BAB 2
LANDASAN TEORI

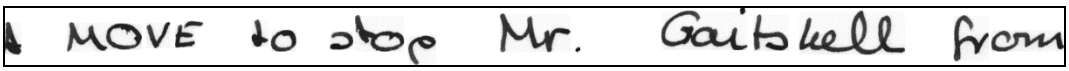
2.1 Grafologi

Grafologi adalah sebuah metode ilmiah yang dapat mengidentifikasi kondisi psikologis maupun kepribadian dari seseorang berdasarkan pola tulisan tangan [1]. Grafologi berasal dari bahasa Yunani, *graph* yang berarti tulisan atau menulis, dan *logos* yang berarti ilmu. Manfaat dari Ilmu Grafologi adalah dapat digunakan untuk mengetahui karakter dan kepribadian seseorang, kestabilan emosi, kondisi mental, serta potensi minat yang disukai seseorang. Grafologi juga bisa digunakan perekrut dari perusahaan untuk mengenal lebih jauh kandidat yang mereka seleksi dan juga bisa menjadi rekomendasi bagi seseorang untuk memilih jenis pekerjaan yang cocok untuk mereka. Saat ini, ada berbagai tes formal yang bisa digunakan untuk menemukan kepribadian seseorang seperti studi bakat (*aptitude*), tes psikometri, dan lainnya. Namun, dengan membandingkan kepraktisan dari berbagai metode diagnostik ini, telah ditemukan bahwa grafologi adalah cara tercepat [12]. Beberapa fitur yang bisa digunakan dalam ilmu grafologi di antaranya adalah tekanan penulisan, dominasi zona tulisan, garis dasar, kemiringan, ukuran huruf, jarak tulisan, dan margin tulisan.

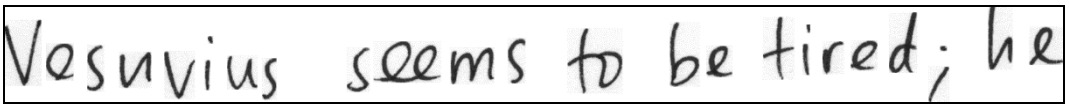
2.1.1 Tekanan Penulisan

Pola tekanan tulisan berhubungan dengan tingkat emosi seseorang. Orang dengan tekanan tulisan yang kuat memiliki tingkat emosional yang tinggi, sulit beradaptasi, selalu serius akan segala sesuatu, tegas, dan memiliki keinginan yang kuat. Contoh tulisan dengan tekanan kuat dapat dilihat pada Gambar 2.1. Orang dengan tekanan tulisan yang sedang memiliki kemampuan untuk mengontrol emosinya dengan baik, nyaman, dan tidak suka memendam kemarahan [13]. Contoh tulisan dengan tekanan sedang dapat dilihat pada Gambar 2.2. Sedangkan orang dengan tekanan tulisan yang ringan cenderung memiliki kepribadian yang tenang dan santai, lebih sensitif, pengertian, dan sulit mengambil keputusan

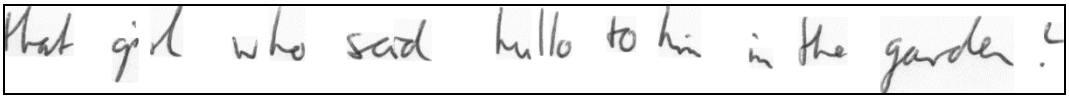
karena mudah terpengaruh [2]. Contoh tulisan dengan tekanan ringan dapat dilihat pada Gambar 2.3.



Gambar 2.1 Tekanan Tulisan Kuat



Gambar 2.2 Tekanan Tulisan Sedang



Gambar 2.3 Tekanan Tulisan Ringan

2.1.2 Dominasi Zona Tulisan

Dominasi zona tulisan berkaitan dengan aspek kehidupan seseorang. Seseorang dengan tulisan yang dominan zona atas lebih memperhatikan aspek spiritual, impian, harapan, dan ambisi dalam hidupnya. Mereka lebih suka melakukan kegiatan berpikir dan memikirkan masa depannya. Contoh tulisan dengan dominasi zona atas dilihat pada Gambar 2.4. Berbeda dengan seseorang dengan tulisan yang dominan zona tengah, mereka lebih mementingkan kehidupan mereka saat ini dan sulit untuk membuat rencana jangka panjang mereka. Contoh tulisan dengan dominasi zona tengah dilihat pada Gambar 2.5. Sedangkan seseorang dengan tulisan yang dominan zona bawah lebih mementingkan aspek fisik kehidupan dan lebih mengandalkan ototnya daripada otaknya [13]. Contoh tulisan dengan dominasi zona bawah dilihat pada Gambar 2.6.

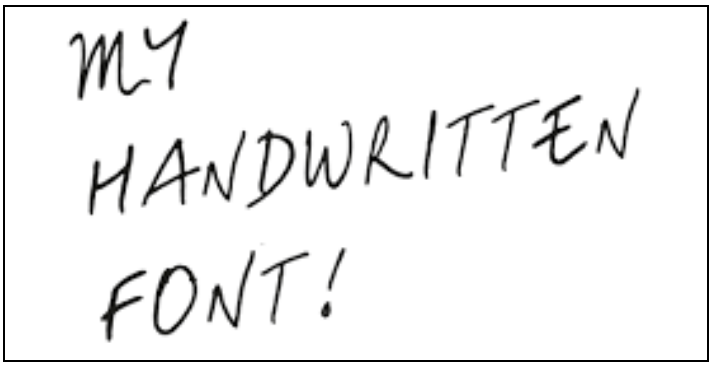
Gambar 2.4 Dominasi Zona Atas

Gambar 2.5 Dominasi Zona Tengah

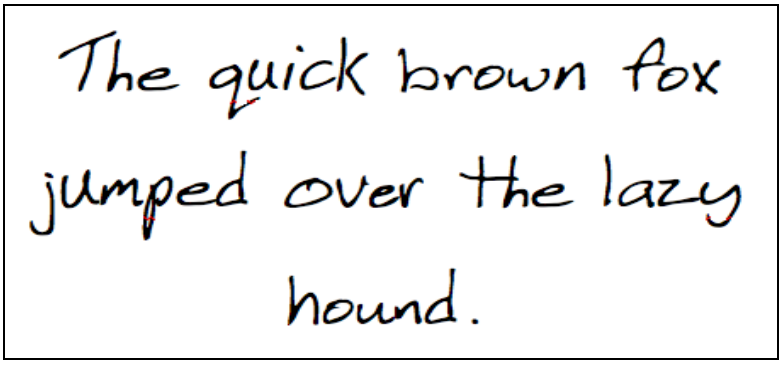
Gambar 2.6 Dominasi Zona Bawah

2.1.3 Garis Dasar (*Baseline*)

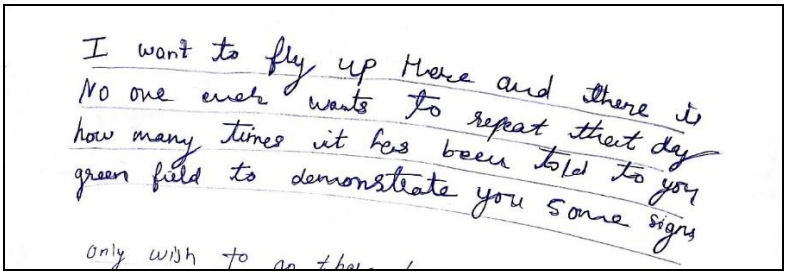
Karakter seseorang dengan pola garis dasar tulisan menaik menandakan bahwa orang tersebut memiliki sifat yang optimis, kegembiraan, aktif, dan memiliki semangat tinggi. Contoh tulisan dengan garis dasar menaik bisa dilihat pada Gambar 2.7. Seseorang dengan pola garis dasar tulisan lurus memiliki karakter yang terkontrol, memiliki stabilitas emosi, dan disiplin [14] [2]. Contoh gambar tulisan dengan garis dasar lurus dilihat pada Gambar 2.8. Sedangkan orang dengan pola garis dasar yang menurun memiliki sifat yang pesimis dan *over thinker* [14]. Contoh tulisan dengan garis dasar menurun dilihat pada Gambar 2.9.



