab ini akan dijelaskan mengenai metode – metode yang digunakan dalam pelatihan jaringan syaraf tiruan menggunakan algoritma genetika untuk meramalkan harga valuta asing.

4. BAB IV HASIL DAN PEMBAHASAN
   Dalam bab ini akan dijelaskan mengenai implementasi program, pengujian dan analisa hasil penelitian.
5. BAB V KESIMPULAN DAN SARAN
   Berisi kesimpulan dari seluruh rangkaian penelitian serta saran kemungkinan pengembangannya.

UNIVERSITAS BRAWIJAYA

BAB II
TINJAUAN PUSTAKA

2.1 Peramalan (*Forecasting)*

Peramalan adalah menduga atau memperkirakan suatu keadaan dimasa yang akan datang berdasarkan keadaan masa lalu dan sekarang yang diperlukan untuk menetapkan kapan suatu peristiwa akan terjadi, sehingga tindakan yang tepat dapat dilakukan. Peramalan merupakan alat bantu yang penting dalam perencanaan yang efektif dan efisien. Peramalan dapat membantu untuk mengurangi ketidakpastian dalam melakukan perencanaan. Oleh karena itu peramalan memegang peranan penting dalam perencanaan dan pengambilan keputusan di berbagai bidang (Makridakis dkk, 1999).

Ada dua hal pokok yang harus diperhatikan dalam proses peramalan yang akurat dan bermanfaat (Makridakis dkk, 1999):

1. Pengumpulan data yang relevan yang berupa informasi yang dapat menghasilkan peramalan yang akurat.
2. Pemilihan teknik peramalan yang tepat yang akan memanfaatkan informasi data yang diperoleh semaksimal mungkin.

Situasi peramalan sangat beragam dalam horison waktu peramalan, faktor yang menentukan hasil sebenarnya, tipe pola data dan aspek lainnya. Untuk menghadapi penggunaan yang luas seperti itu, teknik peramalan dibagi menjadi dua kategori utama yaitu metode kuantitatif dan metode kualitatif. Metode kuantitatif dapat dibagi ke dalam deret berkala (*time series*) dan metode kausal, sedangkan metode kualitatif dapat dibagi menjadi metode eksploratoris dan normatif (Makridakis dkk, 1999).

Pemodelan deret berkala (*time series*) seringkali dikaitkan dengan peramalan suatu nilai karakteristik tertentu pada periode yang akan datang, melakukan pengendalian suatu proses ataupun mengenali pola perilaku sistem. Dengan mendeteksi pola dan kecenderungan data *time series*, kemudian memformulasikannya dalam suatu model, maka dapat digunakan untuk meramalkan data yang akan datang. Model dengan akurasi yang tinggi akan menyebabkan nilai ramalan cukup valid untuk digunakan sebagai pendukung dalam proses pengambilan keputusan (Makridakis dkk., 1999).

2.2 Pengenalan Valuta Asing

Valuta asing atau yang disingkat dengan kata "valas" secara bebas dapat diartikan sebagai mata uang yang dikeluarkan dan digunakan sebagai alat pembayaran yang sah di negara lain. (Berlianta,2005). Valas dibedakan menjadi dua kategori utama, yaitu *hard currency* dan *soft currency. Hard currency* adalah mata uang yang sering digunakan sebagai alat pembayaran dan alat satuan hitung dalam transaksi internasional. Valas yang tergolong dalam kategori ini biasanya mata uang yang nilainya relatif stabil dan cenderung mengalami apresiasi dibandingkan dengan nilai mata uang lainnya. Mata uang semacam ini umumnya dimiliki oleh negara-negara industri maju seperti Dolar AS, Yen Jepang, Deutchmark Jerman, Poundsterling Inggris, dan Euro dari himpunan negara-negara Eropa (Mulyo,2002).

Soft currency adalah mata uang yang jarang digunakan sebagai alat pembayaran transaksi internasional. *Soft currency* memiliki nilai yang lemah, tidak stabil, dan cenderung mengalami depresiasi dibandingkan dengan nilai mata uang negara-negara maju. Contoh mata uang yang termasuk dalam kategori *soft currency* antara lain Rupiah Indonesia, Peso Philipina, Rupee India, dan Bath Thailand (Mulyo,2002).

Berikut ini adalah beberapa mata uang yang termasuk *hard currency* (Berlianta, 2005) :

a. Poundsterling Inggris (GBP)

Mata uang ini disebut juga *PetroCurrency* karena berdasarkan fluktuasi harga masa lalu, pergerakan GBP sangat sensitive dengan pergerakan harga minyak bumi. Sebelum perang dunia kedua, mata uang ini merupakan mata uang utama dunia. Dimana negara-negara di dunia memakainya untuk melakukan transaksi perdagangan dan keuangan internasional. Setelah perang dunia kedua kedudukannya diambil alih oleh US Dollar.

b. Dollar Amerika Serikat (USD)

US Dollar (United States dollar) adalah mata uang utama dunia, ukuran universal untuk mengevaluasi mata uang lain yang diperdagangkan di Forex. Semua mata uang secara umum dikutip dalam US Dollar. Dalam kondisi kerusuhan ekonomi dan politik internasional, US Dollar adalah mata uang utama yang paling aman, yang terbukti sangat baik selama krisis Asia Tenggara

1997-1998. Seperti yang telah ditunjukkan, US Dollar menjadi mata uang utama menjelang akhir Perang Dunia Kedua di sepanjang Bretton Woods Accord. Pengenalan Euro pada tahun 1999 hanya sedikit mengurangi pentingnya US Dollar.

2.2.1 Perdagangan Valuta Asing

Perdagangan valuta asing (*Trading Forex*) adalah perdagangan mata uang dari negara yang berbeda satu sama lain. Tujuan investor dalam perdagangan *forex* adalah mendapatkan keuntungan dari pergerakan mata uang asing. *Forex trading* selalu dilakukan dalam pasangan mata uang. Sebagai contoh, di Inggris mata uang beredar disebut Poundsterling (GBP) dan di Amerika Serikat, mata uang yang beredar disebut Dolar AS (USD). Sebuah contoh dari perdagangan forex adalah membeli GBP, sementara secara bersamaan menjual USD. Hal ini akan disingkat GBP/USD (Hajar, 2009).

Salah satu aktivitas penting yang dilakukan oleh pelaku pasar dalam pasar valuta asing adalah melakukan analisis untuk memprediksi arah kurs valuta asing di masa yang akan datang. Prediksi kurs valuta asing sangatlah penting dilakukan agar para pelaku pasar dapat menghindari kerugian dan bahkan dapat memperoleh keuntungan dari pergerakan kurs valuta asing di pasar. Sampai saat ini ada dua tipe analisis yang digunakan untuk memprediksi kurs valuta asing di masa yang akan datang, yaitu analisis fundamental dan analisis teknikal (Berlianta,2005).

2.2.2 Teknik Peramalan Harga Valuta Asing

Peramalan harga valuta asing dapat dilakukan melalui dua metode, yaitu analisis fundamental dan analisis teknikal. Analisis fundamental adalah suatu cara atau metode peramalan pergerakan kurs valuta asing yang berdasarkan pada pengenalan dan pengukuran faktor-faktor yang mempengaruhi pergerakan kurs valuta asing. Faktor-faktor tersebut biasanya merupakan faktor fundamental ekonomi, politik, kuangan dan lain-lain (Berlianta, 2005).

Analisis teknikal adalah suatu metode peramalan pergerakan harga dengan menggunakan informasi yang terkandung pada pergerakan harga di masa lalu. Berbeda dengan analisis fundamental yang tidak percaya bahwa informasi yang terkandung pada pergerakan harga di masa lalu dapat memprediksi pergerakan harga

di masa yang akan datang maka analisis teknikal percaya bahwa informasi pada pergerakan harga masa lalu dapat memprediksi pergerakan harga di masa yang akan datang (Berlianta, 2005).

Saat ini teknik peramalan yang sering digunakan adalah analisis teknikal. Menurut seorang analis pasar, Martin J. Pring, analisis teknikal bertujuan untuk mencari dan mengidentifikasi kecenderungan (*trend*) berdasarkan perubahan perilaku para pelaku pasar dan investor dalam menyingkapi berbagai peristiwa yang mempengaruhi pasar (Darmawan, 2007). Peristiwa ini biasanya mempengaruhi pola harga *open, low, high* dan *close*. Para analis pasar memiliki tanggapan bahwa perilaku para pelaku pasar dalam menanggapi peristiwa ekonomi yang terjadi di masa lalu sama dengan perilaku masa kini, sehingga terjadi kesamaan pola data atau grafik. Kesamaan pola data ini dapat dimanfaatkan untuk meramalkan pergerakan harga di masa yang akan datang.

Peramalan harga valuta asing dengan melihat pola pergerakan harga valuta asing bukanlah hal yang mudah, karena pergerakan harga valuta asing memiliki variasi yang tinggi dan polanya tidak teratur sehingga sulit untuk diramalkan. Beberapa teknik peramalan konvensional mampu menyediakan prediksi untuk beberapa kasus peramalan, namun teknik tersebut tidak tepat untuk sistem non-stasioner seperti kurs valuta asing, suku bunga dan harga saham. Metode yang secara luas digunakan untuk peramalan *time series* adalah Box-Jenkins Auro-Regresive Integrated Moving Average (ARIMA), namun ARIMA berkembang berdasarkan asumsi bahwa data *time series* yang diramalkan bersifat linear dan stasioner. Hal ini menyebabkan ARIMA bukanlah metode yang tepat untuk prediksi harga valuta asing (Kamruzzaman, 2004).

Berdasarkan penelitian yang dilakukan oleh Zang dan Hu, Yao dkk serta beberapa peneliti lainnya, membuktikan bahwa *Artificial Neural Network* (ANN) lebih baik daripada metode ARIMA. Zang dan Hu menganalisa kemampuan jaringan *backpropagation* untuk meramalkan kurs valuta asing. Yao dkk mengevaluasi kemampuan jaringan *backpropagation* sebagai perangkat peramalan harga. ANN, sebuah fungsi aproksimasi terkenal dalam peramalan dan *system modeling*, telah menunjukkan kemampuan yang baik dalam analisis *time series* dan peramalan. ANN membantu analisis *multivariate*. Model *multivariate* dapat

mengandalkan informasi yang tidak hanya berasal dari waktu selang *time series* yang akan diramalkan, namun juga dapat menggunakan indikatornya seperti indikator teknikal, dan fundamental untuk pasar finansial. Selain itu ANN lebih efektif untuk menggambarkan data *time series* yang non-stasioner dan non-linier (Kamruzzaman, 2004).

2.3 Jaringan Syaraf Tiruan

Artificial Neural Network (ANN) atau disebut juga jaringan syaraf tiruan (JST**)** adalah sistem komputasi dimana arsitektur dan operasi diilhami dari pengetahuan tentang sel saraf biologi di dalam otak. JST dapat digambarkan sebagai model matematis dan komputasi untuk fungsi aproksimasi non-linier, klasifikasi data, *cluster* dan regresi non-parametrik atau sebuah simulasi dari koleksi model syaraf biologi (Kristanto,2004).

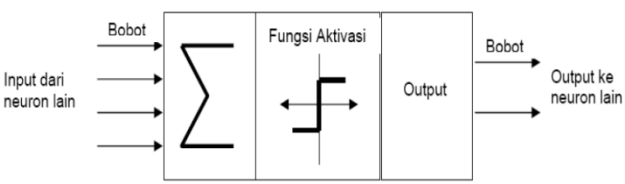
Jaringan syaraf tiruan memiliki 3 karakteristik utama, yaitu (Desiani, 2006) :

1. Arsitektur Jaringan
   Merupakan pola keterhubungan antara neuron. Hubungan neuron-neuron inilah yang membentuk suatu jaringan.
2. Algoritma Pembelajaran
   Merupakan metode yang digunakan untuk menentukan bobot dari hubungan neuron.
3. Fungsi Aktivasi
   Merupakan fungsi untuk menentukan nilai keluaran berdasarkan nilai total masukan pada neuron.

2.3.1 Model Umum Jaringan Syaraf Tiruan

Secara umum, jaringan syaraf tiruan terdiri dari beberapa neuron dan terdapat hubungan antara neuron-neuron tersebut yang dikenal dengan bobot. Neuron akan mentransformasikan informasi atau *input* yang diterima melalui hubungan tersebut ke neuron lainnya. *Input* ini akan diproses oleh suatu fungsi perambatan yang akan menjumlahkan nilai-nilai semua bobot yang datang. Hasil penjumlahan kemudian dibandingkan dengan suatu nilai ambang (*threshold*) tertentu melalui fungsi aktivasi. Apabila fungsi tersebut melewati nilai ambang, maka akan diaktifkan, begitu pula sebaliknya. Neuron yang diaktifkan akan mengirimkan *output* melalui bobot-bobot *output* yang berhubungan dengannya

(Kusumadewi, 2003). Struktur jaringan syaraf tiruan secara umum dapat dilihat pada Gambar 2.1



Gambar 2.1 Struktur neuron jaringan syaraf tiruan

Umumnya, jaringan syaraf tiruan memiliki beberapa tipe, namun hampir semuanya memiliki komponen-komponen yang sama. Komponen-komponen utama dari jaringan syaraf tiruan antara lain sebagai berikut (Puspitaningrum, 2006) :

1. Neuron atau node atau unit
   Sel syaraf tiruan yang merupakan elemen pengolah jaringan syaraf tiruan. Setiap neuron menerima *input*, melakukan pengolahan *input* dan mengirimkan hasilnya berupa sebuah *output*.
2. *Input* atau masukan
   Berkoresponden dengan sebuah atribut tunggal dari sebuah pola atau data lain dari dunia luar. Sinyal-sinyal *input* yang diterima akan diteruskan ke lapisan selanjutnya.
3. Bobot
   Merupakan nilai matematis dari koneksi, yang mentransfer data dari satu lapisan ke lapisan lainnya. Bobot digunakan untuk mengatur jaringan sehingga dapat melakukan pembelajaran untuk menghasilkan *output* yang diinginkan.
4. Fungsi penjumlah
   Fungsi ini menjumlahkan masukan-masukan yang diterima berdasarkan bobot dari masukan tersebut. Masukan yang diterima dikalikan dengan bobotnya lalu hasil seluruh perkalian tersebut dijumlahkan. Fungsi penjumlah dapat didefinisikan melalui Persamaan 2.1 ,

$$net = \sum_{i=0}^{n} w_i x_i \quad (2.1)$$

yang dalam hal ini net adalah hasil keluaran dari fungsi penjumlahan, w_i menyatakan bobot koneksi masukan ke- i, dan x_i menyatakan masukan dari bobot tersebut.

5. Fungsi aktivasi

Fungsi aktivasi adalah fungsi yang menentukan keluaran sebuah neuron dari hasil penjumlahan yang didapat melalui Persamaan 2.1. Fungsi aktivasi ini dilambangkan dengan notasi σ. Terdapat beberapa jenis fungsi aktivasi, yaitu (Kusumadewi, 2003):

a. Fungsi linier

Nilai keluaran neuron sama dengan hasil penjumlahan yang didapat melalui Persamaan 2.2.

$$\sigma (net) = net \quad (2.2)$$

yang dalam hal ini menyatakan nilai hasil penjumlahan yang didapat melalui Persamaan 2.1.

b. Fungsi ambang (*threshold*)

Nilai keluaran neuron dikeluarkan secara diskrit jika nilai hasil penjumlahan dari Persamaan 2.1 melebihi nilai ambang tertentu. Dalam penggunaan fungsi ambang, biasanya batasan nilai tersebut adalah nol karena nilai batasan telah ikut diperhitungkan dari adanya bobot bias yang dimiliki unit neuron. Fungsi ambang dapat didefinisikan sebagai Persamaan 2.3.

$$\sigma\,(net) = \begin{cases} 1 \text{ untuk net} > 0 \\ 0 \text{ untuk net} \leq 0 \end{cases} \quad (2.3)$$

c. Fungsi sigmoid

Nilai keluaran dipetakan dari rentang (-∞,+∞) menjadi bilangan riil dengan rentang antara [0,1] dengan menggunakan fungsi sigmoid biner. Sedangkan nilai keluaran yang dipetakan menjadi bilangan riil dengan rentang antara [-1,1] menggunakan fungsi sigmoid bipolar. Fungsi ini dipilih agar pembelajaran yang menggunakan turunan dari fungsi aktivasi

dapat menggunakan fungsi yang kontinu. Fungsi sigmoid biner dapat didefinisikan melalui Persamaan 2.4.

$$\sigma(net) = \frac{1}{1 + e^{-net}} \quad (2.4)$$

Fungsi sigmoid bipolar dapat didefinisikan melalui Persamaan 2.5.

$$\sigma(net) = \frac{1 - e^{-net}}{1 + e^{-net}} \quad (2.5)$$

6. *Output* atau keluaran

   Koneksi keluaran mengirimkan neuron ke neuron-neuron lainnya atau sebagai keluaran dari jaringan syaraf tiruan. *Output* atau keluaran merupakan solusi atau hasil pemahaman jaringan terhadap data *input*.

2.3.2 Arsitektur Jaringan Syaraf Tiruan

Arsitektur adalah rancangan untuk segala tipe struktur, baik fisik maupun kontekstual, nyata maupun tidak nyata (O'Rourke, 2003). Arsitektur pada dasarnya menggambarkan bentuk konstruksi sistem yang diwujudkan dalam sebuah model yang dilihat dari beberapa sudut pandang yang berbeda (Kurniawan, 2011). Arsitektur dalam jaringan syaraf tiruan merupakan pola keterhubungan antara sebuah neuron dengan neuron lainnya yang membentuk suatu jaringan (Desiani, 2006).

Umumnya, arsitektur jaringan syaraf tiruan terdiri atas beberapa lapisan yang masing-masing lapisan memiliki jumlah node atau neuron yang berbeda-beda. Lapisan penyusun JST dapat dibagi menjadi tiga, yaitu unit-unit (Puspitaningrum,2006) :

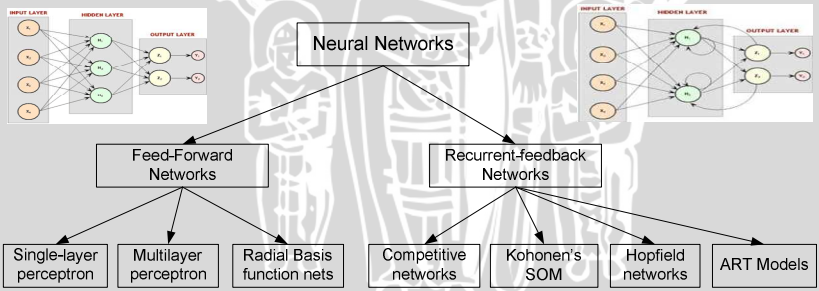
1. Lapisan masukan (*input layer*)

   Node-node di dalam lapisan masukan disebut unit-unit lapisan *input*. Unit-unit *input* menerima *input* dari dunia luar, yang mana *input* tersebut merupakan penggambaran dari suatu masalah yang ingin diselesaikan.

2. Lapisan tersembunyi
   Node-node di dalam lapisan tersembunyi disebut unit-unit tersembunyi atau *hidden node*. Aktifitas setiap unit-unit lapisan tersembunyi ditentukan oleh aktifitas dari unit-unit masukan dan bobot dari koneksi antara unit-unit masukan dan unit-unit lapisan tersembunyi. *Output* dari lapisan ini tidak dapat diamati secara langsung.
3. Lapisan keluaran
   Node-node pada lapisan keluaran disebut unit-unit *output*. Keluaran atau *output* dari lapisan ini merupakan *output* jaringan syaraf tiruan terhadap suatu permasalahan.

Lapisan masukan terhubung dengan lapisan tersembunyi yang selanjutnya terhubung dengn lapisan keluaran.

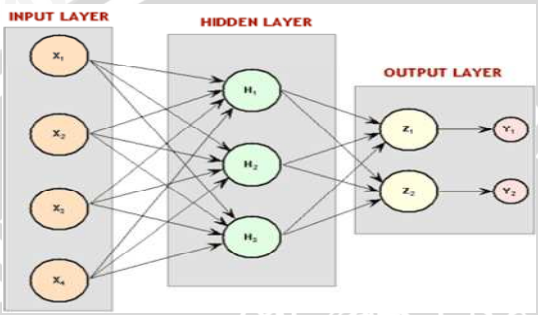
Mengenai pengelompokan jaringan syaraf tiruan, ada yang membagi ke dalam 2 kelompok, yaitu jaringan syaraf tiruan umpan maju (*feed-forward networks*) dan jaringan syaraf tiruan berulang/umpan balik (*recurrent / feedback networks*). Jaringan syaraf tiruan umpan maju adalah graf yang tidak mempunyai *loop*. Sedangkan jaringan syaraf tiruan berulang/ umpan balik dicirikan dengan adanya *loop-loop* koneksi balik. Jika mengikuti aturan ini, taksonomi JST akan tampak seperti pada Gambar 2.2 (Puspitaningrum, 2006).



Gambar 2.2 Taksonomi JST beserta nama-nama algoritma pembelajaran yang termasuk ke dalamnya

2.3.3 Jaringan Umpan Maju (*Feed-Forward Networks*)

Jaringan umpan maju (*Feed-Forward Networks*) membentuk dua atau lebih lapisan yang hanya memiliki alur sinyal maju. Skema arsitektur *feed-forward neural network* ditunjukkan pada Gambar 2.3.



Gambar 2.3 Jaringan *feed-forward* dengan 4 *input* dan 2 *output* (Sumber : Jones,2008)

Pelatihan jaringan pada model FFNN menggunakan algoritma *backpropagation* yaitu tahap umpan maju (*feedforward*) dari pola *input*. Algoritma *feed-forward backpropagation* adalah sebagai berikut (Siang, 2005) :

1. Inisialisasi bobot
   Bobot awal ditentukan secara acak dengan nilai sekecil mungkin.
2. Untuk setiap pasang data pelatihan, lakukan langkah-langkah berikut :
   1. Setiap unit *input* (x_i, i=1,2,…n) menerima sinyal dan menyalurkan sinyal ini ke semua unit lapisan tersembunyi.
   2. mengalikan nilai *input* dan bobot sinyal *input*, lalu dijumlahkan di setiap unit tersembunyi (z_j, j=1,2,…p) dengan Persamaan 2.6.

$$z_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \qquad (2.6)$$

Keterangan :
z_in_j = *input* jaringan ke *hidden layer* z
v_{0j} = bobot awal bias ke *hidden layer* j

n = jumlah unit *input*
v_{ij} = bobot sinyal *input* menuju *hidden layer* j
x_i = nilai *input*

3. Menghitung *output* menggunakan fungsi aktivasi sigmoid bipolar, yaitu dengan menggunakan Persamaan 2.7.

$$z_j = f\left(z_{in_j}\right) = \frac{1 - e^{-z_in_j}}{1 + e^{-z_in_j}} \quad (2.7)$$

Keterangan :
z_j = unit *hidden layer* j
z_in_j = *input* jaringan ke *hidden layer* z

lalu kirimkan sinyal tersebut ke semua unit di lapisan atasnya (lapisan *output*)

4. Setiap unit *output* (y_k; k=1,2,…m) akan menjumlahkan bobot sinyal *output* dengan Persamaan 2.8.

$$y_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk} \quad (2.8)$$

Keterangan :
y_in_k = *input* jaringan ke lapisan *output*
w_{0k} = bobot awal bias ke *output*
p = jumlah unit tersembunyi
z_j = unit tersembunyi j
w_{jk} = bobot menuju *output*

Gunakan fungsi aktivasi untuk menghitung sinyal *output*, ditunjukkan pada Persamaan 2.9.

$$y_k = f(y_{in_k}) = y_{in_k} \quad (2.9)$$

Keterangan :
y_k = unit *output*
y_{in_k} = *input* jaringan ke *output*

Setelah proses pelatihan, tahap pengujian data dilakukan dengan menggunakan tahap perambatan maju (*feedfoward*) yang sama

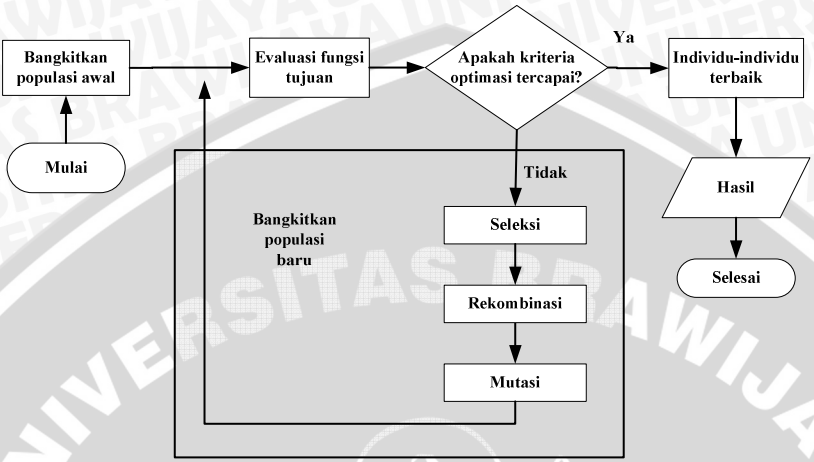
seperti tahap pelatihan. Inisialisasi bobot pada tahap pengujian menggunakan bobot yang dihasilkan pada proses *training* sebelumnya (Kusumadewi, 2003).

2.4 Algoritma Genetika

Algoritma genetik diperkenalkan oleh Holand (1975), sebagai model umum yang diterapkan pada optimasi. Algoritma genetik adalah suatu metode algoritma pengoptimalan dan pencarian yang didasarkan pada prinsip genetika dan seleksi natural. Algoritma genetika mengizinkan sebuah kromosom dalam suatu populasi untuk berkembang dibawah suatu aturan hinggga mencapai nilai fungsi maksimal yang ditentukan (Haupt and Haupt, 2004).

Kromosom adalah sesuatu yang merepresentasikan solusi potensial dari suatu permasalahan. Setiap kromosom akan dievaluasi untuk menentukan nilai *fitness*-nya, *fitness* adalah tolak ukur suatu kromosom yang dihitung berdasarkan fungsi objektifnya. Kromosom tersebut akan mengalami proses evolusi dalam sejumlah iterasi pembentukan generasi baru melalui proses *mutasi* dan *crossover* dimana *mutasi* hanya akan mempengaruhi sebuah kromosom sedangkan *crossover* akan mempengaruhi dua kromosom asal menjadi dua kromosom baru melalui proses persilangan yang disebut dengan *offspring*. Setiap kromosom baru tersebut akan dievaluasi nilai kelayakannya untuk masuk dalam populasi. Proses tersebut akan diteruskan sampai kondisi tertentu dan hasil kromosom pada populasi dengan nilai *fitness* terbaik diharapkan dapat mewakili keadaan optimal dari suatu permasalahan (Gen dan Cheng, 2000).

