OPTIMASI BOBOT PADA *EXTREME LEARNING MACHINE* UNTUK PREDIKSI BEBAN LISTRIK MENGGUNAKAN ALGORITME GENETIKA
(Studi Kasus: PT. PLN (Persero) APD Kalsel dan Kalteng)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Vina Meilia
NIM: 145150201111159

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

OPTIMASI BOBOT PADA EXTREME LEARNING MACHINE UNTUK PREDIKSI BEBAN LISTRIK MENGGUNAKAN ALGORITME GENETIKA

(Studi Kasus: PT. PLN (Persero) APD Kalsel dan Kalteng)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Vina Meilia
NIM: 145150201111159

Skripsi ini telah diuji dan dinyatakan lulus pada
7 Juni 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Budi Darma Setiawan, S.Kom, M.Cs	Nurudin Santoso, S.T., M.T
NIP: 19841015 201404 1 002	NIP: 19740916 200012 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T.,M.T.,Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 30 April 2018

Vina Meilia

NIM: 145150201111159

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT karena berkat rahmat dan hidayah-Nya penulis mampu menyelesaikan skripsi yang berjudul "Optimasi Bobot Pada *Extreme Learning Machine* untuk Prediksi Beban Listrik Menggunakan Algoritme Genetika" dengan baik dan tepat pada waktunya. Skripsi ini disusun untuk memperoleh gelar sarjana pada Program Studi Teknik Informatika, Jurusan Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya.

Di dalam proses penyusunan skripsi ini, penulis tentu banyak mendapatkan bantuan dan dukungan motivasi dari berbagai pihak. Oleh karena itu, pada kesempatan kali ini penulis ingin menyampaikan rasa hormat dan terima kasih yang sedalam – dalamnya kepada:

1. Orang tua penulis tercinta, Bapak Musta'in dan Ibu Ida Suryani, kakak dan adik penulis tersayang, Annisa Rizqiani dan Syarhan Maulana, serta seluruh keluarga yang senantiasa memberikan kasih sayang, dukungan moral maupun material serta doa yang tiada henti untuk penulis.
2. Bapak Budi Darma Setiawan, S.Kom., M.Cs., dan Bapak Nurudin Santoso, S.T., M.T., selaku Dosen Pembimbing I dan Dosen Pembimbing II yang sudah membimbing dan memberikan pengarahan kepada penulis sehingga dapat menyelesaikan skripsi ini dengan baik.
3. Bapak Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D., selaku Dekan Fakultas Ilmu Komputer UB.
4. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D., dan Bapak Agus Wahyu Widodo, S.T., M.Cs., selaku Ketua Jurusan Teknik Informatika UB dan Ketua Program Studi Teknik Informatika UB.
5. Bapak dan ibu dosen serta seluruh civitas akademika Fakultas Ilmu Komputer UB yang telah memberikan ilmu dan membantu penulis selama masa kuliah.
6. Abdul Aziz yang selalu memberikan bantuan, motivasi, serta doa agar segera terselesaikannya skripsi ini.
7. Teman – teman seperantauan penulis Shella Dewi Puspitasari, Alifka Ayu Ratri, Sebtian Assyahlafi, Rizal Luqmana, dan lain – lain. Yang selalu membantu dan mengobati rindu kampung halaman selama masa kuliah penulis.
8. Teman – teman dekat penulis Amelia Achsandini Puteri, Eka Yuni Darmayanti, Anggita Mahardika, Yane Marita Febrianti, Dedy T Nababan yang senantiasa mendukung, membantu, dan menemani penulis selama masa kuliah.

9. Kakak Regina Anky Chandra, Pratomo Adinegoro dan Mohamad Fachrur Rozi yang turut memberikan saran, nasihat dan dukungan kepada penulis dalam pengerjaan skripsi ini.
10. Teman – teman SMA penulis Wahyu Chyntia P, Lita Mutia Sari, Muhammad Najib, dan lain – lain. Yang hingga saat ini masih memberikan dukungan serta doanya kepada penulis.
11. Teman – teman seperjuangan angkatan 2014 Fakultas Ilmu Komputer UB yang tidak dapat penulis sebutkan satu persatu yang telah bersama – sama melewati masa orientasi dan masa perkuliahan dengan penulis serta membantu secara langsung maupun tidak langsung.
12. Teman – teman, kakak – kakak dan adik – adik Fakultas Ilmu Komputer UB khususnya keluarga besar Himpunan Mahasiswa Informatika, Badan Eksekutif Mahasiswa TIIK Kabinet Bersatu III, Eksekutif Mahasiswa Informatika Kabinet Berinovasi, serta Badan Internal Olahraga dan Seni periode EXALT 2017 yang telah memberikan pengalaman berharga, bantuan, serta nasihat kepada penulis selama masa kuliah.
13. Seluruh pihak yang turut membantu secara langsung maupun tidak langsung yang tidak dapat penulis cantumkan satu persatu.

Semoga Allah SWT senantiasa membalas jasa dan kebaikan yang sudah tulus diberikan kepada penulis. Penulis pun menyadari, di dalam penyusunan skripsi ini masih banyak kekurangan dan jauh dari kata sempurna. Oleh karena itu, saran dan kritikan yang membangun akan sangat penulis butuhkan dengan harapan skripsi ini dapat memberikan manfaat yang baik kedepannya.

Malang, 30 April 2018

Penulis

vinamyla@gmail.com

ABSTRAK

Pertumbuhan konsumen listrik di Indonesia terus mengalami peningkatan yang signifikan dari tahun ke tahun. Akan tetapi peningkatan tersebut tidak diimbangi dengan penyediaan infrastruktur yang memadai. Hal ini menyebabkan kapasitas listrik yang tersedia tidak dapat memenuhi permintaan kebutuhan listrik. Sebagai upaya antisipasi, selain dengan menambah kapasitas listrik yang tentu membutuhkan biaya yang tidak sedikit. PLN juga melakukan manajemen operasi sistem, salah satunya adalah dengan dilakukannya prediksi beban listrik. Pada penelitian ini sebuah sistem komputasi cerdas dibangun untuk mengatasi hal tersebut. Dengan menggunakan metode *Extreme Learning Machine,* data input yang digunakan untuk melakukan prediksi beban listrik adalah data beban listrik per-jam. Di dalam metode *Extreme Learning Machine* terdapat bobot awal yang dibangkitkan secara acak dengan *range* -1 hingga 1. Sebelum dilakukan proses prediksi beban listrik, bobot awal tersebut akan dioptimasi terlebih dahulu dengan menggunakan algoritme genetika. Perhitungan nilai akurasi hasil prediksi dilakukan dengan menggunakan metode *Mean Absolute Precentage Error* (MAPE). Berdasarkan hasil pengujian, diperoleh nilai rata-rata *error* MAPE hasil prediksi dengan optimasi bobot sebesar 0.7996% sedangkan nilai rata-rata *error* MAPE tanpa optimasi bobot sebesar 1.1807%. Dari hasil tersebut dapat disimpulkan bahwa metode *Extreme Learning Machine* dengan optimasi bobot menggunakan algoritme genetika dapat diterapkan pada permasalahan prediksi beban listrik dan mampu memberikan hasil prediksi lebih baik.

Kata kunci: Beban Listrik, Optimasi Bobot, Prediksi Beban Listrik, *Extreme Learning Machine,* Algoritme Genetika.

ABSTRACT

The growth of electrical consumers in Indonesia continues to increases every year, but it is not matched by the provision of adequate infrastructure that available. This causes the available electrical capacity can't fulfill the demand for electricity. As an anticipation, beside to add more electrical capacities which will need a lot of costs. PLN also do operations management systems, which is electrical load forecasting. In this study, a smart computing system is build to solves the problem. Electrical load data per hour is being used as an input to do the electrical load forecasting with Extreme Learning Machine method. Extreme Learning Machine method uses random input weight within range -1 to 1. Before the electric load prediction process runs, genetic algorithms first optimizing the input weight. Mean Absolute Percentage Error (MAPE) is being used to calculate the accuration of prediction results. According to the test results with weight optimization, MAPE average error rate is 0.799% while without weight optimization the rate rise to 1.1807%. Thus this study implies that Extreme Learning Machine method with weight optimization using genetics algorithm can be used in electrical load forecasting problem and give better prediction result.

Keywords: Electric Load, Weight Optimization, Electrical Load Forecast, Extreme Learning Machine, Genetics Algorithm.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Listrik merupakan sumber energi yang disalurkan melalui sebuah kabel dan termasuk kedalam kebutuhan utama setiap manusia. Pada era kemajuan teknologi saat ini, semakin banyak alat – alat diciptakan dengan membutuhkan sumber daya listrik. Seperti handphone, laptop, kipas angin, printer, dan lain sebagainya. Berbagai macam aktivitas pun dilakukan dengan membutuhkan tenaga listrik.

Pertumbuhan konsumen listrik di Indonesia terus mengalami peningkatan yang signifikan. Berdasarkan data statistik ketenagalistrikan 2016, sampai pada akhir tahun 2015 jumlah pelanggan mencapai 61.167.980 pelanggan. Dibandingkan dengan tahun 2014 kenaikan yang terjadi sebesar 3.674.746 pelanggan atau 6,39%. Akan tetapi peningkatan tersebut tidak diimbangi dengan penyediaan infrastruktur yang memadai. Hal ini menyebabkan kapasitas listrik yang mampu di-*supply* lebih kecil daripada permintaan kebutuhan listrik (Muin, 2014). Sehingga pembangkit tidak kuat menahan beban yang melebihi kapasitas daya mampu dan mengakibatkan pemadaman. Untuk mengantisipasi hal tersebut selain dengan menambah pembangkit yang tentu membutuhkan biaya yang tidak sedikit bisa dengan melakukan manajemen operasi sistem. Manajemen operasi sistem adalah perencanaan pengoperasian yang meliputi perencaan penyaluran dan pembangkitan serta distribusi untuk mencapai operasi sistem tenaga listrik yang ekonomis, andal, dan berkualitas.

Salah satu manajemen operasi sistem adalah dengan dilakukannya prediksi beban listrik. Prediksi beban ini digunakan sebagai dasar dalam merencanakan pembangkitan dan penyaluran agar sesuai antara pembangkitan dan permintaan kebutuhan. Prediksi beban dibedakan menjadi tiga yaitu, prediksi beban jangka panjang, prediksi beban jangka menengah, dan prediksi beban jangka pendek. Masing – masing memiliki perbedaan yang terletak pada jangka waktu yang diprediksi. Prediksi sendiri memiliki pengertian hal yang akan terjadi di masa datang yang didasari oleh data masa sekarang ataupun data dimasa lampau. Prediksi sudah banyak diterapkan di masyarakat seperti halnya prediksi cuaca, perencanaan produksi, prediksi saham, dan lain - lain.

Sebuah mesin yang mampu memberikan prediksi dengan cepat dan akurat sangat diperlukan dalam permasalahan ini. Dengan banyaknya penelitian yang membahas tentang permasalahan prediksi banyak lahir pula metode – metode atau pendekatan yang mampu melakukan prediksi dengan cepat dan hasil yang lebih akurat. Salah satu metode yang mampu melakukan hal tersebut adalah metode *Extreme Learning Machine*.

Extreme Learning Machine merupakan sebuah metode yang diperkenalkan oleh Huang pada tahun 2004. Metode ini merupakan pengembangan dari *Artificial Neural Network* (Jaringan Syaraf Tiruan disingkat JST) yaitu sebuah algoritme yang mengadaptasi kinerja syaraf pada otak manusia dengan melalui sistem pembelajaran (Hasim, 2008). ELM atau biasa juga disebut dengan *Single-Hidden Layer Feedforward Neural Networks* (SLFNs) hanya memiliki satu *hidden layer* dan memiliki *learning*

accuracy lebih tinggi dibandingkan dengan aloritme JST biasa seperti *Backpropagation*. Hal tersebut dikarenakan pada ELM menggunakan bilangan yang di acak untuk menentukan nilai *input weight* dan biasnya. Akan tetapi, penentuan bilangan secara acak menurut Alencar, et al. (2016) dapat menyebabkan miskin generalisasi karena terbentuknya *hidden neuron* dengan jumlah yang besar. Selain itu nilai yang dibangkitkan memungkinkan memberikan hasil yang kurang optimal. Oleh karena itu, ditemukan solusi dari permasalahan tersebut, yaitu dengan cara menggabungkan metode ELM dengan Algoritme Genetika untuk mendapatkan hasil yang lebih optimal.

Algoritme Genetika adalah suatu metode pencarian solusi dengan melakukan pendekatan – pendekatan yang mengadaptasi dari teori evolusi Darwin yang terdiri dari kromosom, gen, individu, serta populasi. Metode ini sudah banyak digunakan dikehidupan sehari – hari seperti penentuan rute, pembuatan jalan, penentuan komposisi makanan yang bergizi (Mahmudy, 2013). Dengan pembangkitan bilangan secara acak untuk pencarian solusinya, metode ini mampu memberikan solusi optimal pada masalah – masalah yang kompleks dan rumit.

Penelitian sebelumnya dengan metode serupa dan objek yang berbeda pernah dilakukan. Seperti pada penelitian yang dilakukan oleh Wang, et al. pada tahun 2016 melakukan penelitian untuk memprediksi *wind power* dengan menggunakan metode ELM dan Algoritme Genetika. Pada percobaan tersebut dilakukan optimasi pada nilai *hidden node*, bias, dan koefisien regular. Dibandingkan dengan metode ELM biasa, penggabungan metode ini memberikan hasil akurasi yang lebih tinggi dan kemampuan generalisasi yang lebih baik (Wang, et al., 2016).

Kemudian ada penelitian yang dilakukan oleh Mar'atus Sholekhah pada tahun 2017 yang melakukan peramalan nilai tukar rupiah terhadap dolar Amerika dengan menggunakan penggabungan dari metode *Extreme Learning Machine* (ELM) dengan *Genetic Algorithm* (GA). Dari penelitian tersebut didapatkan nilai *Mean Absolute Error* (MAE) sebesar 0.101194 dan nilai rata – rata selisihnya adalah 2.354378% yang berarti peramalan yang dilakukan dengan menggunakan penggabungan metode tersebut mampu memberikan nilai yang mendekati nilai sebenarnya (Sholekhah, 2017).

Pada penelitian – penelitian tersebut dapat disimpulkan bahwa dengan menggabungkan metode ELM dengan Algoritme Genetika mampu memberikan hasil yang lebih baik dibandingkan hanya dengan menggunakan metode ELM biasa. Oleh karena itu penulis mengusulkan permasalahan prediksi beban listrik dapat diselesaikan dengan penggabungan kedua metode tersebut. Harapannya dengan menggunakan metode tersebut dapat memperoleh hasil yang lebih akurat dengan tingkat kesalahan yang lebih kecil. Dan untuk penggunaan lebih lanjut agar sistem ini dapat digunakan untuk membantu pihak penyedia listrik dalam merencanakan pembangkitan dan penyaluran listrik sesuai dengan permintaan kebutuhan.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang sudah dijabarkan, maka didapatkan sebuah rumusan masalah sebagai berikut:

1. Bagaimana performa Algoritme Genetika dalam melakukan optimasi bobot pada metode *Extreme Learning Machine*?
2. Berapa tingkat akurasi prediksi beban listrik dengan penggabungan metode *Extreme Learning Machine* dan Algoritme Genetika dibandingkan metode *Extreme Learning Machine* biasa dengan menggunakan metode *Mean Absolute Precentage Error* (MAPE)?

1.3 Tujuan

Adapun tujuan yang ingin dicapai dari penelitian ini adalah:

1. Mengetahui performa Algoritme Genetika dalam melakukan optimasi bobot pada metode *Extreme Learning Machine*.
2. Mengetahui hasil pengujian tingkat akurasi dari penggabungan metode *Extreme Learning Machine* dan Algoritme Genetika pada prediksi beban listrik dengan menggunakan metode *Mean Absolute Precentage Error* (MAPE).

1.4 Manfaat

Adapun manfaat yang diperoleh dalam penelitian ini adalah sebagai berikut:

1. Sebagai sarana informasi bagi pembaca dan bahan referensi bagi pihak yang membutuhkan.
2. Sebagai perencanaan operasi sistem bagi penyedia listrik agar penyediaan tenaga listrik sesuai dengan permintaan kebutuhan konsumen.

1.5 Batasan Masalah

Untuk mempermudah dan membatasi cakupan pembahasan masalah pada penelitian ini, maka disimpulkan batasan – batasan sebagai berikut:

1. Data yang digunakan adalah data historis beban listrik bulan November – Desember 2016 dari PT. PLN (Persero) Area Pengatur Distribusi Kalsel dan Kalteng.
2. Metode yang digunakan dalam penelitian ini adalah metode *Extreme Learning Machine* yang dioptimasi menggunakan Algoritme Genetika.
3. Metode yang digunakan untuk pengujian akurasi adalah metode *Mean Absolute Precentage Error* (MAPE).

1.6 Sistematika Pembahasan

Penelitian ini ditulis dengan sistematika penulisan yang terbagi menjadi beberapa bab bahasan, antara lain :

BAB 1 : PENDAHULUAN

Bab ini menjabarkan latar belakang penulis dalam memilih topik penelitian ini, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika penulisan.

BAB 2 : TINJAUAN PUSTAKA

Bab ini berisi uraian mengenai dasar teori pendukung yang berkaitan dengan penelitian, serta kajian pustaka dari penelitian-penelitian terdahulu yang dapat digunakan untuk menguatkan dasar teori yang sudah ada serta dapat menjadi dasar dari penelitian tentang prediksi beban listrik, tentang metode *Extreme Learning Machine* dan Algoritme Genetika.

BAB 3 : METODOLOGI PENELITIAN

Bab ini menjelaskan tentang tahapan yang akan dilakukan meliputi studi literatur, pengumpulan data, analisis kebutuhan, perancangan sistem, serta implementasi simulasi dari hasil perancangan sistem.

BAB 4 : PERANCANGAN SISTEM

Bab ini menjabarkan tentang perancangan sistem untuk hasil penelitian, perancangan antar muka, serta perancangan uji coba dan evaluasi sistem.

BAB 5 : IMPLEMENTASI

Bab ini berisi tentang pembahasan proses implementasi yang mengacu pada tinjauan pustaka pada penelitian ini.

BAB 6 : PENGUJIAN DAN ANALISIS

Bab ini berisi penjabaran hasil dari proses pengujian dan analisis kinerja sistem yang telah dibuat.

BAB 7 : PENUTUP

Bab terakhir ini menguraikan tentang kesimpulan yang dapat diambil dari penelitian yang telah dilakukan serta saran untuk penelitian yang selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Bab 2 Landasan kepustakaan ini berisi penjelasan dan uraian kajian pustaka, teori, konsep ataupun metode dari literature penelitian lainnya yang berkaitan dengan permasalahan pada penelitian ini. Pada landasan kepustakaan terdapat dasar teori dari berbagai sumber pustaka penelitian lainnya yang menunjang penelitian ini.

2.1 Kajian Pustaka

Terdapat beberapa penelitian yang meneliti tentang prediksi beban listrik antara lain:

1. Agus Hasim tahun 2008 melakukan sebuah penelitian yang bertujuan untuk memberi prakiraan beban listrik kota Pontianak. Dengan menggunakan metode Jaringan Syaraf Tiruan *Backpropagation*. Dan menggunakan data penelitian yaitu data historis harian beban listrik dalam *Mega Watt* (MW) dan temperatur udara maksimum, minimum, dan rata - rata kota Pontianak dari bulan Januari - Mei 2007. Dengan membandingkan hasil pengujian metode Backpropagation dengan metode konvensional PLN, dan metode penghitungan tingkat error rate – nya menggunakan *Mean Square Error* memperoleh hasil rata rata error metode *Backpropagation* selama satu minggu adalah 5.81%, sedangkan rata-rata error dengan metode konvensional PLN selama satu minggu adalah 8.24% Hasim, 2008).
2. Selanjutnya Aulia Khair tahun 2011 melakukan penelitian tentang peramalan beban listrik jangka pendek dengan menggunakan kombinasi metode *Autoregressive Integrated Moving Average* (ARIMA) dengan Regresi Linear. Pada penelitian ini peneliti mencoba mencari apakah ada hubungan antara konsumsi daya dan nilai suhu pada waktu yang sama. Selain menggunakn data historis beban listrik, digunakan juga data suhu udara daerah Cengkareng. Hasil dari penelitian tersebut menunjukkan bahwa antara suhu dengan konsumsi daya listrik memilik hubungan yang erat dengan MAPE rata – rata peramalan regresi linier sebesar 5%. Sedangkan hasil peramalan beban listrik dengan menggunakan kombinasi metode ARIMA dengan regresi linier memberikan hasil lebih baik dibandingkan dengan peramalan masing – masing metode. Hanya sebesar 4,19% error yang dihasilkan dari kombinasi peramalan tersebut (Khair, 2011).

Dari beberapa penelitian tersebut mengangkat masalah prediksi beban listrik dengan menggunakan metode yang berbeda – beda. Selain menggunakan metode yang disebutkan, metode *Extreme Learning Machine* yang di optimasi dengan Algoritme Genetika juga memberikan hasil yang baik jika digunakan untuk melakukan prediksi seperti yang ada pada penelitian – penelitian berikut:

1. Penelitian yang berasal dari Negara China oleh Xinyou Wang, Chenhua Wang dan Qing Li pada tahun 2016 adalah penelitian tentang tenaga angin. Karena pada saat ini penggunaan tenaga angin semakin banyak digunakan untuk menghemat energi listrik maka peneliti berinisiatif melakukan prediksi terhadap kekuatan tenaga angin untuk pemeliharaan dan penghematan

pengoperasiannya. Peneliti menggunanakan metode gabungan *Extreme Learning Machine* dan Algoritme Genetika dengan melakukan optimasi terhadap *hidden node*, bias dan koefisien regular. Jika dibandingkan dengan metode ELM saja, penggabungan metode ini memberikan hasil yang lebih akurat dan kemampuan generalisasi yang baik. Dengan nilai MAPE sebesar 9.1183 mampu mengalahkan metode ELM yang memiliki tingkat kesalahan sebesar 14.8229 (Wang, et al., 2016).

2. Pada tahun 2017, terdapat sebuah penelitian berjudul "*Hybrid* Jaringan Syaraf Tiruan Metode *Extreme Learning Machine* (ELM) dengan *Genetic Algorithm* (GA) untuk Meramalkan Nilai Tukar Rupiah terhadap Dolar Amerika". Penelitian tersebut dilakukan oleh Mar'atus Sholekhah. Dengan membagi data menjadi 2 yaitu data *training* dan data *testing,* kemudian melakukan optimasi pada bobot awal metode ELM. Tingkat akurasinya diperoleh dengan menggunakan metode *Mean Absolute Error* (MAE) dan mendapatkan hasil sebesar 0,101194 serta nilai selisih rata – ratanya sebesar 2.354378%. dengan hasil tersebut maka dapat ditarik kesimpulan bahwa hasil peramalan mampu mendekati nilai yang sebenarnya (Sholekhah, 2017).

Tabel 2.1 Kajian Pustaka

Tahun	Judul Penelitian	Obyek	Metode	Hasil
2008	Prakiraan Beban Listrik Kota Pontianak dengan Jaringan Syaraf Tiruan (Hasim, 2008)	Beban Listrik per jam dan Temperatur udara maksimum, minimum, rata-rata harian kota Pontianak	*Backpropagation*	Penelitian ini membandingkan metode BP dengan dengan metode konvensional PLN. Hasilnya metode BP menunjukkan hasil lebih baik dibandingkan metode konvensional PLN.
2011	Peramalan Beban Listrik Jangka Pendek Menggunakan Kombinasi Metode *Autoregressive Integrated Moving Average* (ARIMA) dengan Regresi Liniear (Khair, 2011)	Data beban listrik dan data temperatur	*Autoregressive Integrated Moving Average* (ARIMA) dan Regresi Liniear	Penelitian ini membandingkan metode ARIMA dan regresi linier dan memberikan hasil bahwa kombinasi metode tersebut memberikan hasil yang lebih baik dibanding hanya masing – masing metode.

Tahun	Judul Penelitian	Obyek	Metode	Hasil
2016	*Short-term Wind Power Prediction Using GA-ELM* (Wang, et al., 2016)	*Wind power*	*Extreme Learning Machine* dan Algoritme Genetika	Penelitian ini dilakukan untuk melakukan *maintenance* terhadap pengoperasian tenaga angin. Hasil yang didapatkan adalah metode GA-ELM mampu memberikan hasil yang lebih akurat dibandingkan dengan metode ELM saja.
2017	*Hybrid* Jaringan Syaraf Tiruan Metode *Extreme Learning Machine* (ELM) dengan *Genetic Algorithm* (GA) untuk Meramalkan Nilai Tukar Rupiah terhadap Dolar Amerika (Sholekhah, 2017)	Nilai tukar rupiah terhadap Dolar Amerika	*Extreme Learning Machine* dan Algoritme Genetika	Peneliti melakukan optimasi pada bobot awal metode ELM dan memperoleh hasil permalan yang mendekati nilai sebenarnya dengan menggunakan metode *Mean Absolute Error* (MAE).
2018	Optimasi Bobot Pada *Extreme Learning Machine* Untuk Prediksi Beban Listrik Menggunakan Algoritme Genetika	Beban listrik per-setengah jam	*Extreme Learning Machine* dan Algoritme Genetika	-

2.2 Penyedia Tenaga Listrik

PT. PLN (Persero) merupakan perusahaan penyedia tenaga listrik terbesar di Indonesia. Semakin berkembang dan banyaknya teknologi yang membutuhkan energi listrik saat ini, menyebabkan keberadaan PLN sangat dibutuhkan oleh seluruh masyarakat. Sebagai penyedia tenaga listrik, PLN selalu berusaha menyediakan energi listrik sesuai dengan kebutuhan konsumsi listrik yang digunakan atau istilah

tersebut biasa disebut dengan beban listrik. Setiap peralatan yang terhubung dengan sistem daya dan mengkonsumsi energi listrik dapat dikatakan sebagai beban listrik. Sedangkan besar kecilnya beban sangat bergantung pada penggunaan peralatan listrik oleh para konsumen listrik. Upaya pemenuhan kebutuhan yang sesuai sangat lah penting, untuk menghindari defisit energi listrik yang dapat disalurkan dan agar tidak terjadi pemadaman listrik.

Ada beberapa faktor yang mempengaruhi perubahan beban listrik:

1. Pertumbuhan pelanggan: Bertambahnya jumlah konsumen atau pelanggan listrik.
2. Pertumbuhan beban: Bertambahnya tingkat konsumsi tenaga listrik oleh pelanggan, karena adanya penambahan peralatan listrik.
3. Faktor Alam / Cuaca: Jika cuaca sedang panas maka pemakaian alat – alat pendingin akan bertambah. Sebaliknya jika cuaca sedang dingin atau badai maka akan terjadi pengurangan beban untuk menghindari kerusakan peralatan.
4. Penyimpangan terhadap rutinitas: Terdapat kegiatan khusus dalam masyarakat, misalnya seperti pada hari – hari besar Idul Fitri, 17 Agustus, Tahun baru, dll.
5. Realisasi beban periode sebelumnya: Data historis beban periode sebelumnya.

2.2.1 Manajemen Operasi

Manajemen operasi merupakan upaya pemanfaatan sumber daya energi yang tersedia secara efektif, efisien, dan rasional dengan menggunakan pendekatan sistematis. Dengan tujuan memperbaiki kualitas, meminimalisir biaya, serta menjamin keamanan sistem tenaga listrik (PLN, 2012). Salah satu manajemen operasi yang dilakukan dengan mengacu pada sisi beban adalah *Deman Side Management.*

Strategic Load Growth merupakan salah satu metode yang terdapat pada *Demand Side Management* yang digunakan sebagai tindakan setelah dilakukannya prediksi beban listrik. Dengan mengatur pertumbuhan beban sesuai dengan perkembangan beban dan hasil prediksi beban listrik.

2.2.3 Prediksi Beban Listrik

Demi memenuhi permintaan kebutuhan konsumen akan listrik yang semakin meningkat tiap tahunnya. Maka dilakukan sebuah perencanaan operasi agar permintaan kebutuhan terpenuhi sehingga tidak terjadi pemborosan sumber daya. Perencanaan yang dilakukan adalah dengan melakukan prediksi pada beban listrik.

Prediksi atau peramalan sendiri merupakan suatu dugaan atau perkiraan mengenai terjadinya suatu kejadian di waktu yang akan datang. Walaupun ramalan tidak akan pernah tepat 100% karena masa depan mengandung ketidakpastian. Akan tetapi prediksi banyak digunakan dalam berbagai aspek, utamanya sebagai

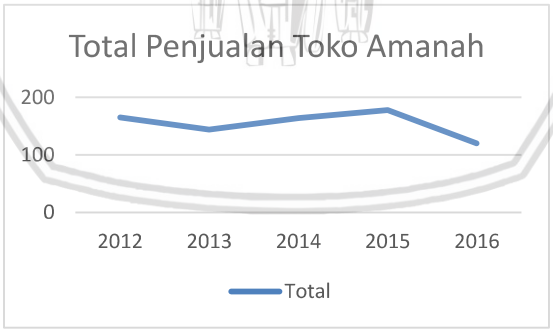
perencanaan untuk mengantisipasi hal – hal yang terjadi di masa datang dengan berbagai macam keadaan. Seperti untuk meramalkan kesuksesan dalam bekerja, dalam berinvestasi, dan lain sebagainya. Dengan pemilihan metode yang tepat prediksi dengan tingkat kesalahan yang kecil atau hasil perkiraan yang sebaik mungkin akan bisa didapat.

Prediksi ini akan berguna untuk penentuan parameter – parameter penyediaan tenaga listrik dan pemeliharaan pembangkit seperti misalnya: kapasitas dan jenis pembangkit, jadwal pemeliharaan pembangkit, kapasitas saluran transmisi, dan lain – lain. Pada Perusahaan Listrik Negara (PLN) sudah dilakukan upaya prediksi beban listrik konvensional dengan menggunakan metode koefisien beban. Akan tetapi metode ini cukup sulit diterapkan oleh orang yang bukan ahli dan memiliki pengalaman dalam memahami karakteristik beban. Sehingga masih diperlukan adanya sebuah metode alternatif yang lebih mudah dan akurat untuk mengatasi hal tersebut (Hasim, 2008).

2.2.2 Jenis Data Prediksi

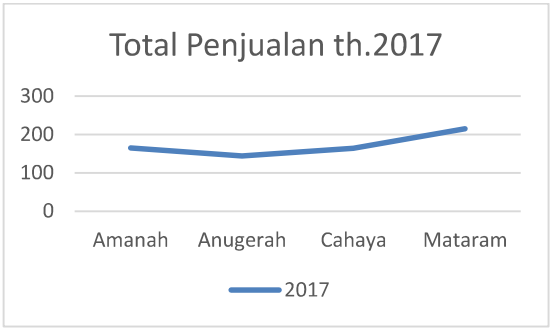
Data yang akan digunakan sebagai alat dan teknik analisis harus benar dan akurat. Oleh karena itu, pengetahuan dan pemahaman terkait pengelompokkan data sangatlah penting, mengingat konsekuensi pengelolaan dan penganalisaan data sangat bergantung dari sumber dan jenis data. Berdasarkan waktunya, data dibagi menjadi 3 kelompok yang terdiri dari (Gani dan Amalia, 2015):

a. Data Runtut Waktu (*time series*) : data yang berasal dari suatu sumber dalam jangka waktu tertentu secara berurutan. Contoh : data total penjualan toko Amanah selama 5 tahun.



Gambar 2.1 Data *Time Series*

b. Data Silang Tempat (*cross section*) : data dalam satu waktu yang berasal dari beberapa sumber data. Contoh : data total penjualan tahun 2017 toko Amanah, toko Anugerah, toko Cahaya dan toko Mataram.

