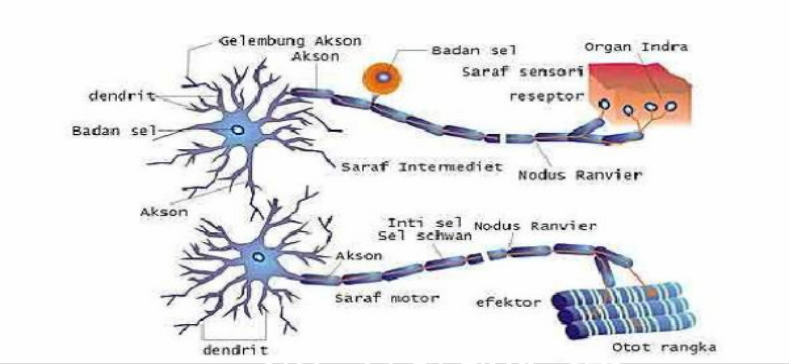
BAB II
LANDASAN TEORI

2.1 Jaringan Syaraf Biologis Manusia

Jaringan syaraf tiruan (*Artificial Neural Network*) atau *ANN* merupakan model yang meniru cara kerja jaringan biologis. Otak manusia terdiri atas sel-sel syaraf yang disebut *neuron*, yang berjumlah sekitar 10^9 *neuron. Neuron – neuron* ini terbagi atas *group – group* yang disebut jaringan, yang dibedakan atas fungsinya dan setiap *group* mengandung ribuan *neuron* yang saling berhubungan. Kecepatan proses tiap jaringan ini sebenarnya jauh lebih kecil dibandingkan dengan kecepatan proses komputer yang ada pada saat ini, namun karena otak terdiri atas jutaan jaringan yang bekerja secara paralel, maka otak dapat mengerjakan pekerjaan yang jauh lebih kompleks dibandingkan dengan apa yang dapat dikerjakan oleh komputer yang hanya mengandalkan kecepatan.



Gambar 2.1 Jaringan Syaraf Manusia

Jaringan syaraf manusia tersusun atas sel – sel syaraf atau *neuron.* Tiap *neuron* terdiri atas badan sel syaraf, cabang *dendrit* dan cabang *akson,* cabang – cabang inilah yang menghubungkan tiap – tiap sel syaraf sehingga membentuk jaringan syaraf.
Terdapat 3 macam sel syaraf :

1. Sel Syaraf Sensorik

   Berfungsi menghantarkan rangsangan dari penerima ransangan (*reseptor)* ke sumsum tulang belakang

2. Sel Syaraf Motorik

   Berfungsi menghantarkan *impuls* motorik dari susunan syaraf pusat ke efektor.

3. Sel Syaraf Penghubung

   Merupakan penghubung sel syaraf yang satu dengan sel syaraf yang lain.

2.2 Paradigma Jaringan Syaraf Tiruan

Paradigma jaringan syaraf tiruan dapat digunakan untuk melakukan komputasi pengenalan suatu objek yang memiliki kemampuan memorisasi dan sekaligus melakukan generalisasi. Memorisasi di sini adalah kemampuan JST untuk memanggil kembali secara sempurna pola yang telah dipelajari. Generalisasi di sini terlihat pada kemampuan sebuah sistem jaringan syaraf tiruan yang telah terlatih untuk menghasilkan *output* yang dapat diterima ketika ia menemukan sebuah *input* yang tidak pernah ia dapatkan selama proses pembelajaran (*learning process*) [1]. Jaringan ini biasanya diimplementasikan dalam bentuk komponen – komponen elektronik atau tersimulasi secara *software* dalam sebuah komputer. Dan untuk memperoleh hasil yang baik, sistem jaringan syaraf tiruan memerlukan sebuah *inter*-koneksi yang besar dari sel – sel komputasi sederhana yang biasa disebut dengan "*neurons"* atau "unit – unit proses"

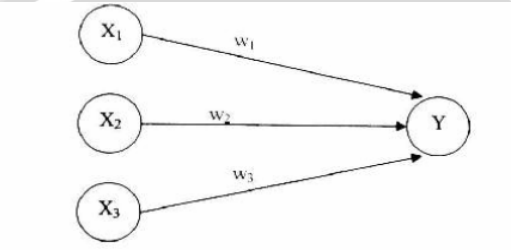
Di sinilah dapat diketahui bahwa paradigma jaringan syaraf tiruan dapat menyerupai dua fungsi otak manusia yaitu :

1. Pengetahuan jaringan syaraf tiruan diperoleh melalui sebuah proses pembelajaran yang kontinu.
2. Koneksi – koneksi antar unit – unit proses memiliki bobot atau nilai informasi yang digunakan untuk menyimpan hasil pembelajaran yang telah dilakukan oleh jaringan syaraf tiruan tersebut.

Jaringan syaraf tiruan terdiri atas sejumlah pemroses yang sangat sederhana yang disebut dengan *node* atau simpul. Simpul ini dianalogikan seperti *neuron* yang ada di otak manusia. Kumpulan simpul – simpul yang membentuk suatu konfigurasi tertentu dikenal sebagai jaringan syaraf tiruan. Jaringan syaraf tiruan terdiri atas lapisan masukan dan keluaran. Tetapi ada juga yang mempunyai lapisan tersembunyi di antara lapisan masukan dan keluaran. Simpul yang ada pada lapisan masukan disebut unit masukan. Pada unit masukan tidak memproses suatu informasi tetapi hanya menyebarkan atau menyalurkan ke unit lain. Sedangkan simpul yang ada pada lapisan

tersembunyi dan lapisan keluaran menghasilkan keluaran yang berupa suatu bentuk *non linear*.

Secara prinsip jaringan dibangkitkan serangkaian masukan (*input*) yang dikalikan dengan suatu faktor penimbang tertentu yang *analog* dengan tegangan sinapsis. Dan kemudian dijumlahkan semua masukan tersebut untuk menentukan tingkat aktivasi *neuron*. Pada gambar 2.2 diperlihatkan kontruksi dasar sebuah *neuron* tiruan.



