BAB II

LANDASAN TEORI

2.1 Pengenalan Teori

Proses utama dari pengenalan karakter adalah menerima karakter *input* dan memeriksa apakah hasil *input* tersebut sesuai dengan salah satu karakter yang ada. Bagian yang paling penting dari seluruh proses adalah mengenali karakter dengan ciri-ciri tertentu.

Ciri-ciri dari sebuah karakter harus mengandung informasi yang diperlukan untuk membedakan karakter tersebut dan harus membedakan penyimpanan variabel ketikan *input*, untuk membatasi jumlah dari pembelajaran data yang diberikan.

2.2 Pengenalan Karakter Huruf Arab

Huruf Arab adalah huruf-huruf yang digunakan dalam Al-Quran. Huruf Arab jumlahnya ada 29, yaitu :

Tabel 2.1 Daftar Huruf Arab

Huruf Latin	Huruf Arab	Huruf Latin	Huruf Arab	Huruf Latin	Huruf Arab
Qaf(Qof)	ق	Zai	ز	Alif	ا
Kaf	ك	Sin	س	Ba	ب
Lam	ل	Syin	ش	Ta	ت
Mim	م	Shad(Shod)	ص	Tsa	ث
Nun	ن	Dhad(Dhod)	ض	Jim	ج

Wau	و	Tha(Tho)	ط	Ha(tipis)	ح
Ha(tebal)	ه	Zha(Zho)	ظ	Kha(Kho)	خ
Hamzah	ء	‘Ain	ع	Dal	د
Ya	ي	Ghain	غ	Dzal	ذ
		Fa	ف	Ra(Ro)	ر

2.3 Teori Dasar Pengolahan Citra

2.3.1 Citra

Definisi citra menurut kamus Webster adalah suatu representasi, kemiripan, atau imitasi dari suatu obyek atau benda. Citra dapat dikelompokan menjadi citra tampak dan citra tak tampak. Contoh citra tampak adalah foto diri kita sendiri, sedangkan contoh citra tak tampak adalah data gambar dalam file (citra digital) dan citra yang direpresentasikan dalam sebuah fungsi matematis.

Diantara jenis-jenis citra tersebut, hanya citra digital yang dapat diolah menggunakan komputer.

Citra merupakan suatu fungsi kontinyu dari intensitas cahaya dalam bidang dua dimensi, dengan (x,y) menyatakan koodinat citra dan nilai f pada koodinat (x,y) menyatakan tingkat kecerahan atau derajat keabuan.

Citra digital merupakan array dua dimensi dengan nilai f(x,y) nya telah dikonversi ke dalam bentuk diskrit baik pada koordinat citra maupun kecerahannya. Pengolahan citra secara umum dapat didefinisikan sebagai

pemrosesan sebuah gambar dua dimensi secara digital. Pengolahan citra merupakan proses pengolahan dan analisis citra yang banyak melibatkan presepsi visual. Proses ini mempunyai data masukan dan informasi keluaran yang berbentuk citra.

Operasi pengolahan citra digital umumnya dilakukan dengan tujuan memperbaiki kualitas suatu gambar sehingga dapat dengan mudah diinterpretasikan oleh mata manusia dan untuk mengolah informasi yang terdapat pada suatu gambar untuk keperluan pengenalan objek secara otomatis.

2.3.1.1 Operasi Pengolahan Citra

Pengolahan citra pada dasarnya dilakukan dengan cara memodifkasi setiap titik dalam citra tersebut sesuai dengan keperluan. Secara garis besar, modifikasi tersebut dikelompokan menjadi:

1. Operasi titik, dimana setiap titik diolah secara tidak gayut terhadap titik yang lain.
2. Operasi global, dimana karakteristik global dari citra digunakan untuk memodifikasi nilai setiap titik.
3. Operasi berbasis bingkai, dimana setiap citra diolah dengan dikombinasikan dengan citra lain.
4. Operasi Geometri, dimana bentuk ukuran, atau orientasi citra dimodifikasi secara geometris.
5. Operasi banyak titik bertetangga, dimana data dari titik-titik yang bersebelahan dengan titik yang ditinjau ikut berperan dalam mngubah nilai.

