BAB II

DASAR TEORI

2.1 Google Maps

Google maps adalah sebuah jasa peta *globe virtual* gratis dan online disediakan oleh *Google* yang dapat diakses di http://maps.google.com. *Google Maps* memberikan informasi mengenai peta dan pemetaan suatu daerah, selain itu *Google Maps* juga mampu memberikan informasi mengenai kepadatan lalu lintas suatu wilayah, bahkan memberikan pencitraan melalui darat ataupun udara.

2.2 Citra Digital

Citra adalah gambar pada bidang dua dimensi. Dalam tinjauan matematis, citra merupakan fungsi kontinu dari intensitas cahaya pada bidang dua dimensi. Ketika sumber cahaya menerangi objek, objek memantulkan kembali sebagian cahaya tersebut. Pantulan ini ditangkap oleh alat-alat pengindera optik, misalnya mata manusia, kamera, scanner dan sebagainya. Bayangan objek tersebut akan terekam sesuai intensitas pantulan cahaya. Ketika alat optik yang merekam pantulan cahaya itu merupakan mesin digital, misalnya kamera digital, maka citra yang dihasilkan merupakan citra digital. Pada citra digital, kontinuitas intensitas cahaya dikuantisasi sesuai resolusi alat perekam.

Di dalam komputer, citra digital disimpan sebagai suatu file dengan format tertentu. Format citra tersebut menunjukan cara sebuah citra digital disimpan, misalnya apakah dengan suatu kompresi atau tidak. Contoh format citra digital adalah .bmp, .jpg, .png, .tif dan sebagainya. Ukuran citra digital dinyatakan dalam *pixel* (*picture element*). Umumnya, nilai setiap *pixel* merupakan kuantisasi harga intensitas cahaya. Dengan demikian, suatu citra digital dapat dipandang sebagai sebuah matriks yang elemen-elemennya menunjukkan intensitas cahaya terkuantisasi. Bedanya terletak pada urutan penyebutan angka ukuran tersebut. Citra digital dengan ukuran 92x112 pixel sebenarnya merupakan sebuah matriks

dengan ukuran 112x92, dimana 112 merupakan banyaknya baris dan 92 merupakan banyaknya kolom.

2.2.1 Citra RGB

Citra RGB adalah citra warna yang setiap *pixel*nya mewakili warna yang merupakan kombinasi tiga warna dasar, yaitu merah, hijau, dan biru (RGB = *Red, Green, Blue*). Setiap warna dasar menggunakan penyimpanan 8 bit = 1 byte (nilai maksimum 255 warna).

2.2.2 Citra Grayscale

Dalam komputasi, suatu citra digital grayscale atau greyscale adalah suatu citra dimana nilai dari setiap *pixel* merupakan sample tunggal. Citra yang ditampilkan dari citra jenis ini terdiri atas warna abu-abu, bervariasi pada warna hitam pada bagian yang intensitas terlemah dan warna putih pada intensitas terkuat. Citra *grayscale* berbeda dengan citra hitam-putih, dimana pada konteks komputer, citra hitam putih hanya terdiri atas 2 warna saja yaitu hitam dan putih saja.

Pada citra *grayscale* warna bervariasi antara hitam dan putih, tetapi variasi warna diantaranya sangat banyak. Citra grayscale seringkali merupakan perhitungan dari intensitas cahaya pada setiap *pixel* pada spektrum elektromagnetik *single band*.

Citra *grayscale* disimpan dalam format 8 bit untuk setiap sample pixel, yang memungkinkan sebanyak 256 intensitas. Format ini sangat membantu dalam pemrograman karena manipulasi bit yang tidak terlalu banyak. Pada aplikasi lain seperti pada aplikasi *medical imaging* dan *remote sensing* biasa juga digunakan format 10 bit, 12 bit dan 16 bit.

2.2.3 Citra Biner

Citra biner (*binary image*) adalah citra yang hanya mempunyai dua nilai derajat keabuan yaitu hitam dan putih, 0 dan 1 dimana 0 menyatakan warna latar belakang (*background*) dengan warna hitam dan 1 menyatakan objek dengan

warna putih. Meskipun saat ini citra berwarna lebih disukai karena memberi kesan yang lebih kaya daripada citra biner, walau begitu citra biner masih tetap dibutuhkan. Misalnya pemanfaatan citra biner untuk penderita buta warna total atau Akromatisme dalam mengenali objek pada gambar berwarna, dimana penderita tidak dapat mengenali warna merah, hijau dan biru, penderita buta warna ini hanya dapat mengenali warna hitam dan putih saja.

