

# Integrasi *Convolutional Autoencoder* dengan *Support Vector Machine* untuk Klasifikasi Varietas Almond

<http://dx.doi.org/10.28932/jutisi.v11i1.9738>

Riwayat Artikel

Received: 13 Agustus 2024 | Final Revision: 14 April 2025 | Accepted: 14 April 2025

Creative Commons License 4.0 (CC BY – NC)



Rizal Fadlullah<sup>#1</sup>, Sri Winarno<sup>✉#2</sup>, Muhammad Naufal<sup>#3</sup>

<sup>#</sup>Teknik Informatika, Universitas Dian Nuswantoro  
Jl. Imam Bonjol No.207, Kota Semarang, 50131, Indonesia

<sup>1</sup>rizal.fadlullah@gmail.com

<sup>2</sup>sri.winarno@dsn.dinus.ac.id

<sup>3</sup>m.naufal@dsn.dinus.ac.id

✉Corresponding author: sri.winarno@dsn.dinus.ac.id

**Abstrak** — Penelitian ini bertujuan mengoptimalkan klasifikasi varietas almond dengan mengintegrasikan *Convolutional Autoencoder* (CAE) sebagai metode ekstraksi fitur dan *Support Vector Machine* (SVM) untuk klasifikasi. Tahapan penelitian mencakup pengumpulan data dari dataset yang tersedia, preprocessing, dan pemisahan data untuk pelatihan serta pengujian. Fitur dari gambar almond diekstraksi menggunakan *Convolutional Autoencoder* (CAE), yang kemudian digunakan dalam model SVM untuk klasifikasi. Selain itu pencarian hiperparameter optimal untuk SVM dilakukan dengan *Random Cross-Validation*, guna meningkatkan performa model. Evaluasi model menunjukkan akurasi klasifikasi sebesar 97% pada data testing, meningkat signifikan dibandingkan dengan akurasi 48% pada SVM konvensional, dengan metode evaluasi menggunakan *train-test split* di mana dataset dibagi menjadi 85% data pelatihan dan 15% data pengujian. Pendekatan CAE-SVM menawarkan representasi fitur yang lebih kompak dan informatif, meningkatkan pengenalan varietas almond secara efektif. Penelitian ini menyoroti potensi penggabungan keunggulan CAE dan SVM dalam meningkatkan analisis citra tanaman, serta mendorong kemajuan lebih lanjut dalam aplikasi machine learning dalam bidang agrikultur.

**Kata kunci**— *Convolutional Autoencoder; Klasifikasi; Machine Learning; Support Vector Machine; Varietas Almond.*

## *Integration of Convolutional Autoencoder with Support Vector Machine for Almond Varieties Classification*

**Abstract** — This research aims to optimize almond variety classification by integrating *Convolutional Autoencoder* (CAE) as a feature extraction method and *Support Vector Machine* (SVM) for classification. The research process includes data collection from available datasets, preprocessing, and splitting data for training and testing. Features from almond images are extracted using CAE, which are then used in the SVM model for classification. Additionally, the search for optimal hyperparameters for SVM is conducted using random cross-validation to enhance model performance. Model evaluation shows a classification accuracy of 97% on the test data, a significant increase compared to the 48% accuracy of conventional SVM, with an evaluation method using *train-test split*, where the

*dataset is divided into 85% training data and 15% testing data. The CAE-SVM approach offers more compact and informative feature representations, effectively improving almond variety recognition. This study highlights the potential of combining CAE and SVM advantages to enhance plant image analysis and encourages further advancements in machine learning applications in agriculture.*

**Keywords—** Almond Varieties; Classification; Convolutional Autoencoder; Machine Learning; Support Vector Machine.

## I. PENDAHULUAN

Almond (*Prunus dulcis*) memiliki sejarah panjang dengan asal usul yang diperkirakan sekitar 5,88 juta tahun yang lalu [1]. Almond adalah salah satu jenis kacang dengan kandungan protein tertinggi [2]. Selain itu Almond juga kaya akan nutrisi penting seperti kalsium, zat besi, vitamin E, magnesium, fosfor, dan seng [3]. Almond mendukung kesehatan jantung [4] dengan lemak yang baik dan membantu mengatur gula darah, sehingga bermanfaat bagi pengelolaan diabetes. Selain manfaat nutrisinya, almond memiliki berbagai aplikasi dalam berbagai industri. Dalam pengobatan tradisional, almond digunakan sebagai solusi alami untuk berbagai masalah kesehatan [5]. Almond juga digunakan sebagai pelembab dalam produk kosmetik dan perawatan kulit, sebagai bahan pengisi dan pemanis dalam industri farmasi, serta sebagai bahan aktif dalam produk pembersih alami. Karena penggunaannya yang beragam ini, almond menjadi produk yang sangat diminati di pasar ekspor internasional [6].

Almond memiliki lebih dari satu varietas dengan yang ditanam di seluruh dunia. Levent et al. [7] mengidentifikasi sepuluh genotipe almond yang berbeda dari berbagai negara, menganalisis komposisi volatil, profil asam lemak, dan kandungan mineralnya. Dengan demikian klasifikasi varietas almond sangat penting untuk penilaian kualitas dan penentuan nilai pasar. Metode tradisional untuk mengklasifikasikan varietas almond sering kali mengandalkan karakteristik morfologi dan analisis genetik. Namun, metode ini bisa memakan waktu, membutuhkan banyak tenaga kerja, dan rentan terhadap kesalahan. Karena karakteristik morfologi dapat bervariasi dan tes genetik mahal, ada kebutuhan akan metode yang lebih cepat, lebih murah, dan inovatif untuk klasifikasi varietas almond.

Untuk mengatasi keterbatasan metode tradisional, pendekatan berbasis teknologi telah mulai diperkenalkan. Dalam beberapa tahun terakhir, metode otomatis yang mengandalkan pengolahan citra dan pembelajaran mesin telah menunjukkan potensi besar dalam klasifikasi varietas almond. Dengan menggunakan teknik ini, gambar almond dianalisis untuk mengekstraksi berbagai fitur seperti bentuk, warna, dan tekstur, yang kemudian digunakan untuk pelatihan model klasifikasi. Pendekatan ini tidak hanya mempercepat proses klasifikasi tetapi juga meningkatkan akurasi dan konsistensi, mengurangi ketergantungan pada pengamatan manusia yang bisa rentan terhadap kesalahan.

Beberapa model telah dikembangkan yang mampu mengklasifikasikan varietas almond dengan akurasi tinggi menggunakan berbagai algoritma pembelajaran mesin. Teimouri et al. [8] mengambil gambar dari lima jenis almond yaitu Normal (NA), Rusak (BA), Ganda (DA), Keriput (WA), Cangkang (SA) dan mengekstraksi set fitur yang komprehensif. Fitur-fitur tersebut diklasifikasikan dengan Jaringan Syaraf Tiruan (ANN) dan mencapai akurasi: 98,92% untuk NA, 99,46% untuk BA, 98,38% untuk DA, 98,92% untuk WA, dan 100% untuk SA. Dalac et al. [9] mengklasifikasikan lima jenis almond menggunakan fitur bentuk dan warna RGB, dan mencapai tingkat akurasi 92,72% dengan algoritma *Principal Component Analysis* (PCA) dan *Support Vector Machine* (SVM). SVM sendiri merupakan metode pembelajaran mesin yang kuat untuk klasifikasi, yang bekerja dengan mencari *hyperplane* optimal yang memisahkan data dari berbagai kelas [10]. Algoritma ini sangat efektif dalam ruang fitur tinggi dan dikenal dengan kemampuannya pada data yang tidak dapat dipisahkan secara linear untuk ditangani dengan menggunakan *kernel trick* [11], [12]. Muljono et al. [13] mengklasifikasikan *Non-Invasive Anemia* dengan mengekstraksi fitur menggunakan *pre-trained* MobileNetV2 kemudian diklasifikasikan dengan SVM, menghasilkan akurasi 93%. A. Nasirahmadi et al. [14] mengklasifikasikan almond sebagai manis atau pahit dengan mengekstraksi berbagai fitur dan menggunakan *Chi-Square Support Vector Machine* (Chi-SVM) sebagai pengklasifikasi, mencapai tingkat akurasi 91% ( $\pm 1,6$ ). Narendra et al. [15] mengusulkan metode berbasis ekstraksi fitur untuk mengklasifikasikan varietas almond sebagai Mission (MI), Nonpareil (NP), Carmel (CR), dan California (C), mencapai akurasi 97,13% dengan algoritma *Decision Tree* (DT). Selain itu CNN juga merupakan metode yang digunakan untuk klasifikasi almond secara otomatis. Vidyarthi et al. [16] mengklasifikasikan kernel almond menjadi lima kategori menggunakan model CNN *pre-trained* (Inception-V3, ResNet50, VGG-16) dan model kustom, dengan ResNet50 mencapai akurasi tertinggi sebesar 98,4% dan model kustom mencapai akurasi 92,2% tetapi dengan jumlah parameter yang lebih sedikit dibandingkan dengan model *pre-trained*.

