BAB 2
LANDASAN TEORI

2.1 Pengertian Umum Diagnosa

Diagnosa berasal dari konteks medis yang berarti proses identifikasi penyakit dengan suatu gejala. Namun diagnosa ini digunakan dalam banyak disiplin ilmu yang berbeda dengan variasi dalam penggunaan logika, analisis, dan pengalaman untuk menentukan sebab dan akibat. Dalam sistem rekayasa dan ilmu komputer, biasanya digunakan untuk menentukan penyebab dari gejala dan solusi[4].

Dalam bidang ilmu komputer khususnya kecerdasan buatan, diagnosa berkaitan dengan pengembangan algoritma dan teknik yang mampu menentukan apakah *behavior* dari suatu sistem tersebut benar. Jika sistem tidak berfungsi dengan benar, algoritma harus dapat menentukan seakurat mungkin bagian manakah yang mengalami kesalahan. Dalam sistem jawaban di proses dan hasil dari jawaban tersebut merupakan ekspresi diagnosa[5].

2.2 Pengertian Umum Sinusitis

Sinusitis berasal dari bahasa latin, yang dalam ilmu kedokteran berarti peradangan, sehingga dapat disimpulkan sinusitis adalah suatu peradangan, atau pembengkakan, dari jaringan yang melapisi sinus. Biasanya sinus berisi udara, tetapi ketika sinus tersumbat dan berisi cairan, kuman (bakteri, virus, dan jamur) dapat berkembang dan menyebabkan infeksi.

Sinusitis sebagian besar sudah dapat didiagnosa hanya berdasarkan pada riwayat keluhan pasien serta pemeriksaan fisik yang dilakukan dokter. Hal ini juga disebabkan karena pemeriksaan menggunakan *CT Scan* dan *MRI* yang walaupun memberikan hasil lebih akurat namun biaya yang dikeluarkan cukup mahal. Pada pemeriksaan fisik akan ditemukan adanya kemerahan dan pembengkakan pada rongga hidung, ingus yang mirip nanah, serta pembengkakan disekitar mata dan dahi. Pemeriksaan menggunakan *CT Scan* dan *MRI* baru diperlukan bila sinusitis gagal

disembuhkan dengan pengobatan awal. *Rhinoscopy,* sebuah cara untuk melihat langsung ke rongga hidung, diperlukan guna melihat lokasi sumbatan *ostia.* Terkadang diperlikan penyedotan cairan sinus dengan menggunakan jarum suntik untuk dilakukan pemeriksaan kuman. Pemeriksaan ini berguna untuk menentukan jenis infeksi yang terjadi.

Sinusitis banyak ditemukan pada penderita *hay fever* yang mana pada penderita ini mengalami pilek menahun akibat dari alergi terhadap debu dan sari bunga. Sinusitis juga dapat disebabkan oleh bahan-bahan iritan seperti bahan kimia yang terdapat pada semprotan hidung serta bahan-bahan kimia lainnya yang masuk melalui hidung. Sinusitis juga bisa disebabkan oleh infeksi atau bakteri.

Sinus atau sering pula disebut dengan sinus *paranasalis* adalah rongga udara yang terdapat pada bagian padat dari tulang tengkorak di sekitar wajah, yang berfungsi untuk memperingan tulang tengkorak. Rongga ini berjumlah empat pasang kiri dan kanan. Sinus *frontalis* terletak dibagian dahi, sedangkan sinus *maksilaris* terletak di belakang pipi. Sementara itu, sinus *sphenoid* dan sinusi *ethmoid* terletak agak lebih dalam dibelakang rongga mata dan di belakang sinus *maksilaris.* Dinding sinus terutama dibentuk oleh sel-sel penghasil cairan *mukus*. Udara masuk ke dalam sinus melalui sebuah lubang kecil yang menghubungkan antara rongga sinus dengn rongga hidung yang disebut *ostia*. Jika karena suatu sebab lubang ini buntu maka udara tidak akan bisa keluar masuk dan cairan mukus yang diproduksi di dalam sinus tidak akan bisa dikeluarkan.

Disekitar rongga hidung terdapat empat sinus yaitu sinus *maksilaris* (terletak di pipi), sinus *etmoidalis* (terletak di kedua mata), sinus *frontalis* (terletak di dahi) dan sinus *sfenoidalis* (terletak di belakang dahi).

Secara klinis sinusitis dibagi atas berbagai jenis, yaitu:

1. Sinusitis Akut

   Sebuah kondisi mendadak seperti gejala seperti pilek, hidung tersumbat dan nyeri wajah yang tidak hilang setelah 10 hari sampai 14 hari. Sinusitis akut biasanya berlangsung 4 minggu atau kurang.

2. Sinusitis Sub-Akut

   Sebuah peradangan yang berlangsung 4 sampai 8 minggu.

3. Sinusitis Kronis

   Suatu kondisi yang ditandai dengan gejala radang sinusi yang berlangsung 8 minggu atau lebih.

Sinusitis dapat terjadi bila terdapat gangguan pengaliran udara dari hidung ke rongga sinus serta adanya gangguan pengeluaran cairan *mukus*. Adanya demam, flu, alergi dan bahan-bahan iritan dapat menyebabkan terjadinya pembengkakan pada *ostia* sehingga lubah *drainase* ini menjadi buntu dan mengganggu aliran udara sinus serta pengeluaran cairan *mukus.* Penyebab lain dari buntunya *ostia* adalah tumor dan trauma. *Drainase* cairan *mukus* keluar dari rongga sinus juga bisa terhambat oleh pengentalan cairan *mukus* itu sendiri. Pengentalan ini terjadi akibat pemberian obat *antihistamin*, penyakit *fibrokistik* dan lain lain. Sel penghasil *mukus* memiliki rambut halus (*silia*) yang selalu bergerak untuk mendorong cairan *mukus* keluar dari rongga sinus. Asap rokok merupakan biang kerok dari rusaknya rambut halus ini sehingga pengeluaran cairan *mukus* menjadi terganggu. Cairan *mukus* yang terakumulasi di rongga sinus dalam jangka waktu yang lama merupakan tempat yang nyaman bagi hidupnya bakteri, virus dan jamur.