Gambar 2.7 Garis Dasar Menaik



Gambar 2.8 Garis Dasar Lurus

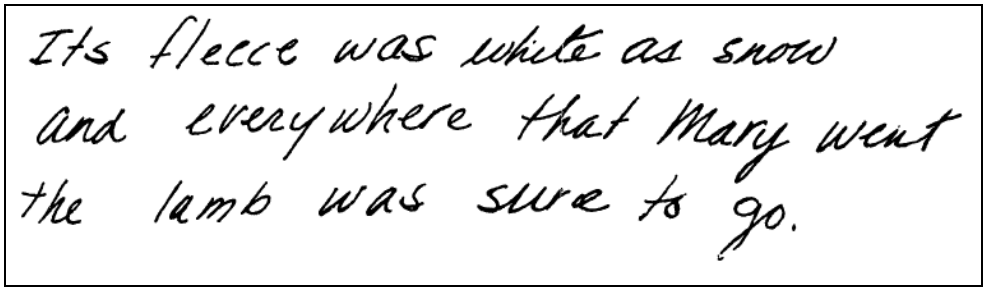


Gambar 2.9 Garis Dasar Menurun

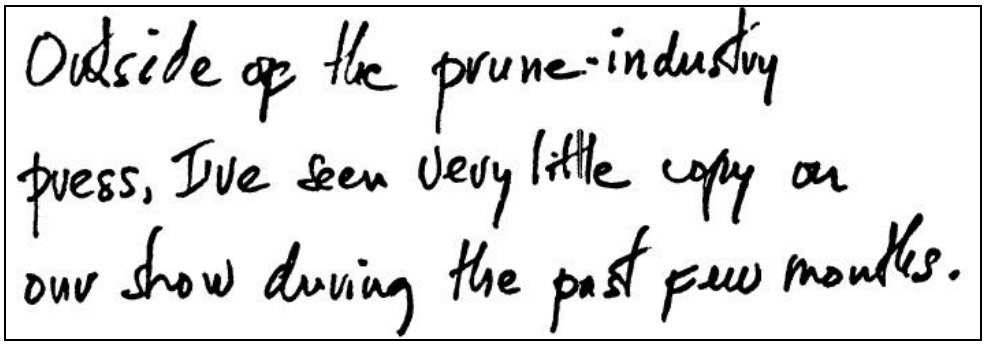
2.1.4 Kemiringan

Kemiringan tulisan mengungkapkan emosi, hubungan, komunikasi, dan kepribadian sosial seseorang [2]. Seseorang dengan pola tulisan yang miring ke kanan memiliki sifat yang ekspresif atau suka menunjukkan apa yang ia rasakan, mampu untuk berpikir dengan jernih, dan memiliki perasaan empati yang kuat. Contoh tulisan yang miring ke kanan dilihat pada Gambar 2.10. Tulisan yang

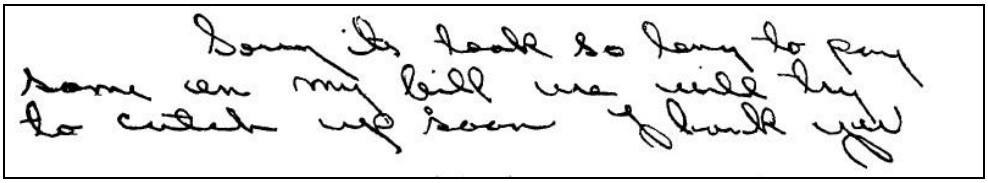
tegak menandakan bahwa orang tersebut lebih suka menggunakan akal dan logikanya untuk memutuskan sesuatu, jarang melibatkan perasaan. Contoh tulisan yang tegak dapat dilihat pada Gambar 2.11. Sedangkan tulisan yang miring ke kiri menandakan bahwa orang tersebut memiliki sikap yang diplomatis, tidak bisa terus terang, dan sulit beradaptasi [13]. Contoh tulisan yang miring ke kiri dilihat pada Gambar 2.12.



Gambar 2.10 Tulisan Miring Ke Kanan



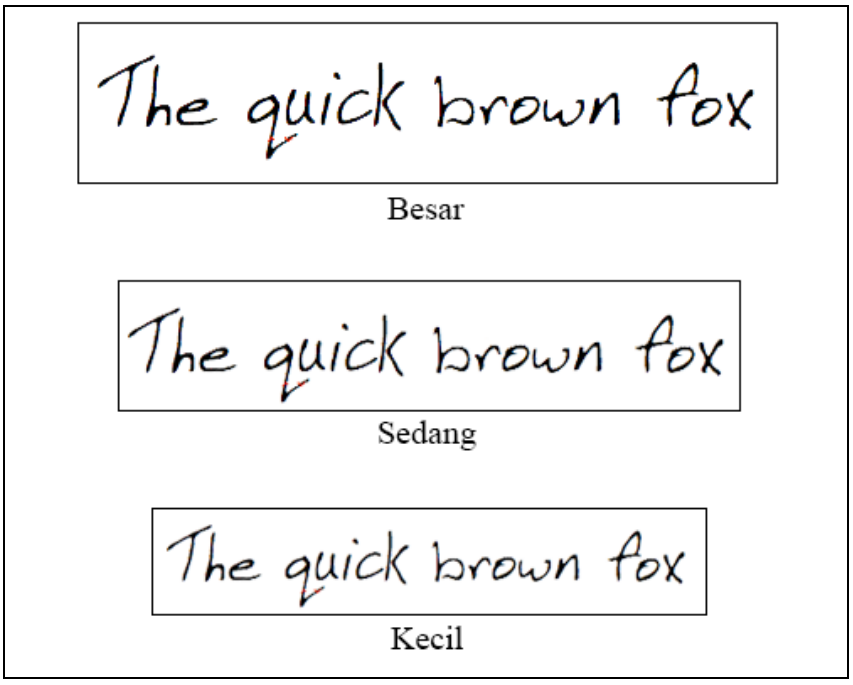
Gambar 2.11 Tulisan Tegak



Gambar 2.12 Tulisan Miring Ke Kiri

2.1.5 Ukuran Huruf

Ukuran huruf mewakili karakteristik seseorang terhadap inspirasi dirinya atas lingkungannya. Seseorang dengan ukuran huruf yang besar mempunyai sifat yang optimis, antusias, ekstrover, suka diperhatikan, dan menyukai dunia luar [2] [13]. Seseorang dengan ukuran huruf yang sedang mempunyai karakter yang nyaman dengan dirinya dan kehidupan sosialnya. Di mana pun dia berada akan tetap merasa nyaman dengan kehidupannya [13]. Sedangkan orang dengan ukuran huruf yang kecil memiliki sifat yang introvert, lebih suka menyendiri, dan tidak terlalu komunikatif kecuali dengan teman dekatnya [2] [13]. Contoh ukuran tulisan tangan bisa dilihat pada Gambar 2.13.

