J. Pilar Sains 6 (2) Juli 2007

ISSN 1412-5595

PENGGUNAAN METODE JARINGAN NEURAL PERCEPTRON UNTUK MENGENAL POLA KARAKTER KAPITAL

Zaiful Bahri[1]
Dosen Program Studi Pendidikan Matematika FKIP Universitas Riau

Abstrak

Pemodelan dengan jaringan neural merupakan pembelajran dan penyesuaian dari suatu objek. Metode perceptron merupakan metode pembelajaran dengan pengawasan dalam sistem jaringan syaraf tiruan. Dalam mengenal pola beberapa huruf diperlukan beberapa neuron untuk membedakannya. Neuron-neuron akan menghasilkan nilai kombinasi yang digunakan untuk mengenal pola huruf-huruf tersebut. Dalam tulisan ini akan dibahas mengenal pola huruf kapital A, B, C, D dan E. MATLAB digunakan untuk mensimulasikan proses pembelajaran menggunakan metode jaringan neural perceptron.

Kata-kunci : Jaringan syaraf, Perceptron, Model, Pengenalan Pola, Pembelajaran

A. Pendahuluan

Pemodelan dengan jaringan neural merupakan pembelajaran dan penyesuaian suatu objek. Model-model jaringan neural dapat diklasifikasikan menurut beberapa kriteria, seperti metode pembelajaran. Menurut arsitekturnya, tipe input dapat berupa biner atau bipolar, begitu juga untuk output (biner atau bipolar). Dalam hal ini pemodelan dipandang dari kumpulan data input-outputnya. Pemodelan dengan perceptron menggambarkan suatu usaha untuk membangun kecerdasan dan sistem pembelajaran sendiri menggunakan komponen sederhana yang berasal dari model jaringan biologi yang diperkenalkan oleh McCuulloch dan Pitts (1943). Berikutnya Rosenbaltt (1950) merancang perceptron dengan menguraikan pemodelan kemampuan sistem pengenalan pola untuk sistem penggambaran biologi. Metode perceptron merupakan metode pembelajaran dalam sistem jaringan neural, sehingga jaringan yang dihasilkan harus mempunyai parameter yang dapat diatur dengan cara mengubah pembelajaran dengan pengawasan. Jaringan neural terdiri dari sejumlah neuron dan sejumlah masukan.

Dalam merancang jaringan neural yang perlu diperhatikan adalah banyaknya spesifikasi yang akan diidentifikasi dan jumlah neuron yang akan digunakan.

B. Metode

Persiapan menggunakan perceptron untuk aplikasi pengenalan pola digambarkan sebagai elemen matrik yang berisi 0 dan 1. Layer pertama pada perceptron dinyatakan dengan suatu kumpulan dari "detector tanda" sebagai isyarat input untuk mengetahui tanda khusus. Layer kedua (output) mengambil output dari tanda khusus dalam layer pertama dan mengklasifikasikan pola data yang diberikan. Pembelajaran dinyatakan dengan membuat aturan hubungan yang relevan (bobot w_i) dengan suatu nilai *threshold* (nilai ambang θ). Untuk persoalan dua kelas, layer output biasanya mempunyai hanya satu *node*(simpul). Untuk persolan kelas- n dengan $n \geq 3$, layer output biasanya mempunyai n simpul yang masing-masing berkorespondensi terhadap suatu kelas.

Setiap fungsi dalam layer 1 adalah fungsi tetap/konstan yang dihitung terdahulu, memetakan semua atau sebagian pola input ke dalam nilai biner $x_i \in \{0,1\}$ atau suatu nilai bipolar $x_i \in \{-1,1\}$. Satuan output adalah suatu unsur ambang linear dengan nilai ambang θ yang dinyatakan dengan :

$$P = f\left(\sum_{i=0}^{n} w_i\, x_i - \theta\right),$$

$$P = f\left(\sum_{i=0}^{n} w_i\, x_i + w_0\right), \qquad w_0 = \theta \qquad (1)$$

$$P = f\left(\sum_{i=0}^{n} w_i\, x_i\right), x_0 = 1$$

dimana w_i adalah suatu bobot yang dapat dimodifikasi berhubungan dengan kedatangan isyarat x_i dan $w_0 = -\theta$ adalah suatu bentuk penyederhanaan.