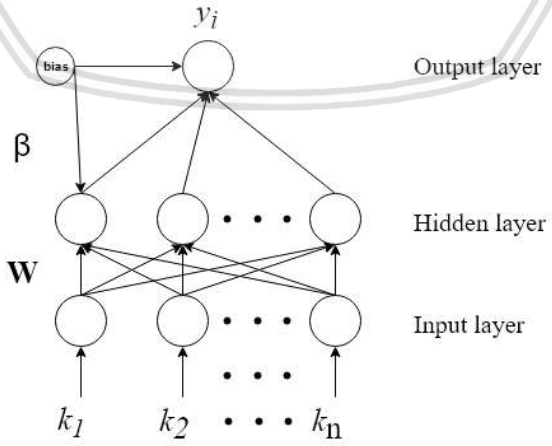


Gambar 2.2 Data Cross Section

c. Data Gabungan (*pooling*) : kombinasi antara data runtut waktu dan data silang tempat. Contoh data total penjualan toko Amanah, toko Anugerah, dan toko Cahaya pada tahun 2012 – 2016.

2.3 *Extreme Learning Machine* (ELM)

Extreme Learning Machine merupakan metode yang pertama kali diperkenalkan oleh Huang, Zhu, dan Siew pada tahun 2004. Metode ini merupakan pengembangan dari metode jaringan syaraf tiruan (disingkat JST) yang mana biasa disebut dengan *Single Hidden Layer Feedforward Neural Networks* (SLNFs). JST sendiri merupakan sebuah metode yang proses pembelajarannya mengadaptasi dari jaringan syaraf sistem otak manusia. Metode ELM muncul sebagai jawaban dari permasalahan yang terdapat dalam JST *feedforward* yaitu dalam hal lama waktu pembelajaran (*learning speed)*. Dengan hanya menggunakan satu *hidden layer* dan proses penentuan parameternya yang dibangkitkan secara acak mampu memberikan hasil prediksi yang lebih stabil dengan waktu yang relatif lebih cepat dibandingkan dengan jaringan *feedworward* biasa (Huang, et. al., 2004).



Gambar 2.3 Struktur Metode ELM

Berdasarkan pada Gambar 2.3, metode ELM memiliki struktur yang terdiri dari 3 lapisan, yaitu lapisan *input,* lapisan tersembunyi, dan lapisan *output.* Setiap node pada masing – masing lapisan saling dihubungkan dengan bobot yang memiliki nilai yang berbeda – beda. Yang mana semua saling terhubung untuk menuju pada satu *output* (Azizah, 2016).

Adapun langkah – langkah dari metode *Extreme Learning Machine* adalah sebagai berikut (Huang, et. al., 2006):

Langkah 1 Inisialisasi Bobot Awal dan Bias

Melakukan pembangkitan *learning parameter* yaitu nilai bobot awal sebagai bobot masukan (W_{jk}) dengan range antara -1 hingga 1 dan nilai bias *b* dengan range nilai antara 0 hingga 1 dengan ukuran matrik 1 x *j*. Indeks *k* menunjukkan banyaknya *node input layer* sedangkan *j* menunjukkan banyaknya *node hidden layer.* Nilai – nilai tersebut dibangkitkan secara acak.

Langkah 2 Menghitung *Output Hidden Layer*

Setelah memperoleh nilai bobot masukan dan bias, kemudian menghitung matriks *output hidden layer* (H_{init}) seperti pada Persamaan 2.1.

$$Hinit_{(i)} = X_{(i)}\, x\, {W_{(i)}}^{T} + b_{(i)} \quad (2.1)$$

Keterangan :

$X_{(i)}$ = data masukan ke-*i*

${W_{(i)}}^{T}$ = bobot masukan yang di-*transpose* ke-*i*

$b_{(i)}$ = bias ke-*i*

Langkah 3 Aktivasi Sigmoid

Proses ini mengaktivasi hasil *output hidden node* agar nilai tersebut mampu untuk digunakan pada proses selanjutnya. Proses ini digunakan untuk memetakan nilai hasil *output hidden node* sebelumnya dengan menggunakan fungsi aktivasi *Sigmoid Biner.* Fungsi aktivasi *sigmoid biner* merupakan fungsi *non linier* yang memiliki rentang nilai *output* antara 0 sampai 1. Rumus untuk melakukan aktivasi sigmoid terdapat pada Persamaan 2.2.

$$H = \frac{1}{(1 + exp^{-Hinit(i)})} \quad (2.2)$$

Keterangan :

H = matriks output *hidden layer*

Langkah 4 Menghitung Matriks *Moore-Penrose Pseudo Inverse*

Kemudian setelah mendapatkan hasil aktivasi tersebut, dilakukan penghitungan matriks *Moore-Penrose Pseudo Inverse* seperti pada Persamaan 2.3.

$$H^{+} = (H^{T} H)^{-1} H^{T} \quad (2.3)$$

Keterangan :

H = matriks output *hidden layer*

H^{+} = *moore-penrose pseudo inverse*

H^{T} = matriks H di-*transpose*

Langkah 5 Menghitung Bobot Output

Proses ini dilakukan untuk menghubungkan antara *hidden layer* dengan *output layer* untuk mendapatkan hasil *output* seperti pada Persamaan 2.4.

$$\beta = H^{+}T \quad (2.4)$$

Keterangan :

β = bobot *output*

H^{+} = matriks *moore-penrose pseudo inverse*

T = matriks target

Langkah 6 Menghitung Hasil Prediksi

Langkah terakhir adalah menghitung hasil prediksi seperti pada Persamaan 2.5.

$$\hat{Y} = H\beta \quad (2.5)$$

Keterangan :

$\hat{Y}$ = hasil prediksi

β = bobot *output*

H = matriks *output hidden layer*

Pada metode ELM proses pembelajaran dibagi menjadi 2 yaitu proses *training* dan proses *testing.* Pada proses *training* urutan langkah yang dilakukan adalah Langkah 1, Langkah 2, Langkah 3, Langkah 4, dan Langkah 5. Sedangkan pada proses *testing* urutan langkah yang dilakukan adalah Langkah 1, Langkah 2, Langkah 3, dan kemudian Langkah 6.

2.4 Algoritme Genetika

Algoritme evolusi merupakan sebuah bidang penelitian interdispliner yang menghubungkan antara biologi, kecerdasan buatan, optimasi numerik, dan sistem pendukung keputusan (Back, 1996). Salah satu algoritme yang termasuk dalam algoritme evolusi yaitu algoritme genetika. Algoritme ini pertama kali diperkenalkan oleh John Holland pada tahun 1975. Sebuah teknik optimasi yang prosesnya

mengadaptasi dari evolusi biologi ini mampu memberikan solusi – solusi dari permasalahan yang kompleks. Selain dapat memberikan solusi pada permasalahan dalam kehidupan sehari – hari seperti penentuan komposisi pakan ternak yang baik (Kusuma, Mahmudy & Indriati, 2015), algoritme ini juga sudah banyak diterapkan di berbagai bidang keilmuan seperti ekonomi, fisika, sosiologi dan masih banyak lainnya.

Gambar 2.4 Proses Algoritme Evolusi

Sumber: (Mahmudy, 2015)

Gambar 2.4 merupakan tahapan algoritme genetika. Bermula dari individu – individu yang berperan sebagai *parent* kemudian saling bereproduksi untuk mendapatkan keturunan (*offspring*). Selanjutnya induk beserta keturunannya berevolusi dan melewati fase seleksi alam, bagi individu yang tidak mampu bertahan hidup maka akan mati. Populasi akan terbentuk dari proses reproduksi individu – individu terpilih. Individu tersebut memiliki peluang untuk menghasilkan keturunan yang lebih baik dari generasi ke generasi (Mahmudy, 2015). Solusi – solusi dari permasalahan direpresentasikan sebagai sebuah individu, kemudian diukur dengan menggunakan fungsi *fitness* untuk mendapatkan individu terbaik. Algoritme ini memiliki sifat *stochastic*, dengan permasalahan yang sama ia mampu menghasilkan kemungkinan solusi yang yang berbeda – beda setiap kali di-*generate* (Smith & Eiben, 2003).

Dalam prosesnya karena diadaptasi dari ilmu genetika, maka banyak istilah yang digunakan juga menggunakan istilah – istilah yang ada pada ilmu genetika, seperti gen, individu, kromosom. Gen merupakan sebuah representasi dari solusi yang dihasilkan. Sekumpulan gen yang disebut dengan kromosom atau individu akan diproses sebagaimana proses evolusi biologi yaitu reproduksi untuk menghasilkan keturunan. Dari kumpulan kromosom dan keturunannya tersebut kemudian dilakukan seleksi dengan menghitung nilai *fitness*-nya untuk kemudian diproses kembali pada generasi selanjutnya dan seterusnya. Hingga didapatkan kromosom yang paling baik atau solusi yang terbaik (Megantara, 2017).

2.4.1 Representasi & Inisialisasi Kromosom

Pada algoritme genetika yang pertama kali diperkenalkan, representasi kromosomnya menggunakan nilai biner. Akan tetapi menurut Wei-Qing Xiong, et al. bahwa representasi dengan menggunakan nilai biner tidak dapat menunjukkan struktur dari solusi secara langsung dan tentu saja memakan lebih banyak waktu serta memori pada saat melakukan transformasi dari biner ke bilangan *real* dan sebaliknya. Oleh karena itu, dengan menggunakan pengkodean bilangan *real* lebih

mampu mengatasi permasalahan pada saat menggunakan bilangan biner (Herrera, Lozano, & Verdegay, 1998).

Inisialisasi kromosom dilakukan dengan cara membangkitkan himpunaan solusi secara *random.* Panjangnya sejumlah banyaknya gen dalam satu kromosom. Misalkan dalam satu kromosom terdiri dari 2 gen maka panjang kromosomnya adalah 2.

2.4.2 Reproduksi

Tahap berikutnya untuk bisa mendapatkan keturunan (offspring) dilakukan proses reproduksi. Pada algoritme genetika terdapat 2 proses reproduksi yang harus dilakukan yaitu *crossover* dan mutasi. Terdapat beberapa metode yang sudah dikembangkan untuk melakukan proses *crossover* dan mutasi*,* salah satunya adalah *Extended Intermediate Crossover* dan *Random Mutation.* Sebelum melakukan proses reproduksi tentukan terlebih dahulu nilai *cr* (*crossover rate*) dan *mr* (*mutation rate*), nilai ini berfungsi untuk menentukan rasio keturunan (*offsipring*) yang harus dihasilkan. Jumlah *offspring* didapat dengan cara mengalikan nilai cr/mr dengan ukuran populasi (ukuran populasi).

1. *Crossover (Extended Intermediate Crossover)*

Proses ini melibatkan dua buah *parent* yang dipilih secara *random* untuk kemduian dikawinkan secara silang dan menghasilkan kromosom baru. Setelah itu membangkitkan nilai alpha (α) secara acak dengan *range* -0,25 sampai 1,25. Kemudian dihitung dengan menggunakan Persamaan 2.6

$$C(1) = P(1) + \alpha\big(\mathrm{P}(2) - P(1)\big)$$
$$C(2) = P(2) + \alpha\big(\mathrm{P}(1) - P(2)\big) \qquad (2.6)$$

Keterangan :

$C(1)$ = Hasil *Crossover* untuk *parent* 1

$C(2)$ = Hasil *Crossover* untuk *parent* 2

$P(1)$ = *Parent* ke 1

$P(2)$ = *Parent* ke 2

α = Nilai *alpha* dengan range [-0.25;1.25]

2. Mutasi (*Random Mutation*)

Jika pada *crossover* melibatkan dua buah *parent* maka pada mutasi hanya melibatkan satu buah induk saja. Tujuan dari proses mutasi adalah untuk menjaga keragaman populasi (Mahmudy, 2015). Dengan *random mutation* kromosom terpilih akan dilakukan pemilihan indeks yang kemudian akan ditambah atau dikurangi dengan bilangan *random* terkecil dengan menggunakan Persamaan berikut.

$$x'(i) = x'(i) + r(max_i - min_i) \qquad (2.7)$$

Keterangan :

$x'(i)$ = *Parent* terpilih yang akan dimutasi

r = nilai *random* dengan range [-0.1;0.1]

max_i = nilai maksimum dari gen variable x_i

min_i = nilai minimum dari gen variable x_i

2.4.3 Evaluasi

Evaluasi digunakan untuk menilai suatu individu atau kromosom apakah dia layak untuk dipertahankan atau tidak. Untuk mengukur suatu individu digunakan sebuah nilai yang disebut dengan nilai *fitness*. Akan tetapi, sebelum melakukan proses penghitungan *fitness* terlebih dahulu dilakukan pengecekan *hard constraint. Hard constraint* merupakan adanya sebuah solusi yang nilainya melebihi batasan atau dalam permasalahan ini adalah nilai dari solusi melebihi dari *range* awal yang sudah ditentukan. Pada permasalahan optimasi memang terkadang perlu untuk menerapkan sebuah *constraint* agar solusi yang dihasilkan tidak melanggar batasan (Gondro dan Kinghorn, 2008).

Dalam penelitian ini, *hard constraint* diterapkan ketika terdapat gen yang nilainya kurang atau lebih dari *range*, gen tersebut akan diperbaiki dengan menggantinya dengan gen baru yang dibangkitkan secara *random* dengan *range* yang sesuai. Untuk perhitungan nilai *fitness* semakin besar nilai *fitness* dari sebuah individu maka semakin baik individu tersebut. Karena fungsi yang digunakan untuk mencari nilai *fitness* tersebut menggunakan nilai MAPE maka rumus *fitness* dibuat seperti pada Persamaan 2.8.

$$f(x) = \frac{1}{(1 + nilaiMAPE(ELM))} \quad (2.8)$$

2.4.4 Seleksi

Pada tahapan terakhir algoritme genetika terdapat proses seleksi, yaitu proses untuk menyeleksi individu – individu terbaik yang mampu bertahan hidup hingga generasi selanjutnya. Dengan menggunakan metode *elitism selection*, seluruh individu diurutkan dari besar ke kecil sesuai dengan nilai *fitness*-nya. Kemudian diambil beberapa individu sebanyak 30% dari ukuran populasi yang memiliki nilai *fitness* terbesar untuk kemudian diproses pada generasi berikutnya (Mahmudy, 2015).

2.5 Gabungan Algoritme Genetika dengan Metode *Extreme Learning Machine* (ELM)

Extreme learning machine merupakan salah satu metode peramalan pada JST yang memiliki keunggulan pada *learning speed* dibandingkan dengan metode JST lainnya. Hal tersebut dipengaruhi oleh penentuan bobot awal dan biasnya yang dilakukan secara *random.* Pembangkitan secara *random* tersebut menyebabkan prediksi memberikan hasil yang berbeda – beda atau berubah – ubah dan belum tentu optimal. Oleh karena itu algoritme genetika hadir sebagai solusi dari permasalahan tersebut untuk mengoptimasi nilai bobot awal agar prediksinya selalu memberikan hasil optimal.

Adapun langkah – langkah perhitungan gabungan algoritme genetika dengan *extreme learning machine* sebagai berikut:

Langkah 1 Melakukan inisialisasi jumlah ukuran populasi, jumlah generasi, nilai cr, dan nilai mr.

Langkah 2 Melakukan inisialisasi kromosom awal.

Langkah 3 Melakukan proses reproduksi yang terdiri dari proses *crossover* dan mutasi dengan menggunakan Persamaan 2.6 dan 2.7.

Langkah 4 Melakukan proses evaluasi pada masing – masing kromosom dengan menggunakannya sebagai *input* bobot awal metode ELM untuk memperoleh nilai MAPE-nya.

Langkah 5 Melakukan proses seleksi untuk mendapatkan nilai *fitness* tertinggi yang diperoleh dari nilai MAPE.

Langkah 6 Mengulang langkah diatas hingga generasi maksimum.

Langkah 7 Setelah didapatkan kromosom terbaik, kemudian melakukan proses yang ada pada metode ELM untuk melihat data hasil prediksi. Proses pertama adalah proses *training* yang dijabarkan sebagai berikut:

a. Menghitung *output hidden layer* dengan data *training* menggunakan Persamaan 2.1.
b. Melakukan aktivasi sigmoid biner dengan menggunakan Persamaan 2.2.
c. Menghitung matrik *moore-penrose pseudo invers* dengan menggunakan Persamaan 2.3.
d. Menghitung bobot output dengan menggunakan Persamaan 2.4.

Langkah 8 Setelah melakukan proses *training* kemudian melakukan proses *testing* pada metode ELM yang dijabarkan sebagai berikut:

a. Menghitung *output hidden layer* dengan data *testing* menggunakan Persamaan 2.1.
b. Melakukan aktivasi sigmoid biner dengan menggunakan Persamaan 2.2.
c. Melakukan proses prediksi dengan menggunakan Persamaan 2.5.

Langkah 9 Setelah mendapatkan data hasil prediksi maka proses selesai.

2.6 Metode Penghitungan Tingkat Akurasi Prediksi

Setiap prediksi atau peramalan pasti tidak ada yang 100% hasilnya sesuai dengan kenyataan. Karena semua itu kehendak alam dan Tuhan, manusia hanya mampu memperkirakannya. Banyak faktor yang mempengaruhi hasil prediksi, salah satunya adalah pemilihan metode yang tepat. Tepat berarti sesuai dengan sifat – sifat data yang akan diprediksi. Memang tidak ada metode yang benar - benar tepat hingga hasil prediksi tersebut sama dengan kenyataan. Akan tetapi kita dapat melihat apakah metode tersebut dapat menghasilkan prediksi mendekati kenyataan dengan menghitung nilai kesalahan peramalannya.

Mean Absolute Precentage Error (MAPE) merupakan metode yang digunakan untung menghitung tingkat akurasi dari sebuah hasil prediksi dengan menghitung total tingkat kesalahan dari prediksi. Metode ini menunjukkan rata – rata prosentase kesalahan dari selisih nilai aktual dan nilai hasil peramalan. Dengan menggunakan rumus sebagai berikut (Khair, 2011):

$$\text{MAPE} = \left| \frac{100}{n} \Sigma_{t=1}^{n} \left(\frac{Y - \hat{Y}}{Y} \right) \right| \quad (2.9)$$

Keterangan :

Y = data aktual

$\hat{Y}$ = data prediksi

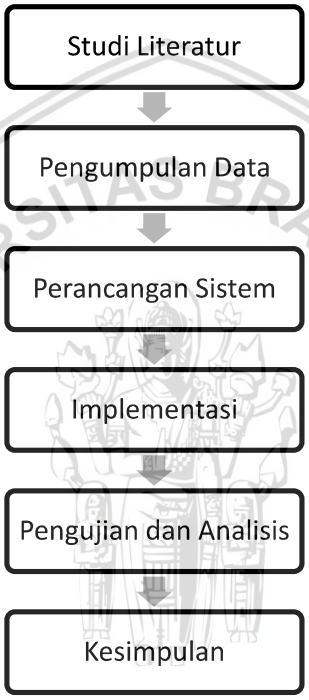
n = total banyak data historis yang di observasi

BAB 3 METODOLOGI PENELITIAN

Metodologi penelitian ini menjelaskan tentang tahapan yang dilakukan penulis dalam mengimplementasikan sistem Optimasi Bobot pada *Extreme Learning Machine* untuk Prediksi Beban Listrik Menggunakan Algoritme Genetika.

3.1 Tahapan Penelitian

Adapun tahapan – tahapannya terdiri atas beberapa langkah yaitu studi literature, pengumpulan data, perancangan sistem, implementasi sistem, pengujian sistem dan penarikan kesimpulan. Berikut ini adalah diagram alur dari



Gambar 3.1 Metode Penelitian

3.2 Studi Literatur

Studi literatur digunakan sebagai tambahan informasi untuk mempelajari dasar teori yang akan digunakan dalam penelitian ini. Studi literatur ini dilakukan dengan cara mengumpulkan beberapa informasi dari buku ataupun dari jurnal – jurnal yang ada di internet. Yang meliputi:

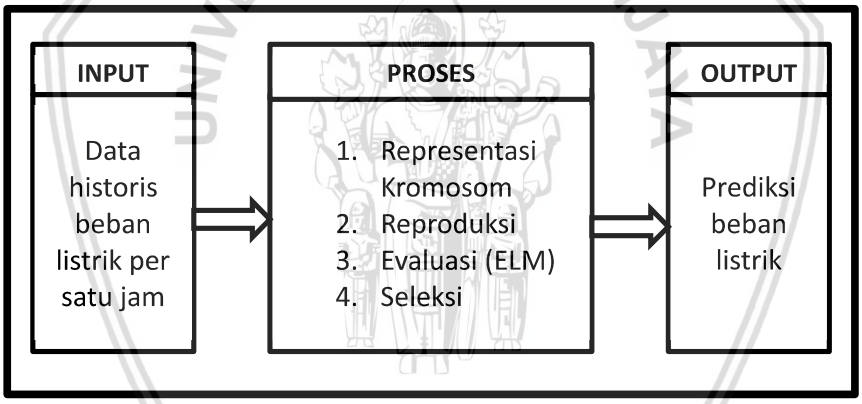
- Beban Listrik
- Metode *Extreme Learning Machine* (ELM).
- Metode Algoritme Genetika
- Metode *Mean Absolute Precentage Error* (MAPE).

3.3 Pengumpulan Data

Teknik pengumpulan data merupakan faktor penting demi keberhasilan penelitian. Hal ini berkaitan dengan bagaimana cara pengumpulan data, siapa sumbernya, dan apa alat yang digunakan. Jenis sumber data adalah mengenai dari mana data diperoleh. Apakah data diperoleh dari sumber langsung (data primer) atau data diperoleh dari sumber tidak langsung (data sekunder). Dalam penelitian ini, data yang didapatkan merupakan data sekunder, yaitu data historis beban listrik bulan November - Desember 2016. Data tersebut berupa data harian yang memperlihatkan perilaku beban listrik selama 24 jam dalam Mega Watt (MW) yang diperoleh dari PT. PLN (Persero) Area Pengatur Distribusi Kalsel dan Kalteng.

3.4 Perancangan Sistem

Tahap ini merupakan tahap yang menjabarkan tentang deskripsi sistem, *flowchart*, perhitungan manual, perancangan antarmuka, serta skenario pengujian dari sistem. Pada tahap perancangan ini digunakan sebagai acuan pada proses implementasi, agar apa yang diimplementasikan sesuai dengan kebutuhan sistem.



Gambar 3.2 Diagram Blok Optimasi Metode ELM dengan Algoritme Genetika

Penjelasan diagram blok tersebut sebagai berikut:

1. Input
   Masukan dari sistem yaitu 5 fitur yang terdiri dari data beban listrik minggu ke 1 – 5 pada hari dan jam yang sama.
2. Proses
   Pada proses dilakukan penghitungan dengan menggunakan metode *Extreme Learning Machine* (ELM) yang telah dilakukan optimasi dengan menggunakan Algoritme Genetika pada nilai bobotnya.
3. Output
   Output yang diharapkan pada sistem ini adalah mampu memberikan prediksi beban listrik untuk periode yang akan datang.

3.5 Implementasi

Implementasi sistem dibuat setelah melakukan proses perancangan pada sistem. Proses implementasi terdiri dari:

1. Implementasi untuk mempermudah pengambilan dan penyimpanan data menggunakan *Microsoft Excel*.
2. Implementasi algoritme dan antarmuka sistem dengan metode *Extreme Learning Machine* (ELM) dan Algoritme Genetika dalam bahasa pemrograman Java dan aplikasi Netbeans IDE 8.2.

3.6 Pengujian dan Analisis

Tahap pengujian dibuat setelah melakukan proses perancangan dan implementasi sistem. Tahap ini bertujuan untuk mengetahui apakah sistem yang dibuat telah sesuai dan juga untuk mendapatkan tingkat akurasi prediksi dari sistem dengan menggunakan metode *Mean Absolute Precentage Error* (MAPE). Terdapat skenario yang digunakan dalam proses pengujian, antara lain:

1. Pengujian kombinasi *crossover rate* dan *mutation rate*.
2. Pengujian ukuran populasi.
3. Pengujian data beban listrik.

Setelah pengujian selesai dilakukan, maka selanjutnya adalah proses analisis hasil. Berdasarkan hasil prediksi beban listrik yang didapatkan dengan menggunakan penggabungan antara metode *Extreme Learning Machine* (ELM) dan Algoritme Genetika.

3.7 Penarikan Kesimpulan dan Saran

Proses penarikan kesimpulan dilakukan setelah penelitian selesai dilakukan. Dari awal proses studi literatur hingga pengujian dan analisis hasil dari sistem Optimasi Bobot Pada *Extreme Learning Machine* (ELM) Untuk Prediksi Beban Listrik dengan Menggunakan Algoritme Genetika. Selain itu, saran juga ditambahkan untuk perbaikan dan pengembangan penelitian yang sudah dibuat.

BAB 4 PERANCANGAN SISTEM

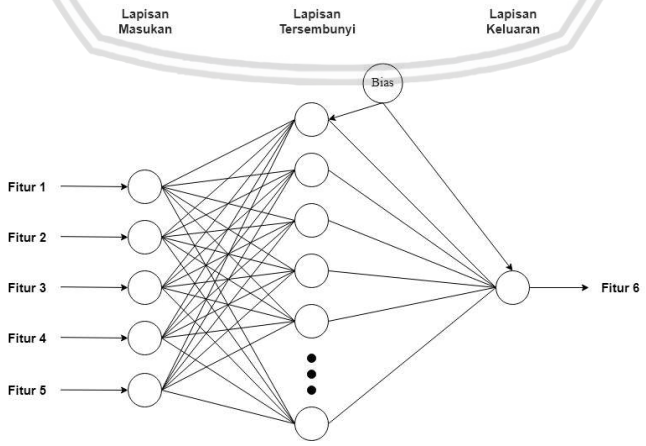
Bab perancangan sistem ini menjelaskan terkait deskripsi sistem, diagram alir yang menjabarkan tentang penyelesaian metode Algoritme Genetika dan *Extreme Learning Machine* (ELM), perhitungan manual, perancangan antarmuka dan perancangan skenario pengujian.

4.1 Formulasi Permasalahan

Dalam penelitian ini masalah yang ingin diselesaikan adalah prediksi beban listrik. Seperti yang sudah dijelaskan pada latar belakang penelitian ini. Guna melakukan manajemen operasi listrik sebagai perencanaan penyaluran dan pembangkitan listrik yang sesuai dengan kebutuhan konsumen maka dilakukan prediksi beban listrik. Oleh karena itu, adanya sebuah perangkat cerdas yang mampu memberikan prediksi secara cepat dengan hasil yang akurat dapat sangat membantu proses manajemen operasi. Dengan menggunakan metode ELM yang telah dioptimasi terlebih dahulu bobotnya menggunakan Algoritme Genetika. Dan menggunakan konsep *similar day* yaitu pola beban pada hari dan jam sekarang dipengaruhi oleh pola beban pada hari dan jam yang sama pada waktu sebelumnya (Perdana, et al 2012).

4.2 Desain Arsitektur Sistem

Pada desain arsitektur metode *Extreme Learning Machine* (ELM) terdapat beberapa lapisan yang memiliki fungsinya masing – masing. Untuk melakukan sebuah prediksi beban listrik jangka pendek maka dibutuhkan data dengan pola beban yang rinci. Data yang digunakan adalah data pola beban harian per jam selama 24 jam dari bulan November - Desember 2016 dengan total 1464 data. Dengan kompleksnya data yang digunakan maka dibuat sebuah desain arsitektur seperti yang ditunjukkan pada Gambar 4.2.



Gambar 4.1 Desain Arsitektur ELM

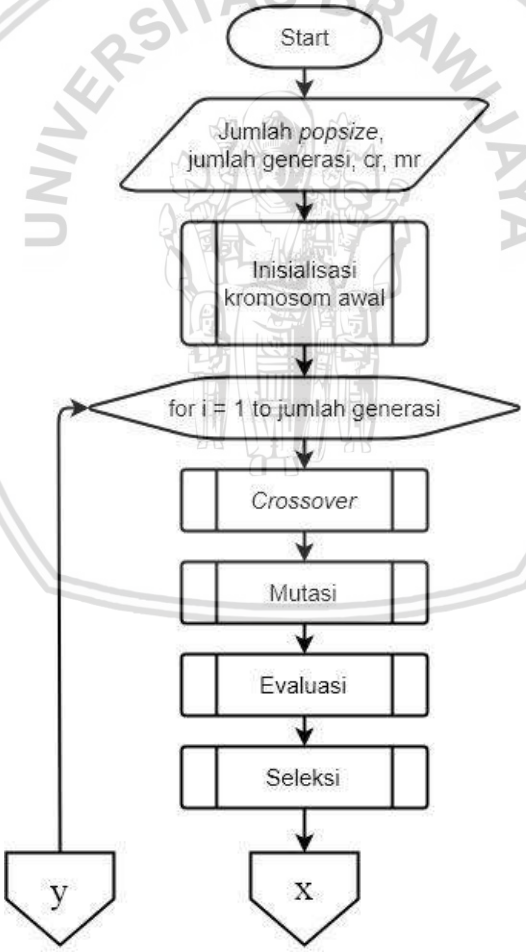
Berikut adalah penjelasan mengenai arsitektur metode ELM dalam melakukan prediksi beban listrik jangka pendek pada Gambar 4.2:

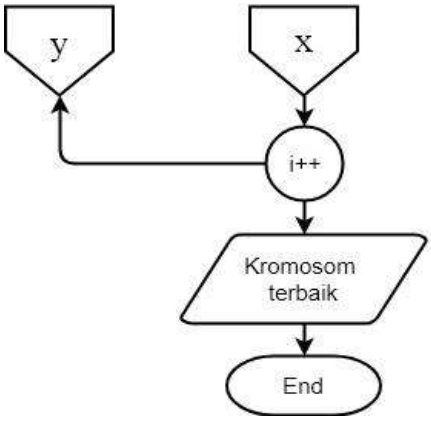
Lapisan Masukan : Lapisan ini berisi masukan yang berupa data beban sebagai data latih yang akan digunakan untuk melakukan prediksi. Terdiri dari 5 unit input dengan data beban berdasarkan jam, hari, dan minggu.

Lapisan Tersembunyi : Lapisan ini berfungsi untuk mengolah data masukan dengan menggunakan metode ELM, dengan *hidden neuron* sebanyak 3.

Lapisan *Output* : Lapisan ini merupakan lapisan yang berfungsi memberikan hasil perhitungan prediksi dengan menggunakan metode ELM.

4.3 Siklus Algoritme Genetika

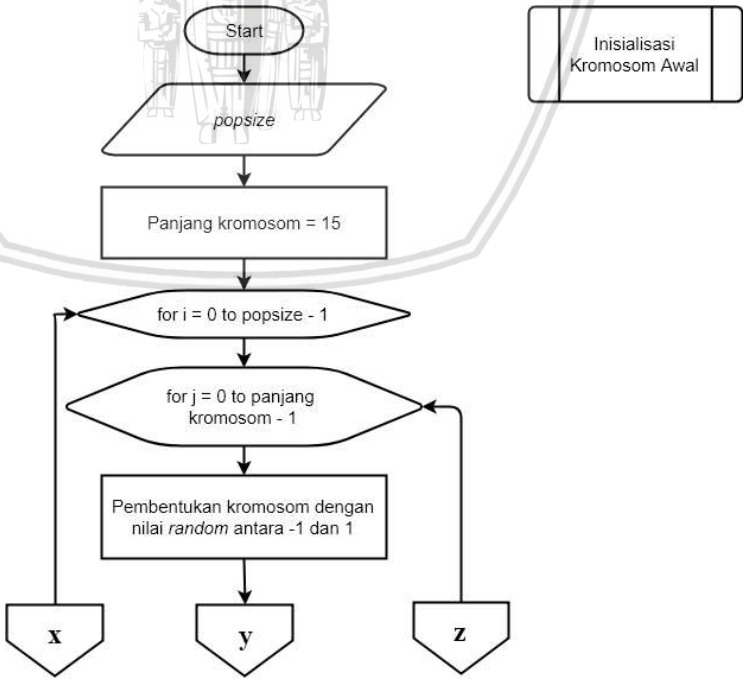


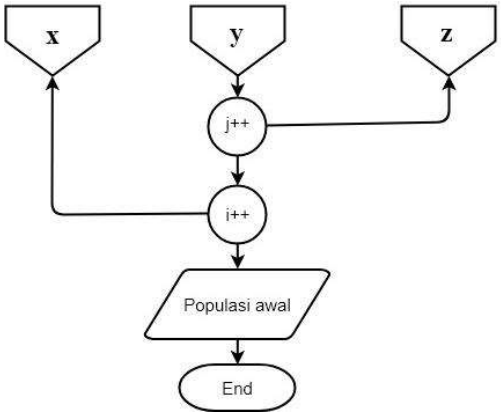


Gambar 4.2 *Flowchart* Algoritme Genetika

Pada Gambar 4.3 secara keseluruhan menjelaskan mengenai proses alur algoritme genetika. Algoritme Genetika sendiri merupakan pendeketan yang mampu mengoptimasi variabel atau factor yang berpengaruh pada sebuah metode. Pada kasus ini variabel yang akan dioptimasi adalah bobot awal dari metode ELM. Untuk proses utama dari algoritme ini terdiri dari inisialisasi kromosom awal, *crossover,* mutasi, evaluasi, dan yang terakhir adalah proses seleksi.