Gambar 2.2 Ilustrasi *Neuron* Tiruan

Misalnya, sebuah *neuron* Y (Gambar 2.2) menerima *input* dari *neuron* X_1, X_2 dan X_3. Bobot nilai yang menghubungkan *neuron* X_1, X_2 dan X_3 ke neuron Y adalah W_1, W_2 dan W_3. Maka input jaringan Y_{in} pada *neuron* Y adalah jumlah dari sinyal bobot dari masing – masing *neuron* X_1, X_2 dan X_3 di mana $Y_{in} = W_1X_1 + W_2X_2 + W_3X_3$

2.3 Komponen Jaringan Syaraf Tiruan

Dengan mengambil ide dari jaringan syaraf manusia, komponen-komponen pada JST adalah :

1. *Neuron* Tiruan (*Artificial Neuron*)

   JST disusun oleh unit dasar yang disebut *neuron* tiruan yang merupakan elemen pemrosesan dalam jaringan, di mana semua proses perhitungan dilakukan disini.

2. Lapisan (*layer*)

   JST disusun oleh kumpulan *neuron* yang berhubungan dan dikelompokkan pada lapisan-lapisan (*layer*). Terdapat tiga lapisan, yaitu : lapisan *input* (*input layer*), lapisan tersembunyi (*hidden layer*), dan lapisan *output* (*output layer*).

3. Masukan (*Input*)

   JST hanya dapat memroses data masukan berupa data numerik. Sehingga apabila masalah melibatkan data kualitatif seperti grafik, *image*, sinyal atau

suara, data harus ditransformasikan dulu ke dalam data *numeric* yang *equivalen* sebelum dapat diproses oleh jaringan syaraf tiruan.

4. Keluaran (*Output*)

   Keluaran dari JST adalah pemecahan terhadap masalah. Data keluaran merupakan data numerik.

5. Bobot (*Weight*)

   Bobot pada JST menyatakan tingkat kepintaran sistem. Walaupun sebenarnya bobot tersebut hanya sebuah deretan angka-angka saja. Bobot sangat penting untuk jaringan syaraf tiruan, di mana bobot yang optimal akan memungkinkan sistem menerjemahkan data masukan secara benar dan menghasilkan keluaran yang diinginkan.

2.4 Arsitektur Jaringan Syaraf Tiruan

Aturan yang menata *neuron – neuron* yang tergabung dalam sebuah jaringan syaraf tiruan terintegrasi dengan algoritma – algoritma pembelajarn yang digunakan untuk melatih jaringan syaraf tiruan tersebut. Dengan kata lain, macam – macam algoritma dan aturan – aturan (*rules)* yang digunakan dalam desain jaringan syaraf tiruan dapat digolongkan sebagai struktur arsitektur dari jaringan syaraf tiruan tersebut. Secara umum, jaringan syaraf tiruan dapat dikelompokan menjadi tiga kelas fundamental yang berbeda menurut arsitekturnya.

2.4.1 Jaringan Syaraf Tiruan *Feedforward Single Layer*

Dalam jaringan syaraf tiruan yang memiliki *layer*, *neuron – neuron* yang ada terorganisir mengikuti bentuk dari pada *layer – layer* yang ada. Dalam sebuah bentuk jaringan sederhana yang memiliki *layer*, bisa ditemukan *layer input* yang terproyeksikan pada *layer output*, namun tidak sebaliknya. Dengan kata lain, jaringan ini benar – benar tertutup untuk bergerak maju (*Feedforward)* atau biasa disebut tipe *acyclic* karena perpindahan informasi hanya terjadi satu arah saja.

Pada jaringan ini baik *layer input* maupun *layer output* sama – sama memiliki *node* yang sama . Jaringan ini disebut jaringan *single – layer* karena mengacu pada *node* komputasi pada *layer output* (*neurons*). Tidak terjadi perhitungan pada apa yang ada pada *input layer* karena tidak ada komputasi yang terjadi di sana.

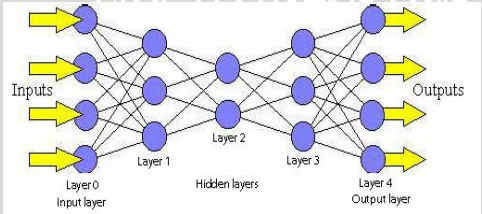
2.4.2 Jaringan Syaraf Tiruan *Feedforward Multilayer*

Yang membedakan jaringan syaraf tiruan tipe *feedforward multilayer* ini dengan jaringan syaraf tiruan tipe *feedforward single-layer* adalah keberadaan satu atau lebih layer tersembunyi atau yang biasa disebut *hidden layer* . Fungsi dari *neuron – neuron* yang tersembunyi ini adalah sebagai jembatan antara *input* eksternal dan *input* dari jaringan demi tujuan tertentu.

Dengan menambahkan satu atau lebih *layer* tersembunyi, jaringan akan mampu melakukan ekstrasi statistik yang lebih baik. Peningkatan kinerja jaringan ini disebabkan oleh tambahan kumpulan sel koneksi sinaptik dan juga oleh tambahan jangkauan dimensi yang diperoleh dari interaksi antar *neuron.*

Titik – titik pada *layer* masukan menjadi masukan pada masing – masing elemen dari setiap vektor *input*, di mana sinyal –sinyal ini akan dipakai lebih lanjut pada titik – titik komputasi pada *layer* tersembunyi pertama. Sinyal – sinyal *output* dari *layer* ini akan menjadi sumber *input* pada *layer* berikutnya dan seterusnya dalam jaringan tersebut. Pada umumnya, setiap *neuron* pada setiap *layer* pada jaringan mendapatkan *input* mereka dari hasil *output* dari *layer* sebelumnya.

Kumpulan sinyal – sinyal *output* dari setiap *neuron* pada *layer output* mewakili respon jaringan syaraf tiruan tersebut secara keseluruhan terhadap pola aktivasi yang diberikan oleh titik – titik *input*. Jaringan syaraf tiruan pada gambar 2.3 dapat dikatakan jaringan yang terkoneksi secara keseluruhan karena setiap titik pada setiap *layer* yang ada terhubung dengan setiap titik yang ada pada *layer* berikutnya. Tetapi apabila ada koneksi sinaptikal yang hilang maka, jaringan syaraf tiruan tersebut dikatakan terhubung sebagian.