Dalam persamaan (1), fungsi f(.) adalah fungsi aktivasi dari perceptron dan hal ini khusus berlaku untuk suatu fungsi signum sgn(x) atau fungsi step(x) :

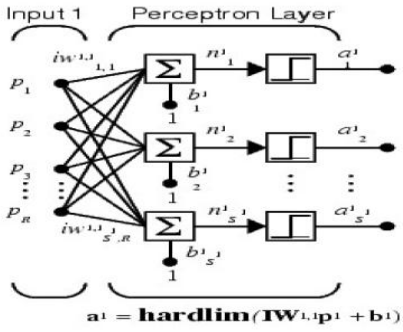
$$sgn(x) = \begin{cases} 1\ jika\ \ x > 0 \\ -1\ jika\ lainnya \end{cases} \qquad (2)$$

$$step(x) = \begin{cases} 1\ jika\ \ x > 0 \\ 0\ jika\ lainnya \end{cases} \qquad (3)$$

Prosedur pembelajaran hanya mengambil bobot yang berhubungan terhadap satuan output dalam layer terakhir. Jika hanya bobot pendahulu pada layer terakhir yang diubah, perceptron dalam gambar 1 biasanya diperlakukan sebagai perceptron layer tunggal. Dimulai dengan suatu himpunan bobot terhubung yang acak, algoritma pembelajaran dasar untuk perceptron layer tunggal diulangi mengikuti tahapan berikut sampai bobot konvergen.

1. Pilih suatu vektor input x dari kumpulan data pelatihan.
2. Jika perceptron memberikan suatu jawaban salah, modifikasi semua bobot terhubung w_i sesuai dengan $\Delta w_i = \eta t_i x_i$ dimana □□ adalah suatu target output dan □ adalah tingkat pembelajaran.

Aturan pembelajarang dapat dispesifikasikan dengan merubah nilai ambang sesuai dengan persamaan (1). Nilai untuk tingkat pembelajaran □ dapat menjadi konstan melalui pelatihan atau proporsional terhadap suatu konvergen lebih cepat tetapi dapat menyebabkan pembelajaran tidak stabil. kesalahan. Nilai □ adalah proporsional terhadap kesalahan biasanya membuat untuk



Gambar-1. Contoh arsitektur neural network dengan s neuron dan R masukan

Misalnya arsitektur neural network yang terdiri dari s neuron dan R masukan dapat dinyatakan seperti gambar 1.

C. Pembahasan

1. Perancangan Arsitektur Jaringan

Mengacu pada gambar 1, dan kebutuhan untuk mengidentifikasi lima huruf, diperlukan tiga neuron untuk membedakannya. Untuk nilai masukan telah diketahui bahwa setiap huruf direpresentasikan dengan 35 nilai yang berbeda. Karena itu, masukan untuk masalah ini adalah sebanyak 35. Ketiga neuron tersebut akan menghasilkan nilai yang kombinasi digunakan untuk mengidentifikasikan furuf-huruf tersebut. Kombinasi yang setara dengan sebuah huruf

Arsitektur jaringan terdiri dari tiga neuron yang mendapatkan masukan sebanyak 35, karena itu ada 3 x 35 nilai bobot yang diperlukan. Masing-masing 35 nilai bobot tersebut diberikan untuk setiap neuron.

Keluaran neuron ($\square_1$ $\square_2$ $\square_3$) akan dimasukkan ke fungsi hardlim (hardlimit), sebuah fungsi pada MATLAB yang akan mengonversikan nilai masukan menjadi nol (0) atau satu (1). Dari fungsi hardlim dapat diidentifikasi huruf-huruf A, B, C, D dan E. Kombinasi keluaran neuron untuk mengenal pola huruf A, B, C, D dan E didefinisikan sebagai 000 untuk A, 001 untuk B, 010 untuk C, 101 untuk D dan 111 untum E.

MATLAB digunakan untuk mensimulasikan proses pembelajaran menggunakan metode perceptron. Diperlukan pengetahuan mengenai instruksi-instruksi pada MATLAB yang berhubungan dengan neural network.

