BAB 2
LANDASAN TEORI

2.1.Pola

Pola adalah suatu entitas yang terdefinisi dan dapat diidentifikasi serta diberi nama melalui ciri-cirinya (feature). Ciri-ciri tersebut digunakan untuk membedakan suatu pola dengan pola yang lainnya. Ciri yang baik adalah ciri yang memiliki daya pembeda yang tinggi, sehingga pengelompokan pola berdasarkan ciri yang dimiliki dapat dilakukan dengan keakuratan yang tinggi.

Pola adalah komposit/gabungan dari ciri yang merupakan sifat dari sebuah objek [5].

Beberapa contoh pola :

1. Huruf, memiliki ciri-ciri seperti tinggi, tebal, titik sudut, dan lengkungan garis.
2. Suara, memiliki ciri-ciri seperti amplitudo, frekuensi, nada, dan intonasi.
3. Tanda tangan, memiliki ciri-ciri seperti panjang, kerumitan, dan tekanan.
4. Sidik jari, memiliki ciri-ciri seperti lengkungan, dan jumlah garis.

Ciri pada suatu pola diperoleh dari hasil pengukuran terhadap objek uji. Khusus pada pola yang terdapat di dalam citra, ciri-ciri yang dapat diperoleh berasal dari informasi [5]:

1. Spasial: intensitas pixel, histogram, …
2. Tepi: arah, kekuatan, …
3. Kontur: garis, elips, lingkaran, …
4. Wilayah/bentuk: keliling, luas, pusat massa, …
5. Hasil transformasi Fourier: frekuensi, …

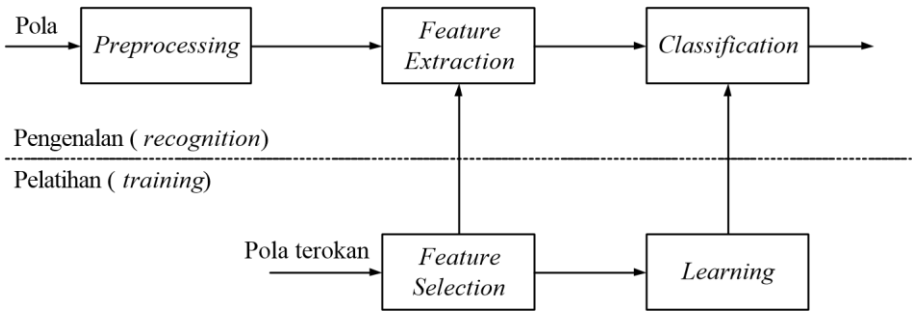
2.1.1. Pengenalan Pola

Pengenalan Pola adalah mengelompokkan data numerik dan simbolik (termasuk citra) secara otomatis oleh mesin (dalam hal ini komputer). Pengenalan pola bertujuan menentukan kelompok atau kategori pola berdasarkan ciri-ciri yang dimiliki oleh pola tersebut. Dengan kata lain, pengenalan pola membedakan suatu objek dengan objek lain. Terdapat dua pendekatan yang dilakukan dalam pengenalan pola: pendekatan secara statistik dan pendekatan secara sintaktik atau struktural [6].

a. Pengenalan Pola secara Statistik

Pendekatan ini menggunakan teori-teori ilmu peluang dan statistik. Ciri-ciri yang dimiliki oleh suatu pola ditentukan distribusi statistiknya. Pola yang berbeda memiliki distribusi yang berbeda pula. Dengan menggunakan teori keputusan di dalam statistik, kita menggunakan distribusi ciri untuk mengklasifikasikan pola.

Sistem pengenalan pola dengan pendekatan statistik ditunjukkkan oleh diagram pada Gambar (2.1).

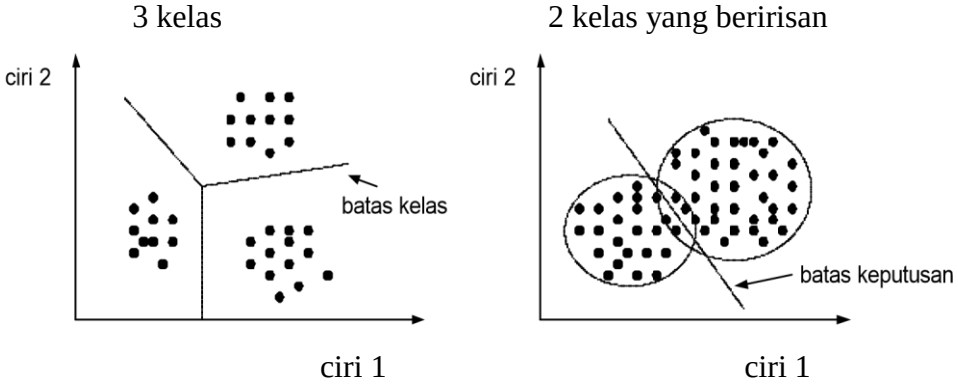


Gambar 2. 1 Sistem pengenalan pola dengan pendekatan statistik.

Ada dua fase dalam sistem pengenalan pola: (i) fase pelatihan dan (ii) fase pengenalan. Pada fase pelatihan, beberapa contoh citra dipelajari untuk menentukan ciri yang akan digunakan dalam proses pengenalan serta prosedur klasifikasinya. Pada fase pengenalan, citra diambil cirinya kemudian ditentukan kelas kelompoknya [5].

Kumpulan ciri dari suatu pola dinyatakan sebagai vektor ciri dalam ruang bahumatra (multi dimensi). Jadi, setiap pola dinyatakan sebagai

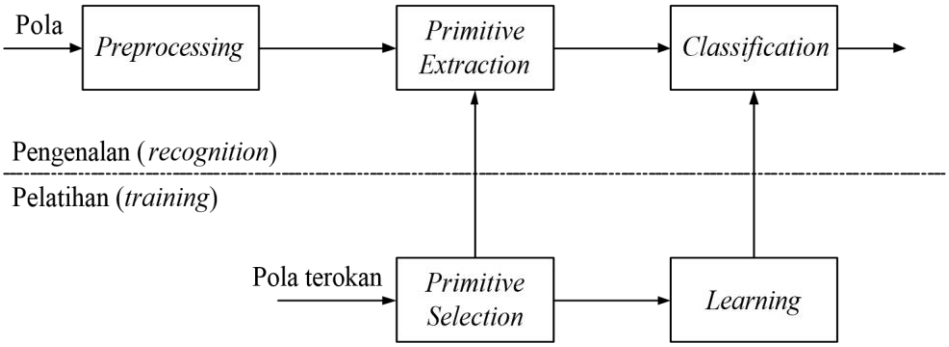
sebuah titik dalam ruang bahumatra. Ruang bahumatra dibagi menjadi sejumlah uparuang (sub-ruang). Tiap uparuang dibentuk berdasarkan pola-pola yang sudah dikenali kategori dan ciri-cirinya (melalui fase pelatihan) [5]. Lihat Gambar (2.2).