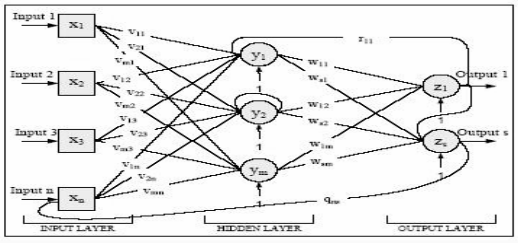


Gambar 2.3 Jaringan Syaraf Tiruan dengan *Hidden Layer*

2.4.3 Jaringan Syaraf Tiruan Berulang

Perbedaan antara sebuah jaringan syaraf tiruan yang berulang dengan sebuah jaringan syaraf tiruan tipe *feedforward* adalah pada jaringan syaraf tiruan berulang (*recurrent network),* terdapat setidaknya satu buah *feedback – loop* atau perulangan

umpan balik. Sebagai contoh, sebuah jaringan Syaraf tiruan berulang dapat hanya memiliki satu buah *layer* saja. Namun ini bukan berarti jaringan syaraf tipe ini tidak dapat memiliki *layer* tersembunyi. Pada gambar 2.4 dapat dilihat bahwa sumber *input*-an sebuah *neuron* dapat diperoleh dari *output neuron – neuron* lainnya dan juga dari *neuron – neuron* yang terdapat pada *layer* tersembunyi



Gambar 2.4 Jaringan Syaraf Tiruan Berulang dengan *Layer* Tersembunyi

2.5 Pembelajaran Jaringan Syaraf Tiruan

Pada otak manusia, informasi yang dilewatkan dari dari *neuron* ke *neuron* yang lainnya berbentuk rangsangan listrik melalui *dendrit*. Jika rangsangan tersebut diterima oleh suatu *neuron*, maka *neuron* tersebut akan membangkitkan *output* ke semua *neuron* yang berhubungan dengannya sampai informasi tersebut sampai ke tujuannya yaitu sampai terjadinya suatu reaksi [3]. Jika rangsangan diterima terlalu halus, maka *output* yang dibangkitkan oleh *neuron* tersebut tidak akan direspon. Tentu saja sangatlah sulit untuk memahami bagaimana otak manusia bisa belajar. Selama proses pembelajaran, terjadi perubahan yang cukup berarti pada bobot – bobot yang menghubungkan antar *neuron*. Apabila ada rangsangan yang sama dengan rangsangan yang telah diterima oleh *neuron*, maka *neuron* akan memberikan reaksi dengan cepat. Namun apabila kelak ada rangsangan yang berbeda dengan apa yang telah diterima oleh *neuron,* maka *neuron* akan segera beradaptasi untuk memberikan reaksi yang sesuai.

Jaringan syaraf akan mencoba untuk mensimulasikan kemampuan otak manusia untuk belajar. Jaringan syaraf tiruan juga tersusun atas *neuron – neuron* dan *dendrit*. Tidak seperti model biologis, jaringan syaraf memiliki struktur yang tidak dapat diubah, dibangun oleh sejumlah *neuron,* dan memiliki nilai tertentu yang menunjukan seberapa besar koneksi antar *neuron* (yang dikenal dengan nama bobot). Perubahan yang terjadi selama proses pembelajaran adalah perubahan nilai bobot. Nilai bobot akan bertambah, jika informasi yang diberikan oleh *neuron* yang bersangkutan tersampaikan, sebaliknya jika informasi tidak disampaikan oleh suatu *neuron* ke *neuron* yang lainnya, maka nilai bobot yang menghubungkan keduanya akan dikurangi. Pada saat pembelajaran

dilakukan pada input yang berbeda, maka nilai bobot akan diubah secara dinamis hingga mencapai suatu nilai yang seimbang. Apabila nilai ini telah tercapai mengindikasikan bahwa tiap – tiap *input* telah berhubungan dengan *output* yang diharapkan.

2.5.1 Pembelajaran Terawasi *(Supervised Learning)*

Pada metoda *Supervised Learning,* setiap pengetahuan yang diberikan nilai acuan untuk pemetaan suatu *input*, akan menjadi suatu keluaran yang dikehendaki [3]. Proses pembelajaran akan dilakukan terus menerus selama kondisi *error* yang dikehendaki belum terjadi. Untuk setiap nilai *error* yang diperoleh di setiap tahap pembelajaran akan di kalkulasikan hingga data atau nilai target yang dikehendaki tercapai.

Contoh : andaikan kita memiliki jaringan syaraf yang akan digunakan untuk mengenali pasangan pola, misalkan pada operasi AND :

Input		Target
0	0	0
0	1	0
1	0	0
1	1	1

Gambar 2.5 Jaringan Syaraf Tiruan Untuk Penyelesaian Operasi *AND*

Pada proses pembelajaran, satu pola akan diberikan ke *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 dicocokan dengan pola *output* targetnya. Apabila terjadi perbedaan antara pola *output* hasil pembelajaran dengan pola target, maka di sini akan muncul *error*, mengindikasikan bahwa masih perlu dilakukan lebih banyak pembelajaran lagi.

2.5.1.1 Hebb Rule

Hebb rule adalah metode pembelajaran yang paling sederhana. Pada metode ini pembelajaran dilakukan dengan cara memperbaiki nilai bobot sedemikian rupa sehingga jika ada dua *neuron* yang terhubung, dan keduannya dalam kondisi "*on*" pada saat yang

sama, maka bobot antara keduanya dinaikan [3]. Apabila data direpresentasikan secara *bipolar*, maka perbaikan bobotnya adalah :

$$W_{ji} = \sum_{i}^{n} u_i u_j \quad (1)$$

$$W_{ji}^{(baru)} = W_{ji}^{(lama)} + U_j^{(baru)} + U_i^{(baru)} \quad (2)$$

Dengan :

W_{ji} : matrik bobot koneksi dari unit j ke unit i

U_j : elemen j dari vektor *U*

U_i : elemen i dari vektor *U*

n : jumlah pola yang dilatihkan

2.5.1.2 Backpropagation

Algoritma pelatihan *backpropagation* atau ada yang menterjemahkannya menjadi propagasi balik, pertama kali dirumuskan oleh Werbos dan dipopulerkan oleh Rumelhart dan McClelland untuk dipakai pada JST, dan selanjutnya algoritma ini biasa disingkat dengan *BP*. Algoritma ini termasuk metoda pelatihan *supervised* dan didesain untuk operasi pada jaringan *feed forward* multi lapis.

