ISSN: 2598-0256, E-ISSN: 2598-0238

Deteksi Objek Daun Tebu Dengan Menggunakan Metode Klasifikasi Pada *Machine Learning*

Sekar Dana Chiatra[1a,*], Hary Sabita[2b]

[a,b] Institut Informatika dan Bisnis Darmajaya
Jalan Z.A. Pagar Alam No. 93 Gedong Meneng Kecamatan Rajabasa,
Bandar Lampung – Indonesia 35142
Telp. (0721) 787214 Fax. (0721) 700261
[d] sekardanachiatra@gmail.com
[e] hary.sabita@darmajaya.ac.id

Abstract

The creation of a machine learning model that combines the Support Vector Machine (SVM) classification algorithm with the Histogram of Oriented Gradient (HOG) feature to detect sugarcane leaf objects. This research utilizes 227 processed images from 829 image data from Kaggle. The data is divided into two classes, namely other (class 0) and sugarcane (class 1). Analysis was performed using Python on the Google Colab platform. The model achieved 96% accuracy, precision 0.96, recall 1.00 and F1-score 0.98. This study provides a solution for automatic detection of tree count objects in Plantation land.

Keywords : *Machine Learning*; *Support Vector Machine*; *Histogram of Oriented Gradients*; *Python*; *Sugarcane*

Abstrak

Pembuatan sebuah model *machine learning* yang menggabungkan algoritma klasifikasi *Support Vector Machine* (SVM) dengan fitur *Histogram of Oriented Gradient* (HOG) untuk mendeteksi objek daun tebu. Penelitian ini memanfaatkan 227 citra hasil pemrosesan dari 829 data citra dari *Kaggle*. Data terbagi dalam dua kelas, yaitu lainnya (kelas 0) dan tebu (kelas 1). Analisis dilakukan menggunakan *Python* pada *platform Google Colab*. Model ini mencapai akurasi 96%, *precision* 0,96, *recall* 1,00 dan *F1-score* 0,98. Studi ini memberikan solusi deteksi objek jumlah pohon di lahan Perkebunan secara otomatis.

Kata kunci : *Machine Learning*; *Support Vector Machine*; *Histogram of Oriented Gradients*; *Python*; *Sugarcane*

1. PENDAHULUAN

Perkebunan merupakan sektor penting dalam perekonomian yang memanfaatkan ilmu pengetahuan, teknologi, modal dan manjemen unutk meningkatkan kesejahteraan masyarakat. Salah satu tanaman unggulan di Indonesia adalah tebu (*Saccharum officinarum*), bahan utama pembuatan gula pasir. Berdasarkan data Badan Pusat Statistik, Provinsi Jawa Timur berada di peringkat pertama dengan penghasilan tebu sebesar 1.116,1 ribu ton, sementara Provinsi Lampung berada di peringkat empat dengan penghasilan tebu mencapai 802,4 ribu ton.

Monitoring berkala sangat penting untuk meningkatkan produktivitas tebu, termasuk menghitung jumlah pohon yang sehat, sakit atau kering. Manual *counting* tidak efisien untuk lahan besar. Oleh karena itu, diperlukan pendekatan berbasis teknologi, seperti klasifikasi objek menggunakan *machine learning* untuk mempermudah proses tersebut.

Penelitian – penelitian sebelumnya menunjukkan efektivitas klasifikasi berbasis objek menggunakan algoritma *K-Nearest Neighbor* dan SVM. Dengan mengadopsi metode ini, penelitian mengusulkan model pendeteksian daun tebu yang lebih akurat menggunakan kombinasi *Support Vector Machine* (SVM) dan *Histogram of Oriented Gradient* (HOG).

2. KERANGKA TEORI

2.1. Tebu

Tanaman semipermanen dengan nilai sosial ekonomi yang signifikan, tebu (*Saccharum officinarum*) ditanam di seluruh wilayah tropis dan subtropis di dunia. Lebih dari 70% dari produksi gula dunia berasal dari tebu, yang merupakan

ISSN: 2598-0256, E-ISSN: 2598-0238

tanaman yang berguna. Selulosa dan *hemicellulose* yang ditemukan dalam sisa-sisa tebu dapat digunakan untuk membuat bioetanol dan biofuel transportasi cair lainnya. Biofuel yang terbuat dari tebu memiliki potensi untuk mengurangi polusi gas rumah kaca (GHG) yang disebabkan oleh pembakaran bahan bakar fosil. Di Brazil, misalnya, bioetanol gandum dapat mengurangi polusi gas rumah kaca dari bahan bakar fosil sebesar 85%. Dalam beberapa dekade terakhir, tebu telah menyebar dengan cepat, mengalihkan produk lain (seperti kedelai, beras, dan jagung), rumput, dan hutan. Di satu sisi, pertanian dan keamanan pangan dapat terancam jika tebu menggantikan produk makanan lainnya. Namun, dengan mengubah albedo permukaan dan evapotranspirasi, pertumbuhan tebu dapat memiliki dampak pada iklim lokal. Untuk tujuan mengelola produksi tebu dan memastikan produksi tebu yang berkelanjutan, perkiraan tepat waktu dan akurat dari area yang ditanam tebu diperlukan untuk melacak kondisi pertumbuhan dan perkiraan hasil(Zheng et al., 2022).

