



Penerapan Algoritma *Support Vector Machine* Dan *Scale Invariant Feature Transform* untuk Pengenalan Tulisan Tangan pada Karakter Hanacaraka Aksara Jawa

Rama Tri Agung
123180053

Penguji:

- Dessyanto Boedi Prasetyo, S.T., M.T.
- Mangaras Yanu Florestiyanto, S.T., M.Eng.
- Oliver Samuel Simanjuntak, S.Kom., M.Eng.
- Wilis Kaswidjanti, S.Si., M.Kom.



Pembahasan

01

Pendahuluan

02

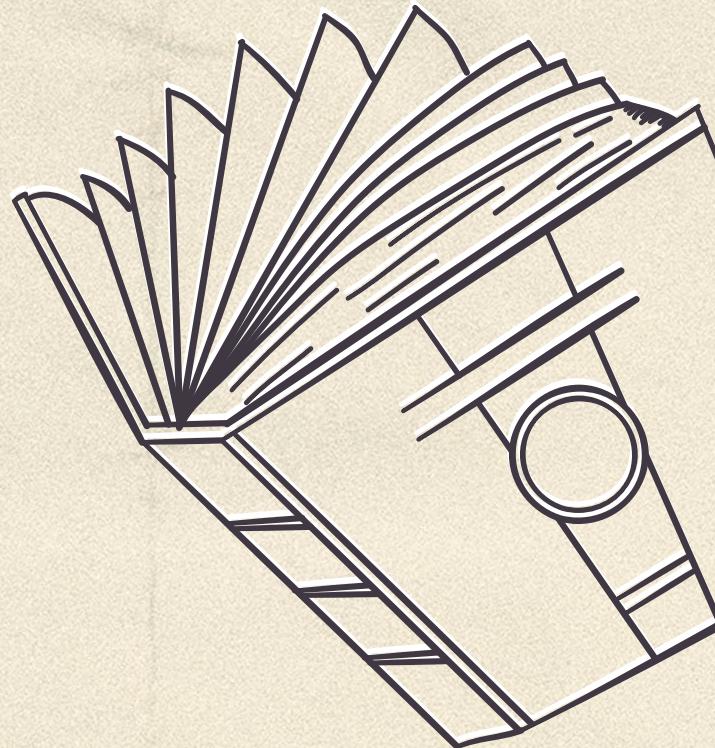
Tinjauan Literatur

03

Metode Penelitian
dan Pengembangan
Sistem

04

Demo Program



01

Pendahuluan

Latar Belakang
Rumusan Masalah
Batasan Penelitian
Tujuan Penelitian
Manfaat Penelitian



Latar Belakang

- Aksara Jawa "ha-na-ca-ra-ka" merupakan salah satu warisan leluhur bangsa Indonesia (Sari et al., 2018).
- Ratusan bahasa daerah di Indonesia terancam punah. Salah satu bahasa daerah yang terancam adalah bahasa Jawa (Lorentius et al., 2019).
- Pengguna bahasa Jawa ini semakin berkurang jumlahnya dan hanya sedikit remaja yang mengenal aksara Jawa dengan jelas (Setiawan et al., 2019)
- Minat masyarakat terhadap aksara jawa juga sangat memprihatinkan (Lorentius et al., 2019).



Latar Belakang

- Melihat kondisi tersebut, bagaimana pentingnya nilai dan eksistensi budaya tersebut, maka perlu sebuah sistem yang dapat mengenali huruf-huruf Hanacaraka Aksara Jawa (Lorentius et al., 2019).
- 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 (Dewa et al., 2018).



Latar Belakang

- CNN memiliki performa klasifikasi yang sangat baik dalam bidang ini dengan tingkat akurasi yang dapat mencapai 94.57% (Wibowo et al., 2018), CNN juga mahir dalam menangani inputan yang bersifat noisy (Rajesh et al., 2016), namun akurasi yang tinggi pada CNN membutuhkan jumlah data training yang banyak (Wibowo et al., 2018)
- KNN adalah metode yang sederhana, efektif, mudah diterapkan, tidak parametrik dan memberikan tingkat kesalahan yang rendah dalam proses pelatihan (Thamilselvana & Sathiaseelan, 2015), kekurangannya pada metode KNN relatif memiliki hasil performa yang kurang baik daripada metode lainnya (Naufal et al., 2021)
- SVM merupakan metode paling efektif dalam klasifikasi, memiliki akurasi yang cukup tinggi (Thamilselvana & Sathiaseelan, 2015), tidak memiliki masalah dalam overfitting (Rajesh et al., 2016), dan tidak membutuhkan jumlah dataset yang sangat besar (Rismiyati et al., 2018).



Latar Belakang

- 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% (Rismiyati et al., 2018) (Sari et al., 2018).
- Pada penilitian pengenalan tulisan tangan karakter lainnya (Thailand, Bangla dan Latin) telah mengusulkan metode yaitu menggunakan Scale Invariant Feature Transform Descriptor (SIFT Descriptor) yang berpengaruh dalam peningkatan akurasi klasifikasi secara signifikan menjadi diatas 95% dan mengungguli performa fitur ekstraksi Histograms of Oriented Gradients (HOG) (Surinta et al., 2015).



Rumusan Masalah



Menerapkan algoritma SVM
dalam klasifikasi

Menerapkan algoritma SIFT
sebagai ekstraksi fitur

Evaluasi performa akurasi
dalam klasifikasi



Batasan Masalah

- Klasifikasi dilakukan hanya pada 20 karakter Hanacaraka Aksara Jawa.
- Sumber data berasal dari dataset yang disediakan di www.kaggle.com oleh Phiard.
- Data diaugmentasi dengan tujuh variasi.
- Data yang digunakan dalam format gambar.
- Analisis dilakukan untuk melihat performa algoritma menggunakan akurasi klasifikasi.



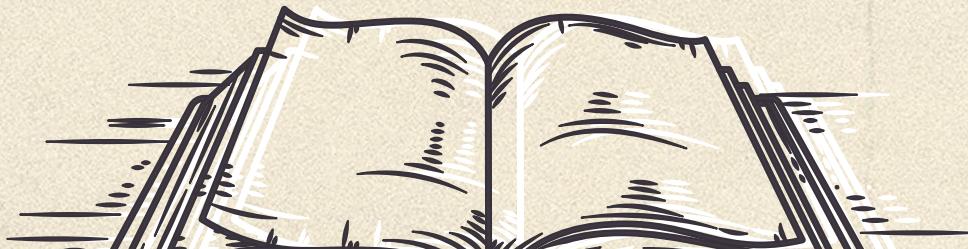
Tujuan Penelitian

Tujuan dari penelitian ini yaitu menerapkan algoritma SVM dengan bantuan SIFT sebagai ekstraksi fitur dalam melakukan klasifikasi tulisan tangan aksara jawa dan mengidentifikasi performa akurasi algoritma yang terbaik dalam melakukan klasifikasi.



Manfaat Penelitian

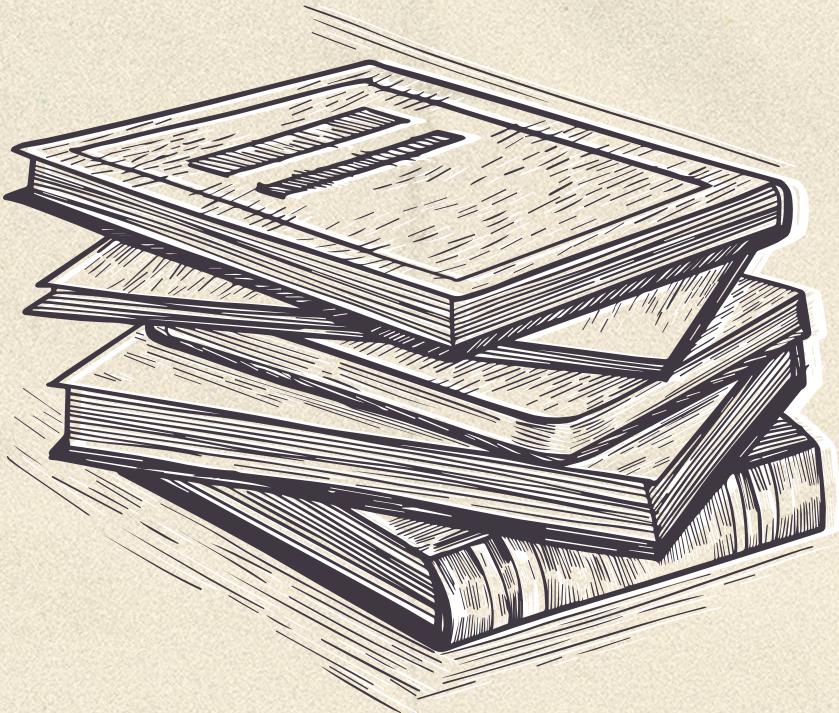
Hasil penelitian dapat dimanfaatkan dalam bantuan pembelajaran baik pada siswa di instansi pendidikan maupun orang lain secara individu dan membangun kembali budaya bahasa khas jawa dengan mengenal lebih mudah terhadap karakter-karakternya.