Adapun *flowchart* dari mekanisme algoritma genetika secara umum digambarkan pada Gambar 2.4 (Kusumadewi, 2003).



Gambar 2.4 Diagram algoritma genetika secara umum (Sumber : Kusumadewi, 2003)

2.4.1 Pengkodean Kromosom

Representasi merupakan permasalahan yang menjadi salah satu kunci keberhasilan dalam algoritma genetika, karena algoritma ini secara langsung memanipulasi kode yang merupakan representasi dari permasalahan, sehingga pemilihan representasi sangat menentukan kualitas hasil penyelesaian (Nurwijaya,2007).

Terdapat tiga skema yang paling umum digunakan dalam pengkodean, yaitu (Suyanto, 2005) :

a. Real-number encoding.

Pada skema ini, nilai gen berada dalam interval [0,R], dimana R adalah bilangan real positif dan biasanya nilai R=1. Pengkodean kromosom dengan bilangan real dapat dilihat pada Gambar 2.5.

0,768	1,000	0,056

Gambar 2.5 Contoh individu dengan *real-number encoding*

b. Discrete decimal encoding.

Pada pengkodean ini, setiap gen bisa bernilai salah satu bilangan bulat dalam interval [0,9]. Pengkodean kromosom dengan bilangan desimal dapat dilihat pada Gambar 2.6.

5	1	3	4	2

Gambar 2.6 Contoh individu dengan *discrete decimal encoding*

c. Binary encoding.

Pada pengkodean ini, setiap gen direpresentasikan sebagai bilangan biner, sehingga hanya bisa bernilai angka 0 atau 1. Pengkodean kromosom dengan bilangan biner dapat dilihat pada Gambar 2.7.

1	1	0	1	0

Gambar 2.7 Contoh individu dengan *binary encoding*

2.4.2 Fungsi *fitness*

Fungsi *fitness* merupakan suatu fungsi yang digunakan untuk menghitung *fitness* atau tingkat kebaikan suatu individu untuk bertahan hidup. Fungsi ini mengambil parameter berupa individu dan menghasilkan keluaran nilai *fitness* dari individu tersebut. Untuk setiap permasalahan yang akan diselesaikan dengan algoritma genetika harus terdefinisi suatu *fitness function* (Nurwijaya, 2007).

Di dalam evolusi alam, individu yang mempunyai nilai *fitness* tinggi akan bertahan hidup, sedangkan individu yang mempunyai nilai *fitness* rendah akan mati. Penentuan nilai *fitness* sangat berpengaruh pada performasi algoritma genetika secara keseluruhan (Suyanto, 2005).

Pada masalah optimasi, jika masalahnya adalah meminimalkan fungsi *h*,maka fungsi *h* tidak bisa digunakan secara langsung. Hal ini disebabkan adanya aturan bahwa individu yang memiliki nilai *fitness* tinggi lebih mampu bertahan hidup pada generasi berikutnya. Maka fungsi *fitness* yang digunakan adalah seperti pada Persamaan 2.10.

$$f = \frac{1}{(h + a)} \qquad (2.10)$$

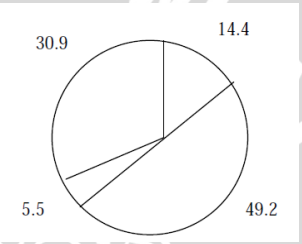
dimana *a* adalah bilangan yang dianggap sangat kecil dan bervariasi sesuai dengan masalah yang akan diselesaikan (Suyanto, 2005).

2.4.3 Operator Genetika

1. *Select Parent*

Proses pemilihan dua individu sebagai *parent* biasanya dilakukan secara proporsional berdasarkan nilai-nilai *fitness*-nya. Salah satu metode seleksi yang umum digunakan adalah *roulette-wheel* . Sesuai dengan namanya, metode ini menirukan permainan *roulette-wheel* dimana masing-masing individu menempati potongan lingkaran pada roda *roulette* secara proporsional sesuai dengan nilai *fitness*-nya (Suyanto, 2007). Ilustrasi penggunaan *roulette wheel* ditunjukkan pada Gambar 2.8.

No	String	Fitness	%total
1	01101	169	14.4
2	11000	576	49.2
3	01000	64	5.5
4	10011	361	30.9
Total		1170	100

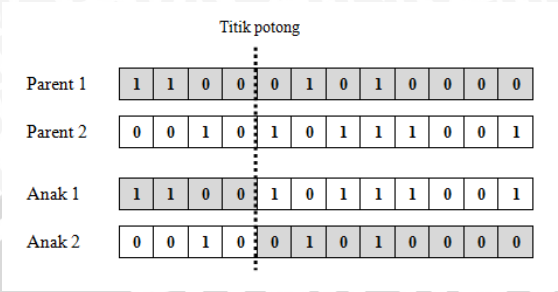


Gambar 2.8 Contoh seleksi orang tua menggunakan metode *roulette wheel*
(Sumber : Desiani, 2006)

Berdasarkan Gambar 2.8 dapat dilihat bahwa semakin besar nilai *fitness* suatu individu maka akan menempati daerah yang lebih luas dibandingkan dengan individu lain yang *fitness*-nya lebih rendah, dengan demikian peluang individu untuk terpilih menjadi *parent* akan semakin besar.

2. *Crossover*

Proses *crossover* pada algoritma genetika menghasilkan variasi atau keanekaragaman kromosom dalam satu populasi dengan menghasilkan *offspring* yang merupakan kombinasi dari 2 kromosom *parents*. Proses ini dimulai dengan memilih 2 kromosom *parent*, dan setelah disilangkan setidaknya menghasilkan satu *offspring* (Rao, 2010). Ada beberapa teknik *crossover* yang dapat digunakan antara lain *one-cut-point crossover*, *n-point crossover* dan *uniform crossover* (Suyanto, 2005). Contoh *crossover* dapat dilihat pada Gambar 2.9.



Gambar 2.9 Contoh *crossover* metode *one-cut-point crossover* (Sumber : Suyanto, 2005)

3. *Mutation*

Mutasi bertujuan untuk mengubah nilai gen secara acak. Jika kromosom berbentuk biner, maka akan mengubah secara acak nilai suatu bit pada posisi tertentu, kemudian mengganti satu atau beberapa nilai gen dari kromosom. Jika kromosom berbentuk real maka akan mengubah nilai gen dengan mengurangkan dan menambahkan nilai gen yang dipilih secara acak, atau dengan mengganti gen tersebut dengan nilai baru yang ditentukan secara acak. Pada mutasi ini sangat dimungkinkan munculnya kromosom baru yang semula belum muncul dalam populasi awal (Triyansyah, 2010). Contoh mutasi ditunjukkan pada Gambar 2.10.

Kromosom sebelum mutasi	1,43	1,09	4,51	9,11	6,94	
Kromosom sesudah mutasi	1,43	**1,19**	4,51	**9,01**	9,64	**(a)**
Kromosom sesudah mutasi	1,43	3,78	4,51	9,11	6,94	**(b)**

Gambar 2.10 Contoh mutasi dengan menambah dan mengurangi konstanta (a) dan penggantian gen dengan bilangan acak baru (b) (Sumber : Desiani, 2006)

2.4.4 Parameter Genetika

Parameter genetik berguna dalam pengendalian operator-operator genetik yang digunakan. Pemilihan nilai-nilai parameter genetik sangat berpengaruh terhadap kinerja algoritma genetik. Parameter yang sering digunakan adalah sebagai berikut :

1. Ukuran Populasi

Ukuran populasi sangat mempengaruhi unjuk kerja dan keefektifan algoritma genetika. Jika populasi yang dibentuk terlalu kecil maka unjuk kerjanya buruk, karena populasi tersebut kurang bisa menyediakan ruang solusi yang dapat merepresentasikan keseluruhan ruang persoalan. Jika jumlah populasi terlalu banyak, maka akan membutuhkan waktu komputasi yang lama dalam proses pencariannya (Desiana, 2006).

2. Probabilitas *Crossover* (Pc)

Probabilitas *crossover* adalah perbandingan jumlah *offspring* (kromosom) yang dihasilkan pada setiap generasi terhadap jumlah kromosom pada satu populasi (Gen dan Cheng, 2000). Dalam setiap populasi sebanyak probabilitas *crossover* * Jum.Populasi kromosom melakukan *crossover*. Misalkan, ukuran populasi sebanyak 100 kromosom, sedangkan probabilitas *crossover* sebesar 0,25, berarti bahwa diharapkan sebanyak 25 kromosom dari 100 kromosom akan mengalami *crossover* (Kusumadewei, 2003). Peluang *crossover* tidak memiliki aturan khusus dalam penentuannya, namun karena *crossover* merupakan operator genetik utama maka pada umumnya probabilitas *crossover* yang digunakan besar sehingga memungkinkan timbulnya domain pencarian baru yang lebih luas (Saputro, 2003).

3. Probabilitas Mutasi (Pm)

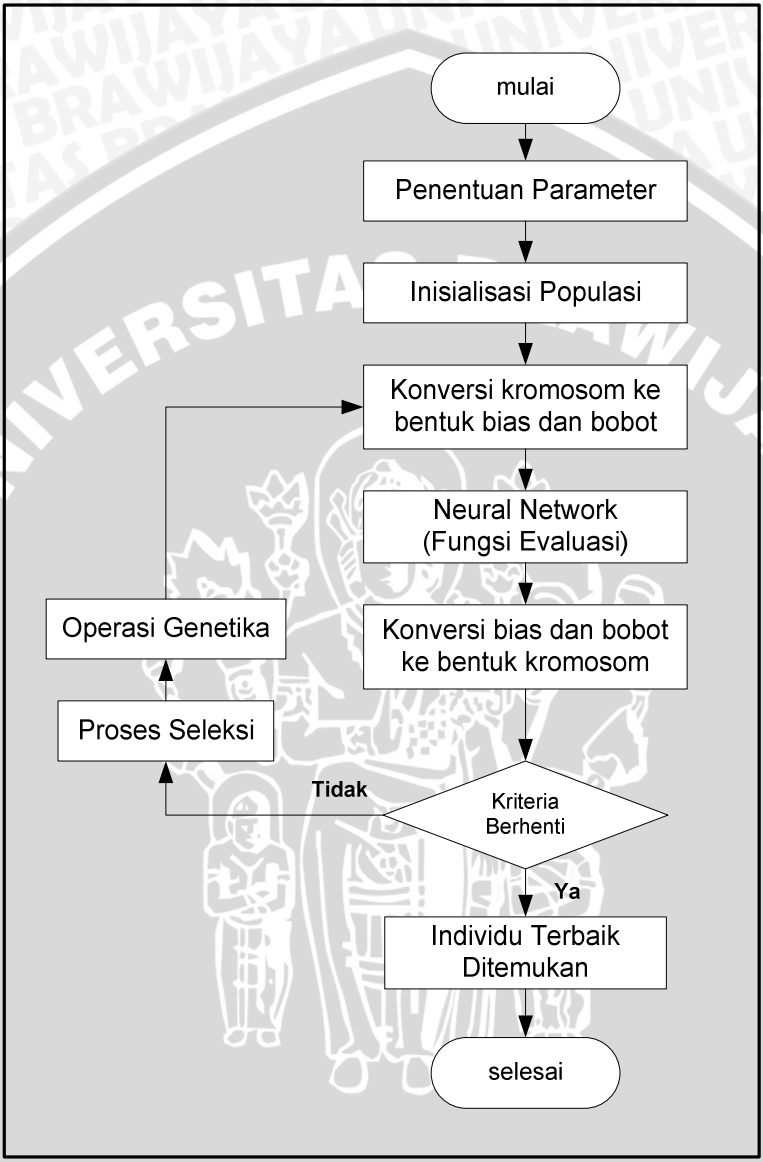
Probabilitas mutasi akan mengendalikan operator mutasi pada setiap generasi. Jika peluang mutasi terlalu kecil, banyak gen yang mungkin berguna tidak dievaluasi, tetapi bila peluang mutasi ini terlalu besar maka akan terlalu banyak gangguan acak, sehingga anak akan kehilangan kemiripan dari induknya dan algoritma juga akan kehilangan kemampuan untuk belajar dari *history* pencarian (Kusumadewi, 2003). Probabilitas mutasi yang digunakan umumnya lebih kecil daripada probabilitas *crossover* karena operator ini dapat menimbulkan perubahan yang cukup besar terhadap titik pencarian. Perubahan inilah yang memungkinkan algoritma genetik memeriksa

titik pencarian lainnya diluar titik pencarian yang sedang dievaluasi sehingga memungkinkan algoritma genetik keluar dari lokal optimal (Saputro, 2003).

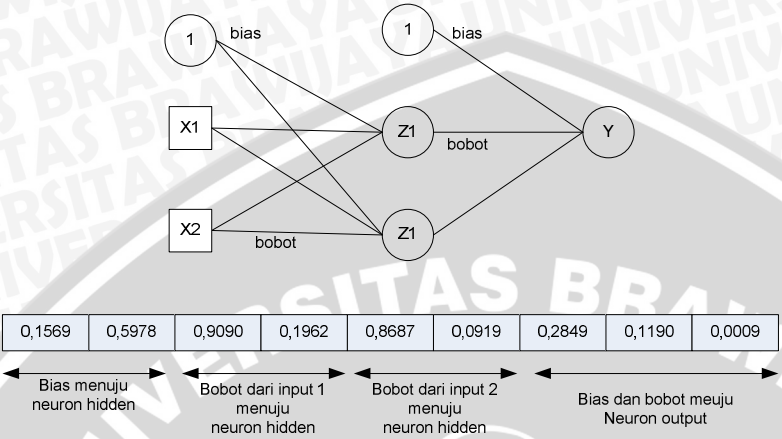
2.5 Pelatihan JST Menggunakan Algoritma Genetika

Algoritma genetika dapat diterapkan pada jaringan syaraf tiruan untuk melatih jaringan sehingga dapat digunakan untuk menyelesaikan suatu permasalahan. Diagram alir penerapan algoritma genetika pada jaringan syaraf tiruan dapat dilihat pada Gambar 2.11 (Noertjahya, 2002).

Hal penting yang perlu diperhatikan dalam penerapan algoritma genetika pada pelatihan jaringan syaraf tiruan adalah bagaimana melakukan pengkodean kromosom yaitu konversi bias dan bobot jaringan ke dalam bentuk individu pada algoritma genetik. Salah satu cara yang dapat digunakan adalah dengan menggabungkan bias dan bobot yang ada pada jaringan syaraf tiruan menjadi suatu *string* yang mewakili individu pada algoritma genetika, dan untuk menkonversikannya kembali ke dalam bentuk bias dan bobot dapat dilakukan pemisahan gen-gen suatu individu pada algoritma genetika menjadi bias dan bobot pada jaringan syaraf tiruan. Representasi dari gen-gen pada sebuah individu dapat menggunakan *real-number encoding* (Noertjahyana, 2002). Untuk lebih jelasnya, proses pengkodean kromosom untuk pelatihan jaringan syaraf tiruan ditunjukkan pada Gambar 2.11.



Gambar 2.11 Diagram alir pelatihan neural network dengan algoritma genetika



Gambar 2.12 Skema pengkodean kromosom untuk pelatihan jaringan syaraf tiruan
(Sumber : Suyanto, 2005)

Langkah selanjutnya adalah menentukan fungsi evaluasi atau *fitness* yang akan menentukan keoptimuman suatu individu. Untuk kasus pelatihan jaringan syaraf tiruan menggunakan algoritma genetika, fungsi *fitness* yang dapat digunakan adalah *invers* dari *mean-square deviation (delta)* antara *output* yang diharapkan dengan *output* yang dihasilkan oleh jaringan syaraf tiruan. Artinya nilai *fitness* untuk kasus ini adalah 1/*delta* , dimana *delta* dapat dihitung menggunakan Persamaan 2.11 (Suyanto, 2005).

$$delta = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(d(i) - y(i))^2} \qquad (2.11)$$

Keterangan :
m = jumlah data
d(i) = *output* yang diharapkan
y(i) = *output* yang dihasilkan oleh jaringan syaraf tiruan

2.6 Normalisasi dan Denormalisasi

Data *input* dan target yang akan diimplementasikan ke dalam jaringan syaraf tiruan harus terlebih dahulu dilakukan penskalaan atau normalisasi untuk menyesuaikan nilai *range* data dengan *logsigmoid threshold function* pada tahap *feedforward backpropagation*. Range *input* yang memenuhi syarat adalah data *input* pada range 0 sampai 1 atau -1 sampai 1. Proses normalisasi dapat dihitung dengan Persamaan 2.12 (Maru'ao, 2010).

$$\text{Nilai baru} = \frac{(\text{Nilai lama} - \text{Ratarata})}{\text{Deviasi Standar}} \quad (2.12)$$

Yang mana deviasi standar dapat dihitung menggunakan Persamaan 2.13.

$$\text{deviasi standar} = \sqrt{\frac{\sum(x-\bar{x})^2}{n-1}} \quad (2.13)$$

Keterangan :
$\bar{x}$ = Rata-rata nilai data
x = nilai data
n = jumlah data

Data hasil proses kemudian diubah kembali menjadi data aslinya melalui proses denormalisasi data. Proses denormalisasi data dapat dihitung dengan Persamaan 2.14 (Maru'ao, 2010).

$$Denormalisasi = (Nilai\ normalisasi * deviasi\ standar) + Ratarata \quad (2.14)$$

2.7 Evaluasi Hasil Peramalan

Suatu model peramalan dapat dikatakan baik setelah diketahui sejauh mana model tersebut mampu memprediksi data yang telah diketahui, untuk itu perlu dilakukan evaluasi kesesuaian model peramalan terhadap suatu kumpulan data yang diberikan. Dalam pemodelan deret waktu (*time series*), sebagian data yang diketahui dapat digunakan untuk meramalkan sisa data berikutnya sehingga memungkinkan seseorang untuk mempelajari ketepatan ramalan secara langsung. Untuk mengevaluasi ketepatan ramalan dapat digunakan beberapa metode diantaranya dengan ukuran

statistik standar yaitu *Root Mean Squared Error* (RMSE) dan *Mean Absolute Percentage Error* (MAPE) (Makridakis dkk., 1999).

Evaluasi jaringan dilakukan dengan mengetahui kesalahan yang dihasilkan oleh jaringan terhadap targetnya menggunakan RMSE dan MAPE. Jika X_i merupakan data aktual untuk periode *i* dan F_i merupakan ramalan yaitu keluaran yang dihasilkan oleh jaringan untuk periode yang sama, maka kesalahan dapat dihitung menggunakan Persamaan 2.15.

$$e_i = X_i - F_i \tag{2.15}$$

Keterangan :

e_i = kesalahan pada periode i

$X_i = t_i$ = nilai aktual harga valas atau target jaringan

$F_i = y_i$ = nilai ramalan atau *output* jaringan

Jika terdapat nilai pengamatan dan ramalan untuk *n* periode waktu, maka akan terdapat *n* buah kesalahan yang dapat dihitung menggunakan ukuran statistik standar pada Persamaan 2.16 dan Persamaan 2.17 (Makridakis dkk., 1999)

.

$$RMSE = \sqrt{\sum_{i=1}^{n} e_i^2 / n} \tag{2.16}$$

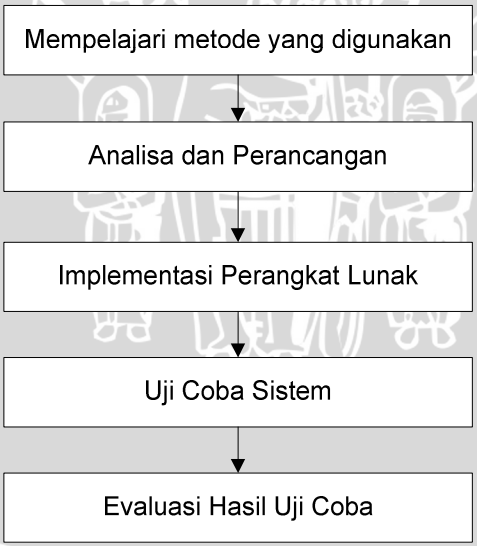
$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{e_i}{Xi}\right| * 100 \tag{2.17}$$

BAB III
METODOLOGI DAN PERANCANGAN

Pada bab metodologi dan perancangan ini akan dibahas metode, rancangan yang digunakan dan langkah – langkah yang dilakukan dalam penelitian untuk meramalkan harga valuta asing menggunakan jaringan syaraf tiruan dengan menerapkan algoritma genetika pada tahap pelatihan jaringan. Penelitian dilakukan dengan tahapan-tahapan sebagai berikut:

1. Mempelajari metode yang digunakan dari berbagai sumber seperti yang telah dijelaskan pada bab 2.
2. Menganalisa dan merancang perangkat lunak dengan metode yang akan digunakan.
3. Membuat perangkat lunak berdasarkan analisis dan perancangan yang dilakukan.
4. Uji coba peramalan harga valuta asing menggunakan perangkat lunak yang telah dibuat.
5. Evaluasi hasil uji coba.

Langkah – langkah penelitian ini ditunjukkan oleh Gambar 3.1



Gambar 3.1 Tahapan penelitian

3.1 Analisa Sistem

3.1.1 Deskripsi Sistem

Sistem yang akan dibuat merupakan sistem peramalan harga valuta asing menggunakan jaringan syaraf tiruan dengan menerapkan algoritma genetika pada proses pelatihan jaringan. Pada sistem ini diperlukan data masukan adalah data harga valuta asing (GBP/USD) berupa harga *open, high* dan *low*. Masukan lain yang digunakan adalah parameter algoritma genetika berupa jumlah generasi, ukuran populasi, probabilitas *crossover* dan probabilitas mutasi. Hasil keluaran sistem berupa ramalan harga *close* valuta asing pada hari yang sama. Sistem yang akan dibangun bertujuan agar dapat memberikan pertimbangan kepada perusahaan maupun individu untuk membeli atau menjual valas yang dimiliki.

3.1.2 Deskripsi Data

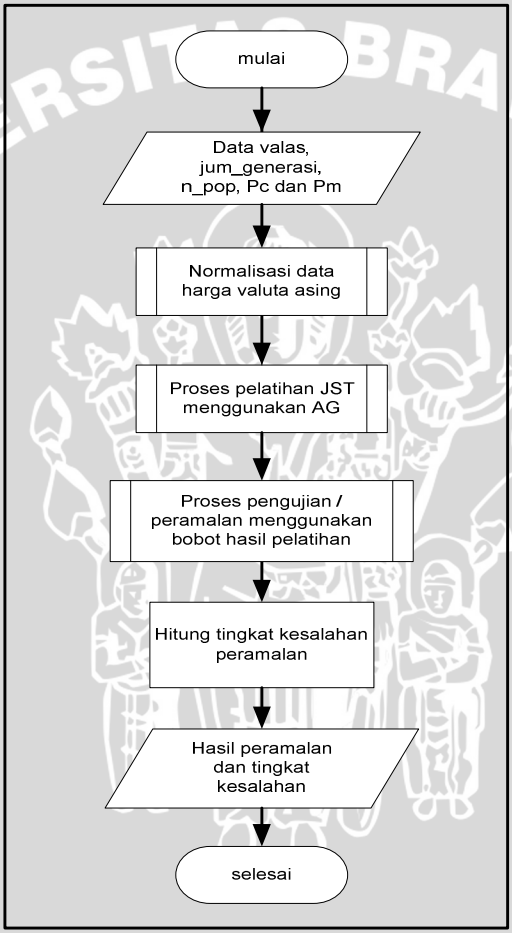
Data harga valuta asing atau pasangan mata uang yang diperdagangkan telah banyak disediakan di berbagai situs *forex*. Data yang digunakan dalam penelitian ini diperoleh dari situs *www.instaforex.com.* Data yang diambil adalah *pair* valas GBP/USD mulai Januari 2010 sampai Februari 2011, berupa harga *open, high , low* dan *close.*

Pada sistem ini akan digunakan data uji dan data latih, maka data dibagi menjadi dua. Untuk data latih menggunakan data harga *pair* valas GBP/USD selama satu tahun, mulai Januari 2010 sampai Desember 2010. Sedangkan untuk data uji menggunakan data selama dua bulan mulai Januari 2011 hingga Februari 2011.

3.2 Perancangan Sistem

Secara keseluruhan sistem yang dibuat menggunakan algoritma genetika untuk mencari bobot jaringan syaraf tiruan selama proses pelatihan. Masukan untuk sistem berupa data valas, dan parameter genetika antara lain jumlah generasi (jum_generasi), banyaknya individu dalam satu populasi (n_pop), probabilitas *crossover* (Pc) dan probabilitas mutasi (Pm). Langkah pertama, seluruh data harga valas dinormalisasi agar sesuai dengan masukan jaringan syaraf tiruan, selanjutnya lakukan proses pelatihan bobot jaringan terhadap data masukan harga valas. Pada proses pelatihan, digunakan algoritma genetika sebagai pengganti algoritma pelatihan

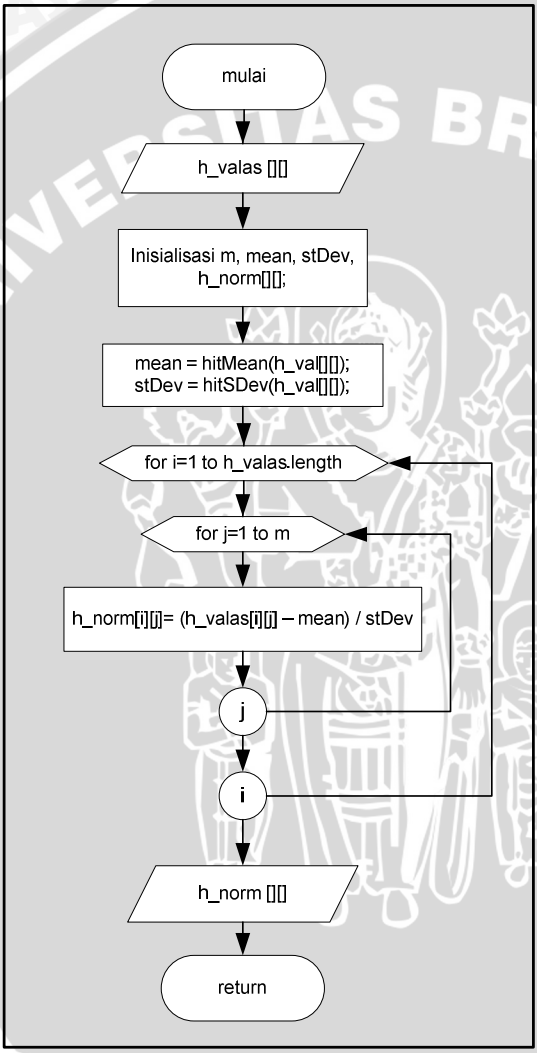
standar sebuah jaringan. Proses pelatihan akan menghasilkan bobot jaringan yang akan digunakan untuk tahap pengujian atau peramalan. Pada proses pengujian akan menghasilkan *output* yang disebut juga hasil ramalan harga valas yang kemudian akan dibandingkan dengan harga valas aktual. Kemudian hitung tingkat kesalahannya untuk mengevaluasi hasil peramalan. Proses ini merupakan gambaran secara umum sistem dan dapat dilihat pada Gambar 3.2.



