PENGENALAN TULISAN TANGAN DENGAN *SMOOTH SUPPORT VECTOR MACHINE* DAN *DIAGONAL BASED FEATURE EXTRACTION*

Muhammad Fadli[1], Kania Evita Dewi[2]

[1,2] Program Studi Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatikukur 112-114 Bandung
Email: mfadli101295@gmail.com[1], kania.evita.dewi@email.unikom.ac.id[2]

ABSTRAK

Penelitian ini bertujuan untuk mengetahui tingkat akurasi pengenalan tulisan tangan menggunakan metode klasifikasi *Smooth Support Vector Machine* (SSVM) dan metode ekstraksi ciri *Diagonal Based Feature Extraction***.** Adapuun tahapan dalam penelitian ini yaitu tahap pengumpulan data karakter tulisan tangan teridiri dari karakter hurf A-Z, a-z dan angka 0-9. Dalam penelitian ini sebelum melakukan proses klasifikasi, citra tulisan tangan akan melalui tahap *preprocessing* yang teridiri dari *Grayscale, Threshold,* Segmentasi, *Scalling* dan ekstraksi fitur *Diagonal Based Feature Extraction.* Selanjutnya dilakukan proses pelatihan dan pengujian dengan metode *Smooth Support Vector Machine*. Sample citra karakter tulisan tangan di peroleh dari 30 orang koresponden dan akan digunakan sebagai data latih dan data uji. Berdasarkan hasil pengujian yang dilakukan terhadap data uji, maka didapatkan akurasi terbaik 72,6%

Kata kunci : Kecerdasan Buatan, *Smoooth Support Vector Machine, Diagonal Based Feature Extraction, Feature Extraction,* pengenalan citra tulisan tangan, pengolahan citra, SSVM.

1. PENDAHULUAN

Karakter pada tulisan tangan cukup sulit untuk dikenali dengan mesin karena setiap orang memiliki gaya penulisan yang berbeda-beda dan tulisan tangan seseorang juga sangat rentan akan kemiripan antara karakter huruf kapital dengan karakter huruf kecil sehingga susah untuk dibedakan. Berdasarkan perbedaan tersebut banyak penelitian yang membahas tentang tulisan tangan.

Pada penelitian sebelumnya metode Support Vector Machine (SVM) sebagai proses klasifikasi dan metode Zoning untuk ekstraksi ciri sudah pernah dilakukan dan mendapatkan akurasi sebesar 77,6%[1]. Support Vector Machine (SVM) dalam melakukan prediksi menggunakan data yang berdimensi tinggi dan data jumlah besar menjadi kurang efisien [2]. Oleh karena itu dikembangkan metode smooth technique yang menggantikan plus function SVM dengan integral dari fungsi sigmoid neural network yang selanjutnya dikenal dengan Smooth Support Vector Machine (SSVM) [2]. Apabila dibandingkan dengan SSVM, SVM memiliki waktu running yang lebih lama dan akurasi yang lebih kecil daripada SSVM [2]. Smooth Support Vector Machine (SSVM) sudah banyak diterapkan kedalam berbagai masalah seperti untuk pengklasifikasian indeks pembangunan manusia kabupaten/kota se-Indonesia dengan tingkat keakuratan prediksi sebesar 84.77% [3], klasifikasi diabetes mellitus menggunakan metode Smooth Support Vector Machine (SSVM) menghasilkan akurasi yang sangat baik yaitu 97,11% [4], tetapi dari penelitian tersebut metode Smooth Support Vector Machine (SSVM) belum diterapkan pada pengenalan tulisan tangan.

Pada penelitian ini diterapkan pada pengenalan tulisan tangan dengan menggunakan algoritma Smooth Support Vector Machine (SSVM) dan Diagonal Based Feature Extraction. Mengacu pada penelitian sebelumnya yang telah menggunakan metode ekstraksi ciri Diagonal Feature Extraction dengan pembahasan yang sama yaitu pengenalan tulisan tangan dengan menggunakan metode diagonal feature extraction dan k-nearest neighbour memperoleh akurasi sebesar 90% [5] dan diagonal feature extraction based handwritten character system using neural network memperoleh akurasi yang sangat tinggi sebesar 98% untuk 54 fitur dan 99% untuk 69 fitur [6].

Berdasarkan pemaparan tersebut maka, akan diteliti serta dilakukan pengujian dan perhitungan akurasi jika kombinasi ini diterapkan pada masalah tulisan tangan guna mengetahui akurasi dari Smooth Support Vector Machine (SSVM) dengan metode ekstraksi ciri Diagonal Based Feature Extraction.

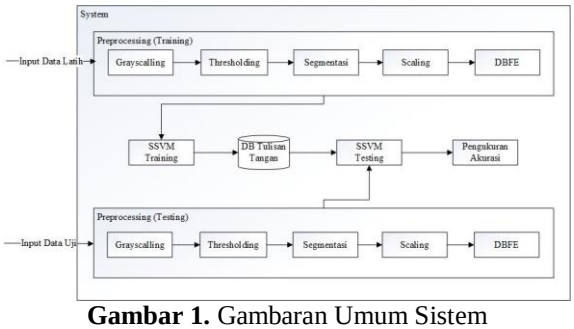
2. ISI PENELITIAN

Pada bagian ini membahas mengenai metode penelitian, landasan teori, analisis sistem, dan hasil pengujian.

