BAB II
TINJAUAN PUSTAKA

2.1 Pepaya (Carica Papaya)

Pepaya (Carica papaya) atau betik adalah tumbuhan yang berasal dari genus Carica yang berasal dari Meksiko bagian selatan dan bagian utara Amerika Selatan, namun kini sudah tersebar luas dan banyak ditanam di seluruh daerah tropis untuk diambil buahnya.

Buah pepaya juga bermanfaat untuk mengobatan bermacam macam penyakit, seperti memperlancarkan pencernaan, menurunkan kolesterol, antioksidan, menghilangkan rasa lelah dan lesu, demam berdarah, dan sariawan. Vitamin C berperan sebagai antioksidan yang berguna untuk melawan serangan radikal bebas penyebab penuaan dini dan berbagai penyakit kanker.

Buah pepaya mengalamai berubahan warna yang nyata selama proses pematangan, yang menunjukan terjadinya perubahan-perubahan secara kimiawi dalam buah. Perubahan warna pepaya dari hijau menjadi kuning disebabkan hilangnya krolofil. Selama proses penyimpanan awalnya buah pepaya berwarna hijau, kemudia berubah menjadi sedikit kuning. Semakin lama penyimpanan warna berubah menjadi kuning matang.

2.2 Penelitian - Penelitian Terdahulu

Beberapa penelitian sebelumnya telah dilakukan untuk menyelesaikan permasalahan klasifikasi yang hampir sama dengan klasifikasi jenis jamur penyebab penyakit antraknos pada buah cabai. Penelitian pertama dilakukan oleh (Desawari et al. 2013) melakukan penelitian tentang identifikasi buah tomat dengan menggunakan metode backpropagation. Hasil dari penelitian ini adalah algoritma backpropagation mampu memberikan hasil identifikasi hingga 71,76%. Pada penelitian ini peneliti melakukan pengidentifikasian kematangan buah tomat yang ditanam pada rumah kaca dengan bantuan webcam dan menggunkan space warna RGB.

Selain itu Liana Fitriani Nunuhitu membahas tentang identifikasi penyakit pada daun cabai serta cara mengatasinya dengan menggunakan metode Laplacian

of Gaussian. Pada penelitian ini, peneliti mengidentifikasipenyakit yang menyerang tumbuhan cabai dengan cara melihatbentuk dari noda yang terdapat pada daun (2011).

Penelitian berikutnya berkaitan dengan metode yang akan digunakan oleh penulis, yaitu penelitian yang dilakukan oleh Nila Anggraini klasifikasi kanker serviks menggunakan jaringan syaraf tiruan backpropagation dengan Graphical User Interface (GUI). Hasil dari penelitian ini mempunyai akurasi rata-rata 85% (Anggraini, 2015).

Selanjutnya adalah penelitian tentang image enhancement yang dilakukan oleh (Dinata, 2014) untuk implementasi metode Multiscale Retinex. Penelitian ini dilakukan untuk normalisasi iluminasi citra.

Santoso, et al (2007) menerapkan model backpropagation dengan struktur jaringan 2-5-4-1-2 untuk meramalkan banyaknya permintaan karet sebagai komoditas pada PT. Perkebunan Nusantara XII Surabaya dan perbandingan tingkat akurasi metode peramalan dengan data pengujian didapatkan persentase kesalahan absolute (MAPE) adalah 17,54%.

Metode backpropagation digunakan oleh Andrijasa dan Mistianingsih (2010) untuk memprediksi jumlah pengangguran di Provinsi Kalimantan Timur dan hasil pengujian di peroleh prediksi jumlah pengangguran Tahun 2009 adalah 133.104 sedangkan hasil prediksi pengangguran Tahun 2009 yang dilakukan oleh BPS Provinsi Kalimantan Timur adalah 139.830.

Pengembangan aplikasi dengan menggunakan metode backpropagation digunakan juga oleh Mulyana (2008) untuk meramalkan tingkat penjualan dan diperoleh tingkat penyimpangan rata-rata sebesar 3.3%.

2.3 Ekstraksi Fitur

Ekstraksi fitur (konten) adalah dasar dari retrivalcitra berbasis konten. Ekstraksi fitur dapat diklasifikasikan sebagai fitur-fitur umum dan fitur dengan spesifik domain. Klasifikasi pertama mencakup fitur warna, tekstur, dan bentuk sedangkan klasifikasi kedua termasuk fitur-fitur yang merupakan fitur spesifik aplikasi sebagai contoh, fitur untuk wajah manusia dan siding jari (Rui, Huang, & Chang, 1999). Fitur-fitur yang digunakan dalam penelitian initermasuk kedalam

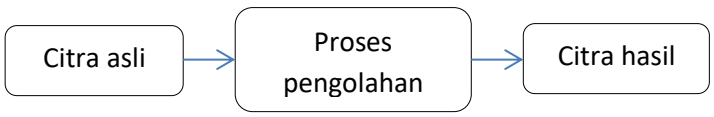
fitu-fitur umum sehingga pembahasan hanya akan mencakup dibagian tersebut saja, fitur-fitur umum tersebut adalah (Rui, Huang, & Chang, 1999).

