Optimasi SVM untuk Klasifikasi Warna: Investigasi Terhadap Pengaruh Fungsi Kernel dan Penyetelan Parameter

Pande Gede Dani Wismagatha[a1], I Wayan Santiyasa[a2]

[a]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam,
Universitas Udayana
Jalan Raya Kampus Udayana, Bukit Jimbaran, Kuta Selatan, Badung, Bali Indonesia
[1]pandedani@student.unud.ac.id
[2]santiyasa@unud.ac.id

Abstract

Color plays a crucial role in visual applications such as object recognition, image processing, computer vision, and computer graphics. Support Vector Machine (SVM) algorithms have gained attention for color classification due to their ability to handle complex data. SVM, a machine learning algorithm for classification and regression, aims to find optimal decision boundaries. In color classification using SVM, color data is represented by feature vectors, and SVM learns patterns to classify colors accurately. The SVM algorithm demonstrates a high accuracy rate, with an average accuracy of approximately 85% in color detection. This indicates the SVM's ability to effectively separate and classify colors with precision. SVM is proven to be effective in handling non-linear color data by utilizing kernel functions to transform the feature space into higher dimensions, enabling accurate classification of complex color data. The outstanding performance of the SVM algorithm in color detection presents vast potential applications in color recognition, image processing, computer vision, and computer graphics. SVM offers accurate and reliable solutions for object classification based on color characteristics in various contexts.

