|  |
| --- |
| **8** ***ARTIFICIAL NEURAL NETWORK*** |
| JUMLAH PERTEMUAN : 1 PERTEMUANTUJUAN INSTRUKSIONAL KHUSUS :1. Memahami konsep sistem jaringan syaraf tiruan dan pemodelannya.
2. Memahami fungsi aktivasi
3. Memahami metode pelatihan dan model jaringan syaraf tiruan.
 |

**Materi :**

**8.1 Pendahuluan**

 Jaringan syaraf tiruan adalah salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia. Istilah buatan ini diimplementasikan dengan menggunakan program computer yang mampu menyelesaikan sejumlah proses perhitungan selama proses pembelajaran.

Otak manusia berisi berjuta-juta sel sayraf yang bertugas untuk memproses informasi. Setiap sel syaraf (neuron) akan memiliki satu inti sel, inti sel ini yang akan bertugas untuk melakukan pemrosesan informasi.

Tahun 1940-an, para ilmuwan menemukan bahwa psikologi otak sama dengan mode pemrosesan yang dilakukan oleh komputer. Tahun 1943, McCulloch dan Pitts merancang model formal yang pertama kali sebagai perhitungan dasar neuron. Tahun 1954, Farley dan Clark mensetup model-model untuk relasi adaptif stimulus-respon dalam jaringan random. Tahun 1958, Rosenblatt mengembangkan konsep dasar tentang perception untuk klasifikasi untuk klasifikasi pola. Tahun 1974, Werbos memperkenalkan algoritma backpropagation. Tahun 1975, Little dan Shaw menggambarkan jaringan syaraf dengan probabilistik

**8.2 Komponen dan Arsitektur**

Pembuatan struktur jaringan syaraf tiruan diilhami oleh struktur jaringan biologi, khususnya jaringan otak manusia.



Gambar 8.1 Struktur dasar jaringan syaraf tiruan dan struktur sederhana sebuah neuron

Jaringan syaraf terdiri dari beberapa komponen berikut :

1. Neuron, sel syaraf yang akan mentransformasikan informasi yang diterima melalui sambungan keluarnya menuju neuron-neuron lain.
2. Hubungan antar neuron dikenal dengan nama bobot.
3. Neuron Layers, neuron-neuron akan dikumpulkan dalam lapisan-lapisan (layer).
4. Informasi yang diberikan pada jaringan syaraf akan dirambatkan dari lapisan ke lapisan, mulai dari lapisan input sampai ke lapisan output melalui lapisan lainnya(hidden layer), tergantung pada algoritma pembelajarannya, bias jadi informasi tersebut akan dirambatkan secara mundur pada jaringan.



Gambar 8.2 Model tiruan sebuah neuron

Faktor terpenting untuk menentukan kelakuan suatu neuron adalah fungsi aktivasi dan pola bobotnya. Ada beberapa arsitektur jaringan syaraf, antara lain :

1. Jaringan dengan lapisan tunggal (single layer net) :
* Memiliki satu lapisan dengan bobot-bobot tehubung
* Menerima input kemudian langsung mengolahnya menjasi output tanpa harus melewati lapisan tersembunyi.



1. Jaringan dengan banyak lapisan (multi layer net) :
* Memiliki 1 atau lebih lapisan yang terletak diantara lapisan input dan lapisan output.
* Ada lapisan yang berbobot yang terletak antara 2 lapisan yang bersebelahan.



1. Jaringan dengan lapisan kompetitif (competitive layer net) :
* Hubungan antar neuron pada lapisan kompetitif tidak diperlihatkan pada diagram arsitektur



**8.3 Fungsi Aktivasi**

Mengaktifkan syaraf tiruan berarti mengaktifkan setiap neuron yang dipakai pada jaringan tersebut. Banyak fungsi yang dipakai sebagai pengaktif, diantaranya :