Algoritma ini juga banyak dipakai pada beberapa aplikasi pengaturan karena proses pelatihannya didasarkan pada hubungan yang sederhana, yaitu: jika keluaran memberikan hasil yang salah, maka bobot (*weight)* dikoreksi supaya nilainya dapat diperkecil dan respon jaringan selanjutnya diharapkan akan lebih mendekati keadaan dengan harga yang benar. *Backpropagation* juga berkemampuan untuk memperbaiki bobot – bobot pada lapisan tersembunyinya.

Secara garis besar algoritma ini disebut sebagai propagasi balik dikarenakan ketika jaringan diberikan pola masukan sebagai pola pelatihan maka pola tersebut menuju ke unit – unit pada lapisan tersembunyi (*hidden layer)* untuk diteruskan ke unit – unit lapisan keluaran [3]. Kemudian unit – unit lapisan keluaran memberikan tanggapan yang disebut dengan keluaran jaringan. Saat keluaran jaringan tidak sama dengan keluaran yang diharapkan maka keluaran akan menyebar mundur (*backward)* pada lapisan tersembunyi diteruskan ke unit pada lapisan masukan. Oleh karenanya maka mekanisme pelatihan tersebut dinamakan *backpropagation* atau propagasi balik.

2.5.2 Pembelajaran Tak Terawasi (*Unsupervised Learning)*

Pada metoda ini tidak menggunakan target pada keluarannya, karena sistem pada metoda ini bergantung sepenuhnya pada hasil komputasi di setiap tahapan pemrosesan untuk mendapatkan nilai target yang dikehendaki. Setiap proses pada metoda ini akan mengkalkulasikan setiap langkah pada nilai bobot yang dikehendaki. Selama proses pembelajaran, nilai bobot disusun dalam suati range tertentu tergantung pada nilai input yang diberikan. Tujuan pembelajaran ini adalah mengelompokan unit – unit yang hampir sama dalam suatu area tertentu.

2.5.2.1 Motode *Hopfield*

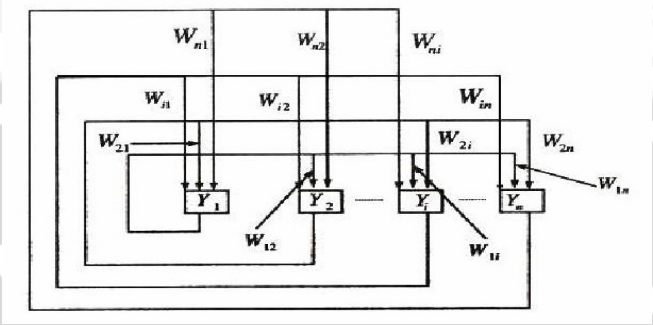
Metoda ini dikembangkan oleh John Hopfield pada tahun 1980. John Hopfield menggambarkan suatu *associative memory* yang dapat diterapkan dan kemudian mendemonstrasikan masalah optimasi dan klasifikasi yang dapat diselesaikan oleh jaringannya [2]. Jaringan *Hopfield* biner mempunyai suatu lapisan pengolah. Setiap unit pengolah mempunyai sebuah nilai aktifitas atau kondisi *(state)* yang bersifat biner [2]. Jaringan *Hopfield* bisa bekerja dengan kondisi 0 dan 1. Jaringan juga dapat bekerja jika digunakan nilai +1 dan -1, hanya saja diperlukan sedikit perubahan dalam persamaannya.

Unit-unit pengolah dalam jaringan *Hopfield* adalah terhubung penuh *(fully connected*) yaitu setiap unit terhubung ke tiap-tiap unit yang lain [1]. Hubungan-hubungan tersebut adalah hubungan langsung dan setiap pasang unit pengolah mempunyai hubungan dalam dua arah. Topologi hubungan ini mempunyai jaringan bersifat *recursive* karena keluaran dari setiap unit memberi masukan ke unit yang lain. Jaringan ini memiliki bobot-bobot yang simetris. Pada jaringan *Hopfield* setiap unit tidak memiliki hubungan dengan dirinya sendiri, secara matematik hal ini memenuhi $W_{ij} = W_{ji}$ (di mana i = baris dan j = kolom) untuk $i \neq j$ dan $W_{ij} = 0$ untuk $i = j$ [1].

$$W_{ji} = \begin{bmatrix} 0 & w_{12} & w_{13} & w_{14} & w_{15} & w_{116} \\ w_{21} & 0 & w_{23} & w_{24} & w_{25} & w_{26} \\ w_{31} & w_{32} & 0 & w_{34} & w_{35} & w_{36} \\ w_{41} & w_{42} & w_{43} & 0 & w_{44} & w_{45} \\ w_{51} & w_{52} & w_{53} & w_{54} & 0 & w_{55} \\ w_{61} & w_{62} & w_{63} & w_{64} & w_{65} & 0 \end{bmatrix}$$

Gambar 2.6 Matrik Bobot Jaringan *Hopfield* Sebagai Vektor W

Perhatikan bahwa bobot –bobot yang terletak pada diagonal utamanya adalah nol (0) yang menunjukan bahwa *neuron-neuron* pada jaringan *Hopfield* tidak memiliki hubungan dengan dirinya sendiri ($W_{ij} = 0$; i=j). Sementara itu kesimetrisan vektor bobot berarti berlakunya $W_{ij} = W_{ji}$ di mana $i \neq j$, sehingga $w_{12} = w_{21}$, $w_{13} = w_{31}$, $w_{23} = w_{32}$, ..., dan seterusnya.