2.2 Analisis Sistem

Sistem yang akan dibangun memiliki beberapa tahapan. Tahapan-tahapan tersebut yaitu tahap training, tahap testing dan tahap klasifikasi dengan menggunakan SSVM serta akurasi yang

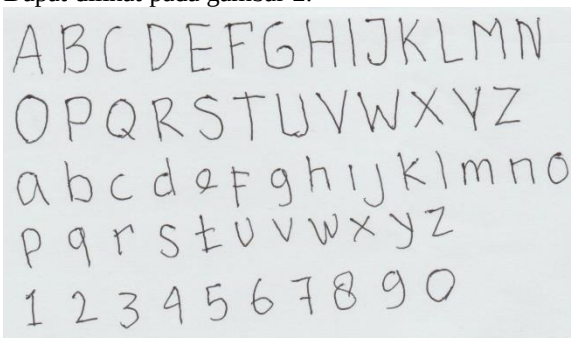
dihasilkan. Gambaran umum sistem dapat dilihat pada gambar 3.1 berikut ini.



Gambar 1. Gambaran Umum Sistem

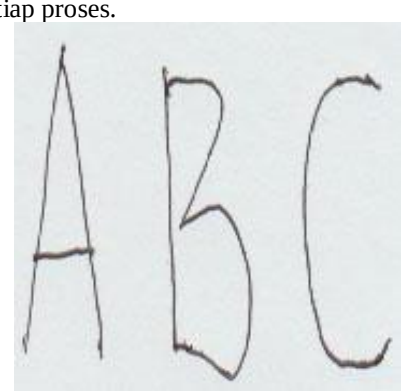
2.3 Analisis Data Masukan

Data masukan yang diperlukan untuk menjalankan sistem ini berupa tulisan tangan yang ditulis pada kertas HVS A4. Untuk mempermudah peneliti dalam memvisualisasikan nilai nilai yang dihasilkan dari setiap proses maka peneliti menggunakan data masukkan berukuran 180 pixel x 180 pixel yang telah di scan dan berformat (.JPG) sebagai data yang akan diolah untuk ke proses selanjutnya. Untuk data latih berupa tulisan tangan yang tersusun dari karakter A-Z, a-z dan 0-9 yang ditulis diatas kertas dengan background putih polos. Dapat dilihat pada gambar 2.



Gambar 2. Contoh Semua Citra Karakter Tangan

Dalam penyusunan penelitian ini data masukkan yang digunakan yaitu sebuah tulisan tangan yang telah di risize menjadi ukuran 180 pixel x 180 pixel dan berformat (JPG) yang hanya terdiri dari tiga karakter yaitu huruf A, B, dan C. Maksudnya guna untuk mempermudah peneliti dalam memvisualisasikan nilai-nilai yang dihasilkan dari setiap proses.



Gambar 3. Data Masukan Citra Ukuran 180x180 pixel

Citra tulisan tangan yang merupakan data masukkan yang digunakan sebagai data latih yang terlihat pada gambar 3, memiliki nilai RGB yang nantinya akan digunakan untuk menghitung pada proses tahapan preprocessing.

2.4 Anallisis Metode

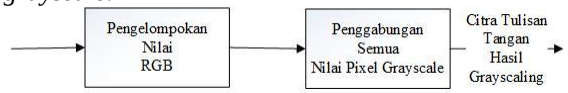
Pada bagian ini yaitu menjelaskan analisis metode yang terjadi pada implementasi *Diagonal Based Feature Extraction* (DBFE) dan *Smooth Support Vector Machine* (SSVM).

2.4.1 Analisis *Preprocessing*

Tahapan dari preprocessing terdiri dari *grayscaling*, *thresholding*, segmentasi, *scalling* dan *Diagonal Based Feature Extraction* (DBFE).

a. Grayscaling

Mengubah format warna menjadi *grayscale* berfungsi untuk mengecilkan range warna menjadi 0 sampai dengan 255. Proses ini akan memudahkan ketika ingi melakukan *threshold* citra menjadi citra hitam putih. Gambar 4 merupakan *flowchart* proses *grayscale.*



Gambar 4. Proses Grayscale

Adapun langkah-langkah yang dilakukan sebagai berikut.

