PENGENALAN TULISAN TANGAN MENGGUNAKKAN METODE *HIDDEN MARKOV MODEL*

Daniel[1], Irfan Maliki[2]

[1,2] Universities Komputer Indonesia
Jalan Dipatiukur No. 112 Bandung, Jawa Barat 40132
E-mail : sembiringdaniel1@gmail.com[1], irfanmaliki007@gmail.com[2]

ABSTRAK

Perkembangan teknologi saat ini memanfaatkan Komputer yang tumbuh pesat pada hampir semua bidang. Manusia semakin tergantung pada kemapuan Komputer untuk melakukan penyimpanan ataupun pengolhan informasi. Pengolahan informasi yang saat ini sedang berkembang adalah tentang pengenalan tulisan tangan, pengenalan tulisan saat ini sedang banyak digunakkan seperti identikasi dokumen-dokumen penting dan lain-lain. Proses awal pada system adalah preprocessing data latih dan uji, kemudian dilakukan ekstrasi ciri dengan metode DEF(*Directional Elment Feature*). DEF salah satu metode ekstrasi ciri yang terbukti memberikan hasil terbaik dalam pengenalan tulisan tangan. Sebagai contohnya adalah pernah diterapkan pada huruf cina berdasarkan kontur, kemudian digabungkan dengan metode HMM (*Hidden Markov Model*) digunakkan sebagai metode klasifikasi pada pengenalan tulisan tangan dengan menggunakkan metode DEF(Directional elment feature), dari hasil pengenalan tulisan tangan pada penelitian ini dimiliki data latih sebanyak 720 data terdiri dari huruf A-Z sebnayak 520 setiap masing-masing huruf memiiki 20 data, dan angka 0-9 memiliki 200 data dengan masing-masing angka terdiri dari 20 data. Dari data latih tersebut digunakkan 20% datanya untuk pengujian pengenalan tulisan dari 100% data yang ada, setelah melakukan pengujian dimiliki akurasi sebesar

Kata kunci : Pengenalan Tulisan Tangan,, *Directional Element Feature (DEF),Hidden Markov Model* (HMM).

1. PENDAHULUAN

Perkembangan teknologi saat ini memanfaatkan komputer yang tumbuh pesat pada hampir semua bidang. Manusia semakin tergantung pada kemampuan komputer untuk melakukan penyimpanan ataupun pengolahan informasi, dan yang sedang berkembang pada saat ini yaitu kemampuan komputer yang dapat mengenali suatu pengenalan tulisan. Penelitian dalam bidang pengenalan tulisan tangan banyak digunakan karena dalam kehidupan sehari-hari, seperti identifikasi dokumen-dokumen penting, seperti bukti pengesahan diperusahaan, di dunia perbankan, dan lainnya melibatkan tulisan tangan didalamnya [1].

Tulisan tangan setiap orang memiliki bentuk dan pola yang berbeda-beda. Pola adalah entitas yang terdefinisi melalui ciri-ciri (*feature*). Ciri tersebut yang membedakan suatu pola. Pengenalan pola (*pattern recognition*).dapat diartikan sebagai proses klasifikasi. Dua hal yang sangat berpengaruh terhadap baik tidaknya pengenalan pola tulisan tangan adalah ekstrasi ciri dan klasifikasi. Berdasarkan penelitan pengenalan tulisan tangan yang telah dilakukan ditemukan kendala pada proses pengenalannya, hal ini dikarenakan beberapa huruf alphabet memiliki kemiripan bentuk dengan huruf lainnya [2], serta pada penerapannya pengenalan tulisan tangan baru diimplementasikan pada kasus penelitian yang secara umum. Pengenalan tulisan tangan akan dipengaruhi oleh hasil akurasi bila tulisan tangan ingin menghasilkan ciri yang efektif dan metode klasifikasi yang dapat mengenali ciri citra tulisan tangan dengan baik dalam proses pengenalan tulisan tangan.

Dalam penelitian ini akan digunakan metode *Directional Element Feature* (DEF) sebagai metode ekstrasi ciri. *Directional Element Feature* (DEF) salah satu metode ekstrasi ciri yang telah digunakan pada judul pengenalan aksara jawa tulisan tangan menggunakkan *Directional Element Feature* dan *Multi Class Support Vector Mechine* didapatkan hasil 93,6 %, [3]. Kemudian metode klasifikasi yang pernah digunakan adalah metode *hidden markov model* (HMM), HMM telah digunakan pada penelitian-penelitian sebelumnya terutama pengenalan pola huruf hijaiyah berharakat menggunakkan *Modified-DFE* dan HMM dan didapatkan hasil 69,4 [4]. Hasil rata-rata akurasi terbaik pengenalan pada penelitian sebelum Implementasi metode Hidden Markov Model untuk deteksi Tulisan Tangan tingkat akurasi sekitar 74,72% [5].HMM telah diimplemantasikan pada beberapa karakter menggunakan citra secara online maupun offline dalam proses klasifikasinya. HMM merupakan perluasan dari rantai markov, dimana statenya tidak dapat diamati secara langsung(tersembunyi), tetapi hanya mengamati variable-variabel yang terpengaruhi oleh state Pembahasan tersebut akan ditulis ke dalam skripsi dengan judul "**PENGENALAN TULISAN**

TANGAN MENGGUNAKKAN METODE *HIDDEN MARKOV MODEL*".