Pendeteksian pohon tebu telah dilakukan sebelumnya oleh (Zheng et al., 2022) menggunakan metode TWDTW (*Time-Weighted Dynamic Time Warping*) atau tergolong metode berbasis fenologi dengan citra *Landsat*-7/8, *Sentinel*-1 dan *Sentinel*-2. Metode yang digunakan dalam penelitian ini berhasil melakukan identifikasi area budidaya tebu secara efektif dan akurat walaupun dengan data sampel pelatihan yang terbatas. Peta tebu yang dihasilkan dapat diterapkan untuk pemantauan kondisi pertumbuhan dan hasil tebu, serta berkontribusi pada produksi tebu berkelanjutan. Selain melakukan pembatasan dengan menggunakan pencampuran spektral jenis vegetasi lain di area tanaman yang heterogen, keakuratan pemetaan juga dapat dilakukan dengan membandingkan secara langsung dengan metode lain, seperti *Machine Learning*.

2.2. Machine Learning

Machine Learning adalah pendekatan *Artificial Intelligence* yang berfokus pada pembuatan mesin (robot) yang dapat belajar tanpa diprogram secara eksplisit (detail). *Machine Learning* adalah teknik untuk melakukan inferensi terhadap data dengan pendekatan matematis (turunan dari matematika dan statistika). Maksud dari inferensi tersebut adalah lebih memfokuskan relasi antar atribut. Intinya, *Machine Learning* digunakan untuk membuat model (matematis) yang merefleksikan pola – pola data. *Machine Learning* dapat diibaratkan menjadi sebuah alat yang identik dengan rumus matematika yang cara menggunakannya bergantung pada domain permasalahan. Terdapat 2 metode pembelajaran yang banyak digunakan pada *Machine Learning*, yaitu pembelajaran terawasi (*supervised learning*) dan pembelajaran tidak terawasi (*unsupervised learning*)(Nurhidayat et al., 2021).

2.3. Supervised Learning

Supervised Learning adalah *Machine Learning* model yang mempelajari data dengan menggunakan data label atau target. Pada *supervised learning*, model akan membutuhkan data training berupa input dan target data. Model ini nantinya akan dilatih untuk dapat melakukan prediksi berdasarkan pola yang ditemukan dalam menjawab data target, yang kemudian dievaluasi dan dibandingkan hasilnya dengan prediksi oleh *test data*. Contoh metode dari model ini adalah klasifikasi dan regresi.

2.4. Classification (Klasifikasi)

Klasifikasi adalah suatu proses penemuan kumpulan model atau fungsi yang menjelaskan dan membedakan data kedalam kelas tertentu. Tujuannya adalah agar model tersebut dapat digunakan dalam penentuan kelas dari suatu objek yang belum diketahui kelasnya. Proses klasifikasi terjadi dalam 2 proses, yaitu :

1. Proses *learning / training*, membangun model menggunakan data *training*.
2. Proses *testing*, *testing* data menggunakan mdel yang telah didapat dari proses *training*.

Contoh algoritma yang sering digunakan untuk klasifikasi, yaitu *Linier Regression, Logistic Regression, Random Forest, XGBoost, K-NN* dan *SVM*.

2.5. Support Vector Machine (SVM)

Pertama kali diperkenalkan oleh Vapnik pada tahun 1992 sebagai rangkaian harmonis konsep – konsep unggulan dalam bidang *pattern recognition* (pengenalan pola). Usia SVM masih terbilang muda sebagai salah satu metode pengenalan pola. Walaupun begitu, evaluasi kemampuan dalam berbagai aplikasi menempatkannya sebagai karya terbaik dalam pengenalan pola. *Support Vector Machine* adalah metode *Machine Learning* yang bekerja atas prinsip *Structural Risk Minimization (SRM)* dengan tujuan menemukan *hyperplane* terbaik yang memisahkan 2 kelas (*class* -1 dan +1) pada *input space*(*Kanker Payudara Invasive*, 2020). Menemukan *hyperplane* terbaik dapat dilakukan dengan

ISSN: 2598-0256, E-ISSN: 2598-0238

menghitung nilai *margin hyperplane* tersebut, yaitu dengan pencarian titik maksimalnya. *Margin* merupakan jarak antara *hyperplane* dengan *pattern* terdekat (*support vector*) dari setiap kelas(Kurnia et al., 2018).

Konsep dasar dari SVM merupakan kombinasi harmonis dari teori – teori komputasi yang telah ada puluhan tahun sebelumnya, seperti *margin hyperplane*, *kernel* yang diperkenalkan oleh Aronszajn (tahun 1950) dan demikian juga dengan konsep – konsep pendukung yang lain. Berbeda dengan strantegi *neural network* yang berusaha mencari *hyperplane* pemisah antar kelas, SVM berusaha menemukan *hyperplane* terbaik pada *input space*(*Kanker Payudara Invasive*, 2020).

Berikut ini adalah beberapa karakteristik dari algoritma *Support Vector Machine* (SVM).