Gambar 3.2 *Flowchart* proses peramalan harga valas secara umum

3.2.1 Normalisasi Data

Data masukan berupa harga valas dilakukan normalisasi terlebih dahulu sehingga berada pada range (-1,1). Proses normalisasi data harga valas dihitung berdasarkan Persamaan 2.12. *Flowchart* proses normalisasi dapat dilihat pada Gambar 3.3.



Gambar 3.3 *Flowchart* normalisasi data

Pada Gambar 3.3, penjelasan dari langkah – langkah proses normalisasi adalah sebagai berikut:

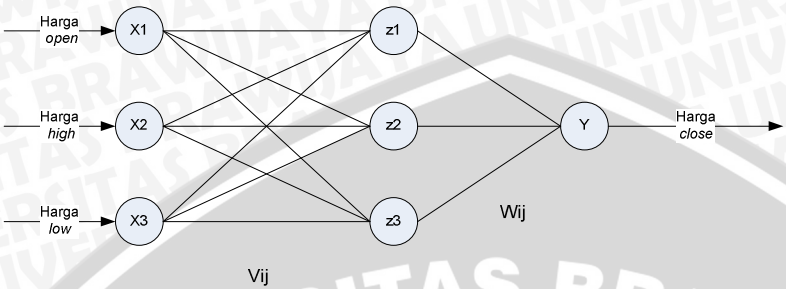
1. Persiapan data *input* harga *open*, *high*, *low* dan *close* valas.
2. Inisialisasi variabel n untuk jumlah macam harga yaitu 4 dan variabel m untuk banyaknya data latih.
3. Untuk i=1 sampai n dilakukan :
   - Untuk j = 1 sampai m, dilakukan proses normalisasi harga valas dengan Persamaan 2.12.
4. Jika i > n, maka proses normalisasi berhenti dan dihasilkan data *input* harga valas yang telah dinormalisasi.

3.2.2 Penentuan Arsitektur JST

Penelitian ini menggunakan 3 *input*, yaitu data harga *open*, *high* dan *low* dan 1 *output* yaitu harga *close* valas GBP terhadap USD. Maka dibutuhkan 3 neuron *input* dan 1 neuron *output*. Penggunaan data *input* berupa harga *open, high* dan *low* valas mengacu pada penelitian sebelumnya yang dilakukan oleh Dini Oktaviani (2010). Harga *open, high, low, close* dan volume digunakan untuk menentukan harga *close* hari berikutnya. Hasil peramalan yang diperoleh menghasilkan kesalahan rata-rata 0.002 terhadap data aktual yang diujikan.

Tidak ada aturan khusus dalam menentukan banyaknya jumlah layer maupun neuron pada hidden layer. Jingtao Yao sebelumnya telah melakukan penelitian untuk meramalkan harga valas menggunakan jaringan multilayer. Hasil yang didapat adalah bahwa dengan menggunakan model jaringan dengan sebuah hidden layer yang terdiri dari tiga neuron hidden, dapat menghasilkan nilai error yang cukup kecil yaitu 0,1. Maka pada penelitian ini, *hidden layer* yang digunakan sebanyak satu *layer* dengan 3 neuron.

Arsitektur *feed-forward* yang digunakan pada penelitian ini dapat dilihat pada Gambar 3.4.



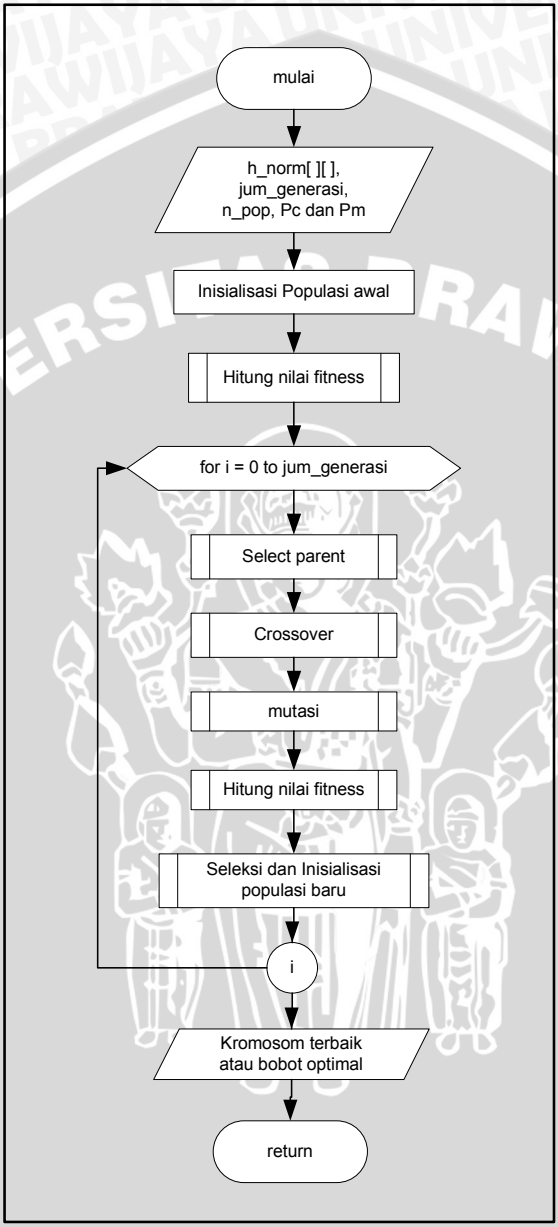
Gambar 3.4 Arsitektur jaringan *feed-forward*

3.2.3 Pelatihan JST menggunakan Algoritma Genetika

Tahapan algoritma genetika digunakan untuk mencari nilai-nilai bobot yang optimal untuk digunakan pada jaringan syaraf tiruan *backpropagation* untuk melakukan peramalan harga valuta asing. Langkah-langkah algoritma genetika untuk mendapatkan bobot optimal jaringan *backpropagation* adalah sebagai berikut :

1. Menentukan parameter awal yang akan digunakan pada proses genetik, yaitu ukuran populasi, jumlah generasi dan probabilitas *crossover* dan mutasi.
2. Membangkitkan populasi awal secara random sebanyak ukuran populasi yang telah ditentukan.
3. Menghitung nilai *fitness* masing-masing individu atau kromosom pada populasi.
4. Selama jumlah generasi belum terpenuhi maka lakukan pembentukkan populasi baru dengan langkah-langkah sebagai berikut :
   a) Menyeleksi *parent* pada populasi dengan metode *roulette wheel.*
   b) Melakukan *crossover* pada *parent* yang terpilih.
   c) Melakukan mutasi pada individu-individu yang terpilih.
   d) Menghitung nilai *fitness* individu hasil operator genetika.
   e) Menyeleksi individu baru sehingga terbentuk populasi atau generasi baru
5. Melakukan kembali langkah no.4 menggunakan populasi baru hingga jumlah generasi terpenuhi.

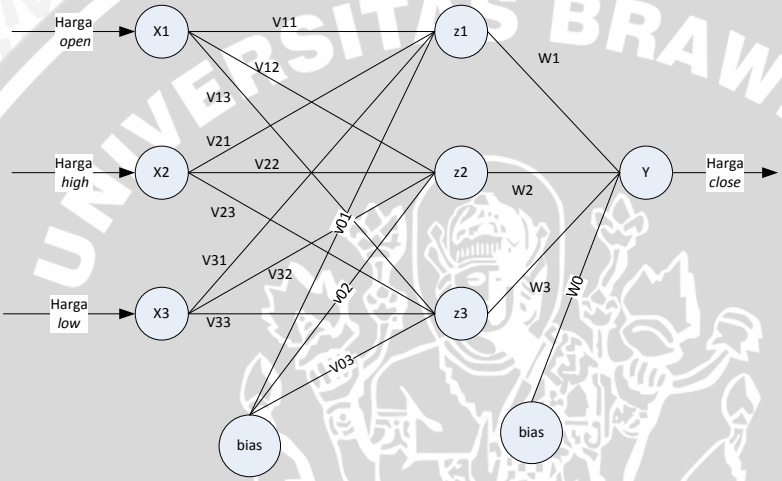
Flowchart proses algoritma genetika ditunjukkan pada Gambar 3.5.



Gambar 3.5 *Flowchart* Algoritma Genetika

3.2.3.1 Pengkodean Kromosom

Kromosom yang digunakan dikodekan dengan *real-number encoding* yang mewakili masing-masing bobot pada jaringan *feed-forward backpropagation*, yaitu semua bobot *input layer* menuju *hidden layer* dan bobot *hidden layer* menuju *output layer*. Arsitektur jaringan yang digunakan dalam penelitian ini dapat dilihat pada Gambar 3.6.



Gambar 3.6 Arsitektur jaringan yang digunakan

Dari Gambar 3.6, maka panjang kromosom yang dibutuhkan untuk membentuk sebuah kromosom adalah sebanyak 16 gen yang masing-masing bernilai riil dengan range 0 sampai 1. Representasi kromosom dijelaskan pada Gambar 3.7.

Kromosom

V01	V02	V03	V11	V12	V13	V21	V22	V23	V31	V32	V33	W1	W2	W3	W0

Kromosom 1	0,7512	0,2310	0,034	...	...	0,3412	0,1561	0,801

Gambar 3.7 Representasi Kromosom

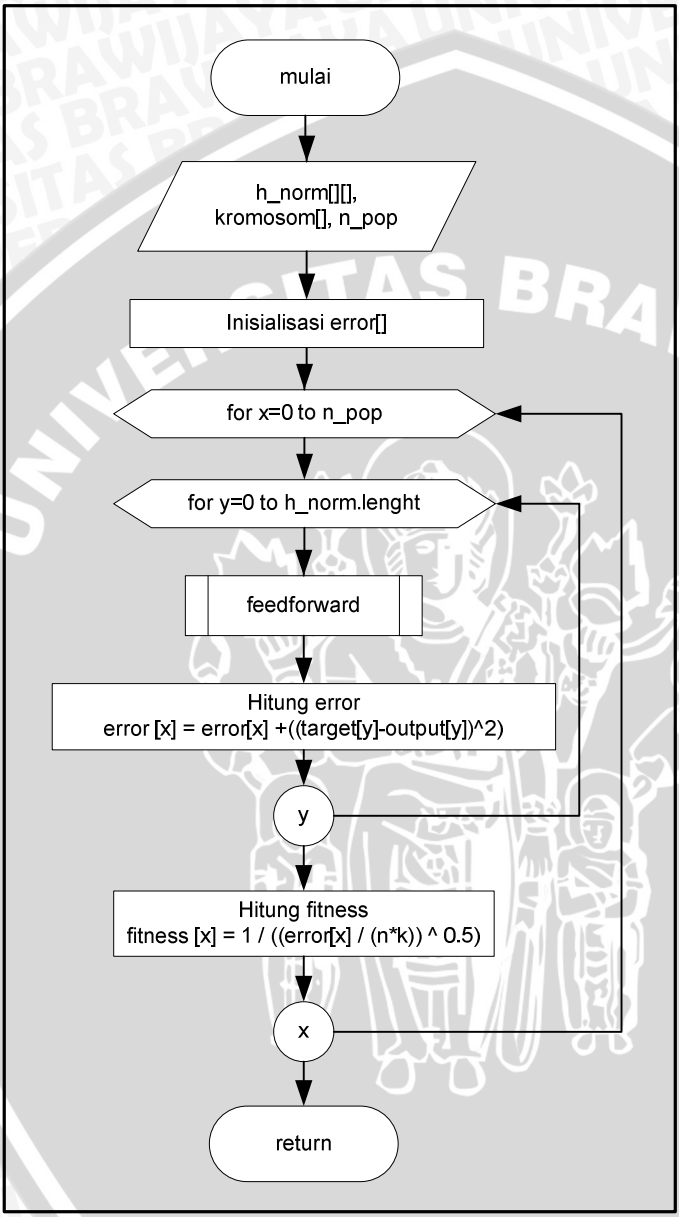
3.2.3.2 Fungsi *Fitness*

Nilai *fitness* dihitung menggunakan proses *feedfoward* algoritma *backpropagation* untuk menghitung *output* jaringan setelah itu menghitung selisihnya terhadap data aktual. Karena proses *feedforward* dilakukan berulang-ulang sesuai dengan jumlah data maka penghitungan nilai selisih antara *output* dan data aktual juga dilakukan berulang-ulang. Fungsi *fitness* yang digunakan dalam penelitian ini adalah 1/*delta*, yang mana *delta* diperoleh dari Persamaan 2.11.

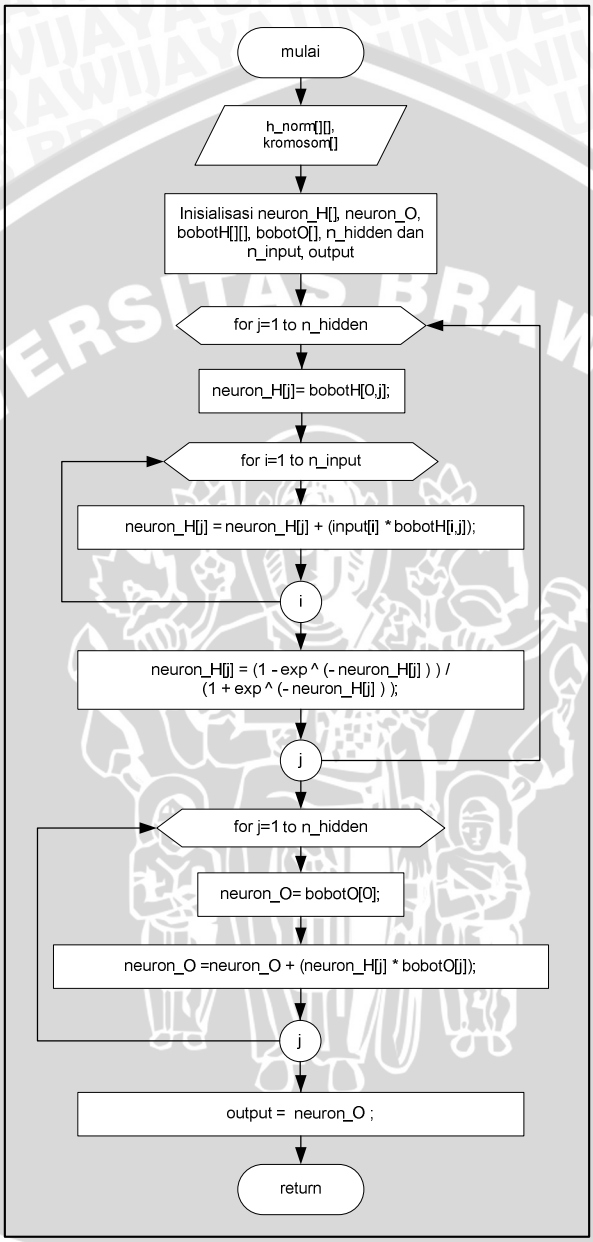
Berikut ini langkah-langkah untuk menghitung nilai *fitness* untuk menentukan kesalahan peramalan :

1. Memasukan data harga valas normalisasi yang menjadi *input* jaringan *feedforward*, kromosom yang telah diinisialisasi pada populasi awal, dan jumlah kromosom dalam populasi (n_pop).
2. Menginisialisasi variabel *error* untuk menghitung kesalahan dan variabel n_data untuk menentukan banyaknya data.
3. Untuk x=1 sampai n_pop, dilakukan :
   - Untuk y=1 sampai n_data, hitung kesalahan peramalan dengan langkah-langkah berikut :
     a) Lakukan proses *feedforward* untuk masing-masing data *input* terhadap sebuah kromosom
     b) Hitung kesalahan yang didapat dengan menghitung kuadrat selisih data *output* dengan data aktual.
     c) Jumlahkan seluruh kesalahan tiap data yang didapat
     d) Hitung *fitness* kromosom
4. Jika x > n_pop maka proses hitung *fitness* berhenti.

Flowchart proses hitung *fitness* dapat dilihat pada Gambar 3.8.



Gambar 3.8 *Flowchart* proses hitung *fitness*



Gambar 3.9 *Flowchart* proses *feedforward* pada perhitungan nilai *fitness*

Berdasarkan Gambar 3.9, penjelasan proses *feedforward* adalah sebagai berikut :

1. Persiapan data *input* berupa harga valas normalisasi dan kromosom awal hasil inisialisasi populasi awal
2. Inisialisasi neuron_H[], *output* untuk menyimpan hasil keluaran jaringan, bobotH[][], dan bobotO[]. bobotH merupakan array 2 dimensi dengan 3 kolom dan 4 baris. Kolom menyatakan banyaknya hidden neuron sedangkan baris adalah jumlah neuron *input* jaringan. BobotO[] merupakan array 1 dimensi dengan panjang array 4 untuk menyimpan bobot neuron hidden menuju neuron *output*. Nilai-nilai untuk bobotH didapat dari kromosom yang merupakan nilai pada gen 1-12, sedangkan bobotO didapat dari nilai kromosom pada posisi gen 13-16.
3. Inisisalisasi n_hidden untuk menentukan jumlah neuron hidden yaitu 3 dan n_*input* untuk menentukan jumlah *input* yaitu 3.
4. Selama j =1 sampai n_hidden, lakukan insialisasi nilai awal neuronH. Nilai awal neuron merupakan nilai bias untuk neuron hidden yang akan dihitung.
   - Selama i=1 sampai n_*input*, dilakukan:
     - Menghitung nilai keluaran masing-masing hidden neuron sesuai dengan Persamaan 2.6.
     - Menghitung nilai aktivasi dari masing-masing neuron hidden menggunakan fungsi aktivasi sigmoid bipolar, sesuai Persamaan 2.7.
5. Jika nilai j > n_hidden maka proses penghitungan keluaran neuron hidden berhenti.
6. Selanjutnya menghitung keluaran jaringan, lakukan insialisasi nilai awal neuron *output*. Nilai awal neuron *output* merupakan nilai bias neuron hidden menuju neuron *output*.
7. Selama j=1 sampai n_hidden, dilakukan:
   - Menghitung nilai keluaran masing-masing neuron *output* sesuai dengan Persamaan 2.8.
   - Menghitung nilai aktivasi neuron *output* menggunakan fungsi aktivasi linier sehingga menghasilkan *output* jaringan, sesuai Persamaan 2.9.

3.2.3.3 Proses Seleksi

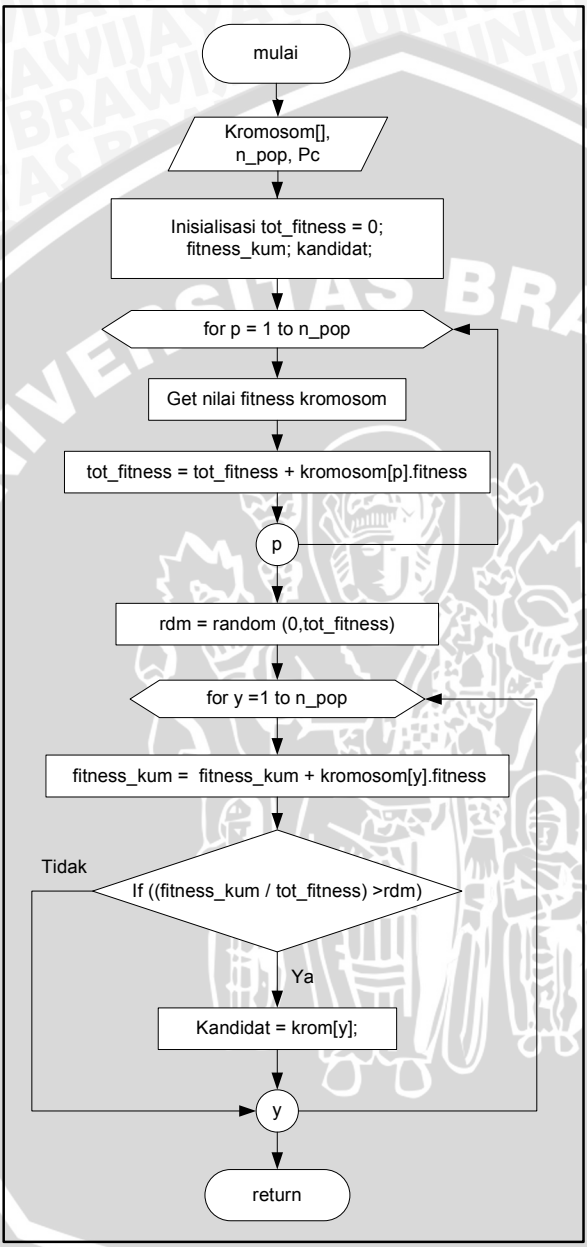
Proses seleksi dilakukan untuk memilih *parent* yang akan melakukan *crossover*. Proses seleksi dilakukan sebanyak jumlah pasangan *parent* yang telah ditentukan sebelumnya.

Proses *select parent* dilakukan berdasarkan nilai *fitness* masing-masing kromosom, artinya semakin baik nilai *fitness*-nya maka kemungkinan besar kromosom itu terpilih untuk melakukan *crossover*. Hal ini bertujuan untuk mendapatkan individu yang semakin baik dari induknya. Metode yang digunakan untuk menentukan *parent* adalah dengan *roulette wheel*.

Langkah-langkah untuk menentukan *parent* dengan metode *roulette wheel* adalah sebagai berikut :

1. Mengambil nilai *fitness* masing-masing kromosom pada populasi.
2. Menjumlahkan seluruh *fitness* untuk mendapatkan total *fitness*.
3. Menghitung *fitness* relatif tiap kromosom dengan membagi nilai *fitness* dengan total *fitness*.
4. Menghitung *fitness* kumulatif tiap kromosom, *fitness* kumulatif digunakan untuk menentukan batasan daerah tiap kromosom.
5. Melakukan proses *roulette* dengan membangkitkan bilangan acak antara 0 sampai nilai total *fitness*.
6. Jika nilai random berada pada salah satu daerah kromosom, maka kromosom itu terpilih menjadi *parent* untuk melakukan *crossover*.
7. Melakukan proses *roulette wheel* sebanyak jumlah *parent* yang telah ditentukan sebelumnya.

Flowchart untuk proses *roulette wheel* dapat dilihat pada Gambar 3.10.



Gambar 3.10 *Flowchart* metode *Roulette Wheel*

3.2.3.4 *Crossover*

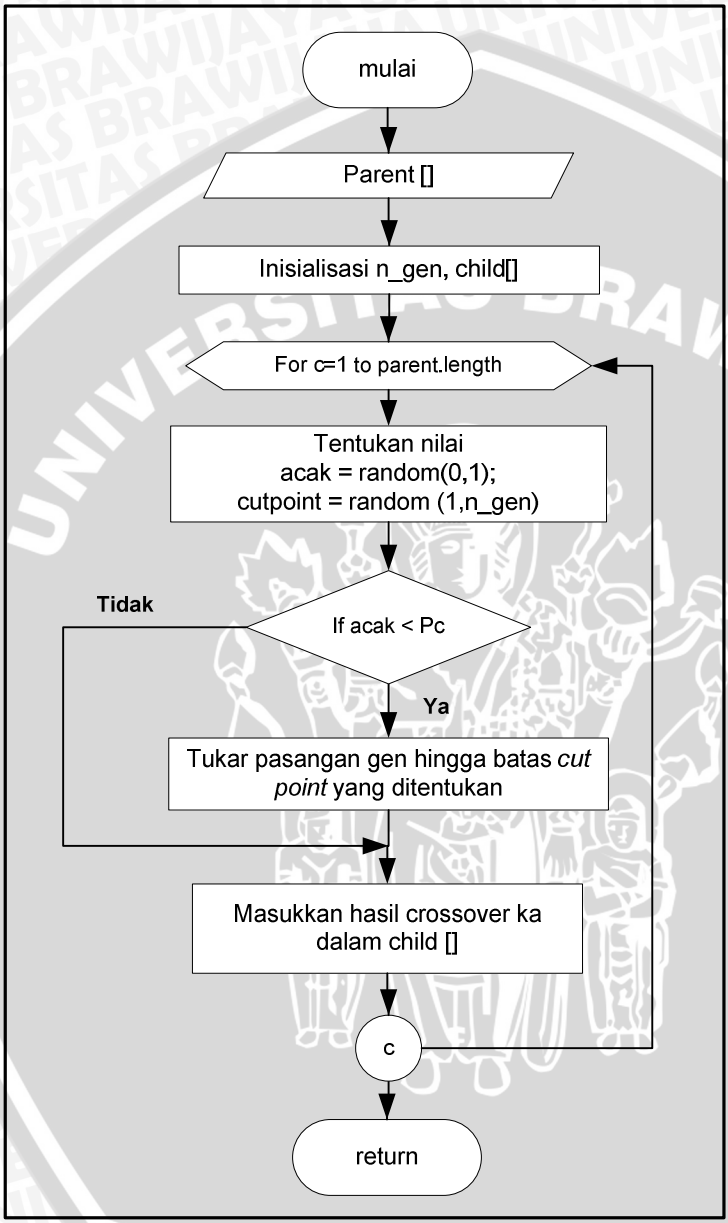
Parent yang telah terpilih melalui proses seleksi dengan *roulette wheel* selanjutnya akan melakukan *crossover*. *Crossover* dilakukan dengan mengkombinasikan gen-gen induk untuk mendapatkan keturunan baru yang diharapkan lebih baik dari induknya. Proses *crossover* dilakukan dengan teknik *one-cut-point crossover*. Titik potong ditentukan secara random dengan nilai antara 1 sampai panjang kromosom. Kemudian tukar gen-gen *parent* 1 dan *parent* 2 sesuai batas titik potong. Contoh proses *crossover* ditunjukkan pada gambar 3.11. *Flowchart* untuk proses *crossover* dapat dilihat pada Gambar 3.12.

Parent 1	0,341	0,856	0,187	0,451	0,910	0,223	0,112	0,876	0,519	0,724	0,301	0,662	0,196	0,349	0,520	0,731
Parent 2	0,981	0,219	0,723	0,409	0,286	0,710	0,813	0,392	0,704	0,211	0,006	0,119	0,076	0,400	0,700	0,105
Child 1	0,341	0,856	0,187	0,451	0,910	0,223	0,112	0,876	0,519	0,724	0,006	0,119	0,076	0,400	0,700	0,105
Child 2	0,981	0,219	0,723	0,409	0,286	0,710	0,813	0,392	0,704	0,211	0,301	0,662	0,196	0,349	0,520	0,731

Gambar 3.11 Contoh proses *crossover*

Dari Gambar 3.12, penjelasan dari proses *crossover* adalah sebagai berikut :

1. Persiapan data *input* yaitu kromosom *parent* yang terpilih dari proses *select parent* (*parent*[]).
2. Inisialisasi variabel n_gen untuk menentukan jumlah gen dalam satu kromosom yaitu 16, dan child[] untuk menyimpan kromosom hasil *crossover*.
3. Selama c =1 sampai *parent.length* dilakukan :
   - Membangkitkan nilai riil secara random antara 0-1, apabila nilai random lebih kecil dari Pc, maka
     - Menentukan *cut-point* dengan membangkitkan bilangan random antara 1-16
     - Menukar pasangan gen hingga batas cut-point
   - Bila nilai random lebih besar dari Pc, maka tidak terjadi proses crossover, sehingga kromosom anak sama persis dengan induknya.
   - Memasukkan hasil *crossover* ke dalam array child
4. Jika c > *parent.length* maka proses *crossover* berhenti.

