Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

Team project ©2017
Dony Pratidana S. Hum | Bima Agus Setyawan S. IIP

BAB II

LANDASAN TEORI

2.1 Pengenalan Tulisan Tangan Offline

Pengenalan tulisan tangan mengarah pada identifikasi karakter (huruf ataupun angka) yang tertulis (Kala. dkk, 2010). Berdasarkan bagaimana *input* didapatkan, pengenalan tulisan tangan dibagi menjadi dua, yaitu pengenalan tulisan tangan *offline* dan pengenalan tulisan tangan *online* (Kumar & Bhatia, 2013). Pengenalan tulisan tangan *offline* merupakan pengenalan tulisan yang didasarkan pada gambar dari sebuah teks tertulis (Chakravarthy. dkk, 2011). Pengenalan tulisan tangan *offline* ini melibatkan konversi otomatis teks dalam gambar ke dalam kode-kode huruf yang digunakan dalam komputer dan aplikasi pengolahan teks (Chakravarthy. dkk, 2011). Data yang diperoleh dianggap sebagai representasi statis tulisan tangan (Chakravarthy. dkk, 2011). Tahapan dalam pengenalan tulisan tangan ini adalah (Kumar & Bhatia, 2013):

1. *Image Scanning*

   Pada tahap ini, dokumen tertulis akan di *scan* dengan mengunakan *scanner* dan kemudian gambar tersebut akan menjadi *input* ke dalam aplikasi, khususnya tahap *image preprocessing* (Rico, 2008). Gambar ini harus memiliki format tertentu seperti JPEG, BMP, dll (Rico, 2008).

2. *Image Preprocessing*

   Keluaran dari tahap sebelumnya merupakan masukkan bagi tahap *image preprocessing* (Kumar & Bhatia, 2013). Data mentah dari tulisan tangan akan

diberikan sejumlah tahapan *preprocessing* untuk membuat data tersebut dapat berguna untuk tahapan selanjutnya (Kumar & Bhatia, 2013). Fase *image preprocessing* bertujuan untuk mengekstraksi bagian-bagian tekstual, bagian yang terdapat tulisan atau karakter, yang relevan dan mempersiapkannya untuk tahap segmentasi dan pengenalan (Kumar & Bhatia, 2013). *Image preprocessing* pada sebuah gambar diartikan sebagai penerapan beberapa prosedur seperti *thresholding*, penghalusan, penyaringan, pengubahan skala, dan normalisasi sehingga algoritma untuk klasifikasi akhir dapat lebih sederhana dan akurat (Suliman. dkk, 2010).

3. Segmentasi

   Sebuah gambar ditampilkan dalam bentuk deretan karakter-karakter yang diuraikan ke dalam bagian-bagian gambar dari individual karakter pada tahapan segmentasi (Kumar & Bhatia, 2013).

4. Ekstraksi ciri

   Ekstraksi ciri berkonsentrasi pada menemukan bentuk pada gambar (Nixon & Aguado, 2008). Setiap karakter memiliki ciri yang berbeda-beda, yang memainkan peranan penting dalam pengenalan pola (Kumar & Bhatia, 2013). Tujuan utama dari ekstraksi ciri adalah untuk menyediakan informasi yang paling relevan dari data asli dan merepresentasikan informasi tersebut pada ruang dimensional yang lebih rendah (Kumar & Bhatia, 2013). Ekstraksi ciri sebenarnya sama seperti *neural network*, tetapi *programmer* harus secara eksplisit dan manual menentukan *properties* yang penting, seperti (Kumar & Bhatia, 2013) :

- Presentase *pixel* secara horizontal
- Presentase *pixel* secara vertikal
- Rata-rata jarak dari titik pusat gambar
- Aspek perbandingan

Ekstraksi ciri dapat menggunakan karakter per karakter, bagian kata, ataupun kata (Lagudu & Sarma, 2013).