- Prinsip dasar dari SVM adalah *linier classifier*, selanjutnya akan dikembangkan agar dapat bekerja pada problem *non-linier* dengan memasukkan konsep *kernel trick* pada ruang kerja berdimensi tinggi.
- *Pattern recognition* dilakukan dengan mentransformasikan data pada *input space* ke ruang yang dimensi lebih tinggi dan ptimasi dilakukan di ruang vector yang baru tersebut. Pembeda SVM dari solusi pattern recognition pada umumnya adalah melakukan optimasi parameter pada ruang hasil transformasi yang berdimensi lebih rendah dari dimensi *input space*.
- Strategi SVM adalah *Structural Risk Minimization* (SRM).
- Prinsip kerjanya hanya mampu melakukan klasifikasi 2 kelas.

Support Vector Machine telah terbukti sukses diaplikasikan dalam menyelesaikan masalah klasifikasi dan estimasi fungsi setelah pengenalan yang dilakukan oleh Vapnik dalam konteks teori *statistical learning* dan *structural risk minimization*. Vapnik mengkonstruksikan SVM standar untuk memisahkan data – data pelatihan menjadi 2 kelas(*Kanker Payudara Invasive*, 2020).

Kelebihan dan kekurangan dari *Support Vector Machine* adalah sebagai berikut.
Kelebihan SVM :

- Generalisasi, sebagai kemampuan suatu metode untuk mengklasifikasikan suatu *pattern*, yang tidak termasuk data yang dipakai dalam fase pembelajaran metode tersebut. *Generalization error* dipengaruhi oleh 2 faktor, yaitu *training set* dan dimensi VC (*Vapnik-Chervonenkis*). Namun SVM dapat meminimalkan error pada kedua faktor tersebut.
- *Curse of Dimensionality*, sebagai masalah yang dihadapi suatu metode *pattern recognition* dalam melakukan estimasi parameter (contoh : jumlah hidden neuron pada neural network, stopping criteria pada proses pembelajaran) dikarenakan jumlah sampel data yang relatif sedikit dibandingkan dimensional ruang vektor data tersebut. Semakin tinggi dimensi dari ruang vektor informasi yang diolah, membawa konsekuensi dibutuhkannya jumlah data dalam proses pembelajaran.
- *Feasibility*, mengimplementasikan SVM relatif mudah, karena proses penentuan *support vector* dapat dirumuskan dalam *Quadratic Programming Problem*. Sehingga kita memiliki *library* untuk menyelesaikannya menggunakan SVM dengan mudah.

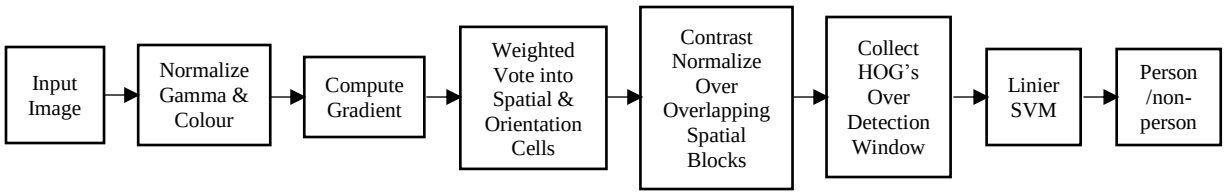
Kekurangan SVM :

- Sulit digunakan dalam permasalahan berskala besar.
- SVM secara teoritik dikembangkan untuk permasalahan klasifikasi dengan 2 kelas atau lebih. Namun, setiap strategi ini memiliki kelemahan, sehingga dapat dikatakan penelitian dan pengembangan SVM pada *multiclass problem* masih menjadi tema penelitian terbuka(*Kanker Payudara Invasive*, 2020).

2.6. Histogram of Oriented Gradients (HOG)

Histogram of oriented gradients atau biasa disingkat HOG adalah suatu fitur yang umum digunakan sebagai algoritma ekstraksi dalam *computer vision* dan pemrosesan gambar. Pertama kali diperkenalkan oleh Dalal dan Triggs pada tahun 2005 pada pendeteksian manusia. Berbagai masalah pendeteksian objek telah diselesaikan menggunakan algoritma ini, seperti deteksi pengenalan dan pelacakan pejalan kaki, deteksi pengenalan gerakan tangan, deteksi pengenalan wajah, deteksi mata dan pengenalan bagian tubuh untuk pelacakan, deteksi mobil, deteksi hewan maupun deteksi buah. HOG telah terbukti efektif untuk klasifikasi dan memberikan akurasi deteksi yang tinggi dengan perhitungan yang sederhana. Fitur HOG ini mampu memberikan gambaran informasi tentang bentuk dan tampilan objek lokal dengan distribusi gradien intensitas lokal atau arah tepi, hal tersebut karena fitur ini memiliki ketahanan yang terhadap perubahan iluminasi dan bayangan. *Histogram of Oriented Gradients* adalah suatu metode yang digunakan untuk melakukan deteksi keberadaan objek sebelum terdeteksi oleh kamera. Ada beberapa tahapan yang dilakukan pada metode ini(Prabowo & Nasahara, 2019).

