OPTIMISASI MODEL *CONVOLUTIONAL NEURAL NETWORK* UNTUK KLASIFIKASI PENYAKIT *LUMPY SKIN* PADA SAPI

Luthfi Adilal Mahbub[1)], Evi Dewi Sri Mulyani[2)], Teguh Ikhlas Ramadhan[3)]
[1,2,3]Prodi Teknik Informatika, Fakultas Teknik, Universitas Perjuangan Tasikmalaya

Correspondence author: L.A. Mahbub, luthfiadilalm26@gmail.com, Tasikmalaya, Indonesia

Abstract

Lumpy Skin Disease is a disease in cattle that causes decreased productivity and economic losses for farmers. Between July 2023 and June 2024, 6,803 cases were recorded in Indonesia, having a significant impact on the livestock industry. Early detection is crucial for controlling the spread of this disease. This study aims to optimize the Convolutional Neural Network for Lumpy Skin Disease classification by comparing the performance of several architectures, DenseNet-121, MobileNetV1, MobileNetV2, and GoogleNet. The dataset used was from Kaggle and consisted of 1,356 images. Results showed that GoogleNet achieved the best performance, with an *Accuracy* of 85.97% without segmentation and increasing to 87.03% with segmentation. However, segmentation does not continually improve *Accuracy*, as evidenced by DenseNet-121 and MobileNetV1, which experienced a slight decrease in *Accuracy*. In contrast, MobileNetV2 increased from 82.65% to 84.50%. This result shows that GoogleNet is more reliable in distinguishing lumpy skin and normal skin images than other architectures.

Keywords: lumpy skin, convolutional neural network, classification, image

Abstrak

Lumpy Skin Disease adalah penyakit viral pada sapi yang menyebabkan penurunan produktivitas dan kerugian ekonomi bagi peternak. Sejak Juli 2023 hingga Juni 2024, tercatat 6.803 kasus di Indonesia, yang berdampak signifikan pada industri peternakan. Deteksi dini sangat penting untuk mengendalikan penyebaran penyakit ini. Penelitian ini bertujuan mengoptimalkan *Convolutional Neural Network* untuk klasifikasi *Lumpy Skin Disease* dengan membandingkan performa beberapa arsitektur, yaitu DenseNet-121, MobileNetV1, MobileNetV2, dan GoogleNet. Dataset yang digunakan diambil dari Kaggle dan terdiri dari 1.356 citra,. Hasil penelitian menunjukkan bahwa GoogleNet memiliki performa terbaik dengan akurasi 85,97% tanpa segmentasi dan meningkat menjadi 87,03% dengan segmentasi. Namun, segmentasi tidak selalu meningkatkan akurasi, terbukti pada DenseNet-121 dan MobileNetV1 yang mengalami sedikit penurunan. Sebaliknya, MobileNetV2 mengalami peningkatan dari 82,65% menjadi 84,50%. Hal ini menunjukkan bahwa GoogleNet lebih andal dalam membedakan citra lumpy skin dan normal dibandingkan arsitektur lainnya.

Kata Kunci: *lumpy skin*, *convolutional neural network*, klasifikasi, citra

JURNAL ELEKTRO & INFORMATIKA
SWADHARMA (JEIS)
VOLUME 05 NOMOR 02 JULI 2025
ISSN 2774 – 5775 | eISSN 2774-5767

A. PENDAHULUAN

Lumpy Skin Disease (LSD) merupakan penyakit menular yang menyerang ternak sapi, disebabkan oleh virus dari *genus Capripoxvirus*. Penyakit ini ditandai dengan munculnya nodul atau benjolan pada kulit, disertai demam, penurunan berat badan, dan penurunan produksi susu (Abutarbush, 2017). Penyebaran LSD di antara sapi dapat terjadi melalui berbagai jalur, seperti kontak langsung antara sapi yang sehat dengan sapi terinfeksi, atau melalui perantara serangga, termasuk nyamuk, kutu, lalat kandang, dan lalat rumah. Selain itu, pakan dan air minum yang terkontaminasi air liur sapi yang terinfeksi juga berpotensi menjadi sumber penularan (Gupta et al., 2020).