Selain itu penyebab sinus bisa terjadi karena *Rhinogenik* (penyebab kelainan atau masalah di hidung), segala sesuatu yang menyebabkan sumbatan pada hidung dapat menyebabkan sinusitis. Termasuk flu biasa, *rhinitis* alergi (pembengkakan pada lapisan hidung), *polip* hidung (pertumbuhan kecil di lapisan hidung), atau *septum* menyimpang (pergeseran di rongga hidung). *Dentogenik / Odotogenik* (penyebabnya kelainan gigi), yang sering menyebabkan sinusitis infeksi pada gigi geraham atas (*pre molar* dan *molar*).

Beberapa gejala sinusitis yang paling umum adalah nyeri di wajah, serta demam. Hampir 25 persen pasien sinusitis akan mengalami demam terkait sinusitis yang diderita. Gejala lainnya berupa perubahan warna pada lendir di hidung, hidung tersumbat, nyeri menelan dan batuk, sakit tenggorokkan, warna lendir pekat dan

berwarna hijau atau kuning. Beberapa pasien akan merasakan sakit bertambah hebat apabila kepala ditundukkan ke depan. Pada sinusitis karena alergi maka penderita juga akan mengalami gejala lain yang berhubungan dengan alerginya seperti gatal pada mata, dan bersin-bersin[6].

Berikut adalah gejala umum penyakit sinusitis menurut jenisnya :

Tabel 2.1 Gejala Sinusitis Menurut Jenisnya

Jenis Sinusitis	Gejala
Sinusitis Akut	Hidung tersumbat, nyeri pada wajah, bersin-bersin disertai batuk, gatal disekitar mata, demam, hidung meler
Sinusitis Sub-Akut	Sakit pada gigi, nyeri pada leher, nyeri pipi dibawah mata, cepat merasa letih dan lesu, hidung meler, hidung tersumbat, batuk, demam, pusing yang berlebihan
Sinusitis Kronis	Peradangan sinus lebih dari 8 minggu, nyeri dipinggir hidung, nyeri antara mata, dahi terasa sakit, selaput lendir merah dan bengkak, cepat merasa letih dan lesu, hidung meler, batuk, nyeri saat bicara dan menelan, demam, pusing yang berlebihan, nyeri pada wajah

2.3 Algoritma

Algoritma merupakan salah satu cabang ilmu komputer yang membahas prosedur penyelesaian suatu permasalahan. Algoritma adalah serangkaian urutan langkah-langkah atau prosedur untuk menyelesaikan suatu masalah dengan memproses nilai masukkan menjadi nilai keluaran.

Suatu algoritma tidak saja harus benar, tetapi juga harus efisien dengan menghitung nilai dari performansinya. Setiap algoritma memiliki nilai performansi yang berbeda-beda. Performansi suatu algoritma dapat diukur dengan menghitung nilai kompleksitas waktunya dimana setiap langkahnya akan dihitung.

2.4 Analisis Algoritma

Analisis algoritma sangat membantu dalam meningkatkan efisiensi program. Kecanggihan suatu program bukan dilihat dari tampilan program, tetapi berdasarkan efisiensi metode atau algoritma yang terdapat didalam program tersebut. Dalam menguji suatu metode, dibutuhkan beberapa kriteria untuk mengukur efisiensi suatu

algoritma. Untuk memeriksa kebenaran metode dapat dilakukan dengan cara perurutan, memeriksa bentuk logika, implementasi metode algoritma, pengujian dengan data dan menggunakan cara matematika untuk membuktikan kebenaran. Penyederhanaan algoritma membagi algoritma menjadi bentuk yang sederhana.

2.5 Kecerdasan Buatan

Kecerdasan buatan atau *Artificial Intelegence* merupakan salah satu bagian ilmu komputer yang membuat mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan manusia[7]. Dengan adanya kecerdasan buatan, diharapkan tidak menutup kemungkinan hanya dengan data pengetahuan yang terbatas, sebuah komputer dapat berfikir seperti manusia dalam menghadapi masalah.

2.5.1 *Naïve Bayes*

Teorema *Bayes* dikemukakan oleh seorang pendeta *presbyterian* Inggris pada tahun 1763 yang bernama *Thomas Bayes*. Teorema *Bayes* digunakan untuk menghitung probabilitas terjadinya suatu peristiwa berdasarkan pengaruh yang didapat dari hasil observasi.

Probabilitas *Bayesian* adalah suatu interpretasi dari kalkulus yang memuat konsep probabilitas sebagai derajat dimana suatu pernyataan dipercaya benar. Teori *Bayesian* juga dapat digunakan sebagai alat pengambilan keputusan untuk memperbaharui tingkat kepercayaan dari suatu informasi.

Metode *Naïve Bayes* merupakan pendekatan statistik untuk melakukan inferensi induksi pada persoalan klasifikasi. Metode ini menggunakan probabilitas bersyarat sebagai dasarnya. Pendekatan ini adalah salah satu cara untuk mengatasi ketidakpastian dengan menggunakan *Formula Bayes*.