1. Fungsi Undak Biner (Hard Limit)

Jaringan dengan lapisan tunggal sering menggunakan fungsi undak untuk mengkonversi input dari suatu variable yang bernilai kontinu ke suatu output biner. Fungsi hard limit dirumuskan

$$Y=\left\{\begin{array}{c}0, jika x\leq 0\\1,jika x>0\end{array}\right.$$

1. Fungsi Undak Biner (Threshold)

Fungsi ini sering disebut fungsi niali ambang atau fungsi Heaviside, dirumuskan :

$$Y=\left\{\begin{array}{c}0, jika x<θ\\1,jika x\geq θ\end{array}\right.$$

1. Fungsi Bipolar Hampir sama dengan undak biner, hanya output yang dihasilkan berupa 1, 0 atau -1. Fungsi ini dirumuskan :

$$Y=\left\{\begin{array}{c}1, jika x>0\\0, jika x=0\\-1, jika x<0\end{array}\right.$$

1. Fungsi Bipolar (dengan Threshold)

Fungsi yang menghasilkan output berupa 1, 0 atau -1

$$Y=\left\{\begin{array}{c}1, jika x \geq θ\\-1,jika x<θ\end{array}\right.$$

1. Fungsi Linear (identitas)

Fungsi linear memiliki nilai output yang sama dengan nilai input, dirumuskan : y = x

1. Fungsi Sturating Linear

Fungsi ini dirumuskan :

$$Y=\left\{\begin{array}{c}1, jika x\geq 0\\x+0,5 , jika-o,5\leq x\leq 0,5\\0, jika x\leq 0\end{array}\right.$$

1. Fungsi Symetric Saturating Linear

Fungsi ini dirumuskan :

 $Y=\left\{\begin{array}{c}1, jika x\geq 0\\x , jika-1\leq x\leq 1\\-1, jika x\leq -1\end{array}\right.$

1. Fungsi Sigmoid Biner

Digunakan untuk jaringan syaraf yang dilatih dengan menggunakan metode backpropagation. Fungsi ini dirumuskan :

$$y=f\left(x\right)=\frac{1}{(1+e^{\acute{α}x})}$$

1. Fungsi Sigmoid Bipolar

Output dari fungsi ini memiliki range antara 1 sampai -1. Fungsi ini dirumuskan :

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

**Contoh program jaringan sederhana**

Contoh jaringan sederhana AND, OR dan NOT dapat dijelaskan seperti dibawah ini:

|  |  |  |
| --- | --- | --- |
| X1 | X2 | Y  |
| 1 | 1 | 1 |
| 1 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

Dari table diatas terlihat ada dua input (x1 dan x2) dan satu output (Y) dengan bobot w1 adalah 1 dan w2 adalah 1 sehingga diperoleh nilai threshold yang sesuai dengan tabel adalah 2. Arsitektur jaringan sederhana untuk kasus AND

1

1

Threshold(Y)=2

Gambar 8.3 Arsitektur jaringan sederhana operasi AND

Contoh program jaringan sederhana operasi AND adalah :

#include<iostream.h>

Main()

{

 Int W1,W2,X1,X2,O;

 W1=1;W2=1

 Cout<<”X1=”; cin>>X1;

 Cout<<”X2=”;cin>>X2;

 O=X1\*W1+X2\*W2

 If(O>=2)

 Cout<<”Output AND =1”<<endl;

 Else

 Cout<<”Output AND=0”<<endl;

}

Buat tabel, arsitektur dan program jaringan sederhana operasi OR dan NOT.

**8.4 Proses Belajar**

Proses pembelajaran terdiri dari beberapa metode.

1. Metode pelatihan terbimbingan

Metode pembelajaran pada jaringan syaraf disebut terawasi jika output yang

diharapkan telah diketahui sebelumnya. Metoda ini memasukan target keluaran

alam data untuk proses pelatihan. Beberapa metoda terbimbing diantaranya :

1. Jaringan Single Perceptron

Biasanya digunakan untuk mengklasifikasikan suatu tipe pola tertentu yag sering dikenal dengan pemisahan secara linear. Algoritma yang digunakan akan mengatur bebasnya melalui proses pembelajaran.

1. Hebb Rule

Metode pembelajaran yang paling sederhana, karena pembelajaran dilakukan dengan cara memperbaiki nilai bobot sedemikian rupa sehingga jika ada 2 neuron yang terhubung dan keduanya dalam kondidi “on” pada saat yang sama, maka bobot keduanya dinaikkan.

1. Delta Rule

Mengubah bobot yang menghubungkan antara jaringan input ke unit output dengan nilai target.

