BAB 2

TINJAUAN PUSTAKA

2.1 Citra Digital

Secara harfiah, citra *(image)* adalah gambar pada bidang dwimatra (dua dimensi) (Ahmad U, 2005). Ditinjau dari sudut pandang matematis, citra merupakan fungsi menerus *(continue)* dari intemsitas cahaya pada bidang dwimatra. Sumber cahaya menerangi objek kemudian objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pantulan cahaya ini ditangkap oleh alat-alat optik, misalnya mata pada manusia, kamera, pemindai *(scanner)* dan sebagainya. Sehingga bayangan objek yang disebut citra tersebut terekam.

2.2 Pengolahan Citra

Pengolahan citra *(image processing)* adalah pemrosesan citra, khususnya dengan menggunakan komputer, menjad citra yang kualitasnya menjadi lebih baik (Ahmad U, 2005). Pengolahan citra ini sangat diperlukan walaupun citra sangat kaya dengan informasi, namun seringkali citra mengalami penurunan mutu *(degradasi),* misalnya mengandung cacat atau derau *(noise),* warnanya terlalu kontars, kurang tajam, kabur *(blurring)* dan sebagainya. Tentu saja citra semacam ini menjadi sulit diinterpretasi karena informasi yang disampaikan oleh citra tersebut menjadi berkurang. Operasi-operasi pada pengolahan citra diterapkan pada citra bila :

1. Perbaikan atau modifikasi citra dilakukan untuk meningkatkan kualitas penampakan citra menonjolkan beberapa aspek informasi yang terkandung dalam citra *(image enchanment).* Contaoh : perbaikan kontras gelap atau terang, perbaikan tepian objek, penajaman, pemberian warna semu dan lain sebagainya.
2. Adanya cacat pada citra sehingga perlu dihilangkan atau diminimumkan *(image restoration).* Contoh : penghilangan kesamaran *(deblurring)* citra

tampak kabur karena pengaturan fokus lensa tidak tepat, penghilangan *noise* dan lan sebagainya.

3. Elemen dalam citra perlu dikelompokan, dicocokan atau diukur *(image segmentation).* Operasi ini berkaitan erar dengan pengenalan pola.
4. Diperlukan ekstraksi ciri-ciri tertentu yang dimiliki citra untuk membantu dalam pengidentifikasian objek *(image analysis).* Proses segmentasi kadangkala diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya.
5. Sebagian citra perlu digabung dengan bagian citra yang lain *(image recontruction).* Contoh : beberapa foto rontgen digunakan untuk membentuk ulang gambar organ tubuh.
6. Citra perlu dimampatkan *(image compression)* contoh : suatu file citra berukuran 256 KB dimampatkan menjadi 49 KB.
7. Menyembunyikan data rahasia (berupa citra atau teks) pada citra sehingga keberadaan data rahasia tersebut tidak diketahui orang *(steganografi* dan *watermaking).*

Beberapa jenis operasi pengolahan citra sebagai berikut :

1. Modifikasi kecemerlangan *(Brightness Modification)*

   Mengubah nilai keabuan atau warna dari gelap menuju terang atau sebaliknya mengubah citra yang terlalu cemerlang menjadi gelap.
2. Peningkatan kontras *(Contrast Enchanment)*

   Dengan peningkatan kontras mak titik yang cendrung gelap menjadi lebih cemerlang.
3. Negasi

   Negai yaitu operasi untuk mendapatkan citra negati*f (negative image)*
4. Pencerminan *(Flipping)*

   Pencerminan merupakan proses menggambar citra ke bentuk kebalikannya seperti ketika bercermin.
5. Rotasi *(Rotation)*

   Rotasi yaitu proses memutar koordinat citra sesuai derajat yang telah ditemtukan.

6. Pemotongan *(Cropping)*

   Pemotongan yaitu proses mengambil bagian dari suatu citra sesuai dengan kebutuhan.

7. Penskalaan *(Scalling)*

   Scalling yaitu proses pengubahan ukuran citra menjadi lebih besar atau enjadi lebih kecil.

8. Model Warna Pada Citra

   Warna adalah persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh objek (Ahmad U, 2005). Setiap warna mempunyai panjang gelombang yang berbeda. Warna merah mempunyai panjang gelombang paling tinggi, sedangkan wrna ungu mempunyai panjang gelombang paling rendah.

   Warna-warna yang diterima oleh mata merupakan hasil kombinasi cahaya dengan panjang gelombang berbeda. Penelitian memperlihatkan bahwa kombinasi warna yang memberikan rentang warna yang paling lebar adalah *red* (R), *green* (G), *blue* (B).