1. Warna citra dikelompokkan berdasarkan nilai *red, green* dan *blue*
2. Kemudian menggunakan rumus grayscale pada persamaan 1 yaitu:
   (0,2989*Red) + (0,5870*Green) + (0,1141 *Blue) (1).
   maka akan didapatkan nilai warna *grayscale* citra.
3. Nilai *grayscale* yang didapat menggunakan nilai RGB pada setiap *pixel.*

Misalkan citra pada *pixel* (0,0) mempunyai nilai *Red* = 229, *Green* = 238, *Blue* = 237, maka berdasarkan persamaan (1) menjadi:

Gray = (0,2989 * R) + (0,5870 * G) + (0,1141 * B)
= (0,2989 * 229) + (0,5870 * 238) + (0,1141 * 237)
= 68,4481 + 139,706 + 27,0417
= 235,1958
= 235

Dari perhitungan di atas , maka *pixel* yang tadinya bernilai *Red* = 229, *Green* = 238, *Blue* = 237 diperbaharui menjadi nilai *grayscale* = 235. Gambar 3.5 di bawah ini merupakan hasil dari proses *grayscale.*

Gambar 5. Gambar Setelah di Grayscale

Berikut adalah gambar matriks warna *Red* (R), *Green* (G), dan *Blue* (B) dari hasil proses *grayscale*.

Tabel 2. Tabel matriks nilai citra grayscale

Y\X	0	1	2	3	4	…	179
0	235	235	235	234	233	…	237
1	235	235	234	234	233	…	238
2	234	234	234	233	233	…	238
3	234	234	234	233	234	…	236
4	235	234	234	234	233	…	233
…	…	…	…	…	…	…	…
179	234	234	234	234	234	…	236

b. Thresholding

Metode threshold yang digunakan dalam penelitian ini menggunakan metode Sauvola Threshold yang bertujuan untuk membedakan objek dan background dari citra agar lebih mudah dikenali pada tahapan ekstraksi fitur. Gambar 6 merupakan proses Sauvola Threshold.

Gambar 6. Proses Sauvola Threshold

$$m(x,y) = \frac{\sum_{i=min}^{i=max} \sum_{j=min}^{j=max} img(i,j)}{\mathrm{i} * \mathrm{j}} \quad (2)$$

$$s(x,y) = \sqrt{\frac{\sum_{i=min}^{i=max} \sum_{j=min}^{j=max} (img(i,j) - m(x,y))^2}{(i*j)-1}} \quad (3)$$

$$T(x,y) = m(x,y) * (1 + k * (\frac{s(x,y)}{R} - 1)) \quad (4)$$

Dalam penelitian ini, parameter yang digunakan sebagai berikut.

1.Jumlah tetangga = 21 pixel. Nilai 21 cocok digunakan karena jika nilai tetangga terlalu besar, maka waktu yang dibutuhkan menjadi lebih lama, atau jika terlalu kecil maka hasil yang didapatkan tidak maksimal.

2. Konstanta R = 128

3. K = 0,3

4. Nilai m(x,y) didapatkan dari persamaan 2

5.Nilai s(x,y) didapatkan dari persamaan standar deviasi 3.

Dari citra tulisan tangan yang digunakan, akan dicari nilai ambang pada pixel (0,0) dengan nilai grayscale 235. Selanjutnya cari pixel tetangga mana saja yang sesuai dengan jumlah tetangga yang sudah ditentukan sebelumnya yaitu 21. Kemudian dari hasil pencarian tersebut didapatkan matriks citra seperti tabel 3., dimana tetangga pixel (0,0) adalah dari pixel (0,0) sampai dengan pixel (10,10).

Tabel 3. Contoh matriks citra yang akan di threshold

(i,j)	0	1	2	3	4	5	6	7	8	9	10	$\sum Img\,(i,j)$
0	235	235	235	234	233	233	233	232	235	234	233	2572
1	235	235	234	234	233	233	232	232	234	234	234	2570
2	234	234	234	233	233	232	232	232	233	234	234	2565
3	234	234	234	233	234	232	232	232	233	233	234	2566
4	235	234	234	234	233	232	232	232	233	233	234	2566
5	235	235	235	233	232	234	233	232	233	234	234	2570
6	236	236	235	235	234	234	233	233	234	234	234	2578
7	236	236	236	235	235	234	234	234	235	234	233	2582
8	233	234	235	236	236	235	234	233	233	233	232	2576
9	234	234	235	236	236	235	234	234	233	233	232	2578
10	235	235	235	236	236	235	235	235	233	233	232	2582
											$\sum_{i=min}^{i=max} \sum_{j=min}^{j=max} img(i,j)$	28307

Langkah pertama cari nilai rata-rata m(0,0) dengan persamaan 2, sehingga hasilnya seperti berikut.

Keterangan:

i min = nilai terkecil dari pixel i
i max = nilai terbesar dari pixel i
j min = nilai terkecil dari pixel j
j max = nilai terbesar dari pixel j

$$m(x,y) = \frac{\sum_{i=min}^{i=max} \sum_{j=min}^{j=max} img(i,j)}{\mathrm{i} * \mathrm{j}}$$

$$m(0,0) = \frac{28307}{11 * 11} = 233{,}942$$

Menurut aturan pembulatan jika angkan dibelakang koma lebih dari 5 maka akan di bulatkan ke atas dari 233,942 menjadi 234 (nilai yang digunakan). Kemudian untuk mencari nilai standar deviasi dipixel x=0, y=0 s(0,0) digunakan persamaan 3 sehingga hasilnya sebagai berikut.

