PENGENALAN AKSARA JAWA TULISAN TANGAN MENGGUNAKAN DIRECTIONAL ELEMENT FEATURE DAN MULTI CLASS SUPPORT VECTOR MACHINE

Aulia Husni Nurul A.I.[1], Mahmud Dwi Sulistiyo[2], Retno Novi Dayawati[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]aulia.ahn@gmail.com, [2]mahmuddwis@telkomuniversity.ac.id,
[3]retnonovidayawati@telkomuniversity.ac.id

ABSTRACT

Javanese character is a set of old traditional letter from Java, Indonesia. It has a complicated structure and it has similiar shape to each other.

Optical Character Recognition (OCR) is a field in computer vision that attempted to recognize a certain character within an image. Various kinds of research have been done by using various methods in order to make an OCR system which able to recognize characters properly. Because of Javanese character's charasteristic, a strong method is needed in order to build a high accurate OCR system in recognizing Javanese character.

Directional Element Feature (DEF) is a feature exctraction method that has been used in many researches and has been proven to be strong enough to recognize Chinese characters which has complicated shape structure. DEF builds feature vector by count up image edge neighborhood element in each character. Support Vector Machine (SVM) is a classification method that works by finding a hyperplane with smallest margin to separate two data classes. In some previous research, SVM has been proven to be strong enough to classify data, especially data that has not been seen by the system before. In some other research, SVM has been proven better than common Artificial Neural Network in classifying data.

In this research, a Javanese character recognition system is built using DEF and SVM. Test result shows the best recognition accuracy is 93.6% by recognizing 250 handwritten Javanese Character which is 10 letters for each character.

Keywords: OCR, handwritten, Javanese character, DEF, SVM

I. PENDAHULUAN

Indonesia adalah negara yang terdiri atas beribu suku bangsa dan kebudayaan. Salah satu kebudayaan di Indonesia yang perlu untuk dilestarikan adalah tulisan tradisional yang berasal dari suku Jawa yang disebut dengan Aksara Jawa.

Seiring berkembangnya teknologi komputer, banyak inovasi yang telah dikembangkan. Salah satunya adalah kemampuan komputer dalam mengenali karakter tulisan tangan, yaitu Optical Character Recognition (OCR). Aksara Jawa merupakan tantangan tersendiri dalam OCR karena setiap bentuk karakternya rumit dan beberapa di antaranya memilki struktur bentuk yang hampir sama.

Beberapa penelitian sebelumnya telah dilakukan dengan mengimplementasikan berbagai macam metode untuk melakukan pengenalan karakter aksara Jawa tulisan tangan. Penerapan Hidden Markov Model pernah dilakukan dan berhasil menghasilkan tingkat akurasi sebesar 86,4% [1]. Kombinasi antara Jaringan Syaraf Tiruan dan Fuzzzy Feature Extraction juga pernah digunakan untuk mengenali aksara Jawa dengan akurasi terbaik sebesar 84,1% [13].

Di lain sisi terdapat metode Directional Element Feature (DEF) yang merupakan metode untuk proses ekstraksi ciri dan telah berhasil digunakan untuk mengenali huruf Cina. Sebagaimana diketahui, huruf Cina memiliki struktur yang kompleks dan beberapa di antaranya memiliki kesamaan struktur [11][12]. Adapun Support Vector Machines (SVM) pada beberapa tahun terakhir banyak dikembangkan untuk berbagai kasus, mulai dari pengenalan wajah, pengenalan dan verifikasi suara,

hingga pengenalan tulisan tangan atau OCR [6][9][10].

DEF merupakan metode ekstraksi ciri pada OCR yang melihat perbedaan kontur dan tanpa mengalami proses *skeletonizing*. Mengingat beberapa aksara Jawa hanya memiliki perbedaan yang sangat kecil, proses *skeletonizing* dikhawatirkan justru membuat adanya informasi yang hilang. Sedangkan *multi class* SVM pernah diterapkan pada pengenalan huruf *Devnagari* dan terbukti lebih baik daripada model Jaringan Syaraf Tiruan (JST) yang umumnya digunakan.

Berdasarkan studi literatur tersebut, metode DEF yang dipadukan dengan multi class SVM akan diterapkan pada sistem OCR untuk mengenali aksara Jawa tulisan tangan. Tentunya di masa yang akan datang penelitian ini dapat dikembangkan, sehingga menjadi produk aplikasi yang lebih bermanfaat, khususnya dalam rangka melestarikan salah satu budaya Indonesia yang amat berharga.

II. AKSARA JAWA

Aksara Jawa adalah huruf tradisional yang berasal dari kebudayaan kuno Jawa. Bentuk aksara Jawa yang sudah ada sekarang merupakan modifikasi dari aksara Kawi yang sudah tetap bentuknya dari sejak masa kesultanan Mataram (abad ke-17). Struktur masing-masing huruf aksara Jawa mewakili paling tidak 2 buah huruf Latin [18].