Pada Juli 2023 hingga awal Juni 2024, terdapat 6.803 kasus yang menyebabkan kematian 7 ekor sapi di Jawa Timur. Pada penelitian yang dilakukan oleh (Namazi & Tafti, 2021) menyatakan bahwa wabah penyakit LSD menyebabkan banyak kerugian ekonomi di beberapa negara, yang menyebabkan penurunan produksi susu hingga 85%. Ini menunjukan bahwa LSD dapat mengancam para peternak sapi di Indonesia. Penularan yang masif ini mengancam produktivitas peternakan, karena menyebabkan penurunan berat badan dan produksi ternak. Urgensi penanganan cepat menjadi sangat penting untuk mencegah kerugian ekonomi yang lebih besar. Untuk menangani gejala utama *Lumpy Skin Disease* (LSD), diperlukan teknologi yang mampu mengidentifikasi penyakit ini melalui analisis gambar sapi secara otomatis. Saat ini, proses identifikasi kesehatan sapi masih sangat bergantung pada tenaga ahli, yang sering memerlukan waktu lama dan dapat menyebabkan keterlambatan dalam penanganan serta pencegahan penyebaran penyakit LSD (Alfiansyah & Litanianda, 2024).

Teknologi identifikasi penyakit berbasis gambar, khususnya melalui *Convolutional Neural Network* (CNN), sangat penting untuk mempercepat penanganan penyakit yang tergantung pada tenaga ahli. CNN efektif dalam klasifikasi gambar dan deteksi dini penyakit, memungkinkan peternak untuk melakukan pencegahan secara lebih cepat dan akurat. Dengan mengurangi jumlah parameter yang dilatih melalui pembagian bobot, CNN meningkatkan generalisasi dan mencegah overfitting, serta mampu mempelajari ekstraksi fitur dan klasifikasi secara bersamaan, menjadikannya lebih mudah diterapkan pada jaringan berskala besar dibandingkan jenis jaringan saraf lainnya (Alzubaidi et al., 2021).

Convolutional Neural Network dapat diimplementasikan menggunakan framework TensorFlow, yang sering dimanfaatkan untuk berbagai eksperimen dalam *Deep Learning*. TensorFlow memudahkan proses pelatihan model dengan dataset berukuran besar dan membantu dalam meningkatkan da memudahkan pengklasifikasian citra dengan model *Convolutional Neural Network* (Martins & Diesel, 2024).

Dikutip yang sudah dilakukan oleh (Alfiansyah & Litanianda, 2024) berdasarkan hasil evaluasi model, metode *Convulational Neural Network* (CNN) dengan arsitektur DenseNet121 untuk klasifikasi gambar kulit sapi antara yang sehat dan terinfeksi dengan penyakit Lumpy Skin. Melalui dataset yang terdiri dari 1024 citra kulit sapi, pelatihan model dilakukan dengan baik dengan mencapai akurasi validasi sebesar 80.21% pada epoch ke-10. Meskipun demikian, adanya fluktuasi pada data validasi juga menunjukkan adanya overfitting yang memerlukan perbaikan lebih lanjut untuk meningkatkan generalisasi model pada data yang belum pernah dilihat sebelumnya.

Berdasarkan hasil tersebut kinerja pada model masih belum optimal, maka perlu dilakukan optimisasi model CNN untuk *Lumpy Skin Disease* pada sapi dengan metode *Convulational Neural Network* (CNN) untuk meningkatkan kinerja model dengan dilakukan optimasi kinerja yaitu dengan memilih arsitektur, melakukan augmentasi

data, *hyperparameter* dan pemilihan *optimizer*.

B. METODE PENELITIAN

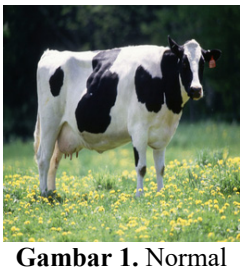
Pengumpulan Data

Dataset yang digunakan untuk klasifikasi penyakit Lumpy Skin pada sapi ini dataset adalah dataset public dari Kaggle dengan username AyushPanwar058. Total data dari datasetnya sebanyak 1356 data, dengan 2 class yaitu class normal skin dan lumpy skin.

Tabel 1. Dataset

Class	Jumlah Data	Sumber Data
Lumpy Skin	656	*Kaggle (Lumpy Cow Dataset)*
Normal Skin	700	
Total	1356	

Pada Gambar 2 menunjukan bahwa sapi terkena Lumpy Skin, pada Gambar 1 menunjukan bahwa kulit sapi normal tidak terkena Lumpy Skin. Dengan memilih data yang relevan, model CNN yang dioptimalkan dapat dilatih lebih baik untuk mengenali karakteristik spesifik dari *Lumpy Skin Disease*, sehingga meningkatkan akurasi klasifikasi penyakit ini. Pada tahap ini juga dilakukan penetapan jumlah data yang akan digunakan, yaitu sebanyak 1356 citra yang terbagi ke dalam 2 kelas Lumpy Skin Images dan Normal Skin Images.