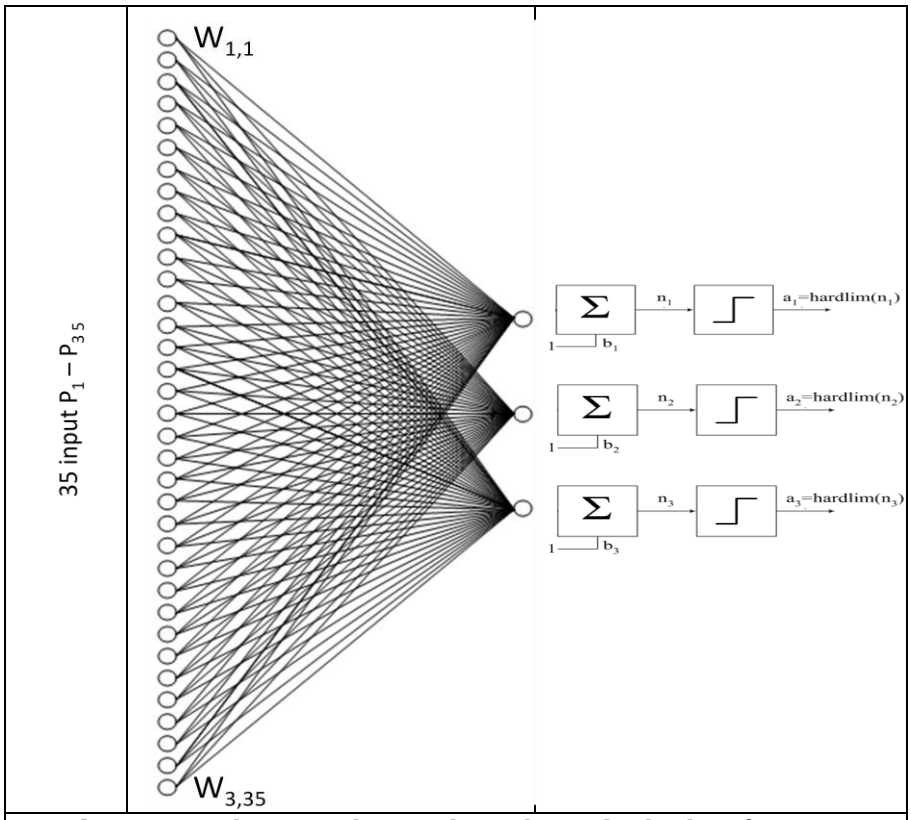
didefinisikan terlebih dahulu dan tidak ada satu aturan yang dapat mengikat tentang kombinasi keluaran neuron dengan sesuatu yang diidentifikasi.

Selanjutnya, arsitektur neural network untuk menyelesaikan persoalan ini adalah seperti pada gambar 2. Dari gambar 2 terlihat nilai $\square_{1,1}$ dan $\square_{3,30}$, nilai sebelum tanda koma menunjukan neuron yang akan menerima bobot w, sedangkan nilai setelah tanda koma menunjukan masukan nilai bobot. Jadi $\square_{3,18}$ adalah nilai bobot ke-21 yang dimasukan ke dalam neuron ke-3.

2. Instruksi dan Inisialisasi Variabel

Instruksi pertama adalah newp, instruksi untuk membuat jaringan neural baru dengan metode perceptron. Instruksi ini memiliki sintaks newp(PR,S), dengan PR adalah matriks berdimensi Rx2 yang nilai minimum dan maksimum R adalah banyaknya masukan ke setiap neuron. Untuk kasus ini, akan ada 35 x 2 matriks yang menjadi masukan bagi setiap neuron dengan nilai minimum dan maksimum masing-masing adalah nol dan satu. Sedangkan S adalah banyaknya neuron, karena banyaknya neuron sama dengan tiga, maka S=3.

neuron harus menghasilkan nilai seperti yang telah didefinisikan sebelumnya. Misalnya, untuk masukan pA, neuron pertama dan kedua harus 0 dan neuron ketiga harus 1. Hal ini akan mempermudah simulasi.



Gambar 2. Arsitektur neural network untuk mendeteksi huruf A, B, C, D dan E

Dari gambar 2 dapat dilihat bahwa arsitektur jaringan tersebut dapat berdiri sendiri. Untuk setiap masukan, hubungan neuron, nilai masukan dan target yang diinginkan dapat dilihat pada tabel 1.

