ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

Handwriting Recognition Hanacaraka Javanese Character using Support Vector Machine and Scale Invariant Feature Transform

Pengenalan Tulisan Tangan Karakter Hanacaraka Aksara Jawa menggunakan *Support Vector Machine* dan *Scale Invariant Feature Transform*

Rama Tri Agung[1], Dessyanto Boedi Prasetyo[2], Mangaras Yanu Florestiyanto[3]

[1,2,3] Informatika, Universitas Pembangunan Nasional Veteran Yogyakarta, Indonesia

[1*]123180053@student.upnyk.ac.id, [2]dess@upnyk.ac.id, [3]mangaras.yanu@upnyk.ac.id

Informasi Artikel
Received:
Revised:
Accepted:
Published:

Abstract

Purpose:
Hanacaraka Javanese script is one of the ancestral heritage of the Indonesian nation inherent in Javanese culture. This Javanese script is threatened with extinction due to the decreasing number of users and also the interest of the community. The Javanese handwritten character recognition system is one way to help the preservation of this culture.
Design/methodology/approach:
As in previous research, the system's application can be developed using a variety of methods, including CNN, KNN, and SVM. Such methods have their own drawbacks. As a result, the SVM method and SIFT as feature extraction to improve classification performance were used in this research. This research used 2940 data with 20 classes that were augmented with seven variations and several preprocessing steps, including resize, grayscaling, and histogram equalization.
Findings/result:
Parameter optimization is accomplished by experimenting with 20 different combinations of several parameters, including image size, K value, C value, and Gamma value. The best parameters were obtained by the test, which were the image size of 192x192 pixels, the value of K = 750, the value of C = 3, and the value of Gamma = 0.10235. The accuracy performance values of the SVM-SIFT model classification using these parameters can reach 92.11% in the training data and 94.55% in the test data. The performance of such models demonstrates that SIFT feature extraction can significantly improve accuracy on SVM models.

Originality/value/state of the art:
This research conducted handwriting recognition on Javanese hancaraka characters using the SVM and SIFT as feature extraction methods. Then 2940 Javanese script hanacaraka data have been augmented with seven variations and preprocessed resizing, grayscaling, and histogram equalization before being trained in the model.

Abstrak

Keywords: javanese script; SVM; SIFT
Kata kunci: aksara jawa; *SVM*; *SIFT*

Tujuan:
Hanacaraka aksara jawa merupakan salah satu warisan leluhur bangsa Indonesia yang melekat dalam budaya Jawa. Bahasa aksara jawa ini terancam punah karena semakin berkurang jumlah penggunanya dan juga pada minat masyarakat. Sistem pengenalan karakter tulisan tangan aksara jawa menjadi salah satu cara untuk menunjang kelestarian budaya tersebut.
Perancangan/metode/pendekatan:
Penerapan sistem dapat dikembangkan dengan beberapa metode seperti yang telah dilakukan pada penelitian sebelumnya yaitu CNN, KNN, dan SVM. Metode tersebut memiliki kelemahannya masing-masing. Sehingga pada penelitian ini menggunakan metode SVM dan SIFT sebagai ekstraksi fitur untuk meningkatkan performa klasifikasi. Penilitian ini menggunakan 2940 data dengan 20 kelas yang telah dilakukan augmentasi tujuh variasi dan beberapa *preprocessing* yaitu *resize*, *grayscaling*, dan *equalization histogram*.
Hasil:
Optimasi parameter dilakukan dengan menguji pada 20 kombinasi terhadap beberapa parameter yaitu *size* gambar, nilai K, nilai C, dan nilai Gamma. Pengujian tersebut mendapatkan parameter terbaik yaitu *size* gambar 192x192 piksel, nilai K = 750, nilai C = 3, dan nilai Gamma = 0.10235. Hasil klasifikasi model SVM-SIFT menggunakan parameter tersebut dapat menghasilkan nilai performa akurasi terbaik yaitu mencapai 92.11% pada data latih dan 94.55% pada data uji. Performa model tersebut membuktikan bahwa ekstraksi fitur SIFT dapat meningkatkan akurasi yang signifikan pada model SVM.
Keaslian/ *state of the art*:
Penelitian ini melakukan pengenalan tulisan tangan pada karakter hancaraka aksara jawa dengan menggunakan metode SVM dan SIFT sebagai ekstraksi fitur. Kemudian 2940 data hanacaraka aksara jawa telah dilakukan

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

diaugmentasi dengan tujuh variasi dan dilakukan *preprocessing resize, grayscaling,* dan *equalization histogram* sebelum dilatih dalam model.

1. Pendahuluan

Aksara Jawa "ha-na-ca-ra-ka" merupakan salah satu warisan leluhur bangsa Indonesia[1]. Aksara Jawa juga bagian dari bahasa Jawa yang melekat dalam budaya Jawa. Ratusan bahasa daerah di Indonesia terancam punah. Salah satu bahasa daerah yang terancam adalah bahasa Jawa[2]. Pengguna bahasa Jawa ini semakin berkurang jumlahnya dan hanya sedikit remaja yang mengenal aksara Jawa dengan jelas[3], karena minat masyarakat terhadap aksara jawa juga sangat memprihatinkan[2]. Melihat kondisi tersebut, maka perlu sebuah sistem yang dapat mengenali huruf-huruf Hanacaraka Aksara Jawa[2]. Pengenalan tulisan tangan ini berguna untuk menunjang kelestarian Aksara Jawa sebagai alat atau perangkat lunak yang memiliki kemampuan untuk mengenali tulisan tangan karakter Aksara Jawa secara otomatis[4].