Gambar 2. 2 Contoh pembagian kelas pola

b. Pengenalan Pola secara Sintaktik

Pendekatan ini menggunakan teori bahasa formal. Ciri-ciri yang terdapat pada suatu pola ditentukan primitif dan hubungan struktural antara primitif kemudian menyusun tata bahasanya. Dari aturan produksi pada tata bahasa tersebut kita dapat menentukan kelompok pola. Gambar (2.3) memperlihatkan sistem pengenalan pola dengan pendekatan sintaktik. Pengenalan pola secara sintaktik lebih dekat ke strategi pengenalan pola yang dilakukan manusia, namun secara praktek penerapannya relatif sulit dibandingkan pengenalan pola secara statistik [5].



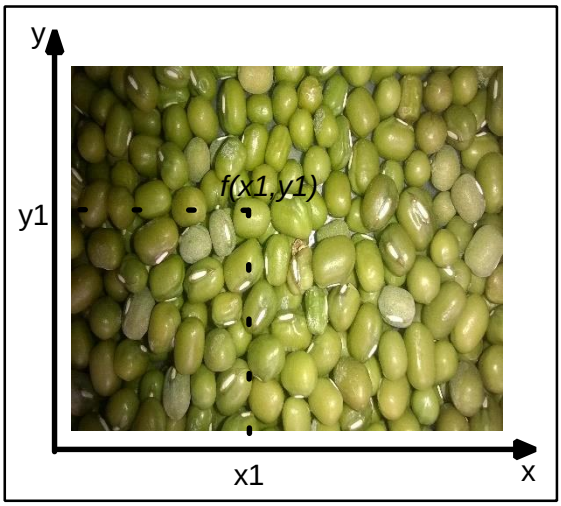
Gambar 2. 3 Sistem pengenalan pola dengan pendekatan sintaktik.

Pendekatan yang digunakan dalam membentuk tata bahasa untuk mengenali pola adalah mengikuti kontur (tepi batas) objek dengan sejumlah segmen garis terhubung satu sama lain, lalu mengkodekan setiap garis tersebut (misalnya dengan kode rantai). Setiap segmen garis merepresentasikan primitif pembentuk objek [5].

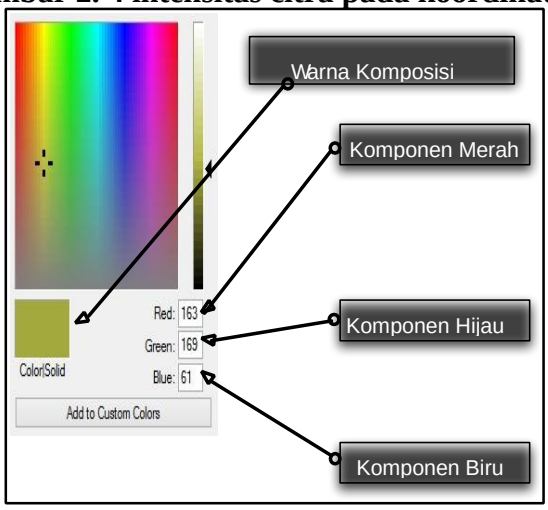
2.2.Citra digital

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 [5].

Citra digital dapat didefinisikan sebagai fungsi dua variabel, f(x,y), dimana x dan y adalah koordinat spasial dan nilai f(x,y) adalah intensitas citra pada koordinat tersebut, hal tersebut diilustrasikan pada gambar(2.4) Teknologi dasar untuk menciptakan dan menampilkan warna pada citra digital berdasarkan pada penelitian bahwa sebuah warna merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau, dan biru (Red, Green, Blue - RGB). Komposisi warna RGB tersebut dapat dijelaskan pada gambar(2.5).



Gambar 2. 4 intensitas citra pada koordinat f(x,y)



Gambar 2. 5 Komposisi warna RGB pada gambar(2.4)

2.2.1. Pengolahan Citra

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.

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 [5].

2.2.1.1.Operasi-operasi pengolahan citra

Operasi-operasi yang dilakukan di dalam pengolahan citra banyak ragamnya. Namun, secara umum, operasi pengolahan citra dapat diklasifikasikan dalam beberapa jenis sebagai berikut [5]:

1. Perbaikan kualitas citra (image enhancement).

   Jenis operasi ini bertujuan untuk memperbaiki kualitas citra dengan cara memanipulasi parameter-parameter citra. Dengan operasi ini, ciri-ciri khusus yang terdapat di dalam citra lebih ditonjolkan. Contoh-contoh operasi perbaikan citra:

   a. perbaikan kontras gelap/terang
   b. perbaikan tepian objek (edge enhancement)
   c. penajaman (sharpening)
   d. pembrian warna semu (pseudocoloring)
   e. penapisan derau (noise filtering)

2. Pemugaran citra (image restoration).

   Operasi ini bertujuan menghilangkan/meminimumkan cacat pada citra. Tujuan pemugaran citra hampir sama dengan operasi perbaikan citra. Bedanya, pada pemugaran citra penyebab degradasi gambar diketahui. Contoh-contoh operasi pemugaran citra:

   a. penghilangan kesamaran (deblurring).
   b. penghilangan derau (noise)

3. Pemampatan citra (image compression).

   Jenis operasi ini dilakukan agar citra dapat direpresentasikan dalam bentuk yang lebih kompak sehingga memerlukan memori yang lebih sedikit. Hal

penting yang harus diperhatikan dalam pemampatan adalah citra yang telah dimampatkan harus tetap mempunyai kualitas gambar yang bagus.

4. Segmentasi citra (image segmentation).

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

5. Pengorakan citra (image analysis)

   Jenis operasi ini bertujuan menghitung besaran kuantitif dari citra untuk menghasilkan deskripsinya. Teknik pengorakan citra mengekstraksi ciri-ciri tertentu yang membantu dalam identifikasi objek. Proses segmentasi kadangkala diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya. Contoh-contoh operasi pengorakan citra:

   b. Pendeteksian tepi objek (edge detection)
   c. Ekstraksi batas (boundary)
   d. Representasi daerah (region)

6. Rekonstruksi citra (image reconstruction)

   Jenis operasi ini bertujuan untuk membentuk ulang objek dari beberapa citra hasil proyeksi. Operasi rekonstruksi citra banyak digunakan dalam bidang medis. Misalnya beberapa foto rontgen dengan sinar X digunakan untuk membentuk ulang gambar organ tubuh.

