BAB 2
TINJAUAN PUSTAKA

2.1 Optical Character Recognition (OCR)

Optical Character Recognition (OCR) adalah teknik mengubah sebuah gambar berisi teks, tulisan tangan, menjadi teks yang dapat diubah untuk proses selanjutnya, Sejarah OCR dimulai sejak 1950 dan masih terus berkembang sampai saat ini**.** Teknologi ini terus berkembang seiring berkembangnya teknologi. Teknologi ini memungkinkan mesin untuk mengenali mengenali teks secara otomatis, teknik ini seperti kombinasi dari mata dan otak manusia, sebuah mata dapat melihat seuah teks dari gambar tapi sebenarnya otak yang memproses dan mengolah teks tersebut sehingga dapat dibaca oleh mata. Dalam pengembangan OCR terdapat beberapa masalah, diantaranya yang pertama adalah terdapat beberapa huruf dan angka yang sulit dibedakan karena memiliki ciri yang hampir sama, faktor kedua adalah faktor cahaya dalam citra yang mempersulit system mengenali huruf [7]

2.2 Dokuman Surat Keputusan

Menurut Kamus Besar Bahasa Indonesia, dokumen adalah surat yang tertulis atau tercetak yang dapat dipakai sebagai bukti keterangan. Surat tugas atau surat keputusan adalah surat yang menerangkan bahwa orang yang diberi surat itu diperintahkan atau diberi tugas untuk menjalankan sesuatu.

Pada penelitian ini, dokumen yang akan digunakan adalah dokumen surat keputusan. Dokumen yang digunakan hanya dokumen surat keputusan yang ditujukan kepada 1 (satu) orang. Lembar dokumen surat keputusan terdiri dari Kepala Surat, Jenis Surat, Nomor Surat, Tentang, Jabatan yang Menetapkan Surat, Menimbang, Mengingat, Putusan, Nama yang Dituju, Nomor Induk yang Dituju, Isi Putusan, Tempat Diputuskan, Tanggal Diputuskan, Organisasi, Nama yang Menetapkan Surat, dan Tembusan.

2.3 Pengolahan Citra

Istilah citra digital sangat polpuler pada masa sekarang. Banyak perlatan elektronik,misalnya *scanner*, kamera digital, mikroskop digital, dan *fingerprint*

reader (pembaca sidik jari), yang menghasilkan citra digital. Perangkat lunak untuk mengolah citra digital jugasangat popular digunakan oleh pengguna untuk mengolah foto atau untuk berbagai keperluan lain. Penolahan citra merupakan proses memanipulasi dan menganalisis citra dengan bantuan komputer, dalam hal ini mengolah informasi yang terdapat pada suatu gambar untuk keperluan pengenalan objek secara otomatis. Ada berbagai teknik pengolahan citra tergantung kebutuhan dan keluaran yang diinginkan [8].

2.4 Prepocessing

Pre-processing adalah bagiaan penting dari setiap system pemrosesan Bahasa alami, karena karakter, kata, dan kalimat yang diintifikasi pada tahap ini adalah unit dasar atau awal sebelum pemrosesan lebih lanjut. *Pre-processing* dilakukan karena data teks sering mengandung berbagai macam format atau berbeda-beda seperti format angka dan kata-kata yag tidak membantu dan dapat dihilagkan sehingga memudahkan proses selanjutya.

Image *preprocessing* adalah suatu bentuk pengolahan atau pemrosesan sinal dengan input berupa gambar (image) dan ditransformasikan menjadi gambar lain sebagai keluatan dengan teknik tertentu. *Image preprocessing* dilakukan untuk memprbaiki kesalahan data sinyal akibat transmisi dan selama akusisi sinyal, serta untuk meningkatkan kualitas penampakan gambar agar lebih mudah diinterpretasi oleh system penglihatan manusia baik dengan melakukan manipulasi dan juga penganalisisan terhadap gambar. Operasi *image preprocessing* dapat dikelompokan berdasarkan tujuan transformasinya.

2.4.1 Grayscale

Grayscale adalah istilah untuk menyebutkan satu citra yang memiliki warna abu-abu, hitam, dan putih. Grayscale adalah koleksi atau kisaran corak monokromik (abu-abu), mulai dari putih murni di ujung yang paling terang hingga hitam murni di ujung yang berlawanan.

Citra *grayscale* adalah citra yang hanya memiliki 1 buah kanal sehinga yang ditapilan hanyalah nilai intensitas atau dikenal juga dengan istilah derajat keabuan. Karena jenis citra ini hanya memiliki 1 kanal saja, maka citra *grayscale* memiliki

tempat penyimpanan yang lebih hemat. Jenis citra ini disebut juga sebagai 8-bit. Foto hitam putih maupun gambar yang ditampilkan oleh televisi hitam putih sebenarnya meggunakan citra *grayscale,* bukan dalam warna hitam dan warna putih. Namun dikalagan masyarakat istilah foto hitam putih maupun televisi hitam putih sudah terbiasa digunakan falam kehidupan sehari-hari [9].