02

Tinjauan Literatur

Penelitian Sebelumnya
Penelitian Yang Akan Dilakukan



Penelitian Terdahulu

**Wibowo et al.
(2018)**

Hasil penelitian menggunakan metode CNN menunjukkan angka yang memuaskan dengan tingkat akurasi sebesar 94.57%, presisi 94.75%, recall 94.57%, dan F1 score 94.66%.

CNN

CNN dan MLP

Hasil pengujian menunjukkan akurasi dari model CNN mampu mengungguli akurasi dari model MLP yaitu sebesar 89% meskipun CNN membutuhkan waktu latih yang lebih lama dibandingkan dengan MLP.

**Dewa et al.
(2018)**

**Sari et al.
(2018)**

Dengan menggunakan dataset yang cukup kecil sebesar 240 data klasifikasi KNN yang dibantu oleh Feature Extraction Roundness dan Eccentricity dengan nilai K sebesar 3 menghasilkan akurasi yang cukup optimal yaitu sebesar 87.5%.

KNN dengan
Feature Extraction
Roundness dan
Eccentricity

Penelitian Terdahulu

Rismiyati et al.
(2018)

Hasil percobaan menunjukkan bahwa fitur HOG mampu menunjukkan akurasi yang lebih tinggi dibandingkan dengan fitur berbasis zona sederhana (88,45%). Di sisi lain, meskipun sederhana, fitur berbasis zona mampu mencapai akurasi 81,98%.

SVM using
HOG or Zone
Base
Features

Random Forest

Xulianti et al.
(2019)

Hasil penelitian menunjukkan bahwa dengan menggunakan data yang banyak sebesar 21000 data training dan 9000 data testing menghasilkan nilai akurasi, presisi, dan recall yang tinggi sebesar 97,7% tanpa menggunakan proses thinning dan HOG serta parameter impurity measure yaitu gini dan jumlah tree yaitu 1800.

Rasyidi et al.
(2021)

Hasil penelitian dengan melakukan preprocessing (greyscaling, binarization, cropping, resizing 28x28, dan thinning), moment invariant sebanyak 112 fitur, dan SVM mendapatkan hasil akurasi optimal sebesar 92,52%.

SVM dengan
Moment Invariant

Penelitian Terdahulu

Susanto et al.
(2021)

Peningkatan akurasi terjadi sekitar 4% ketika menggunakan model KNN-median filter-HOG pada 1000 data karakter aksara jawa. Akurasi tertinggi mencapai 98,5% dengan nilai K adalah 1 dan rasio pembagian dataset 80:20

KNN & HOG

KNN dan SVM
menggunakan
Local Gradient
Feature

Hasil menunjukkan bahwa deskriptor fitur gradien lokal secara signifikan mengungguli secara langsung menggunakan intensitas piksel dari gambar. Fitur HOG dan SIFT memberikan performa yang sangat baik dan signifikan ketika digabungkan dengan metode SVM dengan akurasi mendekati 100%

Surinta et al.
(2015)

Hassan et al.
(2019)

Hasil penelitian yang dilakukan dengan menggunakan 2072 data latih dan beberapa tahapan yaitu preprocessing untuk konversi warna, fitur ekstraksi menggunakan SIFT, bantuan feature selection K-Means & FINN, dan klasifikasi menggunakan SVM menghasilkan akurasi tinggi sebesar 99.08%

SVM dan SIFT

Penelitian yang akan dilakukan

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.