Warna merupakan salah satu fitur visual yang paling banyak digunakan dalam retrival citra., dan histogram warna merupakan fitur paling umum diginakan di dalam representasi fitur warna. Secara statistik, histogram melakukan ekstraksi terhadap probabilitas join dari intensitas tiga saluran warna (RGB). Salah satu fitur dalam penelitian ini menggunakan histogram dan penggunaannya akan dijelaskan lebih lanjut.

2.4 Pengolahan Citra

Pengolahan citra adalah kegiatan memperbaiki citra agar mudah diinterprestasi oleh manusia atau mesin (komputer). Inputnya adalah citra dan outputnya citra tetapi dengan kualitas lebih baik dari pada citra masukan, misal citra warnanya kurang tajam, kabur (bluring), mengandung noise dan lain-lain sehingga perlu ada pemprosesan untuk memperbaiki citra karena citra tersebut menjadi sulit diinterprestasikan karena informasi yang disampaikan menjadi kurang.

Pengolahan citra bertujuan memperbaiki kualitas agar mudah diinterprestasi oleh manusia atau mesin (dalam hal ini komputer). Teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra, namun citra keluarannya harus mempunyai kualitas lebih baik daripada citra masukan.



Gambar 2.1 Contoh Proses Pengolahan Citra

2.5 Representasi Warna RGB

Warna merupakan respon physiological dan intensitas yang berbeda. Persepsi warna dalam pengolahan citra tergantung pada tiga faktor, yaitu spectral reflectance (menentukan bagaimana suatu permukaan memantulkan warna), spectral content (kandungan warna dari cahaya yang menyinari permukaan) dan

spectral response (kemampuan merespon warna dari sensor dalam imaging system).

Representasi warna ini terdiri dari tiga unsur utama yaitu merah (red), hijau (green), dan biru (blue). Gabungan tiga warna ini membentuk warna-warna lainnya berdasarkan intensitas dari masing-masing warna tersebut dengan intesitas maksimal, dan warna hitam merupakan gabungan dari ketiga warna tersebut dengan intensitas minimal.

Dalam tugas akhir ini menggunakan model warna RGB. Tingkat RGB pola bit dikomposisikan dari tiga warna tersebut dan masing-masing warna mempunyai 28 atau 256 bit (0 - 255). Model warna RGB yang dapat dinyatakan dalam bentuk indeks warna RGB dengan cara menormalisasi setiap komponen warna. dengan persamaan sebagai sebagai berikut :

$$R = \frac{R}{R+G+B} \quad (2.1)$$

$$G = \frac{G}{R+G+B} \quad (2.2)$$

$$B = \frac{B}{R+G+B} \quad (2.3)$$

Proses Penghitungan Nilai Pixel Citra:

Untuk melakukan penyisipan pesan ke dalam file citra, terlebih dahulu dilakukan penghitungan nilai pixel (Px) citra dengan rumus:

Nilai R = Px Mod 256 (2.4)

Nilai G = (Px \ 256) Mod 256 (2.5)

Nilai B = (Px \ 256 \ 256) Mod 256 (2.6)

Sebagai contoh nilai piksel (1,1) citra adalah 111100001111000011111111. Nilai komponen Red (R) dilakukan perhitungan modulo dengan bilangan 256 dengan nilai ASCII 10000000 sebagai berikut:

Nilai komponen R dihitung dengan persamaaan (2.4) = nilai piksel Blok-1 mod 10000000

R = 111100001111000011111111 mod 10000000

R = 11111111

Nilai komponen Green (G) dihitung dengan persamaan (2.5):

G = (111100001111000011111111\10000000) mod 100000000

G = 11110000

Nilai komponen Blue (B) dihitung dengan persamaan (2.6):