5. *Neural Network Classification*

Banyak sekali algoritma yang dapat digunakan pada tahap ini, salah satunya menggunakan jaringan syaraf (*Neural network).* Jaringan syaraf belajar dari gambar pelatihan awal (Chakravarthy. dkk, 2011). Jaringan akan dilatih kemudian dapat membuat identifikasi dari karakter (Chakravarthy. dkk, 2011). Setiap jaringan syaraf, secara unik mempelajari *properties* dan perbedaan dari gambar–gambar tersebut (Chakravarthy. dkk, 2011). Kemudian, akan mencari *properties* yang sama pada gambar untuk diidentifikasi (Chakravarthy. dkk, 2011). Jaringan syaraf sangat cepat untuk diatur (Chakravarthy. dkk, 2011). *Backpropagation* merupakan algoritma yang digunakan untuk melatih jaringan syaraf. Jaringan dapat digunakan untuk mempelajari karakter pada pola dan melatih jaringan disimulasikan untuk pengenalan karakter yang ditampilkan dalam bentuk gambar (Kumar & Bhatia, 2013).

2.1.1 Digital Image Processing

Sebuah citra dibuat atau diproduksi oleh berbagai macam perangkat fisik, seperti kamera video, *scanner*, perangkat X-ray, mikroskop elektron, dan radar (Shih, 2009). Kesemuanya dapat digunakan untuk banyak tujuan, yaitu hiburan, bisnis dan industri, militer, keamanan, dan analisis ilmiah (Shih, 2009).

Digital image processing dimulai dengan satu citra dan menghasilkan sebuah citra yang telah dimodifikasi (Shih, 2009). *Digital image processing* adalah sebuah teknologi yang menerapkan sejumlah algoritma komputer untuk memproses citra digital (Zhou. dkk, 2010). Hasil dari proses ini dapat berupa citra ataupun sejumlah karakteristik atau ciri dari citra aslinya (Zhou. dkk, 2010).

Sebuah citra dibagi menjadi bagian-bagian kecil yang disebut elemen citra atau *pixel* (Shih, 2009). Gambar tersebut direpresentasikan dengan sebuah *array* persegi yang berisikan nilai *integer* (Shih, 2009). Angka pada setiap *pixel* mewakili terang atau gelapnya citra pada titik tersebut (atau sering disebut intensitas) (Shih, 2009).

Gambar 2.1 Citra Digital dan Representasi Numerik (Shih, 2009)

Pada *image processing*, operasi dapat dikategorikan menjadi empat macam, yaitu :

1. Operasi *Pixel*

   Keluaran pada *pixel* hanya bergantung pada *input* pada *pixel* tersebut, yang independen terhadap semua *pixel* pada citra (Shih, 2009). Sebagai contoh, thresholding, sebuah proses yang membuat *pixel input* di atas suatu tingkat *thresholding* tertentu berwarna putih, dan yang lainnya berwarna hitam, merupakan contoh sederhana dari sebuah operasi *pixel* (Shih, 2009). Contoh lainnya adalah penambahan/pengurangan kecerahan, transformasi log, pembalikkan citra, dan peluasan kontras (Shih, 2009).

2. Operasi Lokal

   Keluaran pada *pixel* tidak hanya bergantung pada *pixel* di lokasi yang sama pada citra *input*, tetapi juga pada *pixel input* yang berada di sekitarnya (disebut juga *neighborhood*) (Shih, 2009). Pada operasi ini, citra *output* harus dibuat terpisah dari citra *input*nya, sehingga *pixel* sekitarnya pada perhitungan semua berasal dari citra *input* (Shih, 2009). Sebagai contoh adalah penyaringan morfologi, pendeteksian sisi/garis, penyaringan halus, dan penyaringan tajam, dsb (Shih, 2009).

3. Operasi Geometri

   Keluaran pada *pixel* hanya bergantung pada tingkatan *input* pada *pixel* lainnya yang didefinisikan oleh transformasi geometri (Shih, 2009). Operasi geometri berbeda dari operasi global dimana *input* hanya berasal dari beberapa *pixel* yang dikalkulasi menggunakan transformasi geometri (Shih, 2009).

4. Operasi Global

Keluaran pada *pixel* bergantung pada semua *pixel* dalam citra tersebut (Shih, 2009). Sebuah operasi global dirancang untuk merefleksikan informasi statistik yang dikalkulasikan dari semua *pixel* pada citra, tetapi bukan dari sebuah subset *pixel* lokal (Shih, 2009). Sebagai contoh, sebuah transformasi jarak pada citra yang diberikan untuk setiap *pixel* jarak minimal ke *pixel* latar belakang, merupakan operasi global (Shih, 2009). Contoh lainnya Termasuk algoritma *hough transform*, dan *image warping* (Shih, 2009). *Image warping* merupakan sebuah proses transformasi atau perubahan dimana akan memetakan seluruh posisi pada suatu permukaan datar citra ke posisi permukaan datar lainnya.