Gambar 2.7 Arsitektur Jaringan Hopfield

Dari gambar 2.7 di atas, dapat dijelaskan bahwa *output* setiap simpul diumpan balikan ke *input* dari simpul – simpul lainnya melalui bobot koneksi W_{ij} yang tetap. Nilai W_{ij} mula – mula diinisialisasi menggunakan algoritma *Hopfield* untuk sebuah pola yang masuk. Pola yang tidak dikenal dimasukan ke dalam jaringan pada waktu kondisi nol. Kemudian jaringan akan beriterasi sampai output yang dihasilkan konvergen. Konvergen adalah pola *output* yang tidak berubah sampai iterasi selesai. Pola yang dinyatakan oleh simpul – simpul *output* setelah jaringan konvergen adalah *output* jaringan syaraf tiruan.

2.5.2.1.1 Algoritma Pembelajaran JST Hopfield

Ada beberapa versi dari algoritma *Hopfield* . Untuk deskripsi dan algoritma *Hopfield* menggunakan input vektor yang berupa angkatan biner. Untuk menyimpan sekumpulan pola biner digunakan notasi :

$$U_i \ , i = (1,2,3,.....,n) \qquad (3)$$

Keterangan :

n : jumlah pola yang dilatihkan

U_i : elemen i dari vektor U

Sedangkan untuk menyimpan bobot matriks W={W_{ji}}, dengan notasi :

$$W_{ji} = \sum_{i=1}^{n} [2U_j - 1][2U_i - 1] \quad (4)$$

jika $i \neq j$ dan $W_{ji} = 0$ jika i=j

Dengan :

W_{ji} : matrik bobot koneksi dari unit j ke unit i

n : jumlah pola yang dilatihkan

U_i : elemen i dari vektor *U* yang hanya memiliki nilai 0 dan 1

Sedangkan versi dari deskripsi algoritma *Hopfield* lainnya menggunakan *input vektor* yang berupa angka bipolar. Untuk menyimpan sekumpulan angka *bipolar* digunakan notasi :

$$U_i \ , i = (1,2,3,.....,n) \quad (5)$$

Keterangan :

n : jumlah pola yang dilatihkan

U_i : elemen i dari vektor *U*

Sedangkan untuk menyimpan bobot matriks $W = \{W_{ji}\}$, dengan notasi :

$$W_{ji} = \sum_{s=1}^{n} U_{sj} U_{si} \quad (6)$$

jika $j \neq i$ dan $W_{ij} = 0$ jika j = i

Keterangan :

W_{ji} : matrik bobot koneksi dari unit j ke unit i

n : jumlah pola yang dilatihkan

U_{si} : elemen i dari vektor *U* yang memiliki nilai +1 dan -1

U_{sj} : elemen j dari vektor *U* yang memiliki nilai +1 dan -1

Adapun algoritma pembelajaran *Hopfield* [1] dari kedua penjelasan di atas dan apabila diterapkan pada pola angka yang berjumlah sepuluh buah yang berupa angka nol (0) sampai sembilan (9) adalah sebagai berikut :

1. Menciptakan pola vektor *U*, yang merupakan himpunan *n* simpul untuk pola yang ke – i

   $U_i \ , i = (1,2,3,.....,n)$ (7)

2. Menciptakan matriks bobot koneksi sebagai tempat pola contoh yaitu angka 0 sampai pola angka 9 yang disimpan dalam suatu konstanta.

   Siapkan U_i untuk i = 1..10

   Inisialisasi matriks *W* (orde n x n) untuk menyimpan pola.

$$W_{ji} = \sum_{s=1}^{n} U_{sj} U_{si} \qquad (8)$$

jika $j \neq i$ dan $W_{ij} = 0$ jika j = i

Keterangan :

W_{ji} : matrik bobot koneksi dari unit j ke unit i

n : jumlah pola yang dilatihkan

U_{si} : elemen ke i dari vektor *U* yang memiliki nilai +1 dan -1

U_{sj} : elemen ke j dari vektor *U* yang memiliki nilai +1 dan -1

3. Mengerjakan langkah 4-6 selama pola belum konvergen *input* ≠ *output*
4. Untuk tiap vektor U_i , $i = (1,2,3,.....,n)$, hitung nilai y_in_i dengan persamaan $y_in_i = \sum_{i=1}^{n} w_{ji} U_i$ (9)

   Keterangan :

   y_in_i : sinyal output untuk aktivasi

5. Hitung $Outvec_1 = f(y_in_i)$ di mana f adalah fungsi ambang (*threshold function)*

$$f(t) = \begin{cases} 1, jika\ y_in \geq \theta \\ -1, jika y_in < \theta \end{cases} \qquad (10)$$

   Di mana *threshold* θ biasanya sama dengan 0.

6. Update *neuron input* jaringan dengan komponen $Outvec_i$
7. Test konvergensi.

2.5.2.1.2 Fungsi Energi *Hopfield*

Tiap kondisi dari jaringan *Hopfield* mempunyai sekumpulan energi [2]. Nilai ini didefinisikan dengan

$$E = -\frac{1}{2} \sum_{j} \sum_{\substack{i \\ i \neq j}} W_{ji} u_j u_i \qquad (11)$$

Keterangan :

W_{ji} : matrik bobot koneksi dari unit j ke unit i

n : jumlah pola yang dilatihkan

U_i : elemen ke i dari vektor U yang memiliki nilai +1 dan -1

U_j : elemen ke j dari vektor U yang memiliki nilai +1 dan -1

Persamaan ini dihubungkan dengan "energi", meskipun ini tidak menggambarkan energi yang nyata seperti pada sistem fisika. Fungsi energi pada persamaan diatas adalah sebuah fungsi objektif yang diminimalkan oleh jaringan [3]. Pembaharuan dari jaringan *Hopfield* merupakan prosedur konvergen di mana energi dari keseluruhan jaringan akan menjadi semakin kecil, pada akhirnya jaringan akan berada pada kondisi stabil di mana pada kondisi ini energi berada pada nilai minimum. Nilai minimum ini mungkin lokal ataupun global [3].