7. Perubahan Model Warna

   Warna adalah persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh objek [7]. Setiap warna mempunyai panjang gelombang yang berbeda. Warna merah mempunyai panjang gelombang paling tinggi, sedangkan warna ungu mempunyai panjang gelombang paling rendah. Warna-warna yang diterima oleh mata merupakan hasil kombinasi cahaya dengan panjang gelombang berbeda. Penelitian memperlihatkan bahwa kombinasi warna yang memberikan rentang warna yang paling lebar adalah red (R), green (G), blue (B).

   Dalam pembentukan citra digital model warna yang umum digunakan antara lain sebagai berikut :

a. Citra RGB

Citra RGB yang biasa disebut juga citra true color, disimpan dalam citra berukuran (m x n) x 3 yang mendefinisikan warna merah (red), hijau (green) dan biru (blue) untuk setiap pikselnya. Warna pada setiap piksel ditentukan berdasarkan kombinasi dari warna red, green dan blue (RGB). RGB merupakan citra 24 bit dengan komponen merah, hijau, dan biru yang masing-masing umumnya bernilai 8 bit sehingga intensitas kecerahan warna sampai 256 level dan kombinasi warnanya kurang dari sekitar 16 juta warna.

b. Citra Keabuan

Citra dengan derajat keabuan berbeda dengan citra RGB, citra ini didefinisikan oleh satu nilai derajat warna. Umumnya bernilai 8 bit sehingga intensitas kecerahan warna sampai 256 level dan kombinasi warnanya 256 varian. Tingkat kecerahan paling rendah yaitu 0 untuk warna hitam dan putih bernilai 255. Untuk mengkonversikan citra yang memiliki warna RGB ke derajat keabuan bisa menggunakan (persamaan 1) :

$$X = 0.21 \cdot R + 0.72 \cdot G + 0.07 \cdot B \qquad \text{(persamaan 1)}$$

Dimana :

X = Citra Greyscale

R = Red

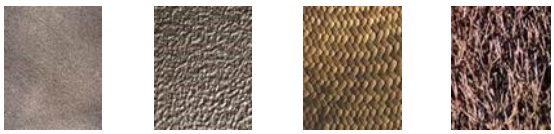
G = Green

B = Blue

2.3.Tekstur

Secara umum tekstur mengacu pada repetisi elemen-elemen tekstur dasar yang sering disebut primitif atau *texel* (*texture element*). Suatu *texel* terdiri dari beberapa *pixel* dengan aturan posisi bersifat periodik, kuasiperiodik, atau acak [7]. Syarat-syarat terbentuknya tekstur setidaknya ada dua, yaitu :

1. Adanya pola-pola primitif yang terdiri dari satu atau lebih pixel. Bentuk-bentuk pola primitif ini dapat berupa titik, garis lurus, garis lengkung, luasan dan lain-lain yang merupakan elemen dasar dari sebuah bentuk.
2. Pola-pola primitif tadi muncul berulang-ulang dengan interval jarak dan arah tertentu sehingga dapat diprediksi atau ditemukan karakteristik pengulangannya.



Gambar 2. 6 Contoh tekstur dari VisTex Database

Suatu citra memberikan interpretasi tekstur yang berbeda apabila dilihat dengan jarak dan sudut yang berbeda. Manusia memandang tekstur berdasarkan deskripsi yang bersifat acak, seperti halus, kasar, teratur, tidak teratur, dan sebagainya. Hal ini merupakan deskripsi yang tidak tepat dan non-kuantitatif, sehingga diperlukan adanya suatu deskripsi yang kuantitatif (matematis) untuk memudahkan analisis [7].

2.3.1. Analisis Tekstur

Analisis tekstur merupakan dasar dari berbagai macam aplikasi, aplikasi dari analisis tekstur antara lain: penginderaan jarak jauh, pencitraan medis, identifikasi kualitas suatu bahan (kayu, kulit, tekstil, dan lain-lain), dan juga berbagai macam aplikasi lainnya. Pada analisis citra, pengukuran tekstur dikategorikan menjadi lima kategori utama yaitu: statistis, struktural, geometri, model dasar, dan pengolahan sinyal. Pendekatan statistis mempertimbangkan bahwa intensitas dibangkitkan oleh medan acak dua dimensi, metode ini berdasar pada frekuensi frekuensi ruang. Contoh metode statistis adalah fungsi autokorelasi, ***Cooccurence Matrix***, transformasi Fourier, frekuensi tepi, run length dan lainya. Teknik struktural berkaitan dengan penyusunan bagian bagian terkecil suatu citra. Contoh metode struktural adalah model fraktal. Metode geometri berdasar atas perangkat geometri yang ada pada elemen tekstur. Contoh metode model dasar

adalah medan acak. Sedangkan metode pengolahan sinyal adalah metode yang berdasarkan analisis frekuensi seperti transformasi Gabor dan transformasi wavelet [7].

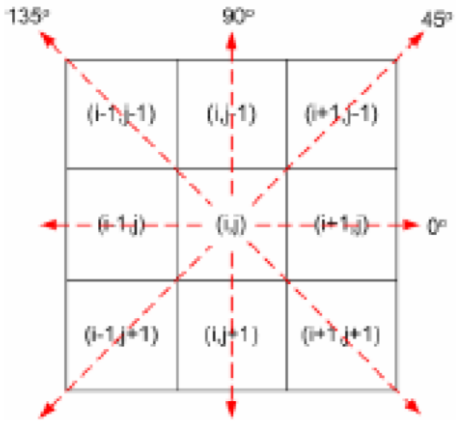
2.3.1.1.Analisis Tekstur Dengan Pendekatan Statistik

Analisis tekstur 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. Metode Statistik ini penggunaannya tidak terbatas untuk tekstur-tekstur alami yang tidak terstruktur dari sub pola dan himpunan aturan (*mikrostruktur*). Berdasarkan orde statistiknya, analisis tekstur dapat dikategorikan menjadi 3, yaitu analisis tekstur orde satu, orde dua, dan orde tiga.