1. Backpropagation

Algoritma pembelajaran yang terawasi dan biasanya digunakan oleh perception dengan banyak lapisan untuk mengubah bobot-bobt yang terhubung dengan neuron-nuron yang ada pada lapisan tersembunyi.

1. Hetroassociative Memory

Jaringan yang bobot-bobotnya ditentukan sedemikian rupe sehingga jaringan tersebut dapat menyimpan kumpulan pola.

1. Bidirectional Associative Memory

Model jaringan syaraf yang memiliki 2 lapisan dan terhubung penuh dari satu lapisan ke lapisan lainnya.

Pada jaringan ini dimungkinkan adanya hubungan timbal balik antara lapisan input dan lapisan ouput.

1. Learning Vector Quantization

Suatu metode uantuk melakukan pembelajaran pada lapisan kompetitif yang terawasi. Suatu lapisan kompetitif akan secara otomatis belajar untuk mengklasifikasikan vektor-vektor input. Kelas-kelas yang didapatkan sebagai hasil hanya tergantung pada jarak antara vektor-vektor input.

1. Metode pelatihan tak terawasi (Jaringan Kohonen)

Jaringan kohonen pertama diperkenalkan oleh Prof. Teuvo Kohenen tahun 1982. Pada jaringan ini, suatu lapisan yang berisi neuron-neuron akan menyusun dirinya sendiri berdasarkan input nilai tertentu dalam suatu kelompok yang dikenal dengan istilah cluster.

Selama proses penyusunan diri, cluster yang memiliki vektor bobot paling cocok dengan pola input akan terpilih sebagai pemenang.

**8.5 Jaringan Single Percentron**

Jaringan lapis tunggal perceptron (single layer perceptron) terdiri dari beberapa unit pemroses (neuron). Jaringan ini terhubung dan mempunyai beberapa masukan serta sebuah atau beberapa keluaran. Single perceptron sering disebut juga dengan Perceptron. Perceptron menghitung nilai perkalian penimbang dan masukan dari parameter permasalahan yang kemudian dibandingkan dengan nilai threshold. Bila nilai keluaran lebih besar dari threshold maka keluarannyadalah satu, sebaliknya adalah nol. 

Gambar 8.4 Single perceptron

Pernyataan ini merupakan hasil proses pelatihan yang dalam bentuk bahasanya adalah pernyataan ya atau tidak atau bukan. Secara matematis dapat dituliskan sebagai berikut:

Jumlah perkalian penimbang dan parameter masukan adalah,

$$I=w\_{ji}x\_{i}$$

Dimana $x\_{i}$, adalah snyal masukan dan $w\_{ji}$adalah penimbangnya. Bila I>T maka keluaran O = 1 sebaliknya O = 0 dimana T adalah treshold.

Pelatihan pada perceptron dilakukan dengan merubah nilai penimbangnya sehingga sesuai dengan kebutuhan yang dilakukan dengan membandingkan keluaran dari jaringan dengan targetnya dan proses tersebut dituliskan seperti berikut ini :

$$wbaru\_{ji}=wlama\_{ji}+α(t\_{j}-O\_{j})x\_{i}$$

$t\_{j}$ adalah target, dan $α$ adalah bilangan konstan bernilai kecil antara 0,1 sampai 0,9 yang disebut sebagai laju pelatihan (learning rate). Proses ini dijalankan pada setiap neuron yang ada pada setiap lapisan (layer) dari jaringan sampai penimbang tersebut sesuaI dengan yang diinginkan. Nilai awal penimbang adalah bilangan kecil antara 0-1 yang dibangkitkan secara acak.

 **8.6 Pelatihan Sebuah Single Perceptron**

Pelatihan pada perceptron dilakukan dnegan merubah nilai penimbangnya sehingga sesuai dengan yang dilakukan dengan membandingkan keluaran dari jaringan dengan targetnya. Contoh kasus operasi And seperti pada tabel 8.1. Bobot (W) didapat dengan cara mengacak (random) kemudian dilakukan perhitungan perkalian antara input (I) seperti pada Gambar 8.5