Pendekatan *Bayes* pada saat klasifikasi adalah mencari probabilitas tertinggi (VMAP) dengan masukkan atribut (a1, a2, a3, … , an) seperti berikut :

$$V_{MAP} = \arg\max \; P_{vj}\, a_1, a_2, a_3, \dots, a_n)$$

Persamaan 2-1

Teorema *Bayes* sendiri berawal dari rumus berikut :

$$P(H|E) = \frac{P(E \cap H)}{P(E)}$$

Persamaan 2-2

Dimana P(H|E) artinya peluang H jika diketahui keadaan E.

Rumus *Formula Bayes*

$$P(H|E) = P(H|E)\frac{P(H)}{P(E)}$$

Persamaan 2-3

Dimana :

P(H|E) : Probabilitas hipotesis H jika terdapat *evidence* E.

P(E|H) : Probabilitas munculnya *evidence* E jika diketahui hipotesa H.

P(H) : Probabilitas hipotesis H tanpa memandang *evidence* apapun.

P(E) : Probabilitas *evidence* E.

Jika setelah dilakukan pengujian terhadap hipotesis, muncul satu atau lebih *evidence* atau observasi baru, maka :

$$P(H|E,e) = P(H|E)\frac{P(e|E,H)}{P(e|E)}$$

Persamaan 2-4

Dimana :

e : *Evidence* lama

E : *Evidence* atau observasi baru

P(H|E,e): Probabilitas adanya hipotesis H jika muncul *evidence* baru E dari *evidence* lama.

P(H|E) : Probabilitas hipotesis H jika terdapat *evidence* E

P(e|E,H): Probabilitas kaitan antara e dan E jika hipotesis H benar.

P(e|E) : Probabilitas kaitan antara r dan E tanpa memandang hipotesis apapun.

2.5.1.1 Alur Metode Naïve Bayes

Alur dari metode *Naïve Bayes* adalah :

1. Masukkan gejala sebagai *input*an.
2. Baca data *training* yang telah tersedia.
3. Hitunglah jumlah dari masing-masing label.
4. Hitung jumlah kasus yang sama dengan label / *class* yang sama.
5. Kalikan semua hasil variabel sesuai dengan kolom hasil.
6. Bandingkan hasil probabilitas tiap label / *class*.
7. Ambil hasil yang paling maksimal probabilitasnya.
8. Menghasilkan hasil dari perhitungan metode.

Berikut adalah alur proses dari metode *Naïve Bayes*.



Gambar 2.1 Alur Proses Metode *Naive Bayes*

2.5.1.2 Kelebihan dan Kekurangan Metode Naïve Bayes

Teori *Naïve Bayes* menurut *Grainner* (1998), mempunyai beberapa kelebihan, yaitu:

1. Mudah untuk dipahami
2. Hanya memerlukan pengkodean yang sederhana
3. Lebih cepat dalam perhitungan
4. Menangani kuantitatif dan data diskrit
5. Cepat dan efisiensi ruang

Kekurangan dari metode *Naïve Bayes* adalah :

1. Tidak berlaku jika probabilitas kondisionalnya adalah nol, apabila nol maka probabilitas prediksi akan bernilai nol juga.
2. Mengasumsikan variabel bebas.

2.5.2 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan (*Neural Network*) adalah salah satu kajian pada kecerdasan buatan. Jaringan syaraf tiruan adalah salah satu representasi buatan dari otak manusia yang selalu mencoba untuk menstimulasikan proses pembelajaran pada otak manusia tersebut. Istilah buatan maksudnya adalah jaringan syaraf yang diimplementasikan menggunakan program komputer untuk menyelesaikan sejumlah proses perhitungan selama proses pembelajaran.

Komponen jaringan syaraf tiruan (JST) terdiri dari *neuron-neuron* yang saling berhubungan. *Neuron – neuron* ini akan mentransformasikan informasi yang diterima melalui sambungan keluarannya menuju *neuron-neuron* yang lain. Pada JST hubungan *neuron-neuron* tersebut dikenal dengan nama bobot. Komponen JST yang terdiri dari *input* (informasi) dan bobot (nilai-nilai tertentu). Fungsi aktivasi berfungsi apabila *input* yang dimasukkan sesuai dengan nilai ambang (*threshold*) yang ditentukan jika tidak sesuai maka fungsi aktivasi tidak di aktifkan, dan apabila *neuron-neuron* tersebut diaktifkan maka *neuron* akan mengirimkan *output* melalui bobot-bobot.

Jaringan syaraf tiruan mempunyai dua macam proses pembelajaran yaitu pembelajaran terawasi dan pembelajaran tidak terawasi. Pembelajaran terawasi adalah jika *output* yang diharapkan telah diketahui sebelumnya, sedangkan pembelajaran tidak terawasi adalah proses pembelajaran yang tidak memerlukan target *output.*

1. Pembelajaran Terawasi (*Supervised Learning*)

   Metode pembelajaran pada jaringan syaraf tiruan disebut terawasi jika *output* yang diharapkan telah diketahui sebelumnya. Pada proses pembelajaran, satu pola input akan diberikan ke satu *neuron* pada lapisan *input*. Pola ini akan dirambatkan disepanjang jaringan syaraf hingga sampai ke *neuron* pada lapisan *output*. Lapisan *output* ini akan membangkitkan pola *output* yang nantinya akan dicocokkan dengan pola *output* targetnya. Apabila terjadi perbedaan antara pola *output* hasil pembelajaran dengan pola target, maka disini akan muncul *error*. Apabila nilai *error* ini masih cukup besar, mengindikasikan bahwa masih perlu dilakukan lebih banyak pembelajaran lagi. Metode pembelajaran terawasi diantaranya *Hebb Rule, Perceptron, Delta Rule, Backpropagation, Learning Vector Quantization, Heteroassociative Memory.*

2. Pembelajaran Tak Terawasi (*Unsupervised Learning*)

   Pada metode pembeajaran yang tak terawasi ini tidak memerlukan target *output*. Pada metode ini, tidak dapat ditentukan hasil yang seperti apakah yang diharapkan selama proses pembelajaran. Metode pembelajaran tidak terawasi salah satunya adalah jaringan kohonen.