2.3.1.1.1. Metode Cooccurence Matrix **[8]**

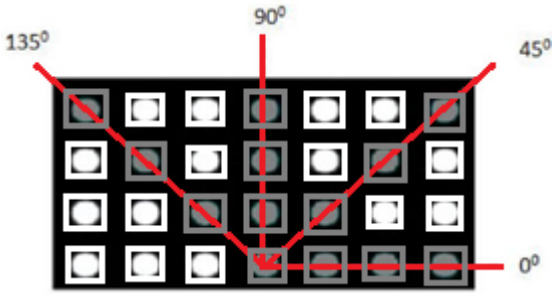
Metode Cooccurence Matrix adalah suatu matriks yang elemen-elemennya merupakan jumlah pasangan piksel yang memiliki tingkat kecerahan tertentu, di mana pasangan piksel itu terpisah dengan jarak *d*, dan dengan suatu sudut inklinasi *θ*. Dengan kata lain, matriks kookurensi adalah probabilitas munculnya graylevel *i* dan *j* dari dua piksel yang terpisah pada jarak *d* dan sudut *θ*. [7]

Suatu piksel yang bertetangga yang memiliki jarak *d* diantara keduanya, dapat terletak di delapan arah yang berlainan, hal ini ditunjukkan pada Gambar(2.7).

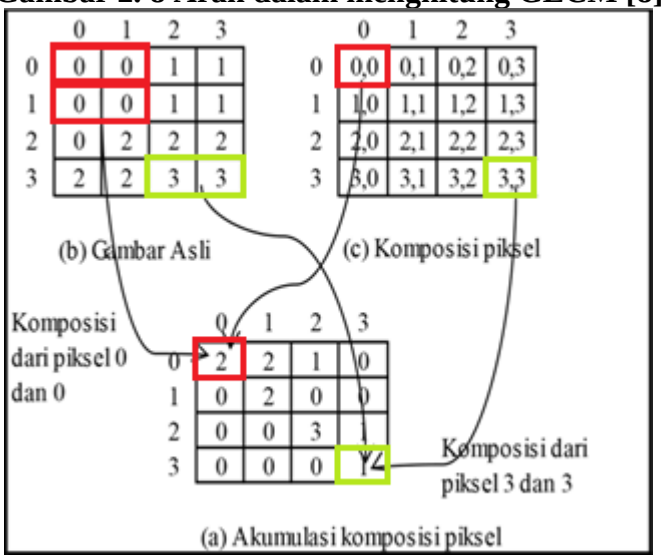


Gambar 2. 7 Hubungan ketetanggaan antar piksel sebagai fungsi orientasi dan jarak spasial (Ganis, 2011)

Arah piksel tetangga untuk mewakili jarak dapat dipilih, misalnya 135^o, 90^o, 45^o, 0^o atau, seperti yang diilustrasikan pada Gambar(2.8). Sedangkan Gambar(2.9). menggambarkan bagaimana untuk menghasilkan matriks menggunakan arah 0^o dan dengan jarak 1 piksel. [8]



Gambar 2. 8 Arah dalam menghitung GLCM [8]



Gambar 2. 9 Langkah pertama mengubah GLCM [8]

Dengan menambahkan transposnya, matriks simetrik akan diperoleh, seperti ditunjukkan pada Gambar(2.10), tapi hasilnya masih belum ternormalisasi. Oleh karena itu, proses normalisasi harus dilakukan untuk menghapus ketergantungan pada ukuran citra dengan mengatur semua elemen dalam matriks sehingga total dari semua nilai elemen sama dengan 1. Gambar(2.11). merupakan hasil dari matriks yang telah ternormalisasi.

$$\begin{bmatrix} 2 & 2 & 1 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 2 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 \\ 1 & 0 & 3 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 2 & 1 & 0 \\ 2 & 4 & 0 & 0 \\ 1 & 0 & 6 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix}$$

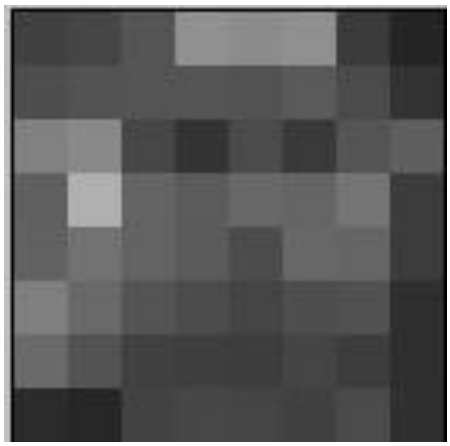
Gambar 2. 10 Prosedure membuat matriks simetrik

Gambar(11) menjelaskan perubahan urutan matriks dari baris ke kolom lalu dijumlahkan dan akan menghasilkan matriks GLCM sebelum normalisasi.

$$\begin{bmatrix} \frac{4}{24} & \frac{2}{24} & \frac{1}{24} & \frac{0}{24} \\ \frac{2}{24} & \frac{4}{24} & \frac{0}{24} & \frac{0}{24} \\ \frac{1}{24} & \frac{0}{24} & \frac{6}{24} & \frac{1}{24} \\ \frac{0}{24} & \frac{0}{24} & \frac{1}{24} & \frac{2}{24} \end{bmatrix}$$

Gambar 2. 11 Matriks ternormalisasi (Kadir dkk, 2011)

Sebagai contoh Gambar(2.12) citra keabuan dengan ukuran 8 piksel yang memiliki derajat keabuan dengan rentang dari 0 sampai 7 atau (0, 7), dalam hal ini 0 menyatakan hitam dan 7 menyatakan putih, nilai antara 0 sampai 7 menyatakan warna keabuan yang terletak antara hitan dan putih.



Gambar 2. 12 Citra keabuan dengan ukuran 8 piksel

Tabel 2. 1 Array ukuran 8x8

1	1	2	5	5	6	2	0
3	3	3	3	3	4	2	1
4	5	3	2	4	3	5	4
3	7	4	3	4	3	6	2

3	4	4	3	2	5	5	2
5	4	3	3	2	4	5	0
4	3	2	2	2	2	3	1
1	1	3	3	3	2	3	1

Dari Tabel(2.1) tersebut diatas dapat dihitung probabilitas hubungan ketetanggaan antara dua piksel pada jarak dan orientasi sudut tertentu. Kemudian dihitung dengan jarak spasial 1 dan sudut 90^0 dan akan diperoleh matriks kookurensi yang dapat dihitung ciri statistik yang merepresentasikan citra yang diamati.

Karena matriks dari table(2.1) tersebut memiliki delapan aras keabuan, maka jumlah nilai piksel ketetanggaan dan nilai piksel referensi pada area kerja matriks berjumlah delapan, seperti tampak pada Tabel(2.2) berikut.