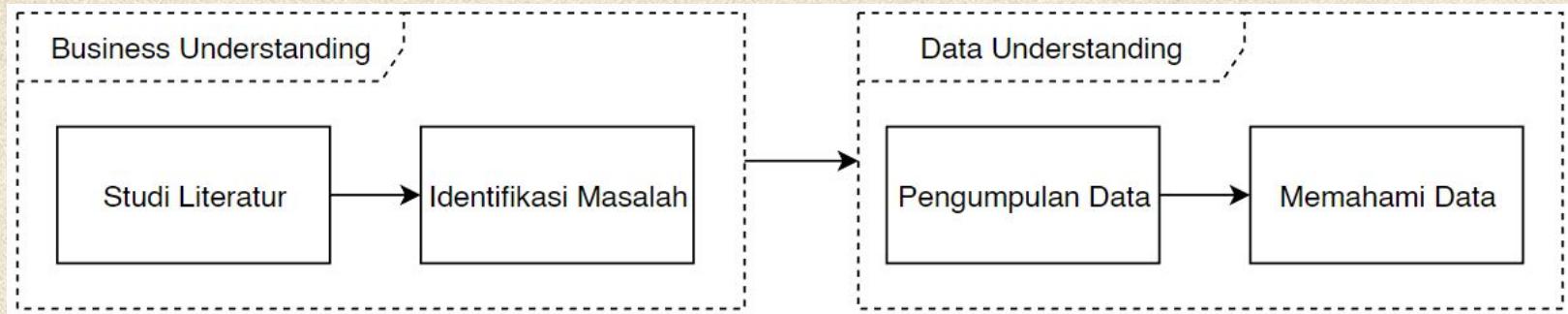




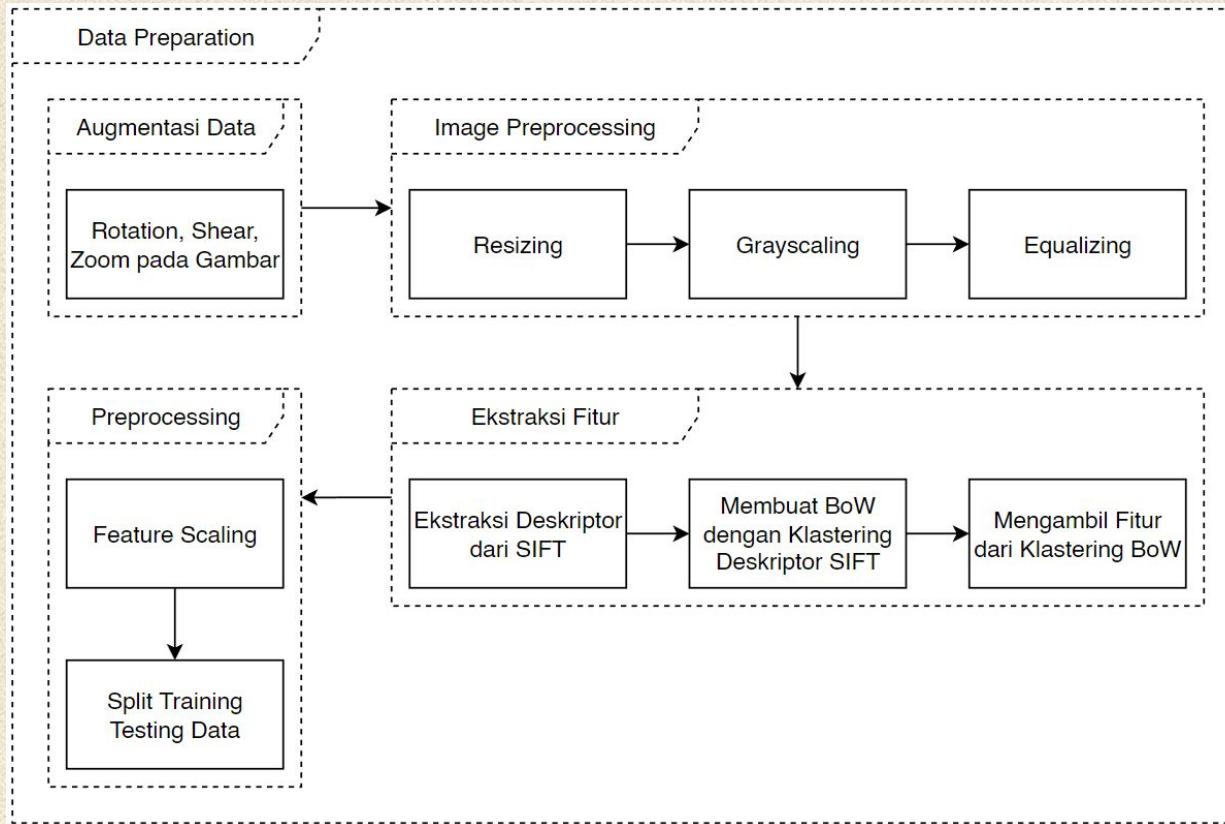
03

Metodologi Penelitian

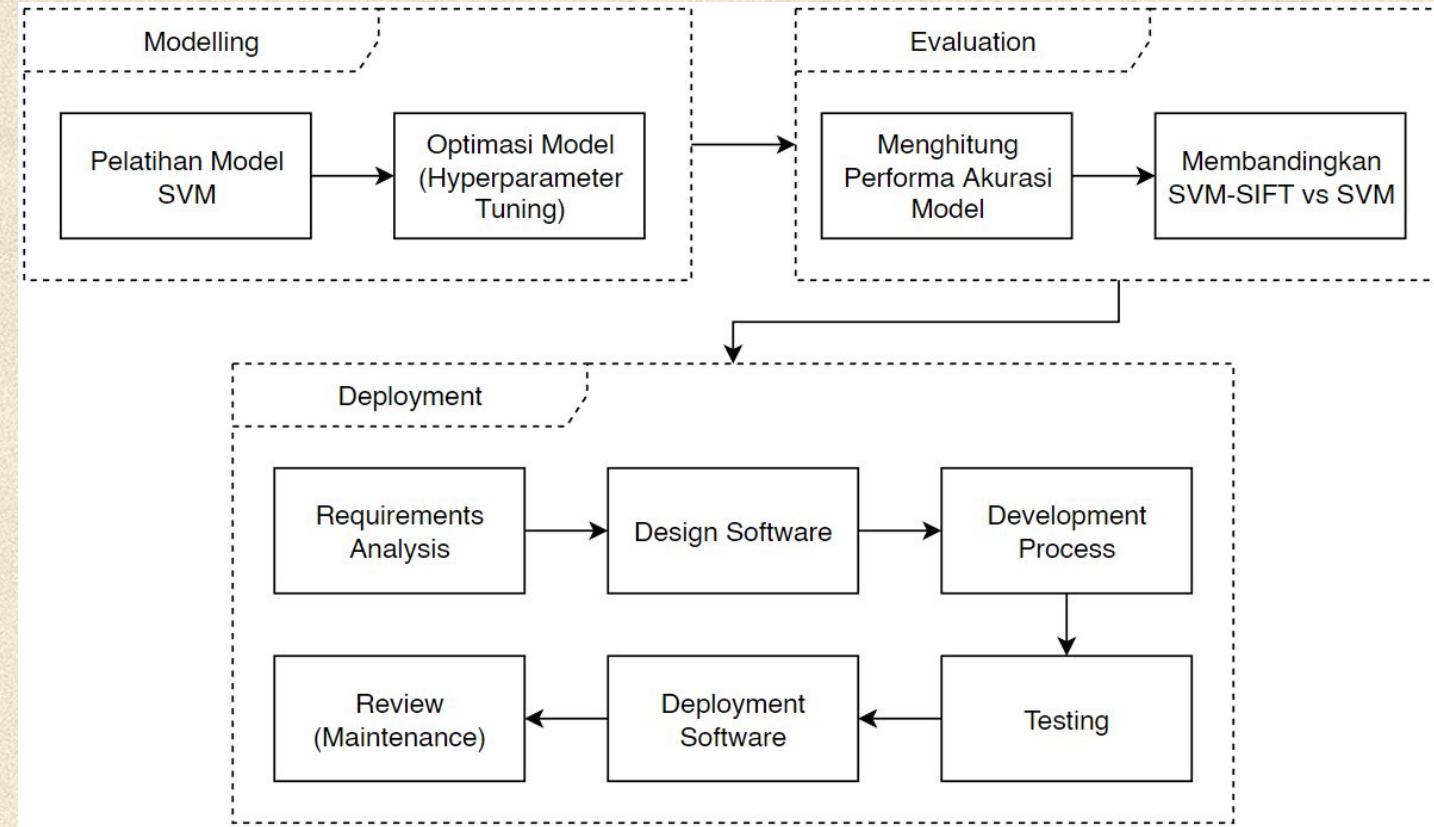
Metodologi Penelitian



Metodologi Penelitian

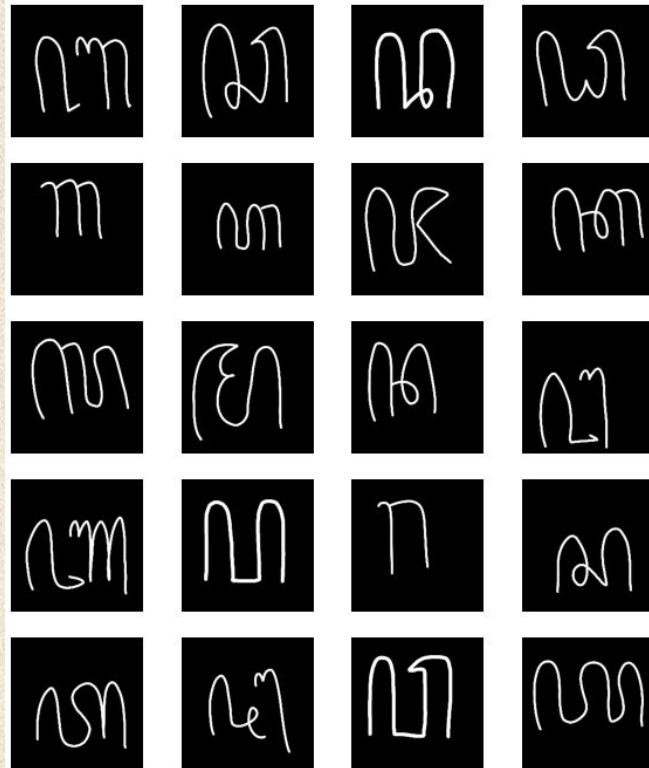


Metodologi Penelitian



Data Understanding

- Pada penelitian ini dataset karakter hanacaraka aksara jawa diambil secara online dari website Kaggle oleh Phiard
- Berjumlah sebanyak 2632 karakter dengan 20 jenis karakter didalamnya.
- Namun pada dataset tersebut telah dilakukan augmentasi sejumlah 6 varian setiap gambarnya termasuk varian normal.
- Augmentasi yang dilakukan oleh Phiard tidak signifikan terlihat berbeda dengan gambar aslinya.
- Pada penelitian ini hanya mengambil gambar yang normal sebanyak 420 karakter dengan 20 jenis karakter tanpa augmentasi oleh Phiard.
- Gambar berwarna hitam putih
- Ukuran 224 x 224 piksel



Augmentasi Data

Tujuh Varian Augmentasi

1. Gambar normal
2. Rotasi 30 derajat searah jarum jam
3. Rotasi 30 derajat berlawanan arah jarum jam
4. Rotasi 15 derajat searah jarum jam dan pengecilan gambar
5. Rotasi 15 derajat berlawanan arah jarum jam dan pengecilan gambar
6. Shear gambar kearah atas kiri
7. Shear gambar kearah bawah kanan

Sebanyak 420 data yang diaugmentasikan hingga menjadi
2940 data



Preprocessing Gambar

Resizing

Melakukan pengecilan ukuran gambar menjadi 192x192 untuk mempercepat pemrosesan

Equalization Hist

Meningkatkan kualitas dan ketajaman warna

Grayscale

Mengubah warna gambar yang awalnya memiliki 3 channel menjadi 1 channel warna keabuan

Ekstraksi Fitur (1. Ekstraksi SIFT)

Membuat gaussian kernel:

$$G_{[0]}M_{[i,j]}(x_i, y_j, \sigma_0) = \frac{1}{2\pi\sigma_s^2} e^{-\frac{x_i^2+y_j^2}{2\sigma_s^2}}$$

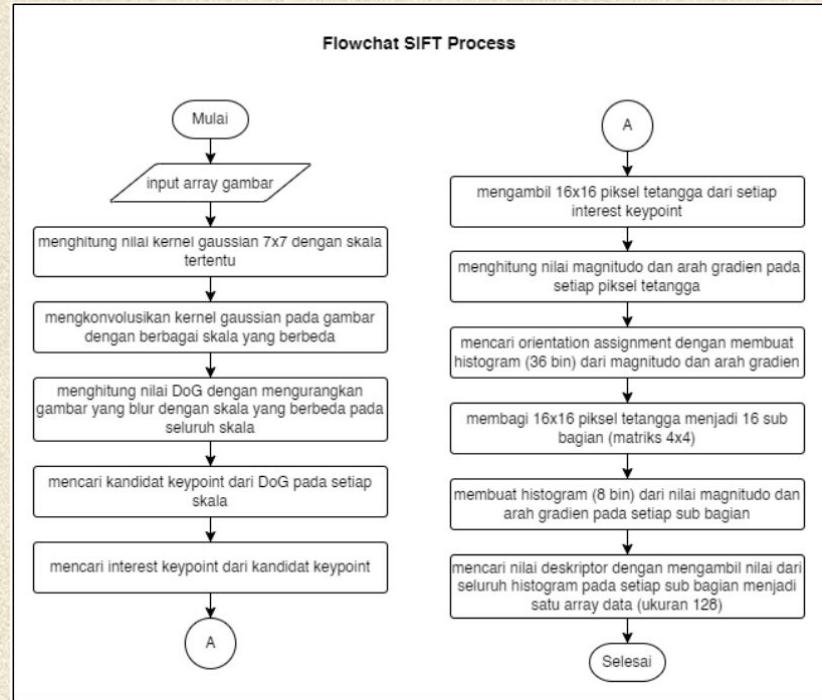
$$G_{[0]}M_{[0,0]}(x_0, y_0, k^0\sigma) = \frac{1}{2\pi(k^0\sigma)^2} e^{-\frac{x_0^2+y_0^2}{2(k^0\sigma)^2}}$$