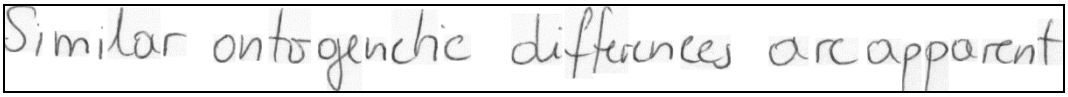


Gambar 2.13 Ukuran Huruf

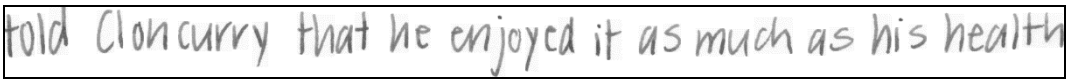
2.1.6 Jarak Tulisan (*Spacing*)

Jarak antar huruf menunjukkan bagaimana seseorang berhubungan dengan orang lain pada tingkat pribadi. Jarak antar huruf yang seimbang menandakan bahwa orang tersebut memiliki hubungan yang seimbang dan fleksibel dengan orang lain. Contoh tulisan dengan jarak antar kata yang seimbang dilihat pada

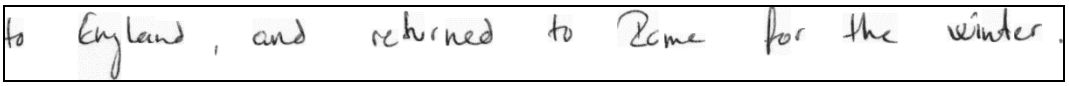
Gambar 2.14. Jika jarak antar hurufnya berdekatan maka menandakan orang tersebut sedang dalam keadaan tertekan, memiliki sifat tertutup dan berpandangan sempit, contoh tulisannya dapat dilihat pada Gambar 2.15. Sedangkan jika jarak antar hurufnya berjauhan menandakan orang tersebut memiliki karakter yang sangat terbuka tetapi dalam hatinya tegang dan hati-hati dengan perasaan sendiri [2]. Contoh tulisan dengan jarak antar kata yang jauh dilihat pada Gambar 2.16.



Gambar 2.14 Jarak Antar Kata Seimbang



Gambar 2.15 Jarak Antar Kata Dekat

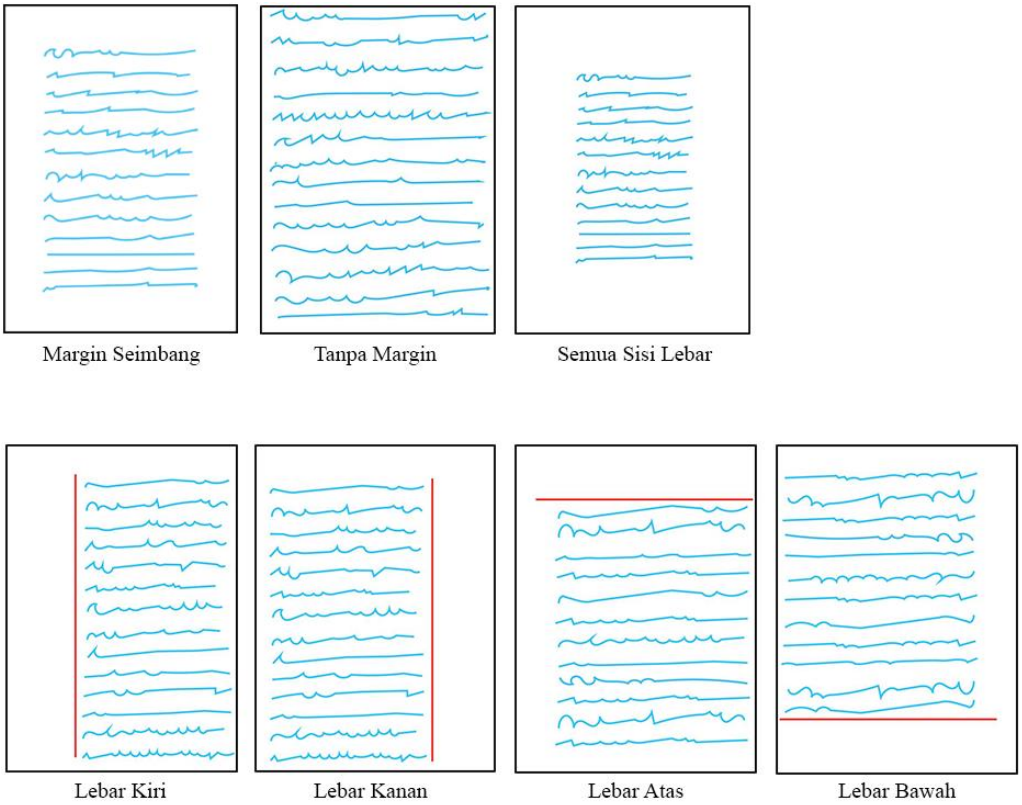


Gambar 2.16 Jarak Antar Kata Jauh

Jarak antar kata mewakili kebutuhan seseorang akan kenyamanan emosional dengan orang lain dan jarak yang ingin dipertahankan antara dirinya dan masyarakat. Jika jarak antar katanya seimbang maka menandakan bahwa orang tersebut memiliki karakter yang matang secara sosial, cerdas, dan memiliki kemampuan untuk menangani sesuatu secara objektif. Jika jarak antar katanya sangat dekat menandakan orang tersebut mendambakan kontak dan kedekatan yang konstan dengan orang lain dan kadang memiliki keegoisan dalam permintaannya. Sedangkan jika jarak antar katanya berjauhan menandakan orang tersebut lebih suka menjaga jarak dari kontak sosial dan lebih suka menjaga privasi [2].

2.1.7 Margin Tulisan

Seseorang dengan pola margin tulisan yang seimbang (rata pada kedua sisi) memiliki sifat yang rapi, tertib, terkontrol, suka memperhatikan penampilan dan sadar akan batasan sosial [2] [13]. Seseorang dengan pola tulisan yang tanpa adanya margin cenderung memiliki sifat yang hemat, efisien, dan sangat percaya diri, namun sulit mempercayai orang lain karena merasa tidak ada orang yang lebih hebat darinya. Seseorang dengan margin tulisan sebelah kiri yang lebar menandakan bahwa orang tersebut takut akan masa lalu yang ia alami, sebaliknya jika margin tulisan yang lebar ada di sebelah kanan menunjukkan bahwa orang tersebut kurang yakin atau tidak siap dengan masa depannya [13]. Jika margin tulisan pada bagian atas lebar menandakan bahwa orang tersebut memiliki sifat yang sederhana, formal, dan hormat terhadap sesama, jika margin yang lebar berada pada sisi bagian bawah menandakan bahwa orang tersebut memiliki sifat yang idealistis, pendiam dan suka menyendiri [2]. Sedangkan jika margin tulisannya lebar pada keempat sisi menandakan bahwa orang tersebut memiliki sifat yang tidak ingin dekat dengan orang lain dan suka bekerja sendirian [13]. Contoh pola margin tulisan dapat dilihat pada Gambar 2.17.



Gambar 2.17 Margin Tulisan

2.2 Citra Digital

Citra adalah kombinasi antara titik, garis, bidang, dan warna untuk menciptakan suatu tiruan dari sebuah objek, seperti manusia, hewan, tumbuhan, ataupun benda lainnya. Citra digital merupakan sebuah larik (*array*) yang berisi nilai-nilai real maupun kompleks yang direpresentasikan dengan deretan bit tertentu. Suatu citra dapat didefinisikan sebagai fungsi f(x,y) berukuran M baris dan N kolom, dengan x dan y adalah koordinat spasial, dan amplitudo f di titik koordinat (x,y) dinamakan intensitas atau tingkat keabuan dari citra pada titik tersebut [15]. Citra disusun oleh sekumpulan piksel-piksel yang mempunyai dua buah parameter yaitu koordinat yang sesuai posisinya pada citra dan nilai intensitas warna. Pada umumnya, citra terbagi menjadi tiga jenis yaitu citra biner, citra *grayscale*, dan citra warna [16].

a. Citra Warna

Citra warna adalah citra yang pada masing-masing pikselnya mempunyai tiga komponen warna yang spesifik yaitu merah (*red*), hijau (*green*), dan biru (*blue*). Warna setiap piksel ditentukan oleh kombinasi dari intensitas warna merah, hijau, dan biru [16]. Setiap piksel citra warna diwakili oleh 24 bit dengan nilai komponen warna masing-masing adalah 8 bit sehingga total kemungkinan kombinasi warna mencapai 16.777.216 kemungkinan. Contoh citra warna dapat dilihat pada Gambar 2.18.