Pada tiap saat unit pengolah diperbaharui, energi dari jaringan akan berkurang atau tetap. Sebagai hasilnya, prosedur pembaharuan ini akan selalu membuat energi dari jaringan untuk konvergen pada nilai minimum. Berikut uraian bahwa prosedur pembaharuan akan mengurangi energi atau tetap sama. Misalkan unit *j* adalah pengolah berikutnya yang akan diperbaharui. Bagian energi yang dipengaruhi oleh unit pengolah *j* adalah :

$$E_j = -\frac{1}{2} \sum_{\substack{i \\ i \neq j}} W_{ji} u_j u_i \tag{12}$$

di mana dapat ditata menjadi :

$$E_j = -\frac{1}{2} u_j \sum_{\substack{i \\ i \neq j}} W_{ji} u_i \tag{13}$$

ketika unit *j* diperbaharui :

$$\Delta E_j = \Delta u_j \left(-\frac{1}{2} \sum_{\substack{i \\ i \neq j}} W_{ji} u_i \right) \tag{14}$$

dengan :

$$\Delta E_j = E_j^{\,baru} - E_j^{\,lama} \tag{15}$$

$$\Delta u_j = \left(u_j^{\,baru} - u_j^{\,lama} \right) \tag{16}$$

di mana jika unit *j* tidak terdapat perubahan kondisi pada saat pembaharuan maka $\Delta u_j = 0$ berdasarkan persamaan maka $\Delta E_j = 0$.

Jika unit u_j berubah dari 1 ke -1, maka :

$$u_j^{lama} = 1$$
$$u_j^{baru} = -1$$
$$\Delta u_j = (-1) - 1$$
$$= -2 < 0$$

pada saat perubahan kondisi unit u_j dari 1 ke -1 $W_{ji} u_i < 0$, sehingga nilai energi : $\Delta E_j < 0$. Jika unit u_j berubah dari -1 ke 1, maka :

$$u_j^{lama} = -1$$
$$u_j^{baru} = 1$$
$$\Delta u_j = 1 - (-1)$$
$$= 2 > 0$$

pada saat perubahan kondisi unit u_j dari -1 ke 1 $W_{ji} u_i \geq 0$, sehingga nilai energi : $\Delta E_j \leq 0$.

Karena ΔE_j adalah hasil dari perkalian tiga bilangan negatif. Jadi, perubahan energi *E* selalu negatif atau nol, tidak peduli bagaimana kondisi pada unit *j.* Jaringan dijamnin konvergen di mana nilai *E* akan menjadi lebih rendah sampai jaringan mencapai kondisi yang tetap.

2.6 Fungsi Aktivasi

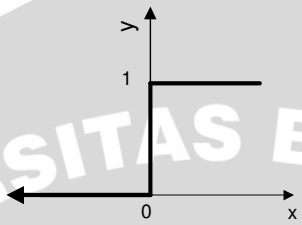
Fungsi aktivasi merupakan bagian terpenting dalam tahapan perhitungan keluaran dari suatu algoritma jaringan syaraf tiruan. Fungsi aktivasi dapat diibaratkan sebagai pendefinisian penguatan non linier dalam sisem analag (*continue).* Penguatan (*gain)* ini dihitung dengan mencari rasio perubahan pada fungsi *output neuron* pada lapisan keluaran. Fungsi aktivasi tidak hanya digunakan pada saat perhitungan nilai keluaran neuron saja, tetapi turunan pertamanya juga bisa untuk menghitung perubahan bobot dan bias pada proses belajar. Dalam penulisan ini digunakan fungsi aktivasi bipolar threshold. Beberapa fungsi aktivasi yang digunakan dalam jaringan syaraf tiruan adalah :

2.6.1 Fungsi Undak Biner (*Hard Limit*)

Jaringan dengan lapisan tunggal sering menggunakan fungsi undak (*step function)* untuk mengkonversikan input dari suatu variable yang bernilai kontinu ke suatu output biner (0 atau 1).

Fungsi undak biner (*hard limit)* dirumuskan sebagai berikut :

$$y = \begin{cases} 0, & jika\ x \le 0 \\ 1, & jika\ x > 0 \end{cases} \qquad (17)$$

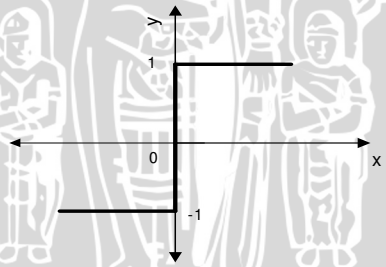


Gambar 2.8 Fungsi Aktivasi: Undak Biner *(Hard Limit)*

2.6.2 Fungsi *Bipolar* (*Symetric Hard Limit)*

Fungsi *Bipola*r sebenarnya hampir sama dengan fungsi undak biner, hanya saja output yang dihasilkan berupa 1, 0 atau -1. Fungsi *Symetric Hard Limit* dirumuskan sebagai :

$$y = \begin{cases} 1, & jika\ x > 0 \\ 0, & jika\ x = 0 \\ -1, & jika\ x < 0 \end{cases} \qquad (18)$$



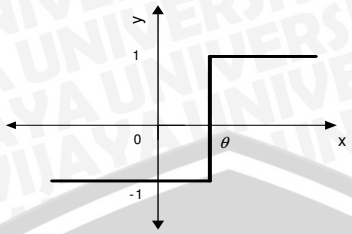
Gambar 2.9 Fungsi Aktivasi: *Bipolar* (*Symmetric Hard Limit*)

2.6.3 Fungsi *Bipolar* (*Threshold Functio)*

Fungsi bipolar sebenarnya hampir sama dengan fungsi undak biner dengan *threshold,* hanya saja output yang dihasilkan berupa 1, atau -1.

Fungsi bipolar (dengan nilai ambang θ) dirumuskan sebagai:

$$y = \begin{cases} 1, & jika\ x \geq \theta \\ -1, & jika\ x < \theta \end{cases} \quad (19)$$



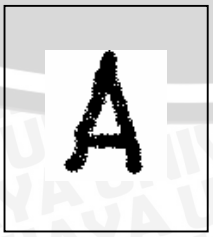
Gambar 2.10 Fungsi Aktivasi: Bipolar *(Threshold Function)*

2.7 Pengenalan Pola

Pengenalan pola mengelompokan data numerik dan simbolik (termasuk citra) secara otomatis oleh mesin (dalam hal ini komputer). Tujuan pengelompokan adalah untuk mengenali suatu objek di dalam citra. Manusia bisa mengenali objek yang dilihatnya karena otak manusia telah belajar mengklasifikasi objek-objek di alam sehingga mampu membedakan suatu objek lainnya. Kamampuan sistem visual manusia inilah yang dicoba ditiru oleh mesin. Komputer menerima masukan berupa citra objek yang akan di identifikasi, memproses citra tersebut, dan memberikan keluaran berupa deskripsi objek di dalam citra.