Tabel 4. Perhitungan nilai rata rata jumlah pixel citra

No	Img (i,j)	m(x,y)	$(img\ (i,j) - m(x,y))^2$
1	235	234	1
2	235	234	1
3	235	234	1
4	234	234	0
5	233	234	1
6	233	234	1
7	233	234	1
8	232	234	4
9	235	234	1
10	234	234	0
11	233	234	1
12	235	234	1
13	235	234	1
14	234	234	0
15	234	234	0
16	233	234	1
17	233	234	1
18	232	234	4
…	…	…	…
121	232	234	4
$\sqrt{\sum \frac{((img\ (i,j)\ - m(x,y))^2}{(i*j)-1}}$			166

$$s(x,y) = \sqrt{\frac{\sum_{i=min}^{i=max}\sum_{j=min}^{j=max}(img(i,j) - m(x,y))^2}{(i*j)-1}}$$

$$s(0,0) = \sqrt{\frac{166}{(11*11)-1}}$$

$$= 1{,}176$$

$$= 1$$

langkah selanjutnya masukkan nilai m(0,0) dan s(0,0) ke dalam persamaan 2.2 untuk mendapatkan nilai ambang T(0,0). Sehingga hasilnya menjadi:

$$T(x,y) = m(x,y) * (1 + k * (\frac{s(x,y)}{R} - 1))$$

$$T(0,0) = m(0,0) * (1 + 0{,}3 * (\frac{s(0,0)}{128} - 1))$$

$$= 234 * (1 + 0{,}3 * (\frac{1}{128} - 1))$$

$$= 164{,}34$$

Karena dua angka dibelakang koma bernilai ganjil dan kurang dari 5 maka dibulatkan kebawah menjadi 164. Dari perhitungan di atas didapatkan nilai ambang 164. Langkah selanjutnya langsung dimasukkan dalam persamaan 2.5 sehingga akan didapatkan nilai *pixel* baru. Dari contoh di atas maka *pixel* yang awalnya dengan nilai 235 akan berubah menjadi 1 karena 235 lebih besar dari 164. Berikut adalah tabel hasil perhitungan menggunakan *sauvola threshold*.

Tabel 5. Hasil perhitungan sauvola threshold

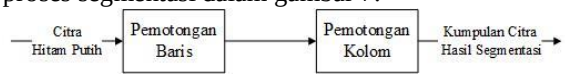
(x,y)	img	m(x,y)	s(x,y)	T(x,y)	*f(x,y)*
(0,0)	235	234	1	164	1
(1,1)	237	236	2	170	1
(1,2)	238	237	2	172	1
…	…	…	…	…	…
(12,0)	114	143	2	102	0
…	…	…	…	…	…
(179,179)	235	234	1	164	1

Berikut adalah gambar citra yang sudah diproses dengan menggunakan metode *sauvola threshold*.

Gambar 6. Hasil Sauvola Threshold

c. Segmentasi

Pada proses ini, input yang digunakan yaitu citra hitam putih hasil *sauvola threshold.* Selanjutnya akan dilakukan pemotongan untuk mendapatkan citra huruf tulisan tangan. Berikut adalah blok *diagram* proses segmentasi dalam gambar 7.

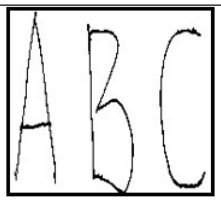
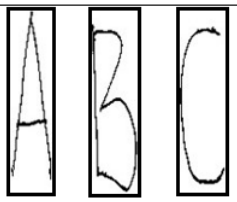


Gambar 7. Proses segmentasi

Pemotongan dilakukan untuk setiap baris(*horizontal)* pada citra input terlebih dahulu, kemudian melakukan pemotongan setiap kolom *(vertical)* pada setiap citra hasil pemotongan secara baris.

Pemotongan untuk setiap baris *(horizontal)* dilakukan dengan menelusuri *pixel* citra dari pixel baris ke-0. Penelusuran terus dilakukan sampai menemukan *pixel* objek, kemudian ditandai sebagai label awal pemotongan. Selanjutnya lakukan penelusuran kembali sampai dalam satu baris *pixel* citra tidak ditemukan *pixel* objek, kemudian tandai sebagai label akhir pemotongan. Label awal dan label akhir ini yang digunakan sebagai acuan untuk memotong citra setiap baris*(horizontal)*. Lakukan hal yang sama untuk pemotongan baris selanjutnya. Untuk pemotongan setiap kolom *(vertical)* sama seperti pemotongan baris, hanya saja penelusuran citra dari *pixel* kolom ke-0. Berikut adalah tabel hasil segmentasi citra tulisan tangan.