Gambar 1. Normal

Gambar 2. Lumpy

Pemrosesan awal

Preprocessing data pada CNN melibatkan serangkaian langkah untuk mempersiapkan gambar agar sesuai dengan kebutuhan model. Langkah pertama adalah mengumpulkan dan memberi label data, lalu membagi beberapa data split data membagi ke 3 bagian seperti training, testing dan validation. Gambar kemudian diubah ukurannya agar memiliki dimensi seragam yang sesuai dengan input model (Susanto & Tinaliah, 2024).

Tabel 2. Split Data

Dataset	Training	Testing
Lumpy Skin	471	135
Normal Skin	477	136
Total	948	271
Dataset	**Validation**	
Lumpy Skin	68	
Normal Skin	69	
Total	137	

Pemodelan

Tahap modeling pada CNN adalah proses merancang, membangun, dan melatih model jaringan saraf untuk mempelajari pola-pola dalam data. Pada tahap ini, arsitektur CNN dirancang berdasarkan kebutuhan masalah, seperti jumlah layer convolusi, ukuran filter, pooling, dan jumlah neuron pada fully connected layer. Setelah arsitektur dirancang, data yang telah dipreproses digunakan untuk melatih model. Tahap modeling juga melibatkan validasi untuk memastikan bahwa model tidak hanya belajar dari data pelatihan tetapi juga mampu melakukan generalisasi pada data baru (Febriyanti, 2024)

Optimasi Model

Tahap optimization model bertujuan untuk meningkatkan performa model CNN agar dapat menghasilkan prediksi yang lebih akurat dan efisien. Pada tahap ini, berbagai teknik dan parameter dioptimalkan, seperti pemilihan optimizer, penyesuaian arsitektur model, dan tuning hyperparameter. Proses ini membantu model belajar secara efektif dari data, meminimalkan loss, dan menghindari permasalahan seperti overfitting atau underfitting. Optimasi dilakukan iteratif

JURNAL ELEKTRO & INFORMATIKA
SWADHARMA (JEIS)
VOLUME 05 NOMOR 02 JULI 2025
ISSN 2774 – 5775 | eISSN 2774-5767

berdasarkan hasil evaluasi performa model pada data validasi (Sentoso et al., 2025)

Tabel 3. Optimisasi Model

Model	**Lapisan**	***Optimizer***
DenseNet-121	121 lapisan	Adam
*MobileNet*V1	28 lapisan	Adam
*MobileNet*V2	53 lapisan	Adam
GoogleNet	22 lapisan	Adam
Model	***Batch size***	**Input *Shape***
DenseNet-121	32	224 x 224
*MobileNet*V1	32	224 x 224
*MobileNet*V2	32	224 x 224
GoogleNet	32	224 x 224
Model	***Epoch***	***Learning rate***
DenseNet-121	100	0.0001
*MobileNet*V1	100	0.0001
*MobileNet*V2	100	0.0001
GoogleNet	100	0.0001

Evaluasi

Pada tahap evaluasi, performa model diukur menggunakan dataset uji yang tidak pernah digunakan selama pelatihan atau validasi. Metrik evaluasi seperti akurasi, *Precision*, *Recall*, *F1-Score*, atau *Confusion Matrix* digunakan untuk memahami sejauh mana model mampu memprediksi data baru dengan benar. *Confusion Matrix* membantu memberikan gambaran detail tentang performa model dalam mengklasifikasikan tiap kelas, termasuk jumlah prediksi yang benar dan salah. Tahap ini penting untuk menilai kualitas model sebelum diterapkan pada data nyata (Raharjo, 2022).

Tabel 4. *Confusion Matrix*

Kelas	Kelas Prediksi	
Sesungguhnya	Positif	Negatif
Positif	TP	FN
Negatif	FP	TN

Keterangan

True Positive : Merujuk pada situasi dimana model berhasil memprediksi hasil positif,

True Negative : Hal ini merujuk pada situasi dimana model berhasil memprediksi hasil negatif dengan tepat.

False Positive : Situasi ini terjadi ketika model membuat kesalahan falam memprediksi hasil positif.

