MENINGKATKAN KEMAMPUAN PENGENALAN POLA SINYAL DENGAN OPTIMALKAN *RULES* PADA *FUZZY NEURAL NETWORK*

Mukhtar Hanafi
Program Studi Teknik Informatika Universitas Muhammadiyah Magelang
Jl. Mayjend Bambang Soegeng Km.5 Mertoyudan, Magelang 56172 Telp (0293)-325554
E-mail: hanafi@ummgl.ac.id

Abstrak
Fuzzy Neural Network (FNN) merupakan suatu model yang dilatih menggunakan jaringan syaraf, namun struktur jaringannya diintepretasikan dengan sekelompok aturan-aturan (rules) fuzzy. Meskipun logika fuzzy dapat menerjemahkan pengetahuan pakar secara langsung melalui aturan-aturan dengan label-label linguistik, tapi umumnya membutuhkan waktu yang lama untuk mendisain dan menyesuaikan fungsi keanggotaan yang dapat memberikan definisi secara kuantitatif label-label linguistik ini. Hal ini menjadi sebih sulit lagi manakala rules yang ada sangat terbatas. Selanjutnya, bagaimana mengoptimalkan keterbatasan rules untuk meningkatkan kemampuan FNN dalam proses pembelajarannya. Tujuan dari penelitian ini adalah untuk mengetahui pengaruh penambahan fuzzy rules dalam proses pembelajaran pada FNN, serta peningkatan kemampuannya dalam mengenal pola sinyal yang dilatihkan. Pengujian dilakukan dengan menggunakan model jaringan FNN empat layer , algoritma pembelajaran back propagation dan tiga rule base fuzzy, yaitu dengan 9, 25 dan 49 rules. Pada penelitian ini, untuk mengembangkan rule base fuzzy dari 9 rules menjadi 25 dan 49 rules adalah dengan cara penambahan fungsi keanggotaan masukan error (e) dan perubahan error (de). Hasil pengujian menunjukkan, dengan penambahan jumlah rules, kemampuan FNN dalam mengenal pola sinyal menjadi lebih baik. Semakin banyak rules yang digunakan, kemampuannya dalam pengenali pola menjadi semakin baik akan tetapi proses belajar yang dilakukan juga semakin lama.

***Kata Kunci** : Fuzzy Neural Network, fuzzy rules, rule base*

1. PENDAHULUAN

Logika fuzzy dan jaringan syaraf tiruan merupakan model yang saling melengkapi dalam membangun suatu sistem cerdas. Jaringan syaraf memiliki struktur komputasi sederhana yang memiliki kemampuan yang baik ketika berhadapan dengan deretan data, sedangkan logika fuzzy memiliki kemampuan penalaran yang lebih tinggi, menggunakan informasi linguistik yang diperoleh dari pakar.

Perpaduan sistem fuzzy dan jaringan syaraf tiruan akan mengkombinasi komputasi sederhana yang bersifat paralel dan kemampuan belajar dari jaringan syaraf dengan representasi pengetahuan seperti manusia dan kemampuan *explanation* dari system fuzzy.

Fuzzy Neural Network (FNN) merupakan suatu model yang dilatih menggunakan jaringan syaraf, namun struktur jaringannya diintepretasikan dengan sekelompok aturan-aturan (*rules*) fuzzy (Fuller, 1995). Meskipun logika fuzzy dapat menerjemahkan pengetahuan pakar secara langsung melalui aturan-aturan dengan label-label linguistik, tapi umumnya membutuhkan waktu yang lama untuk mendisain dan menyesuaikan fungsi keanggotaan yang dapat memberikan definisi secara kuantitatif label-label linguistik ini. Hal ini menjadi sebih sulit lagi manakala *rules* yang ada sangat terbatas.

2. TINJAUAN PUSTAKA

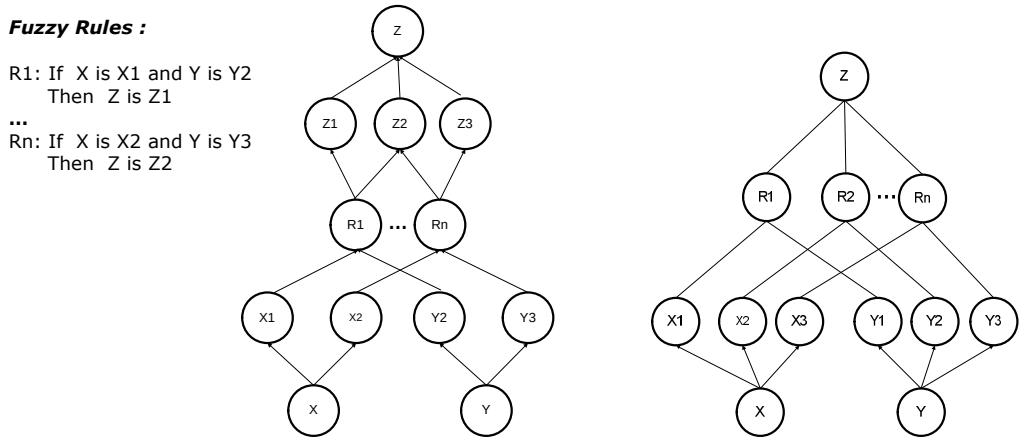
Penelitian yang berhubungan dengan penentuan dan penanganan aturan-aturan (*rules*) fuzzy dilakukan oleh Nazaruddin dan Yamakita (Nazaruddin dan Yamakita, 1999). Pada penelitian ini dikembangkan *adaptive network* yang berbasis aturan fuzzy untuk sistem suspensi. Aturan-aturan (*rules*) fuzzy yang digunakan dievaluasi menggunakan jaringan syaraf dengan proses pelatihan, sehingga diperoleh struktur jaringan yang paling optimal.