2.1.2 Image Binarization

Image binarization merupakan sebuah tahap awal yang utama untuk analisis dokumen berbentuk citra (Feng & Tan, 2004). Proses ini akan mengkonversikan nilai *gray* pada citra ke dalam dua tingkatan representasi untuk area teks dan non-teks (Feng & Tan, 2004). Teknik – teknik yang dilakukan pada proses *binarization* dapat dibedakan menjadi dua kategori, yaitu berbasis *global thresholding* dan berbasis *local thresholding* (Feng & Tan, 2004).

Global thresholding hanya menggunakan sebuah nilai *threshold*, yang biasanya dipilih berdasarkan beberapa statistik dan heuristik dari beberapa *attribute* global sebuah citra, untuk mengklasifikasikan *pixel* pada citra ke dalam teks (latar depan) dan non-teks (latar belakang) (Otsu, 1979). Tetapi, kelemahan

dari *global thresholding* adalah tidak dapat beradaptasi dengan baik pada gangguan yang acak dan iluminasi, dan karena itu menampilkan kualitas yang rendah dari gambar dokumen (Feng & Tan, 2004).

Pada *local thresholding*, nilai dari *threshold* bervariasi berdasarkan konten lokal dari citra dokumen tersebut (Feng & Tan, 2004). Dibandingkan dengan *global thresholding*, *local thresholding* biasanya menampilkan kualitas citra yang lebih baik, khususnya pada pengklasifikasian *pixel* di dekat teks dan garis batas obyek (Feng & Tan, 2004).

2.1.3 Ekstraksi ciri

Ekstraksi ciri merupakan sebuah masalah yang penting dan menantang pada *remote sensing* dan *machine vision* (Ljumic, 2007). Ekstraksi ciri memainkan peran yang utama untuk semua tugas analisis gambar, terutama pada pemisahan pengenalan pola (Ljumic, 2007). Selama beberapa tahun, banyak teknik dan algoritma yang dipakai untuk melakukan ekstraksi ciri, tetapi masih banyak yang masih memiliki kekurangan untuk diaplikasikan dalam *remote sensing* pada citra (Ljumic, 2007).

Tugas ekstraksi ciri adalah untuk menekankan atau mendapatkan informasi pada citra pada tingkatan tertentu, dimana algoritma bekerja (Deserno, 2011). Menurut T.M. Deserno, terdapat beberapa tingkatan ekstraksi ciri seperti berikut ini.

1. Tingkatan Data

   Feature berbasis data bergantung pada informasi gabungan keseluruhan *pixel*. Oleh karena itu, semua manipulasi transformasi seluruh matriks dari citra dapat diartikan sebagai ekstraksi ciri.

2. Tingkatan *Pixel*

   Feature berbasis *pixel* bergantung pada nilai dari *pixel* masing-masing. Seluruh operasi pada titik *pixel*, akan dianggap sebagai ekstraksi ciri pada tingkatan *pixel*.

3. Tingkatan Tepi

   Feature berbasis tepi didefinisikan sebagai kontras lokal, sebagai contoh adalah perbedaan nilai yang besar antara *pixel* yang berdekatan.

4. Tingkatan Tekstur

   Feature berbasis tekstur telah digunakan dalam medis sejak lama. Pada buku patologi dapat dilihat banyak metafora untuk mendeskripsikan tekstur, seperti bentuk batuan *mucosal*, stratifikasi *subintima* yang berbentuk seperti bawang atau jaringan paru-paru yang berbentuk seperti sarang lebah.