Tabel 8.1 perhitungan pelatihan perceptron

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **I1** | **I2** | **I3** | **Penjumlahan** | **Output** |
| 0 | 0 | 1 | (0\*0.5)+(0\*0.4)+(1\*-0.3)=-0.3 | 0 |
| 0 | 1 | 1 | (0\*0.5)+(1\*0.4)+(1\*-0.3)=-0.7 | 0 |
| 1 | 0 | 1 | (1\*0.5)+(0\*0.4)+(1\*-0.3)=0.2 | 1 |
| 1 | 1 | 1 | (1\*0.5)+(1\*0.4)+(1\*-0.3)=-0.2 | 0 |



Gambar 8.5 Ilustrasi input dan perkalian dengan bobot

Output diperoleh dengan fungsi aktivasi dalam kasus AND dilakukan dengan unit step yaitu bila kurang dari nol output = 0, bila lebih output = 1. Error diperoleh apabila terjadi perbedaan antara target dengan output jaringan. Jiak error tidak sama dengan nol maka bobot (W) diperbaiki.

**While(Epoch menghasilkan error)**

**....**

**Err= T – O**

**If Err <> 0 then**

**Wj = Wj + LR \* Ij \* Err**

**End If**

**End**

Keterangan :

**Epoch** : set training pada jaringan

Kasus AND epoch berisi 4 set dari input dipresentasikan pada jaringan ([0,0],[0,1],[1,0],[1,1])

**Nilai Training T** : Training sebuah jaringan

Kita tidak hanya memperhatikan input tetapi juga nilai yang dihasilkan jaringan.

Contoh jika kita mempresentasikan jaringan dengan [1,1] untuk fungsi AND nilai training 1.

**Error,Err** : Nilai error

Adalah perbedaan nilai output dari jaringan dengan nilai training. Contoh jika kita membutuhkan jaringan untuk training 0 dan output 1, maka Err = -1

Output dari Neuron, O :

**Ij** = input dari neuron

**Wj** : Bobot dari input neuron (Ij) ke output neuron

**LR** : Learning Rate.

**Jaringan Konvergen**

1. Bobot diacak. Misalnya (0,1)
2. Kemudian dilakukan perbaikan bobot terus menerus sehingga diperoleh persamaan garis yang korvengen artinya dapat membagi dua kelompok dengan baik.

**Catatan :**

I1  point = W0/W1

I2 point = W0/W2



Gambar 8.6 Proses jaringan konvergen

**8.7 Contoh Jaringan Single perceptron**

Pada kasus operasi AND diatas dapat dibuat suatu disain arsitektur jaringan seperti pada Gambar 8.6



Gambar 8.7 Arsitektur Jaringan Operasi AND

Langkah penyelesaian :

* Struktur jaringan yang digunakan adalah 2-lapis yaitu 2-unit masukan kode biner (00-01-10-11) ditambah 1-bias dan 1-unit keluaran kode biner (0 atau 1)**.**
* Inisialisasi penimbang diambil bilangan acak antara -0.15 sampai 0.15sebanyak 3 penimbang yang menghubungkan antara unit pada lapisan masukan dengan unit pada lapisan keluaran.
* Besarnya nilai laju pelatihan ditentukan : 0,1

Pada kasus aplikasi perseptron multi output untuk mengenali angka desimal yang terbentuk dari BCD 7-segment. Sebagai contoh bagaimana proses perceptron mengenali angka daei 0-9 terlihat seperti pada Gambar 8.7



Gambar 8.8 Pengenalan angka desimal dari BCD 7-Segment

Langkah penyelesaian :

* Struktur JST yang digunakan adalah 7-lapis yaitu 7-unit masukan kode biner (0000000-1111111) ditambah 1-bias dan 10-unit keluaran kode biner (1000000000-000000001).
* Inisialisasi penimbang siambil bilangan acak antara -0.15 sampai 0.15 sebanyak 7 unit pada lapisan masukkan dikali 10 unit pada lapisan keluaran (7\*10).
* Besarnya nilai laju pelatihan ditentukan : 0,1

**8.8 Multi Perceptron**

Single perceptron membagi input dalam dua class dan vektor bobot dari single perceptron terbagi dalam persamaan hyperlane. Persamaan harus linear seperable. Untuk mengatasi permasalahan yang tidak linear seperable seperti Gambar 8.8



Gambar 8.9 Persamaan multi perceptron yang tidak linear seperable

Pada kasus operasi XOR tidak bisa diselesaikan dengan single perceptron tetapi harus dengan multi perceptron. Disain arsitektur jaringan untuk operasi XOR seperti pada gambar 8.10



