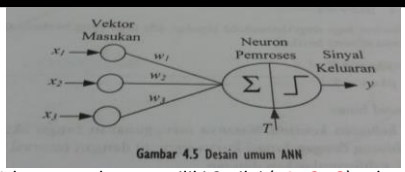
Artificial Neural Network

ANN

ANN

- Merupakan konsep rekayasa pengetahuan dalam bidang kecerdasan buatan yang di desain mengadopsi sistem saraf manusia dimana pemrosesan utama ada di otak.
- Model yang mengadopsi cara kerja neuron secara biologi dengan fokus pada cara kerja saraf otak manusia

ANN



Gambar 4.5 Desain umum ANN

- Vektor masukan memiliki 3 nilai (**x1,x2,x3**) sebagai fitur yang akan diproses, masing-masing nilai masukan melewati jalur berbobot **w**, kemudian menghitung nilai gabungan **∑** dan diproses oleh fungsi aktivasi **F[]** untuk menghasilkan sinyal **y** sebagai output.
- Fungsi aktivasi **F[]** menggunakan nilai threshold utk membatasi nilai keluaran agar selalu dlm batas nilai yg ditetapkan.

ANN

- ANN membutuhkan proses pelatihan agar dpt melakukan prediksi kelas suatu data uji baru yang ditemukan.
- Proses pelatihan ANN menggunakan metode: Perceptron,Backpropagation,Self Organizing Mapping,Delta,Associative Memory,Learning Vector Quantization

Fungsi Aktivasi

- ANN menggunakan fungsi aktivasi utk membatasi keluaran Y agar sesuai dengan batasan sinyal outputnya

1. Fungsi aktivasi Linear
2. Fungsi aktivasi Step
3. Fungsi aktivasi Sigmoid Biner
4. Fungsi aktivasi Sigmoid Bipolar

Fungsi aktivasi:

1. Fungsi aktivasi Linear
   utk output bertipe deskrit
   Y = sign(v) = v
   v : nilai input

2. Fungsi aktivasi Step
   Bipolar
   y=sign(v) = { 1 jk v>= T
   -1 jk v< T
   Biner
   y=sign(v) = { 1 jk v >= T
   0 jk v < T
   v:nilai gabungan dari semua vektor oleh adder

Fungsi aktivasi:

3. Fungsi aktivasi Sigmoid Biner
   Utk output dengan tipe continue

$$Y = \text{sign}(v) = \frac{1}{1 + e^{-v}}$$

4. Fungsi aktivasi Sigmoid Bipolar

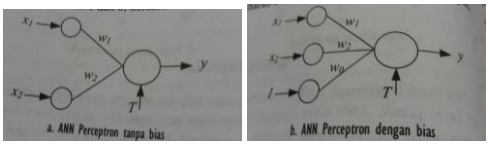
$$Y = \text{sign}(v) = \frac{2}{1 + e^{-v}} - 1$$

Metode pelatihan yang digunakan berdasarkan Jumlah Layer dlm ANN

- Layer Tunggal
  Perceptron,Delta
- Layer Jamak/multi layer
  Backpropagation,Recurrent Network

Perceptron

- Merupakan salah satu ANN berlayer tunggal
- Diperkenalkan oleh Fran Rosenbelt (1958)



- Bias

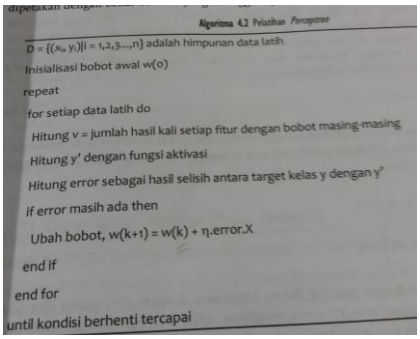
Berupa vektor masukan 1 dengan bobot w0.
Mempunyai pengaruh dalam meningkatkan/menurunkan input ke fungsi aktifasi

Perceptron

Memiliki kelemahan:

- Keterbatasan komputasi tdk dpt membuat generalisasi global pada basis data yang dilatih secara lokal.
- Tdk bisa bekerja dengan baik utk data yang tidak dapat dipisah secara linear.