Tabel 6. Hasil proses segmentasi

No	Pemotongan Baris	Pemotongan Kolom
1		

d. Scalling

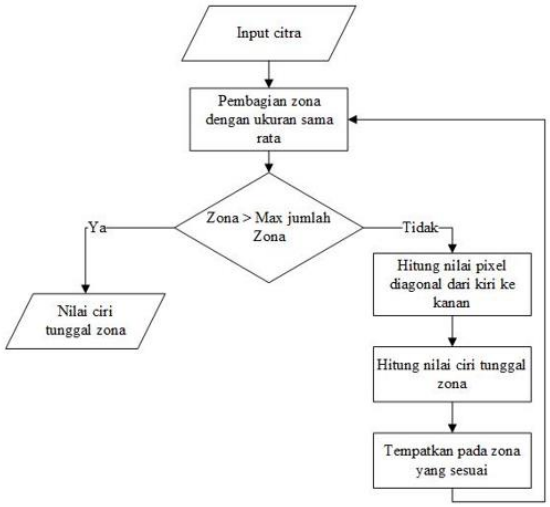
Scaling bertujuan untuk menyamakan ukuran citra hasil segmentasi agar dapat diproses oleh ekstraksi fitur *Diagonal Based feature Extraction.* Dalam penelitan ini digunakan ukuran *scaling* citra 100 *x* 100 piksel.

Tabel 7. Citra Scalling

Image Ukuran Hasil Segmentasi	Image Hasil Scaling 100 x 100 pixel
	
	
	

e. Diagonal Based Feature Extraction

Diagonal Based Feature Extraction adalah algoritma ekstraksi ciri yang membagi ukuran piksel gambar menjadi piksel-piksel yang lebih kecil dan sama rata. Character image citra hasil dari tahapan scalling berukuran 100x100 pixels dibagi menjadi 25 zona yang sama rata. Setiap zona berukuran 25x25 pixels. Ciri diextraksi dari tiap zona dengan bergerak diagonal dari masing-masing 25x25 pixels. Tiap zona memiliki 49 garis diagonal dan foreground pixels yang ada di setiap baris diagonal dijumlahkan untuk mendapatkan satu sub ciri. 49 sub ciri ini akan dirataratakan untuk mendapatkan nilai ciri tunggal dan ditempatkan di zona yang sesuai. Prosedur ini diulangi untuk semua zona. Akan ada beberapa zona yang diagonalnya kosong dari foreground pixels. Nilai ciri untuk zona tersebut adalah nol. Berikut alur dari algoritma *Diagonal Based Feature Extraction* seperti pada gambar 8.



Gambar 8. *Diagonal Based Feature Extraction*

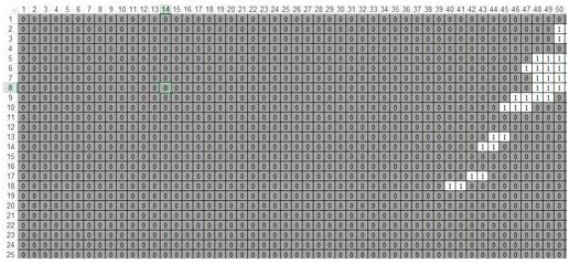
Image citra yang digunakan untuk ekstraksi ciri berupa karakter/kelas dari pengenalan tulisan tangan berupa huruf A, B dan C.

1. Kelas A

Gambar 9. Image citra ukuran 100x100 pixels

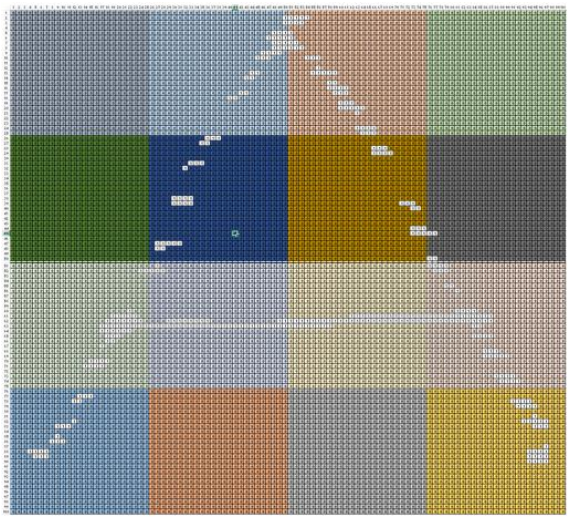
Karena ukuran citra yang terlalu besar maka tidak dapat menampilkan nilai pixel Image citra berukuran 100x100 pixels. Kolom bewarna putih menandakan pixel citra bernilai (1) dan kolom abu-abu bernilai (0).

Tabel 8. Nilai pixel biner image citra huruf A



2. Pembagian Zona Kelas A

Setiap karakter akan dibagi menjadi zona-zona kecil. Setiap karakter berukuran 100 x 100 piksel dibagi menjadi 16 zona kecil dengan ukuran tiap zona sama rata yaitu 25 x 25 piksel.