6. Operasi morfologi, yaitu operasi yang berdasarkan segmen atau bagian dalam citra yang menjadi perhatian.

2.3.1.2 Komponen Citra Digital

Secara umum setiap citra digital memiliki beberapa karakteristik atau komponen, atara lain

1. Ukuran citra
2. Resolusi
3. Format citra

2.3.1.3 Representasi Citra Digital

Komputer dapat mengolah isyarat-isyarat elektronik digital yang merupakan kumpulan sinyal biner. Untuk itu, citra digital harus mempunyai format tertentu yang sesuai sehingga dapat merepresentasikan obyek pencitraan dalam bentuk kombinasi data biner.

Format citra digital yang masih banyak dipakai adalah

1. Citra biner
2. Skala Keabuan
3. Warna
4. Warna berindeks

2.3.1.4 Citra Biner

Pada citra biner, setiap nilai bernilai 0 atau 1, maing-masing merepresentasikan warna tertentu. Contoh yang paling lazim adalah hitam bernilai 0, dan putih bernilai 1.

2.3.1.5 Citra Skala Keabuan

Disebut skala keabuan karena pada umumnya warna yang dipakai adalah antara hitam sebagai warna minimal dan putih sebagai warna maksimal, sehingga warna diatara keduannya adalah abu-abu.

Namun dalam prakteknya warna yang dipakai tidak terbatas pada warna abu-abu, sebagai contoh dipilih warna minimalnya adalah putih dan warna makimalnya adalah merah, maka semakin besar nilainya semakin besar pula intensitas warna merahnya.

2.3.1.6 Citra Warna

Pada citra warna, setiap titik mempunyai warna yang spesifik yang merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau dan biru. Format citra ini sering disebut citra RGB. Setiap warna dasar mempunyai intensitas sendiri dengan nilai maksimum 255 (8 bit).

Jumlah kombnasi warna yang mungkin untuk format citra ini adalah 2^{24} atau lebih dari 16 juta warna, dengan demikian bisa dianggap mencakup semua warna yang ada, hal inilah format ini dinamakan *true color*.

2.3.1.7 Citra Warna Berindeks

Jumlah memori yang dibutuhkan untuk format citra warna *true color* adalah tiga kali jumlah titik yang ada dalam citra yang ditinjau. Di lain pihak, jumlah warna yang ada dalam suatu citra terkadang sangat terbatas, karena banyaknya warna dalam sebuah citra tidak mungkin melebihi banyaknya titik dalam citra itu sendiri. Dengan kasus seperti ini, disediakan format citra warna berindeks. Pada format ini, informasi setiap titik merupakan indeks dari suatu tabel yang berisi informasi warna yang tersedia, yang disebut palet warna atau *color map.*

2.3.2 Histogram Jumlah dan Selisih

Tekstur, dalam citra digital, dinyatakan sebagai hubungan spasial atau penyusunan spasial dari piksel-piksel citra. Secara visual hubungan spasial atau penyusunan spasial piksel-piksel citra tersebut dapat dilihat sebagai perubahan pola intensitas (*gray tones*).

Tekstur dapat digunakan untuk mendeJurnalkan *array* secara dua dimensi yang merupakan variasi-variasi dari e lemen-elemen tekstur. T ekstur juga mendeJurnalkan aturan–aturan penempatan maupun pengaturan jarak antar elemen tekstur tersebut yang dapat acak maupun mempunyai pola berulang.

Salah satu aspek yang harus diperhatikan dari sebuah tekstur adalah distribusi spasial dan ketergantungan secara spasial antar derajat keabuan pada suatu daerah lokal. Distribusi dan ketergantungan secara spasial tersebut dapat ditinjau dengan pendekatan secara statistik, yaitu dengan menggunakan

pendekatan *co-occurrence* Pendekatan *co-occurrence* berdasarkan pada distribusi probabilitas hubungan antar piksel bertetangga. Pendekatan *co-occurance* ini dapat diimplementasikan dengan membentuk matriks kookuransi derajat keabua yang kemudian dikembangkan menjadi histogram jumlah dan selisih oleh Unser pada tahun 1986.

Histogram jumlah "**S**" dibentuk sebagai hasil jumlah derajat keabuan dua piksel yang terpisah pada jarak dx dan dy. Histogram jumlah menyatakan banyaknya kemunculan (frekuensi) hasil jumlah tingkat keabuan dua piksel bertetangga pada jarak dan sudut tertentu.