Berdasarkan literatur yang telah disebutkan, CNN memberikan hasil yang menjanjikan dalam klasifikasi varietas almond. Namun jumlah studi masih terbatas. Oleh karena itu, penggunaan metode berbasis CNN sangat cocok untuk klasifikasi almond. Namun, tantangan utama yang dihadapi adalah bagaimana membuat model CNN yang tidak hanya akurat tetapi juga efisien dalam hal kecepatan dan penggunaan sumber daya, terutama ketika diimplementasikan pada *embedded device* dengan keterbatasan sumber daya. Untuk mengatasi tantangan ini, diusulkan mengusulkan penggabungan *Support Vector Machine* (SVM) yang dikombinasikan dengan *Convolutional Autoencoder*. SVM dikenal memiliki performa yang baik dalam

klasifikasi dan dapat meningkatkan akurasi keseluruhan sistem. Sementara itu, *Convolutional Autoencoder* (CAE) merupakan variasi dari *autoencoder* yang mengimplementasikan CNN di dalamnya yang dapat digunakan untuk mereduksi dimensi data dan menghilangkan redundansi, sehingga hal ini mempercepat proses klasifikasi tanpa kehilangan informasi penting. Kombinasi teknik ini diharapkan dapat menghasilkan model yang tidak hanya akurat tetapi juga ringan dan cepat, memungkinkan implementasi pada *embedded device*.

Dalam perancangan *Convolutional Autoencoder* (CAE), penggunaan metode *optimizer* memegang peranan penting dalam memperbarui bobot atau kernel dalam model yang sedang dikembangkan. *Autoencoder* adalah jenis arsitektur *neural network* yang dapat digunakan baik dalam paradigma *unsupervised* maupun *supervised learning*. Pada paradigma *unsupervised learning*, seperti yang diterapkan dalam CAE, *autoencoder* bertujuan untuk menghasilkan representasi data yang kompak melalui pembelajaran dari data *input* tanpa memerlukan label *output* yang spesifik. Ini dilakukan dengan mencoba merekonstruksi data input pada outputnya sendiri dengan mengurangi dimensi data ke representasi yang lebih rendah.

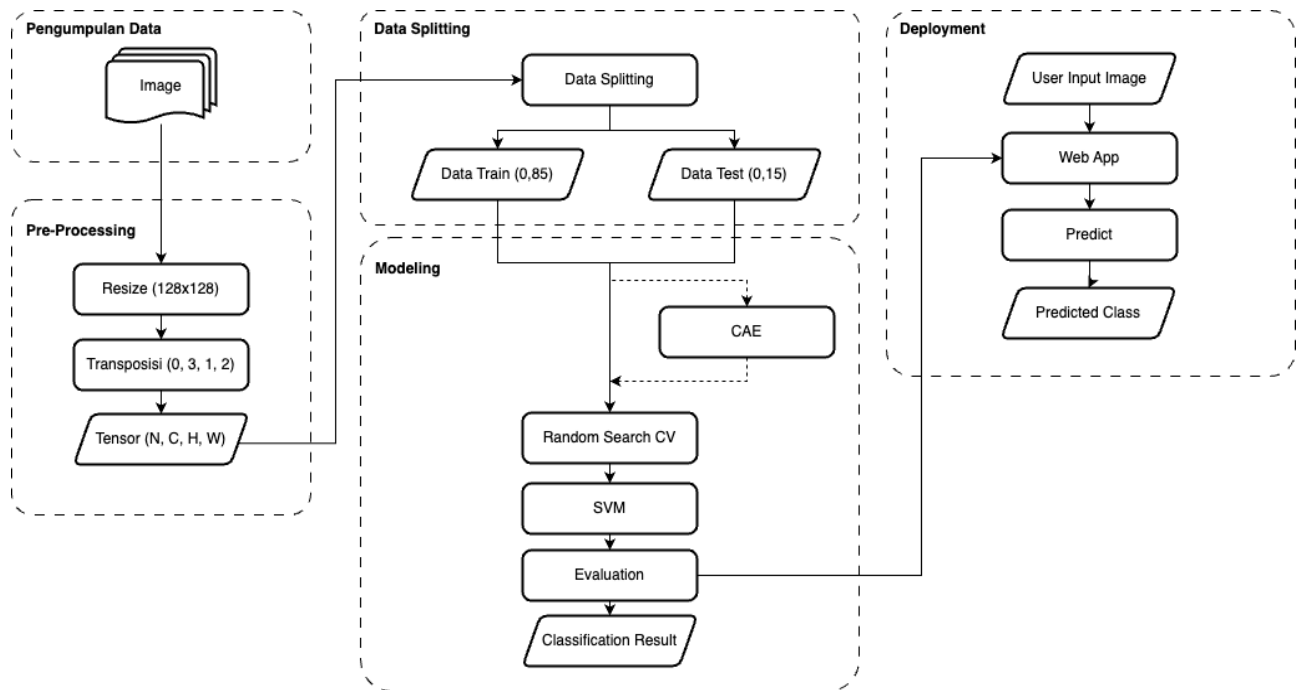
Beberapa *optimizer* umum termasuk Adam (*Adaptive Moment Estimation*), AdamW (*Adaptive Moment Estimation with Decay*), dan SGD (*Stochastic Gradient Descent*). AdamW khususnya relevan dalam konteks penelitian ini karena efisiensi komputasi yang baik, kebutuhan memori yang rendah, serta kemudahan implementasinya dalam berbagai arsitektur model *Deep Learning*, terutama yang berfokus pada *Convolutional Neural Networks* (CNN) [17].

Penelitian ini bertujuan untuk mengembangkan model klasifikasi varietas almond yang lebih akurat dan efisien dengan menggabungkan CAE sebagai *ekstraktor* fitur dan SVM sebagai pengklasifikasi. Model ini diharapkan dapat meningkatkan akurasi klasifikasi dibandingkan dengan metode SVM konvensional, sekaligus mempercepat proses klasifikasi dengan memanfaatkan representasi fitur yang lebih ringkas. Selain itu, penelitian ini juga bertujuan untuk mengevaluasi performa model yang diusulkan dalam skenario nyata serta mengkaji potensi implementasinya pada aplikasi berbasis web pada sisi klien. Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi dalam bidang agrikultur, khususnya dalam klasifikasi varietas almond, serta membuka peluang untuk penerapan serupa pada sistem identifikasi tanaman berbasis citra lainnya.

## II. METODE PENELITIAN

Beberapa tahapan penelitian ini diantaranya pengumpulan data, proses pengumpulan data melibatkan penggunaan dataset yang telah tersedia dari sumber studi sebelumnya, dilanjutkan dengan tahap *preprocessing* untuk mempersiapkan data sebelum dilakukan pembagian data (*data splitting*) untuk pelatihan dan pengujian model. Selanjutnya, fitur-fitur dari gambar-gambar tersebut diekstraksi menggunakan *Convolutional Autoencoder* (CAE) sebagai bagian dari pendekatan *feature extraction*. Setelah itu, klasifikasi dilakukan dengan menggunakan metode *Support Vector Machine* (SVM) pada *latent space*. Hasil dari klasifikasi kemudian divalidasi dan dievaluasi untuk mengukur performa model yang dikembangkan. Implementasi dari seluruh proses ini diuji coba untuk mengonfirmasi validitas dan keefektifan pendekatan yang digunakan dalam penelitian ini, detail dari alur penelitian dapat dilihat pada Gambar 1.