Tabel 9. Pembagian zona

Keterangan:

Tabel 10. Nama zona hasil pembagian

ZONA 1	ZONA 2	ZONA 3	ZONA 4
ZONA 5	ZONA 6	ZONA 7	ZONA 8
ZONA 9	ZONA 10	ZONA 11	ZONA 12
ZONA 13	ZONA 14	ZONA 15	ZONA 16

3. Ekstraksi Diagonl Untuk Setiap Zona

a. Ekstraksi diagonal zona 1

Jumlahkan nilai pixel pada setiap diagonal bergerak dari kiri ke kanan. Jumlah diagonal pada zona ini.

Tabel 11. Ekstraksi diagonal zona 1

JUMLAH PIXEL DIAGONAL	ZONA 1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
JUMLAH PIXEL DIAGONAL		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL JUMLAH PIXEL DIAGONAL		0																								

Total jumlah pixel dari 49 diagonal adalah 0. Untuk mendapatkan nilai tunggal untuk zona 1 yaitu total jumlah pixel diagonal dibagi dengan banyak diagonal yaitu 49 maka didapatkan nilai 0. Nilai ciri tunggal zona 1 yaitu 0 dan ditempatkan di zona yang sesuai. dapat dilihat pada tabel 12.

Tabel 12. Nilai ciri tunggal setiap zona

0	ZONA 2	ZONA 3	ZONA 4
ZONA 5	ZONA 6	ZONA 7	ZONA 8
ZONA 9	ZONA 10	ZONA 11	ZONA 12
ZONA 13	ZONA 14	ZONA 15	ZONA 16

Ekstraksi diagonal untuk zona 2, zona 3 dan seterusnya sampai dengan zona 16 langkah-langkahnya sama dengan menghitung nilai pada zona 1. Sehingga nilai ciri tunggal setiap zona dapat dilihat pada tabel 13.

Tabel 13. Nilai ciri tunggal zona kelas A

0	0,6122448979	0,9795918367	0
0	0,4897959183	0,3673469387	0,0816326530
0,8979591836	0,7959183673	0,9591836734	1,2244897959
0,4081632653	0	0	0,6122448979

Untuk menghitung nilai ciri tunggal untuk zona kelas B sama dengan langkah-langkah menghitung nilai ciri tunggal pada kelas A. Sehingga nilai ciri tunggal untuk zona B dapat dilihat pada tabel 14.

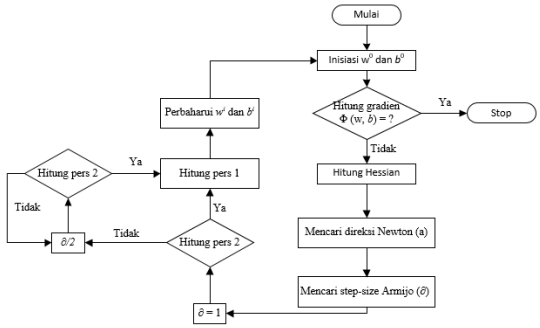
Tabel 14. Nilai ciri tunggal zona kelas B

2,4285714286	1,1836734694	1,5510204082	0
2,0816326531	1,8775510204	1,9387755102	0
2,8775510204	0,8163265306	0,4081632653	2,4693877551
1,8163265306	1,2653061224	1,2857142857	2,3673469388

Nilai-nilai yang dimiliki setiap A, B dan seterusnya merupakan nilai vector. Nilai vector ini akan dijadikan inputan untuk masuk ke tahapan processing yaitu menggunakan algoritma *Smooth Support Vector Machine.*

2.4.2 Metode SSVM

SVM adalah metode klasifikasi yang membagi ruang vector menjadi 2 bagian yaitu kelas positifdan kelas negative oleh hyperplan. Berdasarkan hal tersebut maka digunakan *multiclass* SVM [16]. Berikut ini tahapan dari metode SSVM linier dengan menggunakan algoritma newton armijo dapat dilihat pada gambar 10.



Gambar 10. Tahapan metode SSVM menggunakan algoritma newton armijo

Keterangan :

w^b : Seukuran Inputan

b^0 : Angka

D : Matrik Diagonal berukuran (m x m) dengan nilai [1, -1]

A : Matrik berukuran (m x n)

m : Banyaknya data

n : fitur

w : vektor normal berukuran (n x 1)

e : vektor berukuran (m x 1)

y : Parameter penentu lokasi bidang pemisah terhadap titik asal

v : Parameter positif yang menyeimbangkan bobot dari training error dan margin maximation term.

Tahapan-tahapan dari algoritma Newton Armijo adalah sebagai berikut:

1. Inisialisasi Data

A=

$$\begin{bmatrix} 0 & 0{,}6122448979 & 0{,}9795918367 & 0 & \dots & 0{,}6122448979 \\ 2{,}4285714286 & 1{,}1836734694 & 1{,}5510204082 & 0 & \dots & 2{,}3673469388 \end{bmatrix}$$

(A adalah matrik berukuran (2 x 16)

D = $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ *(matriks diagonal berukuran (2 x 2))*

e = $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ *(e adalah vektor dari banyak data (2 x1))*

2. Menghitung Gradien Matriks Hessian

Initial point untuk menghitung gradien matriks hessian menggunakan formulasi sebagi berikut :