Keywords: *Color classification, SVM, Image processing, Machine Learning*

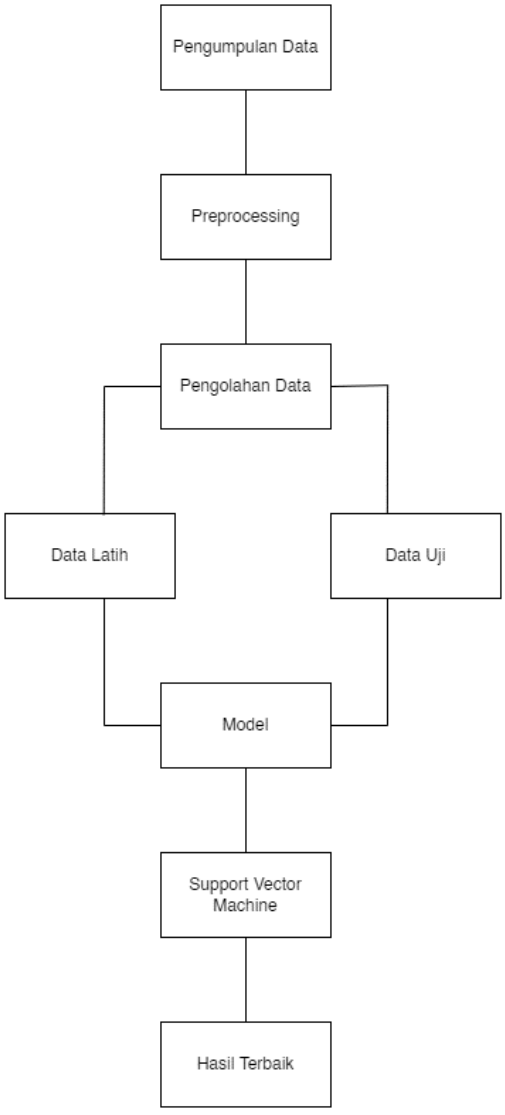
1. Pendahuluan

Warna adalah salah satu aspek penting dalam dunia visual dan memiliki peran yang signifikan dalam berbagai aplikasi, termasuk pengenalan objek, pengolahan citra, visi komputer, dan grafika komputer. Klasifikasi warna adalah suatu proses untuk mengelompokkan objek atau piksel berdasarkan karakteristik warnanya. Penggunaan algoritma Support Vector Machine (SVM) dalam klasifikasi warna menjadi subjek penelitian yang menarik karena kemampuannya dalam membedakan dan mengklasifikasikan data yang kompleks[1]. Support Vector Machine (SVM) merupakan algoritma machine learning yang dapat digunakan untuk klasifikasi ataupun regresi. SVM bertujuan untuk menemukan batas keputusan optimal antara dua atau lebih kelas dengan menggunakan konsep margin maksimal. SVM mampu mengklasifikasikan data dengan baik, terutama dalam kasus-kasus di mana data tidak terpisah secara linear di ruang fitur[2]. Dalam klasifikasi warna menggunakan SVM, data warna diwakili oleh vektor fitur yang terdiri dari komponen warna dalam model warna tertentu. Contohnya, dalam model warna RGB, vektor fitur terdiri dari intensitas merah (R), hijau (G), dan biru (B). SVM kemudian mempelajari pola-pola dalam vektor fitur tersebut untuk memisahkan dan mengklasifikasikan warna dengan akurasi yang tinggi. Metode klasifikasi warna menggunakan SVM memiliki beberapa keuntungan. Pertama, SVM mampu mengatasi data yang tidak linier, dengan menggunakan fungsi kernel untuk mengubah ruang fitur menjadi ruang dimensi yang lebih tinggi. Hal ini memungkinkan SVM untuk mengklasifikasikan data warna yang kompleks dengan presisi yang tinggi. Kedua, SVM mampu menangani data dengan dimensi tinggi, yang sering terjadi dalam representasi warna. SVM dapat memilih fitur-fitur yang relevan dan mengabaikan fitur-fitur yang tidak penting, sehingga meningkatkan efisiensi dan akurasi klasifikasi[3]. Penelitian ini bertujuan untuk mengklasifikasikan warna dengan pend Ekatan SVM Bagdasarian fitur-fitur warna dalam model

warna tertentu, seperti RGB. Metode SVM akan digunakan untuk mempelajari pola-pola dalam vektor fitur warna dan menghasilkan model klasifikasi yang dapat mengklasifikasikan warna dengan akurasi tinggi. Selain itu, penelitian ini juga akan melakukan analisis perbandingan antara beberapa metode terkait dalam klasifikasi warna menggunakan SVM untuk mengetahui kesamaan dan perbedaan dari metode-metode yang digunakan[4]. Diharapkan penelitian ini dapat memberikan kontribusi dalam pengembangan teknik klasifikasi warna dengan menggunakan algoritma SVM. Hasil penelitian ini diharapkan dapat meningkatkan akurasi dan efisiensi dalam pengenalan warna dalam berbagai aplikasi, seperti pengolahan citra, visi komputer, dan grafika komputer.

2. Metode Penelitian

Tahapan-tahapan dalam pelaksanaan penelitian dapat dilihat pada gambar berikut.



Gambar 1. Tahapan Pelaksanaan Penelitian

Berdasarkan gambar 1, tahap-tahap yang dilakukan dalam penelitian ini terdiri atas pengumpulan data, preprocessing, pengolahan data, pembagian data menjadi data latih (train) dan data uji (test), memasukkan data ke model kemudian mencari parameter terbaik untuk menghasilkan hasil yang terbaik.

a. Pengumpulan Data

Data diambil dari dataset pada laman kaggle yang dapat diakses menggunakan link berikut https://www.kaggle.com/datasets/ayanzadeh93/color-classification.
Dataset ini merupakan dataset 9 Jenis warna dengan total gambar berjumlah 107 data latih dan 96 data uji.

b. Pre-processing Data

Pada proses ini data diubah menjadi sebuah matriks dengan representasi angka. Kemudian, data yang telah diolah dapat digunakan untuk melatih dan menguji model klasifikasi.

c. Pengolahan Data

Pengolahan data dilakukan dengan memisahkan dataset yang telah melewati tahap sebelumnya/Pre-processing menjadi 2 (dua) yaitu data uji (train) dan data latih (test) lalu dijalankan pada algoritma Support Vector Machine (SVM). Berikut adalah cara kerja algoritma Support Vector Machine (SVM).

1. Support Vector Machine

Support Vector Machine (SVM) yaitu sistem pembelajaran yang menggunakan fungsi-fungsi linier dalam sebuah fitur yang berdimensi tinggi kemudian dilatih dengan menggunakan algoritma pembelajaran yang didasarkan pada teori optimasi. Akurasi yang dihasilkan oleh model pada algoritma ini sangatlah bergantung dengan penentuan parameter dan fungsi kernel yang digunakan. Algoritma SVM dapat dibedakan menjadi 2 (dua) yaitu SVM linear dan SVM non-linear. SVM linear digunakan untuk mengolah data yang dapat dipisahkan secara linear sedangkan SVM non-linear digunakan untuk data yang tidak bisa dibedakan secara linear sehingga menggunakan kernel untuk memisahkannya.