4.3.1 Inisialisasi Kromosom Awal



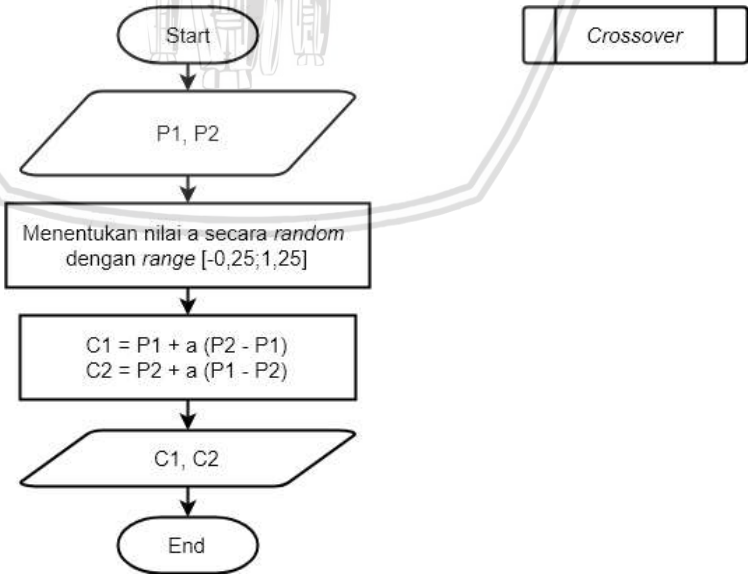


Gambar 4.3 *Flowchart* inisialisasi kromosom awal

Berdasarkan alur proses inisialisasi kromosom awal pada Gambar 4.4, dapat dijelaskan sebagai berikut:

1. Menentukan banyaknya ukuran populasi.
2. Input panjang kromosom atau banyaknya gen.
3. Membangkitkan nilai kromosom secara acak dengan *range* antara -1 sampai dengan 1 sesuai panjang dan banyaknya ukuran populasi yang sudah ditentukan, dilakukan perulangan hingga kondisi terpenuhi.
4. JIka kondisi terpenuhi yaitu kromosom yang dibangkitkan sudah sesuai dengan panjang kromosom dan banyak ukuran populasi maka akan menghasilkan sebuah populasi awal.

4.3.1 *Crossover*

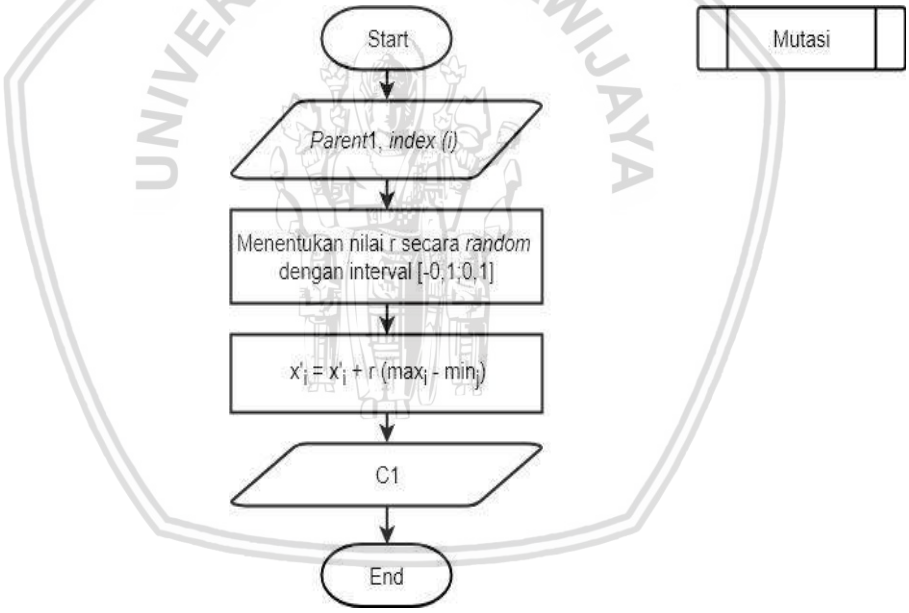


Gambar 4.4 *Flowchart Crossover*

Berdasarkan alur proses *crossover* pada Gambar 4.5, dapat dijelaskan sebagai berikut:

1. Memilih dua buah *parent* yang akan disilangkan pada proses *crossover* dari hasil pembangkitan populasi awal.
2. Membangkitkan nilai alpha secara acak dengan *range* antara -0.25 sampai dengan 1.25 sebanyak jumlah gen dalam satu kromosom
3. Melakukan pembentukan *offspring* yang banyaknya sudah ditentukan sebelumnya yaitu nilai *cr* dikalikan dengan *ukuran populasi.* Penghitungan *crossover* dilakukan dengan menggunakan metode *extended intermediate crossover* yaitu dengan menjumlahkan setiap gen pada 2 *parent* yang terpilih dengan hasil kali antara nilai alpha dan selisih gen pada *parent1* dan *parent2* ataupun sebaliknya.
4. Setelah perhitungan maka akan dihasilkan *offspring* yang akan digunakan pada proses selanjutnya.

4.3.2 Mutasi



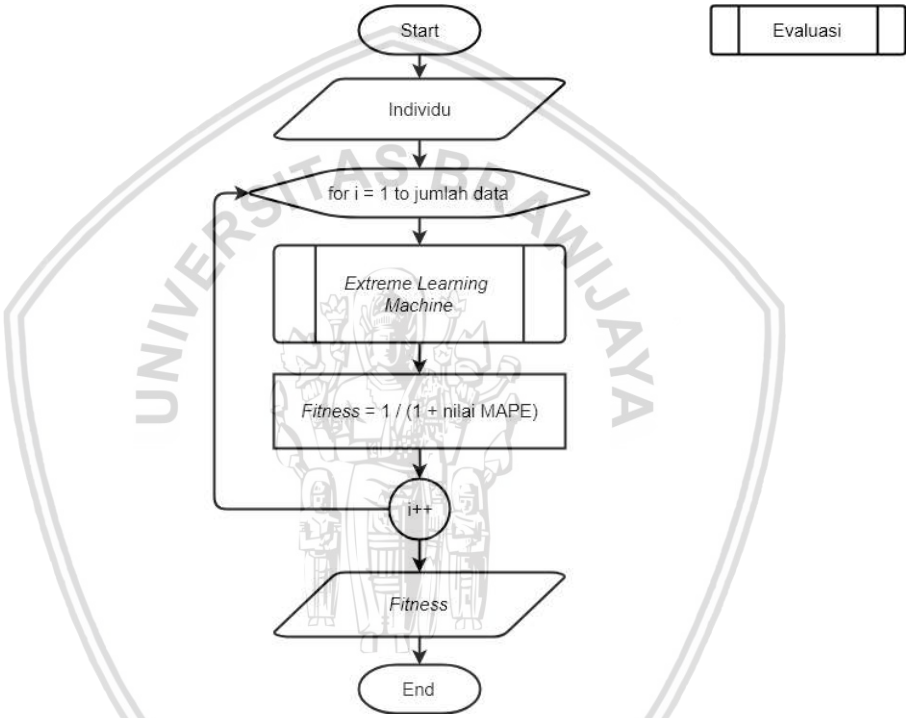
Gambar 4.5 *Flowchart* Mutasi

Berdasarkan alur proses mutasi pada Gambar 4.6, dapat dijelaskan sebagai berikut:

1. Memilih satu *parent* secara acak dari hasil pembangkitan poppulasi awal yang untuk dilakukan proses mutasi.
2. Memilih posisi satu gen secara acak pada *parent* terpilih untuk dilakukan proses mutasi.
3. Membangkitkan nilai r secara acak dengan *range* -0.1 sampai dengan 0.1.

4. Melakukan pembentukan *offspring* yang banyaknya sudah ditentukan sebelumnya yaitu nilai *mr* dikalikan dengan *ukuran populasi.* Penghitungan mutasi dilakukan dengan menggunakan metode *random mutation* yaitu dengan menjumlahkan satu gen terpilih dengan hasil kali antara nilai r dan selisih nilai maksimum minimum dari gen terpilih dengan kondisi yang sudah ditentukan.
5. Setelah perhitungan maka akan dihasilkan *offspring* yang akan digunakan pada proses selanjutnya.

4.3.3 Evaluasi

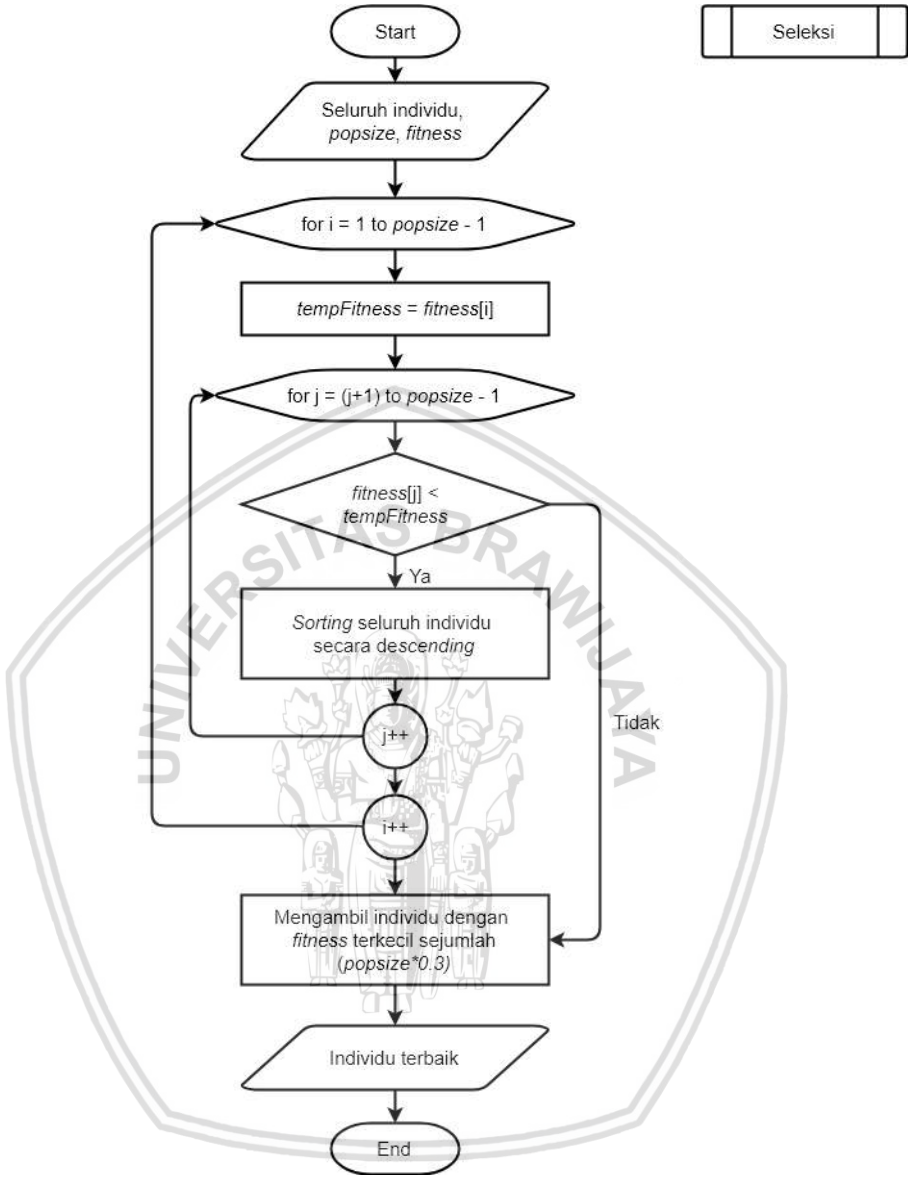


Gambar 4.6 *Flowchart* proses evaluasi

Berdasarkan alur proses evaluasi pada Gambar 4.7, dapat dijelaskan sebagai berikut:

1. Memasukkan individu gabungan yang akan dievaluasi satu persatu sejumlah *ukuran populasi* dan *offspring* dari hasil *crossover* dan mutasi.
2. Memilih satu individu yang akan digunakan sebagai bobot awal untuk melakukan prediksi beban listrik dengan menggunakan metode ELM.
3. Setelah mendapatkan hasil prediksi lalu dihitung tingkat akurasinya dengan menggunakan metode MAPE yang kemudian hasilnya akan digunakan sebagai nilai *fitness.*
4. Nilai *fitness* dari setiap individu gabungan didapatkan.

4.3.4 Seleksi

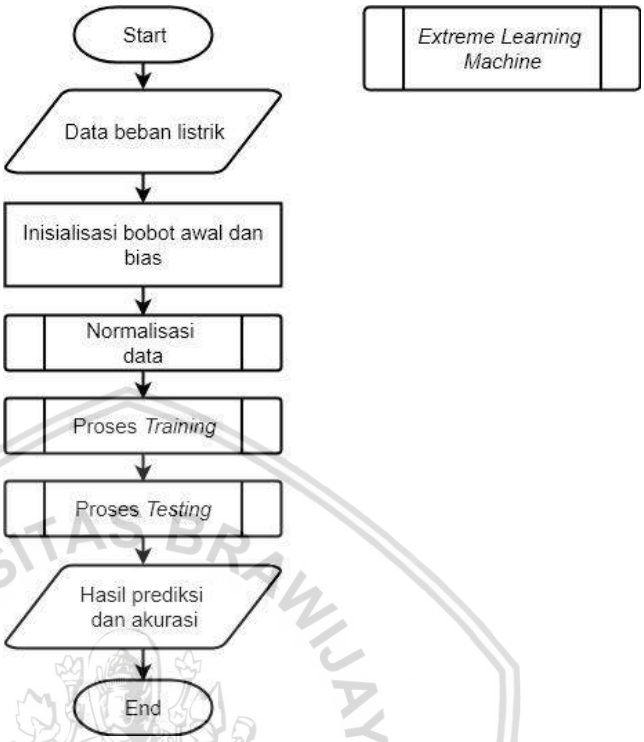


Gambar 4.7 *Flowchart* proses seleksi

Berdasarkan alur proses seleksi pada Gambar 4.8, dapat dijelaskan sebagai berikut:

1. Memasukkan individu gabungan, ukuran populasi, dan nilai *fitness* masing – masing individu.
2. Melakukan *sorting* individu berdasarkan nilai *fitness*-nya secara *ascending* yaitu dari yang paling kecil ke yang paling tinggi.
3. Mengambil individu dari nilai tertinggi sebanyak *ukuran populasi.*
4. Didapatkan individu terbaik yang lolos ke generasi selanjutnya.

4.4 Siklus *Extreme Learning Machine* (ELM)

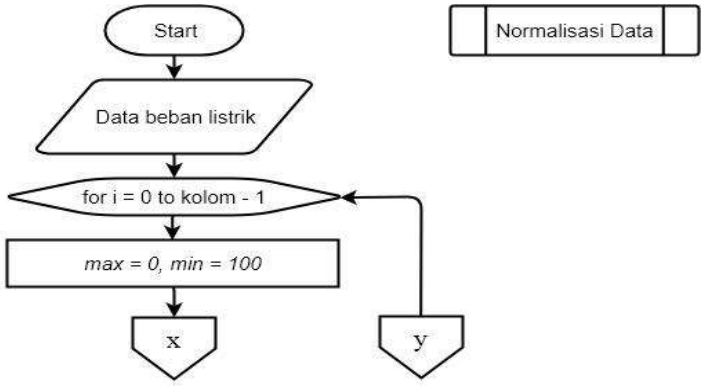


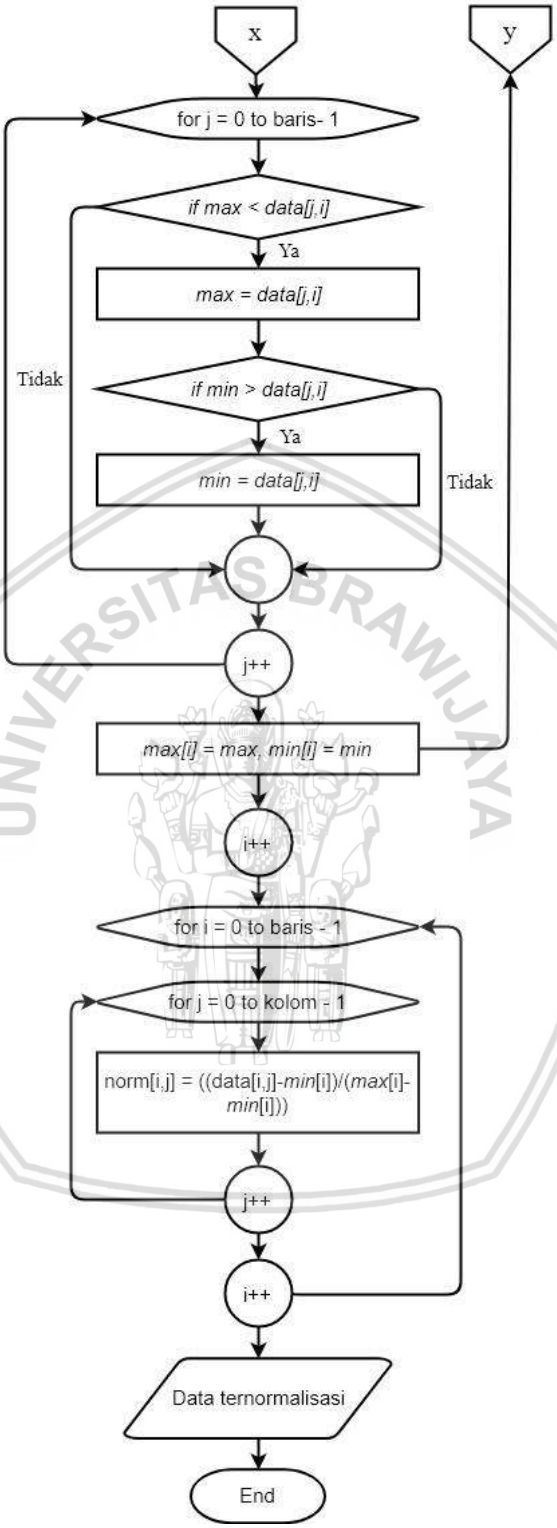
Gambar 4.8 *Flowchart* ELM

Pada Gambar 4.9 secara keseluruhan menjelaskan mengenai alur proses prediksi beban listrik menggunakan metode ELM. Yang terdiri dari beberapa proses utama yaitu normalisasi data, proses *training* dan proses *testing* untuk mendapatkan hasil prediksi.

4.4.1 Normalisasi Data

Di dalam proses ini data input memiliki range yang berbeda sehingga sebelum di proses lebih lanjut disamakan terlebih dahulu *range*-nya dengan cara dinormalisasi dengan menggunakan metode *Min-Max*. Proses normalisasi data dapat dilihat pada Gambar 4.10.





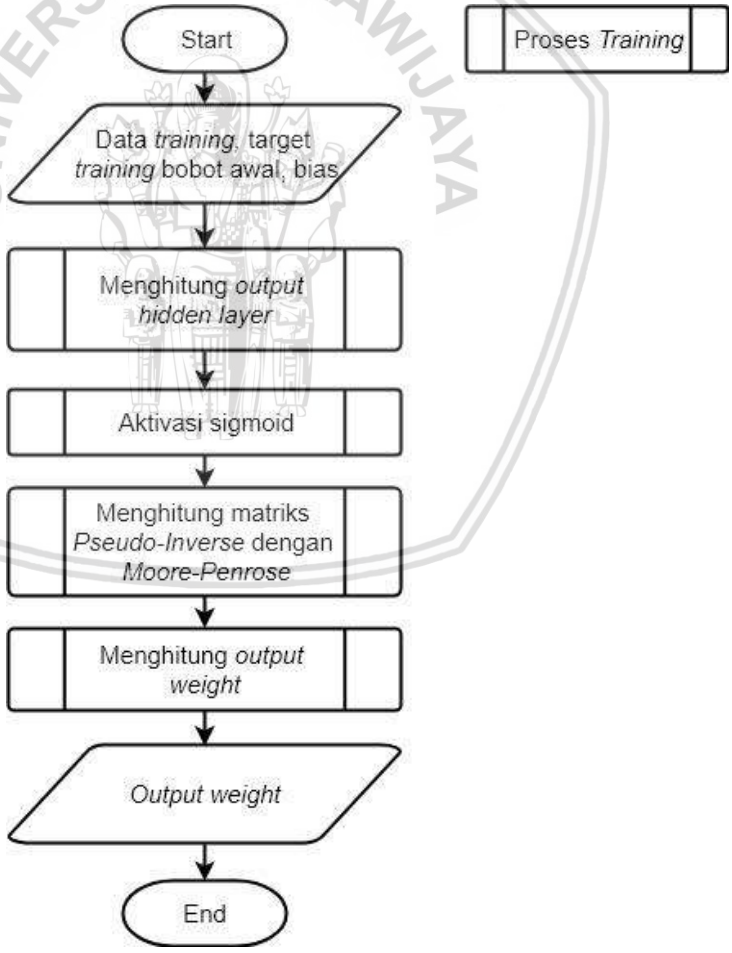
Gambar 4.9 *Flowchart* proses normalisasi

Langkah – langkah pada proses *training* metode ELM berdasarkan Gambar 4.10 akan dijelaskan sebagai berikut:

1. Memasukkan data beban listrik yang dijadikan data *training / testing* dalam file excel berekstensi .xls.
2. Mencari nilai minimum dan maksimum dari setiap parameter.
3. Melakukan penghitungan normalisasi dengan Persamaan $Data_{normi,j} = (data_{i,j} – min_i) / max_i - min_i$.
4. Didapatkan hasil data ternormalisasi dengan rentang 0-1.

4.4.2 Proses *Training*

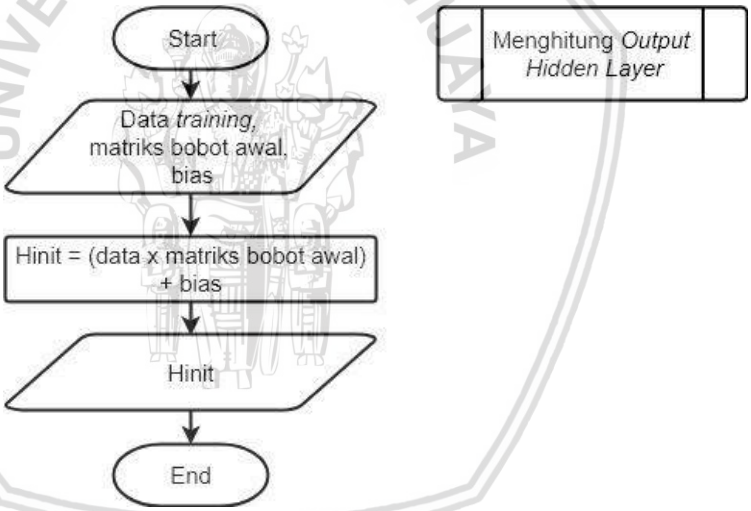
Proses *training* dilakukan untuk memperoleh nilai *output weight* dengan melewati beberapa proses utama yang terdiri dari menghitung *output hidden layer*, aktivasi sigmoid, menghitung matriks *moore-penrose pseudo inverse*, menghitung *output weight* untuk digunakan pada proses *testing*. Proses *training* dapat dilihat pada Gambar 4.11.



Gambar 4.10 *Flowchart* proses *training*

Langkah – langkah pada proses *training* metode ELM berdasarkan Gambar 4.11 akan dijelaskan sebagai berikut:

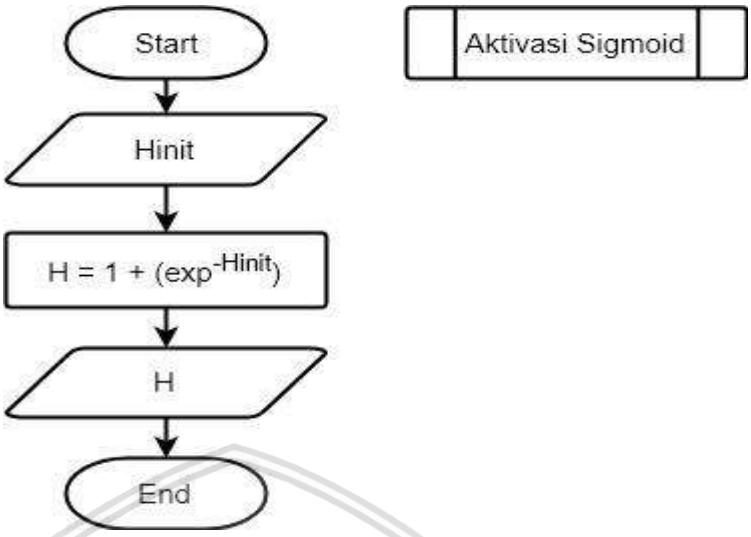
1. Memasukkan data beban listrik yang dijadikan data *training* dan sudah dinormalisasi, target *training,* bobot awal dan bias dari file excel yang berkestensi .xls.
2. Menghitung *output hidden layer* dengan menggunakan Persamaan 2.1. Sedangkan untuk diagram alir proses tersebut terdapat pada Gambar 4.12.
3. Melakukan aktivasi sigmoid dengan menggunakan Persamaan 2.2. Sedangkan untuk diagram alir proses tersebut terdapat pada Gambar 4.13.
4. Menghitung *pseudo inverse* dengan *moore-penrose* menggunakan Persamaan 2.3. Sedangkan untuk diagram alir proses tersebut terdapat pada Gambar 4.14.
5. Menghitung *output weight* dengan menggunakan Persamaan 2.4. Sedangkan untuk diagram alir proses tersebut terdapat pada Gambar 4.15.
6. Didapatkan hasil *output weight* untuk digunakan pada proses *testing.*



Gambar 4.11 *Flowchart* perhitungan *output hidden layer*

Langkah – langkah pada proses hitung *output hidden layer* metode ELM berdasarkan Gambar 4.12 akan dijelaskan sebagai berikut:

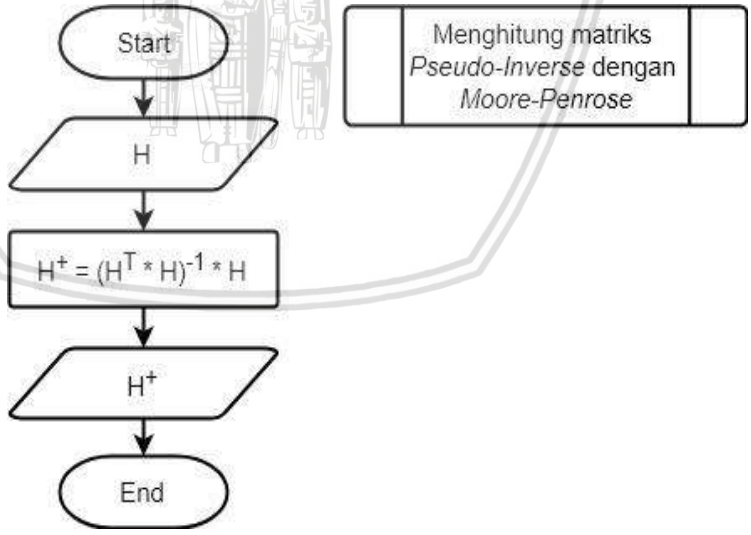
1. Memasukkan data beban listrik yang dijadikan data *training* dan sudah dinormalisasi, bobot awal dan bias dari file excel yang berkestensi .xls.
2. Menghitung *output hidden layer* dengan menggunakan Persamaan 2.1.
3. Didapatkan hasil matriks *output hidden layer* (Hinit).



Gambar 4.12 *Flowchart* aktivasi *sigmoid*

Langkah – langkah pada proses aktivasi sigmoid metode ELM berdasarkan Gambar 4.13 akan dijelaskan sebagai berikut:

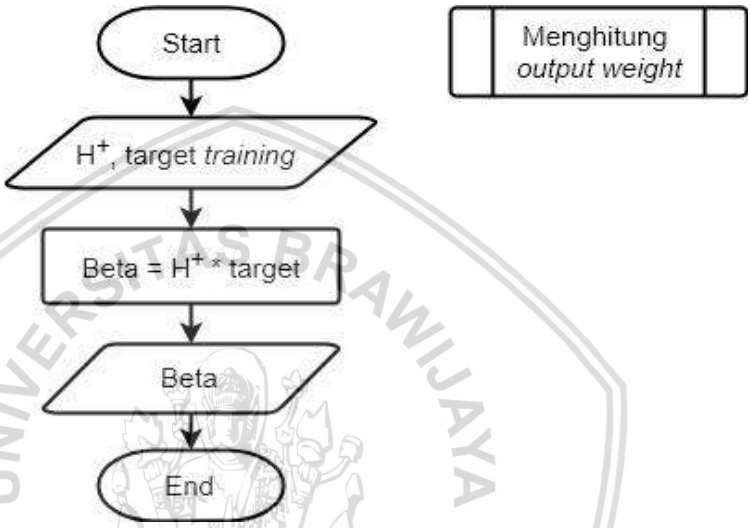
1. Memasukkan matriks Hinit dari proses sebelumnya.
2. Melakukan aktivasi sigmoid dengan menggunakan Persamaan 2.2.
3. Didapatkan hasil matriks H yang sudah diaktivasi.



Gambar 4.13 *Flowchart* perhitungan *Moore-Penrose Pseudo Inverse*

Langkah – langkah pada proses menghitung matriks *pseudo-inverse* dengan *moore-penrose* metode ELM berdasarkan Gambar 4.14 akan dijelaskan sebagai berikut:

1. Memasukkan matriks H teraktivasi.
2. Menghitung *pseudo inverse* dengan *moore-penrose* menggunakan Persamaan 2.3.
3. Didapatkan hasil matriks H^+.