Tabel 2. 2 Area kerja matriks

	0	1	2	3	4	5	6	7
0	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7
1	1,0	1,1	1,2	1,3	1,4	1,5	1,6	1,7
2	2,0	2,1	2,2	2,3	2,4	2,5	2,6	2,7
3	3,0	3,1	3,2	3,3	3,4	3,5	3,6	3,7
4	4,0	4,1	4,2	4,3	4,4	4,5	4,6	4,7
5	5,0	5,1	5,2	5,3	5,4	5,5	5,6	5,7
6	6,0	6,1	6,2	6,3	6,4	6,5	6,6	6,7
7	7,0	7,1	7,2	7,3	7,4	7,5	7,6	7,7

Langkah selanjutnya yaitu menghitung nilai matriks dengan mengisikan jumlah hubungan spasial sehingga akan menghasilkan nilai matriks seperti pada Tabel(2.3)

Tabel 2. 3 Pembentukan matriks kookurensi

GL	0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0

1	2	1	0	1	1	0	0	0
2	0	0	5	2	3	0	0	0
3	0	2	5	6	3	3	0	0
4	0	1	0	3	3	2	1	1
5	0	0	1	3	0	1	1	0
6	0	0	0	0	0	1	0	0
7	0	0	0	0	0	1	0	0

Proses yang ditunjukkan mulai Tabel 2.1, Tabel 2.2 dan Tabel 2.3 merupakan langkah pertama mengubah GLCM. Langkah selanjutnya nilai dari hasil pertama GLCM dicari nilai transposnya. Hasil dari nilai transpose dijumlahkan dengan nilai hasil pertama GLCM dan akan menghasilkan nilai matriks yang belum ternormalisasi seperti berikut.

Tabel 2. 4 Matrix Transpose + Matriks pertama GLCM = Matrik simetris Yang belum ternormalisasi

GL	0	1	2	3	4	5	6	7
0	0	2	0	0	0	0	0	0
1	0	1	0	2	1	0	0	0
2	1	0	5	5	0	1	0	0
3	0	1	2	6	3	3	0	0
4	0	1	3	3	3	0	0	0
5	0	0	0	3	2	1	1	1
6	0	0	0	0	1	1	0	0
7	0	0	0	0	1	0	0	0

+

GL	0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0
1	2	1	0	1	1	0	0	0
2	0	0	5	2	3	0	0	0
3	0	2	5	6	3	3	0	0
4	0	1	0	3	3	2	1	1
5	0	0	1	3	0	1	1	0
6	0	0	0	0	0	1	0	0
7	0	0	0	0	0	1	0	0

=

GL	0	1	2	3	4	5	6	7
0	0	2	1	0	0	0	0	0
1	2	2	0	3	2	0	0	0
2	1	0	10	7	3	1	0	0
3	0	3	7	12	6	6	0	0
4	0	2	3	6	6	2	1	1
5	0	0	1	6	2	2	2	1
6	0	0	0	0	1	2	0	0
7	0	0	0	0	1	1	0	0

Matriks yang telah simetris di atas selanjutnya harus dinormalisasi elemen-elemennya yang dinyatakan dengan probabilitas. Nilai elemen untuk masing-masing sel dibagi dengan jumlah seluruh elemen spasial seperti gambar(2.13) berikut.

$$\begin{bmatrix} \frac{0}{108} & \frac{2}{108} & \frac{1}{108} & \frac{0}{108} & \frac{0}{108} & \frac{0}{108} & \frac{0}{108} & \frac{0}{108} \\ \frac{2}{108} & \frac{2}{108} & \frac{0}{108} & \frac{3}{108} & \frac{2}{108} & \frac{0}{108} & \frac{0}{108} & \frac{0}{108} \\ \frac{1}{108} & \frac{0}{108} & \frac{10}{108} & \frac{7}{108} & \frac{3}{108} & \frac{1}{108} & \frac{0}{108} & \frac{0}{108} \\ \frac{0}{108} & \frac{3}{108} & \frac{7}{108} & \frac{12}{108} & \frac{6}{108} & \frac{6}{108} & \frac{0}{108} & \frac{0}{108} \\ \frac{0}{108} & \frac{2}{108} & \frac{3}{108} & \frac{6}{108} & \frac{6}{108} & \frac{2}{108} & \frac{1}{108} & \frac{1}{108} \\ \frac{0}{108} & \frac{0}{108} & \frac{1}{108} & \frac{6}{108} & \frac{2}{108} & \frac{2}{108} & \frac{2}{108} & \frac{1}{108} \\ \frac{0}{108} & \frac{0}{108} & \frac{0}{108} & \frac{0}{108} & \frac{1}{108} & \frac{2}{108} & \frac{0}{108} & \frac{0}{108} \\ \frac{0}{108} & \frac{0}{108} & \frac{0}{108} & \frac{0}{108} & \frac{1}{108} & \frac{1}{108} & \frac{0}{108} & \frac{0}{108} \end{bmatrix}$$

Gambar 2. 13 Nilai didalam Matriks Simetris dibagi jumlah nilai matriks simetris

Setelah diperoleh matriks kookurensinya dapat dihitung ciri statik ordedua yang merepresentasikan citra yang dipilih.

Haralick dkk mengusulkan berbagai jenis ciri statistik tekstur yang dapat diekstraksi dengan matriks kookurensi. Beberapa di antaranya Antara lain adalah : Kontras(contrast), Homogenitas(homogeneity), Entropi (Entropy), Energi (Energy) dan Dissimilariti (Dissimilarity).

Persamaan untuk fitur tersebut adalah sebagai berikut :

1. Kontras (*Contrast*)

   Menunjukkan ukuran penyebaran (momen inersia) 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. Hasil perhitungan kontras berkaitan dengan jumlah keberagaman intensitas keabuan dalam citra.

   Rumus $Kontras = \sum_{i,j=0}^{n-1} (Pi, j).(i-j)^2$ (persamaan 2)

Dimana:

i dan j adalah sifat keabuan dari resolusi 2 piksel yang berdekatan

p (i,j) adalah *Probabilitas kolom(i,j)*

2. Homogenitas (*Homogenity*)

homogenitas yang menunjukkan kehomogenan citra berderajat keabuan sejenis. Citra homogen akan memiliki harga homogenitas yang besar.

Rumus homogenitas $\sum_{i,j=0}^{n-1} P(i, j) / [1 + (i - j)^2]$ (persamaan 3)

Dimana:

i dan j adalah sifat keabuan dari resolusi 2 piksel yang berdekatan

p (i,j) adalah *Probabilitas kolom(i,j).*

3. Entropi (*Entropy*)

Entropi dapat menunjukkan ketidakteraturan ukuran bentuk, jika nilai Entropinya besar untuk citra dengan transisi derajat keabuan merata dan bernilai kecil jika struktur citra tidak teratur (bervariasi).