False Negative : Terdapat situasi dimana model membuat kesalahan dalam memprediksi hasil negatif.

Adapun istilah yang digunakan untuk mencari nilai dalam metode *Confusion Matrix* :

Akurasi adalah metrik evaluasi yang paling umum digunakan dalam klasifikasi. Metrik ini mengukur persentase prediksi yang benar dibandingkan dengan total jumlah prediksi yang dilakukan model.

$$Accuracy = \frac{(\text{Jumlah prediksi benar})}{(\text{Total prediksi})} \quad \ldots\ldots (1)$$

Precision Presisi adalah perbandingan jumlah prediksi positif yang tepat dengan total prediksi positif yang dilakukan.

$$Precision = TP/(TP+FP) \quad \ldots\ldots\ldots\ldots (2)$$

Recall atau yang juga disebut sebagai *True Positive* (TP) adalah ukuran proporsi kasus positif yang berhasil diidentifikasi seacara actual.

$$Recall = TP/(TP+FN) \quad \ldots\ldots\ldots\ldots\ldots (3)$$

F1-Score merupakan metrik yang menggabungkan presisi dan *Recall* dari sebuah model klasifikasi.

$$F1\text{-}Score = \frac{(Precision \text{ X } Recall)}{(Precision+Recall)} \quad \ldots\ldots\ldots (4)$$

Penerapan

Tahap ini model akan di integrasikan kedalam sebuah API (*Application Programming Interface*), API yang digunakan adalah API yang berjalan di lokal seperti Ngrok, dimana model dapat menerima input dan menghasilkan output. Tujuannya agar prediksi model yang telah dilatih dapat digunakan oleh orang lain (Putri et al., 2024)

C. HASIL DAN PEMBAHASAN

Optimisasi model adalah suatu proses untuk mencapai hasil akurasi yang ideal. Pada penelitian ini data yang digunakan adalah data *public* yang terapat dari *Kaggle* dengan username AyushPanwar058. Jumlah data yang digunakan sebanyak 1356 data dibagi kedalam dua *class* yaitu *lumpy skin* dan *normal skin.* Optimisasi model ini akan dilakukan seperti menyetel *hyperparameter, optimizer,* augmentasi, segmentasi dan membandingkan arsitektur CNN seperti *DenseNet-121, MobileNetV1, MobileNetV2, GoogleNet*.

Pada penelitian sebelumnya dilakukan dengan jumlah dataset 1024 dengan 324 data *lumpy skin,* 1000 data *normal skin*. Pada penelitian sebelumnya mendapatkan hasil akurasi sebesar 80.21% pada *epoch* ke 10. Pada penelitian ini akan dilakukan eksperimen ulang sesuai dengan jumlah dataset dan pengaturan model yang digunakan. Pengaturan model yang digunakan, arsitektur menggunakan *DenseNet-121*, *optimizer Adam*, input 224x224, *batch* 32, *learning* rate 0.0001 dan *epoch* 10, pada penelitian ini akan mencoba mengeksperimen lagi model yang dilakukan pada penelitian sebelumnya, seperti pada Tabel 5.

Tabel 1. Pengaturan model penelitian sebelumnya

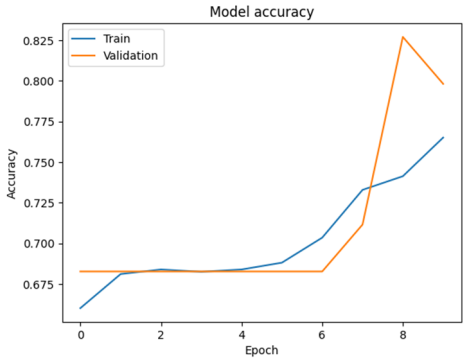
Model	***Optimizer***	**Input**	***Batch***
DenseNet-122	Adam	224x224	32
Learning rate		***Epoch***	
0.0001		10	

Pada tabel 6. menunjukan acc dari tiap epoch meningkat walaupun terkadang ada penuruan apoch. Pada los setiap epoch menurun begitu juga pada val los nya, pada epoch 10 acc menghasilkan 74% dan untuk val acc menghasilkan 79%.