Gambar 2.18 Citra Warna

b. Citra *Grayscale*

Citra *grayscale* adalah citra yang nilai intensitas warna pada pikselnya didasarkan pada derajat keabuan. Citra *grayscale* hanya memiliki satu nilai kanal pada setiap pikselnya (Red = Green = Blue) yang digunakan untuk menunjukkan tingkat intensitas [15]. Setiap piksel pada citra *grayscale* membutuhkan 8 bit memori dimana nilai intensitas setiap pikselnya berkisar antara 0 sampai 255 [16]. Citra *grayscale* tidak memiliki warna lain selain variasi intensitas putih dan hitam, hal ini digunakan untuk menyederhanakan model citra supaya bisa diolah oleh komputer. Contoh citra *grayscale* dapat dilihat pada Gambar 2.19.

Gambar 2.19 Citra *Grayscale*

Untuk mengubah citra berwarna (RGB) menjadi citra *grayscale* dapat dilakukan dengan menggunakan salah satu rumus berikut ini :

$$grayscale = (0.2989 * R) + (0.5870 * G) + (0.1141 * B) \quad \textbf{(2.1)}$$

Di mana R adalah nilai warna merah, G nilai warna hijau, dan B adalah nilai warna biru dari piksel citra.

c. Citra Biner

Citra biner adalah jenis citra yang hanya memiliki dua kemungkinan pada nilai pikselnya yaitu antara hitam (0) atau putih (1) [15]. Pada citra biner, setiap piksel hanya membutuhkan 1 bit memori. Contoh citra biner dapat dilihat pada Gambar 2.20.

Gambar 2.20 Citra Biner

Untuk mengonversi citra *grayscale* menjadi citra biner dapat dilakukan dengan proses *thresholding.* Pada prosesnya dibutuhkan satu nilai ambang sebagai nilai pembatas di mana jika nilai intensitas piksel pada citra lebih besar atau sama dengan nilai ambang ini maka akan dikonversi menjadi 1 (putih), sedangkan jika nilai intensitas pikselnya kurang dari nilai ambang maka akan dikonversi menjadi 0 (hitam).

2.3 Pengolahan Citra Digital

Pengolahan citra digital adalah sebuah cabang ilmu teknologi informasi yang menyangkut manipulasi citra untuk mengekstraksi informasi untuk menekankan atau mengurangi penekanan aspek tertentu dari informasi yang terkandung dalam citra, atau melakukan analisis statistik atau lainnya untuk mengekstraksi informasi non-citra [17]. Pengolahan citra dilakukan dengan cara mengolah piksel-piksel di dalam citra digital dengan tujuan untuk memperbaiki kualitas citra atau mengubah ke bentuk citra yang lain. Pengolahan citra digital dapat diterapkan pada berbagai bidang seperti bidang biomedis, penginderaan jarak jauh, biometrika, fotografi, desain visual, volumetrik, mengidentifikasi suatu objek pada citra, pengenalan

karakter optik, dan *image retrieval* atau *image querying*. Pengolahan citra digital terbagi ke dalam tiga kategori atau tingkat pengolahan [15], yaitu :

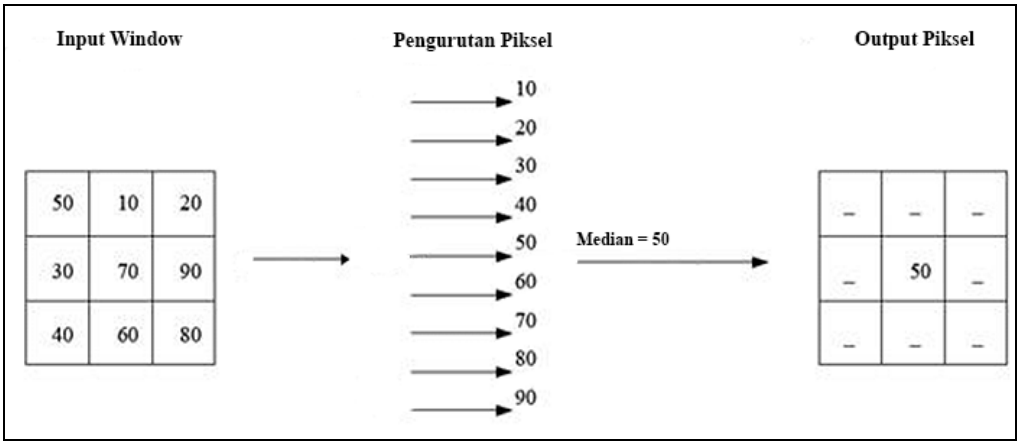
a. Pengolahan Tingkat Rendah (*Low-Level Processing*).
   Pengolahan tingkat rendah melibatkan operasi-operasi dasar dalam pengolahan citra, seperti pengurangan derau (*noise reduction*), praprosesing citra, penambahan atau pengurangan kontras, dan pengaturan ketajaman citra.
b. Pengolahan Tingkat Menengah (*Mid-Level Processing*).
   Pengolahan tingkat menengah meliputi operasi-operasi seperti segmentasi citra dan klasifikasi citra. Proses pada pengolahan ini melibatkan input citra dan output berupa atribut citra yang dipisahkan dari citra input.
c. Pengolahan Tingkat Tinggi (*High-Level Processing*).
   Pengolahan tingkat tinggi meliputi proses pengenalan, deskripsi, dan analisis citra.

2.4 Median Filter

Median filter merupakan filter non linier yang berfungsi untuk mengurangi derau atau gangguan pada citra dan juga menghaluskan citra [18]. Operasi non linier dihitung dengan cara mengurutkan nilai intensitas sekumpulan piksel, kemudian menggantikan nilai piksel yang diproses dengan nilai tertentu. Pada median filter suatu *jendela* atau penapis yang memuat sejumlah piksel ganjil digeser titik per titik pada seluruh daerah citra. Nilai-nilai yang berada pada *window* diurutkan secara *ascending* untuk kemudian dihitung nilai mediannya. Nilai tersebut akan menggantikan nilai yang berada pada pusat bidang *window*. Median filter menggunakan *mask* berukuran 3x3 atau 5x5. Untuk mendapatkan nilai median dapat dilakukan dengan persamaan berikut :

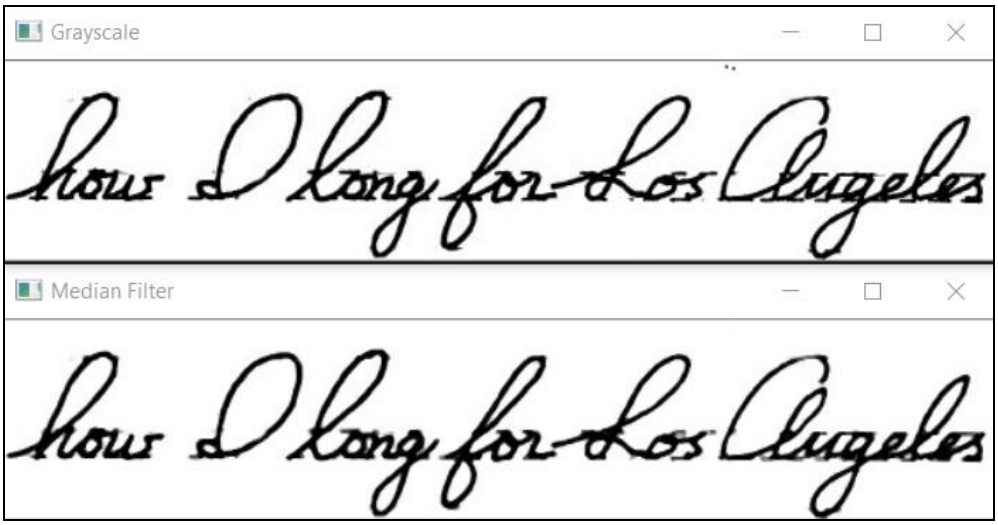
$$g(x,y) = Median\{f(x-i, y-j), (i,j) \in w\} \quad \textbf{(2.2)}$$

di mana *g(x,y)* merupakan citra yang dihasilkan dari citra *f(x,y)* dengan *w* sebagai *window* yang ditempatkan pada bidang citra serta (i,j) adalah elemen dari *window* tersebut. Ilustrasi proses *filtering* dengan metode median filter dapat dilihat pada Gambar 2.21.