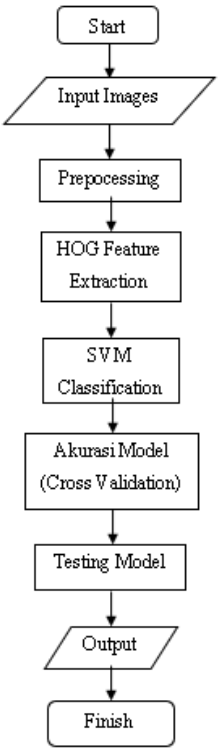
Seminar Nasional Hasil Penelitian dan Pengabdian Masyarakat 2025
Institut Informatika dan Bisnis Darmajaya, 8 Januari 2025
ISSN: 2598-0256, E-ISSN: 2598-0238



Gambar 1. Diagram Alir HOG

3. METODOLOGI

Penelitian ini dilakukan dengan 4 tahapan utama, yaitu (1) *Input Image* (Dataset), (2) HOG *Features Extraction*, (3) SVM *Classification* dan (4) *Cross-Validation*. Berikut adalah representasi dari penelitian ini, yang dapat dilihat pada Gambar 2.



Gambar 2. Alur Penelitian

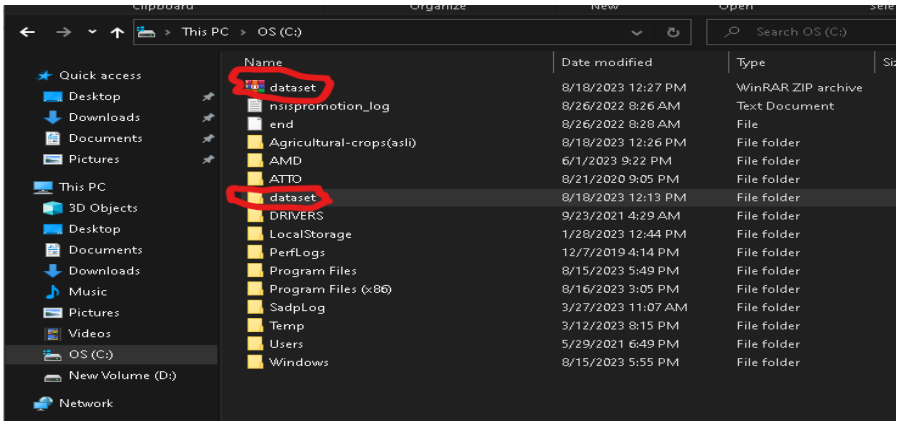
3.1. Input Image

Tahapan pertama pada penelitan ini adalah melakukan *input dataset*, yaitu berupa *image* atau citra. *Dataset* citra yang digunakan dalam penelitian ini, didapatkan dari Kaggle. Dataset berisi sebanyak 829 data yang terbagi dalam 30 kelas tanaman pertanian, yaitu sugarcane (tebu), almond, pisang, cardamom, ceri, cabe, clove, kelapa, kopi, kapas, ketimun, kacang, gram, jowar, jute, lemon, maize, mustard-oil, olive-trees, papaya, pearl-millet, nanas, rice, kedelai, sunflower, teh, tobacco-plant, tomato, vigna-radiati dan wheat. Dataset dari Kaggle ini tidak langsung dilakukan proses klasifikasi, namun dilakukan pengolahan kembali sehingga dataset ini hanya berisi 227 data yang terbagi dalam 2 kelas, yaitu *sugarcane* dan bukan. Seluruh sampel citra ini akan dilakukan proses *resize*, konversi RGB menjadi citra abu, atau yang biasa dikenal dengan *image processing*, dan kemudian citra diekstraksi dengan fitur HOG agar dapat dijadikan sebagai parameter masukan pada SVM pelatihan. Karena SVM linier hanya dapat memprediksi objek penelitian, berdasarkan hasil fitur HOG dan model SVM pada proses pelatihan(Prabowo & Nasahara, 2019)

ISSN: 2598-0256, E-ISSN: 2598-0238

3.1.1. Dataset

Dataset berasal dari Kaggle, dengan jumlah data 829 citra yang terbagi menjadi 30 kelas, kemudian diolah kembali menjadi 227 citra yang dibagi menjadi dua kelas, yaitu kelas lainnya (kelas 0) dan kelas tebu (kelas 1).



Gambar 3. *Dataset* Terbaru (*dataset.zip*)