2.5.2.1 *Backpropagation Neural Network*

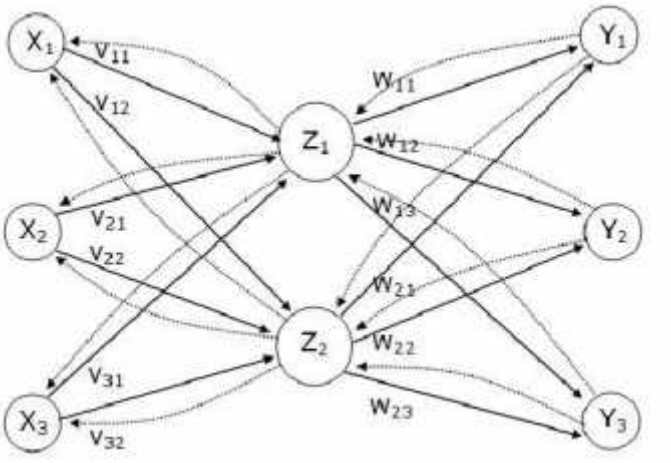
Backpropagation Neural Network merupakan algoritma pembelajaran yang terawasi dan biasanya digunakan oleh *perceptron* dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan *neuron-neuron* yang ada pada lapisan tersembunyinya. Algoritma

Backpropagation Neural Network menggunakan *error output* untuk mengubah nilai bobot-bobotnya dalam arah mudur (*backward*). Untuk mendapatkan *error* ini, tahap perambatan maju (*forward*) harus dikerjakan terlebih dahulu. Pada saat perambatan maju, *neuron-neuron* diaktifkan dengan menggunakan fungsi aktivasi *sigmoid biner (F(X))*, seperti terlihat pada persamaan dibawah ini:

$$f(X) = \frac{1}{1 + e^{-}}$$

Persamaan 2-5

Arsitektur *Backpropagation Neural Network* terlihat seperti Gambar 2.2 Jaringan terdiri atas 3 unit (*neuron*) pada lapisan input yaitu *x1*, *x2,* dan *x3;* 1 lapisan tersembunyi dengan 2 *neuron*, yaitu *z1* dan *z2*; serta 3 unit pada lapisan *output*, yaitu *y*. Bobot yang menghubungkan *x1*, *x2*, dan *x3* dengan *neuron* pertama pada lapisan tersembunyi adalah *v11*, *v21*, *v31* (*vij* : Bobot yang menghubungkan *neuron input* ke-I ke *neuron* ke-j pada lapisan tersembunyi). Bobot yang menghubungkan *z1* dan *z2* dengan *neuron* pada lapisan *output* adalah *w1* dan *w2*. Fungsi aktivasi yang digunakan antara lapisan *input* dan lapisan tersembunyi, dan lapisan tersembunyi dengan lapisan *output* adalah fungsi aktivasi *sigmoid biner*.



Gambar 2.2 Arsitektur Jaringan *Backpropagation*

Pada metode *Backpropagation Neural Network* ini terdapat dua tahapan proses, yaitu :

1. Tahapan belajar atau pelatihan, dimana pada tahap ini metode *Backpropagation Neural Network* diberikan sejumlah data pelatihan dan target
2. Tahap pengujian atau penggunaan, pengujian dilakukan setelah melakukan tahap belajar atau tahap pelatihan.

Untuk menentukan *stopping condition* terdapat dua cara yang biasa dipakai, yaitu:

1. Membatasi iterasi yang ingin dilakukan.

   Misalnya jaringan akan dilatih sampai iterasi yang ke-500. Yang dimaksud dengan satu iterasi adalah perulangan step 3 sampai step 8 untuk semua *training* data yang ada.
2. Membatasi *error*

   Misalnya menentukan besar *Mean Square Error* antara *output* yang dikehendaki dan *output* yang dihasilkan oleh jaringan.

Jika terdapat sebanyak *m training* data, maka untuk menghitung *Mean Square Error* digunakan persamaan berikut.

$$MSE = 0.5 \sum \{ t_{k1} - y_{k1}^{\ 2} + t_{k2} - y_{k2}^{\ 2} + \ldots + t_{kn} - y_{kn}^{\ 2} \}$$

Persamaan 2-6

Dimana :

