IMPLEMENTASI ALGORITMA PELATIHAN LEVENBERG MARQUARDT DAN REGULARISASI BAYES UNTUK PREDIKSI CURAH HUJAN

Yasinta Lisa
STKIP Persada Khatulistiwa, Jl. Pertamina, Sengkuang, Sintang
yasintalisa@gmail.com

Abstract: *Levenberg Marquardt algorithm is used for training feedforward neural networks because of the effectiveness and convergence acceleration. Levenberg Marquardt is nonliear optimization method that used for backpropagation to find adjusted weights. Regularization can improve the performance of the neural network generalization. Regularization technique that often used is the regularization bayes. The data used in this study were monthly data of rainfall, average temperature, humidity, air pressure above the observation stations and the average wind speed at the observation station Pangsuma Putussibau West Kalimantan in 2008-2009. Training data was done on a neural network in a single hidden layer with three neurons and one neuron in the output layer. In this case, the study results showed that Levenberg Marquardt algorithm with regularization was better used.*

Keywords: *Prediction, Neural Networks, Levenberg Marquardt, Bayes Regularization, Rainfall*

Abstrak : Algoritma Levenberg Marquardt digunakan untuk pelatihan *feedforward neural network* karena keefektifan dan kecepatan konvergensinya. Levenberg Marquardt merupakan metode optimasi nonlinier yang digunakan pada saat koreksi error *backpropagation* untuk menemukan bobot yang disesuaikan. Salah satu cara untuk meningkatkan performa generalisasi jaringan syaraf tiruan adalah regularisasi. Tehnik regularisasi yang sering digunakan adalah regularisasi bayes. Penelitian ini dilakukan untuk mengetahui bagaimana kinerja jaringan syaraf tiruan dengan metode pelatihan Levenberg Marquardt dengan penambahan regularisasi untuk prediksi data *time series* serta membandingkan dengan algoritma Levenberg Marquardt tanpa regularisasi. Data yang digunakan dalam penelitian ini adalah data bulanan curah hujan, suhu rata-rata, kelembaban udara, tekanan udara di atas stasiun pengamatan dan kecepatan angin rata-rata di stasiun pengamatan Pangsuma Putussibau Kalimantan Barat dari tahun 2008-2009. Pelatihan data dilakukan pada jaringan syaraf pada satu *hidden layer* dengan tiga buah neuron dan satu buah neuron pada *output layer*. Arsitektur jaringan ditentukan dengan cara coba-coba dan melihat MSE pelatihan terbaik yang dihasilkan. Dalam kasus ini hasil penelitian menunjukan bahwa algoritma Levenberg Marquardt dengan regularisasi menjadi lebih baik.

Kata kunci*:* Prediksi, Jaringan Syaraf Tiruan, Levenberg Marquardt, Bayes Regularisasi, Curah Hujan

PENDAHULUAN

Salah satu faktor yang berpengaruh terhadap tipe iklim adalah curah hujan. Curah hujan adalah ketinggian air hujan yang terkumpul dalam tempat yang datar, tidak menguap, tidak meresap dan tidak mengalir. Secara umum unsur – unsur yang mempengaruhi dalam mendeteksi curah hujan adalah letak garis lintang, letak tinggi tempat (ketinggian), luas daratan, lokasi daerah, suhu udara, kelembapan, jumlah curah hujan, penguapan, kecepatan angin, dan sebagainya. Karena pada kasus ini mendeteksi hanya pada sutu daerah, maka faktor letak, lokasi dan luas daratan tidak digunakan.

Penelitian tentang peramalan curah hujan telah dilakukan pada penelitian sebelumnya oleh Warsito dan Sumiyati (2007) serta Ogwueleka dan Ogwueleka (2009) dengan menggunakan metode jaringan syaraf tiruan. Hasil penelitian menunjukkan bahwa metode jaringan syaraf tiruan mampu memberikan prediksi yang lebih handal. Penelitian lain yang menggunakan jaringan syaraf tiruan selain prediksi curah hujan juga sudah sering dilakukan. Banyak metode yang digunakan untuk menghasilkan mendapatkan hasil yang lebih akurat salah satunya dengan penambahan regularisasi untuk meningkatkan kinerja generalisasi yang dijelaskan dalam Foresee dan Hagan (1993). Secara umum teori tentang jaringan syaraf tiruan telah banyak dipaparkan Fausett (1994).