2.3 Algoritma Levenberg Marquardt

Algoritma *Levenberg-marquardt* merupakan pengembangan algoritma *backpropagation* standar. Pada algoritma *backpropagation*, proses update bobot dan bias menggunakan *Negative Gradient Descent* secara langsung sedangkan, algoritma *Levenberg-Marquardt* menggunakan pendekatan matrik *Hesian* (H) yang dapat dihitung dengan :

$$\boldsymbol{H = J^T J} \quad \text{.....(2.1)}$$

dimana $\boldsymbol{J^T}$ adalah sebuah *tranpose* dari matrik *jacobian* dan $\boldsymbol{J}$ merupakan sebuah matrik *jacobian* yang berisikan turunan pertama dari *error* jaringan terhadap bobot dan bias jaringan.

$$\boldsymbol{J = \frac{\partial er}{\partial w}} \quad \text{.....(2.2)}$$

∂er = turunan pertama *error* jaringan.

∂w = turunan pertama bobot jaringan.

Sedangkan gradient (g) dapat dihitung dengan :

$$\boldsymbol{g = J^T e} \quad \text{...(2.3)}$$

dimana *e* adalah vektor yang menyatakan semua *error* pada *output* jaringan.

Perubahan pembobot (ΔX) dapat dihitung dengan :

$$\Delta X = [\ \boldsymbol{J^T J + \mu I}\ ]^{-}\ \boldsymbol{J^T e} \quad \text{.....(2.4)}$$

dimana $\boldsymbol{\mu}$ adalah konstanta *learning* dan $\boldsymbol{I}$ adalah sebuah matrik identitas.

sehingga dapat dihitung fungsi bobot-bobot jaringan dan bias (X) untuk perbaikan pembobot dengan :

$$X = X + \Delta X \quad \text{.....(2.5a)}$$

$$X = X \quad + [\ \boldsymbol{J^T J} + \boldsymbol{\mu I}\ ]^{-}\ \boldsymbol{J^T e} \quad \text{.....(2.5b)}$$

2.4 Pengolahan Citra

Pengolahan citra merupakan kegiatan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (komputer). Masukannya adalah citra dan keluarannya juga citra tapi dengan kualitas lebih baik daripada citra masukan. Misal citra warnanya kurang tajam, kabur (*blurring*), mengandung *noise* (misal bintik-bintik putih) sehingga perlu ada pemrosesan untuk memperbaiki citra karena citra tersebut menjadi sulit diinterpretasikan karena informasi yang disampaikan menjadi berkurang.

Pengolahan citra merupakan sebuah bentuk pemrosesan sebuah citra atau gambar dengan cara memproses numerik dari gambar tersebut, dalam hal ini yang diproses adalah masing-masing *pixel* dari gambar tersebut.

Citra menurut kamus Webster adalah suatu representasi, kemiripan dan imitasi dari suatu objek atau benda. Contohnya foto sinar-X *thorax* mewakili keadaan bagian dalam tubuh seseorang. Citra dari sudut pandang matematis, merupakan fungsi kontinu dari intensitas cahaya pada bidang dua dimensi.

Citra yang terlihat merupakan cahaya yang direfleksikan dari sebuah objek. Sumber cahaya menerangi objek, objek memantulkan kembali sebagian dari berkas cahaya tersebut dan pantulan cahaya ditangkap oleh alat-alat optik, misal mata manusia, kamera, scanner, sensor satelit dan sebagainya.

Citra sebagai keluaran dari suatu sistem perekaman data dapat bersifat:

1. Optik berupa foto
2. Analog berupa sinyal *spektrum* seperti gambar pada monitor televisi
3. Digital yang dapat langsung disimpan pada media penyimpanan magnetik

Citra juga dapat dikelompokan menjadi dua yaitu citra tampak dan citra tidak tampak.

1. Citra tampak berupa gambar, apa yang tampak di layar monitor atau televisi.
2. Citra tidak tampak berupa data gambar dalam bentuk file, citra yang dipresentasikan dalam fungsi matematis.