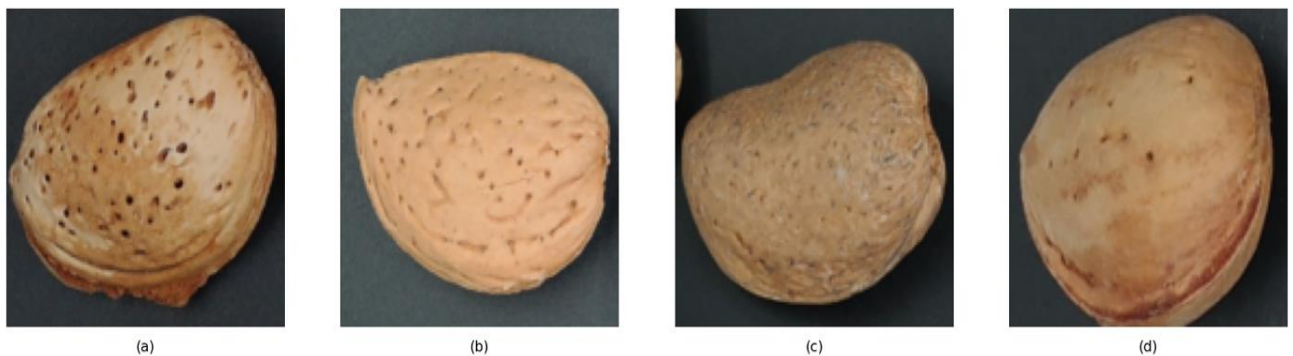
Gambar 1 menjelaskan alur penelitian, proses dimulai dengan pengumpulan dataset gambar yang kemudian diproses melalui *resizing* ke ukuran 128x128 piksel dan transposisi (0,3,1,2) untuk mengubah format dari (N, H, W, C) menjadi (N, C, H, W), di mana N adalah jumlah sampel, C jumlah *channel*, H tinggi, dan W lebar gambar. Data yang telah dikonversi menjadi tensor kemudian dibagi menjadi data latih (85%) dan data uji (15%). Dalam tahap pemodelan, fitur dapat diekstraksi menggunakan *Convolutional Autoencoder* (CAE) sebelum diklasifikasikan oleh *Support Vector Machine* (SVM). Untuk optimasi, digunakan *Random Search Cross-Validation* (CV) pada SVM guna mencari parameter terbaik. Kemudian evaluasi dilakukan untuk menguji performa model SVM dengan dan tanpa bantuan *Convolutional Autoencoder*, kemudian model yang diperoleh diterapkan dalam aplikasi web.



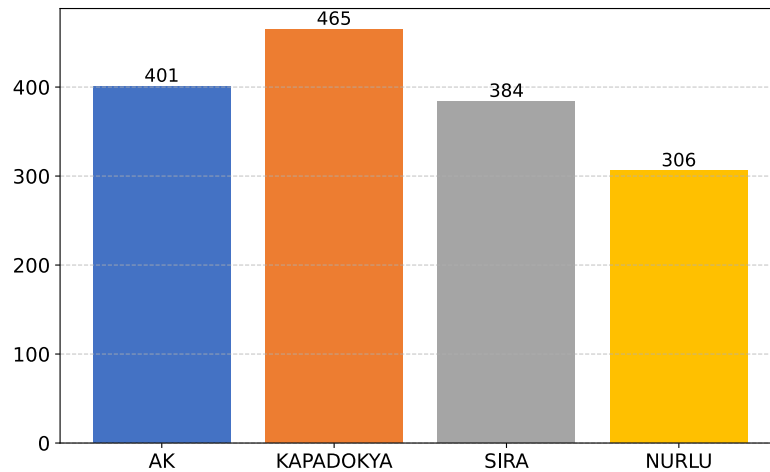
Gambar 1. Alur penelitian mencakup pengumpulan data, pre-processing, pembagian data, modeling, serta deployment.

#### A. Pengumpulan Data

Data yang digunakan dalam penelitian ini berasal dari studi yang dilakukan oleh Yurdakul et al. [18] Dataset terdiri dari 4 kelas yang merepresentasikan varietas almond, yaitu Ak, Kapadokya, Nurlu, dan Sira. Dataset ini terdiri dari total 1556 gambar yang mencakup keempat varietas tersebut. Dataset ini tersedia secara publik di platform Kaggle dengan nama *Almond Varieties*. Sampel dataset dapat dilihat pada Gambar 2.



Gambar 2. Sampel citra almond (a) Ak (b) Kapadokya (c) Nurlu (d) Sira



Gambar 3. Distribusi kelas pada dataset

Gambar 3 distribusi dataset untuk klasifikasi varietas almond adalah sebagai berikut: varietas 'AK' memiliki 401 sampel, varietas 'KAPADOKYA' memiliki 465 sampel, varietas 'SIRA' memiliki 384 sampel, dan varietas 'NURLU' memiliki 306 sampel.

#### B. Preprocessing Data

Tahap *preprocessing* data sangat penting untuk membersihkan dan mempersiapkan data sebelum digunakan dalam pelatihan dan pengujian model. Langkah-langkah *preprocessing* yang dilakukan meliputi:

1) *Resize*: Semua gambar di-*resize* menjadi 128x128 piksel. Proses *resize* ini penting untuk mengurangi ukuran data tanpa kehilangan informasi penting, yang dapat mempercepat proses pelatihan model [19]. Menggunakan ukuran gambar yang seragam juga membantu dalam mengurangi kompleksitas model dan meningkatkan efisiensi komputasi.

2) *Normalisasi Data*: Nilai *piksel* gambar dinormalkan ke rentang 0-1 dengan membagi *tensor* dengan skalar 255 yang merupakan nilai maksimum dari *piksel* dalam format 8bit RGB. Normalisasi dilakukan untuk memastikan bahwa semua fitur *input* berada dalam skala yang sama, yang dapat membantu dalam mempercepat konvergensi selama pelatihan dan meningkatkan performa model [20], [21]. Normalisasi juga membantu dalam mengurangi efek dari perbedaan pencahayaan dalam gambar.

3) *Permutasi Dimensi Data*: Setelah *resize* dan normalisasi, langkah berikutnya adalah melakukan permutasi dimensi data. Permutasi ini mengubah urutan dimensi gambar dari (n, h, w, c) menjadi (n, c, h, w), di mana n adalah jumlah gambar, c adalah jumlah saluran (*channels*), h adalah tinggi gambar, dan w adalah lebar gambar. Format ini sesuai dengan konvensi yang digunakan dalam banyak *framework deep learning* seperti PyTorch. Mengatur data dalam format ini memastikan kompatibilitas dengan model yang akan dilatih dan memfasilitasi operasi *tensor* yang efisien.

#### C. Data Splitting

Dataset dibagi menjadi dua bagian: 85% digunakan sebagai data training sejumlah 1322 gambar dan 15% sebagai data testing sejumlah 234 gambar. Pembagian ini dilakukan dengan menggunakan teknik *stratified sampling* untuk memastikan bahwa proporsi gambar dari setiap varietas almond konsisten dalam kedua subset data. *Stratified sampling* membantu dalam menjaga keseimbangan kelas dalam data training dan data testing, yang penting untuk meningkatkan performa model [22], [23].

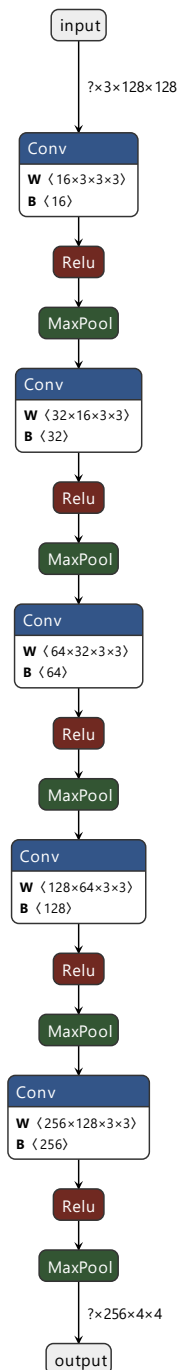
Peningkatan proporsi data training diharapkan dapat meningkatkan kemampuan model untuk belajar dari data dan menghasilkan prediksi yang lebih akurat. Data testing digunakan untuk mengevaluasi performa model dan memastikan bahwa model dapat menggeneralisasi dengan baik pada data yang belum pernah dilihat sebelumnya.

#### D. Feature Extraction dengan CAE

*Convolutional Autoencoder* (CAE) digunakan untuk mengekstraksi fitur dari gambar almond. CAE adalah jenis jaringan saraf tiruan yang dirancang untuk mempelajari representasi data yang lebih ringkas dan informatif melalui proses *encoding* dan *decoding* [24]. Arsitektur CAE yang digunakan dalam penelitian ini terdiri dari dua bagian utama yaitu *Encoder* dan

*Decoder.* CAE dilatih dengan meminimalkan fungsi *loss Mean Squared Error* (MSE) antara gambar *input* dan gambar *output* yang direkonstruksi. MSE mengukur perbedaan rata-rata kuadrat antara nilai *piksel* asli dan nilai *piksel* yang diprediksi [25], sehingga memberikan indikasi seberapa baik model dapat merekonstruksi gambar asli dari representasi fitur yang diekstraksi. Selain itu penggunaan *optimizer AdamW* dengan *learning rate*  $10^{-3}$  pada pelatihan dilakukan untuk memperbarui *weight* pada model CAE sehingga model dapat mencapai *local minima* terkecil.