   Dalam pembentukan citra digital model warna yang umum digunakan antara lain sebagai berikut :

   a. Citra RGB

      Citra RGB yang biasa disubet juga citra *true color*, disimpan dalam citra berukuran (m x n) x 3 yang mendefinisikan warna merah (*red*), hijau (*green*) dan biru (*blue*) untuk setiap pikselnya. Warna pada setiap piksel ditentukan berdasarkan kombinasi dari warna *red*, *green* dan *blue* (RGB). RGB merupakan citra 24 bit dengan komponen merah, hijua, dan biru yang masing-masing umumnya bernilai 8 bit sehingga intensitas kecerahan warna sampai 256 level dan kombinasi warnanya kurang dari sekitar 16 juta warna.

   b. Citra Keabuan

      Citra dengan derajat keabuan berbeda dengan citra RGB, citra ini didefinisikan oleh satu nilai derajat warna. Umumnya bernilai 8 bit

sehingga intensitas kecerahan warna sampai 256 level dan kombinasi warnanya 256 varian. Tingkat kecerahan paling rendah yaitu 0 untuk warna hitam dan putih bernilai 255. Untuk mengkonversikan citra yang memiliki warna RGB ke derajat keabuan bisa menggunakan :

$$Gray = \frac{R+G+B}{3} \quad (2.1)$$

Atau

$$Gray = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \quad (2.2)$$

2.3 Tekstur

Tekstur adalah sifat-sifat atau karakteristik yang dimiliki oleh suatu daerah (di dalam citra) yang cukup besar sehingga secara alami sifat-sifat tadi dapat berulang dalam daerah tersebut. Pengertian dari tekstur dalam hal ini adalah keteraturan pola-pola tertentu yang terbentuk dari susunan piksel-piksel dalam citra digital (Ahmad U, 2005).

Untuk membentuk suatu tekstur setidaknya ada dua persyaratan yang harus dipenuhi antara lain :

1. Terdiri dari satu atau lebih piksel yang membentuk pola-pola primitif (bagian-bagian terkecil). Bentuk-bentuk pola primitif ini dapat berupa titk, garis lurus, garis lengkung, luasan dan lain-lain yang merupakan elemen dasar dari sebuah bentuk.
2. Munculnya pola-pola primitif yang berulang-ulang dengan interval jarak dan arah tertentu sehingga dapat diprediksi atau ditemukan karakteristik perulangannya.



Gambar 2.1. Contoh Citra Tekstur Dari Vistex Database

Suatu citra memberikan interpretasi tekstur yang berbeda apabila dilihat dengan jarak dan sudut pandang yang berbeda, manusia memandang tekstur berdasarkan deskripsi yang bersifat acak, seperti halus, kasar, teratur, tidak teratur, dan lain sebgainya. Hal ini merupakan deskripsi yang tidak tepat dan non-kuantitatif, sehingga diperlukan adanya suatu deskrpsi yang kuantitatif (matematis) untuk memudahkan analisis (Ahmad U, 2005).

2.3.1 Analisis Tekstur

Analisis tekstur merukapan dasar dari berbagai macam aplikasi, aplikasi dari analisis tekstur antara lain adalah penginderaan jarak jauh, pencitraan medis, identifikasi kualitas suatu bahan (kayu, kulit, tekstil dan lain-lain). Pada analisis citra, dikategorikan menjadi lima kategori utama yaitu statistis, struktural, geometri, model dasar dan pengolahan sinyal. Pendekatan statistis mempertimbangakan bahwa bahwa internsitas dibangkitkan oleh medan acak dua dimensi, metode ini berdasar pada prekuensi-frekuensi ruang. Contoh metode statistis adalah fungsi autokorelasi, matriks kookurensi, tranformasi fourier, frekuensi tepi, run lenght. Fteknik struktural berkaitan dengan penyusupan bagian-bagian terkecil suatu citra. contoh metode struktural adalah model fraktal. Metode geometri berdasar atas perangkat geometri yang ada pada elemen tekstur. Contoh model dasar adalah medan acak. Sedangkan metode pengolahan sinyal adalah metode yang berdasarkan analisis frekuensi seperti transformasi gabor dan transformasi wavelet (Ahmad U, 2005).

2.4 Metode Matriks Kookurensi

Matriks kookurensi adalah salah satu metode statistik yang dapat digunakan untuk analisis tekstur. Matiks kookurensi dibentuk dari suatu citra dengan melihat intensitas tertentu. Penggunaan metode ini berdasar pada hipotesis bahwa dalam suatu tekstur akan terjadi perulangan konfigurasi atau pasangan aras keabuan. Misal d didefinisikan sebagai jarak antara dua posisi piksel, yaitu (x1,y1) dan (x2,y2) ;

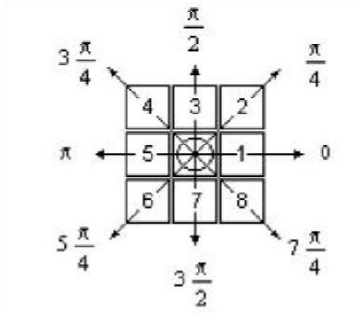
dan θ didefinisikan sebagai sudut diantara keduanya. Maka matriks kookurensi didefinisikan sebagai matriks yang menyatakan disttribusi spasial antara dua piksel yang bertetangga yang memiliki intensitas i dan j, yang memiliki jarak d diantaar keduanya. Matriks kookurensi dinyatakan dengan :

$$P(i,j) = \frac{V(i,j)}{\sum_{i,j=0}^{n-1} V(i,j)} \quad (2.3)$$

Dimana :

- P : Probabilitas munculnya pasangan berderajat keabuan
- $V(i,j)$: Banyaknya pasangan titik-titik dengan tingkat keabuan tertentu

Suatu piksel yang bertetangga yang memiliki jarak d diantara keduanya, dapat terletak di delapan arah yang berlainan, untuk lebih jelasnya dapat dilihat pada Gambar 2.2.



