BAB II

LANDASAN TEORI

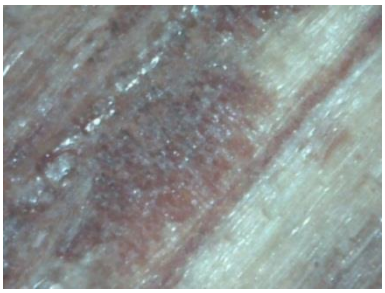
2.1 Citra/Image

Citra atau yang lebih sering dikenal dengan gambar merupakan kumpulan dari titiktitik penyusun citra itu sendiri. Titik-titik tersebut disebut dengan *pixel*. Banyaknya titik yang membangun citra itu sendiri disebut dengan resolusi. Masing-masing *pixel* yang menyusun sebuah citra tersebut dapat memiliki warna dengan variasi yang berbeda-beda, yang disebut dengan *bit depth*. *Bit depth* dinyatakan No Makalah : 299 dengan angka yang bersatuan bit. Misalnya citra dengan *bit depth* = 3, artinya terdapat 23 = 8 variasi warna yang mungkin pada setiap *pixel*nya. Semakin besar nilai *bit depth*, maka semakin besar pula ukuran fungsi citra tersebut (Nauval, 2016)

2.2 Kayu

Kayu adalah bagian batang atau cabang serta ranting tumbuhan yang mengeras karena mengalami lignifikasi (pengayuan). Kayu digunakan untuk berbagai keperluan, mulai dari memasak, membuat perabot (meja, kursi), bahan bangunan (pintu, jendela, rangka atap), bahan kertas, dan banyak lagi. Berikut adalah jenis kayu yang akan diolah:

2.2.1 Kayu Kamper



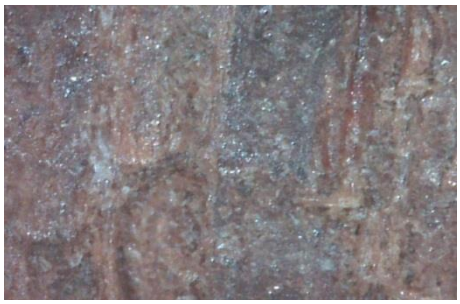
Gambar 2.1 Kayu Kamper

Gambar 2.1 adalah serat kayu pada kayu Kamper. Kayu Kamper merupakan salah satu jenis kayu khas dari daerah tropis, salah satunya seperti Indonesia. Di negara ini, kayu Kamper banyak terdapat di pulau Kalimantan, bahkan Kamper Samarinda telah lama dikenal oleh masyarakat (khususnya masyarakat Indonesia) sebagai jenis kayu Kamper yang memiliki kualitas baik. Dan karena kualitasnya terbukti cukup baik, tentunya kayu Kamper ini memiliki banyak peminat sehingga menjadikan kayu ini sebagai salah satu jenis kayu komersial di Indonesia,

2.2.2 Kayu Keruing

Gambar 2.2 adalah serat kayu pada kayu Keruing. Kayu Keruing merupakan salah satu jenis kayu khas dari daerah tropis salah satunya seperti Indonesia. Di negeri ini, kayu Keruing dapat dengan mudah ditemukan terutama di 3 pulau besar di Indonesia yakni di pulau Jawa, Sumatera dan Kalimantan. Selain dikenal sebagai kayu khas tropis, di Indonesia, kayu Keruing juga dikenal sebagai salah satu jenis kayu yang memiliki nilai jual cukup baik di pasaran, sebab kayu Keruing ini memang merupakan jenis kayu yang banyak dibutuhkan, terutama oleh industri-industri pengolahan kayu.

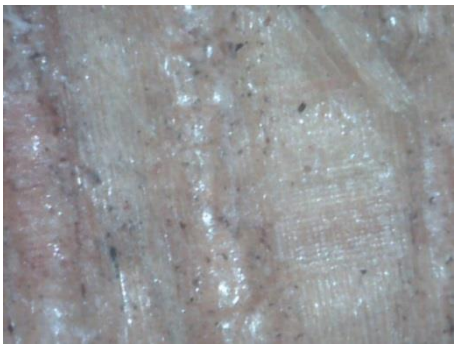
Tentunya, hal tersebut membuat popularitas kayu Keruing tambah naik dan memiliki nilai tawar di pasaran.