Aksara jawa memiliki 20 huruf dasar (aksara *ngelegena*), 20 huruf pasangan sebagai penutup huruf vokal, 8 huruf ‘utama’ (aksara *murda*, ada yang tidak berpasangan), 8 pasangan huruf ‘utama’, 5 aksara *swara*, 5 aksara *rekan* serta 5 pasangannya, beberapa buah *sandhangan* sebagai pengatur vokal, beberapa huruf khusus, tanda baca, dan tanda pengatur tulisan (*pada*) [18].

Pada penelitian ini, pengenalan aksara Jawa tulisan tangan pada sistem OCR yang dibangun dibatasi hanya untuk aksara *ngelegena* dan *swara*. Aksara *ngelegena* adalah aksara dasar yang terdiri dari 20 huruf, yaitu ha, na, ca, ra, ka, da, ta, sa, wa, la, pa ,dha, ja, ya, nya, ma, ga, ba, tha, nga. Sedangkan aksara *swara* merupakan huruf vokal A, I, U, E, O, yang biasanya digunakan untuk menuliskan awalan atau huruf besar vocal [20]. Berikut bentuk dari aksara Jawa yang akan dikenali.

Gambar 1. Aksara Jawa Ngelegena

Gambar 2. Aksara Jawa Swara

III. DIRECTIONAL ELEMENT FEATURE

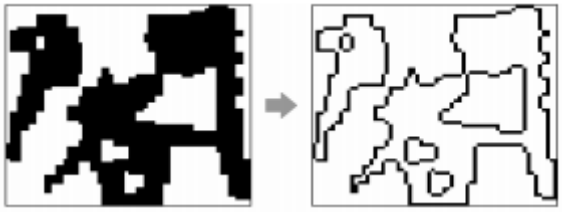
Directional Element Feature (DEF) merupakan metode ekstraksi ciri yang telah banyak digunakan dan terbukti memberikan hasil yang baik dalam pengenalan tulisan tangan. Salah satunya adalah yang pernah diterapkan untuk huruf Cina [14]. DEF bekerja dengan cara mengambil ciri dari sebuah citra berdasarkatn kontur. Algoritma DEF berdasarkan referensi yang diperoleh adalah sebagai berikut [11].

A. Image Scaling

Tahapan pertama pada DEF adalah melakukan *scaling* citra per karakter menjadi ukuran citra yang seragam *NxN*. Hal ini dilakukan agar semua citra karakter memiliki ukuran yang sama untuk mendapatkan ciri yang baik.

B. Countour Extraction (Edge Detection)

Countour Extraction merupakan tahapan untuk mendapatkan tepi dari tiap karakter tulisan tangan. Deteksi tepi dilakukan dengan cara melakukan pengecekan setiap *pixel*. Jika dijumpai sebuah *pixel* hitam yang berbatasan dengan *background* (*pixel* warna putih), maka *pixel* tersebut dikenali sebagai tepi dari karakter yang akan dikenali dan diberi warna hitam yang menunjukkan sebuah tepi. Selebihnya, akan diberikan warna putih [11].



Gambar 3. Contoh Coutour Extraction pada Citra Tulisan Tangan [11]

Salah satu karakteristik dari tulisan tangan adalah adanya bagian *blur* di sekitar tulisan yang sebenarnya masih merupakan bagian dari tulisan. Tujuan dilakukannya *Countour Extraction* dan bukan *skeletonizing* seperti pada teknik ekstraksi ciri lainnya adalah untuk menjaga agar daerah *blur* tersebut tidak mengalami penghapusan. Hal ini dikarenakan pada daerah *blur* tersebut masih mungkin terdapat informasi penting dari tulisan tangan yang dikenali [11].

C. Dot Orientation

Setelah dilakukan tahapan *Countour Extraction*, selanjutnya adalah dilakukan tahapan *Dot Orientation*. Pada tahapan ini, ditentukan hubungan ketetanggan antar *pixel*.

Proses penentuan ketetanggan pada *Dot Orientation* mengikuti aturan sebagai berikut [12].

1	2	3
8		4
7	6	5

Gambar 4. Matriks Ketetanggaan

a. *Pixel* hitam yang berada di tengah merupakan *pixel* warna hitam yang akan dicek ketetanggaannya.
b. Proses pengecekan ketetanggan dilakukan dengan pengecekan searah dengan jarum jam, mulai dari nomor 1 hingga nomor 8.
c. Jika ditemukan ada *pixel* hitam pertama pada proses pengecekan di salah satu tetangga dari *pixel* hitam tersebut, ubah nilai *pixel* hitam di tengah, yaitu
   - jika pixel tetangga pertama ada di posisi 1 atau 5, beri nilai 5;
   - jika pixel tetangga pertama ada di posisi 2 atau 6, beri nilai 2;
   - jika pixel tetangga pertama ada di posisi 3 atau 7, beri nilai 3; dan
   - jika pixel tetangga pertama ada di posisi 4 atau 8, beri nilai 4.

D. Vector Construction

Setelah dilakukan proses *Dot Orientation,* proses selanjutnya adalah proses pembentukan vektor *(Vector Contruction)* yang nantinya akan menjadi vektor ciri dari masing-masing karakter. Pada tahapan ini, citra akan dibagi menjadi $N_1 x N_1$ area yang saling *overlap* sebanyak $N_1/2$ *pixel* terhadap area yang bertetanggaan.