Citra atau gambar bisa diibaratkan sebagai matriks dua dimensi. Gambar digital merupakan suatu fungsi dengan nilai yang berupa intensitas cahaya pada tiap titik pada bidang yang telah dikuantisasi. Titik dimana suatu gambar di-*sampling* disebut *picture element* atau disingkat *pixel*. Nilai intensitas warna pada suatu *pixel* disebut level *grayscale*.

Ada beberapa level *grayscale* berdasarkan banyaknya bit:

- *Binary-valued image*: 1 bit, hanya bernilai 0 atau 1.
- *Gray level* : 8 bit, nilainya antara 0 – 255.
- *High color* : 16 bit, rentang nilainya 2^{16}
- *True color* : 32 bit.

Jika suatu gambar disimpan maka yang disimpan adalah array 2 dimensi, dimana masing-masing merepresentasikan data yang berhubungan dengan *pixel* tersebut.

Pengolahan citra sering diidentikkan dengan *image filtering*. Pengolahan citra sendiri dapat didefinisikan sebagai proses *filtering* sebuah gambar *pixel* demi *pixel*. Tujuan utama dari pengolahan citra adalah untuk meningkatkan kualitas gambar yang diperoleh. Beberapa contoh *filtering* yang biasa dilakukan:

1. *Grayscale Filter*

   Grayscale filter mengubah sebuah gambar berwarna menjadi gambar hitam putih dengan cara mengubah efek warna dari masing-masing *pixel* menjadi derajat keabuan.



Gambar 2. 1 Foto dengan format *grayscale.*

2. *Low pass filter*

   Low pass filter digunakan untuk menghilangkan ruang derau berfrekuensi tinggi dari sebuah gambar digital. Istilah derau atau *noise* digunakan sebagai efek samping dari proses konversi pola dan energi cahaya menjadi energi listrik selama proses konversi gambar dari bentuk analog menjadi bentuk digital. *Noise* merupakan variasi yang tidak diinginkan terjadi dalam sebuah *pixel*. Hasil dari *low pass filter* ini membuat gambar menjadi lebih kabur daripada aslinya.



Gambar 2. 2 Contoh hasil foto *low pass filter*

2.5 Segmentasi Citra

Segmentasi merupakan proses partisi gambar digital ke beberapa daerah dengan tujuan untuk menyederhanakan ataupun merubah representasi gambar menjadi sesuatu yang lebih bermakna dan mudah dianalisa. Ada beberapa metoda yang sering digunakan dalam segmentasi citra antara lain: metode *thresholding*, metode *shapebased*, metode *region growing*, dan metode statistik atau juga disebut metode *clustering*. Masing-masing metoda memiliki kelebihan dan

kelemahan tergantung pada karakteristik dari citra yang akan diproses. Berikut beberapa metode yang umum digunakan dalam segmentasi citra.

a. *Thresholding*

   Metode *thresholding* didasarkan pada pemisahan pixel ke dalam kelas yang berbeda tergantung pada tingkat keabuan masing-masing pixel. Intensitas citra medis seperti tumor dan jaringan pada otak biasanya sangat rumit dan memiliki tingkat keabuan yang sangat dekat sehingga menyebabkan kesulitan penentuan ambang batas (*threshold*). Metode *thresholding* tidak bisa diterapkan untuk citra dengan tingkat keabuan yang berdekatan sehingga biasanya dikombinasikan dengan metode lain.

b. *Growing Region*

   Metode *growing region* seperti menggabungan thresholding dengan kondisi konektivitas atau kriteria daerah homogenitas. Keberhasilan dari metode tersebut bergantung pada kepresisian informasi anatomi untuk meletakkan baik satu mapun beberapa pixel untuk masing-masing daerah homogen. Kelemahan lain dari metode *growing region* adalah metode tersebut hanya dapat bekerja dengan baik pada daerah yang homogen dan membutuhkan operator untuk menentukan daerah yang akan disegmentasi.

c. *Shapebase*

   Metode *shapebased* juga memberikan pendekatan yang cukup sederhana dalam segmentasi citra namun sangat sulit dalam penentuan kontur awal sehingga ketidaktepatan dalam penentuan kontur awal dapat menyebabkan hasil segmentasi yang kurang memuaskan.