Pengenalan tulisan tangan pada karakter hanacaraka aksara jawa telah diusulkan oleh beberapa penelitian sebelumnya dengan menggunakan metode yaitu *Convolutional Neural Network* (CNN)[4][5][6], *K-Nearest Neighbor* (KNN)[1], dan *Support Vector Machine* (SVM)[5]. Pada metode CNN memiliki performa klasifikasi yang sangat baik dalam bidang ini dengan tingkat akurasi yang dapat mencapai 94.57%[6], CNN juga mahir dalam menangani inputan yang bersifat *noisy*[7], namun akurasi yang tinggi pada CNN membutuhkan jumlah data latih yang banyak[6] dan dengan metode yang kompleks ini maka arsitekturnya akan cukup sulit dibangun serta dapat terjadinya *overfitting*[7]. KNN adalah metode yang sederhana, efektif, mudah diterapkan, tidak parametrik dan memberikan tingkat kesalahan yang rendah dalam proses pelatihan[8], metode ini tidak membutuhkan jumlah *dataset* yang banyak[1], tapi kekurangannya pada metode KNN relatif memiliki hasil performa yang kurang baik daripada metode lainnya[9] dan sulit menemukan nilai parameter optimal[8]. Kemudian metode SVM merupakan metode paling efektif dalam klasifikasi, terutama populer dalam klasifikasi teks, memiliki akurasi yang cukup tinggi[8], tidak memilki masalah dalam *overfitting* [7], dan tidak membutuhkan jumlah *dataset* yang sangat besar[5]. Namun sayangnya, metode ini cukup sulit untuk mencari model parameter yang cocok maupun optimal dalam klasifikasi[8].

Beberapa penelitian diatas, metode SVM dapat digunakan secara fleksibel tanpa membutuhkan *dataset* yang besar dan memiliki performa yang cukup baik, namun penelitian sebelumnya yang menggunakan metode tersebut belum dapat menyaingi akurasi dari metode CNN yang diatas 90%[1][5]. Untuk dapat meningkatkan akurasi dibutuhkan bantuan fitur ekstraksi, pada penilitian pengenalan tulisan tangan karakter lainnya (Thailand, Bangla dan Latin) telah mengusulkan metode yaitu menggunakan *Scale Invariant Feature Transform Descriptor* (SIFT) yang berpengaruh dalam peningkatan akurasi klasifikasi secara signifikan menjadi diatas 95% dan mengungguli performa fitur ekstraksi *Histograms of Oriented Gradients* (HOG)[10].

Sehingga pada penelitian ini pengenalan tulisan tangan pada karakter hanacaraka aksara jawa akan menerapkan metode SVM dengan bantuan SIFT dalam meningkatkan performa akurasi. Dari metode tersebut akan dibandingkan dan menemukan bagaimana pengaruh terhadap akurasi jika metode SIFT diterapkan dalam ekstraksi fitur.

2. Metode/Perancangan

Pada penelitian ini terdapat metodologi penelitian dengan beberapa tahapan secara sistematis dengan menerapkan metode pengembangan *machine learning*[17][4][18] yang konten dari tahapan itu sendiri telah disesuaikan dengan kebutuhan dari penelitian ini. Adapun tahapan tersebut dipecah menjadi beberapa tahapan yaitu pengumpulan data, persiapan data, pemodelan data, pengujian data.

2.1. Pengumpulan Data

Pada penelitian ini dataset karakter hanacaraka aksara jawa diambil dari *website* Kaggle oleh Phiard yang berjumlah 2632 karakter dengan 20 jenis karakter didalamnya. Namun pada dataset tersebut telah dilakukan augmentasi sejumlah 6 varian setiap gambarnya termasuk varian normal. Sayangnya augmentasi yang dilakukan oleh Phiard tidak signifikan terlihat berbeda dengan gambar asli nya sehingga pada penelitian ini hanya mengambil gambar yang normal sebanyak 420 karakter dengan 20 jenis karakter tanpa augmentasi oleh Phiard. Data gambar yang telah dikumpulkan tersebut merupakan gambar yang berwarna hitam putih dengan ukuran 224x224 piksel.

2.2. Persiapan Data

Setelah mengetahui kebutuhan data yang digunakan, pada tahap persiapan data yaitu data akan diolah terlebih dahulu sebelum masuk pada tahapan pemodelan data, sehingga tahap ini akan menghasilkan keluaran sebuah fitur yang sudah rapi dan siap dilatih. Tahapan ini akan dibagi menjadi tiga tahapan penting yaitu augmentasi data & *preprocessing* gambar, ekstraksi fitur, dan *preprocessing* fitur.

2.2.1. Augmentasi Data & *Preprocessing* Gambar

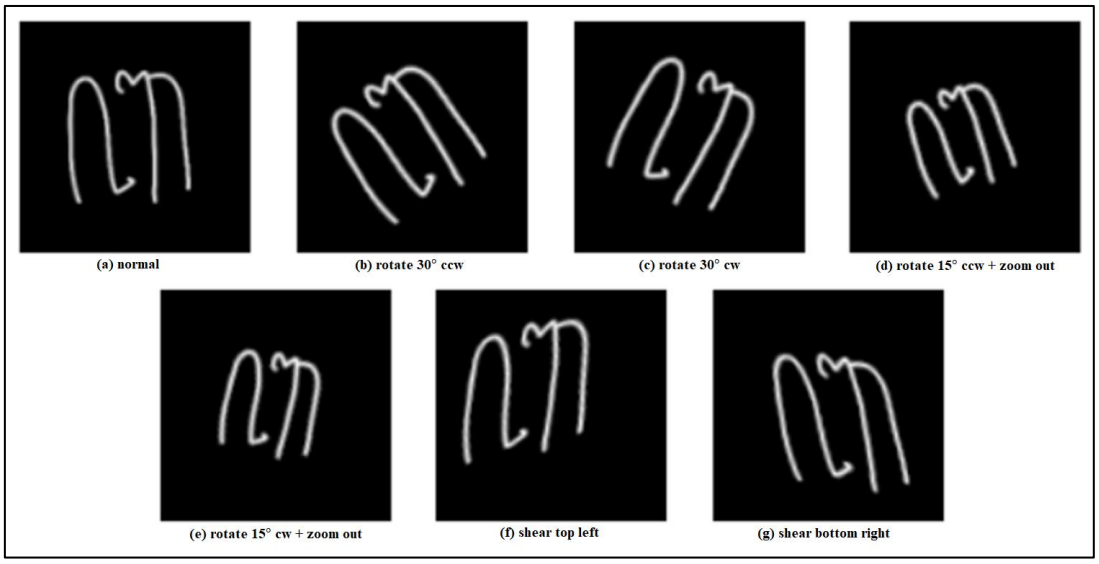
Augmentasi data adalah strategi yang memungkinkan praktisi untuk secara signifikan meningkatkan keragaman data yang tersedia untuk model pelatihan, tanpa benar-benar mengumpulkan data baru[11]. Terdapat banyak metode augmentasi yang dapat dilakukan, namun pada penelitian ini menggunakan tiga metode augmentasi gambar yaitu rotasi gambar, pengecilan gambar, dan *shear* gambar. Penggunaan ketiga metode tersebut masih tetap menjaga kualitas gambar yang ada dan tidak membuat gambar menjadi rusak ataupun sulit dilihat sehingga variasi *dataset* masih layak untuk digunakan. Setelah augmentasi dilakukan maka jumlah *dataset* menjadi 2940 dengan tujuh variasi gambar. Beberapa varian hasil augmentasi yang dilakukan dapat dilihat pada **Tabel 1.**