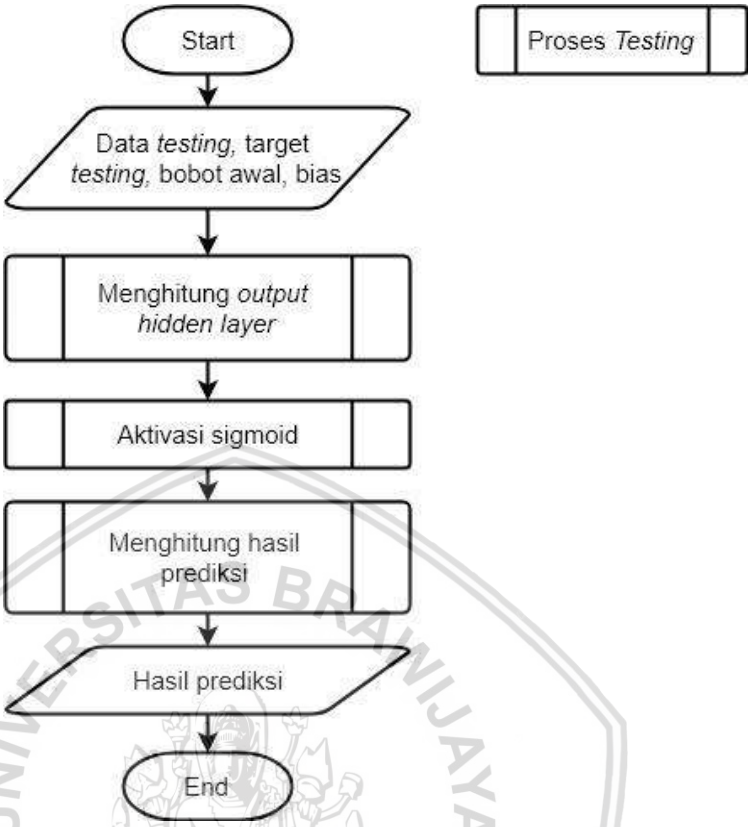
Gambar 4.14 *Flowchart* perhitungan *output weight*

Langkah – langkah pada proses menghitung *output weight* metode ELM berdasarkan Gambar 4.15 akan dijelaskan sebagai berikut:

1. Memasukkan matriks H^+ dan target dari data *training* yang sudah dinormalisasi.
2. Menghitung *output weight* dengan menggunakan Persamaan 2.4.
3. Didapatkan hasil *output weight* (β).

4.4.3 Proses *Testing*

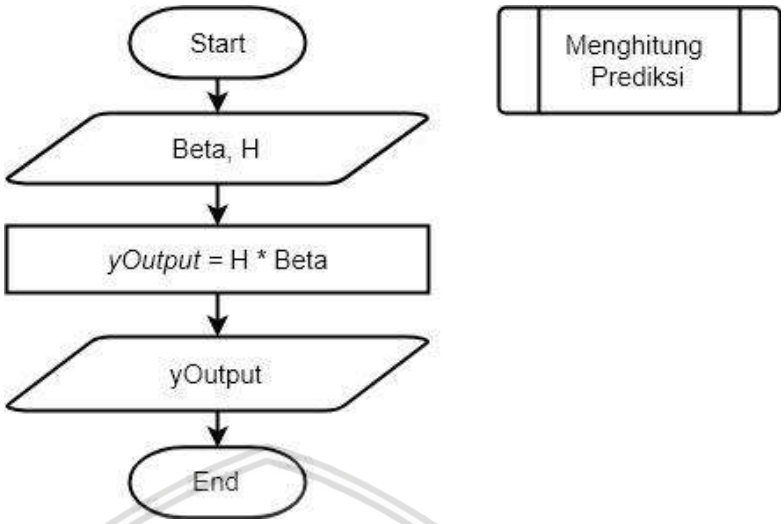
Proses *testing* dilakukan untuk memperoleh hasil prediksi beban listrik dengan melewati beberapa proses utama yang terdiri dari menghitung *output hidden layer*, aktivasi sigmoid, menghitung hasil prediksi. Proses *testing* dapat dilihat pada Gambar 4.16.



Gambar 4.15 *Flowchart* proses *testing*

Langkah – langkah pada proses *testing* metode ELM berdasarkan Gambar 4.16 akan dijelaskan sebagai berikut:

1. Memasukkan data beban listrik yang dijadikan data *testing* dan sudah dinormalisasi, target *testing,* bobot awal dan bias dari file excel yang berkestensi .xls.
2. Menghitung *output hidden layer* dengan menggunakan Persamaan 2.1. Sedangkan untuk diagram alir proses tersebut terdapat pada Gambar 4.12.
3. Melakukan aktivasi sigmoid dengan menggunakan Persamaan 2.2. Sedangkan untuk diagram alir proses tersebut terdapat pada Gambar 4.13.
4. Menghitung hasil prediksi menggunakan Persamaan 2.5. Sedangkan untuk diagram alir proses tersebut terdapat pada Gambar 4.17.
5. Didapatkan hasil prediksi.



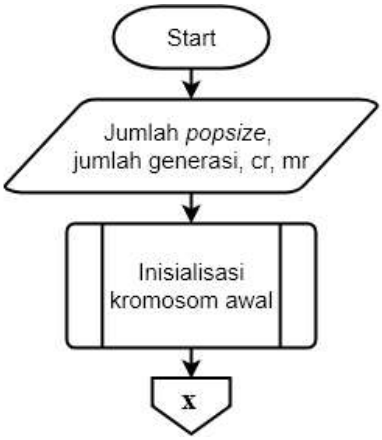
Gambar 4.16 *Flowchart* perhitungan prediksi

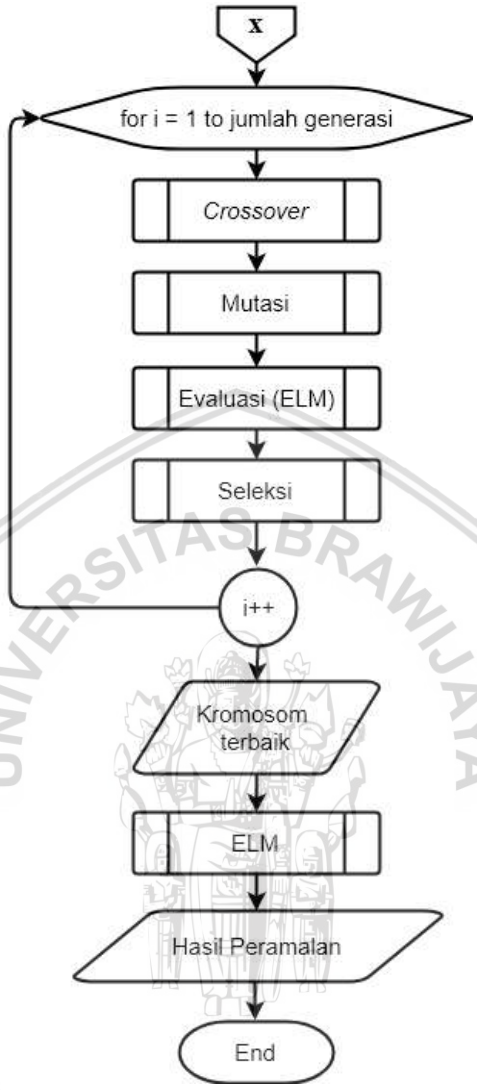
Langkah – langkah pada proses menghitung hasil prediksi metode ELM berdasarkan Gambar 4.17 akan dijelaskan sebagai berikut:

1. Memasukkan hasil *output weight* (β) dari proses training dan nilai H dari proses aktivasi sigmoid data *testing*.
2. Menghitung hasil prediksi menggunakan Persamaan 2.5.
3. Didapatkan hasil prediksi beban listrik.

4.5 Siklus Algoritme Genetika dan *Extreme Learning Machine*

Pada penelitian digunakan penggabungan 2 metode untuk melakukan prediksi beban listrik. Metode tersebut antara lain algoritme genetika dan *Extreme Learning Machine*. Algoritme genetika bertugas untuk melakukan optimasi terhadap bobot awal yang terdapat pada metode ELM. Adapun siklus dari penggabungan kedua metode ini dapat dilihat dari alur pada Gambar 4.17.





Gambar 4.17 Siklus GA-ELM

4.6 Perhitungan Manual

Perhitungan manual digunakan untuk memberikan gambaran terkait siklus algoritme proses perhitungan sebelum diimplementasikan kedalam sistem. Manualisasi yang ditampilkan hanya sebagian tidak secara keseluruhan. Dataset yang digunakan pada proses perhitungan manualisasi terdiri dari:

1. 35 data beban yang digunakan sebagai data *training.*
2. 35 data beban yang digunakan sebagai data *testing.*
3. Terdapat 5 input yang terdiri dari data beban minggu ke-1, minggu ke-2, minggu ke-3, minggu ke-4, minggu ke-5.
4. Terdapat 1 output yaitu data beban minggu ke-6 sebagai data target.

4.5.1 Representasi Kromosom Awal

Pembangkitan nilai kromosom awal pada algoritme genetika dilakukan secara acak. Pada penelitian ini kromosom yang dibangkitkan merupakan bobot yang akan digunakan pada perhitungan prediksi dengan metode ELM. Oleh karena itu *range* kromosom yang dibangkitkan yaitu antara -1 sampai 1. Adapun parameter algoritme genetika yang digunakan sebagai berikut:

- Ukuran populasi : 4
- *Crossover rate* (cr) : 0.4
- *Mutation rate* (mr) : 0.2

Didapatkan 4 individu yang sudah dibangkitkan secara *random* dan masing – masing terdiri dari 15 gen yang terdiri dari 15 bobot yang akan digunakan sebagai bobot awal pada metode ELM. Hal tersebut ditunjukkan pada Tabel 4.1.

Tabel 4.1 Representasi Kromosom Awal

P1	0.44	0.41	-0.03	0.98	-0.45	-0.21	-0.02	0.24	-0.25	0.56	0.07	0.79	0.32	0.39	-0.62
P2	0.68	0.71	0.68	0.45	-0.78	-0.55	0.61	0.13	0.48	-0.65	0.02	-0.61	-0.68	0.8	0.32
P3	-0.95	-0.42	0.31	-0.69	-0.37	0.37	0.1	0.52	0.88	-0.08	0.59	0.43	-0.28	-0.63	0.75
P4	-0.73	-0.76	-0.39	-0.66	-0.23	0.79	0.34	-0.95	0.62	-0.16	-0.02	0.68	0.75	-0.94	0.65

4.5.2 *Crossover* dan Mutasi

Setelah proses inisialisasi kromosom awal, selanjutnya adalah proses reproduksi yang terdiri dari *crossover* dan mutasi. Metode *crossover* yang digunakan adalah *extended intermediate crossover,* sedangkan metode mutasi yang digunakan adalah *random mutation*.

- ***Crossover***

Proses *crossover* merupakan proses pemilihan dua *parent* yang kemudian keduanya akan disilangkan untuk menghasilkan keturunan (*offspring*). Dalam satu kali eksekusi proses *crossover* mampu menghasilkan 2 *offspring* sekaligus. Akan tetapi banyaknya *offspring* yang dihasilkan dapat ditentukan oleh nilai *crossover rate*. Pada penelitian ini *crossover rate* yang digunakan sebesar 0,4. Untuk menentukan banyaknya *offspring* yang harus dihasilkan maka cr dikalikan dengan ukuran populasi.

Offspring = cr x ukuran populasi

= 0.4 x 4

= 1.6 ≈ 2

Kemudian pada perhitungan menggunakan metode *extended intermediate crossover* membutuhkan bantuan 1 variabel lagi yaitu a atau *alpha*. Variabel tersebut dibangkitkan secara *random* dengan *range* [-0.25; 1.25]. Seperti pada Tabel 4.2 berikut.

Tabel 4.2 Pembangkitan nilai *alpha*

a	-0.17	0.8	0.37	-0.24	0.92	1.19	0.08	1.07	0.43	0.69	0.32	0.03	0.54	0.6	0.6

Setelah membangkitkan nilai *alpha* kemudian memilih 2 *parent* P1 dan P2 yang telah dipilih secara acak untuk disilangkan dengan menggunakan Persamaan 2.7.

Tabel 4.3 *Parent* terpilih (P1 dan P2)

P1	0.44	0.41	-0.03	0.98	-0.45	-0.21	-0.02	0.24	-0.25	0.56	0.07	0.79	0.32	0.39	-0.62
P2	-0.95	-0.42	0.31	-0.69	-0.37	0.37	0.1	0.52	0.88	-0.08	0.59	0.43	-0.28	-0.63	0.75

C1 = P1 + a (P2 – P1)

= 0.44 + (-0.17)(-0.95 -0.44)

= 0.676

C2 = P2 + a (P1 – P2)

= -0.95 + (-0.17)(0.44 – (- 0.95))

= -1.19

Hasil *crossover* dapat dilihat pada Tabel 4.4.

Tabel 4.4 Hasil *Crossover*

C1	0.676	-0.25	0.096	1.381	-0.38	0.48	-0.01	0.54	0.236	0.118	0.236	0.779	-0	-0.22	0.202
C2	-1.19	0.244	0.184	-1.09	-0.44	-0.32	0.09	0.22	0.394	0.362	0.424	0.441	0.044	-0.02	-0.07

- **Mutasi**

Setelah melakukan proses *crossover* maka proses selanjutnya adalah mutasi. Proses mutasi adalah proses menghasilkan keturunan dari satu *parent* yang dipilih secara *random.* Dengan menggunakan metode *random mutation* selain memilih satu *parent* secara acak, setelah itu memilih satu gen yang akan dimutasi. Pada metode ini juga membutuhkan bantuan variabel lain yaitu r yang dibangkitkan secara *random* dengan *range* [-0.1; 0.1] dan nilai batasan maksimum minimum untuk gen terpilih. Pada proses ini *mutation rate* yang digunakan sebesar 0.2.

Offspring = mr x ukuran populasi

= 0.2 x 4

= 0.8 ≈ 1

r = -0.09 max = 3 min = 0

Setelah itu melakukan proses mutasi dengan menggunakan Persamaan 2.8.

Tabel 4.5 *Parent* terpilih (P3)

P3	0.68	0.71	0.68	0.45	-0.78	-0.55	0.61	0.13	0.48	-0.65	0.02	-0.61	-0.68	0.8	0.32

Didapatkan parent terpilih yaitu P3 dan indeks yang dipilih secara acak yaitu 5, maka didapatkan hasil mutasi sebagai berikut:

x'_5 = -0.78 + -0.09(3 – 0)

= -0.78 + -0.27

= -0.83

Hasil mutasi dapat dilihat pada Tabel 4.6.

Tabel 4.6 Hasil mutasi

C3	0.68	0.71	0.68	0.45	-0.83	-0.55	0.61	0.13	0.48	-0.65	0.02	-0.61	-0.68	0.8	0.32

Hasil *offspring* dari proses *crossover* dan mutasi tersebut akan digabungkan dengan seluruh *parent* menjadi individu gabungan seperti pada Tabel 4.7.

Tabel 4.7 Individu gabungan

P1	0.44	0.41	-0.03	0.98	-0.45	-0.21	-0.02	0.24	-0.25	0.56	0.07	0.79	0.32	0.39	-0.62
P2	0.68	0.71	0.68	0.45	-0.78	-0.55	0.61	0.13	0.48	-0.65	0.02	-0.61	-0.68	0.8	0.32
P3	-0.95	-0.42	0.31	-0.69	-0.37	0.37	0.1	0.52	0.88	-0.08	0.59	0.43	-0.28	-0.63	0.75
P4	-0.73	-0.76	-0.39	-0.66	-0.23	0.79	0.34	-0.95	0.62	-0.16	-0.02	0.68	0.75	-0.94	0.65
C1	0.676	-0.25	0.096	1.381	-0.38	0.48	-0.01	0.54	0.236	0.118	0.236	0.779	-0	-0.22	0.202
C2	-1.19	0.244	0.184	-1.09	-0.44	-0.32	0.09	0.22	0.394	0.362	0.424	0.441	0.044	-0.02	-0.07
C3	-0.22	0.15	0.01	-0.04	-0.83	-0.2	0.29	-0.14	-0.26	0.17	0.35	0.37	0.05	-0.2	0.16

4.5.3 Penghitungan ELM

Setelah mendapatkan hasil *offspring* langkah selanjutnya adalah melakukan perhitungan prediksi beban listrik dengan menggunakan metode ELM. Tujuannya adalah untuk mendapatkan nilai *fitness* dari masing – masing individu.

4.5.3.1 Normalisasi Data

Tabel 4.8 Data *training* aktual

No	X1	X2	X3	X4	X5	T
1	987.95	999.43	1027.92	1028.72	1008.65	1011.85
2	957.31	968.72	1001.88	1008.82	988.51	986.97
3	935.47	943.14	940.07	938.67	938.29	940.84
4	854.97	874.84	885.38	873.41	890.05	871.93
...	...	...	...	...	...	...
33	944.67	912.3	924.96	805.3	895.37	917.12
34	881	854.27	865.64	728.7	840.09	869.38
35	820.8	785.45	783.54	650.2	770.1	795.88

Tabel 4.9 Data *testing* aktual

No	X1	X2	X3	X4	X5	T
1	999.43	1027.92	1028.72	1008.65	1011.85	944.87
2	968.72	1001.88	1008.82	988.51	986.97	934.73
3	943.14	940.07	938.67	938.29	940.84	877.91
4	874.84	885.38	873.41	890.05	871.93	798.5
...	...	...	...	...	...	...
33	912.3	924.96	805.3	895.37	917.12	875.88
34	854.27	865.64	728.7	840.09	869.38	805.5
35	785.45	783.54	650.2	770.1	795.88	736

Sebelum masuk pada proses *training* dan *testing* data *training* dan *testing* yang masing – masingnya berjumlah 35 data yang terdapat pada Tabel 4.8 dan Tabel 4.9 dinormalisasi terlebih dahulu untuk mengurangi perbedaan *range* data. Berikut contoh perhitungan manual normalisasi data.

$$'X_i = \frac{data_{Xi} - data_{min}}{data_{max} - data_{min}}$$

$$'X_1 = \frac{987.95 - 757.52}{1004.9 - 757.52}$$

$$= 0.9316$$

Hasil normalisasi data *training* dapat dilihat pada Tabel 4.10.

Tabel 4.10 Normalisasi data *training*

No	X1	X2	X3	X4	X5	T
1	0.931633	0.875138	0.963426	1	0.912706	0.970341
2	0.807755	0.74954	0.878239	0.94743	0.848282	0.868839
3	0.719455	0.644922	0.676034	0.7621	0.68764	0.680646
4	0.393992	0.365588	0.497121	0.58969	0.533331	0.399518
...	...	...	...	...	...	...
33	0.756651	0.518793	0.626603	0.40975	0.550349	0.583877
34	0.499232	0.281461	0.432544	0.20739	0.373521	0.389115
35	0.255842	0	0.163962	0	0.149639	0.089262

Hasil normalisasi data *testing* dapat dilihat pada Tabel 4.11.

Tabel 4.11 Normalisasi data *testing*

No	X1	X2	X3	X4	X5	T
1	0.875138	0.963426	1	0.91271	0.970341	0
2	0.74954	0.878239	0.947427	0.84828	0.86884	0.106061
3	0.644922	0.676034	0.7621	0.68764	0.680646	0.878788
4	0.365588	0.497121	0.589691	0.53333	0.399519	0.69697
...	...	...	...	...	...	...
33	0.518793	0.626603	0.409754	0.55035	0.583877	0.469697
34	0.281461	0.432544	0.207387	0.37352	0.389116	0.984848
35	0	0.163962	0	0.14964	0.089262	0.651515

4.5.3.2 Proses *Training*

Proses yang pertama adalah proses *training*, dengan menggunakan data *training* sebanyak 35 data, total *hidden neuron* sebanyak 3 dan fungsi aktivasi sigmoid biner. Langkah – langkah proses *training* pada ELM adalah sebagai berikut:

Langkah 1 Inisialisasi Bobot Awal dan Bias

Langkah pertama yang harus dilakukan adalah inisialisasi parameter yang terdiri dari bobot awal yang dioptimasi menggunakan algoritme genetika, bias, dan data beban listrik. Bobot awal didapatkan dari salah satu individu gabungan (lihat Tabel 4.7). Ukuran matriks yang dibutuhkan adalah 3x5. Oleh karena itu matriks individu gabungan perlu diubah terlebih dahulu dari matriks 1x15 menjadi matriks 3x5 seperti contoh individu gabungan P1 pada Tabel 4.12 dan hasil transformasi individu P1 ada pada Tabel 4.13.

Tabel 4.12 Individu P1

P1	0.44	0.41	-0.03	0.98	-0.45	-0.21	-0.02	0.24	-0.25	0.56	0.07	0.79	0.32	0.39	-0.62

Tabel 4.13 Bobot awal

w	1	2	3	4	5
1	0.44	0.41	-0.03	0.98	-0.45
2	-0.21	-0.02	0.24	-0.25	0.56
3	0.07	0.79	0.32	0.39	-0.62

Nilai bias dibangkitkan secara *random* dengan *range* [0,1] serta ukuran matriksnya adalah 3x1. Nilai bias yang dibangkitkan dapat dilihat pada Tabel 4.14.

Tabel 4.14 Bias

b	1
1	0.14
2	0.2
3	0.57

Langkah 2 **Menghitung *Output Hidden Layer***

Dengan menggunakan Persamaan 2.1 data beban listrik yang merupakan data *input* dengan ordo 35x3 dikalikan dengan bobot awal transpose (w^T) yang ordonya menjadi 5x3 kemudian ditambah dengan nilai bias. Matriks *output hidden layer* (H_{init}) akan memiliki ordo 35x3. Hasil bobot awal yang telah ditranspose terdapat pada Tabel 4.15.

Tabel 4.15 Bobot *transpose* (w^T)

w^T	1	2	3
1	0.44	-0.21	0.07
2	0.41	-0.02	0.79
3	-0.03	0.24	0.32
4	0.98	-0.25	0.39
5	-0.45	0.56	-0.62

$H_{init\ 1,1} = X_{training(1)} . w_{(1)}^T + b_{(1)}$

= ((0.9316*0.44) + (0.8751*0.41) + (0.9634*-0.03) + (1*0.98) + (0.9127*-0.45)) + 0.14

= 1.449105

Hasil perhitungan *output hidden layer* terdapat pada Tabel 4.16.

Tabel 4.16 Hasil *output hidden layer*

H_{init}	1	2	3
1	1.449105	0.479192	1.458992
2	1.323127	0.464339	1.343277
3	1.138117	0.392818	1.217064
4	0.786233	0.380502	0.944787
...	...	...	...
34	0.497239	0.350667	0.815012
35	0.180314	0.269422	0.547601

Langkah 3 Hitung Aktivasi Sigmoid

Setelah mendapatkan *output hidden layer* dengan data *training* maka langkah selanjutnya adalah melakukan aktivasi sigmoid biner pada matriks *output hidden layer* dengan menggunakan Persamaan 2.2 seperti dibawah.

$$H_{1,1} = \frac{1}{(1 + exp^{-Hinit})}$$
$$= \frac{1}{(1 + exp^{-(1.449105)})}$$
$$= 0.8099$$

Hasil perhitungan aktivasi sigmoid biner terdapat pada Tabel 4.17.

Tabel 4.17 Hasil aktivasi *sigmoid*

H	1	2	3
1	0.809861	0.617557	0.811378
2	0.789702	0.614043	0.793028
3	0.757334	0.596961	0.771546
4	0.687022	0.593994	0.720066
...	...	...	...
34	0.62181	0.586779	0.693177
35	0.544957	0.566951	0.633579

Langkah 4 Menghitung Matriks *Moore-Penrose Pseudo Inverse*

Dengan menggunakan Persamaan 2.3 untuk menghitung matriks *Moore-Penrose* dengan *Pseudo Inverse*. Proses awalnya adalah dimulai dengan melakukan *transpose* pada matriks *output hidden layer* yang sudah diaktivasi menggunakan fungsi aktivasi *sigmoid* biner. Hasil transpose dari matriks H terdapat pada Tabel 4.18.

Tabel 4.18 Matriks H *transpose* (H^T)

H^T	1	2	3	4	5	...	34	35
1	0.809861	0.789702	0.757334	0.687022	0.628906	...	0.62181	0.544957
2	0.617557	0.614043	0.596961	0.593994	0.569817	...	0.586779	0.566951
3	0.811378	0.793028	0.771546	0.720066	0.668016	...	0.693177	0.633579

Setelah itu matriks H^T dikalikan dengan matriks H dan menghasilkan matriks berukuran 3x3. Berikut contoh perhitungannya.

$(H^T H)_{1,1}$ = (0.809861 * 0.809861) + (0.789702 *0.789702) + (0.757334 * 0.757334) + (0.687022 * 0.687022) + (0.628906 * 0.628906) + ... + (0.62181 * 0.62181) + (0.544957 * 0.544957)

= 18.6639

Hasil perkalian matriks H transpose dengan matriks H terdpat pada Tabel 4.19.

Tabel 4.19 Hasil perkalian H^T dengan H

$H^T H$	1	2	3
1	18.6639	14.99813	19.45439
2	14.99813	12.1321	15.66044
3	19.45439	15.66044	20.29671

Selanjutnya dari hasil perkalian tersebut dihitung matriks inversnya. Berikut adalah contoh perhitungan invers matriks menggunakan Persamaan Operasi Baris Elementer (OBE).

a. Membuat matriks indentitas (*I*)

$$[H^T H \mid I] = \begin{bmatrix} 18.664 & 14.998 & 19.454 \\ 14.998 & 12.132 & 15.66 \\ 19.454 & 15.66 & 20.297 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

b. B1 ÷ 18.664 → B1

$$[H^T H \mid I] = \begin{bmatrix} 1 & 0.803 & 1.042 \\ 14.998 & 12.132 & 15.66 \\ 19.454 & 15.66 & 20.297 \end{bmatrix} \begin{bmatrix} 0.053 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

c. B2 – (14.998 * B1) → B2

$$[H^T H \mid I] = \begin{bmatrix} 1 & 0.803 & 1.042 \\ 0 & 0.08 & 0.032 \\ 19.454 & 15.66 & 20.297 \end{bmatrix} \begin{bmatrix} 0.053 & 0 & 0 \\ -0.794 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

d. B3 – (19.454 * B1) → B3

$$[H^TH \mid I] = \begin{bmatrix} 1 & 0.803 & 1.042 \\ 0 & 0.08 & 0.032 \\ 0 & 0.038 & 0.025 \end{bmatrix} \begin{bmatrix} 0.053 & 0 & 0 \\ -0.794 & 1 & 0 \\ -1.031 & 0 & 1 \end{bmatrix}$$

e. B2 ÷ 0.08 → B2

$$[H^TH \mid I] = \begin{bmatrix} 1 & 0.803 & 1.042 \\ 0 & 1 & 0.4 \\ 0 & 0.038 & 0.025 \end{bmatrix} \begin{bmatrix} 0.053 & 0 & 0 \\ -9.925 & 12.5 & 0 \\ -1.031 & 0 & 1 \end{bmatrix}$$

f. B1 – (0.803 * B2) → B1

$$[H^TH \mid I] = \begin{bmatrix} 1 & 0 & 0.72 \\ 0 & 1 & 0.4 \\ 0 & 0.038 & 0.025 \end{bmatrix} \begin{bmatrix} 8.022 & -10.03 & 0 \\ -9.925 & 12.5 & 0 \\ -1.031 & 0 & 1 \end{bmatrix}$$

g. B3 – (0.038 * B2) → B3

$$[H^TH \mid I] = \begin{bmatrix} 1 & 0 & 0.72 \\ 0 & 1 & 0.4 \\ 0 & 0 & 0.009 \end{bmatrix} \begin{bmatrix} 8.022 & -10.03 & 0 \\ -9.925 & 12.5 & 0 \\ -0.653 & -0.475 & 1 \end{bmatrix}$$

h. B3 ÷ (0.009) → B3

$$[H^TH \mid I] = \begin{bmatrix} 1 & 0 & 0.72 \\ 0 & 1 & 0.4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 8.022 & -10.03 & 0 \\ -9.925 & 12.5 & 0 \\ -72.5 & -52.7 & 111.1 \end{bmatrix}$$

i. B1 – (0.72 * B3) → B1

$$[H^TH \mid I] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0.4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 60.22 & 27.914 & -79.99 \\ -9.925 & 12.5 & 0 \\ -72.5 & -52.7 & 111.1 \end{bmatrix}$$

j. B2 – (0.4 * B3) → B2

$$[H^TH \mid I] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 60.22 & 27.914 & -79.99 \\ 19.07 & 33.58 & -44.44 \\ -72.5 & -52.7 & 111.1 \end{bmatrix}$$

Berikut adalah hasil perhitungan invers matriks H^TH pada Tabel 4.20.

Tabel 4.20 Matriks H^TH *invers*

$(H^TH)^{-1}$	1	2	3
1	60.22	27.914	-79.99
2	19.07	33.58	-44.44
3	-72.5	-52.7	111.1

Selanjutnya adalah hasil perhitungan manual matriks *Moore-Penrose Pseudo Inverse* pada Tabel 4.21.

$H^{+} = (H^{T}.H)^{-1}.H^{T}$

Tabel 4.21 Matriks *Moore-Penrose Pseudo-Inverse*

H+	1	2	3	4	...	34	35
1	2.186609	2.196119	1.328383	0.479916	...	-2.14418	-3.09838
2	0.381318	0.60146	0.370905	0.90741	...	0.518373	0.811255
3	-2.3501	-2.52998	-1.52142	-1.12466	...	1.689383	2.375065

Langkah 5 Menghitung Bobot *Output*

Langkah selanjutnya adalah menghitung bobot *output* (β) yang dihasilkan dari perkalian matriks *moore-penrose pseudo inverse* dengan data target seperti pada Persamaan 2.4. Matriks data target dapat dilihat pada Tabel 4.22.

Tabel 4.22 Matriks data target (T)

T
0.970341
0.868839
0.680646
...
0.389115
0.089262

Berikut adalah hasil perhitungan manual matriks bobot *output* dapat dilihat pada Tabel 4.23:

β = (2.186609 * 0.970341) + (2.196119 * 0.868839) + (1.328383 * 0.680646) + ... + (-2.14418 * 0.389115) + (-3.09838 * 0.089262)

= 5.8592

Tabel 4.23 Hasil bobot *output*

β	1
1	5.8592
2	-0.9936
3	-4.1562

4.5.3.3 Proses *Testing*

Pada proses *testing* langkah yang dilakukan hampir sama yaitu menghitung *output hidden layer* dan aktivasi sigmoid. Perbedaannya adalah pada proses *testing*

tidak perlu menghitung matriks *moore-penrose* dan bobot *output* tetapi menggunakan bobot *output* yang sudah dihitung pada proses *training* untuk menghitung hasil prediksi.

Langkah 1 Menghitung *Output Hidden Layer*

Dengan menggunakan *Bobot Awal* dan nilai bias yang sama, serta data *testing* sebanyak 35. Berikut contoh perhitungan manual dan hasil dari *output hidden layer* yang terdapat pada Tabel 4.24.

$H_{init\ 1,1} = X_{testing(1)} . w_{(1)}^T + b_{(1)}$

= ((0.8751*0.44) + (0.9634*0.41) + (1*-0.03) + (0.913*0.98) + (0.9703*-0.45)) + 0.14

= 1.347863

Tabel 4.24 Hasil *output hidden layer* (Proses *Testing*)

H_{init}	1	2	3
1	1.347863	0.552167	1.46671
2	1.241791	0.526894	1.411602
3	1.045673	0.443201	1.239262
4	0.829869	0.345208	1.137316
...	...	...	...
34	0.625912	0.30654	0.902197
35	0.313702	0.209298	0.702547

Langkah 2 Hitung Aktivasi *Sigmoid*

Berikut adalah contoh perhitungan manual dan hasil aktivasi sigmoid yang terdapat pada Tabel 4.25.

$H_{1,1} = \frac{1}{(1 + exp^{-Hinit})}$

$= \frac{1}{(1 + exp^{-(1.347863)})}$

= 0.79378

Tabel 4.25 Hasil aktivasi *sigmoid* (Proses *Testing*)

H	1	2	3
1	0.79378	0.634638	0.812557
2	0.775876	0.628758	0.804019
3	0.739943	0.609022	0.775436
4	0.696327	0.585455	0.757186
...	...	...	...
34	0.651562	0.57604	0.711401
35	0.577789	0.552134	0.668752

Langkah 3 Menghitung Hasil Prediksi

Cara menghitung hasil prediksi adalah dengan mengalikan matriks hasil aktivasi *sigmoid* data *testing* dengan matriks bobot *output* yang didapatkan pada proses *training* seperti pada Persamaan 2.6. Berikut adalah contoh perhitungan manual dan hasil prediksi yang terdapat pada Tabel 4.26.

