Jurnal Elektronik Ilmu Komputer Udayana
Volume 15, No 1. August 2026

p-ISSN: 2301-5373
e-ISSN: 2654-5101

Klasifikasi Spesies Jamur Menggunakan Metode Hibrida DenseNet-121 dan Support Vector Machine

I Dewa Made Mardana[a1], Luh Gede Astuti[a2], I Dewa Made Bayu Atmaja Darmawan[a3], I Wayan Supriana[a4]

Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Udayana
Jln. Raya Kampus UNUD, Bukit Jimbaran, Kuta Selatan, Badung, 08261, Bali, Indonesia

[1]dewamardana33@email.com (Corresponding author)
[2]lg.astuti@unud.ac.id
[3]dewabayu@unud.ac.id
[4]wayan.supriana@unud.ac.id

Abstract

The public's inability to distinguish between the visual characteristics of poisonous and edible mushrooms contributes to a high risk of wild mushroom poisoning. This study developed a mushroom image classification system using a hybrid method combining a Convolutional Neural Network (CNN) based on the DenseNet-121 architecture and a Support Vector Machine (SVM), which was tested through four comparative modeling schemes. The four schemes include: (1) training DenseNet-121 from scratch, (2) integrating DenseNet-121 with SVM without Transfer Learning, (3) Transfer Learning of DenseNet-121 using ImageNet weights, and (4) Transfer Learning + Fine-Tuning + SVM. Each scheme was evaluated using three batch size variations (4, 8, 16) and two optimizers (Adam and RMSProp). The results show that Scheme 4, using the Adam optimizer and a batch size of 4, achieved the highest average accuracy of 92.37%, far surpassing the baseline model in Scheme 1, which reached only 82.40%. The best model was then converted to the TensorFlow Lite format and implemented in an Android application. Testing on 180 new images (unseen data) recorded an accuracy of 94.44% with a total inference time of only 0.50 seconds per image. This study proves that the DenseNet-121 + SVM hybrid architecture with a fine-tuning strategy excels in mushroom classification and is suitable as a mobile-based educational tool for preventing poisoning.

Keywords: *Mushroom Classification, Convolutional Neural Networks, DenseNet-121, Support Vector Machine, Transfer Learning, TensorFlow Lite*

1. Pendahuluan

Layout Jamur merupakan organisme yang berperan penting dalam keseimbangan ekosistem dan banyak dimanfaatkan di bidang pangan maupun medis [1], [2]. Dari estimasi 1,5 juta spesies jamur di dunia, diperkirakan sekitar 200.000 spesies tumbuh di Indonesia [3]. Namun, keberagaman tersebut berbanding lurus dengan tingginya risiko keracunan akibat mengonsumsi jamur liar yang mengandung toksin mematikan [4]. Penelitian menunjukkan bahwa dari 66 spesies jamur liar di satu kawasan hutan, hanya 6 spesies yang aman dikonsumsi, sementara sisanya beracun [5]. Di Tiongkok, pada tahun 2024 tercatat 599 insiden keracunan yang mengakibatkan 1.486 korban dirawat intensif dan 13 di antaranya meninggal dunia [6].

Di Indonesia, kondisi serupa juga mengkhawatirkan. Banyak spesies jamur liar beracun memiliki morfologi yang sangat mirip dengan jamur yang aman dikonsumsi [7]. Kemiripan inilah yang memicu kesalahan identifikasi, seperti kasus kekeliruan antara jamur liar beracun dari kelompok Coprinoid dengan jamur merang (Volvariella volvacea) yang mencatatkan 48 korban di berbagai provinsi Indonesia [8]. Sepanjang tahun 2010–2020, terdapat 76 kasus keracunan jamur liar di Nusantara dengan 550 korban dan 9 kematian [9].

Untuk mengatasi permasalahan ini, teknologi Deep Learning, khususnya Convolutional Neural Network (CNN), muncul sebagai solusi yang lebih unggul dibandingkan metode Machine Learning tradisional karena mampu mengekstrak fitur visual secara otomatis tanpa rekayasa fitur manual [10]. Studi komparasi arsitektur CNN oleh Gehlot dan Saini (2020) membuktikan bahwa DenseNet-121 mampu mencapai akurasi setara model-model berat seperti ResNet dan VGG, namun dengan ukuran model yang jauh lebih kecil dan efisien [11]. Penelitian lain juga menegaskan bahwa DenseNet secara konsisten menawarkan keseimbangan terbaik antara performa tinggi dan efisiensi parameter [12]. Keunggulan ini dimungkinkan oleh mekanisme dense connectivity yang memungkinkan model menggunakan ulang informasi fitur dari lapisan-lapisan sebelumnya secara optimal [13].

Meskipun efisien, melatih DenseNet-121 dari awal menggunakan data terbatas rentan terhadap overfitting. Pendekatan Transfer Learning menjadi solusi efektif dengan memanfaatkan bobot pre-trained dari dataset skala besar seperti ImageNet agar model tidak perlu belajar dari nol [14]. Studi terdahulu membuktikan bahwa Transfer Learning mampu meningkatkan akurasi DenseNet-121 dari 83% menjadi 87% pada dataset terbatas [15]. Penambahan teknik Fine-Tuning pada lapisan akhir jaringan selanjutnya memungkinkan model mengadaptasi fitur generik tersebut menjadi representasi yang lebih spesifik terhadap karakteristik visual jamur [16].

Untuk mengoptimalkan akurasi klasifikasi lebih lanjut, khususnya dalam memisahkan spesies yang memiliki kemiripan visual tinggi, Support Vector Machine (SVM) diintegrasikan sebagai klasifikator akhir menggantikan fungsi Softmax bawaan CNN. SVM secara teoritis sangat tangguh dalam memisahkan data pada ruang berdimensi tinggi melalui prinsip maximum margin hyperplane dan memiliki kemampuan generalisasi yang baik pada dataset terbatas [17]. Metode hibrida yang menempatkan CNN sebagai ekstraktor fitur otomatis dan SVM sebagai klasifikator akhir ini terbukti menghasilkan performa pemisahan kelas yang lebih stabil dan superior dibandingkan penggunaan metode tunggal [18], [19].