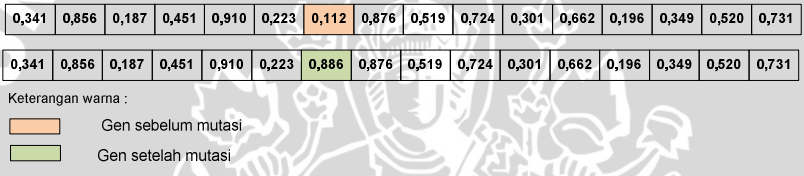


Gambar 3.12 *Flowchart* proses *one-cut-point crossover*

3.2.3.5 Mutasi

Pada algoritma genetika, mutasi dilakukan untuk mendapatkan nilai solusi yang baru dari hasil mutasi gen yang terpilih. Selain itu mutasi dilakukan untuk menghindari solusi terjebak pada lokal optimal dan menemukan global optimal. Proses mutasi tidak dilakukan pada setiap kromosom, namun hanya dilakukan pada kromom terpilih.

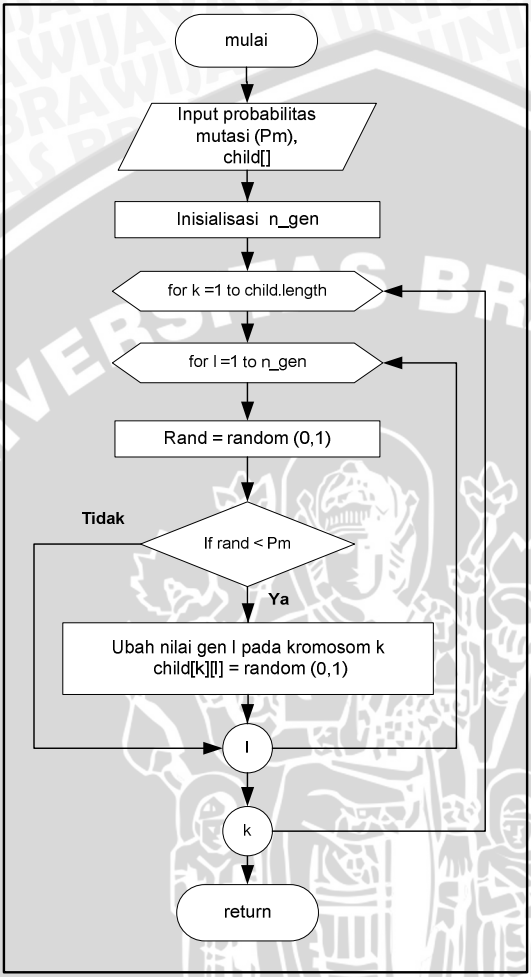
Proses mutasi yang digunakan pada penelitian ini adalah dengan mengganti nilai gen terpilih dengan nilai random yang baru. Langkah pertama adalah memilih gen yang akan dimutasi. Kemudian bangkitkan bilangan random antara 0 dan 1 untuk menggantikan nilai pada gen terpilih untuk dimutasi. Contoh mutasi ditunjukkan pada Gambar 3.13 dan *flowchart* mutasi dapat dilihat pada Gambar 3.14.



Gambar 3.13 Contoh proses mutasi

Dari Gambar 3.14, penjelasan dari proses mutasi adalah sebagai berikut :

1. Persiapan data *input* yaitu kromosom anak (child[]) dan probabilitas mutasi (Pm).
2. Inisialisasi variabel n_gen untuk menentukan jumlah gen dalam satu kromosom yaitu 16.
3. Selama k =1 sampai child.length lakukan :
   - Selama l = 1 sampai n_gen, lakukan :
     - Membangkitkan nilai riil secara random antara 0-1, apabila nilai random lebih kecil dari Pm, maka
     - Mengubah nilai gen dari posisi gen dengan nilai riil baru yang dibangkitkan secara random antara 0-1
4. Jika k > n_child maka proses mutasi berhenti.

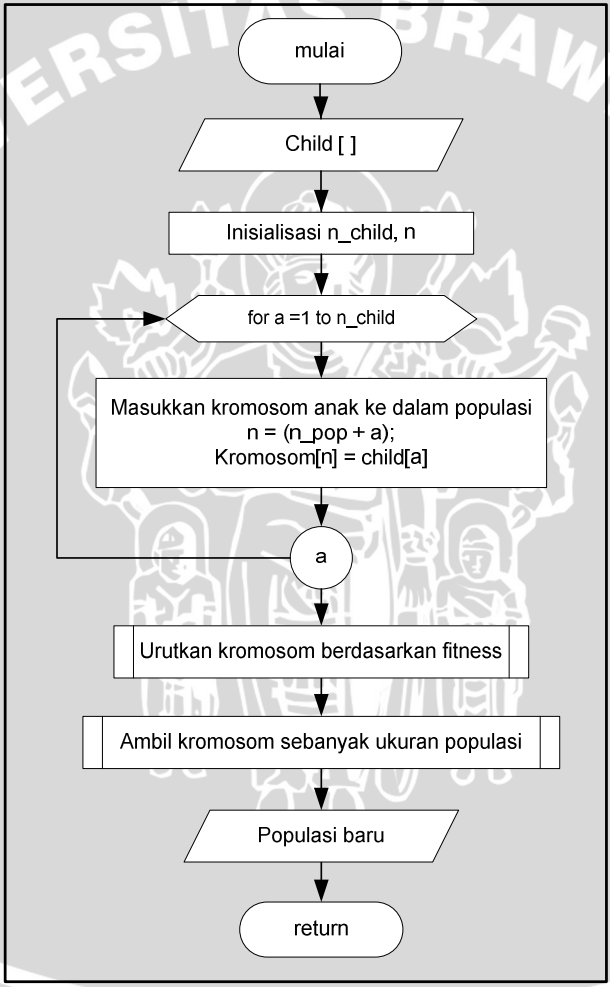


Gambar 3.14 *Flowchart* proses mutasi

3.2.3.6 Seleksi Populasi Baru

Setelah proses *crossover* dan mutasi selesai dilakukan, kromosom-kromosom baru yang terbentuk dihitung *fitness*-nya. Ukuran populasi pada proses genetika selalu sama, oleh karena itu hanya individu atau kromosom baru yang terbaik yang dapat masuk ke populasi membentuk generasi baru yang lebih baik. Untuk itu perlu dilakukan seleksi. Metode yang digunakan untuk proses seleksi adalah *Rank based Selection*.

Semua kromosom atau individu baru yang terbentuk dimasukkan terlebih dahulu ke dalam populasi, kemudian seluruh kromosom diurutkan berdasarkan nilai *fitness* yang dimiliki. Setelah itu diambil kromosom yang terbaik sebanyak ukuran populasi yang telah ditentukan sebelumnya sehingga membentuk populasi baru. Individu yang nilai *fitness*-nya tidak mencukupi untuk masuk ke dalam populasi akan dihilangkan. *Flowchart* proses seleksi populasi baru dapat dilihat pada Gambar 3.15.

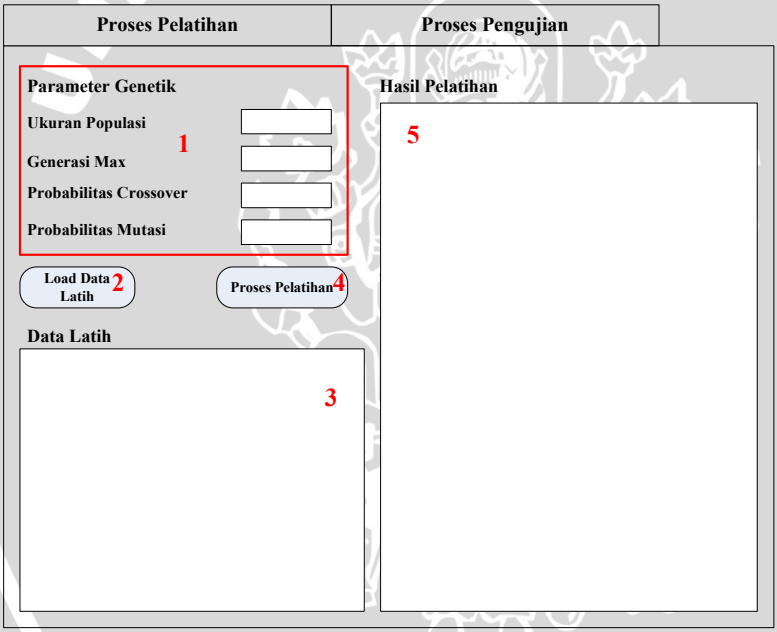


Gambar 3.15 *Flowchart* proses pembentukan populasi baru

3.3 Rancangan Antarmuka

Antarmuka (*interface*) untuk sistem ini terdiri dari dua bagian utama, yaitu:

1. Bagian untuk proses pelatihan
   Bagian ini bertujuan untuk menentukan bobot jaringan optimal yang akan digunakan pada proses peramalan. Antarmuka dari bagian ini ditunjukkan pada gambar 3.16.
2. Bagian untuk proses pengujian
   Bagian ini bertujuan untuk melakukan pengujian terhadap bobot jaringan yang terbentuk serta peramalan yang dihasilkan terhadap data aktual. Antarmuka dari bagian ini ditunjukkan pada gambar 3.17.

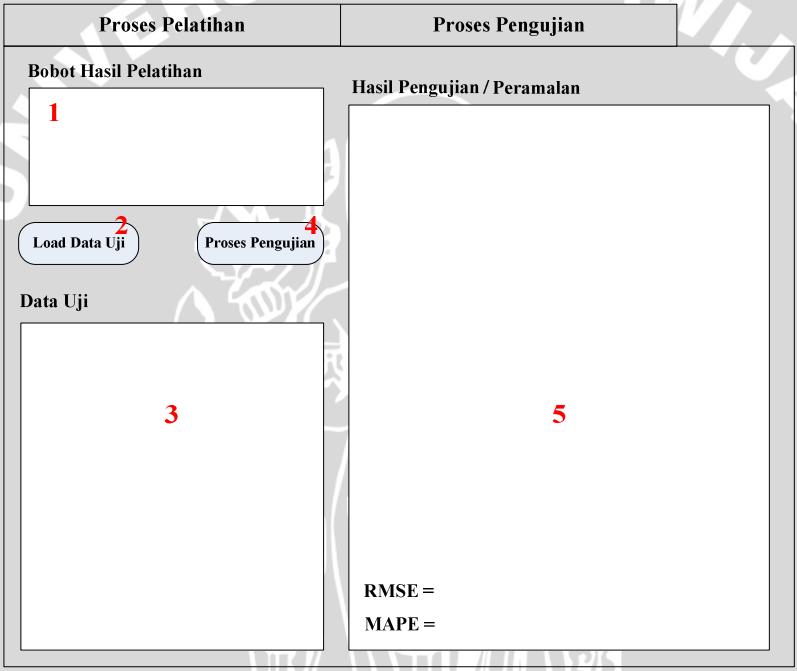


Gambar 3.16 Rancangan Antarmuka untuk Proses Pelatihan

Pada gambar 3.16, antar muka bagian proses pelatihan bobot terdiri dari :

1. Area berisi *text field* untuk memasukkan nilai parameter yang akan digunakan pada algoritma genetika.

2. Tombol *load data* latih, untuk mengambil data yang akan digunakan pada proses pelatihan.
3. *Text area* untuk menampilkan data yang akan digunakan pada proses pelatihan.
4. Tombol proses pelatihan, untuk melatih bobot jaringan terhadap data latih yang digunakan untuk mendapatkan bobot optimal.
5. *Text area* untuk menampilkan proses pelatihan sampai menghasilkan bobot optimal.



Gambar 3.17 Rancangan Antarmuka untuk Proses Pengujian

Pada gambar 3.17, antarmuka bagian proses pelatihan bobot terdiri dari :

1. *Text area* untuk menampilkan bobot hasil pelatihan yang akan digunakan pada proses pengujian atau peramalan.
2. Tombol *load data* uji, untuk mengambil data yang akan digunakan pada proses pengujian.

3. *Text area* untuk menampilkan data yang akan digunakan pada proses pelatihan.
4. Tombol proses pengujian, untuk menghitung nilai ramalan yang dihasilkan menggunakan bobot sekaligus menghitung kesalahan peramalan terhadap data aktual.
5. *Text area* untuk menampilkan hasil peramalan dan kesalahan yang diperoleh.

3.4 Perhitungan Manual

3.4.1 Normalisasi Data

Normalisasi dihitung menggunakan Persamaan 2.12. Sebelumnya hitung nilai rata-rata dan nilai deviasi standar dari data. Data sebelum normalisasi dapat dilihat pada Tabel 3.1.

Tabel 3.1 Data awal valas sebelum dilakukan normalisasi

No	Date	open	high	low	close
1	2010.01.04	1.6111	1.6241	1.6057	1.6086
2	2010.01.05	1.6084	1.6153	1.5965	1.5993
3	2010.01.06	1.5994	1.6066	1.594	1.6021
4	2010.01.07	1.602	1.6061	1.5899	1.5938
5	2010.01.08	1.5937	1.611	1.5912	1.6023
...	...	...	...	...	...
...	...	...	...	...	...
256	2010.12.27	1.5441	1.5456	1.5369	1.542
257	2010.12.28	1.5419	1.5511	1.5344	1.5367
258	2010.12.29	1.5365	1.5517	1.5349	1.5498
259	2010.12.30	1.5499	1.5535	1.5366	1.5424
260	2010.12.31	1.5425	1.5664	1.5419	1.5586

Langkah pertama pada normaliasi data adalah mencari nilai rata-rata dan nilai deviasi standar dari data. Nilai rata-rata dapat dihitung dengan menjumlahkan seluruh data kemudian dibagi dengan jumlah data. Dengan menggunakan data pada Tabel 3.1, nilai rata-ratanya

adalah 1.5453. Kemudian menghitung nilai deviasi standar dengan Persamaan 2.13.

$$\text{deviasi standar} = \sqrt{\frac{\sum(x-\bar{x})^2}{n-1}} = \sqrt{\frac{0.0043+0.0062+0.0037+\cdots+1.13E-05}{779}}$$

$$= \sqrt{\frac{1.775621}{779}} = 0.047743$$

Kemudian menghitung normalisasi data dengan Persamaan 2.12 :

$$\text{Nilai baru} = \frac{(\text{Nilai lama} - \text{Ratarata})}{\text{Deviasi Standar}}$$

Normalisasi data pertama adalah sebagai berikut:

$$\text{untuk data } open, \text{normalisasi} = \frac{(1.6111 - 1.5453)}{0.047743} = 1.3795$$

$$\text{untuk data } high, \text{normalisasi} = \frac{(1.6241 - 1.5453)}{0.047743} = 1.652$$

$$\text{untuk data } low, \text{normalisasi} = \frac{(1.6057 - 1.5453)}{0.047743} = 1.2662$$

$$\text{untuk data } close, \text{normalisasi} = \frac{(1.6068 - 1.5453)}{0.047743} = 1.3468$$

Seluruh data yang akan digunakan dinormalisasi terlebih dahulu seperti pada contoh normalisasi data pertama. Hasil normalisasi data valas dapat dilihat pada Tabel 3.2.

Tabel 3.2 Hasil Normalisasi data valas

No	Date	Open	high	low	close
1	2010.01.04	1.3795	1.652	1.2663	1.3468
2	2010.01.05	1.3229	1.4675	1.0734	1.1489
3	2010.01.06	1.1342	1.2851	1.021	1.2085
4	2010.01.07	1.1887	1.2746	0.935	1.0319
5	2010.01.08	1.0147	1.3774	0.9623	1.2128
...	...	...	...	...	...
...	...	...	...	...	...
256	2010.12.27	-0.0252	0.0063	-0.176	-0.0702
257	2010.12.28	-0.071	0.1216	-0.229	-0.183
258	2010.12.29	-0.184	0.1342	-0.218	0.0957
259	2010.12.30	0.0964	0.1719	-0.182	-0.0617
260	2010.12.31	-0.0587	0.4423	-0.071	0.2823

3.4.2 Pembangkitan Populasi Awal

Inisialisasi populasi awal dilakukan dengan membangkitkan bilangan random antara 0 dan 1 sebanyak jumlah gen dalam satu kromosom dan dilakukan sebanyak jumlah populasi yang akan dibentuk. Misalnya telah dibangkitkan sebuah populasi sebanyak 4 kromosom sebagai berikut :

Kromosom 1	0.978	0.331	0.968	0.069	0.47	0.274	0.696	0.744
	0.073	0.803	0.939	0.495	0.041	0.53	0.325	0.879
Kromosom 2	0.003	0.295	0.333	0.6	0.573	0.754	0.264	0.892
	0.449	0.088	0.218	0.135	0.048	0.579	0.844	0.705
Kromosom 3	0.372	0.728	0.792	0.85	0.02	0.703	0.926	0.259
	0.669	0.341	0.09	0.494	0.274	0.471	0.331	0.124

Kromosom 4	0.084	0.63	0.344	0.818	0.81	0.12	0.681	0.844
	0.733	0.564	0.556	0.572	0.719	0.207	0.788	0.137

3.4.3 Perhitungan *Fitness*

Perhitungan *fitness* dilakukan dengan menggunakan tahapan *feedforward* jaringan *backpropagation* yaitu Persamaan 2.5 sampai Persamaan 2.8. Setiap kromosom berupa susunan bobot jaringan dikalikan dengan data valas ternormalisasi, kemudian dihitung total kesalahan yang terjadi. Dengan menggunakan kromosom pertama, perhitungan *fitness*-nya sebagai berikut :

Kromosom 1	0.978	0.331	0.968	0.069	0.47	0.274	0.696	0.744
	v01	v02	v03	v11	v12	v13	v21	v22
	0.073	0.803	0.939	0.495	0.041	0.53	0.325	0.879
	v23	v31	v32	v33	w1	w2	w3	w0

Data 1 :

Menghitung keluaran unit tersembunyi dengan Persamaan 2.6:

$$z_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

Keterangan :
nilai x_i = nilai normalisasi harga valas yang berasal dari Tabel 3.3

$$z_in_1 = v_{01} + x_1 v_{11} + x_2 v_{21} + x_3 v_{31}$$

z_in_1 = 0.978 + (1.3795)(0.069)+ (1.652)(0.696)+(1.2663)(0.803)
= 3.2398

$$z_in_2 = v_{02} + x_1 v_{12} + x_2 v_{22} + x_3 v_{32}$$

z_in_2 = 0.331 + (1.3795)(0.47)+(1.652)(0.744)+ (1.2663)(0.939)
= 3.3974

$z_in_3 = v_{03} + x_1 v_{13} + x_2 v_{23} + x_3 v_{33}$

z_in_3 = 0.968 + (1.3795)(0.274)+ (1.652)(0.073)+ (1.2663)(0.495)
= 2.0934

Menghitung aktivasi unit tersembunyi dengan Persamaan 2.7 :

$$z_j = f\left(z_{in_j}\right) = \frac{1 - e^{-z_in_j}}{1 + e^{-z_in_j}}$$

$$z_1 = f\,(z_in_1) = \frac{1 - e^{-3.2398}}{1 + e^{-3.2398}} = 0.9246$$

$$z_2 = f\,(z_in_2) = \frac{1 - e^{-3.3974}}{1 + e^{-3.3974}} = 0.9352$$

$$z_3 = f\,(z_in_3) = \frac{1 - e^{-2.0934}}{1 + e^{-2.0934}} = 0.7805$$

Hitung keluaran unit y_k menggunakan Persamaan 2.8 :

$$y_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

Untuk menghitung nilai *output* hanya digunakan bobot yang terhubung ke neuron *output*. Berdasarkan arsitektur jaringan yang dibuat sebelumnya, tedapat satu neuron *output* dan bobot yang terhubung ke neron *output* sebanyak empat buah, yaitu yang berasal dari ketiga hidden neuron dan neuron bias. Maka persamaan 2.8 dapat dijabarkan sebagai berikut :

$y_in = w_0 + z_1 w_1 + z_2 w_2 + z_3 w_3$

y_in = 0.879 + (0.9246)(0.041)+(0.9352)(0.53)+(0.7805)(0.325)
= 1.6662

Aktivasi keluaran unit y_k dengan Persamaan 2.9 :

$y_k = f(y_{in_k}) = y_{in_k}$

$y = f(y_{in}) = 1.6662$

Hitung kembali nilai y untuk data kedua dan seterusnya dengan langkah yang sama seperti contoh perhitungan data 1 dan tetap menggunakan kromosom atau nilai bobot yang sama.

Hasil perhitungan *feedforward* untuk seluruh data dengan kromosom 1 ditunjukkan pada Tabel 3.3.

Tabel 3.3 Hasil perhitungan *feedforward* kromosom ke-1

data	Zin 1	Zin2	Zin3	Z1	Z2	Z3	y_in	y
1	3.2398	3.3974	2.0934	0.9246	0.9352	0.7805	1.6662	1.6662
2	2.9526	3.0525	1.9689	0.9007	0.9098	0.7549	1.6435	1.6435
3	2.7705	2.7789	1.878	0.8821	0.8830	0.7347	1.6219	1.6219
4	2.698	2.716	1.8496	0.8738	0.8759	0.7281	1.6157	1.6157
5	2.7794	2.7362	1.8229	0.8831	0.8782	0.7218	1.6152	1.6152
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…
256	0.8392	0.1585	0.8744	0.3966	0.0791	0.4113	1.0708	1.0708
257	0.8742	0.1734	0.8442	0.4112	0.0865	0.3987	1.0713	1.0713
258	0.8836	0.1394	0.8193	0.4151	0.0696	0.3881	1.059	1.0590
259	0.9579	0.333	0.9167	0.4453	0.1649	0.4287	1.1240	1.1240
260	1.2246	0.5656	0.9489	0.5457	0.2755	0.4418	1.1909	1.1909

Setelah semua nilai *y* dari data diketahui, selanjutnya hitung selisih nilai *y* (*output* jaringan) terhadap target, yang dalam penelitian ini adalah harga penutupan (*close price*). Nilai selisih digunakan untuk mendapatkan nilai *delta*, yang dihitung menggunakan Persamaan 2.11.

$$delta = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(d(i) - y(i))^2}$$

Keterangan : nilai d_i adalah nilai target yaitu harga *close* yang telah dinormalisasi yang ada pada Tabel 3.3

$$\sqrt{\frac{(1.3468-1.6662)^2+ (1.1489-1.6435)^2+\cdots +(-0.0617-1.124)^2+(0.283-1.1909)^2}{260}}$$

= 1.2069

$fitness = \frac{1}{delta} = \frac{1}{1.2069} = 0.82857$

Lakukan cara yang sama untuk menghitung kromosom yang ada dalam populasi. Hasil *fitness* seluruh kromosom dalam populasi, yang pada contoh ini sebanyak 4 kromosom dalam satu populasi dapat dilihat pada Tabel 3.4.

Tabel 3.4 Hasil *fitness* tiap kromosom

Kromosom ke-	***fitness***
1	**0.82857**
2	**1.02689**
3	**1.40317**
4	**2.41029**

3.4.4 Seleksi *Parent* dengan *Roulette Wheel*

Langkah awal untuk menjalankan metode *roulette wheel* adalah dengan menghitung total *fitness* untuk memperoleh *fitness* kumulatif tiap kromosom.

Total *fitness* = 0.82857+ 1.02689+ 1.40317+ 2.41029= 5.66892

Hitung masing-masing *fitness relative* terhadap total *fitness* yaitu dengan membagi masing-masing *fitness* individu dengan total *fitness.*

Fitness relatif kromosom 1 = 0.82857/ 5.66892 = 0.1462

Fitness relatif kromosom 1 = 1.02689/ 5.66892 = 0.1811
Fitness relatif kromosom 1 = 1.40317/ 5.66892 = 0.2475
Fitness relatif kromosom 1 = 2.41029/ 5.66892 = 0.4252

Hitung *fitness* kumulatif tiap kromosom untuk memperoleh skala atau daerah *roulette wheel*.

Kromosom 1 = 0 + 0.1462 = 0.1462
Kromosom 2 = 0.1462+ 0.1811 = 0.3273
Kromosom 3 = 0.3273+ 0.2475 = 0.5748
Kromosom 4 = 0.5748+ 0.4252 = 1

Dari *fitness* kumulatif tersebut maka masing masing kromosom memiliki daerah sebagai berikut :

Kromosom 1	**0 - 0.1462**
Kromosom 2	**0.1463 - 0.3273**
Kromosom 3	**0.3274 - 0.5748**
Kromosom 4	**0.5749 - 1**

Bangkitkan nilai random antara 0 sampai 1 untuk menentukan kromosom yang akan terpilih melakukan *crossover*. Misalnya nilai random pertama 0.7410 dan nilai random kedua 0.0245. Berdasarkan batasan atau daerah masing-masing kromosom, maka untuk random pertama yang terpilih adalah kromosom ke-4 sedangkan random kedua yang terpilih adalah kromosom ke-1. Maka selanjutnya proses *crossover* antara kromosom 4 dan 1 akan dilakukan.

3.4.5 *Crossover*

Proses *crossover* dilakukan dengan cara mengkombinasikan 2 kromosom *parent* yang terpilih dengan satu titik potong yang ditentukan secara acak.

Langkah 1 :
***Parent* 1** :

Kromosom 4	**0.084**	**0.63**	**0.344**	**0.818**	**0.81**	**0.12**	**0.681**	**0.844**
	0.733	**0.564**	**0.556**	**0.572**	**0.719**	**0.207**	**0.788**	**0.137**

***Parent* 2** :

Kromosom 1	0.978	0.331	0.968	0.069	0.47	0.274	0.696	0.744
	0.073	0.803	0.939	0.495	0.041	0.53	0.325	0.879

Langkah 2 :

Menentukan bilangan random antara 1 sampai (banyaknya gen – 1) yaitu 15. Misalnya nilai random yang muncul adalah 5. Maka titik potong berada pada posisi ke 5.