2. ISI PENELITAN

2.1 Tinjauan Pustaka

2.1.1 Citra

Citra adalah suatu representasi (gambaran), kemiripan, atau imitasi dari suatu objek. Citra terbagi 2 yaitu citra yang bersifat analog dan ada citra yang bersifat digital. [7].

2.1.2 Citra Analog

Citra analog adalah citra yang bersifat *continue*, seperti gambar pada monitor televisi, foto sinar X, foto yang tercetak di kertas foto, lukisan, pemandangan alam, hasil CT scan, gambar-gambar yang terekam pada pita kaset, dan lain sebagainya. Citra analog tidak dapat direpresentasikan dalam komputer, sehingga tidak bisa diproses di komputer secara langsung [7]

2.1.3 Citra Digital

Citra digital merupakan representatif dari citra yang diambil oleh mesin dengan bentuk pendekatan berdasarkan sampling dan kuantisasi. Sampling menyatakan besarnya kotak-kotak yang disusun dalam baris dan kolom. Dengan kata lain, sampling pada citra menyatakan besar kecilnya ukuran *pixel* (titik) pada citra, dan kuantisasi menyatakan besarnya nilai tingkat kecerahan yang dinyatakan dalam nilai tingkat keabuan (*grayscale*). [7]

2.1.4 *Grayscale*

Citra grayscale menangani gradasi warna hitam dan putih, yang menghasilkan efek warna abu-abu, warna gambar dinyatakan dengan intensitas. Dalam hal ini, intensitas berkisar antara 0 sampai dengan 255. Nilai 0 menyatakan hitam dan nilai 255 menyatakan putih Berikut adalah rumus konversi citra berwarna (RGB) menjadi grayscale.

$$I = (0.2989 * R) + (0.5870 * G) + (0.1141 * B) \quad (2.1)$$

Keterangan :

I = fungsi pencarian nilai skala keabu-abuan
R = komponen nilai merah (*Red)* dari suatu titik *pixel*
G = komponen nilai hijau (*Green*) dari suatu titik *pixel*
B = komponen nilai biru (*Blue*) dari suatu titik *pixel*

Persamaan di atas merupakan salah satu rumus yang digunakan untuk mengkonversikan citra berwarna menjadi grayscale. [7].

2.1.5 *Sauvola threshold*

Sauvola threshold merupakan metode *threshold* yang mampu mendeteksi citra tulisan tangan dengan cara menghitung ambang menggunakan rentang nilai dinamis dari nilai standar deviasi citra *grayscale* [9]. *Sauvola* termasuk dalam *local threshold* dimana nilai ambang ditentukan oleh nilai pixel tetangga**,** tujuan dilakukannya proses *threshold* adalah untuk menyederhanakan bentuk citra.

$$T(x,y) = m(m,y)n * (1 + k * \left(\frac{s(x,y)}{R} - 1\right)) \quad (2.2)$$

Pada rumus (2.2) terdapat rumus unutk menghitung m(x,y) yang dapat dihitung menggunakan rumus (2.3) dan rumus unutk menghitung s(x,y) dapat dilihat pada rumus (2.4)

$$m(x,y) = \frac{\sum_{i=min}^{i=max} \sum_{j=min}^{j=max} img(i,j)}{i*j} \quad (2.3)$$

$$s(x,y) = \frac{\sum_{i=min}^{i=max} \sum_{j=min}^{j=max} (img(i,j) - m(x,y))^2}{(i*j)-1} \quad (2.4)$$

Keterangan :

R : nilai maksimum dari standar deviasi (128 untuk citra grayscale)
k : kernel dengan nilai antara 0.2 – 0.5
m : fungsi yang menghasilkan nilai rata-rata dari sejumlah pixel citra.
s : fungsi yang menghasilkan deviasi dari sejumlah pixel citra
T : fungsi yang menghasilkan nilai threshold (ambang)
x : nilai koordinat lebar citra dalam rumus (i)
y : nilai koordinat tinggi citra dalam rumus (j)

Setelah nilai ambang T(x,y) sudah didaptkan, selanjutnya masukkan ke persamaan (2.5) di bawah ini [9].

$$f(x,y) = \begin{cases} 0, img(x,y) < T(x,y) \\ 255, img(x,y) \geq T(x,y) \end{cases} \quad (2.5)$$

Keterangan :

f: fungsi yang menghasilkan nilai 0 atau 255
img: nilai grayscale citra

2.1.6 Segmentasi

Segmentasi citra adalah pemisahan objek yang satu dengan objek yang lain dalam suatu citra atau antara objek dengan latar yang terdapat dalam sebuah citra [10].