Beberapa penelitian terdahulu telah mengeksplorasi penggunaan arsitektur CNN dan metode hibrida untuk klasifikasi citra. Gehlot dan Saini [11] membuktikan DenseNet-121 mencapai akurasi 99,69% pada klasifikasi penyakit daun tomat dengan ukuran model paling ringan dibandingkan ResNet-101, dikonfirmasi pula oleh Ahad et al. [12] bahwa DenseNet konsisten menawarkan keseimbangan terbaik antara akurasi dan efisiensi parameter. Pada domain metode hibrida, Liao dan Zhang [20] menggabungkan DenseNet dengan SVM mencapai 99,12% pada klasifikasi bunga, sementara Minarno et al. [21] membuktikan DenseNet-201 + SVM mencapai 99,65% pada klasifikasi tumor otak MRI, yang lebih tinggi dari DenseNet-201 tanpa SVM (99,51%). Sarkar et al. [22] dan J et al. [23] juga mengonfirmasi bahwa model hibrida DenseNet-SVM secara konsisten menghasilkan akurasi dan kemampuan generalisasi yang lebih baik dibandingkan model tunggal. Mujiono et al. [18] memvalidasi efektivitas hibrida CNN-SVM yang mencapai akurasi 95% pada klasifikasi kesegaran daging ayam.

Pada domain klasifikasi jamur berbasis CNN, Zahan et al. [24] menggunakan InceptionV3, VGG16, dan ResNet50 pada 8.190 citra dari 45 spesies dengan akurasi terbaik 88,40%, sedangkan Ketwongsa et al. [25] mencapai 99,0% menggunakan AlexNet yang ditingkatkan namun pada dataset sangat terbatas (623 gambar, 5 kelas). Sahasranamam et al. [26] membuktikan DenseNet-121 mencapai akurasi 97,16% untuk klasifikasi penyakit daun padi, mengungguli ResNet50 dan VGG-16. Sari dan Elvitaria [27] menggunakan EfficientNet-B4 dengan Transfer Learning dan Fine-Tuning untuk klasifikasi 10 kategori jamur dan mencapai akurasi 90%. Terkait Transfer Learning, Rahman et al. [14] menunjukkan peningkatan akurasi dari 42% menjadi 73% pada dataset terbatas, sementara Bello et al. [15] membuktikan Fine-Tuning DenseNet-121 meningkatkan akurasi dari 83% menjadi 87% pada citra medis, dan Pasha dan Budiman [16] menegaskan bahwa membuka lapisan blok akhir saat Fine-Tuning mengoptimalkan performa klasifikasi domain spesifik.

Penelitian sebelumnya yang menggunakan CNN untuk klasifikasi jamur umumnya hanya menggunakan satu arsitektur tanpa membandingkan skema hibrida secara komparatif, serta jarang mengujikan model pada perangkat bergerak (mobile deployment) [24], [25]. Penelitian ini hadir untuk mengisi celah tersebut dengan tujuan: (1) membandingkan performa empat skema pemodelan yang berbasis DenseNet-121 secara komprehensif, (2) mengevaluasi pengaruh Transfer Learning dan Fine-Tuning terhadap akurasi dan efisiensi komputasi, serta (3) mengimplementasikan model terbaik pada aplikasi Android dan menguji performanya pada perangkat dengan sumber daya komputasi terbatas.

Kontribusi utama penelitian ini adalah rancangan dan evaluasi sistematis arsitektur hibrida DenseNet-121 + SVM dengan strategi Transfer Learning dan Fine-Tuning untuk klasifikasi 12 spesies jamur, beserta implementasinya sebagai aplikasi offline berbasis Android menggunakan TensorFlow Lite. Model yang dihasilkan diharapkan dapat berfungsi sebagai alat bantu edukasi yang cepat, akurat, dan praktis untuk pencegahan keracunan jamur liar di masyarakat.

p-ISSN: 2301-5373
e-ISSN: 2654-5101

2. Metode Penelitian

Contains Penelitian ini menggunakan pendekatan eksperimental komparatif dengan tahapan yang mencakup pengumpulan data, prapemrosesan citra, pembangunan empat skema model, pengujian hyperparameter, evaluasi performa, dan implementasi pada perangkat Android. Alur metodologi secara keseluruhan dimulai dari pengambilan data citra jamur melalui web scraping, dilanjutkan dengan preprocessing, pelatihan model di lingkungan berbasis Python menggunakan platform Kaggle, hingga konversi dan pengujian akhir pada perangkat mobile.

2.1 Dataset dan Pengumpulan Data

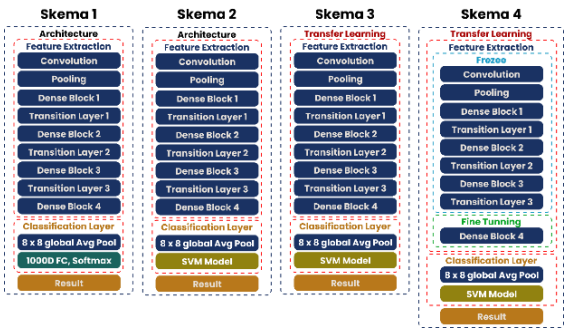
Data yang digunakan merupakan data sekunder berupa citra digital yang diperoleh dari platform iNaturalist.org menggunakan teknik web scraping berbasis Application Programming Interface (API). iNaturalist merupakan komunitas global yang menyediakan dokumentasi dan identifikasi berbagai spesies flora dan fauna, termasuk jamur. Dataset mencakup 12 spesies jamur, terdiri dari enam spesies beracun (Amanita muscaria, Amanita pantherina, Coprinopsis atramentaria, Hypholoma fasciculare, Lactarius torminosus, Paxillus involutus) dan enam spesies aman konsumsi (Boletus edulis, Cerioporus squamosus, Coprinus comatus, Lactarius deliciosus, Laetiporus sulphureus, Macrolepiota procera). Proses scraping mengunduh citra beresolusi penuh (format original) dengan mekanisme retry otomatis dan exponential backoff untuk menjaga stabilitas koneksi.

2.2 Prapemrosesan Data

Prapemrosesan dilakukan dalam dua tahapan utama. Pertama, standarisasi dimensi citra melalui pendekatan bertingkat: setiap gambar diubah ukurannya menjadi 300×300 piksel, kemudian dilakukan central cropping dengan fraksi 0,85 untuk membuang noise latar belakang, dan diubah kembali ke dimensi target 256×256 piksel. Kedua, dilakukan kurasi manual (visual inspection) untuk menyaring citra yang tidak memuat objek jamur, memiliki skala objek terlalu kecil, atau terdapat obstruksi visual (blur, noise).

2.3 Desain Skema Model

Penelitian ini merancang empat skema pemodelan komparatif yang semuanya berbasis arsitektur DenseNet-121 [13]. Arsitektur ini dipilih karena mekanisme dense connectivity-nya memungkinkan setiap lapisan menerima input dari semua lapisan sebelumnya, sehingga informasi fitur digunakan secara maksimal tanpa redundansi. DenseNet-121 terdiri dari empat Dense Block (6, 12, 24, 16 lapisan konvolusi) yang diselingi Transition Layer, diawali konvolusi 7×7 dan pooling 3×3, serta diakhiri Global Average Pooling yang menghasilkan vektor fitur 1D berdimensi 1024. Perbandingan keempat skema diilustrasikan secara ringkas pada Gambar 1.