***Child* 1**	0.084	0.63	0.344	0.818	0.81	0.274	0.696	0.744
	0.073	0.803	0.939	0.495	0.041	0.53	0.325	0.879
***Child* 2**	0.978	0.331	0.968	0.069	0.47	0.12	0.681	0.844
	0.733	0.564	0.556	0.572	0.719	0.207	0.788	0.137

Keterangan warna :

: Kromosom dari *parent* 1

: Kromosom dari *parent* 2

3.4.6 Mutasi

Proses mutasi ditentukan dengan sebuah parameter yaitu probabilitas mutasi. Misalnya nilai probabilitas mutasi adalah 15% maka diharapkan 15% dari jumlah gen pada seluruh individu dalam satu populasi akan mengalami mutasi. Proses mutasi diawali dengan membangkitkan nilai acak untuk menetukan apakah gen pada individu tersebut dimutasi atau tidak. Misalnya kromosom *child* pertama, pada tiap gennya akan dibangkitkan bilangan riil antara 0-1 secara acak. Apabila nilai acak yang muncul kurang dari probabilitas mutasi, maka gen tersebut akan dimutasi. Langkah mutasi yang digunakan dalam penelitian ini adalah sebagai berikut :

1. Lakukan pembangkitan bilangan riil secara acak antara 0-1 pada setiap gen dalam populasi, mulai dari kromosom awal hingga kromosom akhir.
2. Misalnya, pada kromosom pertama gen ke-11 nilai acak yang muncul adalah 0.08, maka gen tersebut dimutasi.

***Child* 1**	0.084	0.63	0.344	0.818	0.81	0.274	0.696	0.744
	0.073	0.803	0.939	0.495	0.041	0.53	0.325	0.879

3. Random nilai antara 0 sampai 1 untuk menggantikan nilai gen yang akan dimutasi. Misalkan nilai yang muncul adalah 0.417.
4. Tukar nilai gen ke 11 yang lama dengan nilai random yang baru.

***Child* 1**	0.084	0.63	0.344	0.818	0.81	0.274	0.696	0.744
	0.073	0.803	0.417	0.495	0.041	0.53	0.325	0.879

Keterangan warna :

: Gen yang terpilih untuk dimutasi

: Gen sesudah dimutasi

3.4.7 Pembentukan Populasi baru

Individu-individu yang telah melalui proses *crossover* dan mutasi kemudian dimasukkan kedalam satu populasi sementara untuk diseleksi membentuk satu populasi baru. Pembentukan populasi baru dilakukan berdasarkan nilai *fitness* terbaik. Untuk itu perlu dilakukan perhitungan *fitness* lagi untuk masing-masing kromosom. Dengan cara yang sama seperti sebelumnya, *fitness* masing-masing kromosom dapat dilihat pada Tabel 3.5

Tabel 3.5 Hasil *fitness* tiap kromosom setelah proses reproduksi

	fitness
Kromosom 1	**0.82857**
Kromosom 2	**1.02689**
Kromosom 3	**1.40317**
Kromosom 4	**2.41029**
Child 1	**0.87125**
Child 2	**1.3284**
Child 1 (mutasi)	**0.84838**

Selanjutnya urutkan kromosom berdasarkan *fitness* dimulai dari yang terbesar hingga terkecil. Hasil pengurutan dapat dilihat pada Tabel 3.6.

Tabel 3.6 Hasil pengurutan individu berdasarkan *fitness*

	fitness
Kromosom 4	**2.41029**
Kromosom 3	**1.40317**
Child 2	**1.3284**
Kromosom 2	**1.02689**
Child 1	**0.87125**
Child 1 (mutasi)	**0.84838**
Kromosom 1	**0.82857**

Kemudian ambil empat kromosom teratas untuk dijadikan populasi baru, sedangkan individu yang *fitness*-nya tidak mencukupi akan dibuang. Individu yang terpilih untuk membentuk populasi baru disebut juga generasi ke-1. Proses genetika ini diulang terus hingga generasi maksimum yang ditentukan. Individu pada populasi baru dapat dilihat pada Tabel 3.7.

Tabel 3.7 Individu generasi ke-1

	Fitness
Kromosom 4	**2.41029**
Kromosom 3	**1.40317**
Child 2	**1.3284**
Kromosom 2	**1.02689**

3.4.8 Proses Peramalan

Setelah proses pelatihan selesai dilakukan, maka akan didapat bobot optimal yang akan digunakan pada jaringan untuk proses peramalan. Proses peramalan menggunakan data uji sebanyak 41 record data, berupa data harga valas bulan Januari-Februari 2011. Sama seperti data latihnya, data uji juga terdiri dari harga *open*, *high*, *low* dan *close* valas.

Tabel 3.8 Data Uji

No	Date	open	high	low	close
1	2011.01.03	1.5556	1.5562	1.5431	1.5482
2	2011.01.04	1.5484	1.5649	1.5449	1.5588
3	2011.01.05	1.5589	1.5627	1.5449	1.5514
4	2011.01.06	1.5513	1.5564	1.5448	1.5472
5	2011.01.07	1.5467	1.5578	1.5404	1.5545
...	...	...	...	...	...
...	...	...	...	...	...
37	2011.02.22	1.6224	1.6234	1.6102	1.6135
38	2011.02.23	1.6136	1.6274	1.6132	1.6209
39	2011.02.24	1.6208	1.6254	1.6083	1.6134
40	2011.02.25	1.6135	1.616	1.603	1.6116
41	2011.02.28	1.6112	1.6272	1.6067	1.6251

Peramalan dilakukan dengan proses *feedforward* jaringan *backpropagation*. Sebelumnya data dinormalisasi terlebih dahulu. Langkah-langkah peramalan adalah sebagai berikut :

1. Normalisasi data

   Untuk menghitung normalisai data, hitung nilai rata-rata terlebih dahulu dengan menjumlahkan seluruh data kemudian dibagi dengan jumlah data. Dengan menggunakan data pada Tabel 3.8, nilai rata-ratanya adalah 1.5942. Kemudian menghitung nilai deviasi standar dengan Persamaan 2.13.

$$\text{deviasi standar} = \sqrt{\frac{\sum(x-\bar{x})^2}{n-1}} = \sqrt{\frac{0.0015+0.0014+\cdots+0.0002+0.001}{122}}$$

$$= \sqrt{\frac{0.006796}{122}} = 0.0236$$

   Sama seperti cara sebelumnya untuk menghitung normalisasi data menggunakan Persamaan 2.12 :

$$\text{Nilai baru} = \frac{(\text{Nilai lama} - \text{Ratarata})}{\text{Deviasi Standar}}$$

Hasil normalisasi ditampilkan pada Tabel 3.9.

Tabel 3.9 Data uji yang sudah dinormalisasi

No	Date	open	high	low	close
1	2011.01.03	-1.63559	-1.61017	-2.16525	-2.07895
2	2011.01.04	-1.94068	-1.24153	-2.08898	-1.61404
3	2011.01.05	-1.49576	-1.33475	-2.08898	-1.9386
4	2011.01.06	-1.8178	-1.60169	-2.09322	-2.12281
5	2011.01.07	-2.01271	-1.54237	-2.27966	-1.80263
...	...	...	...	...	...
...	...	...	...	...	...
37	2011.02.22	1.194915	1.237288	0.677966	0.785088
38	2011.02.23	0.822034	1.40678	0.805085	1.109649
39	2011.02.24	1.127119	1.322034	0.597458	0.780702
40	2011.02.25	0.817797	0.923729	0.372881	0.701754
41	2011.02.28	0.720339	1.398305	0.529661	1.29386

2. Proses *feedforward*

   Proses ini menggunakan bobot hasil pelatihan dan data uji. Misalnya digunakan kromosom 4, karena nilai *fitness*-nya paling tinggi yaitu 2.41029.

Kromosom 4	0.084	0.63	0.344	0.818	0.81	0.12	0.681	0.844
	v01	v02	v03	V11	V12	V13	V21	V22
	0.733	0.564	0.556	0.572	0.719	0.207	0.788	0.137
	v23	v31	v32	v33	w1	w2	w3	w0

Untuk data pertama, hitung keluaran unit tersembunyi dengan Persamaan 2.6 :

$$z_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

z_in_1 = 0.084+(-1.63559)(0.818)+(-1.61017)(0.681)+
(-2.16525)(0.564)
= -3,57164

z_in_2 = 0.63+(-1.63559)(0.81)+(-1.61017)(0.844)+
(-2.16525)(0.556)
= -3,25769

z_in_3 = 0.344+(-1.63559)(0.12)+(-1.61017)(0.733)+
(-2.16525)(0.573)
= -2,27105

Hitung aktivasi unit tersembunyi dengan Persamaan 2.7 :

$$z_j = f\left(z_{in\,j}\right) = \frac{1 - e^{-z_in_j}}{1 + e^{-z_in_j}}$$

$$z_1 = f\,(z_in_1) = \frac{1 - e^{-(-3,57164)}}{1 + e^{-(-3,57164)}} = -0.9453$$

$$z_2 = f\,(z_in_2) = \frac{1 - e^{-(-3,25769)}}{1 + e^{-(-3,25769)}} = -0.92587$$

$$z_3 = f\,(z_in_3) = \frac{1 - e^{-(-2,27105)}}{1 + e^{-(-2,27105)}} = -0.81286$$

Hitung keluaran unit y_k dengan Persamaan 2.8 :

$$y_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

$$y_in = w_0 + z_1 w_1 + z_2 w_2 + z_3 w_3$$

y_in = 0.137 + (-0.9453) (0.719) + (-0.92587)(0.207) +
(- 0.81286)(0.788)
= -1.3749

Aktivasi keluaran unit y_k dengan Persamaan 2.9 :

$$y_k = f(y_{in_k}) = y_{in_k}$$

$$y = f(y_{in}) = -1.3749$$

Hitung kembali nilai y untuk data kedua dan seterusnya dengan langkah yang sama seperti contoh perhitungan data 1 dan tetap menggunakan kromosom atau nilai bobot yang sama.

Hasil perhitungan *feedforward* untuk seluruh data uji dengan menggunakan kromosom 4 ditunjukkan pada Tabel 3.10.

Tabel 3.10 Hasil perhitungan *feedforward* data uji

Data	Zin 1	Zin2	Zin3	Z1	Z2	Z3	y_in	y
1	-3.572	-3.258	-2.271	-0.945	-0.926	-0.813	-1.375	-1.375
2	-3.527	-3.151	-1.994	-0.943	-0.918	-0.760	-1.33	-1.33
3	-3.227	-2.870	-2.009	-0.924	-0.893	-0.763	-1.313	-1.313
4	-3.674	-3.358	-2.246	-0.951	-0.933	-0.808	-1.377	-1.377
5	-3.899	-3.570	-2.332	-0.960	-0.945	-0.823	-1.398	-1.398
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…
37	2.2864	3.0191	1.7821	0.815	0.9068	0.7119	1.472	1.472
38	2.1685	2.9308	1.9343	0.795	0.8987	0.747	1.4834	1.4834
39	2.2433	2.991	1.7901	0.8081	0.9043	0.7138	1.4677	1.4677
40	1.5923	2.2794	1.3325	0.6618	0.8143	0.5825	1.2404	1.2404
41	1.9242	2.6881	1.7584	0.745	0.8726	0.706	1.4097	1.4097

3. Denormalisasi Data

   Proses *feeedforward* menghasilkan nilai *output* peramalan yang berupa nilai ternormalisasi. Maka perlu dilakukan proses

denormalisasi untuk mengubah nilai *output* kembali berada pada range data harga *close* aktual. Proses denormalisasi dihitung menggunakan Persamaan 2.13.

$$Denormalisasi = (Nilai\ normalisasi * deviasi\ standar) + Ratarata$$

Contoh perhitungannya sebagai berikut :
Data 1 = (-0.84016 x 0.02277)+ 1.5956 = 1.5764445
Data 2 = (-0.81832 x 0.02277)+ 1.5956 = 1.5769423
Data 3 = (-0.80217 x 0.02277)+ 1.5956 = 1.5773104

Tabel 3.11 Hasil *output* yang telah di-denormalisasi

No	Date	y	Denormalisasi
1	2011.01.03	-1.37486	1.56425318
2	2011.01.04	-1.33001	1.56527575
3	2011.01.05	-1.31341	1.56565435
4	2011.01.06	-1.37657	1.56421427
5	2011.01.07	-1.39756	1.56373568
...	...	...	...
...	...	...	...
37	2011.02.22	1.471977	1.62916108
38	2011.02.23	1.483394	1.62942138
39	2011.02.24	1.467701	1.62906357
40	2011.02.25	1.2404	1.62388112
41	2011.02.28	1.409708	1.62774134

4. Evaluasi peramalan

   Error yang dihasilkan dari selisih nilai antara data harga *close* aktual dengan harga *close* hasil ramalan. *Error* dihitung dengan Persamaan 2.16 dan Persamaan 2.17. Hasil perhitungan *error* untuk semua data uji ditampilkan pada Tabel 3.12.

Tabel 3.12 *Error* peramalan harga *close* valas

No	Date	Close aktual	Close Peramalan	$Error^2$
1	2011.01.03	1.5482	1.56425318	0.000257705
2	2011.01.04	1.5588	1.56527575	4.19354E-05
3	2011.01.05	1.5514	1.56565435	0.000203187
4	2011.01.06	1.5472	1.56421427	0.000289485
5	2011.01.07	1.5545	1.56373568	8.52978E-05
...	...	...	...	...
...	...	...	...	...
37	2011.02.22	1.6135	1.62916108	0.000245269
38	2011.02.23	1.6209	1.62942138	7.26139E-05
39	2011.02.24	1.6134	1.62906357	0.000245348
40	2011.02.25	1.6116	1.62388112	0.000150826
41	2011.02.28	1.6251	1.62774134	6.97665E-06

Hitung nilai kesalahan dengan RMSE sesuai Persamaan 2.16.

$$RMSE = \sqrt{\sum_{i=1}^{n} e_i^2 / n}$$

$$\sqrt{\frac{0.002862994 + 0.001852052 + \cdots + 5.14606E{-}05 + 0.000421137}{41}}$$

$$= 0.52228$$

Selanjutnya hitung nilai kesalahn dengan MAPE sesuai Persamaan 2.17 :

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{e_i}{Xi}\right|$$

$$= 16.302 / 41$$
$$= 0.398$$

3.5 Perancangan Desain Data

Struktur data pada penelitian ini adalah `valuta`, `kromosom`, dan `parent`. Masing-masing tipe data akan dijelaskan pada Tabel 3.13.

Tabel 3.13 Tipe Data yang Digunakan

No	Tipe data	
1	Nama	`valuta`
	Deskripsi	untuk menyimpan data valuta asing yang akan digunakan pada penelitian ini. Untuk memudahkannya satu tipe valuta terdiri dari 4 harga yang dibedakan berdasarkan jenis harga

	Deklarasi	`public class valuta {`
		`  double open;`
		`  double high;`
		`  double low;`
		`  double close;`
		`}`

	Operasi	*getValas, getClose, getOpen, getHigh, getLow*
2	Nama	`Kromosom`
	Deskripsi	untuk menyimpan nilai dari gen, bobot V, bobot W dan nilai *fitness* untuk masing-masing kromosom

	Deklarasi	`public class kromosom {`
		`  double gen[] = new double [16];`
		`  double V [][]= new double [4][3];`
		`  double W[]= new double[4];`
		`  double fitness;`
		`}`

	Opeasi	*Generate, getFitness*
3	Nama	`parent`
	Deskripsi	untuk menyimpan pasangan kromosom yang akan digunakan untuk proses *crossover*

	Deklarasi	`public class parent {`
		`  kromosom P1;`
		`  kromosom P2;`
		`}`

	Operasi	*getP1. getP2*

3.6 Perancangan Uji Coba

Uji coba sistem untuk peramalan harga valas dilakukan dengan memasukkan nilai-nilai parameter genetik yang berbeda terhadap data latih. Tujuan dari uji coba adalah untuk mengetahui pengaruh parameter genetik yang digunakan pada proses pelatihan. Setelah proses pelatihan dilakukan, maka bobot terbaik yang dihasilkan akan dievaluasi menggunakan sejumlah data uji, kemudian hitung tingkat kesalahan peramalannya.

Uji pengaruh parameter genetik dilakukan pada sejumlah data latih yaitu data harga valas selama satu tahun. Uji parameter genetik dilakukan untuk mendapatkan kromosom atau individu dengan nilai *fitness* terbaik pada parameter tertentu untuk digunakan pada proses peramalan. Uji parameter pertama yang dilakukan adalah pengaruh probabilitas *crossover* dan mutasi terhadap nilai *fitness* dengan jumlah generasi dan ukuran populasi yang sama. Untuk setiap pengujian dilakukan sepuh kali percobaan kemudian diambil nilai rata-ratanya. Tabel uji untuk pengaruh probabilitas *crossover* dan mutasi dapat dilihat pada Tabel 3.14.

Tabel 3.14 Percobaan Pengaruh Perubahan Probabilitas *Crossover* dan Probabilitas Mutasi terhadap Nilai *Fitness*

Pm	Probabilitas *Crossover* (Pc)				
	0.1	0.3	0.5	0.7	0.9
0.1					
0.3					
0.5					
0.7					
0.9					

Uji parameter selanjutnya ialah pengaruh ukuran populasi dan jumlah generasi terhadp nilai *fitness*. Sama seperti pengaruh probabilitas *crossover* dan mutasi, untuk setiap pengujian dilakukan sepuh kali percobaan kemudian diambil nilai rata-ratanya. Tabel uji untuk pengaruh jumlah generasi terhadap *fitness* dapat dilihat pada Tabel 3.15, sedangkan tabel uji untuk pengaruh ukuran populasi terhadap *fitness* dapat dilihat pada Tabel 3.16.

Tabel 3.15 Percobaan Pengaruh Jumlah Generasi terhadap Nilai *Fitness*

No	Jumlah Generasi	Pc 90% ; Pm 30%	
		Percobaan ke-	*Fitness* rata-rata
1	10		
2	50		
3	100		
4	…		
5	n		

Tabel 3.16 Percobaan Pengaruh Ukuran Populasi terhadap Nilai *Fitness*

No	Ukuran Populasi	Pc 90% ; Pm 30%	
		Percobaan ke-	*Fitness* rata-rata
1	10		
2	50		
3	100		
4	…		
5	n		

Setelah didapatkan parameter genetika yang menghasilkan kromosom dengan *fitness* terbaik, kromosom tersebut digunakan sebagai bobot jaringan untuk menentukan *output* jaringan pada peramalan harga valuta asing. Tabel uji untuk hasil peramalan terhadap data aktual harga valuta asing dapat dilihat pada Tabel 3.17.

Tabel 3.17 Hasil peramalan terhadap data aktual harga valuta asing

No	Date	*Close* aktual	*Close* Peramalan	*error*

Dari Tabel 3.17 dapat dihitung selisih harga valuta asing hasil peramalan terhadap data harga valuta asing aktual. Selanjutnya hitung nilai RMSE dan MAPE untuk mengevaluasi kesalahan peramalan sekaligus menentukan tingkat akurasi hasil peramalan tersebut.

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

4.1 Lingkungan Implementasi

Implementasi merupakan representasi rancangan berupa aplikasi peramalan nilai tukar valuta asing menggunakan jaringan syaraf tiruan yang menggunakan algoritma genetika pada proses pelatihan jaringan syaraf tiruan. Adapun yang akan dijelaskan dalam subbab ini meliputi lingkungan implementasi perangkat keras dan perangkat lunak.

4.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan dan pengujian aplikasi peramalan harga valuta asing ini adalah sebuah Notebook dengan spesifikasi sebagai berikut :

1. Prosesor Intel(R) Core(TM)2 Duo CPU T7100 @1.80GHz
2. Memori 1 Gb
3. Harddisk 160 Gb
4. Monitor 14”
5. Keyboard
6. Mouse

4.1.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan program peramalan harga valuta asing dan uji coba adalah:

1. Sistem operasi Windows 7 32-bit
2. Bahasa pemrograman Java
3. NetBeans IDE 7

4.2 Implementasi Program

Berdasarkan analisa dan perancangan proses yang telah dipaparkan pada Bab III, maka pada bab ini akan dijelaskan proses-proses implementasinya.

4.2.1 Implementasi tipe data `valuta`

Tahap awal yang dilakukan pada peramalan harga valuta asing adalah memanggil data valas berupa *file excel* yang akan digunakan sebagai data latih pada proses pelatihan. Setelah file dipilih, isi data selanjutnya dibaca dan disimpan dalam array 2 dimensi. Langkah

selanjutnya adalah membuat setiap baris data menjadi sebuah data bertipe `valuta`.

Proses untuk mengubah data menjadi tipe valuta menggunakan sebuah prosedur `GenerateValuta`, yang dapat dilihat pada *sourcecode* 4.1. Pada baris ke-4 mendeklarasikan setiap baris `datlat` menjadi tipe `valuta`. Selanjutnya pada kode baris ke-5 sampai baris ke-8 adalah proses mengisi nilai dari variabel `datlat` dari array 2-dimensi `data`.

```
1  public valuta[] GenerateValuta(double[][] data) {
2     datlat = new valuta[data.length];
3      for (int i = 0; i < data.length; i++) {
4        datlat[i] = new valuta();
5        datlat[i].open = data[i][0];
6        datlat[i].high = data[i][1];
7        datlat[i].low = data[i][2];
8        datlat[i].close = data[i][3];
9       }
10     return datlat;
11  }
```

***Sourcecode* 4.1** Pembuatan data bertipe `valuta`

4.2.2 Implementasi normalisasi data

Masing-masing jenis harga valas, yaitu harga *open*, *high*, *low* dan *close* akan dinormalisasi berdasarkan persamaan 2.11. Implementasi dari proses normalisasi data dapat dilihat pada *sourcecode* 4.2.

```
1  public valuta[] normval(valuta val[]) {
2     norm = new valuta[val.length];
3     mean = mean(val);
4     SD = hitSTDEV(val);
5      for (int i = 0; i < norm.length; i++) {
6        norm[i] = new valuta();
7        norm[i].open = (val[i].open - mean) / SD;
8        norm[i].high = (val[i].high - mean) / SD;
9        norm[i].low = (val[i].low - mean) / SD;
10       norm[i].close = (val[i].close - mean) / SD;
11      }
12    return norm;
13 }
```

***Sourcecode* 4.2** Proses normalisasi data

Untuk melakukan normalisasi data, harus terlebih dahulu menghitung nilai rata-rata dan standar deviasi dari data yang akan dinormalisasi. Pada *sourcecode* 4.2 baris ke-2 dan ke-3 terdapat pemanggilan fungsi `mean`*`Input`* untuk mendapatkan nilai rata-rata data dan `hitSTDEV` untuk mendapatkan nilai standar deviasi data. Setelah itu hitung normalisasi tiap data seperti pada kode baris ke-7 sampai baris ke-10 berdasarkan rumus normalisasi yang digunakan.

4.2.3 Implementasi inisialisasi populasi awal

Populasi awal dibuat dengan membangkitkan nilai random untuk setiap gen pada masing-masing kromosom. Nilai random yang dibangkitkan bertipe *double* antara 0 sampai 1, dan nilai tiap gen ini merupan bobot-bobot pada model jaringan syaraf tiruan yang digunakan. Banyaknya jumlah individu atau kromosom dalam satu populasi ditentukan oleh nilai masukan. Implementasi untuk inisialisasi populasi awal dapat dilihat pada *sourcecode* 4.3.

```
1  public void GeneratePop() {
2   krom = new kromosom[nPop];
3     for (int i = 0; i < krom.length; i++) {
4       krom[i] = new kromosom();
5         for (int j = 0; j < krom[i].gen.length; j++) {
6           krom[i].gen[j] = rdm.nextDouble();
7         }
8     }
9   }
```

***Sourcecode* 4.3** Inisialisasi populasi awal

Deklarasi variabel `krom` yang bertipe array kromosom berada pada kode baris ke-2. Kode baris ke-3 sampai baris ke-6 merupakan proses pembangkitan nilai awal gen yang bernilai random antara 0-1.

4.2.4 Implementasi prosedur hitung *fitness*

Nilai *fitness* didapat dengan menghitung *output* jaringan menggunakan tahap *feedforward* jaringan syaraf tiruan untuk tiap data, dan menggunakan kromosom yang berisi bobot-bobot jaringan syaraf tiruan. Pada proses hitung *fitness* akan membawa variabel kromosom dan nilai normalisasi data latih valuta. Untuk lebih jelas, proses hitung *fitness* menggunakan tahap *feedforward* dapat dilihat pada *sourcecode* 4.4.

```
1  public double FitnessKrom(kromosom kr, valuta[] norm) {
2    for (int j = 0; j < norm.length; j++) {
3     // perulangan 3x karna menggunakan 3 hidden layer
4    for (int k = 0; k < 3; k++) {
5     zin[j][k] = ((kr.V[0][k]) + (norm[j].open *
kr.V[1][k])+
6          (norm[j].high * kr.V[2][k]) +
7          (norm[j].low * kr.V[3][k]));
8
9     Z[j][k]=(1-(Math.exp(-1*(zin[j][k]))))/(1+(Math.exp(-
1*(zin[j][k]))));
10    }
11
12   yin = kr.W[3];
13    for (int j2 = 0; j2 < 3; j2++) {
14      yin = yin + (Z[j][j2] * kr.W[j2]);
15    }
16
17   Y = yin;
18    error = Math.pow((norm[j].close - Y), 2);
19    sse = sse + error;
20    }
21    rmse = Math.sqrt(sse / norm.length);
22    fit = 1 / rmse;
23    sse = 0;
24    return fit;
25  }
```

***Sourcecode* 4.4** Prosedur hitung *fitness*

Pada *sourecode* 4.4, proses hitung *fitness* dilakukan dengan mencari nilai *output* untuk masing-masing data valuta asing yang telah dinormalisasi, oleh karena itu terdapat perulangan sebanyak jumlah data, seperti ditunjukkan pada kode baris ke-2. Pada kode baris ke-4 sampai baris ke-7 dihitung nilai Zin untuk tiap data, selanjutnya hitung nilai Z dengan fungsi aktivasi bipolar yang ditunjukkan pada kode baris ke-9. Nilai Z yang telah didapat merupakan *input*an pada layer *output*, maka hitung nilai Yin. Bobot bias untuk lapisan *output* pada kasus ini berada pada array indeks ketiga, sehingga implementasinya seperti kode baris ke-12. Baris kode ke-13 sampai baris ke-15 merupakan implementasi hitung Yin. Selanjutnya hitung nilai Y dengan fungsi aktivasi linier yang ditunjukkan pada kode baris ke-17. Kode baris ke-18 sampai baris ke-24 merupakan proses untuk mendapatkan nilai *fitness* kromosom.