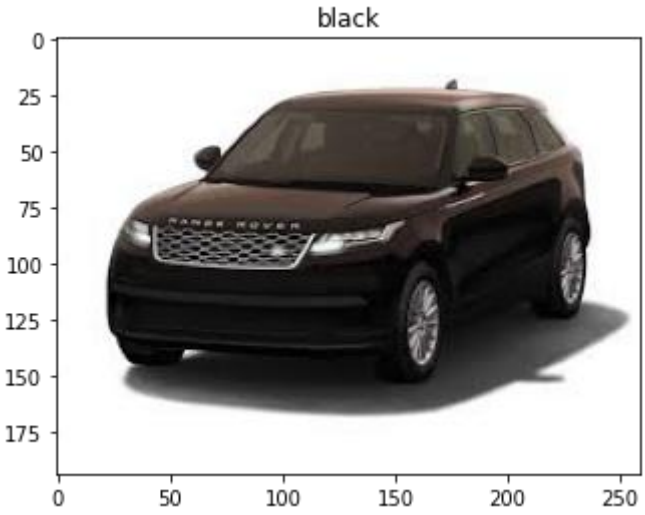
3. Hasil dan Diskusi

3.1. Persiapan Data

Dataset yang digunakan pada penelitian ini diperoleh dari salah satu dataset kaggle yang dibuat oleh Aydin Ayanzadeh dalam bentuk image (gambar). Pada dataset terdapat 107 gambar yang dibagi menjadi 9 klasifikasi gambar berbeda dan 96 tanpa label. Berikut adalah contoh gambar data uji. Peneliti menggunakan 107 gambar yang memilki label untuk memprediksi akurasi dari algoritma SVM.

3.2. Pre-Processing

Dataset yang digunakan dalam penelitian ini masih harus diubah menjadi sebuah nilai numerik agar mesin dapat mengerti akan masukan yang digunakan. Serta dataset harus diperiksa kembali guna mencari kesalahan klasifikasi pada data uji. Berikut ini contoh data yang telah diubah menjadi nilai numerikal yang dapat dimengerti mesin.



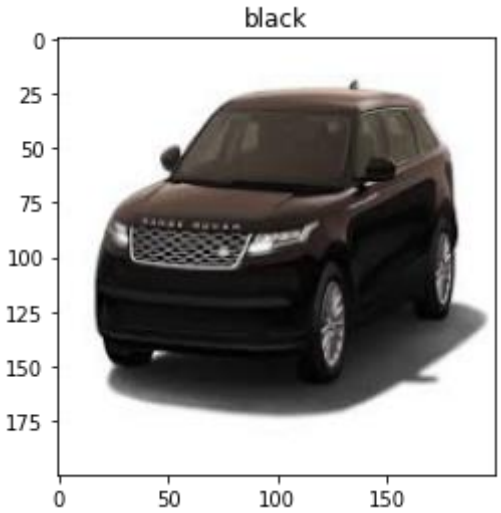
Gambar 2. Data Gambar

```
[[[255 255 255]
  [255 255 255]
  [255 255 255]
  ...
  [255 255 255]
  [255 255 255]
  [255 255 255]]

 [[255 255 255]
  [255 255 255]
  [255 255 255]
  ...
  [255 255 255]
  [255 255 255]
  [255 255 255]]

 [[255 255 255]
  [255 255 255]
  [255 255 255]
  ...
  [255 255 255]
  [255 255 255]
  [255 255 255]]

 ...

...
  ...
  [255 255 255]
  [255 255 255]
  [255 255 255]]]
```

