PERAMALAN BEBAN LISTRIK JANGKA PENDEK MENGGUNAKAN JARINGAN SYARAF TIRUAN ALGORITMA *RESILIENT PROPAGATION*

Skripsi
Disusun untuk melengkapi syarat-syarat
guna memperoleh gelar Sarjana Sains

SYIFA AULIA
3125111209

PROGRAM STUDI MATEMATIKA
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI JAKARTA
2015

LEMBAR PERSETUJUAN HASIL SIDANG SKRIPSI

PERAMALAN BEBAN LISTRIK JANGKA PENDEK MENGGUNAKAN JARINGAN SYARAF TIRUAN ALGORITMA *RESILIENT PROPAGATION*

Nama : Syifa Aulia

No. Registrasi : 3125111209

	Nama	Tanda Tangan	Tanggal
Penanggung Jawab			
Dekan	: Prof. Dr. Suyono, M.Si.	.....	.....
	NIP. 19671218 199303 1 005		
Wakil Penanggung Jawab			
Pembantu Dekan I	: Dr. Muktiningsih, M.Si.	.....	.....
	NIP. 19640511 198903 2 001		
Ketua	: Drs. Mulyono, M.Kom.	.....	.....
	NIP. 19660517 199403 1 003		
Sekretaris	: Ria Arafiah, M.Si.	.....	.....
	NIP. 19751121 200501 2 004		
Penguji	: Ir. Fariani Hermin, M.T.	.....	.....
	NIP. 19600211 198703 2 001		
Pembimbing I	: Ratna Widyati, S.Si, M.Kom.	.....	.....
	NIP. 19750925 200212 2 002		
Pembimbing II	: Med Irzal, M.Kom.	.....	.....
	NIP. 19770615 200312 1 001		

Dinyatakan lulus ujian skripsi tanggal: 8 Desember 2015

ABSTRACT

SYIFA AULIA, 3125111209. Short Term Electrical Load Forecasting Using Artificial Neural Network Resilient Propagation Algorithm. Thesis. Faculty of Mathematics and Natural Science Jakarta State University. 2015.

Electricity is a very crucial needed to people nowadays, even more we could say that electricity is being primary needed for people. The demand of electric power from consumer is not fixed by time to time. While the electrical load can not be stored on a large scale, but it should be available when people needed it. Many forecasting methods were developed to optimize the result. Artificial Neural Network (ANN) is one of the methods which are developed to get result of forecast near to the actual data. This thesis implemented artificial neural network Resilient Propagation to predict the consumption of electrical load. Used data for forecasting is daily data from July 2015 to August 2015. Prediction result of the smallest MSE 0,00061 with the architecture 4-10-6-1 that is 4 input neuron, 10 neuron in first hidden layer, 6 neuron in second hidden layer, and 1 output neuron.

Keywords *: artificial neural network, resilient propagation, forecasting.*

ABSTRAK

SYIFA AULIA, 3125111209. Peramalan Beban Listrik Jangka Pendek Menggunakan Jaringan Syaraf Tiruan Algoritma *Resilient Propagation*. Skripsi. Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta. 2015.

Listrik merupakan kebutuhan yang sangat krusial bagi masyarakat pada saat ini, bahkan dapat dikatakan bahwa listrik sudah menjadi kebutuhan primer masyarakat. Kebutuhan konsumen terhadap daya listrik tidak tetap dari waktu ke waktu. Sedangkan listrik tidak dapat disimpan dalam skala besar, tetapi harus tersedia pada saat dibutuhkan. Banyak metode peramalan yang telah dikembangkan untuk mendapatkan hasil peramalan yang optimal. Jaringan syaraf tiruan (JST) adalah salah satu metode yang saat ini dikembangkan untuk mendapatkan hasil peramalan yang mendekati dengan data sebenarnya. Dalam skripsi ini dipaparkan implementasi jaringan syaraf tiruan *Resilient Propagation* (RPROP) untuk memprediksi pemakaian beban listrik. Data yang digunakan dalam peramalan adalah data harian periode Juli 2015 sampai Agustus 2015. Dari hasil peramalan didapatkan MSE terkecil yaitu 0,00061 dengan arsitektur jaringan 4-10-6-1 yaitu 4 *neuron input*, 10 *neuron* di *hidden layer* pertama, 6 *neuron* di *hidden layer* kedua, dan 1 *neuron output*.

Kata kunci : jaringan syaraf tiruan, *resilient propagation*, peramalan.

PERSEMBAHAN

"*Fa inna ma'al 'usri yusraa, inna ma'al 'usri yusraa*" (94:5-6)

"*Teruslah bergerak, karena masa depan hanya milik mereka yang berani berjuang*"
-Syifa Aulia, 2015

Skripsi ini kupersembahkan untuk (Alm) Ali Achmad, Siti Maryanah, dan Awin Maryadi.

"Terima kasih untuk do'a yang selalu terhampar di bumi dan langit-Nya untukku".

KATA PENGANTAR

Segala puji bagi Allah Subhanallahuata'ala atas nikmat sehat yang diberikan sehingga penulis dapat menyelesaikan skripsi yang berjudul "Peramalan Beban Listrik Jangka Pendek Menggunakan Jaringan Syaraf Tiruan Algoritma *Resilient Propagation*" yang merupakan salah satu syarat dalam memperoleh gelar Sarjana Jurusan Matematika Universitas Negeri Jakarta.

Skripsi ini berhasil diselesaikan tidak terlepas dari adanya bantuan berbagai pihak. Oleh karena itu, dalam kesempatan ini penulis ingin menyampaikan terima kasih terutama kepada:

1. Ibu Ratna Widyati, S.Si, M.Kom selaku Dosen Pembimbing I dan Bapak Med Irzal, M.Kom selaku Dosen Pembimbing II, yang telah memberikan ilmu dan waktunya dalam memberikan bimbingan, saran, serta arahan sehingga skripsi ini menjadi lebih baik. Semoga kebahagiaan selalu mengelilingi Ibu dan Bapak.

2. Ibu Dr. Lukita Ambarwati, S.Pd, M.Si, selaku Ketua Prodi Matematika FMIPA UNJ yang telah banyak membantu penulis.

3. Bapak Drs. Mulyono, M.Kom., selaku Pembimbing Akademik atas segala bimbingan dan kerja sama Bapak selama perkuliahan, dan seluruh Bapak dan Ibu dosen atas pengajarannya yang telah diberikan, serta karyawan dan karyawati FMIPA UNJ yang telah memberikan informasi yang penulis butuhkan dalam menyelesaikan skripsi.

4. (Alm) Ali Achmad dan Mamah tercinta, Siti Maryanah, yang selalu mengangkat kedua tangannya di sepertiga malam untuk terus mendoakan kesuksesan bagi anak-anaknya. Skripsi ini hanya sebagian kecil dari keberhasilan kalian dalam mendidik seorang anak.

5. Kelima kakak penulis, Awin Maryadi, Fachrul Maryadi, Fikri Maryadi, Ria Amalia, dan Riza Amalia serta kakak ipar penulis, Bubu, Teh Irma, Bang Nur yang selalu mendukung dan mendoakan. Dan juga cimit kesayangan Aufa, Rafa, Hafidz, Akila, Azka yang selalu mampu memberikan tawa dan menghilangkan lelah dikala penulis mengerjakan skripsi.

6. Sahabat SMA penulis Azza, Yuanita, Herli, Aul, Cemara, Vivi, Aghnia, Yolanda, dan Fary yang selalu ada selama hampir delapan tahun terakhir. Sahabat yang tanpa penulis minta selalu mengulurkan tangannya memberikan bantuan.

7. Anisa Idam, selaku sahabat penulis. Terima kasih untuk ada selama empat tahun terakhir, untuk canda tawa serta suka duka yang terlewati bersama.

8. Sahabat yang selalu menghibur: Vira, Ambar, Cika, Indahdwi, Desya, Iyus dan partner sidang skripsi Hamas Fahmi dan juga teman-teman di Matematika 2011. Terima kasih telah mengisi empat tahun ini menjadi kenangan yang tak terlupa dengan persahabatan yang luar biasa, dan perhatian serta kebersamaan yang semoga akan abadi. Semoga ada banyak kesempatan baik untuk kita berbagi cerita.

9. TASEM. Pepu, Nida, Monic, Idam, Firdha, dan Nancy yang selalu menjadi teman yang mampu menghibur penulis, terima kasih untuk ada dikala penulis membutuhkan bantuan dan masukkan.

10. Kakak dan adik tingkat khususnya Mega dan Fara yang selalu memberikan dukungan kepada penulis.

11. Teman-teman serta berbagai pihak yang telah membantu dalam penulisan skripsi ini yang tidak dapat penulis sebutkan satu per satu, semoga Allah membalas semua kebaikan kalian.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Masukan dan kritikan akan sangat berarti. Semoga skripsi ini dapat bermanfaat bagi pembaca sekalian.

Jakarta, Desember 2015

Syifa Aulia

DAFTAR ISI

DAFTAR SIMBOL

			halaman
X_i	:	unit input ke-i; $i = 1, 2, ..., p$	16
W_{ij}	:	bobot unit input ke-i pada unit hidden ke-j; $j = 1, 2, ..., n$	16
W_{0j}	:	bobot bias unit input pada unit hidden ke-j	16
Z_in_j	:	unit hidden ke-j dari input ke-i	16
Z_j	:	unit hidden ke-j	16
V_{jk}	:	bobot unit hidden ke-j pada unit output ke-k; $k = 1, 2, ..., m$	17
V_{0k}	:	bobot bias unit hidden pada unit output ke-k	17
Y_in_k	:	unit output ke-k dari unit hidden ke-j	17
Y_k	:	unit output ke-k	17
δ_k	:	koreksi kesalahan untuk pengaturan bobot V_{jk}	17
t_k	:	output yang diinginkan	17
$\frac{\partial E}{\partial V_{jk}}(t)$	:	turunan parsial dari error terhadap bobot V_{jk}	17
$\frac{\partial E}{\partial V_{jk}}(t)$	:	turunan parsial dari error terhadap bobot bias V_{0k}	18
$\Delta_{jk}(t)$	:	koreksi bobot ke-jk	18
$\Delta_{jk}(t-1)$	:	koreksi bobot ke-jk pada waktu sebelumnya	18
$\frac{\partial E}{\partial V_{jk}}(t-1)$	:	turunan parsial dari error terhadap bobot V_{jk} pada waktu sebelumnya	18
η^+	:	faktor penaik	18
η^-	:	faktor penurun	18
Δ_{maks}	:	perubahan bobot maksimum	18
Δ_{min}	:	perubahan bobot minimum	18

$\Delta V_{jk}(t)$	:	perubahan bobot pada unit V_{jk}	18
δ_in_j	:	selisih input pada unit ke-j	18
$\delta 1_j$	:	koreksi kesalahan untuk pengaturan bobot unit ke-W_{ij}	19
$\frac{\partial E}{\partial W_{ij}}(t)$	:	turunan parsial dari error terhadap bobot W_{ij}	19
$\frac{\partial E}{\partial W_{0j}}(t)$	:	turunan parsial dari error terhadap bobot bias W_{0j}	19
$\frac{\partial E}{\partial W_{ij}}(t-1)$	:	turunan parsial dari error terhadap bobot	19
		W_{ij} pada waktu sebelumnya	
$\Delta_{ij}(t)$	:	koreksi bobot ke-ij	19
$\Delta_{ij}(t-1)$	:	koreksi bobot ke-ij pada waktu sebelumnya	19
$\Delta W_{ij}(t)$	:	perubahan bobot pada unit W_{ij}	19
$\Delta V_{0k}(t)$	:	perubahan bobot bias pada unit ke-k	20
$\Delta W_{0j}(t)$	:	perubahan bobot bias pada unit ke-j	20
$V_{jk}(baru)$	:	bobot baru unit hidden ke-j pada	20
		unit output ke-k; $k = 1, 2, ..., m$	
$W_{ij}(baru)$	:	bobot baru unit input ke-i pada	20
		unit hidden ke-j; $j = 1, 2, ..., n$	

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Listrik merupakan kebutuhan yang sangat krusial bagi masyarakat pada saat ini. Berbagai aktivitas tidak dapat lepas dari peranan listrik, bahkan dapat dikatakan bahwa saat ini listrik sudah menjadi kebutuhan primer masyarakat. Listrik menunjang berbagai sektor kehidupan, seperti industri, rumah tangga, sosial, dan pemerintahan. Peningkatan kebutuhan listrik menyebabkan pihak penyedia listrik, yakni PT. PLN Persero harus menyalurkan kebutuhan listrik agar tidak mengganggu rutinitas masyarakat.

Kebutuhan konsumen terhadap daya listrik tidak tetap dari waktu ke waktu. Sedangkan listrik tidak dapat disimpan dalam skala besar, tetapi harus tersedia pada saat dibutuhkan. Akibatnya timbul masalah dalam menghadapi pemenuhan daya listrik kepada konsumen. Karena listrik tidak dapat disimpan dalam skala besar, maka perlu adanya suatu sistem untuk meramalkan kebutuhan listrik setiap harinya. Prediksi pemenuhan kebutuhan daya listrik merupakan salah satu bidang yang banyak diteliti karena listrik merupakan jenis energi utama, sehingga diperlukan perencanaan yang baik untuk mengetahui permintaan listrik dimasa depan.

Hal ini menjadi sangat penting, khususnya untuk PT.PLN Persero sebagai penyedia daya listrik. PT.PLN Persero harus menyuplai daya listrik sesuai de-

ngan permintaan konsumen. Dari pihak penyedia daya listrik, total daya yang dihasilkan oleh pembangkit dan didistribusikan haruslah sesuai dengan permintaan. Jika beban yang dikirim dari pembangkit terlalu besar, maka frekuensi sistem akan naik dan berpotensi merusak elemen pembangkit maupun distribusi. Akan tetapi, jika daya yang disuplai kurang, maka frekuensi sistem akan menurun dan menyebabkan pemadaman di beberapa titik. Untuk mencegah hal tersebut, diperlukan suatu sistem peramalan beban listrik yang dapat menyesuaikan daya yang dibangkitkan dan didistribusikan dengan permintaan konsumen dari waktu ke waktu.

Beban listrik sebaiknya diprediksi dalam jangka pendek, karena untuk jangka panjang lebih banyak dipengaruhi oleh masalah makro ekonomi dan perlu adanya arahan dari pemerintah. Peramalan beban jangka pendek merupakan peramalan dalam jangka waktu harian setiap jam. Biasa digunakan untuk membandingkan beban listrik hasil peramalan dengan beban aktual.

Sejumlah variasi statistik dan *artificial intellegence* telah dikembangkan sebagai metode peramalan jangka pendek, salah satunya adalah *Artificial Neural Network* (Jaringan Syaraf Tiruan). Jaringan syaraf tiruan merupakan cabang kecerdasan buatan yang meniru cara kerja otak makhluk hidup yaitu sel syaraf (*neuron*). Jaringan syaraf tiruan telah banyak digunakan sebagai studi pembelajaran peramalan beban listrik (Aulia, 2011).