2.6 Ekstraksi Ciri Orde Dua

Salah satu teknik untuk memperoleh ciri statistik orde dua adalah dengan menghitung probabilitas hubungan ketetanggaan antara dua *pixel* pada jarak dan orientasi sudut tertentu. Pendekatan ini bekerja dengan membentuk sebuah matriks kookurensi dari data citra, dilanjutkan dengan menentukan ciri sebagai fungsi dari matriks tersebut.

Kookurensi berarti kejadian bersama, yaitu jumlah kejadian satu level nilai *pixel* bertetangga dengan satu level nilai *pixel* lain dalam jarak (*d*) dan orientasi sudut (θ) tertentu. Jarak dinyatakan dalam *pixel* dan orientasi dinyatakan dalam derajat. Orientasi dibentuk dalam empat arah sudut dengan interval sudut 45°, yaitu 0°, 45°, 90°, dan 135°. Sedangkan jarak antar *pixel* biasanya ditetapkan sebesar 1 *pixel*.

Matriks kookurensi merupakan matriks bujursangkar dengan jumlah elemen sebanyak kuadrat jumlah level intensitas *pixel* pada citra. Setiap titik (*p*,*q*) pada matriks kookurensi berorientasi θ berisi peluang kejadian *pixel* bernilai *p* bertetangga dengan *pixel* bernilai *q* pada jarak *d* serta orientasi θ dan (180−θ).

Setelah memperoleh matriks kookurensi tersebut, kita dapat menghitung ciri statistik orde dua yang merepresentasikan citra yang diamati. Terdapat enam ciri tekstural yang dapat diekstraksi dari matriks kookurensi yaitu *Angular Second Moment, Contrast, Correlation, Variance, Inverse Difference Moment*, dan *Entropy*.

a. *Angular Second Moment* (ASM)

   Menunjukkan ukuran sifat homogenitas citra.

$$ASM = \sum_i \sum_j \{p(i,j)\}^2 \quad \text{.....(2.6)}$$

   dimana *p*(*i*,*j*) menyatakan nilai pada baris i dan kolom j pada matriks kookurensi.

b. *Contrast* (CON)

   Menunjukkan ukuran penyebaran atau momen *inersia* (k) elemen-elemen matriks citra. Jika letaknya jauh dari diagonal utama, nilai kekontrasan besar. Secara visual, nilai kekontrasan adalah ukuran variasi antar derajat keabuan suatu daerah citra.

$$CON = \sum_k k^2 \left[\sum_i \sum_j p(i,j) \right] \quad \text{.....(2.7)}$$
$$|i-j| = k$$

c. *Correlation* (COR)

Menunjukkan ukuran ketergantungan linear derajat keabuan citra sehingga dapat memberikan petunjuk adanya struktur linear dalam citra.

$$COR = \frac{\sum_i \sum_j (ij).p(i,j) - \mu_x \mu_y}{\sigma_x \sigma_y} \quad \text{.....(2.8)}$$

d. *Variance* (VAR)

Menunjukkan variasi elemen-elemen matriks kookurensi. Citra dengan transisi derajat keabuan kecil akan memiliki variansi yang kecil pula.

$$VAR = \sum_i \sum_j (i - \mu_x)(j - \mu_y) p(i,j) \quad \text{.....(2.9)}$$

e. *Inverse Different Moment* (IDM)

Menunjukkan kehomogenan citra yang berderajat keabuan sejenis. Citra homogen akan memiliki harga *IDM* yang besar.

$$IDM = \sum_i \sum_j \frac{1}{1+(i-j)^2} p(i,j) \quad \text{.....(2.10)}$$

f. *Entropy* (ENT)

Menunjukkan ukuran ketidakteraturan bentuk. Harga *ENT* besar untuk citra dengan transisi derajat keabuan merata dan bernilai kecil jika struktur citra tidak teratur (bervariasi).

$$ENT_2 = -\sum_i \sum_j p(i,j) \cdot {}^2\log p(i,j) \quad \text{.....(2.11)}$$

2.7 Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan adalah suatu teknologi komputasi yang berbasis hanya pada model syaraf biologis dan mencoba mensimulasikan tingkah laku dan kerja model syaraf terhadap berbagai macam masukan. Sebagai suatu teknologi komputasi, Jaringan Syaraf Tiruan merupakan satu teknik pemrosesan informasi yang menggunakan model kuantitatif. Syaraf biologis mengilhami terciptanya

suatu proses komputasi yang identik dengan kerja *neuron* dalam sistem syaraf manusia. Seperti halnya jaringan syaraf biologis, model matematik Jaringan Syaraf Tiruan menghubungkan sejumlah masukan dan keluaran suatu sistem secara adaptif yang diorganisasikan dalam lapisan elemen pemroses seperti layaknya hubungan antar *neuron* syaraf biologis.