Rumus $Entropi = -\sum_{i,j=0}^{n-1} P(i, j) \ \log(P(i, j))$ (persamaan 4)

Dimana:

i dan j adalah sifat keabuan dari resolusi 2 piksel yang berdekatan

p (i,j) adalah *Cooccurence Matrix* Simetris Ternormalisasi

4. Energi (*Energy*)

Energi menyatakan ukuran konsentrasi pasangan dengan intensitas keabuan tertentu pada matriks, dimana (i, j) menyatakan nilai pada baris i dan kolom j pada matriks kookurensi.

Rumus $Energi = \sum_{i,j=0}^{n-1} P(i,j)^2$ (persamaan

Dimana:

i dan j adalah sifat keabuan dari resolusi 2 piksel yang berdekatan

p (i,j) adalah *Cooccurence Matrix* Simetris Ternormalisasi

5. Dissimilarity

Dissimilarity menyatakan ukuran ketidaksamaan linear derajat keabuan citra sehingga dapat memberikan petunjuk adanya struktur linear dalam citra.

Rumus Dissimilarity : $Dissimilarity = \sum_{i,j=0}^{n-1} P(i,j).|i-j|$ (persamaan 6)

Dimana:

i dan j adalah sifat keabuan dari resolusi 2 piksel yang berdekatan

p (i,j) adalah frekuensi relatif matriks dari resolusi 2 piksel yang berdekatan.

2.4. Klasifikasi

Klasifikasi merupakan suatu pekerjaan yang melakukan penilaian terhadap suatu obyek data untuk masuk dalam suatu kelas tertentu dari sejumlah kelas yang tersedia. [9].

Ada dua pekerjaan utama:

1. Pembangunan model sebagai prototype untuk disimpan sebagai memori
2. Menggunakan model tersebut untuk melakukan pengenalan/klasifikasi/prediksi pada suatu obyek data lain masuk pada kelas mana

Contoh aplikasi:

1. Klasifikasi pengenalan jenis hewan spesies baru.
2. Diagnosis penyakir pasien
3. Pengenalan jenis bunga
4. Pemeriksaan kualitas bunga kamboja (bahan teh)
5. Absensi karyawan (dengan sidik jari, mata, wajah, tangan, dsb.)

2.4.1. Algoritma klasifikasi

Algoritma klasifikasi dibagi menjadi dua macam: *eager learner* dan *lazy learner* [9].

Eager learner

1. Didesain untuk melakukan pembacaan/ pelatihan/ pembelajaran pada data latih untuk dapat memetakan dengan benar setiap vektor masukan ke label kelas keluarannya.
2. Model (berupa bobot / sejumlah nilai kuantitas tertentu) disimpan sebagai memori, sedangkan semua data latihnya dibuang.
3. Prediksi dilakukan menggunakan model yang tersimpan, tidak melibat data uji sama sekali

Trade-off:

4. Proses prediksi berjalan dengan cepat
5. Proses pelatihan lama

Contoh: Artificial Neural Network (ANN), Support Vector Machine (SVM), Decision Tree, Bayesian, dan sebagainya.

Lazy learner

1. Hanya sedikit melakukan pelatihan (bahkan tidak sama sekali)
2. Hanya menyimpan sebagian atau seluruh data latih, kemudian menggunakan data latih tersebut ketika proses prediksi.

Trade-off

3. Proses prediksi menjadi lama karena model harus membaca kembali semua data latihnya untuk dapat memberikan keluaran label kelas dengan benar pada data uji yang diberikan.
4. Proses pelatihan berjalan dengan cepat

Contoh: K-Nearest Neighbor (K-NN), Fuzzy K-Nearest Neighbor (FK-NN), Regresi Linear, dan sebagainya.

2.4.2. Klasifikasi naïve bayes

Naïve bayes adalah Teknik prediksi berbasis probabilistik sederhana yang berdasar pada penerapan teorema Bayes [9]

1. Asumsi independensi (ketidaktergantungan) yang kuat (naif).
2. Model yang digunakan adalah "model fitur independen"

Independensi yang kuat pada fitur adalah bahwa sebuah fitur pada sebuah data ***tidak ada kaitannya*** dengan ***adanya*** atau ***tidak adanya fitur yang lain*** dalam data yang sama. Contoh: kasus klasifikasi hewan dengan fitur: penutup kulit, melahirkan, berat, dan menyusui

1. Dalam dunia nyata, hewan yang berkembang biak dengan cara melahirkan dipastikan hewan tersebut menyusui juga, disini ada ketergantungan pada fitur menyusui karena hewan yang menyusui biasanya melahirkan, atau hewan yang bertelur biasanya tidak menyusui.
2. Dalam Bayes, hal tersebut tidak dipandang, sehingga masing-masing fitur seolah tidak ada hubungan apa-apa.

Kasus lain: prediksi hujan

1. Hujan tergantung angin, cuaca kemarin, kelembaba udara (tidak ada kaitan satu sama lain)
2. Tapi juga tidak boleh memasukkan fitur lain yang tidak ada hubungannya dengan hujan, seperti: gempa bumi, kebakaran, dsb.

Naïve bayes adalah suatu metode pengklasifikasian paling sederhana dengan menggunakan peluang yang ada, dimana diasumsikan bahwa setiap variable X bersifat bebas (*independence*). Karena asumsi variabel tidak saling terikat, maka didapatkan persamaan (7).

$$P(X \mid Y = y) = \prod_{i=1}^{q} P(X_i \mid Y = y) \qquad \text{(persamaan 7)}$$

Data yang digunakan dapat bersifat kategorial (diskrit) maupun kontinyu. Namun, pada tugas akhir ini akan digunakan data kontinyu, karena hasil ekstraksi ciri citra merupakan data kontinyu berupa angka angka hasil pengukuran tingkat kontras, homogenitas, entropy, energy, dan dissimilarity pada ekstraksi ciri. Maka dari itu

Untuk data kontinyu dapat diselesaikan dengan menggunakan langkah-langkah berikut.