MSE : *Mean Square Error*

t_k : target *output*

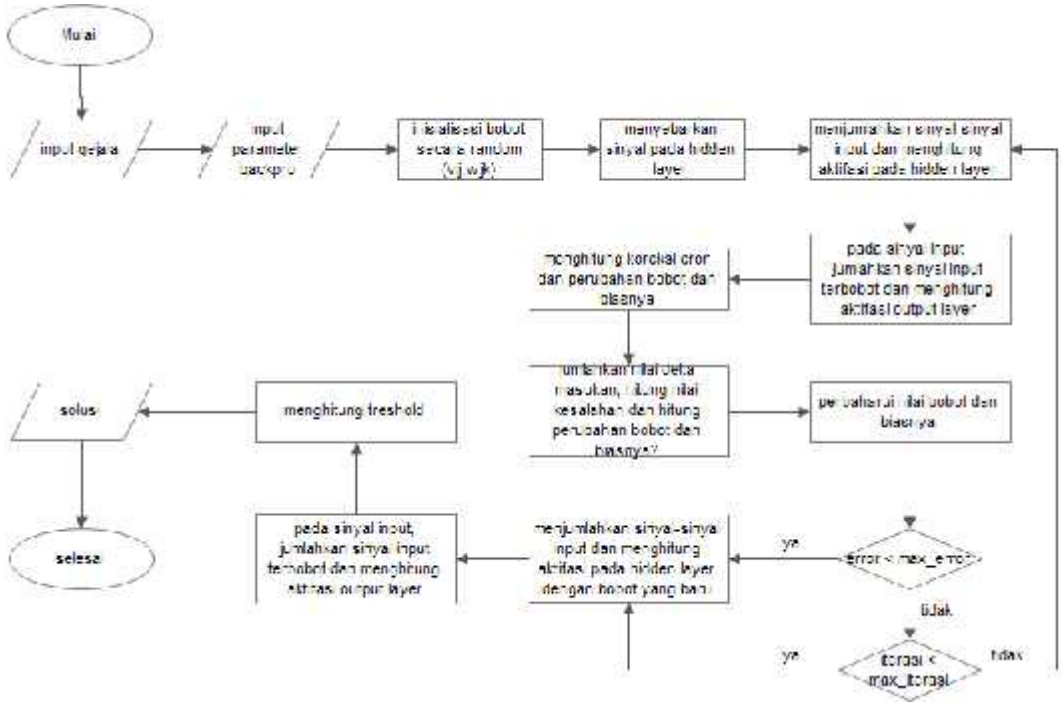
y_k : *output* yang dihasilkan

Setelah pelatihan selesai, *Backpropagation Neural Network* dianggap telah pintar sehingga apabila diberi *input* tertentu jaringan akan menghasilkan *output* seperti yang diharapkan. Cara mendapatkan *output* tersebut adalah dengan mengimplementasikan metode *Backpropagation*

Neural Network yang sama seperti proses belajar, tetapi hanya pada bagian umpan majunya saja.

2.5.2.2 Alur Metode *Backpropagation Neural Network*

Alur pada metode *Backpropagation Neural Network* dapat terlihat seperti gambar.



Gambar 2.3 Alur Proses Metode *Backpropagation Neural Network*

Berikut adalah deskripsi alur proses metode *Backpropagation Neural Network* :

Inisialisasi bobot (ambil bobot awal [V_{ij}, W_{jk}] dengan nilai *random*).

Kerjakan langkah-langkah berikut selama kondisi berhenti bernilai *FALSE* :

1. Untuk tiap-tiap pasangan elemen yang akan dilakukan pembelajaran, kerjakan:

Feedforward:

a. Tiap-tiap unit *input* (*Xi, i* = 1, 2, 3, … , *n*) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua unit pada lapisan yang ada di atasnya (lapisan tersembunyi).

b. Tiap-tiap unit tersembunyi (*Zi, j* = 1, 2, 3, … , *p*) menjumlahkan sinyal-sinyal *input* terbobot:

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

Persamaan 2-7

Dimana :

z_in_j : Bobot sinyal unit tersembunyi.

$v0_j$: Bobot bias menuju ke *hidden layer*.

x_i : Unit (*neuron*) dari sinyal input.

v_{ij} : Bobot yang menghubungkan *neuron input* ke-*i* ke *neuron* ke-*j* pada lapisan tersembunyi.

Setelah menjumlahkan sinyal-sinyal *input* maka lakukan perhitungan fungsi aktivasi. Fungsi aktivasi ini ditentukan untuk menghitung sinyal *output* dari *hidden* unit yang bersangkutan, lalu mengirim sinyal *output* ke seluruh unit pada unit *output*.

$$z_j = f(z_in_j)$$

Persamaan 2-8

Dimana :

z_j : Keluaran unit tersembunyi.

z_in_j : Bobot unit tersembunyi.

c. Langkah selanjutnya adalah setiap unit *output* (Y_k, *k=1,* … *m*) akan menjumlahkan sinyal – sinyal *input* yang sudah berbobot, termasuk biasnya.

$$y_{in_k} = w_{0j} + \sum_{j=1}^{n} z_j w_{jk}$$

Persamaan 2-9

Dimana :

y_in_k : Bobot sinyal lapisan *output.*

$w0_j$: Bobot bias menuju *output.*

z_j : Keluaran unit tersembunyi.

w_{jk} : Bobot awal menuju *output.*

Sinyal *input* yang sudah terbobot sudah diketahui maka hitung fungsi aktivasi sinyal *output* dari unit *output* yang bersangkutan.

$$y_k = f(y_in_k)$$

Persamaan 2-10

Dimana :

y_k : Keluaran unit *output.*

y_in_k : Bobot sinyal lapisan *output.*

Backpropagation

d. Tiap-tiap unit *output* (y_k, *k=1, 2, 3, … ,m*) menerima target pola yang berhubungan dengan pola *input* pembelajaran, hitung informasi *error*nya.

$$\delta_k = t_k - y_k f'(y_in_k)$$

Persamaan 2-11

Dimana :

δ_k : Informasi *error.*

t_k : Target keluaran.

y_k : Keluaran unit *output.*

y_in_k : Bobot sinyal lapisan *output*.

Faktor δ_k ini digunakan untuk menghitung koreksi *error* (Δw_{jk}) yang nantinya akan dipakai untuk memperbaharui w_{jk} , dimana :

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

Persamaan 2-12

Dimana :

Δw_{jk} : Koreksi *error.*

α : Rasio pembelajaran (*learning rate*).

δ_k : Informasi *error.*

z_j : Keluaran unit tersembunyi.

Selain itu juga dihitung koreksi bias (Δw_{0k}) yang nantinya akan dipakai untuk memperbaharui $w0_k$,dimana :

$$\Delta w_{0k} = \alpha \delta_k$$

Persamaan 2-13

Dimana :

Δw_{0k} : Koreksi bias.