Diketahui: $w = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \dots \\ 0 \end{bmatrix}$ dan y = 0

(w berukuran matriks (16 x 1) karena banyaknya data yaitu 16)

$$\lim_{a\to\infty} \nabla\Psi_\alpha\,(w,y) = \begin{bmatrix} \boldsymbol{w - vA^T D\big(\, e - D(Aw - ey)\big)} \\ \boldsymbol{y + ve^T D\big(\, e - D(Aw - ey)\big)} \end{bmatrix}$$

Menggunakan initial point tersebut dihitung $\boldsymbol{e - D(Aw - ey)}$ setelah itu hitung nilai $\boldsymbol{w - vA^T D\big(\, e - D(Aw - ey)\big)}$ lalu hitung nilai $\boldsymbol{y + ve^T D\big(\, e - D(Aw - ey)\big)}$ sehingga hasil dari

$$\lim_{a\to\infty} \nabla\Psi_\alpha\,(w,y) = \begin{bmatrix} 4{,}8571428572 \\ 1{,}1428571430 \\ 1{,}1428571430 \\ 0{,}0000000000 \\ 4{,}1632653062 \\ 2{,}7755102042 \\ 3{,}1428571430 \\ -0{,}1632653060 \\ 3{,}9591836736 \\ 0{,}0408163266 \\ -1{,}1020408162 \\ 2{,}4897959184 \\ 2{,}8163265306 \\ 2{,}5270122448 \\ 2{,}5714285714 \\ 3{,}5102040818 \\ 0 \end{bmatrix}$$

3. Mengecek Gradien > 0

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

$$\begin{bmatrix} 4{,}8571428572 & 1{,}1428571430 & 1{,}1428571430 & \dots & 0 \end{bmatrix} \begin{bmatrix} 4{,}8571428572 \\ 1{,}1428571430 \\ 1{,}1428571430 \\ 0{,}0000000000 \\ 4{,}1632653062 \\ 2{,}7755102042 \\ 3{,}1428571430 \\ -0{,}1632653060 \\ 3{,}9591836736 \\ 0{,}0408163266 \\ -1{,}1020408162 \\ 2{,}4897959184 \\ 2{,}8163265306 \\ 2{,}5270122448 \\ 2{,}5714285714 \\ 3{,}5102040818 \\ 0 \end{bmatrix}$$

= 117,4861657344

Ternyata gradien > 0 (117,4861657344 > 0) sehingga proses dilanjutkan pada tahapan Newton Direction.

4. Menghitung Gradien Dari Matriks Hessian

Pada tahapan Newton Direction, matriks Hessian dihitung terlebih dahulu menggunakan formulasi sebagai berikut:

Diketahui : $\boldsymbol{e - D(Aw - ey)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

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

$$S_\infty\left(\, \boldsymbol{e - D(Aw - ey)}\right) = \begin{bmatrix} \frac{1+sign(1)}{2} \\ \frac{1+sign(1)}{2} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$diag\,(S_\infty\left(\, \boldsymbol{e - D(Aw - ey)}\right)) = \begin{bmatrix} \boldsymbol{1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}$$

Hitung matriks hessian menggunakan rumus $H_{11} = A^T\, diag\,(S_\infty\left(\, \boldsymbol{e - D(Aw - ey)}\right))$ **A,** $H_{12} = -A^T\, diag\,(S_\infty\left(\, \boldsymbol{e - D(Aw - ey)}\right))$ **e,** $H_{21} = -e^T\, diag\,(S_\infty\left(\, \boldsymbol{e - D(Aw - ey)}\right))$ **A,** $H_{22} = e^T\, diag\,(S_\infty\left(\, \boldsymbol{e - D(Aw - ey)}\right))$ **e.** Setelah nilai $H_{11}, H_{12}, H_{21}, H_{22}$ dihitung lalu masukkan ke dalam rumus dibawah ini.

$$\lim_{a\to\infty} \nabla^2\Psi_\alpha\,(w,y) = \begin{bmatrix} \frac{\partial^2\Psi_a(w,\gamma)}{\partial w^2} & \frac{\partial^2\Psi_a(w,\gamma)}{\partial w \partial \gamma} \\ \frac{\partial^2\Psi_a(w,\gamma)}{\partial^2 \partial w} & \frac{\partial^2\Psi_a(w,\gamma)}{\partial^2 \gamma^2} \end{bmatrix}$$