B = (11110000111100001111111\10000000\10000000) mode 10000000 =

B = 11110000

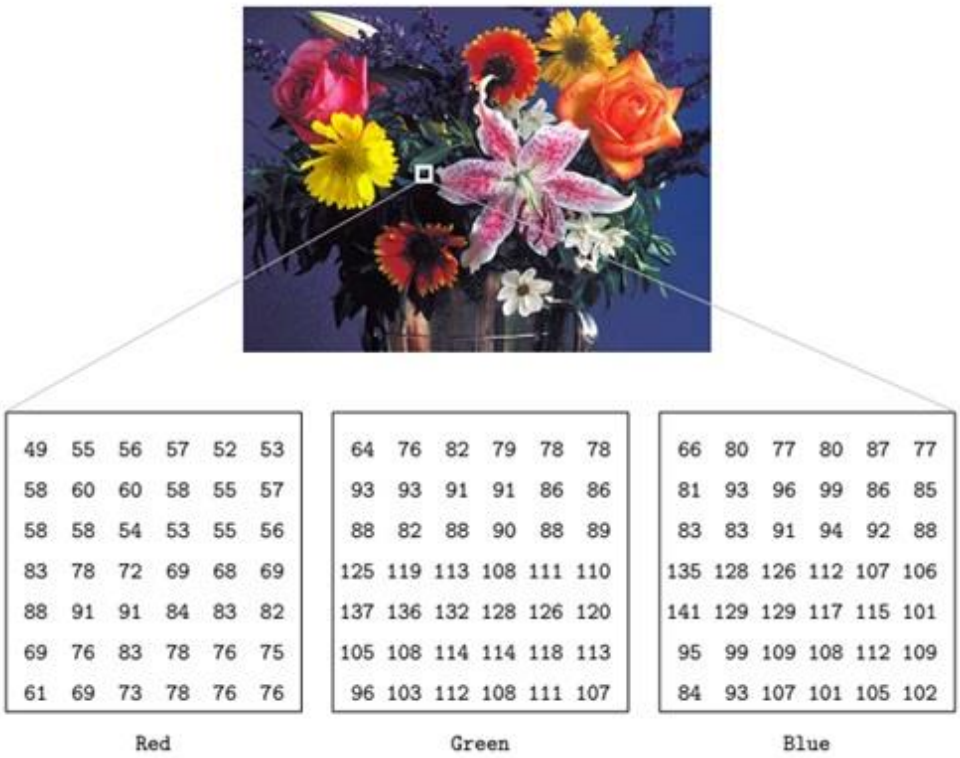
Sehingga diperoleh nilai RGB piksel citra frame-1 pada Blok-1 adalah: 11110000 11110000 11111111:

R = 11111111= 254, G = 11110000 = 240 dan B = 11110000 = 240

Sehingga diperoleh nilai piksel (1,1) dengan komponen RGB = (254,240,240). Untuk perhitungan nilai piksel selanjutnya adalah sama dengan di atas dan hasil perhitungan digambarkan seperti Gambar 3.5.

254,240,240	200,200,45	210,230,101	190, 200,20	100, 190,100
169,190,210	187,180,74	50,120,56	150,150,200	100,150,10
145,200,67	201,160,55	80,110,60	150,190,205	125,190,157
178,204,45	196,120,50	70,100,45	150,200,158	180,200,158
187,200,120	106,100,150	170,100,100	150,210,133	90,100,138

Gambar 2.2 Matriks Citra RGB



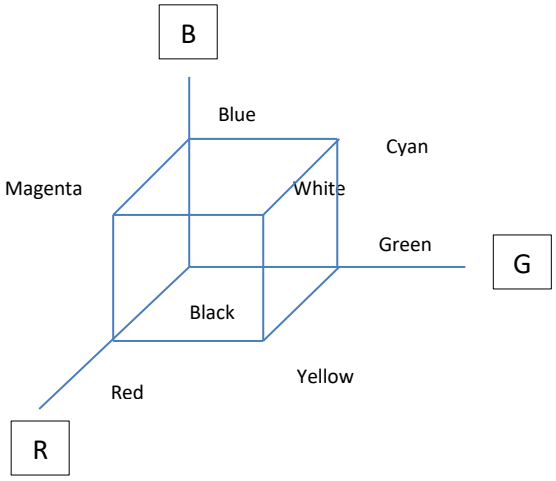
Gambar 2.3 Contoh Citra RGB

2.6 Ekstraksi Histogram

Histogram warna atau color histogram adalah representasi distribusi warna dalam sebuah gambar yang didapatkan dengan menghitung jumlah pixel dari setiap bagian range warna, secara tipikal dalam dua dimensi atau tiga dimensi. Dalam pembuatan histogram, nilai RGB mempunyai range dari 0 sampai 255 akan punya kemungkinan kombinasi warna sebesar 16777216 (didapat dari: 255 x 255 x 255).

Retrival citra berbasis konten memainkan peranan penting dalam area aplikasi khususnya sistem database multimedia dalam beberapa tahun terakhir. Pekerjaan terfokus dalam penggunaan fitur-fitur tingkat rendah seperti warna untuk representasi gambar. Diantara fitur visual, warna mungkin adalah salah satu yang paling membedakan di banyak aplikasi (Suhasini, krishna, & Krishna 2009).

Isi visual dari gambar seperti warna banyak digunakan dalam retrival citra berbasis konten. Warna adalah salah satu fitur visual yang paling dapat diandalkan yang juga lebih mudah untuk diterapkan dalam sistem pengambilan gambar. Warna tidak tergantung pada ukuran dan orientasi gambar. Karena warna kuat untuk komplikasi latar belakang. Histogram warna adalah teknik paling umum untuk mengekstraksi fitur warna dari gambar berwarna. Pada kasus retrival citra histogram warna banyak digunakan untuk sistem retrival citra berbasis konten. Hal ini dikarenakan metode ini adalah metode paling umum untuk memprediksi karakteristik dari gambar (Sai & Patil, 2010).