Neuron	Masukan	Target
1	pA, pB, pC pD, pE	0 1
2	pA, pB, pD pC, pE	0 1
3	pA, pC pB, pD, pE	0 1

Tabel 1. Korespondensi neuron, nilai masukan dan target

Untuk inisialisasi jaringan bagi ketiga neuron tersebut pada MATLAB, digunakan instruksi newp, dimana jaringan yang terbentuk untuk masing-masing net1, net2 dan net3, begitu juga untuk masukan masing-masing huruf.

```
net1=newp([0 1;0 1;0 1;0 1;0 1; 0 1;0 1;
           0 1;0 1;0 1; 0 1;0 1;0 1;0 1;
           0 1; 0 1;0 1;0 1;0 1;0 1; 0 1;
           0 1;0 1;0 1;0 1; 0 1;0 1;0 1;
           0 1;0 1; 0 1;0 1;0 1;0 1;0 1],1);      untuk neuron 1
net2=newp([0 1;0 1;0 1;0 1;0 1; 0 1;0 1;
           0 1;0 1;0 1; 0 1;0 1; 0 1;0 1;
           0 1; 0 1;0 1;0 1; 0 1;0 1; 0 1;
           0 1;0 1;0 1; 0 1; 0 1;0 1;0 1;
           0 1;0 1;0 1;0 1;0 1;0 1;0 1],1);      untuk neuron 2
net3=newp([0 1;0 1;0 1;0 1;0 1; 0 1;0 1;
```

0 1;0 1;0 1; 0 1;0 1;0 1;0 1;
0 1; 0 1;0 1;0 1;0 1;0 1; 0 1;
0 1;0 1;0 1;0 1; 0 1;0 1;0 1;
0 1;0 1; 0 1;0 1;0 1;0 1;0 1],1); ⟶ untuk neuron 3

Sedangkan data masukan untuk masing-masing huruf adalah seperti berikut ini :

pA=[0;0;1;0;0; 1;1;0;1;1; 1;1;0;1;1; 1;1;0;1;1; 1;1;0;1;1; 1;1;1;1;1; 1;1;0;1;1; 1;1;0;1;1];
pB=[1;1;1;1;0; 1;1;1;1;1; 1;1;0;0;1; 1;1;1;1;0; 1;1;0;0;1; 1;1;1;1;1; 1;1;1;1;0];
pC=[0;1;1;1;0; 1;1;1;1;1; 1;1;0;0;1; 1;1;0;0;0; 1;1;0;0;1; 1;1;1;1;1; 0;1;1;1;0];
pD=[1;1;1;1;0; 1;1;1;1;1; 1;1;0;1;1; 1;1;0;1;1; 1;1;0;1;1; 1;1;1;1;1; 1;1;1;1;0];
pE=[1;1;1;1;1; 1;1;1;1;1; 1;1;0;0;0; 1;1;1;1;1; 1;1;0;0;0; 1;1;1;1;1; 1;1;1;1;1];

Jika data masukan tersebut disatukan menjadi aliran data untuk huruf A, B, C, D dan E, maka data tersebut dapat dideklarasikan dalam MATLAB sebagai berikut : pTOTAL=[pA pB pC pD pE]. Selajutnya target untuk masing masing network adalah sebagai berikut :

t1=[0 0 0 1 1] ⟶ untuk neuron 1
t2=[0 0 1 0 1] ⟶ untuk neuron 2
t3=[0 1 0 1 1] ⟶ untuk neuron 3

Instruksi selanjutnya yang diperlukan adalah train. Instruksi ini memilik sintaks [net,tr,Y,E,Pf,Af]=train(net,P,T,Pi,Ai,VV,TV). Dalam tulisan ini, *return value* dari instruksi adalah Y sebagai output dari network.