Salah satu model peramalan yang dapat digunakan untuk memprediksi besarnya konsumsi listrik adalah dengan jaringan syaraf tiruan algoritma *Resilient Propagation*. Algoritma *Resilient Propagation* dipilih karena merupakan algoritma terbaik dalam hal kecepatan konvergensi (Lobo dan Santosa, 2014). Hal ini mungkin terjadi karena besarnya turunan parsial tidak mempengaruhi pembaruan bobot dan hanya bergantung pada tanda-tanda dari turunan parsial (Saputro,

2006). Oleh karena itu, Algoritma *Resilient Propagation* sangat memungkinkan untuk konvergensi yang lebih cepat dibandingkan dengan *backpropagation* tradisional.

1.2 Perumusan Masalah

Perumusan masalah yang akan dikaji adalah bagaimana arsitektur jaringan syaraf tiruan peramalan beban listrik jangka pendek dengan Algoritma *Resilient Propagation*?

1.3 Pembatasan Masalah

Pembatasan masalah dalam penulisan ini adalah:

1. *Layer* yang digunakan berupa 1 *input layer*, 2 *hidden layer* dan 1 *output layer*.

2. Data yang digunakan dalam penulisan ini didapatkan dari PT.PLN Persero Area Pengatur Distribusi Beban Penyulang Cawang Baru pada tanggal 17 Juni 2015 - 17 Agustus 2015.

1.4 Tujuan Penulisan

Tujuan yang ingin dicapai dalam skripsi ini adalah memperoleh arsitektur jaringan syaraf tiruan untuk peramalan beban listrik jangka pendek dengan nilai eror terkecil menggunakan Algoritma *Resilient Propagation*.

1.5 Manfaat Penulisan

Manfaat yang diharapkan dari skripsi ini adalah

1. Didapatkan arsitektur jaringan syaraf tiruan untuk peramalan beban listrik jangka pendek dengan nilai eror terkecil menggunakan Algoritma *Resilient Propagation*.

2. Diharapkan dapat memberikan informasi mengenai tingkat akurasi *Resilient Propagation* dalam bidang peramalan.

1.6 Metode Penelitian

Skripsi ini merupakan kajian pustaka dalam bidang komputasi dan *time series* yang didasarkan pada buku-buku dan jurnal-jurnal tentang teori Jaringan Syaraf Tiruan (JST), khususnya Algoritma *Resilient Propagation*.

BAB II

LANDASAN TEORI

Pada bab ini, akan dijelaskan mengenai peramalan, jaringan syaraf tiruan, dan algoritma *resilient propagation.*

2.1 Peramalan

Peramalan adalah ilmu yang memprediksi peristiwa-peristiwa yang akan terjadi dengan menggunakan data historis dan memproyeksikannya ke waktu yang akan datang dengan beberapa model matematis. Untuk melakukan peramalan digunakan metode tertentu yang bergantung dari informasi yang tersedia. Dalam praktiknya, ada berbagai metode peramalan, yaitu:

1. Metode Kuantitatif

   Metode kuantitatif adalah metode peramalan dengan menggunakan data yang dimiliki. Metode ini digunakan jika memenuhi kondisi sebagai berikut:

   - Tersedianya informasi atau data tentang masa lalu.
   - Informasi atau data masa lalu tersebut dapat dikuantitatifkan dalam bentuk data.
   - Informasi atau data masa lalu yang diperoleh dapat diasumsikan sehingga polanya terus berlanjut sampai ke masa yang akan datang.

Metode kuantitatif terbagi menjadi dua bagian, pertama metode deret Berkala (*Time Series*). Peramalan dengan menggunakan *Time Series* bergantung kepada nilai seluruh variabel dari masalalu atau kesalahan yang dilakukan sebelumnya. Tujuan dari metode ini adalah meneliti pola data yang dipakai untuk meramal dan melakukan perkiraan ke masa depan. Kedua, metode kausal. Metode ini adalah metode yang mengasumsikan bahwa faktor yang diramalkan menunjukkan hubungan sebab akibat dengan satu atau lebih variabel bebas.

2. Metode Kualitatif

   Metode kualitatif tidak memerlukan data historis seperti metode kuantitatif. Input yang dibutuhkan bergantung kepada pertimbangan dan pengetahuan yang telah didapat.

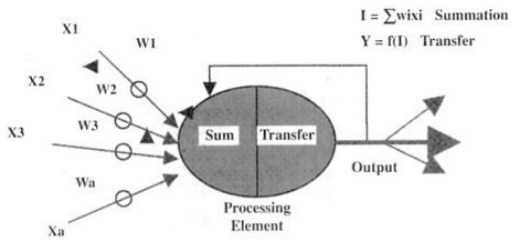
2.2 Jaringan Syaraf Tiruan

Otak manusia terdiri dari 10 milyar *neuron* yang saling berhubungan satu sama lain. Hubungan *neuron* ini disebut dengan *synapses*. *Neuron* secara garis besar terbagi menjadi tiga bagian, yaitu *dendrites, cell body*, dan *axon*. Proses perjalanan sinyal adalah sebagai berikut. Pertama, sinyal masuk melalui *dendrites* menuju *cell body*. Kemudian di dalam *cell body* sinyal diproses dan akan diteruskan menuju ke *axon* dan akhirnya menuju ke *neuron* lainnya lewat *synapse* (yaitu celah sempit antara *axon* dari suatu *neuron* dan *dendrites* dari neuorn lainnya). Model *neuron* inilah yang selanjutnya menjadi dasar dari jaringan syaraf tiruan.

Pada jaringan syaraf tiruan, neuron-neuron akan dikumpulkan dalam lapisan yang disebut dengan *layer*. *Neuron* dalam satu lapisan akan dihubungkan

dengan neuron pada lapisan lainnya. Terkadang muncul juga *layer* tersembunyi (*hidden layer*) untuk menambah keakuratan.

Sebuah *neuron* pada jaringan syaraf tiruan dianalogikan sebagai *neuron* biologis. *Dendrites* sebagai tempat masuk sinyal yang berupa impuls elektrik yang ditransmisikan melalui sebuah celah sinapsis dengan bantuan proses kimia. Proses kimia inilah yang memodifikasi sinyal masuk dimana proses kimia erat kaitanyna dengan fungsi aktivasi pada jaringan syaraf tiruan. Komponen kedua, *cell body* berfungsi sebagai penjumlahan dari sinyal masuk. Dimana aktivitasi di *cell body* ini didapat dari proses aktivasi *dendrites* melalui jalur yang dikenal dengan axon. Jalur ini pada jaringan syaraf tiruan disimbolkan dengan *weight* (bobot), dimana *weight* inilah yang membedakan nilai koneksi dari setiap jalur yang ada.



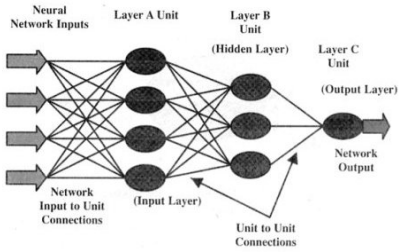
Gambar 2.1: Simulasi jaringan syaraf tiruan

2.2.1 Arsitektur Jaringan Syaraf Tiruan

Arsitektur jaringan syaraf tiruan adalah suatu pola di mana neuron-neuron pada jaringan syaraf tiruan disusun berhubungan erat dengan algoritma belajar yang digunakan untuk melatih jaringan. Jaringan syaraf tiruan dikelompokkan menjadi dua bagian dilihat dari arah datangnya sinyal, yaitu:

- *Feedforward Network* (Jaringan Umpan Maju)

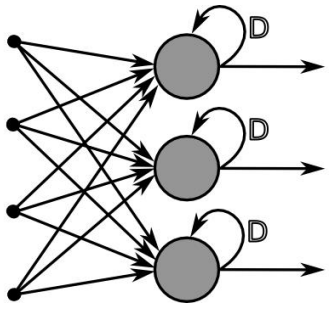
  Pada *feedforward network*, sinyal bergerak dari unit *input* ke unit *output* dengan arah maju.



Gambar 2.2: Jaringan umpan maju (*feedforward network*)

- *Recurrent Network* (Jaringan Umpan Balik)

  Recurrent network adalah jaringan yang memiliki minimal satu *feedback loop*. Sebagai contoh, suatu *recurrent network* bisa terdiri dari satu lapisan *neuron* tunggal dengan masing-masing neuron memberikan kembali *output*-nya sebagai *input* pada semua *neuron* yang lain.

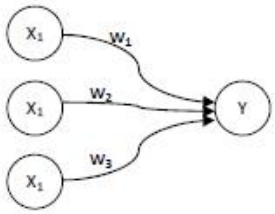


Gambar 2.3: *Recurrent Network*

Dan jaringan syaraf tiruan dapat diklasifikasikan menjadi dua kelas yang berbeda jika dilihat dari jumlah layer, yaitu:

- *Single Layer*

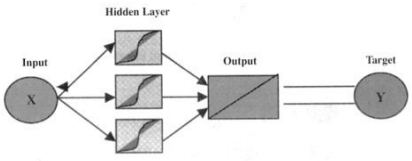
  Single layer merupakan suatu jaringan syaraf tiruan yang dibentuk dengan lapisan-lapisan (*layer*). Pada bentuk paling sederhana hanya terdapat *input layer* dan *output layer* seperti gambar di bawah ini:



Gambar 2.4: *Single layer*

- *Multi Layer*

  Multi layer merupakan suatu jaringan syaraf tiruan dengan satu atau lebih lapisan (*layer*) yang tersembunyi (*hidden layer*). Dengan adanya hidden layer. sistem akan semakin fleksibel dalam memecahkan berbagai permasalahan, yang mungkin tidak dapat diselesaikan oleh *single layer*

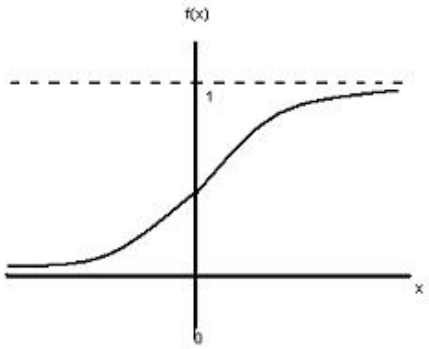


Gambar 2.5: *Multi layer*

2.2.2 Fungsi Aktivasi

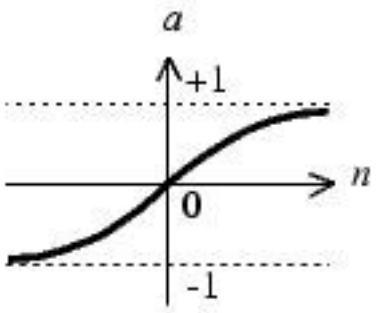
Fungsi aktivasi merupakan suatu fungsi transfer yang terdapat pada sebuah neuron yang akan digunakan untuk menghitung nilai *output* dari suatu *neuron*. Secara umum, fungsi aktivasi terdiri dari beberapa fungsi antara lain fungsi threshold, linear, Gaussian, sigmoid biner (logsig) dan lain-lain. Pada penulisan ini digunakan fungsi aktivasi tipe sigmoid biner, sigmoid bipolar, dan purelin dengan persamaan:

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



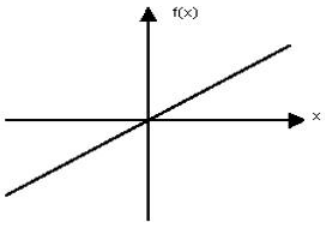
Gambar 2.6: Fungsi Aktivasi Sigmoid Biner

- $f(x) = \frac{e^{\alpha x} - e^{-\alpha x}}{e^{\alpha x} + e^{-\alpha x}}$



Gambar 2.7: Fungsi Aktivasi Sigmoid Bipolar

- $f(x) = x$



Gambar 2.8: Fungsi Aktivasi Purelin

2.2.3 Bias

Bobot jaringan syaraf input ke unit Y_j (dengan bias pada unit j) adalah sebagai berikut:

$$\begin{aligned} y\_in_j &= XW_j \\ &= \sum_{i=1}^{j} X_i W_j \end{aligned}$$

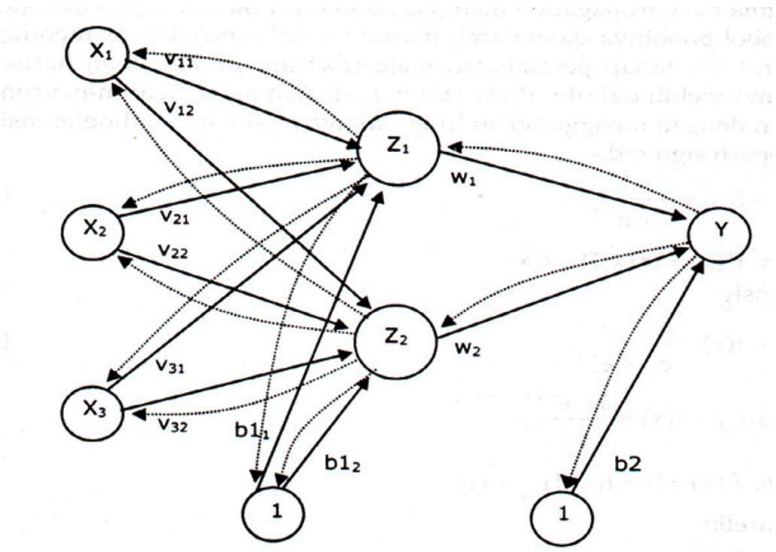
Bias dapat dinotasikan dengan bobot yaitu: $W_{0j} = b_j$. Jaringan input ke unit Y_j didefinisikan sebagai:

$$\begin{aligned} y\_in_j &= XW_j \\ &= \sum_{i=1}^{j} X_i W_j \\ &= W_{0j} + \sum_{i=1}^{j} X_i W_{ij} \\ &= b_j + \sum_{i=1}^{j} X_i W_{ij} \end{aligned}$$

2.3 Algoritma *Backpropagation*

Backpropagation merupakan salah satu dari beberapa metode yang digunakan dalam jaringan syaraf tiruan dan yang paling sering digunakan dalam berbagai bidang aplikasi, seperti pengenalan pola, peramalan dan optimalisasi. Hal ini memungkinkan karena metode ini menggunakan *supervised learning. Backpropagation* merupakan *multilayer neural network* yang terdiri dari *input layer, hidden layer,* dan *output layer*, dan setiap *layer* terdiri dari satu atau lebih atificial *neuron. Backpropagation* bertujuan untuk memperkecil tinggkat *error* dengan menyesuaikan bobot berdasarkan perbedaan *output* dan target yang diinginkan.