Histogram selisih "**D**" dibentuk sebagai hasil selisih derajat keabuan antara piksel yang terpisah pada jarak dx dan dy. Histogram selisih menyatakan banyaknya kemunculan (frekuensi) hasil selisih tingkat keabuan dua piksel bertetangga pada jarak dan sudut tertentu, sehingga histogram yang dihasilkan akan memiliki selang nilai antara -255 sampai +255 untuk citra 8 bit.

Kedua histogram tersebut kemudian dinormalisasi dengan cara membagi setiap elemen dengan total jumlah semua elemen *array* pada maing-masing histogram sehingga dihasilkan suatu probabilitas P_S dan P_d. P_S adalah histogram jumlah yang ternormalisai, sedangkan P_d adalah histogram selisih yang sudah ternormalisasi. Dari kedua histogram ini kemudian dapat diturunkan 8 macam *descriptor* fitur yang dapat merepresentasikan citra digital 2 dimensi kedalam bentuk vector.

2.4 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan (*artificial neural network*) atau disingkat JST adalah sistem komputasi dimana arsitektur dan operasi diilhami dari pengetahuan tentang sel syaraf biologi didalam otak. JST dapat digambarkan sebagai model matematis dan komputasi untuk fungsi aproksimasi nonlinear, klasifikasi data, cluster dan regresi non parametric atau sebagai sebuah simulasi dari koleksi model syaraf biologi.

JST menyerupai otak manusia dalam 2 (dua) hal, yaitu :

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

2.4.1 Arsitektur (Model) JST

Dalam JST ada beberapa model yang digunakan, antara lain :

2.4.1.1 Jaringan dengan lapisan tunggal (*Single Layer Net*)

Jaringan dengan lapisan tunggal hanya memiliki satu lapisan dengan bobot-bobot terhubung. Jaringan ini hanya menerima *input* kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi.

2.4.1.2 Jaringan dengan banyak lapisan (*Multilayer Net*)

Dalam model ini sebenarnya merupakan model JST satu lapisan yang jumlahnya banyak dan prinsip kerja dari model JST ini sama dengan JST satu

lapisan. *Output* dari tiap lapisan sebelumnya dalam model ini merupakan *input* dari lapisan sebelumnya.

2.4.1.3 Jaringan dua lapisan dengan umpan balik

Dengan jumlah *neuron* yang sangat besar, JST memiliki sifat yaitu *fault tolerance*. Sifat ini mengandung maksud kerusakan sedikit atau sebagian pada sel-sel dalam jaringan tidak akan mempengaruhi *output* yang akan dikeluarkan.

Jaringan dua lapisan ini mempunyai sifat umpan balik, sehingga *output* yang dihasilkan akan mempengaruhi *input* yang akan masuk lagi kedalam jaringan syaraf tersebut.

2.5 Komponen JST

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

a) Neuron Tiruan (*Artificial Neuron*)

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

b) Lapisan (*layer*)

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

c) 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 kedalam data numerik yang ekuivalen sebelum dapat diproses oleh JST.

d) Keluaran (*Output*)

Keluaran dari JST adalah pemecahan terhadap masalah.

Data keluaran merupakan data numerik.

e) Bobot (*Weight*)

Bobot pada JST menyatakan tingkat kepintaran sistem. Walaupun sebenarnya bobot tersebut hanya sebuah deretan angka-angka saja. Bobot sangat penting untuk JST, dimana bobot yang optimal akan memungkinkan sistem menerjemahkan data masukan secara benar dan

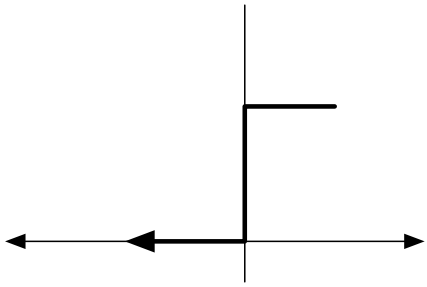
2.6 Fungsi Aktivasi JST

1. Fungsi Undak Biner

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

Fungsi undak biner dirumuskan sebagai berikut :

$$y = \begin{cases} 0, \; jikax < 0 \\ 1, \; jikax \geq 1 \end{cases} \qquad (2.1)$$