Jaringan Syaraf Tiruan terdiri atas elemen pemroses bernama *neuron*, yang dihubungkan dengan elemen pemroses lain oleh suatu aturan dan bobot. Umumnya, Jaringan Syaraf Tiruan merupakan suatu kumpulan pemroses sinyal analog yang dihubungkan melalui link yang disebut interkoneksi sederhana. Secara skematis, Jaringan Syaraf Tiruan digambarkan dalam bentuk grafik yang mempunyai arah menuju suatu simpul dari elemen pemroses. Arah panah menunjukkan arah normal suatu aliran sinyal. Pemrosesan sinyal di dalam jaringan dilakukan melalui proses komputasi. Dengan demikian, Jaringan Syaraf Tiruan merupakan suatu teknik komputasi pada *software* yang mengemulasikan *neuron* biologis dalam melakukan operasi pengambilan informasi.

Teknologi Jaringan Syaraf Tiruan yang berkembang pesat merupakan solusi persoalan komputasi yang tidak dapat diselesaikan oleh komputer konvensional. Kemudian, Jaringan Syaraf Tiruan belajar dari contoh yang disebut set pelatihan. Karena belajar dari contoh, Jaringan Syaraf Tiruan mempunyai potensi membangun sistem komputasi sebagai hasil pemetaan hubungan masukan dan keluaran yang ada dalam sistem. Set pelatihan dikenal sebagai pola pelatihan berupa suatu vektor dan didapatkan dari sumber seperti citra, sinyal suara, data dari sensor, data keuangan, dan informasi.

Secara garis besar, proses belajar Jaringan Syaraf Tiruan dapat dibagi menjadi dua:

1. Jaringan Syaraf Tiruan yang menggunakan paket pelatihan sebagai proses belajar dan dikenal sebagai proses belajar dengan pengawasan.
2. Jaringan Syaraf Tiruan tanpa paket pelatihan pada proses belajar dan umumnya disebut proses belajar tanpa pengawasan.

Kemampuan dan proses komputasi pada Jaringan Syaraf Tiruan memberikan keuntungan sebagai berikut :

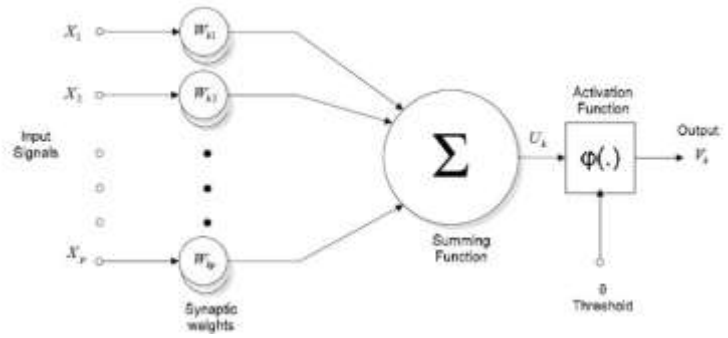
1. Jaringan Syaraf Tiruan bersifat adaptif terhadap perubahan parameter yang mempengaruhi karakteristik sistem, sehingga pada proses belajar Jaringan Syaraf Tiruan mampu belajar secara adaptif dan melaksanakan tugas berbasis pada data yang diberikan saat pelatihan.
2. Jaringan Syaraf Tiruan memiliki kekebalan atau toleran terhadap kesalahan. Artinya, Jaringan Syaraf Tiruan tetap berfungsi walaupun ada ketidak-lengkapan data yang dimasukkan. Jaringan Syaraf Tiruan mempunyai kemampuan mengisi bagian masukan yang kurang lengkap sedemikian rupa sehingga tetap diperoleh keluaran dibutuhkan.
3. Jaringan Syaraf Tiruan dapat dilatih memberikan keputusan dengan memberikan set pelatihan sebelumnya untuk mencapai target tertentu, sehingga Jaringan Syaraf Tiruan mampu membangun dan memberikan jawaban sesuai dengan informasi yang diterima pada proses pelatihan.
4. Jaringan Syaraf Tiruan mempunyai struktur paralel dan terdistribusi. Artinya, komputasi dapat dilakukan oleh lebih dari satu elemen pemroses yang bekerja secara simultan.
5. Jaringan Syaraf Tiruan mampu mengklasifikasi pola masukan dan pola keluaran. Melalui proses penyesuaian, pola keluaran dihubungkan dengan masukan yang diberikan oleh Jaringan Syaraf Tiruan.
6. Jaringan Syaraf Tiruan mengurangi derau, sehingga dihasilkan keluaran yang lebih bersih.
7. Jaringan Syaraf Tiruan dapat dimanfaatkan pada proses optimisasi penyelesaian suatu masalah.
8. Jaringan Syaraf Tiruan dapat digunakan pada proses pengendalian sistem agar masukan memperoleh tanggapan yang diinginkan.