Secara teori ada beberapa cara dalam mengonversi citra berwarna RGB ke dalam citra *grayscale.* Untuk mendapatkan hasil konversi yang baik, persamaan (2.1) berikut dapat digunakan :

$$GS = 0.299R' + 0.587G' + 0.114B' \quad (2.1)$$

Keterangan :

GS : Citra *grayscale*

R’ : Komponen merah dari citra RGB

G’ : Komponen hijau dari citra RGB

B’ : Komponen biru dari citra RGB

2.4.2 Thresholding

Citra biner atau citra hitam putih adalah citra yang hanya memiliku 2 kemungkinan nilai untuk setiap pikselnya, yaitu 0 atau 1. Nilai 0 akan tampil sebagai sebagai warna hitam sedangkan nilai 1 akan tampil sebagai warna putih. Maka dari itu, jenis citra ini hanya membutuhkan 1-bit untuk menyimpan setiap nilai pada setiap pikselnya. Jenis citra ini sering digunakan untuk proses *masking* ataupun proses segmentasi citra [9]. Proses thresholding digunakan untuk mengekstrak *foreground* (tinta) dan *background* (kertas) dan mengubah menjadi citra biner. Proses *thresholding* mengubah warna gambar menjadi citra biner (*binary image*) dimana ditentukan sebuah nilai level *threshold* kemudian piksel yang memiliki nilai level dibawah level *threshold* di set menjadi warna putih (1 pada nilai biner) dan nilai diatas nilai *threshold* di set menjadi warna hitam (0 pada nilai biner).

2.4.3 Resize

Resize adalah proses yang digunakan untuk mengubah ukuran citra digital dalam piksel, baik menjadi lebih kecil atau lebih besar dari ukuran sebenarnya. Proses resize pada penelitian ini tidak memerlukan metode khusus, caranya hanya

dengan dilakukan perbandingan ukuran antara citra hasil thresholding (pada tahap latih) dan hasil segmentasi (pada tahap uji) dengan menggunakan persamaan (2.2 **)** untuk mendapatkan posisi koordinat dari titik x yang baru dan persamaan (2.3) untuk mendapatkan posisi koordinat dari titik y yang baru[9]. Dimana kedua persamaan tersebut adalah sebagai berikut.

$$Xbaru = \frac{pb \times pp}{pa} \qquad (2.2)$$

Keterangan:

$Xbaru$ = Posisi koordinat x baru

pb = Ukuran panjang dari matriks baru

pp = Posisi koordinat x lama

pa = Ukuran panjang dari matriks lama

Untuk mencari koordinat *y* yang baru, digunakan persamaan (2.3 **)** sebagai berikut.

$$Ybaru = \frac{lb \times pp}{la} \qquad (2.3)$$

Keterangan:

$Ybaru$ = Posisi koordinat y baru

lb = Ukuran lebar dari matriks baru

pp = Posisi koordinat y lama

la = Ukuran lebar dari matriks lama

Pada penelitian ini *resize* digunakan untuk merubah ukuran *pixel* sesuai dengan kebutuhan dalam melakukan proses ektraksi ciri. Hal ini dilakukan agar semua citra karakter dari hasil segmentasi mempunyai ukuran yang sama (normalisasi ukuran citra) sebelum masuk ke tahap ekstraksi fitur.

2.4.4 Segmentasi

Segmentasi bertujuan untuk memotong huruf per-huruf. Pemotongan tersebut dilakukan dengan cara mencari pixel-pixel terluar dari setiap sisi (atas, bawah, kiri, kanan). Pixel-pixel terluar itulah yang akan menjadi batas pemotongan, sehingga didapat citra segiempat yang siap diproses lebih lanjut.

2.5 Profile Projection

Metode *Profile Projection* digunakan untuk memisahkan tulisan perbaris dan perkarakter. Metode *Profile Projection* akan menghitung jumlah piksel non-background secara vertikal dan horizontal dan nilai tersebut akan dibandingkan dengan nilai ambang tertentu untuk memisahkan tulisan perbaris dan perkarakter [10].