$$G_{[0]}M_{[0,0]}(x_0, y_0, k^0\sigma) = \frac{1}{2\pi(\sqrt{2}^0 \cdot 1,6)^2} e^{-\frac{-3^2+ -3^2}{2(\sqrt{2}^0 \cdot 1,6)^2}}$$

$$G_{[0]}M_{[0,0]}(x_0, y_0, k^0\sigma) = \frac{1}{16.08} e^{-3.515625}$$

$$G_{[0]}M_{[0,0]}(x_0, y_0, k^0\sigma) = 0.00184826$$

Tabel 3.1 Kernel Matriks dari Gaussian Filter						
0.0018483	0.0049077	0.0088176	0.0107194	0.0088176	0.0049077	0.0018483
0.0049077	0.0130315	0.0234134	0.0284635	0.0234134	0.0130315	0.0049077
0.0088176	0.0234134	0.0420663	0.0511396	0.0420663	0.0234134	0.0088176
0.0107194	0.0284635	0.0511396	0.0621699	0.0511396	0.0284635	0.0107194
0.0088176	0.0234134	0.0420663	0.0511396	0.0420663	0.0234134	0.0088176
0.0049077	0.0130315	0.0234134	0.0284635	0.0234134	0.0130315	0.0049077
0.0018483	0.0049077	0.0088176	0.0107194	0.0088176	0.0049077	0.0018483



Ekstraksi Fitur (1. Ekstraksi SIFT)

Konvolusikan kernel dengan gambar

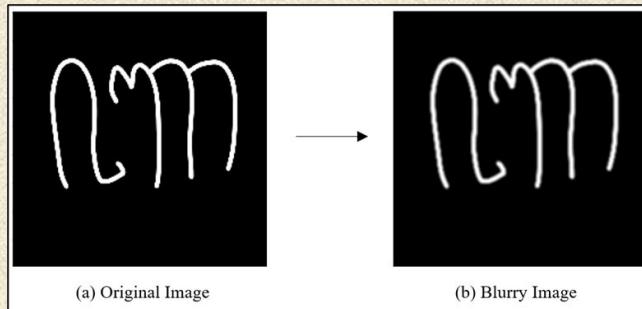
gambar awal

gambar hasil konvolusi (blur)

Tabel 3.2 Contoh Ilustrasi Konvolusi Matriks

Tabel 3.3 Hasil Matriks Setelah Dikonvolusi

...	255	250	249	250	255	242	196	111	38	1
	231	196	175	201	236	255	240	183	96	26
	202	131	82	118	183	235	255	231	170	83
	176	81	7	38	121	204	253	255	228	153
	163	70	0	5	59	148	228	255	253	210
	134	46	0	0	23	82	169	230	255	240
	102	18	0	0	0	25	96	184	241	255
	83	3	0	0	0	0	38	125	213	255
	80	0	0	0	0	0	11	68	156	230
	79	0	0	0	0	0	0	25	93	180



Ekstraksi Fitur (1. Ekstraksi SIFT)

Membuat Difference of Gaussian (DoG)

$$D(x, y, \sigma) = L(x, y, k^0\sigma) - L(x, y, k^1\sigma)$$

Tabel 3.4 Matriks Blur Skala 0

255	250	249	250	255	242	196	111	38	1	...
231	196	175	201	236	255	240	183	96	26	
202	131	82	118	183	235	255	231	170	83	
176	81	7	38	121	204	253	255	228	153	
163	70	0	5	59	148	228	255	253	210	
134	46	0	0	23	82	169	230	255	240	...
102	18	0	0	0	25	96	184	241	255	
83	3	0	0	0	0	38	125	213	255	
80	0	0	0	0	0	11	68	156	230	
79	0	0	0	0	0	0	25	93	180	...
										...

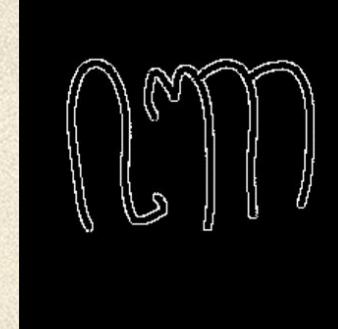
Tabel 3.5 Matriks Blur Skala 1

255	250	249	250	255	240	195	111	39	1	...
229	195	173	200	234	255	239	183	97	28	
200	135	85	119	183	234	255	229	169	84	
173	84	7	40	122	203	253	255	226	152	
161	72	0	6	60	146	226	255	253	209	
134	47	0	0	25	83	168	228	255	239	...
104	19	0	0	0	27	97	183	240	255	
86	3	0	0	0	0	40	125	212	255	
83	0	0	0	0	0	12	68	155	228	
82	0	0	0	0	0	0	26	94	180	...
										...

Tabel 3.6 Matriks Hasil Perhitungan DoG

0	0	0	0	0	2	1	0	255	0	...
2	1	2	1	2	0	1	0	255	254	
2	252	253	255	0	1	0	2	1	255	
3	253	0	254	255	1	0	0	2	1	
2	254	0	255	255	2	2	0	0	1	
0	255	0	0	254	255	1	2	0	1	...
254	255	0	0	0	254	255	1	1	0	
253	0	0	0	0	0	254	0	1	0	
253	0	0	0	0	0	0	255	0	1	2
253	0	0	0	0	0	0	255	255	0	...
										...

=



Ekstraksi Fitur (1. Ekstraksi SIFT)

Setiap DoG yang dihasilkan, *keypoint* akan diidentifikasi berdasarkan lokal maxima/minima pada setiap skala. Ini dilakukan dengan cara membandingkan setiap piksel yang ada di dalam citra DoG terhadap 8 piksel tetangganya di skala yang sama dan 9 piksel tetangga lainnya pada setiap skala tetangganya. Jika nilai piksel tersebut merupakan maxima/minima dari perbandingan seluruh piksel tetangganya (26 piksel), maka nilai piksel tersebut menjadi sebagai kandidat *keypoint* pada skala tersebut.

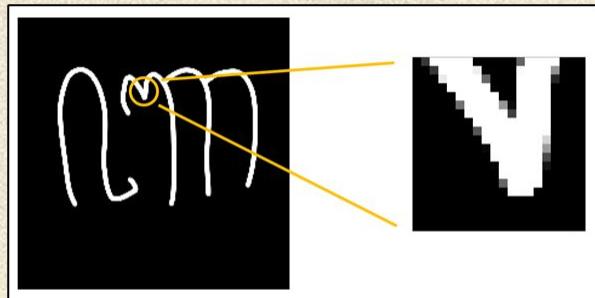


Ekstraksi Fitur (1. Ekstraksi SIFT)

Mencari Orientation Assignment pada setiap keypoint dengan menghitung nilai magnitudo dan gradien

$$m(x, y) = \sqrt{\left(L(x+1, y) - L(x-1, y)\right)^2 + \left(L(x, y+1) - L(x, y-1)\right)^2}$$

$$\theta(x, y) = \text{atan2}(L(x, y + 1) - L(x, y - 1), L(x + 1, y) - L(x - 1, y)) \dots$$



Tabel 3.7 Sampel Matriks Keypoint

Ekstraksi Fitur (1. Ekstraksi SIFT)

Hasil Perhitungan Magnitudo dan Arah Gradien

Tabel 3.8 Matriks Hasil Perhitungan Magnitudo

281	0	0	0	356	350	0	0	0	100	358	155	0	0	277	255
361	201	0	0	16	361	260	0	0	251	298	4	0	107	335	148
240	361	21	0	0	155	361	101	0	255	255	0	0	217	295	38
0	346	359	0	0	0	296	361	2	255	255	0	0	255	258	0
0	2	361	290	0	0	0	359	296	255	255	0	0	255	255	0
0	0	113	361	142	0	0	104	352	123	243	0	0	255	255	0
0	0	0	275	361	4	0	0	216	243	189	0	0	255	255	0
0	0	0	0	358	305	0	0	0	189	0	0	0	255	255	0
0	0	0	0	87	361	168	0	0	0	0	0	2	259	253	0
0	0	0	0	0	269	354	0	0	0	0	0	43	273	212	0
0	0	0	0	9	361	246	0	0	0	0	0	100	290	155	0
0	0	0	0	0	0	252	357	3	0	0	0	182	288	73	0
0	0	0	0	0	5	359	250	0	0	0	234	265	21	0	
0	0	0	0	0	0	255	297	0	0	0	255	256	0	0	
0	0	0	0	0	0	102	361	153	0	0	360	255	0	0	
0	0	0	0	0	0	0	275	361	255	360	361	1	0	0	