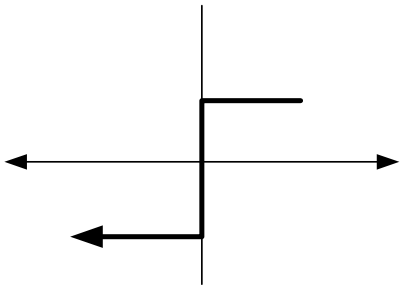
Gambar 2.1 Fungsi aktifasi Undak Biner

2. Fungsi Bipolar

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

Fungsi bipolar dirumuskan sebagai berikut :

$$y = \begin{cases} 1, \; jikax \geq 0 \\ -1, \; jikax < 0 \end{cases} \qquad (2.2)$$

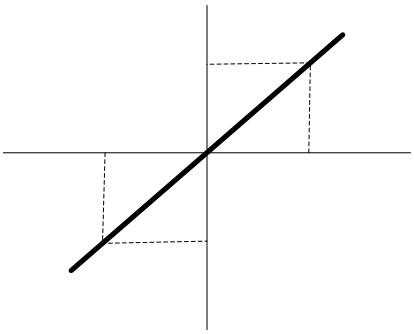


Gambar 2.2 Fungsi aktifasi Bipolar

3. Fungsi Linear

Fungsi linear mempunyai *output* yang sama dengan nilai *input*nya. Fungsi linear dirumuskan sebagai berikut :

$$y = x \qquad (2.3)$$



Gambar 2.3 Fungsi aktifasi Linear

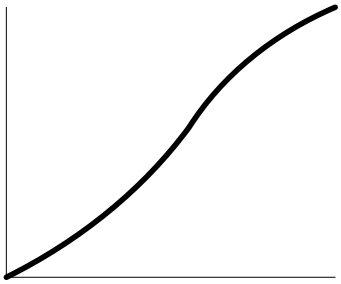
4. Fungsi Sigmoid Biner

Fungsi ini digunakan untuk JST yang dilatih dengan menggunakan metode backpropagation. Fungsi sigmoid biner mempunyai nilai antara 0 sampai 1. Oleh karena itu, fungsi ini sering digunakan untuk JST yang membutuhkan nilai *output* yang terletak pada interval 0 sampai 1.

Fungsi sigmoid biner dirumuskan sebagai berikut:

$$y = f(x) = \frac{1}{1 + e^{-\sigma x}} \tag{2.4}$$

dengan : $f'(x) = \omega f(x)[1 - f(x)]$ (2.5)



Gambar 2.4 Fungsi aktifasi Sigmoid Biner

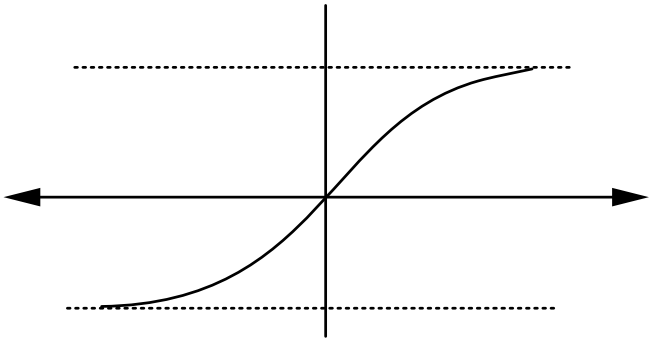
5. Fungsi Sigmoid Bipolar

Fungsi sigmoid bipolar hampir sama dengan fungsi sigmoid biner, hanya saja *output* dari fungsi ini memiliki range antara 1 sampai -1.