Gambar 2.4 Ruang Warna RGB (Candan & Sapino, 2010, p35)

Contoh perhitungan histogram

- RGB → per plane warna
- Plotting dari persamaan:

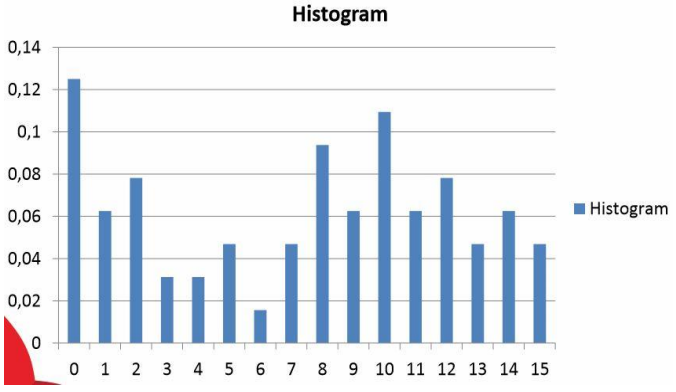
$$p_r(r_k) = \frac{n_k}{n}; \quad 0 \le r_k \le 1; \quad k = 0,1,\dots,L-1 \qquad (2.7)$$

  - L: jumlah level
  - $p_r(r_k)$: probabilitas kemunculan level ke-k
  - n_k : jumlah kemunculan level k pada citra
  - n : total jumlah pixel dalam citra

misalkan matriks dibawah ini menyatakan citra digital yang berukuran **8 x 8** ***pixel*** dengan derajat keabuan dari 0 sampai 15 (ada 16 buah derajat keabuan → 16 level).

$$\begin{bmatrix} 3 & 7 & 7 & 8 & 10 & 12 & 14 & 10 \\ 2 & 0 & 0 & 0 & 1 & 8 & 15 & 15 \\ 14 & 6 & 5 & 9 & 8 & 10 & 9 & 12 \\ 12 & 12 & 11 & 8 & 8 & 10 & 11 & 1 \\ 0 & 2 & 3 & 4 & 5 & 13 & 10 & 14 \\ 4 & 5 & 0 & 0 & 1 & 0 & 2 & 2 \\ 15 & 13 & 11 & 10 & 9 & 9 & 8 & 7 \\ 2 & 1 & 0 & 10 & 11 & 14 & 13 & 12 \end{bmatrix}$$

k	n_k	$P_r(r_k)$
0	8	0.125
1	4	0.0625
2	5	0.078125
3	2	0.03125
4	2	0.03125
5	3	0.046875
6	1	0.015625
7	3	0.046875
8	6	0.09375
9	4	0.0625
10	7	0.109375
11	4	0.0625
12	5	0.078125
13	3	0.046875
14	4	0.0625
15	3	0.046875



Gambar 2.5 Contoh Histogram

2.7 Pengenalan Pola

Menurut Gonzalez dan Woods, sebagaimana dikutip oleh Abidin (2010), pola adalah suatu kuantitatif atau gambaran struktural dari suatu objek atau beberapa entitas yang menarik dari suatu citra. Secara umum, pola terbentuk oleh satu atau lebih ciri citra. Dengan kata lain, pola adalah susunan dari ciri-ciri citra. Ciri-ciri yang dapat diperoleh dari suatu citra, misalnya histrogram, arah, dan magnitudo tepi, warna, luas daerah dan sebagainya. Ilmu yang mempelajari klasifikasi atau penggambaran pola dari suatu objek berdasarkan ciri-cirinya adalah pengenalan pola. Pengenalan pola dibedakan menjadi tiga, yaitu:

2.7.1 Deteksi

Deteksi adalah usaha untuk menemukan keberadaan, anggapan, atau kenyataan (Poerwadarminta, 2007). Deteksi umumnya berkaitan dengan segmentasi dan proses thresholding, misalnya dalam mendeteksi daun pada suatu gambar, maka benda yang berwarna hijau akan terdeteksi sebagai daun (Rupam, 2011).

2.7.2 Klasifikasi

Klasifikasi adalah proses menemukan sekumpulan model/ fungsi yang menjelaskan dan membedakan data ke dalam kelas-kelas tertentu dengan tujuan menggunakan model tersebut dalam menentukan kelas dari suatu obyek yang belum diketahui kelasnya (Maharani, 2009). Misalnya dalam pengklasifikasian mobil.

2.7.3 Pengenalan

Pengenalan pola bertujuan menentukan kelompok atau kategori pola berdasarkan ciri-ciri yang dimiliki oleh pola tersebut. Tujuan pengelompokan adalah untuk mengenali suatu objek dalam citra (Sari, 2010). Misalnya, dalam mengenali suatu wajah pada gambar, maka wajah akan dideteksi dan diproses untuk dibandingkan dengan database wajah yang dikenal sebelumnya untuk menentukan siapa orang tersebut.