1) *Encoder.* *Encoder* dirancang dengan menggunakan lima lapisan konvolusi dan lima lapisan *pooling* yang berurutan untuk mengekstraksi fitur dari gambar *input*. Detail arsitektur *encoder* dapat dilihat pada Gambar 4.



Gambar 4. Arsitektur Encoder

Gambar 4 menunjukkan arsitektur *encoder* dalam model *Convolutional Autoencoder* (CAE), yang terdiri dari beberapa lapisan konvolusi (Conv), fungsi aktivasi *ReLU*, dan lapisan *MaxPooling*. Arsitektur ini bertujuan untuk mengekstraksi fitur dari gambar *input* dengan cara mengurangi ukuran spasial sambil meningkatkan jumlah *channel* fitur. Proses dimulai dengan *input layer* yang memiliki dimensi  $? \times 3 \times 128 \times 128$ , di mana ? adalah ukuran *batch* atau jumlah gambar yang akan diproses secara bersamaan, pada proses pelatihan dan pengujian digunakan *batch size* 32 angka ini tidak ditetapkan secara konstan pada model untuk fleksibilitas jumlah input gambar pada saat pelatihan, pengujian, dan implementasi, 3 menunjukkan jumlah *channel Red, Green, dan Blue* (RGB), dan  $128 \times 128$  adalah dimensi gambar (tinggi dan lebar). Selanjutnya, gambar melewati lapisan konvolusi pertama dengan bobot  $W: 16 \times 3 \times 3 \times 3$  dan bias  $B: 16$ , yang berarti terdapat 16 filter dan 16 bias dengan ukuran *kernel*  $3 \times 3$  yang diterapkan pada 3 *channel input*. aktivasi *ReLU* diterapkan untuk menambahkan non-linearitas, diikuti oleh *MaxPooling* untuk mengurangi dimensi fitur map.

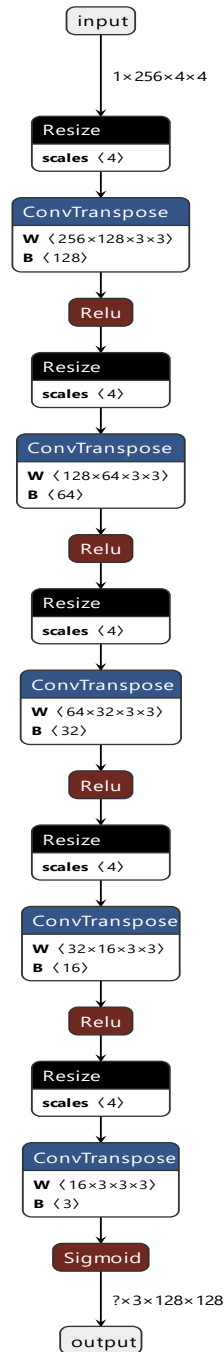
Lapisan berikutnya, lapisan konvolusi kedua, memiliki  $W: 32 \times 16 \times 3 \times 3$  dan  $B: 32$ , di mana 32 filter dan 32 bias digunakan dengan *input* dari 16 filter sebelumnya. Proses ini diikuti oleh aktivasi *ReLU* dan *MaxPooling*. Selanjutnya, lapisan konvolusi ketiga menggunakan  $W: 64 \times 32 \times 3 \times 3$  dengan bias  $B: 64$ , diikuti oleh aktivasi *ReLU* dan *MaxPooling*.

Pada lapisan konvolusi keempat, jumlah filter meningkat menjadi 128 dengan  $W: 128 \times 64 \times 3 \times 3$  dan  $B: 128$ , diikuti oleh aktivasi *ReLU* dan *MaxPooling*. Proses berlanjut ke lapisan konvolusi kelima, di mana jumlah filter bertambah menjadi 256 dengan  $W: 256 \times 128 \times 3 \times 3$  dan  $B: 256$ , lalu kembali diterapkan aktivasi *ReLU* dan *MaxPooling*.

Setelah melalui beberapa tahap konvolusi dan *MaxPooling*, ukuran fitur map menjadi  $? \times 256 \times 4 \times 4$ , di mana ? adalah *batch size* yaitu 32 pada proses pelatihan dan pengujian, 256 adalah jumlah *channel* fitur yang diekstraksi, dan  $4 \times 4$  adalah ukuran fitur map setelah *downsampling*. Dengan arsitektur ini, *encoder* berhasil menangkap pola penting dalam *input* sambil mereduksi dimensinya, sehingga dapat digunakan dalam tahap *decoding* atau klasifikasi lebih lanjut.

2) *Decoder*: *Decoder* dirancang dengan menggunakan lima lapisan *dekonvolusi* dan *upsampling* yang berurutan untuk merekonstruksi gambar dari fitur yang diekstraksi oleh *encoder*. Detail arsitektur *decoder* dapat dilihat pada Gambar 5.

Gambar 5 menunjukkan arsitektur *decoder* dalam model *Convolutional Autoencoder* (CAE), yang bertanggung jawab untuk merekonstruksi kembali gambar dari fitur laten yang telah diekstraksi oleh *encoder*. Arsitektur ini terdiri dari beberapa tahap peningkatan resolusi (*upsampling*) menggunakan metode *nearest neighbor*, diikuti oleh lapisan *transposed convolution* (ConvTranspose) untuk memperbaiki detail gambar yang dihasilkan.



Gambar 5. Arsitektur Decoder

Proses dimulai dengan fitur laten berukuran  $? \times 256 \times 4 \times 4$ , di mana ? adalah ukuran *batch*, 256 merupakan jumlah channel fitur, dan  $4 \times 4$  adalah dimensi fitur map hasil kompresi dari tahap *encoder*. Agar dapat mengembalikan gambar ke ukuran aslinya, fitur ini harus diperbesar secara bertahap. Oleh karena itu, pada setiap tahap, resolusi ditingkatkan menggunakan operasi *resize* dengan faktor skala 2, yang dilakukan menggunakan metode *nearest neighbor*. Metode ini bekerja dengan menyalin nilai piksel terdekat ke posisi baru dalam citra yang diperbesar, sehingga menghasilkan interpolasi sederhana yang cepat dan efisien.

Setelah proses *resize*, fitur dilewatkan ke lapisan *transposed convolution* pertama dengan bobot  $W: 128 \times 256 \times 3 \times 3$  dan bias  $B: 128$ , yang berarti terdapat 128 filter dan 128 bias dengan *kernel*  $3 \times 3$  yang diterapkan pada *feature map* sebelumnya.



Fungsi aktivasi *ReLU* digunakan setelahnya untuk menambahkan elemen non-linearitas dan meningkatkan kemampuan jaringan dalam menangkap pola kompleks.

Proses ini berlanjut dengan tahapan *resize* dan *transposed convolution* berikutnya. Pada tahap kedua, jumlah *channel* berkurang menjadi 64 dengan bobot  $W: 64 \times 128 \times 3 \times 3$ , sementara resolusi meningkat menjadi  $32 \times 32$ . Tahapan ini terus berulang, di mana resolusi terus bertambah menjadi  $64 \times 64$  dan  $128 \times 128$ , sementara jumlah *channel* fitur semakin berkurang (32 kemudian 16). Setiap tahap tetap menggunakan fungsi aktivasi *ReLU* untuk mempertahankan karakteristik *non-linear* pada rekonstruksi.

Pada lapisan terakhir, dilakukan operasi *transposed convolution* dengan bobot  $W: 3 \times 16 \times 3 \times 3$  dan bias  $B: 3$ , yang mengurangi jumlah *channel* menjadi 3, sesuai dengan format gambar RGB. Fungsi aktivasi *Sigmoid* diterapkan untuk memastikan bahwa nilai *piksel* hasil rekonstruksi berada dalam rentang 0-1. Fungsi *Sigmoid* didefinisikan pada persamaan (1).

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (1)$$

Pada persamaan (1),  $x$  merupakan input ke fungsi sigmoid, yang bisa berupa nilai apapun dari  $-\infty$  hingga  $\infty$ , dan  $\sigma(x)$  adalah output dari fungsi sigmoid, yang merupakan nilai dalam rentang  $[0, 1]$ .

