BAB 2
LANDASAN TEORI

2.1 Citra

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

Citra dapat berupa citra diam dan citra bergerak. Citra diam adalah citra tunggal yang tidak bergerak. Citra bergerak (*moving images*) adalah rangkaian citra diam yang ditampilkan secara beruntun (sekuensial) sehingga memberi kesan pada mata kita sebagai gambar yang bergerak. Setiap citra di dalam rangkaian itu disebut *frame*. Gambar-gambar yang tampak pada film layar lebar atau televisi pada hakikatnya terdiri atas ratusan sampai ribuan frame. [3]

2.2 Pengolahan Citra

Meskipun sebuah citra kaya informasi, namun seringkali citra yang kita miliki mengalami penurunan mutu (degradasi), misalnya mengandung cacat atau derau (*noise*), warnanya terlalu kontras, kurang tajam, kabur (*blurring*), dan sebagainya. Tentu saja citra semacam ini menjadi lebih sulit diinterpretasi karena informasi yang disampaikan oleh citra tersebut menjadi berkurang.

Agar citra yang mengalami gangguan mudah diinterpretasi (baik oleh manusia maupun mesin), maka citra tersebut perlu dimanipulasi menjadi citra lain yang kualitasnya lebih baik. Bidang studi yang menyangkut hal ini adalah pengolahan citra (*image processing*). Pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakan komputer, menjadi citra yang kualitasnya lebih baik. Umumnya, operasi-operasi pada pengolahan citra diterapkan pada citra bila :

1) Perbaikan atau memodifikasi citra perlu dilakukan untuk meningkatkan kualitas penampakan atau untuk menonjolkan beberapa aspek informasi yang terkandung di dalam citra.
2) Elemen di dalam citra perlu dikelompokkan, dicocokkan, atau diukur.
3) Sebagian citra perlu digabung dengan bagian citra yang lain.

Di dalam bidang komputer, sebenarnya ada tiga bidang studi yang berkaitan dengan data citra yakni grafika komputer (*computer graphics*), pengolahan citra (*image processing*), dan pengenalan pola (*pattern recognition/image interpretation*), namun ketiganya memiliki tujuan yang berbeda.

Pengolahan Citra bertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (dalam hal ini komputer). Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas lebih baik daripada citra masukan. Termasuk ke dalam bidang ini juga adalah pemampatan citra (*image compression*).

Secara umum, operasi pengolahan citra dapat diklasifikasikan menjadi :

1) Perbaikan kualitas citra (*image enhancement*)

   Operasi ini bertujuan untuk memperbaiki kualitas citra dengan cara memanipulasi parameter-parameter citra. Dengan operasi ini, ciri-ciri khusus dalam citra dapat lebih ditonjolkan. Contoh-contoh operasi dalam perbaikan kualitas citra adalah perbaikan kontras gelap / terang, perbaikan tepian objek (*edge enhancement*), penajaman (*sharpening*), pemberian warna semu (*pseudocoloring*), dan penapisan derau (*noise*).
2) Pemugaran citra (*image restoration*)

   Hampir sama dengan operasi perbaikan kualitas citra, operasi ini bertujuan untuk menghilangkan / meminimumkan cacat pada citra. Contoh-contoh operasi dalam pemugaran citra antara lain penghilangan kesamaran (*deblurring*) dan penghilangan derau (*noise*)
3) Pemampatan citra (*image compression*)

   Operasi ini bertujuan supaya citra dapat direpresentasikan dalam bentuk yang lebih kompak sehingga memori yang digunakan lebih sedikit. Perlu

diperhatikan dalam pemampatan citra, citra yang dimampatkan harus tetap memiliki kualitas yang baik.

4) Segmentasi citra (*image segmentation*)

   Operasi ini bertujuan untuk memecah suatu citra ke dalam beberapa segmen dengan suatu kriteria tertentu. Jenis operasi ini erat kaitannya dengan pengenalan pola.

5) Pengorakan citra (*image analysis*)

   Operasi ini bertujuan untuk menghitung besaran kumulatif dari citra untuk menghasilkan deskripsinya. Teknik pengorakan citra mengekstraksi ciri-ciri tertentu yang membantu dalam identifikasi objek. Contoh-contoh operasi pengorakan citra antara lain pendeteksian tepian objek (*edge detection*), ekstraksi batas (*boundary*), dan representasi daerah (*region*).

6) Rekonstruksi citra (*image reconstruction*)

   Operasi ini bertujuan untuk membentuk ulang objek dari beberapa citra hasil proyeksi. Operasi ini banyak digunakan dalam bidang medis seperti pada foto *rontgen* dengan sinar X yang digunakan untuk membentuk ulang gambar organ tubuh.