Training :

1. Hitung rata-rata (*mean*) tiap fitur dalam dataset training dengan persamaan (8).

   $\mu = \frac{\sum x_i}{n}$ (persamaan 8)

   Dimana:

   μ = mean

   n = banyaknya data

   $\sum x_i$ = jumlah nilai data

2. Kemudian hitung nilai varian dari diat dataset training tersebut seperti pada persamaan (9).

   $\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2$ (persamaan 9)

   Dimana:

   σ^2= varians

   μ= mean

   x_i = nilai data

   $n =$ banyaknya data

Testing :

1. Hitung probabilitas (*Prior*) tiap kelas yang ada dengan cara menghitung jumlah data tiap kelas dibagi jumlah total data secara keseluruhan.
2. Selanjutnya menghitung densitas probabilitasnya menggunakan persamaan (10). Fungsi densitas mengekspresikan probabilitas relatif. Data dengan mean μ dan standar deviasi σ, fungsi densitas probabilitasnya adalah :

   $\varphi_{\mu\sigma}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ (persamaan 10)

   Dimana :

   x = data masukan

   π = 3,14

   $\sigma =$ standar deviasi

μ = mean

3. Setelah didapatkan nilai densitas probabilitasnya, selanjutnya menghitung posterior masing-masing kelas dengan menggunakan persamaan (11).

$$Posterior = \frac{P(C_i) x\, P(X1 \dots \dots \dots \dots \dots \dots \dots Xn|C_i)}{P(X1 \dots \dots \dots \dots \dots \dots \dots Xn|C_i)} \qquad \text{(persamaan 11)}$$

$atau\ bisa\ ditulis$

$$posterior = \frac{prior\ x\ likehood}{evidnce}$$

4. Nilai posterior terbesar adalah kelas yang sesuai.

2.3.OOP(Object Oriented Programing)

Pemrograman berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapann prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktifitas analisis beorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek [10]

Pada saat ini, metode berorientais objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentranformasi hasil dari satu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan saat ini sangat beragam (aplikasi bisnis, *real-time, utility* dan sebagainya) dengan platform yang berbeda-beda, sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi ke semua jenis aplikasi.

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut :

a. Meningkatkan Produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reuseable*).

b. Kecepata Pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan mennyababkan berkurangnya kesalahan pada saat pengkodean

c. Kemudahan Pemeliharaan

   Karena dengan model objek, pola-pola yang cendrung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering diubah-ubah.

d. Adanya Konsistensi

Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

e. Meningkatkan Kualitas Perangkat Lunak

Karena adanya pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengambangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Berikut beberapa contoh bahasa pemograman yang mendukung pemrograman berorinentasi objek adalah :

a. Smalltalk

Smalltalk adalah salah satu bahasa pemograman yang diekmbangkan untuk mendukung pemrograman beroirentasi objek.

b. Bahasa Pemrograman Eiffel

Eiffel merupakan bahsa pemrograman yang kembangkan untuk mendukung pemrograman berorientasi objek oleh Bertrand Meyer dan *compiler*.

c. Bahasa Pemrograman (Web) PHP

Php dibuat pertama kali oleh seorang perekayasa perangkat (software engineering) yang bernama Rasmus Lerdoff.

d. Bahasa Pemrograman C++

C++ merupakan pengembangan lebih lanjut dari bahasa pemrograman C untuk mendukung pemrograman berorientasi objek.

e. Bahasa Pemrograman Java

Java dikembangkan oleh perusahaan Sun Microsystem. Java menurut definisi dari Sun Microsystem adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *standalone* ataupun pada lingkungan jaringan.

2.3.1. Konsep Dasar Berorientasi Objek

Berikut adalah konsep dasar pemrograman berorientasi objek :

a. Objek (*Object*)

Objek adalah abtraksi dan sesuatu yang mewakili dunia nyata seperti benda, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal

lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi dan mempunyai operasi yang dapat diterapkan atau dapat berpengaruh pada status objeknya.

b. Kelas (*Class*)

Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama yang mungkin lahir atau diciptakan dalam kelas tersebut.

c. Pembungkusan (*Encapsulation*)

Pembungkusan atribut data dan layanan yang mempunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

d. Pewarisan (*Inheritance*) dan Generalisasi/Spesialisasi

Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dirinya.

e. Metode

Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural.

f. Polimorfisme

Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

2.3.2. UML (*Unified Modelling Language*)

Unified Modelling Language (UML) adalah sekumpulan spesifikasi yang dikeluarkan oleh OMG. UML terbaru adalah UML 2.3 yang terdiri dari 4 macam spesifikasi, yaitu : *Diagram Interchange Spesification,* UML *Infrastrukture,* UML *Superstrukture, dan Object Constraint Language* (OCL). Pada UML 2.3 terdiri 13 macam diagram yang dikelompokan pada 3 kategori, yaitu [11]:

A. Struktur Diagram, yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. Pada Struktur Diagram dibagi menjadi 6 bagian [11]:

1. Diagam Kelas

Diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut attribut dan metode atau operasi.

2. Diagram Objek

   Digram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem.

3. Diagram Komponen

   Diagram komponen dibuat untuk menunjukan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem.

4. *Composite Structure Diagram*

   Composite Structure Diagram bru muali ada pada UML versi 2.0. diagram ini dapat digunakan untuk menggambarkan struktur dari bagian-bagian yang saling terhubung maupun mendeskripsikan struktur pada saat berjalan (*runtime*) .

5. *Package Diagram*

   Package diagram menyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram UML. Hampir semua diagram dalam UML dapat dikelompokan menggunakan *Package Diagram*.

6. *Deployment Diagram*

   Deployment diagram menunjukan konfigurasi komponen dalam proses eksekusi aplikasi.

B. *Behavior Diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. Pada *Behavior Diagram* dibagi menjadi 3 bagian :

1. *Use Case Diagram*

   Use case diagram merupakan pemodelan untuk kelakuan (behavior) sistem informasi yang akan dibuat,. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat.

2. *Activity Diagram*

   Activity diagram menggambarkan *workflow* atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak.

3. *State Machine Diagram*

   State machine diagram digunakan untuk menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem atau objek.

C. *Interactions Diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi antar subsistem pada suatu sistem. Pada Interactions Diagram dibagi menjadi 4 bagian :

1. *Sequence Diagram*

   Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek.

2. *Communication Diagram*

   Communication diagram menggambarkan interaksi antar bojek/bagian dalam bentuk urutan pengiriman pesan. Diagram komunikasi merepresentasikan informasi yang diperoleh dari diagram kelas, diagram *sequence*, dan diagram *use case* untuk mendeskripsikan gabungan antara struktur statis dan tingkah laku dinamis dari suatu sistem.

3. *Timing Diagram*

   Timing diagram merupakan diagram yang fokus pada penggambaran terkait batas waktu.

4. *Interaction Overview Diagram*

   Interaction overview diagram mirip dengan diagram aktivitas yang berfungsi untuk menggarbarkan sekumpulan urutan aktivitas, diagram ini adlaah bentuk aktivias diagram yang setiap titik merepresentasikan diagram interaksi.

2.4. Bahasa C#

C# (dibaca “See-Sharp”) adalah bahasa pemrograman baru yang diciptakan oleh Microsoft (dikembangkan dibawah kepemimpinan Anders Hejlsberg yang notabene juga telah menciptakan berbagai macam bahasa pemrograman termasuk Borland Turbo C++ dan Borland Delphi). Bahasa C# juga telah di standarisasi secara internasional oleh ECMA [12].