Gambar 8.10 Arsitektur jaringan operasi XOR

 Langkah penyelesaian :

* Lapisan input : N input {xi}, i = 0,1,…, N-1
* Lapisan Tersembunyi : M neuron $Y\_{i}$ = $\sum\_{}^{}w\_{ji}x\_{i}$
* Lapisan Output : 1 neuron $z=\sum\_{}^{}S\_{k}y\_{k}$
* Input : $\overbar{x}$
* Minimum Square Error

$E=\frac{1}{2}(T\_{\overbar{x}}-O\_{\overbar{x}})^{2}$

* Pembelajaran dengan merubah parameter w

$$w\leftarrow w-η\frac{∂E}{∂w}$$

* Pembelajaran untuk lapisan output

$$\frac{∂E}{∂S\_{k}}=\left(T\_{\overbar{x}}-O\_{\overbar{x}}\right)\frac{∂z}{∂s\_{k}}=(T\_{\overbar{x}}-O\_{\overbar{x}})\grave{g}s\_{k}$$

$$\acute{g}=\frac{dg(x)}{dx},x=\sum\_{k}^{}s\_{k}y\_{k}$$

$$s\_{k}\leftarrow s\_{k}-η(T\_{\overbar{x}}-O\_{\overbar{x}})\dot{g}y\_{k}$$

Pembelajaran pada lapisan tersembunyi

$$\frac{∂E}{∂w\_{ji}}=(T\_{\overbar{x}}-O\_{\overbar{x}})\sum\_{k}^{}\frac{∂z}{∂y\_{k}}\frac{∂y\_{k}}{∂w\_{ji}}=(T\_{\overbar{x}}-O\_{\overbar{x}})\dot{g}s\_{k}$$

$$=\left(T\_{\overbar{x}}-O\_{\overbar{x}}\right)\sum\_{k}^{}\dot{g}y\_{k}\dot{f}δ\_{ki}x\_{j}=(T\_{\overbar{x}}-O\_{\overbar{x}})\dot{g}s\_{k}\dot{f}x\_{j}$$

$$\acute{f}=\frac{df(x)}{dx},x=\sum\_{j}^{}w\_{ji}w\_{j}$$

$$w\_{ji}\leftarrow w\_{ji}-η\dot{\left(T\_{\overbar{x}}-O\_{\overbar{x}}\right)\acute{g}s\_{i}f}x\_{j}$$

**8.9 Metode Backpropagation**

Algoritma pelatihan Backpropagation(BP) atau propagasi balik, pertama kali dirumuskan oleh Werbos dan dipopulerkan oleh Rumelhart dan McClellacd untuk dipakai pada JST. Algoritma ini termasuk metode pelatihan supervised dan didisain untuk operasi pada jaringan feed forward multi lapis.

Metoda BP banyak dipakai pada aplikasi pengaturan karena proses pelatihannya didasarkan pada hubungan yang sederhana, yaitu : jika keluaran memberikan hasil yang salah, maka penimbang dikoreksi supaya galatnya dapat diperkecil dan respon jaringan selanjutnya diharapkan akan lebih mendekati harga yang benar. BP juga berkemampuan untuk memperbaiki penimbang pada laisan tersembunyi (hidden layer).

Secara garis besar BP dapat dideskripsikan sebagai berikut : ketika jaringan diberikan pola masukan sebagai pola pelatihan maka pola tersebut menuju ke unit-unit pada lapisan tersembunyi untuk diteruskan ke unit-unit lapisan keluaran. Kemudian unit-unti tersebut memberikan tanggapan yang disebut sebagai keluaran jaringan. Saat keluaran jaringan tidak sama dengan keluaran yang diharapkan maka keluaran akan menyebar mundur (backward) pada lapisan tersembunyi diteruskan ke unit pada lapisan masukan.

Tahap pelatihan ini merupakan langkah bagaimana suatu jaringa saraf itu berlatih, yaitu dengan cara melakukan perubahan penimbang (sambungan antara lapisan yang membentuk jaringan melalui masing-masing unitnya). Sedangkan pemecahan masalah baru akan dilakukan jika proses pelatihan telah selesai, fase tersebut adalah fase mapping atau proses pengujian/testing.