Gambar 2.2. Piksel Bertetangga Dalam Delapan Arah

Setelah memperoleh matriks kookurensi, fitur ciri yang merepresentasikan citra dapat dihitung. Dalam matriks kookurensi, terdapat beberapa ciri tekstur yang dapat diperoleh dari suatu citra yang digunakan sebagai pembeda antara citra dengan kelas tertentu, dengan kelas lainnya. Ciri-ciri tersebut adalah :

1. Energi (*Energy*)

   Energi adalah fitur untuk mengukur konsentrasi pasangan intensitas pada matriks kookurensi. Nilai energi akan semakin membesar bila pasangan piksel yang memenuhi syarat matriks intensitas kookurensi terkosentrasi pada beberapa koordinat dan mengecil bila letaknya menyebar. Rumus yang digunakan untuk menghitung energi adalah :

   $$\sum_{i,j=0}^{n-1} P^2(i,j) \quad (2.4)$$

2. Entropi (*Entropy*)

   Entropi menunjukan ketidakteraturan distribusi intensitas suatu citra pada matriks kookurensi persamaanya untuk menghitung entropi adalah :

   $$\sum_{i,j=0}^{n-1} \mathrm{P}(i,j)(-\ln P(i,j)) \quad (2.5)$$

3. Kontras (*Contrast*)

   Kontras adalah fitur yang digunakan untuk mengukur kekuatan perbedaan intensitas dalam citra. nilai kontras membesar jika variasi intensitas citra tinggi dan menurun bila variasi rendah. Persamaan yang digunakan untuk menghitung kontras adalah :

   $$\sum_{i,j=0}^{n-1} Pi,j(i-j)^2 \quad (2.6)$$

4. Homogenitas (*Homogenity*)

   Homogenitas digunakan untuk mengukur kehomogenan variasi intensitas citra. Nilai homogenitas akan semakin membesar bila variasi intensitas dalam citra mengecil. Persamaan yang digunakan untuk menghitung homogenitas adalah :

   $$\sum_{i,j=0}^{n-1} \frac{P(i,j)}{1+|\mathrm{i}-\mathrm{j}|} \quad (2.7)$$

5. Korelasi (*Correlation*)

   Ciri ini menunjukan tingkat korelasi antar piksel dalam sutau citra. persamaan yang digunakan untuk menghitung korelasi adalah :

   $$\sum_{i,j=0}^{n-1} \frac{(i-\mu x)(j-\mu y)Pd(i,j)}{\sigma x \sigma y} \quad (2.8)$$

   Dimana :

   $$\mu x = \sum_{i,j=0}^{n-1} \mathrm{p}(i,j)$$

   $$\mu y = \sum_{j.i=0}^{n-1} \mathrm{p}(i,j)$$

   $$\sigma x^2 = \Sigma \mathrm{i}(i-\mu x)^2 \Sigma \mathrm{j} P_d(\mathrm{i},\mathrm{j})$$

   $$\sigma y^2 = \Sigma \mathrm{j}(j-\mu y)^2 \Sigma \mathrm{i} P_d(\mathrm{i},\mathrm{j})$$

Dalam persamaan diatas, notasi P melambangkan probabilitas, yang bernilai mulai dari nol hingga satu, yaitu nilai elemen dalam matriks kookurensi, sedangkan i dan j, melambangkan pasangan intensitas yang berdekatan, yang dalam matriks

kookurensi masing-masing menjadi nomor baris dan nomor kolom. Ciri-ciri dari matriks kookurensi yang memiiki nilai sama atau hampir sama dapat digunakan salah satunya saja, sehingga dapat mengurangi beban komputasi dan untuk menghindari pembentukan matriks kookurensi yang terlalu besar dan menyebabkan proses perhitungan menjadi lambat biasanya ukuran matriks kookurensi dibatasi (Ahmad U, 2005).