Gambar di bawah ini merupakan arsitektur backpropagation dengan n buah masukan (ditambah sebuah bias), sebuah hidden layer yang terdiri dari p unit (ditambah dengan sebuah bias), serta m buah unit output.



Gambar 2.9: Jaringan *backpropagation*

Seperti yang diperlihatkan pada gambar di atas bahwa jaringan Backpropagation terdiri dari 3 sel *neuron* pada *input layer* dan x_1, x_2 dan x_3 sedangkan pada lapisan tersembunyi terdapat 2 sel *neuron* yaitu z_1 dan z_2 serta 1 sel *neuron* pada *output layer* yaitu y. Nilai bias b_1 yang diberikan pada lapisan tersembunyi bertujuan untuk mengolah data *input* ditambah bobot v_{ij} yang masuk ke dalam sel-sel pada *hidden layer* dengan bantuan fungsi aktifasi. Begitu pula dengan nilai bias b_1 yang diberikan pada lapisan keluaran adalah untuk mengolah data yang berasal dari keluaran sel pada *hidden layer* ditambah bobot w_{ij} yang masuk kedalam *output layer* dengan bantuan fungsi aktivasi.

1. Propagasi maju

   Selama propagasi maju, sinyal masukan dipropagasikan ke *hidden layer* menggunakan fungsi aktivasi yang telah ditentukan hingga menghasilkan keluaran jaringan. Keluaran jaringan dibandingkan dengan target yang

harus dicapai. Selisih antara target dengan keluaran merupakan kesalahan yang terjadi. Jika kesalahan lebih kecil dari batas toleransi, maka iterasi dihentikan. Akan tetapi jika kesalahan lebih besar, maka bobot setiap garis dalam jaringan akan dimodifikasi untuk mengurangi kesalahan yang terjadi.

2. Propagasi mundur

   Kesalahan yang terjadi di propagasi mundur mulai dari garis yang berhubungan langsung dengan neuron-neuron dilayar keluaran.

3. Perubahan bobot

   Pada fase ini bobot semua garis dimodifikasi secara bersamaan. Ketiga fase tersebut diulang- ulang terus hingga kondisi penghentian dipenuhi. Kondisi penghentian yang sering dipakai adalah jumlah maksimal iterasi *(epoch)* atau minimal kesalahan *(error)*.

2.4 *Resilient Propagation*

Resilient propagation atau biasa disingkat RPROP adalah satu algoritma yang digunakan untuk mempercepat *learning rate* pada pelatihan jaringan syaraf tiruan *backpropagation*. *Resilient propagation* (RPROP) dikembangkan oleh Martin Riedmiller dan Heinrich Braun pada tahun 1992. Metode ini adalah salah satu modifikasi dari proses standar *backpropagation* yang digunakan untuk mempercepat laju pembelajaran pada pelatihan jaringan syaraf tiruan *backpropagation*.

RPROP merupakan algoritma yang digunakan untuk mempercepat pembelajaran pada pelatihan jaringan syaraf tiruan backpropagation. Algoritma *resilient propagation* menggunakan tanda dari nilai turunan untuk menentukan arah perbaikan bobot. Besarnya perubahan bobot ditentukan oleh parameter faktor

naik (η^+) dan faktor turun (η^-). Pada awal iterasi, besarnya perubahan bobot diinisialisasikan dengan parameter Δ_0. Besarnya perubahan bobot tidak boleh melebihi parameter Δ_{maks} dan tidak boleh dibawah Δ_{min}. Apabila perubahan bobot melebihi batas maksimum, maka perubahan bobot ditentukan sama dengan maksimum perubahan bobot. Proses propagasi maju pada algoritma JST *resilient propagation* sama dengan algoritma *backpropagation*, yang berbeda pada proses propagasi mundur.

2.4.1 Algoritma Pelatihan *Resilient Propagation*

Algoritma *Resilient Propagation* sebenarnya hampir sama dengan *backpropagation* pada propagasi maju *(forward)*. Hanya saja untuk propagasi mundur *(backward)* ada beberapa syarat yang harus dipenuhi. Sebelum dilakukan pelatihan pada data, maka terlebih dahulu harus dinormalisasi. Normalisasi data adalah proses pengubahan data asli menjadi data dengan *range* antara 0,1 sampai 0,9. Hal ini disebabkan karena fungsi aktivasi yang digunakan adalah fungsi sigmoid yang nilai fungsinya tidak pernah mencapai 0 ataupun 1 (Siang JJ, 2009). Berikut adalah rumus untuk normalisasi data (Siang JJ, 2009).

$$x' = \frac{0,8(x-a)}{b-a} + 0,1 \tag{2.1}$$

Dengan:

x'=data yang sudah dinormalisasi

x=data yang belum dinormalisasi

a=nilai minimum dari seluruh data

b=nilai maksimum dari seluruh data

Setelah data dinormalisasi lakukan langkah berikut ini untuk menjalankan algo-

ritma *resilient propagation* (Donny, 2006):

Langkah 0

Inisialisasi bobot random dengan bilangan acak antara $[-1, 1]$

Langkah 1

Jika kondisi berhenti masih belum terpenuhi, maka lakukan langkah 2-9

Langkah 2

Untuk setiap pasang pelatihan, lakukan langkah 3-8

Langkah 3

Setiap input $(X_i, i = 1, ..., n)$ menerima sinyal input X_i dan meneruskan sinyal ini ke semua neuron pada *layer* di atasnya *(hidden neuron).*

Langkah 4

Setiap hidden neuron $(Z_j, j = 1, ..., p)$ menjumlahkan bobot dari sinyal-sinyal inputnya.

$$Z\_in_j = W_{0j} + \sum_{i=1}^{n} X_i W_{ij} \tag{2.2}$$

Kemudian gunakan fungsi aktivasi untuk menghitung nilai sinyal outputnya:

$$Z_j = f(Z\_in_j) \tag{2.3}$$

Fungsi aktivasi yang digunakan pada penulisan ini adalah fungsi aktivasi sigmoid biner:

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

Dan kirimkan sinyal ini ke semua neuron yang berada pada *layer* di atasnya *(output neuron)*.

Langkah 5

Setiap unit output $(Y_k, k = 1, ..., m)$ menghitung total sinyal masukan terbobot

$$Y\_in_k = V_{0k} + \sum_{j=1}^{p} z_j V_{jk} \tag{2.4}$$

Lalu menghitung sinyal keluaran dengan fungsi aktivasi

$$Y_k = f(Y\_in_k) \tag{2.5}$$

Langkah 6

Setiap unit output $(Y_k, k = 1, ..., m)$ menerima sebuah pola target yang sesuai dengan pola masukan pelatihannya. Unit tersebut menghitung informasi kesalahan.

$$\delta_k = (t_k - y_k) f'(Y\_in_k) \tag{2.6}$$

Untuk hidden neuron

$$\frac{\partial E}{\partial V_{jk}}(t) = \delta_k \times Z_j \tag{2.7}$$

Untuk bias di hidden neuron

$$\frac{\partial E}{\partial V_{0k}}(t) = \delta_k \tag{2.8}$$

Kemudian koreksi bobot,

$$\Delta_{jk}(t) = \begin{cases} \eta^+ \times \Delta_{jk}(t-1), \frac{\partial E}{\partial V_{jk}}(t-1) \times \frac{\partial E}{\partial V_{jk}}(t) > 0 \\ \eta^- \times \Delta_{jk}(t-1), \frac{\partial E}{\partial V_{jk}}(t-1) \times \frac{\partial E}{\partial V_{jk}}(t) < 0 \\ \Delta_{jk}(t-1), else \end{cases} \tag{2.9}$$

Selanjutnya,

$$\Delta_{jk}(t) = \begin{cases} min(\Delta_{jk}(t-1) \times \eta^+, \Delta_{maks}) \\ maks(\Delta_{jk}(t-1) \times \eta^-, \Delta_{min}) \end{cases} \tag{2.10}$$

Perubahan bobotnya adalah,

$$\Delta V_{jk}(t) = \begin{cases} -\Delta_{jk}(t), \frac{\partial E}{\partial V_{jk}}(t) > 0 \\ +\Delta_{jk}(t), \frac{\partial E}{\partial V_{jk}}(t) < 0 \\ \Delta_{jk}(t-1), lainnya \end{cases} \tag{2.11}$$

Hitung koreksi bias dengan cara yang sama

Langkah 7

Setiap unit tersembunyi $(Z_j, j = 1, ..., p)$ menghitung selisih input dengan cara,

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k \times W_{ij} \tag{2.12}$$

Lalu mengalikannya dengan turunan fungsi aktivasi untuk menghitung informasi kesalahan

$$\delta 1_j = \delta\_in_j f'(Z\_in_j) \tag{2.13}$$

Untuk hidden neuron

$$\frac{\partial E}{\partial W_{ij}}(t) = \delta 1_j \times X_i \tag{2.14}$$

Untuk bias di hidden neuron

$$\frac{\partial E}{\partial W_{0j}}(t) = \delta 1_j \tag{2.15}$$

Kemudian koreksi bobot,

$$\Delta_{ij}(t) = \begin{cases} \eta^+ \times \Delta_{ij}(t-1), \frac{\partial E}{\partial W_{ij}}(t-1) \times \frac{\partial E}{\partial W_{ij}}(t) > 0 \\ \eta^- \times \Delta_{ij}(t-1), \frac{\partial E}{\partial W_{ij}}(t-1) \times \frac{\partial E}{\partial W_{ij}}(t) < 0 \\ \Delta_{ij}(t-1), lainnya \end{cases} \tag{2.16}$$

Selanjutnya,

$$\Delta_{ij}(t) = \begin{cases} min(\Delta_{ij}(t-1) \times \eta^+, \Delta_{maks}) \\ maks(\Delta_{ij}(t-1) \times \eta^-, \Delta_{min}) \end{cases} \tag{2.17}$$

Perubahan bobotnya adalah,

$$\Delta W_{ij}(t) = \begin{cases} -\Delta_{ij}(t), \frac{\partial E}{\partial W_{ij}}(t) > 0 \\ +\Delta_{ij}(t), \frac{\partial E}{\partial W_{ij}}(t) < 0 \\ \Delta_{ij}(t-1), lainnya \end{cases} \tag{2.18}$$

Hitung perubahan bobot pada bias dengan cara yang sama.

Langkah 8

Setiap unit output $(Y_k, k = 1, ..., m)$ mengubah bobot dan bias

$$V_{jk}(baru) = V_{jk}(lama) + \Delta V_{jk} \tag{2.19}$$

$$V_{0k}(baru) = V_{0k}(lama) + \Delta V_{0k} \tag{2.20}$$

Setiap unit tersembunyi $(Z_j, j = 1, ..., p)$ mengubah bobot dan bias

$$W_{ij}(baru) = W_{ij}(lama) + \Delta W_{ij} \tag{2.21}$$

$$W_{0j}(baru) = W_{0j}(lama) + \Delta W_{0j} \tag{2.22}$$

Langkah 9

Jika besar nilai $MSE = \frac{1}{n}\sum_{k=1}^{n}(t_k - Y_k)^2$ lebih kecil dari toleransi yang telah ditentukan atau jumlah epoch pelatihan sudah mencapai epoch maksimum, maka selesai, jika tidak maka ulangi kembali.

2.4.2 Algoritma Pengujian *Resilient Propagation*

Pada proses pengujian jaringan syaraf tiruan hanya akan diterapkan tahap propagasi maju. Setelah pelatihan selesai dilakukan, maka bobot-bobot yang terpilih akan digunakan untuk menginisialisasi bobot pada proses pengujian jaringan syaraf tiruan. Adapun tahapannya adalah sebagai berikut:

1. Masukkan nilai input dari data *testing*.

2. Lakukan langkah berikut:

Langkah 1

Setiap hidden neuron $(Z_j, j = 1, ..., p)$ menjumlahkan bobot dari sinyal-sinyal inputnya.

$$Z\_in_j = W_{0j} + \sum_{i=1}^{n} X_i W_{ij}$$

Kemudian gunakan fungsi aktivasi untuk menghitung nilai sinyal outputnya:

$$Z_j = f(Z\_in_j)$$

Fungsi aktivasi yang digunakan pada penulisan ini adalah fungsi aktivasi sigmoid biner:

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

Dan kirimkan sinyal ini ke semua neuron yang berada pada *layer* di atasnya *(output neuron)*.

Langkah 2

Setiap unit output $(Y_k, k = 1, ..., m)$ menghitung total sinyal masukan terbobot

$$Y\_in_k = V_{0k} + \sum_{j=1}^{p} X_i V_{jk}$$

Lalu menghitung sinyal keluaran dengan fungsi aktivasi

$$Y_k = f(Y\_in_k)$$

2.4.3 Contoh Kasus

Data berikut didapatkan dari PT.PLN Persero Area Pengatur Distribusi Beban Penyulang Cawang Baru pada tanggal 17 Juni 2015 - 17 Agustus 2015. Data yang menjadi target adalah data pada tanggal 10 Agustus 2015, dan yang menjadi data pelatihan adalah data pada tanggal 13, 20, 27 Juli 2015 dan 3 Agustus 2015. Sebelum dimasukkan ke model jaringan syaraf tiruan, data terlebih dahulu dinormalisasi dengan persamaan (2.1) agar interval data berkisar antara 0 sampai 1. Berikut ini adalah tabel data yang belum dan sudah dinormalisasi.

Tabel 2.1: Data Peramalan Sebelum dinormalisasi

Data Pelatihan	Data Target
2170	
1740	1820
1980	
1780	

Tabel 2.2: Data Peramalan Sesudah dinormalisasi

Data Pelatihan	Data Target
0,9	
0,1	0,2488
0,5465	
0,1744	

Selanjutnya menentukan nilai parameter untuk algoritma pelatihan *resilient propagation*. Nilai parameter ditentukan mengacu pada jurnal University of Karlsruhe. Nilai parameter tersebut yaitu $\Delta_0 = 0,1$, $\Delta_{maks} = 50$, $\Delta_{min} = 0,0025$, $\eta^+ = 1,2$, dan $\eta^- = 0,5$. Arsitektur *resilient propagation* yang digunakan berupa 4 neuron pada input layer, 2 neuron pada hidden layer, dan 1 *neuron* pada *output*

layer. Selain itu ada beberapa parameter lain yang digunakan, yaitu epoch maksimum=100 dan toleransi error=0,001. Lakukan langkah-langkah pada algoritma pelatihan RPROP.

Langkah 0

Inisialisasi bobot random dengan bilangan acak antara $[-1, 1]$

Tabel 2.3: Inisialisasi Bobot Input ke Hidden Layer

	Z1	Z2
X1	0.1	0.6
X2	0.3	0.1
X3	0.4	0.8
X4	0.5	0.3
b1	0.2	0.4

Tabel 2.4: Inisialisasi Bobot Hidden Layer ke Output

	Y1
Z1	0.1
Z2	0.3
b2	0.4

Selanjutnya lakukan langkah 1-9.