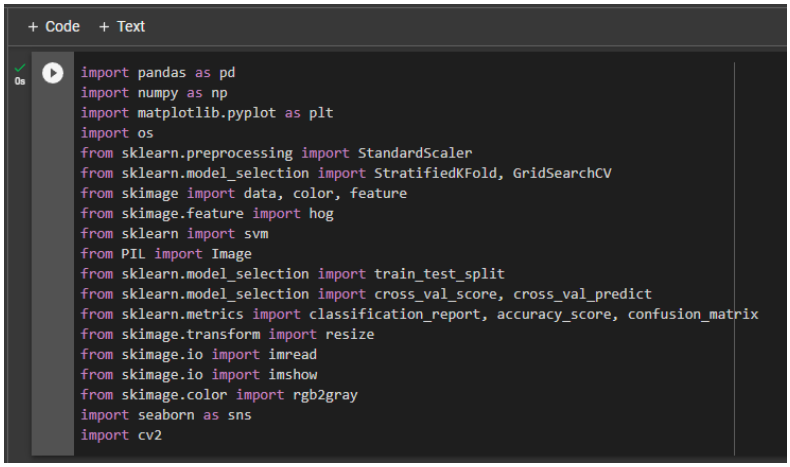
Coding *Import dataset.zip* :

```
#from google.colab import files

# Mengunggah dataset
uploaded = files.upload()

# Nama file yang diunggah
for filename in uploaded.keys():
    print(f"Uploaded file: {filename}")

import zipfile

# Nama file ZIP yang telah diunggah
zip_file_name = 'dataset.zip'

# Ekstrak ZIP
with zipfile.ZipFile(zip_file_name, 'r') as zip_ref:
    zip_ref.extractall('')  # Ganti 'temp_folder' dengan folder tujuan
ekstraksi
```

3.1.2. Preprocessing

Sebelum melakukan proses *image processing*, hal pertama yang perlu dilakukan untuk memulai meng-*coding* di *GoogleColab* adalah melakukan *import library* yang diperlukan pada model yang akan dibuat. Library yang digunakan untuk model pada penelitian ini, dapat dilihat pada Gambar 4.

Seminar Nasional Hasil Penelitian dan Pengabdian Masyarakat 2025
Institut Informatika dan Bisnis Darmajaya, 8 Januari 2025
ISSN: 2598-0256, E-ISSN: 2598-0238



Gambar 4. *Import Libraries*

Kemudian hal kedua yang perlu dilakukan adalah melihat isi *directory* untuk memastikan bahwa dataset telah diekstrak secara benar dengan menggunakan *coding* berikut ini.

```
base_dir = '/content/dataset/dataset'
print(os.listdir(base_dir))
```

Dan hal ketiga yang perlu dilakukan adalah mengetahui daftar kelas atau label gambar dari direktori dengan menggunakan *coding* berikut ini.

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Definisikan generator
data_generator = ImageDataGenerator(base_dir)  # Isi dengan parameter yang sesuai

# Muat data dari direktori
train_data = data_generator.flow_from_directory(
    base_dir,
    target_size=(64, 128),
    #batch_size=batch_size,
    class_mode='categorical'  # atau 'binary' tergantung pada kasus Anda
)

# Menampilkan daftar kelas atau label gambar
class_indices = train_data.class_indices
print("Daftar kelas:", class_indices)
```

Setelah selesai melakukan 3 hal diatas, sekarang gambar sudah dapat diproses (*Image Processing*). Pada codingan ini, gambar diproses dengan melakukan pengecekan (apakah data gambar termasuk RGB atau bukan?), perubahan ukuran (*resize*), dan perubahan gambar dari RGB menjadi *gray*.

```
Categories = {'others': 0, 'sugarcane': 1}


# Process the data
```

ISSN: 2598-0256, E-ISSN: 2598-0238


```
data = []
target = []

for category in os.listdir(base_dir):
    if category in Categories:
        class_idx = Categories[category]
        path = os.path.join(base_dir, category)
        for img in os.listdir(path):
            try:
                img_array = imread(os.path.join(path, img))

                # Periksa format gambar
                img_pil = Image.open(os.path.join(path, img))
                if img_array.shape[2] == 3 and img_pil.mode == 'RGB':
                    img_resized = resize(img_array, (64, 128, 3))
                    img_gray = rgb2gray(img_resized)
```

3.2. HOG Features Extraction

Setelah dilakukan image processing, tahap berikutnya adalah melakukan ekstraksi fitur dengan Histogram of Oriented Gradients (HOG). Langkah pertama pada HOG adalah melakukan konversi citra dari RGB ke gray, dan langkah ini telah dilakukan pada coding Gambar 6. Setelah melakukan konversi citra RGB ke gray, selanjutnya adalah penentuan jumlah bin, pixel per cell, dan cells per block (orientations = 9, pixels_per_cell = (8,8), cells_per_block = (2,2)).

```
fd = hog(img_gray, orientations=9, pixels_per_cell=(8, 8),
cells_per_block=(2, 2), visualize=False)
                    data.append(fd)
                    target.append(class_idx)
                    print(f'loaded label: {class_idx} successfully')
                    print(f'loaded category: {category} successfully')
                    print(f'loaded image: {img} successfully')
                else:
                    print(f'Skipped image (not RGB): {img}')
            except Exception as e:
                print(f'Error loading image: {img}. Error message: {e}')

# Buat DataFrame dari data dan target
df = pd.DataFrame({'data': data, 'target': target})

# Pisahkan fitur dan target
x = np.array(df['data'].tolist())  # Konversi list of arrays ke array numpy
y = np.array(df['target'])