2.1.7 Ekstrasi ciri DEF

. Metode DEF pada OCR yang melihat perbedaan kontur dan tanpa mengalami proses *skeletonizing* [12]

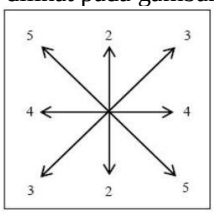
Berikut adalah tahapan-tahapan yang ada dalam ekstrasi DEF [12]

1.*Image Scaling*

Citra akan di-*scaling* sehingga ukurunnya menjadi seragam *NxN pixel*,proses ini dilakukan agar semua citra yang telah disegmentasi mempunyai ukuran yang sama untuk mendapatkan ciri yang baik

2. *Dot Orientation*

Directional Element Feature adalah pencarian nilai *feature* berdasarkan label arah dari sebuah pixel. bisa dilihat pada gambar Contoh pembagi 3 zona pada citra biner dapat dilihat pada gambar [3] 1



Gambar 1

3.Vektor Construction

Tahapan ini untuk menghitung nilai dari setiap *pixel* ketetanggaan yang sebelumnya, melalui proses *dot orientation*, awalnya citra akan dibagi menjadi (piksel)/N_1 *x* N_1 area yang saling overlap sebanyak $N_1/2$ *pixel* terhadap area yang bertetanggaan.

Selanjutnya area, setiap area dibagi kembali menjadi 4 *sub-area*, yaitu A, B, C dan D, yang eksklusif satu sama lainnya. *sub-area* dengan bobot(*w*) pada *sub-*

area tersebut dan dijumlahkan dari setiap elemen di *sub-area* dengan rumus sebagai berikut [3]:

$$x_j = w_A x_j + w_B x_j + w_C x_j + w_D x_j$$
$$j = 1,\ldots,4$$

Dimana j merupakan pixel ketetanggaan.Selanjutnya vektor ciri DEF yang diperolh dengan menggabungkan jumlah elemen disetiap area menjadi satu kolom vektor *V*.

$$V = [X_1^1, X_2^1, X_3^1, X_4^1, \ldots, X_1^N, X_2^N, X_3^N, X_4^N$$

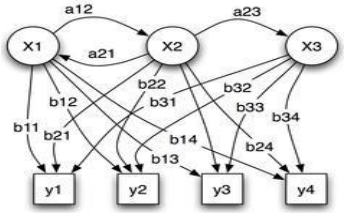
Dimana *N* merupakan jumlah area dari setiap karakter. Vektor *V* inilah yang nanntinya jadi input untuk proses *learning* dan *testing.*

.2.1.7 *Hidden Markov Model*

Hidden Markov Model (HMM) adalah pemodelan probabilitas suatu sistem dengan mencari parameter-parameter yang tidak diketahui untuk mempermudah proses analisis sistem tersebut [15]. *Hidden Markov Model* dapat digunakan untuk aplikasi dibidang *temporal pattern recognition* _pengenalan pola temporal _ seperti pengenalan suara, tulisan, gestur, bioinformatika, kompresi kalimat, *computer vision*, ekonomi, finansial, dan pengenalan not balok.

HMM adalah variasi dari *finite state machine* yang memiliki kondisi tersembunyi *Q*, suatu nilai output *O* (observasi), kemungkinan transisi *A*, kemungkinan output *B*, sebuah kondisi awal π. Kondisi saat ini tidak terobservasi. Tetapi, setiap keadaan menghasilkan output kemungkinan *B*. biasanya, *Q* dam *O* dimengerti, jadi HMM disebut triple (*A*,*B*, π).

1 Himpunan observed state: $O=o_1,o_2,\ldots,oN$.
2 Himpunan hidden state: $Q=q_1,q_2,..qN$
3 Probabilitas transisi: $A = a0_1, a0_2\ldots,an_1$
4 …anm ; a_{ij} adalah probabilitas untuk pindah dari state i ke state j.
5 Probabilitas emisi atau observation likehood: $B=b_i(O_t)$, merupakan probabilitas observasi O_t dibangkitkan oleh state i.
6 State awal dan akhir: q_0, q_{end}, yang tidak terkait dengan observasi.



Gambar 2 Representasi Parameter HMM

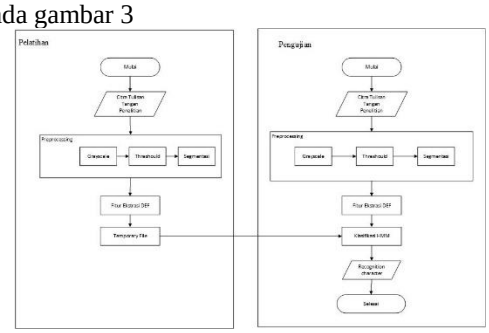
Penjelasan Gambar 2.5:

x = kondisi
y = observasi yang mungkin
a = kemungkinan keadaan transisi
b = kemungkinan output contoh kasus HMM

2.2 Analisa dan Implementasi

2.2.1 Analisis Proses

Proses yang dilakukan dalam aplikasi ini dibagi menjadi beberapa bagian dimana setiap proses memiliki peranan masing-masing dalam melakukan pengenalan tulisan tangan. Proses yang dilakukan pada gambar 3



Gambar 3 Gambaran Umum Sistem

Pada blok diagram diatas, proses awal yang dilakukan yaitu memasukkan citra tulisan tangan ke dalam aplikasi. Langkah selanjutnya melalui tahap pengolahan citra meliputi *grayscale*, *thresholding dan feature extraction.* Dari tahap pengolahan citra akan didapatkan array nilai desimal dari setiap ciri karakter, *array* desimal kemudian diolah kembali pada tahap klasifikasi dengan metode HMM baik pelatihan maupun pengujian. Hasil dari pengenalan tulisan tangan didapatkan teks digital

2.2.2 *Grayscale*

Mengubah format warna menjadi *grayscale* berfungsi untuk mengecilkan range warna menjadi 0 sampai dengan 255.