2.2 Algoritma Generalized Hough Transform

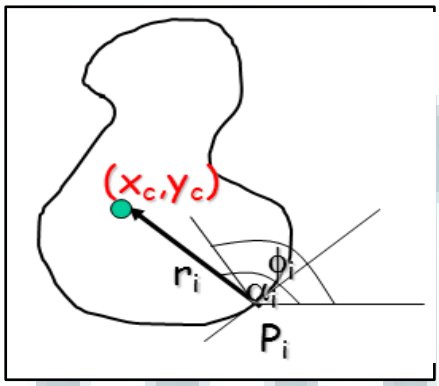
Hough Transform merupakan salah satu algoritma yang dapat digunakan untuk ekstraksi ciri. Ide dasar dari *Hough Transform* adalah untuk memetakan titik tepi citra ke parameter ruang, yang mewakili semua *instance* dari *feature* yang terlihat pada citra (Touj. dkk, 2005). *Hough Transform* dapat mengekstrak bentuk – bentuk yang sederhana seperti garis dan bentuk dua dimensi lainnya,

tetapi tidak untuk bentuk – bentuk yang lebih kompleks dan berubah-ubah. Oleh karena itu, digunakanlah algoritma *Generalized Hough Transform*.

Generalized Hough Transform yang sering dipakai sekarang merupakan penemuan dari DH. Ballard pada 1981. *Generalized Hough Transform* merupakan perkembangan dari *Hough Transform* untuk mendeteksi obyek yang berubah-ubah pada citra. Semua bentuk tidak dapat secara mudah dinyatakan dengan sekelompok kecil parameter (Touj. dkk, 2005). Solusi yang diusulkan kemudian untuk membuat tabel untuk menyimpan semua *pixel* sisi dari target bentuk kita (Touj. dkk, 2005).

Implementasi *Generalized Hough Transform* menggunakan informasi sisi untuk mendefinisikan pemetaan dari orientasi titik tepi ke titik referensi dari bentuk tersebut. Dalam menerapkan algoritma *Generalized Hough Transform* terdapat beberapa tahapan, yaitu (Barnes, 2011) :

1. Menetapkan titik acuan (x_c, y_c) yang dapat dihitung menggunakan rumus berikut dengan contoh obyek citra seperti pada Gambar 2.2 di bawah ini.



Gambar 2.2 Contoh Obyek Citra untuk Implementasi GHT

(Barnes, 2011)

$$x_c = x_i + r_i\cos(\alpha_i)$$
$$y_c = y_i + r_i\sin(\alpha_i)$$

Gambar 2.3 Rumus Titik Acuan (Barnes, 2011)

2. Menggambar garis dari titik acuan ke titik pada tepi obyek (P_i) dan menghitung panjang dari r_i dan sudut α_i..
3. Menuliskan koordinat dari titik acuan (x_c, y_c) sebagai fungsi dari r_i dan α_i.
4. Menyimpan orientasi gradien (ϕ_i) pada P_i.
5. Membuat sebuah tabel dengan data-data di atas, dan sebagai indeks ϕ_i.

Gambar 2.4 Contoh Obyek untuk Pembuatan Tabel

(Barnes, 2011)

Gambar 2.5 Contoh R-Tabel pada *Generalized Hough Transform*
(Barnes, 2011)

6. Menemukan versi kurva obyek setelah diputar, diperbesar/perkecil, dan ditranslasikan dengan membuat *accumulator array* dari titik acuan, diperbesar/perkecil dengan faktor S dan diputar dengan sudut θ. Untuk setiap tepi (x,y) pada citra, menghitung $\phi(x,y)$.

   Kemudian untuk setiap (r,α) yang sesuai dengan $\phi(x,y)$ lakukan :

   - Untuk setiap S dan θ :

$$x_c = x_i + r(f)\ S \cos[a(f) + q]$$
$$y_c = y_i + r(f)\ S \sin[a(f) + q]$$
$$A(x_c,y_c,S,q) = A(xc,yc,S,q) + 1$$

Gambar 2.6 Rumus Menghitung *Accumulator Array*
(Barnes, 2011)

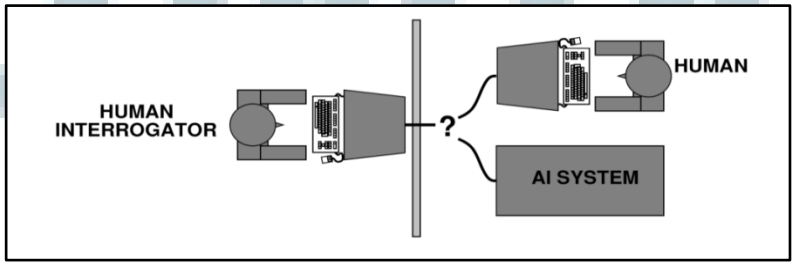
   Dan mencari maxima dari A.