Penelitian lain yang berhubungan dengan penanganan aturan-aturan (*rules*) fuzzy menggunakan FNN, dilakukan oleh Lin, Roopaei dan Chen (Lin dkk, 2010). Dalam penelitian ini Lin dkk, menggunakan *Indirect Adaptive Interval Type-2* pada *Fuzzy Neural Network Controller* untuk menangani adanya aturan-aturan (*rules*) fuzzy yang tidak pasti yang dapat mempengaruhi kinerja pengontrolan suspensi. Ketidakpastian aturan-aturan fuzzy dapat muncul akibat tidak lengkapnya data pelatihan atau adanya gangguan (*noise*) selama proses pelatihan sehingga proses pengenalan pola sinyal control dapat terganggu. Dengan *Indirect Adaptive Interval Type-2*, munculnya ketidakpastian aturan fuzzy tersebut dapat diatasi

2.1 FUZZY NEURAL NETWORK

Fuzzy neural network (FNN) merupakan jaringan syaraf tiruan dengan basis aturan fuzzy (*fuzzy rule base*), sehingga memungkinkan adanya pemetaan antara anteseden dan konsekuen dalam bentuk IF-THEN ke dalam jaringan syaraf, dan untuk mengimplementasikan aturan fuzzy kedalam jaringan syaraf tiruan diperlukan level tambahan yang berupa level aturan-aturan fuzzy (*fuzzy rules level*). Banyaknya unit tersembunyi (*hidden units*) pada FNN ditentukan oleh banyaknya *rule*, sedangkan banyaknya lapisan tersebunyi (*hidden layer*) ditentukan oleh level dalam suatu hirarki aturan. Misalnya untuk suatu aturan: IF A AND B THEN X , secara hirarki aturan, terdiri dari tiga level yaitu input pada contoh ini adalah A dan B, penghubung (*conjunction*) yang berupa "AND" dan output yaitu X yang merupakan konklusi (Fu, 1994).

Seperti diilustasikan pada Gambar 1a, fuzzy neural network terdiri dari lima lapisan: lapisan input lapisan input fuzzy, lapisan penghubung (*conjuction*), lapisan output fuzzy dan lapisan output. Fungsi aktivasi dari unit input adalah nilai suatu variabel masukan yang sesuai dengan nilai yang diberikan. Nilai input dilewatkan dalam unit-unit himpunan fuzzy, yang akan merubah nilai kedalam suatu derajat keanggotaan sebagai fungsi aktifasi dari unit himpunan fuzzy (*fuzzy set unit*). Unit penghubung (*conjuction unit*) akan mengambil 'min' dari input (derajat keanggotaan) yang diterima dari masukan unit *fuzzy set* sebelumnya.



a). lima lapisan/*layer* b). empat lapisan/*layer*

Gambar 1. Struktur FNN Sebagai Jaringan Syaraf Berbasis Aturan

Model lebih sederhana dari FNN ditunjukkan pada Gambar 1b. FNN ini terdiri terdiri dari empat lapisan. Pada FNN ini tidak digunakan lapisan 'ouput_fuzzy' dan dari *fuzzy rule* yang digunakan fungsi keanggotaan keluaran Z1 dan Z2 adalah singletons yang menyertai bobot koneksi pada lapisan terakhir (Kasabov, 1998) . Untuk menentukan besarnya keluaran atau output z rumus yang digunakan sebagai berikut,

$$z = \sum_{i=1}^{m} \mu_i \omega_i \Big/ \sum_{i=1}^{m} \mu_i \quad \text{..... (1)}$$

dimana z adalah level aktivasi dari unit output, μ_i dan ω_i berturut-turut adalah fungsi keanggotaan dan jalur terbobot unit ke i pada lapisan antara output fuzzy ke output.