Gambar 1. Perbedaan desain pada keempat skema pemodelan

Pada skema pertama, seluruh bobot DenseNet-121 diinisialisasi secara acak dan dilatih sepenuhnya dari awal (training from scratch) menggunakan dataset jamur. Keluaran Global Average Pooling (vektor 1×1024) diteruskan langsung ke Fully Connected Layer bawaan dengan fungsi aktivasi Softmax untuk menghasilkan probabilitas 12 kelas. Skema ini berfungsi sebagai baseline untuk mengukur peningkatan performa pada skema-skema berikutnya.

Pada skema kedua, arsitektur identik dengan Skema 1 namun lapisan Fully Connected dan Softmax digantikan oleh algoritma Support Vector Machine (SVM). Pemilihan SVM dibandingkan Random

Forest, XGBoost, maupun Multi-Layer Perceptron (MLP) didasarkan pada keunggulan prinsip maximum margin hyperplane SVM pada ruang fitur berdimensi tinggi (1024), serta ketahanannya terhadap risiko overfitting pada ukuran dataset moderat dibandingkan MLP yang membutuhkan parameter terlatih jauh lebih banyak [28]. Vektor fitur 1×1024 dari Global Average Pooling diumpankan langsung ke SVM dengan kernel Radial Basis Function (RBF), dipilih karena kemampuannya memetakan fitur non-linier berdimensi tinggi ke ruang yang lebih mudah dipisahkan dibandingkan kernel linear, yang kurang memadai untuk memisahkan kelas dengan kemiripan morfologi visual tinggi [29]. Parameter SVM ditetapkan C=1,0 dan gamma='scale' (nilai default scikit-learn) mengingat keterbatasan alokasi GPU untuk eksplorasi grid search penuh terhadap kedua parameter tersebut. Skema ini bertujuan mengisolasi pengaruh penggantian metode klasifikasi akhir terhadap akurasi, tanpa variabel Transfer Learning.

Pada skema ketiga, DenseNet-121 menggunakan bobot pre-trained ImageNet dengan seluruh lapisan konvolusi dibekukan sepenuhnya. Model berfungsi murni sebagai ekstraktor fitur statis. Vektor fitur yang dihasilkan kemudian diklasifikasikan oleh SVM. Skema ini dirancang untuk mengevaluasi efektivitas fitur generik ImageNet secara langsung tanpa proses adaptasi, sekaligus mengukur besaran domain gap antara ImageNet dan dataset jamur.

Skema keempat merupakan arsitektur paling komprehensif. DenseNet-121 diinisialisasi dengan bobot ImageNet, kemudian diterapkan partial fine-tuning: seluruh lapisan awal (Initial Conv, Dense Block 1–3, dan Transition Layer 1–3) dibekukan, sementara Dense Block 4 dibuka (unfreeze) untuk dilatih ulang. Pemilihan Dense Block 4 didasarkan pada prinsip hierarki fitur CNN, di mana lapisan akhir bertugas mengekstrak fitur semantik tingkat tinggi yang spesifik terhadap objek dataset [16], [30]. Lapisan BatchNormalization tetap dibekukan untuk menjaga stabilitas statistik pelatihan. Arsitektur dilengkapi dua lapisan Dense (512 dan 128 neuron) dengan Dropout 0,5. Setelah fine-tuning selesai, model dialihfungsikan sebagai ekstraktor fitur domain-specific, dan vektor 1×1024 dari Global Average Pooling diklasifikasikan oleh SVM kernel RBF.

2.4 Skenario Pengujian Hyperparameter

Untuk mencari konfigurasi optimal, setiap skema diuji dengan enam kombinasi hyperparameter yang terdiri dari dua variasi optimizer (Adam dan RMSProp) dikombinasikan dengan tiga variasi batch size (4, 8, dan 16). Learning rate ditetapkan konstan di 1×10−4 dan jumlah epoch maksimal 16 iterasi. Penetapan nilai-nilai ini didasarkan pada pertimbangan berikut: (1) Adam dipilih karena mengintegrasikan estimasi momen pertama dan kedua yang memberikan stabilitas konvergensi tinggi [31]; (2) RMSProp diuji sebagai pembanding karena kemampuannya menyesuaikan learning rate secara dinamis [31]; (3) batch size kecil (4) terbukti memberikan efek regularisasi implisit yang membantu model keluar dari local minima [32]; (4) learning rate rendah mencegah degradasi bobot pre-trained saat fine-tuning. Mekanisme ModelCheckpoint diterapkan untuk menyimpan bobot terbaik secara otomatis. Rincian skenario disajikan pada Tabel 1.

Tabel 1. Skenario pengujian hyperparameter

Skenario	Optimizer	Batch Size	Learning Rate	Epoch
1	Adam	4	0,0001	16
2	Adam	8	0,0001	16
3	Adam	16	0,0001	16
4	RMSProp	4	0,0001	16
5	RMSProp	8	0,0001	16
6	RMSProp	16	0,0001	16

Evaluasi pada Skema 2, 3, dan 4 menggunakan Stratified 5-Fold Cross Validation (K=5) untuk memastikan setiap fold mempertahankan distribusi kelas yang proporsional. Rata-rata akurasi dari lima fold digunakan sebagai metrik evaluasi final untuk menjamin ketangguhan (robustness) dan kemampuan generalisasi model.

p-ISSN: 2301-5373
e-ISSN: 2654-5101

2.5 Evaluasi Model

Evaluasi dilakukan dalam dua dimensi. Pertama, evaluasi performa klasifikasi menggunakan Confusion Matrix yang menghasilkan empat metrik utama: Accuracy, Precision, Recall, dan F1-Score (Persamaan 1–4). Kedua, evaluasi efisiensi komputasi yang mengukur waktu pelatihan, waktu inferensi total, dan ukuran model (MB) di lingkungan Python (Kaggle GPU).