Pengolahan citra telah dimanfaatkan dalam berbagai bidang, beberapa di antaranya adalah [3] :

1) Bidang perdagangan
   a. Pembacaan kode batang (*bar code*) yang tertera pada barang (umum digunakan di pasar swalayan/supermarket).
   b. Mengenali huruf/angka pada suatu formulir secara otomatis.
2) Bidang militer
   a. Mengenali sasaran peluru kendali mela lui sensor visual.
   b. Mengidentifikasi jenis pesawat musuh.
3) Bidang kedokteran
   a. Pengolahan citra sinar X untuk mammografi (deteksi kanker payudara)
   b. NMR (*Nuclear Magnetic Resonance*)

c. Mendeteksi kelainan tubuh dari foto sinar X.
d. Rekonstruksi foto janin hasil USG

4) Bidang biologi
   Pengenalan jenis kromosom melalui gambar mikroskopik
5) Bidang komunikasi data
   Pemampatan citra yang ditransmisi.
6) Bidang hiburan
   Pemampatan video (*MPEG*)
7) Bidang robotika
   Visualy-guided autonomous navigation
8) Bidang pemetaan
   Klasifikasi penggunaan tanah melalui foto udara/LANDSAT
9) Bidang geologi
   Mengenali jenis batu-batuan melalui foto udara/LANDSAT
10) Bidang hukum
    a. Pengenalan sidik jari
    b. Pengenalan foto narapidana.

2.3 Ekstraksi Ciri

Ekstraksi ciri adalah proses untuk mengubah tiap gambar yang terdapat dalam file gambar menjadi susunan kode biner (angka 0 dan 1). Suatu ekstraksi ciri dapat meminimumkan dimensi data dengan mempertahankan informasi penting yang terkandung di dalam sebuah objek.

Ekstraksi ciri merupakan langkah awal dalam melakukan klasifikasi dan interpretasi citra. Proses ini berkaitan dengan kuantisasi karakteristis citra ke dalam sekelompok nilai ciri yang sesuai.

Tujuan dari pemilihan ciri khas (*feature selection*) citra adalah untuk :

1) Mencari ciri yang paling optimum dari suatu objek yang dapat digunakan untuk membedakan objek tersebut dengan objek lainnya.
2) Menentukan prosedur langkah pengambilan ciri yang akan digunakan serta menentukan prosedur klasifikasinya.

Beberapa metode yang dapat digunakan untuk melakukan ekstraksi ciri antara lain :

1) Metode statistik

   Metode statistik menggunakan perhitungan statistik distribusi derajat keabuan (histogram) dengan mengukur tingkat kekontrasan, granularitas, dan kekasaran suatu daerah dari hubungan ketetanggaan antar piksel di dalam citra.

2) Metode spektral

   Metode spektral berdasarkan pada fungsi autokorelasi suatu daerah atau power distribution pada domain transformasi *Fourier* dalam mendeteksi periodisitas tekstur.

3) Metode struktural

   Analisis dengan metode ini menggunakan deskripsi primitif tekstur dan aturan sintaktik. Metode struktural banyak digunakan untuk pola-pola makrostruktur. [4]

2.4 Ekstraksi Ciri Statistik

Ekstraksi ciri statistik merupakan metode pengambilan ciri yang didasarkan pada karakteristik histogram citra. Histogram menunjukkan probabilitas kemunculan nilai derajat keabuan piksel pada suatu citra. Ekstraksi ciri statistik terbagi menjadi dua metode yaitu [5] :

1) Ekstraksi ciri orde pertama

   Parameter ciri yang dihasilkan dari ekstraksi ciri orde pertama yakni :

   a. *Mean*

      Menunjukkan ukuran dispersi dari suatu citra.

$$\boldsymbol{\mu = \sum_n f_n p(f_n)} \qquad (2.1)$$

      dengan

      fn : nilai skala keabuan

      P(fn) : probabilitas kemunculan skala keabuan

   b. *Variance*

      Menunjukkan variasi elemen pada histogram dari suatu citra.

$$\sigma^2 = \sum_n (f_n - \mu)^2 p(f_n) \qquad (2.2)$$

c. *Skewness*

Menunjukkan tigkat kemencengan relatif kurva histogram dari suatu citra.

$$\alpha_3 = \frac{1}{\sigma^3} \sum_n (f_n - \mu)^3 p(f_n) \qquad (2.3)$$

d. *Kurtosis*

Menunjukkan tingkat keruncingan realatif kurva histogram dari suatu citra.

$$\alpha_4 = \frac{1}{\sigma^4} \sum_n (f_n - \mu)^4 p(f_n) - 3 \quad (2.4)$$

e. *Entrophy*

Menunjukkan ukuran ketidakteraturan bentuk dari suatu citra.

$$H = -\sum_n p(f_n) \cdot {}^2\log p(f_n) \qquad (2.5)$$

2) Ekstraksi ciri orde kedua

Ekstraksi ciri orde kedua digunakan bila ekstraksi ciri orde pertama tidak dapat lagi digunakna untuk mengenali perbedaan antar citra. Parameter ciri yang dihasilkan dari ekstraksi ciri orde kedua yakni :

a. *Angular Second Moment*

Menunjukkan ukuran sifat homogenitas citra.