Gambar 2.2 Kayu Keruing

2.2.3 Kayu Meranti

Gambar 2.3 adalah serat kayu pada kayu Meranti. Kayu Meranti merupakan salah satu jenis *kayu* khas daerah tropis yang cukup terkenal. Dan kayu ini juga termasuk salah satu jenis kayu komersial yang banyak peminatnya. Di Indonesia, kayu Meranti berasal dari beberapa daerah yakni Sumatra, Kalimantan, Maluku, dan beberapa juga ada yang berasal dari Sulawesi. Dari keempat daerah itu, Kalimantan merupakan daerah penghasil kayu Meranti yang paling bagus dari pada tiga daerah lainnya, sehingga orang-orang banyak yang menyebut Meranti sebagai kayu kalimantan. Di pasaran, kayu Meranti dikenal memiliki tiga jenis yang berbeda, yakni Meranti merah, Meranti putih dan Meranti kuning. Khusus untuk jenis kayu Meranti kuning, daerah penghasilnya hanya ada dua yakni Sumatra dan Kalimantan saja.



Gambar 2.3 Kayu Meranti

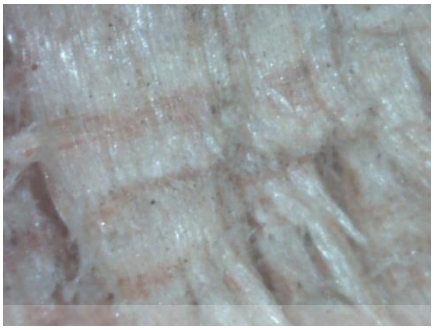
2.2.4 Kayu Pinus

Gambar 2.4 Kayu Pinus

Gambar 2.4 adalah serat kayu pada kayu Pinus. Kayu Pinus adalah salah satu jenis kayu khas dari daerah tropis yang bernilai komersial cukup baik di pasaran. Kayu Pinus ini terdiri dari banyak jenis yang berbeda-beda, tetapi hanya ada dua jenis yang banyak beredar di pasaran, sebab kedua jenis Pinus ini memang dikenal memiliki kualitas paling baik diantara jenis-jenis Pinus lainnya yakni Pinus radiata dan Pinus merkusii. Baik jenis Pinus radiata ataupun Pinus merkusii, keduanya adalah jenis Pinus yang cukup populer di Indonesia sebab kedua jenis Pinus tersebut merupakan jenis Pinus yang banyak digunakan oleh industri-industri perkayuan ataupun oleh individu

(masyarakat umum) sebagai bahan baku untuk membuat aneka macam furniture indoor ataupun jenis produk lainnya.

2.2.5 Kayu Randu



Gambar 2.5 Kayu Randu

Gambar 2.5 adalah serat kayu pada kayu Randu. Kayu Randu atau yang juga dikenal dengan nama kayu kapuk merupakan jenis kayu yang cukup familier bagi sebagian besar masyarakat tanah air. Sebab setiap kali seseorang membangun rumah, bisa dipastikan hampir selalu membutuhkan kayu Randu ini untuk material pendukung proses pengerjaan bangunannya yakni untuk media membuat cor-coran.

2.3 Grayscalling

Grayscalling adalah proses perubahan warna dari citra dari berwarna menjadi keabuan. *Grayscalling* citra diperlukan karena GLCM mengekstraksi fitur dengan memperhitungkan level aras keabuan ketetanggaan. Pengubahan dari citra berwarna menjadi bentuk *grayscale* biasanya mengikuti aturan seperti berikut :

$$I(i,j) = \frac{R(i,j) + G(i,j) + B(i,j)}{3}$$

Dimana:

I(i, j) = Intensitas citra *grayscale*

R(i, j) = Intensitas warna merah dari citra asal

G(i, j) = Intensitas warna hijau dari citra asal

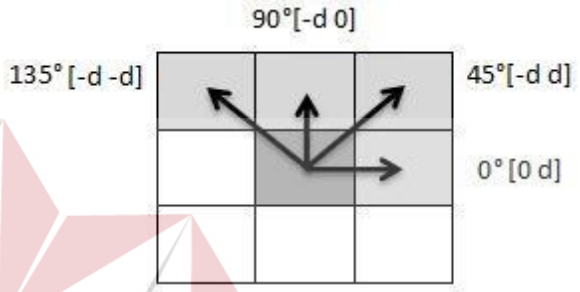
B(i, j) = Intensitas warna biru dari citra asal

Dalam sistem ini, *grayscalling* dilakukan pada citra data latih dan citra data uji. Untuk melakukan *grayscalling* di MATLAB, bisa menggunakan fungsi: I = *rgb2gray*(„variabel_citra‟). Gambar 2.6 adalah contoh hasil dari proses *grayscalling* dimana citra yang atas adalah citra yang bersifat RGB dan citra yang bawah adalah citra yang telah diubah menjadi *grayscale*.(Nauval,2016)

Gambar 2.6 Citra RGB (Atas) Dan Citra *Grayscale* (Bawah)

2.4 Gray Level Co-occurrence Matrix

Setelah dilakukan *grayscalling*, maka dicari matriks GLCM. *Gray Level Co-occurrence Matrix* (GLCM) merupakan metode yang paling sering digunakan untuk analisis tekstur. Dalam membuat matriks GLCM digunakan 2 variabel utama, yaitu orientasi sudut (θ) dan jarak piksel (d). Pada penelitian ini digunakan orientasi sudut (θ) 0˚, 45˚, 90˚, 135˚ dan jarak piksel (d) yang digunakan mulai dari 1 sampai 10 piksel.