Penerepan Bayes regularisasi pada algoritma Levenberg Marquardt telah dilakukan dibeberapa penelitian oleh Eslamloueyan, Khademi, dan Mazinani (2010), Li dan Wang (2009), Kaminski dan Orlowska-Kowalska (2011), serta Aggarwal, dkk (2005). Bayes regularisasi juga akan digunakan dalam penelitian ini, untuk mengetahui bagaimana kinerja jaringan untuk meramalkan curah hujan.

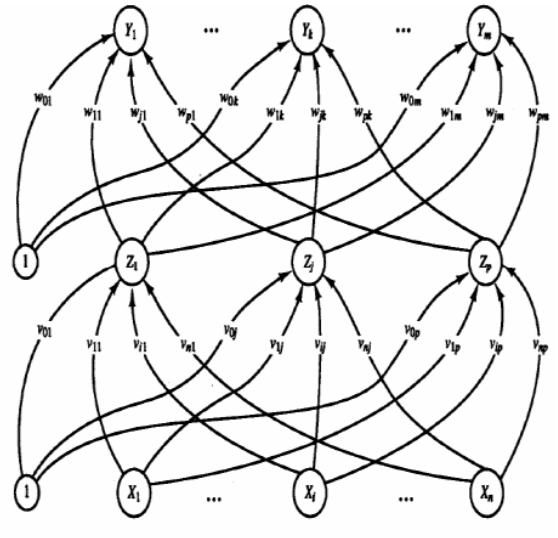
METODOLOGI PENELITIAN

a. Backpropagation

Pada dasarnya Neural Network (NN) merupakan suatu kumpulan elemen-elemen pemrosesan sederhana yang saling berhubungan, yang disebut neuron (unit, sel atau node). Setiap neuron dihubungkan dengan neuron lain dengan link komunikasi langsung melalui pola hubungan yang disebut arsitektur jaringan (Fausset, 1994). Tiap-tiap hubungan tersebut mempunyai bobot koneksi (*weight*) yang dilatih untuk mencapai respon yang diinginkan. Sehingga dengan pelatihan terhadap data berdasarkan bobot-bobot koneksi tersebut diharapkan memperoleh output yang diinginkan. Metode yang digunakan untuk menentukan bobot koneksi tersebut dinamakan algoritma pelatihan (*training algorithm*).

Backpropagation memiliki beberapa unit yang ada dalam satu atau lebih layar tersembunyi. Gambar 1 adalah arsitektur backpropagation dengan n buah masukan (ditambah sebuah bias), sebuah layar tersembunyi yang terdiri dari p unit

(ditambah sebuah bias), serta m buah unit keluaran.



Gambar 1. Arsitektur Backpropagation

Pelatihan backpropagation menggunakan metode pancarian titik minimum untuk mencari bobot dengan *error* minimum. Algoritma backpropagation menggunakan *error output* untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*) (Fausset, 1994). Pelatihan backpropagation meliputi 3 fase yaitu fase maju, fase mundur dan fase perubahan bobot.

Fase pertama ialah tahap maju (*feedforward*). Pada tahap ini seluruh proses awal inisialisasi bobot-bobot input dilakukan. Pada tahap ini juga ditentukan angka pembelajaran (α), nilai toleransi error dan jumlah epoch (siklus setiap pola pelatihan) yang diperlukan selama proses komputasi berlangsung. Setelah semua proses inisialisasi dilakukan, maka langkah selanjutnya ialah proses maju. Setiap unit masukan xi akan mengirimkan sinyal masukan ke lapisan tersembunyi. Setelah dihitung dengan menggunakan fungsi aktivasi maka keluarannya akan dikirimkan ke lapisan di atasnya, yaitu lapisan *output*. Setelah nilai keluaran (*yk*) diperoleh, maka dibandingkan dengan target keluaran sebenarnya (*tk*). Selisih *yk – tk* disebut dengan *error* (δk). Jika nilai *error* lebih kecil atau sama dengan dari nilai ambang maka proses terasi dihentikan, tetapi jika tidak maka nilai *error* tersebut digunakan untuk memodifikasi bobot-bobot untuk mengoreksi kesalahan yang terjadi.