Selanjutnya, setiap area dibagi kembali menjadi 4 sub-area, yaitu A, B, C, dan D, yang *exclusive* satu sama lain. Kemudian, dihitung jumlah elemen ketetanggan pixel di setiap sub-area, yakni *pixel* yang memiliki ketetanggan secara vertikal (1), horizontal (2), miring ke kanan (3), dan miring ke kiri (4).

Setelah diperoleh jumlah *pixel* di setiap sub-area, akan dikalikan masing-masing jumlah dari elemen ketetanggan di setiap sub-area dengan bobot (*w*) pada sub-area tersebut dan dijumlahkan dari setiap elemen di sub-area dengan rumus sebagai berikut [12].

$$x_j = w_A x_j + w_B x_j + w_C x_j + w_D x_j \quad (1)$$

$$j = 1, \ldots, 4$$

di mana *j* merupakan *pixel* ketetanggaan.

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

$$V = [X_1^1, X_2^1, X_3^1, X_4^1, \ldots, X_1^N, X_2^N, X_3^N, X_4^N] \quad (2)$$

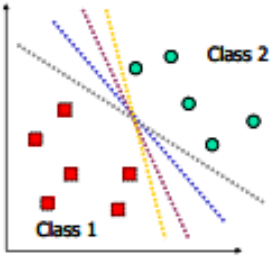
di mana *N* merupakan jumlah area dari setiap karakter. Vektor *V* inilah yang disebut vektor DEF yang menjadi ciri karakter, yang menjadi input untuk proses *learning* dan *testing*.

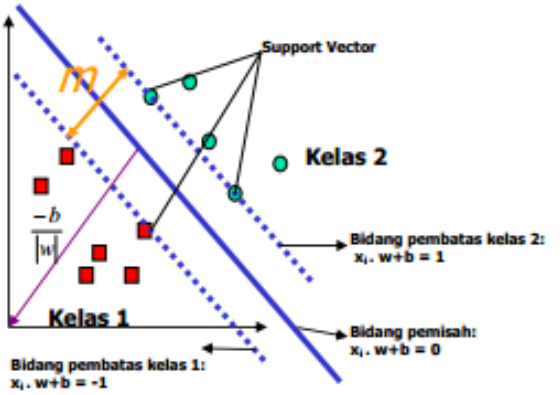
IV. SUPPORT VECTOR MACHINE

A. *Dasar Support Vector Machine*

Support Vector Machine (SVM) merupakan metode yang berfungsi untuk memisahkan dua buah kelas (*binary classification*). Ide dasar dari SVM adalah membentuk suatu bidang pemisah (*hyperplane*) yang memiliki margin terbesar dalam memisahkan suatu kelompok kelas data positif (+) dan negatif (-). Misalnya terdapat data latih $\boldsymbol{x_i} \in \boldsymbol{R^d}$ dengan label $y_i \in \{-1, +1\}$, untuk setiap data latih *i* = 1,…,*l* , di mana *l* adalah jumlah data dan *d* adalah dimensi permasalahan.

Ketika dua buah kelas dapat dipisahkan secara linear (*linearly separable*) pada R^d, SVM akan mencoba mencari *hyperplane* pemisah yang memberikan *generalization error* terkecil diantara jumlah *hyperplane* yang bermacam jumlahnya. Margin yang dimaksudkan adalah jumlah jarak dari *hyperplane* hingga titik terdekat dari data dari tiap kelas. Selanjutnya, data terdekat tersebut disebut dengan *support vector.*





Gambar 5. Ilustrasi Penentuan Hyperplane. Alternatif pilihan hyperplane (atas) dan Hyperplane terbaik dengan margin terbesar (bawah)

Pada ilustrasi di gambar 5 di atas, kelas 1 dan kelas 2 dipisahkan oleh *d-dimensional hyperplane* yangdidefinisikan sebagai

$$w \,.\, x + b = 0 \quad (3)$$

Untuk setiap kelas yang terpisahkan oleh *hyperplane* masing-masing memiliki persamaan $x_i . w + b \geq 1$ untuk kelas positif dan $x_i . w + b \leq 1$ untuk kelas negatif. Dengan *w* merupakan bobot *hyperplane*, dan *b* merupakan nilai *bias* yang mendefinisikan pergeseran *hyperplane* terhadap pusat koordinat. SVM bekerja dengan menemukan *hyperplane* yang memiliki margin terbesar. *Hyperplane* dengan margin terbesar ini dapat ditemukan dengan cara meminumkan fungsi $^1/_2 \|w\|^2$.

$$\min_{w,b} \frac{1}{2}\|w\|^2$$
$$subject\ to: y_i(w^T x_i + b) \geq 1 \quad i = 1, \ldots, n \quad (4)$$