2.5.1 Horizontal Profile projection

Horizontal Profile Projection adalah jumlah banyaknya piksel hitam yang tegak lurus dengan sumbu x. *Horizontal Profile Projection* dipresentasikan dengan suatu vektor Ph berukuran M. *Horizontal Profile Projection* pada kolom ke-j, yaitu Ph[j], didefinisikan sebagai berikut :

$$P_h[h] = \sum_{j=1}^{N} S[i,j] \qquad (2.4)$$

2.5.2 Vertical Profile Projection

Vertical Profile Projection adalah jumlah banyaknya piksel hitam yang tegak lurus dengan sumbu y. *Vertical Profile Projection* dipresentasikan dengan suatu vektor Pv berukuran N. *Vertical Profile Projection* pada baris ke-I, yaitu Pv[i], didefinisikan sebagai berikut :

$$P_v[v] = \sum_{i=1}^{M} S[i,j] \qquad (2.5)$$

Keterangan :

S : Citra Biner

M : Banyak kolom pada citra

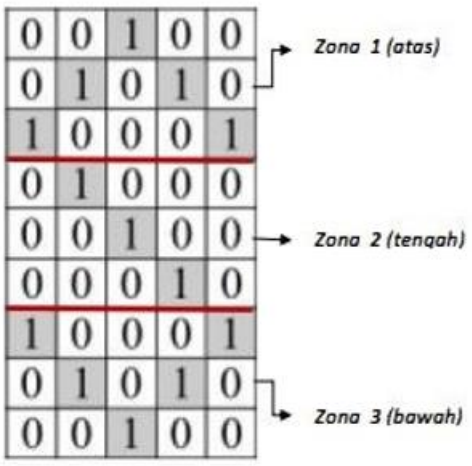
N : Banyak baris pada citra

2.6 Ekstraksi Fitur

Ekstraksi fitur bertujuan untuk mendapatkan karakteristik suatu karakter yang membedakan dari karakter lain. dalam penelitian ini, untuk mendapatkan karakteristik dengan cara memasukan nilai pixel gambar ke dalam sebuah vektor atau array 1 dimensi. Sebelum dimasukkan ke array, pixel diubah terlebih dahulu menjadi nilai 1 dan 0 dimana 1 mewakilkan warna putih dan 0 mewakilkan warna hitam pada gambar.

2.6.1 Zone Based Feature Extraction

Zoning adalah salah satu ekstraksi fitur yang paling popular dan sederhana untuk diimplementasikan [11]. Setiap citra dibagi menjadi N x M zona dan dari setiap zona tersebut dihitung nilai fitur sehingga didapatkan fitur dengan panjang N x M. Salah satu cara menghitung nilai fitur setiap zona adalah dengan menghitung jumlah piksel hitam setiap zona dan membaginya dengan jumlah piksel hitam terbanyak pada yang terdapat pada salah satu zona. Contoh pembagian 3 zona pada citra biner dapat dilihat dari **Gambar 2.1**.



Gambar 2.1 Pembagian Zona pada Citra Biner

Metode ekstraksi fitur berbasis zona memberikan hasil yang baik bahkan ketika langkah sebelum proses tertentu dimulai seperti filtering. Smoothing dan menghapus zona yang dianggap tidak ada. Konsep metode ekstraksi ciri yang

digunakan untuk mengekstraksi fitur untuk klasifikasi yang efisien dan pengenalan yaitu :

1. Hitung *centroid* dari citra. Menghitung centroid dari citra biner masukan dengan persamaan (2.6) dan (2.7 **)**.

   Rumus mencari *centroid* X

   $$C_x = \frac{(x_1.p_1 + x_2.p_2 + \cdots + x_n.p_n)}{(p_1 + p_2 + \cdots + p_n)} \qquad (2.6)$$

   Rumus mencari *centroid* Y

   $$C_y = \frac{(y_1.p_1 + y_2.p_2 + \cdots + y_n.p_n)}{(p_1 + p_2 + \cdots + p_n)} \qquad (2.7)$$

   Keterangan :

   a. Cx = *centroid* koordinat x
   b. Cy = *centroid* koordinat y
   c. Xn = koordinat x dari piksel ke-n
   d. Yn = koordinat y dari piksel ke-n
   e. Pn = nilai piksel ke-n

2. Bagi matriks kedalam *n* buah zona yang sama besar proporsinya.
3. Hitung jarak antara titik *centroid* dengan koordinat *pixel* yang memiliki nilai. Persamaan untuk menghitung jarak piksel.

   $$d(P, C) = \sqrt{(x_p - C_x)^2 + (y_p - C_y)^2} \qquad (2.8)$$

   Keterangan :

   a. d = jarak antara dua titik
   b. P = koordinat piksel
   c. C = koordinat centroid
   d. X_p = koordinat piksel X
   e. Y_p = koordinat piksel Y
   f. C_x = koordinat centroid X
   g. C_y = koordinat centroid Y