Algortima pelatihan BackPropagation terdiri dari dua proses, feed forward dan backpropagation dari galatnya. Rinciannya sebagai berikut :

Langkah 0 :

 Pemberian inisialisasi penimbang (diberi nilai kecil secara acak)

Langkah 1 :

 Ulangi langkah 2 hingga 9 sampai kondisi akhir iterasi dipenuhi

Langkah 2 :

Untuk masing-masing pasangan data pelatihan (training data) lakukan langkah 3 hingga 8

 Propagasi maju (Feedforward)

Langkah 3 :

Masing-masing unit masukan (Xi, I = 1,…,n) menerima sinyal masukan Xi dan sinyal tersebut disebarkan ke unit-unit bagian berikutnya (unit-unit lapisan tersembunyi)

 Langkah 4 :

Masing-masing unit dilapisan tersembunyi dikalikan dengan faktor penimbang dan dijumlahkan serta ditambahkan dengan biasnya :

$$Z\_{in\_{j}}=V\_{oj}+\sum\_{I=1}^{n}X\_{i}V\_{ij}$$

Kemudian menghitung sesuai dengan fungsi aktifasi yang digunakan :

$$Z\_{j}=f(Z\_{in\_{j}})$$

Bila yang digunakan adalah fungsi sigmoid maka bentuk fungsi tersebut adalah :

$$Z\_{j}=\frac{1}{1+exp^{-z\\_in\_{j}}}$$

Masing-masing unit keluaran (Yk, k=1,2,3,…m) dikalikan dengan faktor penimbang dan dijumlahkan :

$$Y\_{in\_{k}}=W\_{ok}+\sum\_{j=1}^{p}Z\_{j}W\_{jk}$$

Menghitung kembali sesuai dengan fungsi aktifasi

$$y\_{k}=f(y\_{in\_{k}})$$

Backpropagation dan galatnya

 Langkah 6 :

Masing-masing unit keluaran (Yk, k=1,…,m) menerima pola target sesuai dengan pola masukan saat pelatihan dan dihitung galatnya :

$$δ\_{k}=\left(t\_{k}-y\_{k}\right)f^{'}(y\\_in\_{k})$$

Karena $f^{'}(y\\_in\_{k})=y\_{k}$ menggunakan fungsi sigmoid, maka :

$$f^{'}(y\\_in\_{k})=f(y\\_in\_{k})\left(1-f(y\\_in\_{k})\right)$$

$$=y\_{k}(1-y\_{k})$$

Menghitung perbaikan faktor penimbang (kemudian untuk memperbaiki $w\_{jk}$

$$∆W\_{kj}=α.∂\_{k}.Z\_{j}$$

Menghitung perbaikan koreksi :

$$∆W\_{0j}=α.∂\_{k}$$

Dan menggunakan nilai $δ\_{k}$ pada semua unit lapisan sebelumnya.

 Langkah 7 :

 Masing-masing penimbang yang menghubungkan unit-unit l;apisan keluaran dengan unit-unit pada lapisan tersembunyi ($Z\_{j}, j=1,…,p)$ dikalikan delta dan dijumlahkan sebagai masukan ke unit-unit lapisan berikutnya.

$$δ\\_in\_{j}=\sum\_{k=1}^{n}δ\_{k}W\_{jk}$$

Selanjutnya dikalikan dengan turunan dari fungsi aktifasinya untuk menghitung galat.

$δ\_{j}=δ\\_in\_{j}f'(y \\_in\_{j}$)

Kemudian menghitung perbaikan penimbang (digunakan untuk memperbaiki $V\_{ij}$)

$$∆V\_{ij}=αδ\_{j}X\_{i}$$

Kemudian menghitung perbaikan bias (untuk memperbaiki $V\_{oj}$)



Memperbaiki penimbang dan bias

 Langkah 8 :

Masing-masing keluaran unit (yk, k=1,…,m) diperbaiki bias dan penimbangnya (j=0,…,p)



Masing-masing unit tersembunyi (Zj, j:1,…,p) diperbaiki bias dan penimbangnya (j=0,…,n)



 Langkah 9 :

 Uji kondisi pemberhentian (akhir iterasi)