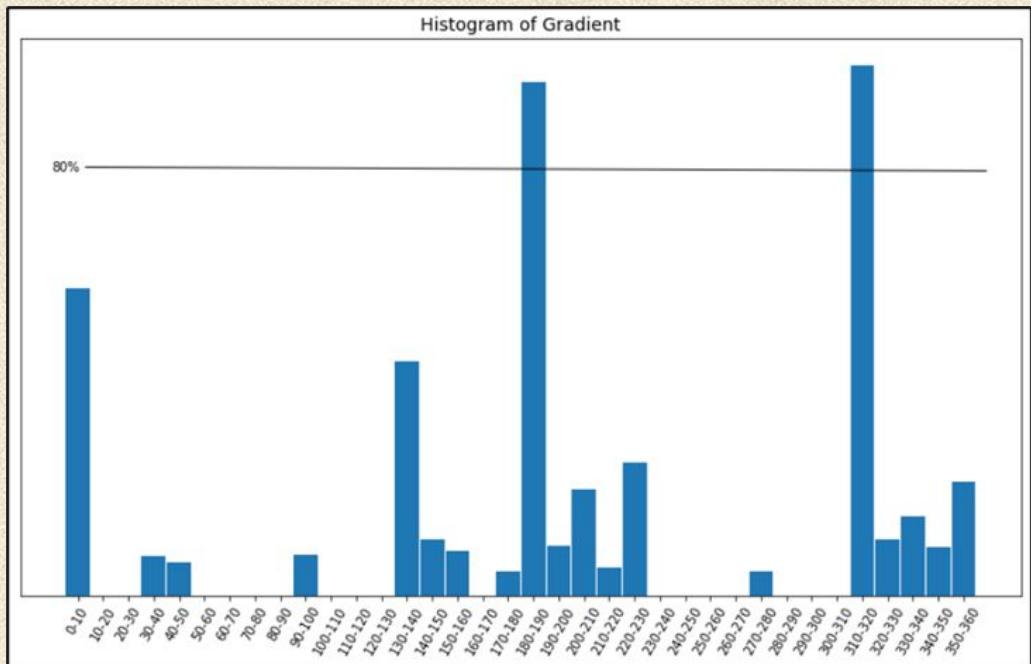
Tabel 3.9 Matriks Hasil Perhitungan Arah Gradien

315°	0°	0°	0°	136°	137°	0°	0°	0°	0°	45°	0°	0°	0°	203°	180°
315°	354°	0°	0°	180°	135°	157°	0°	0°	0°	31°	0°	0°	0°	180°	220°
347°	315°	0°	0°	0°	174°	135°	179°	0°	0°	1°	0°	0°	0°	180°	210°
0°	317°	315°	0°	0°	0°	149°	135°	180°	0°	0°	0°	0°	0°	180°	188°
0°	0°	315°	331°	0°	0°	0°	135°	149°	3°	0°	0°	0°	0°	180°	180°
0°	0°	359°	315°	358°	0°	0°	180°	134°	32°	0°	0°	0°	0°	180°	180°
0°	0°	0°	336°	315°	0°	0°	0°	151°	90°	0°	0°	0°	0°	180°	180°
0°	0°	0°	0°	315°	327°	0°	0°	0°	90°	0°	0°	0°	0°	180°	180°
0°	0°	0°	0°	0°	315°	0°	0°	0°	0°	0°	0°	0°	0°	180°	180°
0°	0°	0°	0°	0°	341°	316°	0°	0°	0°	0°	0°	0°	0°	180°	201°
0°	0°	0°	0°	0°	0°	315°	359°	0°	0°	0°	0°	0°	0°	180°	209°
0°	0°	0°	0°	0°	0°	358°	316°	0°	0°	0°	0°	0°	0°	180°	208°
0°	0°	0°	0°	0°	0°	0°	315°	0°	0°	0°	0°	0°	0°	180°	196°
0°	0°	0°	0°	0°	0°	0°	359°	329°	0°	0°	0°	0°	0°	180°	185°
0°	0°	0°	0°	0°	0°	0°	0°	315°	0°	0°	0°	0°	0°	225°	180°
0°	0°	0°	0°	0°	0°	0°	0°	338°	315°	270°	225°	225°	180°	0°	0°

Ekstraksi Fitur (1. Ekstraksi SIFT)

Membuat histogram dengan sumbu x sebagai arah gradien per 10 derajat dan sumbu y sebagai nilai magnitudo

Mengambil nilai yang tertinggi yaitu 310 derajat, sehingga gambar akan dirotasi sebesar 310 derajat



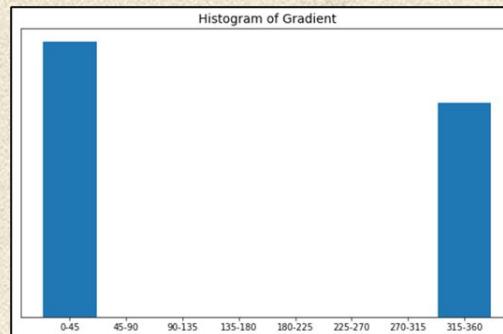
Ekstraksi Fitur (1. Ekstraksi SIFT)

Terakhir mencari nilai deskriptor atau fitur dari keypoint dengan cara menghitung kembali nilai derajat dan membuat 16 subblok.

Tabel 3.10 Matriks Hasil Perhitungan Arah Gradien Tiap Blok

0	344	321	319	322	0	157	144	133	180	0	0	0	0	0	0	0
0	0	323	323	359	0	0	131	136	130	90	0	0	0	0	0	0
0	0	0	339	319	279	0	0	124	131	131	117	0	0	0	0	0
0	0	0	319	313	311	293	0	0	117	131	131	126	0	0	0	0
0	90	90	0	297	311	312	306	0	0	92	132	140	137	0	0	0
89	84	101	99	90	87	340	326	342	0	180	179	137	145	156	0	0
86	102	108	88	95	98	103	35	0	0	0	153	158	140	143	173	0
90	90	90	90	90	94	95	87	68	0	0	0	90	143	139	131	0
0	270	270	0	270	0	90	90	0	0	0	0	0	0	131	135	0
267	269	277	272	270	270	270	271	270	0	270	0	0	0	0	0	150
253	278	290	287	276	264	289	285	280	282	270	271	0	270	0	200	0
270	270	270	272	270	277	275	269	289	295	286	295	295	270	235	223	0
0	0	0	0	270	0	270	270	270	271	270	296	289	290	254	227	0
0	0	0	0	0	0	0	0	0	0	270	270	270	271	270	266	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	270	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Membuat histogram setiap blok yang berisi jumlah gradien pada derajat tertentu per 45 derajat

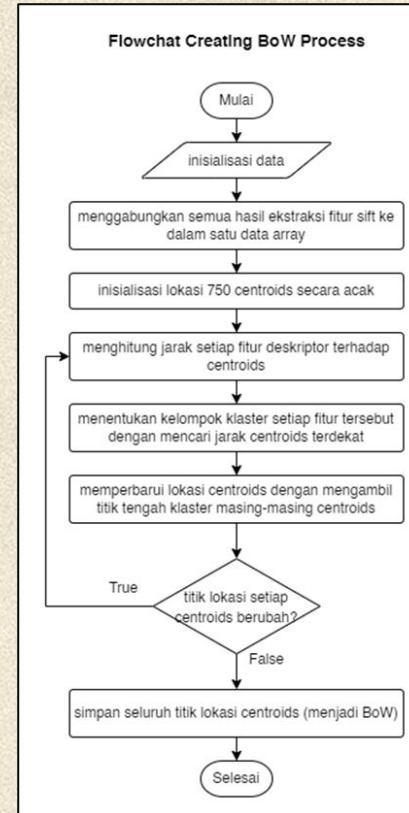


$$\text{descriptor}_0 = [9, 0, 0, 0, 0, 0, 0, 7]$$

Hasil fitur akhir akan menjadi $8 \times 16 = 128$ fitur pada setiap keypoint

Ekstraksi Fitur (2. Membuat BoW)