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (1)$$

$$\text{Precision} = \frac{TP}{(TP+FP)} \quad (2)$$

$$\text{Recall} = \frac{TP}{(TP+FN)} \quad (3)$$

$$F1 - Score = 2 \times \frac{(Precision \times Recall)}{(Precision+Recall)} \quad (4)$$

2.6 Implementasi pada Perangkat Mobile

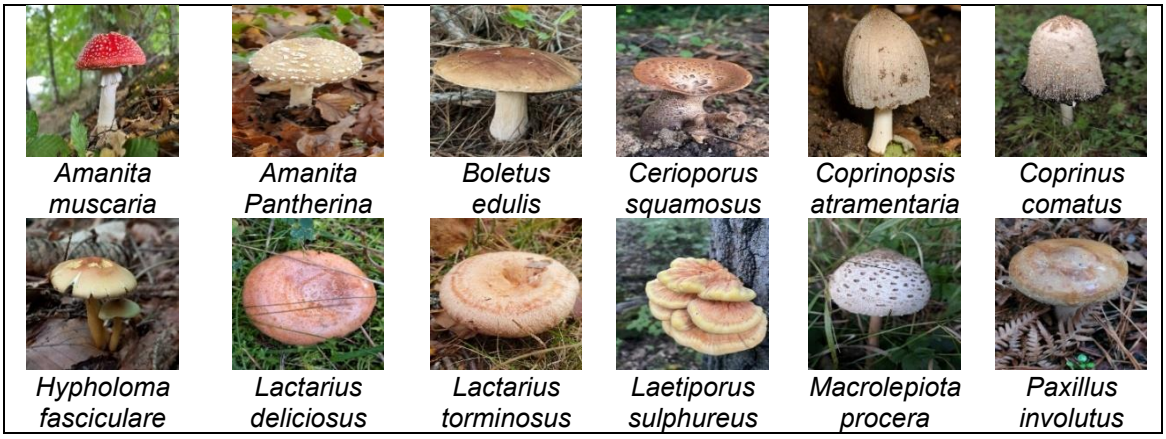
Model terbaik hasil eksperimen dikonversi ke format TensorFlow Lite (TFLite) untuk implementasi Android. Proses konversi dibagi menjadi dua komponen: (1) ekstraktor fitur DenseNet-121 dikonversi ke format .tflite dengan teknik weight quantization untuk mereduksi ukuran model; (2) parameter SVM (koefisien dan bias) serta normalisasi StandardScaler diekspor ke format JSON agar dapat direkonstruksi secara native menggunakan Kotlin. Aplikasi Android dibangun menggunakan bahasa Kotlin dan XML, dengan antarmuka yang mencakup fitur klasifikasi kamera/galeri, ensiklopedia jamur, dan halaman bantuan. Evaluasi akhir dilakukan menggunakan 180 citra baru (unseen data) yang belum pernah dilihat model (15 citra per spesies) pada perangkat Redmi Note 13 (Snapdragon 685, 8 GB RAM).

3. Hasil dan Pembahasan

This Bagian ini menyajikan seluruh hasil penelitian secara sistematis, dimulai dari hasil pengumpulan data dan prapemrosesan citra, dilanjutkan dengan komparasi performa keempat skema pemodelan, evaluasi efisiensi komputasi, implementasi model terbaik pada perangkat Android, hingga pembahasan komparatif dengan penelitian terdahulu.

3.1 Hasil Pengumpulan Data

Proses Pengumpulan data dilakukan melalui web scraping berbasis API terhadap iNaturalist.org, menghasilkan 39.276 citra mentah yang terdistribusi merata ke dalam 12 kelas spesies (3.273 citra per kelas) dengan total ukuran 71,9 GB. Distribusi yang seimbang ini penting untuk mencegah bias model. Namun, data mentah belum dapat langsung digunakan karena masih mengandung variasi dimensi dan citra tidak relevan, sehingga diperlukan prapemrosesan. Contoh visual sampel setiap kelas spesies ditunjukkan pada Gambar 2.

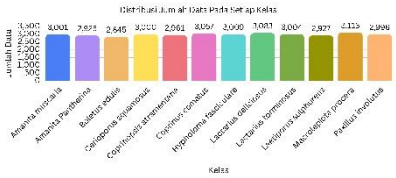


Gambar 2. Sample gambar jamur per kelas spesies

3.2 Hasil Prapemrosesan Data

Setiap citra distandarisasi melalui tiga langkah, yaitu resize ke 300×300 piksel dan kemudian dilakukan central cropping (fraksi 0,85) dan terakhir dilakukan resize kembali ke ukuran final yaitu 256×256 piksel. Proses ini mereduksi ukuran dataset secara signifikan dari 71,9 GB menjadi 1,81 GB. Selanjutnya, kurasi manual menyaring tiga kategori citra tidak layak, yaitu citra tidak relevan, skala objek terlalu kecil, dan citra dengan blur/obstruksi tinggi.

Setelah seluruh rangkaian kurasi manual selesai, diperoleh dataset final berjumlah 35.921 citra, berkurang 3.355 file (8,5%) dari total awal 39.276 citra. Pengurangan ini bukan merupakan kerugian, melainkan peningkatan kualitas data yang presisi. Distribusi jumlah citra valid per kelas pasca-kurasi divisualisasikan pada Gambar 3.



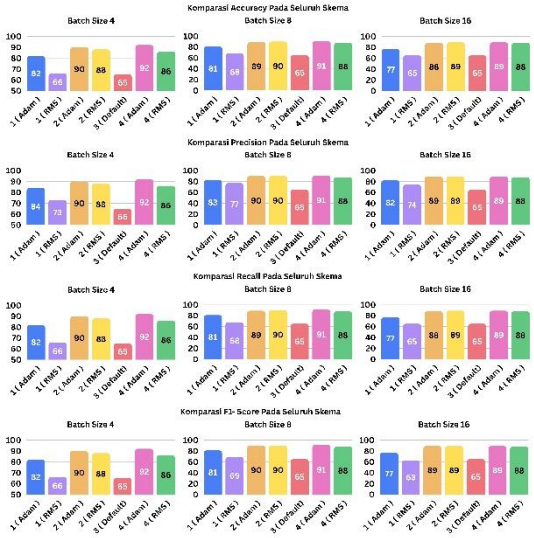
Gambar 3. Distribusi jumlah citra final per kelas setelah kurasi manual

Distribusi antar kelas tetap relatif seimbang pasca-kurasi, menunjukkan bahwa tidak ada kelas yang kehilangan citra secara tidak proporsional. Keseimbangan distribusi ini penting untuk memastikan model tidak bersifat bias dalam proses pelatihan. Total keseluruhan dataset kemudian didistribusikan ke dalam tiga bagian utama, yaitu data latih sebesar 80%, serta data validasi dan data uji yang masing-masing sebesar 10%.