 Daftar Notasi

|  |  |  |
| --- | --- | --- |
| **Notasi** |  | **Keterangan** |
| Xp | = | Pola masukan pelatihan ke-p, p=1,2,…,p<=1. (X1,X2,X3, …,Xn) |
| tp | = | Pola keluaran target dari pelatihan. (t1, t2, t3, …, tn) |
| xi | = | Unit ke-I pada lapisan masukan |
| Xi | = | Nilai aktifasi dari unit Xi |
| Zj | = | Unit ke-j pada lapisan tersembunyi |
| Z\_inj | = | Keluaran untuk unit Zj |
| zj | = | Nilai aktifasi dari unit Zj |
| Yk | = | Unit ke-k pada lapisan keluaran |
| Y\_ink | = | Net masukan untuk unit Yk |
| Wk0 | = | Nilai penimbang sambungan pada bias untuk unit Yk |
| Wkj | = | Nilai penimbang sambungan dari Zij ke unit Yk |
| $∇W$kj | = | Selisih Wkj(t) dengan Wkj(t+1) |
| Vio | = | Nilai penimbang sambungan pada bias untuk unit Zi |
| $∇W$ij | = | Selisih antara Vij(t) dengan Vij(t+1) |
| $δ$k | = | Faktor pengaturan nilai penimbang sambungan pada lapisan keluaran |
| $δ$j | = | Faktor pengaturan niloai penimbang sambungan pada lapisan tersembunyi |
| α | = | Konstanta laju pelatihan (learning rate) 0<α<1 |
| E | = | Total galat |

**8.10 Metode Pelatihan tak Terbimbing/Mandiri**

Metode pelatihan tak terbimbing adalah pelatihan tanpa memerlukan target pada keluarannya. Proses pelatihan berdasarkan proses transformasi dari bentuk variabel kontinyu menjadi variabel diskrit yang dikenal dengan kuantisasi vektor. Jaringan yang digunakan untuk proses pelatihan tak terbimbing ini adalah jaringan umpan balik (feedback network).

1. **Metode Kohonen / Self-Organizing Map (SOM)**

Teknik self-organizing map dikenalkan pertama kali oleh Teuvo Kohonen, merupakan proses unsupervised learning yang mempelajari distribusi himpunan pola-pola tanpa informasi kelas.

Ide dasar teknik ini diilhami dari bagaimana proses otak manusia menyimpan gambar/pola yang telah dikenalinya melalui mata, kemudian mampu mengungkapkan kembali gambar/pola tersebut. Oleh karenanya aplikasi model JST ini banyak digunakan pada pengenalan obyek/citra visual (visual image).

Proses pemetaan terjadi bila sebuah pola berdimensi bebas diproyeksikan dari ruang masukan ke posisi pada array berdimensi satu atau dua. Metode ekstraksi informasi tersebut dapat dinyatakan sebagai observasi terhadap sebuah pola melalui jendela yang terbentuk oleh lokasi unit-unit luasan pola. Pola yang dikenali hanya pola yang batasan unit lokasinya jelas berbeda. Biasanya observasi hanya dapat dilakukan bila lokasi pola tersebut mendapat iluminasi/pencahayaan yang cukup normal.

Meskipun SOM adalah proses klasifikasi, namun tidak seperti teknik klasifikasi atau pengelompokan yang umum digunakan, yang hanya menyediakan penataan kelas-kelas berdasarkan topologinya. Kemiripan pada pola masukan dipertahankan agar tidak berubah sampai keluaran proses.

Pada JST ini lapisan masukan (pertama) terhubung secara penuh dengan lapisan kompetitif (kedua). Jadi setiap unit masukan terhubung ke semua unit keluaran dan pada hubungan tersebut terdapat nilai penimbang (weight) tertentu.

1. Metode Hopfield

Metode ini dikembangkan oleh John Hopfield pada tahun 1980. Cara pemahaman paling mudah JST Hopfield bila dinyatakan sebagai sebuah memori asosiatif yang gambaran umumnya adalah sebagai berikut :

Bila ada sebuah JST yang terbentuk dari N X N neuron dimana N adalah jumlah varibel dari objek yang akan dioptimasi. Setiap neuron terhubung penuh satu sama lainnya. Ilustrasi pernyataan tersebut adalah sebagai berikut :





Gambar 8.11 Layout dari JST Hopfield

JST Hopfield merupakan salah satu metode optimasi untuk pencarian nilai minimum dari kombinasi fungsi objektif.