Gambar 2.7 Ilustrasi Dari Orientasi Sudut Dan Jarak Piksel

Co-occurrence matriks adalah matriks bujur sangkar dengan jumlah elemen sebanyak kuadrat jumlah level intensitas piksel pada citra. Setiap titik (i,j) pada matriks kookurensi berorientasi berisi peluang kejadian piksel bernilai (i) bertetangga dengan piksel bernilai (j) pada jarak (d) serta orientasi (180 − θ) (Pradnyana I., 2015). Tahapan dalam membuat matriks kookurensi adalah:

1. Membuat area kerja matriks.
2. Menentukan hubungan spasial antara piksel referensi dengan piksel tetangga dengan memperhatikan nilai sudut (θ) dan jarak piksel (d).
3. Menghitung jumlah kookurensi dan mengisikannya pada area kerja matriks.
4. Menjumlahkan matriks kookurensi dengan transposenya untuk menjadikan simetris.

5. Normalisasi matriks untuk mengubahnya ke bentuk probabilitas.

Kookurensi berarti kejadian bersama, yaitu jumlah kejadian satu level nilai piksel bertetangga dengan satu level nilai piksel lain dalam jarak (*d*) dan orientasi sudut θ. Orientasi dibentuk dalam mepat arah sudut dengan interval 45° yaitu 0°,45°,90°, 135°. Sedangkan jarak antar piksel biasanya ditetapkan sebesar 1 piksel, 2 piksel, 3 piksel dan seterusnya.

Matriks kookurensi merupakn matriks bujur sangkar dengan jumlah elemen sebanyak kuadrat jumlah level internsitas piksel pada citra. Setiap titik (*i*,*j*) pada matriks kookurensi berorientasi berisi peluang kejadian piksel bernilai *i* bertetangga dengan piksel bernilai *j* pada jarak *d* serta orientasi (180- θ) (Pradnyana, 2015). Sebagai contoh matriks 4 x 4 memiliki tingkat keabuan dari 0 sampai 6. Matriks kookurensi akan dihitung dengan nilai d=1 dan θ=0°. Jumlah frekuensi munculnya pasangan (*i*,*j*) dihitung untuk keseluruhan matriks. Jumlah kookurensi diisikan pada matriks GLCM pada posisi sel yang bersesuaian. Gambar 2.8, 2.9, 2.10, 2.11 secara berurutan menunjukan contoh proses perhitungan matriks kookurensi. (Nauval, 2016)

4	3	2	1
0	4	3	3
5	4	0	0
1	1	5	5

Gambar 2.8 Matriks Bebas, Matriks I

Karena matriks I memiliki enam aras keabuan, maka jumlah nilai piksel tetangga dan nilai piksel referensi pada area kerja matriks berjumlah enam. Berikut adalah area kerja matriks.

Tabel 2.1 Area Kerja Matriks

Nilai Piksel Tetangga / Nilai Piksel Referensi	**0**	**1**	**2**	**3**	**4**	**5**
0	0,0	0,1	0,2	0,3	0,4	0,5
1	1,0	1,1	1,2	1,3	1,4	1,5
2	2,0	2,1	2,2	2,3	2,4	2,5
3	3,0	3,1	3,2	3,3	3,4	3,5
4	4.0	4,1	4,2	4,3	4,4	4,5
5	5,0	5,1	5,2	5,3	5,4	5,5

Hubungan spasial untuk d=1 dan θ=0° pada matriks diatas dapat dituliskan dalam matriks berkut:

Tabel 2.2 Matriks Co-occurrence

Nilai Piksel Tetangga / Nilai Piksel Referensi	0	1	2	3	4	5
0	1	0	0	0	1	0
1	0	1	0	0	0	1
2	0	1	0	0	0	0
3	0	0	1	1	0	0
4	1	0	0	2	0	0
5	0	0	0	0	1	1