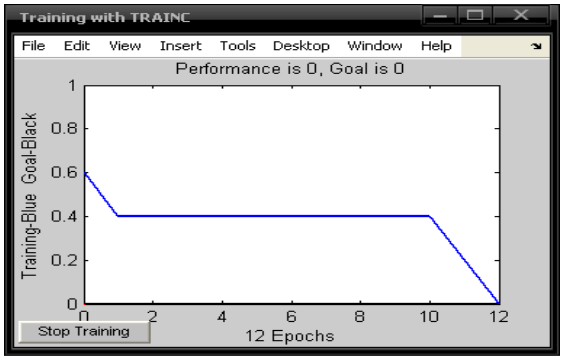
Untuk mensimulasikan neuron ini, dapat digunakan instruksi *sim*, dengan sintaks sebagi berikut [Y,Pf,Af,E,perf]=sim(net,P,Pi,Ai,T). Dalam hal ini, *return value* yang akan digunakan adalah *net*, sedangkan *passing value*nya adalah P(nilai masukan) dan T(target). Sedangkan *passing value* yang akan digunakan *net*, P(nilai masukan) dan T (target). Dengan instruksi *sim* tidak lagi diperlukan instruksi *hardlim* karena sudah disimulasikan dalam fungsi *sim.* Langkah awal adalah melakukan simulasi dengan mendeklarasikan semua variabel pada *command window* MATLAB. Variabel tersebut adalah ketiga neuron, data masukan, dan data target setiap neuron.

3. Skenario Simulasi Jaringan dan Hasil

Setelah arsitektur jaringan serta variable-variabel diketahui, simulasi pada MATLAB dapat dilakukan. Hasil inisialisasi jaringan dan variabel dapat dilihat seperti gambar 3.

s

Gambar 3. Inisialisasi jaringan dan variabel- variabel neuron pada Matlab

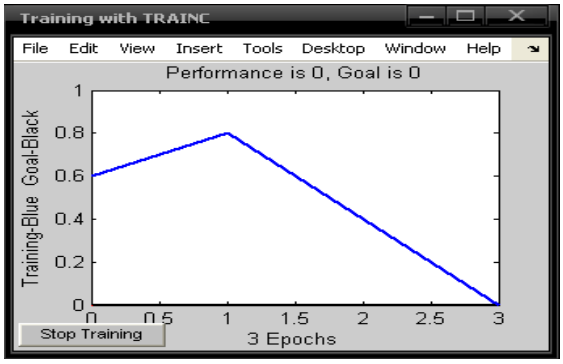


Gambar 4. Konvergensi keluaran neuron pertama setelah epoch ke 12

Nilai bobot yang dihasilkan dari pembelajaran neuron pertama, yang disimpan dalam variabel w1, adalah sebagai berikut :

w1 ={ 4 4 -1 4 3 -1 -1 4 -1 -1 -1 -1 0 2 -4 -1 -1 -3 -1 5 -1
-1 -5 2 -4 -1 -1 4 -1 -1 -1 -1 4 -1 -2}

sedangkan nilai bias yang disimpan dalam variabel b1 adalah -1. Proses simulasi selanjutnya adalah untuk neuron kedua dan ketiga dengan instruksi yang sama dengan neuron pertama dengan variabel disesuaikan, dan diperoleh :



Gambar 5. Konvergensi keluaran neuron kedua setelah epoch ke 3

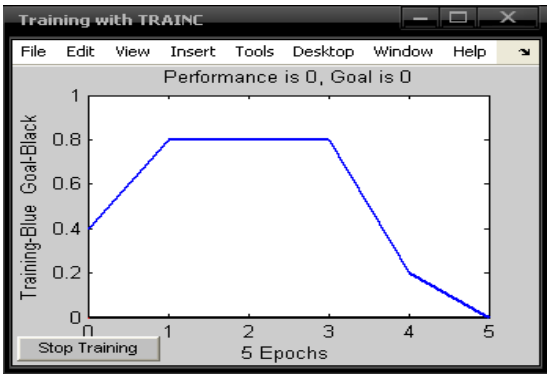
Perbedaan nilai keluaran neuron kedua dengan nilai keluaran yang diharapkan sejak pembelajaran dimulai, dapat digambarkan sebagai berikut (gambar 5). Nilai bobot yang dihasilkan dari pembelajaran neuron kedua, yang disimpan pada w2 adalah :

w2 ={ -1 1 0 1 1 0 0 1 0 0 0 0 0 -2 -1 0 0 0 -2 -1 0 0 -1 -2 -1 0 0 1 0 0 -2 0 1 0 0}