4.2.5 Implementasi *select parent*

Setelah masing-masing kromosom dihitung nilai *fitness*-nya, maka proses selanjutnya adalah proses reproduksi yang terdiri dari proses pindah silang (*crossover*) dan mutasi. Sebelum melakukan proses reproduksi, perlu dipilih pasangan *parent* yang memiliki *fitness* baik agar menghasilkan keturunan yang baik pula. Proses pemilihan orang tua berdasarkan *fitness* pada penelitian ini menggunakan metode *roulette wheel*. Implementasi metode *roulette wheel* dapat dilihat pada *sourcecode* 4.5.

```
1  public kromosom rouletteWheel(kromosom krom[]) {
2    kromosom kandidat = new kromosom();
3     // hitung fitness kumulatif
4    for (int i = 0; i < krom.length; i++) {
5      totalfitness = totalfitness + krom[i].fitness;
6    }
7    //proses roda roulette
8    double rand = rdm.nextDouble();
9    while (y <= krom.length) {
10     fitkum = fitkum + krom[y].fitness;
11       if ((fitkum / totalfitness) > rand) {
12          kandidat = krom[y];
13          break;
14       }
15      y++;
16    }
17   return kandidat;
18 }
```

***Sourcecode* 4.5** Implementasi metode *roulette wheel*

Tahap awal dari proses seleksi *roulette wheel* adalah menghitung nilai *fitness* kumulatif untuk setiap populasi, yang ditunjukkan pada *sourecode* 4.5 baris ke-4 sampai ke-6. Selanjutnya baru menentukan daerah yang terpilih secara random yang diimplementasikan pada baris ke-8 sampai baris ke-16. Nilai *return* berupa kromsom dari daerah yang terpilih dan merupakan kromosom `kandidat` yang akan digunakan untuk proses *crossover*, seperti pada kode baris ke-12 lalu baris ke-17.

Suatu populasi yang terdiri dari beberapa individu dalam satu waktu akan melakukan reproduksi secara bersama-sama. Oleh karena itu akan dibentuk sekumpulan *parent* yang akan melakukan

crossover dalam satu waktu reproduksi. Setiap individu dalam populasi dapat dipasangkan dengan individu manapun dalam populasi tersebut selama tidak dengan dirinya sendiri. Implementasi dari pembentukan pasangan *parent* untuk *crossover* dapat dilihat pada *sourcecode* 4.6.

```
1  public parent[] getParent(kromosom krom[]) {
2    parent ot[] = new parent[krom.length];
3     for (int i = 0; i < ot.length; i++) {
4       ot[i] = new parent();
5       ot[i].P1 = new kromosom();
6       ot[i].P2 = new kromosom();
7       ot[i].P1 = rouletteWheel(krom);
8       ot[i].P2 = rouletteWheel(krom);
9   // syarat supaya tidak poligami
10    while (ot[i].P1.gen.equals(ot[i].P2.gen)) {
11      ot[i].P2 = rouletteWheel(krom);
12     }
13    }
14   return ot;
15  }
```

***Sourcecode* 4.6** Implementasi pembentukan *parent*

Proses pemilihan *parent* pertama dan kedua menggunakan metode *roulette wheel* diimplementasikan pada kode baris ke-7 dan ke-8. Selanjutnya untuk menghindari adanya pasangan kromosom yang sama dalam *parent*, dilakukan pengecekan yang diimplementasikan pada kode baris ke-10 sampai baris ke-12.

4.2.6 Implementasi operasi *crossover*

Proses *crossover* atau pindah silang merupakan proses dimana terjadi rekombinasi gen antara dua individu atau kromosom. Metode *crossover* yang digunakan dalam penelitian ini adalah *one-cut-point crossover*. Implementasi *crossover* dapat dilihat pada *sourcecode* 4.7.

```
1  public kromosom[] Crossover(parent ot) {
2    kromosom anak[] = new kromosom[2];
3    int cutpoint = rdm.nextInt(15);
4    double random = rdm.nextDouble();
5    while (cutpoint == 0 || cutpoint == 14) {
6       cutpoint = rdm.nextInt(15);
7    }
```

```
8     for (int i = 0; i < anak.length; i++) {
9       anak[i] = new kromosom();
10    }
11    if (random < Pc) {
12      // tukar gen
13      for (int j = 0; j < cutpoint; j++) {
14        anak[0].gen[j] = ot.P1.gen[j];
15        anak[1].gen[j] = ot.P2.gen[j];
16      }
17      for (int j2 = cutpoint; j2 < 16; j2++) {
18        anak[0].gen[j2] = ot.P2.gen[j2];
19        anak[1].gen[j2] = ot.P1.gen[j2];
20      }
21    } else {
22      //gen anak = gen parent
23        anak[0] = ot.P1;
24        anak[1] = ot.P2;
25    }
26   return anak;
27  }
```

***Sourcecode* 4.7** Implementasi operasi *crossover*

Pada kode baris ke-5 sampai baris ke-7 merupakan proses penentuan titik potong persilangan kromosom, yaitu dengan pembangkitan bilangan random antara 1-14. Pada kode baris ke-11 sampai baris ke-25 adalah proses *crossover* yang ditentukan oleh nilai probabilitas *crossover*.

4.2.7 Implementasi operasi mutasi

Teknik mutasi yang digunakan pada penelitian ini adalah mutasi acak. Implementasi dari operasi mutasi dapat dilihat pada *sourcecode* 4.8.

```
1   public kromosom[] MutKrom(kromosom[] krom, double pm) {
2    kromMut = new kromosom[krom.length];
3     for (int i = 0; i < krom.length; i++) {
4       kromMut[i] = new kromosom();
5        for (int j = 0; j < kromMut[i].gen.length; j++) {
6          acak = rdm.nextDouble();
7          newgen = rdm.nextDouble();
8           if (acak < pm) {
9             kromMut[i].gen[j] = newgen;
10          }
```

```
11          else {
12            kromMut[i].gen[j] = krom[i].gen[j];
13          }
14        }
15      }
16    return kromMut;
17  }
```

***Sourcecode* 4.8** Implementasi operasi mutasi

Kode baris ke-3 dan ke-5 merupakan proses perulangan yang digunakan untuk menentukan gen yang akan mengalami mutasi. Kode baris ke-6 merupakan proses untuk menetukan nilai acak secara random untuk dibandingkan dengan probabilitas mutasi pada kode baris ke-8, sedangkan pada baris ke-7 merupakan proses membangkitkan nilai gen baru yang akan menggantikan gen yang mengalami mutasi. Proses mutasi diimplementasikan pada kode baris ke-8 sampai baris ke-12. Apabila gen mengalami mutasi maka akan digantikan dengan nilai acak baru (newgen) yang diimplementasikan pada kode baris ke-8, apabila tidak maka nilai gen tetap, seperti diimplementasikan pada kode baris ke-12.

4.2.8 Implementasi seleksi populasi baru

Proses seleksi populasi baru pada penelitian ini menggunakan metode *rank selection*. Implementasi dari proses seleksi populasi dapat dilihat pada *sourcecode* 4.9.

```
1   public kromosom[] seleksiKrom(kromosom[] krom,
    kromosom[] anak) {
2     tempkrom = new kromosom[krom.length + anak.length];
3     for (int i = 0; i < krom.length; i++) {
4       tempkrom[i] = new kromosom();
5       tempkrom[i] = krom[i];
6      }
7     for (int i = 0; i < anak.length; i++) {
8       tempkrom[i + krom.length] = new kromosom();
9       tempkrom[i + krom.length] = anak[i];
10    }
11  // mengurutkan kromosom
12    kromosom temp;
13    int d = tempkrom.length - 1;
14    for (int i = 0; i < d; i++) {
15      for (int j = 0; j < d - i; j++) {
```

```
16          if (tempkrom[j].fitness < tempkrom[(j +
1)].fitness) {
17            temp = tempkrom[j];
18            tempkrom[j] = tempkrom[j + 1];
19            tempkrom[j + 1] = temp;
20          }
21       }
22     }
23    return tempkrom;
24  }
```

***Sourcecode* 4.9** Implementasi proses seleksi

Untuk melakukan seleksi terhadap kromosom hasil rekombinasi, seluruh kromosom digabung dalam satu array yaitu `tempkrom`, yang ditunjukkan pada kode baris ke-2. Kode baris ke-3 hingga baris ke-6 adalah proses memasukkan kromosom awal (`krom`) ke dalam `tempkrom`, sedangkan baris ke-7 hingga baris ke-10 adalah proses memasukkan kromosom anak kedalam `tempkrom`. Selanjutnya seluruh kromosom diurutkan mulai *fitness* tertinggi hingga terendah, yang ditunjukkan pada kode baris ke-12 hingga baris ke-22.

Setelah proses seleksi dilakukan maka akan terbentuk generasi baru yang memiliki nilai *fitness* lebih baik dari individu pada populasi sebelumnya. Proses regenerasi terus dilakukan hingga didapatkan populasi yang konvergen yang terdiri dari kromosom dengan nilai *fitness* terbaik.

4.2.9 Implementasi pengujian *output* jaringan

Setelah proses pelatihan selesai dilakukan maka akan terdapat sebuah populasi yang konvergen, yang mana semua kromosom di dalamnya bernilai sama. Kromosom tersebut dianggap sebagai kromosom terbaik dan merupakan solusi permasalahan. Nilai dari kromosom tersebut kemudian diambil untuk digunakan pada proses pengujian atau peramalan sebagai parameter atau bobot jaringan syaraf tiruan yang akan digunakan.

Proses pengujian dilakukan dengan menggunakan proses *feedforward* jaringan syaraf tiruan untuk menghasilkan *output* atau disebut juga hasil peramalan. *Output* yang dihasilkan oleh jaringan masih berupa nilai normalisasi, untuk itu perlu dilakukan denormalisasi data sehingga data akan kembali berada pada kisaran

awal sebelum dinormalisasi. Kemudian dilakukan pencocokan dengan data aktual yaitu harga *close* valuta asing. Setelah itu dicari nilai selisih kesalahannya, kemudian dihitung nilai RMSE dan MAPE. Implementasi dari proses pengujian dapat dilihat pada *sourcecode* 4.10.

```
1   public void peramalan(kromosom kr, valuta[] norm) {
2     for (int j = 0; j < norm.length; j++) {
3     // perulangan 3x karna menggunakan 3 hidden layer
4       for (int k = 0; k < 3; k++) {
5         zin[j][k] = ((kr.V[0][k]) + (norm[j].open *
      kr.V[1][k])
6                 + (norm[j].high * kr.V[2][k]) +
7                   (norm[j].low * kr.V[3][k]));
8       Z[j][k]=(1-(Math.exp(-1*(zin[j][k]))))/(1+(Math.exp(-
      1*(zin[j][k]))));
9       }
10       yin = kr.W[3];
11      for (int j2 = 0; j2 < 3; j2++) {
12        yin = yin + (Z[j][j2] * kr.W[j2]);
13       }
14      Y[j] = yin;
15      denormalisasi den = new denormalisasi(Y, datauji);
16      ramalan = den.GetOutputramal();
17      error = Math.pow((datauji[j].close - ramalan[j]), 2);
18      sse = sse + error;
19      error2[j]= Math.abs((datauji[j].close -
      ramalan[j])/datauji[j].close);
20      sse2 = sse2 + error2[j];
21    }
22    rmse = Math.sqrt(sse / datauji.length);
23    mape = sse2/datauji.length;
24  }
25
```

***Sourcecode* 4. 10** Implementasi proses pengujian

Proses untuk mendapatkan nilai *output* jaringan sama dengan proses *feedforward* pada proses pelatihan. Selanjutnya nilai *output* yang dihasilkan di-denormalisasi sehingga nilainya kembali pada *range* semula. Implementasi proses denormalisasi ditunjukkan pada *sourcecode* 4.11 baris ke-15. Selanjutnya hasil denormalisasi disimpan dalam variabel `ramalan`, yang diimplementasikan pada kode baris ke-16. Untuk mengevaluasi hasil peramalan harga valas

digunakan RMSE dan MAPE, maka hitung selisih harga antara harga aktual dan harga ramalan. Baris ke-19 sampai baris ke-23 merupakan implementasi untuk mengevaluasi hasil peramalan.

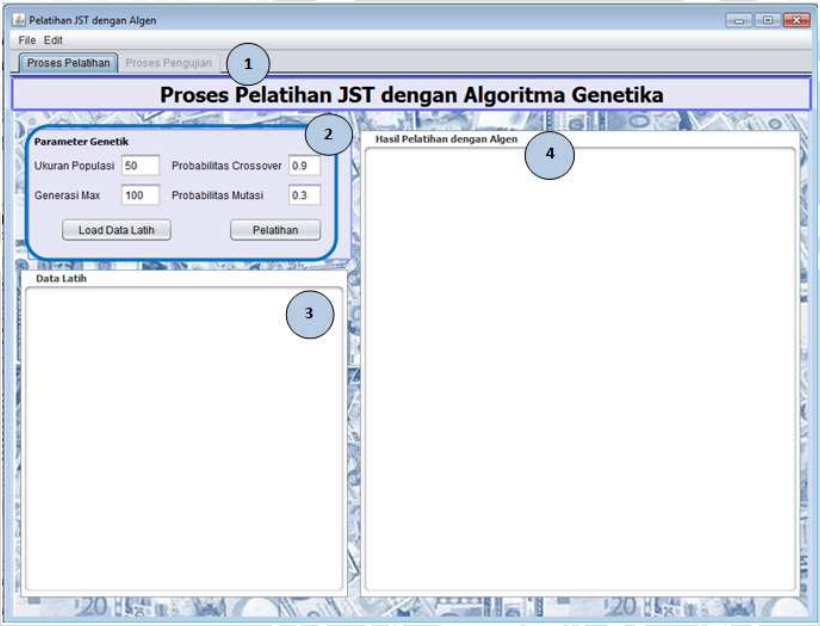
```
1  public double[] denorm(double[] d, valuta[] d2) {
2    output = new double[d.length];
3    meanOutput = meanInput(d2);
4    SDoutput = hitSTDEV(d2);
5    for (int i = 0; i < d.length; i++) {
6       output[i] = (d[i] * SDoutput) + meanOutput;
7    }
8   return output;
9  }
```

***Sourcecode* 4. 21** Implementasi proses denormalisasi data

Untuk melakukan denormalisasi data, hitung terlebih dahulu nilai rata-rata dan standar deviasi dari data yang akan di-denormalisasi. Kode baris ke-3 merupakan proses pemanggilan fungsi `meanInput` untuk mendapatkan nilai rata-rata data, sedangkan baris ke-4 merupakan proses pemanggilan fungsi `hitSTDEV` untuk mendapatkan nilai standar deviasi data. Setelah itu hitung denormalisasi tiap data seperti pada kode baris ke-6 berdasarkan rumus denormalisasi yang digunakan.

4.3 Implementasi Antar Muka

Berdasarkan rancangan antar muka yang telah dikemukakan pada Bab 3 maka dihasilkan antarmuka pada Gambar 4.1.

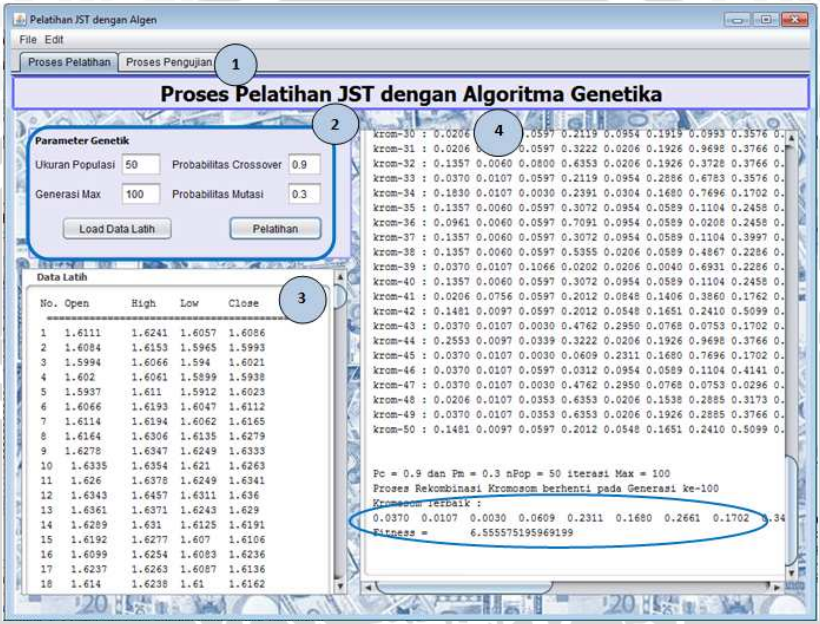


Gambar 4.1 Antar muka utama

Tampilan antarmuka pada aplikasi ini terdiri dari dua panel utama yang mewakili masing-masing proses yang akan dilakukan, yaitu panel pelatihan untuk menjalankan proses pelatihan dan panel pengujian untuk proses pengujian atau peramalan, yang ditunjukan pada bagian nomor 1. Hal pertama yang harus dilakukan sebelum memulai proses pelatihan adalah memasukkan nilai untuk masing-masing parameter genetika, yaitu ukuran populasi, jumlah iterasi maksimal, probabilitas *crossover* dan probabilitas mutasi, yang berada di daerah nomor 2.

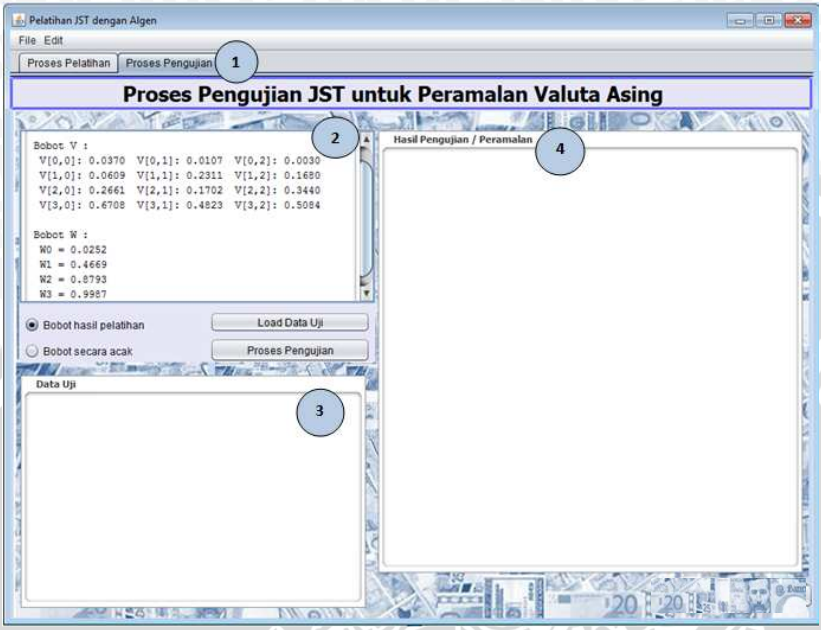
Apabila semua masukan parameter genetik telah diisi, masukan data latih yang akan digunakan dengan menekan tombol load data latih. Maka nantinya data latih akan tampil pada bagian nomor 3. Setelah itu baru menekan tombol proses pelatihan untuk mendapatkan nilai-nilai bobot jaringan syaraf tiruan untuk

melakukan peramalan data valuta asing. Hasil dari proses pelatihan akan ditampilkan pada bagian nomor 4. Untuk lebih jelasnya dapat dilihat pada Gambar 4.2.



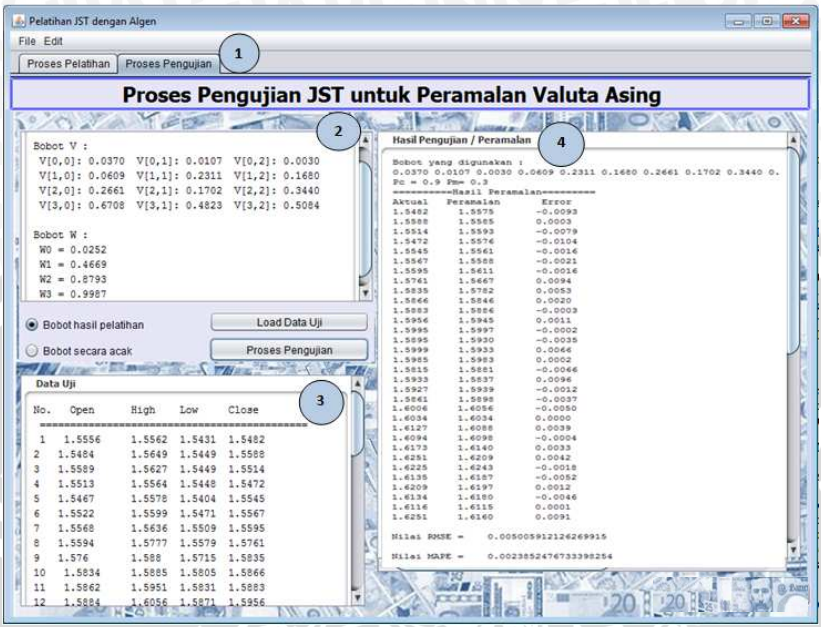
Gambar 4.2 Tampilan proses pelatihan

Proses selanjutnya adalah pengujian atau peramalan yang ditunjukkan pada Gambar 4.3. User menekan tab proses pengujian pada bagian nomor 1. Kromosom yang dihasilkan oleh algoritma genetika pada proses pelatihan selanjutnya digunakan untuk proses pengujian dan akan tampil pada bagian nomor 2. Kromosom yang didapat telah diubah formatnya menjadi bobot V dan bobot W untuk jaringan syarf tiruan.



Gambar 4.3 Tampilan kromosom pada proses pelatihan

User memasukkan data uji yang akan digunakan dengan menekan tombol load data uji, setelah itu data uji akan ditampilkan pada bagian nomor 3. Setelah itu tekan tombol proses pengujian, maka akan dihasilkan nilai-nilai *output* atau peramalan harga *close* valuta asing berdasarkan harga *open*, *high* dan *low* masing-masing hari. Hasil peramalan, nilai kesalahan peramalan, serta RMSE akan tampil pada bagian nomor 4. Untuk lebih jelasnya dapat dilihat pada Gambar 4.4.



Gambar 4.4 Tampilan proses pengujian

4.4 Sistematika Pengujian

Proses peramalan sangat dipengaruhi oleh nilai bobot jaringan yang dihasilkan oleh proses pelatihan. Oleh karena itu perlu dilakukan uji coba untuk nilai-nilai parameter genetic terhadap data latih dan data uji sehingga didapat bobot terbaik yang menghasilkan peramalan yang optimal. Sistematika pengujian yang akan dilakukan sesuai dengan penjelasan pada bab 3 tentang rancangan pengujian.

1. Pengaruh perubahan probabilitas *crossover* dan probabilitas mutasi terhadap nilai *fitness*.

   Pada percobaan ini akan dilakukan kombinasi-kombinasi dari nilai-nilai probabilitas *crossover* dan mutasi antara 0.1 sampa1 0.9. Percobaan dilakukan sebanyak sepuluh kali dan kemudian dihitung nilai rata-rata setiap percobaan. Kemudian hasilnya akan ditampilkan dalam bentuk tabel dan grafik sehingga lebih mudah diamati.

2. Pengaruh perubahan jumlah generasi dan pengaruh ukuran populasi terhadap nilai *fitness*.
   Sama halnya pengujian sebelumnya, untuk pengujian ini percobaan juga dilakukan sebanyak sepuluh kali lalu dihitung rata-ratanya. Hasil dari percobaan juga akan dibuat dalam bentuk tabel dan grafik sehingga mudah untuk diamati.
3. Pengujian hasil peramalan terhadap harga aktual.
   Harga *close* valuta asing yang dihasilkan *output* jaringan dibandingkan dengan data harga *close* valas sebenarnya. Pengujian ini menggunakan rumus evaluasi *error* yaitu RMSE dan MAPE untuk mengevaluasi kesalahan peramalan sekaligus mengetahui tingkat akurasi nilai peramalan yang dihasilkan oleh sistem.

4.5 Uji Coba dan Analisis Hasil

Pada penelitian ini terjadi perubahan nilai *fitness* yang dipengaruhi oleh parameter genetic, antara lain ukuran populasi, jumlah generasi, probabilitas *crossover*, dan probabilitas mutasi.

4.5.1 Pengaruh Probabilitas *Crossover* dan Probabilitas Mutasi terhadap Nilai *Fitness*

Uji coba pertama dilakukan untuk mengetahui pengaruh probabilitas *crossover* dan probabilitas mutasi terhadap nilai *fitness*. Untuk setiap kombinasi nilai probabilitas *crossover* dan probabilitas mutasi dilakukan percobaan sebanyak 10 kali dan diambil rata-ratanya. Parameter genetika yang nilainya selalu sama pada uji coba ini adalah ukuran populasi sebanyak 50 individu atau kromosom, dan maksimum generasi sebanyak 10.000 generasi. Hasil pengujian dapat dilihat pada Tabel 4.1 hingga Tabel 4.5.