2.7.1 Model *Neuron*

Mengadopsi esensi dasar dari system syaraf biologi, syaraf tiruan digambarkan sebagai berikut : menerima sinyal *input* (X), baik dari data yang dimasukkan atau dari *output* sel syaraf pada jaringan syaraf. Setiap sinyal input datang melalui suatu koneksi atau hubungan yang mempunyai sebuah bobot (*synaptic weight*). Setiap sel syaraf mempunyai sebuah nilai ambang. Jumlah

bobot dari input dan dikurangi dengan nilai ambang kemudian akan mendapatkan suatu aktivasi dari sel syaraf *(post synaptic potential*, PSP, dari sel syaraf). Signal aktivasi kemudian menjadi fungsi aktivasi atau fungsi transfer untuk menghasilkan *output* dari sel syaraf.

Satu sel syaraf terdiri dari tiga bagian, yaitu: fungsi penjumlah (*summing function*), fungsi aktivasi (*activation function*), dan keluaran (*output*).



Gambar 2. 3 Model *Neuron*

Jika kita lihat, neoron buatan diatas mirip dengan sel biologis. Informasi (input) akan dikirim ke *neuron* dengan bobot tertentu. Input ini akan diproses oleh suatu fungsi yang akan menjumlahkan nilai-nilai bobot yang ada. Hasil penjumlahan kemudian akan dibandingkan dengan suatu nilai ambang (*threshold*) tertentu melalui fungsi aktifasi setiap *neuron*. Apabila input tersebut melewati suatu nilai ambang tertentu, maka *neuron* tersebut akan diaktifkan, jika tidak, maka *neuron* tidak akan diaktifkan. Apabila *neuron* tersebut diaktifkan, maka *neuron* tersebut akan mengirimkan *output* melalui bobot-bobot *output*nya ke semua *neuron* yang berhubungan dengannya.

2.7.2 Bobot

Bobot merupakan suatu nilai yang mendefinisikan tingkat atau kepentingan hubungan antara suatu node dengan node yang lain. Semakin besar bobot suatu hubungan menandakan semakin pentingnya hubungan kedua node tersebut.

Bobot merupakan suatu hubungan berupa bilangan real maupun integer, tergantung dari jenis permasalahan dan model yang digunakan. Bobot-bobot

tersebut bisa ditentukan untuk berada didalam interval tertentu. selama proses pelatihan, bobot tersebut dapat menyesuaikan dengan pola-pola input.

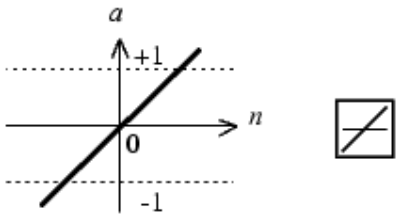
Jaringan dengan sendirinya akan memperbaiki diri terus-menerus karena adanya kemampuan untuk belajar. Setiap ada suatu masalah baru, jaringan dapat belajar dari masalah baru tersebut, yaitu dengan mengatur kembali nilai bobot untuk menyesuaikan karakter nilai.