sedangkan nilai bias yang disimpan dalam variabel b2 adalah 0. Nilai bobot yang dihasilkan dari pembelajaran neuron ketiga (gambar 6), yang disimpan dalam variabel w3, adalah sebagai berikut :

w3 ={ 6 3 -1 3 0 -1 -1 3 -1 -1 -1 -1 0 -1 -1 -1 -1 3 2 -1 -1 -1 -4 -1 -1 -1 -1 3 -1 -1 2 -1 3 -1 -4}

sedangkan nilai bias yang disimpan dalam variabel b3 adalah -1. Setelah semua neuron konvergen, jaringan dapat digunakan untuk mengidentifikasi huruf A, B, C, D dan E. Untuk simulasi huruf A digunakan instruksi: aTOTAL = [sim(net1,pA) sim(net2,pA) sim(net3,pA)].



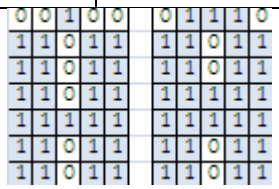
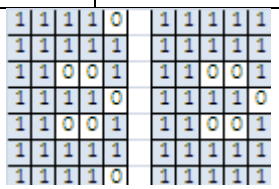
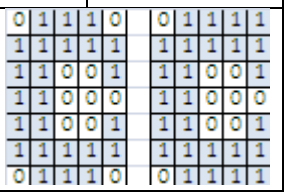
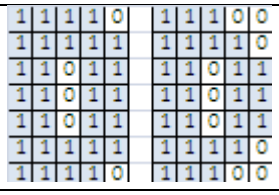
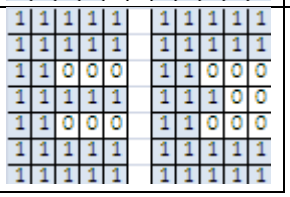
Gambar 6. Konvergensi keluaran neuron ketiga setelah epoch ke 5

Instruksi untuk data masukan huruf lain mengikuti instruksi di atas. Hasil simulasi keluaran ketiga neuron adalah [0 0 0], [0 0 1], [0 1 0], [1 0 1], [1 1 1] masing-masing untuk huruf A, B, C, D, dan E. Hasil ini sama dengan representasi huruf-huruf tersebut.

Untuk neuron pertama dengan data masukan pTOTAL dihasilkan keluaran yang kemudian disimpan dalam variabel Y1. Variabel Y1 tersebut adalah Y1=[0 0 0 1 1]. Terlihat bahwa nilai Y1 sama dengan t1 (target pertama), sehingga error untuk keluaran pertama yaitu e1=Y1-t1, maka diperoleh e1=[0 0 0 0 0]. Untuk neuron kedua dan ketiga digunakan variabel Y2 dan Y3 dan diperoleh berturut - turut Y2=[0 0 1 0 1] dan Y3=[0 1 0 1 1], yang sama dengan target t2 dan t3, yang menghasilkan e2=[0 0 0 0 0] dan e3=[0 0 0 0 0]

4. Pengujian

Untuk menguji kinerja neuron ini diperlukan masukan yang sudah dimodifikasi sedemikian rupa berbeda dari data masukan awal, namun polanya masih kelihatan. Berikut adalah data awal dan data yang telah dimodifikasi (Tabel 2) untuk masukan masing huruf A,B,C, D dan E. Dengan menggunakan variabel target yang sama dengan saat pembelajaran, dapat disimulasikan apakah arsitektur jaringan yang dibuat dapat mengidentifikasi data masukan yang telah dimodifikasi. Berhasil tidaknya proses identifikasi ditentukan dengan kesamaan keluaran jaringan neuron dengan target yang telah ditetapkan. Selanjutnya, dilakukan langkah-langkah yang sama seperti pada saat pembelajaran. Perbedaannya adalah pada saat pembelajaran ini tidak diperlukan lagi instruksi train. Berikut data masukan untuk masing-masing huruf yang sudah dimodifikas

Huruf	Awal	Modifikasi	Huruf	Awal	Modifikasi	Huruf	Awal	Modifikasi
A			B			C		
D			E					