Selain itu *preprocessing* gambar dilakukan untuk memperbaiki gambar dan menyediakan data gambar yang terbaik. Beberapa hal yang dilakukan pada tahapan ini yaitu mengganti ukuran gambar, mengganti warna gambar menjadi hitam putih dan melakukan perataan histogram pada gambar. Ukuran gambar diganti menjadi 192x192 piksel, kemudian gambar diubah menjadi hitam putih dengan menggunakan metode weighted grayscaling, setelah itu proses terakhir dilakukan dengan meratakan histogram pada sebuah matriks gambar agar membuat gambar menjadi lebih tajam. Berikut **Gambar 1** merupakan hasil dari semua proses pengolahan gambar beserta hasil augmentasinya.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

Tabel 1. Variasi Hasil Augmentasi Gambar

No.	Varian Gambar
1.	Normal
2.	Rotasi 30 derajat berlawanan jarum jam
3.	Rotasi 30 derajat searah arah jarum jam
4.	Rotasi 15 derajat berlawanan jarum jam dan pengecilan gambar
5.	Rotasi 15 derajat searah arah jarum jam dan pengecilan gambar
6.	*Shear* gambar kearah atas kiri
7.	*Shear* gambar kearah bawah kanan



Gambar 1. Hasil Augmentasi dan Preprocessing Gambar pada Karakter ‘ba’

2.2.2. Ekstraksi Fitur

Dalam penelitian ini ekstraksi fitur menggunakan *Scale Invariant Feature Transform* (SIFT). Deskriptor SIFT menciptakan lingkungan 16x16 yang dipartisi menjadi 16 subwilayah masing-masing 4x4 piksel. Untuk setiap piksel dalam subwilayah, SIFT menambahkan vektor gradien piksel ke histogram arah gradien dengan mengkuantisasi setiap orientasi ke salah satu dari 8 arah dan memberi bobot kontribusi setiap vektor berdasarkan besarnya. Setiap arah gradien selanjutnya dibobot dengan skala Gaussian = n/2 di mana n adalah ukuran lingkungan dan nilai-nilai didistribusikan ke bin tetangga menggunakan interpolasi trilinear untuk mengurangi efek batas saat sampel bergerak di antara posisi dan orientasi[12]. Hasil fitur dari SIFT akan diolah kembali ke dalam bentuk *Bag of Feature* (BoF) atau *Bag of Words* (BoW) dengan bantuan metode K-Means dalam klasterisasi seluruh fitur dari semua data gambar yang telah diekstrak. Kemudian fitur akhir akan dihasilkan dengan cara mencari nilai klaster menggunakan K-Means terhadap BoW yang telah dibuat dengan jumlah fitur sebanyak jumlah klaster yang ditentukan. Fitur akhir itulah yang akan menjadi input model yang dibangun. Maka dari itu proses ekstraksi fitur ini akan dibagi menjadi tiga proses utama, yaitu ekstraksi deskriptor dari SIFT, membuat BoW dari deskriptor menggunakan K-Means, dan membuat fitur akhir dari BoW.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

1. Ekstraksi deskriptor dari SIFT

Ada empat langkah utama yang terlibat dalam algoritma ekstraksi fitur SIFT yaitu *scale-space peak selection*, *keypoint localization*, *orientation assignment*, dan *keypoint descriptor*[13][14]. Pertama *scale-space peak selection* dilakukan dengan mendeteksi *points of interest*, atau yang disebut juga *keypoint* pada SIFT. Dambar asli diambil dan menghasilkan urutan gambar blur terus-menerus, kemudian mengubah ukuran gambar asli menjadi 50% dari ukurannya dan menghasilkan gambar blur berulang kali. Citra dikonvolusikan dengan filter *Gaussian* pada skala yang berbeda, kemudian diambil perbedaan dari citra *Gaussian* yang berurutan. *Keypoint* kemudian diambil sebagai maxima/minima dari *Difference of Gaussians* (DoG) yang terjadi pada beberapa skala. Berikut persamaan DoG.

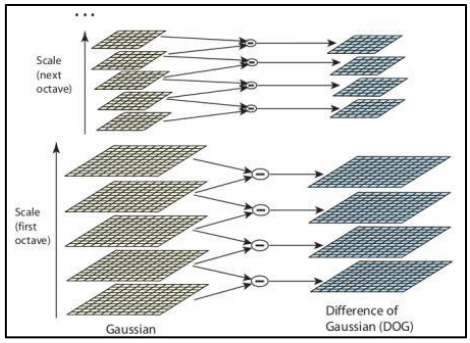
$$D(x, y, \sigma) = L(x, y, k_i\sigma) - L(x, y, k_j\sigma) \quad (1)$$

Dimana $L(x, y, k\sigma)$ adalah konvolusi dari citra asli $I(x, y)$ dengan *Gaussian Blur* $G(x, y, k\sigma)$ dalam skala $k\sigma$ yaitu.