Langkah 1

Jika kondisi berhenti masih belum terpenuhi, maka lakukan langkah 2-9

Langkah 2

Untuk setiap pasang pelatihan, lakukan langkah 3-8

Langkah 3

Setiap input $(X_i, i = 1, ..., n)$ menerima sinyal *input* X_i dan meneruskan sinyal ini

kesemua *neuron* pada *layer* di atasnya *(hidden neuron).*

Langkah 4

Setiap hidden neuron $(Z_j, j = 1, ..., p)$ menjumlahkan bobot dari sinyal-sinyal inputnya dengan menggunakan persamaan (2.23).

$Z\_in_1 = 0,2 + (0,1 \times 0,9 + 0,3 \times 0,1 + 0,4 \times 0,5465 + 0,5 \times 0,1744) = 0,6258$

$Z\_in_2 = 0,4 + (0,6 \times 0,9 + 0,1 \times 0,1 + 0,8 \times 0,5465 + 0,3 \times 0,1744) = 1,4395$

Kemudian gunakan fungsi aktivasi untuk menghitung nilai sinyal outputnya seperti pada persamaan (2.23), dalam kasus ini digunakan fungsi aktivasi sigmoid.

$Z_j = f(Z\_in_j) = \frac{1}{1+e^{-Z\_in_j}}$

$Z_1 = \frac{1}{1+e^{-0,6258}} = 0,6515$

$Z_2 = \frac{1}{1+e^{-1,4395}} = 0,8084$

Dan kirimkan sinyal ini ke semua neuron yang berada pada *layer* di atasnya *(output neuron).*

Langkah 5

Setiap unit *output* $(Y_k, k = 1, ..., m)$ menghitung total sinyal masukan terbobot dengan menggunakan persamaan (2.23):

$Y\_in_1 = 0,4 + (0,3 \times 0,6515 + 0,2 \times 0,8084) = 0,7571$. Lalu menghitung sinyal keluaran dengan fungsi aktivasi menggunakan persamaan (2.23):

$Y = f(Y_{net}) = \frac{1}{1+e^{-0,7571}} = 0,6807$

Langkah 6

Setiap unit output $(Y_k, k = 1, ..., m)$ menerima sebuah pola target yang sesuai

dengan pola masukan pelatihannya dengan menggunakan persamaan (2.6):

$$\begin{aligned}\delta_k &= (t_k - Y_k)(f'(y\_in_k)) \\ &= (t_k - Y_k)Y_k(1 - Y_k) \\ &= (0,2488 - 0,687)(0,6807)(1 - 0,6807) \\ &= -0,0939\end{aligned}$$

Untuk hidden neuron dengan menggunakan persamaan (2.7):

$\frac{\partial E}{\partial V_{11}} = \delta_1 Z_1 = -0,0939 \times 0,6516 = -0,0611$

$\frac{\partial E}{\partial V_{12}} = \delta_1 Z_2 = -0,0939 \times 0,8084 = -0,0759$

Untuk bias di hidden neuron dengan menggunakan persamaan (2.8):

$\frac{\partial E}{\partial V_{01}} = \delta_1 = -0,0939$

Kemudian koreksi bobot menggunakan persamaan (2.9):

$\frac{\partial E}{\partial V_{11}}(t) \times \frac{\partial E}{\partial V_{11}}(t-1) = -0,0611 \times 0 = 0$

$\frac{\partial E}{\partial V_{21}}(t) \times \frac{\partial E}{\partial V_{21}}(t-1) = -0,0759 \times 0 = 0$

$\frac{\partial E}{\partial V_{01}}(t) \times \frac{\partial E}{\partial V_{01}}(t-1) = -0,0939 \times 0 = 0$

Selanjutnya,karena hasil dari $\frac{\partial E}{\partial V_{jk}}(t) \times \frac{\partial E}{\partial V_{jk}}(t-1)$ untuk V_{11}, V_{12}. dan V_{01} bernilai 0, maka, $\Delta_{11}(t) = \Delta_{21}(t) = \Delta_{01} = \Delta_0 = 0,1$

Selanjutnya dengan menggunakan persamaan (2.11), didapatkan

$\frac{\partial E}{\partial V_{11}} < 0$, $\frac{\partial E}{\partial V_{21}} < 0$, dan $\frac{\partial E}{\partial V_{01}} < 0$

Sehingga,

$\Delta V_{11}(t) = +\Delta_{11}(t) = +0,1$

$\Delta V_{21}(t) = +\Delta_{21}(t) = +0,1$

$\Delta V_{01}(t) = +0,1$

Langkah 7

Setiap unit tersembunyi ($Z_j, j = 1, ..., p$) menghitung selisih input dengan menggunakan persamaan (2.12):

$$\begin{aligned}
\delta_in_1 &= \delta_1 \times V_{11} \\
&= -0,0939 \times 0,3 \\
&= -0,2817 \\
\delta_in_2 &= \delta_2 \times V_{21} \\
&= -0,0939 \times 0,2 \\
&= -0,1878
\end{aligned}$$

Lalu mengalikannya dengan turunan fungsi aktivasi untuk menghitung informasi kesalahan dengan menggunakan persamaan (2.13):

$$\begin{aligned}
\delta_1 &= \delta_in_1 \times f'(Z_in_1) \\
&= (-0,02817) \times (1 - 0,6515) \times (0,6515) \\
&= -0,0064 \\
\delta_2 &= \delta_in_2 \times f'(Z_in_2) \\
&= (-0,1878) \times (1 - 0,8084) \times (0,8084) \\
&= -0,0029
\end{aligned}$$

Selanjutnya hitung suku perubahan bobot W_{ij} dengan persamaan (2.14) dan (2.15):

$\frac{\partial E}{\partial W_{11}} = -0,0064 \times 0,9 = -0,0576$

$\frac{\partial E}{\partial W_{21}} = -0,0064 \times 0,1 = -0,00064$

$\frac{\partial E}{\partial W_{31}} = -0,0064 \times 0,5465 = -0,0035$

$\frac{\partial E}{\partial W_{41}} = -0,0064 \times 0,1744 = -0,00112$

$\frac{\partial E}{\partial W_{12}} = -0,0029 \times 0,9 = -0,00261$

$\frac{\partial E}{\partial W_{22}} = -0,0029 \times 0,1 = -0,00029$

$\frac{\partial E}{\partial W_{32}} = -0,0029 \times 0,5465 = -0,00158$

$\frac{\partial E}{\partial W_{42}} = -0,0029 \times 0,1744 = -0,00051$

$\frac{\partial E}{\partial W_{01}} = -0,0064$

$\frac{\partial E}{\partial W_{02}} = -0,0029$

Kemudian koreksi bobot dengan persamaan (2.16):

$\frac{\partial E}{\partial W_{11}}(t) \times \frac{\partial E}{\partial W_{11}}(t-1) = -0,0576 \times 0 = 0$

$\frac{\partial E}{\partial W_{21}}(t) \times \frac{\partial E}{\partial W_{21}}(t-1) = -0,00064 \times 0 = 0$

$\frac{\partial E}{\partial W_{31}}(t) \times \frac{\partial E}{\partial W_{31}}(t-1) = -0,0035 \times 0 = 0$

$\frac{\partial E}{\partial W_{41}}(t) \times \frac{\partial E}{\partial W_{41}}(t-1) = -0,00112 \times 0 = 0$

$\frac{\partial E}{\partial W_{12}}(t) \times \frac{\partial E}{\partial W_{12}}(t-1) = -0,00261 \times 0 = 0$

$\frac{\partial E}{\partial W_{22}}(t) \times \frac{\partial E}{\partial W_{22}}(t-1) = -0,00029 \times 0 = 0$

$\frac{\partial E}{\partial W_{32}}(t) \times \frac{\partial E}{\partial W_{32}}(t-1) = -0,00158 \times 0 = 0$

$\frac{\partial E}{\partial W_{42}}(t) \times \frac{\partial E}{\partial W_{42}}(t-1) = -0,00051 \times 0 = 0$

$\frac{\partial E}{\partial W_{01}}(t) \times \frac{\partial E}{\partial b_{11}}(t-1) = -0,0064 \times 0 = 0$

$\frac{\partial E}{\partial W_{02}}(t) \times \frac{\partial E}{\partial b_{12}}(t-1) = -0,0029 \times 0 = 0$

Selanjutnya, hasil dari $\frac{\partial E}{\partial W_{ij}}(t) \times \frac{\partial E}{\partial W_{ij}}(t-1) = 0$ untuk setiap W_{ij} pada input neuron dan W_{0j} pada bias menghasilkan:

$\Delta_{11}(t) = \Delta_{11}(t-1) = \Delta_0 = 0,1$

$\Delta_{21}(t) = \Delta_{21}(t-1) = \Delta_0 = 0,1$

$\Delta_{31}(t) = \Delta_{31}(t-1) = \Delta_0 = 0,1$

$\Delta_{41}(t) = \Delta_{41}(t-1) = \Delta_0 = 0,1$

$\Delta_{12}(t) = \Delta_{12}(t-1) = \Delta_0 = 0,1$

$\Delta_{22}(t) = \Delta_{22}(t-1) = \Delta_0 = 0,1$

$\Delta_{32}(t) = \Delta_{32}(t-1) = \Delta_0 = 0,1$

$\Delta_{42}(t) = \Delta_{42}(t-1) = \Delta_0 = 0,1$

$\Delta_{01}(t) = \Delta_{11}(t-1) = \Delta_0 = 0,1$

$\Delta_{02}(t) = \Delta_{12}(t-1) = \Delta_0 = 0,1$

Perubahan bobotnya sesuai dengan persamaan (2.18):

$\Delta W_{11}(t) = +0,1$

$\Delta W_{21}(t) = +0,1$

$\Delta W_{31}(t) = +0,1$

$\Delta W_{41}(t) = +0,1$

$\Delta W_{12}(t) = +0,1$

$\Delta W_{22}(t) = +0,1$

$\Delta W_{32}(t) = +0,1$

$\Delta W_{42}(t) = +0,1$

$\Delta W_{01}(t) = +0,1$

$\Delta W_{02}(t) = +0,1$

Langkah 8

Setiap unit output $(Y_k, k = 1, ..., m)$ mengubah bobot dan bias dengan menggunakan persamaan (2.19) dan (2.20).

$\Delta V_{11}(baru) = 0,3 + 0,1 = 0,4$

$\Delta V_{21}(baru) = 0,2 + 0,1 = 0,3$

$\Delta V_{01}(baru) = 0,4 + 0,1 = 0,5$

Setiap unit tersembunyi $(Z_j, j = 1, ..., p)$ mengubah bobot dan bias dengan menggunakan persamaan (2.21) dan (2.22).

$\Delta W_{11}(baru) = 0,1 + 0,1 = 0,2$

$\Delta W_{21}(baru) = 0,3 + 0,1 = 0,4$

$\Delta W_{31}(baru) = 0,4 + 0,1 = 0,5$

$\Delta W_{41}(baru) = 0,5 + 0,1 = 0,6$

$\Delta W_{12}(baru) = 0,6 + 0,1 = 0,7$

$\Delta W_{22}(baru) = 0,1 + 0,1 = 0,2$

$\Delta W_{32}(baru) = 0,8 + 0,1 = 0,9$

$\Delta W_{42}(baru) = 0,3 + 0,1 = 0,4$

$\Delta W_{01}(baru) = 0,2 + 0,1 = 0,3$

$\Delta W_{02}(baru) = 0,4 + 0,1 = 0,5$

Langkah 9

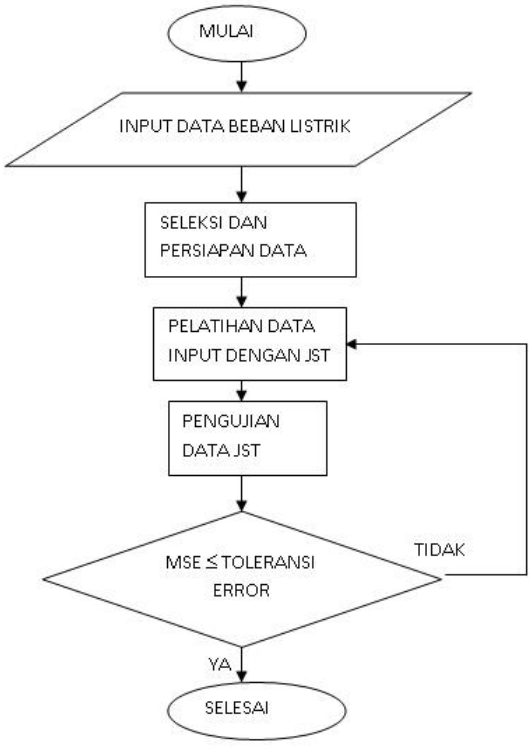
$MSE = \frac{1}{n}\sum_{k=1}^{n}(t_k - Y_k)^2 = (0,2488 - 0,6807)^2 = 0,1856 > 0,001$

Kesimpulan: Karena MSE masih lebih besar dari target error, maka ulangi langkah 4-9 hingga didapatkan MSE yang lebih kecil dari target, atau lakukan sampai epoch maksimum tercapai untuk memenuhi kondisi berhenti.

BAB III

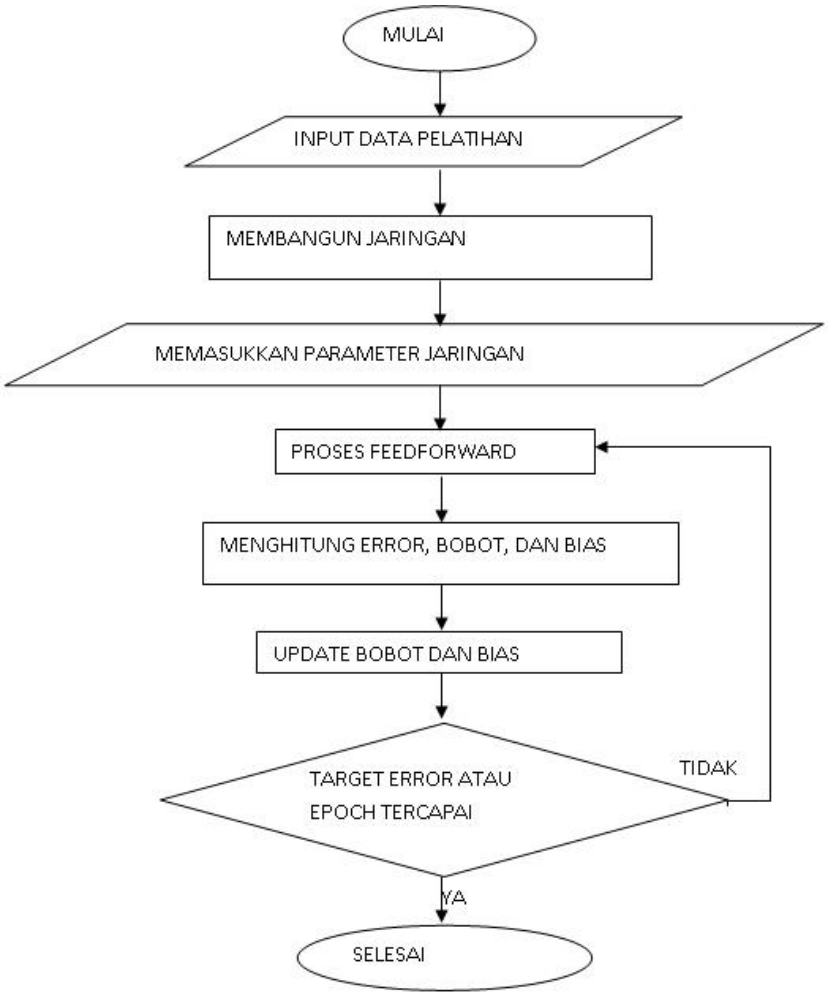
PEMBAHASAN

Peramalan beban yang dilakukan merupakan peramalan beban dengan jangka waktu satu minggu kedepan, sehingga dapat dikategorikan kedalam beban listrik jangka pendek. Data beban listrik yang digunakan adalah data per jam selama 24 jam. Diagram alir pengolahan data dapat dilihat pada gambar 3.1.