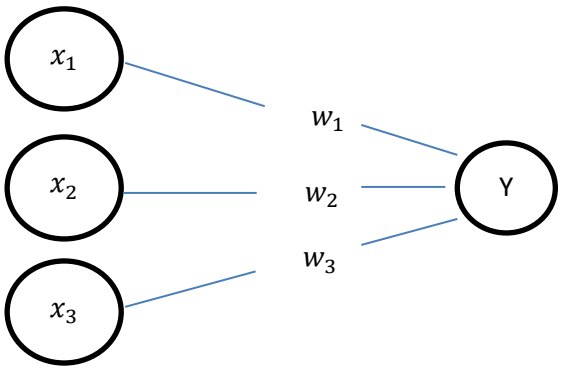
2.8 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan (JST) diinspirasikan oleh struktur jaringan sel-sel syaraf di dalam otak. Menurut Haykin, S. (1994), ”Sebuah jaringan saraf adalah sebuah prosesor yang terdistribusi paralel dan mempunyai kecenderungan untuk menyimpan pengetahuan yang didapatkannya dari pengalaman dan membuatnya tetap tersedia untuk digunakan. Hal ini menyerupai kerja otak dalam dua hal yaitu:

1. Pengetahuan diperoleh oleh jaringan melalui suatu proses belajar.
2. Kekuatan hubungan antar sel saraf yang dikenal dengan bobot sinapsis digunakan untuk menyimpan pengetahuan.

JST merupakan suatu model komputasi yang meniru cara kerja sistem otak manusia. JST merupakan sebuah model komputasi dari otak manusia yang mampu melakukan perhitungan, pengenalan, pengamatan serta pengambilan keputusan. Jaringan syaraf tiruan memanfaatkan struktur pengolahan paralel atas sejumlah pengolah sederhana dan hubungan antar pengolah tersebut.

Dalam jaringan syaraf tiruan, terdapat elemen pengolah yang merupakan model dari neuron. Setiap hubungan elemen pengolah menentukan kemampuan JST. Seperti halnya jaringan syaraf biologis, JST juga memiliki kemampuan untuk belajar dan beradaptasi terhadap masukan-masukannya. JST tidak perlu diprogram secara eksplisit, karena JST dapat belajar dari beberapa contoh pelatihan. Sebagai contoh, perhatikan neuron Y pada gambar berikut :



Gambar 2.6 Contoh Arsitektur

JST ditentukan oleh 3 hal :

1. Pola hubungan antar neuron (arsitektur).

2. Metode menentukan bobot penghubung (metode training/learning/algoritma).
3. Fungsi aktivasi

Y menerima input dari neuron x_1, x_2 dan x_3 dengan bobot hubungan masing-masing adalah w_1, w_2 dan w_3. Ketiga impuls neuron yang ada dijumlahkan:

net = $x_1w_1 + x_2w_2 + x_3w_3$

Besarnya impuls yang diterima oleh Y mengikuti fungsi aktivasi :

y = f(net)

Apabila nilai fungsi aktivasi cukup kuat, maka sinyal akan diteruskan. Nilai fungsi aktivasi (keluaran model jaringan) juga dapat dipakai sebagai dasar untuk merubah bobot. Beberapa arsitektur jaringan yang sering dipakai dalam jaringan syaraf tiruan antara lain :

1. Jaringan Layar Tunggal (single layer network). Dalam jaringan ini, sekumpulan input neuron dihubungkan langsung dengan sekumpulan outputnya. Dalam beberapa model hanya ada sebuah unit neuron output. Contoh dapat dilihat pada gambar 2.1.
2. Jaringan Layar Jamak (multi layer network) Jaringan ini merupakan perluasan dari layar tunggal. Dalam jaringan ini, selain unit input dan output, ada unitunit lain (layar tersembunyi). Dimungkinkan pula ada beberapa layar tersebunyi. Sama seperti pada unit input dan output, unit-unit dalam satu layar tidak saling berhubungan. Contoh dapat dilihat pada gambar 4, simbol z merupakan layar tersembunyi.

Ada banyak model arrsitektur JST dimulai dari yang sangat sederhana dan analitik yang ditemukan oleh McCullochPitts, Jaringan Hebb hingga jaringan dengan supervisi yang umum dipakai seperti Perceptron, ADALINE serta Back Propagation.

2.9 Komponen Jaringan Syaraf Tiruan

Pada umumnya JST memiliki dua lapisan, yaitu input layer dan output layer. Tetapi pada perkembangannya, adapula JST yang memiliki satu lapisan lagi yang terletak di antara input layer dan output layer. Lapisan ini disebut lapisan hidden layer. Menurut Halim et all. (2004: 12), berikut penjelasan mengenai komponen JST.

2.9.1 Input Layer

Input layer berisi node-node yang masing-masing menyimpan sebuah nilai masukan yang tidak berubah pada fase latih dan hanya bisa berubah jika diberikan nilai masukan baru. Node pada lapisan ini tergantung pada banyaknya input dari suatu pola.

2.9.2 Hidden Layer

Lapisan ini tidak pernah muncul sehingga dinamakan hidden layer. Akan tetapi semua proses pada fase pelatihan dan fase pengenalan dijalankan di lapisan ini. Jumlah lapisan ini tergantung dari arsitektur yang akan dirancang, tetapi pada umumnya terdiri dari satu lapisan hidden layer.

2.9.3 Output Layer

Output layer berfungsi untuk menampilkan hasil perhitungan sistem oleh fungsi aktivasi pada lapisan hidden layer berdasarkan input yang diterima.