2.5 Jaringan Saraf Tiruan (JST)

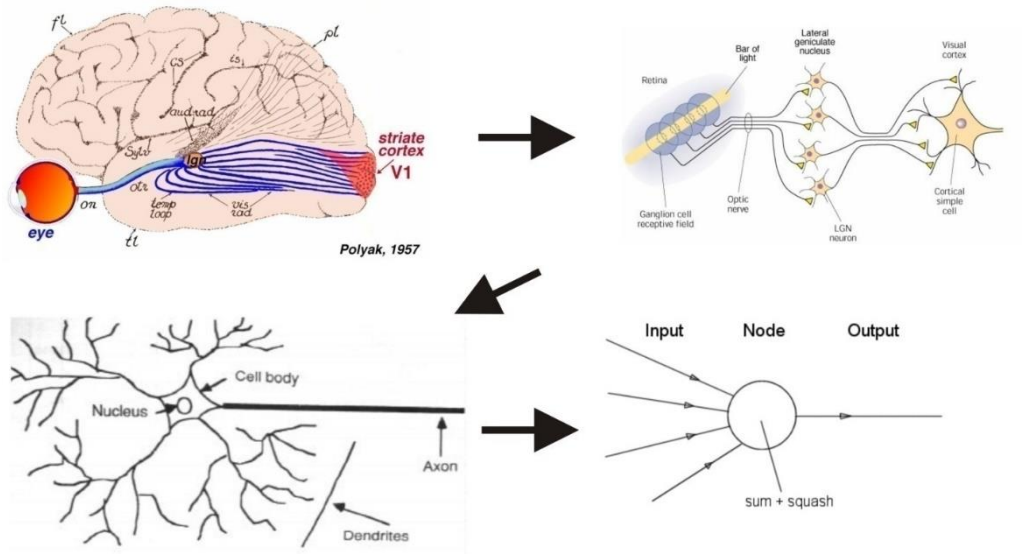
Jaringan saraf tiruan adalah salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia. Istilah buatan ini diimplementasikan dengan menggunakan program komputer yang mampu menyelesaikan sejumlah proses perhitungan selama proses pembelajaran (Kristanto A, 2004).

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

Tahun 1940-an, para ilmuwan menenmukan bahwa psikologi otak sama dengan mode pemrosesan yang dilakukan oleh komputer. Tahun 1943, Mculloch dan Pitts merancang model formal yanag pertama kali sebagai perhitungan dasar neuron. Tahun 1954, Farley dan Clark mensetup model-model untuk relasi adaptif stimulus-respon dalam jaringan random. Tahun 1958, Rosenblatt mengembangkan konsep dasar tentang perception untuk klasifikasi pola. Tahun 1974, Werbos memperkenalkan algoritma backpropagation. Tahun 1975, Little dan Shaw menggambarkan jaringan saraf dengan probabilistik (Kristanto A, 2004).

2.5.1 Komponen dan Arsitektur

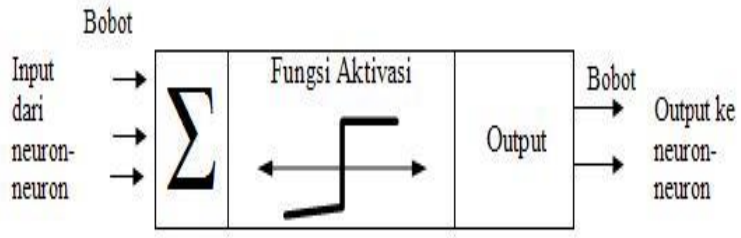
Pembuatan struktur jaringan saraf tiruan diilhami oleh struktur jaringan biologi, khususnya jaringan otak manusia.



Gambar 2.3. Struktur Dasar JST Dan Struktur Sederhana Sebuah Neuron

Jaringan saraf terdiri dari beberapa komponen berikut (Muis S, 2006) :

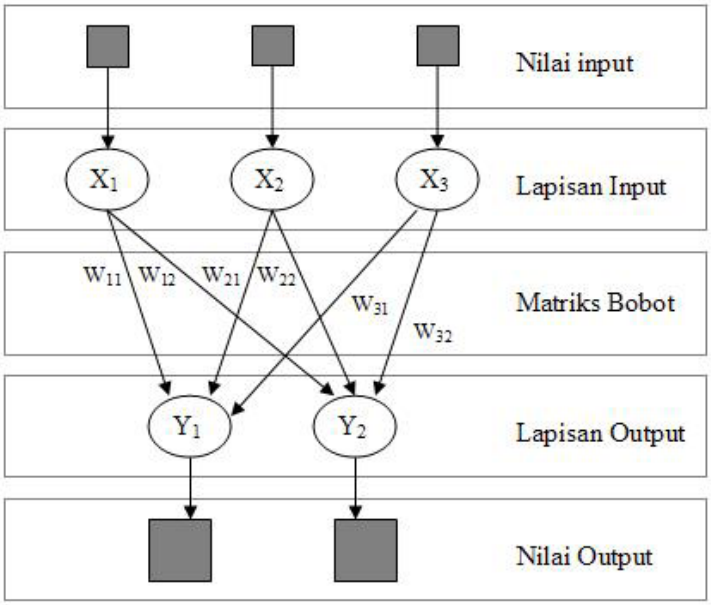
1. Neuron, sel saraf yang akan mentrsnformasikan informasi yang diterima melalui sambungan keluarnya menuju neuron-neuron lain.
2. Hubungan antar neuron dikenal dengan nama bobot.
3. Neuron layers, neuron-neuron aka dikumpulkan dalam lapisan-lapisan (layer).
4. Informasi yang diberikan pada jaringan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