Tahap kedua adalah tahap mundur atau backpropagation. Pada tahap ini, nilai *error* (δk) yang diperoleh pada di lapisan *output* digunakan untuk mengoreksi bobot-bobot yang ada pada lapisan tersembunyi yang berhubungan langsung dengan lapisan *output*. Setelah itu nilai *error* (δj) di setiap unit pada lapisan tersembunyi juga dihitung untuk mengoreksi bobot-bobot yang menghubungkan lapisan input dengan lapisan tersembunyi.

Tahap ketiga adalah tahap pengoreksian bobot. Setelah seluruh bobot pada lapisan input dan lapisan tersembunyi dimodifikasi sesuai dengan besar faktor *error*nya, maka ketiga fase ini diulang secara terus menerus sampai kondisi berhenti dipenuhi. Kondisi berhenti yang dimaksud adalah jika jumlah *epoch* yang ditetapkan tercapai

atau jika nilai *error* jaringan telah sama dengan atau lebih kecil dari nilai toleransi *error* yang ditetapkan sebelumnya. Pada tahap pelatihan, jaringan diharapkan dapat melatih seluruh data pelatihan yang diberikan untuk mendapatkan bobot akhir jaringan yang akan digunakan pada tahap pengujian. Struktur algoritma pelatihan Backpropagation antara lain :

1. Inisialisasi bobot-bobot Mentukan angka pembelajaran (α). Tentukan pula nilai toleransi *error* yang diinginkan dan set maksimal *epoch* jika ingin membatasi jumlah *epoch* yang digunakan.
2. Selama kondisi berhenti tidak terpenuhi, lakukan langkah ke-2 sampai langkah ke-9.
3. Untuk setiap pasangan pola pelatihan, lakukan langkah ke-3 sampai langkahke-8.

Tahap maju (Feedforward)

4. Tiap-tiap unit input (*xi, i = 1, 2, 3, ..., o*) menerima sinyal input dan meneruskan sinyal tersebut ke tiap-tiap unit pada lapisan tersembunyi.
5. Tiap-tiap unit di lapisan tersembunyi (*zj,j = 1, 2, 3, ..., p*) menjumlahkan sinyal-sinyal input yang berbobot, yaitu:

$$z\_net_j = v_{j0} + \sum_{i=1} x_i v_{ji} \qquad (1)$$

Fungsi aktivasi untuk menghitung sinyal outputnya, yaitu:

$$z_j = f(v_{j0} + \sum_{i=1} x_i v_{ji}) \qquad (2)$$

dan mengirimkan sinyal tersebut ke semua unit pada lapisan di atasnya (lapisan output).

6. Tiap-tiap unit di lapisan *output* (*yk, k = 1, 2, 3, ..., m*) menjumlahkan sinyal input yang berbobot, yaitu:

$$y\_net_k = w_{k0} + \sum_{j=1}^{p} z_j w_{kj} \qquad (3)$$

Fungsi aktivasi untuk menghitung sinyal outputnya, yaitu:

$$y_k = f(w_{k0} + \sum_{j=1}^{p} z_j w_{kj}) \qquad (4)$$

Tahap mundur(*Backpropagation*)

7. Tiap-tiap unit output yk menerima pola target tk untuk menghitung *error* (δk), yaitu:

$$\delta_k = (t_k - y_k) f'(y\_net_k)$$
$$= (t_k - y_k) y_k (1 - y_k) \qquad (5)$$

Kemudian menghitung nilai koreksi bobot yang nantinya digunakan untuk memperbaiki nilai bobot antara lapisan tersembunyi dan lapisan *output* (*wjk*), yaitu:

$$\Delta w_{jk} = \alpha \delta_k z_j \qquad (6)$$

menghitung juga koreksi bias yang digunakan untuk memperbaiki nilai bias antara lapisan tersembunyi dan lapisan *output* (*wk0*), yaitu:

$$\Delta w_{k0} = \alpha \delta_k \qquad (7)$$

8. Tiap-tiap unit pada lapisan tersembunyi (*zj, j = 1, 2, 3, ..., p*)

menjumlahkan sinyal-sinyal input dari lapisan output, yaitu:

$$\delta\_net_j = \sum_{k=1}^{n} \delta_k w_{jk} \quad (8)$$

mengalikan nilai ini dengan fungsi aktivasi untuk menghitung *error* pada lapisan tersembunyi *(δj*), yaitu:

$$\delta_j = \delta\_net_j z_j (1 - z_j)$$

$$= \delta\_net_j f'(z\_net_j) \quad (9)$$

Kemudian hitung koreksi bobot untuk memperbaiki nilai bobot antara lapisan *input* dan lapisan tersembunyi (*vji*), yaitu:

$$\Delta v_{ji} = \alpha \delta_j x_i \quad (10)$$

Kemudian menghitung koreksi bias untuk memperbaiki nilai bobot antara lapisan *input* dan lapisan tersembunyi (*vj0*), yaitu:

$$\Delta v_{j0} = \alpha \delta_j \quad (11)$$

Tahap pengoreksian bobot

9. Tiap-tiap unit keluaran (yk, k = 1, 2, 3,..., m) memperbaiki bobot dan bias, yaitu:

$$w_{kj}(baru) = w_{kj}(lama) + w_{kj},$$

$$(k = 1, 2, ..., m; j = 0, 1, ..., p) \quad (12)$$

Tiap-tiap unit tersembunyi memperbaiki bobot dan bias, yaitu:

$$v_{ji}(baru) = v_{ji}(lama) + v_{ji},$$

$$(j = 1, 2, ..., p; i = 0, 1, ..., n) \quad (13)$$

10. Tes kondisi berhenti.

b. Levenberg Marquardt

Algoritma Levenberg-marquardt merupakan pengembangan algoritma backpropagation standar. Pada algoritma backpropagation, proses update bobot dan bias menggunakan negative gradient descent secara langsung sedangkan. Algoritma Levenberg-Marquardt menggunakan pendekatan matrik *Hesian* (H) yang dapat dihitung dengan persamaan berikut:

$$H = J^T J \quad (14)$$

Sedangkan gradient dapat digitung dengan persamaan:

$$g = J^T e \quad (15)$$

Dalam hal ini J adalah sebuah matrik *jacobian* yang merupakan turunan pertama dari *error* jaringan terhadap bobot dan bias jaringan.

$$J(x) = \begin{bmatrix} \frac{\partial e_1(x)}{\partial x_1} & \frac{\partial e_1(x)}{\partial x_2} & \cdots & \frac{\partial e_1(x)}{\partial x_n} \\ \frac{\partial e_2(x)}{\partial x_1} & \frac{\partial e_2(x)}{\partial x_2} & \cdots & \frac{\partial e_2(x)}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_N(x)}{\partial x_1} & \frac{\partial e_N(x)}{\partial x_2} & \cdots & \frac{\partial e_N(x)}{\partial x_n} \end{bmatrix} \quad (16)$$

Perubahan bobot dapat dihitung dengan:

$$\Delta X = \left[J^T J + \mu I \right]^{-} J^T e \quad (17)$$

Sehingga perbaikan pembobot dapat ditentukan dengan persamaan:

$$X = X + \Delta X \quad (18)$$

X = fungsi bobot-bobot jaringan dan bias

X = [v_{11}, v_{12}, . . . , v_{ij}; v_{01}, v_{02}, . . . v_{0j} ; w_{11}, w_{12}, . . . w_{jk} ; w_{01}, w_{02} . . .w_{0k}]

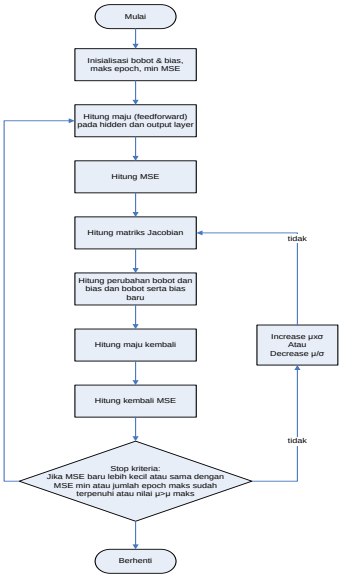
e adalah vector yang menyatakan semua error pada output jaringan.

$$e = \left[t_1 - y_1 t_2 - y_2 \cdots t_p - y_p \right]^T$$

μ = konstanta learning

I = matriks identitas

Secara garis besar jalannya proses Levenberg-Marquardt dari sistem dapat digambarkan pada Gambar 1.