2.10 Jaringan Saraf Tiruan Backpropagation

Secara umum, jaringan seperti ini terdiri dari sejumlah unit neuron sebagai lapisan input, satu atau lebih lapisan simpul-simpul neuron komputasi hidden (lapisan tersembunyi), dan sebuah lapisan simpul-simpul neuron komputasi output. Sinyal input dipropagasikan ke arah depan (arah lapisan output), lapisan demi lapisan. Jenis jaringan ini adalah hasil generalisasi dari arsitektur perceptorn satu lapisan, jadi biasa disebut sebagai multilayer perceptorn (MLPs).

Error back propagation adalah algoritma MLPs yang menggunakan prinsip supervised learning. Propagasi balik (ke arah lapisan input) terjadi setelah jaringan menghasilkan output yang mengandung error. Pada fase ini seluruh bobot synaptic (yang tidak memiliki aktivasi nol) dalam jaringan akan disesuaikan untuk mengkoreksi/memperkecil error yang terjadi (error correction rule). Untuk pelatihan jaringan, pasangan fase propagasi ke depan dan balik dilakukan secara berulang untuk satu set data latihan, kemudian diulangi untuk sejumlah epoch (satu sesi lewatan untuk seluruh data latihan dalam sebuah proses pelatihan jaringan) sampai error yang terjadi mencapai batas kecil toleransi tertentu atau nol.

Fungsi aktivasi yang digunakan pada arsitektur ini adalah yang menghasilkan nilai kontinu, jadi output jaringan juga akan bernilai kontinu. Fungsi aktivasi yang umum digunakan disini adalah sigmoidal (biner atau bipolar, mengacu pada pembuatan program, akan ditunjukkan yang bipolar, range : ([-1,1]) :

$$f = (v_n) = y_n = \frac{2}{1+\exp(-v_n)} - 1 \quad (2.8)$$

y_j adalah nilai aktivasi setiap neuron/node, dan v_j adalah fungsi integrasi untuk setiap neuron, dimana :

$$v_n = \sum_m w_{mn} y_{mn} - \theta_n \quad (2.9)$$

w_{ij} adalah bobot synaptic link-link antarneuron, dan θ adalah treshold (nilai ambang) yang dimiliki setiap neuron. Untuk menyederhanakan perhitungan, pada setiap lapisan (input, hidden, dan output) diberikan neuron tambahan yang selalu memiliki nilai aktivasi 1, dan bobot synaptic setiap link-nya dengan neuron tertentu sama dengan nilai treshold neuron tersebut. Sehingga v_n menjadi :

$$v_n = \sum_m w_{mn} y_{mn} \quad (2.10)$$

Untuk pelatihan, diberikan satu set data latihan (s t) dimana s adalah input dan t adalah target (output) yang valid untuk s. Error terjadi bila output yang dihasilkan jaringan dengan bobot synaptic yang digunakan saat itu tidak sama dengan target (pada satu atau lebih neuron output).

Jumlahan kuadrat sinyal error digunakan sebagai acuan untuk melihat apakah jaringan sudah terlatih dengan baik atau tidak. Semakin kecil nilainya menunjukkan bobot synaptic setiap link semakin menyesuaikan untuk menghasilkan output yang benar. (n = jumlah output).

$$E = \frac{1}{2} \sum_n e_n^2 \quad (2.11)$$

Error kuadrat rerata (average squared error) adalah rata-rata E terhadap cacah data latihan (P).

$$E_v = \frac{1}{p} \sum_p E_p \quad (2.12)$$

2.11 Algoritma Backpropagation

Menurut Haykin (1994), prosedur algoritma Backpropagation akan mengikuti langkah-langkah sebagai berikut :

1. Inisialisasi ; menentukan konfigurasi jaringan, kemudian menetapkan seluruh bobot synaptic dan treshold dengan nilai acak kecil yang terdistribusi secara seragam.
2. Menyiapkan data pelatihan ; untuk setiap pasangan (s t) dilakukan langkah 3 (komputasi ke depan) dan langkah 4 (komputasi balik) secara berurutan.
3. Komputasi ke depan ; input yang tersedia adalah nilai aktivasi bagi neuronneuron sesudah lapisan input. Kemudian untuk lapisan-lapisan berikutnya nilai aktivasi dihasilkan kemudian di propagasikan dengan memakai fungsi aktivasi sigmoid (rumus 1). Pada lapisan output nilai aktivasinya adalah sebagai keluaran.
4. Komputasi balik ; seluruh bobot synaptic disesuaikan untuk memperkecil error. Mulai dari link-link yang menuju lapisan output, sampai link-link yang menuju lapisan hidden pertama. Untuk penyesuaian bobot ini antara lain bisa menggunakan rumus 7 – rumus 10, tergantung posisi link. Untuk mempercepat konvergensi, bisa ditambahkan parameter momentum (α). Untuk data latihan ke-p

   $$w_{mn}(p) = w_{mn}(p-1) + \Delta w_{mn}(p) + \alpha(w_{mn}(p-1)) - w_{mn} = w_{mn}(p-1) + \Delta w_{mn}(p) + \alpha \Delta w_{mn}(p-1) \quad (2.13)$$
5. Iterasi ; iterasi dilakukan untuk sejumlah epoch, jika masih terjadi error, sampai average squared error(rumus 5) mencapai batas kecil toleransi tertentu atau nol.
6. Langkah 2 s/d 4 adalah algoritma untuk melatih jaringan. Untuk pengujian pola, cukup dilakukan komputasi ke depan satu lewatan (langkah 3).