Gambar 3.1: Diagram Alir Pengolahan Data

Resilient propagation (RPROP) dikembangkan oleh Martin Riedmiller dan Heinrich Braun pada tahun 1992. RPROP merupakan algoritma yang digunakan untuk mempercepat pembelajaran pada pelatihan jaringan syaraf tiruan *backpropagation*. Berikut ini adalah diagram alir algoritma pelatihan RPROP.



Gambar 3.2: Diagram Alir Algoritma Pelatihan RPROP

3.1 Pengumpulan Data

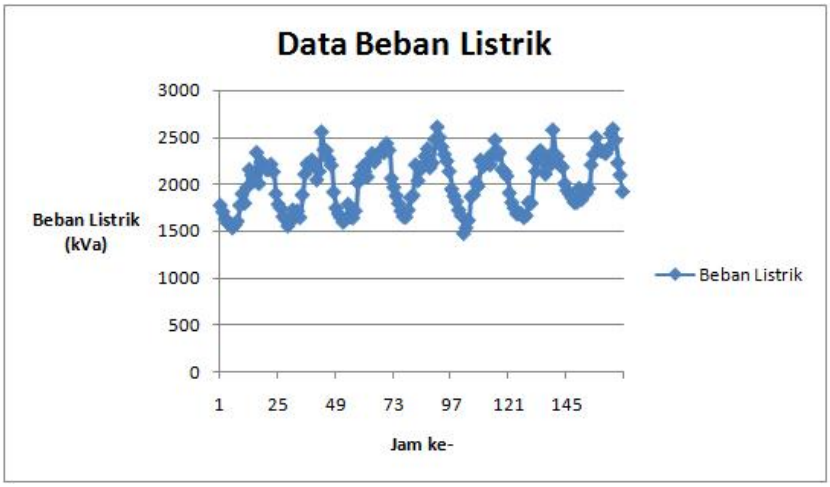
Peramalan beban yang dilakukan merupakan peramalan beban dengan jangka waktu satu minggu ke depan, sehingga dapat dikategorikan ke dalam peramalan beban listrik jangka pendek. Data beban listrik yang digunakan adalah data perjam selama 24 jam. Data yang digunakan merupakan data pada hari yang sama. Misal hari yang akan diramal adalah hari senin, maka data input beban listrik yang dimasukkan juga merupakan data historis pada hari senin.

Hal ini dilakukan dengan mempertimbangkan bahwa pola beban di suatu hari akan cenderung berbeda dengan hari lainnya, misalkan pola beban dihari kerja akan berbeda dengan diakhir pekan. Data yang digunakan berasal dari PT.PLN Persero Area Jaringan Kramat Jati pada periode 13 Juli 2015 sampai 16 Agustus 2015. Data pada tangga tersebut digunakan karena lengkap dan tidak ada nilai pencilan yang akan mengurangi akurasi peramalan. Berikut adalah contoh data input yang digunakan pada Tabel (3.1).

Tabel 3.1: Data Beban Listrik (dalam kVa)

Jam ke-	13-07-2015	20-07-2015	27-07-2015	03-08-2015	10-08-2015 (Target)
00.00-01.00	1780	1980	1740	2170	1820
01.00-02.00	1710	1930	1700	2085	1735
02.00-03.00	1640	1850	1610	2040	1670
....	....	....	....	....	....

Umumnya pergerakan beban listrik akan meningkat pada pagi ke siang hari dan akan mulai kembali menurun pada sore ke malam hari. Untuk lebih jelas lagi, pergerakan data beban listrik dapat dilihat pada gambar.



Gambar 3.3: Perubahan Beban Listrik Setiap Jam Selama Satu Minggu

3.1.1 Data *Training* dan Data *Testing*

Pada jaringan syaraf tiruan data yang akan digunakan dibagi menjadi dua terlebih dahulu, yaitu data *training* dan data *testing*. Data *training* digunakan untuk membentuk model, baik model regresi logistik ordinal ataupun model jaringan syaraf tiruan. Sedangkan data *testing* digunakan untuk menguji ketepatan klasifikasi dari model yang telah terbentuk. Ketepatan klasifikasi kedua model nantinya akan dibandingkan, baik untuk data *training* maupun data *testing*.

Data beban listrik yang digunakan adalah data dari 13 Juli 2015 sampai dengan 16 Agustus 2015 berjumlah 168 data. Data yang dipakai untuk *training* sebesar 90% dan untuk *testing* sebesar 10% dari seluruh data. Nilai 90% dan 10% didapatkan dari uji coba data *training* mulai dari 75%, 80%, dan 85%. Pada ketiga nilai tersebut, MSE masih cukup besar, tetapi ketika data *training* diperbesar menjadi 90%, nilai MSE sudah cukup kecil. Uji coba pembagian data *training* dan *testing* sehingga didapatkan 90% dan 10% dapat dilihat pada lampiran.

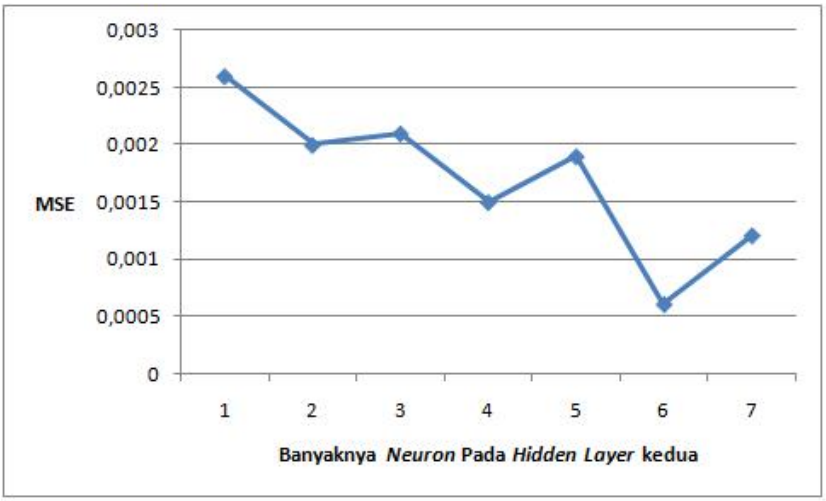
- Data *Training*: data 1 sampai 151 (90%)
- Data *Testing*: data 152 sampai 168 (10%)

3.2 Parameter dan Arsitektur *Resilient Propagation*

Arsitektur jaringan syaraf tiruan yang dipakai adalah *multi layer networks*. *Multi layer networks* memiliki satu atau lebih lapisan yang terletak diantara *input layer* dan *output layer* (memiliki satu atau lebih lapisan tersembunyi). Variasi jumlah *hidden layer* dapat ditentukan dalam pelatihan (Kusumadewi, 2006).

Model jaringan terdiri atas *input layer*, *hidden layer* dan *output layer*. Jumlah *neuron hidden layer* ditentukan dengan cara *trial and error* hingga ditemukan arsitektur terbaik dengan MSE terkecil. Pada penelitian ini telah dilakukan beberapa percobaan *trial and error* sehingga dipilih arsitektur dengan MSE yang cukup kecil. Jaringan terdiri dari satu *input layer* dengan empat *neuron*, dua *hidden layer* dengan *hidden layer* pertama terdiri dari sepuluh atau dua belas *neuron*, dan *hidden layer* kedua enam *neuron*, serta satu *output layer* dengan satu *neuron*. Data dilatihkan ke jaringan hingga *error* atau *epoch* tercapai.

Pada *hidden layer* kedua dipilih neuron sebanyak enam buah karena dari beberapa uji coba yang dilakukan mulai dari satu sampai tujuh *neuron*, didapatkan MSE terkecil pada enam *neuron* di *hidden layer* kedua. Berikut adalah grafik uji coba *neuron* pada *hidden layer* kedua.



Gambar 3.4: Uji Coba Jumlah *Neuron* Pada *Hidden Layer* Kedua

Dari grafik terlihat MSE terus turun, dan berada dititik terendah ketika neuron pada *hidden layer* kedua berjumlah enam. Tetapi kemudian meningkat lagi ketika neuron ditambahkan menjadi tujuh. Oleh karena itu, dipilih neuron berjumlah enam pada *hidden layer* kedua.

Selanjutnya ada beberapa nilai pada parameter yang ditetapkan yaitu $\Delta_{maks} = 50$, $\Delta_{min} = 10^{-6}$, $\eta^+ = 1,2$, $\eta^- = 0,5$, dan $\Delta_0 = 0.1$ (Riedmiller dan Braun, 1993). Untuk nilai Δ_{maks}, Δ_{min}, η^+, η^-, Δ_0 dibeberapa jurnal dalam maupun luar negeri, nilainya relatif sama dan mengacu pada jurnal *Resilient Propagation* milik Riedmiller dan Braun.

Selain itu, dilakukan juga uji coba pada fungsi aktivasi yang digunakan. Yaitu fungsi aktivasi di *hidden layer* pertama dan kedua, sedangkan untuk fungsi aktivasi pada *output layer* ditetapkan dengan menggunakan fungsi purelin. Kombinasi fungsi aktivasi yang digunakan adalah logsig-tansig-purelin, tansig-logsig-purelin, logsig-logsig-purelin, dan tansig-tansig-purelin.

3.3 Hasil dan Pembahasan

Dalam penulisan ini, *training* dan *testing* data dilakukan dengan menggunakan GUI pada Matlab R2012b.

3.3.1 Normalisasi Data

Normalisasi data adalah proses pengubahan data asli menjadi data dengan *range* antara 0,1 sampai 0,9. Akan dilakukan normalisasi data dengan menggunakan rumus pada persamaan 2.1. Berikut adalah hasil normalisasi data target pada data *training* dan *testing*.

Tabel 3.2: Normalisasi data target pada data *Training*

Data Target Sebelum dinormalisasi	Data Target Sesudah dinormalisasi
1820	0,2984
1735	0,2440
1670	0,2024
1630	0,1768
1600	0,1576
1590	0,1512
.	.
.	.
.	.

Tabel 3.3: Normalisasi data target pada data *Testing*

Data Target Sebelum dinormalisasi	Data Target Sesudah dinormalisasi
1760	0,1
1830	0,1518
1760	0,1
2430	0,5962
2390	0,5666
2450	0,6111
.	.
.	.
.	.

3.3.2 Hasil *Training* dan *Testing* Data

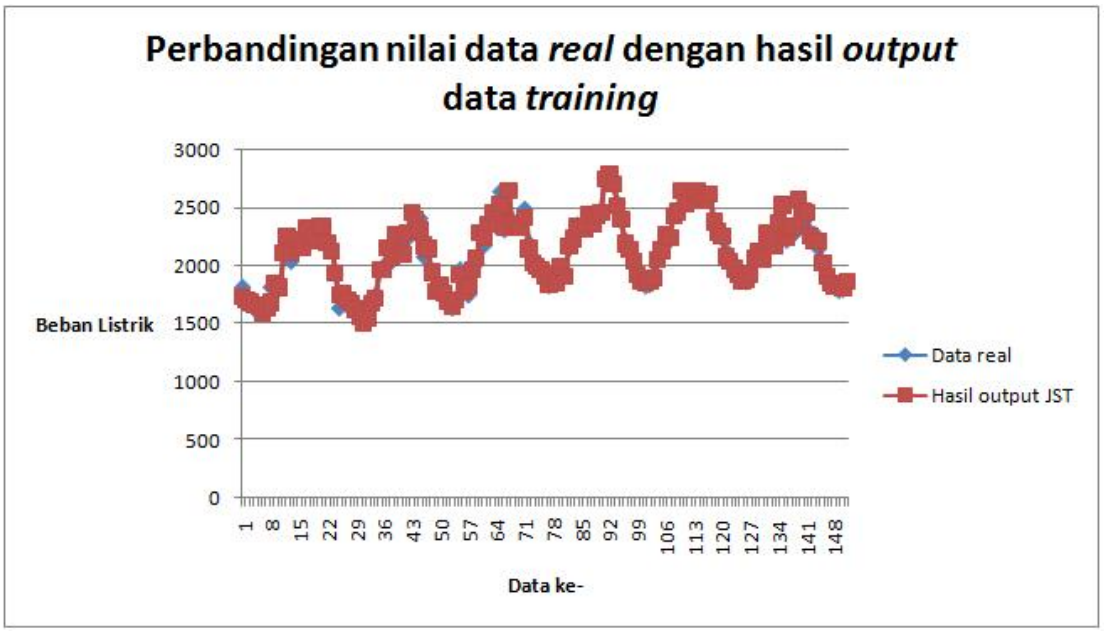
Keberhasilan JST RPROP diukur dari *Mean Square Error* (MSE). Semakin kecil nilai MSE, maka kinerja JST RPROP semakin baik. Uji coba dilakukan dengan memodifikasi jumlah *neuron* pada *hidden layer* pertama, dan juga kombinasi fungsi aktivasi pada *hidden layer* pertama dan kedua. Berikut adalah tabel hasil uji coba Algoritma *Resilient Propagation*

Tabel 3.4: MSE Terkecil

Neuron pada *hidden layer* pertama	MSE Training	MSE Testing	Fungsi Aktivasi
10	0,0017	0,0371	logsig-tansig-purelin
	0,0017	0,0198	tansig-logsig-purelin
	0,00061	0,0366	logsig-logsig-purelin
	0,0018	0,0394	tansig-tansig-purelin
12	0,0027	0,0414	logsig-tansig-purelin
	0,0015	0,0425	tansig-logsig-purelin
	0,0016	0,0720	logsig-logsig-purelin
	0,0015	0,0355	tansig-tansig-purelin