$\hat{Y}$ = Hβ

$\hat{Y}_{1,1}$ = (0.79378*5.8592) + (0.634638*-0.9936) + (0.812557*-4.1562)

= 0.6432

Tabel 4.26 Hasil prediksi

$\hat{Y}$
0.6432
0.5797
0.5075
0.3512
...
0.2886
0.0573

Langkah 4 Hitung MAPE

Langkah terakhir adalah menghitung akurasi prediksi dengan menggunakan metode MAPE (*Mean Absolute Precentage Error*). Akan tetapi sebelum menghitung tingkat akurasi data hasil prediksi didenormalisasi terlebih dahulu. Dengan menggunakan rumus sebagai berikut.

X_i = $\hat{Y}_{Xi}$ (($data_{max}$ – $data_{min}$) + $data_{min}$)

= 0.6432 ((1033.8-702.72) + 702.72)

= 915.69

Hasil perhitungan denormalisasi dapat dilihat pada Tabel 4.27.

Tabel 4.27 Denormalisasi data

$\hat{Y}_{denorm}$
915.69
894.64
870.76
819.01
...
798.27
721.7

Setelah itu hitung akurasi prediksi dengan menggunakan Persamaan 2.9 seperti dibawah.

MAPE = $\left| \frac{100}{n} \Sigma_{t=1}^{n} \left(\frac{Y-\hat{Y}}{Y} \right) \right|$

$= \frac{100}{35} \left(\left(\frac{944.87-915.69}{944.87} \right) + \left(\frac{934.73-894.64}{934.73} \right) + \left(\frac{877.91-870.76}{877.91} \right) + \right.$

$\left. \ldots + \left(\frac{805.5-798.27}{805.5} \right) + \left(\frac{736-721.7}{736} \right) \right)$

$= \frac{100}{35} \; x \; 1.699$

= 4.8535

4.5.4 Evaluasi dan Seleksi

Setelah didapatkan nilai akurasi dari masing – masing individu, nilai tersebut merupakan nilai *fitness* yang akan digunakan sebagai bahan evaluasi dari tiap individu. Adapun nilai *fitness* dari masing – masing individu yang didapat dari hasil perhitungan ELM dapat dilihat pada Tabel 4.28.

f(x) = $\frac{1}{(1+MAPE(x)}$

f(P1) = $\frac{1}{(1+MAPE(P1)}$

= 0.170838683

Tabel 4.28 Hasil evaluasi

Individu	MAPE	*Fitness*
P1	4.8535	0.170838683
P2	1.21411	0.45164842
P3	1.19293	0.456011738
P4	0.61364	0.619717504
C1	1.75317	0.363218275
C2	1.7880	0.358679898
C3	1.21799	0.450858253

Proses selanjutnya adalah seleksi dengan menggunakan metode *elitism selection* yaitu seluruh individu dilakukan *sorting* atau pengurutan berdasarkan nilai *fitness*-nya. Karena semakin besar *fitness* maka semakin baik tingkat akurasi, maka seluruh individu diurutkan dari yang terbesar hingga terkecil. Kemudian diambil sebanyak 30% dari ukuran populasi untuk kemudian diproses kembali pada generasi selanjutnya hingga mendapatkan hasil yang paling optimal.

Tabel 4.29 Hasil seleksi

Individu	Fitness
P4	0.619717504
P3	0.456011738
P2	0.45164842
C3	0.450858253

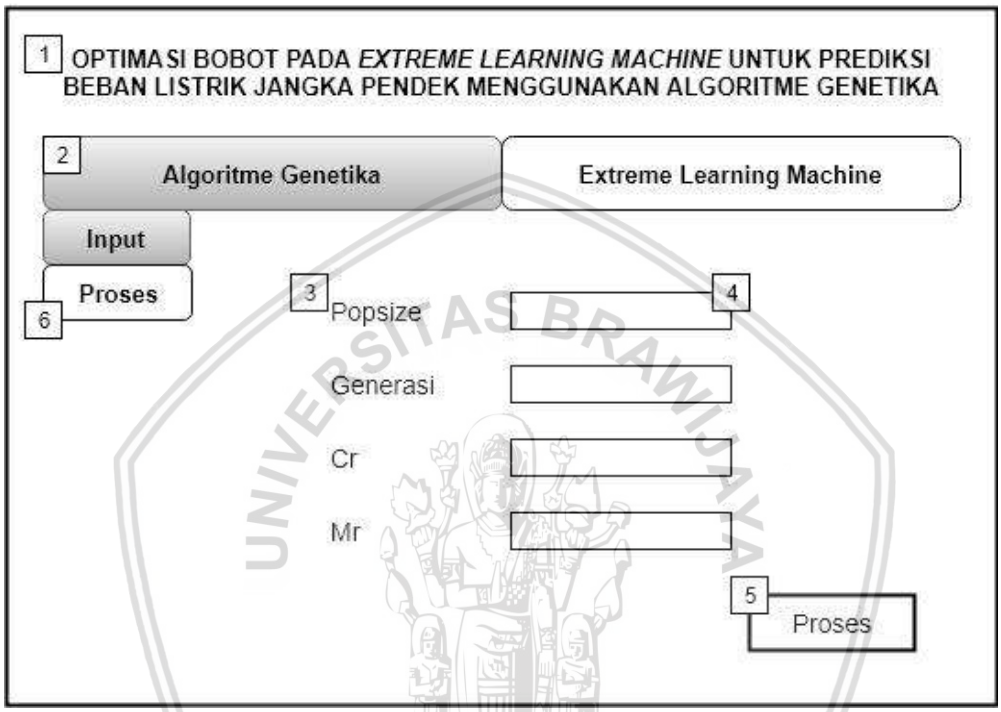
Individu dengan nilai *fitness* tebesar akan menjadi individu terbaik atau yang paling optimal. Dalam hal ini individu P4 merupakan individu terbaik yang artinya menjadi bobot awal terbaik dapat dilihat pada Tabel 4.29.

4.6 Perancangan Antarmuka

Perancangan ini bertujuan untuk memberikan gambaran pada sistem yang akan diimplementasikan pada penelitian ini. Terdiri dari 2 antarmuka utama yaitu halaman Algoritme Genetika dan *Extreme Learning Machine*. Pada masing – masing halaman tersebut memiliki beberapa menu diantaranya 2 menu pada Algoritme Genetika yaitu halaman *input* dan hasil, sementara pada ELM terdiri dari 3 menu yaitu halaman normalisasi data, bobot, bias, serta halaman evaluasi dan hasil.

4.6.1 Halaman Antarmuka Antarmuka Halaman Awal

Pada halaman ini akan menampilkan sebuah sarana untuk melakukan *input* parameter – parameter yang akan digunakan pada proses Algoritme Genetika untuk mencari nilai bobot yang selanjutnya akan digunakan pada metode ELM. Pada Gambar 4.18 berikut adalah rancangan antarmuka dari halaman *input* Algoritme Genetika.



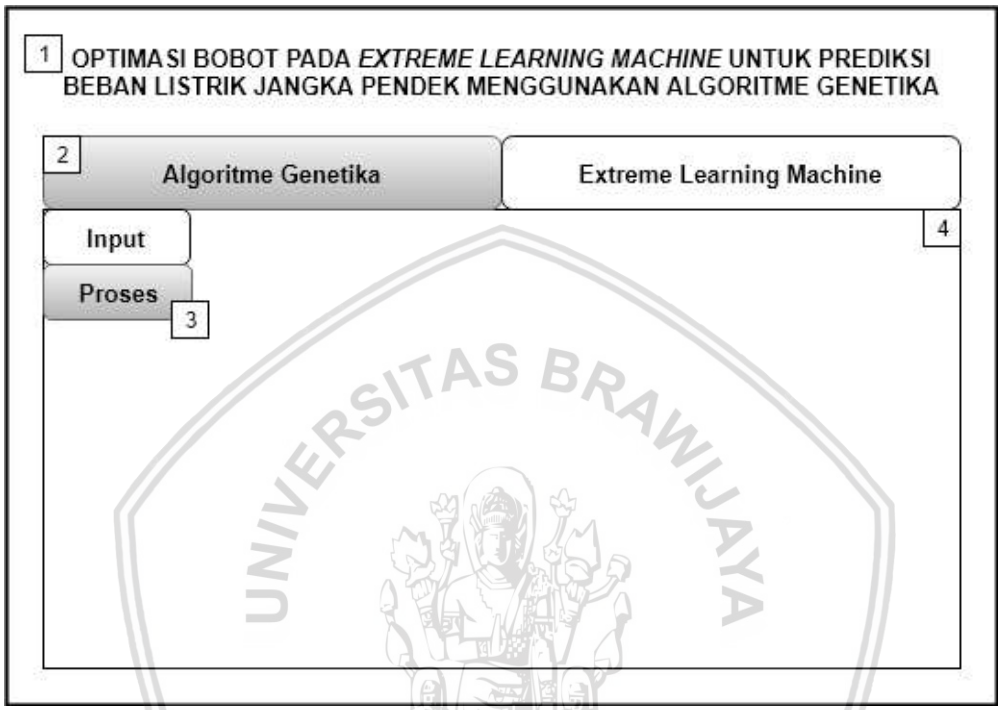
Gambar 4.18 Rancangan antarmuka *Input* Algoritme Genetika

Berikut penjelasan rancangan antarmuka *input* Algoritme Genetika pada Gambar 4.18:

1. *Header* program yang menampilkan judul sistem
2. *Tab* menu untuk menampilkan proses masing – masing metode, menu yang sedang aktif ditunjukkan dengan warna abu – abu
3. *Label* untuk menunjukkan nama parameter Algoritme Genetika yang di *input*-kan
4. *Textbox* untuk memasukkan nilai masing – masing parameter Algoritme Genetika
5. *Button* untuk melakukan proses penghitungan Algoritme Genetika
6. *Tab* menu untuk menampilkan pilihan menu pada menu Algoritme Genetika, menu yang sedang aktif ditunjukkan dengan warna abu – abu

4.6.2 Halaman Antarmuka Hasil Algortima Genetika

Pada halaman ini akan menampilkan hasil perhitungan dari Algoritme Genetika berdasarkan parameter yang sudah di *input*-kan sebelumnya. Hasil yang ditampilkan sesuai dengan langkah – langkah pada metode Algoritme Genetika. Pada Gambar 4.19 berikut adalah rancangan antarmuka dari halaman hasil Algoritme Genetika.



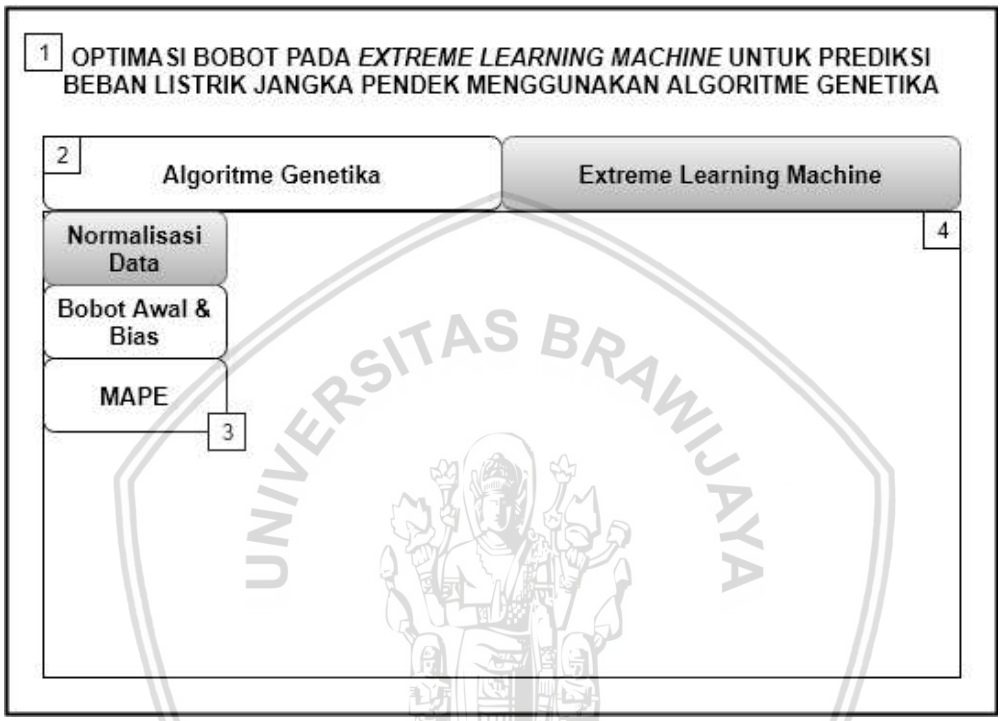
Gambar 4.19 Rancangan antarmuka hasil Algoritme Genetika

Berikut penjelasan rancangan antarmuka hasil Algoritme Genetika pada Gambar 4.19:

1. *Header* program yang menampilkan judul sistem
2. *Tab* menu untuk menampilkan proses masing – masing metode, menu yang sedang aktif ditunjukkan dengan warna abu – abu
3. *Tab* menu untuk menampilkan pilihan menu pada menu Algoritme Genetika, menu yang sedang aktif ditunjukkan dengan warna abu – abu
4. *Textbox* untuk menampilkan hasil perhitungan metode Algoritme Genetika

4.6.3 Halaman Antarmuka Normalisasi Data

Pada halaman ini akan menampilkan hasil normalisasi data sebelum dilakukan proses penghitungan dengan metode ELM. Pada Gambar 4.20 berikut adalah rancangan antarmuka dari halaman normalisasi data.



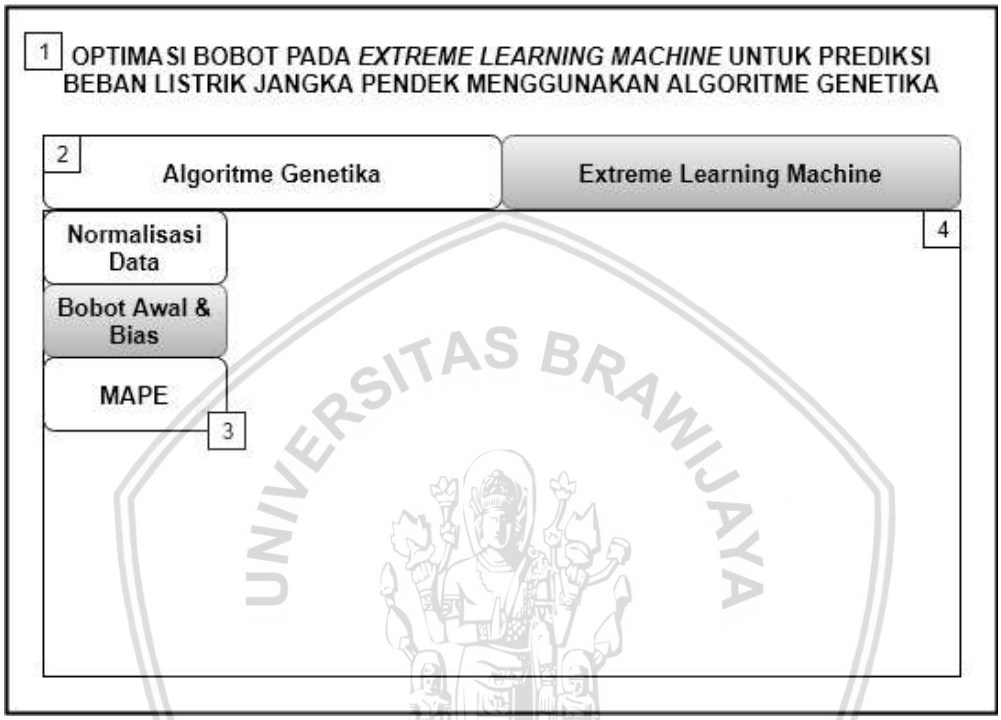
Gambar 4.20 Rancangan antarmuka normalisasi data

Berikut penjelasan rancangan antarmuka normalisasi data pada Gambar 4.20:

1. *Header* program yang menampilkan judul sistem
2. *Tab* menu untuk menampilkan proses masing – masing metode, menu yang sedang aktif ditunjukkan dengan warna abu – abu
3. *Tab* menu untuk menampilkan proses yang ada pada menu ELM, menu yang sedang aktif ditunjukkan dengan warna abu – abu
4. *Textbox* untuk menampilkan hasil normalisasi data

4.6.4 Halaman Antarmuka Bobot Awal & Bias

Pada halaman ini akan menampilkan nilai bobot awal yang sudah dioptimasi dengan menggunakan Algoritme Genetika dan bias yang digunakan untuk proses penghitungan dengan metode ELM. Pada Gambar 4.21 berikut adalah rancangan antarmuka dari halaman bobot.



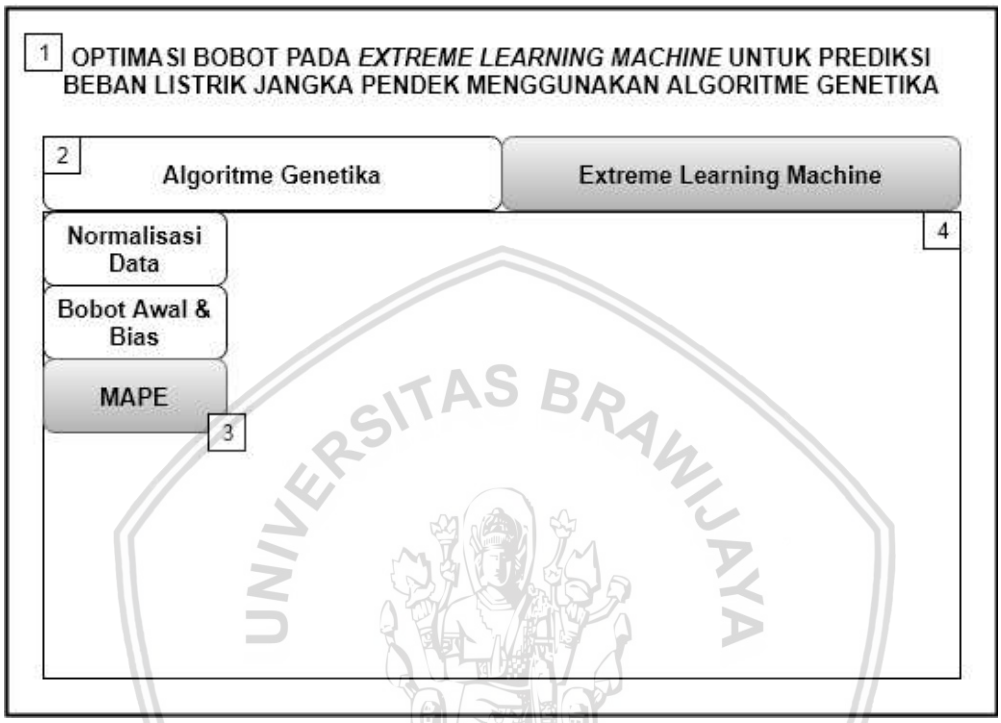
Gambar 4.21 Rancangan antarmuka bobot awal & bias

Berikut penjelasan rancangan antarmuka bobot pada Gambar 4.21:

1. *Header* program yang menampilkan judul sistem
2. *Tab* menu untuk menampilkan proses masing – masing metode, menu heuuyang sedang aktif ditunjukkan dengan warna abu – abu
3. *Tab* menu untuk menampilkan proses yang ada pada metode ELM, menu yang sedang aktif ditunjukkan dengan warna abu – abu
4. *Textbox* untuk menampilkan bobot awal dan bias yang digunakan

4.6.5 Halaman Antarmuka MAPE

Pada halaman ini akan menampilkan hasil prediksi dan hasil perhitungan akurasi dengan menggunakan metode MAPE. Pada Gambar 4.22 berikut adalah rancangan antarmuka dari halaman evaluasi dan hasil.



Gambar 4.22 Rancangan antarmuka MAPE

Berikut penjelasan rancangan antarmuka normalisasi data pada Gambar 4.22:

1. *Header* program yang menampilkan judul sistem
2. *Tab* menu untuk menampilkan proses masing – masing metode, menu yang sedang aktif ditunjukkan dengan warna abu – abu
3. *Tab* menu untuk menampilkan proses yang ada pada metode ELM, menu yang sedang aktif ditunjukkan dengan warna abu – abu
4. *Textbox* untuk menampilkan evaluasi dan hasil prediksi

4.7 Skenario Pengujian

Pada tahapan ini bertujuan untuk mengetahui hasil dari penelitian yang dilakukan. Karena tidak adanya sebuah pendekatan yang pasti untuk memperoleh parameter yang optimal pada algoritme genetika, maka dilakukan sebuah pengujian untuk mengetahui berapa parameter algoritme genetika yang memperleh hasil paling optimal. Pengujian ini terdiri dari pengujian populasi, kombinasi cr dan mr, dan jumlah generasi. Sedangkan untuk penghitungan tingkat *error* menggunakan metode *Mean Absolute Precentage Error* (MAPE).

4.7.1 Pengujian Kombinasi *Crossover Rate* (Cr) & *Mutation Rate* (Mr)

Pengujian ini dilakukan untuk mengetahui kombinasi nilai Cr dan Mr yang menghasilkan nilai fitness terbaik. Dengan melakukan pengubahan pada nilai Cr dan Mr pada setiap percobaan. Pada Tabel 4.28 berikut adalah contoh dari skenario pengujian kombinasi Cr dan Mr.

Tabel 4.30 Skenario Pengujian Kombinasi Cr dan Mr

Kombinasi		Nilai MAPE Percobaan Ke-										Rata - Rata
Cr	Mr	1		2		3		4		5		
		Fitness	Gen	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	

4.7.2 Pengujian Ukuran Populasi

Pengujian ini dilakukan untuk mengukur berapakah ukuran populasi yang memiliki hasil paling optimal dan menghasilkan individu paling baik. Dengan melakukan pengubahan ukuran populasi pada setiap percobaan. Pada Tabel 4.27 berikut adalah contoh dari skenario pengujian populasi.

Tabel 4.31 Skenario pengujian populasi

Banyak Populasi	Nilai MAPE Percobaan Ke-										Rata - Rata
	1		2		3		4		5		
	Fitness	Gen	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	

4.7.3 Pengujian Data Beban Listrik

Pengujian ini dilakukan untuk menguji akurasi hasil prediksi dan membandingkan dengan menggunakan input data berdasarkan jam, hari dan minggu. Data yang digunakan masing – masing dengan pola input yang berbeda berdasarkan jam, hari, dan minggu akan dibandingkan mana yang memberikan hasil paling mendekati nilai sebenarnya.

BAB 5 IMPLEMENTASI

Bab implementasi ini menjabarkan terkait implementasi sistem dan antarmuka sistem berdasarkan perncangan pada bab sebelumnya. Pada bab ini dibahas mengenai perangkat yang digunakan untuk melakukan proses implementasi serta penjelasan terkait implementasi yang dibangun.

5.1 Perangkat Implementasi

Dalam sub-bab ini akan dijelaskan perangkat apa saja yang digunakan dalam membantu proses implementasi sistem. Berikut perangkat yang digunakan dalam proses implementasi antara lain:

1. Perangkat Keras
   a. Laptop
      - Intel(R) Celeron(R) CPU B830 @ 1.80GHz
      - RAM 4 GB
      - Monitor 14 inch
   b. Printer
2. Perangkat Lunak
   a. Netbeans IDE 8.0.1
   b. Microsoft Excel 2013

5.2 Implementasi Algoritme Genetika

Pada implementasi algoritme genetika akan dijelaskan mengenai implementasi sistem prediksi beban listrik yang mengacu pada perancangan yang telah dibuat sebelumnya. Bahasa yang digunakan dalam implementasi adalah bahasa pemrograman JAVA serta sebuah library yaitu JXL yang digunakan untuk mengakses *file excel*. Pada bab ini akan dijelaskan implementasi dari proses – proses yang ada pada algoritme genetika mulai dari inisialisasi kromosom awal, proses *crossover,* proses mutasi, proses evaluasi, serta proses seleksi.

5.2.1 Implementasi Inisialisasi Kromosom Awal

Tabel 5.1 Kode Program Inisialisasi Kromosom Awal

No	Kode Program
1	`public void populasiAwal() {`
2	`   individu = new double[popsize][panjangIndv];`
3	`   double max = 1, min = -1;`
4	`   for (int i = 0; i < individu.length; i++) {`
5	`      for (int j = 0; j < panjangIndv; j++) {`
6	`      scaled = random.nextDouble() * (max - min);`
7	`      shifted = scaled + min;`
8	`      temp = Math.pow(10, angka);`
9	`      rand = (double) Math.round(shifted * temp) / temp;`
10	`      individu[i][j] = rand;`
11	`      }`
12	`   }`
13	`}`

Pada Tabel 5.1 menunjukkan *source code* dari proses inisialisasi populasi awal. Proses tersebut dilakukan dengan cara pembangkitan individu secara *random*. Individu yang dibangkitkan merupakan representasi dari bobot awal yang akan dioptimasi dan digunakan pada proses ELM. Bobot awal tersebut memiliki *range* sebesar -1 hingga 1.

5.2.2 Implementasi Proses *Crossover*

Tabel 5.2 Kode Program *Extended Intermediate Crossover*

No	Kode Program
1	`public double[][] Crossover(double[][] individu) {`
2	
3	`double maxAlpha = 1.25, minAlpha = -0.25;`
4	`double range = maxAlpha - minAlpha;`
5	`double[][] p1 = new double[1][panjangIndv];`
6	`double[][] p2 = new double[1][panjangIndv];`
7	`double[] newAlpha = new double[panjangIndv];`
8	`int parent1 = 0, parent2 = 0;`
9	
10	`    //mengambil 2 parent scr acak`
11	`    do {`
12	`        parent1 = random.nextInt(popsize);`
13	`        parent2 = random.nextInt(popsize);`
14	`    } while ((parent1 == parent2));`
15	
16	`  //membangkitkan nilai alpha`
17	`  for (int j = 0; j < panjangIndv; j++) {`
18	`     scaled = random.nextDouble() * range;`
19	`     shifted = scaled + minAlpha;`
20	`     temp = Math.pow(10, angka);`
21	`     rand = (double) Math.round(shifted * temp) / temp;`
22	`     newAlpha[j] = rand;`
23	`  }`
24	
25	`   //membangkitkan parent1 & parent2`
26	`   for (int j = 0; j < panjangIndv; j++) {`
27	`      p1[0][j] = individu[parent1][j];`
28	`   for (int j = 0; j < panjangIndv; j++) {`
29	`      p2[0][j] = individu[parent2][j];`
30	
31	`   //menghasilkan crossover`
32	`   for (int j = 0; j < panjangIndv; j++) {`
33	`     child1[0][j] = p1[0][j] + (newAlpha[j] * (p2[0][j]`
34	`- p1[0][j]));`
35	`     child1[1][j] = p2[0][j] + (newAlpha[j] * (p1[0][j]`
36	`- p2[0][j]));`
37	`   }`
38	
39	`  return child1;`
40	`}`

Pada Tabel 5.2 diatas merupakan *source code* dari *method Crossover*. Fungsinya adalah untuk menghasilkan *offspring* dengan menyilangkan 2 buah *parent*. Berikut adalah penjelasan *code* dari Tabel 5.2:

Baris 3 – 8 : Inisialisasi variabel yang digunakan pada perhitungan proses *Extended Intermediate Crossover.*

Baris 11 – 14 : Proses pemilihan *parent 1* dan *parent 2* dari populasi yang dibangkitkan secara *random* untuk disilangkan.

Baris 17 – 23 : Proses pembangkitan nilai *alpha* sebagai variabel tambahan pada metode *Extended Intermediate Crossover* secara *random* dengan *range* -0.25 hingga 1.25.

Baris 32 – 40 : Proses menghasilkan *offspring* dengan cara menyilangkan *parent* yang sudah didapatkan sebelumnya dan nilai *alpha* menggunakan Persamaan 2.7.

5.2.3 Implementasi Proses Mutasi

Tabel 5.3 Kode Program *Random Mutation*

No	Kode Program
1	`public double[][] Mutasi(double[][] individu) {`
2	
3	`double r;`
4	`double maxR = 0.1, minR = -0.1;`
5	`double range = maxR - minR;`
6	
7	`for (int i = 0; i < mRate; i++) {`
8	
9	`//random nilai r`
10	`scaled = random.nextDouble() * range;`
11	`shifted = scaled + minR;`
12	`temp = Math.pow(10, angka);`
13	`rand = (double) Math.round(shifted * temp) / temp;`
14	`r = rand;`
15	
16	`// mengambil 1 parent scr acak`
17	`int parents = random.nextInt(popsize);`
18	`for (int j = 0; j < panjangIndv; j++) {`
19	`child2[i][j] = individu[parents][j];`
20	`}`
21	
22	`//memilih index yang akan dimutasi`
23	`int index = random.nextInt(panjangIndv);`
24	
25	`//mencari nilai min max`
26	`double max = individu[0][0], min = individu[0][0];`
27	`for (int j = 0; j < individu.length; j++) {`
28	`if (individu[j][index] > max) {`
29	`max = individu[j][index];`

```
30          }
31          if (individu[j][index] < min) {
32             min = individu[i][index];
33          }
34       }
35
36      //menghasilkan child dr mutasi
37      double mResult = individu[parents][index] + (r *
38 (max - min));
39      child2[i][index] = mResult;
40      }
41    return child2;
42 }
```

Pada Tabel 5.3 diatas merupakan *source code* dari *method* Mutasi. Fungsinya adalah untuk menghasilkan *offspring* dengan melakukan mutasi pada 1 gen dari 1 *parent* terpilih. Berikut adalah penjelasan *code* dari Tabel 5.3:

Baris 3 – 5 : Inisialisasi variabel yang digunakan pada perhitungan proses *Random Mutation.*

Baris 10 - 14 : Proses pembangkitan nilai r sebagai variabel tambahan pada metode *Random Mutation* secara *random* dengan *range* -0.1 hingga 0.1.

Baris 17 – 20 : Proses pemilihan 1 *parent* dari populasi

Baris 23 : Proses memilih 1 indeks dari *parent* terpilih yang dibangkitkan secara *random.*

Baris 26 – 34 : Proses penentuan nilai *max* dan *min* untuk variabel tambahan pada metode *Random Mutation*. Nilai tersebut didapatkan dari pencarian nilai maksimal dan minimal diantara gen ke – x sesuai index terpilih pada seluruh populasi.

Baris 37 – 42 : Proses menghasilkan *offspring* dengan metode *Random Mutation* menggunakan Persamaan 2.8.