Gambar 3. Data Numerik

Setelah melaksanakan konversi citra menjadi representasi data numerik, diperlukan tahapan penyesuaian ukuran pada seluruh data uji untuk memastikan konsistensi dimensi data. Resize (penyesuaian ukuran) pada gambar dalam machine learning merujuk pada proses mengubah dimensi fisik (ukuran) gambar menjadi ukuran yang ditentukan sebelum digunakan dalam model machine learning. Proses resize ini biasanya dilakukan untuk memastikan bahwa semua gambar memiliki dimensi yang seragam, sehingga mempermudah pemrosesan dan analisis dalam algoritma machine learning. Resize pada gambar melibatkan perubahan ukuran gambar secara proporsional, baik secara peningkatan (upscaling) maupun penurunan (downscaling) ukuran

gambar. Hal ini dapat dilakukan dengan menggunakan metode atau algoritma yang berbeda, seperti metode bilinear, metode nearest neighbor, atau metode bicubic. Tujuan utama dari resize gambar dalam konteks machine learning adalah untuk mencapai konsistensi ukuran gambar, sehingga memungkinkan model machine learning untuk memproses dan mempelajari fitur-fitur yang terdapat pada gambar dengan cara yang seragam. Resize juga dapat membantu mengurangi beban komputasi dan memori yang diperlukan dalam proses pelatihan model, terutama ketika menghadapi dataset gambar dengan variasi ukuran yang berbeda-beda[5]. Berikut adalah contoh gambar setelah proses resize.



Gambar 4. Gambar setelah proses resize.

3.3. Pre-Processing

Pada tahap selanjutnya data uji akan dibagi menjadi 2 tipe data yaitu data latih (train) dan data uji (test). Data yang digunakan sebagai data latih (train) sebanyak 80% dan data uji (test) sebanyak 20%. Namun sebelum itu, data warna dan label dipisah terlebih dahulu ke dalam sebuah variabel agar lebih mudah digunakan kedepannya. Berikut adalah potongan kode untuk memisahkan data warna dan label.

Tabel 1. Kode untuk memisahkan data warna dan label

```
X = []
y = []

for data_warna,label in training_list:
    X.append(data_warna)
    y.append(label)
```

Kemudian, data warna yang sebelumnya berupa matriks dengan dimensi 200x200 (setelah resize) perlu diubah menjadi sebuah matriks 1 dimensi. Berikut adalah potongan kode yang dapat digunakan untuk mengubah dimensi matriks.

Tabel 2. Kode untuk mengubah dimensi matriks menjadi 1 dimensi

```
X = np.array(X).reshape(len(X),-1)
```

Setelah data warna menjadi 1 dimensi, kedua data warna dan label dipisah menjadi 2 (dua) tipe data yang telah disebutkan sebelumnya yaitu 80% dan 20%. Berikut potongan kode untuk membagi dataset.

Tabel 3. Kode untuk memisahkan data menjasi data latih dan data uji

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42, train_size=0.8,
test_size=0.2)
```

3.4. Proses Klasifikasi

Kemudian proses dilanjutkan ke proses klasifikasi, proses klasifikasi memerlukan beberapa modules antara lain:

Tabel 4. Import modules untuk proses klasifikasi

```
from sklearn.svm import SVC
from tqdm import tqdm
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
```

Kemudian, proses dilanjutkan dengan memanggil algoritma pertama yaitu SVC (Support Vector Classifier). Peneliti membuat sebuah fungsi yang akan memanggil SVC dengan kernel, C (Parameter Regularisasi), gamma (koefisien kernel) dan decision function shape (bentuk fungsi keputusan) yang beragam dengan parameter awal kernel = linear, C = 0.1, gamma = scale dan decision function shape = ovo.

Tabel 5. Fungsi SVC (Support Vector Classifier)

```
def svm_train(kernel:str='linear', C:float=1.0, gamma:str='scale', literal:str='ovo'):
    svc = SVC(kernel=kernel, C=C)
    svc.fit(X_train, y_train)
    y2 = svc.predict(X_test)
    return accuracy_score(y_test, y2)
```

Kemudian peneliti membuat sebuah fungsi baru untuk memanggil kernel, C, gamma, dan decision function shape yang berbeda-beda guna membandingkan akurasi yang dihasilkan. Berikut adalah potongan kode yang dapat digunakan.

Tabel 6. Fungsi pemanggilan SVC