Persoalan tersebut akan lebih mudah diselesaikan apabila diubah ke dalam formula *Lagrangian* yang menggunakan *Lagrange Multipier.* Dengan demikian, permasalahan optimasi dapat diubah menjadi

$$\min_{w,b} L_p(w, b, \alpha) \equiv \frac{1}{2}|w|^2 - \sum_{i=1}^{n} \alpha_i y_i (x_i . w + b) + \sum_{i=1}^{n} \alpha_i \quad (5)$$

dengan tambahan konstrain, yaitu $\alpha_i \geq 0$ (nilai dari koefisien *lagrange*). Dengan meminimumkan *Lp* terhadap *w* dan *b*, maka dari $\frac{\partial}{\partial b} L_p(w, b, \alpha) = 0$, diperoleh

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \quad (6)$$

dan dari $\frac{\partial}{\partial w} L_p(w, b, \alpha) = 0$, diperoleh

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i \quad (7)$$

Kemudian, persamaan di atas dapat diubah ke dalam bentuk *dual problem* L_D, sehingga persoalan pencarian bidang pemisah terbaik dapat dirumuskan menjadi

$$\max_{\alpha} L_D \equiv \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_i y_i y_j x_i x_j$$
$$s.t \sum_{i=1}^{n} \alpha_i y_i = 0, \alpha_i \geq 0 \quad (8)$$

Dengan demikian, dapat diperoleh nilai α_i yang akan dipergunakan untuk mencari nilai *w.* Data pelatihan yang memiliki $\alpha_i \geq 0$ merupakan *support vector*, sedangkan sisanya memiliki nilai $\alpha_i = 0$. Jadi, fungsi keputusan yang dihasilkan hanya dipengaruhi oleh *support vector*.

Formula pencarian bidang pemisah terbaik ini merupakan permasalahan *quadraic programming*. Setelah solusi permasalah ditemukan, maka kelas dari data pengujian *x* dapa ditentukan dengan mengikuti persamaan sebagai berikut.

$$f(x) = sgn(\langle w, x \rangle + b) \quad (9)$$

Substitute $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$

OR

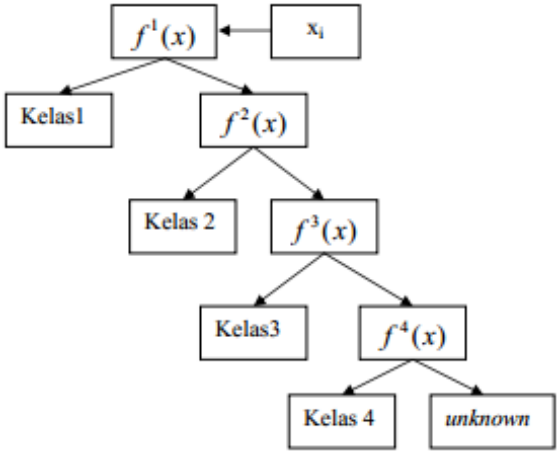
$$f(\mathbf{x}) = \text{sgn}\left(\sum_{i=1}^{N} \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b \right) \quad (10)$$

B. *Multi Class SVM*

Pada awal ditemukan, SVM hanya berfungsi untuk mengklasifikasikan dua buah kelas saja. Namun, seiring perkembangan permasalahan yang hendak diselesaikan, peneliti mengembangkan SVM sehingga memungkinkan untuk dapat mengklasifikasikan data dengan lebih dari dua kelas.

Salah satu pendekatan yang dapat digunakan sehingga membuat SVM mampu mengklasifikasikan data dengan lebih dari dua kelas adalah dengan metode "one-against-all". Dengan metode tersebut, akan terdapat hyperplane sebanyak jumlah kelas. Setiap hyperplane bertugas membedakan apakah setiap data masuk ke suatu kelas tertentu atau selainnya.

Secara sederhana, dengan metode tersebut, sebuah data pada mulanya dicek dengan hyperplane-1, apakah termasuk ke kelas ke-1 atau bukan. Jika bukan, maka dilanjutkan dengan hyperplane-2, apakah termasuk ke kelas ke-2 atau bukan. Jika bukan, maka dilanjutkan lagi hingga hyperplane terakhir. Jika masih bukan, maka data tersebut tidak masuk ke kelas manapun.

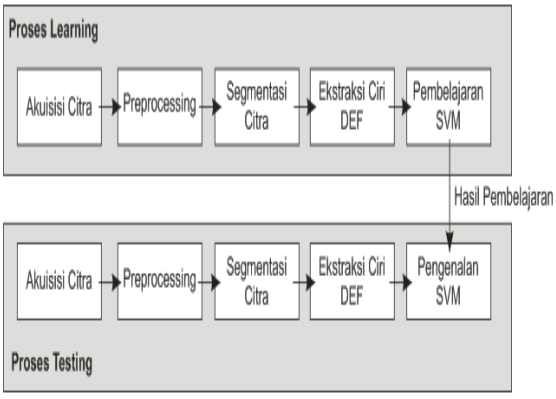


Gambar 6. Ilustrasi Metode "one-against-all" [7]

V. SISTEM OCR

A. *Rancangan Umum Sistem*

Penelitian ini secara garis besar dibagi menjadi dua fase, yakni fase *learning* (pembelajaran) dan fase *testing* (pengujian). Secara umum, tahapan dalam setiap fase dapat digambarkan pada blok diagram berikut ini.