5.2.4 Implementasi Proses Evaluasi

Tabel 5.4 Kode Program Proses Evaluasi

No	Kode Program
1	`public double[][] calcFitness() throws IOException,`
2	`BiffException {`
3	`   fitness = new double[mixIndv.length];`
4	`   for (int i = 0; i < mixIndv.length; i++) {`
5	`        konversiMatriks(i);`
6	`        proses(inpW);`
7	`        fitness[i] = 1 / (1  + mixMape[0][0]);`
8	`   }return inpW;`
9	`}`

Pada Tabel 5.4 menunjukkan *source code* dari proses evaluasi. Proses tersebut dilakukan dengan melakukan perhitungan *fitness* dari masing – masing individu. Yang mana pada penelitian ini nilai *fitness* tersebut merupakan satu per nilai MAPE dari proses ELM. Pada *source code* tersebut proses perhitungan *fitness* terdapat pada baris 8, yaitu dengan memanggil *method* proses yang berisikan perhitungan MAPE metode ELM. Hasil nilai *fitness* kemudian disimpan dalam array *fitness.*

5.2.5 Implementasi Proses Seleksi

Tabel 5.5 Kode Program Proses Seleksi

No	Kode Program
1	`public double[][] seleksiElitism(double mixFitness[]) {`
2	
3	`  double fIndex[][] = new double[mixFitness.length][2];`
4	`  double hasilSeleksi[][] = new double[cutSel][2];`
5	
6	`  for (int i = 0; i < fIndex.length; i++) {`
7	`  fIndex[i][0] = i;`
8	`  fIndex[i][1] = mixFitness[i];`
9	`  }`
10	
11	`  for (int i = 0; i < fIndex.length; i++) {`
12	`  for (int j = 1; j < fIndex.length - 1; j++) {`
13	`  if (fIndex[j - 1][1] < fIndex[j][1]) {`
14	`  double temp0 = fIndex[j - 1][0];`
15	`  double temp1 = fIndex[j - 1][1];`
16	`  fIndex[j - 1][0] = fIndex[j][0];`
17	`  fIndex[j - 1][1] = fIndex[j][1];`
18	`  fIndex[j][0] = temp0;`
19	`  fIndex[j][1] = temp1;`
20	`  }`
21	`  }`
22	`  }`
23	
24	`  for (int i = 0; i < hasilSeleksi.length; i++) {`
25	`  hasilSeleksi[i][0] = fIndex[i][0];`
26	`  hasilSeleksi[i][1] = fIndex[i][1];`
27	`  }`
28	`  return hasilSeleksi;`
29	`}`

Pada Tabel 5.5 diatas merupakan *source code* dari proses seleksi. Metode seleksinya adalah *elitism selection* yaitu hasil nilai *fitness* yang diperoleh dari proses evaluasi diurutkan secara *descending* dari yang paling terbesar hingga yang paling terkecil. Pada *source code* tersebut proses pengurutan indeks nilai *fitness* terdapat pada baris 11 – 22. Kemudian indeks tersebut digabungkan dengan nilai *fitness*-nya dalam satu *array*.

5.3 Implementasi *Extreme Learning Machine*

Pada implementasi ELM akan dijelaskan mengenai implementasi sistem prediksi beban listrik yang mengacu pada perancangan yang telah dibuat sebelumnya. Bahasa yang digunakan dalam implementasi adalah bahasa pemrograman JAVA serta sebuah library yaitu JAMA yang digunakan untuk melakukan perhitungan matriks. Pada bab ini akan dijelaskan implementasi dari proses – proses yang ada pada metode ELM mulai dari normalisasi data, pembangkitan bobot awal dan bias, perhitungan *output hidden layer*, aktivasi sigmoid, perhitungan *output weight*, perhitungan hasil prediksi, perhitungan akurasi menggunakan metode MAPE.

5.3.1 Implementasi Normalisasi Data

Tabel 5.6 Kode Program Normalisasi Data

No	Kode Program
1	`public void dataNormTr(int x) { //menormalisasi data`
2	
3	`max = 0;`
4	`min = 1000;`
5	
6	`for (int i = 0; i < dataTr.length; i++) {`
7	`if (dataTr[i][x] > max) {`
8	`max = dataTr[i][x];`
9	`}`
10	`if (dataTr[i][x] < min) {`
11	`min = dataTr[i][x];`
12	`}`
13	`}`
14	
15	`for (int i = 0; i < dataTr.length; i++) {`
16	`dataTr[i][x] = (dataTr[i][x] - min) / (max - min);`
17	`}`
18	`}`

Pada Tabel 5.6 diatas merupakan proses awal yang dilakukan pada metode ELM yaitu normalisasi data. Data yang dinormalisasi merupakan data yang digunakan pada proses *training* dan *testing.* Data yang tersimpan dalam variable dataTr dinormalisasi dengan menggunakan Persamaan $Data_{normi,j} = (data_{i,j} – min_i) / max_i - min_i$. Pada baris ke 6 – 13 merupakan proses pencarian nilai *min* dan *max* variabel x mewakili masing – masing kolom yang dinormalisasi. Kemudian pada baris 15 – 17 merupakan proses perhitungan normalisasi data.

5.3.2 Implementasi Pembangkitan Bias

Tabel 5.7 Kode Program Pembangkitan Bias

No	Kode Program
1	`public void pembangkitanBias() throws IOException {`
2	
3	`File inputWb = new File("C:\\Users\\Vina`
4	`Myla\\Documents\\Exercise\\SkripsiVM\\bias.xls");`
5	`Workbook wb;`
6	
7	`try {`
8	`wb = Workbook.getWorkbook(inputWb);`
9	`Sheet sheet = wb.getSheet(0);`
10	`populasi = sheet.getRows();`
11	`bias = new double[populasi];`
12	
13	`for (int i = 0; i < populasi; i++) {`
14	`Cell cell = sheet.getCell(0, i);`
15	`bias[i] = Double.valueOf(cell.getContents());`
16	`}`
17	`} catch (BiffException e) {`
18	`e.printStackTrace();`
19	`}`
20	`}`

Pada Tabel 5.7 menunjukkan *source code* dari proses pembangkitan bias. Nilai bias dibangkitkan secara *random* dengan *range* antara 0 hingga 1. Akan tetapi proses pembangkitan bias pada *source code* tersebut dengan cara mengambil nilai bias yang sudah dibangkitkan dan disimpan didalam *file excel*. Pengaksesan *file excel* tersebut dengan menggunkan *library* JXL. Bias yang didapatkan dari *file excel* disimpan dalam satu array yang bernama bias.

5.3.3 Implementasi Proses Perhitungan *Output Hidden Layer*

Tabel 5.8 Kode Program *Output Hidden Layer*

No	Kode Program
1	`public void outputHiddenL(double[][] data, double[][]`
2	`inputWeight) {`
3	
4	`double[][] wTransp = transposeMx(inputWeight);`
5	`hInit = crossMx(data, wTransp);`
6	
7	`for (int i = 0; i < hInit.length; i++) {`
8	`for (int j = 0; j < hInit[0].length; j++) {`
9	`hInit[i][j] += bias[j];`
10	`}`
11	`}`
12	`}`

Pada Tabel 5.8 merupakan *source code* dari proses perhitungan *output hidden layer,* proses ini dilakukan pada proses *training* dan *testing* metode ELM. Didalam proses ini terdapat perkalian yang berbentuk matriks. Pada baris 4 merupakan proses pemanggilan method untuk melakukan proses transpose matriks. Sedangkan pada baris 5 merupakan proses pemanggilan method untuk melakukan perkalian matriks antara data dan bobot awal. Proses perhitungan *output hidden layer* yang ditambahkan dengan nilai bias yang telah dibangkitkan terdapat pada baris ke 7 – 11 dengan menggunakan Persamaan 2.1.

5.3.4 Implementasi Proses Aktivasi Sigmoid

Tabel 5.9 Kode Program Aktivasi Sigmoid

No	Kode Program
1	`public void aktivasiSigmoid() {`
2	
3	`  h = new double[hInit.length][hInit[0].length];`
4	
5	`    for (int i = 0; i < hInit.length; i++) {`
6	`      for (int j = 0; j < hInit[0].length; j++) {`
7	`        h[i][j] = 1 / (1 + Math.exp(-hInit[i][j]));`
8	`      }`
9	`    }`
10	`}`

Pada Tabel 5.9 menunjukkan *source code* untuk proses aktivasi sigmoid. Proses ini juga dilakukan pada proses *training* dan *testing* metode ELM. Pada baris 3 merupakan inisialisasi variabel untuk menyimpan hasil aktivasi sigmoid. Selanjutnya pada baris 5 – 9 merupakan proses perhitungan aktivasi sigmoid dengan menggunakan Persamaan 2.2.

5.3.5 Implementasi Proses Perhitungan *Output Weight*

Tabel 5.10 Kode Program Pehitungan *Output Weight*

No	Kode Program
1	`public void outputWeight() throws IOException {`
2	
3	`  double[][] hTrans  = transposeMx(h);`
4	`  double[][] hTh     = crossMx(hTrans, h);`
5	`  double[][] hInvers = inversMx(hTh);`
6	`  double[][] hPlus   = crossMx(hInvers, hTrans);`
7	`            beta     = crossMx(hPlus, targetTr);`
8	
9	`}`

Pada Tabel 5.10 merupakan *source code* dari proses perhitungan *output weight*. Proses ini hanya ada dalam proses *training* metode ELM. Dalam implementasi proses

tersebut terdapat proses perhitungan *Moore-Penrose Pseudo Inverse.* Operasi perhitungan dalam proses *moore-penrose* meliputi *transpose, inverse,* dan perkalian. Proses perhitungan matriks *moore-penrose pseudo inverse* terdapat pada baris 3 – 6 menggunakan Persamaan 2.3. Baris 7 merupakan proses pemanggilan method perkalian matriks antara data target *training* yang sudah dinormalisasi dengan hasil perhitungan *moore-penrose* dengan menggunakan Persamaan 2.4.

5.3.6 Implementasi Hasil Prediksi

Tabel 5.11 Kode Program Hasil Prediksi

No	Kode Program
1	`public void hasilPrediksi() {`
2	
3	`  yPrediksi = crossMx(h, beta);`
4	
5	`}`

Pada Tabel 5.11 menunjukkan *source code* proses perhitungan hasil prediksi. Proses ini hanya terdapat pada proses *testing* metode ELM. Seperti pada baris 3 merupakan proses pemanggilan method perkalian matriks hasil bobot *output* yang didapatkan dari proses *training* dengan hasil *output hidden layer* yang telah diaktivasi menggunakan Persamaan 2.5.

5.3.7 Implementasi Proses MAPE

Tabel 5.12 Kode Program Perhitungan Akurasi

No	Kode Program
1	`public void hitungAkurasi() throws IOException {`
2	
3	`   targetTest();`
4	`   denorm = new double[targetTs.length][1];`
5	`   mape = new double[targetTs.length][1];`
6	`   mixMape = new double[1][1];`
7	`   max = 0;`
8	`   min = 1000;`
9	
10	`     for (int i = 0; i < targetTs.length; i++) {`
11	`         if (targetTs[i][0] > max) {`
12	`             max = targetTs[i][0];`
13	`         }`
14	`         if (targetTs[i][0] < min) {`
15	`             min = targetTs[i][0];`
16	`         }`
17	`     }`
18	
19	`  for (int i = 0; i < targetTs.length; i++) {`
20	`     denorm[i][0] = yPrediksi[i][0] * (max - min) + min;`

```
21    }
22
23    for (int i = 0; i < targetTs.length; i++) {
24       mape[i][0] = (((targetTs[i][0] - denorm[i][0]) /
25 targetTs[i][0])*100);
26       mixMape[0][0] += mape[i][0];
27    }
28
29       mixMape[0][0]    =    Math.abs    (mixMape[0][0]    /
30 targetTs.length);
31 }
```

Pada Tabel 5.12 menunjukkan *source code* dari proses perhitungan akurasi metode ELM untuk mengetahui seberapa besar tingkat error dari prediksi yang dihasilkan. Pada penelitian ini proses perhitungan akurasinya menggunakan metode MAPE. Berikut adalah penjelasan *code* dari Tabel 5.12:

Baris 3 : Proses memanggil *method* targetTest() untuk mendapatkan data target *testing* yang telah dinormalisasi.

Baris 4 – 8 : Inisialisasi variabel yang digunakan pada proses perhitungan akurasi.

Baris 10 – 17 : Proses mencari nilai *max* dan *min* pada hasil prediksi untuk proses denormalisasi data.

Baris 19 – 21 : Proses denormalisasi data dari data hasil prediksi dengan menggunakan Persamaan $Data_{denormi} = Data_i * (max_i - min_i) + min_i$.

Baris 23 – 30 : Proses perhitungan MAPE dengan menggunakan Persamaan 2.9.

5.4 Implementasi Antarmuka

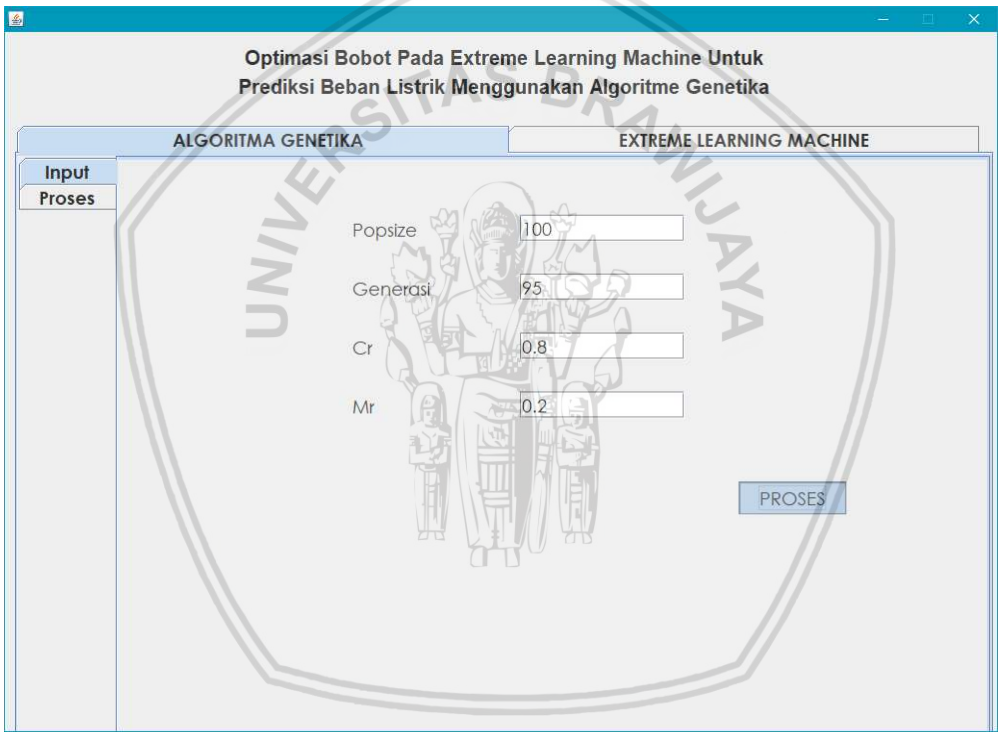
Pada sub-bab ini menampilkan terkait antramuka sistem optimasi bobot pada *extreme learning machine* untuk prediksi beban listrik menggunakan algoritme genetika yang sudah diimplementasikan kedalam sebuah program mengacu pada rancangan yang telah dibuat pada bab sebelumnya. Antarmuka terdiri dari 2 menu utama yaitu algortime genetika dan *extreme learning machine.*

5.4.1 Implementasi Antarmuka Algoritme Genetika

Pada menu antarmuka yang pertama yaitu algoritme genetika merupakan antarmuka yang berfungsi untuk melakukan *input* parameter dan menampilkan beberapa proses yang terdapat pada algoritme genetika seperti inisialisasi populasi awal, *crossover*, mutasi, seleksi, dan evaluasi.

5.4.1.1 Implementasi Antarmuka Halaman Awal

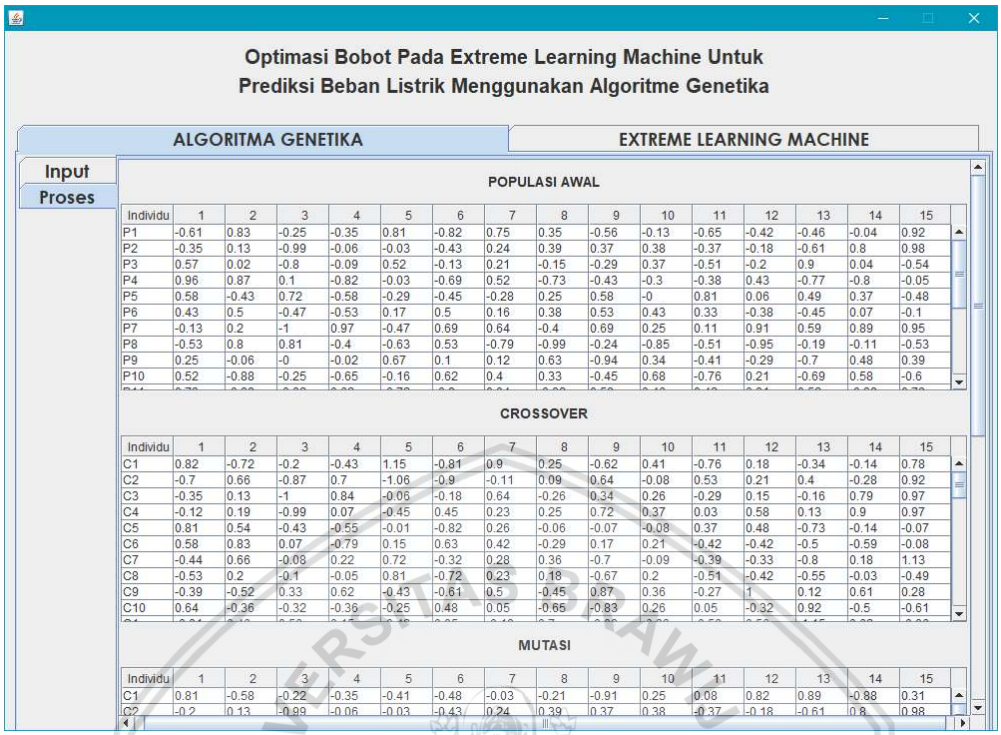
Ketika pertama kali program dijalankan maka akan ditampilkan sebuah antarmuka untuk *user* melakukan *input* parameter yang dibutuhkan pada proses algoritme genetika yang terdiri dari 4 *textbox* dengan masing-masing label yaitu *popsize* (ukuran populasi), generasi, *crossover rate,* dan *mutation rate.* Dan terdapat satu buah *button* untuk memulai proses. Berikut implementasi antarmuka halaman input yang terdapat pada Gambar 5.1.



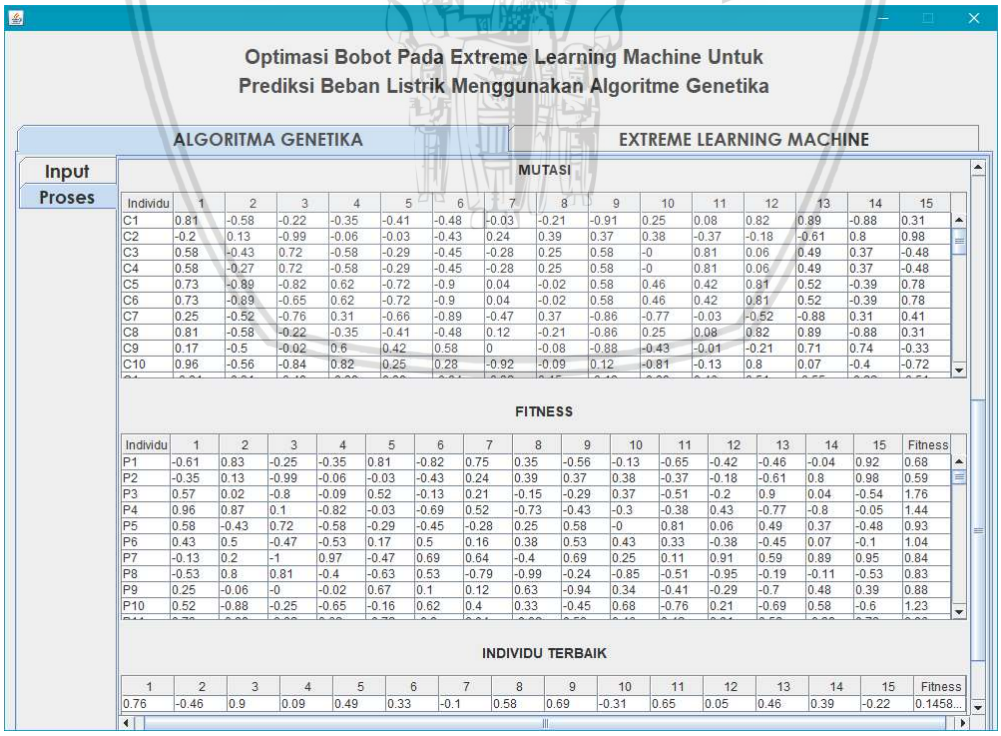
Gambar 5.1 Implementasi Antarmuka Halaman *Input*

5.4.1.2 Implementasi Antarmuka Halaman Proses

Pada halaman proses akan ditampilkan antarmuka yang berisikan tabel – tabel hasil dari proses perhitungan algoritme genetika. Proses yang ditampilkan terdiri dari inisialisasi populasi awal, *crossover*, mutasi, evaluasi, dan seleksi yang menampilkan individu terbaik. Berikut implementasi antaramuka halaman proses yang terdapat pada Gambar 5.2 dan Gambar 5.3.



Gambar 5.2 Implementasi Antarmuka Halaman Proses



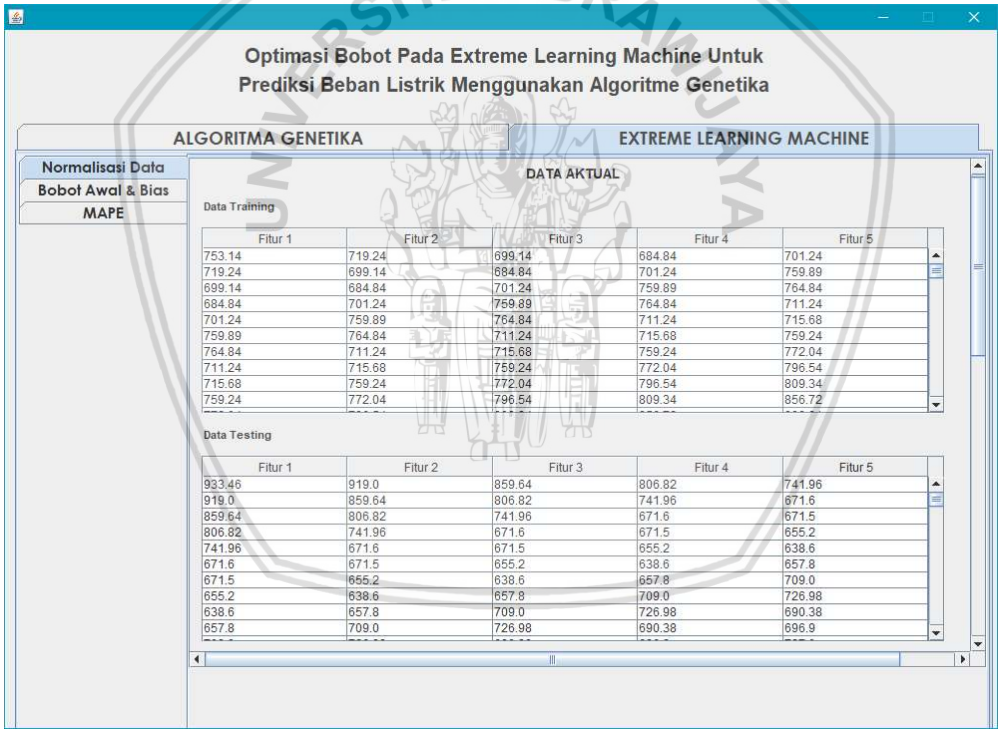
Gambar 5.3 Implementasi Antarmuka Halaman Proses (lanjutan)

5.4.2 Implementasi Antarmuka *Extreme Learning Machine*

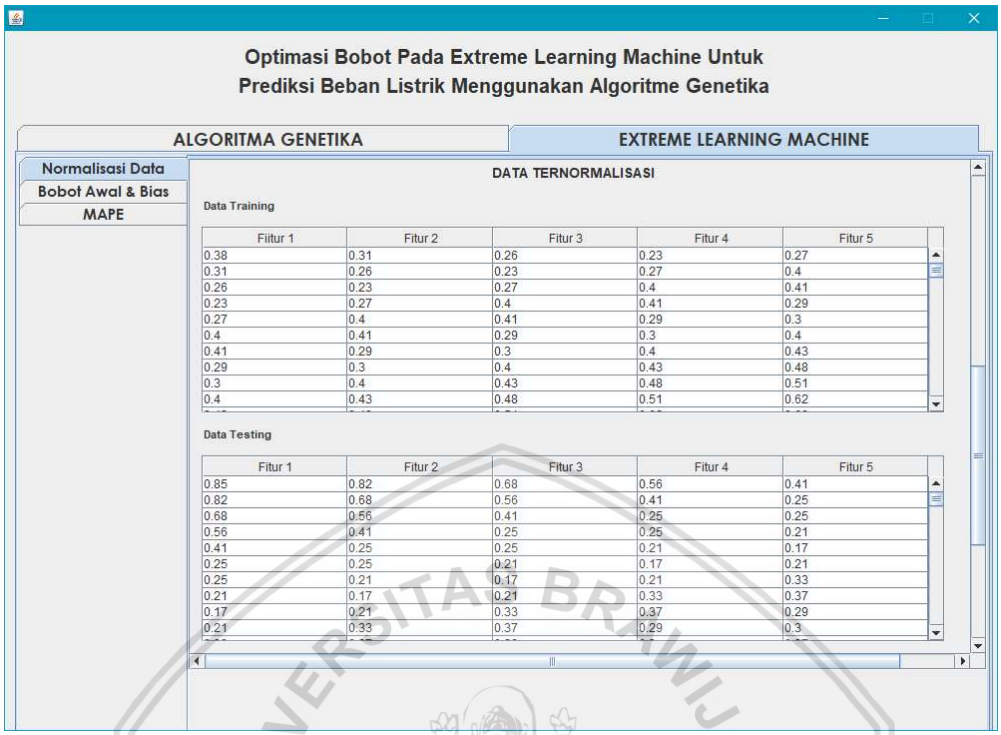
Pada menu antarmuka yang kedua yaitu *extreme learning machine* merupakan antarmuka yang berfungsi untuk menampilkan beberapa proses yang terdapat pada metode ELM. Didalamnya terdapat 3 sub-menu yang memiliki fungsinya masing-masing terdiri dari normalisasi data, bobot awal & bias, dan MAPE.

5.4.2.1 Implementasi Antarmuka Normalisasi Data

Sub-menu yang pertama adalah normalisasi data. Pada halaman antarmuka ini akan menampilkan tabel – tabel yang berisikan data *training* dan data *testing* aktual serta data *training* dan data *testing* ternormalisasi. Data diambil dari sebuah *file excel* yang kemudian dibaca oleh program, dinormalisasi dan ditampilkan pada halaman antarmuka. Berikut implementasi antarmuka normalisasi data yang terdapat pada Gambar 5.4 dan Gambar 5.5.



Gambar 5.4 Implementasi Antarmuka Normalisasi Data



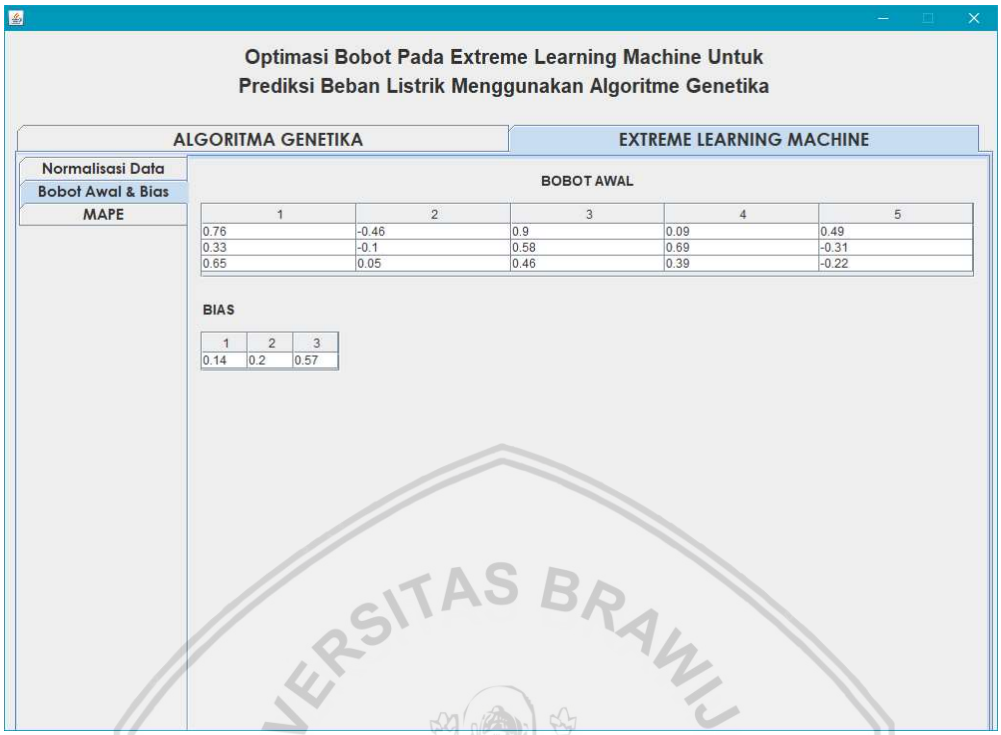
Gambar 5.5 Implementasi Antarmuka Normalisasi Data (lanjutan)

5.4.2.2 Implementasi Antarmuka Bobot Awal & Bias

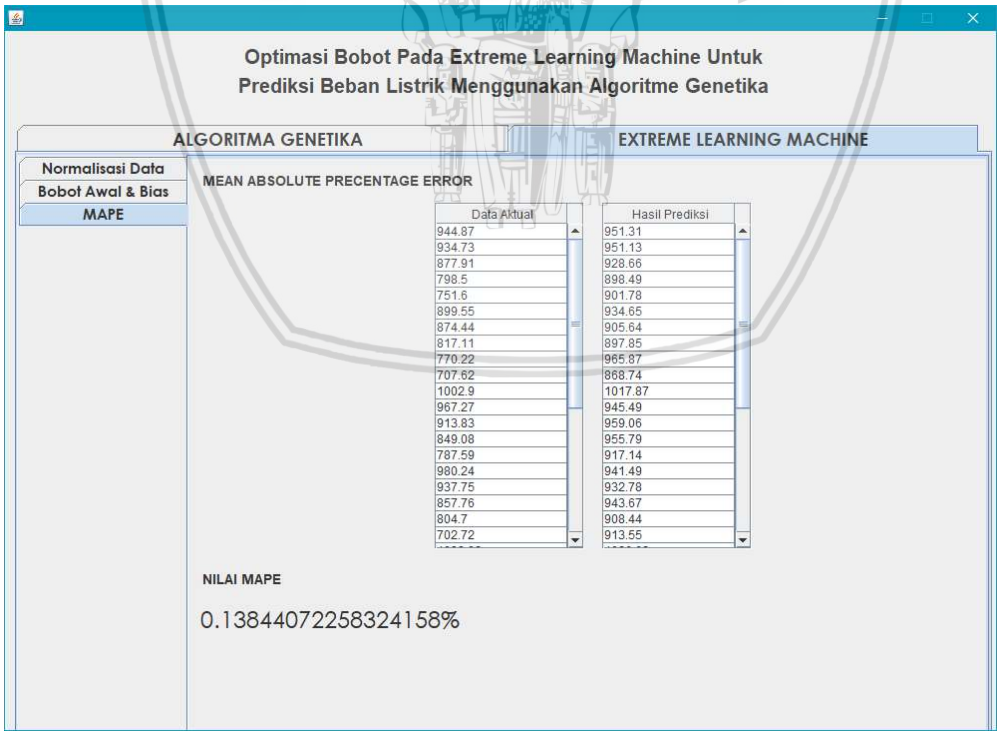
Sub-menu kedua adalah halaman antarmuka bobot awal & bias yang menampilkan tabel – tabel berisikan bobot awal hasil dari individu terbaik algoritme genetika dan bias yang diambil dari sebuah *file excel* yang akan digunakan untuk memprediksi beban listrik. Implementasi antarmuka bobot awal & bias terdapat pada Gambar 5.6.