3) *Ekstraksi Fitur*: Setelah gambar direkonstruksi oleh *decoder*, fitur yang diekstraksi dari *encoder* digunakan untuk klasifikasi dengan *Support Vector Machine* (SVM). Dalam proses ini, *tensor input* ( $N, C, H, W$ ) melewati serangkaian lapisan konvolusi dengan filter  $3 \times 3$ , aktivasi *ReLU*, dan *max pooling* untuk mereduksi dimensi spasial sambil mempertahankan informasi penting.

Pada awalnya, setiap gambar memiliki dimensi ( $N, 3, 128, 128$ ), yang berarti terdapat  $N$  gambar berwarna dengan 3 *channel* (RGB) dan resolusi  $128 \times 128$  *piksel*. Setelah melewati seluruh lapisan *encoder*, *tensor* tersebut mengalami transformasi bertahap hingga akhirnya memiliki dimensi ( $N, 256, 4, 4$ ), yang menunjukkan bahwa setiap gambar telah direpresentasikan sebagai 256 *feature maps* berukuran  $4 \times 4$ . Agar dapat digunakan oleh SVM, *tensor* ini di-*flatten* menjadi vektor berdimensi 4096 ( $256 \times 4 \times 4$ ). Vektor fitur ini kemudian menjadi masukan utama bagi SVM dalam proses klasifikasi.

#### E. Randomized Search CV

Pada penelitian ini, *Random Search CV* (*Random Search Cross-Validation*) adalah teknik yang diterapkan pada *Support Vector Machine* (SVM) dengan fokus pada dua parameter utama, yaitu  $C$  dan  $\gamma$ . Parameter  $C$  berperan dalam mengontrol tingkat regulasi SVM, di mana nilai yang lebih besar menyebabkan model lebih ketat dalam memisahkan kelas, tetapi berisiko mengalami *overfitting*, sementara nilai yang lebih kecil membuat model lebih toleran terhadap kesalahan klasifikasi, sehingga lebih *generalizable*. Berbeda dengan *Grid Search CV* yang mencoba semua kombinasi *hyperparameter* yang mungkin, *Random Search CV* lebih cepat dan efisien, terutama ketika jumlah *hyperparameter* yang harus dicoba sangat banyak [26]. Dalam *Random Search CV*, distribusi probabilitas dapat ditentukan untuk masing-masing *hyperparameter* sehingga beberapa nilai lebih mungkin dipilih daripada yang lain.

TABEL 1  
DISTRIBUSI PARAMETER RANDOMIZED SEARCH CV

Parameter	Nilai
C	$[10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2, 10^3]$
Gamma ( $\gamma$ )	$[10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2, 10^3]$

Tabel 1 parameter  $C$  dan  $\gamma$ , digunakan distribusi logaritmik dari antara  $10^{-3}$  hingga  $10^4$ . dan menggunakan K-Fold=10 untuk melakukan *cross validation* pada *Random Search CV*. Pendekatan ini memungkinkan *Random Search CV* untuk secara efisien mengeksplorasi ruang pencarian *hyperparameter* yang luas dengan mencoba sejumlah kombinasi acak.

#### F. Klasifikasi dengan SVM

Setelah fitur diekstraksi menggunakan CAE, *Support Vector Machine* (SVM) digunakan untuk mengklasifikasikan citra almond berdasarkan fitur tersebut. SVM adalah algoritma klasifikasi yang efektif dalam memisahkan data ke dalam kelas yang berbeda dengan menemukan garis linear atau *hyperplane* yang memaksimalkan margin antara kelas [27]. Garis ini diposisikan untuk memiliki jarak terbesar antara garis dengan titik-titik dari kelas-kelas yang berbeda (*support vector*) [28].

Dalam penelitian ini, *kernel Radial Basis Function* (RBF) digunakan untuk meningkatkan fleksibilitas SVM dalam menangani data yang tidak linear. *Kernel* RBF memungkinkan SVM untuk memetakan data ke dalam ruang dimensi yang lebih tinggi, di mana data lebih mudah dipisahkan oleh *hyperplane* [29]. Penggunaan *kernel* RBF dapat membantu dalam

meningkatkan akurasi model, terutama ketika data memiliki kompleksitas dan variasi yang tinggi. *Kernel* RBF didefinisikan pada persamaan (2).

$$K(x_i, x_j) = \exp(-\gamma ||x_i - x_j||^2) \quad (2)$$

Pada persamaan (2)  $x_i$  dan  $x_j$  adalah vektor fitur yang sedang dihitung kernelnya.  $|x_i - x_j|^2$  mengacu pada kuadrat jarak *Euclidean* antara dua vektor fitur tersebut. Parameter  $\gamma$  mengontrol lebar atau skala dari kernel RBF. Semakin besar nilai  $\gamma$ , kernel RBF akan lebih sempit. Sebaliknya, nilai  $\gamma$  yang lebih kecil akan menghasilkan kernel RBF yang lebih luas, memungkinkan sampel yang lebih jauh memiliki nilai kernel yang tinggi.

#### G. Evaluasi

Performa model dievaluasi menggunakan beberapa metrik, yaitu akurasi, presisi, *recall*, dan *F1-score*. Metrik-metrik ini dihitung berdasarkan hasil klasifikasi pada data testing dan memberikan indikasi seberapa baik model dapat mengklasifikasikan varietas almond. Evaluasi performa model dilakukan dengan membandingkan hasil prediksi dengan label asli pada data testing. Ini membantu dalam mengidentifikasi kekuatan dan kelemahan model serta area di mana model dapat ditingkatkan. Evaluasi performa model dilakukan dengan membandingkan hasil prediksi dengan label asli pada data testing. Ini membantu dalam mengidentifikasi kekuatan dan kelemahan model serta area di mana model dapat ditingkatkan.

1) *Akurasi*: Akurasi mengukur persentase prediksi yang benar dari total prediksi yang dibuat oleh model. Akurasi menilai kemampuan model secara keseluruhan untuk memprediksi dengan benar. Hal ini sangat penting untuk mengevaluasi kinerja model secara keseluruhan.

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

Pada persamaan (3) TP adalah *True Positive* (prediksi positif yang benar), TN adalah *True Negative* (prediksi negatif yang benar), FP adalah *False Positive* (prediksi positif yang salah), dan FN adalah *False Negative* (prediksi negatif yang salah).

2) *Presisi*: Presisi mengukur jumlah prediksi positif yang benar dibagi dengan total prediksi positif yang dibuat oleh model. Presisi dapat didefinisikan pada persamaan (4).

$$Presisi = \frac{TP}{TP+FP} \quad (4)$$

Presisi memberikan informasi tentang seberapa banyak dari prediksi positif yang sebenarnya benar [30].

3) *Recall*: *Recall* mengukur jumlah prediksi positif yang benar dibagi dengan total data aktual yang positif. *Recall* memberikan gambaran tentang kemampuan model untuk mengidentifikasi semua contoh positif yang sebenarnya. Rumus untuk menghitung *Recall* didefinisikan pada persamaan (5).

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

4) *F1-Score*: *F1-Score* menghitung rata-rata harmonis dari presisi dan *recall*, memberikan indikasi keseimbangan antara keduanya. *F1-Score* berguna ketika ada ketidakseimbangan kelas (*imbalance class*) dan memperhitungkan baik presisi maupun *recall*. Rumus untuk menghitung *F1-Score* dituliskan pada persamaan (6).