4. Ulangi langkah 3 untuk *pixel* yang ada di semua zona.

5. Hitung rata-rata dari jarak yang telah didapat pada langkah 3.

$$rerata\ jarak = \sum \frac{d(P,C)}{\sum P} \qquad (2.9)$$

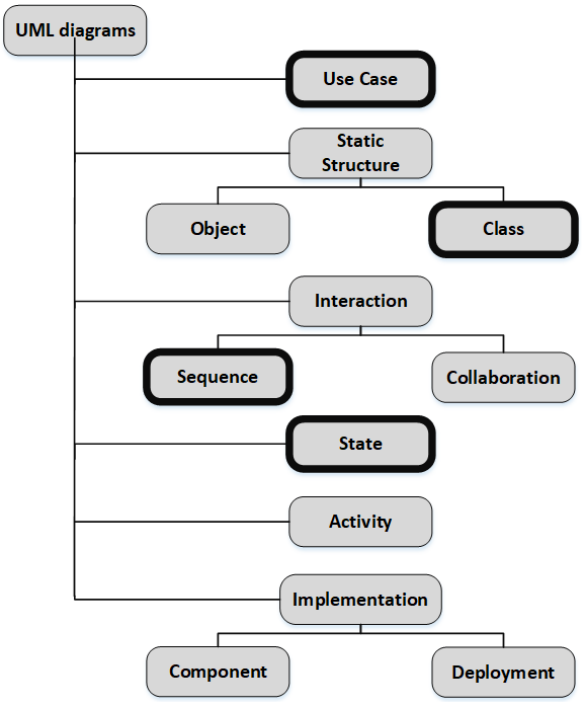
6. Ulangi langkah 5 hingga didapat masing-masing rata-rata jarak dari setiap zona.
7. Akhirnya *n* buah fitur akan didapat untuk melakukan klasifikasi dan pengenalan

2.7 UML (*Unified Modeling Language*)

Unified Modeling Language (UML) adalah bahasa pemodelan visual yang digunakan untuk menspesifikasikan, menvisualisasikan, membangun, dan mendokumentasikan rancangan dari suatu sistem perangkat lunak [12].

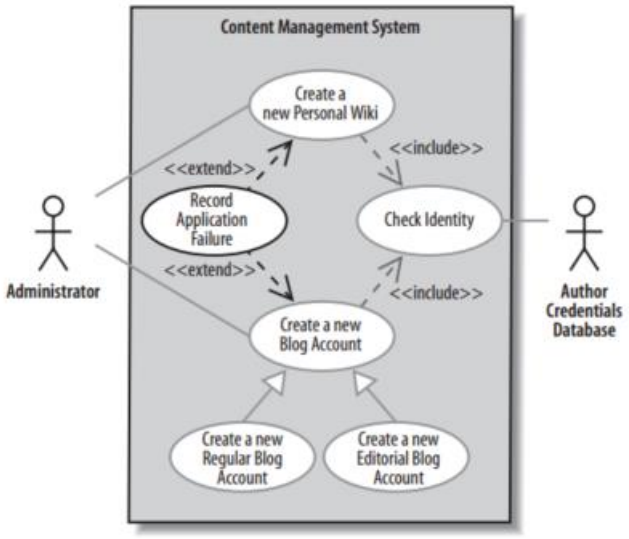
Pemodelan memberikan gambaran yang jelas mengenai sistem yang akan dibangun baik dari sisi struktural ataupun fungsional. UML dapat diterapkan pada semua model pengembangan, tingkatan siklus sistem, dan berbagai macam domain aplikasi. Dalam UML terdapat konsep semantik, notasi, dan panduan masing-masing diagram. UML bertujuan menyatukan teknik-teknik pemodelan berorientasi objek-objek menjadi terstandarisasi [12].

UML dibuat oleh Grady Booch, James Rumbaugh, dan Ivar Jacobson di bawah bendera Rational Software Corps. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai prespetktif. UML tidak hanya digunakan dalam pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan [13].



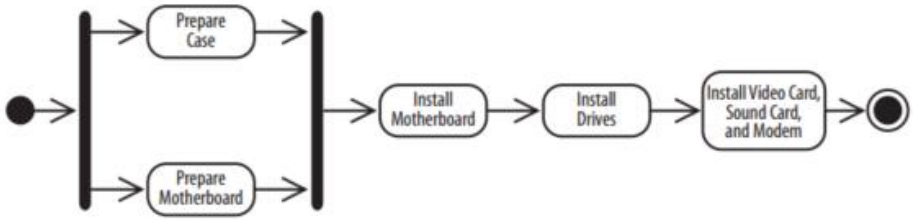
Gambar 2.2 UML

Use case diagram menyajikan interakasi antara use case dan aktor. Dimana, aktor dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang dibangun. Use case menggambarkan fungsionalitas sistem atau persyaratan-persyaratan yang harus dipenuhi sistem dari pandangan pemakai. Sebuah use case digambarkan sebagai elips horizontal dalam suatu diagram UML use case [13].