2.2 ALGORITMA *BACKPROPAGATION*

Algoritma backpropagation memiliki dasar matematis yang kuat dan obyektif. Algoritma ini mendapatkan bentuk persamaan dan nilai koefisien dalam formula dengan meminimalkan jumlah kuadrat galat error atau *Mean Squared Error* (MSE) melalui model yang dikembangkan (*training set*). Pelatihan suatu jaringan dengan algoritma *backpropagation* meliputi dua tahap, yaitu perambatan maju dan perambatan mundur. Untuk langkah selengkapnya adalah : (Fausset, 1994)

Tahap perambatan maju meliputi :

1. Tiap unit masukan (x_i ,i = *1,…, n*) menerima sinyal x_i dan menghantarkan sinyal ini ke semua unit lapisan di atasnya (unit tersembunyi),
2. Setiap unit tersembunyi (x_i ,i*=1,…,p*) dijumlahkan bobot sinyal masukannya dengan,

$$z\_in_j = v_{oj} + \sum_{i=1}^{n} x_i v_{ij} \quad \text{...} (2)$$

v_{oj} = bias pada unit tersembunyi *j* mengaplikasikan fungsi aktivasinya untuk menghitung sinyal keluarannya, $z_j = f\ (z\_in_j)$, dan mengirimkan sinyal ini keseluruh unit pada lapisan diatasnya (unit keluaran).

3. Tiap unit keluaran (*yk, k = 1,…, m*) dijumlahkan bobot sinyal masukannya dengan,

$$y\_in_k = w_{ok} + \sum_{i=1}^{n} z_i w_{ij} \quad \text{..} (3)$$

w_{ok} = bias pada unit keluaran *k* dan mengaplikasikan fungsi aktivasinya untuk menghitung sinyal keluarannya, $y_k = f\,(y\_in_k)$.

Sedangkan tahap perambatan mundur meliputi:

1. Tiap unit keluaran (y_k , *k = 1,…, m*) menerima pola target yang saling berhubungan pada masukan pola pelatihan, menghitung kesalahan informasinya dengan,

$$\delta_k = (t_k - y_k)\, f'(y\_in_k) \quad \text{...} (4)$$

menghitung koreksi bobotnya (digunakan untuk memperbaharui w_{jk} nantinya),

$$\Delta w_{jk} = \alpha \quad \delta_k z_j \quad \text{...} (5)$$

menghitung koreksi biasnya (digunakan untuk memperbaharui w_{ok} nantinya), dan mengirimkan δ_k ke unit-unit pada lapisan dibawahnya,

2. Setiap unit lapisan tersembunyi (z_j, *j = 1,…, p*) dijumlahkan hasil perubahan masukannya (dari unit-unit lapisan diatasnya) dengan,

$$\Delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad \text{..} (6)$$

dikalikan dengan turunan fungsi aktivasinya untuk menghitung informasi kesalahannya,

$$\delta_j = \delta\_in_j\, f'(z\_in_j) \quad \text{..} (7)$$

menghitung koreksi bobotnya (digunakan untuk memperbaharui v_{oj} nanti),

3. Tiap unit keluaran (yk, k = 1,…,m) mengupdate bias dan bobotnya (j = 0,…, p) dengan:

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad \text{..................................} (8)$$

Tiap unit lapisan tersembunyi (*zj, j=1,…,p*) mengupdate bias dan bobotnya (*I = 0,…,n*) dengan:

$$v_{jk}(baru) = v_{jk}(lama) + \Delta v_{jk} \quad \text{..................................} (9)$$

Untuk mempercepat proses pembelajaran, *backpropagation* standar dimodifikasi dengan menambahkan momentum. Penambahan momentum bertujuan untuk menghindari perubahan bobot yang terlalu besar akibat adanya data *outlier* atau data yang sangat berbeda dengan yang lain. Dengan penambahan momentum, bobot baru pada waktu ke (t+1) didasarkan atas bobot pada waktu t dan (t-1). Jika β adalah konstanta $(0 \le \beta \le 1)$ yang menyatakan parameter momentum, maka bobot baru dapat dihitung dengan persamaan: (Fausset, 1994)

$$w_{jk}(t+1) = w_{jk}(t) + \Delta w_{jk} + \beta(w_{jk}(t) - w_{jk}(t-1)) \quad \text{......................} (10)$$

dengan $\Delta w_{jk} = \alpha \delta_k z_j$, maka persamaan (3.13) dapat dituliskan kembali dengan

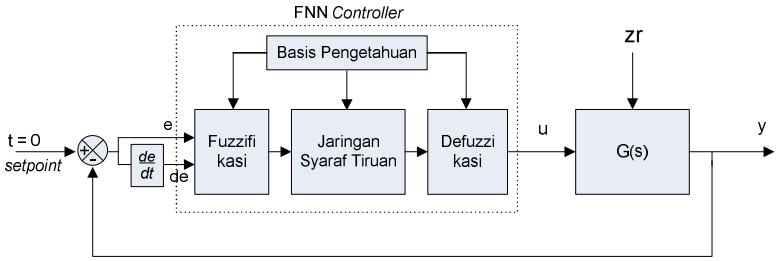
$$w_{jk}(t+1) = w_{jk}(t) + \alpha\delta_k z_j + \beta(w_{jk}(t) - w_{jk}(t-1)) \quad \text{....................} (11)$$

dimana $\boldsymbol{\alpha}$ adalah *learning factor* dan $\boldsymbol{\beta}$ adalah *momentum factor* .

3. METODE PENELITIAN

Pada penelitian ini semua model dan proses disimulasikan dengan menggunakan *software* Matlab. Pola sinyal yang digunakan adalah pola sinyal kontrol *u* yang dihasilkan oleh FNN *Controller* yang blok diagramnya seperti yang terlihat pada Gambar 2.