2.12 Fungsi Aktivasi

Dalam backpropagation, fungsi aktivasi yang dipakai harus memenuhi beberapa syarat, yaitu kontinu, terdiferensial dengan mudah, dan merupakan fungsi yang tidak turun (Siang, 2009: 99). Fungsi aktivasi diharapkan jenuh (mendekati nilai-nilai maksimum dan minimum secara asimtot) (Puspitaningrum, 2006: 133).

Beberapa fungsi aktivasi dalam JST adalah sebagai berikut (Puspitaningrum, 2006: 133).

2.12.1 Fungsi Sigmoid Biner

Fungsi ini merupakan fungsi yang umum digunakan. Range-nya adalah (0,1) dan didefinisikan sebagai berikut.

$$f_1(x) = \frac{1}{1+e^{-x}} \tag{2.14}$$

Dengan turunan

$$f_1(x) = f_1(x)(1 - f_1(x)) \tag{2.15}$$

2.12.2 Fungsi Sigmoid Bipolar

Fungsi sigmoid bipolar merupakan fungsi yang umum digunakan dan memiliki range (-1,1) yang didefinisikan sebagai :

$$f_2(x) = 2f_1(x) - 1 \tag{2.16}$$

Dengan turunan

$$f_2(x) = \frac{1}{2}(1 + f_2(x))(1 - f_2(x)) \tag{2.17}$$

2.12.3 Fungsi Tangen Hiperbolik

Fungsi tangen hiperbolik didefinisikan sebagai

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.18}$$

Dengan turunan

$$tanh'(x) = (1 + \tanh(x))(1 - \tanh(x)) \tag{2.19}$$

2.13 Pelatihan Standar Backpropagation

Ada tiga fase pelatihan backpropagation, antara lain (Siang, 2009:100-101):

2.13.1 Fase 1, Propagasi Maju

Dalam propagasi maju, setiap sinyal masukan (=x_i) dipropagasikan ke layer tersembunyi menggunakan fungsi aktivasi yang ditentukan. Keluaran dari setiap unit layer tersembunyi (=z_j) tersebut selanjutnya dipropagasikan maju lagi ke layer tersembunyi diatasnya menggunakan fungsi aktivasi, demikian seterusnya hingga menghasilkan keluaran jaringan(=y_k). Berikutnya, keluaran jaringan (=y_k) 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 yang

ditentukan, maka iterasi dihentikan. Akan tetapi, apabila kesalahan masih lebih besar dari batas toleransinya, maka bobot setiap garis dalam jaringan akan dimodifikasi untuk mengurangi kesalahan yang terjadi.

Langkah 3 : sinyal dari unit input dikirim ke unit tersembunyi di atasnya

Langkah 4 : Hitung semua keluaran di unit tersembunyi z_j (j= 1,2,..P)

$$z\_net_j = v_{jo} + \sum_{i=1}^{n} x_i \, v_{ji}$$

$$z_j = f(Z\_net_j) = \frac{1}{1 + e^{-z\_net_j}}$$

Langkah 5 : Hitung semua keluaran jaringan di unit y_k (k=1,2,..,m)

$$y\_net_j = w_{jo} + \sum_{i=1}^{p} z_j \, w_{kj}$$

$$y_k = f(y\_net_k) = \frac{1}{1 + e^{-y\_net_k}}$$

2.13.2 Fase 2, Propagasi Mundur

Berdasarkan kesalahan $t_k - y_k$, dihitung faktor δ_k (k = 1,…,m) yang dipakai untuk mendistribusikan kesalahan di unit y_k ke semua unit tersembunyi yang terhubung langsung dengan y_k. δ_k juga dipakai untuk mengubah bobot garis yang berhubungan langsung dengan unit keluaran. Dengan cara yang sama, dihitung faktor δ_j di setiap unit di layer tersembunyi sebagai dasar perubahan bobot semua garis yang berasal dari unit tersembunyi di layer dibawahnya. Demikian seterusnya hingga semua faktor δ di unit tersembunyi yang berhubungan langsung dengan unit masukan dihitung.