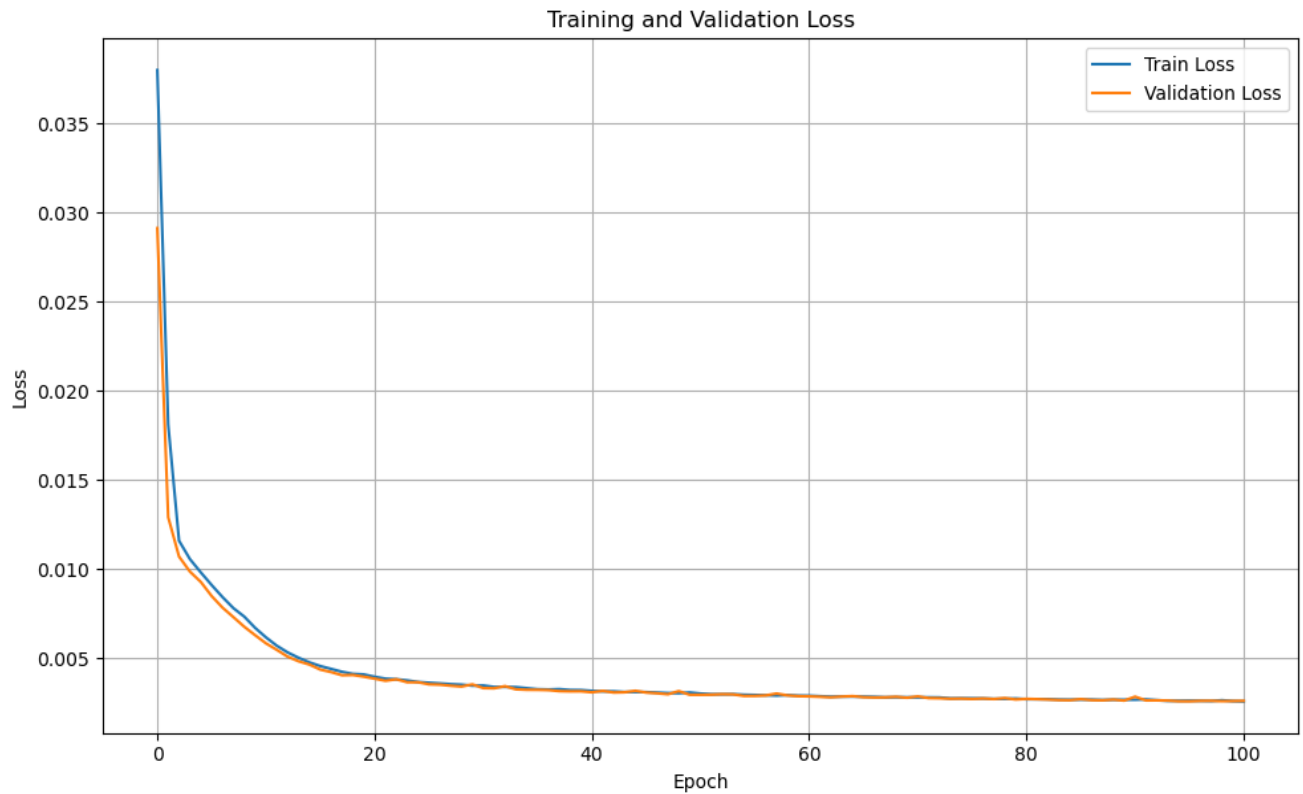
$$F1-Score = 2 \times \frac{Presisi \times Recall}{Presisi + Recall} \quad (6)$$

#### H. Implementasi

Implementasi pada penelitian ini menggabungkan *Encoder* dari *Convolutional Autoencoder* (CAE) yang telah dilatih untuk mengekstraksi fitur dari gambar daun almond dan *Support Vector Machine* (SVM) sebagai *classifier* untuk mengklasifikasikan varietas almond berdasarkan fitur yang diekstraksi. Pada proses ini juga dilakukan penyisipan lapisan *Flatten* antara *encoder* dan *classifier* untuk menyatukan representasi fitur dalam format yang sesuai dengan input yang diterima oleh SVM. Setelah semua model terintegrasi, model akan di-*deploy* dalam aplikasi web *client-side*.

### III. HASIL DAN PEMBAHASAN

#### A. Feature Extraction dengan CAE



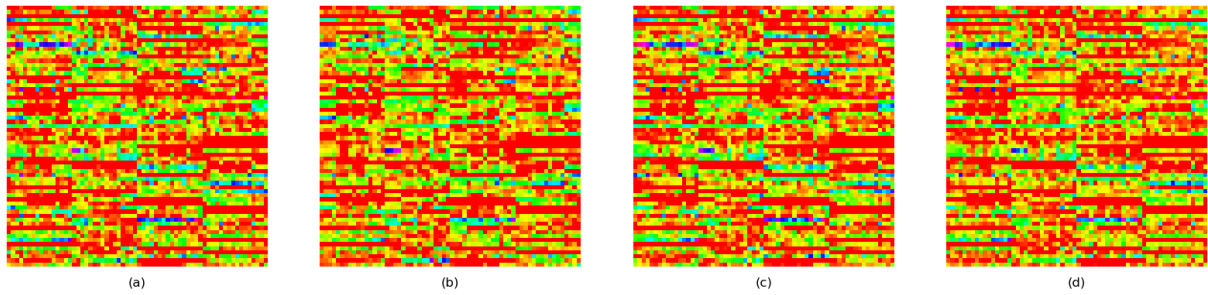
Gambar 6. Plot loss Convolutional Autoencoder

Gambar 6 menjelaskan bahwa proses pelatihan model berhenti setelah mencapai *validation loss* sekitar 0.00258, dengan *training loss* sekitar 0.00256 pada *epoch* ke-100, proses pelatihan dihentikan karena tidak terdapat *improvement* lagi pada lima *epoch* terakhir. Hasil ini menunjukkan bahwa model telah mencapai tingkat performa yang sangat baik dalam mempelajari pola-pola yang ada dalam dataset. Penggunaan *learning rate*  $10^{-3}$  dan *optimizer* AdamW telah membantu model untuk konvergen dengan efisien, menghasilkan representasi yang lebih kompak dan informatif dari data.



Gambar 7. Reconstructed Image (a) Ak (b) Kapadokya (c) Nurlu (d) Sira

Gambar 7 menunjukkan citra rekonstruksi dari *Convolutional Autoencoder*, terlihat empat citra hasil rekonstruksi (a), (b), (c), dan (d) yang tampak seperti bentuk asli objek yang direkonstruksi oleh *autoencoder*. Meskipun ada sedikit *blur*, bentuk dan struktur utama dari citra aslinya masih dapat dikenali, menunjukkan kemampuan *autoencoder* dalam menangkap dan merekonstruksi fitur penting dari data *input*.



Gambar 8. Latent Space (a) Ak (b) Kapadokya (c) Nurlu (d) Sira

Gambar 8 menampilkan representasi *latent space* dari *convolutional autoencoder* untuk empat varietas almond (Ak, Kapadokya, Nurlu, Sira), dengan perbedaan warna menunjukkan nilai fitur yang telah dinormalisasi antara 0-1 untuk setiap varietas. Variasi pola warna ini mengindikasikan bagaimana encoder mengekstrak dan memetakan karakteristik unik dari masing-masing varietas almond.

#### B. Randomized Search CV

Dalam penelitian ini, dilakukan perbandingan antara SVM dan CAE-SVM menggunakan metode *Randomized Search CV* untuk mengoptimalkan parameter C dan Gamma.

TABEL 2  
RANDOMIZED SEARCH CV SVM

<i>Gamma</i>	<i>C</i>	Rata-Rata Akurasi	Rata-Rata Waktu(m)
0.01	10.0	0.422	29.101
0.001	0.01	0.299	25.811
0.01	100.0	0.422	26.186
1.0	100.0	0.300	26.425
0.1	1000.0	0.300	23.223
1.0	0.01	0.299	34.106
10.0	1.0	0.300	32.313
100.0	0.001	0.299	34.779
0.1	0.1	0.299	26.175
1000.0	100.0	0.300	17.032
Rata-Rata			25.498

Tabel 2 menunjukkan hasil dari SVM, di mana parameter terbaik yang ditemukan adalah  $C=100.0$  dan  $\text{Gamma}=0.01$ , dengan rata-rata akurasi sebesar 0.422. Waktu total yang dibutuhkan untuk setiap *cross-validation (CV)* mencatat rata-rata 25.498 menit.