Sinyal masukan atau *input* yang diterima oleh FNN adalah sinyal *error* (*e*) dan perubahan *error* (*de*) yang terjadi pada *plant*/proses G(s). Sinyal tersebut kemudian difuzzifikasi kedalam bentuk fuzzy untuk selanjutnya diolah dengan jaringan syaraf, hasilnya didefuzzikasi kembali ke dalam bentuk *script* menjadi sinyal kontrol *u* sebagai *output* yang digunakan untuk mengendalikan *plant* G(s). Oleh sebab itu, ketepatan sinyal kontrol *u* sangat berpengaruh terhadap hasil akhir proses pengontrolan sistem secara keseluruhan.



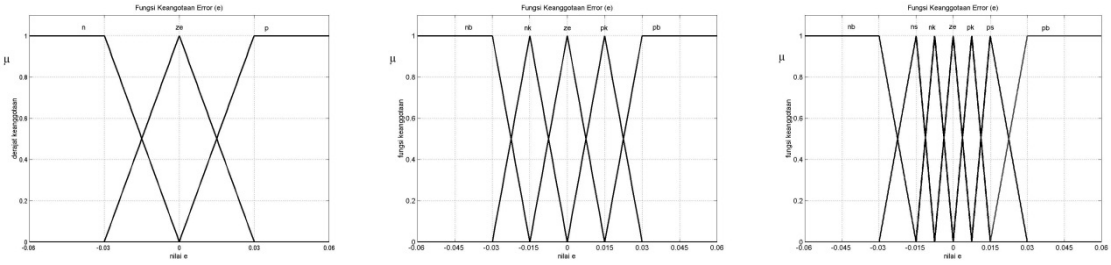
Gambar 2. Blok diagram FNN *Controller*

Semua proses pengambilan keputusan pada pengontrol FNN didukung oleh basis pengetahuan. Basis pengetahuan yang ada pada sistem ini terdiri dari tiga bagian, yaitu: informasi tentang definisi dan parameter fungsi-fungsi keanggotaan dari himpunan fuzzy, basis pengetahuan (*rule base*) dan bobot koneksi dari jaringan syaraf. Bobot koneksi dari jaringan syaraf disini diperoleh melalui proses pelatihan yang dilakukan secara terpisah atau *off-line* menggunakan algoritma *backpropagation*.

3.1 Menentukan Fungsi Keanggotaan dan *Rules*

Fungsi keanggotaan yang digunakan pada penelitian ini adalah fungsi keanggotaan segitiga untuk masukan *e* dan *de*. Untuk mendapatkan variasi fungsi keanggotaan masukan sebagai bahan pengujian, dilakukan modifikasi terhadap fungsi keanggotaan *input* tersebut.

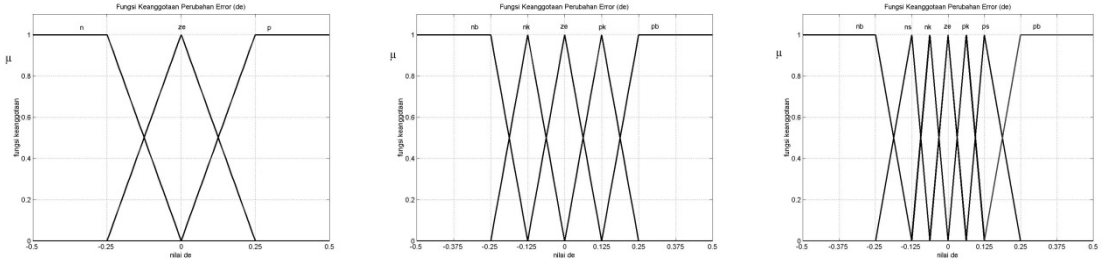
Modifikasi dilakukan dengan mengembangkan fungsi keanggotaan masukan *error* (*e*)**,** dari 3 fungsi keanggotaan yaitu: positif (***p***), negatif (***n***) dan zero (***ze***) seperti pada Gambar 3a, menjadi 5 fungsi keanggotaan yaitu: positif besar (***pb***), positif kecil (***pk***), zero (***ze***), negatif kecil (***nk***) dan negatif besar (***nb***) seperti yang terlihat pada Gambar 3b. Kemudian dari 5 fungsi keanggotaan dikembangkan lagi menjadi 7, yaitu positif besar (***pb***), positif sedang (***ps***), positif kecil (***pk***), zero (***ze***), negative besar (***nb***), negatif sedang (***ns***) dan negatif kecil (***nk***) seperti pada Gambar 3c.



a) tiga fungsi keanggotaan b) lima fungsi keanggotaan c) tujuh fungsi keanggotaan

Gambar 3. Fungsi Keanggotaan *error* (e)

Selain fungsi keanggotaan *error* (*e*), dengan cara yang sama modifikasi dilakukan juga pada fungsi keanggotaan perubahan *error* (*de*). Hasilnya seperti terlihat pada Gambar 5a,b dan c.