Adapun langkah-langkah yang dilakukan sebagai berikut.

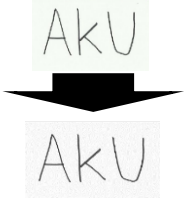
1. Warna citra dikelompokkan berdasarkan nilai *red, green* dan *blue*
2. Kemudian menggunakan persamaan 2.1, maka akan didapatkan nilai warna *grayscale* citra.
3. Nilai *grayscale* yang didapat menggunakan nilai RGB pada setiap *pixel.*

Misalkan citra pada *pixel* (0,0) mempunyai nilai *Red* = 255, *Green* = 219, *Blue* = 255 maka berdasarkan persamaan 2.1 menjadi

I = (0,2989 * R) + (0,5870 * G) + (0,1141 * B)
= (0,2989 * 255) + (0,5870 * 219) + (0,1141 * 255)
= 76,2195 + 128,553 + 29,0955
= 233,868
= 234

Dari perhitungan di atas , maka *pixel* yang tadinya bernilai *Red* = 255, *Green* = 219, *Blue* = 255 diperbaharui menjadi nilai *grayscale* = 234.

Pada gambar 5 di bawah ini merupakan hasil dari proses *grayscale* Pada citra data masukan.



Gambar 5 Hasil Proses grayscale

img(x,y)	Nilai Gray	m(x,y)	s(x,y)	T(x,y)	Nilai Baru
(0,0)	234	248	1.85	174	255
(1,0)	234	248	1.85	174	255
(2,0)	234	248	1.80	174	255
(3,0)	234	246	3.21	247	255
(4,0)	234	248	1.81	147	255
.	.	.	.	.	.
(143,77)	37	44	0.09	341	0
.	.	.	.	.	.
(360,173)	251	250	0,09	175	255

Berikut adalah tabel matrik warna *Red* (R), *Green* (G), dan *Blue* (B) dari hasil prose *grayscale.*

Tabel 1

X/Y	0	1	2	3	.....	173
0	234	248	251	251		249
1	234	248	251	251		249
2	234	248	251	250		250
3	234	234	251	250		250
.....						
360	234	250	249	249		251

2.2.3 Thresholding

Metode *threshold* digunakan dalam penelitian ini yaitu menggunakan metode *sauvola.*

Dalam penelitian ini, parameter yang digunakan sebagai berikut.

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 didaptkan tidak maksimal.

1. Konstanta R = 128
2. K = 0,3
3. Nilai m(x,y) didaptkan dari persamaan 2.3
4. Nilai s(x,y) didapatkan dari persamaan standar deviasi 2.4

).

Tabel 2 Contoh Matriks Citra Yang Akan Di Threshold

X/Y	0	1	2	3	4	5	6	7	8	9	10
0	234	248	251	251	249	250	250	249	248	249	249
1	234	248	251	251	249	250	250	249	248	249	249
2	234	248	251	250	248	250	250	249	248	249	249
3	234	234	251	250	248	250	250	249	248	249	249
4	234	247	251	250	248	250	250	249	248	249	249
5	234	248	251	250	248	250	250	249	248	249	249
6	234	248	251	251	250	250	250	249	248	249	249
7	234	248	251	251	249	250	250	249	248	249	249
8	234	250	250	250	250	250	250	250	250	250	250
9	252	250	249	250	250	250	249	249	249	249	249
10	234	249	249	250	250	249	249	249	249	249	249

Langkah pertama cari nilai rata-rata m(0,0) dengan persamaan 2.3, sehingga hasilnya seperti berikut.

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

$$K(0,0) = \frac{234+248+\cdots+249+249}{11*11} = 248$$

Kemudian untuk mencari nilai standar deviasi s(0,0) digunakan persamaan 2.4 sehingga hasilnya sebagai berikut.

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

$$s(0,0) = \sqrt{\frac{(234-248)^2+(251-248)^2+\cdots+(249-248)^2+(249-248)^2}{121-1}}\ 1.86$$

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

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

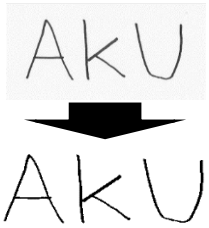
$$T(0,0) = m(0,0) * (1 + 0{,}3 * (\frac{s(0,0)}{121} - 1))$$

$$= 248 * (1 + 0{,}3 * (\frac{1.86}{121} - 1))$$

$$= 174$$

Dari perhitungan di atas didapatkan nilai ambang 107. Langkah selanjutnya langsung dimasukkan dalam persamaan 2.5 sehingga akan didapatkan nilai *pixel* baru. Dari contoh di atas maka *pixel* yang awalnya dengan nilai 153 akan berubah menjadi 255 karena 153 lebih besar dari 107. Berikut adalah tabel hasil perhitungan menggunakan *sauvola threshol*

Berikut adalah gambar 4 citra yang sudah diproses dengan menggunakan metode *sauvola threshold*



Gambar 4 *Hasil Sauvola Threshold*

Setelah dilakukan proses *threshold*, selanjutnya adalah proses binerisasi atau merubah nilai matriks nilai *threshold* yang sebelumnya telah didapatkan ke dalam nilai biner yaitu 0 dan 1, maka nilai 0 diubah menjadi 1 dan nilai 255 akan diubah menjadi 0. Berikut adalah *flowchart* dari proses binerisasi citra *threshold*:

Berikut ini merupakan tabel 4 yang merupakan hasil dari proses perhitungan matriks nilai *threshold* menjadi biner:

Tabel 4 *Matriks Nilai Biner*

X/Y	0	1	2	3	…	173
0	0	0	0	0	…	0
1	0	0	0	0	…	0
2	0	0	0	0	…	0
3	0	0	0	0	…	0
…	…	…	…	…	…	…
360	0	0	0	0	…	0

2.2.4 Segmentasi Citra

Pada proses ini, input yang digunakan yaitu citra hitam putih hasil *sauvola threshold.* Selanjutnya akan dilakukan pemotongan untuk mendapatkan citra huruf tulisan tangan.

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

Gambar* 5 Citra yang akan *disegmentasi

Berikut adalah gambar 6 hasil segmentasi citra tulisan tangan.

Gambar 6 *Pemotongan baris dan kolom pada citra A,K,U*

2.2.5 Ekstrasi ciri

Pada proses ini, input yang digunakan adalah citra karakter hasil dari segmentasi yang telah dilakukan sebelumnya. Selanjutnya akan dilakukan prose ekstraksi cirii [12]

1. Image Scaling

Pada Proses ini Citra yang telah disegmentasi akan diubah ukurannya menjadi 15x15 *pixel*.

2. Dot Orientation

Pada Proses ini akan ditentukan hubungan ketetanggaan antar *pixel.* Proses penentuan ketetanggaan mengikuti aturan seperti pada gambar 2.2, dan nilai arahnya seperti pada gambar 2.3 Pada tabel 3.7 yang merupakan matriks nilai hitam putih dari citra A akan ditentukan ketetanggan antar *pixel*-nya. Setelah itu dilakukan pada citra, maka citra tersebut akan dibagi kembali menjadi 4 sub-area yaitu A,B,C dan D. kemudian akan dihitung jumlah elmen ketetanggan pixel disetiap sub-area.

Area 1

Sub-Area A = (4.4) = 3

Sub-Area B (2.5) + (3.5) + (4.5) + (5.4) = 2 + 2 + 2 + 2 = 8

Sub-Area C (1.5) + (1.6) + (6.3) + (6.4) = 3 + 2 +3 + 2 = 10

Sub-Area D (0.6) + (0.7) + (1.7) + (7.3) = 4 + 2 + 5 + 2 = 13

Area 2

Sub-Area A = 0

Sub-Area B = (4.2) + (5.2) = 5 + 2 = 7

Sub-Area C = (2.1) + (3.1) + (6.2) + (6.3) = 5 + 2 + 2 + 5 = 14

Sub-Area D = (0.0) + (1.0) + (7.3) = 2 + 5 + 5 = 12

Area 3

Sub-Area A = 0

Sub–Area B = (2.2) + (2.3) + (2.4) + (2.5) + (3.2) = 3 + 2 + 5 + 4 + 2 = 16

Sub-Area C = (1.3) + (2.6) + (4.1) + (5.1) = 2 + 4 + 3 + 2 = 11

Sub-Area D = (0.3) + (2.7) + (6.0) + (7.0) = 2 + 4 + 3 + 2 = 11

Area 4

Sub-area A = (3.3) + (3.4) + (4.4) = 5 + 5 + 5 = 15

Sub–Area B = (2.2) + (2.3) + (2.4) + (3.5) + (4.5) + (5.5) = 3 + 2 + 5 + 5 + 5 + 5 = 25

Sub-Area C = (1.3) + (2.1) + (5.6) + (6.6) = 2 + 4 + 5 + 5 = 16

Sub-Area D = (0.3) + (2.0) + (6.7) + (7.7) = 5 + 4 + 5 + 5 = 19

Setelah diperoleh jumlah *pixel* di setiap sub-area, akan dikalikan masing-masing jumlah dari elemen ketetanggaan disetiap sub-area dengan bobot (*w*) pada sub-area tersebut dan dijumlahkan dari setiap elemen di sub-area dengan persamaan 2.6, sebagai berikut

$$x_j = w_A x_j + w_B x_j + w_C x_j + w_D x_j$$
$$j = 1, \ldots, 4$$

J= Pixel Tetangga

W= Sub Area

X1= (3.1) + (8.4) + (10.4) + (13.4) = 3 + 32 + 40 + 52

X2= (0) + (7.2) + (14.4) + (12.3) = 0 + 14 + 56 + 36

X3 = (0) + (16.5) + (11.4) + (11.4) = 0 + 80 + 44 + 44

X4= (15.3) + (25.6) + (16.4) + (19.4) = 45 + 150 + 64 +76

Selanjutnya dapat dibentuk vektor ciri DEF yang diperoleh dengan menggabungkan jumlah elemen di setiap area menjadi satu kolom vektor *V*. dengan persamaan 2.7:

$$V = [X_1^1, X_2^1, X_3^1, X_4^1, \ldots, X_1^N, X_2^N, X_3^N, X_4^N]$$

Vektor Karakter A:

v =[3, 32, 40, 52, 0, 14, 40, 36, 0, 80, 21, 21, 45, 150, 64, 76]

Vektor Karakter A:

V[3, 32, 40, 52, 0, 14, 56, 36, 0, 80, 44, 44, 45, 150, 64, 76]

Vektor karakter K

V[0, 0, 80, 308, 4, 24, 14, 10, 0, 100, 410, 280, 5, 68, 68, 56]

Vektor karakter U

V[0, 0, 36, 90 ,0 ,0 ,0 98, 20, 44, 20, 100, 0, 10, 60, 184]

Setelah itu vektor akan dinormalisasi dengan metode min-max.