Gambar 2. Algoritma Levenberg Marquardt

c. Regularisasi Bayes

Regularisasi berperan meningkatkan proses generalisasi dengan membatasi ukuran bobot suatu jaringan. Jika nilai bobot jaringan lebih kecil maka jaringan akan menanggapi dengan lebih halus. Dengan regularisasi, sebuah jaringan besar yang disederhanakan harus mampu mewakili fungsi yang sebenarnya.

Dalam algoritma Backpropagation klasik bertujuan untuk meminimalkan fungsi $F = Ed$, dimana:

$$E_d = \sum_{i=1}^{n} \left(t_i - a_i \right)^2 \quad (19)$$

Dalam hal ini n adalah jumlah input pada *training* set, t_i adalah nilai target pada data ke-i dan a_i adalah keluaran untuk data ke-i yang diperoleh sebagai respon jaringan saraf.

Metode regularisasi merubah kinerja kesalahan fungsi dengan menambahkan standar deviasi dari bobot dan bias, yaitu:

$$F = \beta E_d + \alpha E_w \quad (20)$$

α, β adalah parameter regularisasi, dan E_w didefinisikan sebagai :

$$E_w = \frac{1}{n} \sum_{i=1}^{n} \left(W_i \right)^2 \quad (21)$$

W_i adalah sebuah bobot atau batas ambang.

Dengan menggunakan persamaan (20) untuk mengubah fungsi kinerja *error* memungkinkan jaringan untuk mendapatkan bobot dan batas ambang terkecil, tetapi tidak bisa menentukan bobot dan batas ambang jaringan yang efektif.

Metode konvensional seringkali sulit untuk menentukan ukuran parameter biasa, Mackay (1992) mengusulkan jaringan yang dapat menyesuaikan ukuran parameter adaptif dengan menggunakan kerangka teori Bayesian, dan memungkinkan tercapainya kinerja yang

optimal. Rumus untuk menentukan parameter regularisasi adalah:

$$\begin{cases} \alpha = \dfrac{\gamma}{2E_w} \\ \beta = \dfrac{n-\gamma}{2E_D} \end{cases} \qquad (22)$$

Dimana $\gamma = n - 2\alpha tr\left(H\right)^{-1}$, H adalah matiks *Hessian* dari fungsi kinerja *F*.

d. Menggabungkan Metode Regularisasi Bayes dengan Algoritma Optimal Levenberg Marquardt

Algoritma Levenberg Marquardt dalam jaringan saraf jelas memiliki keuntungan untuk meningkatkan tingkat konvergensi dan mengurangi kesalahan dari proses pelatihan, dan algoritma Regularisasi Bayes dalam jaringan memiliki kelebihan untuk mengurangi kesalahan generalisasi. Tujuan penggunaan regularisasi Bayes adalah untuk menentukan parameter efektif yang digunakan oeh jaringan. Penerapan algoritma Levenberg Marquardt yang digabungkan dengan Regularisasi Bayes dalam penelitian ini bertujuan untuk meningkatkan kinerja jaringan. Langkah-langkah pelatihan dari Regularisasi Bayes adalah sebagai berikut:

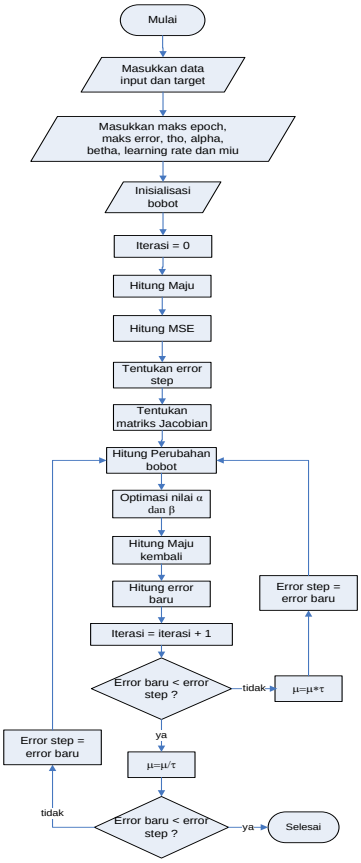
1. Inisialisasi nilai α, β serta bobot dan bias.
2. Minimasi fungsi *error* berdasarkan algoritma Levenberg Marquardt menurut persamaan (20).
3. Hitung pendekatan matriks *Hessian*