5.4.2.3 Implementasi Antarmuka Hasil Prediksi

Pada sub-menu selanjutnya adalah antarmuka hasil prediksi yang menampilkan hasil dari prediksi beban menggunakan metode ELM. Dalam antarmuka tersebut akan terdapat 2 tabel, tabel yang pertama merupakan tabel untuk menampilkan data aktual yang diambil dari sebuah *file excel* sedangkan tabel kedua merupakan tabel pembanding untuk menampilkan hasil perhitungan prediksi beban listrik menggunakan metode ELM. Kemudian terdapat *text* yang menampilkan hasil akurasi prediksi menggunakan metode MAPE. Implementasi antarmuka hasil prediksi terdapat pada Gambar 5.7



Gambar 5.6 Implementasi Antarmuka Bobot Awal dan Bias



Gambar 5.7 Implementasi Antarmuka Hasil Prediksi

BAB 6 PENGUJIAN DAN ANALISIS

Bab pengujian ini menjabarkan terkait pengujian dari sistem yang sudah diimplementasikan kedalam kode program. Pengujian dilakukan dengan mengacu pada bab perancangan sebelumnya. Terdapat 4 tahapan dalam melakukan pengujian yaitu pengujian kombinasi cr dan mr, pengujian ukuran populasi, pengujian jumlah generasi, serta pengujian data.

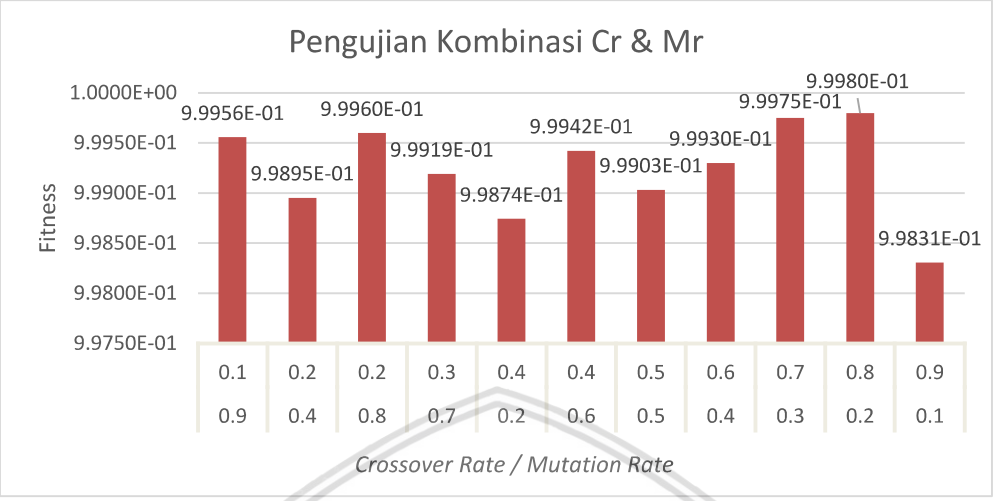
6.1 Pengujian Kombinasi *Crossover Rate* (Cr) & *Mutation Rate* (Mr)

Sesuai pada Tabel 4.27, pengujian kombinasi *crossover rate* dan *mutation rate* ini bertujuan untuk mendapatkan kombinasi nilai cr dan mr yang memberikan hasil paling optimal yaitu memiliki nilai *fitness* terbesar. Terdapat 11 variasi kombinasi yang akan digunakan untuk melakukan pengujian. Masing – masing diuji dengan melakukan sebanyak 5 kali percobaan. Selain kombinasi nilai cr dan mr, terdapat parameter lainnya yang harus diinputkan yaitu ukuran populasi dan jumlah generasi. Ukuran populasi akan di *set* dengan nilai tetap sebanyak 20, sedangkan jumlah generasi juga di *set* sebanyak 100. Nilai kombinasi cr dan mr terbaik akan digunakan untuk skenario pengujian selanjutnya.

Tabel 6.1 Nilai *Fitness* Hasil Pengujian Kombinasi Cr dan Mr

Kombinasi		Nilai *Fitness* Percobaan Ke-										Nilai Rata-Rata *Fitness*
		1		2		3		4		5		
Cr	Mr	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	
0.1	0.9	9.9998E-01	83	9.9940E-01	36	9.9952E-01	35	9.9988E-01	75	9.9900E-01	15	9.9956E-01
0.2	0.4	9.9807E-01	86	9.9934E-01	34	9.9883E-01	42	9.9967E-01	73	9.9886E-01	23	9.9895E-01
0.2	0.8	9.9995E-01	70	9.9996E-01	38	9.9889E-01	72	9.9965E-01	93	9.9953E-01	15	9.9960E-01
0.3	0.7	9.9842E-01	24	9.9949E-01	81	9.9816E-01	50	9.9990E-01	81	9.9998E-01	61	9.9919E-01
0.4	0.2	9.9600E-01	70	9.9866E-01	90	9.9979E-01	21	9.9972E-01	55	9.9954E-01	52	9.9874E-01
0.4	0.6	9.9873E-01	54	9.9966E-01	92	9.9983E-01	27	9.9892E-01	69	9.9998E-01	73	9.9942E-01
0.5	0.5	9.9638E-01	81	9.9983E-01	67	9.9970E-01	49	9.9965E-01	32	9.9960E-01	49	9.9903E-01
0.6	0.4	9.9947E-01	42	9.9942E-01	47	9.9973E-01	20	9.9978E-01	8	9.9810E-01	23	9.9930E-01
0.7	0.3	9.9979E-01	51	9.9972E-01	33	9.9975E-01	59	9.9949E-01	74	1.0000E+00	43	9.9975E-01
0.8	0.2	9.9986E-01	80	9.9985E-01	23	9.9982E-01	31	9.9965E-01	56	9.9982E-01	28	9.9980E-01
0.9	0.1	9.9953E-01	75	9.9906E-01	52	9.9975E-01	95	9.9367E-01	61	9.9952E-01	84	9.9831E-01

Tabel 6.1 menunjukkan hasil pengujian kombinasi nilai *crossover rate* dan *mutation rate.* Berikut grafik hasil pengujian kombinasi nilai *crossover rate* dan *mutation rate* yang ditunjukkan pada Gambar 6.1.



Gambar 6.1 Grafik Nilai *Fitness* Hasil Pengujian Kombinasi Cr & Mr

Dari grafik tersebut dapat dilihat bahwa nilai *fitness* yang dihasilkan mengalami naik turun atau bisa dikatakan tidak stabil, hal tersebut dapat disebabkan oleh pembangkitan nilai yang dilakukan secara *random* setiap kali proses dijalankan. Kombinasi nilai cr dan mr dapat memberikan hasil yang berbeda – beda pada setiap permasalahan. Pada permasalahan ini nilai cr dan mr yang memperoleh hasil paling optimal didapatkan dari kombinasi cr 0.8 dan mr 0.2 yaitu dengan nilai rata – rata *fitness* sebesar $9.998x10^{-1}$. Sedangkan rata - rata nilai *fitness* terkecil didapatkan dari kombinasi cr 0.9 dan mr 0.1 sebesar $9.983x10^{-1}$.

Dapat disimpulkan dari hasil tersebut bahwa *offspring* banyak dihasilkan dari proses *crossover* dibandingkan dengan proses mutasi. Akan tetapi jika nilai antara cr dan mr sama maka hasil yang diberikan adalah kurang optimal. Pada beberapa penelitian banyak yang menggunakan nilai cr yang lebih tinggi dibandingkan nilai mr. Menurut Desiani & Arhami alasannya adalah karena fokus dari proses *crossover* adalah untuk menemukan individu – individu baru sehingga daerah pencarian akan semakin melebar. Akan tetapi terdapat suatu kondisi dimana proses pencarian solusi terjebak pada optimum lokal karena terjadinya konvergensi dini. Konvergensi dini mungkin terjadi karena pemilihan individu yang berdasarkan *fitness* terbaik saja sehingga pencarian solusi hanya terfokus pada satu bagian ruang pencarian dan tidak mengeksplor ke bagian lain. Proses mutasi dilakukan untuk mengatasi hal tersebut akan tetapi dengan kemungkinan yang rendah. Karena proses mutasi menghasilkan *offspring* dengan melakukan pengubahan secara acak pada satu gen saja. Jadi kemungkinannya adalah individu yang dihasilkan dapat memberikan *fitness* yang lebih baik atau malah mengubah kualitas individu tersebut menjadi lebih buruk (Desiani & Arhami, 2006).

Selain itu, pada setiap kali percobaan dengan menggunakan generasi sebanyak 100 sebagai *default,* kecepatan pada saat ditemukannya solusi akan berbeda - beda.

Pada saat pengujian nilai cr 0.8 dan mr 0.2 yang mana merupakan solusi paling optimal, dari 5 percobaan solusi paling optimal dapat ditemukan pada generasi antara 23 hingga 80 nilai tersebut sangatlah acak. Artinya pengaruh banyaknya generasi terhadap nilai cr & mr sangatlah sedikit. Oleh karena itu, untuk penentuan banyaknya generasi yang akan digunakan dilakukan dengan cara mengambil nilai generasi paling maksimal yang diperoleh dari keseluruhan pengujian kombinasi cr dan mr untuk digunakan pada pengujian berikutnya yaitu pengujian ukuran populasi. Yang mana pada pengujian ini nilai generasi paling maksimal adalah sebesar 95 artinya dari 100 generasi solusi paling optimal ditemukan pada saat generasi ke – 95. Nilai generasi paling maksimal ini digunakan sebagai generasi ideal untuk penemuan solusi paling optimal yang bisa diterapkan pada permasalahan ini. Semakin banyak nilai generasi akan membuat waktu komputasi semakin lama (Mahmudy, 2015). Maka jika banyaknya generasi diberikan lebih dari 95 padahal pada generasi 95 sudah bisa didapatkan solusi, maka hanya akan membuang waktu saja.

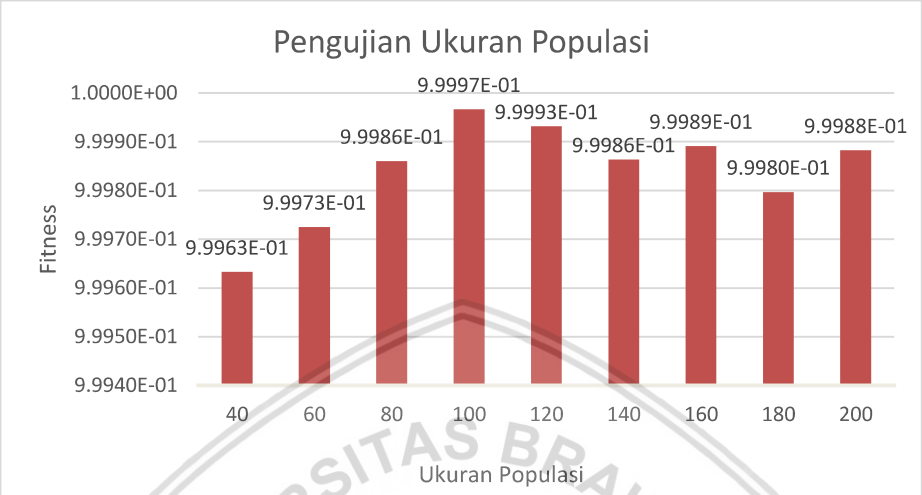
6.2 Pengujian Ukuran Populasi

Sesuai pada Tabel 4.28, pengujian ukuran populasi ini bertujuan untuk mendapatkan nilai ukuran populasi yang memberikan hasil paling optimal dengan nilai *fitness* terbaik. Terdapat 9 variasi kombinasi yang akan digunakan untuk melakukan pengujian. Masing – masing diuji dengan melakukan sebanyak 5 kali percobaan. Parameter lainnya di *set* dengan nilai tetap. Nilai kombinasi cr dan mr didapatkan dari hasil terbaik pengujian pertama yaitu nilai cr 0.8 dan nilai mr 0.2, kemudian banyaknya jumlah generasi didaptkan berdasarkan generasi maksimal pada pengujian pertama yaitu 95. Pada proses pengujian ini, banyaknya ukuran populasi akan diubah-ubah dari 40 hingga 200 populasi dengan interval setiap variasi yaitu 20.

Tabel 6.2 Nilai *Fitness* Hasil Pengujian Ukuran Populasi

Banyak Populasi	Nilai *Fitness* Percobaan Ke-										Nilai Rata-Rata *Fitness*
	1		2		3		4		5		
	Fitness	Gen	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	*Fitness*	Gen	
40	9.9987E-01	39	9.9886E-01	58	9.9993E-01	62	9.9963E-01	38	9.9987E-01	51	9.9963E-01
60	9.9978E-01	45	9.9988E-01	85	9.9982E-01	77	9.9982E-01	65	9.9933E-01	72	9.9973E-01
80	9.9988E-01	66	9.9989E-01	80	9.9996E-01	36	9.9995E-01	87	9.9963E-01	42	9.9986E-01
100	9.9986E-01	9	1.0000E+00	59	1.0000E+00	30	9.9999E-01	69	9.9999E-01	20	9.9997E-01
120	9.9989E-01	29	9.9996E-01	89	9.9993E-01	82	9.9997E-01	30	9.9991E-01	80	9.9993E-01
140	9.9970E-01	41	9.9995E-01	59	1.0000E+00	85	9.9972E-01	91	9.9995E-01	57	9.9986E-01
160	9.9997E-01	56	9.9974E-01	56	9.9989E-01	90	9.9988E-01	34	9.9997E-01	79	9.9989E-01
180	9.9984E-01	53	9.9942E-01	92	1.0000E+00	69	9.9979E-01	94	9.9994E-01	70	9.9980E-01
200	9.9990E-01	79	9.9978E-01	90	9.9981E-01	57	9.9998E-01	42	9.9995E-01	55	9.9988E-01

Tabel 6.2 menunjukkan hasil pengujian ukuran populasi. Berikut grafik hasil pengujian ukuran populasi yang ditunjukkan pada Gambar 6.2.



Gambar 6.2 Grafik Hasil Pengujian Ukuran Populasi

Berdasarkan grafik hasil pengujian ukuran populasi di atas menunjukkan bahwa semakin besar ukuran populasi belum tentu menghasilkan nilai *fitness* yang semakin optimal. Ketidakstabilan yang terjadi dapat disebabkan oleh nilai yang dibangkitkan secara *random* setiap kali proses dijalankan. Pada permasalahan ini rata – rata nilai *fitness* terkecil terdapat pada ukuran populasi sebesar 40. Sedangkan rata – rata nilai *fitness* yang memperoleh hasil paling optimal terdapat pada ukuran populasi sebesar 100 dengan nilai *fitness* sebesar $9.997x10^{-1}$. Pada rentang 40 hingga 100 nilai *fitness* mengalami kenaikan, akan tetapi ketika ukuran populasi diatas 100 nilai *fitness* yang dihasilkan mengalami naik turun. Semakin besar ukuran populasi mengakibatkan daerah yang dieksplorasi semakin luas sehingga membutuhkan waktu komputasi yang lama untuk pencarian solusi (Mahmudy, 2015). Untuk nilai generasi pada pengujian ini cara penentuan generasi paling ideal sama dengan pengujian sebelumnya yaitu dengan mengambil nilai generasi paling maksimal dari keseluruhan percobaan. Didapatkan nilai generasi paling maksimal adalah sebesar 95. Nilai tersebut akan digunakan sebagai nilai generasi paling ideal yang akan digunakan untuk melakukan prediksi beban listrik.

6.3 Pengujian Data Beban Listrik

Pengujian data beban listrik merupakan pengujian yang dilakukan untuk mencari pola input yang memberikan hasil *fitness* terbaik. Dengan menggunakan data yang sama akan tetapi berbeda pada penyusunan pola inputnya. Pola input yang diujikan berdasarkan jam, hari dan minggu berikut penjelasannya:

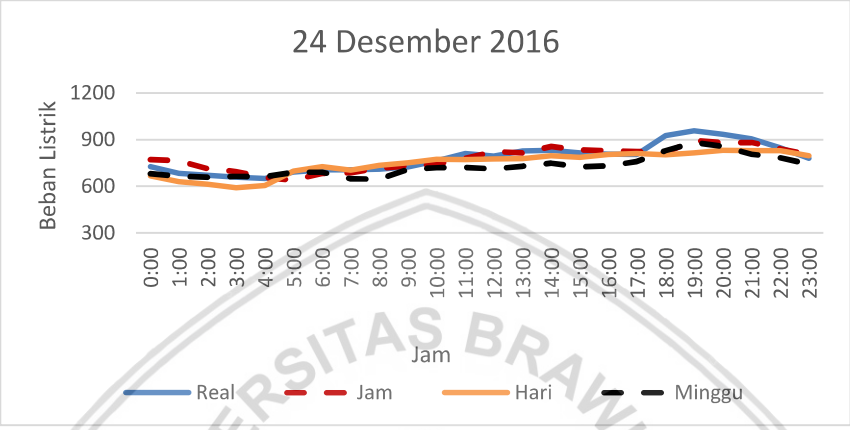
1. Pola input berdasarkan jam
   Yang menjadi data input merupakan data beban listrik 5 jam sebelumnya untuk memprediksi beban pada jam selanjutnya.
2. Pola input berdasarkan hari
   Yang menjadi data input merupakan data beban listrik 5 hari sebelumnya untuk memprediksi beban pada hari selanjutnya pada jam yang sama.
3. Pola input berdasarkan minggu
   Yang menjadi data input merupakan data beban listrik 5 minggu sebelumnya untuk memprediksi beban pada minggu selanjutnya pada hari dan jam yang sama.

Dengan menggunakan parameter kombinasi nilai cr & mr, ukuran populasi serta generasi terbaik yang sudah didapatkan dari pengujian 1 dan pengujian 2. Berikut adalah hasil pengujian data beban listrik pada tanggal 24 Desember 2016 yang digunakan sebagai *sample* ditunjukkan pada Tabel 6.3.

Tabel 6.3 Hasil Pengujian Data Beban Listrik

Waktu	Real (MW)	JAM (MW)	JAM *Error* %	HARI (MW)	HARI *Error* %	MINGGU (MW)	MINGGU *Error* %
0:00	726.96	771.22	-6.09	665.74	8.42	680.87	6.34
1:00	683.36	764.59	-11.89	628.93	7.96	665.55	2.61
2:00	670.36	713.38	-6.42	611.57	8.77	656.83	2.02
3:00	658.08	693.82	-5.43	590.53	10.27	664.00	-0.90
4:00	650.68	656.08	-0.83	605.48	6.95	660.86	-1.56
5:00	690.97	639.82	7.40	696.80	-0.84	691.11	-0.02
6:00	705.07	682.18	3.25	726.06	-2.98	690.27	2.10
7:00	707.06	687.51	2.76	703.85	0.45	648.48	8.29
8:00	710.8	716.61	-0.82	734.24	-3.30	645.88	9.13
9:00	725.2	730.67	-0.75	751.11	-3.57	708.60	2.29
10:00	763.64	742.84	2.72	774.79	-1.46	718.66	5.89
11:00	810.83	780.16	3.78	773.30	4.63	722.27	10.92
12:00	793.93	823.56	-3.73	775.92	2.27	711.41	10.39
13:00	826.75	813.96	1.55	778.47	5.84	729.86	11.72
14:00	832.38	855.57	-2.79	795.52	4.43	747.45	10.20
15:00	814.56	832.26	-2.17	785.03	3.63	725.87	10.89
16:00	806.66	829.31	-2.81	804.51	0.27	730.07	9.50
17:00	804.76	824.44	-2.45	811.99	-0.90	759.17	5.66
18:00	926.76	815.45	12.01	803.35	13.32	830.20	10.42
19:00	958.06	894.15	6.67	814.53	14.98	881.41	8.00
20:00	934.86	879.76	5.89	831.35	11.07	855.46	8.49
21:00	904.96	881.19	2.63	828.39	8.46	805.88	10.95
22:00	847.76	841.55	0.73	828.60	2.26	783.72	7.55
23:00	780.86	803.86	-2.94	796.56	-2.01	743.22	4.82
Rata – Rata MAPE			**0.0120**		4.1211		6.4875

Dari Tabel 6.1 dapat kita lihat hasil nilai rata – rata MAPE terbaik yang diperoleh sebesar 0.0120% dan didapatkan dari prediksi beban listrik dengan menggunakan pola input berdasarkan data 5 jam sebelumnya. Dapat ditarik kesimpulan bahwa semakin dekat jarak waktu dapat meningkatkan tingkat akurasi yang diperoleh. Dibawah ini merupakan grafik perbandingan hasil pengujian data beban listrik.



Gambar 6.3 Grafik Hasil Pengujian Data Berdasarkan Waktu

6.4 Hasil Prediksi Beban Listrik Menggunakan Metode *Extreme Learning Machine*

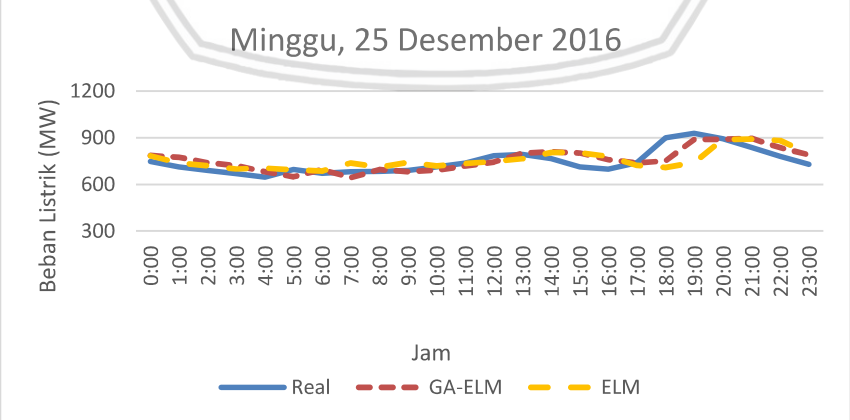
Setelah dilakukan beberapa pengujian sebelumnya, didapatkan hasil nilai terbaik dari beberapa parameter yang diuji yaitu, kombinasi cr & mr, ukuran populasi dan data beban listrik. Hasil tersebut akan digunakan untuk menghitung hasil prediksi beban listrik dengan menggunakan metode *Extreme Learning Machine.* Nilai hasil parameter terbaik tersebut adalah:

1. *Crossover Rate* : 0.8
2. *Mutation Rate* : 0.2
3. Ukuran Populasi : 100
4. Jumlah Generasi : 95
5. Pola input data : Jam

Dengan menggunakan nilai hasil yang paling optimal tersebut, maka didapatkan hasil prediksi beban listrik yang ditunjukkan pada Gambar grafik dibawah ini. Data yang ditampilkan merupakan data dari tanggal 25 Desember – 31 Desember 2016 yang diilustrasikan secara grafis dan dibandingkan dengan nilai aktual dari beban pada jam yang sama. Selain itu hasil prediksi dengan menggunakan metode GA-ELM juga dibandingkan dengan hasil prediksi yang hanya dengan menggunakan metode ELM biasa.

Tabel 6.4 Hasil Prediksi Beban 25 Desember 2016

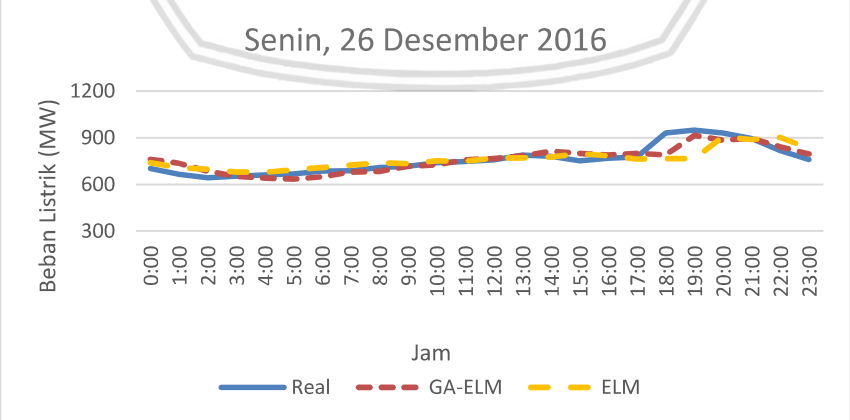
Waktu	Real (MW)	GA-ELM (MW)	ELM (MW)	GA-ELM *Error %*	ELM *Error %*
0:00	748.7	786.02	784.61	-4.98	-4.80
1:00	711.6	774.25	737.98	-8.80	-3.71
2:00	690.5	738.72	719.81	-6.98	-4.24
3:00	668.31	719.47	699.22	-7.65	-4.63
4:00	646.49	682.28	704.87	-5.54	-9.03
5:00	696.7	648.29	692.95	6.95	0.54
6:00	671.84	695.47	684.96	-3.52	-1.95
7:00	681.21	643.14	737.19	5.59	-8.22
8:00	683.1	695.45	710.53	-1.81	-4.02
9:00	689.5	682.16	741.32	1.06	-7.52
10:00	711.5	690.92	717.46	2.89	-0.84
11:00	735.8	718.21	733.41	2.39	0.33
12:00	784.3	744.30	748.09	5.10	4.62
13:00	793.1	801.94	765.06	-1.12	3.54
14:00	764.6	809.64	805.19	-5.89	-5.31
15:00	711.6	801.06	802.94	-12.57	-12.84
16:00	698.1	758.00	780.30	-8.58	-11.78
17:00	738.6	737.40	722.83	0.16	2.14
18:00	900.6	750.77	708.58	16.64	21.32
19:00	926.7	889.29	741.43	4.04	19.99
20:00	892	889.34	887.86	0.30	0.46
21:00	838.19	895.36	891.72	-6.82	-6.39
22:00	780.69	835.02	882.49	-6.96	-13.04
23:00	729.6	789.85	797.12	-8.26	-9.25
Rata-rata MAPE				**1.8486**	2.2757



Gambar 6.4 Grafik Hasil Prediksi Beban 25 Desember 2016

Tabel 6.5 Hasil Prediksi Beban 26 Desember 2016

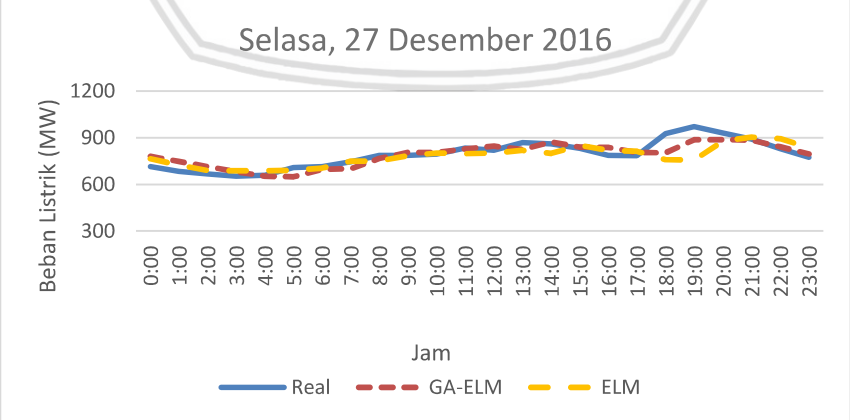
Waktu	Real (MW)	GA-ELM (MW)	ELM (MW)	GA-ELM *Error* %	ELM *Error* %
0:00	701.75	761.36	740.17	-8.49	-5.48
1:00	664.39	735.69	705.75	-10.73	-6.23
2:00	641.34	684.69	697.59	-6.76	-8.77
3:00	653.21	649.99	679.61	0.49	-4.04
4:00	662.09	641.54	677.62	3.10	-2.35
5:00	667.88	632.55	694.13	5.29	-3.93
6:00	686.81	648.49	709.20	5.58	-3.26
7:00	689.05	679.99	723.54	1.31	-5.01
8:00	707.77	683.25	736.87	3.46	-4.11
9:00	716.90	717.26	733.30	-0.05	-2.29
10:00	737.90	725.26	751.10	1.71	-1.79
11:00	748.30	756.27	749.09	-1.07	-0.11
12:00	757.00	767.14	768.87	-1.34	-1.57
13:00	788.70	782.93	769.46	0.73	2.44
14:00	781.30	812.43	775.49	-3.98	0.74
15:00	752.70	799.83	796.61	-6.26	-5.83
16:00	766.80	789.95	783.97	-3.02	-2.24
17:00	775.30	799.79	763.42	-3.16	1.53
18:00	929.48	789.32	767.20	15.08	17.46
19:00	948.28	915.30	764.21	3.48	19.41
20:00	930.38	884.42	905.07	4.94	2.72
21:00	894.93	895.67	888.22	-0.08	0.75
22:00	818.43	840.75	901.10	-2.73	-10.10
23:00	759.79	795.73	832.51	-4.73	-9.57
Rata-rata MAPE				**0.3007**	1.3169



Gambar 6.5 Grafik Hasil Prediksi Beban 26 Desember 2016

Tabel 6.6 Hasil Prediksi Beban 27 Desember 2016

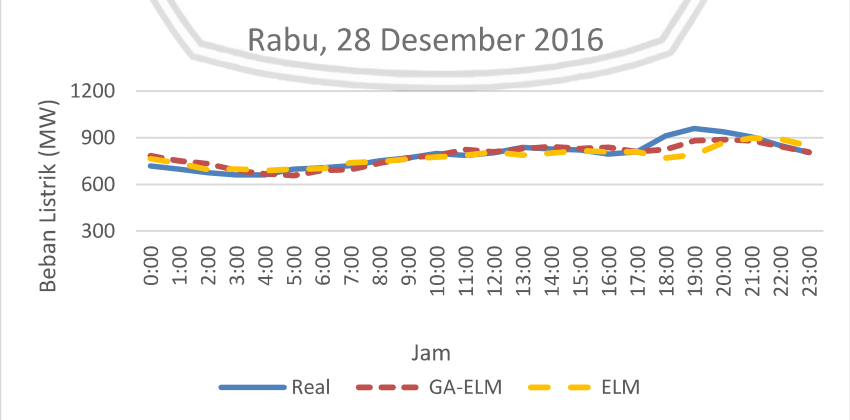
Waktu	Real (MW)	GA-ELM (MW)	ELM (MW)	GA-ELM *Error %*	ELM *Error %*
0:00	714.10	780.40	763.77	-9.28	-6.96
1:00	684.90	746.52	724.36	-9.00	-5.76
2:00	666.50	715.35	690.17	-7.33	-3.55
3:00	654.26	681.06	687.72	-4.10	-5.11
4:00	660.02	652.98	688.33	1.07	-4.29
5:00	707.69	649.46	690.99	8.23	2.36
6:00	713.53	697.39	703.81	2.26	1.36
7:00	743.70	701.46	749.39	5.68	-0.76
8:00	787.60	766.77	751.67	2.65	4.56
9:00	787.30	805.54	783.57	-2.32	0.47
10:00	794.50	805.74	800.97	-1.41	-0.81
11:00	833.40	827.58	797.31	0.70	4.33
12:00	818.90	845.42	800.90	-3.24	2.20
13:00	870.40	823.92	818.38	5.34	5.98
14:00	860.40	872.79	799.04	-1.44	7.13
15:00	830.54	838.79	851.48	-0.99	-2.52
16:00	787.15	838.00	818.71	-6.46	-4.01
17:00	784.40	805.35	812.34	-2.67	-3.56
18:00	925.34	802.97	759.32	13.22	17.94
19:00	970.96	887.27	756.93	8.62	22.04
20:00	930.07	887.32	877.97	4.60	5.60
21:00	891.03	883.57	902.78	0.84	-1.32
22:00	828.32	841.67	892.94	-1.61	-7.80
23:00	774.15	794.95	835.07	-2.69	-7.87
Rata-rata MAPE				**0.0272**	0.8185