Secara rinci, subset latih terdiri dari 28.736 citra yang difungsikan untuk pelatihan arsitektur CNN dengan penambahan proses augmentasi data, dan ekstraksi fitur SVM tanpa penambahan proses augmentasi data. Sementara itu, subset validasi sebanyak 3.592 citra digunakan untuk memantau konvergensi model selama proses pelatihan berlangsung. Sisanya, yakni subset uji sebanyak 3.593 citra, dialokasikan secara khusus untuk mengevaluasi performa akhir model.

3.3 Hasil Pelatihan Keempat Skema

Keempat skema pemodelan diuji dengan enam kombinasi hyperparameter (optimizer Adam dan RMSProp, batch size 4, 8, dan 16) untuk memperoleh konfigurasi terbaik pada masing-masing skema. Gambar 4 menyajikan komparasi visual Accuracy, Precision, Recall, dan F1-Score dari seluruh skema pada ketiga variasi batch size tersebut.



Gambar 4. Komparasi Accuracy, Precision, Recall, dan F1-Score pada Seluruh Skema dan Variasi Batch Size

Berdasarkan Gambar 4, konfigurasi Adam dengan Batch Size 4 secara konsisten unggul di seluruh skema dan menjadi acuan pembahasan berikut.

Pada Skema 1, DenseNet-121 dilatih sepenuhnya dari awal (training from scratch) tanpa bantuan bobot pre-trained, dengan Softmax sebagai klasifikator akhir. Konfigurasi Adam Batch Size 4 mencatatkan

p-ISSN: 2301-5373
e-ISSN: 2654-5101

akurasi 82%, menjadikannya baseline penelitian. Performa ini relatif terbatas karena arsitektur harus mempelajari seluruh representasi fitur visual jamur dari nol hanya berdasarkan data latih yang tersedia, tanpa pengetahuan visual eksternal.

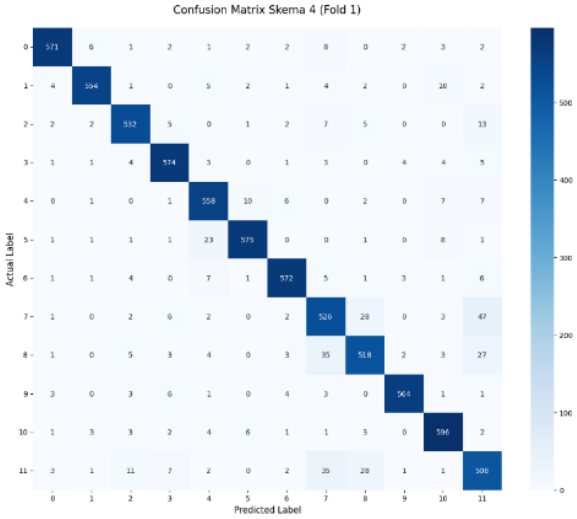
Pada Skema 2, arsitektur dasar tetap identik dengan Skema 1, namun lapisan Softmax digantikan oleh Support Vector Machine (SVM) sebagai klasifikator akhir. Penggantian ini menghasilkan peningkatan akurasi signifikan menjadi 90% pada konfigurasi Adam Batch Size 4, atau naik 8 poin dari Skema 1. Peningkatan ini menunjukkan bahwa SVM dengan prinsip maximum margin hyperplane lebih unggul dibandingkan Softmax dalam menarik batas keputusan pada ruang fitur berdimensi tinggi (1024), terlepas dari ada atau tidaknya Transfer Learning.

Pada Skema 3, DenseNet-121 menggunakan bobot pre-trained ImageNet dengan seluruh lapisan konvolusi dibekukan sepenuhnya (frozen), sehingga model hanya berfungsi sebagai ekstraktor fitur statis tanpa proses adaptasi domain. Hasilnya justru menjadi yang terendah di antara seluruh skema, yaitu hanya 65% pada konfigurasi terbaiknya. Penurunan tajam ini mengindikasikan adanya domain gap yang besar antara fitur generik objek umum pada ImageNet dengan karakteristik tekstur dan morfologi spesifik jamur, sehingga fitur yang diekstraksi tanpa adaptasi tidak cukup representatif untuk membedakan ke-12 kelas spesies.

Pada Skema 4, DenseNet-121 diinisialisasi dengan bobot ImageNet kemudian diterapkan partial fine-tuning pada Dense Block 4, dikombinasikan dengan SVM sebagai klasifikator akhir. Skema ini mencapai performa tertinggi di antara seluruh skema, yaitu 92% pada konfigurasi Adam Batch Size 4, meningkat 27 poin dari Skema 3 dan 2 poin dari Skema 2. Hasil ini membuktikan bahwa kombinasi Transfer Learning, Fine-Tuning pada lapisan semantik tingkat tinggi, dan SVM sebagai klasifikator akhir saling melengkapi: bobot pre-trained mempercepat konvergensi, fine-tuning mengadaptasi fitur generik menjadi spesifik domain jamur, dan SVM menghasilkan batas keputusan yang lebih tegas antar kelas.

Secara konsisten pada keempat skema, optimizer Adam dengan Batch Size 4 selalu mengungguli RMSProp maupun batch size yang lebih besar, sebagaimana terlihat pada seluruh panel Gambar 4. Hal ini menunjukkan bahwa mekanisme adaptive moment estimation pada Adam, dikombinasikan dengan efek regularisasi implisit dari batch kecil, secara konsisten lebih efektif menavigasi loss landscape pada domain klasifikasi jamur dibandingkan kombinasi hyperparameter lainnya.

Skema 4 dengan konfigurasi Adam Batch Size 4 ditetapkan sebagai skema terbaik penelitian ini. Detail performa per kelas serta Confusion Matrix dari skema tersebut disajikan pada bagian Gambar 5 dan Tabel 2.



Gambar 5. Confusion Matrix Skema 4 (Adam, Batch Size 4)

Tabel 2. Laporan Klasifikasi Skema 4 (Adam, Batch Size 4)

Kelas Jamur	Precision	Recall	F1-Score
Amanita muscaria	0,97	0,95	0,96

Amanita pantherina	0,97	0,95	0,96
Boletus edulis	0,94	0,93	0,94
Cerioporus squamosus	0,95	0,96	0,95
Coprinopsis atramentaria	0,91	0,94	0,93
Coprinus comatus	0,96	0,94	0,95
Hypholoma fasciculare	0,96	0,95	0,95
Lactarius deliciosus	0,84	0,85	0,85
Lactarius torminosus	0,88	0,86	0,87
Laetiporus sulphureus	0,98	0,96	0,97
Macrolepiota procera	0,94	0,96	0,95
Paxillus involutus	0,82	0,85	0,83
Rata-rata / Akurasi	0,92	0,92	0,92

3.4 Evaluasi Komparatif Seluruh Skema

Tabel 3 menyajikan perbandingan efisiensi komputasi keempat skema pada konfigurasi terbaik masing-masing.