# Cetak bentuk dari x dan y
print("Shape of x:", x.shape)
print("Shape of y:", y.shape)
```

Seminar Nasional Hasil Penelitian dan Pengabdian Masyarakat 2025
Institut Informatika dan Bisnis Darmajaya, 8 Januari 2025
ISSN: 2598-0256, E-ISSN: 2598-0238

3.3. SVM Classification

Tahap selanjutnya pada penelitian ini, setelah melakukan ekstrasi fitur HOG yaitu melakukan klasifikasi dengan algoritma *Support Vector Machine* (SVM). Pada tahapan inilah algoritma *Support Vector Machine* bekerja sebagai *classifier* untuk menentukan objek yang dideteksi adalah jenis tanaman pertanian apa menggunakan SVM terlatih. Namun sebelum melakukan klasifikasi, satu hal yang penting dilakukan yaitu, pembagian data (*Data Splitting*). *Data Splitting* adalah membagi dataset menjadi set pelatihan (*training set*) dan set pengujian (*testing set*). Tujuan dari proses ini adalah untuk memastikan bahwa model yang dibangun tidak hanya "menghafal" data yang telah dilihatnya, tetapi juga mampu melakukan generalisasi pada data yang belum pernah dilihat sebelumnya.

Coding *Data Splitting* :

```
(x_train, x_test, y_train, y_test) = train_test_split(x, y, test_size=0.2,
random_state=0)
```

Setelah dilakukan pembagian data, ada satu proses lagi yang perlu dilakukan, yaitu normalisasi fitur. Normalisasi fitur ini dilakukan menggunakan *StandardScaler* dari *Scikit-learn*.

```
# Normalisasi fitur
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

Selanjutnya, setelah melalui 2 proses tersebut, data gambar akhirnya melakukan klasifikasi *Support Vector Machine* (SVM) dengan kernel linear dan penyetelan parameter terbaik menggunakan *GridSearchCV*.

```
# Inisialisasi model SVM
regularized_svc = svm.SVC(kernel='linear')
```

```
# Penyetelan parameter menggunakan GridSearchCV
param_grid = {'C': [0.001, 0.01, 0.1, 1, 10]}
              #'gamma': [0.01, 0.1, 1, 10]
grid_search = GridSearchCV(regularized_svc , param_grid,
cv=StratifiedKFold(n_splits=10, shuffle=True, random_state=42))
grid_search.fit(x_train_scaled, y_train.ravel())
```

```
# Hasil penelusuran parameter
print("Best parameters found:", grid_search.best_params_)
```

```
# Evaluasi skor menggunakan model terbaik
best_svc = grid_search.best_estimator_
print("Dimensi best_svc:", best_svc)
print("Dimensi best_svc:", best_svc.n_features_in_)
```

3.4. Cross Validation

Cross Validation atau validasi silang merupakan suatu *model validation* yang digunakan untuk memprediksi atau memperkirakan keakuratan model saat dijalankan (*running*). *k-fold cross-validation* merupakan salah satu dari *cross validation* yang bekerja dengan cara memecah data menjadi beberapa bagian dengan ukuran yang sama dan secara berulang melakukan pelatihan dan pengujian dengan bagian yang berbeda(Azis et al., 2020).

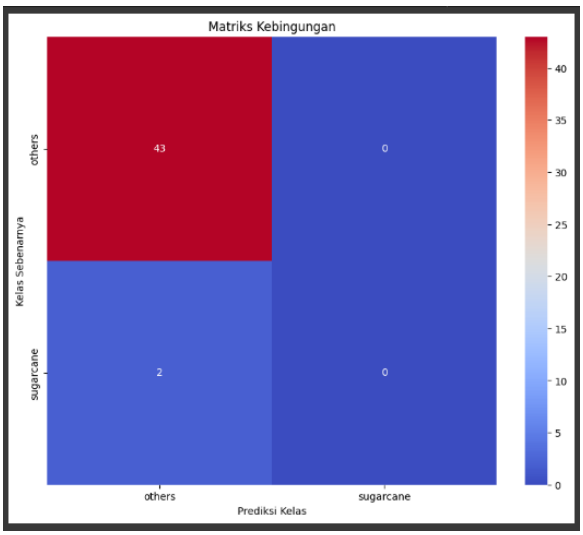
```
cv_train_score = cross_val_score(best_svc, x_train_scaled, y_train.ravel(),
cv=StratifiedKFold(n_splits=5, shuffle=True, random_state=1))
```

ISSN: 2598-0256, E-ISSN: 2598-0238

```
print("Train Score: %.2f" % cv_train_score.mean())

cv_test_score = cross_val_score(best_svc, x_test_scaled, y_test.ravel(),
cv=StratifiedKFold(n_splits=2, shuffle=True, random_state=1))
print("Test Score: %.2f" % cv_test_score.mean())
```

3.4.1. Confusion Matrix

Untuk dapat melakukan evaluasi performa suatu model klasifikasi pada *machine learning*, salah satu *tools* yang dapat digunakan yaitu *confusion matrix*. Dengan menggunakan *confusion matrix* ini, dapat dihasilkan nilai *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN). Berikut hasil *confusion matrix* dari model ini, dapat dilihat pada Gambar 5.



Gambar 5. Hasil *Confusion Matrix*

3.4.2. Testing Model

Untuk dapat membuktikan kebenaran dan ketepatan hasil validasi keakuratan serta evaluasi performa model, maka perlu untuk dilakukan percobaan pada model atau biasa disebut *testing model*.