Gambar 2.4. Model Tiruan Sebuah Neuron

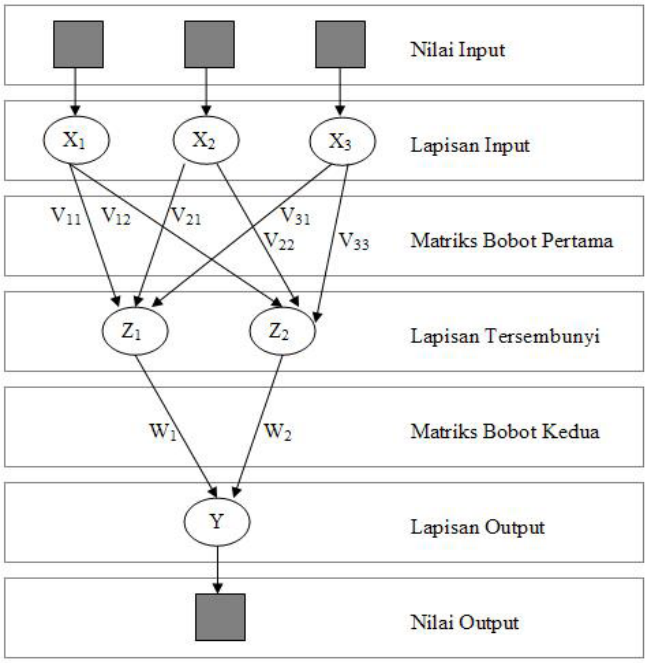
Faktor terpenting untuk menentukan kelakuan suatu neuron adalah fungsi aktivasi dan pola bobotnya. Ada beberapa arsitektur jaringan saraf, antara lain :

1. Jaringan dengan lapisan tunggal (*single layer net*)
   a. Memiliki satu lapisan dengan bobot terhubung
   b. Menerima input kamudian langsung mengolahnya menjadi output tanpa harus melewati lapisan tersembunyi.



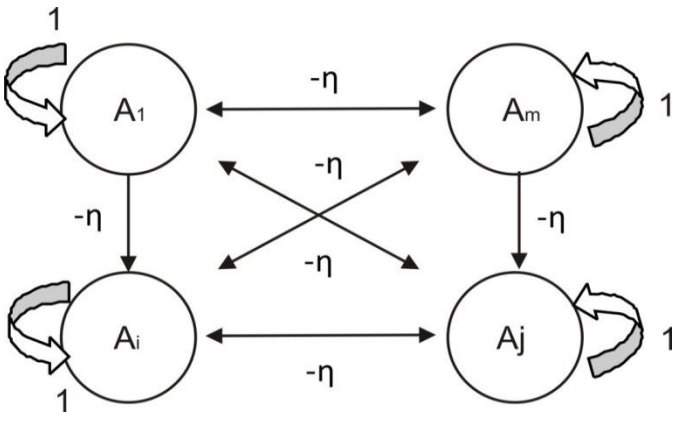
Gambar 2.5. Model Single Layer

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



Gambar 2.6. Model Multi Layer

3. Jaringan dengan lapisan kompetitif (*Competitive layer net*)
   - Hubungan antar neuron pada lapisan kompetitif tidak diperlihatkan pada diagram arsitektur



Gambar 2.7. Model Competitive Layer

2.5.2 Fungsi aktivasi

Mengaktifkan saraf tiruan berarti mengaktifkan setiap neuron yang dipakai pada jaringan tersebut. Banyak fungsi yang dipakai sebagai pengaktif, diantaranya (Kristanto A, 2004) :

1. Fungsi Undak Biner (*Hard Limit*)

   Jaringa dengan lapisan unggal sering menggunakan fungsi undak untuk mengkonversi input dari suatu variabel yang bernilai kontinu ke suatu output biner. Fungsi hard limit dirumuskan

   $$Y = \begin{cases} 0, jika\ x \leq 0 \\ 1, jika\ x > 0 \end{cases} \quad (2.9)$$

2. Fungsi Undak Biner (*Threshold*)

   Fungsi ini sering disebut fungsi nilai ambang atau fungsi *headviside*, dirumuskan :

   $$Y = \begin{cases} 0, jika\ x < \theta \\ 1, jika\ x \geq \theta \end{cases} \quad (2.10)$$

3. Fungsi Bipolar

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

   $$Y = \begin{cases} 1, jika\ x > 0 \\ 0, jika\ x = 0 \\ -1, jika\ x < 0 \end{cases} \quad (2.11)$$

4. Fungsi Bipolar (Dengan *Threshold*)

   Fungsi yang menghasilkan output berupa 1,0 atau -1

   $$Y = \begin{cases} 1, jika\ x\ \geq \theta \\ -1, jika\ x < \theta \end{cases} \quad (2.12)$$

5. Fungsi Linear (Identitas)

   Fungsi linear memilki nilai output yang sama dengan nilai input, dirumuskan :

   $$Y = X \quad (2.13)$$

6. Fungsi Sturating Linear

Fungsi ini dirumuskan :

$$Y = \begin{cases} 1, jika\ x \geq 0 \\ x + 0{,}5\ , jika - o, 5 \leq x \leq 0{,}5 \\ 0, jika\ x \leq 0 \end{cases} \quad (2.14)$$

7. Fungsi Symetric Saturating Linear

   Fungsi ini dirumuskan :

$$Y = \begin{cases} 1, jika\ x \geq 0 \\ x\ , jika - 1 \leq x \leq 1 \\ -1, jika\ x \leq -1 \end{cases} \quad (2.15)$$

8. Fungsi Sigmoid Biner

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

$$y = f(x) = \frac{1}{(1+e^{\acute{\alpha}x})} \quad (2.16)$$

9. Fungsi Sigmoid Bipolar

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

$$y = f(x) = \frac{1-e^{-x}}{(1+e^{-x})} \quad (2.17)$$

2.5.3 Proses Pembelajaran

Metode pembelajaran terdiri dari beberapa metode (Muis S, 2006) :

1. Metode Pelatihan Terbimbing

   Metode pembelajaran pada jaringan saraf disebut terawasi jika output yang diharapkan telah diketahui sebelumnya. Metode ini memasukan target keluaran alam data untuk proses pelatihan. Beberapa metode terbimbing diantaranya : Jaringan Single Perceptron, Hebb Rule, Delat Rule, Backpropagation, Hetroassociative Memory, Bidirectional Associative Memory dan Learning Vector Quantization.