α : Rasio pembelajaran (*learning rate*).

δ_k : Informasi *error.*

Faktor δ_k ini kemudian dikirimkan ke layer didepannya.

e. Setiap *hidden* unit (*zj, j = 1, … , p*) menjumlah *input* delta (yang dikirim dari *layer* pada step 6) yang sudah berbobot.

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

Persamaan 2-14

Dimana :

δ_in_j : *Input* delta.

δ_k : Informasi *error.*

w_{jk} : Koreksi *error*.

Kemudian hasilnya dikalikan dengan turunan dari fungsi aktivasi yang digunakan jaringan untuk menghasilkan faktor koreksi *error* δ_j, dimana :

$$\delta_j = \delta_in_j f'(z_in_j)$$

Persamaan 2-15

Dimana :

δ_j : Faktor koreksi *error*

δ_in_j : *Input* delta.

z_in_j : Bobot unit tersembunyi.

Faktor δ_j ini digunakan untuk menghitung koreksi *error* (Δv_{ij}) yang nantinya akan dipakai untuk memperbaharui v_{ij}, dimana:

$$\Delta v_{ij} = \alpha \delta_j x_i$$

Persamaan 2-16

Dimana :

Δv_{ij} : Koreksi *error* untuk memperbaharui v_{ij}.

α : Rasio pembelajaran (*learning rate*).

x_i : Masukkan (*input*).

Selain itu juga dihitung koreksi bias (Δv_{0j}) yang nantinya akan dipakai untuk memperbaharui v_{0j}, dimana :

$$\Delta v_{0j} = \alpha \delta_j$$

Persamaan 2-17

Dimana :

Δv_{0j} : Koreksi bias.

α : Rasio pembelajaran (*learning rate*).

δ_j : Faktor koreksi *error*.

f. Tiap-tiap unit *output* (y_k, $k = 1, 2, 3, \dots, m$) memperbaiki bias dan bobotnya ($j=0, 1, 2, \dots, p$) dimana :

$$w_{jk}\ baru = w_{jk}\ lama + \Delta w_{jk}$$

Persamaan 2-18

Dimana :

w_{jk} : Koreksi *error*.

Δw_{jk} : Delta koreksi *error*.

Tiap-tiap unit tersembunyi (z_j, $j = 1, 2, 3, \dots, p$) memperbaiki bias dan bobotnya ($i = 0, 1, 2, \dots, n$) dimana :

$$v_{ij}\ baru = v_{ij}\ lama + \Delta v_{ij}$$

Persamaan 2-19

Dimana :

v_{ij} : Unit tersembunyi.

Δv_{ij} : Delta unit tersembunyi.

2. Tes kondisi berhenti.

Keterangan persamaan :

z_in : Bobot sinyal unit tersembunyi.

z : Keluaran unit tersembunyi.

y_in : Bobot sinyal lapisan keluaran.

y : Keluaran unit keluaran.

δ : Kesalahan.

Δw : Koreksi bobot unit keluaran.

v_{ij} : Bobot yang menghubungkan *neuron input* ke-i ke *neuron* ke-j pada lapisan tersembunyi.

Δv : Koreksi bobot unit tersembunyi.

x_i : Unit (*neuron*) pada lapisan *input*.

n : Jumlah *neuron* (unit) pada lapisan *input*.

v : Bobot awal masuk ke *hidden*.

v_o : Bobot bias yang menuju ke *hidden*.

w : Bobot awal *hidden* ke *output*.

w_o : Bobot bias yang menuju ke *output*.

α : *Learning rate* / rasio pembelajaran.

2.5.2.3 Parameter *Backpropagation Neural Network*

Dalam metode *Backpropagation Neural Network* terdapat beberapa parameter yang sangat penting pada proses pelatihan, yaitu jumlah *neuron* pada *hidden layer, learning rate,* jumlah iterasi dan batasan *error.*

Sampai saat ini belum ada formula khusus yang bisa menemukan jumlah *neuron* pada *hidden layer* yang optimal. Suatu formula yang bisa digunakan untuk memperkirakan jumlah *neuron* pada *hidden layer* adalah :

$$N_h = \overline{N_i N_o}$$

Persamaan 2-20

Dimana :

N_h : Jumlah *neuron hidden layer.*

N_i : Jumlah *neuron input.*

N_o : Jumlah *neuron output*.

Dimana N_h adalah jumlah *neuron* pada *hidden layer*, N_i jumlah *node* pada *input layer* (jumlah variabel pada pola masukkan), dan N_o adalah jumlah *neuron* pada *output layer*. Misalkan N_i = 3 dan N_o = 1. Oleh karena itu $N_h \approx \sqrt{3 \square 1} \approx 2$ (pembulatan ke atas). Jumlah *neuron* pada *hidden layer* yang optimal bisa lebih atau kurang dari hasil perhitungan tersebut.

Parameter *learning rate* sangat mempengaruhi proses pelatihan. *Learning rate* yang terlalu besar (misalkan 0,9) akan mengakibatkan *MSE* (*Mean Square Error*) menurun tajam pada awal iterasi, tetapi akan mengakibatkan *MSE* menjadi berosilasi atau naik turun tidak terkendali. Sebaliknya *learning rate* yang terlalu kecil (misalkan 0,0001) akan mengakibatkan *MSE* menurun sangat pelan.

Banyak strategi yang diusulkan untuk menentukan *learning rate*. Misalkan pada awal iterasi (*epoch*), kita gunakan *learning rate* secara gradual pada iterasi-iterasi berikutnya.