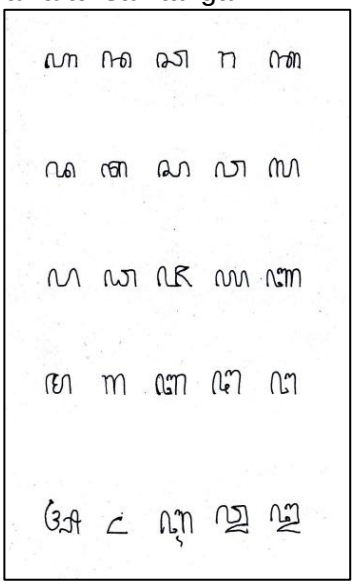


Gambar 7. Alur Sistem pada *Learning* dan *Testing*

B. *Akuisisi Citra*

Tahap akuisisi citra adalah tahap pertama yang akan mengubah aksara Jawa yang ditulis pada kertas menjadi citra digital sehingga selanjutnya dapat diproses lebih jauh oleh komputer.

Pengumpulan sample aksara Jawa tulisan tangan dilakukan di SMA N Jumapolo, Karanganyar. Responden yang berhasil dikumpulkan berjumlah 125 orang. Setiap orang diminta untuk menulis 2 set aksara jawa *nglegena* dan *swara*. Jadi, total data yang dikumpulkan adalah 250 set aksara Jawa tulisan tangan.



Gambar 8. Sample Hasil *Scan* Aksara Jawa

C. Segmentasi Citra Karakter

Agar dapat mengenali setiap karakter dari data yang dikumpulkan, dilakukan segmentasi citra secara vertikal dan horizontal. Segmentasi vertikal pada citra akan memecah citra input seperti pada gambar 8 menjadi citra baru yang berisi citra karakter per baris.

Gambar 9. Contoh Hasil Segmentasi Vertikal

Citra hasil segmentasi vertikal selanjutnya akan di-segmentasi kembali secara horizontal sehingga diperoleh beberapa citra, di mana setiap citra berisi 1 karakter aksara Jawa.

Gambar 10. Contoh Hasil Segmentasi Horizontal

Setiap citra hasil segementasi per karakter ini kemudian diolah dengan beberapa teknik pemrosesan citra digital, yaitu normalisasi ukuran, deteksi tepi, dan *dot orientation*.

D. Ekstraksi Ciri

Pada tahap ini, akan dilakukan ekstraksi ciri terhadap masing- masing citra aksara jawa tulisan tangan. *Pertama*, citra akan dibagi menjadi 7x7 area dengan ukuran masing-masing 16x16 *pixel*. Tiap area akan beririsan satu sama lain dengan area di sekitarnya.

Kedua, setiap area yang ada akan dibagi menjadi menjadi sub-area yang berada di dalam setiap area yang bersangkutan. Setiap area akan memiliki 4 buah sub-area A, B, C, dan D. A adalah sub-area sebesar 4x4 di tengah, B adalah sub-area 8x8 yang eksklusif di luar A, C adalah sub-area 12x12 yang eksklusif terhadap A dan B, dan D adalah sub-area 16x16 area di luar A, B, dan C.

Ketiga, bobot untuk setiap sub-area ditentukan. Untuk mengurangi efek negatif yang disebabkan oleh posisi citra yang bervariasi, besar bobot dibuat semakin besar terhadap sub-area yang semakin ke dalam pada setiap area. Bobot A, B, C, dan D secara berurutan adalah 4, 3, 2, dan 1.

Keempat, untuk setiap area yang ada pada citra akan dibentuk menjadi vektor 4 dimensi (x1, x2, x3, x4) yang merepresentasikan jumlah elemen arah di setiap sub-area. Jumlah elemen di setiap area dihitung sesuai persamaan (1).

Terakhir, jumlah vektor yang didapat dari tiap area disusun menjadi sebuah vektor sesuai dengan persamaan (2), sehingga untuk 1 karakter akan memiliki vektor ciri dengan ukuran 1x196. Vektor ciri inilah yang disebut sebagai DEF yang kemudian akan digunakan, baik dalam proses pembelajaran maupun pengujian.