```
def get_svm(gamma:str='scale', literal:str='ovo'):
    df = pd.DataFrame(columns=['kernel', 'C', 'accuracy','gamma','decision_function_shape'])
    for x in tqdm([0.1, 1, 10, 100, 1000]):
        for y in ['linear', 'poly', 'rbf', 'sigmoid']:
            acc = svm_train(y, x, gamma, literal)
            df = df.append({'kernel': y, 'C': x, 'accuracy': acc, 'gamma': gamma,
'decision_function_shape':literal}, ignore_index=True)
    return df
```

Peneliti menjalankan fungsi dengan beberapa variasi yang ada. Berikut adalah hasil yang didapatkan.

C	gamma	decision_function_shape	kernel	accuracy
0.1	scale	ovo	linear	0.851852
			poly	0.592593
			rbf	0.333333
			sigmoid	0.185185
1.0	scale	ovo	linear	0.851852
			poly	0.703704
			rbf	0.777778
			sigmoid	0.074074
10.0	scale	ovo	linear	0.851852
			poly	0.703704
			rbf	0.814815
			sigmoid	0.000000
100.0	scale	ovo	linear	0.851852
			poly	0.740741
			rbf	0.814815
			sigmoid	0.037037
1000.0	scale	ovo	linear	0.851852
			poly	0.777778
			rbf	0.814815
			sigmoid	0.148148

C	gamma	decision_function_shape	kernel	accuracy
0.1	auto	ovo	linear	0.851852
			poly	0.592593
			rbf	0.333333
			sigmoid	0.185185
1.0	auto	ovo	linear	0.851852
			poly	0.703704
			rbf	0.777778
			sigmoid	0.074074
10.0	auto	ovo	linear	0.851852
			poly	0.703704
			rbf	0.814815
			sigmoid	0.000000
100.0	auto	ovo	linear	0.851852
			poly	0.740741
			rbf	0.814815
			sigmoid	0.037037
1000.0	auto	ovo	linear	0.851852
			poly	0.777778
			rbf	0.814815
			sigmoid	0.148148

Gambar 5. Perbandingan hasil akurasi SVC dengan mengubah parameter gamma dan decision function shape bernilai ovo (one-vs-one)

C	gamma	decision_function_shape	kernel	accuracy
0.1	scale	ovr	linear	0.851852
			poly	0.592593
			rbf	0.333333
			sigmoid	0.185185
1.0	scale	ovr	linear	0.851852
			poly	0.703704
			rbf	0.777778
			sigmoid	0.074074
10.0	scale	ovr	linear	0.851852
			poly	0.703704
			rbf	0.814815
			sigmoid	0.000000
100.0	scale	ovr	linear	0.851852
			poly	0.740741
			rbf	0.814815
			sigmoid	0.037037
1000.0	scale	ovr	linear	0.851852
			poly	0.777778
			rbf	0.814815
			sigmoid	0.148148

C	gamma	decision_function_shape	kernel	accuracy
0.1	auto	ovr	linear	0.851852
			poly	0.592593
			rbf	0.333333
			sigmoid	0.185185
1.0	auto	ovr	linear	0.851852
			poly	0.703704
			rbf	0.777778
			sigmoid	0.074074
10.0	auto	ovr	linear	0.851852
			poly	0.703704
			rbf	0.814815
			sigmoid	0.000000
100.0	auto	ovr	linear	0.851852
			poly	0.740741
			rbf	0.814815
			sigmoid	0.037037
1000.0	auto	ovr	linear	0.851852
			poly	0.777778
			rbf	0.814815
			sigmoid	0.148148

Gambar 6. Perbandingan hasil akurasi SVC dengan mengubah parameter gamma dan decision function shape bernilai ovr (one-vr-rest)

Dengan menjalankan program tersebut, peneliti dapat mengambil hasil terbaik yang ada pada tiap perubahan parameter. Berikut adalah tabel parameter yang mendapat akurasi paling tinggi.

	kernel	C	accuracy	gamma	decision_function_shape
0	linear	0.1	0.851852	scale	ovo
1	linear	1.0	0.851852	scale	ovo
2	linear	10.0	0.851852	scale	ovo
3	linear	100.0	0.851852	scale	ovo
4	linear	1000.0	0.851852	scale	ovo
5	linear	0.1	0.851852	auto	ovo
6	linear	1.0	0.851852	auto	ovo
7	linear	10.0	0.851852	auto	ovo
8	linear	100.0	0.851852	auto	ovo
9	linear	1000.0	0.851852	auto	ovo
10	linear	0.1	0.851852	scale	ovr