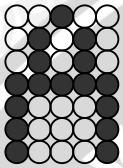
Gambar 2.11 Proses Pengenalan Pola

Contoh pengenalan pola misalnya citra pada gambar 2.12 adalah tulisan tangan yang digunakan sebagai data masukan untuk mengenali karakter ‘A’. dengan menggunakan suatu algoritma pengenalan pola, diharapkan komputer dapat menenali bahwa karakter tersebut adalah ‘A’.



Gambar 2.12 Citra Karakter ‘A’ Yang Digunakan Sebagai Masukan Untuk Pengenalan Huruf.

Salah satu contoh aplikasi pengenalan pola yang cukup kompleks adalah pengenalan karakter alfanumerik (['A'…'Z'] dan ['0'…'9]). Diasumsikan ingin dibuat sebuah desain program komputer yang bertugas mengenali karakter alfanumerik dengan jalan menerjemahkan sebuah matriks yang berisikan bilangan-bilangan biner yang menggambarkan citra *pixel* (*picture element*) dari sebuah pemetaan-bit *(bit-mappet pixel image)* dari dari sebuah karakter alfanumerik ke dalam bentuk kode ASCII 8 bit.



$=0010001010010101111110001100011000 1_2$

$=114AFC631_{16}$

$=65_{10}ASCII$

Gambar 2.13 Setiap Citra Karakter Dipetakan Ke Dalam Kode ASCII Yang Ditunjuknya

2.8 Citra *Digital*

Citra adalah gambar pada bidang dwimatra (dua dimensi). Citra merupakan fungsi menerus (*continue*) dari intensitas cahaya pada bidang dwimatra. Sumber cahaya menerangi objek, objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pemantulan cahaya ini ditangkap oleh alat-alat optik, misalnya pada mata manusia, kamera, pemindai (*scanner*), dan lain sebagainya, sehingga bayangan objek yang disebut citra tersebut terekam.

Citra sebagai keluaran dari suatu sistem perekaman data dapat bersifat:

1. Optik berupa foto,
2. Analog berupa sinyal video seperti gambar pada monitor televisi,
3. Digital yang dapat langsung disimpan pada suatu pita magnetik.

Citra *digital* dapat didefinisikan sebagai fungsi dua variabel, $f(x, y)$, di mana x dan y adalah koordinat spasial dan nilai $f(x, y)$ adalah intensitas cahaya citra pada koordinat tersebut. Teknologi dasar untuk menciptakan dan menampilkan warna merupakan kombinasi dari tiga warna dasar yaitu merah, hijau dan biru (*Red, Green,*

Blue – RGB). Karena intensitas yang dimaksud berasal dari sumber cahaya dan cahaya adalah suatu bentuk energi, maka berlakulah keadaan di mana fungsi intensitas terletak di antara:

$$0 < f(x, y) < \infty \qquad (20)$$

Keterangan :

f(x,y) merupakan intensitas cahaya pada titik *(x,y).*

Pada hakekatnya gambar yang dilihat oleh mata kita terdiri atas berkas-berkas cahaya yang dipantulkan oleh benda-benda sekitar kita. Sehingga bentuk dasar dari *f(x,y)* dapat dipandang sebagai karakterisasi dari dua komponen. Komponen pertama ialah pencahayaan (*illumination)* yaitu menyatakan jumlah sumber cahaya yang datang pada bidang yang diamati, dinotasikan dengan i(x,y), sedangkan komponen lainnya dipantulkan oleh objek pada bidang tersebut, dinotasikan dengan r(x,y). Kombinasi dari perkalian fungsi i(x,y) dan r(x,y) membentuk f(x,y).

$$f(x,y) = i(x,y) * r(x,y) \qquad (21)$$

dengan :

i(x,y) : *illumination* / pencahayaan

r(x,y) : *reflactane component* / komponen pemantulan

di mana :

$$0 < i(x, y) < \infty$$

$$0 < r(x, y) < 1 \qquad (22)$$

Persamaan-persamaan di atas menyatakan bahwa reflektansi dibatasi oleh nol (0) yang berarti terjadi penyerapan total oleh obyek tersebut dan satu (1) berarti cahaya yang dipantulkan total oleh obyek tersebut.

2.9 Pemrosesan Citra (*Image Processing*)

Secara umum dan sederhana, citra dapat didefinisikan sebagai representasi visual dari suatu objek. Lebih jauh citra dapat juga diartikan sebagai gambaran yang representatif mengenai suatu objek sedemikian sehingga citra tersebut dapat memberikan kesan yang mendalam mengenai objek yang dimaksud. Jika ingin mendefinisikannya lebih bebas lagi, citra dapat didefinisikan sebagai bentuk visual yang dapat diterima secara baik oleh indera penglihatan, apapun bentuknya. Komputer memiliki cara pandang tersendiri terhadap suatu citra. Berbeda dengan citra konvensional yang misalnya melalui proses fotografis seperti pada foto dapat dihasilkan

suatu citra nyata yang langsung dapat dinikmati oleh indera penglihatan, citra pada computer harus melalui beberapa tahapan yang cukup rumit. Tahapan tersebut dapat digambarkan sebagai suatu rangkaian proses dari proses akuisi data, manipulasi data, visualisasi data, serta proses penyimpanan data.

Seringkali dokumen hasil *scan* memiliki banyak *noise.* Hal ini disebabkan oleh beberapa hal, antara lain : dokumen yang akan di *scan* memiliki kualitas yang jelek, kaca kotor atau dokumen yang akan di *scan* berada dalam posisi miring, *noise* pada dokumen *scan* akan mengurangi keakuratan sistem dalam melakukan pengenalan karakter, oleh karena itu diperlukan suatu proses untuk menghilangkan *noise* yang ada. Kinerja proses *image processing* ini akan mempengaruhi keakuratan keseluruhan sistem pengenalan karakter.