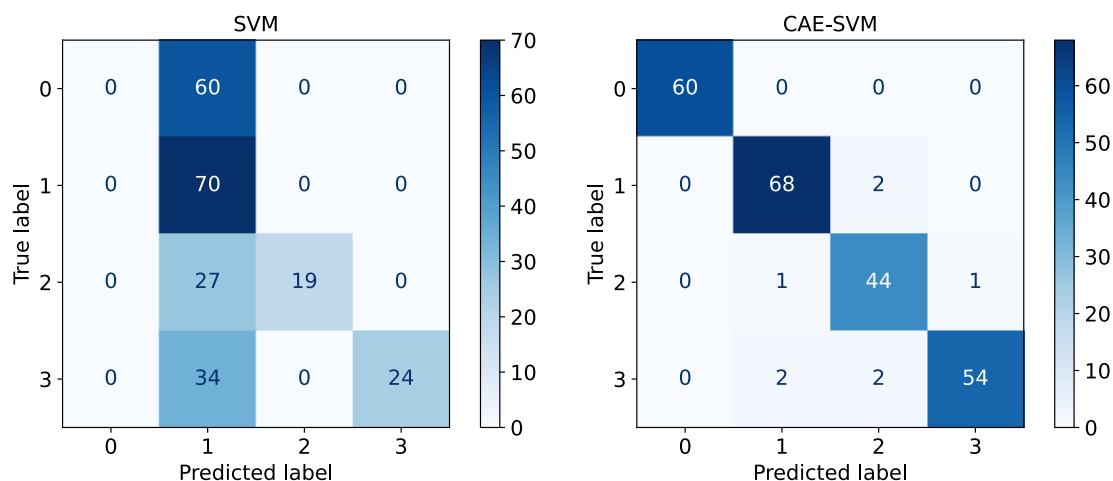
TABEL 3  
RANDOMIZED SEARCH CV CAE-SVM

<i>Gamma</i>	<i>C</i>	Rata-Rata Akurasi	Rata-Rata Waktu(m)
0.01	10.0	0.938	0.338
0.001	0.01	0.299	1.123
0.01	100.0	0.939	0.320
1.0	100.0	0.347	1.321
0.1	1000.0	0.924	0.794
1.0	0.01	0.299	1.255
10.0	1.0	0.302	1.315
100.0	0.001	0.299	1.314
0.1	0.1	0.771	1.051
1000.0	100.0	0.300	1.130
Rata-rata			0.996

Tabel 3 menggambarkan hasil dari CAE-SVM menunjukkan parameter terbaik dengan  $C=10.0$  dan  $\text{Gamma}=0.01$ , memberikan rata-rata akurasi sebesar 0.938. Waktu yang diperlukan untuk setiap CV pada CAE-SVM rata-rata 0.996 menit. Perbandingan ini menunjukkan bahwa meskipun SVM menghasilkan waktu komputasi yang lebih lama akan tetapi tidak memberikan akurasi yang lebih baik, sedangkan CAE-SVM memberikan performa yang lebih baik dalam hal akurasi pada dataset yang digunakan.

### C. Evaluasi

Dalam penelitian ini, dilakukan perbandingan antara model *Support Vector Machine* (SVM) konvensional dengan SVM yang dioptimasi dengan fitur yang diekstraksi menggunakan *Convolutional Autoencoder* (CAE) pada dataset klasifikasi varietas almond. Penggunaan CAE bertujuan untuk meningkatkan kemampuan SVM dalam memahami fitur-fitur yang lebih abstrak dan mendalam dari data gambar, dibandingkan dengan representasi fitur yang hanya didasarkan pada *piksel* mentah.



Gambar 9. Confusion Matrix SVM dan CAE-SVM

Gambar 9 menunjukkan bahwa CAE-SVM memiliki performa yang lebih baik dalam klasifikasi data dibandingkan dengan SVM biasa. Dalam *confusion matrix* SVM, terlihat bahwa kelas 2 dan kelas 3 mengalami jumlah kesalahan prediksi yang signifikan, di mana banyak sampel kelas 2 diprediksi sebagai kelas 0 atau kelas 3, dan banyak sampel kelas 3 diprediksi sebagai kelas 0. Sebaliknya, matriks kebingungan CAE-SVM menunjukkan peningkatan akurasi dengan lebih banyak prediksi yang benar untuk kelas 2 dan kelas 3. Model CAE-SVM memprediksi dengan benar lebih banyak sampel pada kelas-kelas tersebut dan mengurangi jumlah kesalahan prediksi. Distribusi kesalahan pada model CAE-SVM juga lebih merata dan jumlah kesalahan prediksi lebih sedikit dibandingkan SVM.

TABEL 4  
CLASSIFICATION REPORT SVM

Kelas	Precision	Recall	F1-Score
AK	0.00	0.00	0.00
KAPADOKYA	0.37	1.00	0.54
NURLU	1.00	0.41	0.58
SIRA	1.00	0.41	0.59
Rata-Rata	0.55	0.48	0.42
Akurasi	0.48		

Pada Tabel 4 model SVM konvensional yang dilatih langsung pada data gambar menghasilkan akurasi rata-rata sebesar 48% pada data testing, dengan nilai presisi 55%, *recall* 48%, dan *F1-score* 42%. Model ini menggunakan *kernel rbf* dan dijalankan dengan parameter  $C$  (*regularization*)  $10^2$  dan  $\text{gamma}$  (*kernel parameter*)  $10^{-2}$ .

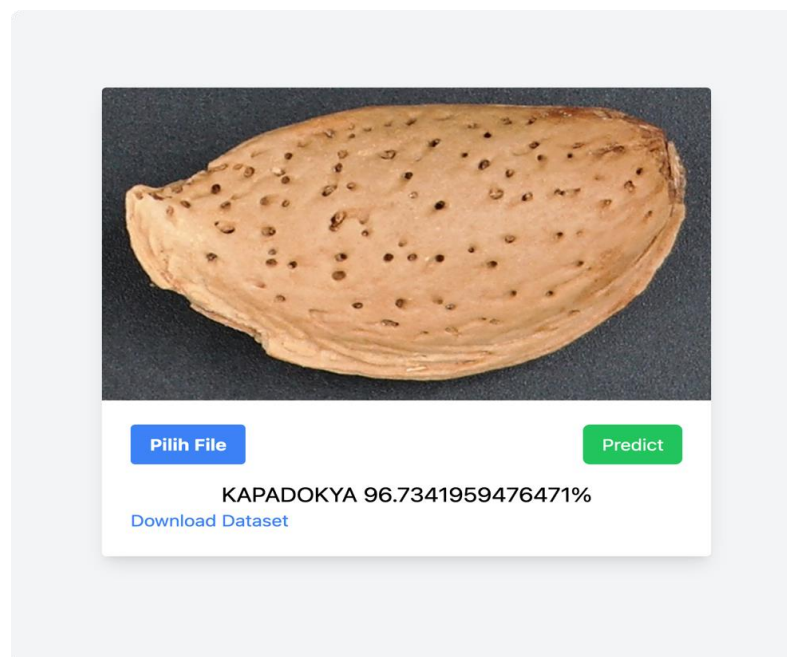
TABEL 5  
CLASSIFICATION REPORT SVM DENGAN CAE

Kelas	Precision	Recall	F1-Score
AK	1.00	1.00	1.00
KAPADOKYA	0.96	0.97	0.96
NURLU	0.92	0.96	0.94
SIRA	0.98	0.93	0.96
Rata-Rata	0.97	0.97	0.97
Accuracy			0.97

Tabel 5 SVM yang dioptimasi dengan fitur yang diekstraksi menggunakan CAE menunjukkan peningkatan signifikan dalam performa. Model ini menggunakan *encoder* dari CAE untuk mengekstraksi representasi fitur yang lebih ringkas dan informatif dari gambar. Pada eksperimen, SVM dengan CAE mencapai akurasi rata-rata sebesar 97% pada data testing, dengan nilai presisi 97%, *recall* 97%, dan *F1-score* 97% yang dijalankan dengan parameter  $C=10$  dan  $\gamma=10^{-1}$ . Hasil ini menunjukkan bahwa penggunaan CAE sebagai fitur *ekstraktor* dapat meningkatkan kemampuan SVM dalam memisahkan kelas-kelas yang kompleks dalam dataset gambar varietas almond.

#### D. Implementasi

Setelah *Encoder* dan SVM diintegrasikan model tersebut dapat berjalan dengan baik pada aplikasi web.



Gambar 10. Implementasi pada aplikasi web *client side*

Gambar 10 menunjukkan bahwa model CAE-SVM dapat diimplementasikan dengan baik pada aplikasi web *client side*. Aplikasi web dapat diakses pada url <https://izzalDev.github.io/almost-classification>.

#### IV. SIMPULAN

Penelitian ini bertujuan untuk mengoptimalkan model *Support Vector Machine* (SVM) dalam klasifikasi varietas almond menggunakan *Convolutional Autoencoder* (CAE) sebagai pendekatan untuk ekstraksi fitur. Data yang digunakan berasal dari dataset yang terdiri dari empat kelas varietas almond, yaitu AK, Kapadokya, Nurlu, dan Sira. Proses eksperimen melibatkan tahap pengumpulan data, *preprocessing*, dan pembagian data untuk *training* dan *testing*. CAE digunakan untuk mengekstraksi fitur dari gambar almond, sementara SVM digunakan untuk klasifikasi berdasarkan fitur-fitur tersebut. Hasil evaluasi menunjukkan bahwa penggunaan CAE meningkatkan secara signifikan performa SVM dalam memisahkan kelas-kelas yang kompleks, dengan akurasi rata-rata mencapai 97% pada data testing. Implementasi model ini juga menunjukkan

bahwa CAE-SVM dapat diintegrasikan dengan baik pada aplikasi web *client side*, menunjukkan potensi untuk aplikasi praktis dalam identifikasi varietas almond secara otomatis.