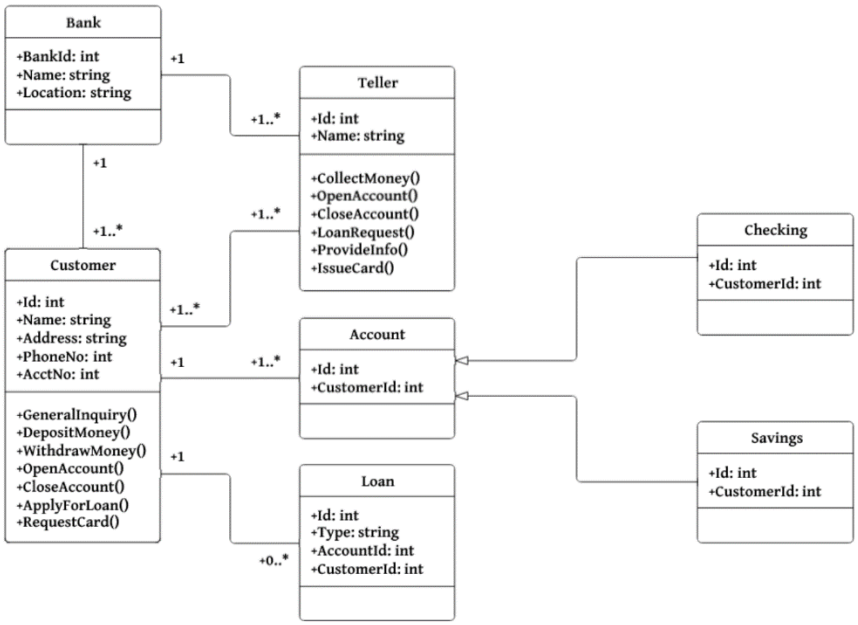
Gambar 2.3 Use Case Diagram

Activity diagram merupakan diagram yang menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. *Activity* diagram juga digunakan untuk mendefinisikan urutan atau pengelompokan tampilan dari sistem/*user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antar muka tampilan serta rancang menu yang ditampilkan pada perangakat lunak. [13].



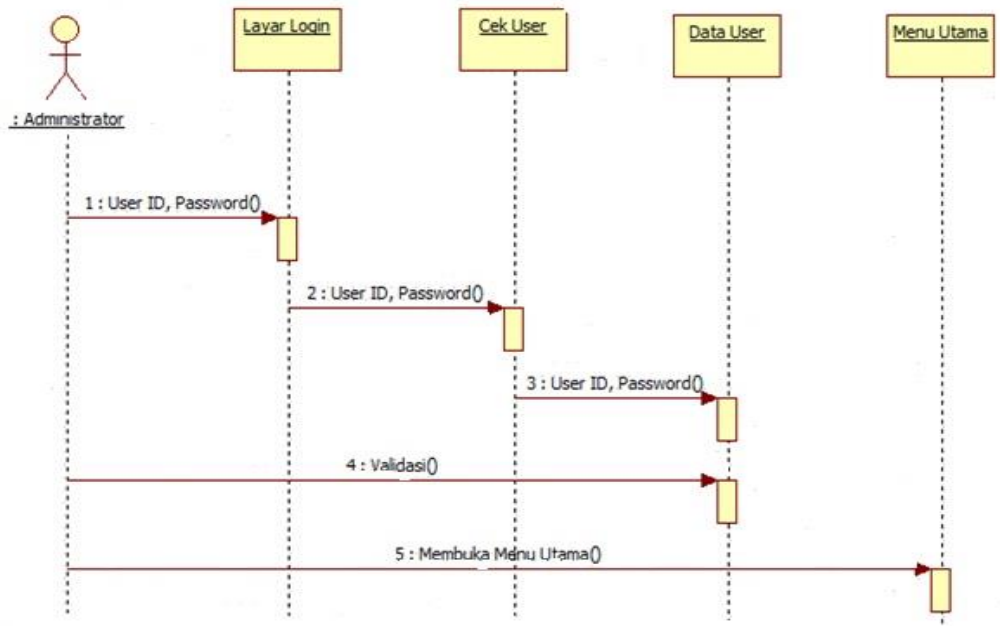
Gambar 2.4 Activity Diagram

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. *Class* diagram memiliki apa yang disebut atribut dan metode atau operasi. *Class* diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat di antara mereka. *Class* Diagram juga menunjukkan properti dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut.[13]