Gambar 2.21 Proses Median Filter

Contoh hasil median filter dengan *mask* berukuran 3x3 pada citra tulisan tangan dapat dilihat pada Gambar 2.22.



Gambar 2.22 Hasil Median Filter

2.5 Segmentasi Citra

Segmentasi citra adalah proses pembagian daerah pada citra menjadi bagian-bagian daerah yang lebih kecil berdasarkan letak piksel dan intensitasnya yang masih berdekatan. Tujuan dari proses segmentasi adalah untuk membagi citra menjadi daerah yang homogen dan konsisten yang merepresentasikan sebuah objek yang berbeda dalam citra [17]. Segmentasi citra biasanya digunakan untuk

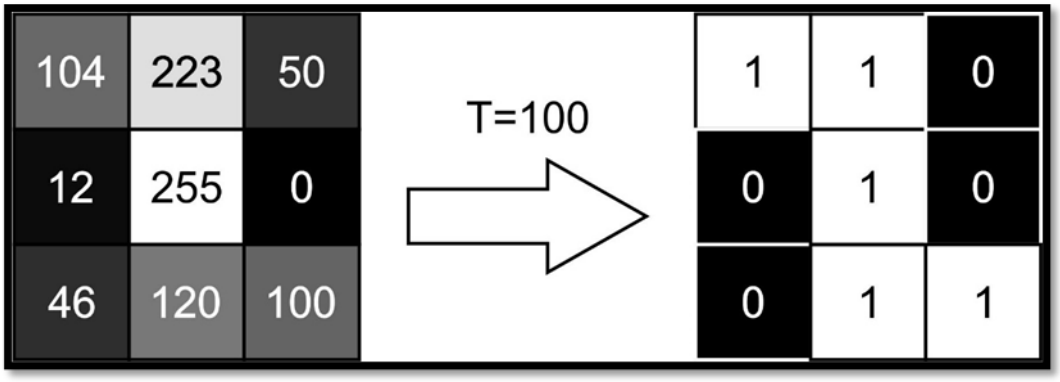
menemukan objek dan batas garis atau kurva dalam citra. Hasil segmentasi citra adalah seperangkat segmen yang secara kolektif mencakup seluruh citra, atau seperangkat kontur yang diekstraksi dari citra. Ada beberapa metode yang bisa digunakan dalam proses segmentasi citra, di antaranya adalah *thresholding* dan *edge detection*.

2.5.1 Ambang Batas (*Thresholding*)

Ambang batas (*thresholding*) merupakan salah satu metode segmentasi citra yang memisahkan antara objek dengan *background* dalam suatu citra berdasarkan pada perbedaan tingkat kecerahannya. Bagian citra yang cenderung gelap akan dibuat semakin gelap dengan nilai intensitas 0, sedangkan bagian citra yang cenderung terang akan dibuat semakin terang dengan nilai intensitas 1. Keluaran dari proses *thresholding* ini adalah citra biner atau hitam putih dengan nilai intensitas antara 0 dan 1, sehingga dapat diketahui daerah mana yang termasuk objek dan *background* dari citra secara jelas. Persamaan yang dapat digunakan untuk mengubah citra *grayscale* menjadi citra biner dengan *thresholding* adalah sebagai berikut :

$$g(x,y) = \begin{cases} 1, & jika\ f(x,y) \geq T \\ 0, & jika\ f(x,y) < T \end{cases} \qquad \textbf{(2.3)}$$

Dimana g(x,y) adalah citra biner, f(x,y) adalah nilai piksel pada citra *grayscale* dan T adalah nilai ambang batas (*threshold*). Ilustrasi dari proses metode *thresholding* dapat dilihat pada Gambar 2.23.



Gambar 2.23 Proses Thresholding

Namun proses thresholding dengan cara seperti di atas memiliki kekurangan, yaitu nilai ambang batasnya harus ditetapkan secara manual dan hasilnya belum tentu optimal untuk semua citra yang diolah. Untuk mengatasi masalah ini maka proses *thresholding* menggunakan metode otsu yang dapat menentukan nilai ambang secara otomatis pada setiap citra yang diolah. Metode otsu bekerja dengan cara membagi histogram citra grayscale menjadi dua daerah yang berbeda secara otomatis tanpa ada bantuan dari pengguna untuk menetapkan nilai ambang [19]. Nilai ambang batas yang akan dicari dari citra *grayscale* dinyatakan dengan k yang bernilai antara 1 sampai dengan L, dengan nilai L = 255. Probabilitas setiap piksel pada level ke i dapat dinyatakan dengan persamaan berikut :

$$p_i = n_i / N \quad \textbf{(2.4)}$$

Dengan n_i menyatakan jumlah piksel pada level ke i dan N menyatakan total jumlah piksel pada citra. Nilai jumlah kumulatif, rerata kumulatif, dan rerata intensitas global berturut-turut dapat dinyatakan dengan rumus sebagai berikut.

$$\omega(k) = \sum_{i=1}^{k} p_i \quad \textbf{(2.5)}$$

$$\mu(k) = \sum_{i=1}^{k} i \cdot p_i \quad \textbf{(2.6)}$$

$$\mu_T = \mu(L) = \sum_{i=1}^{L} i \cdot p_i \qquad \textbf{(2.7)}$$

Dan nilai ambang k dapat ditentukan dengan menggunakan persamaan sebagai berikut :

$$\sigma_B^2(K^*) = \max_{1 \le k < L} \sigma_B^2(k) \qquad \textbf{(2.8)}$$

dengan

$$\sigma_B^2(k) = \frac{[\mu_T \omega(k) - \mu(k)]^2}{\omega(k)[1 - \omega(k)]} \qquad \textbf{(2.9)}$$

Contoh hasil dari proses *thresholding* dengan metode otsu dapat dilihat pada Gambar 2.24.

Gambar 2.24 Hasil Otsu Thresholding

2.6 Ekstraksi Ciri (*Feature Extraction*)

Ekstraksi ciri (*feature extraction*) merupakan suatu pengambilan ciri (*feature*) atau karakteristik dari suatu objek citra yang digunakan sebagai pembeda dari objek-objek lainnya [16]. Ciri atau karakteristik inilah yang digunakan sebagai parameter untuk menggambarkan sebuah objek. Nilai dari parameter-parameter inilah yang nantinya akan digunakan sebagai data masukan pada proses klasifikasi citra. Ekstraksi ciri dapat dilakukan setelah tahapan segmentasi citra (memisahkan antara objek dengan background) maupun tanpa segmentasi citra. Ekstraksi ciri dilakukan dengan cara menghitung jumlah titik atau piksel yang ditemui dalam setiap pengecekan, di mana pengecekan dilakukan dalam berbagai arah pengecekan pada koordinat kartesian dari citra digital yang dianalisis, yaitu vertikal, horizontal, diagonal kanan, dan diagonal kiri.