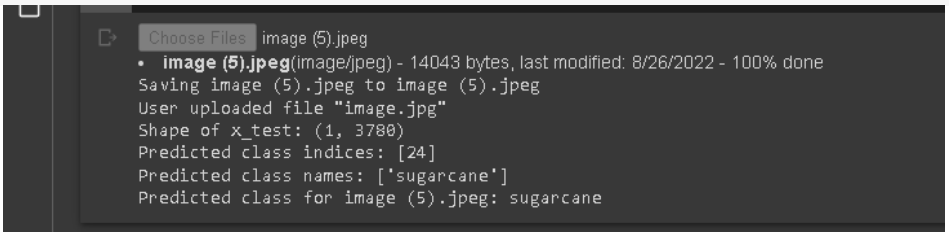
```
from google.colab import files
from skimage.io import imread
from skimage.transform import resize
import os
import io

Categories = {'others': 0, 'sugarcane': 1}

# Definisikan direktori data uji
uploaded = files.upload()

# Misalnya, jika Anda mengunggah file "image.jpg", maka:
test_data_filename = 'image.jpg'

print('User uploaded file "{}"'.format(test_data_filename))
```

Seminar Nasional Hasil Penelitian dan Pengabdian Masyarakat 2025
Institut Informatika dan Bisnis Darmajaya, 8 Januari 2025
ISSN: 2598-0256, E-ISSN: 2598-0238

```
# Inisialisasi variabel untuk data uji
x_test = []

# Loop melalui data gambar yang diunggah
for img_data in uploaded.values():
    try:
        img_array = imread(io.BytesIO(img_data))

        if img_array.shape[2] == 3:  # Gambar RGB
            img_resized = resize(img_array, (64, 128, 3))
            img_gray = rgb2gray(img_resized)
            fd = hog(img_gray, orientations=9, pixels_per_cell=(8, 8),
cells_per_block=(2, 2), visualize=False)
            if len(fd.shape) > 1:
                fd = fd.flatten()
                print("Bentuk dari fd:", fd.shape)

            x_test.append(fd)
        else:
            print("Skipped image (not RGB)")
    except Exception as e:
        print(f'Error loading image. Error message: {e}')

# Konversi ke array numpy
x_test = np.array(x_test)

# Print the shape of x_test for debugging purposes
print("Shape of x_test:", x_test.shape)

if x_test.shape[0] > 0:  # Periksa apakah ada data untuk diuji
    # Normalisasi data uji
    x_test_scaled = scaler.transform(x_test)  # Menggunakan scaler yang
telah didefinisikan sebelumnya

    # Memprediksi kelas data uji
    predicted_class_indices = best_svc.predict(x_test_scaled)

    print("Predicted class indices:", predicted_class_indices)  # Cetak
prediksi indeks kelas

    # Konversi indeks yang diprediksi menjadi nama kelas
    predicted_class_names = [key for key, value in class_indices.items() if
value == predicted_class_indices[0]]

    print("Predicted class names:", predicted_class_names)  # Cetak
prediksi nama kelas
```

ISSN: 2598-0256, E-ISSN: 2598-0238

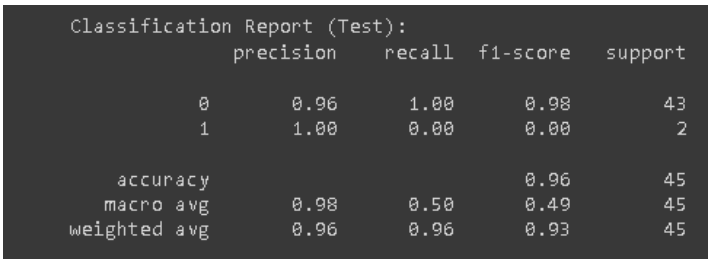
```
    # Cetak hasil prediksi untuk setiap gambar yang diunggah
    for img_file, predicted_class_name in zip(uploaded.keys(),
predicted_class_names):
        print(f"Predicted class for {img_file}: {predicted_class_name}")
else:
    print("No valid images to predict.")
```



Gambar 6. *Output Testing Model*

4. HASIL DAN PEMBAHASAN

Berdasarkan hasil *confusion* matrixpada Gambar 5, artinya dari 45 data uji, terdapat 43 data gambar positif bukan daun tebu (*non-sugarcane/others)*, dan model memprediksi ada 2 data gambar negatif bukan daun tebu (sugarcane). Dari proses *confusion matrix*, nilai – nilai yang telah dihasilkan ini dapat digunakan untuk menghitung *performance matrix* (*accuracy*, *recall*, *precision* dan *F1-score*). Hasil *performance matrix* ini, dapat dilihat pada Gambar 6 dan Tabel 1.