Tabel 3. Komparasi Efisiensi Komputasi Seluruh Skema (Konfigurasi Terbaik)

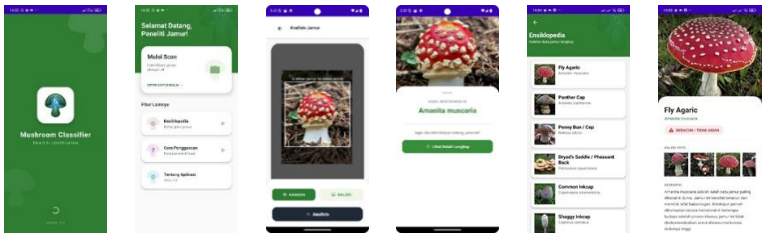
Skema	Konfigurasi	Waktu Latih (detik)	Waktu Uji (detik)	Ukuran Model (MB)
Skema 1	Adam, Batch 4	8.844	29	139,8
Skema 2	Adam, Batch 4	9.768	70	249,06
Skema 3	ImageNet Murni	2.099	88	219,75
Skema 4	Adam, Batch 4	8.590	73	146,18

Skema 3 mencatat waktu pelatihan tercepat (2.099 detik) karena tidak ada proses backpropagation. Skema 4 menyelesaikan pelatihan dalam 8.590 detik lebih cepat dari Skema 2 (9.768 detik) membuktikan bobot pre-trained membantu model konvergen lebih awal. Waktu inferensi 73 detik untuk 3.593 citra uji setara ~0,02 detik per gambar, tergolong near real-time. Ukuran model 146,18 MB relatif ringan untuk arsitektur deep learning kompleks dan masih dapat ditekan lebih lanjut melalui konversi TensorFlow Lite.

3.5 Implementasi pada Perangkat Android

Model Skema 4 (146,18 MB) dikonversi ke TensorFlow Lite melalui weight quantization (float32 ke int8) dan ekstraksi parameter SVM ke format JSON, menghasilkan penyusutan ukuran 94,93% menjadi hanya 7,41 MB.

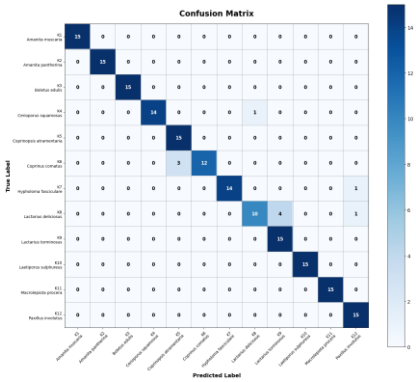
Kemudian aplikasi android dibangun menggunakan Kotlin dan XML dengan enam halaman utama, yaitu Splash Screen, Halaman Utama, Menu Klasifikasi, Hasil Klasifikasi, Ensiklopedia Jamur, dan Bantuan/Tentang Aplikasi. Tampilan antarmuka aplikasi diilustrasikan pada Gambar 6.



p-ISSN: 2301-5373
e-ISSN: 2654-5101

Gambar 6. Dari kiri ke kanan tampilan Splash Screen, Halaman Utama aplikasi, Tampilan Menu Klasifikasi, Halaman Hasil Klasifikasi, Tampilan Daftar Ensiklopedia, Detail Informasi Jamur.

Kemudian pengujian akhir menggunakan 180 citra baru (unseen data, 15 citra per spesies) pada perangkat Redmi Note 13 (Snapdragon 685, 8 GB RAM). Confusion Matrix pengujian Android disajikan pada Gambar 7.



Gambar 7. Confusion Matrix hasil klasifikasi pada aplikasi Android (180 citra, 15 per spesies)

Terdapat 10 kesalahan prediksi dari total 180 citra, menghasilkan 170 prediksi benar. Kesalahan mengelompok dalam dua pola: (1) kebingungan intra-genus pada Lactarius 4 citra K8 (Lactarius deliciosus) teridentifikasi sebagai K9 (Lactarius torminosus) akibat kemiripan morfologi antar-spesies satu genus; (2) kebingungan warna-bentuk pada Coprinoid 3 citra K6 (Coprinus comatus) teridentifikasi sebagai K5 (Coprinopsis atramentaria) karena kesamaan palet warna dan bentuk tudung jamur. Rekapitulasi metrik per kelas disajikan pada Tabel 4.

Tabel 4. Rekapitulasi metrik performa klasifikasi pada perangkat Android

Kelas Jamur	Precision	Recall	F1-Score
Amanita muscaria	1,00	1,00	1,00
Amanita pantherina	1,00	1,00	1,00
Boletus edulis	1,00	1,00	1,00
Cerioporus squamosus	1,00	0,93	0,97
Coprinopsis atramentaria	0,83	1,00	0,91
Coprinus comatus	1,00	0,80	0,89
Hypholoma fasciculare	1,00	0,93	0,97
Lactarius deliciosus	0,91	0,67	0,77
Lactarius torminosus	0,79	1,00	0,88
Laetiporus sulphureus	1,00	1,00	1,00
Macrolepiota procera	1,00	1,00	1,00
Paxillus involutus	0,88	1,00	0,94
Rata-rata	**0,95**	**0,94**	**0,94**
Akurasi Global			**94,44%**

Akurasi global 94,44% melampaui rata-rata cross-validation 92,37%, mengindikasikan model tidak mengalami overfitting dan adaptif terhadap variasi baru. Lima spesies mencapai Precision dan Recall sempurna 1,00: Amanita muscaria, Amanita pantherina, Boletus edulis, Laetiporus sulphureus, dan Macrolepiota procera. Performa terendah pada Lactarius deliciosus (F1-Score 0,77) disebabkan oleh

kombinasi kemiripan morfologi intra-genus dan precision loss akibat rekonstruksi SVM menggunakan tipe data presisi lebih rendah di lingkungan Kotlin.