2.6.1 Ekstraksi Ciri Tekanan Tulisan

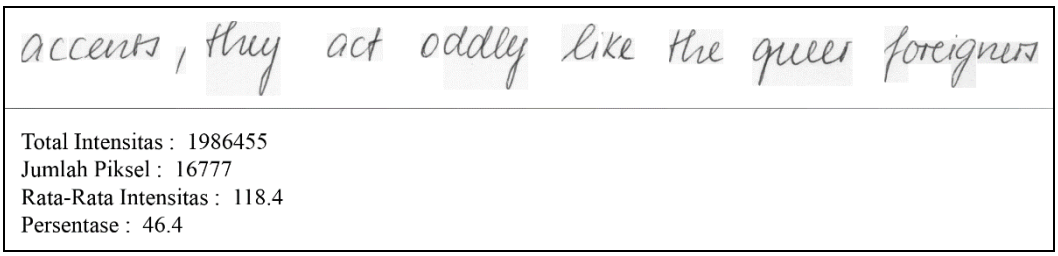
Ekstraksi ciri untuk mendapatkan nilai ciri tekanan tulisan pada citra dapat dilakukan dengan menggunakan algoritma *handwritten text feature extraction* dari Mukherjee dan Ishita [20]. Ciri tekanan tulisan yang didapat menggunakan algoritma tersebut adalah rata-rata intensitas piksel dari citra *grayscale* yang nilai intensitasnya kurang dari 150, serta nilai persentase dari nilai rata-rata tersebut. Berikut adalah algoritma yang digunakan untuk mendapatkan nilai ekstraksi ciri tekanan tulisan : [20]

1. Input citra *grayscale* yang sudah melalui tahap median filter.
2. Ambil nilai baris dan kolom dari citra, misalkan m = jumlah baris dan n = jumlah kolom.
3. for i = 1 to m
   for j = 1 to n
   if (grayscale(i,j) < 150)
   total_intensitas = total_intensitas + grayscale(i,j)
   jumlah_piksel = jumlah_piksel + 1
   end if
   end for

end for

4. rerata = total_intensitas / jumlah_piksel
5. persentase = (rerata * 100) / 255

Contoh hasil penerapan metode ekstraksi ciri tekanan tulisan dapat dilihat pada Gambar 2.25.



Gambar 2.25 Contoh Hasil Ekstraksi Ciri Tekanan Tulisan

Dari data hasil ekstraksi ciri tersebut, nilai rata-rata intensitas dan persentase akan digunakan pada proses klasifikasi.

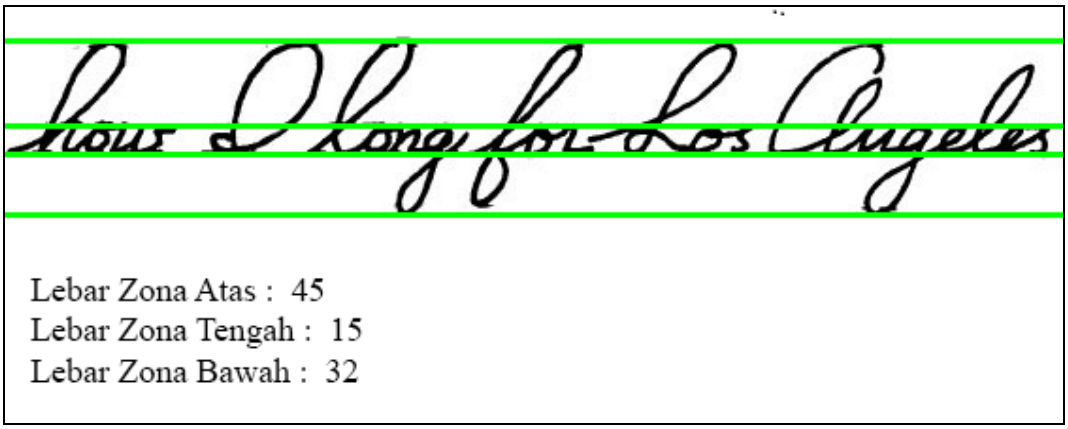
2.6.2 Ekstraksi Ciri Dominasi Zona Tulisan

Ekstraksi ciri untuk mendapatkan nilai ciri dominasi zona tulisan juga akan dilakukan menggunakan algoritma *handwritten text feature extraction* dari Mukherjee dan Ishita [20]. Ciri atau karakteristik dari dominasi zona tulisan yang akan digunakan pada proses klasifikasi adalah lebar daerah piksel dari tiga zona tulisan yaitu zona atas, tengah dan bawah. Lebar daerah ketiga zona tersebut didapatkan dengan mencari 4 koordinat y pada citra yang menjadi pembatas antar zona. Berikut adalah algoritma yang digunakan untuk mendapatkan nilai ekstraksi ciri dominasi zona tulisan : [20]

1. Input citra biner yang sudah melalui tahap *preprocessing*.
2. Hitung jumlah piksel hitam di setiap baris.
3. Temukan nilai frekuensi horizontal tertinggi dari hasil perhitungan sebelumnya.
4. Hitung 1/3 dari frekuensi horizontal maksimum. Nilai ini disebut T.

5. Cari baris pertama dari atas yang nilai frekuensinya tidak nol. Baris ini dinamakan top.
6. Cari baris pertama dari bawah yang nilai frekuensinya tidak nol. Baris ini dinamakan bottom.
7. Lintasi citra dari baris bawah ke baris atas dan periksa di mana jumlah piksel hitam dari baris yang melebihi nilai T. Nomor baris ini dimanakan r1.
8. Lintasi citra dari baris atas ke baris bawah dan periksa di mana jumlah piksel hitam dari baris yang melebihi nilai T. Nomor baris ini dinamakan r2.
9. Hitung lebar zona atas dengan r2 – top.
10. Hitung lebar zona tengah dengan r1– r2.
11. Hitung lebar zona bawah dengan bottom – r1.

Contoh hasil penerapan metode ekstraksi ciri dominasi zona tulisan dapat dilihat pada Gambar 2.26.



Gambar 2.26 Contoh Hasil Ekstraksi Ciri Dominasi Zona

Dari data hasil ekstraksi ciri tersebut, nilai-nilai lebar ketiga zona tulisan akan digunakan pada proses klasifikasi.

2.7 Pembelajaran Mesin (*Machine Learning*)

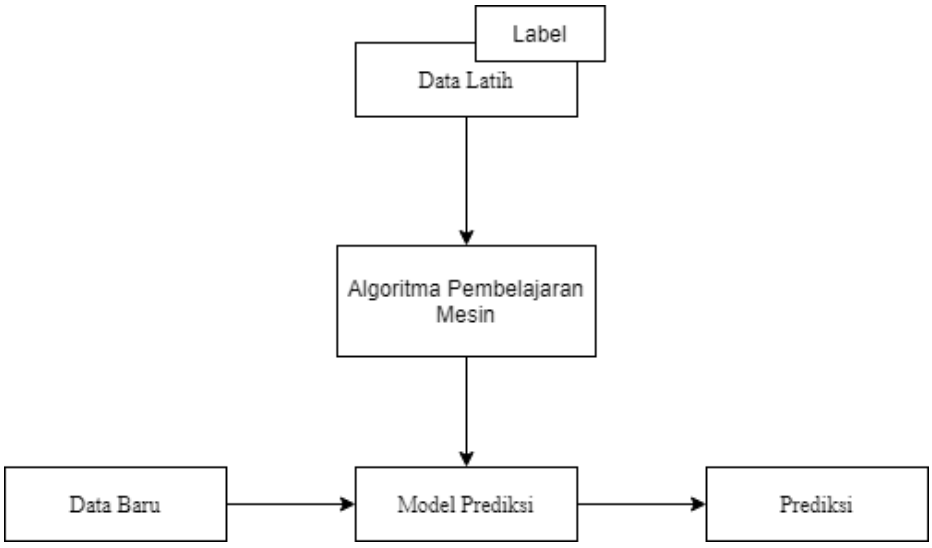
Pembelajaran mesin (*machine learning*) adalah salah satu sub bidang dari kecerdasan buatan (*artificial intelligence*) yang konsepnya berupa memberikan kemampuan kepada komputer untuk belajar secara mandiri dari sekumpulan data

yang diberikan menggunakan algoritma dan model statistik untuk membuat prediksi [21]. Fokus utama dari pembelajaran mesin adalah menemukan sebuah pola yang tepat dalam suatu kumpulan data. Pola tersebut yang nantinya akan digunakan dalam melakukan prediksi atau keputusan tanpa diprogram secara eksplisit untuk melakukan tugas tersebut. Beberapa manfaat dari pembelajaran mesin adalah dapat digunakan untuk melakukan prediksi harga barang, klasifikasi tumbuhan, mendeteksi spam pada *email*, merekomendasikan sebuah produk, bahkan mengenali wajah manusia.