2.3 Kecerdasan Buatan

Kecerdasan buatan (*Artificial Intelligence*) adalah sebuah cabang dari ilmu komputer yang berkonsentrasi dengan otomatisasi tingkah laku yang cerdas (Luger & Stubblefield, 1993). Terdapat empat kategori dari kecerdasan buatan, yaitu sebagai berikut (Russel & Norvig, 1995).

1. Sistem yang bertindak seperti manusia

   Kategori ini memberikan penekanan bagaimana mengatakan bahwa mesin cerdas, bukan pada bagaimana membuatnya cerdas. Sebagai contoh adalah *Turing Test*. *Turing test* merupakan penemuan dari Alan Turing pada 1950, yang dirancang untuk menyediakan sebuah definisi operasional yang memuaskan tentang kecerdasan (Russel & Norvig, 1995). Turing mendefinisikan bahwa tingkah laku yang cerdas sebagai kemampuan untuk menerima performa manusia dalam segala tugas kognitif, untuk menipu *interrogator* (Russel & Norvig, 1995). *Turing Test* terdiri dari *interrogator* (manusia), manusia, dan sistem kecerdasan buatan. *Interrogator* akan dipisahkan dari manusia dan sistem. *Interrogator* harus menentukan mana yang manusia dan mana yang sistem dengan memberikan pertanyaan. Jika mesin dapat mengelabui *interrogator*, maka mesin tersebut lulus dari *Turing Test*.



Gambar 2.7 *Turing Test* (Russel & Norvig, 1995)

2. Sistem yang berpikir seperti manusia

   Kategori ini menekankan pada bagaimana cara manusia berpikir dan bagaimana pikiran kita bekerja. Ketika sebuah *input/output* dan waktu dari tingkah laku program sesuai dengan tingkah laku manusia, itu adalah bukti bahwa beberapa mekanisme pada program dapat beroperasi pada manusia (Russel & Norvig, 1995).

3. Sistem yang berpikir secara rasional

   Mencoba untuk mengerti bagaimana manusia berpikir merupakan salah satu subyek dari kecerdasan buatan. Tetapi, pada kategori ini menetapkan pada bagaimana seharusnya manusia berpikir.

4. Sistem yang bertindak secara rasional

   Bertindak secara rasional merupakan bertindak untuk meraih suatu tujuan, diberikan suatu keyakinan (Russel & Norvig, 1995). Contohnya adalah seorang *agent*. *Agent* adalah sesuatu yang menerima dan bertindak, sedangkan *rational agent* merupakan *agent* yang bertindak secara rasional. Salah satu cara untuk bertindak rasional adalah dengan memiliki alasan yang memiliki logika pada kesimpulan bahwa suatu tindakan akan mencapai tujuan, dan bertindak sesuai kesimpulan tersebut (Russel & Norvig, 1995).

2.4 Jaringan syaraf tiruan

Jaringan syaraf tiruan telah ada pada 1950an, tetapi pada tahun 80an jaringan syaraf tiruan sudah tidak banyak lagi dipakai untuk pembuatan aplikasi (Lagudu & Sarma, 2013). Sekarang ini, jaringan syaraf tiruan sudah diaplikasikan ke dalam persoalan nyata (Buffa, 1997).

Jaringan syaraf tiruan merupakan sebuah model seperti jaringan syaraf pada manusia, yang dapat digunakan untuk pengenalan pola dan pembelajaran mesin (Lejap, 2008). Jaringan syaraf tiruan adalah sekumpulan sel neuron yang saling terhubung secara sederhana (Lagudu & Sarma, 2013). Sel-sel neuron yang ada diatur sehingga setiap sel dapat mengirimkan masukkan dari satu sel ke sel lainnya (Lagudu & Sarma, 2013). Keluaran yang dihasilkan dari jaringan ini didapatkan dari masukkan yang disebarkan keseluruh jaringan dari sel-sel yang aktif (Lagudu & Sarma, 2013). Jaringan syaraf tiruan ditentukan oleh tiga hal, yaitu (Siang, 2009):

1. Pola hubungan antar neuron (disebut arsitektur jaringan).
2. Metode untuk menentukan bobot penghubung (disebut dengan metode *training/leraning/*pelatihan).
3. Fungsi aktivasi.