$$ASM = \sum_i \sum_j P_d^2 (i, j) \qquad (2.6)$$

b. *Contrast*

Menunjukkan ukuran penyebaran (momen inersaia) elemen-elemen matriks citra. Jika letaknya jauh dari diagonal utama, maka nilai kekontrasan besar. Nilai kekontrasan adalah ukuran variasi antar derajat keabuan suatu daerah citra.

$$CONTRAST = \sum_i \sum_j (i - j)^2 P_d^2 (i, j) \quad (2.7)$$

c. *Correlation*

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

$$\boldsymbol{COR} = \sum_{\boldsymbol{i}} \sum_{\boldsymbol{j}} \frac{\boldsymbol{ijPd(i,j)} - \boldsymbol{\mu_x \mu_y}}{\boldsymbol{\sigma_x \sigma_y}} \qquad (2.8)$$

d. *Inverse Different Moment* (IDM)

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

$$\boldsymbol{IDM} = \sum_{\boldsymbol{i}} \sum_{\boldsymbol{j}} \frac{\mathbf{1}}{\mathbf{1}+(\boldsymbol{i}-\boldsymbol{j})^2} \boldsymbol{Pd(i,j)} \qquad (2.9)$$

e. *Entrophy*

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

$$\boldsymbol{ENT} = -\sum_{\boldsymbol{i}} \sum_{\boldsymbol{j}} \boldsymbol{P_d(i,j)} \log(\boldsymbol{P_d(i,j)}) \quad (2.10)$$

2.5 Deteksi Tepi

Faktor kunci dalam mengekstraksi ciri adalah kemampuan mendeteksi keberadaan tepi (*edge*) dari objek di dalam citra. Yang dimaksud dengan tepi di sini adalah perubahan nilai intensitas derajat keabuan yang mendadak (besar) dalam jarak yang singkat. Perbedaan intensitas inilah yang menampakkan rincian pada gambar. Tepi biasanya terdapat pada batas antara dua daerah berbeda pada suatu citra. Tepi dapat diorientasikan dengan suatu arah, dan arah ini berbeda-beda pada bergantung pada perubahan intensitas.

Ada tiga macam tepi yang terdapat di dalam citra digital yakni :

1) Tepi curam

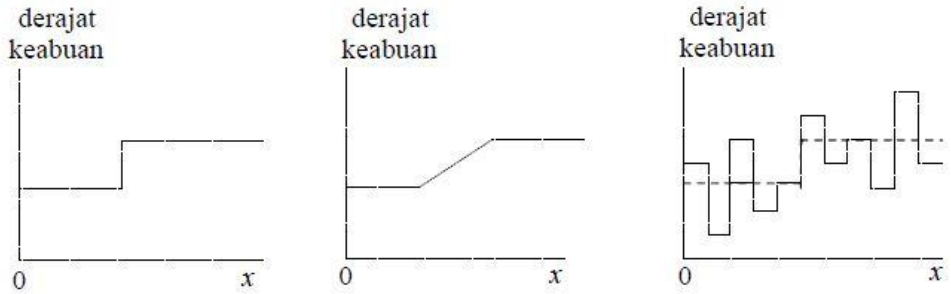
   Tepi dengan perubahan intensitas yang tajam. Arah tepi berkisar 90°.

2) Tepi landai

   Disebut juga tepi lebar, yaitu tepi dengan sudut arah yang kecil. Tepi landai dapat dianggap terdiri dari sejumlah tepi-tepi lokal yang lokasinya berdekatan.

3) Tepi yang mengandung derau (*noise*)

   Umumnya tepi yang terdapat pada aplikasi *computer vision* mengandung derau. Operasi peningkatan kualitas citra (*image enhancement*) dapat dilakukan terlebih dahulu sebelum pendeteksian tepi.



Gambar 2.1 Jenis-Jenis Tepi Citra Digital (dari kiri ke kanan)
Tepi Curam, Tepi Landai, Tepi yang Mengandung Derau

Tujuan operasi pendeteksian tepi adalah untuk meningkatkan penampakan garis batas suatu daerah atau objek di dalam citra. Karena tepi termasuk ke dalam komponen berfrekuensi tinggi, maka pendeteksian tepi dapat dilakukan dengan penapis lolos-tinggi. Terdapat beberapa teknik yang digunakan untuk mendeteksi tepi, antara lain [3] :

1) Operator gradien pertama (*differential gradient*)
2) Operator turunan kedua (*Laplacian*)
3) Operator kompas (*compass operator*)

2.6 *Content Based Image Retrieval* (CBIR)

Content Based Image Retrieval System (CBIR) merupakan teknik pencarian kembali gambar yang mempunyai kemiripan karakteristik atau content dari sekumpulan gambar. Proses secara umum dari CBIR pada image query dilakukan proses ekstraksi fitur. Parameter fitur gambar yang dapat digunakan untuk retrieval pada sistem ini seperti histogram, susunan warna, tekstur, bentuk, tipe spesifik dari obyek, tipe event tertentu, nama individu, lokasi, emosi. Image query terdiri atas 3 level, yaitu:

1) Level 1

   Retrieval dengan *primitive feature*, seperti color, shape, teksture.
2) Level 2

   Retrieval dengan *logical feature*, seperti tipe obyek, individu obyek atau orang.