2. Metode Pelatihan Tak Terbimbing

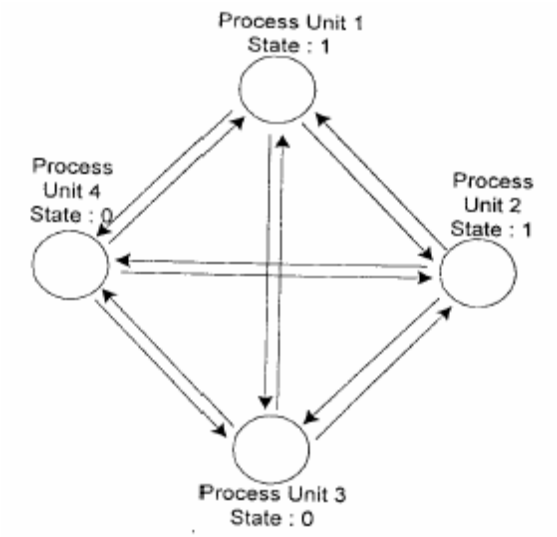
   Metode pelatihan tak terbimbing adalah pelatihan tanpa memerlukan target pada keluarannya. Proses pelatihan berdasarkan proses transformasi dari bentuk variabel kontinyu menjadi variabel diskrit yang dikenal dengan kuantisasi vektor. Jaringan yang digunakan untuk proses pelatihan tak terbimbing ini adalah jaringan umpan balik (*feedback network*). Beberapa metode tak terbimbing adalah : Metode Kohonen/Self-Organizing Map (SOM) dan Metode Hopfield.

2.6 Metode Hopfield

Pada tahun 1982, John Hopfield dari *California Institute of Technology* merancang sebuah jaringan saraf tiruan yang kemudian dikenal dengan nama jaringan hopfield. Dalam jaringan hopfield, semua neuron saling berhubungan penuh. Neuron yang satu mengeluarkan output dan kemudian menjadi input bagi semua neuron yang lain. Proses pengiriman dan penerimaan sinyal antar neuron ini secara *feedback* tertutup dan terus menerus sampai dicapai kondisi stabil (Kristanto A, 2004).

Dalam model diskritnya, jaringan hopfield bobot sinaptiknya menggunakan vektor biner dimensi n. Model semacam ini berisi n neuron dan jaringannya terdiri dari n(n-1) interkoneksi dua jalur. Cara pemahaman paling mudah JST Hopfield berdasarkan konsep di atas dapat disajikan dalam gambaran umumnya adalah sebagai berikut :

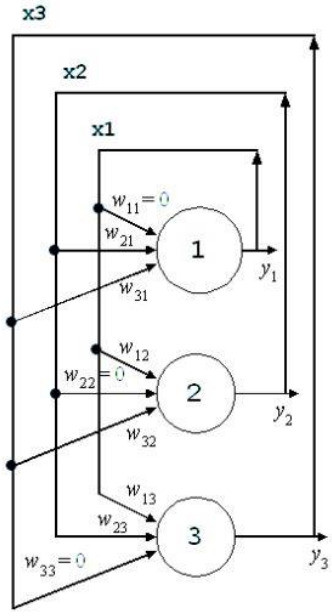
Bila ada sebuah JST yang terbentuk dari N X N neuron dimana N adalah jumlah variabel dari objek yang akan dioptimasi. Setiap neuron terhubung penuh satu sama lainnya. Ilustrasi pernyataan tersebut adalah sebagai berikut :



Gambar 2.8. Layout dari JST Hopfield

JST Hopfield merupakan salah satu metode optimasi untuk pencarian nilai minimum dari kombinasi fungsi objektif. Pada gambar 2.9 semua neuron saling berhubungan penuh. Neuron yang satu mengeluarkan output dan kemudian menjadi input bagi neuron yang lain ouptut dari setiap simpul diumpanbalikan ke input dari simpu lainnya melalui bobot koneksi W_{ij} yang tetap. Nilai W_{ij} mula-mula

diinisialisasikan menggunakan rumus yang diberikan hopfield untuk seluruh pola-pola contoh (Kristanto A, 2004).