**8.11 Metode Pelatihan Hibrida**

Semakin tinggi tingkat ketidaklinieran suatu sistem, sering kali sulit diselesaikan dengan kedua metode yang telah diuraikan sebelumnya yaitu metode pelatihan terbimbing dan tak terbimbing. Untuk mengatasi problem tersebut, banyak yang mencoba menggabungkannya dan diperoleh hasil yang lebih baik. Salah satu metode hibrida yang berhasil dan banyak digunakan adalah **metode radial basis function network (RBFN).**

Model jaringan berbasis fungsi radial ini, neuron-neuron keluarannya adalah hasil kombinasi linier fungsi basis neuron-neuron pada lapisan tersembunyi. Sebagai fungsi basis yang umum digunakan adalah Gaussian. Perbedaan utama antara jaringan multi lapis perceptron (MLP) dengan jaringan berbasis fungsi radial yaitu penggunaan gaussian pada lapisan tersembunyi jaringan RBF, sedangkan jaringan MLP menggunakan fungsi sigmoid.

Pada prinsipnya RBF adalah emulasi sifat jaringan biologi yang umumnya sel/neuron yang paling aktif adalah sel/neuron yang paling sensitif menerima rangsangan sinyal masukan. Sehingga orientasi sensitivitas respon tersebut hanya terhadap beberapa daerah (local response) dalam wilayah masukan. JST dengan lapisan tersembunyi tunggal, pada dasarnya lapisan tersebut berisi neuron-neuron yang sensitif atau aktif lokal. Sedangkan keluarannya terdiri dari unit-unit linier.

Pada unit-unit dalam lapisan tersembunyi, respon unitnya bersifat lokal dan berkurang sebagai fungsi jarak masukan dari pusat unit penerima rangsangan. Metode ini menjadai terkenal sejak Broomhead dan Lowe’s pada tahun 1988 menyampaikan makalahnya yang berjudul “Multivariate functional interpolation and adaptive network”.

Fungsi Radial

Fungsi radial adalah suatu fungsi yang mempunyai karakteristik merespon pengurangan ataupun penambahan secara monoton dengan jarak yang berasal dari nilai tengahnya. Jenis fungsi radial yang banyak digunakan adalah fungsi Gaussian seperti pada persamaan berikut :

$$h\left(x\right)=Φ(\left(x-c\right)^{T}R^{-1}\left(x-c\right))$$

dimana $Φ$ adalah jneis fungsi aktivasi yang digunakan dalam RBF

 c adalah pusat (nilai tengah)

 R adalah matrik dengfan R:r2 dan r merupakan jari-jari skalar

$\left(x-c\right)^{T}R^{-1}\left(x-c\right)$ adalah jarak antara input x dan pusat c dalam matriks yang ditetapkan oleh R.

 Beberapa tipe fungsi aktivasi RBF adalah sebagai berikut :

* Fungsi Thinplate-Splane

$$Φ\left(z,1\right)=z^{2}log⁡(z)$$

* Fungsi Multiquadratic

$$Φ\left(z,σ\right)=(z^{2}+σ^{2})^{1/2}$$

* Fungsi Inverse Multiquadratic

$$Φ\left(z,σ\right)=^{1}/\_{(z^{2}+σ^{2})^{1/2}}$$

* Fungsi Gauss

$$Φ\left(z,σ\right)=exp⁡(-z^{2}/σ^{2})$$

Hasil penelitian menyatakan bahwa seleksi dari keempat fungsi nonlinier tersebut tidak dominan menentukan kinerja RBF. Bila jarak Euclidien antara vektor masukan dan unit-unit dalam lapisan tersembunyi mempunyai nilai yang berbeda, maka jarak yang sama untuk setiap unitnya cukup untuk pendekatan secara universal. Ini berarti bahwa semua jarak dapat disesuaikan pada sebuah nilai $σ$ untuk menyederhanakan strategi pelatihannya.

**LATIHAN**

Cari artikel tentang aplikasi jaringan syaraf tiruan. Buat resume tentang bagaimana cara sistem belajar.