3) Level 3

   Retrieval dengan *abstract feature*, seperti nama even, tipe aktifitas, emotional, religius.

Beberapa pokok persoalan yang sering terjadi dalam pembangunan CBIR dapat diuraikan sebagai berikut:

1) Pemahaman gambar yang dibutuhkan oleh user dan pencarian informasi.
2) Pengidentifikasian cara yang sesuai dalam *image content* atau karakteristik dari gambar.
3) Proses ekstraksi fitur dari gambar.

Ekstraksi fitur merupakan proses penting pada sistem CBIR karena hasil dari proses ini perbedaan pada setiap gambar dapat diketahui berdasarkan cirinya seperti ciri bentuk, warna, tekstur dan lain – lain. Beberapa teknik yang dapat digunakan untuk ekstraksi fitur antara lain : teknik analisis komponen utama, besaran statistik , histogram warna, *wavelet transform* dan lain – lain. [6]

2.7 Kecerdasan Buatan

Kecerdasan buatan atau *Artificial Intelligence* merupakan cabang dari ilmu komputer yang mempelajari bagaimana membuat mesin (komputer) agar dapat melakukan pekerjaan dengan baik seperti apa yang dilakukan manusia atau bahkan lebih baik.

Agar mesin bisa menjadi cerdas, maka mesin harus diberi bekal pengetahuan dan kemampuan untuk menalar. Cerdas yang dimaksud di sini adalah memilki pengetahuan dan pengalaman yang dapat digunakan untuk mengambil suatu keputusan atau suatu tindakan dengan baik. Terdapat dua bagian utama yang dibutuhkan dalam pengembangan aplikasi kecerdasan buatan, yakni [7] :

1) Basis pengetahuan (*Knowledge base*)

   Basis pengetahuan berisi fakta-fakta, teori, pemikiran, dan hubungan antara satu dengan yang lainnya.

2) Motor inferensi (*Inference agent*)

   Motor inferensi merupakan kemampuan untuk menarik kesimpulan berdasarkan pengetahuan.

Definisi – definisi kecerdasan buatan, diorganisasikan ke dalam empat kategori yakni [8] :

1) *Thinking humanly* (*The cognitive modelling approach*)

   Pendekatan ini mencoba menangkap pemikiran – pemikiran melalui introspeksi dan melalui eksperimen – eksperimen psikologi.

2) *Acting humanly* (*The Turing test approach*)

   Seperti eksperimen yang dilakukan Alan Turing (1950), dirancang suatu ujian bagi komputer yang menggunakan kecerdasan buatan, apakah mampu mengelabui seorang manuasi yang menginterogasi komputer tersebut melalui *teletype.* Jika *interrogator* tidak dapat membedakan apakah yang diinterogasi adalah manusia atau komputer, maka komputer tersebut lolos dari *Turing test*. Komputer tersebut perlu memiliki kemampuan: *Natural Language Processing*, *Knowledge Representation*, *Automated Reasoning*, *Machine Learning*, *Computer Vision*, *Robotics*. *Turing Test* sengaja menghindari interaksi fisik antara *interrogator* dan komputer karena simulasi fisik manusia tidak memerlukan kecerdasan.

3) *Thinking rationally* (*The laws of thought approach*)

   Terdapat dua masalah dalam pendekatan ini, yang pertama tidak mudah untuk membuat pengetahuan informal dan menyatakannya dalam *formal term* yang diperlukan oleh notasi logika, khususnya ketika pengetahuan tersebut memiliki kepastian kurang dari 100%. Selain itu, masalah yang kedua adalah terdapat perbedaan besar antara dapat memecahkan masalah "dalam prinsip" dan juga memecahkannya dalam praktek.

4) *Acting rationally* (*The rational agent approach*)

   Membuat inferensi yang benar kadang-kadang merupakan bagian dari suatu *rational agent*, karena satu cara untuk melakukan aksi secara rasional adalah menalar secara logika untuk mendapatkan kesimpulan bahwa aksi yang diberikan akan mencapai tujuan, dan kemudian melakukan aksi atas kesimpulan tersebut.

2.8 Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan (JST) atau yang disebut juga dengan *neural network* merupakan salah satu cabang dari kecerdasan buatan. JST adalah simulasi perangkat lunak dengan arsitektur dan operasi yang diilhami dari pengetahuan tentang sel syaraf biologis di dalam otak (neuron). JST merupakan salah satu representasi buatan dari otak manusia yang bekerja dengan menstimulasi proses pembelajaran yang diterima otak manusia tersebut. Ide dari ilmu tentang *neural network* ini pertama kali ditemukan oleh seorang ahli psikologi, William James, pada tahun 1890 yang kemudian dilakukan pengembangan model neuron secara matematik oleh McCulloch dan Pitts pada tahun 1943. JST menyerupai otak manusia dalam dua hal, yaitu:

1) Pengetahuan diperoleh jaringan melalui proses belajar.
2) Kekuatan hubungan antar sel syaraf (*neuron*) yang dikenal sebagai bobot-bobot sinaptik digunakan untuk menyimpan pengetahuan.