Sudut orientasi menentukan arah hubungan tetangga dari piksel-piksel referensi, orientasi θ=0° berarti acuan dalam arah horizontal atau sumbu x positif dari piksel-piksel refrensi. Acuan sudut berlawanan arah jarum jam. Angka 2 pada (1,1) berarti jumlah hubungan pasangan (1,1) pada matriks asal berjumlah 2. Matriks kookurensi yang didapat kemudian ditambahkan dengan matriks transposenya untuk menjadikanya simestris terhadap sumbu diagonal. Berkut ini adalah (*i,j*) dari matriks asal ditambahkan dengan transposenya, dan hasilnya simetris seperti berikut :

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 & 2 & 0 \\ 0 & 2 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 & 2 & 0 \\ 2 & 0 & 0 & 2 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 2 \end{bmatrix}$$

2.5 Ekstraksi Fitur–Fitur GLCM

Setelah didapat matriks kookurensi GLCM, dilanjutkan dengan mencari nilai fitur-fitur dari GLCM. Dalam penelitian ini, fitur GLCM yang digunakan antara lain fitur *energy, contrast, correlation* dan *homogeneity* dari 4 orientasi sudut dengan jarak piksel 1 sampai 10. Berikut penjelasan dari masing-masing fitur (Hartadi, 2011).

2.5.1 *Contrast*

Contrast menunjukkan ukuran penyebaran (momen inersia) elemen-elemen matriks citra. Jika letaknya jauh dari diagonal utama, maka nilai kekontrasannya besar. Secara visual, nilai kekontrasan adalah ukuran variasi antar derajat keabuan suatu daerah citra.

$$\sum_{i,j=0}^{N-1} P(i,j)(i-j)2$$

Dimana :

P(i,j) = nilai elemen matriks kookurensi

2.5.2 *Correlation*

Correlation menunjukkan ukuran ketergantungan linear derajat keabuan citra sehingga dapat memberikan petunjuk adanya struktur linear dalam citra.

$$\sum_{i,j=0}^{N-1} P\ i,j\ [\frac{i - \mu x\ (j - \mu y)}{\sqrt{(\sigma x2)(\sigma y2)}}]$$

Dimana :

μx = nilai rata-rata elemen kolom pada matriks Pθ(i , j)

μy = nilai rata-rata elemen baris pada matriks Pθ(i , j)

σx = nilai standar deviasi elemen kolom pada matriks Pθ(i , j)

σy = nilai standar deviasi elemen baris pada matriks Pθ(i , j)

(Matlab, 2016)

2.5.3 *Energy*

Energy menunjukkan ukuran konsentrasi pasangan intensitas pada matriks kookurensi. Nilai *energy* makin membesar bila pasangan piksel yang memenuhi syarat matriks intensitas kookurensi terkonsentrasi pada beberapa koordinat dan mengecil bila letaknya menyebar.

$$\sum_{i,j} p(i,j)^2$$

Dimana :

P(i,j) = nilai elemen matriks kookurensi

Nilai energy makin membesar bila pasangan piksel yang memenuhi syarat matriks intensitas kookurensi terkonsentrasi pada beberapa koordinat dan mengecil bila letaknya menyebar

(Matlab, 2016)

2.5.4 *Homogeneity*

Homogeneity menunjukkan kehomogenan variasi intensitas dalam citra. Citra homogen akan memiliki nilai *homogeneity* yang besar. Nilai *homogeneity* membesar bila variasi intensitas dalam citra mengecil dan sebaliknya.

$$\sum_{i,j} \frac{p(i,j)}{1+|i-j|}$$

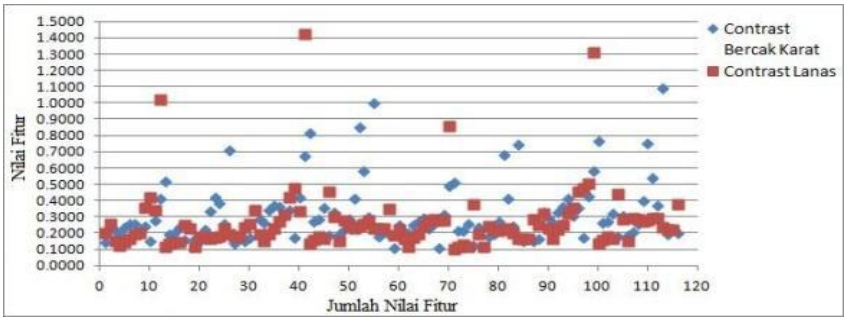
Dimana :