Table 4.1 Nilai fitness pada probabilitas crossover 0.1 dan probabilitas mutasi yang berbeda

Uji Coba	Probabilitas Mutasi				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
1	7.86000	8.08782	7.62127	6.83481	5.62381
2	7.98901	8.13700	7.74577	6.87967	5.84774
3	7.95295	8.12773	7.62409	6.86402	6.31828
4	8.03171	8.13191	7.82562	6.89986	5.74050

Uji Coba	Probabilitas Mutasi				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
5	7.88577	8.13406	7.78291	6.73695	6.02378
6	7.96754	8.07519	7.63684	7.62398	6.32826
7	8.04594	8.12489	7.76858	6.95388	5.76732
8	8.02230	8.14339	7.63990	6.67273	5.78719
9	7.75668	8.11831	7.63025	7.27582	5.64668
10	7.88299	8.10702	7.81492	6.90642	6.07453
Rata-rata	**7.93949**	**8.11873**	**7.70901**	**6.96481**	**5.91581**

Table 4.2 Nilai fitness pada probabilitas crossover 0.3 dan probabilitas mutasi yang berbeda

Uji Coba	Probabilitas Mutasi				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
1	8.074288	8.141633	7.666457	6.687707	6.383444
2	7.806202	8.115774	7.757139	6.691426	6.361191
3	8.14272	8.100842	7.601691	7.083243	6.147065
4	7.96107	8.149858	7.593588	6.642706	5.671496
5	7.660566	8.119708	7.754881	6.905883	6.110745
6	8.075578	8.13865	7.611585	6.894071	5.894755
7	8.012164	8.155581	7.708818	6.79802	5.837278
8	7.939622	8.133116	7.593742	7.089178	5.474117
9	8.11584	8.099221	7.654633	6.873892	5.585545
10	8.06113	8.165068	7.673971	6.512297	6.072803
Rata-rata	**7.984918**	**8.131945**	**7.661651**	**6.817842**	**5.953844**

Table 4.3 Nilai fitness pada probabilitas crossover 0.5 dan probabilitas mutasi yang berbeda

Uji Coba	Probabilitas Mutasi				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
1	7.840554	8.163954	7.713903	7.156166	5.705264
2	8.082857	8.093199	7.80253	6.884808	5.5104

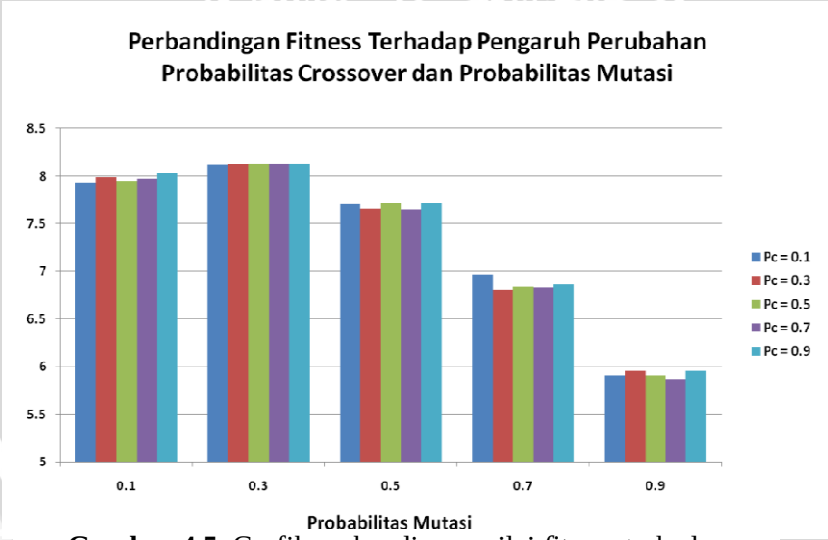
Uji Coba	Probabilitas Mutasi				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
3	8.063401	8.143861	7.803301	7.037487	5.575041
4	7.857614	8.140562	7.572153	6.635997	6.347013
5	8.010245	8.121429	7.789326	6.722264	6.010256
6	8.032833	8.142842	7.836178	7.05875	5.954092
7	7.806047	8.140571	7.733796	6.761137	5.903745
8	7.782506	8.151847	7.78164	6.644151	5.808785
9	7.986604	8.126546	7.550978	6.839819	6.058744
10	8.06549	8.11181	7.656289	6.723368	6.221641
Rata-rata	**7.952815**	**8.133662**	**7.72401**	**6.846395**	**5.909498**

Table 4.4 Nilai fitness pada probabilitas crossover 0.7 dan probabilitas mutasi yang berbeda

Uji Coba	Probabilitas Mutasi				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
1	7.923387	8.11792	7.753887	6.794244	5.703091
2	7.958921	8.145592	7.583166	6.774902	5.723056
3	7.866509	8.158893	7.63519	7.011886	5.805949
4	8.130784	8.124055	7.610582	7.022615	6.075202
5	8.073929	8.161104	7.58636	6.874526	5.844667
6	8.001017	8.152476	7.760551	6.556977	6.031215
7	8.004829	8.135402	7.764874	6.689778	5.908661
8	8.138516	8.082354	7.550792	6.915104	6.150285
9	8.020891	8.161064	7.623931	6.714974	5.630665
10	7.615453	8.157718	7.685508	7.041905	5.784493
Rata-rata	**7.973424**	**8.139658**	**7.655484**	**6.839691**	**5.865728**

Table 4.5 Nilai *fitness* pada probabilitas *crossover* 0.9 dan probabilitas mutasi yang berbeda

Uji Coba	Probabilitas Mutasi				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
1	8.068662	8.163088	7.628426	6.852559	6.121747
2	8.074824	8.10998	7.615149	6.558832	6.04669
3	8.024586	8.141469	7.628615	7.056507	6.458951
4	8.09604	8.155621	7.824585	6.754208	6.117791
5	8.011604	8.108232	7.609388	6.679474	5.966302
6	8.091396	8.133358	7.866144	7.042484	5.507581
7	7.994775	8.15023	7.764359	6.990416	6.161362
8	7.991168	8.139809	7.678945	6.958752	5.901215
9	7.99606	8.137209	7.689262	6.918808	5.685432
10	8.012596	8.149591	7.810329	6.885345	5.633438
Rata-rata	**8.036171**	**8.138859**	**7.71152**	**6.869739**	**5.960051**



Gambar 4.5 Grafik perbandingan nilai *fitness* terhadap perunahan Pc dan Pm

Table 4.6 Hasil perbandingan nilai fitness terhadap perunahan Pc dan Pm

Pc	Pm				
	0.1	0.3	0.5	0.7	0.9
0.1	7.939487	8.118732	7.709014	6.964813	5.915809
0.3	7.984918	8.131945	7.661651	6.817842	5.953844
0.5	7.952815	8.133662	7.72401	6.846395	5.909498
0.7	7.973424	8.139658	7.655484	6.839691	5.865728
0.9	8.036171	8.138859	7.71152	6.869739	5.960051

Pada Gambar 4.5 parameter genetika yang paling mempengaruhi nilai *fitness* adalah probabilitas mutasi, semakin besar nilai probabilitas mutasi maka nilai *fitness* yang dihasilkan semakin menurun. Nilai probabilitas mutasi dibawah 30% menghasilkan nilai *fitness* yang baik, namun nilai probabilitas mutasi diatas 30% menyebabkan penurunan nilai *fitness* yang sangat besar. Hal ini terjadi karena metode mutasi yang digunakan, yaitu metode random akan memunculkan nilai baru untuk setiap gen yang dimutasi. Apabila probabilitas mutasi besar, terlalu banyak perubahan gen sehingga sifat induk yang awalnya sudah baik akan hilang digantikan oleh nilai gen hasil mutasi yang belum tentu menghasilkan nilai *fitness* yang baik.

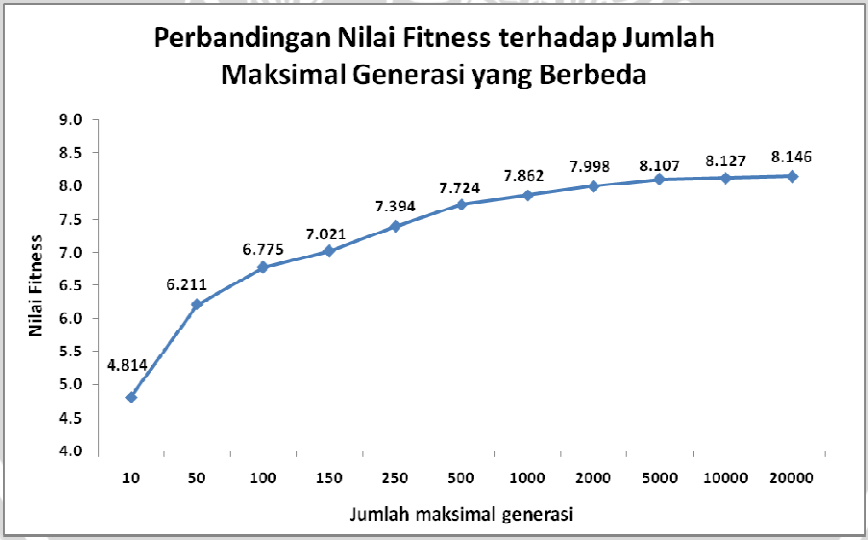
Pada Gambar 4.5 diketahui pula bahwa probabilitas *crossover* tidak menunjukkan perubahan yang signifikan terhadap nilai *fitness*, namun secara keseluruhan nilai probabilitas *crossover* yang tinggi akan menghasilkan nilai *fitness* yang lebih baik yaitu nilai probabilitas *crossover* diatas 50%. Hal ini terjadi karena probabilitas *crossover* yang besar akan menghasilkan banyak anak dengan variasi kromosom induk yang semakin beragam sehingga kemungkinan untuk memperoleh nilai *fitness* yang optimal semakin basar pula.

4.5.2 Pengaruh jumlah generasi terhadap nilai *fitness*

Uji coba kedua akan dilakukan pada jumlah generasi yang berbeda. Uji coba dilakukan sebanyak 10 kali percobaan untuk masing-masing jumlah generasi, kemudian diambil nilai rata-ratanya. Pengujian ini dilakukan dengan jumlah populasi awal = 50, Pc= 0.9 dan Pm = 0.3. Hasil pengujian dapat dilihat pada Tabel 4.7.

Table 4.7 Nilai *fitness* dengan jumlah maksimal generasi yang berbeda

	PERCOBAAN KE-										Rata-Rata
	I	II	III	IV	V	VI	VII	VIII	IX	X	
10	4.703	4.310	4.182	4.699	4.656	4.793	5.207	5.143	5.317	5.129	4.814
50	5.916	6.478	5.889	6.330	6.448	6.067	6.199	6.228	5.646	6.914	6.211
100	7.115	6.531	6.817	6.796	7.000	6.716	6.287	6.929	6.836	6.725	6.775
150	6.937	7.044	7.464	7.010	6.877	7.071	6.928	6.896	6.717	7.270	7.021
250	7.421	7.415	7.389	7.468	7.401	7.375	7.538	7.185	7.354	7.396	7.394
500	7.604	7.725	7.762	7.819	7.601	7.671	7.943	7.768	7.675	7.670	7.724
1000	7.947	7.806	7.859	7.789	7.909	7.989	7.902	7.819	7.776	7.828	7.862
2000	7.975	7.980	8.032	7.927	8.025	8.022	8.018	8.008	7.963	8.025	7.998
5000	8.142	8.092	8.130	8.121	8.082	8.093	8.090	8.108	8.122	8.090	8.107
10000	8.150	8.147	8.113	8.131	8.163	8.096	8.103	8.119	8.106	8.144	8.127
20000	8.165	8.145	8.153	8.156	8.130	8.135	8.134	8.146	8.169	8.131	8.146



Gambar 4.6 Grafik perbandingan nilai fitness terhadap jumlah generasi

Pengujian nilai *fitness* terhadap jumlah maksimal generasi terlihat pada Gambar 4.6. Pada jumlah generasi maksimal 10, nilai *fitness* yang dihasilkan rata-rata sebesar 4.814, sedangkan pada jumlah maksimal generasi 50 nilai *fitness* mengalami kenaikan menjadi 6.211. Kenaikan nilai *fitness* terus berlanjut hingga iterasi maksimal sebanyak 10000. Pada generasi maksimal sebanyak 20000 nilai *fitness* yang dihasilkan tidak jauh berbeda dengan generasi maksimal 10000, sehingga diketahui bahwa batas maksimal *fitness* untuk penelitian ini sekitar 8.1. Dengan demikian dapat disimpulkan bahwa semakin banyak jumlah generasi maksimal, nilai fitness yang dicapai akan semakin tinggi dengan batasan tertentu. Perubahan nilai *fitness* terhadap jumlah generasi terjadi karena semakin beragamnya kromosom yang terbentuk melalui proses rekombinasi sehingga memiliki peluang yang besar untuk mendapatkan nilai *fitness* yang tinggi.

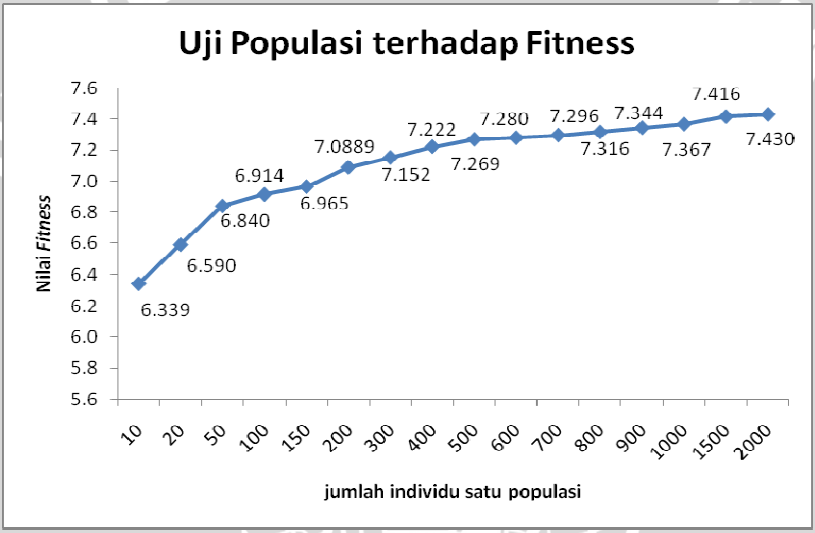
4.5.3 Pengaruh ukuran populasi terhadap nilai *fitness*

Uji coba ketiga akan dilakukan pada ukuran populasi yang berbeda. Uji coba dilakukan sebanyak 10 kali percobaan untuk masing-masing ukuran populasi, kemudian diambil nilai rata-ratanya. Pengujian ini dilakukan dengan jumlah generasi maksimal = 100, Pc= 0.9 dan Pm = 0.3. Hasil pengujian dapat dilihat pada Tabel 4.8.

Table 4.8 Nilai *fitness* dengan ukuran populasi yang berbeda

nPop	Percobaan Ke-										Rata-rata
	I	II	III	IV	V	VI	VII	VIII	IX	X	
10	5.873	6.092	6.594	6.161	6.038	5.942	6.829	6.681	6.232	6.950	6.339
20	6.472	6.279	6.539	6.693	7.182	6.680	6.557	6.767	6.222	6.513	6.590
50	6.764	6.850	7.038	6.810	6.718	6.993	7.424	6.370	6.895	6.534	6.840
100	7.112	6.989	6.604	6.986	6.858	7.228	6.739	7.005	6.977	6.641	6.914
150	6.888	6.817	6.953	6.835	7.075	7.132	7.035	6.918	6.872	7.123	6.965
200	7.135	7.189	7.214	6.939	7.147	6.939	7.018	7.060	7.142	7.107	7.0889
300	7.245	7.105	7.087	7.208	7.191	7.170	7.017	7.336	7.016	7.140	7.152
400	7.363	7.333	7.229	7.135	7.234	7.191	7.306	7.099	7.163	7.167	7.222
500	7.254	7.327	7.196	7.277	7.364	7.118	7.069	7.454	7.264	7.371	7.269
600	7.308	7.169	7.326	7.344	7.289	7.299	7.275	7.346	7.254	7.186	7.280

nPop	Percobaan Ke-										Rata-rata
	I	II	III	IV	V	VI	VII	VIII	IX	X	
700	7.342	7.277	7.457	7.351	7.263	7.213	7.376	7.137	7.307	7.239	7.296
800	7.319	7.352	7.452	7.331	7.314	7.435	7.331	7.212	7.174	7.242	7.316
900	7.269	7.200	7.440	7.288	7.372	7.314	7.431	7.383	7.343	7.397	7.344
1000	7.276	7.487	7.408	7.299	7.335	7.386	7.283	7.311	7.313	7.571	7.367
1500	7.363	7.585	7.431	7.353	7.348	7.549	7.450	7.377	7.424	7.276	7.416
2000	7.404	7.401	7.452	7.452	7.320	7.329	7.457	7.668	7.374	7.439	7.430



Gambar 4.7 Grafik perbandingan nilai *fitness* terhadap ukuran populasi

Pengujian nilai *fitness* terhadap ukuran populasi terlihat pada Gambar 4.7. Pada ukuran populasi 10 nilai *fitness* yang dihasilkan sebesar 6.339, selanjutnya mengalami kenaikan seiring ditambahnya ukuran populasi. Pada ukuran populasi 200 sampai 1000 perubahan nilai fitness yang dihasilkan tidak terlalu besar. Pada ukuran populasi 1500 dan 2000 nilai *fitness* yang dihasilkan juga tidak jauh berbeda yaitu sekitar 7.4 dan merupakan batas teringgi nilai fitness yang dicapai. Maka dapat disimpulkan bahwa semakin besar ukuran populasi maka semakin besar pula nilai *fitness* yang dihasilkan.

Seperti pada pengujian jumlah generasi, hal ini terjadi karena semakin beragamnya kromosom yang terbentuk pada saat pembentukan populasi awal secara random dan proses rekombinasi sehingga memiliki peluang yang besar untuk mendapatkan nilai *fitness* yang tinggi.

4.5.4 Perbandingan harga saham aktual dengan *output* jaringan

Data yang akan diramalkan adalah nilai tukar Poundsterling Inggris (GBP) terhadap Dollar Amerika Serikat (USD) selama 2 bulan yaitu pada januari 2011 – Februari 2011. Peramalan harga valas ini menggunakan parameter genetik yaitu ukuran populasi = 50, jumlah maksimal generasi = 10000, Pc = 0.9 dan Pm = 0.3. Kromosom yang digunakan sebagai bobot jaringan merupakan hasil dari proses pelatihan dan memiliki nilai *fitness* sebesar 8.1631 sehingga bobot jaringan yang digunakan diambil dari kromosom tersebut. Untuk lebih jelas dapat dilihat pada Tabel 4.9.

Table 4.9 Kromosom hasil pelatihan jaringan

Kromosom	**0.0272**	**0.0014**	**0.0127**	**0.0008**	**0.0011**	**0.0017**	**0.6800**	**0.1897**
	V01	**V02**	**V03**	**V11**	**V12**	**V13**	**V21**	**V22**
	0.0348	**0.0930**	**0.5082**	**0.6541**	**0.9979**	**0.9964**	**0.9993**	**0.0016**
	V23	**V31**	**V32**	**V33**	**W1**	**W2**	**W3**	**W0**

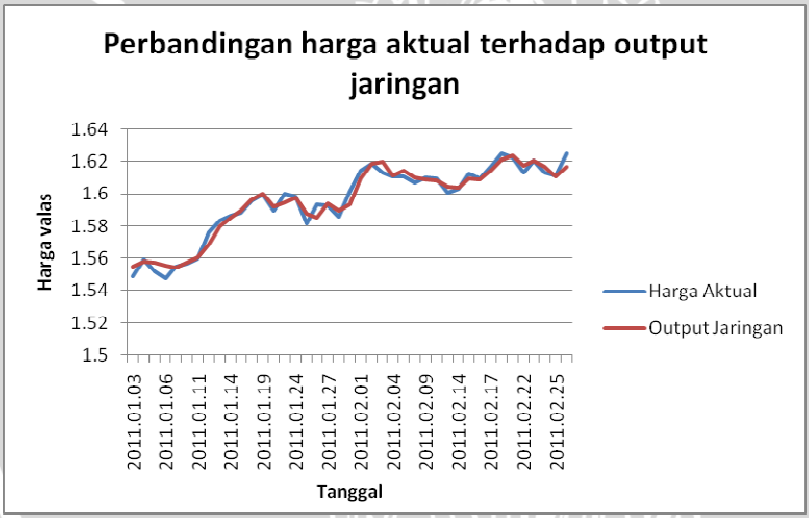
Kromosom pada Tabel 4.9 digunakan untuk menentukan *output* dari data uji yang digunakan, atau dengan kata lain bobot tersebut digunakan untuk proses peramalan. Perbandingan nilai tukar aktual dan *output* jaringan dapat dilihat pada Tabel 4.10 dan Gambar 4.8.

Table 4.10 Perbandingan harga aktual dengan *output* jaringan

Tanggal	Harga Aktual	*Output* Jaringan	Error
2011.01.03	1.5482	1.5549	-0.0067
2011.01.04	1.5588	1.5583	0.0005
2011.01.05	1.5514	1.5576	-0.0062

Tanggal	Harga Aktual	*Output* Jaringan	Error
2011.01.06	1.5472	1.5556	-0.0084
2011.01.07	1.5545	1.5544	0.0001
2011.01.10	1.5567	1.5576	-0.0009
2011.01.11	1.5595	1.5604	-0.0009
2011.01.12	1.5761	1.5687	0.0074
2011.01.13	1.5835	1.5800	0.0035
2011.01.14	1.5866	1.5854	0.0012
2011.01.17	1.5883	1.5898	-0.0015
2011.01.18	1.5956	1.5967	-0.0011
2011.01.19	1.5995	1.6003	-0.0008
2011.01.20	1.5895	1.5925	-0.003
2011.01.21	1.5999	1.5947	0.0052
2011.01.24	1.5985	1.5978	0.0007
2011.01.25	1.5815	1.5878	-0,0063
2011.01.26	1.5933	1.5855	0.0078
2011.01.27	1.5927	1.5943	-0.0016
2011.01.28	1.5861	1.5900	-0.0039
2011.01.31	1.6014	1.5933	0.0081
2011.02.01	1.6143	1.6096	0.0047
2011.02.02	1.619	1.6188	0.0002
2011.02.03	1.6135	1.6201	-0.0066
2011.02.04	1.6109	1.6113	-0.0004
2011.02.07	1.6108	1.6148	-0.004
2011.02.08	1.6068	1.6104	-0.0036
2011.02.09	1.6102	1.6092	0.001
2011.02.10	1.6096	1.6085	0.0011
2011.02.11	1.6006	1.6046	-0.004
2011.02.14	1.6034	1.6039	-0.0005
2011.02.15	1.6127	1.6097	0.003

Tanggal	Harga Aktual	*Output* Jaringan	Error
2011.02.16	1.6094	1.6091	0.0003
2011.02.17	1.6173	1.6143	0.003
2011.02.18	1.6251	1.6212	0.0039
2011.02.21	1.6225	1.6238	-0.0013
2011.02.22	1.6135	1.6175	-0.004
2011.02.23	1.6209	1.6205	0.0004
2011.02.24	1.6134	1.6171	-0.0037
2011.02.25	1.6116	1.6105	0.0011
2011.02.28	1.6251	1.6169	0.0082



Gambar 4.8 Grafik Perbandingan harga valas aktual terhadap output jaringan

Berdasarkan Gambar 4.8 terlihat bahwa *output* jaringan mampu mengikuti pola dari harga valas aktual. Pada bagian awal yaitu mulai tanggal 07/01/2011 hingga tanggal 19/01/2011 harga nilai tukar terus naik dan *output* aktual berhasil mengikuti harga aktual dengan selisih terkecil 0.0001 terdapat pada tanggal 07/01/2011, sedangkan selisih paling tinggi adalah pada tanggal

12/01/2011 sebesar 0.0074. Pada tanggal setelahnya harga mengalami penurunan dan kenaikan berulang-ulang mulai tanggal 20/01/2011 hingga tanggal 31/01/2011. Selisih terbesar adalah pada tanggal 25/01/2011 sebesar -0.0063 dan pada tanggal 26/01/2011 sebesar 0.0078, sedangkan selisih terkecil pada tanggal 24/01/2011 dengan nilai aktual 1.5985 dan nilai *output* jaringan 1.5978 sehingga kesalahan yang terjadi hanya 0.0007. Bedasarkan Gambar 4.8 dapat diketahui pula bahwa harga valas selalu mengalami penurunan setelah terjadi kenaikan yang cukup tinggi, namun secara keseluruhan harga valas mengalami peningkatan selama bulan Januari dan Februari.

Hasil peramalan atau *output* yang dikeluarkan oleh sistem setelah dicocokkan dengan data aktual akan dievaluasi tingkat kesalahannya untuk mengetahui tingkat akurasi peramalan. Rumus yang digunakan untuk evaluasi kesalahan adalah RMSE dan MAPE sesuai dengan penjelasan pada bab sebelumnya. Berdasarkan Tabel 4.8 dihasilkan nilai RMSE sebesar 0.00412 dan nilai MAPE sebesar 0.0020. Agar hasil pengujian lebih valid, maka dilakukan percobaan sebanyak 10 kali kemudian diambil nilai rata-ratanya. Hasil percobaan dapat dilihat pada Tabel 4.11.

Table 4.11 Nilai RMSE dan MAPE pada 10 percobaan

Uji Coba ke -	Evaluasi Kesalahan	
	RMSE	MAPE
1	0.004122367	0.002003215
2	0.004246015	0.002030639
3	0.004174882	0.002016645
4	0.004151801	0.002013174
5	0.004264418	0.002042592
6	0.004191393	0.002019829
7	0.004126161	0.002004746
8	0.004174111	0.002019552
9	0.004181899	0.002018936
10	0.004146113	0.002008593
Rata-rata	0.004177916	0.002017792

Berdasarkan Tabel 4.11 diketahui bahwa nilai RMSE dan MAPE untuk sepuluh kali percobaan menunjukkan nilai yang cukup stabil, yaitu dengan nilai rata-rata RMSE sebesar 0.00418 dan nilai rata-rata MAPE sebesar 0.0020. Nilai RMSE yang dihasilkan dapat dikatakan sangat kecil sehingga tingkat kebenaran peramalan pun dapat dikatakan sangat baik. Begitu juga dengan nilai MAPE yang dihasilkan dapat dikatakan sangat kecil. Pada beberapa penelitian nilai MAPE ditunjukkan dalam bentuk persentase (%), jadi apabila nilai MAPE senilai 0,002 artinya kesalahan yang terjadi sebanyak 0.2% terhadap data sebenarnya. Berdasarkan tingkat kesalahan yang diperoleh menunjukkan bahwa sistem mampu meramalkan harga valuta asing dengan baik, dengan tingkat akurasi 99.8%.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah melakukan penelitian maka dapat disimpulkan bahwa :

1. Algoritma genetika dapat diterapkan pada jaringan syaraf tiruan dengan baik, untuk meramalkan harga valuta asing. Algoritma genetika digunakan untuk mendapatkan bobot jaringan syaraf tiruan yang paling optimal. Model jaringan yang digunakan adalah 3-3-1, artinya terdiri dari 3 *neuron input*, 3 *neuron* tersembunyi dan 1 *neuron output*. Pengkodean kromosom berupa susunan bobot pada jaringan syaraf tiruan yang terdiri dari bobot V (bobot yang menghubungkan *neuron input* dan *neuron* tersembunyi) dan bobot W (bobot yang menghubungkan *neuron* tersembunyi dan *neuron output*)serta bobot bias, sebanyak 16 gen. Setelah bobot jaringan syaraf didapat, bobot digunakan terhadap data uji untuk menghitung peramalan harga valuta asing.
2. Tingkat kesalahan yang dihasilkan oleh jaringan syaraf tiruan dengan pelatihan menggunakan algoritma genetika dapat dikatakan sangat kecil. Evaluasi kesalahan menggunakan RMSE dan MAPE. Nilai RMSE yang dihasilkan sebesar 0.00418 sedangkan nilai MAPE yang dihasilkan sebesar 0.0020 atau dalam persentase kesalahan sebesar 0.2%. Hal ini menunjukkan bahwa tingkat akurasi yang dihasilkan sistem sangat baik karena sangat mendekati nilai sebenarnya.
3. Parameter-parameter genetika sangat bepengaruh pada pelatihan bobot jaringan syaraf tiruan. Parameter ukuran populasi dan jumlah generasi maksimal menunjukkan bahwa semakin banyak jumlah individu dalam satu populasi dan semakin banyak jumlah generasi maka nilai *fitness* yang dihasilkan semakin baik. Untuk probabilitas *crossover*, pada penelitian ini kurang memperngaruhi perubahan nilai *fitness* dibandingkan dengan probabilitas mutasi. Pengaruh probabilitas mutasi terhadap nilai *fitness* sangat jelas terlihat. Untuk probabilitas mutasi > 50% maka nilai *fitness*nya akan menurun. Parameter genetika yang menghasilkan nilai

kesalahan peramalan terkecil yaitu dengan menggunakan ukuran populasi sebanyak 50 kromosom, jumlah generasi maksimal 10000, probabilitas *crossover* 0.9 dan probabilitas mutasi 0.3.