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (2)$$

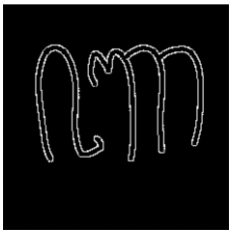
$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (3)$$

Nilai k merupakan nilai kontinu dari beberapa nilai skala yang berbeda-beda untuk membandingkan beberapa hasil dari pengurangan konvolusi *Gaussian Blur*. Pada penelitian ini nilai sigma yang digunakan yaitu $\sigma = 1{,}6$ dan nilai $k = \sqrt{2}$.



Gambar 2. *Scale Space* dan DoG

Salah satu contoh hasil gambar perhitungan DoG yang dilakukan dapat dilihat pada **Gambar 3**.

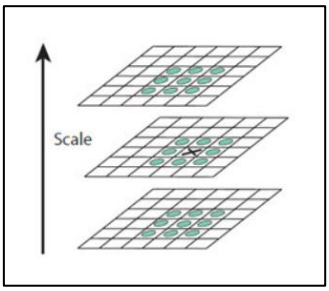


Gambar 3. Hasil Perhitungan DoG

Setelah gambar DoG diperoleh, *keypoints* diidentifikasi sebagai *local minima/maxima* dari gambar DoG di seluruh skala. Hal ini dilakukan dengan membandingkan setiap piksel dalam

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

gambar DoG dengan delapan tetangganya pada skala yang sama dan sembilan piksel tetangga yang sesuai di setiap skala tetangga. Jika nilai piksel adalah maksimum atau minimum di antara semua piksel yang dibandingkan, maka dipilih sebagai *candidate keypoint*.



Gambar 4. Proses pencarian kandidat *keypoint* pada tiap skala

Setelah itu, *keypoint* tersebut akan tersebar disetiap adanya extrema, sehingga banyak *keypoint* yang terbentuk tidak akurat atau tidak tepat karena tidak adanya batasan ataupun *threshold* dalam pencarian sebelumnya. Maka selanjutnya *keypoint localization* dilakukan untuk pencocokan detail fitur ke data terdekat untuk lokasi, skala, dan rasio kelengkungan utama yang akurat. Pertama, untuk setiap titik kunci kandidat, interpolasi data terdekat digunakan untuk menentukan posisinya secara akurat. Interpolasi ini dilakukan menggunakan *Quadratic Taylor Expansion* dari fungsi skala DoG, dengan kandidat *keypoint* sebagai titik asal. Kedua, untuk membuang *keypoint* yang memiliki kontras rendah, maka nilai *Taylor Expansion* orde kedua dihitung pada suatu *offset* tertentu. Terakhir, perlu menghilangkan *keypoint* yang kurang tepat terhadap sejumlah *noise* yang kecil. Pada setiap tepi objek terdapat nilai kelengkungan utama untuk menjadi *keypoint* yang stabil, untuk mengatasinya dapat dilakukan dengan memecahkan nilai *eigenvalues* pada matriks *Hessian* orde kedua. Hasil tahapan ini dilihat pada **Gambar 5**.

Gambar 5. Hasil identifikasi *interest keypoints*

Selanjutnya setiap *keypoint* diberikan satu atau lebih orientasi berdasarkan arah gradien gambar lokal. Ini adalah langkah kunci dalam mencapai invarian terhadap rotasi karena *deskriptor keypoint* dapat direpresentasikan relatif terhadap orientasi ini dan karenanya mencapai invarian terhadap rotasi gambar. Besaran gradien dan orientasi diperoleh dengan menggunakan persamaan. Besaran dan orientasi dihitung untuk semua piksel yang mengelilingi titik-titik kunci. Setelah itu, histogram dapat dibuat.

$$m(x,y) = \sqrt{\big(L(x+1,y) - L(x-1,y)\big)^2 + \big(L(x,y+1) - L(x,y-1)\big)^2} \quad (4)$$

$$\theta(x,y) = atan2(L(x,y+1) - L(x,y-1), L(x+1,y) - L(x-1,y)) \quad (5)$$

Citra $L(x, y, \sigma)$ yang dihaluskan *Gaussian* pada skala *keypoint* σ diambil sehingga semua komputasi dilakukan dengan cara *scale-invariant*. Untuk sampel gambar $L(x, y)$ pada skala σ, besaran gradien $m(x, y)$, dan orientasi $\theta(x, y)$, dihitung terlebih dahulu menggunakan rumus diatas.

Terakhir, menghitung vektor deskriptor untuk setiap *keypoints* sedemikian rupa sehingga deskriptor sangat unik/berbeda dan sebagian tidak berubah untuk variasi yang tersisa seperti iluminasi, sudut pandang 3D, dll. Dengan representasi ini, dimungkinkan dapat dengan mudah memperoleh fitur yang diperlukan. Untuk melakukanya, matriks 16x16 yang disekitar *keypoints* diatur dan matriks ini dibagi menjadi 16 matriks ukuran 4x4. Di dalam setiap matriks 4x4, besaran gradien dan orientasi dapat diperoleh. Histogram ini dibagi menjadi delapan bin dan jumlah orientasi yang ditambahkan ke bin tergantung pada besaran gradien. Sehingga, setiap titik kunci dideskripsikan oleh 4*4*8=128 dimensi vektor.

2. Membuat BoW dari deskriptor menggunakan K-Means

Pada tahapan ini seluruh fitur deskriptor dari setiap gambar dikumpulkan dalam satu data yang digunakan untuk membuat *Bag of Words* (BOW) dengan menerapkan metode *K-Means Clustering*. BoW menjadi kamus dari seluruh fitur penting yang telah di ekstrak sebelumnya. Data berukuran $Nx128$ dimana N adalah jumlah gabungan fitur seluruh citra dan 128 adalah ukuran hasil deskriptor pada setiap fitur. Penggunaan metode K-Means dilakukan untuk mencari satu kelompok/klaster yang dapat mewakili fitur-fitur yang berdekatan menjadi fitur penting. Seluruh nilai *centroids* dari hasil perhitungan K-Means menjadi fitur yang disimpan dalam BoW. Berikut beberapa persamaan yang dibutuhkan pada metode K-Means menggunakan *Euclidean Distance*.

$$\min\left(\sum_{k=1}^{k} d_{ik}\right) = \|x_i - c_k\|^2 \quad (6)$$

Selanjutnya, kelompokkan data-data yang menjadi anggota pada setiap *cluster*. Nilai pusat *cluster* yang baru dapat dihitung dengan cara mencari nilai rata-rata dari data-data yang menjadi anggota pada *cluster* tersebut menggunakan rumus berikut.

$$c_k = \frac{\sum_{i=1}^{p} x_i}{p} \quad (7)$$

Dimana $x_i \in$ *cluster* ke-k dan p adalah banyaknya anggota *cluster* ke-k. Perhitungan dilakukan berulang hingga sudah tidak ada lagi data yang berpindah ke *cluster* yang lain.