Tabel* 5 *Matriks Vektor Biner Vb

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	1	1	0	0	0	1	1	0	0	1	0	0	1	1	0	0
K	0	0	1	1	1	1	0	0	0	1	1	1	0	0	1	0
U	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1

Vektor Biner A:

V[1,1,0,0,0,1,1,0,0,1,0,0,1,1,0,0]

Vektor Biner K:
V[0,0,1,1,1,1,0,0,0,1,1,1,0,0,1,0]
Vektor Biner U:
V[0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1]

Berikut adalah langkah-langkah algoritma Viterbi untuk menentukan kelas huruf A
V[1,1,0,0,0,1,1,0,0,0,0,0,0,1,1,0]

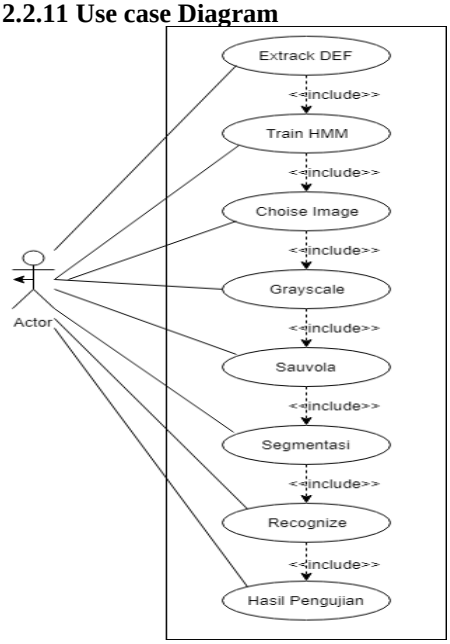
2.2.7 Inisialisasi
Tahapan ini menggunakan nilai pertama dari vektor (z_1), yaitu 1.
$\delta_1(i) = \pi_i * B_{i,z_i},\ i = 1, \dots, N$
$\psi_1(i) = 0$
State A:
$\delta_1(A) = \pi_A * B_{A,1}$
$\delta_1(A) = 0{,}33 * 0{,}4375$
$\delta_1(A) = 0{,}144375$
$\psi_1(A) = 0$
State K:
$\delta_1(K) = \pi_K * B_{K,1}$
$\delta_1(K) = 0{,}33 * 0{,}5$
$\delta_1(K) = 0{,}165$
$\psi_1(K) = 0$
State U:
$\delta_1(U) = \pi_U * B_{U,1}$
$\delta_1(U) = 0{,}33 * 0{,}1875$
$\delta_1(U) = 0{,}061875$
$\psi_1(U) = 0$

2.2.8 Rekursif
Tahapan ini dilakukan untuk nilai-nilai vector sisanya yaitu nilai kedua hingga nilai keenambelas (z_2,…,z_{16}).
$\delta_n(j) = \max_{1<i\leq 3}(\delta_{n-1}(i) * A_{ij}) * B_{j,z_n}$
$\psi_n(j) = \operatorname{argmax}_{1<i\leq 3}(\delta_{n-1}(i) * A_{ij})$
Contoh untuk nilai kedua (z_2) yaitu 1, maka perhitungannya adalah sebagai berikut.
State A:
$\delta_2(A) = \max(\delta_1(A) * A_{AA}, \delta_1(K) * A_{AK}, \delta_1(U) * A_{AU}) * B_{A,1}$
$\delta_2(A) = \max(0{,}144375 * 1, 0{,}165 * 0, 0{,}061875 * 0) * 0{,}4375$
$\delta_2(A) = 0{,}144375 * 0{,}4375$
$\delta_2(A) = 0{,}063164$
$\psi_2(A) = \operatorname{argmax}(\delta_1(A) * A_{AA}, \delta_1(K) * A_{AK}, \delta_1(U) * A_{AU})$
$\psi_2(A) = \operatorname{argmax}(0{,}144375 * 1, 0{,}165 * 0, 0{,}061875 * 0)$
$\psi_2(A) = A$
State K:
$\delta_2(K) = \max(\delta_1(A) * A_{KA}, \delta_1(K) * A_{KK}, \delta_1(U) * A_{KU}) * B_{K,1}$
$\delta_2(K) = \max(0{,}144375 * 0, 0{,}165 * 1, 0{,}061875 * 0) * 0{,}5$
$\delta_2(K) = 0{,}165 * 0{,}5$
$\delta_2(K) = 0{,}0825$
$\psi_2(K) = \operatorname{argmax}(\delta_1(A) * A_{KA}, \delta_1(K) * A_{KK}, \delta_1(U) * A_{KU})$
$\psi_2(K) = \operatorname{argmax}(0{,}144375 * 0, 0{,}165 * 1, 0{,}061875 * 0)$
$\psi_2(K) = K$
State U:
$\delta_2(U) = \max(\delta_1(A) * A_{UA}, \delta_1(K) * A_{UK}, \delta_1(U) * A_{UU}) * B_{U,1}$
$\delta_2(U) = \max(0{,}144375 * 0, 0{,}165 * 0, 0{,}061875 * 1) * 0{,}1875$
$\delta_2(U) = 0{,}061875 * 0{,}1875$
$\delta_2(U) = 0{,}011602$
$\psi_2(U) = \operatorname{argmax}(\delta_1(A) * A_{UA}, \delta_1(K) * A_{UK}, \delta_1(U) * A_{UU})$
$\psi_2(U) = \operatorname{argmax}(0{,}144375 * 0, 0{,}165 * 0, 0{,}061875 * 1)$
$\psi_2(U) = U$
Berikut adalah hasil perhitungan tahap rekursif untuk nilai-nilai z_i lainnya.