Gambar 2.5 Class Diagram

Sequence diagram merupakan salah satu diagram *interaction* yang menjelaskan bagaimana suatu operasi itu dilakukan, *message* (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu dan objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut. [13]



Gambar 2.6 *Sequence* diagram

2.8 RSVM (Reduced Support Vector Machines)

Support Vector Machine merupakan metode berbasis machine learning yang sangat menjanjikan untukdikembangkan karena memiliki performansi tinggi dan dapat diaplikasikan secara luas untuk klasifikasi dan estimasi. SVM memanfaatkan optimasi dengan quadratic programming, sehingga untuk data berdimensi tinggi dan berjumlah besar, SVM menjadi kurang efisien. Untuk mengatasi hal tersebut, dikembangkan Smooth Support Vector Machine (SSVM). Pada jumlah data yang besar SSVM juga tidak efisien kemudian dikembangkan Reduced Support Vector Machine (RSVM) yang melakukan klasifikasi dengan menggunakan sebagian karakteristik dari data yang dipilih secara random. [13]

Terdapat dua masalah besar dalam klasifikasi data besar yang non linier, yang pertama kesulitan komputasi dalam memecahkan masalah optimasi tanpa kendala dan melibatkan fungsi kernel yang membutuhkan memori sangat besar. Pada umumnya komputer mengalami *out of memory* bahkan sebelum dimulai proses pencarian solusi. Yang kedua kesulitan dalam menggunakan formula yang tidak terlihat, untuk bidang pemisah . untuk menyelesaikan masalah tersebut muncul sebuah ide menyelesakan bidang pemisah non linier pada data besar dengan hanya menggunakan sebagian karakteristik dari data. Hal inilah yang mengawali ide dasar dari RSVM. Dengan formulasi data penuh (*full set*) $A \in R^{\mathrm{mxn}}$ dengan *square kernel* $K(A, A^T) \in R^{\mathrm{mxn}}$ dimodifikasi sedemikian hingga *reduced* dataset $A \in R^{\mathrm{mxn}}$ dengan diagonal matriks D dan matriks kernel . Selanjutnya algoritma tersebut diselesaikan dengan *smoothing technique*. Hasil modifikasi tersebut dirumuskan dengan menggantikan $\mathbf{A}^{\mathrm{T}}$ dengan $\bar{\mathbf{A}}^{\mathrm{T}}$ sebagai berikut[3]

Diberikan masalah klasifikasi dari n objek dalam ruang dimensi R^{p} sehingga susunan data berupa matriks A berukuran n x p dan keanggotaan tiap titik terhadap kelas {+1} atau {-1} yang didefinisikan pada diagonal matriks D berukuran ,problem optimasinya adalah :

$$min_{w,b,\xi} \quad \frac{c}{2}\xi'\xi + \frac{1}{2}\,(w'w + b^2) \tag{2.10}$$

dengan kendala

$$D\,(Aw + eb) + \,\xi \;\geq e, \xi \;\geq\; 0 \tag{2.11}$$

Solusi problem adalah

$$\xi = \left(e - D(Aw + eb)\right) \quad (2.12)$$

Dimana ξ adalah variabel *slack* yang mengukur kesalahan klasifikasi. Kemudian dilakukan substitusi dan konversi, sehingga persamaan dapat ditulis sebagai berikut:

$$min_{w,b} \ \frac{c}{2}||(e - D(Aw - eb)||_2^2 + \frac{1}{2}\ (w'w + b^2) \quad (2.13)$$

Fungsi objektif dalam persamaan tidak memiliki turunan kedua. Teknik smoothing yang diusulkan dilakukan dengan mengganti fungsi plus dengan p(x,a) yaitu integral dari fungsi sigmoid neural network atau dapat dituliskan sebagai berikut:

$$p\ (x, a) = x + \frac{1}{a}\log(1 + \ \varepsilon^{-ax}), a > 0 \quad (2.14)$$

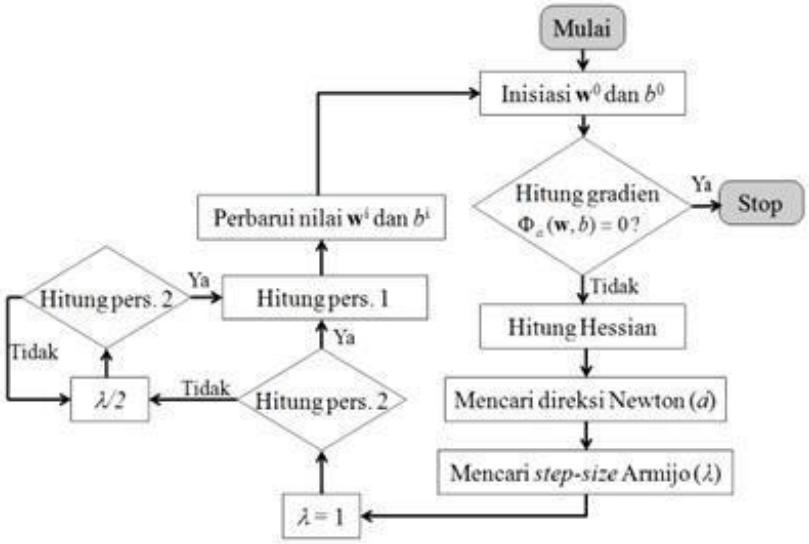
dimana α adalah parameter smoothing. Dengan menggantikan fungsi plus dengan p (x,a) maka diperoleh model SSVM sebagai berikut:

$$min_{(w,b)\in Rp+1}\phi_\alpha(w, b) = min_{(w,b)\in Rp+1}\ \frac{c}{2}||p(e - D(Aw - eb)||_2^2 + \frac{1}{2}\ (w'w + b^2)$$

Secara umum, problem optimasi dapat ditulis sebagai berikut:

$$\Psi_\alpha\ (w^i, y^i) \ = \frac{v}{2}\|p(e - D)(Aw - \gamma)\alpha \ \|_2^2 + \frac{1}{2}(W^TW + \gamma^2) \quad (2.16)$$

Yang diselesaikan dengan iterasi Newton Armijo (Gambar 2.7) dan $K(X_{i,}X'_j)$ merupakan fungsi kernel yang dalam penelitian ini digunakan kernel Gaussian atau bisa dirumuskan berikut $K(X_{i,}X'_j)$ = *exp* (-*y*($\|X_{i,}X'_j\|^2$) dengan parameter kernel y.



Gambar 2.7 Alur Algoritma Newton Armijo

Menghitung Gradien dengan persamaan (2.17) kemudian cek apakah nilai gradien = 0? Menggunakan persamaan (2.18) , jika nilai gradien lebih dari 0 , maka harus menghitung nilai hessian dengan persamaan (2.19) berikut.

$$\lim_{a\to\infty} \nabla\Psi_\alpha\,(w, y) \quad = \tag{2.17}$$

$$\begin{bmatrix} w - vA^T D\big(\, e - D(Aw - ey)\big) \\ y + ve^T D\big(\, e - D(Aw - ey)\big) \end{bmatrix}$$

$$\left\| \lim_{a\to\infty} \nabla\Psi_\alpha\,(w, y) \quad \right\|_2^2 \tag{2.18}$$

$$S_\infty(x) = \lim_{a\to\infty} \left(\frac{1}{1 + \varepsilon^{-ax}} \right) \quad = \frac{1 + sign(x)}{2} \tag{2.19}$$

Menentukan nilai direksi newton

$$d^i = \nabla^2\Psi_\alpha\,(w^i, y^i)^{-1}(-\Psi_\alpha\,(w^i, y^i) \tag{2.20}$$

Persamaan 1 :

$$\Psi_\alpha\,(w^i, y^i) \quad - \Psi_\alpha(w^i, y^i) + \lambda_i d^i \;\geq\; -\delta\lambda\nabla\Psi_\alpha(w^i, y^i)\; d^i \tag{2.21}$$

Persamaan 2 :

$$\Psi_\alpha(w^i, y^i) + \lambda_i d^i \tag{2.22}$$

Saat iterasi pada algoritma Newton-Armijo berhenti, diperoleh nilai w dan b yang konvergen. Dengan demikian fungsi pemisah yang diperoleh untuk kasus klasifikasi linier adalah :

$$\boldsymbol{f(x) = sign(g(x))} \tag{2.23}$$

Sedangkan fungsi permisah untuk kasus klasifikasi nonlinier adalah sebagai berikut:

$$F(x) = sign\,(\text{w'x} + \gamma) = sign\,(\text{u'}D'K(\bar{W}'\bar{W}) + \gamma) \tag{3.24}$$

Setelah menyelesaikan persamaan SSVM (2.35) yang telah dimodifikasi dan diselesaikan dengan algoritma *Newton-Armijo* dimana W' digantikan oleh $\bar{W}'$ dengan $\bar{D} \subset D$. Maka diperoleh model RSVM sebagai berikut.

$$\boldsymbol{\Psi_\alpha\,(w^i, y^i) = \frac{v}{2}\left\|p(e - D)(Aw - \gamma)\alpha\right\|_2^2 + \frac{1}{2}(\overline{W}^T\overline{W} + \gamma^2)} \tag{2.24}$$

2.9 Pengujian Sistem

Pengujian sistem adalah proses pemeriksaan atau evaluasi sistem atau komponen sistem secara manual atau otomatis untuk memverifikasi apakah sistem memenuhi kebutuhan-kebutuhan yang dispesifikasikan atau mengidentifikasi perbedaan-perbedaan antara hasil yang diterapkan dengan hasil yang terjadi [14]
Pengujian seharusnya meliputi tiga konsep berikut :

1. Demonstrasi validasi perangkat lunak pada masing-masing tahap diskusi pengembangan sistem.
2. Penentuan validitas sistem akhir dikaitkan dengan kebutuhan pemakai.
3. Pemeriksaan prilaku sistem dengan mengeksekusi sistem pada data sample pengujian.