3. Membuat fitur akhir dari BoW

Tahapan ekstraksi fitur terakhir membentuk fitur akhir pada setiap citra dari BoW. Pembentukan fitur ini dilakukan mirip seperti proses sebelumnya yaitu melakukan ekstraksi pada sebuah citra menggunakan SIFT untuk mendapatkan beberapa deskriptor unik citra tersebut, kemudian dari fitur deskriptor tersebut dilakukan perhitungan jarak *Euclidean Distance* pada **Persamaan 6** terhadap BoW untuk mendapatkan fitur akhir dari citra tersebut. Fitur dari citra tersebut sama dengan ukuran klaster yang sebelumnya dilakukan yaitu berukuran 750 fitur, isinya merupakan jumlah deskriptor pada klaster-klaster terdekat diantara 750 *centroids* (BoW).

2.2.3. *Preprocessing* Fitur

Pada tahap ini *feature scaling* dan *split dataset* akan dilakukan untuk menyediakan data dengan distribusi yang baik, sehingga perlakuan pada *training* model dapat diterapkan dengan lebih

baik. *Feature scaling* dilakukan dengan metode normalisasi *MinMaxScaling* yang mengubah nilai menjadi rentang 0-1. Selanjutnya *splitting dataset* dilakukan dengan rasio perbandingan 85% : 15%, Sehingga data terpecah menjadi 2499 data latih dan 441 data uji.

2.3. Pemodelan Data

Penelitian ini menggunakan SVM dalam pembuatan model *machine learning*. *Support Vector Machine* (SVM), adalah salah satu algoritma pembelajaran mesin terbaik, yang diusulkan pada 1990-an dan sebagian besar digunakan untuk pengenalan pola[15]. Pada penelitian ini kernel yang digunakan adalah *Radial Basis Function* (RBF). Kernel RBF cocok digunakan dalam kelas yang banyak dan fitur yang banyak. Model yang optimal dari himpunan *hyperplanes* di *data training* dihitung dengan algoritma optimasi SVM[10]. Berikut rumus kernel RBF yang digunakan.

$$f_i = K(x, x'_i) \tag{8}$$

$$K(x, x') = \exp\left(-\frac{\|x,x'\|^2}{2\sigma^2}\right) \tag{9}$$

Dalam kasus klasifikasi *multiclass*, perhitungan dilakukan sebanyak *K* kelas sehingga nilai prediksi akhir berbentuk vektor dengan panjang *K*. Dan nilai bobot akan sebanyak *K*n* fitur.

Dalam penelitian ini, SVM *multiclass* diterapkan dengan metode *one-against-rest (one-vs-all)*. Metode ini melakukan perlakuan terhadap masing-masing kelas dengan membandingkan seluruh sisa kelasnya, apabila terdapat *k* kelas proses klasifikasi dilakukan sebanyak *k* kelas tersebut dengan membandingkan kelas pertama dengan seluruh kelas lainnya begitu pula perlakuan yang sama dilakukan pada kelas lainnya. Ketika salah satu kelas akan dilatih maka kelas lainnya akan digabung menjadi nilai negatif dari kelas yang dituju.

2.4. Pengujian Data

Pada penelitian ini pengujian berfokus pada performa akurasi terhadap model klasifikasi. Pengujian ini dilakukan dengan melakukan optimasi *hyperparameter tuning* untuk membantu model menemukan parameter terbaik agar mendapatkan kinerja yang maksimal. *Hyperparameter tuning* adalah proses mencari nilai parameter optimal dengan menentukan daftar parameter dan rentang pencarian untuk setiap parameter[16]. Ukuran gambar, nilai K pada K-Means, nilai C pada SVM, dan nilai Gamma pada RBF menjadi skenario pengujian optimasi parameter. Pada parameter ukuran gambar terdapat 4 nilai yang akan dikombinasikan dalam satuan piksel yaitu 128x128, 160x160, 192x192, dan 224x224. Kemudian pada parameter nilai K terdapat 5 nilai yang akan dikombinasikan yaitu 180, 250, 500, 750, dan 1000. Selanjutnya pada parameter nilai C dan G memiliki nilai yang dinamis menyesuaikan ukuran dan distribusi masing-masing data.

3. Hasil dan Pembahasan

3.1. Hasil Pengujian

Pengujian dilakukan dengan membandingkan model SVM-SIFT dengan model SVM tanpa ekstraksi fitur. Pada pengujian model SVM-SIFT memiliki 20 skenario dari 4 parameter yang berbeda yaitu *Size* gambar, nilai K, nilai C, dan nilai Gamma. Sedangkan pengujian model SVM tanpa ekstraksi fitur hanya memiliki 2 kombinasi parameter yaitu C dan Gamma. Hasil pengujian skenario SVM-SIFT dapat dilihat pada **Tabel 2**.