a) tiga fungsi keanggotaan b) lima fungsi keanggotaan c) tujuh fungsi keanggotaan

Gambar 4. Fungsi Keanggotaan perubahan *error* (*de*)

Selanjutnya, pada jaringan FNN dilapisan ketiga yang merupakan lapisan *conjuction*, akan digunakan sejumlah neuron untuk merealisasikan *rule base fuzzy*. Dengan menggunakan penghubung **AND** pada fungsi keanggotaan

fuzzy masukan *e* dan *de* dan dengan menggunakan jaringan FNN empat lapisan, dimana pada model jaringan ini tidak digunakan lapisan 'ouput_fuzzy' dan dari *fuzzy rule* yang digunakan fungsi keanggotaan keluaran adalah *singletons* yang menyertai bobot koneksi *w* pada lapisan terakhir, maka diperoleh matrik *rules* untuk masing-masing fungsi keanggotaan seperti terlihat pada Tabel 1 dan Tabel 2.

Tabel 1 adalah matrik 9 *rule* yang diperoleh dari hasil kombinasi dengan *cojunction* AND dari 3 fungsi keanggotaan *e* dan 3 fungsi keanggotaan *de*. Sedangkan matrik 25 *rules* pada Tabel 2 diperoleh dari kombinasi 5 fungsi keanggotaan *e* dan 5 fungsi keangotaan *de*. Demikian juga untuk 49 *rules* diperoleh dari kombinasi 7 fungsi keanggotaan *e* dan 7 *de*.

Table 1. Matrik 9 *rules*

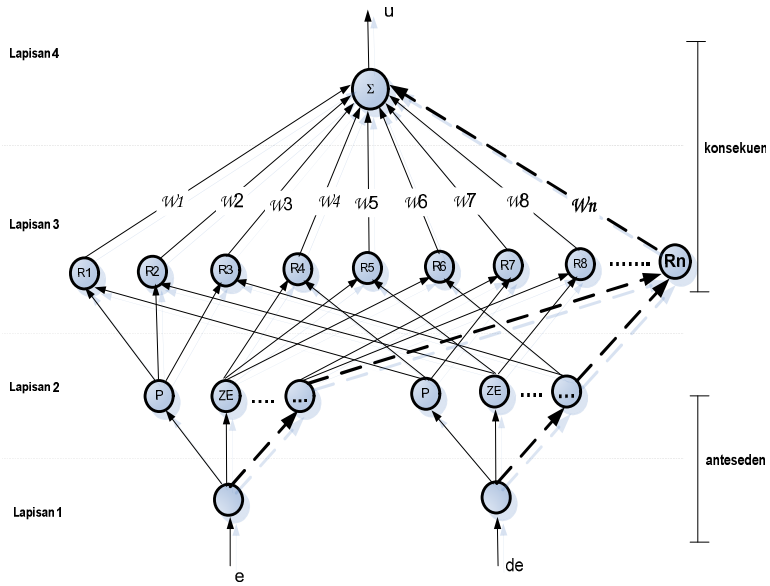
e / *de*	**n**	**ze**	**p**
n	*1.w*	*1.w*	*1.w*
ze	*1.w*	*1.w*	*1.w*
p	*1.w*	*1.w*	*1.w*

Table 2. Matrik 25 dan 49 *rules*

e / *de*	***nb***	***nk***	***ze***	***pk***	***pb***	***ns***	***ps***
ze	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*
nk	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*
np	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*
pk	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*
pb	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*
ns	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*
ps	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*	*1.w*

3.2 Struktur Jaringan FNN

Model struktur jaringan syaraf berbasis fuzzy atau FNN yang digunakan adalah model jaringan FNN 4 lapisan/layer, seperti ditunjukkan pada Gambar 5 berikut,



Gambar 5. Struktur Jaringan FNN

Lapisan pertama dari struktur FNN diatas merupakan lapisan input yang akan digunakan untuk melewatkan variabel masukan dan output dari lapisan pertama ini adalah: $y_{1,i}^{(1)} = e$ dan $y_{2,j}^{(1)} = de$. Kemudian pada lapisan kedua akan dilakukan proses fuzzikasi dengan memetakan nilai input yang dilewatkan (*e* dan *de*) kedalam suatu derajat keanggotaan fuzzy, output dari lapisan kedua ini adalah: $y_{1,i}^{(2)} = \mu(y_{1,i}^{(1)})$ dan $y_{2,j}^{(2)} = \mu(y_{1,j}^{(1)})$. Selanjutnya, pada lapisan ketiga yang merupakan lapisan *conjuction*, akan digunakan sejumlah neuron untuk merealisasikan *rule base fuzzy*. Dengan menggunakan penghubung **AND** pada fungsi keanggotaan fuzzy masukan e dan de, maka fungsi aktifasi yang digunakan pada lapisan ini adalah $\mu_{i,j} = \min(\mu(e_i), \mu(de_j))$ atau $\mu_{i,j} = \min(\mu(y_{1,i}^{(2)}), \mu(y_{2,j}^{(2)}))$, Sehingga keluarannya dapat dirumuskan dengan : $y_{i,j}^{(3)} = \mu_{i,j}$. Lapisan keempat

merupakan lapisan *output*, dimana pada lapisan ini akan dilakukan defuzzikasi atas hasil keluaran dari neuron-neuron pada lapisan sebelumnya. Pada lapisan ini dilakukan pembobotan dengan w_i pada setiap masukan. Output dari lapisan ini dihitung dengan persamaan:

$y^{(4)} = \frac{\sum_{i,j=1}^{n} y_{i,j}^{(3)}.w_i}{\sum_{i,j=1}^{n} y_{i,j}^{(3)}}$, dengan n adalah banyaknya *rules*. sehingga: $u = \frac{\sum_{i,j=1}^{n} \mu_{i,j}.w_i}{\sum_{i,j=1}^{n} \mu_{i,j}}$.

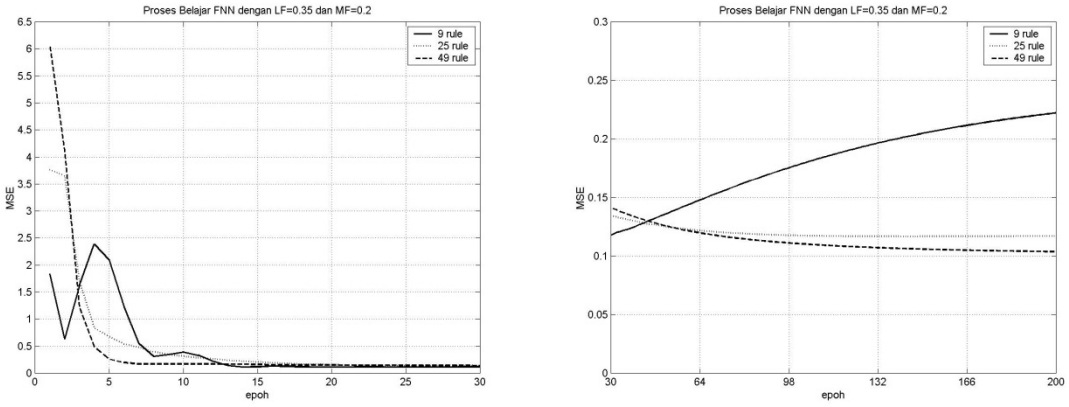
3.3 Pelatihan dan Pengujian

Proses pelatihan yang dilakukan pada FNN ini menggunakan algoritma *backpropagation*. Algoritma ini akan mengubah bobot-bobot koneksi w_i dengan metode perambatan mundur *error* keluaran dari lapisan *output*. Proses pelatihan ini hanya untuk melatih bobot koneksi lapisan ke 4, untuk lapisan yang lain bobot koneksi diset dengan nilai satu. Jika *Error* antara keluaran FNN yang sebenarnya ($u(t)$) dengan keluaran yang diinginkan ($ud(t)$), ditentukan dengan rumusan (Fu,1994): $E = \frac{1}{2}(ud(t) - u(t))^2$ dan untuk modifikasi bobot koneksi jaringan digunakan persamaan (Fausset,1994): $w_{jk}(t+1) = w_{jk}(t) + \alpha\delta_k Z_j + \beta(w_{jk}(t) - w_{jk}(t-1))$, dimana $\delta_k = (t_k - y_k) f'(y\_in_k)$. Maka modifikasi bobot koneksi pada lapisan ke 4 pada jaringan FNN yang dirancang ini dilakukan dengan persamaan: $w_{i4}(t+1) = w_{i4}(t) + \alpha(-\frac{\partial E}{\partial w_{i4}}) + \beta(w_{i4}(t) - w_{i4}(t-1))$, dimana α adalah *learning factor*, β adalah *momentum factor* dan $\frac{\partial E}{\partial w_{i4}} = \frac{\partial E}{\partial u}\frac{\partial u}{\partial w_{i4}} = (ud(t) - u(t))\frac{\mu_i}{\sum_{i=1}^{n}\mu_i}$

Pengujian dilakukan dengan menggunakan dua kombinasi nilai *learning factor* α dan *momentum factor* β yaitu : α = 0.35 , β = 0.2 dan α = 0.5 , β = 0.25. Dengan batasan iterasi atau epoh maksimalnya 200.

4. HASIL DAN PEMBAHASAN

Proses belajar FNN dilakukan dengan algoritma *backpropagation*. Grafik proses belajar dari FNN dengan menggunakan kombinasi *learning factor* α dan *momentum factor* β ditunjukkan oleh Gambar 6 dan Gambar 7. Pada Gambar 6 proses belajar dari FNN menggunakan α = 0.35 dan β = 0.2, sedangkan pada Gambar 7, nilai α yang digunakan 0.35 dan β = 0.2.