Gambar 6.6 Grafik Hasil Prediksi Beban 27 Desember 2016

Tabel 6.7 Hasil Prediksi Beban 28 Desember 2016

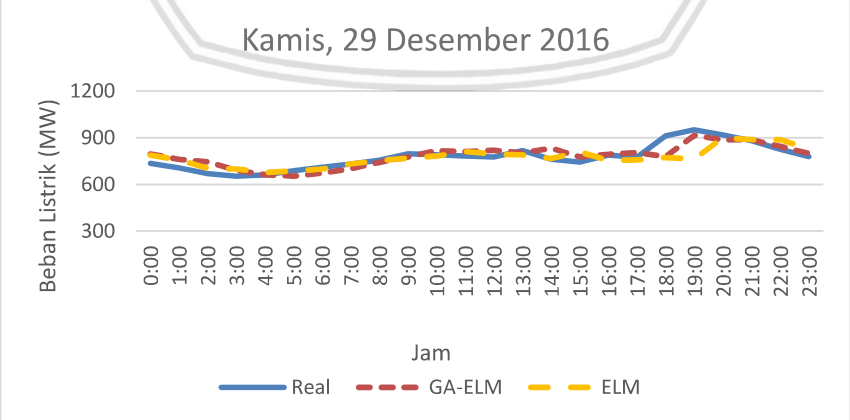
Waktu	Real (MW)	GA-ELM (MW)	ELM (MW)	GA-ELM *Error* %	ELM *Error* %
0:00	718.67	784.27	764.40	-9.13	-6.36
1:00	698.84	751.54	734.79	-7.54	-5.14
2:00	674.10	733.22	695.28	-8.77	-3.14
3:00	661.35	688.78	698.72	-4.15	-5.65
4:00	662.17	669.10	688.41	-1.05	-3.96
5:00	698.24	656.89	697.38	5.92	0.12
6:00	705.55	690.23	701.30	2.17	0.60
7:00	720.77	694.92	739.60	3.59	-2.61
8:00	752.11	736.58	744.73	2.07	0.98
9:00	769.30	770.85	763.27	-0.20	0.78
10:00	798.82	786.35	776.23	1.56	2.83
11:00	787.41	823.47	785.60	-4.58	0.23
12:00	802.00	809.19	809.85	-0.90	-0.98
13:00	838.19	830.41	787.17	0.93	6.09
14:00	826.88	842.24	800.61	-1.86	3.18
15:00	820.78	830.16	817.03	-1.14	0.46
16:00	794.44	838.44	808.28	-5.54	-1.74
17:00	810.52	809.94	808.53	0.07	0.25
18:00	911.36	824.70	770.98	9.51	15.40
19:00	957.98	879.28	788.70	8.22	17.67
20:00	939.42	888.40	866.53	5.43	7.76
21:00	906.24	880.19	897.86	2.87	0.93
22:00	850.32	844.54	891.31	0.68	-4.82
23:00	803.62	806.32	847.38	-0.34	-5.45
Rata-rata MAPE				**0.0905**	0.7255



Gambar 6.7 Grafik Hasil Prediksi Beban 28 Desember 2016

Tabel 6.8 Hasil Prediksi Beban 29 Desember 2016

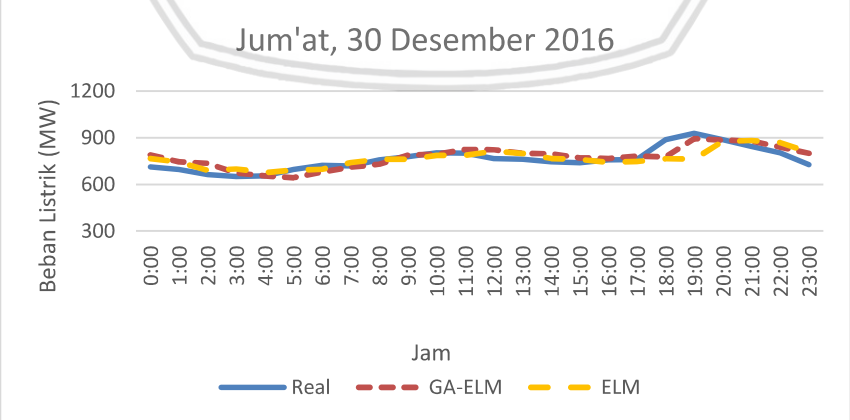
Waktu	Real (MW)	GA-ELM (MW)	ELM (MW)	GA-ELM *Error* %	ELM *Error* %
0:00	735.14	795.77	786.85	-8.25	-7.03
1:00	704.92	761.33	756.83	-8.00	-7.36
2:00	668.96	745.87	704.33	-11.50	-5.29
3:00	652.46	690.46	699.81	-5.82	-7.26
4:00	661.71	663.11	675.47	-0.21	-2.08
5:00	688.58	653.16	685.29	5.14	0.48
6:00	709.53	671.83	699.13	5.31	1.47
7:00	730.25	700.95	730.98	4.01	-0.10
8:00	756.43	740.74	752.61	2.07	0.51
9:00	797.25	776.14	768.89	2.65	3.56
10:00	791.13	817.29	781.78	-3.31	1.18
11:00	782.97	810.79	809.87	-3.55	-3.44
12:00	774.89	817.97	796.22	-5.56	-2.75
13:00	817.84	802.38	789.90	1.89	3.42
14:00	761.54	831.53	766.99	-9.19	-0.72
15:00	742.15	776.79	808.27	-4.67	-8.91
16:00	787.83	793.79	753.81	-0.76	4.32
17:00	772.33	805.20	756.44	-4.26	2.06
18:00	912.61	778.53	772.86	14.69	15.31
19:00	951.52	915.84	763.69	3.75	19.74
20:00	917.52	886.09	899.28	3.43	1.99
21:00	880.58	885.37	887.73	-0.54	-0.81
22:00	825.04	843.78	888.26	-2.27	-7.66
23:00	779.54	800.05	828.72	-2.63	-6.31
Rata-rata MAPE				1.1488	**0.2374**



Gambar 6.8 Grafik Hasil Prediksi Beban 29 Desember 2016

Tabel 6.9 Hasil Prediksi Beban 30 Desember 2016

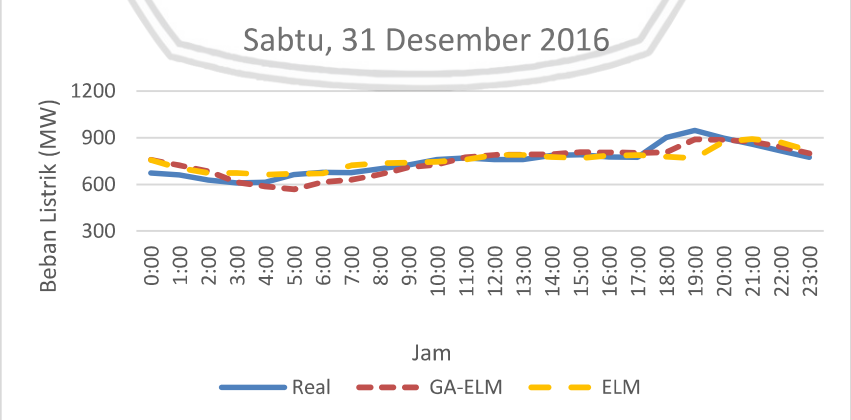
Waktu	Real (MW)	GA-ELM (MW)	ELM (MW)	GA-ELM *Error %*	ELM *Error %*
0:00	710.80	789.06	767.42	-11.01	-7.97
1:00	695.50	745.85	742.78	-7.24	-6.80
2:00	663.90	735.17	690.72	-10.73	-4.04
3:00	650.00	673.89	699.62	-3.68	-7.63
4:00	655.50	655.15	676.34	0.05	-3.18
5:00	695.20	643.52	691.18	7.43	0.58
6:00	722.30	679.30	696.23	5.95	3.61
7:00	718.38	710.64	738.81	1.08	-2.84
8:00	757.68	729.84	762.69	3.67	-0.66
9:00	779.87	787.64	760.08	-1.00	2.54
10:00	801.46	792.86	785.78	1.07	1.96
11:00	799.28	824.74	787.33	-3.19	1.50
12:00	766.04	822.20	811.85	-7.33	-5.98
13:00	761.55	798.51	798.47	-4.85	-4.85
14:00	746.73	796.52	766.79	-6.67	-2.69
15:00	737.88	769.44	760.47	-4.28	-3.06
16:00	758.03	766.80	742.07	-1.16	2.11
17:00	759.51	781.74	747.22	-2.93	1.62
18:00	887.07	776.31	764.87	12.49	13.78
19:00	927.60	893.50	762.46	3.68	17.80
20:00	886.45	883.36	876.59	0.35	1.11
21:00	843.89	877.61	881.36	-4.00	-4.44
22:00	801.30	840.24	868.08	-4.86	-8.33
23:00	727.80	798.88	807.23	-9.77	-10.91
Rata-rata MAPE				1.9541	**1.1164**



Gambar 6.9 Grafik Hasil Prediksi Beban 30 Desember 2016

Tabel 6.10 Hasil Prediksi Beban 31 Desember 2016

Waktu	Real (MW)	GA-ELM (MW)	ELM (MW)	GA-ELM *Error %*	ELM *Error %*
0:00	673.26	759.16	756.14	-12.76	-12.31
1:00	662.26	721.66	705.00	-8.97	-6.45
2:00	627.96	684.76	674.48	-9.04	-7.41
3:00	609.86	613.61	674.20	-0.61	-10.55
4:00	613.66	586.55	661.61	4.42	-7.81
5:00	662.86	567.65	668.71	14.36	-0.88
6:00	676.90	613.36	670.50	9.39	0.95
7:00	674.79	628.64	722.68	6.84	-7.10
8:00	700.12	664.53	734.98	5.08	-4.98
9:00	725.58	708.47	739.20	2.36	-1.88
10:00	759.70	727.26	746.09	4.27	1.79
11:00	769.45	774.06	757.86	-0.60	1.51
12:00	759.88	790.05	789.21	-3.97	-3.86
13:00	759.88	792.31	788.70	-4.27	-3.79
14:00	786.84	793.27	776.96	-0.82	1.26
15:00	791.08	807.66	767.51	-2.10	2.98
16:00	774.21	805.68	785.44	-4.07	-1.45
17:00	774.05	803.53	788.56	-3.81	-1.87
18:00	903.00	804.98	778.90	10.86	13.74
19:00	947.50	890.16	768.73	6.05	18.87
20:00	895.90	885.46	872.34	1.17	2.63
21:00	857.80	874.70	891.90	-1.97	-3.98
22:00	816.40	841.69	870.12	-3.10	-6.58
23:00	773.90	799.17	815.74	-3.27	-5.41
	Rata-rata MAPE			**0.2271**	1.7747



Gambar 6.10 Grafik Hasil Prediksi Beban 31 Desember 2016

Berdasarkan keseluruhan tabel –tabel diatas yang merupakan perbandingan hasil predikasi beban dengan menggunakan metode ELM yang dioptimasi dengan algoritme genetika dan metode ELM biasa. Maka didapatkan hasil rata – rata MAPE selama satu minggu yang ditunjukkan pada Tabel 6.11 berikut ini.

Tabel 6.11 Rata-rata MAPE Hasil Prediksi GA-ELM dan ELM

Tanggal	GA-ELM	ELM	Keterangan
	Error %	*Error %*	
25-Dec-2016	**1.8486**	2.2757	GA-ELM < ELM
26-Dec-2016	**0.3007**	1.3169	GA-ELM < ELM
27-Dec-2016	**0.0272**	0.8185	GA-ELM < ELM
28-Dec-2016	**0.0905**	0.7255	GA-ELM < ELM
29-Dec-2016	1.1488	**0.2374**	GA-ELM > ELM
30-Dec-2016	1.9541	**1.1164**	GA-ELM > ELM
31-Dec-2016	**0.2271**	1.7747	GA-ELM < ELM
Rata-rata MAPE%	**0.7996**	1.1807	GA-ELM < ELM

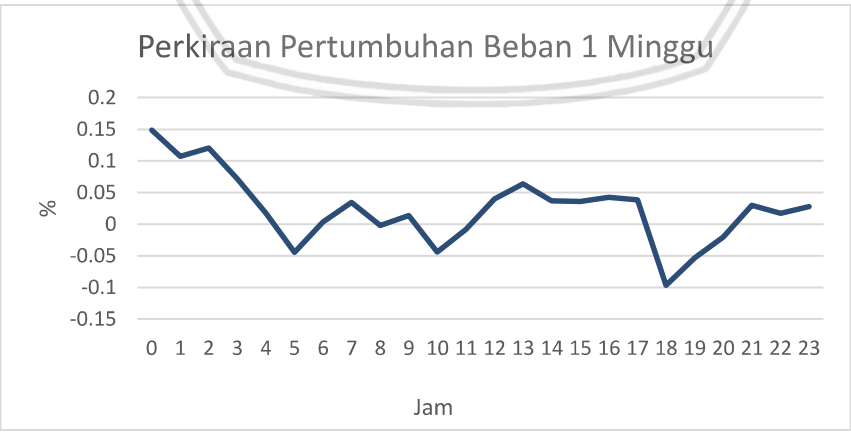
Dapat disimpulkan bahwa dari hasil perhitungan rata – rata MAPE diatas prediksi beban listrik selama satu minggu dengan menggunakan metode GA-ELM mampu memberikan solusi yang lebih baik daripada solusi yang dihasilkan hanya dengan menggunakan metode ELM tanpa dilakukan optimasi. Rata – rata MAPE hasil prediksi menggunakan metode GA-ELM sebesar 0.7996% sedangkan rata – rata MAPE hasil prediksi dengan menggunakan metode ELM biasa adalah sebesar 1.1807%. Demikian juga dengan nilai *error* tertinggi prediksi GA-ELM adalah 1.9541% sedangkan prediksi ELM adalah 2.2757%. Nilai *error* yang dihasilkan oleh metode ELM biasa dapat dipengaruhi oleh beberapa faktor salah satunya adalah bobot awal yang dibangkitkan secara acak yang masih belum optimal.

6.5 Analisis Perkiraan Pertumbuhan Beban Listrik

Subbab ini akan menjelaskan mengenai analisis perkiraan pertumbuhan beban listrik berdasarkan hasil prediksi. Analisis akan dilakukan selama satu minggu hingga dua minggu, dan hari yang digunakan sebagai patokan adalah tanggal 31 November 2016. Sedangkan hasil prediksi adalah dari tanggal 1 Desember 2016 – 14 Desember 2016. Dibawah ini merupakan hasil analisis perkiraan pertumbuhan beban selama satu minggu yang ditunjukkan pada Tabel 6.12 dan Gambar 6.11.

Tabel 6.12 Perkiraan Pertumbuhan Beban dalam 1 Minggu

31 Nov 2016	Perkiraan Pertumbuhan Beban (%)							Rata – Rata Pertumbuhan Beban
	1 Des 2016	2 Des 2016	3 Des 2016	4 Des 2016	5 Des 2016	6 Des 2016	7 Des 2016	
673.26	13.70%	15.15%	15.75%	15.40%	17.06%	13.57%	13.33%	↑ 14.85%
662.26	8.40%	15.54%	12.23%	5.29%	10.08%	8.38%	15.05%	↑ 10.71%
627.96	11.75%	16.77%	14.47%	8.74%	13.86%	7.23%	11.53%	↑ 12.05%
609.86	5.36%	15.80%	9.35%	1.19%	7.98%	0.85%	9.66%	↑ 7.17%
613.66	1.88%	8.81%	4.37%	-5.88%	1.12%	-4.32%	5.89%	↑ 1.70%
662.86	-3.04%	1.96%	-1.33%	-9.46%	-6.47%	-8.66%	-4.16%	↓ -4.45%
676.90	1.72%	4.60%	3.38%	-10.36%	1.59%	-0.82%	2.33%	↑ 0.35%
674.79	6.07%	6.78%	4.04%	-4.34%	2.32%	0.89%	8.07%	↑ 3.41%
700.12	1.11%	3.82%	-0.15%	-11.30%	2.41%	1.66%	1.02%	↓ -0.20%
725.58	0.89%	4.25%	0.01%	-10.15%	7.86%	3.56%	3.10%	↑ 1.36%
759.7	-3.25%	-0.37%	-7.20%	-16.57%	-0.22%	-0.02%	-3.22%	↓ -4.41%
769.45	-0.54%	1.24%	-1.45%	-14.78%	7.20%	2.84%	-0.26%	↓ -0.82%
759.88	5.04%	5.71%	0.72%	-10.48%	11.81%	10.90%	4.29%	↑ 4.00%
759.88	8.65%	1.75%	3.59%	-4.69%	13.62%	11.56%	9.94%	↑ 6.35%
786.84	9.58%	3.53%	-2.83%	-7.27%	9.56%	7.97%	5.47%	↑ 3.72%
791.08	7.87%	1.27%	-5.22%	-1.02%	3.50%	9.81%	9.07%	↑ 3.61%
774.21	11.03%	3.76%	-3.31%	-1.69%	1.09%	9.43%	9.27%	↑ 4.23%
774.05	6.30%	4.53%	-3.74%	0.49%	0.41%	9.66%	9.43%	↑ 3.87%
903	-8.00%	-8.33%	-11.16%	-12.91%	-12.13%	-7.61%	-7.59%	↓ -9.68%
947.5	-4.92%	-5.60%	-5.50%	-3.73%	-7.12%	-5.82%	-4.42%	↓ -5.30%
895.9	-1.72%	-0.22%	-2.80%	-2.74%	-0.18%	-3.33%	-3.41%	↓ -2.06%
857.8	2.71%	3.40%	2.98%	6.02%	2.84%	1.35%	1.69%	↑ 3.00%
816.4	0.72%	2.22%	1.28%	3.79%	2.63%	0.57%	0.80%	↑ 1.71%
773.9	3.11%	4.11%	1.26%	4.33%	3.04%	1.68%	1.79%	↑ 2.76%

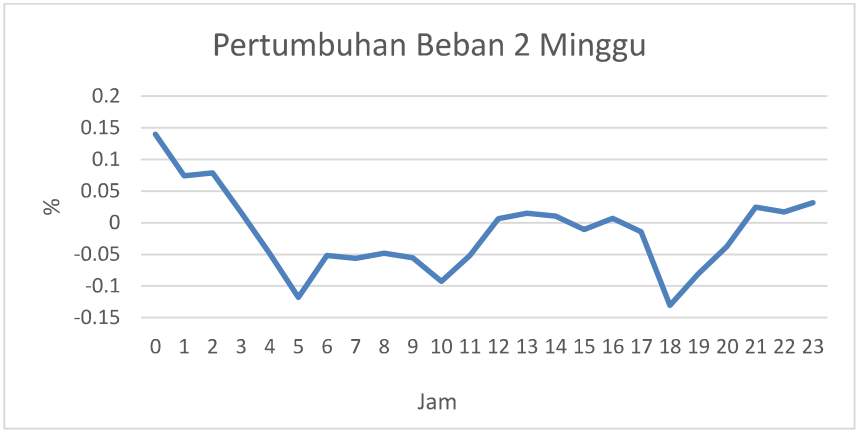


Gambar 6.11 Perkiraan Pertumbuhan Beban dalam 1 Minggu

Berdasarkan hasil prediksi beban dapat dilihat perkiraan pertumbuhan beban untuk satu minggu kedepan bahkan dua minggu. Tabel 6.12 dan Gambar 6.11 menunjukkan perkiraan pertumbuhan beban selama satu minggu, dari hasil prediksi beban tersebut ada yang mengalami kenaikan dan ada juga yang mengalami penurunan. Angka positif menunjukkan kenaikan, sedangkan angka negatif menunjukkan penurunan. Dalam satu minggu rata – rata kenaikan beban paling besar adalah sebesar 14.85%, sedangkan rata – rata penurunan beban paling besar adalah 9.68%. Akan tetapi secara keseluruhan perkiraan beban selama satu minggu pada tanggal 1 Desember hingga 7 Desember 2016 banyak mengalami kenaikan.

Tabel 6.13 Perkiraan Pertumbuhan Beban dalam 2 Minggu

31 Nov 2016	Perkiraan Pertumbuhan Beban (%)							Rata – Rata Pertumbuhan Beban
	8 Des 2016	9 Des 2016	10 Des 2016	11 Des 2016	12 Des 2016	13 Des 2016	14 Des 2016	
673.26	16.56%	16.32%	18.01%	17.17%	11.34%	1.59%	17.18%	↑ 14.03%
662.26	11.95%	10.69%	6.08%	14.39%	5.04%	-8.10%	12.07%	↑ 7.45%
627.96	17.14%	11.91%	7.99%	13.88%	1.16%	-10.36%	13.12%	↑ 7.84%
609.86	11.76%	5.61%	-1.16%	9.57%	-5.21%	-15.95%	6.23%	↑ 1.55%
613.66	6.39%	-0.56%	-8.56%	2.72%	-10.62%	-23.59%	-0.12%	↓ -4.91%
662.86	-0.82%	-7.21%	-11.61%	-6.03%	-21.78%	-29.26%	-5.86%	↓ -11.80%
676.90	4.47%	-1.26%	-2.98%	0.77%	-17.79%	-19.00%	-0.67%	↓ -5.21%
674.79	3.51%	-1.49%	-2.96%	-8.25%	-16.67%	-15.06%	1.47%	↓ -5.64%
700.12	0.95%	-1.35%	-5.02%	-4.28%	-11.75%	-13.31%	1.05%	↓ -4.81%
725.58	2.63%	-0.75%	-8.52%	-13.92%	-14.81%	-7.30%	4.10%	↓ -5.51%
759.7	-2.85%	-3.98%	-13.32%	-19.84%	-13.60%	-8.19%	-3.20%	↓ -9.29%
769.45	4.25%	3.74%	-9.47%	-21.17%	-14.70%	-4.76%	5.66%	↓ -5.21%
759.88	8.53%	9.18%	-0.92%	-17.50%	-11.05%	7.11%	8.97%	↑ 0.62%
759.88	10.07%	8.71%	0.52%	-18.87%	-10.74%	8.85%	11.78%	↑ 1.47%
786.84	11.39%	7.26%	3.03%	-18.85%	-16.59%	10.23%	10.58%	↑ 1.01%
791.08	8.18%	5.11%	0.69%	-17.35%	-17.25%	7.73%	5.27%	↓ -1.09%
774.21	10.62%	7.72%	4.68%	-16.78%	-17.43%	8.62%	7.40%	↑ 0.69%
774.05	5.10%	6.70%	1.72%	-14.33%	-19.47%	6.06%	3.94%	↓ -1.47%
903	-11.08%	-9.17%	-9.60%	-18.75%	-23.82%	-9.39%	-9.76%	↓ -13.08%
947.5	-8.30%	-6.92%	-6.00%	-6.65%	-12.56%	-7.06%	-8.79%	↓ -8.04%
895.9	-5.02%	-4.76%	-2.28%	-1.01%	-4.00%	-5.28%	-4.08%	↓ -3.78%
857.8	2.72%	2.16%	3.15%	4.02%	1.15%	1.88%	2.09%	↑ 2.45%
816.4	2.46%	2.17%	1.97%	2.08%	-0.99%	1.56%	2.49%	↑ 1.68%
773.9	4.08%	4.26%	5.59%	0.63%	-2.63%	4.85%	5.24%	↑ 3.15%



Gambar 6.12 Pertumbuhan Beban dalam 2 Minggu

Dari Tabel 6.13 dan Gambar 6.12 dapat kita lihat perkiraan pertumbuhan beban dalam dua minggu. Dalam dua minggu rata – rata kenaikan beban paling besar adalah sebesar 14.03%, sedangkan rata – rata penurunan beban paling besar adalah 13.08%. Akan tetapi secara keseluruhan perkiraan beban selama dua minggu pada tanggal 8 Desember hingga 14 Desember 2016 banyak mengalami penurunan.

BAB 7 PENUTUP

Pada bab penutup ini akan membahas terkait kesimpulan dan saran berdasarkan hasil penelitian optimasi bobot pada metode *extreme learning machine* untuk prediksi beban listrik menggunakan algoritme genetika. Yang mana kesimpulan mengacu pada rumusan masalah yang sudah dibuat sebelumnya. Selain itu, pada bab ini juga akan dibahas terkait saran untuk perbaikan dari penelitian ini guna sebagai masukan untuk penelitian yang berkaitan dengan prediksi beban listrik atau penerapan metode ELM maupun algoritme genetika.

7.1 Kesimpulan

Dari hasil pengujian dan analisis hasil yang sudah dilakukan pada bab sebelumnya, maka dapat ditarik beberapa kesimpulan sebagai berikut:

1. Algoritme genetika mampu dengan baik melakukan optimasi bobot yang ada pada metode *extreme learning machine*. Parameter yang terdapat pada algoritme genetika dapat mempengaruhi tingkat akurasi dan lama waktu komputasi metode *extreme learning machine.* Besarnya ukuran populasi dan jumlah generasi sangat mempengaruhi, semakin besar ukuran populasi dan jumlah generasi akan membuat waktu komputasi dalam melakukan prediksi dengan metode ELM membutuhkan waktu yang lama. Akan tetapi tingkat akurasi yang dihasilkan akan lebih baik jika pemilihan parameternya tepat. Dalam penelitian ini berdasarkan hasil pengujian sebelumnya didapatkan kombinasi nilai cr dan mr, ukuran populasi, pola input data yang menghasilkan solusi paling optimal. Kombinasi nilai cr dan mr yang paling optimal adalah cr 0.8 dan mr 0.2 dengan nilai rata-rata *fitness* sebesar 9.998×10^{-1}. Ukuran populasi dengan nilai rata – rata *fitness* sebesar 9.997×10^{-1} diperoleh dari ukuran populasi sebesar 100. Untuk jumlah generasi didapatkan dari hasil pengujian 1 dan 2 bahwa generasi paling ideal dalam pencarian solusi permasalahn ini adalah 95. Sedangkan pada pengujian data beban listrik didapatkan hasil nilai rata-rata MAPE terbaik sebesar 0.0120% dari pola input data berdasarkan 5 jam sebelumnya.

2. Setelah melakukan penelitian mengenai optimasi bobot pada *extreme learning machine* menggunakan algoritme genetika untuk memprediksi beban listrik, hasil yang diperoleh lebih baik dibandingkan hanya dengan melakukan prediksi menggunakan metode ELM tanpa optimasi bobot. Nilai rata – rata *error* dengan optimasi bobot yang didapatkan adalah sebesar 0.7996% dengan simpangan sebesar 35.5014 MW sedangkan nilai rata – rata *error* tanpa optimasi adalah sebesar 1.1807% dengan simpangan sebesar 41.2630 MW. Hal tersebut menunjukkan bahwa metode GA-ELM dapat diterapkan dengan lebih baik untuk melakukan prediksi beban listrik dibandingkan metode ELM biasa.

7.2 Saran

Dalam melakukan penelitian ini masih terdapat beberapa kekurangan pada solusi yang dihasilkan, sehingga penulis memberikan beberapa saran untuk penelitian selanjutnya, sebagai berikut:

1. Pada penelitian ini hanya menggunakan satu variabel masukan yaitu beban listrik perjam, untuk penelitian yang akan datang dapat ditambahkan beberapa variabel lain yang dapat mendukung prediksi seperti jumlah pelanggan, total biaya, dan variabel lain. Sehingga prediksi dapat diperluas lagi tujuannya untuk mengoptimalkan biaya yang dapat dihemat.

2. Pada penelitian selanjutnya dapat menggunakan *hybrid* metode lain untuk melakukan perbandingan dengan metode GA-ELM agar dapat membandingkan metode mana yang lebih baik.

DAFTAR PUSTAKA

Alencar, Alisson S.C., Ajalmar R. Rocha Neto, dan Joao Paulo P. Gomes. 2016. *A New Pruning Method for Extreme Learning Machine via Genetic Algorithms.* Applied Soft Computing 1568-4946.

Azizah, Alvia Nur. 2016. Penentuan Kualitas Air Sungai Menggunakan Metode Extreme Learning Machine. Fakultas Ilmu Komputer Universitas Brawijaya. Malang.

Back, Thomas. 1996. *Evolutionary Algorithms in Theory and Practice*. New York. Oxford University Press.

Desiani, A., & Arhami, M. 2006. Konsep Kecerdasan Buatan. Yogyakarta: Penerbit: Andi.

Direktorat Jenderal Ketenagalistrikan. 2016. Statistik Ketenagalistrikan 2015.[pdf] Kementrian Energi dan Sumber Daya Mineral. Tersedia di: https://djk.esdm.go.id/pdf/Buku%20Statistik%20Ketenagalistrikan/Statistik%20Ketenagalistrikan%20T.A.%202016.pdf[Diakses 15 Oktober 2017].

Gani, Irwan Dan Amalia, Siti. 2015. Alat Dan Analisis Data; Aplikasi Statistik Untuk Penelitian Bidang Ekonomi Dan Sosial.Andi Yogyakarta.

Gondro, C. & Kinghorn, B., 2008. Application of Evolutionary Algorithms to solve complex problems in Quantitative Genetics and Bioinformatics. Armidale: University of New England.

Hasim, Agus. 2008. Prakiraan Beban Listrik Kota Pontianak Dengan Jaringan Syaraf Tiruan. Sekolah Pasca Sarjana Institut Pertanian Bogor. Bogor.

Herrera, F, M. Lozano, dan J.L. Vergeday. 1998. *Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis*. *Artificial Intelligence Review* 12: 265-319, 1998.

Huang, Guang-Bin, Qin-Yu Zhu, dan Chee-Kheong Siew. 2004. *Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks. Proceedings of International Joint Conference on Neural Networks.* Budapest, Hungary, 25-29 Juli 2004. Singapura: Nanyang Avenue.

Huang, G.B., Zhu, Q.Y. and Siew, C.K., 2006. Extreme learning machine: theory and applications. Neurocomputing, 70(1), pp.489-501.

Khair, Aulia. 2011. Peramalan Listrik Jangka Pendek Menggunakan Kombinasi Metode *Autoregressive Integrated Moving Average* (ARIMA) Dengan Regresi Linear Antara Suhu dan Daya Listrik. Fakultas Teknik Universitas Indonesia. Depok.

Kusuma, JI, Mahmudy, WF, dan Indriati. 2015. Optimasi Komposisi Pakan Sapi Potong Menggunakan Algoritme Genetika. DORO: *Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya*, vol. 5, no. 15.

Mahmudy, Wayan Firdaus. 2013. Algoritme Evolusi. Program Teknologi Informasil dan Ilmu Komputer (PTIIK) Universitas Brawijaya. Malang.

Mahmudy, Wayan Firdaus. 2015. Dasar-Dasar Algoritme Evolusi. Program Teknologi Informasi dan Ilmu Komputer (PTIIK) Universitas Brawijaya. Malang.

Megantara, Agung Nurjaya. 2017. Optimasi *Fuzzy Inference System* Mamdani Menggunakan Algoritme Genetika Untuk Menentukan Lama Waktu Siram Pada Tanaman *Strawberry*. Fakultas Ilmu Komputer Universitas Brawijaya. Malang.

Muin, Harli. 2014. Mencari Solusi Strategis Untuk Memenuhi Kebutuhan Energi Listrik. Bau Bau, Sulawesi Tenggara, 16 Desember 2014.

Perusahaan Listrik Negara. 2012. Perencanaan Operasi Sistem. Pusat Pendidikan dan Pelatihan.

Perdana, Januar Adi, Adi Soeprijanto, dan Rony Seto Wibowo. 2012. Peramalan Beban Listrik Jangka Pendek Menggunakan *Optimally Pruned Extreme Learning Machine* (OPELM) pada Sistem Kelistrikan Jawa Timur. Jurnal Teknik ITS Vol. 1, No. 1, September 2012.

Sholekhah, Mar'atus. 2017. *Hybrid* Jaringan Syaraf Tiruan Metode *Extreme Learning Machine* (ELM) dengan *Genetic Algorithm* (GA) untuk Meramalkan Nilai Tukar Rupiah terhadap Dolar Amerika. Fakultas Sains dan Teknologi Universitas Airlangga. Surabaya.

Smith, JE & Eiben, AE. 2003. *Introduction to Evolutionary Computing*. Springer. London.

Sun, Zhan-Li, Tsan-Ming Choi, Kin-Fan Au, dan Yong Yu. 2008. *Sales Forecasting Using Extreme Learning Machine With Applications in Fashion Retailing. Decision Support Systems* 46 (2008) 411-419.

Wang, Xinyou, Chenhua Wang, dan Qing Li. 2016. *Short-term Wind Power Prediction Using* GA-ELM. *The Open Electrical & Electronic Engineering Journal* 2017, 11, 48-56.