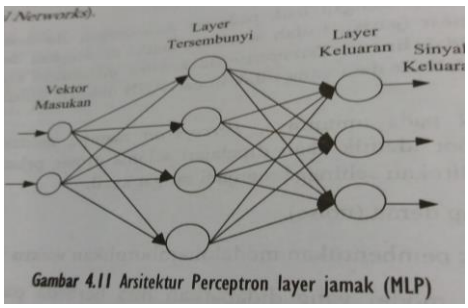
Algoritma Pelatihan Perceptron



Algoritma 4.2 Pelatihan Perceptron

$D = \{(x_i, y_i) | i = 1,2,3...,n\}$ adalah himpunan data latih
Inisialisasi bobot awal w(o)
repeat
 for setiap data latih do
  Hitung v = jumlah hasil kali setiap fitur dengan bobot masing-masing
  Hitung y' dengan fungsi aktivasi
  Hitung error sebagai hasil selisih antara target kelas y dengan y'
  if error masih ada then
   Ubah bobot, w(k+1) = w(k) + η.error.X
  end if
 end for
until kondisi berhenti tercapai

Multilayer Perceptron



Gambar 4.11 Arsitektur Perceptron layer jamak (MLP)

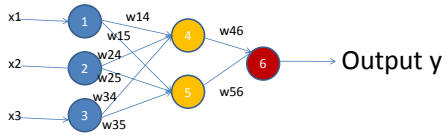
Multilayer Perceptron

- Merupakan ANN turunan dari perceptron
- Berupa ANN *feedforward* dengan satu atau lebih *hidden layer*

Backpropagation

- Pertama kali diperkenalkan oleh Bryson & Ho tahun 1969.
- Merupakan metode pembelajaran *ANN*(jaringan saraf tiruan) yang bekerja melalui proses secara iteratif dengan menggunakan *data training* membandingkan nilai prediksi dari jaringan dengan setiap contoh data.
- Dalam setiap proses bobot relasi jaringan dimodifikasi untuk meminimalkan nilai MSE(Mean Squared Error) antara nilai prediksi dari jaringan dengan nilai sesungguhnya .
- Modifikasi relasi jaringan tsb dilakukan dalam arah mundur dari layer output ke layer pertama dari hidden layer.

Backpropagation



- Vektor pola masukan diberikan pada **layer input** kemudian dirambatkan ke **layer hidden** sampai nilai output dibangkitkan oleh **layer output**
- Jika nilai/pola output berbeda dengan nilai yang diinginkan hitung error yang dirambatkan balik dari layer output ke layer input dengan melakukan modifikasi bobot selama proses perambatan balik

Algoritma Backpropagation

- Inisialisasi
  - Tentukan bobot [-0.5 s/d 0.5, haykin 1999]
  - Tentukan fungsi aktivasi
  - Tentukan learning rate
- Repeat
  - Aktivasi
    - Hitung keluaran pd neuron hidden layer
    - Hitung keluaran pd neuron output
  - Perbaharui bobot
    - Hitung gradien error pd neuron output
    - Hitung gradien error pd neuron hidden layer
- Until Kondisi berhenti tercapai

Algoritma Backpropagation

- Inisialisasi
  - Tentukan bobot [-0.5 s/d 0.5, haykin 1999], fungsi aktivasi,learning rate
- Repeat
  - Aktivasi
    - Hitung keluaran pd neuron hidden layer
    - Hitung keluaran pd neuron output

$$y_j(p) = \frac{1}{1 + e^{-v_j(p)}} \qquad y_k(p) = \frac{1}{1 + e^{-v_k(p)}}$$

  - Perbaharui bobot
    - Hitung gradien error pd neuron output

$$\delta_k(p) = y_k(p) \times [1 - y_k(p)] \times e_k(p)$$

Hitung koreksi bobot:

$$\Delta w_{jk}(p) = \eta \times y_j(p) \times \delta_k(p)$$

Perbarui bobot pada neuron layer keluaran:

$$w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p)$$

    - Hitung gradien error pd neuron hidden layer

$$\delta_j(p) = y_j(p) \times [1 - y_j(p)] \times \sum_{k=1}^{l} \delta_k(p) \cdot w_{jk}(p)$$

Hitung koreksi bobot:

$$\Delta w_{ij}(p) = \eta \times x_i(p) \times \delta_j(p)$$

Perbarui bobot pada neuron layer tersembunyi:

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

- Until Kondisi berhenti tercapai

Keterangan

- Learning rate (η)

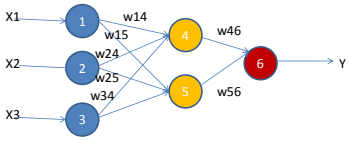
  0-1, semakin besar nilainya semakin cepat selesai proses pelatihan namun bisa terjebak dlm local optima, semakin kecil nilai semakin lama namun menjamin hasil model yg lebih baik (umumnya 0,1 – 0,3)
- Momentum

  0-1 Penambahan momentum utk membantu mempercepat proses pencapaian target error tapi dengan learning rate yg kecil . (umumnya 0,9)

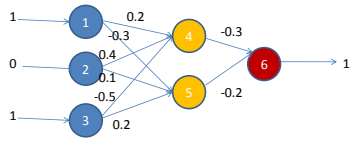
Contoh:

X1	X2	X3	Y
1	0	1	1
1	1	0	1

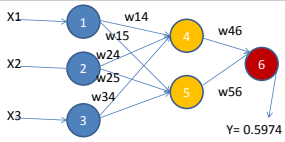
Untuk data 1:
X1: 1, X2: 0, X3 : 1, Output : 1
Learning rate (η) : 0.9



X1: 1, X2: 0, X3 : 1, Output : 1
Inisialisasi bobot awal secara acak.
W14 = 0.2 W34 = -0.5
W15 = -0.3 W35 = 0.2
W24 = 0.4 W46 = -0.3
W25= 0.1 W56 = -0.2



Iterasi 1:
a. Hitung keluaran tiap node

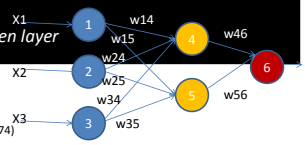


Neuron 4
Input = X1* W14+ X2* W24+ X3* W34
= 1* 0.2 + 0 *0.4+1*(-0.5)
= -0.3
Y_4 = $1/(1+e^{-0.3})$
= $1/(1+2.718^{-03})$ = 0.5744

Neuron 5
Input = X1*W15+X2*W25+I3*W35
= 1*(-0.3) + 0 * (0.1) + 1*(0.2)
= -0.1
Y_5 = $1/(1+e^{-0.1})$
= $1/(1+2.718^{-0.1})$ = 0.525

Neuron 6
Input = $Y_4 * W_{46} + Y_5 * W_{56}$
= 0.5744 * (-0.3) + (1.1111) * (-0.2)
= -0.3945
Y_6 = $1/(1+e^{-0.3945})$
= $1/(1+2.718^{-0.3945})$ = 0.5974

Iterasi1:
b. Hitung nilai *error output & hidden layer*



Neuron 6
Err6 = $Y_6 * (\eta - Y_6) * (Y_{target} - Y_6)$
= 0.5974 * (0.9 – 0.5974) * (1 – 0.5974)
= 0.0728

Neuron 5
Err_5 = $Y_5 * (\eta - Y_5) * Err_6 * W_{56}$
= 0.525 * (0.9 - 0.525) * 0.0728 * (-0.2)
= -0.00287

Neuron 4
Err_4 = $Y_4 * (\eta - Y_4) * Err_6 * W_{46}$
= 0.5744 * (0.9 – 0.5744) * 0.0728 * (-0.3)
= -0.0041

Iterasi 1:
c. Modifikasi / Hitung bobot baru

$W_{46} = W_{46} + \eta * Err_6 * Y_4$
= (-0.3)+0.9*0.0728*0.5744
= -0.2624

$W_{56} = W_{56} + \eta * Err_6 * Y_5$
= (-0.2)+0.9*0.0728 * 1.1111
= -0.1272

$W_{14} = W_{14} + \eta * Err_4 * X_1$
= 0.2+0.9*(-0.0041)*1
= 0.1963

$W_{15} = W_{15} + \eta * Err_5 * X_1$
= (-0.3) + 0.9 * 0.0034 * 1
= -0.2941

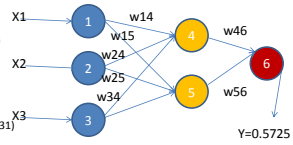
$W_{24} = W_{24} + \eta * Err_4 * X_2$
= 0.4 + 0.9 * (-0.0041) * 0
= 0.4