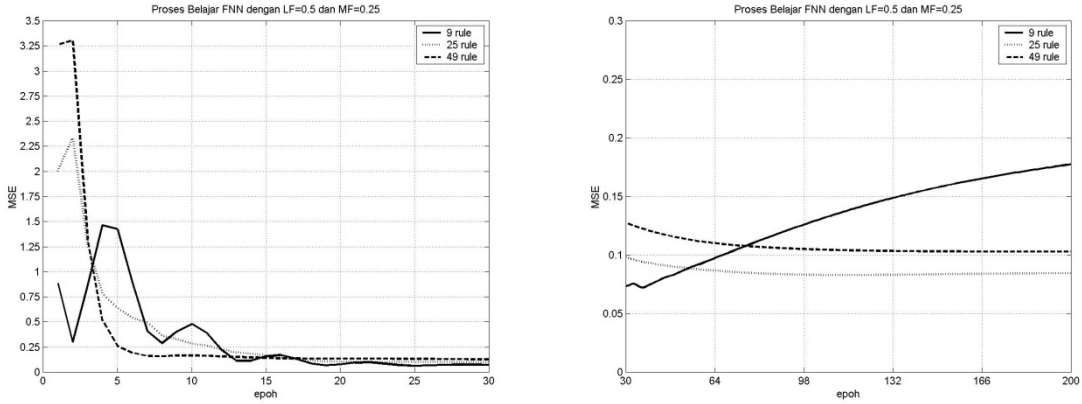


a) epoh 0 – 30 b) epoh 30 - 200

Gambar 6. Proses belajar FNN dengan α = 0.35 β = 0.2

Grafik pada Gambar 6a menunjukkan karakteristik proses belajar dari epoh 0 sampai epoh 30. Pada grafik tersebut ditunjukkan bahwa ketiga *rule base* yang digunakan yaitu 9, 25 dan 49 *rules* memberikan respon yang baik selama proses yaitu dengan semakin menurunnya nilai MSE disetiap kenaikan epohnya. Sementara itu untuk grafik pada Gambar 6b, dari epoh ke 30 sampai epoh ke 200 menunjukkan bahwa FNN dengan 9 *rule* sudah memberikan respon yang kurang baik dengan naiknya nilai MSE. Nilai terendah MSE untuk FNN dengan 9 rules adalah pada epoh ke-20 seperti yang terlihat pada Tabel 3. Setelah itu nilai MSE terus bertambah besar.

Walaupun memiliki nilai α dan β yang berbeda yaitu α = 0.5 dan β = 0.25, tetapi grafik pada Gambar 7 a dan b menunjukkan karakteristik yang hampir sama dengan grafik pada Gambar 6. Proses belajar pada FNN dengan 9 *rules* mengalami proses naik turun pada nilai MSEnya terutama pada epoh-epoh awal.



a) epoh 0 – 30 b) epoh 30 - 200

Gambar 7. Proses belajar FNN dengan α = 0.5 β = 0.25

Data proses belajar selengkapnya ditunjukkan pada Tabel 3. Untuk FNN dengan *rule* terbanyak yaitu 49 *rules* memiliki nilai MSE yang hampir sama pada nilai α dan β yang berbeda. Dan nilai MSE terendah diperoleh pada epoh terbesar. MSE terendah pada epoh ke-200 ini diperoleh bukan karena terkecil tapi karena telah mencapai batas iterasinya yaitu 200. Hal ini menunjukkan bahwa semakin banyak *rule* yang digunakan proses belajarnya juga akan semakin lama.

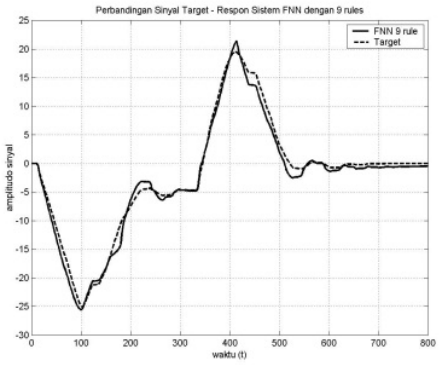
Tabel 3. Proses Belajar FNN

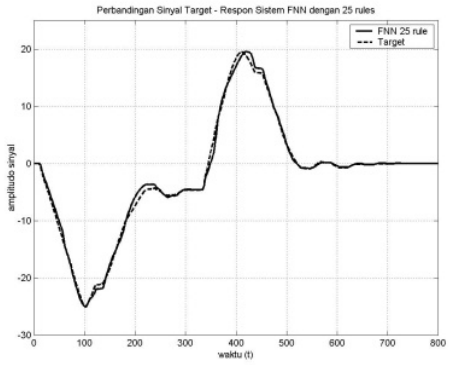
Nilai α dan β	Banyaknya rule	MSE terkecil	#Epoh MSE terkecil
α = 0.35 β = 0.2	9 rule	0.1114	20
	25 rule	0.1168	145
	49 rule	0.1038	200*
α = 0.5 β = 0.25	9 rule	0.0643	25
	25 rule	0.0829	115
	49 rule	0.1030	200*

Ket. * : batas iterasi

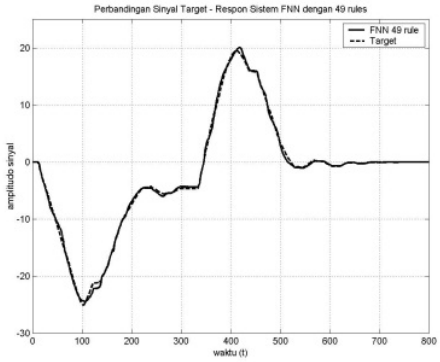
Setelah dilakukan proses belajar pada masing-masing FNN dengan jumlah rules yang berbeda serta nilai α dan β yang berbeda juga, kemudian dilakukan pengujian terhadap kemampunnya dalam mengenali pola sinyal yang dilatihkan. Hasilnya ditunjukkan pada Gambar 8 dan Gambar 9.