JST mempunyai sifat dan kemampuan:

1) Nonlinieritas (*Nonlinearity*)
2) Pemetaan Input-Output (*Input-Output Mapping*)
3) Adaptivitas (*Adaptivity*)
4) Respon Yang Jelas (*Evidential Response*)
5) Informasi Yang Sesuai Dengan Keadaan (*Contextual Information*)
6) Toleransi Kesalahan (*Fault Tolerance*)
7) Kemampuan Implementasi Pada VLSI (*VLSI Implementability*)
8) Keseragaman Analisis Dan Perancangan (*Unifomity of Analysis and Design*)
9) Analogi Sel Syaraf Biologi (*Neurobiological Analogy*)

Dengan menggunakan analogi sistem kerja otak manusia, JST terdiri dari sebuah unit pemroses yang disebut neuron yang mana merupakan representasi dari akson dalam otak manusia. Neuron berisi penambah (*adder*) dan fungsi aktivasi, sejumlah bobot sebagai representasi sinaps dalam otak manusia, dan sejumlah vektor masukan sebagai representasi dendrit dalam otak manusia. JST dikembangkan berdasarkan asumsi-asumsi [11] :

1) Pemrosesan informasi terjadi pada elemen - elemen sederhana yang disebut neuron.
2) Sinyal – sinyal merambat di antara neuran melalui suatu sambungan penghubung.
3) Setiap sambungan penghubung memiliki bobot yang bersesuaian yang berfungsi untuk menggandakan / mengalikan sinyal yang dikirim.
4) Setiap neuron menerapkan fungsi aktifasi terhadap sinyal hasil penjumlahan berbobot yang masuk untuk menentukan sinyal keluaran.

Pada dasarnya karakteristik JST ditentukan oleh [9] :

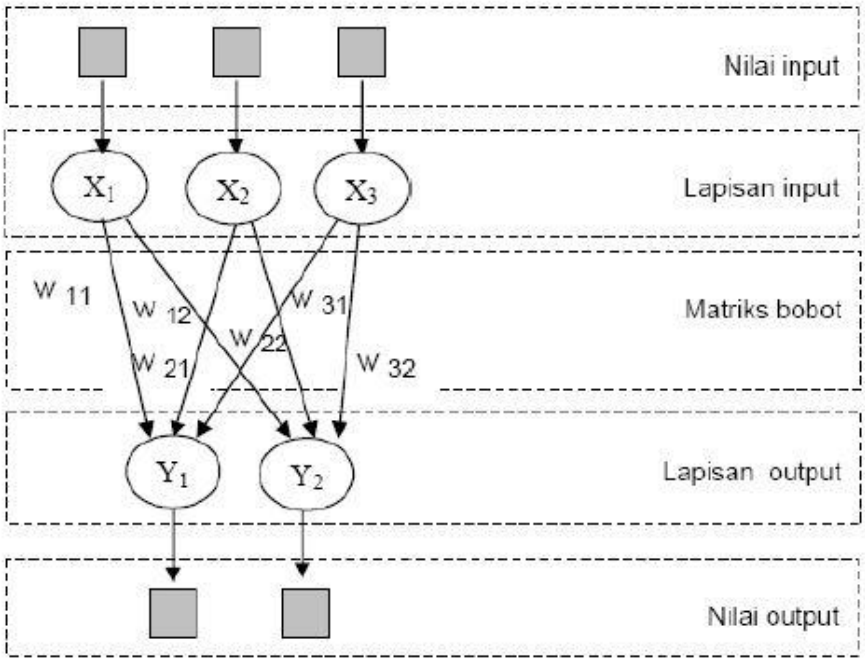
1) Pola hubungan antar neuron (disebut arsitektur jaringan).
2) Metode penentuan bobot-bobot sambungan (disebut dengan pelatihan atau proses belajar jaringan).
3) Fungsi aktivasi.

Pada JST, neuron-neuron akan dikumpulkan dalam lapisan-lapisan (*layer*) yang disebut dengan lapisan neuron (*neuron layers*). Neuron-neuron pada satu lapisan akan dihubungkan dengan lapisan-lapisan sebelum dan sesudahnya. Informasi yang diberikan pada jaringan syaraf akan dirambatkan lapisan ke lapisan, mulai dari lapisan masukan sampai ke lapisan keluaran melalui lapisan tersembunyi (*hidden layer*). Faktor terpenting dalam menentukan kelakuan suatu neuron adalah fungsi aktivasi dan pola bobotnya. Umumnya neuron-neuron yang terletak pada lapisan yang sama akan memiliki keadaan yang sama sehingga pada setiap lapisan yang sama neuron-neuron memiliki fungsi aktivasi yang sama. Bila neuron-neuron pada suatu lapisan (misal lapisan tersembunyi) akan dihubungkan dengan neuron-neuron pada lapisan lain (misal lapisan keluaran) maka setiap neuron pada lapisan tersebut (lapisan tersembunyi) juga harus dihubungkan dengan setiap neuron pada lapisan lainnya (lapisan keluaran). Terdapat 3 macam arsitektur JST, yakni:

1) Jaringan dengan lapisan tunggal (*single layer net*)

   Jaringan ini memiliki 1 lapisan dengan bobot-bobot terhubung. Jaringan ini hanya menerima masukan kemudian secara langsung akan mengolahnya menjadi keluaran tanpa melalui lapisan tersembunyi.

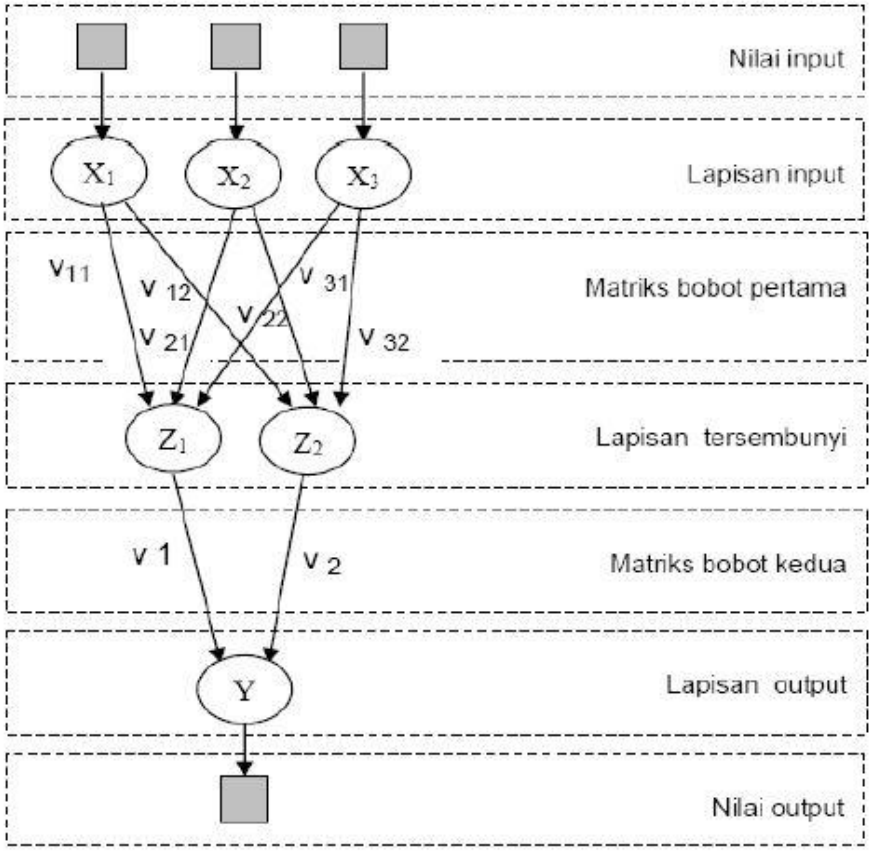
Seberapa besar hubungan antara 2 neuron ditentukan oleh bobot yang bersesuaian. Semua unit masukan akan dihubungkan dengan setiap unit keluaran.



Gambar 2.2 JST dengan Lapisan Tunggal

2) Jaringan dengan banyak lapisan (*multilayer net*)

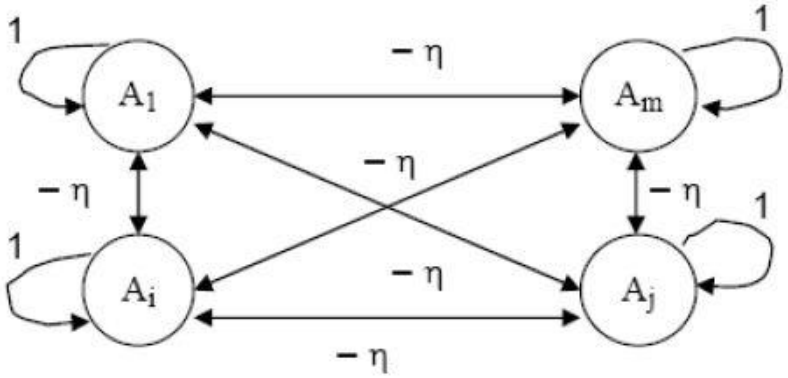
Jaringan ini memiliki 1 atau lebih lapisan yang terletak diantara lapisan masukan dan lapisan keluaran. Umumnya ada lapisan bobot-bobot yang terletak antara 2 lapisan yang bersebelahan. Jaringan dengan banyak lapisan dapat menyelesaikan permasalahan yang lebih sulit daripada lapisan tunggal, namun dengan pembelajaran yang lebih rumit. Pada banyak kasus, pembelajaran pada jaringan dengan banyak lapisan ini terbukti lebih baik dalam menyelesaikan masalah.