- Bag of Words atau BoW merupakan teknik menyimpan fitur penting dalam satu data/kamus menggunakan K-Means Clustering
- Jumlah isi data/kamus yang dihasilkan bergantung kepada jumlah centroids yang akan digunakan
- Jumlah centroids yang digunakan sebanyak 750 centroids
- Proses yang dilakukan selayaknya sama dengan K-Means pada umumnya
 - Inisialisasi centroids
 - Menghitung jarak centroids dengan data
 - Memperbarui titik centroids
- Setelah proses K-Means dilakukan semua centroids akan disimpan ke dalam BoW



Ekstraksi Fitur (2. Membuat BoW)

Menghitung jarak centroids pada K-Means

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

Contoh sampel 3 data

$$x_1 = \begin{bmatrix} 3 \\ 2 \\ 4 \\ \dots \\ 0 \end{bmatrix}, c_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \\ \dots \\ 1 \end{bmatrix}, c_2 = \begin{bmatrix} 0 \\ 0 \\ 2 \\ \dots \\ 4 \end{bmatrix}, c_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dots \\ 2 \end{bmatrix}$$

Jarak x_1 ke c_1

$$\min\left(\sum_{k=1}^k d_{11}\right) = \left\| \begin{bmatrix} 3 \\ 2 \\ 4 \\ \dots \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 2 \\ 3 \\ \dots \\ 1 \end{bmatrix} \right\|^2$$

$$\min\left(\sum_{k=1}^k d_{11}\right) = \left\| \begin{bmatrix} 2 \\ 0 \\ 1 \\ \dots \\ -1 \end{bmatrix} \right\|^2 = 398$$

Jarak x_1 ke c_2

$$\min\left(\sum_{k=1}^k d_{12}\right) = \left\| \begin{bmatrix} 3 \\ 2 \\ 2 \\ \dots \\ -4 \end{bmatrix} \right\|^2 = 436$$

Jarak x_1 ke c_3

$$\min\left(\sum_{k=1}^k d_{13}\right) = \left\| \begin{bmatrix} 3 \\ 2 \\ 4 \\ \dots \\ -2 \end{bmatrix} \right\|^2 = 580$$

x_1 masuk c_1

Menghitung jarak centroids pada K-Means

$$c_k = \frac{\sum_{i=1}^p x_i}{p}$$

Contoh sampel 3 data

$$x_1 = \begin{bmatrix} 3 \\ 2 \\ 4 \\ \dots \\ 0 \end{bmatrix}, x_2 = \begin{bmatrix} 2 \\ 2 \\ 3 \\ \dots \\ 2 \end{bmatrix}, x_3 = \begin{bmatrix} 1 \\ 2 \\ 5 \\ \dots \\ 1 \end{bmatrix}, c_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \\ \dots \\ 1 \end{bmatrix}$$

$$c_1 = \frac{\begin{bmatrix} 3 \\ 2 \\ 4 \\ \dots \\ 0 \end{bmatrix} + \begin{bmatrix} 2 \\ 2 \\ 3 \\ \dots \\ 2 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 5 \\ \dots \\ 1 \end{bmatrix}}{3}$$

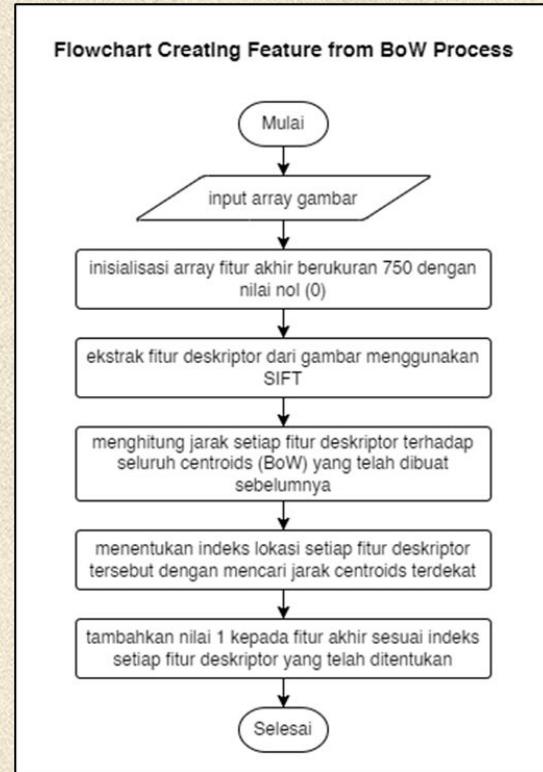
$$c_1 = \frac{\begin{bmatrix} 6 \\ 6 \\ 12 \\ \dots \\ 3 \end{bmatrix}}{3}$$

Hasil lokasi centroids terbaru

$$c_1 = \begin{bmatrix} 2 \\ 2 \\ 4 \\ \dots \\ 1 \end{bmatrix}$$

Ekstraksi Fitur (3. Membuat Fitur dari BoW)

- Pembuatan fitur melibatkan data BoW
- Setiap gambar memiliki jumlah keypoint yang berbeda-beda, setiap keypoint tersebut diklasifikasikan terhadap centroids yang terdekat
- Ekstrak SIFT Deskriptor pada gambar.
- Fitur akhir akan berukuran 750 dengan nilai awal nol (0)
- Indeks centroids yang terdekat menjadi penambahan nilai 1 pada fitur



Ekstraksi Fitur (3. Membuat Fitur dari BoW)

Inisialisasi fitur

$$F = [0,0,0, \dots, 0] \text{ dengan } \text{panjang}(F) = 750.$$

Contoh sampel 3 data

$$x_1 = \begin{bmatrix} 5 \\ 1 \\ 3 \\ \dots \\ 1 \end{bmatrix}, c_1 = \begin{bmatrix} 4 \\ 2 \\ 3 \\ \dots \\ 2 \end{bmatrix}, c_2 = \begin{bmatrix} 0 \\ 3 \\ 1 \\ \dots \\ 8 \end{bmatrix}, c_3 = \begin{bmatrix} 3 \\ 3 \\ 2 \\ \dots \\ 0 \end{bmatrix}$$

Jarak x_1 ke c_1

$$\min\left(\sum_{k=1}^k d_{11}\right) = \left\| \begin{bmatrix} 5 \\ 1 \\ 3 \\ \dots \\ 1 \end{bmatrix} - \begin{bmatrix} 4 \\ 2 \\ 3 \\ \dots \\ 2 \end{bmatrix} \right\|^2$$

$$\min\left(\sum_{k=1}^k d_{11}\right) = \left\| \begin{bmatrix} 1 \\ -1 \\ 0 \\ \dots \\ -1 \end{bmatrix} \right\|^2 = 244$$

c_1 = terdekat

Jarak x_1 ke c_2

$$\min\left(\sum_{k=1}^k d_{12}\right) = \left\| \begin{bmatrix} 5 \\ -2 \\ 2 \\ \dots \\ -7 \end{bmatrix} \right\|^2 = 691$$

Jarak x_1 ke c_3

$$\min\left(\sum_{k=1}^k d_{13}\right) = \left\| \begin{bmatrix} 2 \\ -2 \\ 1 \\ \dots \\ 1 \end{bmatrix} \right\|^2 = 403$$

Memperbarui fitur

$$F = [1,0,0, \dots, 0]$$

Fitur Akhir

$$F = [5,2,4, \dots, 0]$$

Preprocessing Fitur

- Preprocessing fitur berisi dua langkah yaitu feature scaling dan splitting data
- Feature scaling menggunakan metode normalisasi MinMaxScaling
- Splitting data dibagi menjadi data latih dan data uji dengan rasio 85% data latih dan 15% data uji

Perhitungan normalisasi:

Contoh sampel 9 data $x = [7,3,2,6,7,2,1,0,8]$

$$\max(x) = 8 \quad \min(x) = 0$$

$$x'_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}$$

$$x'_1 = \frac{7 - 0}{8} = 0.875$$

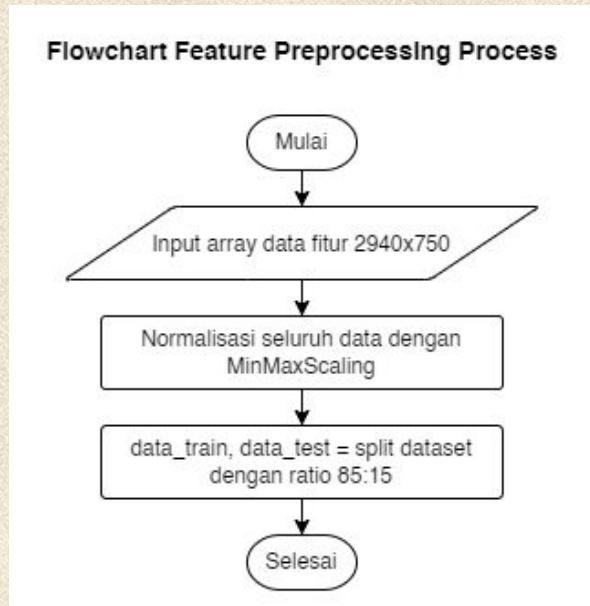
$$x'_2 = \frac{3 - 0}{8} = 0,375$$

...