Gambar 2.9. Topologi JST Hopfield

ada beberapa versi dari algortima jaringan Hopfield. Untuk deskripsi dari algoritma Hopfield menggunakan input vektor yang berupa angka biner. Untuk menyimpan sekumpulan pola biner digunakan notasi s(p), p=1 ,..., P,

dimana:

$$Wij = \sum_p[2\text{si(p)} - 1][2\text{sj(p)} - 1] \quad (2.18)$$

untuk i|=j dan Wij=0

sedangkan versi dari deskripsi algortima Hopfiled lainnya menggunakan vektor input yang berupa angka bipolar. Untuk menyimpan sekumpulan pola bipolar digunakan notasi s(p),p=1,...,p, dimana :

$$Wij = \sum_p[\text{si(p)sj(p)}] \quad (2.19)$$

untuk i|=j dan Wij=0

untuk menghitung jarak terdekat adalah dengan rumus seperti berikut :

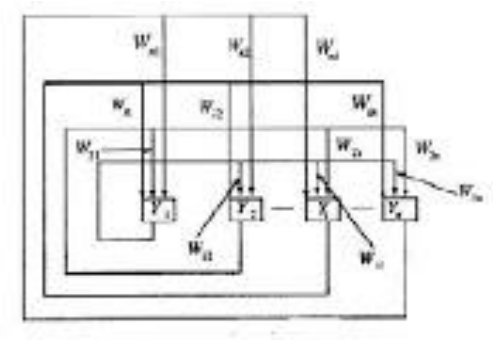
$$\sqrt{(a-c)^2 + (b-d)^2} \quad (2.20)$$

Pola yang tidak dikenal dimasukkan ke jaringan satu kali saja yaitu pada waktu nol. Kemudian jaringan beriterasi hingga konvergen. Konvergen adalah kondisi dimana output yang sudah tidak bisa berubah lagi untuk iterasi selanjutnya. Pola yang dinyatakan oleh simpul-simpul output setelah jaringan konvergen adalah output jst.

Kemampuan JST Hopfield untuk mengenali pola ditentukan dengan rumus 0.12 x N. N adalah banyaknya dimensi yang digunakan. Misalkan dimensi yang digunakan 100, maka pola yang akan dikenali sebanyak 15 buah. Dimensi adalah suatu vektor yang mewakili banyaknya baris dan kolom yang digunakan dari contoh diatas berarti baris yang digunakan berjumlah 100 dan kolom yang digunakan berjumlah 100 juga.

Jaringan saraf tiruan dilatih untuk mengenali berbagai input atau masukan yang diberikna kepadanya. Misalkan, jaringan tersebut dilatih untuk mengenali berbagai versi angka 2. Setelah mengalami cukup latihan, maka jaringan itu dapat menghasilkan angka 2 secara sempurna dari masukan angka 2 yang rusak atau kena distorsi.

Model atau arsitektur dari jaringan Hopfield ditunjukan pada gambar 2.10., dimana dalam gambar tersebut terdiri N buah simpul yang mengandung *nonlinieritas hard limiter* dan input serta output biner yang dapat bernilai 1 atau -1 (Kristanto A, 2004).



Gambar 2.10. Arsitektur Jaringan Hopfield

2.7 OOP (Object Oriented Progamming)

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapann prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktifitas analisis beorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek (Nugroho A, 2005).

Pada saat ini, metode berorientais objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentranformasi hasil dari satu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan saat ini sangat beragam (aplikasi bisnis, *real-time, utility* dan sebagainya) dengan platform yang berbeda-beda, sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi ke semua jenis aplikasi.

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut :

a. Meningkatkan Produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reuseable*).

b. Kecepata Pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan mennyababkan berkurangnya kesalahan pada saat pengkodean

c. Kemudahan Pemeliharaan

   Karena dengan model objek, pola-pola yang cendrung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering diubah-ubah.

d. Adanya Konsistensi

Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

e. Meningkatkan Kualitas Perangkat Lunak

Karena adanya pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengambangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Berikut beberapa contoh bahasa pemograman yang mendukung pemrograman berorinentasi objek adalah :

a. Smalltalk

Smalltalk adalah salah satu bahasa pemograman yang diekmbangkan untuk mendukung pemrograman beroirentasi objek.

b. Bahasa Pemrograman Eiffel

Eiffel merupakan bahsa pemrograman yang kembangkan untuk mendukung pemrograman berorientasi objek oleh Bertrand Meyer dan *compiler*.

c. Bahasa Pemrograman (Web) PHP

Php dibuat pertama kali oleh seorang perekayasa perangkat (software engineering) yang bernama Rasmus Lerdoff.

d. Bahasa Pemrograman C++

C++ merupakan pengembangan lebih lanjut dari bahasa pemrograman C untuk mendukung pemrograman berorientasi objek.

e. Bahasa Pemrograman Java

Java dikembangkan oleh perusahaan Sun Microsystem. Java menurut definisi dari Sun Microsystem adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *standalone* ataupun pada lingkungan jaringan.