Seperti halnya bahasa pemrograman yang lain, C# bisa digunakan untuk membangun berbagai macam jenis aplikasi, seperti aplikasi berbasis windows (desktop) dan aplikasi berbasis web serta aplikasi berbasis web services [13].

Kelebihan C#, yaitu :

a. Sederhana (simple) C# menghilangkan beberapa hal yang bersifat kompleks yang terdapat dalam beberapa macam bahasa pemrograman seperti Java dan C++, termasuk diantaranya mengilangkan macro, templates, multiple inheritance dan virtual base classes. Hal-hal tersebut yang dapat menyebabkan kebingunan pada saat menggunakannya, dan juga berpotensial dapat menjadi masalah bagi para programmer C++.
b. C# bersifat sederhana, karena bahasa ini didasarkan kepada bahasa C dan C++.
c. Modern Apa karena adanya beberapa fitur seperti exception handling, garbage collection, extensible data types, dan code security (keamanan kode/bahasa pemrograman). Dengan adanya fitur-fitur tersebut, menjadikan bahasa C# sebagai bahasa pemrograman yang modern.
d. Powerfull dan fleksibel, C# bisa digunakan untuk membuat berbagai macam aplikasi, seperti aplikasi pengolah kata, grapik, spreadsheets, atau bahkan membuat kompiler untuk sebuah bahasa permrograman.
e. Efisien, C# adalah bahasa pemrograman yang menggunakan jumlah kata-kata yang tidak terlalu banyak. C# hanya berisi kata-kata yang biasa disebut dengan keywords. Keywords ini digunakan untuk menjelaskan berbagai macam informasi.

Modular Kode C# ditulis dengan pembagian masing Class-Class (classes) yang terdiri dari beberapa routines yang disebut sebagai member methods. Class-Class dan metode-metode ini dapat digunakan kembali oleh program atau aplikasi lain. Hanya dengan memberikan informasi yang dibutuhkan oleh Class dan metode yang dimaksud, maka kita akan dapat membuat suata kode yang dapat digunakan oleh satu atau beberapa aplikasi dan program (reusable code).

2.5. Visual Studio.NET

Visual Studio .NET merupakan editor yang paling ideal untuk membuat aplikasi yang berbasis Framework .NET, termasuk aplikasi dengan bahasa C#. Dengan editor ini, maka akan bisa memanfaatkan kemampuan C# secara maksimal. Editor ini tidak hanya menyediakan berbagai macam tools dan wizard untuk membuat aplikasi C#, tapi juga termasuk fitur-fitur produktif seperti IntelliSense dan bantuan yang dinamis [13].

Dengan IntelliSense, jika mengetikan nama sebuah namespace atau nama Class, maka anggota dari namespace atau Class itu akan secara otomatis di munculkan sehingga kita tidak usah mengingat anggota dari semua I atau semua Class yang kita gunakan. IntelliSense, juga akan menampilkan semua argumen dan jenis typenya ketika kita mengetikan nama dari sebuah metode. Visual Studio 6 juga telah memiliki kemampuan ini, hanya saja Visual Studio 6 tidak mensupport jenis type dan Class-Class yang termasuk di dalam lingkungan Framework .NET.

Fitur bantuan yang dinamis (dynamic help) merupakan fitur yang baru yang ada di dalam "keluarga" editor Visual Studio. Ketika mengetikan sebuah code pada editor, sebuah jendela yang terpisah akan menampilkan topik-topik bantuan dan penjelasan yang berhubungan dengan kata-kata tempat di mana kita menempatkan cursor. Misalnya, ketika mengetikan kata namespace, maka di jendela yang terpisah tadi akan dimunculkan topik-topik yang berhubungan dengan kata kunci namespace. Hal ini tentu akan sangat membantu pada saat pembuatan program untuk dapat lebih memahami dan mengerti tentang kata-kata / kode yang ditulis.

2.6. Microsoft Visio

Microsoft Visio 2010 adalah aplikasi untuk menggambarkan diagram yang dapat membantu pekerja IT dan staf profesional untuk memvisualisasikan informasi yang kompleks. Berawal dari teks yang rumit dan tabel yang kompleks yang sulit dipahami, Microsoft Visio berusaha untuk membantu kondisi tersebut.

Microsoft Visio 2010 adalah tools untuk membuat pelbagai jenis diagram, mulai dari diagram jaringan sampai ke kelender dan mulai dari tampilan teks biasa, sampai ke bentuk flowchart. Visio 2010 membuat dokument visual secara

profesional untuk membantu analisa dan komunikasi informasi, sistem dan proses yang kompleks [14].

2.7. Metode pengujian keakurasian K-Fold Cross validation

Dalam machine learning weka, pengujian keakurasian dapat dilakukan dengan 3 tipe pengujian, yaitu

1. Training set test
2. Supplied set test
3. K-fold cross validation

1. Training set test

Metode pengujian menggunakan data yang telah di training, dengan kata lain, data training dan data uji adalah data yang sama

2. Supplied set test

Metode pengujian menggunakan data yang berbeda, dengan kata lain, data training berbeda dengan data yang akan diujikan

3. K-fold cross validation

Cross Validation merupakan salah satu teknik untuk menilai/memvalidasi keakuratan sebuah model yang dibangun berdasarkan dataset tertentu. Pembuatan model biasanya bertujuan untuk melakukan prediksi maupun klasifikasi terhadap suatu data baru yang boleh jadi belum pernah muncul di dalam dataset. Data yang digunakan dalam proses pembangunan model disebut data latih/training, sedangkan data yang akan digunakan untuk memvalidasi model disebut sebagai data test.

Salah satu metode cross-validation yang populer adalah K-Fold Cross Validation. Dalam teknik ini dataset dibagi menjadi sejumlah K-buah partisi secara acak. Kemudian dilakukan sejumlah K-kali eksperimen, dimana masing-masing eksperimen menggunakan data partisi ke-K sebagai data testing dan memanfaatkan sisa partisi lainnya sebagai data training. Sebagai gambaran, jika melakukan 5-Fold Cross-Validation maka desain data eksperimennya sebagai berikut:

Dataset:

| K1 | K2 | K3 | K4 | K5 |

Data Eksperimen:

Eksperimen Ke	Data Latih	Data Test
1	K2,K3,K4,K5	K1
2	K1,K3,K4,K5	K2
3	K1,K2,K4,K5	K3
4	K1,K2,K3,K5	K4
5	K1,K2,K3,K4	K5

Untuk mendapatkan nilai akurasi ataupun ukuran penilaian lainnya dari hasil eksperimen yang kita lakukan, dapat diambil nilai rataan dari seluruh eksperimen tersebut.