Tabel 5. Hasil pengukuran efisiensi komputasi pada perangkat Android (Redmi Note 13)

Parameter Evaluasi	Deskripsi	Hasil
Model Load Time	Inisialisasi seluruh model ke RAM saat aplikasi pertama dibuka	0,77 detik
CNN Feature Extraction	Ekstraksi piksel → vektor fitur 1×1024 oleh DenseNet-121	0,48 detik
Image Processing & SVM	Prapemrosesan + normalisasi StandardScaler + klasifikasi SVM	0,02 detik
Total Inference Time	Total waktu komputasi murni per citra	0,50 detik
Ukuran Model (AI)	Total memori seluruh komponen AI pada perangkat	8,47 MB
Ukuran Aplikasi (APK)	Total ukuran instalasi aplikasi	94,70 MB

Tiga temuan kunci: (1) Cold Start Load Time 0,77 detik memvalidasi keberhasilan weight quantization dalam menekan beban I/O; (2) total inferensi 0,50 detik per citra tergolong near real-time dan layak untuk aplikasi mobile praktis; (3) bottleneck pemrosesan terpusat pada CNN Feature Extraction (0,48 detik dari total 0,50 detik), sementara klasifikasi SVM hanya 0,02 detik membuktikan strategi ekstraksi koefisien SVM ke JSON berhasil mengeksekusi decision boundary tanpa beban tambahan pada CPU smartphone.

4. Kesimpulan dan Saran

Penelitian ini mengembangkan dan mengevaluasi sistem klasifikasi hibrida DenseNet-121 + SVM untuk 12 spesies jamur melalui empat skema pemodelan komparatif. Tiga simpulan utama dapat ditarik.

Pertama, arsitektur hibrida CNN-SVM secara konsisten mengungguli klasifikasi berbasis Softmax, penggantian Softmax dengan SVM saja menghasilkan peningkatan akurasi 7,97 poin, sementara pipeline Skema 4 lengkap (Transfer Learning + Fine-Tuning + SVM) mencapai rata-rata akurasi tertinggi 92,37% yang meningkat 10,0 poin dari baseline.

Kedua, Transfer Learning tanpa adaptasi domain (Skema 3) menurunkan performa secara signifikan akibat domain gap, sementara Fine-Tuning parsial pada Dense Block 4 memulihkan dan secara substansial melampaui baseline, membuktikan kalibrasi lapisan fitur semantik akhir sebagai intervensi paling krusial.

Ketiga, model terbukti dapat diimplementasikan pada perangkat Android terbatas: weight quantization dan rekonstruksi SVM berbasis JSON mengkompresi model dari 146,18 MB menjadi 7,41 MB berkurang sekitar 94,93%, dan pengujian di perangkat mencapai akurasi 94,44% pada 180 citra baru dengan total waktu inferensi 0,50 detik per citra.

Penelitian ini memiliki beberapa keterbatasan: dataset terbatas pada 12 spesies dari platform iNaturalist dan belum merepresentasikan keanekaragaman jamur liar regional di Indonesia; kualitas gambar bergantung pada unggahan pengguna; dan model belum diuji pada kondisi pencahayaan dan sudut pengambilan gambar yang sangat bervariasi di lapangan. Penelitian ke depan disarankan untuk: (1) memperluas cakupan spesies, terutama spesies endemik Indonesia; (2) mengintegrasikan teknik explainability seperti Grad-CAM untuk memvisualisasikan basis keputusan model; (3) menambahkan konteks toksisitas berbasis lokasi GPS; dan (4) mereduksi latensi inferensi melalui model distillation untuk deployment pada perangkat dengan sumber daya sangat terbatas.

References

[1] A. M. Cahaya, R. Pratama, and M. Herlina, “Keanekaragaman jamur makroskopis di taman hutan raya rajo lelo bengkulu,” *J. Ris. DAN Inov. Pendidik. SAINS*, vol. 4, no. 2, pp. 93–99, 2025.

[2] R. Nurfitri, J. Hardini, and I. K. Muskin, “IDENTIFIKASI JAMUR ENDOFIT YANG BERASOSIASI DENGAN DAUN JERUK LIMAU Citrus x amblycarpa (Hassk.) OCHSE DI KEBUN KALPATARU, DENPASAR, BALI,” *SIMBIOSIS XII*, no. 1, pp. 100–107, 2024.

[3] S. W. Ulfa, N. H. Siagian, P. K. Putri, R. A. Harahap, and A. Sari, “Identifikasi Produk Bahan Makanan Yang Berbahan Dasar Fungi Makroskopis Pada Pasar Tradisional Dan Modern Yang Ada Di Kota Medan,” *J. Soc. Sci. Res. Vol.*, vol. 3, pp. 11487–11502, 2023.

[4] M. R. Al Aziz, M. T. Furqon, and L. Muflikhah, “Klasifikasi Jamur Dapat Dimakan atau Beracun Menggunakan Naïve Bayes dan Seleksi Fitur berbasis Association Rule Mining,” *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 6, no. 8, pp. 3948–3955, 2022.

[5] J. E. Putri, R. Lingga, and H. Helmi, “Keanekaragaman Jenis dan Pemanfaatan Jamur Makroskopis di Taman Hutan Raya Bukit Mangkol Desa Teru, Provinsi Kepulauan Bangka Belitung,” *J. BIOS LOGOS*, vol. 14, no. 3, pp. 50–63, 2024.

[6] H. Li *et al.*, “Mushroom Poisoning Outbreaks — China , 2024,” *Chinese Cent. Dis. Control Prev.*, vol. 7, no. 19, 2024.

[7] P. A. N. K. Permatananda, I. G. S. Pandit, and I. P. Putra, “ULASAN : BEBERAPA JAMUR LIAR YANG BERPOTENSI MENYEBABKAN KERACUNAN DI INDONESIA,” *J. Biol.*, vol. 18, no. 1, pp. 42–56, 2025.

[8] M. P. Amelya, I. P. Putra, R. Hermawan, and R. Nurzakiah, “Peluang Pemanfaatan dan Resiko Keracunan Jamur Coprinoid pada Jerami Padi di Indonesia,” *Quagga J. Pendidik. dan Biol.*, vol. 15, pp. 1–8, 2023, doi: 10.25134/quagga.v15i1.5180.Received.

[9] I. P. Putra, “Kasus-Kasus Keracunan Jamur Liar Di Indonesia,” *J. Ekol. Kesehat.*, vol. 20, no. 3, pp. 215–230, 2022, doi: 10.22435/jek.v20i3.4943.

[10] P. A. Prayesy, “STUDI PERBANDINGAN METODE SUPPORT VECTOR MACHINE, RANDOM FOREST, DAN CONVOLUTIONAL NEURAL NETWORK UNTUK KLASIFIKASI PENYAKIT KULIT,” *J. Kecerdasan Buatan dan Teknol. Inf.*, vol. 4, no. 1, pp. 70–76, 2025.

[11] M. Gehlot and M. L. Saini, “Analysis of Different CNN Architectures for Tomato Leaf Disease Classification,” *2020 5th IEEE Int. Conf. Recent Adv. Innov. Eng. ICRAIE 2020 - Proceeding*, vol. 2020, 2020, doi: 10.1109/ICRAIE51050.2020.9358279.