Fungsi sigmoid bipolar dirumuskan sebagai berikut :

$$y - f(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{2.6}$$

dengan: $f'(x) = \frac{\sigma}{2}[1 + f(x)][1 - f(x)]$ (2.7)



Gambar 2.5 Fungsi aktifasi Sigmoid Bipolar

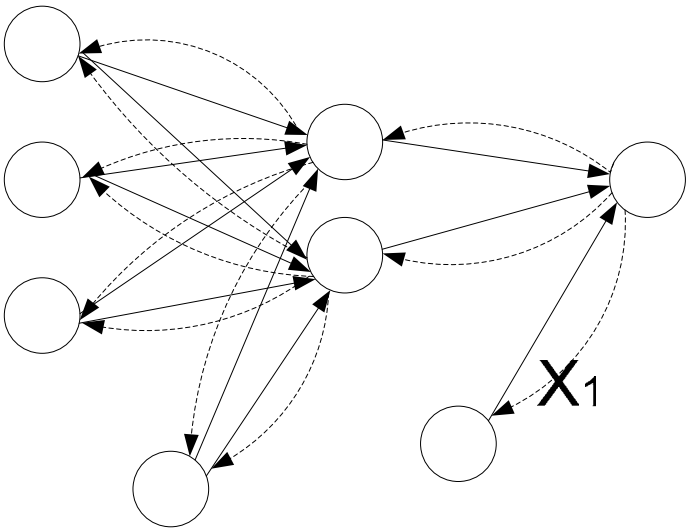
2.7 Laju Pembelajaran (*Learning rate)*

Laju pembelajaran berfungsi untuk memperkecil perubahan bobot. Jika nilai laju pembelajaran dimulai dengan nilai yang besar maka banyak perhitungan terhadap nilai bobot dan error akan besar. Jika perubahan-perubahan dilakukan secara terus-menerus nilai laju pembelajaran yang besar kemungkinan membuat sistem menjadi lambat, tetapi jika perubahan laju pembelajaran dilakukan dengan perubahan yang kecil maka kecepatan dari sistem memungkinkan lebih cepat.

2.8 Backpropagation

Backpropagatin merupakan algoritma pembelajaran yang terawasi dan biasanya digunakan oleh perceptron dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron-neuron yang ada pada lapisan tersembunyinya. Pelatihan sebuah jaringan yang menggunakan *backpropagation* terdiri dari 3 langkah, yaitu : pelatihan pola *input* secara *feedforward*, perhitungan dan *backpropagation* dari kumpulan kesalahan dan penyesuaian bobot. Sesudah pelatihan, aplikasi dari jaringan hanya terdiri dari fase *feedforward*. Bahkan, jika pelatihan menjadi lambat, sebuah jaringan yang dilatih dapat menghasilkan *output*nya sendiri secara cepat.

Arsitektur jaringan *backpropagation* seperti terlihat dalam gambar 2.6. Pada gambar 2.6 jaringan terdiri atas unit(neuron) pada lapisan *input* yaitu x_1, x_2, x_3, satu lapisan tersembunyi dengan dengan 2 neuron, yaitu z_1 dan z_2, serta 1 lapisan *output* yaitu y. Bobot yang menghubungkan antara lapisan *input* dan lapisan tersembunyi adalah v_{11}, v_{21}, v_{31} (v_{ij} bobot yang mengandung *input* ke-i ke neuron ke-j pada lapisan tersembunyi).



Gambar 2.6 Arsitektur jaringan backpropagation

Algoritma *backpropagation* :

1. Inisialisasikan bobot (ambil bobot awal dengan nilai random yang kecil)
2. Kerjakan langkah-langkah berikut selama kondisi berhenti bernilai FALSE :
   a. Untuk tiap-tiap pasang elemen yang akan dilakukan pembelajaran, kerjakan :

      Feedfoward :

      1. Tiap-tiap unit *input* (X_i , $i = 1,2,3,...,n$) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua unit pada lapisan yang ada di atasnya (lapisan tersembunyi)
      2. Tiap-tiap unit tersembunyi (Z_j , $j = 1,2,3,...,p$) menjumlahkan sinyal-sinyal *input* terbobot :

$$z_in_j = v_{oj} + \sum_{i=1}^{n} x_i v_{ij}$$

gunakan fungsi aktivasi untuk menghitung sinyal *output*nya :

$$z_j = f(z_in_j)$$

dan kirimkan sinyal tersebut ke semua unit di lapisan atasnya (unit-unit *output*).