$= \mathrm{I} + \upsilon \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$ maka hessian yang terbentuk adalah sebagai berikut :

$$=\begin{bmatrix} 12{,}795918368 & 5{,}7492711372 & \cdots & 11{,}4985422744 & 4{,}8571428572 \\ 5{,}7492711372 & 4{,}5518533944 & & 6{,}3540191586 & 3{,}5918367346 \\ 7{,}5335276970 & 4{,}8713036234 & \cdots & 8{,}5431070388 & 5{,}0612244898 \\ 0 & 0 & \vdots & 0 & 0 \\ 10{,}110787172 & 4{,}927946689 & \vdots & 9{,}8558933780 & 4{,}1632653062 \\ 9{,}1195335278 & 5{,}0445647646 & \vdots & 9{,}4893794252 & 4{,}7346938774 \\ 9{,}4169096210 & 5{,}0395668470 & \vdots & 9{,}6293211162 & 4{,}6122448978 \\ 0 & 0{,}0999583506 & \vdots & 0{,}0999583506 & 0{,}1632653060 \\ 13{,}976676385 & 7{,}9117034568 & \vdots & 14{,}7238650562 & 7{,}5510204080 \\ 3{,}9650145772 & 2{,}9071220324 & \vdots & 4{,}8396501456 & 3{,}2244897958 \\ 1{,}9825072886 & 2{,}1407746770 & \vdots & 3{,}1070387336 & 1{,}3673469387 \\ 11{,}994169096 & 7{,}3452728028 & \vdots & 13{,}1911703456 & 7{,}3877551020 \\ 8{,}8221574344 & 4{,}7996668054 & \vdots & 9{,}0995418576 & 4{,}4489795918 \\ 6{,}1370297374 & 2{,}9911573510 & \cdots & 5{,}9823147020 & 2{,}5270122448 \\ 6{,}2448979592 & 3{,}0437317784 & \cdots & 6{,}0874635568 & 2{,}5714285714 \\ 11{,}498542274 & 6{,}3540191586 & & 12{,}9583506874 & 5{,}9591836734 \\ 4{,}8571428572 & 3{,}5918367346 & & 5{,}9591836734 & 5 \end{bmatrix}$$

(matriks berukuran (17 x 17))

5. Menentukan Newton Direction (d^i)

   Misalkan $H = \nabla^2 \Psi_\alpha (\boldsymbol{w^i}, \boldsymbol{y^i})$ dan $G = \nabla \Psi_\alpha (\boldsymbol{w^i}, \boldsymbol{y^i})$ maka $d^i = \boldsymbol{H^{-1}}(-\boldsymbol{G}) =$

3. PENUTUP

3.1 Kesimpulan

Berdasarkan hasil pengujian sistem pengenalan tulisan tangan menggunakan metode *Smootth Support Vector Machine* dan *Diagonal Based Feature Extraction* maka diperoleh akurasi terbaik sebesar 72,6%.. Akurasi ini dipengaruhi oleh parameter pelatihan, data training dan data uji yang digunakan.

3.2 Saran

Agar penelitian selanjutnya tentang pengenalan tulisan tangan menggunakan metode *Smootth Support Vector Machine* dan *Diagonal Based Feature Extraction* memiliki akurasi yang lebih tinggi, maka berikut adalah saran yang dapat dijadikan pertimbangan, yaitu:

1. Dibutuhkan metode segmentasi citra tulisan tangan yang lain.
2. Dibutuhkan metode ekstraksi fitur lain yang lebih baik untuk gambar seperti *Template matching* atau yang lain yang terbaru..
3. Dibutuhkan Metode *Smootth Support Vector Machine* menggunakan metode Newton Armijo dengan kernel yang berbeda.

DAFTAR PUSTAKA

[1] U. Rohwana, M. Isa, J. Matematika, F. Matematika, and P. Alam, "Pengenalan Tulisan Tangan Huruf Latin Bersambung Secara Real Time Menggunakan Algoritma Learning Vector Quantization," J. Sains Dan Seni Pomits, vol. 2, no. 1, pp. 1–6, 2013.

[2] L. Herman, Syafie and D. Indra, "Pengenalan Angka Tulisan Tangan Mengunakan Jaringan Syaraf Tiruan," Ilk. J. Ilm., vol. 10, no. 2, pp. 201–206, 2018.

[3] F. P. Putri and A. Kusnadi, "Pengenalan Tulisan Tangan Offline Dengan Algoritma Generalized Hough Transform dan Backpropagation," Ultim. Comput., vol. 10, no. 1, pp. 5–12, 2018.

[4] E. Suryanto and S. W. Purnami, "Perbandingan Reduced Support Vector Machine dan Smooth Support Vector Machine untuk Klasifikasi Large Data," J. Sains dan Seni ITS, vol. 4, no. 1, pp. D25–D30, 2015.

[5] R. A. Nugroho, Tarno, and A. Prahutama, "Klasifikasi Pasien Diabetes Mellitus Menggunakan Metode Smooth Support Vector Machine (Ssvm)," Gaussian, vol. 6, pp. 439–448, 2017.

[6] T. W. Fauzi, Fatkhurokhman, Darsyah, Moh.Yamin, Utami, "Smooth Support Vector Machine (Ssvm) Untuk Pengklasifikasian Indeks Pembangunan Manusia Kabupaten / Kota Se-Indonesia," Statiska, vol. 5, no. 2, pp. 14-23, 2017.

[16] N. I. Widiastuti, E. Rainarli, and K. E. Dewi, "Peringkasan dan Support Vector Machine pada Klasifikasi Dokumen," *J. Infotel*, vol. 9, no. 4, p. 416-421, 2017.