Gambar 2.3 JST dengan Banyak Lapisan

3) Jaringan dengan lapisan kompetitif (*competitive layer net*)

Pada jaringan ini sekumpulan neuron bersaing untuk mendapatkan hak menjadi aktif. Umumnya hubungan antar neuron pada lapisan kompetitif ini tidak diperlihatkan pada diagram arsitektur.



Gambar 2.4 JST dengan Lapisan Kompetitif

JST dapat belajar dari pengalaman, melakukan generalisasi atas contoh – contoh yang diperolehnya dan mengabstraksi karakteristik esensial masukan bahkan untuk data yang tidak relevan. Algoritma untuk JST beroperasi secara langsung dengan angka sehingga data yang tidak numerik harus diubah menjadi data numerik. JST tidak diprogram untuk menghasilkan keluaran tertentu. Semua keluaran atau kesimpulan yang ditarik oleh jaringan didasarkan pada pengalamannya selama mengikuti proses pembelajaran. Pada proses pembelajaran, ke dalam JST dimasukkan pola-pola masukan (dan keluaran) lalu jaringan akan diajari untuk memberikan jawaban yang bisa diterima. [8]

Pelatihan Jaringan Syaraf bertujuan untuk mencari bobot-bobot yang terdapat pada setiap *layer*. Terdapat dua jenis pelatihan dalam sistem JST, yakni [11] :

1) *Supervised Learning*

   Dalam proses pelatihan ini, jaringan dilatih dengan cara diberikan data-data yang disebut *training data* yang terdiri atas pasangan *input-output* yang diharapkan dan disebut *associative memory*. Setelah jaringan dilatih, *associative memory* dapat mengingat suatu pola. Jika jaringan diberi *input* baru, jaringan dapat menghasilkan *output* seperti yang diharapkan berdasarkan pola yang sudah ada.

2) *Unsupervised Learning*

   Dalam proses pelatihan ini, jaringan dilatih hanya dengan diberi data *input* yang memiliki kesamaan sifat tanpa disertai *output*.

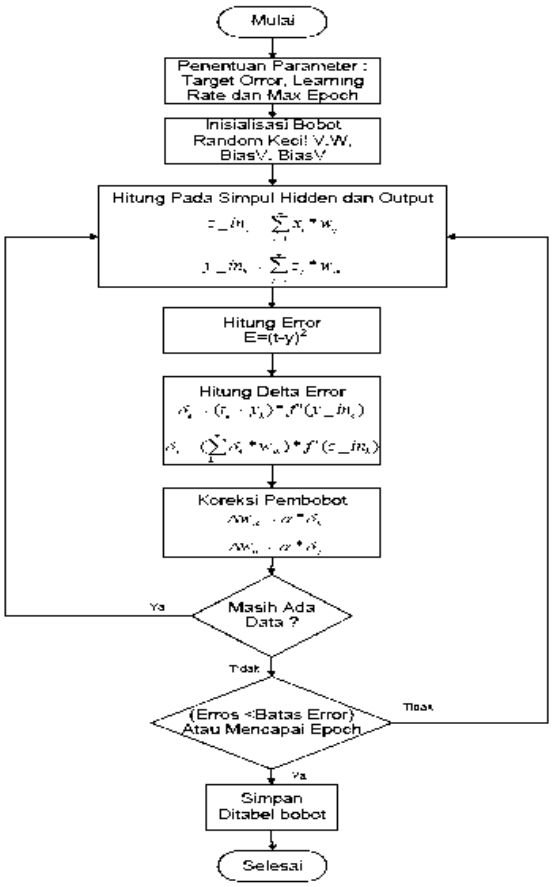
2.9 Algoritma *Backpropagation*

Algoritma *backpropagation* yang biasa disebut juga dengan algoritma propagasi balik merupakan salah satu model dari jaringan syaraf tiruan. Salah satu fungsi dari *backpropagation* adalah untuk melatih keseimbangan antara kemampuan jaringan untuk mengenali pola-pola yang digunakan dalam pelatihan. Selain itu fungsi lainnya adalah untuk melatih kemampuan sebuah jaringan dalam pemberian sebuah respon yang benar terhadap pola masukan dengan pola yang dipakai selama pelatihan. Algoritma *backpropagation* telah banyak di aplikasikan dalam segala bidang, antara lain untuk pediksi harga saham, pengenalan emosi

lewat suara, pendeteksi penyakit dan masih banyak lagi penerapan algoritma *backpropagation* ini.