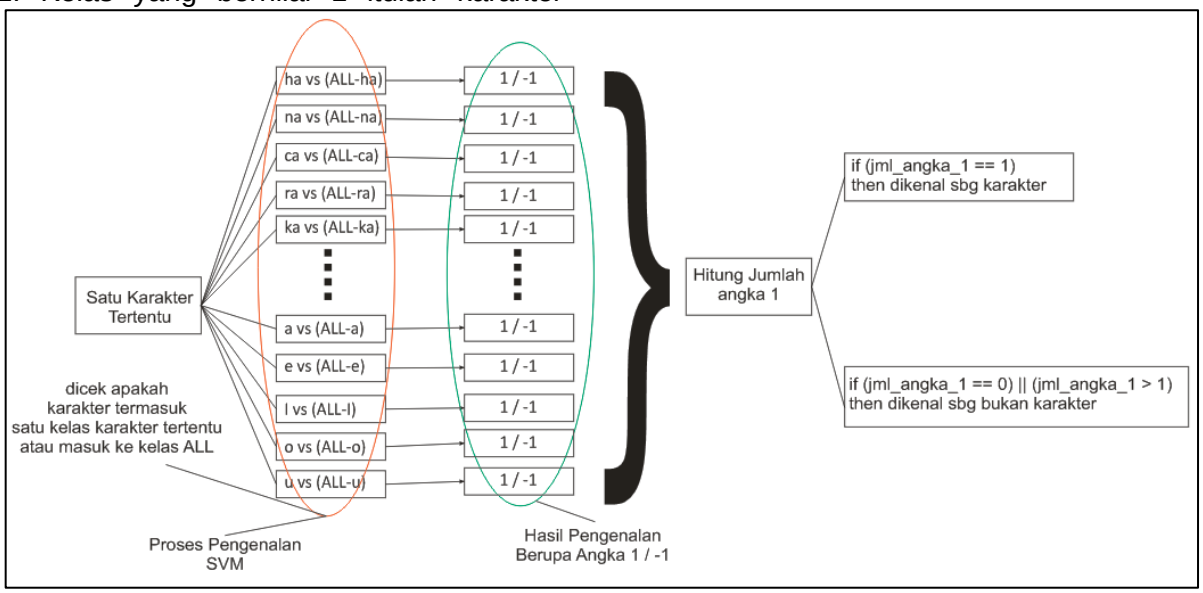
E. Pembelajaran SVM

Pada prinsipnya, proses pembelajaran pada SVM adalah proses untuk menemukan posisi dari *hyperplane* yang optimal sebagai pemisah antar kelas. Metode yang digunakan pada *multi class* SVM di sini ialah “one-againts-all“ sehingga dari proses pembelajaran ini akan dihasilkan 25 *hyperplane*. Masing-masing *hyperplane* tersebut berfungsi untuk memisahkan satu karakter dengan keseluruhan karakter lainnya.

F. Proses Pengenalan

Hyperplane yang dihasilkan pada tahap pembelajaran disimpan dan dijadikan sebagai *classifier* pada proses pengenalan ini. Setiap karakter yang akan dikenali memiliki 25 kemungkinan kelas. Output dari proses pengenalan ini adalah nilai 1 atau -1 untuk setiap kelas.

Dari ke-25 kelas hasil keluaran tersebut, akan dicek manakah yang bernilai 1. Kelas yang bernilai 1 itulah karakter aksara Jawa hasil pengenalan oleh sistem. Namun, apabila suatu karakter dikenali di lebih dari satu kelas, maka karakter tersebut dianggap gagal untuk dikenali sebagai sebuah karakter. Begitu pula jika tidak ada satupun kelas yang menghasilkan nilai 1, maka karakter tersebut dianggap tidak dikenali oleh sistem.



Gambar 11. Ilustrasi Pengenalan Aksara Jawa Tulisan Tangan

VI. PENGUJIAN DAN ANALISIS

A. Skenario Pengujian

Tujuan dari pengujian di sini adalah untuk melihat pengaruh jumlah data latih yang digunakan pada saat fase pelatihan terhadap akurasi sistem dalam mengenali aksara Jawa tulisan tangan. Adapun skenario pengujian yang dilakukan adalah sebagai berikut.

1. Untuk parameter Gamma dan C digunakan nilai 15 dan 30.
2. Jumah citra data latih akan divariasikan untuk setiap pengujian, yakni 5, 10, 15, 20, 25, 30, 35, 40, 45, dan 50 citra untuk setiap karakter aksara jawa.
3. Hasil pelatihan pada masing-masing pengujian, digunakan untuk mengenali aksara Jawa (ha, na, ca, ra, ka, ..., a, e, i, o, u), di mana untuk setiap karakter uji terdiri dari 10 citra (total citra uji = 250).

B. Hasil Pengujian dan Analisis

Dari hasil pengujian yang dilakukan, terlihat bahwa jumlah data latih yang digunakan secara umum dapat mempengaruhi tingkat akurasi sistem dalam mengenali karakter uji. Dari 10 pengujian yang dilakukan, akurasi tertinggi dicapai dengan menggunakan jumlah data latih sebanyak 50 citra untuk setiap karakter aksara jawa, yakni mencapai besar akurasi 93,6%. Sedangkan paling kecil diperoleh dengan jumlah data latih 5 citra per karakter. Tentu hal ini diperoleh dengan ‘pengorbanan‘ waktu proses.

Untuk lebih lengkapnya, berikut rangkuman hasil pengujian pengaruh jumlah data (citra) per karakter.

Tabel 1. Hasil Pengujian Pengaruh Jumlah Data Latih Per Karakter

Jml. Data Latih /karakter	Akurasi
5	70.8
10	82.8
15	84
20	87.6
25	88.4
30	89.6
35	90.4
40	92
45	93.6
50	93.6

Adapun perubahan akurasi setiap karakter seiring dengan perubahan jumlah data latih secara lebih rinci dapat dilihat pada tabel 2.