P(i,j) = nilai elemen matriks kookurensi

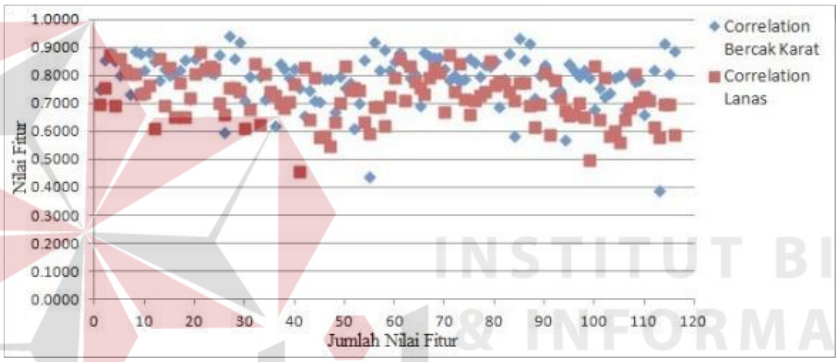
(Matlab, 2016)

2.5.5 Kelinearan Nilai Fitur GLCM

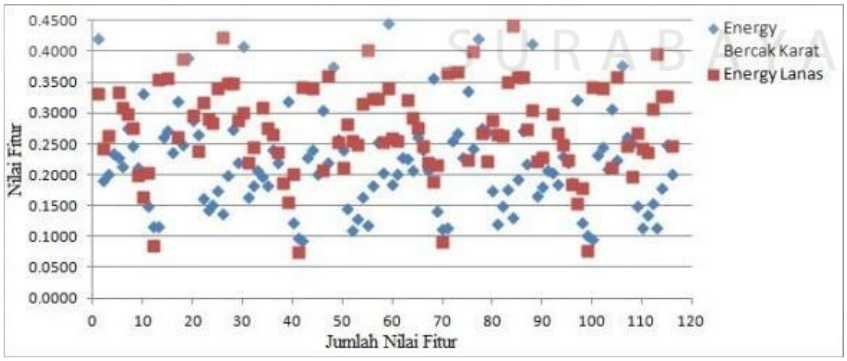
Setelah didapat nilai ekstraksi fitur GLCM, maka tahap selanjutnya adalah menentukan kelinearan citra tersebut dengan menggunakan *Microsoft Office Excel* secara manual dengan melihat pesebaran nilai fitur GLCM. Apabila dari hasil ekstraksi tersebut didapat bahwa citra tersebut bisa dipisahkan secara linear, maka penggunaan kernel untuk pelatihan SVM adalah kernel linear (*default*). Sedangkan jika hasil ekstraksi menunjukkan bahwa hasil tersebut tidak linear, maka diperlukan kernel untuk menyelesaikannya. Berikut pesebaran nilai fitur GLCM .



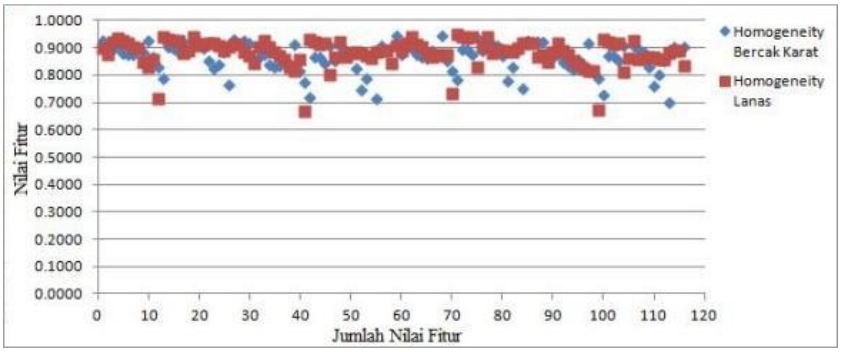
Gambar 2.12 Penyebaran Nilai Fitur *Contrast*



Gambar 2.13 Penyebaran Nilai Fitur *Correlation*



Gambar 2.14 Penyebaran Nilai Fitur *Energy*



Gambar 2.15 Penyebaran Nilai Fitur *Homogeneity*

Gambar 2.12 sampai gambar 2.15 merupakan Penyebaran dari nilai tiap fitur GLCM dari citra data latih. Dari gambar tersebut terlihat bahwa nilai fitur GLCM menunjukkan ketidaklinearan (tidak bisa dipisahkan secara linear).

2.6 Jaringan Saraf Tiruan (JST) *Backpropagation*

2.6.1 Pengertian

Perambatan galat mundur (*Backpropagation*) menurut Kiki (2004) adalah sebuah metode sistematik untuk pelatihan *multilayer* jaringan saraf tiruan. Metode ini memiliki dasar matematis yang kuat, obyektif dan algoritma ini mendapatkan bentuk persamaan dan nilai koefisien dalam formula dengan meminimalkan jumlah kuadrat galat *error* melalui model yang dikembangkan (*training set*).