$$x'_9 = \frac{8 - 0}{8} = 1$$

Hasil normalisasi:

$$x' = [0.875, 0.375, 0.25, 0.75, 0.875, 0.25, 0.125, 0, 1]$$



Support Vector Machine

Contoh pada kelas 'ba' mengambil 5 sampel ['ba', 'ba', 'nya', 'ma', 'ma'] dengan nilai label kelas menjadi [1, 1, -1, -1, -1]

Inisialisasi data:

$$\lambda=0.5, \gamma=0.001, \beta=1, \epsilon=0.0005, \alpha=0, \sigma=15.81$$

Membuat hessian matriks dari kernel RBF:

$$K(x_1, x_1) = \exp\left(-\frac{\|x_1, x_1\|^2}{2\sigma^2}\right)$$

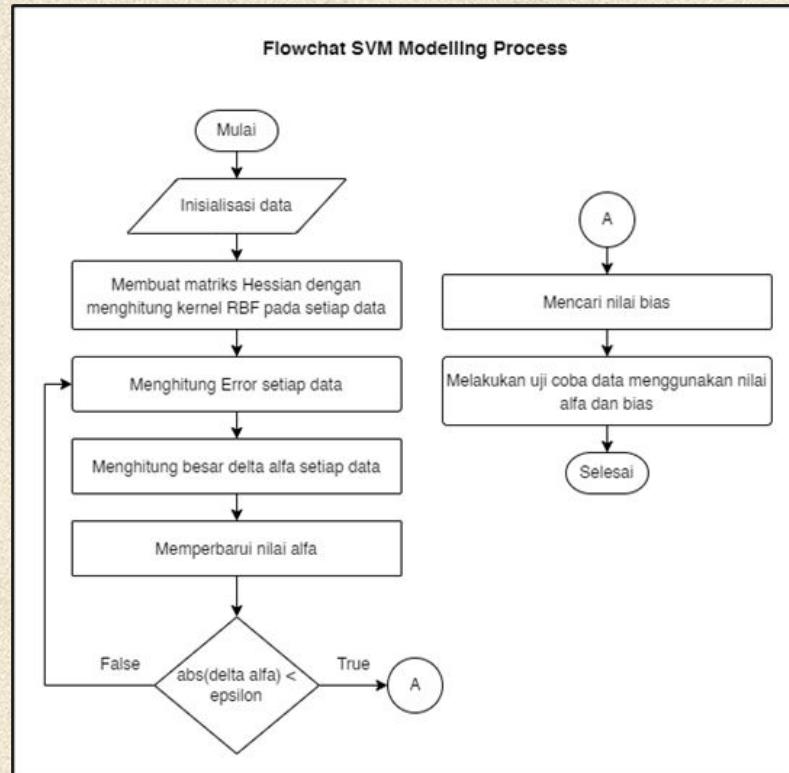
$$K(x_1, x_1) = \exp\left(-\frac{0}{2(15.81)^2}\right)$$

$$K(x_1, x_1) = \exp(0)$$

$$K(x_1, x_1) = 1$$

Tabel 3.11 Matriks Hasil Kernel RBF

D	1	2	3	4	5
1	1	0.33070886	0.02627371	0.0303213	0.03641921
2	0.33070886	1	0.02452981	0.03978491	0.04552518
3	0.02627371	0.02452981	1	0.00481004	0.00508454
4	0.0303213	0.03978491	0.00481004	1	0.0541797
5	0.03641921	0.04552518	0.00508454	0.0541797	1



Support Vector Machine

Menghitung nilai hessian matriks:

$$D_{11} = y_1 y_1 (K(x_1, x_1) + \lambda^2)$$

$$D_{11} = (1)(1)(1 + 0.5^2)$$

$$D_{11} = 1.25$$

Tabel 3.12 Hasil Perhitungan Matriks Hessian

D	1	2	3	4	5
1	1.25	0.58070886	-0.27627371	-0.2803213	-0.28641921
2	0.58070886	1.25	-0.27452981	-0.28978491	-0.29552518
3	-0.27627371	-0.27452981	1.25	0.25481004	0.25508454
4	-0.2803213	-0.28978491	0.25481004	1.25	0.3041797
5	-0.28641921	-0.29552518	0.25508454	0.3041797	1.25

Mencari nilai error:

$$E_1 = \sum_{j=1}^n \alpha_j D_{1j}$$

$$E_1 = (0 \times 1.25) + (0 \times 0.58070886) + (0 \times -0.27627371) + (0 \times -0.2803213) + (0 \times -0.28641921)$$

$$E_1 = 0$$

Tabel 3.13 Hasil Perhitungan Error

D	E
1	0
2	0
3	0
4	0
5	0

Menghitung delta alfa:

$$\delta\alpha_1 = \min(\max[\gamma(1 - E_1), \alpha_1], C - \alpha_1)$$

$$\delta\alpha_1 = \min(\max[0.001(1 - 0), 0], 1 - 0)$$

$$\delta\alpha_1 = \min(\max[0.001, 0], 1)$$

$$\delta\alpha_1 = \min(0.001, 1)$$

$$\delta\alpha_1 = 0.001$$

Tabel 3.14 Hasil Perhitungan Delta Alfa

D	$\delta\alpha$
1	0.001
2	0.001
3	0.001
4	0.001
5	0.001

Support Vector Machine

Memperbarui nilai alfa

$$\alpha_1 = \alpha_1 + \delta\alpha_1$$

$$\alpha_1 = 0 + 0.001$$

$$\alpha_1 = 0.001$$

Tabel 3.15 Hasil Perhitungan Pembaruan Alfa

D	α
1	0.001
2	0.001
3	0.001
4	0.001
5	0.001

Iterasi ini akan terus dilakukan hingga nilai $|\delta\alpha| < \varepsilon$. Tahapan iterasi ini diawali dengan menghitung nilai error hingga pembaruan nilai alfa

Tabel 3.17 Hasil Akhir Perhitungan Pembaruan Alfa

D	α
1	1
2	1
3	1
4	1
5	1

Mencari nilai bias dengan menghitung nilai kernel positif dan negatif terlebih dahulu

$$K(x_i, x^+) = \sum_{i=1}^m \alpha_i y_i K(x_i, x^+)$$

$$K(x_i, x^+) = (1 \times 1 \times 1.25) + (1 \times 1 \times 0.58070886) + (1 \times -1 \times -0.27627371) + (1 \times -1 \times -0.2803213) + (1 \times -1 \times -0.28641921)$$

$$K(x_i, x^+) = 2.67372308$$

$$K(x_i, x^-) = \sum_{i=1}^m \alpha_i y_i K(x_i, x^-)$$

$$K(x_i, x^-) = (1 \times 1 \times -0.27627371) + (1 \times 1 \times -0.27452981) + (1 \times -1 \times 1.25) + (1 \times -1 \times 0.25481004) + (1 \times -1 \times 0.25508454)$$

$$K(x_i, x^-) = -2.37909594$$

Menghitung nilai bias

$$b = -\frac{1}{2} \left[\sum_{i=1}^m \alpha_i y_i K(x_i, x^+) + \sum_{i=1}^m \alpha_i y_i K(x_i, x^-) \right]$$

$$b = -\frac{1}{2} [2.67372308 + (-2.37909594)]$$

$$b = -\frac{0.29462714}{2}$$

$$b = -0.14731357$$

Support Vector Machine

Melakukan uji coba dari hasil nilai alfa dan nilai bias.
Mengambil contoh sampel data kelas ‘ba’

Menghitung terlebih dahulu nilai kernel tiap data uji

$$K(x_{testing}, x_1) = \exp\left(-\frac{\|x_{testing}, x_1\|^2}{2\sigma^2}\right)$$

$$K(x_{testing}, x_1) = \exp\left(-\frac{785.396}{2(15.81)^2}\right) = 0.20782308$$

Tabel 3.18 Hasil Perhitungan Kernel Data Testing

$K(x_{testing}, x_1)$	$K(x_{testing}, x_2)$	$K(x_{testing}, x_3)$	$K(x_{testing}, x_4)$	$K(x_{testing}, x_5)$
0.20782308	0.23541007	0.02091285	0.03079944	0.03741394