11	linear	1.0	0.851852	scale	ovr
12	linear	10.0	0.851852	scale	ovr
13	linear	100.0	0.851852	scale	ovr
14	linear	1000.0	0.851852	scale	ovr
15	linear	0.1	0.851852	auto	ovr
16	linear	1.0	0.851852	auto	ovr
17	linear	10.0	0.851852	auto	ovr
18	linear	100.0	0.851852	auto	ovr
19	linear	1000.0	0.851852	auto	ovr

Gambar 7. Akurasi terbaik saat menjalankan fungsi SVC

Semua data yang memiliki akurasi terbaik berasal dari kernel yang sama walaupun dengan menggunakan parameter yang berbeda. Maka dari itu, peneliti mencoba mencari laporan klasifikasi yang dijalankan oleh mesin dan mendapatkan hasil berkut.

	precision	recall	f1-score	support
Black	1.00	1.00	1.00	3
Blue	1.00	0.67	0.80	3
Brown	0.62	1.00	0.77	5
Green	1.00	1.00	1.00	2
Violet	1.00	0.67	0.80	3
White	0.80	1.00	0.89	4
orange	1.00	0.50	0.67	2
red	1.00	0.67	0.80	3
yellow	1.00	1.00	1.00	2
accuracy			0.85	27
macro avg	0.94	0.83	0.86	27
weighted avg	0.90	0.85	0.85	27

Gambar 8. Classification Report kernel linear

4. Kesimpulan

Algoritma SVM menunjukkan tingkat akurasi yang tinggi dalam mendeteksi warna. Berdasarkan pengujian yang dilakukan, algoritma SVM mencapai tingkat akurasi rata-rata sekitar 85%. Hal ini menunjukkan kemampuan SVM untuk memisahkan dan mengklasifikasikan warna dengan presisi yang baik. Algoritma deteksi warna berbasis Support Vector Machine (SVM) menunjukkan hasil yang menjanjikan dengan tingkat presisi, recall, dan F1-score yang tinggi. Presisi sebesar 90% mengindikasikan kemampuan algoritma dalam mengklasifikasikan warna dengan akurasi tinggi, dengan minimalisasi hasil positif palsu. Recall sebesar 85% menunjukkan kemampuan algoritma dalam mengidentifikasi dan menangkap instansi positif sebenarnya dari warna. F1-score sebesar 85% juga menunjukkan kinerja yang baik secara keseluruhan dalam deteksi warna. SVM terbukti efektif dalam menangani data warna yang tidak terpisah secara linier. Dengan menggunakan fungsi kernel, SVM dapat mentransformasikan ruang fitur menjadi ruang dimensi yang lebih tinggi, sehingga memungkinkan SVM untuk mengklasifikasikan data warna yang kompleks dengan akurasi yang tinggi. Dengan kinerja yang baik dalam mendeteksi warna, algoritma SVM memiliki potensi aplikasi yang luas dalam bidang pengenalan warna, pengolahan citra, visi komputer, dan grafika komputer. Dalam berbagai konteks, SVM dapat memberikan solusi yang akurat dan handal dalam mengklasifikasikan objek berdasarkan warnany

Daftar Pustaka

[1] M. F. Naufal, “ANALISIS PERBANDINGAN ALGORITMA SVM, KNN, DAN CNN UNTUK KLASIFIKASI CITRA CUACA”, doi: 10.25126/jtiik.202184553.

[2] W. Styorini and dan Wahyuni Khabzli, “Jurnal Politeknik Caltex Riau Analisis Perbandingan Machine Learning SVM Dan Adaboost Face Detection Dengan Metode Viola Jones,” 2018. [Online]. Available: http://jurnal.pcr.ac.id|

[3] M. Fahmi and I. Suhartana, “Perbandingan Algoritma Decision Tree Dan Support Vector Machine Dalam Prediksi Kualitas Udara,” 2022. [Online]. Available: https://data.jakarta.go.id/.

[4] R. L. Thiosdor, K. Gunadi, and L. P. Dewi, “Implementasi Program Presensi Mahasiswa dengan menggunakan Face Recognition.”

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks.” [Online]. Available: http://code.google.com/p/cuda-convnet/

Halaman ini sengaja dibiarkan kosong