$W_{25} = W_{25} + \eta * Err_5 * X_2$
= 0.1 + 0.9 * (0.0034) * 0
=0.1

$W_{34} = W_{34} + \eta * Err_4 * X_3$
= -0.5 + 0.9 * (-0.0041) * 1
= -0.537

$W_{35} = W_{35} + \eta * Err_5 * X_3$
= 0.2 + 0.9 * (0.034) * 1
= 0.2031

Iterasi 2:
a. Hitung keluaran tiap node

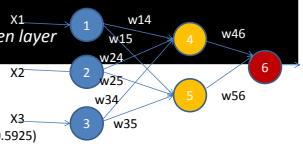


Neuron 4
Input = X1* W14+ X2* W24+ X3* W34
= 1 * 0.1963 + 0 * 0.4 + 1 * (-0.5037)
= -0.3074
Y_4 = $1/(1+e^{-0.3074})$
= $1/(1+2.718^{-03074})$ = 0.5763

Neuron 5
Input = X1*W15+X2*W25+I3*W35
= 1 * (-0.2969) + 0 * (0.1) + 1 * (0.2031)
= -0.0938
Y_5 = $1/(1+e^{-0.0938})$
= $1/(1+2.718^{-0.0938})$ = 1.1035

Neuron 6
Input = $Y_4 * W_{46} + Y_5 * W_{56}$
= 0.5763 * (-0.2622) + (1.1035) * (-0.1276)
= -0.2919
Y_6 =$1/(1+e^{-0.2919})$
= $1/(1+2.718^{-0.2919})$ = 0.5725

Iterasi2:
b. Hitung nilai *error output & hidden layer*



Neuron 6

Err6 $= Y_6 * (\eta - Y_6) * (Y_{target} - Y_6)$
$= 0.5925 * (0.9 - 0.5925) * (1 - 0.5925)$
$= 0.0802$

Neuron 5

$Err_5 = Y_5 * (\eta - Y_5) * Err_6 * W_{56}$
$= 1.1035 * (0.9 - 1.1035) * 0.0802 * (0.1276)$
$= -0.0023$

Neuron 4

$Err_4 = Y_4 * (\eta - Y_4) * Err_6 * W_{46}$
$= 0.5763 * (0.9 - 0.5763) * 0.0802 * (-0.2622)$
$= -0.0039$

Iterasi 2:
c. Modifikasi / Hitung bobot baru

$W_{46} = W_{46} + \eta * Err_6 * Y_4$
$= (-0.2622) + 0.9 * 0.0802 * 0.5763$
$= -0.2206$

$W_{56} = W_{56} + \eta * Err_6 * Y_5$
$= (-0.1276) + 0.9 * 0.0802 * 1.1035$
$= -0.0479$

$W_{14} = W_{14} + \eta * Err_4 * X_1$
$= 0.1963 + 0.9 * (-0.0039) * 1$
$= 0.1928$

$W_{15} = W_{15} + \eta * Err_5 * X_1$
$= (-0.2969) + 0.9 * 0.0023 * 1$
$= -0.2948$

$W_{24} = W_{24} + \eta * Err_4 * X_2$
$= 0.4 + 0.9 * (-0.0039) * 0$
$= 0.4$

$W_{25} = W_{25} + \eta * Err_5 * X_2$
$= 0.1 + 0.9 * 0.0023 * 0.5763$
$= 0.1$

$W_{34} = W_{34} + \eta * Err_4 * X_3$
$= 0.1 + 0.9 * (-0.0039) * 1$
$= -0.5072$

$W_{35} = W_{35} + \eta * Err_5 * X_3$
$= 0.2031 + 0.9 * 0.0023 * 1$
$= 0.2052$

Ulangi iterasi hingga

- Nilai MSE antara $Y_{prediksi\ dengan}$ Y_{target} Minimum
- $Err < Err_{target}$ tercapai.

Latihan

- Buatlah ANN dengan menggunakan metode backpropagation untuk data berikut:

No	Lampu	Jarak	Waktu
	X1	X2	Y
1	2	0,50	9,95
2	8	1,10	24,45
3	11	1,20	31,75
4	10	5,50	35,00
5	8	2,95	25,02