[12] M. T. Ahad, Y. Li, B. Song, and T. Bhuiyan, “Comparison of CNN-based deep learning architectures for rice diseases classificatio,” *Artif. Intell. Agric.*, vol. 9, pp. 22–35, 2023, doi: 10.1016/j.aiia.2023.07.001.

[13] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 2261–2269, 2017, doi: 10.1109/CVPR.2017.243.

[14] A. T. Rahman, A. Setyanto, and H. Al Fatta, “Klasifikasi Penyakit Daun Apel Mengunakan Arsitektur CNN Dengan Transfer Learning,” *J. SENOPATI*, vol. 6, pp. 42–49, 2024.

[15] A. Bello, S. C. Ng, and M. F. Leung, “Skin Cancer Classification Using Fine-Tuned Transfer Learning of DENSENET-121,” *Appl. Sci.*, vol. 14, no. 17, 2024, doi: 10.3390/app14177707.

[16] R. K. Pasha and K. Budiman, “Optimasi Model CNN Berbasis Transfer Learning Untuk Klasifikasi Pneumonia pada Citra X-Ray Dada,” *SMATIKA STIKI Inform. J.*, vol. 15, no. 1, pp. 167–178, 2025.

[17] M. R. P. Soewarno, R. Ardianto, R. Suryani, and C. I. Al-Hakim, Rosyid Ridlo Wahyudi, “Analisis Perbandingan Algoritma KNN dan SVM untuk Prediksi Risiko Kesehatan Ibu Hamil,” *J. Kolaborasi Ris. Sarj.*, vol. 2, pp. 1–8, 2025.

[18] A. A. Mujiono, Kartini, and E. Y. Puspaningrum, “IMPLEMENTASI MODEL HYBRID CNN-SVM PADA KLASIFIKASI KONDISI KESEGARAN DAGING AYAM,” *JATI (Jurnal Mhs. Tek. Inform.*, vol. 8, no. 1, pp. 756–763, 2024, doi: 10.36040/jati.v8i1.8855.

[19] Y. U. Ladopurab and Y. R. Kaesmetan, “ANALISIS PERBANDINGAN CONVOLUTIONAL NEURAL NETWORK DAN SUPPORT VECTOR MACHINE PADA ANALISIS SENTIMEN RUU PILKADA DI APLIKASI X Comparative Analysis of Convolutional Neural Network and Support Vector Machine in Sentiment Analysis of the Regional Election B,” vol. 9, no. 2, pp.

908–927, 2025, doi: 10.52362/jisamar.v9i2.1887.

[20] L. Liao and S. Zhang, “A Flower Classification Method Combining DenseNet Architecture with SVM,” *Proc. - 2020 16th Int. Conf. Comput. Intell. Secur. CIS 2020*, pp. 25–29, 2020, doi: 10.1109/CIS52066.2020.00014.

[21] A. E. Minarno, I. S. Kantomo, F. D. S. Sumadi, H. A. Nugroho, and Z. Ibrahim, “Classification of Brain Tumors on MRI Images Using DenseNet and Support Vector Machine,” *Int. J. Informatics Vis.*, vol. 6, no. 2, pp. 404–410, 2022, doi: 10.30630/joiv.6.2.991.

[22] S. Sarkar, J. A. Ray, C. Mukherjee, S. Ghosh, N. Jayanthi, and C. L. K. R, “Plant Leaf Disease Classification Based on SVM Based Densenets,” *2023 Int. Conf. Adv. Comput. Commun. Inf. Technol.*, pp. 636–641, 2023, doi: 10.1109/ICAICCIT60255.2023.10465886.

[23] R. R. J, S. Barola, and V. Narawade, “Enhancing Image Classification Accuracy With A Lightweight Hybrid Densenet And Machine Learning Model,” *Nanotechnol. Perceptions*, vol. 11, pp. 1141–1155, 2024.

[24] N. Zahan, M. Z. Hasan, M. A. Malek, and S. S. Reya, “A Deep Learning-Based Approach for Edible, Inedible and Poisonous Mushroom Classification,” *2021 Int. Conf. Inf. Commun. Technol. Sustain. Dev. ICICT4SD 2021 - Proc.*, pp. 440–444, 2021, doi: 10.1109/ICICT4SD50815.2021.9396845.

[25] W. Ketwongsa, S. Boonlue, and U. Kokaew, “A New Deep Learning Model for The Classification of Poisonous and Edible Mushrooms Based on Improved AlexNet Convolutional Neural Network,” *Appl. Sci.*, vol. 12, no. 7, 2022, doi: 10.3390/app12073409.

[26] V.Sahasranamam, T. Ramesh, R. Rajeswari, A. Karthikkumar, and D. Muthumanickam, “AI-Driven Paddy Leaf Disease Classification and Prediction using DenseNet-121,” *Adv. Intell. Syst. Res.*, vol. 193, no. Icdlair 2024, pp. 272–292, 2025, doi: 10.2991/978-94-6463-740-3.

[27] I. P. Sari and L. Elvitaria, “Classification of mushroom types based on digital image processing using convolutional neural network,” vol. 13, no. 4, pp. 379–388, 2025.

[28] S. A. Ngorima, A. Helberg, and M. H. Davel, “Feature extraction for plant growth estimation,” *Commun. Comput. Inf. Sci.*, pp. 1–13, 2026, doi: 10.1007/978-3-032-11733-5_5.

[29] A. H. Baradan, “Predictive Maintenance of Electric Motors Using Supervised Learning Models: A Comparative Analysis,” *Predict. Maint. Electr. Mot. Using Supervised Learn. Model. A Comp. Anal.*, p. 8, 2025.

[30] S. Saluja, M. C. Trivedi, and S. S. Sarangdevot, “Advancing glioma diagnosis : Integrating custom U-Net and VGG-16 for improved grading in MR imaging,” vol. 21, no. 3, pp. 4328–4350, 2024, doi: 10.3934/mbe.2024191.

[31] D. Kurniadi, R. M. Shidiq, and A. Mulyani, “Perbandingan Penggunaan Optimizer dalam Klasifikasi Sel Darah Putih Menggunakan Convolutional Neural Network,” *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 14, pp. 77–86, 2025, doi: 10.22146/jnteti.v14i1.17162.

[32] R. Lin, “Analysis on the Selection of the Appropriate Batch Size in CNN Neural Network,” *Int. Conf. Mach. Learn. Knowl. Eng.*, pp. 106–109, 2022, doi: 10.1109/MLKE55170.2022.00026.