Dari tabel dapat dilihat bahwa MSE *training* paling kecil ada pada baris

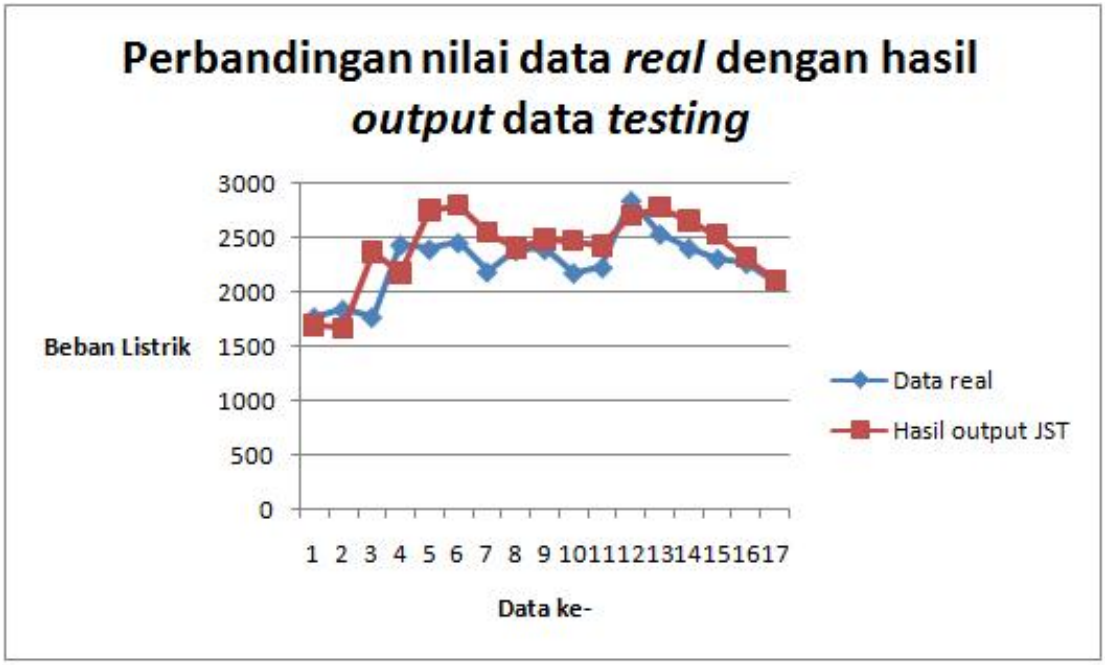
ketiga pada tabel (3.4), dengan *neuron* pada *hidden layer* pertama sebanyak 10 buah. Ini berarti, arsitektur JST yang digunakan adalah 4-10-6-1 yaitu 4 neuron pada *input layer*, 10 neuron pada *hidden layer* pertama, 6 *neuron* pada *hidden layer* kedua, dan 1 neuron pada *output layer*, dan dengan kombinasi fungsi aktivasi logsig-logsig-purelin. Tetapi untuk data *testing*, MSE terkecil ada pada arsitektur JST 4-10-6-1 dengan kombinasi fungsi aktivasi tansig-logsig-purelin. Berikut adalah grafik hasil perbandingan data *training* MSE terkecil dengan beban yang sebenarnya.



Gambar 3.5: Perbandingan beban real data *training* dengan hasil JST

Dapat dilihat bahwa data hasil prediksi dapat mengikuti pola meskipun masih terdapat selisih. Hal ini menunjukkan bahwa JST Algoritma *Resilient Propagation* dengan arsitektur 4-10-6-1 dan fungsi aktivasi logsig-logsig-purelin mampu dijadikan salah satu alat untuk meramalkan beban listrik jangka pendek karena memiliki MSE yang cukup kecil yaitu 0,00061.

Sedangkan untuk data *testing* dengan arsitektur dan fungsi aktivasi yang sama, berikut adalah hasil perbandingan dengan beban yang sebenarnya.



Gambar 3.6: Perbandingan beban real data *testing* dengan hasil JST

Pada grafik terlihat bahwa masih terjadi beberapa perbedaan pada data target dengan hasil *output* JST. Hal ini dikarenakan MSE sebesar 0,0366. Umumnya, pada jaringan syaraf tiruan, nilai MSE *testing* lebih besar dari MSE *training*. Tetapi, karena MSE *training* pada arsitektur 4-10-6-1 dan fungsi aktivasi logsig-logsig-purelin cukup kecil yaitu 0,00061, maka dapat dikatakan Algoritma *Resilient Propagation* dapat dijadikan salah satu referensi untuk meramalkan beban listrik jangka pendek.

Untuk perbandingan data pada uji coba arsitektur dan fungsi aktivasi yang lainnya dapat dilihat pada lampiran.

BAB IV

PENUTUP

4.1 Kesimpulan

1. Dari uji coba yang telah dilakukan menggunakan algoritma *resilient propagation*, menghasilkan MSE terkecil sebesar 0,00061 dengan arsitektur JST 4-10-6-1 yang berarti 4 *neuron* pada *input layer*, 10 *neuron* pada *hidden layer* pertama, 6 *neuron* pada *hidden layer* kedua, dan 1 *neuron* pada *output layer*.

2. Dari hasil MSE dapat disimpulkan tingkat akurasi dengan metode JST algoritma *Resilient Propagation* sudah cukup baik dan dapat digunakan untuk meramalkan beban listrik jangka pendek di PT.PLN Persero Area Jaringan Kramat Jati.

4.2 Saran

1. Jumlah *hidden layer* dan *neuron* di *hidden layer* dapat ditambahkan.

2. Variasi kombinasi fungsi aktivasi dapat diperluas dengan menambahkan fungsi aktivasi jenis lain.

DAFTAR PUSTAKA

Ismawan, Anggara., Setiawardhana, dan Dwi Kurnia Basuki, "Aplikasi Jaringan Syaraf Tiruan Sebagai Penerjemah Karakter Braille ke Bentuk Abjad", Institut Teknologi Sepuluh November, 2011.

Khair, Aulia. 2011. "Peramalan Beban Listrik Jangka Pendek Menggunakan Kombinasi Metode *Autoregressive Integrated Moving Average* (ARIMA) dengan Regresi Linear Antara Suhu dan Daya Listrik". Depok.

Kusumadewi, Sri. 2004. "Membangun Jaringan Syaraf Tiruan (Menggunakan Matlab & Excel (ink)". Yogyakarta: Graha Ilmu.

Lobo, Deogracias Gama Da Costa., dan Stefanus Santosa , "Prediksi Penjualan Air Minum dalam Kemasan Menggunakan Jaringan Syaraf Tiruan Resilient Propagation", Jurnal Teknologi Informasi Universitas Dian Nuswantoro Volume 10 Nomor 2, 2014.

Mishra, Dharmendra Kumar., A.K.D Dwivedi, dan S.P Tripathi, "Efficient Algorithms For Load Forecasting in Electric Power System Using Artificial Nural Network", International Journal of Latest Research in Science and Technology, 2012.

Rahman, Fajar Alya. 2012. "Peramalan Beban Listrik Jangka Pendek Menggunakan Jaringan Syaraf Tiruan". Depok.

Ramadhani, Syaifudin., dan Urifatun Anis, "Klasifikasi Penyakit Kencing Manis Menggunakan Jaringan Syaraf Tiruan dengan Metode Backpropagation", Jurnal Fakultas Teknik Universitas Islam Lamongan, 2009.

Riedmiller, Martin., dan Heinrich Braun , "A Direct Adaptive Method For Faster Backpropagation Learning: The RPROP Algorithm", Institute fur Logik, Komplexitat und Deduktionssyteme University of Karlsruhe, 2012.

Rojas, Raul. 1996. " *Neural Networks: a Systematic Introduction*". Berlin: Springer-Verlag.

Saputro, Donny Wahyu. 2006. "Pengenalan Karakter Tulisan Tangan dengan Menggunakan Jaringan Syaraf Tiruan Propagasi Balil Resilient". Bogor.

Siang, Jong Jek. 2009. Jaringan Syaraf Tiruan dan Pemrogramannya Menggunakan Matlab". Yogyakarta: CV Andi Offset.

Simsar, Shima., Mahmood Alborzi, Jamshid Nazemi, dan Mahmood Abbasi Layegh, "Forecasting Power Demand Using Neural Networks Model", International Journal of Engineering and Advanced Technology, 2013.

LAMPIRAN-LAMPIRAN

SOURCE CODE *RESILIENT PROPAGATION* PADA MATLAB R2012B

```
%-----TRAINING-----

%data training 90%
p = [1780 1710 1640 1590 1580 1540 1570 1610 1780 1900 1800 1970 2160
2100 ... ];
t= [1820 1735 1670 1630 1600 1590 1670 1820 1860 1800 2120 2250 2040
2220 ... ];

%Normalisasi data:
min_p = min(min(p)); %nilai terkecil dari p
max_p = max(max(p)); %nilai terbesar dari p
pn = (0.8*(p-min_p)/(max_p-min_p))+0.1; %normalisasi p
min_t = min(min(t)); %nilai terkecil dari t
max_t = max(max(t)); %nilai terbesar dari t
tn=(0.8*(t-min_t)/(max_t-min_t))+0.1; %normalisasi t

%Menentukan banyaknya neuron hidden layer
%dan fungsi aktivasi yang digunakan
net = newff(minmax(pn),[12 6 1],{'tansig' 'logsig' 'purelin'},'trainrp');

%set bobot awal
```

```
net=init(net);

%input layer ke hidden layer pertama
net.IW{1,1} = [-0.68 0.15 0.36 0.39;
-0.26 0.08 0.98 0.42;
-0.97 0.45 0.94 0.34;
0.66 -0.90 0.45 0.06;
0.93 -0.88 0.99 0.89;
0.69 -0.83 0.62 0.56;
0.87 0.50 -0.09 0.51;
0.73 0.94 -0.42 0.08;
0.19 0.93 -0.30 0.95;
0.81 0.32 0.32 -0.02;
0.19 0.93 -0.30 0.95;
0.81 0.32 0.32 -0.02];

%bias input ke hidden 1
net.b{1,1} = [-0.80;
0.70;
-0.40;
0.67;
-0.87;
0.24;
0.52;
-0.77;
0.76;
```

```
-0.41;
0.76;
-0.41];

%=====================================================

%hidden 1 ke hidden 2

net.LW{2,1} = [0.61 0.54 0.66 0.89 0.97 0.30 0.16 0.29 0.25 0.72 0.25
0.72;
0.70 0.50 0.22 0.27 0.85 0.27 0.24 0.44 0.98 0.44 0.25 0.72;
0.92 0.42 0.41 0.15 0.03 0.81 0.35 0.54 0.88 0.44 0.25 0.72;
0.99 0.90 0.06 0.39 0.21 0.17 0.49 0.76 0.19 0.52 0.25 0.72;
0.38 0.38 0.67 0.81 0.36 0.76 0.21 0.30 0.18 0.29 0.25 0.72;
0.38 0.38 0.67 0.81 0.36 0.76 0.21 0.30 0.18 0.29 0.25 0.72];

%bias hidden 1 ke hidden 2
net.b{2,1} = [0.51;
0.63;
0.87;
0.68;
0.07;
0.65];

%hidden 2 ke output
net.LW{3,2} = [0.64 0.89 0.46 0.62 0.58 0.85];
```

```
%bias hidden 2 ke output
net.b{3,1} = 0.95;

%Melihat bobot awal input, lapisan, dan bias
bobotAkhir_input=net.IW{1,1};
bobotAkhir_bias_input=net.b{1,1};
bobotAkhir_lapisan1=net.LW{2,1};
bobotAkhir_bias_lapisan1=net.b{2,1};
bobotAkhir_lapisan2=net.LW{3,2};
bobotAkhir_bias_lapisan2=net.b{3,1};

%Menentukan parameter jaringan
%Parameter tetap pada RPROP
net.trainParam.deltamax = 50;
net.trainParam.delt_inc = 1.2;
net.trainParam.delt_dec = 0.5;
net.trainParam.delta0 = 0.1;

%Parameter lain pada RPROP
net.trainParam.epochs = 20000;
net.trainParam.show = 10;
net.trainParam.goal = 1e-5;
net.trainParam.time = inf;
net.trainParam.lr = 0.1;
net.trainParam.min_grad = 1e-6;
```

```
%Melakukan pembelajaran
net=train(net,pn,tn); %training pn dan tn

%Melakukan simulasi (Hasil bobot baru)
an=sim(net,pn);

%Denormalisasi
td = round(((an-0.1)*(max_t-min_t))/0.8)+min_t;

%MSE Training
xx=(tn-an).^2;
MSE_TR=mean(xx);

%-----TESTING-----

%Input baru Q akan dites dengan target TQ
%data testing 10%

Q = [1900 1910 1960 2210 2320 2500 2390 2360 2350 2330 2380 2540 2590
2470  2230 2100 1925;
1760 1770 1850 1850 1990 1980 1980 2090 1990 2060 2280 2220 2270 2250
2080  2115 2020;
1720 1670 1920 2120 2220 2140 2210 2130 2340 2010 2260 2310 2500 2510
2500  2350 2100;
1580 1620 1670 1870 1770 1790 1970 1960 1990 1890 1930 2060 2250 2200
```

```
2090  2005 1870];

TQ = [1760 1830 1760 2430 2390 2450 2180 2380 2390 2170 2220 2840 2530
2400  2300 2260 2090];

%Normalisasi input baru
%Normalisasi:
min_Q = min(min(Q)); %nilai terkecil dari Q
max_Q = max(max(Q)); %nilai terbesar dari Q
Qn = (0.8*(Q-min_Q)/(max_Q-min_Q))+0.1; %normalisasi Q
min_TQ = min(min(TQ)); %nilai terkecil dari TQ
max_TQ = max(max(TQ)); %nilai terbesar dari TQ
TQn=(0.8*(TQ-min_TQ)/(max_TQ-min_TQ))+0.1; %normalisasi TQ

%Melakukan simulasi
bn=sim(net,Qn);

%denormalisasi
tqd = round(((bn-0.1)*(max_TQ-min_TQ))/0.8)+min_TQ;

%MSE testing
XX=(TQn-bn).^2;
MSE_TES=mean(XX);
```

DATA BEBAN PENYULANG GARDU INDUK CAWANG BARU

Tabel 4.1: Data Beban Penyulang Gardu Induk Cawang Baru (kVa)