2.2.9 Terminasi
Cari nilai max dan argmax dari δ tahap terminasi terakhir.
$p^*(X|\Theta) = \max_{1<i\leq 3}(\delta_{16}(i))$
$p^*(X|\Theta) = \max(\delta_{16}(A), \delta_{16}(K), \delta_{16}(U))$
$p^*(X|\Theta) = \max(0{,}00000734, 0{,}00000504, 0{,}00000180)$
$p^*(X|\Theta) = 0{,}00000734$
$q_{16}^* = \operatorname{argmax}_{1<i\leq 3}(\delta_{16}(i))$
$q_{16}^* = \operatorname{argmax}(\delta_{16}(A), \delta_{16}(K), \delta_{16}(U))$
$q_{16}^* = \operatorname{argmax}(0{,}00000734, 0{,}00000504, 0{,}00000180)$
$q_{16}^* = A$
Maka *state* terakhir dalam observasi adalah A.

2.2.10 Lacak Balik
Pada tahap ini akan dibuat *sequence* terbaik dari *state* berdasarkan hasil observasi. *Sequence* terbaik dapat dilihat dari nilai-nilai ψ. *Sequence* dibuat terurut dari akhir ke awal.
$q_n^* = \psi_{n+1}(q_{n+1}^*), n = 16-1, 16-2, \dots, 1$
Contoh untuk nilai ke-15 (z_{15}), perhitungannya adalah sebagai berikut.
$q_{15}^* = \psi_{16}(q_{16}^*)$
$q_{15}^* = \psi_{16}(A)$
$q_{15}^* = A$
Lakukan untuk semua z_i, sehingga menghasilkan *sequence* sebagai berikut.
Q* = A, A, A, A, A, A, A, A, A, A, A, A, A, A, A, A
Ambil nilai *sequence* terakhir sebagai prediksi kelas huruf. Maka hasil prediksi kelas huruf pada gambar yang diuji adalah A.

2.2.11 Use case Diagram



Gambar 7 usa case diagram

2.2.12 Implementasi Perangkat keras

Berikut adalah implementasi antarmuka dari sistem pengenalan tulisan

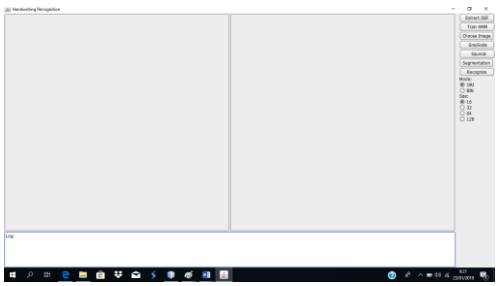
Komponen	Spesifikasi
Processor	AMD A9-9420 (2CPUs), ~3,0GHz
Memory	4GB DDR 4

2.2.13 Implementasi perangkat lunak

Perangkat lunak yang digunakan dalam membangun aplikasi seperti pada table

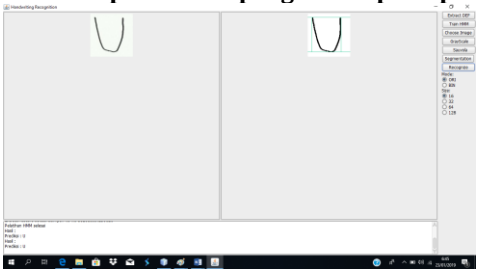
Komponen	Spesifikasi
Sistem Operasi	Windows 10 Home 64-bit
Java	NetBeans IDE 8,2

2.2.14 Implementasi Antarmuka



Gambar 8 Tampilan antarmuka tampilan pada program

2.2.15 Implementasi pengenalan pada program



2.2.16 Pengujian Akurasi

Pengujian akurasi Pengujian pertama dilakukan dengan data latih sebanyak 820 data, 820 terdiri dari huruf A-Z sebanyak 520 setiap masing-masing huruf memiliki 20 data sampel. Dan angka 0-9 memiliki 300 data sampel yang masing-masing angka memiliki 30 data sampel sebelumnya telah dilatih dan citra yang belum dilatih. Berikut adalah rumus untuk menghitung nilai akurasi.

$$\textbf{Akurasi} = \frac{\textbf{TP}}{(\textbf{TP} + \textbf{FN}_1 + \cdots + \textbf{FN}_6)} \times 100\%$$

Keterangan :

1. True *Positive* (TP) merupakan data positif yang terdeteksi benar, dan

2. *False Negative* (FN) merupakan kebalikan dari *True Positive*, sehingga data positif, namun terdeteksi sebagai data negatif.