1. Dimulai dengan lapisan masukan, hitung keluaran dari setiap elemen pemroses melalui lapisan luar.
2. Menghitung kesalahan pada lapisan luar yang merupakan selisih antara data aktual dan target.

3. Mentransformasikan kesalahan tersebut pada kesalahan yang sesuai di sisi masukan elemen pemroses.
4. Mempropagasi balik kesalahan-kesalahan ini pada keluaran setiap elemen pemroses ke kesalahan yang terdapat pada masukan. Ulangi proses ini
5. Mengubah seluruh bobot dengan menggunakan kesalahan pada sisi masukan elemen dan luaran elemen pemroses yang terhubung.

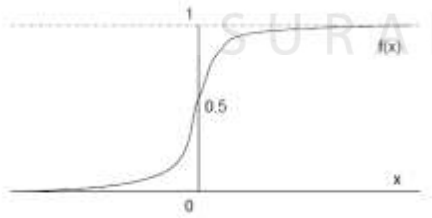
2.5.6 Arsitektur Model *Backpropagation*

Fungsi Aktivasi menurut Jong J.S: Syarat fungsi aktivasi yang dapat dipakai adalah kontinu, terdeferensial dengan mudah dan merupakan fungsi yang tidak turun Fungsi yang sering dipakai adalah:

- *sigmoid* biner yang memiliki range (0,1)

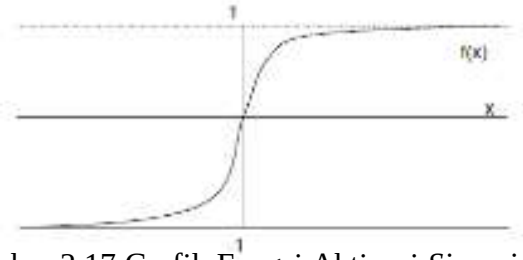
  Grafik fungsinya:

  $f(x) = 1/(1 + e^{-x})$ dengan turunan $f'(x) = f(x)(1 – f(x))$



Gambar 2.16 Grafik Fungsi Aktivasi *Sigmoid* Biner

- Fungsi *sigmoid* bipolar dengan *range* (1, -1)



Gambar 2.17 Grafik Fungsi Aktivasi *Sigmoid* Bipolar

Grafik fungsinya:

$f(x) = 2/(1 + e^{-x})-1$, $f'(x) = (1+f(x))(1-f(x))/2$

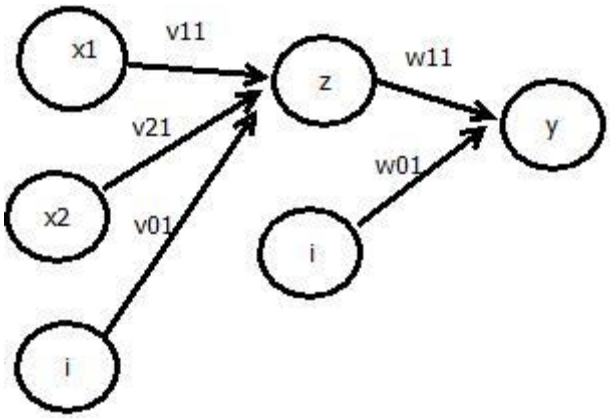
Adapun pelatihan standar *backpropagation* menurut Jong, J.S. yang terdapat pada gambar2.17 adalah:

Meliputi 3 fase, maju, mundur, dan modifikasi bobot

- Fase I Propagasi maju, sinyal masukan(x_i) dikalikan dengan bobot garis(w), kemudian dipropagasikan ke *hidden layer* menggunakan fungsi aktivasi(f(x)) yang ditentukan. Keluaran dari setiap unit *hidden*(z_j) selanjutnya dipropagasikan maju lagi ke layar *hidden* diatasnya menggunakan fungsi aktivasi yang ditentukan, demikian seterusnya hingga menghasilkan keluaran jaringan (y_k). Berikutnya, keluaran jaringan dibandingkan dengan target yang harus dicapai (t_k). Selisih (t_k - y_k) adalah kesalahan yang terjadi. Jika kesalahan ini lebih kecil dari batas toleransi maka iterasi dihentikan, tetapi bila kesalahan masih lebih besar maka bobot setiap garis(w) dalam jaringan akan dimodifikasi untuk mengurangi kesalahan yang terjadi.