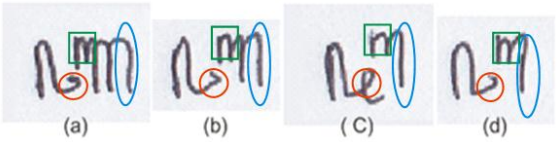
Berdasarkan tabel 2, dapat diamati bahwa akurasi yang paling stabil adalah karakter **ra**, **la**, **ya**, dan **ga**. Keempat karakter tersebut merupakan karakter yang memiliki bentuk yang jauh berbeda dibandingkan karakter lainnya. Karena itulah, meskipun hanya dengan sedikit jumlah data latih, keempat karakter tersebut sudah dapat dikenali dengan tingkat akurasi yang tinggi. Untuk karakter lainnya, kondisinya bervariasi tergantung pada keunikan dan kerumitan bentuknya.

Tabel 2. Perubahan tingkat akurasi setiap karakter seiring dengan perubahan jumlah data latih

Indeks Karakter	Nama Karakter	JUMLAH DATA LATIH/KARAKTER										Rata2/ huruf
		5	10	15	20	25	30	35	40	45	50	
1	ha	70	70	70	80	90	90	90	90	90	90	83
2	na	90	70	70	70	70	80	70	70	80	80	75
3	ca	50	90	90	90	90	90	90	90	90	90	86
4	ra	100	100	100	100	100	100	100	100	100	100	100
5	ka	90	90	100	100	100	100	100	100	100	100	98
6	da	40	40	60	70	70	70	70	80	90	90	68
7	ta	50	60	60	60	70	90	90	90	100	100	77
8	sa	60	80	80	80	80	80	80	90	90	90	81
9	wa	70	90	100	100	100	100	100	100	100	100	96
10	la	100	100	100	100	100	100	100	100	100	100	100
11	pa	50	90	80	100	100	100	100	100	100	100	92
12	dha	40	80	90	90	80	80	80	100	100	100	84
13	ja	90	100	100	100	100	100	100	100	100	100	99
14	ya	100	100	100	100	100	100	100	100	100	100	100
15	nya	50	80	80	90	90	90	90	90	90	90	84
16	ma	90	100	100	100	100	100	100	100	100	100	99
17	ga	100	100	100	100	100	100	100	100	100	100	100
18	ba	50	60	50	50	50	50	50	50	60	60	53
19	tha	60	60	70	70	70	70	70	70	70	70	68
20	nga	40	80	70	80	80	80	80	80	80	80	75
21	a	90	100	100	100	100	100	100	100	100	100	99
22	e	70	80	100	100	100	100	100	100	100	100	95
23	i	90	90	80	100	90	90	100	100	100	100	94
24	o	50	70	70	90	100	100	100	100	100	100	88
25	u	80	90	80	70	80	80	100	100	100	100	88
Rata2 tiap pertambahan data latih		70.8	82.8	84	87.6	88.4	89.6	90.4	92	93.6	93.6	

Untuk karakter **ka**, dibutuhkan data latih sebanyak 15 citra/karakter hingga semuanya dapat dikenali dengan baik. Karakter **ta** membutuhkan data latih 45 citra/karakter; **wa** 15 citra/karakter, **pa** 20 citra/karakter; **dha** 40 citra/karakter; **ja**, **ma**, dan **a** 10 citra/karakter; **e** 15 citra/karakter; **o** 25 citra/karakter; dan **u** 35 citra/karakter. Selebihnya, untuk karakter **ha**, **na**, **ca**, **da**, **sa**, **nya**, **ba**, **tha**, **nga**, **e**, dan **i**, tidak pernah mencapai akurasi sebesar 100% meskipun telah ditambahkan data latih hingga 50 citra/karakter. Khusus untuk karakter **na**, setelah diamati, ternyata 2 karakter ujinya memiliki bentuk yang tidak sempurna dan mirip karakter **da**, sehingga menyebabkan kesalahan pengenalan.

Dilihat dari bentuknya, karakter **nya**, **ba**, **tha**, dan **nga** merupakan karakter yang sulit untuk dikenali karena keempat karakter tersebut memiliki struktur bentuk yang hampir sama dan masing-masing karakternya memiliki dua buah bagian terpisah. Akibatnya, kemungkinan terjadi *error* akan lebih besar pada kondisi-kondisi tertentu. Misalnya, jarak antar dua bagian dalam satu karakter yang terlalu rapat dapat mengakibatkan karakter tidak dikenali. Gambar 12 berikut menunjukkan bagian mana saja yang sering mengakibatkan kesalahan pengenalan karakter **nya**, **ba**, **tha**, dan **nga**.



Gambar 12. Bagian-bagian yang sering mengakibatkan kesalahan klasifikasi pada karakter **nya**, **ba**, **tha**, dan **nga**.

Namun demikian, hasil pada pengujian secara keseluran menunjukkan bahwa metode DEF dan SVM yang digunakan sudah mampu dengan baik melakukan pengenalan aksara Jawa. Kesalahan-kesalahan yang dilakukan lebih banyak disebabkan oleh keunikan dan kerumitan yang dimiliki oleh aksara Jawa itu sendiri. Ke depannya, metode ini dapat dikembangkan lagi ke permasalahan yang lebih spesifik sesuai yang telah diungkapkan di atas, agar performansinya menjadi lebih baik.