Terakhir hitung nilai masing-masing kernel dengan alfa dan bias

$$f(x_{testing}) = \sum_{i=1}^m \alpha_i y_i K(x_{testing}, x_i) + b$$

$$\begin{aligned} f(x_{testing}) &= (1 \times 1 \times 0.20782308) + (1 \times 1 \times 0.23541007) \\ &\quad + (1 \times -1 \times 0.02091285) + (1 \times -1 \times 0.03079944) \\ &\quad + (1 \times -1 \times 0.03741394) + (-0.14731357) \end{aligned}$$

$$f(x_{testing}) = 0.20679335$$

Nilai > 0 menjelaskan bahwa data tersebut masuk kedalam kategori kelas 1 yaitu kelas ‘ba’, sedangkan nilai < 0 menjelaskan bahwa data tersebut termasuk dalam kategori kelas -1 yaitu kelas selain ‘ba’

$$f(x_{testing}) = 0.20679335 \text{ terprediksi sebagai kelas ‘ba’}$$

Optimasi Model

- Optimasi dilakukan dengan 4 parameter, yaitu
 - Size (ukuran gambar), K (jumlah centroids) secara manual
 - C (parameter SVM), G (gamma SVM) secara GridSearch

Tabel 3.19 Parameter Optimasi Model

Size	K	C	G
128x128	180	[1,3,6,10,15]	['auto', 0.0039, 0.0047, 0.0063, 0.0071]
	250		['auto', 0.0024, 0.0032, 0.0048, 0.0056]
	500		['auto', 0.001, 0.0015, 0.0025, 0.003]
	750		['auto', 0.00090, 0.00095, 0.0018, 0.0023]
	1000		['auto', 0.00090, 0.00095, 0.0015, 0.0020]
160x160	180	[1,3,6,10,15]	['auto', 0.0039, 0.0047, 0.0063, 0.0071]
	250		['auto', 0.0024, 0.0032, 0.0048, 0.0056]
	500		['auto', 0.001, 0.0015, 0.0025, 0.003]
	750		['auto', 0.00090, 0.00095, 0.0018, 0.0023]
	1000		['auto', 0.00090, 0.00095, 0.0015, 0.0020]
192x192	180	[1,3,6,10,15]	['auto', 0.0039, 0.0047, 0.0063, 0.0071]
	250		['auto', 0.0024, 0.0032, 0.0048, 0.0056]
	500		['auto', 0.001, 0.0015, 0.0025, 0.003]
	750		['auto', 0.00090, 0.00095, 0.0018, 0.0023]
	1000		['auto', 0.00090, 0.00095, 0.0015, 0.0020]
224x224	180	[1,3,6,10,15]	['auto', 0.0039, 0.0047, 0.0063, 0.0071]
	250		['auto', 0.0024, 0.0032, 0.0048, 0.0056]
	500		['auto', 0.001, 0.0015, 0.0025, 0.003]
	750		['auto', 0.00090, 0.00095, 0.0018, 0.0023]
	1000		['auto', 0.00090, 0.00095, 0.0015, 0.0020]

Evaluasi Model

Penelitian ini berfokus terhadap **akurasi** dalam evaluasi performa model.

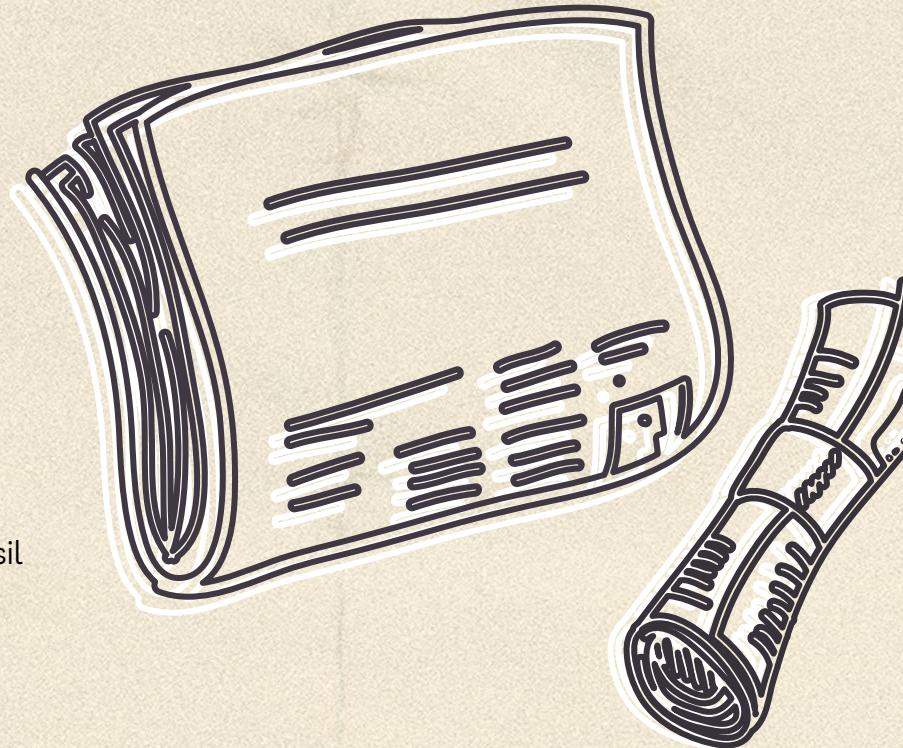
Berikut contoh perhitungan akurasi

$$\text{accuracy} = \frac{\text{total_true_predict}}{\text{total_data}}$$

$$\text{accuracy} = \frac{403}{441}$$

$$\text{accuracy} = 0.9138 = 91.38\%$$

Evaluasi model juga dilakukan dengan membandingkan hasil performa dari model SVM-SIFT dengan model SVM



Deployment Process

Deployment process merupakan tahap perancangan perangkat lunak dengan menggunakan prinsip metode dari Agile



Requirements

Analisis kebutuhan perangkat lunak yang akan dibangun



Design

Perancangan desain arsitektur (sistem aplikasi), desain proses (DFD) dan desain antarmuka (prototype aplikasi)



Development

Implementasi dari perancangan menggunakan bahasa python, HTML sederhana, framework Flask.

Deployment Process



Testing

Pengujian sistem dengan metode black box testing



Deployment

Integrasi aplikasi terhadap platform Github dan platform Heroku

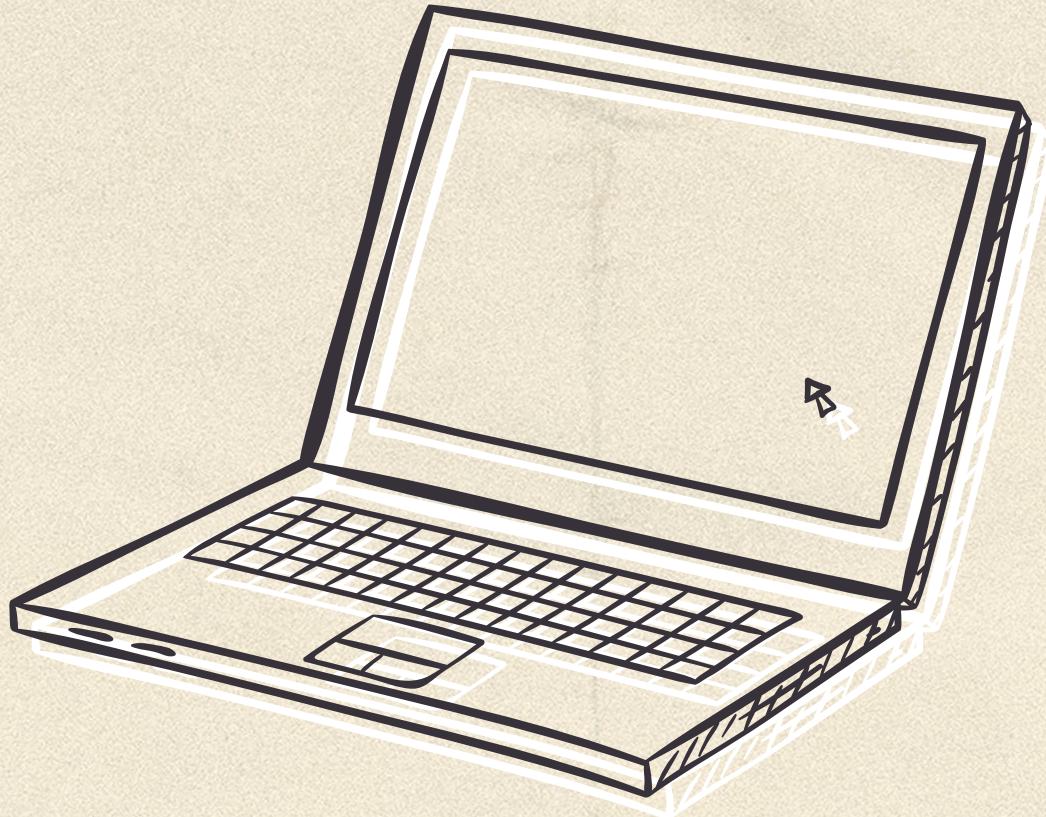


Review

Proses monitoring ataupun maintenance

04

Demo Program



Matur Nuwun

CREDITS:

This presentation template was created by Slidesgo, including icons by Flaticon, infographics & images by Freepik