Jumlah iterasi dan batasan *error* digunakan sebagai kondisi berhenti pada proses pelatihan. Jika batasan *error* yang didefinisikan terlalu kecil mungkin akan terjadi *overfit*, artinya memiliki akurasi tinggi untuk data *training* tapi akurasi sangat rendah pada data *test*.

2.6 Metode Pengujian

2.6.1 Pengujian *White-Box*

Pengujian *white-box* dilakukan untuk menguji prosedur-prosedur yang mempunyai lintasan *logic* yang dilalui oleh setiap bagian prosedur. Prosedur tersebut diuji dengan memberikan kondisi / *loop* spesifik. Pengujian *white-box* menjamin pengujian terhadap semua lintasan yang tidak bergantungan minimal satu kali, mencoba semua keputusan *logic* dari sisi *trus* dan *false*, eksekusi semua *loop* dalam batasan kondisi dan batasan operasional data pengujian validasi dan internal.

Konsep pengujian basis *path* yaitu :

a. Merupakan bagian dari pengujian *white-box* dalam hal pengujian prosedur.
b. Mempergunakan notasi aliran *graph* (*node, link* untuk merepresentasikan *sequence, if, while, until*, dll).
c. Konsep kompleksitas *cyclomatic* antara lain cara perhitungan daerah tertutup pada *graph planar* dimana dapat menghubungkan batas atas jumlah pengujian yang harus direncanakan dan di eksekusi untuk menjamin pengujian seluruh *statement* program.
d. Memunculkan kasus-kasus yang akan diuji dengan membuat daftar lintasan kasus pengujian berdasarkan kompleksitas dan *cyclomatic* yang didapat.
e. Membuat alat bantu untuk *matrix graph* yang membantu pengawasan pengujian.

2.6.2 Pengujian *Black-Box*

Pengujian *black-box* dilakukan untuk menguji antarmuka dari perangkat lunak yang telah dibangun. Pengujian ini dilakukan untuk memperlihatkan bahwa hasil fungsi-fungsi bekerja dengan baik dalam arti masukkan yang diterima dengan benar dan keluaran yang dihasilkan benar-benar tepat dan pengintegrasian dari eksternal data berjalan dengan baik.

Metode pengujian *black-box* memfokuskan pada kebutuhan fungsi dari perangkat lunak. Pengujian ini merupakan bagian dari pengujian *white-box*. Pengujian *white-box* dilakukan terlebih dahulu pada proses pengujian, sedangkan pengujian *black-box* dilakukan pada tahap akhir dari pengujian perangkat lunak.

Proses yang terdapat dalam proses pengujian *black-box* antara lain sebagai berikut :

a. Pembagian kelas data pengujian setiap kasus yang muncul pada pengujian *white-box*.
b. Analisis batasan nilai yang berlaku untuk setiap data.

2.7 *WAMPServer* 2.2

WAMP (*Windows, Apache, MySQL, PHP*) merupakan *server* yang dapat dijalankan komputer tanpa memerlukan sambungan internet. *Server* di komputer ini disebut dengan *Local Server* (*localhost*) yang mana *server* ini nantinya akan kita *install*. Proses instalasi *website hosting* yang sudah memiliki sistem *CMS* (*Content Management System*) disebut juga proses pembuatan *database* di komputer atau *local server* (*localhost*).

2.8 *PHP (Hypertext Preprocessor)*

Pada awalnya *PHP* merupakan kependekkan dari *Personal Home Page* (situs personal). *PHP* pertama kali dibuat oleh *Rasmus Lerdorf* pada tahun 1995. Pada waktu itu *PHP* masih benama *Form Interpreted* (*FI*), yang wujudnya berupa sekumpulan *script* yang digunakan untuk mengolah data formulir dari *web*.

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya *PHP/FI*. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan *PHP*.

Pada November 1997, dirilis *PHP/FI 2.0*. Pada rilis ini, *interpreter PHP* sudah diimplementasikan dalam program *C*. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan *PHP/FI* secara signifikan.

Pada tahun 1997, sebuah perusahaan bernama *Zend* menulis ulang *interpreter PHP* menjadi lebih bersih, lebih baik dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis *interpreter* baru untuk PHP dan meresmikan rilis tersebut sebagai *PHP 3.0* dan singkatan *PHP* diubah menjadi akronim berulang *PHP* yaitu *Hypertext Preprocessing*.

Pada pertengahan tahun 1999, *Zend* merilis *interpreter PHP* baru dan rilis tersebut dikenal dengan *PHP 4.0*. *PHP 4.0* adalah versi *PHP* yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai karena kemampuannya untuk membangun aplikasi *web* kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, *Zend* merilis *PHP 5.0*. Dalam versi ini, inti dari *interpreter PHP* mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam *PHP* untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

2.9 *MySQL*

MySQL adalah sebuah perangkat lunak sistem Manajemen Basis Data *SQL* atau *Database Management System* (*DBMS*) yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi diseluruh dunia.

Tidak sama dengan proyek-proyek seperti *Apache*, dimana perangkat lunak dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, *MySQL* dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia *MySQL AB*, dimana memegang hak cipta hampir atas semua kode

sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan *MySQL AB* adalah *David Axmark*, *Allan Larsson*, dan *Michael Monty Widenius*.

MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional (*RDBMS*) yang didistribusikan secara gratis dibawah lisensi *GPL* (*General Public License*). Setiap pengguna dapat secara bebas menggunakan *MySQL*, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. *MySQL* sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya yaitu *SQL* (*Structured Query Language*). *SQL* adalah sebuah konsep pengoperasian basis data, terutama untuk pemilihan atau seleksi dan pemasukkan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basis data (*DBMS*) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah *SQL* yang dibuat oleh penggua maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data non-transaksional. Pada modus operasi non-transaksional, *MySQL* dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basis data kompetitor lainnya. Namun demikian pada modus non-transaksional tidak ada jaminan atas reliabilitas data seperti aplikasi *blogging* berbasis *web* (*wordpress*), *CMS* dan sejenisnya. Untuk kebutuhan sistem yang ditunjukkan untuk bisnis sangat disarankan untuk menggunakan modus basis data transaksional, hanya saja sebagai konsekuensinya unjuk kerja *MySQL* para modus transaksional tidak secepat unjuk kerja pada modus non-transaksional.