2.4.1 Neuron (*Artificial Neuron*)

Neuron merupakan sebuah sel sama seperti halnya dengan neuron pada syaraf manusia. Neuron memproduksi sebuah keluaran (Lagudu & Sarma, 2013). Biasanya neuron yang menghasilkan satu keluaran bersifat linier karena tidak ada

fungsi aktivasi yang akan digunakan (Lagudu & Sarma, 2013). Pada jaringan syaraf tiruan, sebuah neuron terdiri dari tiga bagian yaitu fungsi penjumlahan atau fungsi masukkan (*input/summing function*), fungsi aktivasi (*activation function*), dan keluaran (*output*) (Siang, 2009).

Gambar 2.8 Neuron (Russel & Norvig, 1995)

Pada Gambar 2.8 di atas, dapat terlihat bahwa masukkan akan dikirim ke neuron dengan bobot tertentu. Masukkan ini akan diproses oleh suatu fungsi yang akan menjumlahkan perkalian nilai-nilai bobot dengan masukkan. Hasil penjumlahan kemudian akan dihitung dalam fungsi aktivasi dan kemudian akan dibandingkan hasilnya dengan suatu batas ambang (*threshold*). Jika hasil keluaran melewati suatu batas ambang tertentu maka neuron tersebut akan diaktifkan, jika tidak neuron tersebut akan mengirimkan keluaran melalui bobot-bobot keluaran ke semua neuron yang berhubungan dengannya melalui *links* atau penghubung (Agustin, 2012).

2.4.2 Layar (*Layer*)

Pada JST, layar dapat berjumlah 2 (layar masukkan dan layar keluaran) atau lebih. Setiap layar dapat terdiri dari beberapa neuron, sesuai yang diinginkan. Layar-layar penyusun jaringan syaraf dapat dibagi menjadi tiga, yaitu sebagai berikut.

1. Layar Masukkan (*input layer*)

   Unit-unit di dalam layar masukkan disebut unit-unit masukkan. Unit-unit tersebut menerima pola masukkan data dari luar yang menggambarkan suatu permasalahan yang akan dipecahkan (Agustin, 2012).

2. Layar Tersembunyi (*hidden layer*)

   Unit-unit di dalam layar tersembunyi disebut unit-unit tersembunyi dimana keluarannya tidak dapat secara langsung diamati (Agustin, 2012). Layar tersembunyi merupakan layar dari elemen pemrosesan (neuron) yang tersebunyi dari koneksi langsung ke titik luar dari jaringan Syaraf (Priddy & Keller, 2005). Semua koneksi ke dan dari layar tersembunyi bersifat internal pada jaringan Syaraf (Priddy & Keller, 2005).

3. Layar Keluaran (*output layer*)

   Unit-unit di dalam layar keluaran disebut unit-unit keluaran. Keluaran dari lapisan ini merupakan solusi jaringan syaraf terhadap suatu permasalahan (Agustin ,2012).

2.4.3 Arsitektur Jaringan Syaraf Tiruan

Beberapa arsitektur jaringan yang sering digunakan dalam berbagai aplikasi antara lain (Siang, 2009).

1. Jaringan Layar Tunggal (*single layer network*)

   Dalam jaringan ini, sekumpulan neuron masukkan dihubungkan langsung dengan sekumpulan neuron keluaran (Siang, 2009).

   Gambar 2.9 Jaringan Layar Tunggal (Beckenkamp, 2002)

   Pada Gambar 2.9, terlihat bahwa satu neuron pada layar masukkan akan dihubungkan langsung ke semua neuron pada lapisan keluaran tanpa dihubungkan dengan neuron lain. Contoh dari jaringan layar tunggal adalah ADALINE, *Hopfield*, *Perceptron*, LVQ (*Learning Vector Quantization*), dll (Siang, 2009).

2. Jaringan Layar Jamak (*multi layer network*)

   Jaringan layar jamak merupakan perluasan dari layar tunggal (Siang, 2009). Dalam jaringan ini, selain unit masukkan dan keluaran, ada unit lain yang sering disebut layar tersembunyi (Siang, 2009). Dimungkinkan pula ada

beberapa layar tersembunyi (Siang, 2009). Sama seperti pada unit masukkan dan keluaran, unit-unit dalam satu layar tidak saling berhubungan (Siang, 2009). Contoh dari jaringan layar jamak antara lain : MADALINE, *Backpropagation*, *Neocognitron*, dll (Siang, 2009).