   $H = \beta J^T J + \alpha I$

   Dan menentukan:

   $\gamma = N - 2\alpha tr\left(H\right)^{-1}$
4. Gunakan formula (22) untuk memperbaharui nilai α, β.
5. Ulangi langkah 2-4 hingga jaringan mencapai konvergen.

Algoritma LMBR digambarkan menggunakan flowchart pada Gambar 3.



Gambar 3. Algoritma LMBR

e. Arsitektur Jaringan Syaraf Tiruan

Arsitektur jaringan syaraf tiruan dari sistem menggunakan arsitektur *multilayer network* atau jaringan

multilapis yang ditunjukkan pada Gambar 4. Terdapat 3 lapis layer , yaitu :

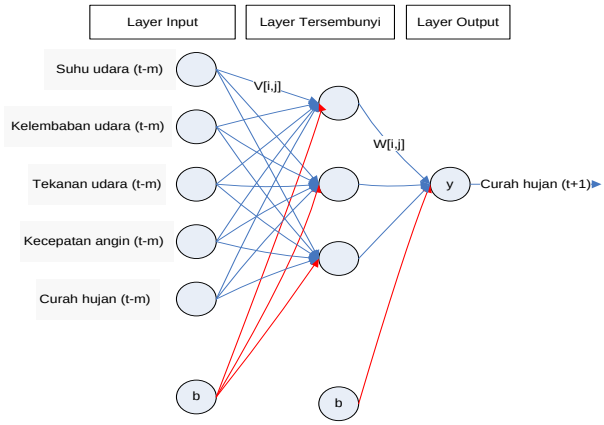
- input layer yang terdiri dari maksimum 30 unit neuron
- hidden layer yang terdiri dari 3 unit neuron
- output layer yang terdiri dari 1 unit neuron

Metode pembelajaran yang digunakan sistem adalah supervised learning. Sistem melakukan pembelajaran terhadap input dengan target yang sudah ditentukan untuk memperoleh nilai-nilai bobot dan bias yang optimum pada tiap unit neuron hidden layer dan output layer dengan error minimum. Fungsi error yang digunakan sistem adalah Mean Square Error.

Input data yang digunakan untuk pelatihan adalah suhu rata-rata, kelembaban udara, tekanan udara di atas stasiun pengamatan dan kecepatan angin rata-rata yang diambil dari data bulanan selama Januari 2008 hingga Desember 2009 yang diambil sebanyak 70%, dan 30% sisanya digunakan sebagai data pengujian. Data yang menjadi target adalah data curah hujan. Deskripsi variabel input dan output dapat dilihat pada Tabel 1.

Tabel 1. Deskripsi Variabel Input

Variabel	Suhu rata-rata (^{0}C)	Kelembaban udara (%)	Tekanan udara (mbar)	Kecepatan angin (km/jam)	Curah hujan (mm)
Mean	28.17	90.07	1018	3.41	17.36
Std. dev	0.62	1.41	61.1	0.54	6.03
Maks	29.17	93.05	1304.84	4.44	33.77
Min	27.09	87.49	1004.61	2.28	4.89



Gambar 4. Arsitektur Jaringan pelatihan

HASIL DAN PEMBAHASAN

Data yang digunakan dalam penelitian ini adalah data bulanan suhu rata-rata, kelembaban udara, tekanan udara di atas stasiun pengamatan dan kecepatan angin rata-rata serta curah hujan selama Januari 2008 hingga Desember 2009 yang diperoleh dari http://banyumilih.blogspot.com.
Deskripsi variabel input dan output dapat dilihat pada Tabel 2.

Penelitian dilakukan dengan menguji kinerja tiap algoritma berdasarkan panjang data runtun waktu yang dipilih dari 2-6. Dimana didapat hasil peramalan terbaik yaitu panjang data runtun waktu = 2. Hasil dapat dilihat pada Tabel 2.