VII. SIMPULAN DAN SARAN

A. Simpulan

Penelitian ini menghasilkan beberapa simpulan sebagai berikut.

1. Metode DEF yang dipadukan dengan SVM telah berhasil diterapkan untuk pengenalan aksara Jawa tulisan tangan dengan tingkat akurasi terbaik untuk data uji sebesar 93,6%.
2. Karakter **ra**, **la**, **ya**, dan **ga** merupakan karakter-karakter aksara Jawa yang berbeda dari karakter lainnya sehingga mudah untuk dikenali dengan benar.
3. Karakter **nya**, **ba**, **tha**, dan **nga**, merupakan karakter-karakter yang sulit untuk dikenali karena memiiliki karakteristik bentuk yang rumit, terdiri dari dua bagian terpisah, dan struktur bentuk yang hampir sama.

B. Saran

Adapun saran untuk penelitian selanjutnya antara lain sebagai berikut.

1. Aksara Jawa yang dikenali hendaknya dikembangkan ke jenis aksara yang lain, seperti aksara *murda*, *sandhangan*, pasangan, dan sebagainya.
2. Perlu ada penanganan khusus pada tahap *preprocessing* untuk aksara Jawa yang ditulis miring.
3. Perlu ada penanganan khusus untuk mengatasi karakter-karakter yang sulit dibedakan, yaitu **nya**, **ba**, **tha**, dan **nga**.

DAFTAR PUSTAKA

[1] Anastasia Rita Widiarti and Phalita Nari Wastu, "Javanese Character Recognition Using Hidden Markov Model," World Academy of Science, pp. 261-264, 2009.

[2] Asa Ben-Hur and Jackson Weston, "A User's Guide to Support Vector Machines".

[3] Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet, and Sayan Mukherjee, "Choosing Multiple Parameters for Support Vector Machines," pp. 131-159, 2002.

[4] Christopher J.C Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and Knowledge DIscovery, pp. 121-167, 1998.

[5] Krisantus Sembiring, "Tutorial SVM Bahasa Indonesia," ITB, 2007.

[6] Sandhya Arora et al., "Performance Comparison of SVM and ANN for Handwritten Devnagari Character Recognition," IJCSI, vol. 7, no. International Journal of Computer Science, pp. 18-26, May 2010.

[7] Hyeran Byun and Seong-Whan Lee, "Applications of Support Vector Machines for Pattern Recognition: A Survey," Springer, pp. 213-236, 2002.

[8] Youbin Chen, Youshou Wu, and J.Ross Beveridge, "Analysis and Improvement of Directional Element Feature for Off-line Handwritten Chinese Character Recognition," SPIE, vol. 3305, pp. 72-80, 1998.

[9] Javad Sadri, Ching Y. Suen, and Tien D. Bui, "Application of Support Vector Machines for Recognition of Handwritten Arabic/Persian Digits," MVIP, vol. 1, pp. 300-307, February 2003.

[10] Luiz S. Oliveira and Robert Sabourin, "Support Vector Machines for Handwritten Numerical String Recognition," IEEE, 2004.

[11] Nei Kato, Masato Suzuki, Shin'ichiro Omachi, Hirotomo Aso, and Yoshiaki Nemoto, "A Handwritten Character Recognition System Using Directional Element Feature and Asymmetric Mahalanobis Distance," IEEE, vol. 21, pp. 258-262, March 1999.

[12] Xianliang Wang, Xiaoqing Ding, and Hailong Liu, "Writer Identification Using Directional Element Features and Linear Transform," IEEE, 2003.

[13] Aditya Wibowo, Achmad Hidayatno, and Ajub Ajulian, "Pengenalan Huruf Jawa Tulisan Tangan Menggunakan Jaringan Saraf Tiruan Perambatan Balik dengan Fuzzy Feature Extraction".

[14] Anto Satriyo Nugroho, Arif Budi Witarto, and Dwi Handoko. (2003) IlmuKomputer.com. [Online]. ilmukomputer.com

[15] Brian Karundeng and Kho I Eng, "An Evaluation of Feature Extraction Algorithms for Automatic Language Transcription System for Ancient Handwritting Javanese Manuscripts," in International Seminar on Industrial Engineering and Management, pp. 659-665.

[16] Darma Putra, Pengolahan Citra Digital, Westriningsih, Ed. Yogyakarta, Indonesia: Andi Offset, 2010.

[17] Imran Khan, Smitha U.V, and Suresh Kumar D.S, "Isolated Kannada Character Recognition using Chain Code Features", in International Journal of Science and Research, vol.2, pp. 67-70, 2013

[18] Simon Haykin, Neural Networks and Learning Machines, 3rd ed., Alice Dworkin, Ed. New Jersey: Pearson, 2009.
[19] (2014, March) Wikipedia. [Online]. http://id.wikipedia.org/wiki/Aksara_Jawa
[20] Sidiq Harmawan, "Aplikasi Multimedia Pembelajaran Aksara Jawa untuk Anak-Anak," Universitas Sebelas Maret, Surakarta, Tugas Akhir 2007.
[21] Swastyan Dwi Saputra. (2013) [Online]. http://swastyan.blogspot.com/2013/05/aksara-jawa.html