Gambar 2.10 Jaringan Layar Jamak (Beckenkamp, 2002)

3. Jaringan *Recurrent*

   Model jaringan *recurrent* mirip dengan jaringan layar tunggal ataupun ganda (Siang, 2009). Hanya saja, ada neuron keluaran yang memberikan sinyal pada unit masukkan (sering disebut *feedback loop*) (Siang, 2009). Model dari jaringan *recurrent* adalah BAM (*Bidirectional Associative Memory*), *Boltzman Machine*, *Hopfield*, dll (Siang, 2009).

2.4.4 Fungsi Masukkan

Sebelum melakukan perhitungan pada fungsi aktivasi, masukkan (*input*) yang diterima oleh sebuah neuron harus terlebih dahulu melalui fungsi masukkan (*input function*). Persamaan yang dipakai dalam fungsi masukkan adalah

$$in_i = \sum_j W_{j,i} a_j = \mathbf{W}_i \mathbf{a}_i$$

……. Rumus 2.1

Keterangan rumus :

in_i = jumlah masukkan dan bobot pada unit ke *i*

$W_{j,i}$ = bobot masukkan ke *j* pada unit ke *i*

a_j = masukkan ke *j*

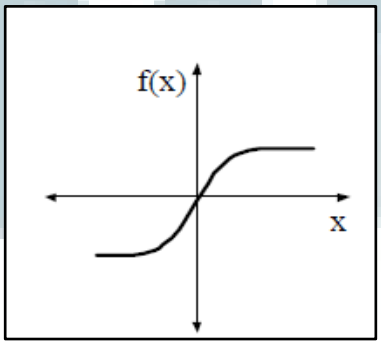
W_i = vektor bobot pada unit ke *i*

a_i = vektor nilai aktivasi dari masukkan ke unit *i*

2.4.5 Fungsi Sigmoid

Fungsi aktivasi merupakan bagian terpenting dalam tahapan perhitungan dalam suatu algoritma (Lejap, 2008). Fungsi aktivasi adalah salah satu perhitungan pada neuron yang bersifat *nonlinear* (Russel & Norvig, 1995). Perhitungan pada fungsi aktivasi melibatkan hasil perhitungan dari fungsi masukkan.

Fungsi Sigmoid adalah fungsi yang paling terkenal digunakan dalam *feed forward* jaringan syaraf karena ketidaklinierannya dan kesederhanaan turunan dari fungsi ini (Mira & Sandoval, 1995). Fungsi sigmoid yang paling umum adalah fungsi pendistribusian logistik yang hanya menyediakan nilai keluaran dari 0 sampai 1 (Beckenkamp, 2002).



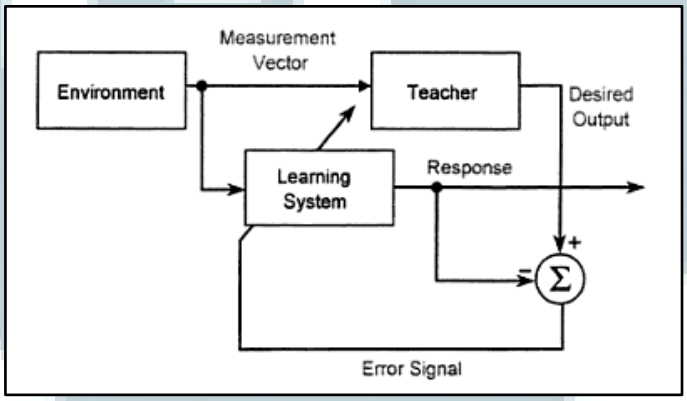
Gambar 2.11 Grafik Fungsi Sigmoid Biner (Beckenkamp, 2002)

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

……. Rumus 2.2

2.4.6 Metode Pembelajaran Terawasi

Metode pembelajaran terawasi adalah sebuah proses yang digunakan untuk melatih jaringan syaraf dengan menggunakan keluaran yang dinginkan (Priddy & Keller, 2005). Metode ini digambarkan seperti seorang murid yang diarahkan oleh seorang guru.