Backpropagation digunakan oleh perceptron dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron-neuron yang terdapat pada lapisan tersembunyi. Algoritma propagasi balik menggunakan *error output* agar dapat mengubah nilai bobot – bobotnya dalam proses mundur *(backward)* untuk mendapatkan error ini, tahap perambatan maju, neuron – neuron diaktifkan dengan menggunakan fungsi aktivasi sigmoid, sedangkan dalam melakukan sebuah proses perambatan maju dan mundur disebut *epoch* atau iterasi. [12]

Berikut merupakan tahapan pada proses pelatihan algoritma *backpropagation* :



Gambar 2.5 *Flowchart* Proses Pelatihan *Backpropagation*

2.10 Algoritma *Levenberg-Marquardt*

Algoritma *Levenberg-Marquardt* pengembangan algoritma *backpropagation* standar yang secara umum prosesnya masih seperti algoritma *backpropagation*. Bila pada algoritma *backpropagation*, proses *update* bobot dan bias menggunakan *negative gradient descent* secara langsung, algoritma *Levenberg-Marquardt* menggunakan pendekatan matriks Hesian (H) yang dapat dihitung dengan persamaan 2.11 :

$$\boldsymbol{H = J^T J} \qquad (2.11)$$

sedangkan gradient dapat dihitung dengan persamaan 2.12 :

$$\boldsymbol{g = J^T e} \qquad (2.12)$$

J merupakan sebuah matrik jacobian yang berisikan turunan pertama dari *error* jaringan terhadap bobot dan bias jaringan. Matriks Jacobian dapat dihitung dengan persamaan 2.13 [12] :

$$\boldsymbol{J = \left[\frac{\partial e}{\partial w}\right]} \qquad (2.13)$$

Perubahan pembobot dapat dihitung dengan persamaan 2.14 [13] :

$$\boldsymbol{\Delta W = [J^T J + \mu I]^{-} J^T e} \qquad (2.14)$$

di mana μ merupakan konstanta *learning*, *I* merupakan matrik identitas dan *e* merupakan vektor yang menyatakan semua *error* pada *output* jaringan

Berikut tahapan pada proses pelatihan algoritma *Levenberg-Marquardt* :

Langkah 1 : Inisialisasi bobot dan parameter.

Langkah 2 : Hitung *input* setiap *node* pada *hidden layer*

$$\boldsymbol{z_{-in_j} = \sum_{i=1}^{n} x_i \times w_{ij}} \qquad (2.15)$$

di mana :

z_inj : jumlah total *input* untuk *node* ke-j

n : jumlah *node* pada *hidden layer* sebelumnya, untuk *hidden layer* yang pertama n adalah jumlah *node input*

xi : nilai *node* ke-i yang dikeluarkan oleh *hidden layer*

sebelumnya, untuk *hidden layer* yang pertama xi adalah *input* yang diterima oleh sistem

wij : bobot yang menghubungkan antara *node* ke-i dan *node* ke-j

Hitung *output* setiap *node* pada *hidden layer* dengan fungsi aktivasi

$$\boldsymbol{z_j = f(z_in_j)} \qquad (2.16)$$

di mana :

zj : *output* dari node ke-j

$$\boldsymbol{f(x) = \frac{1}{1+\exp(-x)}} \qquad (2.17)$$

Langkah 3 : Hitung *input* setiap *node* pada *output layer*

$$\boldsymbol{y_{_in_k} = \sum_{j=1}^{n} z_j \times w_{jk}} \qquad (2.18)$$

dimana :

y_ink : jumlah total input untuk *node* ke-k

n : jumlah node pada *hidden layer* sebelum *output layer*

wjk : bobot yang menghubungkan antara *node* ke-j dan *node* ke-k

Hitung output pada setiap *node* pada *output layer*

$$\boldsymbol{y_k = f(y_in_k)} \qquad (2.19)$$

dimana :

yk : *output* dari *node* ke-k

Langkah 4 : Hitung *error* setiap *node* pada *output layer*

$$\boldsymbol{e = [t_k - y_k]^T} \qquad (2.20)$$

dimana :

k : banyaknya unit pada output layer

Langkah 5 : Hitung matriks *Jacobian* dengan persamaan 2.13

Langkah 6 : Hitung selisih bobot dengan persamaan 2.14

Langkah 7 : *Update error* dengan persamaan :

$$\boldsymbol{e(new) = \frac{1}{2} e^T e} \qquad (2.21)$$

Langkah 8 : *Update* bobot pada setiap *node* pada *output layer* dan *hidden layer* dengan persamaan :

$$\boldsymbol{w_{jk}(new) = w_{jk}(old) + \Delta w_{jk}} \quad (2.22)$$

$$\boldsymbol{w_{ij}(new) = w_{ij}(old) + \Delta w_{ij}} \quad (2.23)$$

dimana Δw_{jk} dan Δw_{ij} berasal dari matriks ΔW

Jika iterasi masih berlanjut (*epoch* < *max epoch* dan *error* > *max error*), terdapat 2 kemungkinan [14] :

jika *e* naik maka *reduce* dengan $\boldsymbol{\mu(new) = \frac{\mu(old)}{\beta}}$ (2.24),

jika *e* turun maka *increase* dengan $\boldsymbol{\mu(new) = \mu(old) \times \beta}$ (2.25),

lalu kembali ke langkah 2.