No.	Target	t-1	t-2	t-3	t-4
1	1820	1780	1980	1740	2170
2	1735	1710	1930	1700	2085
3	1670	1640	1850	1610	2040
4	1630	1590	1810	1540	2000
5	1600	1580	1730	1470	2015
6	1590	1540	1720	1460	2140
7	1670	1570	1750	1560	2100
8	1820	1610	1810	1640	2080
9	1860	1780	1820	1645	1960
10	1800	1900	1950	1770	2350
11	2120	1800	2190	1900	2380
12	2250	1970	2520	1920	2410
13	2040	2160	2410	2160	2490
14	2220	2100	2310	2120	2610
15	2150	2040	2310	2060	2510
16	2170	2340	2540	2030	2470
17	2320	2010	2310	2160	2420
18	2230	2240	2500	2070	2600
19	2290	2210	2570	2250	2570
20	2310	2160	2510	2285	2640
21	2340	2160	2640	2070	2580
22	2180	2215	2450	2020	2530
23	2150	2135	2500	1930	2500
24	1930	1900	2280	2035	2420
25	1640	1790	2000	1850	2200
26	1720	1750	1940	1780	2130
27	1660	1660	1870	1690	2090
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
166	2300	2230	2080	2500	2090
167	2260	2115	2350	2005	2005
168	2090	2020	2100	1870	1870

Tabel 4.2: Hasil Output Data Training dan Testing dengan Arsitektur 4-10-6-1 logsig-logsig-purelin

No.	Target Training	Output Training	Target Testing	Output Testing
1	1820	1735	1760	1692
2	1735	1698	1830	1664
3	1670	1682	1760	2359
4	1630	1655	2430	2169
5	1600	1655	2390	2741
6	1590	1589	2450	2784
7	1670	1636	2180	2535
8	1820	1680	2380	2399
9	1860	1852	2390	2475
10	1800	1810	2170	2465
11	2120	2106	2220	2413
12	2250	2251	2840	2697
13	2040	2116	2530	2766
14	2220	2230	2400	2642
15	2150	2150	2300	2515
16	2170	2162	2260	2306
17	2320	2331	2090	2100
18	2230	2162		
19	2290	2249		
20	2310	2335		
21	2340	2247		
22	2180	2192		
23	2150	2127		
24	1930	1935		
25	1640	1742		
26	1720	1753		
27	1660	1703		
.	.	.		
.	.	.		
.	.	.		
149	1790	1815		
150	1800	1799		
151	1860	1861		

Tabel 4.3: Hasil Output Data Training dan Testing dengan Arsitektur 4-10-6-1 logsig-tansig-purelin

No.	Target Training	Output Training	Target Testing	Output Testing
1	1820	1762	1760	2260
2	1735	1716	1830	2268
3	1670	1662	1760	2226
4	1630	1592	2430	2179
5	1600	1619	2390	2392
6	1590	1598	2450	2261
7	1670	1678	2180	2281
8	1820	1711	2380	2459
9	1860	1755	2390	2383
10	1800	1762	2170	2427
11	2120	2043	2220	2451
12	2250	2236	2840	2690
13	2040	2190	2530	2750
14	2220	2139	2400	2715
15	2150	2114	2300	2532
16	2170	2148	2260	2417
17	2320	2135	2090	2169
18	2230	2204		
19	2290	2275		
20	2310	2305		
21	2340	2295		
22	2180	2160		
23	2150	2221		
24	1930	2093		
25	1640	1797		
26	1720	1766		
27	1660	1727		
28	1660	1710		
.	.	.		
.	.	.		
.	.	.		
149	1790	1807		
150	1800	1794		
151	1860	1870		

Tabel 4.4: Hasil Output Data Training dan Testing dengan Arsitektur 4-10-6-1 tansig-tansig-purelin

No.	Target Training	Output Training	Target Testing	Output Testing
1	1820	1750	1760	2278
2	1735	1725	1830	2095
3	1670	1685	1760	2180
4	1630	1665	2430	2299
5	1600	1644	2390	2601
6	1590	1612	2450	2717
7	1670	1628	2180	2534
8	1820	665	2380	2495
9	1860	1800	2390	2530
10	1800	1749	2170	2488
11	2120	2072	2220	2464
12	2250	2281	2840	2800
13	2040	2140	2530	2932
14	2220	2228	2400	2832
15	2150	2241	2300	2538
16	2170	2211	2260	2362
17	2320	2217	2090	2167
18	2230	2190		
19	2290	2301		
20	2310	2310		
21	2340	2269		
22	2180	2141		
23	2150	2188		
24	1930	1954		
25	1640	1788		
26	1720	1749		
27	1660	1696		
28	1660	1669		
.	.	.		
.	.	.		
.	.	.		
149	1790	1808		
150	1800	1839		
151	1860	1866		

Tabel 4.5: Hasil Output Data Training dan Testing dengan Arsitektur 4-10-6-1 tansig-logsig-purelin

No.	Target Training	Output Training	Target Testing	Output Testing
1	1820	1815	1760	1833
2	1735	1708	1830	1878
3	1670	1739	1760	2023
4	1630	1681	2430	2191
5	1600	1568	2390	2341
6	1590	1566	2450	2199
7	1670	1657	2180	2504
8	1820	1785	2380	2303
9	1860	1825	2390	2604
10	1800	1801	2170	2068
11	2120	2041	2220	2516
12	2250	2221	2840	2640
13	2040	2211	2530	2628
14	2220	2143	2400	2573
15	2150	2127	2300	2474
16	2170	2143	2260	2462
17	2320	2168	2090	2177
18	2230	2224		
19	2290	2286		
20	2310	2276		
21	2340	2320		
22	2180	2163		
23	2150	2188		
24	1930	2129		
25	1640	1700		
26	1720	1760		
27	1660	1681		
28	1660	1640		
.	.	.		
.	.	.		
.	.	.		
149	1790	1808		
150	1800	1839		
151	1860	1866		

Tabel 4.6: Hasil Output Data Training dan Testing dengan Arsitektur 4-12-6-1 logsig-logsig-purelin

No.	Target Training	Output Training	Target Testing	Output Testing
1	1820	1745	1760	2568
2	1735	1680	1830	2574
3	1670	1700	1760	2192
4	1630	1649	2430	2079
5	1600	1607	2390	2153
6	1590	1573	2450	2057
7	1670	1706	2180	2419
8	1820	1731	2380	2219
9	1860	1829	2390	2604
10	1800	1831	2170	2035
11	2120	2093	2220	2461
12	2250	2224	2840	2579
13	2040	2191	2530	2795
14	2220	2150	2400	2739
15	2150	2120	2300	2745
16	2170	2150	2260	2418
17	2320	2115	2090	2078
18	2230	2216		
19	2290	2270		
20	2310	2239		
21	2340	2278		
22	2180	2177		
23	2150	2226		
24	1930	2077		
25	1640	1784		
26	1720	1721		
27	1660	1684		
28	1660	1668		
.	.	.		
.	.	.		
.	.	.		
149	1790	1791		
150	1800	1771		
151	1860	1907		

Tabel 4.7: Hasil Output Data Training dan Testing dengan Arsitektur 4-12-6-1 logsig-tansig-purelin

No.	Target Training	Output Training	Target Testing	Output Testing
1	1820	1731	1760	2430
2	1735	1635	1830	2383
3	1670	1676	1760	2329
4	1630	1729	2430	2129
5	1600	1581	2390	2236
6	1590	1598	2450	2089
7	1670	1700	2180	2373
8	1820	1786	2380	2355
9	1860	1831	2390	2434
10	1800	1816	2170	2214
11	2120	1976	2220	2435
12	2250	2150	2840	2508
13	2040	2209	2530	2722
14	2220	2175	2400	2678
15	2150	2134	2300	2535
16	2170	2194	2260	2395
17	2320	2139	2090	2086
18	2230	2225		
19	2290	2306		
20	2310	2297		
21	2340	2273		
22	2180	2177		
23	2150	2156		
24	1930	2080		
25	1640	1789		
26	1720	1696		
27	1660	1646		
28	1660	1696		
.	.	.		
.	.	.		
.	.	.		
149	1790	1819		
150	1800	1795		
151	1860	1896		

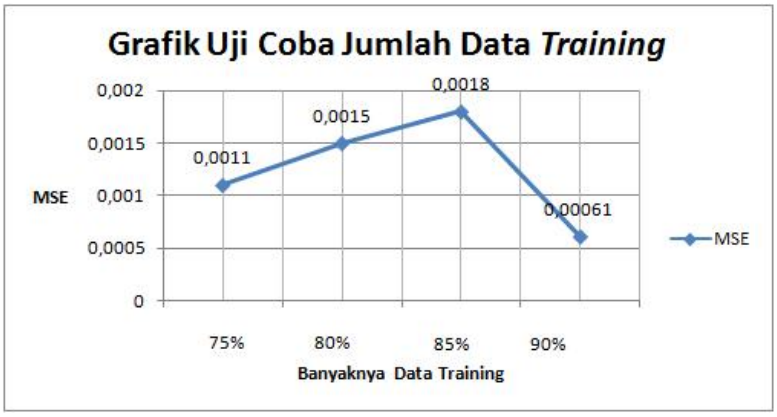
Tabel 4.8: Hasil Output Data Training dan Testing dengan Arsitektur 4-12-6-1 tansig-tansig-purelin

No.	Target Training	Output Training	Target Testing	Output Testing
1	1820	1719	1760	2072
2	1735	1693	1830	2035
3	1670	1669	1760	2117
4	1630	1646	2430	2207
5	1600	1652	2390	2372
6	1590	1630	2450	2680
7	1670	1646	2180	2517
8	1820	1669	2380	2474
9	1860	1775	2390	2452
10	1800	1825	2170	2298
11	2120	2001	2220	2420
12	2250	2249	2840	2565
13	2040	2253	2530	2860
14	2220	2272	2400	2820
15	2150	2084	2300	2565
16	2170	2246	2260	2387
17	2320	2287	2090	2118
18	2230	2192		
19	2290	2286		
20	2310	2248		
21	2340	2350		
22	2180	2132		
23	2150	2146		
24	1930	2096		
25	1640	1761		
26	1720	1735		
27	1660	1689		
28	1660	1673		
.	.	.		
.	.	.		
.	.	.		
149	1790	1760		
150	1800	1812		
151	1860	1901		

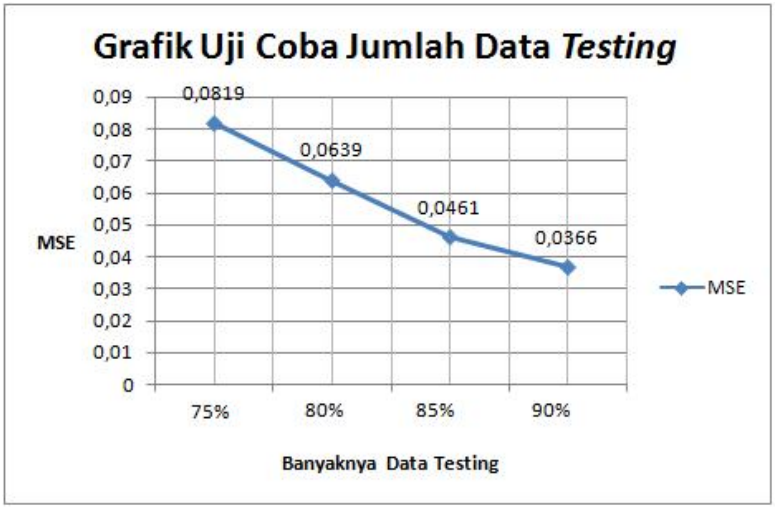
Tabel 4.9: Hasil Output Data Training dan Testing dengan Arsitektur 4-12-6-1 tansig-logsig-purelin

No.	Target Training	Output Training	Target Testing	Output Testing
1	1820	1748	1760	1863
2	1735	1743	1830	2031
3	1670	1590	1760	2318
4	1630	1588	2430	2371
5	1600	1622	2390	3076
6	1590	1611	2450	2857
7	1670	1623	2180	2457
8	1820	1695	2380	2838
9	1860	1717	2390	2443
10	1800	1813	2170	2621
11	2120	2074	2220	1804
12	2250	2263	2840	2056
13	2040	2187	2530	2640
14	2220	2222	2400	2740
15	2150	2118	2300	2476
16	2170	2226	2260	2449
17	2320	2094	2090	2101
18	2230	2218		
19	2290	2288		
20	2310	2310		
21	2340	2287		
22	2180	2156		
23	2150	2173		
24	1930	2055		
25	1640	1727		
26	1720	1728		
27	1660	1768		
28	1660	1726		
.	.	.		
.	.	.		
.	.	.		
149	1790	1808		
150	1800	1801		
151	1860	1877		