Tabel 2. Korespondensi Data awal sat pelatihan dan data modifikasi untuk pengujian

```
pA1=[0;1;1;1;0;  1;1;0;1;1;  1;1;0;1;1; 1;1;0;1;1;  1;1;1;1;1; 1;1;0;1;1; 1;1;0;1;1];
pB1=[1;1;1;1;1;  1;1;1;1;1;  1;1;0;0;1; 1;1;1;1;0;  1;1;0;0;1; 1;1;1;1;1; 1;1;1;1;1];
pC1=[0;1;1;1;1;  1;1;1;1;1;  1;1;0;0;1; 1;1;0;0;0;  1;1;0;0;1;  1;1;1;1;1; 0;1;1;1;1];
pD1=[1;1;1;0;0;  1;1;1;1;0;  1;1;0;1;1; 1;1;0;1;1;  1;1;0;1;1;  1;1;1;1;0; 1;1;1;0;0];
pE1=[1;1;1;1;1;  1;1;1;1;1;  1;1;0;0;0; 1;1;1;0;0;  1;1;0;0;0;  1;1;1;1;1; 1;1;1;1;1];
```

Setelah data masukan yang baru dideklarasikan di MATLAB, keluaran dapat dilakukan dengan instruksi simulasi *sim* untuk masing-masing huruf. Instruksinya adalah:

aTOTAL=[sim(net1,pA1) sim(net2,pA1) sim(net3,pA1)]

Instruksi di atas dapat digunakan untuk masing-masing huruf dengan variabel yang disesuaikan. Terlihat, pada saat data masukan adalah adata uji untuk huruf A, jaringan neuron menghasilkan keluaran [0 0 0], sama seperti representasi untuk huruf A. Begitu juga untuk huruf B, C, D dan E yang masing-masing menghasilkan [0 0 1], [0 1 0], [1 0 1] dan [1 1 1].

Tingkat ketidakteraturan data uji sesungguhnya masih sangat sederhana, sehingga kemampuan jaringan neuron masih bisa mengenal pola masukan sebagai huruf yang diidentifikasi. Namun jika data uji diubah dengan dengan data yang polanya agak berbeda seperti pola yang pertama:

```
0 1 1 0 0          1 1 1 1 1
1 1 1 1 0          1 1 1 1 1
1 1 0 0 0          1 1 0 1 1
1 1 0 0 0    dan   1 1 0 1 1
1 1 0 0 0          1 1 0 1 1
1 1 1 1 0          1 1 1 1 1
0 1 1 0 0          1 1 1 1 1
```

Hasil simulasi menunjukkan keluaran [1 1 0] yang bukan representasi dari kelima huruf, walaupun polanya masih kelihatan sebagai huruf C, begitu juga dengan pola yang kedua, hasil simulasi menunjukkan keluaran [1 0 0] yang bukan representasi dari kelima huruf, walaupun polanya masih kelihatan sebagai huruf D.

D. Kesimpulan

Jaringan neural terdiri dari sejumlah neuron dan sejumlah masukan. Dalam merancang jaringan neural yang perlu diperhatikan adalah banyaknya spesifikasi yang akan diidentifikasi.

Dari beberapa data uji yang dilakukan dapat diambil kesimpulan bahwa :

1. Jika pola masukan masih mirip dengan pola awal maka arsitektur jaringan neuron yang dibanguan masih mampu mengidentifikasi pola masukan tersebut.
2. Jika pola masukan tidak mirip dengan pola awal seperti pola huruf D, jaringan neural tidak mengenal pola masukan tersebut, karena polanya mirip dengan huruf O, sehingga keluarannya tidak seperti yang diharapkan.

E. Daftar Pustaka.

[1] Diah Puspitaningrum, Pengantar Jaringan Syaraf Tiruan, Andi Yokyakarta, 2006.

[2] Jong Jek Siang, Jaringan Syaraf Tiruan dan Pemogramannya Menggunakan MATLAB, Andi Yoyakarta, 2009.

[3] Mike Susmikanti dan Arya Adhiyaksa. Identifikasi Huruf Menggunakan Metode Pembelajaran Perceptron Dalam Jaringan Neural. Prosiding Semiloka Teknologi Simulasi dan Komputasi serta Aplikasi, 2005.

[4] Pandya, A. S., Pattern Reconition with Neural Networks in C++, CRC Press 1994.

[5]. Sergios Theodoridis and Konstantinos Koutroumbas, Pattern Recognition, Third Edition, Elsevier (USA), 2006.