Gambar 2.12 Diagram Metode Pembelajaran Terawasi
(Priddy & Keller, 2005)

Diagram di atas merupakan diagram metode pembelajran terawasi dimana sistem pembelajaran diekspos ke lingkungan (*environment),* yang direpresentasikan oleh vektor kepastian (Priddy & Keller, 2005). Vektor kepastian juga disampaikan kepada guru, yang berdasarkan pengalaman, menentukan respon yang dinginkan (Priddy & Keller, 2005). Respon tersebut digunakan untuk membuat sinyal kesalahan yang menyesuaikan bobot pada system pembelajaran (Priddy & Keller, 2005).

2.4.7 Algoritma Backpropagation

Algoritma *Backpropagation* merupakan salah satu algoritma pembelajaran yang bekerja pada jaringan *multilayer feed-forward*, menggunakan gradien turunan pada bobot untuk meminimalkan kesalahan keluaran (Russel & Norvig, 1995). Terdapat dua fase pada algoritma *Backpropagation* (Lagudu & Sarma ,2013) :

1. Fase *Forward*

   Fase ini merupakan tempat dimana fungsi aktivasi merambat dari layar masukkan ke layar keluaran (Lagudu & Sarma, 2013).

2. Fase *Backward*

   Fase ini merupakan tempat dimana nilai aktual yang diamati dan nilai nominal yang diminta pada layar keluaran dirambatkan mundur ke layar masukkan sehingga bobot dan nilai bisa dapat dimodifikasi sampai nilai keluaran yang dihasilkan mendekati atau bahkan sama dengan dinginkan (Lagudu & Sarma,2013).

Gambar 2.13 Fase *Forward* dan *Backward Backpropagation*
(Lagudu & Sarma ,2013)

Algoritma dari *Backpropagation* itu sendiri adalah sebagai berikut.

1. Menginisialisasi nilai bobot dan bias dengan nilai secara acak yang bernilai antara 0 sampai 1 atau -0.5 – 0.5.
2. Menghitung keluaran dari layar tersembunyi.

$$O_j^h = \frac{1}{1+e^{-\alpha \sum_{i=1}^{m} w_{ij} x_i}}$$

……. Rumus 2.3

Keterangan rumus :

O^h_j = keluaran dari neuron ke j pada layar tersembunyi

X_i = masukkan ke unit ke i

α = fungsi aktivasi

W_{ij} = bobot dari unit ke i ke layar tersembunyi ke j

3. Menghitung keluaran.

$$O_k = \frac{1}{1+e^{-\alpha \sum_{j=1}^{n} w_{jk} u_j}}$$

……. Rumus 2.4

Keterangan rumus :

O_k = keluaran dari unit keluaran ke k

W_{jk} = bobot dari layar tersembunyi ke j ke unit keluaran ke k

u_j = masukkan ke unit keluaran k

4. Menghitung kesalahan untuk unit pada layar tersembunyi dan layar keluaran.

$$\delta_k^o = O_k\,(1 - O_k)(T_k - O_k)$$

……. Rumus 2.5

Keterangan rumus :

T_k = target keluaran untuk unit keluaran ke k

δ^0_k = sinyal *error* untuk unit keluaran ke k

$$\delta_j^h = O_j\,(1 - O_j)(\sum_{k=1}^{p} \delta_k^o\, W_{jk}$$

……. Rumus 2.6

Keterangan rumus :

δ^h_j = sinyal *error* untuk unit ke j pada layar tersembunyi

5. Memperbaharui bobot pada layar keluaran

$$W_{ij}^h(t+1) = W_{ij}^h(t) + (\eta\, \delta_j^h X_i)$$

……. Rumus 2.7

Keterangan rumus :

η = *laju pembelajaran*

Laju pembelajaran merupakan sebuah parameter skalar, yang dianalogikan sebagai besar urutan pada integrasi numerik, yang digunakan untuk mengatur tingkat penyesuaian (Tiwari. dkk, 2013).

6. Memperbaharui bobot pada layar tersembunyi

$$W_{jk}^{o}\ (t+1) = W_{jk}^{o}\ (t) + (\square\ \delta_{k}^{o}\ O_{j})$$

……. Rumus 2.8

7. Mengulang langkah 2 sampai 6 sampai nilai keluaran mendekati yang dinginkan.