Untuk penelitian selanjutnya, disarankan untuk memperluas dataset dengan lebih banyak variasi kondisi pencahayaan dan konteks tumbuhnya almond. Hal ini dapat membantu dalam meningkatkan generalisasi model terhadap situasi dunia nyata yang lebih kompleks. Selain itu, eksplorasi lebih lanjut dalam pengoptimalan parameter CAE dan SVM, serta penggunaan teknik *ensemble* atau *deep learning* yang lebih canggih, dapat menjadi langkah-langkah berikutnya untuk meningkatkan akurasi dan keandalan sistem klasifikasi varietas almond berbasis gambar ini.

#### DAFTAR PUSTAKA

- [1] M. Zeinalabedini, M. Khayam-Nekoui, V. Grigorian, T. M. Gradziel, and P. Martínez-Gómez, "The origin and dissemination of the cultivated almond as determined by nuclear and chloroplast SSR marker analysis," *Sci Hortic*, vol. 125, no. 4, pp. 593–601, 2010.
- [2] K. W. C. Sze-Tao and S. K. Sathe, "Functional properties and in vitro digestibility of almond (*Prunus dulcis* L.) protein isolate," *Food Chem*, vol. 69, no. 2, pp. 153–160, 2000.
- [3] D. Barreca et al., "Almonds (*Prunus Dulcis* Mill. D. A. Webb): A Source of Nutrients and Health-Promoting Compounds," *Nutrients*, vol. 12, no. 3, p. 672, Mar. 2020.
- [4] T. Banjanin et al., "Physicochemical properties, fatty acids, phenolic compounds, and mineral contents of 12 Serbia regional and commercial almond cultivars," *J Food Process Preserv*, vol. 45, no. 1, 2021.
- [5] J. Ghafarzadeh, K. Sadeghniai-Haghighi, O. Sadeghpour, S. Akbarpour, and F. Amini-Behbahani, "Investigating the Prevalence of Sleep Disorder and the Impact of Sweet Almond on the Quality of Sleep in Students of Tehran, Iran.," *Iran J Public Health*, vol. 48, no. 6, pp. 1149–1154, 2019.
- [6] D. A. Sumner, W. A. Matthews, J. Medellín-Azuara, and A. Bradley, "The Economic Impacts of the California Almond Industry A Report Prepared for the Almond Board of California." Accessed: Feb. 20, 2025. [Online]. Available: [https://www.almonds.com/sites/default/files/2020-04/the\\_economic\\_impacts\\_of\\_the\\_california\\_almond\\_industry.pdf](https://www.almonds.com/sites/default/files/2020-04/the_economic_impacts_of_the_california_almond_industry.pdf)
- [7] O. Levent, "A Detailed Comparative Study on Some Physicochemical Properties, Volatile Composition, Fatty Acid, and Mineral Profile of Different Almond (*Prunus dulcis* L.) Varieties," *Horticulturae*, vol. 8, no. 6, 2022.
- [8] N. Teimouri, M. Omid, K. Mollazade, and A. Rajabipour, "An Artificial Neural Network-Based Method to Identify Five Classes of Almond According to Visual Features," *J Food Process Eng*, vol. 39, no. 6, pp. 625–635, 2016.
- [9] D. Halac, E. Sokic, and E. Turajlic, "Almonds classification using supervised learning methods," in 2017 XXVI International Conference on Information, Communication and Automation Technologies (ICAT), IEEE, 2017, pp. 1–6.
- [10] S. Ding, X. Hua, and J. Yu, "An overview on nonparallel hyperplane support vector machine algorithms," *Neural Comput Appl*, vol. 25, no. 5, pp. 975–982, 2014.
- [11] M. A. Araaf, K. Nugroho, and D. R. I. M. Setiadi, "Comprehensive Analysis and Classification of Skin Diseases based on Image Texture Features using K-Nearest Neighbors Algorithm," *Journal of Computing Theories and Applications*, vol. 1, no. 1, pp. 31–40, 2023.
- [12] F. Mustofa, A. N. Safriandono, A. R. Muslikh, and D. R. I. M. Setiadi, "Dataset and Feature Analysis for Diabetes Mellitus Classification using Random Forest," *Journal of Computing Theories and Applications*, vol. 1, no. 1, pp. 41–48, 2023.
- [13] Muljono, S. A. Wulandari, H. Al Azies, M. Naufal, W. A. Prasetyanto, and F. A. Zahra, "Breaking Boundaries in Diagnosis: Non-Invasive Anemia Detection Empowered by AI," *IEEE Access*, vol. 12, pp. 9292–9307, 2024.
- [14] A. Nasirahmadi and S.-H. Miraei Ashtiani, "Bag-of-Feature model for sweet and bitter almond classification," *Biosyst Eng*, vol. 156, pp. 51–60, 2017.
- [15] P. Kamath, "ALMOND KERNEL VARIETY IDENTIFICATION AND CLASSIFICATION USING DECISION TREE," *Journal of Engineering Science and Technology*, vol. 16, no. 5, pp. 3923–3942, 2021.
- [16] S. K. Vidyarthi, S. K. Singh, H. Xiao, and R. Tiwari, "Deep learnt grading of almond kernels," *J Food Process Eng*, vol. 44, no. 4, 2021.
- [17] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Dec. 2014, [Online]. Available: <http://arxiv.org/abs/1412.6980>
- [18] M. Yurdakul, İ. Atabaş, and Ş. Taşdemir, "Almond (*Prunus dulcis*) varieties classification with genetic designed lightweight CNN architecture," *European Food Research and Technology*, 2024.
- [19] M. Liu et al., "An Improved Flexible Spatiotemporal Data Fusion (IFSADF) method for producing high spatiotemporal resolution normalized difference vegetation index time series," *Remote Sens Environ*, vol. 227, pp. 74–89, 2019.
- [20] G. A. Kaissis, M. R. Makowski, D. Rückert, and R. F. Braren, "Secure, privacy-preserving and federated machine learning in medical imaging," *Nat Mach Intell*, vol. 2, no. 6, pp. 305–311, 2020.
- [21] N. Moshkov, B. Mathe, A. Kertesz-Farkas, R. Hollandi, and P. Horvath, "Test-time augmentation for deep learning-based cell segmentation on microscopy images," *Sci Rep*, vol. 10, no. 1, 2020.
- [22] P. Wagner et al., "PTB-XL, a large publicly available electrocardiography dataset," *Sci Data*, vol. 7, no. 1, 2020.
- [23] S. Demir and E. K. Sahin, "Comparison of tree-based machine learning algorithms for predicting liquefaction potential using canonical correlation forest, rotation forest, and random forest based on CPT data," *Soil Dynamics and Earthquake Engineering*, vol. 154, p. 107130, 2022.
- [24] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction," in Springer Berlin Heidelberg, vol. 6791, 2011, pp. 52–59.
- [25] U. Sara, M. Akter, and M. S. Uddin, "Image Quality Assessment through FSIM, SSIM, MSE and PSNR—A Comparative Study," *Journal of Computer and Communications*, vol. 07, no. 03, pp. 8–18, 2019.
- [26] P. Liashchynskyi and P. Liashchynskyi, "Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS," Dec. 2019, [Online]. Available: <http://arxiv.org/abs/1912.06059>
- [27] X. Yang et al., "Preferred vector machine for forest fire detection," *Pattern Recognit*, vol. 143, p. 109722, 2023.
- [28] N. Shafaf and H. Malek, "Applications of Machine Learning Approaches in Emergency Medicine; a Review Article.," *Arch Acad Emerg Med*, vol. 7, no. 1, p. 34, 2019, [Online]. Available: <http://www.ncbi.nlm.nih.gov/pubmed/31555764>
- [29] Z. Liu, Z. Dang, and J. Yu, "Stock Price Prediction Model Based on RBF-SVM Algorithm," in 2020 International Conference on Computer Engineering and Intelligent Control (ICCEIC), IEEE, 2020, pp. 124–127.
- [30] M. Grandini, E. Bagli, and G. Visani, "Metrics for Multi-Class Classification: an Overview," Aug. 2020, [Online]. Available: <http://arxiv.org/abs/2008.05756>