Tabel 2. Kinerja LMBR untuk runtun data = 2 dengan *mu* = 0,001

lr	MSE	MSE	Peramalan	
	Pelatihan	Pengujian	MSE	MAPE
0,1	0.038019	0.0867951	0.027829	39.2274
0,2	0.039599	0.104284	0.031313	42.4383
0,3	0.036386	0.147972	0.032467	50.3175
0,4	0.050426	0.122697	0.037283	34.2272
0,5	0.036948	0.107433	0.024281	36.7668

Hasil ini akan dilakukan perbandingan dengan algoritma LM untuk menentukan kehandalan peramalan tiap algoritma seperti yang terlihat pada Tabel 3.

Tabel 3. Pengujian kehandalan peramalan

Algoritma	Pelatihan		Pengujian	Peramalan	
	MSE	epoch	MSE	MAPE	MSE
LMBR	0.0415	0	0.103	40.199	0.02806
LM	0.0468	28	0.118	30.7304	0.030792

SIMPULAN DAN SARAN

Dengan melakukan seluruh tahapan-tahapan penelitian berdasarkan hasil yang didapat selama analisis dan pengujian, maka penulis dapat menyimpulkan bahwa: Berdasarkan hasil pelatihan, algoritma LMBR menunjukkan nilai MSE terkecil dibandingkan algoritma LM pada rentang waktu = 2 yaitu sebesar 0.0280639. Hasil prediksi algoritma LMBR lebih mendekati data aktual untuk curah hujan bulan Januari 2010. Dari kedua algoritma pelatihan baik LMBR maupun LM menunjukkan nilai prediksi yang tidak jauh berbeda dan jangkauan nilai tidak terlalu jauh hal ini bisa disebabkan dari penyebaran pola data yang masih kurang dikenali oleh jaringan selama pelatihan karena jaringan terlalu cepat konvergen pada saat pelatihan. Untuk hasil yang lebih baik maka penggunaan metode ini dapat digabungkan dengan analisis data input yang lebih cocok sehingga data yang dimasukkan untuk pembelajaran merupakan data yang mewakili kejadian misalnya dengan analisis statistik.

DAFTAR RUJUKAN

Aggarwal, K.K., Singh, Y., Chandra, P. dan Puri, M., 2005, *Bayesian Regulation in Neural Network to Estimate Linse of Code Using Function Points*, Journal of Computer Sciences Research, 1, 505-509.

Eslamloueyan, R., Khademi, M.H. dan Mazinani, S., 2010, *Using a Multylayer Perceptron Networks for Thermal Conductivity Prediction of Aqueous Electrolyte Solutions*, Industrial & Engineering Chemistry Research, 50, 4050-4056.

Fausett, L., 1994, *Fundamentals of Neural Networks; rchitectures,algorithms and applications*, Prentice-Hall Inc., Englewoods Cliffs, New Jersey

Foresee, F.D. dan Hagan, M.T., 1993, *Gauss-Newton Approximation to Bayesian Learning*, IEEE International Conference on Neural Networks, 3, 1930-1935.

Kaminski, M. dan Orlowska-Kowalska, T., 2011, *Optimization of neural state variables estimators of two-mass drive system using the Bayesian regularization method*, Bulettin of the polish academy of sciences Technical Sciences, 59, 1, 33 – 38.

Li, X. dan Wang, D., 2009, *A Sensor Registration Method Using Improved Bayesian Regularization Algorithm*, International Joint Conference on Computational Sciences and Optimization, 194 – 199.

Mackay, D.,J.,C., 1992, *Bayesian Interpolation, Neural Computation,* vol. 4, pp. 415-447.

Ogwueleka, T.C. dan Ogwueleka, F.N., 2009, *Feed-forward Neural Networks for Precipitation and River Level Prediction*, Advances in Natural and Applied Sciences, 3, 350-356.

Warsito, B. dan Sumiyati, S., 2007, *Prediksi Curah Hujan Kota Semarang Dengan Feedforward Neural Network Menggunakan Algoritma Quasi Newton BFGS Dan Levenberg-Marquardt*, Jurnal PRESIPITASI, Semarang.