Tabel 2. Skenario Pengujian SVM-SIFT

No.	Size	K	C	G	R_Train	R_Test
1	128	180	[1,**3**,6,10,15]	[**0.24574**, 0.25086, 0.25598, 0.2611, 0.26622]	89.31%	92.51%
2	128	250	[1,3,**6**,10,15]	[**0.20504**, 0.20931, 0.21358, 0.21785, 0.22212]	90.63%	92.97%
3	128	500	[1,3,**6**,10,15]	[**0.13567**, 0.1385, 0.14133, 0.14416, 0.14699]	91.55%	95.01%
4	128	750	[1,3,**6**,10,15]	[**0.10508**, 0.10727, 0.10946, 0.11165, 0.11384]	91.99%	94.10%
5	128	1000	[1,3,**6**,10,15]	[**0.09142**, 0.09332, 0.09522, 0.09712, 0.09902]	91.27%	94.55%
6	160	180	[1,**3**,6,10,15]	[**0.25495**, 0.26026, 0.26557, 0.27088, 0.27619]	89.07%	91.69%
7	160	250	[1,**3**,6,10,15]	[0.20114, **0.20533**, 0.20952, 0.21371, 0.2179]	89.95%	90.92%
8	160	500	[1,3,**6**,10,15]	[**0.13022**, 0.13293, 0.13564, 0.13835, 0.14106]	90.63%	91.83%
9	160	750	[1,3,**6**,10,15]	[**0.10009**, 0.10217, 0.10425, 0.10633, 0.10841]	90.79%	93.19%
10	160	1000	[1,**3**,6,10,15]	[**0.08725**, 0.08907, 0.09089, 0.09271, 0.09453]	90.07%	92.97%
11	192	180	[1,**3**,6,10,15]	[0.26274, **0.26821**, 0.27368, 0.27915, 0.28462]	88.47%	90.92%
12	192	250	[1,**3**,6,10,15]	[**0.20183**, 0.20603, 0.21023, 0.21443, 0.21863]	90.91%	91.60%
13	192	500	[1,**3**,6,10,15]	[0.12668, 0.12932, **0.13196**, 0.1346, 0.13724]	91.83%	93.42%
14	192	750	[1,3,**6**,10,15]	[**0.10235**, 0.10448, 0.10661, 0.10874, 0.11087]	92.11%	94.55%
15	192	1000	[1,**3**,6,10,15]	[**0.08487**, 0.08664, 0.08841, 0.09018, 0.09195]	92.67%	95.01%
16	224	180	[1,3,6,**10**,15]	[0.23885, **0.24383**, 0.24881, 0.25379, 0.25877]	85.83%	90.47%
17	224	250	[1,3,**6**,10,15]	[**0.19131**, 0.1953, 0.19929, 0.20328, 0.20727]	87.99%	90.92%
18	224	500	[1,**3**,6,10,15]	[0.11421, **0.11659**, 0.11897, 0.12135, 0.12373]	89.19%	93.19%
19	224	750	[1,3,**6**,10,15]	[**0.09026**, 0.09214, 0.09402, 0.0959, 0.09778]	90.91%	92.51%
20	224	1000	[1,3,**6**,10,15]	[**0.07609**, 0.07767, 0.07925, 0.08083, 0.08241]	90.47%	92.74%

Hasil pengujian performa akurasi model SVM-SIFT tertinggi yaitu pada skenario 14 dengan parameter Size = 192, K = 750, C = 6, dan Gamma = 0.10235. Akurasi yang diperoleh dari skenario tersebut yaitu mencapai 92.11% pada data latih dan 94.55% pada data uji. Selanjutnya pengujian SVM tanpa ekstraksi fitur dengan parameter C = [1, 3, 6, 10, 15] dan untuk nilai G = [0.02265, 0.02312, 0.02359, 0.02406, 0.02453] dapat dilihat pada **Tabel 3.**

Tabel 3. Skenario Pengujian SVM Tanpa Ekstraksi Fitur

kernel	params	rank_test_score	mean_test_score	std_test_score
3_0.02312	{'C': 3, 'gamma': 0.02312}	1	0.511807	0.011146
3_0.02265	{'C': 3, 'gamma': 0.02265}	2	0.511407	0.011168
3_0.02406	{'C': 3, 'gamma': 0.02406}	3	0.510608	0.010277
3_0.02359	{'C': 3, 'gamma': 0.02359}	4	0.510207	0.010708
15_0.02265	{'C': 15, 'gamma': 0.02265}	5	0.509408	0.013446
10_0.02265	{'C': 10, 'gamma': 0.02265}	5	0.509408	0.013446
6_0.02265	{'C': 6, 'gamma': 0.02265}	5	0.509408	0.012712
6_0.02312	{'C': 6, 'gamma': 0.02312}	8	0.509407	0.013052
3_0.02453	{'C': 3, 'gamma': 0.02453}	9	0.509007	0.010085
10_0.02312	{'C': 10, 'gamma': 0.02312}	9	0.509007	0.012622
15_0.02312	{'C': 15, 'gamma': 0.02312}	9	0.509007	0.012622
6_0.02406	{'C': 6, 'gamma': 0.02406}	12	0.507407	0.014696
10_0.02406	{'C': 10, 'gamma': 0.02406}	13	0.507406	0.013294
10_0.02359	{'C': 10, 'gamma': 0.02359}	13	0.507406	0.012423
15_0.02359	{'C': 15, 'gamma': 0.02359}	13	0.507406	0.012423
15_0.02406	{'C': 15, 'gamma': 0.02406}	13	0.507406	0.013294
6_0.02359	{'C': 6, 'gamma': 0.02359}	17	0.507006	0.013539
6_0.02453	{'C': 6, 'gamma': 0.02453}	18	0.507004	0.014284
10_0.02453	{'C': 10, 'gamma': 0.02453}	19	0.506604	0.014053

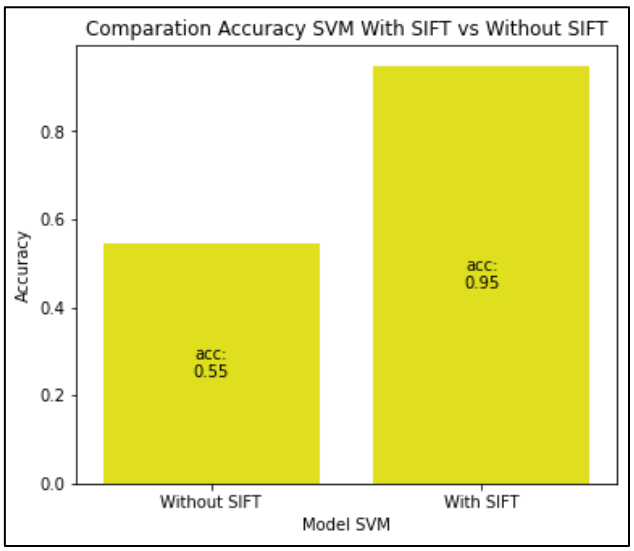
ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