2.9.1 Representasi Citra *Digital*

Cita *digital* adalah sebuah citra *f(x,y)* yang telah di-diskretasi ke dalam koordinat spasial dan tingkat keabuan. Cita digital dinyatakan sebagai sebuah matrik *n x n* yang terdiri atas baris dan kolom untuk menyatakan sebuah titik pada citra dan elemen nilai matrik yang berupa nilai diskrit menyatakan tingkat keabuan pada titik tersebut. Citra digital tiap elemen dikenal sebagai elemen gambar (*picture element)* atau *pixel*.

$$f(x,y)=\begin{vmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ . & . & . & . \\ . & . & . & . \\ f(N-1,0) & f(N-1,1) & . & f(N-1,N-1) \end{vmatrix}$$

Gambar 2.14 Representasi Citra *Digital* Dalam Bentuk Matrik

Untuk mendapatkan suatu citra *digital* diperlukan suatu konversi sehingga dapat diproses oleh komputer. Proses konversi tersebut dengan membuat kisi-kisi arah *horizontal* dan *vertical* sehingga diperoleh gambar dalam bentuk array dua dimensi. Proses tersebut disebut digitasi atau *sampling.*

Semakin tinggi resolusi berarti semakin kecil ukuran *pixel*-nya, berarti semakin halus gambar yang diperoleh karena informasi yang hilang akibat pengelompokan tingkat keabuan pada proses pembuatan kisi-kisi semakin kecil, tetapi membutuhkan penyimpanan bit yang makin besar pula.

2.9.2 Representasi Warna (*Color Representation*)

Karena persepsi manusia akan warna berdasarkan respon dari tiga *cones* yang berada pada bola mata manusia, maka representasi warna pun didasarkan pada hal tersebut, yang biasa disebut sebagai *tristimulus value.* Salah satu model representasi warna berdasarkan *tristimulus value* adalah representasi warna RGB yang nantinya dapat dijadikan representasi *grayscale* untuk memudahkan pemrosesan citra.

2.9.2.1 Representasi Warna *RGB*

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.

2.9.2.2 Representasi *Grayscale*

Dengan menggunakan representasi warna *RGB* gambar yang berwarna dapat diubah menjadi gambar yang terdiri dari warna putih dan gradiasi warna hitam yang biasa disebut gambar *grayscale.* Setiap pixel dari gambar 24-bit mempunyai 8-bit *Red*, 8-bit *Green* dan 8-bit *Blue*. Suatu gambar warna dapat dikonversi ke *grayscale* dengan menghitung nilai "Y" untuk tiap warna pixel [8].

$$Y = 0{,}299R + 0{,}587G + 0{,}114B \quad (23)$$

Nilai *"Y"* merupakan komponen *grayscale* dalam sistem *YIQ* yang digunakan dalam sistem televisi *NTSC*. Koefisiennya menyatakan sensifitas terang-gelapnya (*brightness)* mata manusia terhadap warna utama. Metode lainnya dengan menghitung nilai rata-rata dari RGB itu sendri.

$$Y = (R + G + B) / 3 \quad (24)$$

2.9.3 Thresholding

Thresholding adalah proses untuk memisahkan *foreground* dengan *background* dari suatu *image.* Proses *Thresholding* dilakukan dengan cara melihat perbedaan intensitas warna dari suatu *image*. *Input* untuk proses *thresholding* adalah *grayscale image* dan *colour image. Output* dari proses ini adalah *binary image* di mana *pixel*

hitam mewakili *foreground* dan *pixel* putih mewakili *background,* atau sebaliknya. *Binary Image* adalah suatu *image* yang mana *pixel*nya hanya memiliki dua nilai intensitas. Nilai intensitas yang sering digunakan yaitu 0 untuk *pixel* hitam, 1 atau 255 untuk *pixel* putih ataupun sebaliknya.

Proses *thresholding* menggunakan sebuah parameter yang disebut dengan *intensity threshold.* Setiap *pixel* dari *image* akan dibandingkan dengan *intensity threshold.* Apabila intensitas *pixel* lebih tinggi dari *intensity threshold* maka *pixel* diset warna putih, sebaliknya apabila intensitas *pixel* lebih rendah maka *pixel* diset warna hitam, bisa juga sebaliknya.

2.9.4 Segmentasi Citra

Segmentasi citra merupakan suatu proses pengelompokkan citra menjadi beberapa *region* berdasarkan kriteria tertentu, dengan kata lain, merupakan suatu proses pembagian citra ke dalam wilayah *(region)* yang mempunyai kesamaan fitur antara lain tingkat keabuan *(grayscale),* teksture *(texture)*, warna *(color),* gerakan *(motion).*
Berdasarkan pengertiannya, segmentasi memiliki tujuan menemukan karakteristik khusus yang dimiliki suatu citra. Oleh karena itulah, segmentasi sangat diperlukan pada proses pengenalan pola. Semakin baik kualitas segmentasi maka semakin baik pula kualitas pengenalan polanya.

Secara umum ada beberapa pendekatan yang banyak digunakan dalam proses segmentasi antara lain :

1. Teknik *threshold*, yaitu pengelompokan citra sesuai dengan distribusi properti *pixel* penyusun citra.
2. Teknik *region-based*, yaitu pengelompokkan citra ke dalam *region-region* tertentu secara langsung berdasar persamaan karakteristik suatu area citranya.
3. *Edge-based methods*, yaitu pengelompokkan citra ke dalam wilayah berbeda yang terpisahkan karena adanya perbedaan perubahan warna tepi dan warna dasar citra yang mendadak.

Pendekatan pertama dan kedua merupakan contoh kategori pemisahan *image* berdasarkan kemiripan area citra, sedangkan pendekatan ketiga merupakan salah satu contoh pemisahan daerah berdasarkan perubahan intensitas yang cepat terhadap suatu daerah. Performansi citra segmentasi dapat dilakukan secara kuantitatif dan kualitatif.

Penilaian secara kuantitatif dilihat dari segi performansi yang dihasilkan melalui perhitungan yang dilakukan dengan menghitung nilai koefisien variansi dari hasil segmentasi. Sedangkan, untuk pernilaian kualitatif ditinjau dari segi pengamatan mata manusia *(Mean Opinion Score/*MOS).