2.7.3 Fungsi Aktivasi

Pada setiap layer pada jaringan syaraf tiruan terdapat fungsi aktivasi. fungsi ini adalah fungsi umum yang akan digunakan untuk membawa input menuju *output* yang diinginkan. Fungsi aktivasi inilah yang akan menentukan besarnya bobot. Penggunaan fungsi aktivasi tergantung pada kebutuhan dan desired *output*. Contoh dari fungsi aktivasi ini antara lain:

1. Linier atau Pureline

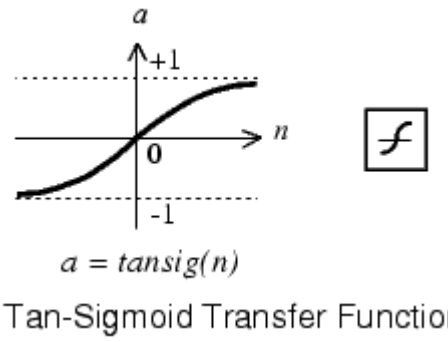
   Fungsi linier akan membawa input ke *output* yang sebanding. fungsi ini digambarkan sebagai berikut:



Gambar 2.4 *Linear Transfer Function*

2. Tansig

   Tansig adalah fungsi sigmoid tangen yang digunakan sebagai fungsi aktivasi. Fungsi ini akan membawa nilai input pada *output* dengan menggunakan rumus *hyperbolic tangen sigmoid*. Nilai maksimal *output* dari fungsi ini adalah 1 dan minimal -1.



Gambar 2.5 *Tan-Sigmoid Transfer Function*

2.7.4 Konsep Dasar Jaringan Syaraf Tiruan

Setiap pola-pola informasi input dan *output* yang diberikan kedalam Jaringan Syaraf Tiruan diproses dalam *neuron*. *Neuron-neuron* tersebut terkumpul di dalam lapisan-lapisan penyusun Jaringan Syaraf Tiruan tersebut dapat dibagi menjadi tiga, yaitu:

1. Lapisan Input

   Unit-unit di dalam lapisan input disebut unit-unit input. Unit-unit input tersebut menerima pola inputan data dari luar yang menggambarkan suatu permasalahan.
2. Lapisan Tersembunyi

   Unit-unit di dalam lapisan tersembunyi disebut unit-unit tesembunyi. Dimana *output*nya tidak dapat secara langsung diamati.
3. Lapisan *output*

   Unit-unit di dalam lapisan *output* disebut unit-unit *output*. *Output* dari lapisan ini merupakan solusi Jaringan Syaraf Tiruan terhadap suatu permasalahan.

2.7.5 Arsitektur Jaringan

Jaringan Syaraf Tiruan memiliki beberapa arsitektur jaringan yang sering digunakan dalam berbagai aplikasi. Arsitektur Jaringan Syaraf Tiruan tersebut, antara lain:

1. Jaringan Layer Tunggal (*Single Layer Network*)
   Jaringan dengan lapisan tunggal terdiri dari 1 *layer* input dan 1 *layer output*. Setiap *neuron*/unit yang terdapat di dalam lapisan/*layer* input selalu terhubung dengan setiap *neuron* yang terdapat pada layer *output*. Jaringan ini hanya menerima input kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi.
2. Jaringan Layer Jamak (*multi layer network)*
   Jaringan dengan lapisan jamak memiliki ciri khas tertentu yaitu memiliki tiga jenis *layer* yakni *layer* input, *layer output* dan *layer* tersembunyi (*Hidden layer*). Jaringan dengan banyak lapisan ini dapat menyelesaikan permasalahan yang lebih kompleks dibandingkan jaringan dengan lapisan tunggal. Namun, proses pelatihan sering mebutuhkan waktu yang cenderung lama.
3. Jaringan Layer Kompetitif
   Pada jaringan ini sekumpulan *neuron* bersaing untuk menjadi aktif. Algoritma yang menggunakan metode ini adalah *Learning Vector Quantization* (LVQ)

2.8 Perangkat lunak

2.8.1 Software Matlab

Matlab adalah sebuah bahasa (pemrograman) dengan kinerja yang tinggi (*high-performance*) untuk komputasi teknis, yang mengintegrasikan komputasi, visualisasi dan pemrograman di dalam lingkungan yang mudah penggunaannya dalam memecahkan persoalan dengan solusinya yang dinyataka dengan notasi matematik. Penggunaan matlab, yaitu:

- Matematik dan Komputasi
- Pengembangan Algoritma
- Pemodelan, simulasi dan pembuatan *Prototipe*

- Grafik untuk Sains dan Teknik
- Pengembangan aplikasi, termasuk pembuatan antarmuka grafis untuk pengguna (*Graphical User Interface*).