5.2 Saran

Untuk pengembangan lanjut penelitian ini maka ada beberapa saran yang dapat diberikan :

1. Pada penelitian ini pengkodean kromosom berupa susunan bobot JST untuk peramalan, maka untuk penelitian selanjutnya diharapkan dapat menggunakan algoritma genetika untuk mengoptimalkan komponen jaringan syaraf tiruan yang berbeda.
2. Pada penelitian ini data yang digunakan adalah data harga valas harian, untuk penelitian selanjutnya dapat menggunakan interval data yang lain.
3. Banyaknya iterasi yang dibutuhkan untuk mendapatkan nilai *fitness* terbaik pada pelatihan ini sangatlah besar yaitu 10.000 iterasi. Untuk penelitian selanjutnya dapat digunakan metode lain pada pengkodean kromosom, proses mutasi atau *crossover* sehingga dapat mencapai *fitness* tertinggi tanpa melakukan banyak iterasi.

LAMPIRAN

Data latih harga valuta asing USD terhadap GBP

No.	Date	Open	High	Low	Close
1	2010.01.04	1.6111	1.6241	1.6057	1.6086
2	2010.01.05	1.6084	1.6153	1.5965	1.5993
3	2010.01.06	1.5994	1.6066	1.594	1.6021
4	2010.01.07	1.602	1.6061	1.5899	1.5938
5	2010.01.08	1.5937	1.611	1.5912	1.6023
6	2010.01.11	1.6066	1.6193	1.6047	1.6112
7	2010.01.12	1.6114	1.6194	1.6062	1.6165
8	2010.01.13	1.6164	1.6306	1.6135	1.6279
9	2010.01.14	1.6278	1.6347	1.6249	1.6333
10	2010.01.15	1.6335	1.6354	1.621	1.6263
11	2010.01.18	1.626	1.6378	1.6249	1.6341
12	2010.01.19	1.6343	1.6457	1.6311	1.636
13	2010.01.20	1.6361	1.6371	1.6243	1.629
14	2010.01.21	1.6289	1.631	1.6125	1.6191
15	2010.01.22	1.6192	1.6277	1.607	1.6106
16	2010.01.25	1.6099	1.6254	1.6083	1.6236
17	2010.01.26	1.6237	1.6263	1.6087	1.6136
18	2010.01.27	1.614	1.6238	1.61	1.6162
19	2010.01.28	1.6165	1.6269	1.6117	1.6139
20	2010.01.29	1.614	1.6173	1.5973	1.5974
21	2010.02.01	1.5943	1.5972	1.5844	1.595
22	2010.02.02	1.5953	1.5989	1.5896	1.5966
23	2010.02.03	1.597	1.6063	1.587	1.5886
24	2010.02.04	1.5889	1.5914	1.5727	1.5749
25	2010.02.05	1.5751	1.5772	1.5552	1.5634
26	2010.02.08	1.5586	1.5653	1.5528	1.5577

27	2010.02.09	1.558	1.5742	1.5555	1.5715
28	2010.02.10	1.5714	1.5758	1.5565	1.5589
29	2010.02.11	1.5588	1.5722	1.5553	1.5709
30	2010.02.12	1.5706	1.5746	1.5586	1.5707
31	2010.02.15	1.5683	1.5726	1.5618	1.5664
32	2010.02.16	1.5665	1.58	1.5633	1.5795
33	2010.02.17	1.5797	1.5821	1.567	1.5676
34	2010.02.18	1.5675	1.5693	1.55	1.5531
35	2010.02.19	1.5532	1.5542	1.5341	1.5465
36	2010.02.22	1.5474	1.5514	1.5426	1.5474
37	2010.02.23	1.5478	1.5569	1.5388	1.5414
38	2010.02.24	1.5418	1.547	1.5379	1.5401
39	2010.02.25	1.54	1.5417	1.5186	1.5261
40	2010.02.26	1.526	1.5311	1.5146	1.5229
41	2010.03.01	1.5163	1.5197	1.4776	1.4982
42	2010.03.02	1.4983	1.4995	1.4847	1.4965
43	2010.03.03	1.4969	1.5126	1.4951	1.5091
44	2010.03.04	1.5095	1.513	1.4999	1.5025
45	2010.03.05	1.5028	1.516	1.4986	1.513
46	2010.03.08	1.5129	1.5199	1.5034	1.5067
47	2010.03.09	1.5068	1.5071	1.4939	1.4999
48	2010.03.10	1.5	1.5019	1.4876	1.4975
49	2010.03.11	1.4976	1.5064	1.4943	1.5059
50	2010.03.12	1.5058	1.5215	1.5023	1.5202
51	2010.03.15	1.5171	1.5201	1.5012	1.5049
52	2010.03.16	1.5053	1.5254	1.497	1.5235
53	2010.03.17	1.5236	1.5375	1.5202	1.5319
54	2010.03.18	1.532	1.5322	1.521	1.5237
55	2010.03.19	1.5241	1.5252	1.4982	1.5006
56	2010.03.22	1.4998	1.5101	1.4926	1.5096
57	2010.03.23	1.5095	1.5109	1.4966	1.5039

58	2010.03.24	1.504	1.5043	1.4856	1.486
59	2010.03.25	1.4859	1.4999	1.4791	1.4805
60	2010.03.26	1.4808	1.4917	1.4798	1.489
61	2010.03.29	1.4917	1.5012	1.4886	1.498
62	2010.03.30	1.4983	1.512	1.4967	1.5065
63	2010.03.31	1.5066	1.5198	1.5037	1.5174
64	2010.04.01	1.5176	1.5292	1.5166	1.5284
65	2010.04.02	1.5289	1.5293	1.5175	1.52
66	2010.04.05	1.528	1.5313	1.5217	1.5289
67	2010.04.06	1.5288	1.5301	1.5122	1.5262
68	2010.04.07	1.526	1.528	1.5132	1.5233
69	2010.04.08	1.5231	1.5276	1.5135	1.5273
70	2010.04.09	1.5272	1.5391	1.5261	1.5371
71	2010.04.12	1.5457	1.5485	1.5352	1.5367
72	2010.04.13	1.5368	1.5449	1.5336	1.5379
73	2010.04.14	1.5374	1.549	1.5371	1.5465
74	2010.04.15	1.5466	1.5522	1.5385	1.5494
75	2010.04.16	1.5495	1.5505	1.5356	1.5356
76	2010.04.19	1.5274	1.534	1.5191	1.5337
77	2010.04.20	1.5336	1.5433	1.5289	1.5357
78	2010.04.21	1.5355	1.5439	1.5331	1.5409
79	2010.04.22	1.5407	1.5472	1.5342	1.5381
80	2010.04.23	1.5382	1.5403	1.5301	1.5382
81	2010.04.26	1.5394	1.5501	1.5377	1.5462
82	2010.04.27	1.5461	1.5489	1.5238	1.5261
83	2010.04.28	1.5263	1.5284	1.5121	1.5205
84	2010.04.29	1.5204	1.5341	1.514	1.5315
85	2010.04.30	1.5319	1.539	1.5258	1.5279
86	2010.05.03	1.5338	1.534	1.521	1.5245
87	2010.05.04	1.5244	1.5268	1.5094	1.5146
88	2010.05.05	1.5143	1.5166	1.5061	1.5095

89	2010.05.06	1.5094	1.5141	1.4704	1.4821
90	2010.05.07	1.482	1.4938	1.4479	1.4808
91	2010.05.10	1.4883	1.5058	1.4765	1.4846
92	2010.05.11	1.4848	1.5008	1.4725	1.4958
93	2010.05.12	1.4957	1.5047	1.4822	1.4826
94	2010.05.13	1.4824	1.492	1.4597	1.4616
95	2010.05.14	1.4615	1.4642	1.4499	1.4543
96	2010.05.17	1.4508	1.4544	1.4246	1.447
97	2010.05.18	1.4468	1.4514	1.4301	1.4329
98	2010.05.19	1.4331	1.4445	1.4233	1.444
99	2010.05.20	1.4441	1.4461	1.4224	1.4359
100	2010.05.21	1.4354	1.45	1.4313	1.4462
101	2010.05.24	1.4463	1.453	1.4354	1.4425
102	2010.05.25	1.4426	1.4427	1.426	1.4409
103	2010.05.26	1.4408	1.4449	1.4331	1.4384
104	2010.05.27	1.4386	1.4606	1.4366	1.4581
105	2010.05.28	1.458	1.4611	1.4434	1.4457
106	2010.05.31	1.4417	1.4549	1.4404	1.4538
107	2010.06.01	1.4536	1.4718	1.4438	1.4651
108	2010.06.02	1.4648	1.4768	1.4553	1.465
109	2010.06.03	1.4651	1.4742	1.4585	1.4612
110	2010.06.04	1.4607	1.4681	1.4451	1.4452
111	2010.06.07	1.4408	1.4561	1.4386	1.4464
112	2010.06.08	1.4465	1.4527	1.4345	1.4467
113	2010.06.09	1.4465	1.4606	1.4394	1.4525
114	2010.06.10	1.4524	1.4715	1.4507	1.4711
115	2010.06.11	1.4712	1.4757	1.4502	1.4549
116	2010.06.14	1.457	1.4807	1.455	1.4742
117	2010.06.15	1.4743	1.4837	1.4683	1.4805
118	2010.06.16	1.4804	1.4856	1.4726	1.4731
119	2010.06.17	1.473	1.4838	1.4645	1.4823

120	2010.06.18	1.4824	1.4886	1.4771	1.4826
121	2010.06.21	1.4857	1.4936	1.4738	1.4755
122	2010.06.22	1.4756	1.4857	1.4686	1.4814
123	2010.06.23	1.4815	1.4976	1.4805	1.4958
124	2010.06.24	1.4957	1.5014	1.4918	1.4938
125	2010.06.25	1.4937	1.508	1.4859	1.5063
126	2010.06.28	1.5068	1.5128	1.5016	1.5102
127	2010.06.29	1.5104	1.5119	1.5012	1.5065
128	2010.06.30	1.506	1.5073	1.4931	1.4941
129	2010.07.01	1.4943	1.5189	1.4873	1.5175
130	2010.07.02	1.5174	1.5228	1.5147	1.5194
131	2010.07.05	1.5203	1.5205	1.5089	1.5135
132	2010.07.06	1.5134	1.5226	1.5081	1.5148
133	2010.07.07	1.5149	1.5219	1.508	1.5186
134	2010.07.08	1.5187	1.524	1.5102	1.5166
135	2010.07.09	1.5165	1.5204	1.505	1.5064
136	2010.07.12	1.5079	1.5085	1.4948	1.5031
137	2010.07.13	1.503	1.5191	1.4964	1.5175
138	2010.07.14	1.5176	1.5297	1.5171	1.5266
139	2010.07.15	1.5265	1.5471	1.5235	1.546
140	2010.07.16	1.5461	1.5463	1.5278	1.53
141	2010.07.19	1.5276	1.5351	1.5202	1.5228
142	2010.07.20	1.5227	1.5309	1.5152	1.5263
143	2010.07.21	1.5261	1.5335	1.5123	1.5161
144	2010.07.22	1.5162	1.5295	1.515	1.5258
145	2010.07.23	1.5259	1.5448	1.5251	1.542
146	2010.07.26	1.5418	1.5519	1.5409	1.549
147	2010.07.27	1.5489	1.5598	1.5442	1.5592
148	2010.07.28	1.5594	1.5637	1.5546	1.5596
149	2010.07.29	1.5597	1.5661	1.5579	1.5611
150	2010.07.30	1.5612	1.5721	1.5552	1.5695

151	2010.08.02	1.5711	1.5909	1.5698	1.5889
152	2010.08.03	1.5888	1.597	1.5864	1.5954
153	2010.08.04	1.5953	1.5966	1.5858	1.5886
154	2010.08.05	1.5887	1.5927	1.5822	1.5898
155	2010.08.06	1.59	1.6	1.5842	1.5943
156	2010.08.09	1.5967	1.5998	1.5893	1.5893
157	2010.08.10	1.5894	1.5911	1.5712	1.5854
158	2010.08.11	1.5853	1.5865	1.5631	1.566
159	2010.08.12	1.5659	1.5715	1.5564	1.558
160	2010.08.13	1.5579	1.5681	1.5568	1.5594
161	2010.08.16	1.5611	1.5705	1.5537	1.5665
162	2010.08.17	1.5664	1.5698	1.5554	1.5588
163	2010.08.18	1.559	1.569	1.55	1.5595
164	2010.08.19	1.5596	1.5674	1.5511	1.5602
165	2010.08.20	1.5604	1.5605	1.5466	1.5539
166	2010.08.23	1.5538	1.5621	1.5502	1.5514
167	2010.08.24	1.5515	1.5517	1.5375	1.539
168	2010.08.25	1.5392	1.5469	1.5386	1.5457
169	2010.08.26	1.5456	1.5597	1.5456	1.5528
170	2010.08.27	1.5527	1.5544	1.5441	1.5528
171	2010.08.30	1.5529	1.5575	1.5455	1.5461
172	2010.08.31	1.546	1.5474	1.5326	1.5347
173	2010.09.01	1.5348	1.5491	1.5336	1.5453
174	2010.09.02	1.5454	1.5459	1.535	1.5401
175	2010.09.03	1.54	1.5467	1.5388	1.5451
176	2010.09.06	1.5429	1.5488	1.5345	1.5391
177	2010.09.07	1.5392	1.5424	1.5296	1.5355
178	2010.09.08	1.5356	1.5532	1.5337	1.5468
179	2010.09.09	1.547	1.5477	1.5375	1.5428
180	2010.09.10	1.5429	1.5466	1.5345	1.5356
181	2010.09.13	1.5358	1.5487	1.5354	1.5428

182	2010.09.14	1.5426	1.5591	1.5347	1.5537
183	2010.09.15	1.5538	1.5651	1.5448	1.5621
184	2010.09.16	1.5622	1.5648	1.5538	1.5623
185	2010.09.17	1.5624	1.5728	1.5595	1.563
186	2010.09.20	1.5651	1.5685	1.5527	1.5544
187	2010.09.21	1.5543	1.5643	1.5502	1.5621
188	2010.09.22	1.562	1.5715	1.5601	1.5663
189	2010.09.23	1.5662	1.5741	1.561	1.5682
190	2010.09.24	1.5683	1.5842	1.5641	1.5819
191	2010.09.27	1.582	1.5866	1.5786	1.5826
192	2010.09.28	1.5827	1.5894	1.5718	1.58
193	2010.09.29	1.5801	1.5874	1.5761	1.5786
194	2010.09.30	1.579	1.5922	1.5669	1.5717
195	2010.10.01	1.5719	1.5872	1.5703	1.5819
196	2010.10.04	1.5799	1.5869	1.5751	1.5827
197	2010.10.05	1.5828	1.5933	1.5751	1.5889
198	2010.10.06	1.5888	1.5938	1.5832	1.5891
199	2010.10.07	1.5889	1.6019	1.5826	1.5876
200	2010.10.08	1.5875	1.5965	1.5822	1.596
201	2010.10.11	1.5951	1.5962	1.587	1.5884
202	2010.10.12	1.5883	1.592	1.5757	1.581
203	2010.10.13	1.5811	1.5908	1.5776	1.5896
204	2010.10.14	1.5895	1.6066	1.5888	1.6011
205	2010.10.15	1.601	1.6106	1.597	1.599
206	2010.10.18	1.5977	1.6003	1.5836	1.5876
207	2010.10.19	1.5875	1.5946	1.5679	1.5705
208	2010.10.20	1.5706	1.5877	1.5653	1.5846
209	2010.10.21	1.5847	1.5848	1.5685	1.5703
210	2010.10.22	1.5704	1.5749	1.5651	1.5682
211	2010.10.25	1.568	1.5772	1.5663	1.5721
212	2010.10.26	1.572	1.5895	1.569	1.5844

213	2010.10.27	1.5842	1.5865	1.5728	1.5765
214	2010.10.28	1.5766	1.5977	1.5759	1.5946
215	2010.10.29	1.5944	1.6044	1.5875	1.604
216	2010.11.01	1.6048	1.6091	1.5993	1.6037
217	2010.11.02	1.6038	1.6082	1.5963	1.6043
218	2010.11.03	1.6042	1.6161	1.6009	1.6081
219	2010.11.04	1.6082	1.6299	1.6078	1.6263
220	2010.11.05	1.6266	1.6294	1.6165	1.6182
221	2010.11.08	1.6196	1.6213	1.6103	1.6139
222	2010.11.09	1.6137	1.6183	1.595	1.5983
223	2010.11.10	1.5981	1.6135	1.596	1.612
224	2010.11.11	1.6119	1.6178	1.6081	1.6121
225	2010.11.12	1.6122	1.6183	1.5985	1.6114
226	2010.11.15	1.6147	1.6153	1.6041	1.6051
227	2010.11.16	1.6052	1.6086	1.5839	1.5884
228	2010.11.17	1.5885	1.5949	1.5853	1.5906
229	2010.11.18	1.5905	1.6055	1.5887	1.604
230	2010.11.19	1.6039	1.6094	1.5936	1.5978
231	2010.11.22	1.5979	1.6083	1.5897	1.5954
232	2010.11.23	1.5955	1.5964	1.5757	1.5776
233	2010.11.24	1.5775	1.5837	1.574	1.5769
234	2010.11.25	1.5771	1.5793	1.5725	1.5759
235	2010.11.26	1.576	1.5771	1.5574	1.5589
236	2010.11.29	1.5593	1.5646	1.5527	1.5573
237	2010.11.30	1.5572	1.5595	1.5483	1.5561
238	2010.12.01	1.5563	1.5647	1.5546	1.562
239	2010.12.02	1.5622	1.5667	1.5511	1.5599
240	2010.12.03	1.56	1.5787	1.558	1.5774
241	2010.12.06	1.576	1.5774	1.5655	1.5713
242	2010.12.07	1.5714	1.5822	1.5702	1.5757
243	2010.12.08	1.5756	1.5836	1.5668	1.5803

244	2010.12.09	1.5802	1.584	1.571	1.5768
245	2010.12.10	1.5769	1.5862	1.5747	1.58
246	2010.12.13	1.5799	1.5897	1.5718	1.5862
247	2010.12.14	1.586	1.591	1.5738	1.5773
248	2010.12.15	1.5774	1.5774	1.553	1.5543
249	2010.12.16	1.5544	1.5636	1.554	1.5632
250	2010.12.17	1.5631	1.5646	1.5454	1.5532
251	2010.12.20	1.5515	1.5576	1.5475	1.551
252	2010.12.21	1.5511	1.5567	1.5435	1.5468
253	2010.12.22	1.5469	1.5493	1.5355	1.5383
254	2010.12.23	1.5385	1.5437	1.5373	1.5427
255	2010.12.24	1.5426	1.5475	1.5415	1.5445
256	2010.12.27	1.5441	1.5456	1.5369	1.542
257	2010.12.28	1.5419	1.5511	1.5344	1.5367
258	2010.12.29	1.5365	1.5517	1.5349	1.5498
259	2010.12.30	1.5499	1.5535	1.5366	1.5424
260	2010.12.31	1.5425	1.5664	1.5419	1.5586

Data uji harga valuta asing USD terhadap GBP

No.	Date	Open	High	Low	Close
1	2011.01.03	1.5556	1.5562	1.5431	1.5482
2	2011.01.04	1.5484	1.5649	1.5449	1.5588
3	2011.01.05	1.5589	1.5627	1.5449	1.5514
4	2011.01.06	1.5513	1.5564	1.5448	1.5472
5	2011.01.07	1.5467	1.5578	1.5404	1.5545
6	2011.01.10	1.5522	1.5599	1.5471	1.5567
7	2011.01.11	1.5568	1.5636	1.5509	1.5595
8	2011.01.12	1.5594	1.5777	1.5579	1.5761
9	2011.01.13	1.576	1.588	1.5715	1.5835
10	2011.01.14	1.5834	1.5885	1.5805	1.5866

11	2011.01.17	1.5862	1.5951	1.5831	1.5883
12	2011.01.18	1.5884	1.6056	1.5871	1.5956
13	2011.01.19	1.5957	1.6037	1.5944	1.5995
14	2011.01.20	1.5993	1.6009	1.5834	1.5895
15	2011.01.21	1.5896	1.6017	1.5866	1.5999
16	2011.01.24	1.6001	1.6013	1.5919	1.5985
17	2011.01.25	1.5987	1.6016	1.575	1.5815
18	2011.01.26	1.5814	1.5936	1.5769	1.5933
19	2011.01.27	1.5934	1.599	1.5878	1.5927
20	2011.01.28	1.5926	1.5965	1.5824	1.5861
21	2011.01.31	1.585	1.6048	1.582	1.6014
22	2011.02.01	1.6015	1.6165	1.6011	1.6143
23	2011.02.02	1.6142	1.623	1.6129	1.619
24	2011.02.03	1.6191	1.6277	1.6121	1.6135
25	2011.02.04	1.6134	1.6171	1.6036	1.6109
26	2011.02.07	1.6098	1.6184	1.6088	1.6108
27	2011.02.08	1.6109	1.6162	1.6027	1.6068
28	2011.02.09	1.6067	1.6124	1.6032	1.6102
29	2011.02.10	1.61	1.6137	1.6011	1.6096
30	2011.02.11	1.6097	1.6112	1.5962	1.6006
31	2011.02.14	1.5986	1.6074	1.5978	1.6034
32	2011.02.15	1.6033	1.6172	1.6008	1.6127
33	2011.02.16	1.6126	1.6187	1.5988	1.6094
34	2011.02.17	1.6095	1.6186	1.6077	1.6173
35	2011.02.18	1.6172	1.6262	1.6151	1.6251
36	2011.02.21	1.625	1.6258	1.6203	1.6225
37	2011.02.22	1.6224	1.6234	1.6102	1.6135
38	2011.02.23	1.6136	1.6274	1.6132	1.6209
39	2011.02.24	1.6208	1.6254	1.6083	1.6134
40	2011.02.25	1.6135	1.616	1.603	1.6116
41	2011.02.28	1.6112	1.6272	1.6067	1.6251

DAFTAR PUSTAKA

Berlianta, Heli Charisma. 2005. *Mengenal Valuta Asing.* Gajah Mada Universitu Press.Yogyakarta.

Darmawan, M. Daud. 2007. *Mengenal BIsnis Valuta Asing.* Penerbit Pinus. Yogyakarta.

Desiani, Anita dan Muhammad Arhani. 2006. *Konsep Kecerdasan Buatan*. Penerbit Andi. Yogyakarta.

Gen,M and R. Cheng. 2000. *Genetic Algorithms and Engineering Optimization*. John Wiley & Sons. New York.)

Hajar, Ibnu. 2009. Mengenal Forex Lebih Dekat. http://www.seputarforex.com/belajar/forex/ diakses pada tanggal 11 Juni 2011.

Haupt R.L., Haupt S.E.2004. *Practical Genetic Algorithms*. John Wiley & Sons, Inc., Hoboken, New Jersey.

Hermawan, A. 2006. *Jaringan Syaraf Tiruan: Teori dan Aplikasi.* Andi. Yogyakarta.

Jingtao Yao *et al. Foreign Exchange Rate Forecasting with Neural Networks*. Department of Information System and Computer Science, National University of Singapore.

Kamruzzaman, Joarder and Ruhul A. Sarker. 2004. *ANN-Based Forecasting of Foreign Currency Exchange Rates.* Neural Information Processing – Letters and Reviews Vol.3 No.2

Khan, Asif Ullah, *et al*. 2008. *Genetic Algorithm Based Backpropagation Neural Network Performs better than Backpropagation Neural Network in Stock Rates Prediction*. IJCSNS International Journal of Computer Science and Network Security VOL.8 No.7

Kristanto, Andri. 2004. *Jaringan Syaraf Tiruan (Konsep Dasar, Algoritma dan Aplikasi)*. Gaya Media. Yogyakarta.

Kurniawan, Bobi. 2011. *Enterprise Architecture Planning Sistem Informasi pada Perguruan Tinggi Swasta dengan Zachman Framework*. Majalah Ilmiah UNIKOM vol.9 No.1

Kusumadewi, S. 2003. *Artificial Intelligence (Teknik dan Aplikasinya)*. Graha Ilmu. Jakarta.

Kusumadewi, Sri dan Kiki. 2009. *Analisis Jaringan Saraf Tiruan dengan Metode Backpropagation Untuk Mendeteksi Gangguan Psikologi*. Laboratorium Komputasi & Sistem Cerdas Jurusan Teknik Informatika, Fakultas Teknologi Industri Universitas Islam Indonesia.

Makridakis, S., Wheelwright, S.,C. & McGee, V., E. 1999. *Metode Dan Aplikasi Peramalan Edisi.2 Jilid.1*. Penerbit Erlangga. Jakarta.

Maru'ao, Dini O. 2010. *Implementasi Jaringan Saraf Tiruan Backpropagation dalam Memprediksi Kurs Valuta Asing*. Jurusan Teknik Informatika Universitas Gunadarma.

Mulyo, Ade S. 2002. *Ada Apa di Balik Perdagangan Valuta Asing? (Sebuah Kajian Praktikal)*. Jurnal ISSN Panutan Binis Vol.5 No.1

Noertjahyana, A dan Yulia. 2002. *Studi Analisa Pelatihan Jaringan Syaraf Tiruan Dengan Dan Tanpa Algoritma Genetika*. Jurnal Informatika Vol.3 No.1.

Nurwijaya. 2007. *Analisis Penggunaan Algoritma Genetika Untuk Optimalisasi Jaringan Syaraf Tiruan*. Institut Teknologi Bandung. Bandung.

O'Rourke, Carol, *Enterprise Architecture Using the Zachman Framework*, Thomson, Canada, 2003.

Puspitaningrum, Diyah. 2006. *Pengantar Jaringan Syaraf Tiruan*. Andi. Yogyakarta.

Rao, T. Venkat Narayana and Srikanth Madiraju. 2010. *Genetic algorithms and Programming An Evolutionary Methodology*. International Journal of Hybrid Information Technology Vol.3 No.4

Saputro, Nico. 2003. *Pengenalan Huruf dengan memakai Algoritma Genetik.* Integral, Vol. 8 No. 2.

Siang,JJ. 2005. *Jaringan Syaraf Tiruan dan Pemrogramannya Menggunakan Matlab*. Andi, Yogykarta.

Suyanto. 2005. *Algoritma Genetika dalam Matlab.* Andi. Yogyakarta.

Suyanto.ST,Msc. 2007. *Artificial Inteligence : Searching, Reasoning, Planning and Learning.* Informatika Bandung. Bandung.

Triyansyah, A. 2010. Implementasi Solver Optimasi Integer dan Mixed Integer Menggunakan Real Coded Genetic Algorithm. Jurusan Teknik Informatika, Institut Teknologi Sepuluh Nopember. Surabaya.