Pada dasarnya pengujian diartikan sebagai aktivitas yang dapat atau hanya dilakukan setelah pengkodean (kode program selesai). Namun, pengujian seharusnya dilakukan dalam skala lebih luas. Pengujian dilakukan bagitu skesifikasi kebutuhan telah dapat didefinisikan. Evaluasi terhadap spesifikasi dan perancangan juga merupakan teknik pengujian. Kategori pengujian dapat dikategorikan menjadi dua, yaitu :

1. Berdasarkan ketersediaan *logic* sistem, terdiri dari *Black Box testing* dan *White Box testing.*
2. Berdasarkan arah pengujian, terdiri dari *top down* dan pengujian *Bottom up.*

2.9.1 Pengujian *Blackbox*

Konsep *black box* digunakan untuk mempresentasikan sistem yang cara kerja di dalamnya tidak tersedia untuk diinspeksi. Dalam *black box,* item-item yang diuji dianggap "gelap" karena logiknya tidak diketahui, yang diketahui hanya apa yang masuk dana pa yang keluar dari *black box.*

2.9.2 Pengujian Akurasi

Akurasi merupakan seberapa dekat suatu angka hasil pengukuran terhadap angka sebenarnya (*true value* dan *reference value*). Tingkat akurasi diperoleh dengan persamaan sebagai berikut [22] :

$$Akurasi = \frac{jumlah\ karakter\ sama}{jumlah\ seluruh\ karakter} \times 100\% \qquad (2.10)$$

2.10 Bahasa Pemrograman Java

Java merupakan sebuah Bahasa pemrograman berorientasi objek yang dapat berjalan pada platform yang berbeda, baik di Windows, Linux, serta system operasi lainnya. Dengan menggunakan Java kita dapat mengembangkan banyak aplikasi yang dapat digunakan pada lingkungan yang berbeda, seprti pada Desktop, Mobile, Internet dan lain-lain. Untuk menginstalasi dan menggunakan Java, Sun Micro System selaku pengembang Java menyediakan paket instalasi sesuai dengan kebutuhan kita dalam membangun suatu aplikasi. Berikut ini uraian singkat mengenai paket aplikasi Java yang tersedia.

1. J2ME (Java 2 Micro Edition) Paket instalasi ini dapat digunakan untuk membangun software yang berjalan pada perangkat yang memiliki memori dan sumber daya yang kecil, seperti pada handphone, PDA, dan Smartcard.

2. J2SE (Java 2 Standard Edition) Paket instalasi ini digunakan untuk mengembangkan aplikasi yang berjalan pada lingkungan workstation, seperti aplikasi desktop.

3. J2EE (Java 2 Enterprise Edition) Paket instalasi ini dapat digunakan untuk mengembangkan aplikasi pada lingkungan internet maupun aplikasi skala enterprise.

Java juga merupakan bahasa pemrograman resmi yang digunakan untuk pembangunan aplikasi android yang didukung penuh oleh Google. Namun meskipun demikian saat ini java bukanlah satu-satunya Bahasa yang dapat digunakan untuk membangun aplikasi Android seperti Xamarin dengan menggunakan Bahasa pemrograman C#, Cordova dengan menggunakan bahasa pemrograman web seperti HTML, CSS, dan Javascript dan lain-lain. [15].

2.11 Xampp

XAMPP adalah perangkat lunak bebas (free software), yang mendukung untuk banyak sistem operasi, yang merupakan kompilasi dari beberapa program. Fungsi XAMPP sendiri adalah sebagai server yang berdiri sendiri (localhost), yang terdiri beberapa program antara lain : Apache HTTP Server, MySQL database, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. Nama XAMPP sendiri merupakan singkatan dari X (empat sistem operasi apapun), Apache, MySQL, PHP dan Perl. Program ini tersedia dalam GNU General Public License dan bebas, merupakan web server yang mudah untuk digunakan yang dapat menampilkan halaman web yang dinamis. Untuk mendapatkanya XAMPP anda dapat mendownload langsung dari web resminya [16]. Pada penelitian ini XAMPP berguna sebagai web server dan database server.