Matlab adalah sistem interaktif yang menggunakan elemen data dasarnya adalah *array* yang tidak membutuhkan dimensi. Hal ini mempermudah untuk menyelesaikan masalah komputasi terutama menyangkut matriks dan vector.

Nama Matlab merupakan singkatan dari *matrix laboratory*. Pada awalnya Matlab dibuat untuk mempermudah pengembangan perangkat lunak berbasis matriks oleh proyek LINPACK dan EISPACK.

Matlab memiliki beberapa keunggulan dan kemudahan dalam desain dan analisis sistem kendali dan sistem linear. Fungsi-fungsi pendukung terangkum dalam *Control System Toolbox* dan *Signal Proccesing Toolbox*. Kebanyakan operasi dalam sistem kendali dan sistem linear berhubungan dengan operasi matriks, aritmatika kompleks, menghitung akar, model konversi, dan penggambaran fungsi-fungsi yang kompleks. Seperti telah kita ketahui sebelumnya, Matlab telah didesain untuk memudahkan pengerjaan operasi-operasi tersebut diatas, dan analisa fungsi. Pelatihan ini lebih ditekankan pada pemrograman dan menggunakan fungsi-fungsi dasar dalam bahasa Matlab sehinga teori mengenai ilmu sistem itu sendiri hanya akan disinggung sedikit pada topik-topik yang relevan.

2.8.2 Sistem Matlab

Sebagai sebuah sistem Matlab tersusun dari lima bagian utama, yaitu :

1. *Development Environment*. Merupakan sekumpulan perangkat dan fasilitas yang membantu anda untuk menggunakan fungsi-fungsi dan file-file Matlab. Beberapa perangkat ini merupakan sebuah Graphical User Interfaces (GUI). Termasuk didalamnya adalah Matlab *desktop*, *Command window*, *Command history*, sebuah editor, *debugger*, dan *browsers* untuk melihat *help*, *workspace*, *files*, dan *search path*.

2. Matlab *Mathematical Function Library*. Merupakan sekumpulan algoritma komputasi mulai dari fungsi-fungsi dasar sepertri: *sum, sin, cos*, dan *complex arithmetic*, sampai dengan fungsi-fungsi yang lebih komplek seperti *matrix inverse, matrix eigenvalues, Bessel functions*, dan *fast Fourier transforms.*
3. Matlab *Language*. Merupakan suatu *high level matrix* atau *array language* dengan *control flow statements, functions*, data *structures*, input atau *output*, dan fitur-fitur *object oriented programming*. Ini memungkinkan bagi kita untuk melakukan kedua hal baik pemrograman dalam lingkup sederhana untuk mendapatkan hasil yang cepat dan pemrograman dalam lingkup yang lebih besar untuk memperoleh hasil-hasil dan aplikasi yang komplek.
4. *Graphics***.** Matlab memiliki fasilitas untuk menampilkan vektor dan matriks sebagai suatu grafik. Di dalamnya melibatkan *high level functions* (fungsi-fungsi level tinggi) untuk visualisasi data dua dikensi dan data tiga dimensi, *image processing*, *animation*, dan *presentation graphics*. Ini juga melibatkan fungsi level rendah yang memungkinkan bagi anda untuk membiasakan diri untuk memunculkan grafik mulai dari bentuk yang sederhana sampai dengan tingkatan *graphical user interfaces* pada aplikasi Matlab.

Matlab *Application Program Interface (API)*. Merupakan suatu library yang memungkinkan program yang telah anda tulis dalam bahasa C dan Fortran mampu berinterakasi dengan Matlab. Ini melibatkan fasilitas untuk pemanggilan *routines* dari Matlab (*dynamic linking*), pemanggilan Matlab sebagai sebuah computational engine, dan untuk membaca dan menuliskan Mat-*files*.