Classification Report (Test):

	precision	recall	f1-score	support
0	0.96	1.00	0.98	43
1	1.00	0.00	0.00	2
accuracy			0.96	45
macro avg	0.98	0.50	0.49	45
weighted avg	0.96	0.96	0.93	45

Gambar 7. Hasil *Performance Matrix*

Tabel 1. Hasil Evaluasi Performa Model

Confusion Matrix		*Performance Matrix*	
TP	43	*Accuracy*	0,96
TN	0	*Precision*	0,96
FP	0	*Recall*	1,00
FN	2	*F1-Score*	0,98

Berdasarkan hasil pada Tabel 1, dapat terlihat bahwa model ini memiliki performa yang sangat baik, dengan menggunakan algoritma *Support Vector Machine* (SVM) dan *Histogram of Oriented Gradient* (HOG) sebagai fitur ekstraksi dapat melakukan klasifikasi pada 45 data uji dengan nilai akurasi yang tinggi, yaitu sebesar 0,96 (96%), *precision* sebesar 0,96, *recall* sebesar *1,00 dan f1-score* sebesar *0,98.* Dengan begitu, artinya model ini dapat dijadikan acuan sebagai model klasifikasi yang akurat dan cepat.

ISSN: 2598-0256, E-ISSN: 2598-0238

5. KESIMPULAN

Berikut kesimpulan yang dapat ditarik dari penelitian ini.

1. Awalnya terdapat 829 data (terbagi dalam 30 kelas) yang diambil dari Kaggle, kemudian diolah kembali menjadi berjumlah 227 data (yang terbagi menjadi 2 kelas, yaitu sugarcane dan bukan).

2. Dari 227 data citra, hanya sebanyak 225 data citra yang dapat melanjutkan ke tahap berikutnya, karena termasuk citra RGB (*Red, Green and Blue*), mulai dari proses *resize* menjadi ukuran 64 x 128, kemudian konversi citra RGB ke *gray*, baru selanjutnya ekstraksi fitur HOG. Ekstraksi fitur HOG dilakukan pada orientasi bin = 9, pixels per cell = 8,8 dan cells per block = 2,2.

3. Setelah ekstraksi fitur, tahap selanjutnya adalah *splitting data*. Dari proses ini dihasilkan data uji sebanyak 45 data citra dan kemudian langsung dilakukan normalisasi fitur menggunakan StandardScaler.

4. Setelah normalisasi fitur, tahap berikutnya adalah klasifikasi SVM, dilakukan dengan memberikan parameter terbaik menggunakan GridSearchCV (kernel = liniear dan C = 0,001).

5. Model yang dihasilkan dapat dijadikan acuan sebagai model klasifikasi yang akurat karena telah melalui proses pengujian model dengan hasil akurasi model sebesar 96% (lebih detail dapat dilihat pada Tabel 1)

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Bapak Hary Sabita, S.T., M.T.I. atas bimbingan, dukungan, dan kontribusi yang diberikan selama proses penelitian ini. Penulis juga berterima kasih kepada Institut Informatika dan Bisnis Darmajaya atas fasilitas yang diberikan untuk mendukung penelitian ini.

Selain itu, penghargaan yang tulus juga disampaikan kepada keluarga dan teman-teman yang telah memberikan dorongan moral selama proses penyusunan jurnal ini.

DAFTAR PUSTAKA

Azis, H., Purnawansyah, P., Fattah, F., & Putri, I. P. (2020). Performa Klasifikasi K-NN dan Cross Validation Pada Data Pasien Pengidap Penyakit Jantung. *ILKOM Jurnal Ilmiah*, *12*(2), 81–86. https://doi.org/10.33096/ilkom.v12i2.507.81-86

Geomatika, J. T., Teknik, F., & Sepuluh, I. T. (2016). *Menggunakan Metode Klasifikasi Berbasis Obyek*. *October*.

Ichwan, M., Dewi, I. A., & S, Z. M. (2019). Klasifikasi Support Vector Machine (SVM) Untuk Menentukan TingkatKemanisan Mangga Berdasarkan Fitur Warna. *MIND Journal*, *3*(2), 16–23. https://doi.org/10.26760/mindjournal.v3i2.16-23

Kanker Payudara Invasive. (2020). [Binus]. http://library.binus.ac.id/eColls/eThesisdoc/Bab2HTML/2012100292IFBab2/body.html

Kurnia, A. I., Furqon, M. T., & Rahayudi, B. (2018). Klasifikasi Kualitas Susu Sapi Menggunakan Algoritme Support Vector Machine (SVM) (Studi Kasus: Perbandingan Fungsi Kernel Linier dan RBF Gaussian). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *2*(11), 4453–4461. http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/2935

Nurhidayat, A., Asmunin, A., & Suyatno, D. F. (2021). Prediksi Kinerja Akademik Mahasiswa Menggunakan Machine Learning dengan Sequential Minimal Optimization untuk Pengelola Program Studi. *Journal of Information Engineering and Educational Technology*, *5*(2), 84–91. https://doi.org/10.26740/jieet.v5n2.p84-91

Prabowo, Y., & Nasahara, K. N. (2019). Detecting and Counting Coconut Trees in Pleiades Satellite Imagery Using Histogram of Oriented Gradients and Support Vector Machine. *International Journal of Remote Sensing and Earth Sciences (IJReSES)*, *16*(1), 87. https://doi.org/10.30536/j.ijreses.2019.v16.a3089

Zheng, Y., Li, Z., Pan, B., Lin, S., Dong, J., Li, X., & Yuan, W. (2022). Development of a Phenology-Based Method for Identifying Sugarcane Plantation Areas in China Using High-Resolution Satellite Datasets. *Remote Sensing*, *14*(5). https://doi.org/10.3390/rs14051274