2.7.1 Pembelajaran Terarah (*Supervised Learning*)

Pembelajaran terarah (*supervised learning*) adalah salah satu metode dari pembelajaran mesin yang menggunakan data latih yang sudah diberi label dalam proses pembelajarannya untuk melakukan prediksi pada data baru atau data yang akan datang [21]. Pembelajaran terarah banyak digunakan dalam memprediksi pola di mana pola tersebut sudah ada contoh data yang lengkap, sehingga pola yang terbentuk adalah hasil pembelajaran data lengkap tersebut. Proses dari pembelajaran terarah dapat dilihat pada

Gambar 2.27. Salah satu contohnya adalah memfilter *email* dengan melatih suatu model menggunakan algoritma pembelajaran mesin terarah pada kumpulan *email* berlabel dengan *email* yang ditandai sebagai spam atau bukan spam, untuk memprediksi apakah *email* masuk yang baru adalah salah satu dari dua kategori tersebut. Tugas dari pembelajaran terarah dengan label kelas diskrit seperti dalam contoh penyaringan spam *email* sebelumnya, juga disebut tugas klasifikasi.



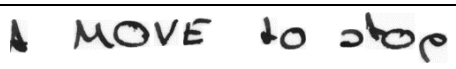
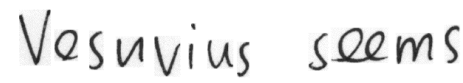
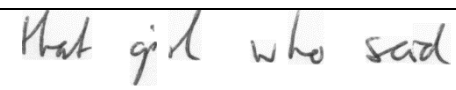
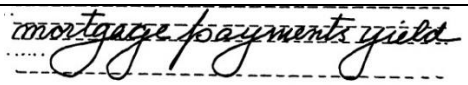
Gambar 2.27 Proses Pembelajaran Terarah

2.7.1.1 Klasifikasi

Menurut Towa P. Hamakonda dan J.N.B. Tairas (1983) klasifikasi adalah pengelompokan yang sistematis dari objek, gagasan, buku atau benda-benda lain ke dalam kelas atau golongan tertentu berdasarkan ciri-ciri yang sama. Dalam

konsep pembelajaran mesin, klasifikasi adalah sub kategori pembelajaran terarah di mana tujuannya adalah untuk memprediksi label kelas kategorikal dari data baru berdasarkan kumpulan data sebelumnya [21]. Label kelas tersebut adalah nilai diskrit dan tidak berurutan yang dapat dipahami sebagai anggota dari grup data yang sudah ada. Klasifikasi memprediksi hasil dari data masukan yang diberikan di mana keluaran dari teknik ini berupa kategori-kategori seperti laki-laki atau perempuan, panjang atau pendek, besar atau kecil, dan lain-lain. Pada penelitian ini, data masukan yang akan diklasifikasi adalah citra tulisan tangan di mana hasil keluaran adalah label kelas dari tulisan tangan tersebut. Penggolongan kelas tulisan tangan dapat dilihat pada Tabel 2.1.

Tabel 2.1. Fitur Tulisan Tangan

Fitur	Kelas	Contoh Tulisan
Tekanan Tulisan	Tekanan Kuat	
	Tekanan Sedang	
	Tekanan Ringan	
Dominasi Zona Tulisan	Dominan Atas	
	Dominan Tengah	
	Dominan Bawah	

2.8 Smooth Support Vector Machine (SSVM)

Smooth Support Vector Machine (SSVM) merupakan metode pengembangan dari *Support Vector Machine* (SVM) dengan menggunakan teknik *smoothing*. SSVM pertama kali diperkenalkan oleh Lee pada tahun 2001. SVM

memanfaatkan optimasi dengan *quadratic programming*, sehingga menjadi kurang efisien saat bekerja dengan data berdimensi tinggi dan berjumlah besar. Oleh karena itu dikembangkan teknik *smoothing* yang menggantikan *plus function* SVM dengan integral dari fungsi sigmoid *neural network* yang selanjutnya dikenal dengan SSVM. Permasalahan SVM yang dimodifikasi dari pengembangan ini adalah sebagai berikut.

$$\min_{w,\gamma,y} \frac{v}{2} y'y + \frac{1}{2}(w'w + \gamma^2) \qquad \textbf{(2.10)}$$

dengan kendala $D(Aw - e\gamma) + y \geq e$

$y \geq 0$

Solusi dari permasalahan ((2.10) adalah

$$y = (e - D(Aw - e\gamma))_+ \qquad \textbf{(2.11)}$$

Dimana y adalah variabel *slack* yang mengukur kesalahan klasifikasi. Kemudian dilakukan substitusi dan konversi, sehingga persamaan (2.10 dapat ditulis sebagai berikut:

$$\min_{w,\gamma} \frac{v}{2} \left\| \left(e - D(Aw - e\gamma)\right)_+ \right\|_2^2 + \frac{1}{2}(w'w + \gamma^2) \qquad \textbf{(2.12)}$$

Fungsi objektif pada persamaan (2.16 tidak memiliki turunan kedua. Teknik *smoothing* yang diusulkan dilakukan dengan mengganti *plus function* dengan *p(x,a)* yaitu integral dari fungsi sigmoid *neural network* yang dapat dituliskan sebagai berikut.

$$p(x, \alpha) = x + \frac{1}{\alpha} log(1 + exp(-\alpha x)) \qquad \textbf{(2.13)}$$

Fungsi *p* dengan parameter smoothing α ini digunakan untuk menggantikan *plus function* dari persamaan (2.13 untuk mendapatkan model SSVM sebagai berikut.

$$\min_{(w,\gamma)\in R^{n+1}} \frac{v}{2} \|p(e - D(Aw - e\gamma), \alpha)\|_2^2 + \frac{1}{2}(w'w + \gamma^2) \qquad \textbf{(2.14)}$$

Permasalahan SSVM diatas dapat diselesaikan dengan metode Newton–Armijo. Algoritma Newton–Armijo untuk menyelesaikan masalah SSVM adalah sebagai berikut [13].

Dimulai dengan menginisialisasi nilai $(w^0, \gamma^0) \in R^{nx1}$. Selanjutnya adalah mengulanginya hingga nilai gradien dari fungsi objektif (2.14 sama dengan 0 atau $\nabla\phi_\alpha\,(w^i, \gamma^i) = 0$. Jika nilai gradien tersebut belum 0, maka hitung (w^i, γ^i) sebagai berikut.

i. Newton *Direction* : menentukan nilai direction $d^i \in R^{nx1}$ dengan persamaan berikut:

$$\nabla^2\phi_\alpha\,(w^i, \gamma^i)d^i = -\nabla\phi_\alpha\,(w^i, \gamma^i)' \qquad \textbf{(2.15)}$$

ii. Armijo *Stepsize* : memilih nilai *stepsize* $\lambda_i \in R$ sehingga

$$(w^{i+1}, \gamma^{i+1}) = (w^i, \gamma^i) + \lambda_i d^i \qquad \textbf{(2.16)}$$

dimana $\lambda_i = \max\{1, \frac{1}{2}, \frac{1}{4}, \dots\}$ sehingga

$$\phi_\alpha(w^i, \gamma^i) - \phi_\alpha((w^i, \gamma^i) + \lambda_i d^i) \geq -\delta\lambda_i \nabla\phi_\alpha(w^i, \gamma^i)d^i \qquad \textbf{(2.17)}$$

dimana $\delta \in (0, \frac{1}{2})$.