- Fase II Propagasi mundur, Berdasarkan kesalahan (t_k - y_k), dihitung factor δ_k(k=1,2,3,..,m) yang dipakai untuk mendistribusikan kesalahan di unit (y_k) ke semua unit *hidden* yang terhubung langsung dengan y_k. δ_k juga dipakai untuk mengubah bobot garis(w) yang berhubungan langsung dengan unit keluaran. Dengan cara yang sama, dihitung faktor δ_j di setiap unit di *hidden layer* sebagai dasar perubahan bobot semua garis yang berasal dari unit tersembunyi di layar di bawahnya. Demikian seterusnya hingga semua faktor δ di unit *hidden* yang berhubungan langsung dengan unit masukan dihitung
- Fase III Perubahan bobot, bobot semua garis dimodifikasi bersamaan. Perubahan bobot suatu garis didasarkan atas faktor δ neuron di layar atasnya. Sebagai contoh, perubahan bobot garis yang menuju ke layar keluaran didasarkan atas δ_k yang ada di unit keluaran. Fase tersebut diulang hingga penghentian terpenuhi. Umumnya kondisi penghentian yang dipakai adalah jumlah iterasi atau kesalahan. (Joseph, 2016)

Berikut contoh perhitungan manual algoritma *backpropagation*: Misalnya sebuah jaringan terdiri atas dua unit input, satu unit tersembunyi, dan satu unit keluaran. Fungsi aktivasi yang digunakan adalah Sigmoid Biner, learning rate / alpha (α) = 0.01, toleransi *error* yang diperkenankan adalah 0.41. Jaringan digunakan untuk menyelesaikan fungsi XOR. Pada Gambar 2.17 menunjukkan arsitektur jaringan yang akan dilatih.



Gambar 2.18 Contoh Jaringan Arsitektur Neural network

Gambar 2.18 Arsitektur Jaringan Pada Contoh Adapun data training yang digunakan terdiri atas empat pasang masukkan dan keluaran yakni:

Tabel 2.3 Data *Training* Contoh

No	Masukkan 1	Masukkan 2	Keluaran
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	0

Langkah-langkah pada proses training adalah sebagai berikut:

Langkah 0 : Inisialisasi Sembarang bobot dan bias, misalnya $v01 = 1{,}718946$ $v11 = -1{,}263178$ $v21 = -1{,}083092$ $w01 = -0{,}541180$ $w11 = 0{,}543960$

Langkah 1 : Dengan bobot sembarang tersebut, tentukan *error* untuk data training secara keseluruhan dengan rumus sebagai berikut:

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

$$z_j = f(z\_in_j)$$

$z\_in11 = 1{,}718946 + 0 \times -1{,}263178 + 0 \times -1{,}083092 = 1{,}718946$

$z11 = f\ z\_in11 = 0{,}847993$

$z\_in12 = 1{,}718946 + 0 \times -1{,}263178 + 1 \times -1{,}083092 = 0{,}635854$

$z12 = f\ z\_in12 = 0{,}653816$

$z\_in13 = 1{,}718946 + 1 \times -1{,}263178 + 0 \times -1{,}083092 = 0{,}455768$

$z13 = f\ z\_in13 = 0{,}612009$

$z\_in14 = 1{,}718946 + 1 \times -1{,}263178 + 1 \times -1{,}083092 = -0{,}627324$

$z14 = f\ z\_in14 = 0{,}348118$

Dimana indeks zjn berarti bobot untuk unit tersembunyi ke-j dan data training ke-n

$y\_ink = w0k + \sum_{i=1}^{n} z_j w_{jk}$

$yk = f(y\_ink)$

$y\_in11 = -0{,}541180 + (0{,}847993 \times 0{,}543960) = 0{,}079906$

$y11 = f\ y\_in11 = 0{,}480034$

$y\_in12 = -0{,}541180 + 0{,}653816 \times 0{,}543960 = -0{,}185530$

$y12 = f\ y\_in12 = 0{,}453750$

$y\_in13 = -0{,}541180 + (0{,}612009 \times 0{,}543960) = 0{,}208271$

$y13 = f\ y\_in13 = 0{,}448119$

$y\_in14 = -0{,}541180 + 0{,}348118 \times 0{,}543960 = -0{,}351818$

$y14 = f\ y\_in14 = 0{,}412941$

Sehingga, $E = 0{,}5 \times \{\ 0 - 0{,}480034\ 2 + 1 - 0{,}453750\ 2 + 1 - 0{,}448119\ 2 + 0 - 0{,}412941\ 2\ \} = 0{,}501957$

Langkah 2 : Karena, data *error* training masih lebih besar dari toleransi yakni 0.41. Maka, pelatihan dilanjutkan pada langkah 3-8

Langkah 3 : $x1 = 0$, $x2 = 0$; (Training untuk data pertama)

Langkah 4 : $z\_in11 = 1{,}718946 + 0 \times -1{,}263178 + 0 \times -1{,}083092 = 1{,}718946$