Langkah 6 **:** Hitung faktor δ unit keluaran berdasarkan kesalahan setiap unit keluaran yk (K=1,2,…,m)

δ_k → komponen kesalahan yg akan dipakai pada perubahan bobot layar di bawahnya (langkah 7)

Hitung suku perubahan bobot w_{kj} (yang akan dipakai untuk merubah bobot w_{kj}) dengan laju percepatan ά

$$\Delta w_{kj} = \alpha \delta_k z_j$$

$$k = 1,2,\dots,m; j = 0,1,\dots,P$$

Langkah 7 : Hitung faktor δ unit tersembunyi :

$$\delta\_net_j = \sum_{k=1}^{m} \delta_k w_{jk}$$

Faktor δ unit tersembunyi :

$$\delta_j = \delta\_net_j f'(z\_net_j) = \delta\_net_j z_j(1 - z_j)$$

Hitung suku perubahan bobot vji →untuk ubah bobot vji)

$$\Delta v_{ji} = \alpha \delta_j x_i \quad ; \quad j = 1,2,\dots P \quad ; \quad 1 = 0,1,\dots,n$$

2.13.3 Fase 3, Perubahan Bobot

Setelah semua faktor δ dihitung, bobot semua garis dimodifikasi bersamaan. Perubahan bobot suatu garis didasarkan atas faktor δ neuron di layer atasnya. Sebagai contoh, perubahan bobot garis yang menuju ke layer keluaran didasarkan atas δ_k yang ada di unit keluaran.

Ketiga fase tersebut diulang-ulang terus hingga kondisi penghentian dipenuhi. Umumnya, kondisi penghentian yang sering dipakai adalah jumlah iterasi atau kesalahan. Iterasi akan dihentikan jika jumlah iterasi yang dilakukan sudah melebihi jumlah maksimum iterasi yang ditetapkan, atau jika kesalahan yang terjadi sudah lebih kecil dari batas toleransi yang diijinkan.

Langkah 8 : Hitung semua perubahan bobot Perubahan bobot garis yang menuju ke unit keluaran :

$$w_{kj}(baru) = w_{kj}(lama) + \Delta w_{kj}$$

$$(k = 1,2,\dots m \quad ; \quad j = 0,1,\dots,P)$$

Perubahan bobot garis yang menuju ke hiden layer :

$$v_{ji}(baru) = v_{ji}(lama) + \Delta v_{ji}$$

$$(j = 1,2,\dots m \quad ; \quad i = 0,1,\dots,n)$$

Setelah pelatihan → jaringan dapat dipakai untuk pengenalan pola dengan propagasi maju (langkah 4 dan 5)

2.14 Mean Square Error (MSE)

Menurut Kusumadewi (2004: 116), dalam pelatihan dengan backpropagation sama halnya seperti pelatihan pada jaringan syaraf yang lain. Pada jaringan feedforward (umpan maju), pelatihan dilakukan dalam rangka perhitungan bobot

sehingga pada akhir pelatihan akan diperoleh bobot-bobot yang baik. Selama proses pelatihan, bobot-bobot diatur secara iteratif untuk meminimumkan error (kesalahan) yang terjadi. Kesalahan dihitung berdasarkan rata-rata kuadrat kesalahan (MSE). Rata-rata kuadrat kesalahan juga dijadikan dasar perhitungan unjuk kerja fungsi aktivasi.

Menurut Pratiwi (2011), pelatihan dilakukan berulang-ulang dan berhenti jika telah mencapai batas iterasi maksimum yang ditentukan dan nilai error kurang dari Mean Square Error. Semakin kecil nilai MSE, maka dapat dianggap bahwa arsitektur jaringan semakin baik, demikian sebaliknya.

2.15 Pelatihan Dengan dan Tanpa Supervisi

Berdasarkan cara memodifikasi bobotnya, ada 2 macam pelatihan yang dikenal yaitu dengan supervisi (supervised) dan tanpa supervisi (unsurpervised).

Dalam pelatihan dengan supervisi, terdapat sejumlah pasangan data (masukan-target keluaran) yang dipakai untuk melatih jaringan hingga diperoleh bobot yang diinginkan. Sebaliknya dalam pelatihan tanpa supervisi tidak ada pelatihan yang akan mengarahkan proses, sehingga tidak ada proses pelatihan untuk mengenali objek. Perubahan bobot jaringan dilakukan berdasarkan parameter tertentu dan jaringan dimodifikasi menurut ukuran parameter tersebut. Berdasarkan hasil yang pernah dilaporkan, model pelatihan dengan supervisi lebih banyak digunakan dan terbukti cocok dipakai dalam berbagai aplikasi (Siang, 2005). Dalam pembuatan sistem ini juga digunakan pelatihan dengan supervisi.

Menurut Siang (2005:30) algoritma belajar atau pelatihan digolongkan menjadi sebagai berikut :

Dengan Supervisi (Supervised Training) Dalam pelatihan dengan supervisi, terdapat sejumlah pasangan data (masukan-target keluaran) yang dipakai untuk melatih jaringan. Pada setiap pelatihan, suatu masukan diberikan ke jaringan. Jaringan akan memproses dan mengeluarkan keluaran. Selisih antara keluaran jaringan dengan target (keluaran yang diinginkan) merupakan kesalahan yang terjadi. Jaringan akan memodifikasi bobot sesuai dengan kesalahan tersebut. Model yang menggunakan pelatihan dengan supervisi antara lain : Perceptron, ADALINE, MADALINE, Backpropagation, LVQ.

Tanpa Supervisi (Unsupervised Training) Dalam pelatihannya, perubahan bobot jaringan dilakukan berdasarkan parameter tertentu dan jaringan dimodifikasi menurut ukuran parameter tersebut. Model yang menggunakan pelatihan ini adalah model jaringan kompetitif.