Saat iterasi dari algoritma Newton-Armijo berhenti, diperoleh nilai *w* dan *b* yang konvergen.

Dalam menyelesaikan permasalahan non linear, dapat dilakukan dengan cara menambahkan *kernel trick*, salah satunya adalah kernel *radial basis function* (RBF) dengan persamaan sebagai berikut.

$$K(A, A') = \exp(-gamma\,\|A - A'\|^2) \qquad \textbf{(2.18)}$$

Dengan *gamma* merupakan parameter kernel. Sehingga model SSVM dengan *kernel* non linear adalah sebagai berikut.

$$\min_{u,\gamma} \frac{v}{2}||p(e - D(K(A, A')Du - e\gamma), \alpha)||_2^2 + \frac{1}{2}(u'u + \gamma^2) \qquad \textbf{(2.19)}$$

Dimana $K(A, A')$ merupakan peta *kernel* dari R^{mxn} x R^{nxm} menjadi R^{mxm}. Pada permasalahan SSVM non linear diatas, masih terdapat 2 kesulitan. Pertama, masalah (2.19 adalah masalah dalam variabel *m* + 1, di mana *m* bisa menjadi susunan jutaan untuk dataset besar. Kedua, permukaan pemisah nonlinear yang dihasilkan tergantung pada seluruh dataset yang diwakili oleh matriks A. Ini menciptakan kesulitan penyimpanan untuk dataset yang sangat besar dan membuat penggunaan *kernel* nonlinear tidak praktis untuk masalah seperti itu. Untuk menghindari dua kesulitan ini, maka perhatian dialihkan ke *Reduced Support Vector Machine* (RSVM).

2.9 Reduced Support Vector Machine (RSVM)

Algoritma *Reduced Support Vector Machine* (RSVM) pertama kali diperkenalkan oleh Lee dan Mangasarian pada tahun 2001 dengan tujuan untuk mengatasi permasalahan kesulitan komputasi serta untuk mengurangi kompleksitas model saat bekerja dengan *dataset* yang berjumlah sangat besar [23]. Konsep dasar dari algoritma ini adalah melakukan proses klasifikasi dengan hanya menggunakan sebagian data yang dipilih secara acak. RSVM merupakan algoritma yang menghasilkan permukaan pemisah berbasis kernel non linier yang membutuhkan sedikitnya 1% dari keseluruhan *dataset* untuk evaluasi eksplisitnya [8]. Untuk menghasilkan permukaan non linier ini, seluruh *dataset* digunakan sebagai pembatas dalam masalah optimasi dengan sangat sedikit variabel yang sesuai dengan 1% dari data yang disimpan. Sisa data dapat dibuang setelah menyelesaikan masalah optimasi. Secara garis besar, langkah-langkah dalam algoritma RSVM dapat dituliskan sebagai berikut [8]:

1. Pilih matriks subset acak $\bar{A} \in R^{\bar{m}xn}$ dari data matriks awal $A \in R^{mxn}$ secara acak. Biasanya $\bar{m}$ adalah 1% hingga 10% dari m.
2. Selesaikan persamaan SSVM yang telah dimodifikasi berikut dan diselesaikan dengan algoritma *Newton-Armijo* [13] di mana A' saja yang digantikan oleh $\bar{A}'$ dengan $\bar{D} \subset D$:

$$\min_{(\bar{u},\gamma)\in R^{\bar{m}+1}} \frac{v}{2}||p(e - D(K(A,\bar{A}')\bar{D}\bar{u} - e\gamma),\alpha)||_2^2 + \frac{1}{2}(\bar{u}'\bar{u} + \gamma^2) \quad \textbf{(2.20)}$$

3. Bidang pemisah dengan A' diganti dengan $\bar{A}'$ sebagai berikut:

$$K(x', \bar{A}')\bar{D}\bar{u} = \gamma \quad \textbf{(2.21)}$$

di mana $(\bar{u}, \gamma) \in R^{\bar{m}+1}$ adalah solusi unik dari persamaan (2.20, dan $x \in R^n$ adalah variabel ruang input bebas dari titik baru.

4. Titik input baru $x \in R^n$ diklasifikasikan ke dalam kelas +1 atau −1 tergantung pada fungsi berikut.

$$f(x) = (K(x', \bar{A}')\bar{D}\bar{u} - \gamma) \quad \textbf{(2.22)}$$

dengan *f(x)* adalah plus function yang masing-masing nilainya adalah +1 atau 0.

2.10 Pengujian Perangkat Lunak

Pengujian perangkat lunak adalah proses pengecekan sebuah sistem perangkat lunak yang telah dibangun untuk mendapatkan informasi tingkat kualitas serta menemukan kesalahan dari perangkat lunak tersebut sebelum dikirimkan kepada pengguna akhir (*end user*) [24]. Pengujian perangkat lunak bertujuan untuk mengevaluasi fungsi-fungsi atau kemampuan sebuah perangkat lunak dan penentuan apakah sudah sesuai dengan hasil yang diharapkan atau belum. Pengujian perangkat lunak juga dapat memberikan pandangan yang objektif dan independen terhadap perangkat lunak untuk memahami risiko implementasi perangkat lunak tersebut. Teknik pengujian meliputi proses menjalankan program atau aplikasi dengan tujuan menemukan *bug* perangkat lunak (kesalahan atau ketidaksesuaian lainnya), dan memverifikasi bahwa produk perangkat lunak tersebut layak untuk digunakan. Ada dua kategori pengujian perangkat lunak yang bisa digunakan berdasarkan pelaksanaannya, yaitu pengujian secara manual (*Manual Testing*) dan pengujian secara otomatisasi (*Automation Testing*) [25].

2.10.1 Pengujian Manual (*Manual Testing*)

Pengujian manual adalah teknik pengujian perangkat lunak yang dilakukan oleh penguji di mana penguji tersebut menyiapkan skenario pengujian secara manual dan mengeksekusinya untuk mengidentifikasi kesalahan pada perangkat lunak [25]. Pengujian ini biasanya mencakup memverifikasi semua fitur yang ditentukan dalam dokumen persyaratan, tetapi sering juga mencakup penguji yang mencoba perangkat lunak dengan perspektif pengguna mereka. Skenario pengujian manual bervariasi dari kasus pengujian yang sepenuhnya ditulis, memberikan langkah-langkah rinci penguji dan hasil yang diharapkan, hingga panduan tingkat tinggi yang mengarahkan sesi pengujian eksplorasi.

2.11 K-Fold Cross Validation

Cross-validation adalah metode statistik yang bisa digunakan untuk mengevaluasi kinerja dari suatu model atau algoritma. Pada metode *cross validation,* data dipisahkan menjadi dua bagian yaitu data pelatihan dan data validasi / uji dimana model atau algoritma dilatih oleh subset pelatihan dan divalidasi oleh subset validasi. Salah satu metode *cross validation* yang biasa digunakan adalah *k-fold cross validation,* dimana pada metode ini *dataset* dibagi menjadi sejumlah K bagian dan mengulangi eksperimennya sebanyak K juga. Misalnya, terdapat jumlah data sebanyak 150 dengan nilai K=5, maka data tersebut kemudian dibagi menjadi 5 lipatan/partisi yang berisi masing-masing 30 data. Data pada partisi ke-K akan digunakan sebagai data uji, sedangkan sisanya akan digunakan sebagai data latih. Ilustrasi dari metode *k-fold cross validation* dapat dilihat pada Tabel 2.2.

Tabel 2.2. Ilustrasi Metode K-Fold Cross Validation

Eksperimen ke-	K1	K2	K3	K4	K5
1	Test	Train	Train	Train	Train
2	Train	Test	Train	Train	Train

3	Train	Train	Test	Train	Train
4	Train	Train	Train	Test	Train
5	Train	Train	Train	Train	Test

Nilai akurasi dari eksperimen yang dilakukan dapat diambil dari nilai rata-rata keseluruhan eksperimen tersebut.