$z11 = f\ z\_in11 = 0{,}847993$

Langkah 5 : $y\_in11 = -0{,}541180 + (0{,}847993 \times 0{,}543960) = 0{,}079906$

$y11 = f\ y\_in11 = 0{,}480034$

Langkah 6 : $\delta 1 = 0 - 0{,}480034\ f\ '\ 0{,}079906 = -0{,}119817$

$\Delta w11 = 0{,}01 \times -0{,}119817 \times 0{,}847993 = -0{,}001016$

$\Delta w01 = 0{,}01 \times -0{,}119817 = -0{,}00119817$

Langkah 7 : $\delta\_in1 = -0{,}119817 \times 0{,}543960 = -0{,}065176$

$\delta 1 = -0{,}065176 \times f\ '\ 1{,}718946 = -0{,}008401$

$\Delta v11 = 0{,}01 \times -0{,}008401 \times 0 = 0$

$\Delta v21 = 0{,}01 \times -0{,}008401 \times 0 = 0$

$\Delta v01 = 0{,}01 \times -0{,}008401 = -0{,}00008401$

Langkah 8 : $w01\ baru = -0{,}541180 + -0{,}00119817 = -0{,}542378$

$w11\ baru = 0{,}543960 + -0{,}001016 = 0{,}542944$

$v01\ baru = 1{,}718946 + -0{,}00008401 = 1{,}718862$

$v11\ baru = -1{,}263178 + 0 = -1{,}263178$

$v21\ baru = -1{,}083092 + 0 = -1{,}083092$

Setelah langkah 3-8 untuk data training pertama dikerjakan, ulangi kembali langkah 3-8 untuk data training ke-2,3 dan 4. Setelah seluruh data training dikerjakan itu berarti satu iterasi telah diproses. Bobot yang dihasilkan pada iterasi pertama untuk data training ke-2,3, dan 4 adalah:

Data Training ke-2

$w01 = -0{,}541023$

$w11 = 0{,}543830$

$v01 = 1{,}718862$

$v11 = -1{,}263178$

$v21 = -1{,}083092$

Data Training ke-3

$w01 = -0{,}539659$

$w11 = 0{,}544665$

$v01 = 1{,}719205$

$v11 = -1{,}263002$

$v21 = -1{,}082925$

Data Training ke-4

$w01 = -0{,}540661$

$w11 = 0{,}544316$

$v01 = 1{,}719081$

$v11 = -1{,}263126$

$v21 = -1{,}083049$

Setelah sampai pada data training ke-4, maka iterasi pertama selesai dikerjakan. Proses training dilanjutkan pada langkah ke-9 yaitu memeriksa kondisi STOP dan kembali pada langkah ke-2. Demikian seterusnya sampai *error* yang dihasilkan memenuhi toleransi *error* yang ditentukan. Setelah proses training selesai, bobot akhir yang diperoleh untuk contoh XOR adalah sebagai berikut:

$w01 = -5{,}018457$

$w11 = 5{,}719889$

$v01 = 12{,}719601$

$v11 = -6{,}779127$

$v21 = -6{,}779127$

Jika terdapat masukkan baru, misalnya $x1 = 0{,}2$ dan $x2 = 0{,}9$ maka keluarannya dapat dicari dengan menggunakan langkah-langkah umpan maju berikut ini:

Langkah 0 : Bobot yang digunakan adalah bobot akhir hasil pelatihan di atas.

Langkah 1 : Perhitungan dilakukan pada langkah 2 – 4.

Langkah 2 : Dalam contoh ini, bilangan yang digunakan telah berada dalam interval 0 dan 1, jadi tidak perlu diskalakan lagi.

Langkah 3 : $zin\ 1 = 12{,}719601 + 0{,}2 \times -6{,}779127 + 0{,}9 \times -6{,}779127 = 5{,}262561$ $z1 = f\ z\_in1 = 0{,}994845$

Langkah 4 : $y\_in1 = -5{,}018457 + (5{,}719889 \times 0{,}994845) = 0{,}671944$

$y1 = f\ y\_in1 = 0{,}661938$.

Jadi, jika input data adalah $x1 = 0{,}2$ dan $x2 = 0{,}9$; output jaringan yang dihasilkan adalah 0,661938. Dengan menunjukkan perhitungan manual algoritma

backpropagation, sangat tidak memungkinkan untuk melakukan perhitungan manual pada penelitian prediksi jumlah dokter keluarga dengan menggunakan 78 data training dan 13 data testing, karena itu diperlukan perangkat lunak untuk dapat melakukan proses komputasi pada penelitian ini.