2.7.1 Konsep Dasar Berorientasi Objek

Berikut adalah konsep dasar pemrograman berorientasi objek :

a. Objek (*Object*)

Objek adalah abtraksi dan sesuatu yang mewakili dunia nyata seperti benda, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi dan mempunyai operasi yang dapat diterapkan atau dapat berpengaruh pada status objeknya.

b. Kelas (*Class*)

Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama yang mungkin lahir atau diciptakan dalam kelas tersebut.

c. Pembungkusan (*Encapsulation*)

Pembungkusan atribut data dan layanan yang mempunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

d. Pewarisan (*Inheritance*) dan Generalisasi/Spesialisasi

Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dirinya.

e. Metode

Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural.

f. Polimorfisme

Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

2.7.2 UML (*Unified Modelling Language*)

Unified Modelling Language (UML) adalah sekumpulan spesifikasi yang dikeluarkan oleh OMG. UML terbaru adalah UML 2.3 yang terdiri dari 4 macam spesifikasi, yaitu : *Diagram Interchange Spesification,* UML *Infrastrukture*, UML *Superstrukture, dan Object Constraint Language* (OCL). Pada UML 2.3 terdiri 13 macam diagram yang dikelompokan pada 3 kategori, yaitu (Sutopo & Masya, 2005):

A. Struktur Diagram, yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. Pada Struktur Diagram dibagi menjadi 6 bagian (Sutopo & Masya, 2005) :

1. Diagam Kelas

   Diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut attribut dan metode atau operasi.

2. Diagram Objek

   Digram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem.

3. Diagram Komponen

   Diagram komponen dibuat untuk menunjukan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem.

4. *Composite Structure Diagram*

   Composite Structure Diagram baru mulai ada pada UML versi 2.0. diagram ini dapat digunakan untuk menggambarkan struktur dari bagian-bagian yang saling terhubung maupun mendeskripsikan struktur pada saat berjalan (*runtime*) .

5. *Package Diagram*

   Package diagram menyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram UML. Hampir semua diagram dalam UML dapat dikelompokan menggunakan *Package Diagram*.

6. *Deployment Diagram*

   Deployment diagram menunjukan konfigurasi komponen dalam proses eksekusi aplikasi.

B. *Behavior Diagram,* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. Pada *Behavior Diagram* dibagi menjadi 3 bagian :

1. *Use Case Diagram*

   Use case diagram merupakan pemodelan untuk kelakuan (behavior) sistem informasi yang akan dibuat,. *Use case* mendeskripsikan sebuah

interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat.

2. *Activity Diagram*

   Activity diagram menggambarkan *workflow* atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak.

3. *State Machine Diagram*

   State machine diagram digunakan untuk menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem atau objek.

C. *Interactions Diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi antar subsistem pada suatu sistem. Pada Interactions Diagram dibagi menjadi 4 bagian :

1. *Sequence Diagram*

   Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek.

2. *Communication Diagram*

   Communication diagram menggambarkan interaksi antar bojek/bagian dalam bentuk urutan pengiriman pesan. Diagram komunikasi merepresentasikan informasi yang diperoleh dari diagram kelas, diagram *sequence*, dan diagram *use case* untuk mendeskripsikan gabungan antara struktur statis dan tingkah laku dinamis dari suatu sistem.

3. *Timing Diagram*

   Timing diagram merupakan diagram yang fokus pada penggambaran terkait batas waktu.

4. *Interaction Overview Diagram*

   Interaction overview diagram mirip dengan diagram aktivitas yang berfungsi untuk menggarbarkan sekumpulan urutan aktivitas, diagram ini adlaah bentuk aktivias diagram yang setiap titik merepresentasikan diagram interaksi.

2.8 Bahasa Pemograman C#

C# (tanda '#' dibaca "Sharp") merupakan bahasa pemograman baru yang diciptakan Microsoft seacra khusus sebagai salah satu bahasa pemrograman dalam teknologi .Net sebagai bahasa baru, C# tidak berevolusi dari bahasa C# versi bukan teknologi .Net. dengan demikian C# dapat memaksimalkan kemampuannya tanpa khawatir dengan masalah kompatibilitas dengan versi-versi sebelumnya. Keharusan sebuah perangkat lunak untuk tetap dapat kompatibel dengan versi-versi sebelumnya sebagaimana yang terjadi pada Visual Basic (VB) maupun C++biasanya menghambat optimalitas kemampuan dari perangkat lunak tersebut (Hartanto, B., 2008).

Sejak diluncurkan pada tahun 2000, C# dengan cepat merebut hati progammer C++ bahkan VB. Dengan tata cara penulisan yang mirip C++ dan *interface* mirip VB 6.0 menurut wikipedia, sebuah ensiklopedia gratis di internet pengguna C# .Net pada saat ini sudah melebihi pengguna VB.Net. sementara itu jumlah pengguna bahasa pemrograman lain masih berada dibawah jumlah pengguna VB .Net. masih menurut wikipedia, jumlah buku C# yang terjual pun berada dikisaran 2 hingga 3 kali lebih banyak dari jumlah buku VB yang terjual.

Dari informasi ini dapat disimpulkan bahwa C# merupakan bahasa pemrogrman baru yang sedang berkembang dan dapat diterima dengan baik oleh kebanyakan progammer dan kalangan industri. Di Microsoft sendiri, C# merupakan bahasa pemrograman yang digunakan untuk membuat perangkat lunak yang bertteknologi .Net dengan demikian dapat diperkirakan bahwa C# akan menjadi bahasa pemrograman yang akan banyak digunakan di masa-masa mendatang (Hartanto, B., 2008).