3. Setiap kolom dalam matriks merepresentasikan kelas yang diprediksikan, sedangkan setiap baris merepresentasikan kelas yang sebenarnya.

Tabel 6 Hasil Pangujian Confusion matrix

Sebanarnya	Prediksi										
	0	1	2	3	4	5	6	7	8	9	
0	1										50 %
1		1	1								50 %
2			1								50 %
3				1		1					50 %
4					1						50 %
5						1					50 %
6										1	0 %
7					1						0 %
8		1							1		50 %
9				1						1	50 %

Sebanarnya	Prediksi										
	0	1	2	3	4	5	6	7	8	9	
	Rata – Rata Akurasi										40 %

Dari hasil akurasi pada Tabel 6 merupakan contoh perhitungan confusion matrix dari percobaan 13 angkat yang di ujikan

3. PENUTUP

3.1 Kesimpulan

Berdasarkan hasil dari pengujian akurasi menggunakkan *Confusion matrix* terhadap 72 data uji yang telah selesai dilakukan, kesimpulan hasil akurasi yang didapatkan terhadap pengenalan tulisan tangan adalah sebagai berikut:

1. Data latih dari metode *HMM* yang digunakan pada saat pelatihan sangat mempengaruhi akurasi yang dilakukan.
2. Kerapihan dalam data training yang dibuat sangat berpengaruh untuk mencapai hasil yang akurasi yang bagus
3. Setelah menguji dengan 72 data uji maka hasil akurasi yang memiliki presentase paling tinggi didapatkan hasil akurasi sebesar 56,94%
4. hasil pengujian pada table 6 merupakan contoh perhitungan confusion matrix yang

3.2 Saran

Saran dari peneliti setelah menyelesaikan tugas akhir ini untuk penelitian selanjutnya yang berhubungan dengan pengenalan tulisan tangan menggunakan metode *Hidden Markov Model*. Sebagai berikut:

1. Dibutuhkan metode segmentasi citra tulisan tangan untuk tulisan yang saling berdempet atau menempel.
2. Kerapihan dalam data latih yang dibuat sangat berpengaruh untuk mencapai hasil yang akurasi yang bagus
3. Dibutuhkan metode ekstraksi fitur yang lebih baik untuk gambar seperti *Principal Components Analysis* (PCA), *Wavelet Transform.*

DAFTAR PUSTAKA

[1] R. Tiofan, "Pengenalan Tulisan Tangan Menggunakan Metode Curvelet Transform Dan Jaringan Saraf Tiruan Backprogation Untuk Kasus Penilaian Esai," in *Unikom*, Bandung, 2017.

[2] I. A. Ersyaputra, "Analisis Dan Implementasi Pengenalan Citra Tulisan Tangan Alfabet Menggunakan Metode Modified Direction Feature dan Klasifikasi Jaringan Saraf Tiruan-Learning Vector Quantization," *Skripsi Fakultas Teknik Informatika, Universitas Telkom*, 2013.

[3] X. D. H. L. Xianliang Wang, "Writer Identification Using Directional Element Features and Linear Transform," *Department of Electronic Engineering*, 2003.

[4] A. A. Nugraha, "Pengenalan Pola Huruf Hijaiyah Berharakat Menggunakan Modified-DFE dan HMM.," *Institut Teknologi Telkom*, 2013.

[5] E. F. Yuwitaning, D.. Bambang Hidayat and S. M. Nur Andini, "Implementasi Metode hidden Markov Model Untuk Deteksi Tulisan Tangan," *e-Proceeding of Engineering* , vol. I, p. 396, Desember 2014.

[6] Sugiyono, Metode Penelitian Kombinasi (Mixed Methods), Bandung: Alfabeta, 2013.

[7] K. D. A and S. A, Teori dan Aplikasi Pengolahan Citra, Yogyakarta: ANDI, 2013.

[9] K. Khurshid, I. siddiqi, C.Feaure dan N. Vincent, "Comparasion of Niblack Inspired Binarization Methods for Ancient Document," *Document Recognition and Retrival XVI*, 2009.

[10] P. A. Cahyan, I. M. Muhammad Aswin and S. M. Ali Mustofa, "Segementasi citra dengan menggunakkan algoritma watershed dan lowpass filter sebagai prosedur awal," *Jurusan Teknik Elektro, Universitas Brawijaya*, november 2013.

[11] S. Suyanto, Artificial Intelligence (Searching, Reasoning, Planning dan Learning), Bandung: Informatika, 2014.

[12] Aulia Husni Nurul A I , Mahmud Dwi Sulistiyo and Retno Novi Dayawati, "Pengenalan aksara jawa tulisan tangan menggunakkan directional element feature dan multi class support vector machine," *Konferensi Nasional Teknologi Informasi dan Aplikasinya*, 13 September 2014.

[13] J. a. K. M. Han, Data Mining Concept and Techniques, McGraw Hill, Inc, 2006.

[15] P. M. E. Budi, "Teori Dasar Hidden Markov Model," *Makalah II2092 Probabilitas dan statistik*, 2010/2011.