GRAFIK UJI COBA JUMLAH DATA *TRAINING* DAN DATA *TESTING*

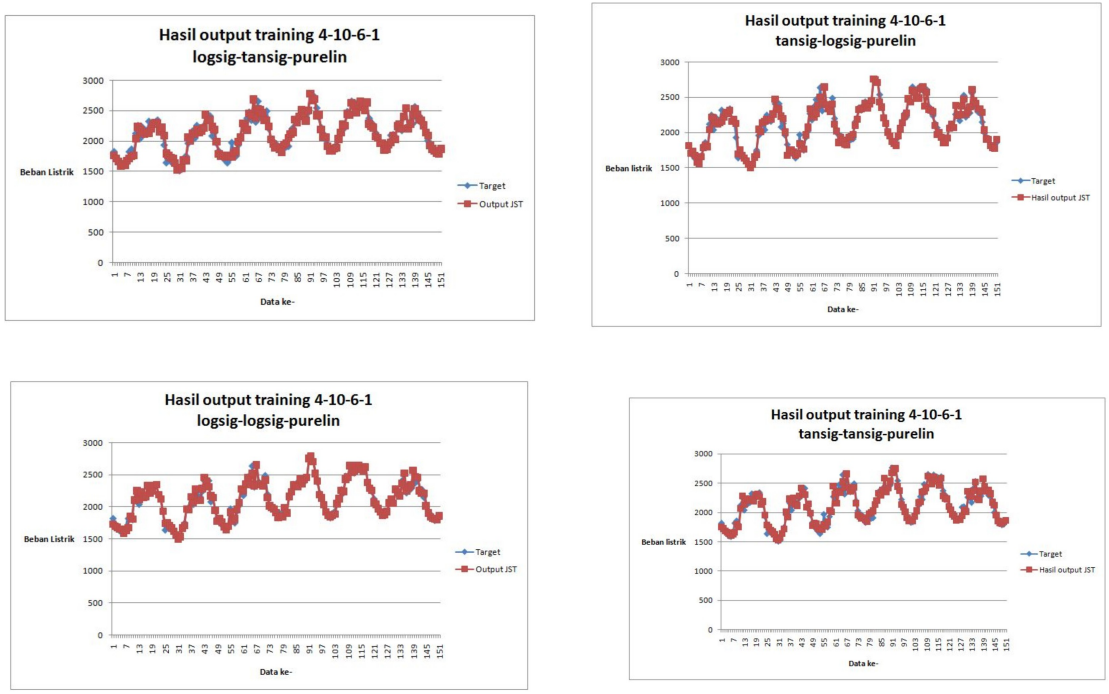


Gambar 4.1: Uji Coba Jumlah Data *Training*

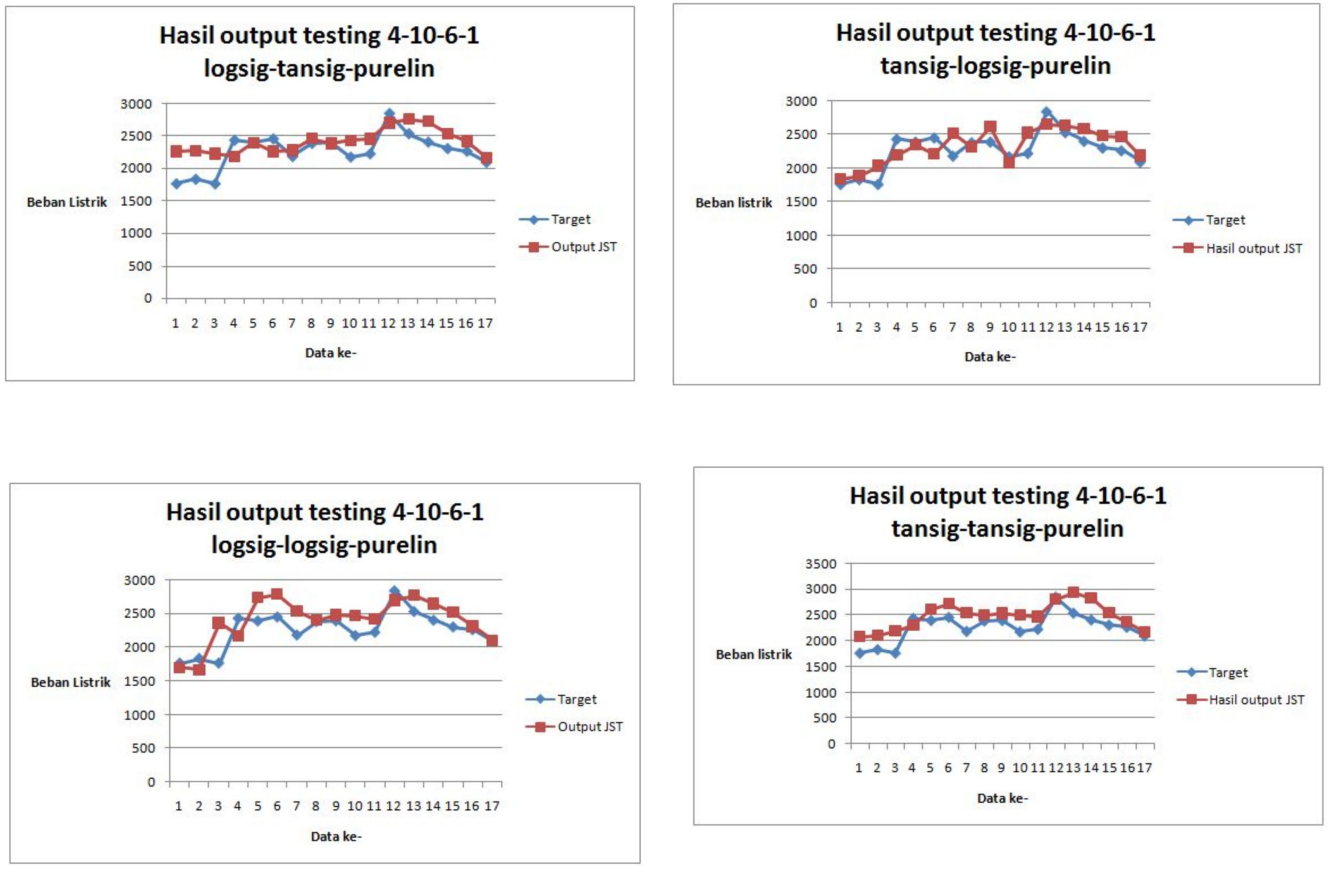


Gambar 4.2: Uji Coba Jumlah *Data Testing*

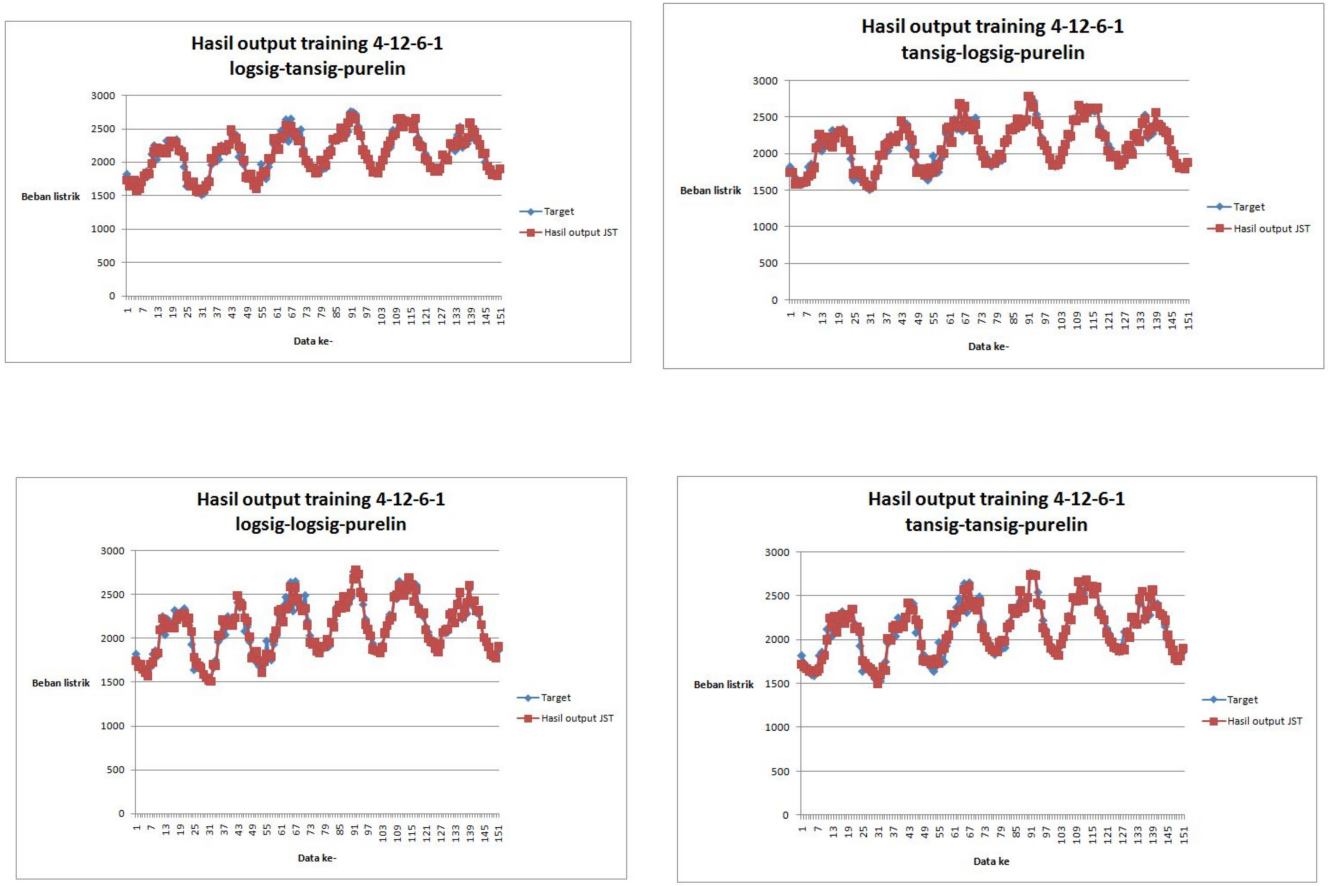
GRAFIK PERBANDINGAN HASIL *OUTPUT* DATA *TRAINING* DAN DATA *TESTING* DENGAN DATA SEBENARNYA



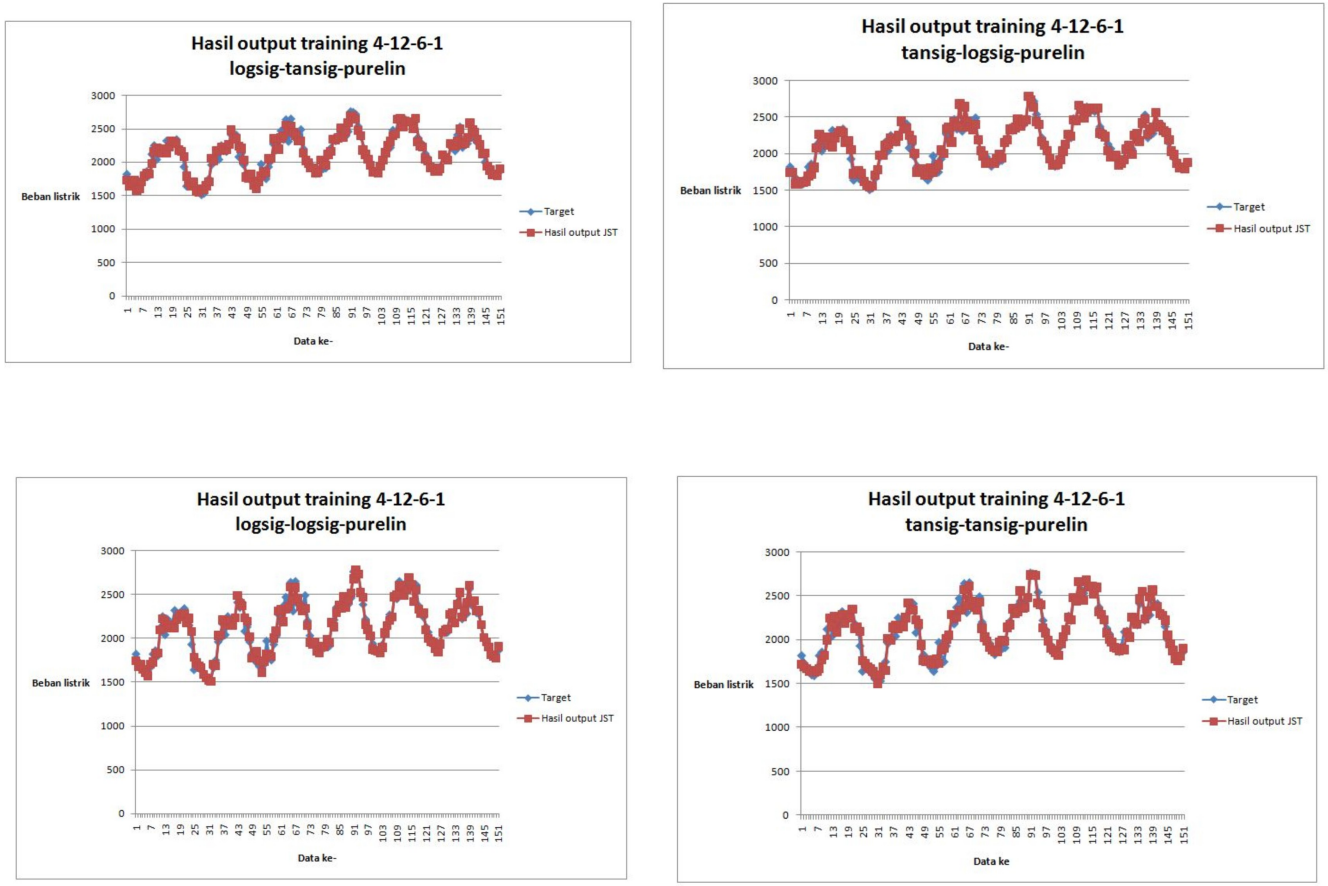
Gambar 4.3: Perbandingan *Output Training* JST dengan Beban Sebenarnya Pada Arsitektur 4-10-6-1



Gambar 4.4: Perbandingan *Output Testing* JST dengan Beban Sebenarnya Pada Arsitektur 4-10-6-1



Gambar 4.5: Perbandingan *Output Training* JST dengan Beban Sebenarnya Pada Arsitektur 4-12-6-1



Gambar 4.6: Perbandingan *Output Testing* JST dengan Beban Sebenarnya Pada Arsitektur 4-12-6-1

SURAT PERNYATAAN KEASLIAN SKRIPSI

Dengan ini saya yang bertanda tangan di bawah ini, mahasiswa Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta:

Nama	:	Syifa Aulia
No. Registrasi	:	3125111209
Jurusan	:	Matematika
Program Studi	:	Matematika

Menyatakan bahwa skripsi ini yang saya buat dengan judul "**Peramalan Beban Listrik Jangka Pendek Menggunakan Jaringan Syaraf Tiruan Algoritma *Resilient Propagation***" adalah :

1. Dibuat dan diselesaikan oleh saya sendiri.
2. Bukan merupakan duplikat skripsi yang pernah dibuat oleh orang lain atau jiplakan karya tulis orang lain.

Pernyataan ini dibuat dengan sesungguhnya dan saya bersedia menanggung segala akibat yang timbul jika pernyataan saya tidak benar.

Jakarta, Desember 2015

Yang membuat pernyataan

Syifa Aulia

DAFTAR RIWAYAT HIDUP

SYIFA AULIA. Lahir di Jakarta, 28 Mei 1992. Anak keenam dari pasangan Bapak (Alm) Ali Achmad dan Ibu Siti Maryanah. Saat ini bertempat tinggal di Jalan Batu Agung RT 009 / RW 015 No 32 Cililitan 13640.

No. Ponsel : 08989754954

Email : s.aulia28@gmail.com

Riwayat Pendidikan : Penulis mengawali pendidikan di SD Negeri 01 Cililitan Jakarta pada tahun 1998 - 2004. Setelah itu, penulis melanjutkan ke SMP Negeri 20 jakarta hingga tahun 2007. Kemudian kembali melanjutkan ke SMA Negeri 67 Jakarta dan lulus tahun 2010. Ditahun yang sama penulis melanjutkan ke Universitas Negeri Yogyakarta (UNY), jurusan Matematika melalui jalur SNMPTN Tertulis. Ditahun 2011 penulis diterima di Universitas Negeri Jakarta (UNJ), jurusan Matematika melalu jalur SNMPTN Tertulis. Di akhir tahun 2015 penulis telah memperoleh gelar Sarjana Sains untuk Jurusan Matematika, Program Studi Matematika, FMIPA, UNJ.

Riwayat Organisasi : Selama dibangku perkuliahan, penulis aktif diberbagai organisasi kemahasiswaaan. Dalam dua tahun pertama, penulis mendapat kepercayaan sebagai staff Departemen Informasi dan Komunikasi BEMJ Matematika. Penulis aktif diberbagai kegiatan seperti Pelangi Matematika, Calculus Cup, Anjangsana, MPA Fakultas dan lain sebagainya.

Riwayat Pekerjaan : Penulis mulai menjadi pengajar matematika sejak tahun 2010. Pada tahun 2014, penulis menjalani Praktik Kerja Lapangan di Kementerian Tenaga Kerja dan Transmigrasi.