2.10*Adobe Dreamweaver CS 5*

Adobe Dreamweaver merupakan program penyunting halaman web keluaran *Adobe Systems* yang dulu dikenal sebagai *Macromedia Dreamweaver* keluaran *Macromedia*. Program ini banyak digunakan oleh pengembang *web* karena fitur-fiturnya yang menarik dan kemudahan penggunanya. Versi terakhir *Macromedia Dreamweaver* sebelum *Macromedia* dibeli oleh *Adobe Systems* yaitu versi 8. Versi

terakhir *Dreamweaver* keluaran *Adobe Systems* adalah versi 12 yang ada dalam *Adobe Creative Suite* 6 (sering disingkat *Adobe CS6*).

Adobe Dreamweaver adalah aplilasi desain dan pengembangan *web* yang menyediakan editor *WSYWYG* visual (bahasa sehari-hari yang disebut sebagai *Design View*) dan kode editor dengan fitur standar seperti *syntax high lighting*, *code completion*, dan *code collapsing* serta fitur lebih canggih seperti *real-time syntax checking* dan *code introspection* untuk menghasilkan petunjuk kode untuk membantu pengguna dalam menulis kode. Tata letak tampilan desain memfasilitasi desain cepat dan pembuatan kode seperti memungkinkan pengguna dengan cepat membuat tata letak dan manipulasi elemen *HTML*.

Dreamweaver memiliki fitur browser yang terintegrasi untuk melihat halaman web yang dikembangkan di jendela pratinjau program sendiri agar konten memungkinkan untuk terbuka di *web browser* yang telah terinstal. Aplikasi ini menyediakan transfer dan fitur sinkronisasi, kemampuan untuk mencari dan mengganti baris teks atau kode untuk mencari kata atau kalimat biasa di seluruh situs, dan *templating feature* yang memungkinkan untuk berbagi satu sumber kode atau memperbarui tata letak diseluruh situs tanpa *server side includes* atau *scripting*. *Behavior Panel* juga memungkinkan penggunaan *JavaScript* dasar tanpa pengetahuan coding, dan integrasi dengan *Adobe Spry Ajax Framework* menawarkan akses mudah ke konten yang dibuat secara dinamis dan *interface*.

Dreamweaver dapat menggunakan ekstensi dari pihak ketiga untuk memperpanjang fungsionalitas inti dari aplikasi, yang setiap pengembang *web* bisa menulis (sebagian besar dalam *HTML* dan *JavaScript*). *Dreamweaver* didukung oleh komunitas besar pengembang ekstensi yang membuat ekstensi yang tersedia (baik komersial maupun yang gratis) untuk pengembangan *web* dari efek *rollover* sederhana sampai *full-featured shooping cart*.

Dreamweaver seperti editor *HTML* lainnya, edit *file* secara lokal kemudian di*upload* ke *web server remote* menggunakan *FTP*, *SFTP*, atau *WebDAV*. *Dreamweaver CS4* sekarang mendukung sistem kontrol versi *SubVersion* (*SVN*).

2.11*Adobe Photoshop CS 3*

Adobe Photoshop atau biasa disebut *Photoshop* adalah perangkat lunak editor citra buatan *Adobe Systems* yang dikhususkan untuk pengeditan foto/gambar dan pembuatan efek. Perangkat lunak ini banyak digunakan oleh fotografer digital dan perusahaan iklan sehingga dianggap sebagai pemimpin pasar (*market leader*) untuk perangkat lunak pengolah gambar/foto dan bersama *Adobe Acrobat*, dianggap sebagai produk terbaik yang pernah diproduksi oleh *Adobe Systems*. Versi kedelapan aplikasi ini disebut dengan nama *Photoshop CS* (*Creative Suite*), versi sembilan disebut *Adobe Photoshop CS2*, versi sepuluh disebut *Adobe Photoshop CS3*, versi kesebelas adalah *Adobe Photoshop CS4*, versi keduabelas adalah *Adobe Photoshop CS5*, dan versi yang terakhir (ketigabelas) adalah *Adobe Photoshop CS6*.

Photoshop tersedia untuk *Microsoft Windows*, *Mac OS X* dan *Mac OS* versi 9 keatas juga dapat digunakan oleh sistem operasi lain seperti *Linux* dengan bantuan perangkat lunak tertentu seperti *CrossOver*.

Pada tahun 1987, *Thomas Knoll*, mahasiswa *PhD* di Universitas *Michigan*, mulai menulis sebuah program pada *Macintosh Plus*-nya untuk menampilkan gambar *grayscale* pada layar monokrom. Program ini, yang disebut *Display*, menarik perhatian saudaranya *John Knoll*, seorang karyawan di *Industrial Light and Magic*, yang merekomendasikan *Thomas* agar mengubah programnya menjadi program penyunting gambar penuh. *Thomas* mengambil enam bulan istirahat dari studi pada tahun 1988 untuk berkolaborasi dengan saudaranya pada program itu, yang telah diubah namanya menjadi *ImagePro*. Setelah tahun itu, *Thomas* mengubah nama programnya menjadi *Photoshop* dan bekerja dalam jangka pendek dengan produsen *scanner Barneyscan* untuk mendistribusikan salinan dari program tersebut dengan *slide scanner*.