15_0.02453	{'C': 15, 'gamma': 0.02453}	19	0.506604	0.014053
1_0.02453	{'C': 1, 'gamma': 0.02453}	21	0.449780	0.006010
1_0.02312	{'C': 1, 'gamma': 0.02312}	22	0.447782	0.010193
1_0.02406	{'C': 1, 'gamma': 0.02406}	23	0.447780	0.006575
1_0.02359	{'C': 1, 'gamma': 0.02359}	24	0.447381	0.007565
1_0.02265	{'C': 1, 'gamma': 0.02265}	25	0.445781	0.009792

Hasil pengujian performa akurasi model SVM tanpa ekstraksi fitur tertinggi yaitu pada parameter C = 3, dan Gamma = 0.02312. Akurasi yang diperoleh dari parameter tersebut yaitu mencapai 51.18% pada data latih dan 54.64% pada data uji.

3.2. Pembahasan

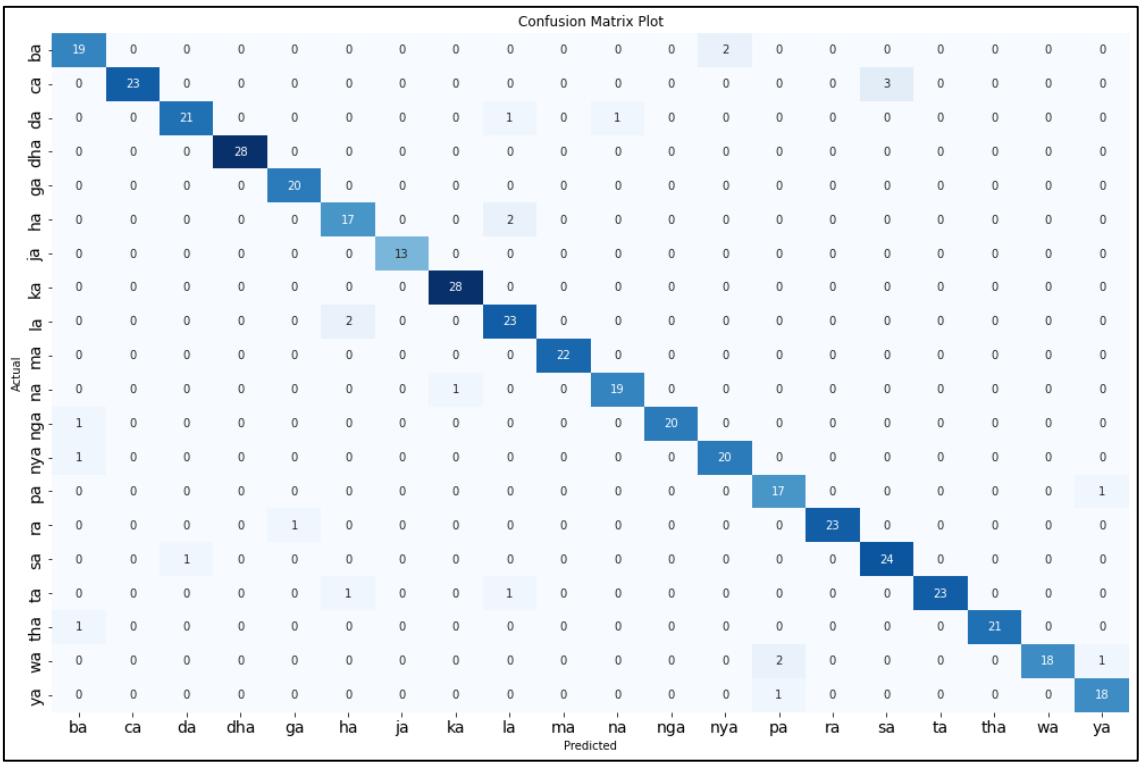
Optimasi model SVM-SIFT diterapkan dengan melakukan *hyperparameter tuning* pada beberapa kombinasi parameter *Size*, nilai K, nilai C, dan nilai G yang telah dijelaskan sebelumnya pada Tabel 2. Optimasi tersebut memperoleh hasil terbaik dengan nilai performa akurasi yang tinggi pada skenario 14 dengan parameter Size = 192, K = 750, C = 6, dan G = 0.10235, yaitu memperoleh akurasi terhadap data latih mencapai 92.11% dan data uji mencapai 94.55%. Pengujian juga dilakukan pada model SVM tanpa SIFT dengan optimasi parameter yang dilakukan pada nilai C dan nilai G. Model SVM tanpa SIFT memperoleh performa akurasi yang tidak cukup baik yaitu mencapai 51.18% pada data latih dan 54.64% pada data uji dengan parameter nilai C = 3 dan nilai G = 0.02312.



Gambar 6. Perbandingan Akurasi Model

Perbedaan performa akurasi terhadap model SVM-SIFT dan SVM tanpa SIFT memiliki selisih yang sangat besar, perbedaan tersebut membuktikan bahwa penggunaan metode ekstraksi fitur SIFT terhadap model SVM memiliki pengaruh yang besar terhadap peningkatan akurasi klasifikasi. Tidak hanya membantu meningkatkan akurasi, model SVM-SIFT ini juga berhasil diterapkan dengan performa akurasi yang diperoleh sangat baik diatas 90% pada dataset yang digunakan penelitian ini. Berikut hasil *confussion matrix* model SVM-SIFT pada **Gambar 7**.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx



Gambar 7. *Confussion Matrix* Model SVM-SIFT

4. Kesimpulan dan Saran

Pengenalan 20 karakter hanacaraka aksara jawa menggunakan metode SVM dan SIFT sebagai ekstraksi fitur berhasil diterapkan dengan jumlah data 2940 yang telah dilakukan augmentasi dan *image preprocessing*. Metode SIFT berhasil bekerja sebagai ekstraksi fitur, kemudian fitur *keypoint* dipilih dan dimasukkan ke dalam BoW menggunakan metode K-Means. BoW tersebut digunakan untuk membuat fitur akhir dari SIFT. Metode SVM berhasil bekerja menggunakan kernel RBF dari data latih berjumlah sebesar 2499 dan data uji sebesar 441. Evaluasi juga dilakukan dengan mengidentifikasi performa akurasi dari model SVM-SIFT yang telah dibangun. Hasil performa terbaik yang diperoleh yaitu pada pengujian skenario 14 dimana ukuran gambar = 192, K = 750, C = 6, dan G = 0.10235. Skenario tersebut memperoleh tingkat akurasi yang tinggi yaitu 92.11% pada data latih dan 94.55% pada data uji. Performa akurasi yang diperoleh pada model SVM-SIFT sangat tinggi dibandingkan dengan performa akurasi yang diperoleh pada model SVM tanpa SIFT yaitu hanya dapat mencapai akurasi 54.64% pada data uji.

Saran pada penelitian ini yaitu pada proses tahapan *image preprocessing* dapat menggunakan metode lainnya untuk dapat meningkatkan kualitas dari data gambar. Selanjutnya dalam penelitian lebih lanjut dapat juga melakukan segmentasi huruf-huruf hanacaraka dari serangkaian kata ataupun kalimat. Kemudian dalam hal evaluasi penelitian juga dapat diperluas lebih lanjut dengan memperhatikan bagaimana performa selain akurasi seperti kompleksitas waktu maupun komputasi.

Daftar Pustaka

[1] C. A. Sari, M. W. Kuncoro, D. R. I. M. Setiadi, and E. H. Rachmawanto, "Roundness and eccentricity feature extraction for Javanese handwritten character recognition based on K-nearest neighbor" in *2018 International Seminar on Research of Information Technology and Intelligent Systems*, 2018, pp. 5-10.

[2] C. A. Lorentius, R. Adipranata, and A. Tjondrowiguno, "Pengenalan Aksara Jawa dengan Menggunakan Metode Convolutional Neural Network" in *Jurnal Infra*, vol. 7, no.1, 2019.

[3] A. Setiawan, and A. M. Sulaiman, "Hancaraka: Aksara Jawa Dalam Karakter Font dan Aplikasinya Sebagai Brand Image" in *Ornamen Jurnal Kriya*, vol. 12, no. 1, pp. 33-47, 2015.

[4] C. K. Dewa, A. L. Fadhilah, and A. Afiahayati, "Convolutional Neural Networks for Handwritten Javanese Character Recognition" in *Indonesian Journal of Computing and Cybernetics Systems*, vol. 12, no. 1, pp. 83-94, 2018.

[5] Rismiyati, Khadijah, and N. Adi, "Deep learning for handwritten Javanese character recognition" in *2017 1st International Conference on Informatics and Computational Sciences*, 2017, pp. 59-63.

[6] M. A. Wibowo, M. Soleh, W. Pradani, A. N. Hidayanto, and A. M. Arymurthy, "Handwritten Javanese Character Recognition using Descriminative Deep Learning Technique" in *2017 2nd International Conferences on Information Technology, Information Systems and Electrical Engineering*, 2017, pp. 325-330.

[7] S. R. Rajesh, A. Beaula, P. Marikkannu, A. Sungheetha, and C Sahana, "Comparative study of distinctive image classification techniques" in *2016 10th International Conference on Intelligent Systems and Control*, 2016.

[8] P. Thamilselvana, and J. G. R. Sathiaseelan, "A Comparative Study of Data Mining Algorithms for Image Classification" in *International Journal of Education and Management Engineering*, vol. 5, no. 2, pp. 1-9, 2015.

[9] M. F. Naufal, S. F. Kusuma, Z. A. Prayuska, and A. Alexander, "Comparative Analysis of Image Classification Algorithms for Face Mask Detection" in *Journal of Information Systems Engineering and Business Intelligence*, vol. 7, no. 1, pp. 56-66, 2021.

[10] O. Surinta, M. F. Karaaba, L. R. B. Schomaker, and M. A. Wiering, "Recognition of handwritten characters using local gradient feature descriptors" in *Engineering Applications of Artificial Intelligence*, vol. 45, pp. 405-414, 2015.

[11] J. Sanjaya, and M. Ayub, "Augmentasi DataPengenalan Citra Mobil Menggunakan Pendekatan Random Crop, Rotate, dan Mixup" in *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 6, no. 2, pp. 311-323, 2020.

[12] E. N. Mortensen, H. Deng, and L. Shapiro, "A SIFT Descriptor with Global Context" in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005.

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI:10.31515/telematika.v18i1.xxxx

[13] Y. Wang, Z. Li, L. Wang, and M. Wang, "A Scale Invariant Feature Transform Based Method" in *Journal of Information Hiding and Multimedia Signal Processing*, vol. 4, no. 2, pp. 73- 89, 2013.

[14] A. K. A. Hassan, B. S. Mahdi, and A. A. Mohammed, "Arabic Handwriting Word Recognition Based on Scale Invariant Feature Transform and Support Vector Machine" in *Iraqi Journal of Science*, vol. 60, no. 2, pp. 381-387, 2019.

[15] A. Pradhan, "Support Vector Machine-A Survey" in *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, no. 8, pp. 82-85, 2012.

[16] E. Andini, M. R. Faisal, R. Herteno, R. A. Nugroho, F. Abadi, and Muliadi, "Peningkatan Kinerja Prediksi Cacat Software Dengan Hyperparameter Tuning Pada Algoritma Klasifikasi Deep Forest" in *Jurnal MNEMONIC*, vol. 5, no. 2, pp. 119-127, 2022.

[17] M. A. Rasyidi, T. Bariyah, Y. I. Riskajaya, and A. D. Septyani, "Classification of handwritten javanese script using random forest algorithm" in *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 3, pp. 1308-1315, 2021.

[18] C. Schröera, F. Kruse, and J. M. Gómez, "A Systematic Literature Review on Applying CRISP-DM Process Model" in *Procedia Computer Science*, vol. 181, pp. 526-534, 2021.