3. Tiap-tiap unit *output* (Y_k, $k = 1,2,3,...,m$) menjumlahkan sinyal-sinyal *input* terbobot.

$$y_in_k = w_{0k} + \sum_{i=1}^{p} z_i w_{jk}$$

gunakan fungsi aktivasi untuk menghitung sinyal *output*nya :

$$y_k = f(y_in_k)$$

dan kirimkan sinyal tersebut ke semua unit di lapisan atasnya (unit-unit *output*)

Backpropagation

4. Tiap-tiap unit *output* ($Y_k, k = 1,2,3,...,m$) menerima target pola yang berhubungan dengan pola *input* pembelajaran, hitung informasi errornya :

$$\delta_k = (t_k - y_k) f(y_in_k)$$

kemudian hitung koreksi bobot (yang nantinya akan digunakan untuk memperbaikinya nilai w_{jk}) :

$$\Delta w_{jk} = \alpha \delta_k z_j$$

hitung juga koreksi bias (yang nantinya akan digunakan untuk memperbaiki nilai w_{0k}) :

$$\Delta w_{0k} = \alpha \delta_k$$

kirimkan δ_k ini ke unit-unit yang ada di lapisan bawahnya.

5. Tiap-tiap unit tersembunyi ($Z_j, j = 1,2,3,..., p$) menjumlahkan delta *input*nya (dari unit-unit yang berada pada lapisan di atasnya) :

$$\delta_in_j = \sum_{k=1}^{m} \delta_k w_{jk}$$

kalikan nilai ini dengan turunan dari fungsi aktivasinya untuk menghitung informasi error :

$$\delta_j = \delta_in_j f'(z_in_j)$$

kemudian hitung koreksi bobot (yang nantinya akan digunakan untuk memperbaiki nilai v_{ij}) :

$$\Delta v_{jk} = \alpha \delta_j x_i$$

hitung juga koreksi bias (yang nantinya akan digunakan untuk memperbaiki nilai v_{0j}) :

$$\Delta v_{0j} = \alpha \delta_j$$

6. Tiap-tiap unit *output* (Y_k, k=1,2,3,…,m) memperbaiki bias dan bobotnya (j=0,1,2,…,p) :

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk}$$

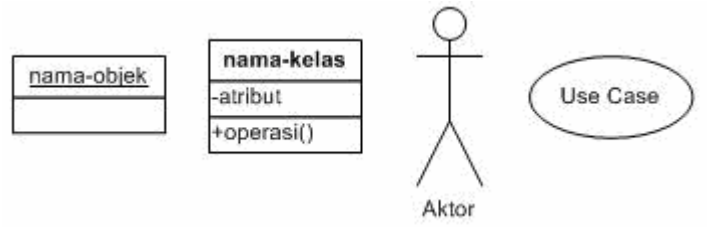
Tiap-tiap unit tersembunyi (Z_j, j=1,2,3,…,p) memperbaiki bias dan bobotnya (i=0,1,2,…,n) :

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij}$$

b. Tes kondisi berhenti.

2.9 UML (*Unified Modeling Language*)

Merupakan notasi-notasi baku untuk melakukan pemodelan visual. Pemodelan visual merupakan proses penggambaran informasi-informasi secara grafis. Notasi-notasi UML terbentuk atas kerjasama dan upaya **Graddy Booch**, **DR. James Rumbaugh**, serta **Ivar Jacobson**.



Gambar 2.7 Beberapa notasi UML

Beberapa notasi UML yang sering digunakan antara lain ditunjukkan pada Gambar 2.7. Aktor merupakan entitas yang berada di luar sistem; bersifat eksternal. Salah satu aktor sistem adalah pengguna (*user*).

Use case berfokus pada perilaku sistem, misalnya *use case* "Pengaktifan Sistem". Kelas merupakan prototipe suatu objek sehingga memiliki atribut (variabel) dan operasi (fungsi atau prosedur).

Dalam UML terdapat beberapa jenis diagram yang dapat digunakan dalam pemodelan sistem, diantaranya adalah:

1. *Activity Diagram*. Bersifat dinamis, memodelkan fungsi-fungsi dalam suatu sistem dan menekankan pada aliran kendali antar objek.
2. *Use Case Diagram*. Bersifat statis, menggambarkan hubungan (*relation*) antara aktor dan *use case*.

3. *Sequence Diagram*. Bersifat dinamis, merupakan diagram interaksi yang menekankan pada pengiriman pesan (*message*) secara berurutan berdasarkan waktu.
4. *Class Diagram*. Bersifat statis, menggambarkan himpunan kelas-kelas, antarmuka dan relasi antar kelas.