Gambar 8 a,b dan c menunjukkan hasil pengenalan pola sinyal untuk FNN dengan 9, 25 dan 49 *rules*. Grafik pada Gambar 8 tersebut menunjukkan perbandingan antara pola sinyal yang dihasilkan oleh FNN dengan sinyal target yang diharapkan. Pola sinyal yang dihasilkan oleh FNN dengan 49 *rules* adalah pola sinyal yang paling identik dengan pola sinyal target (Gambar 8c) dibandingkan dengan pola sinyal dari FNN dengan jumlah *rule* yang lebih sedikit.





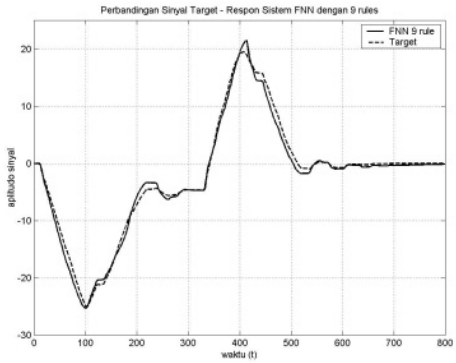
a) 9 *rules* b) 25 *rules*

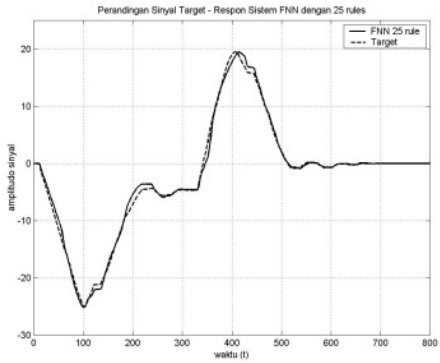


c) 49 *rules*

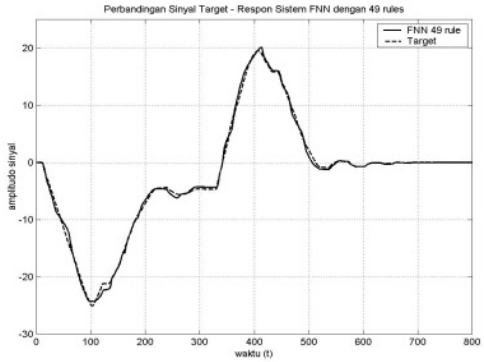
Gambar 8. Hasil pengenalan pola dengan $\alpha = 0.35$ $\beta = 0.2$

Kemampuan pengenalan pola sinyal dari FNN yang dilatih dengan nilai $\alpha = 0.5$ dan $\beta = 0.25$ ditunjukkan pada Gambar 9 a,b dan c.





a) 9 *rules* b) 25 *rules*



c) 49 *rules*

Gambar 9. Hasil pengenalan pola dengan $\alpha = 0.5$ $\beta = 0.25$

Grafik pada Gambar 9 diatas juga menunjukkan bahwa kemampuan mengenali pola terbaik dimiliki oleh FNN dengan jumlah *rule* terbanyak yaitu 49 (Gambar 9c).

5. KESIMPULAN

Kemampuan FNN yang merupakan jaringan syaraf tiruan dengan basis aturan fuzzy dapat ditingkatkan dengan mengoptimalkan *rules* yang digunakan. Mengoptimalkan *rules* FNN dengan 4 layer dapat dilakukan dengan penambahan jumlah *rules*. Dengan cara tersebut kemampuan FNN dalam pengenali pola sinyal menjadi semakin baik, akan tetapi semakin banyak *rule* yang digunakan proses belajar yang dilakukan juga semakin lama.

DAFTAR PUSTAKA

Fausset, L., 1994, *Fundamental of Neural Network*, Prentice-Hall.

Fu, L.M., 1994, *Neural Network in Computer Intelligence*, International Edition, McGraw Hill.

Fuller, R. , 1995, *Neural Fuzzy Systems*, Abo Akademi University, Abo, Turki.

Kasabov, N.K., 1998, *Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering*, Massachusetts Institute of Technology, England.

Lin, T.C., Roopaei, M., and Chen, M.C., 2010, Car Suspension Control By Indirect Adaptive Interval Type-2 Fuzzy Neural Network Control, *World Applied Sciences Journal 8* (5): 555-564.

Nazaruddin, Y.Y., Yamakita, M., 1999, Neuro-fuzzy based modeling of vehicle suspension system, *Control Application Proceedings of the 1999*, IEEE International Conference.