Tabel 6. Nilai acc tiap epoch

Epoch	**Data *Train***		**Data *Testing***	
	Acc	**Los**	**Val Acc**	**Val Los**
1	0.6227	0.6705	0.6827	0.6174
2	0.6708	0.6431	0.6827	0.6110
3	0.6790	0.6225	0.6827	0.5974
4	0.6816	0.5970	0.6827	0.5838
5	0.6715	0.5804	0.6827	0.5664
6	0.7014	0.5411	0.6827	0.5496
7	0.6896	0.5332	0.6827	0.5469
8	0.6897	0.5252	0.7115	0.4950
9	0.7454	0.4736	0.8269	0.4302
10	0.7472	0.5004	0.7981	0.3978

Pada Gambar 3. grafik menunjukkan tren peningkatan akurasi model selama pelatihan, dengan akurasi data *training* (garis biru) dan validasi (garis oranye) yang awalnya hampir sama, lalu meningkat seiring bertambahnya *epoch*. Akurasi validasi mengalami lonjakan signifikan di *epoch* terakhir, yang bisa menandakan model mulai menggeneralisasi dengan baik atau mengalami fluktuasi karena batch kecil. Namun, perbedaan antara akurasi training dan validasi yang tidak terlalu besar menunjukkan bahwa model masih dalam tahap pembelajaran yang baik, meskipun perlu dianalisis lebih lanjut menggunakan loss *function* untuk memastikan tidak terjadi *overfitting*.

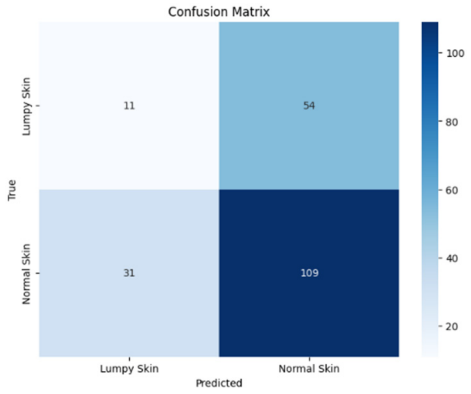


Gambar 3. Model akurasi penelitian sebelumnya

Setelah melakukan train model dan menghasilkan akurasi per *epoch* nya , setelah itu melihat hasil evaluasi model berupa *Confusion Matrix*, dari *Confusion Matrix*

JURNAL ELEKTRO & INFORMATIKA
SWADHARMA (JEIS)
VOLUME 05 NOMOR 02 JULI 2025
ISSN 2774 – 5775 | eISSN 2774-5767

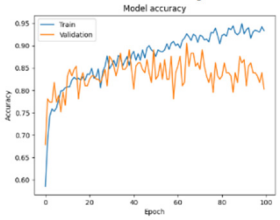
dapat dilihat data gambar yang diterima oleh model dan juga yang *loss* atau tidak diterima oleh model. Pada gambar 4 bisa dilihat bahwa data gambar *Lumpy Skin* yang diterima yaitu (11) dan yang tidak diterima yaitu (31), untuk normal skin data yang diterima yaitu sebesar (109) dan yang tidak diterima yaitu (54). Disini berarti model belum mampu mengenali data gambar *Lumpy Skin* tapi model sudah mampu mengenali data gambar normal skin. Hasil *Confusion Matrix* dapat dilihat pada Gambar 4 .



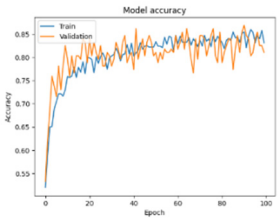
Gambar 4. *Confusion Matrix* penelitian sebelumnya

Optimisasi Model

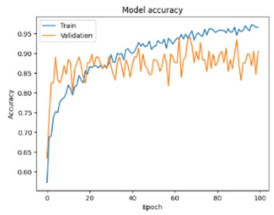
Hasil dari penelitian sebelumnya bahwa model CNN masih belum cukup baik untuk mengklasifikasi *lumpy skin*. Maka dilakukanlah optimisasi model guna meningkatkan akurasi menjadi lebih baik.



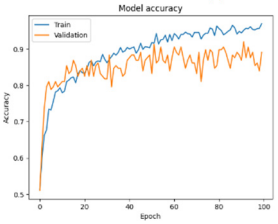
Gambar 1. GoogleNet tanpa segmentasi



Gambar 2. MobileNetV2 tanpa segmentasi

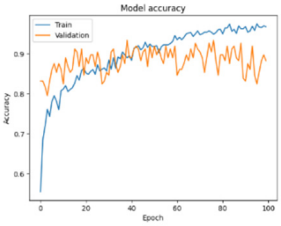


Gambar 3 MobileNetV1 tanpa segmentasi

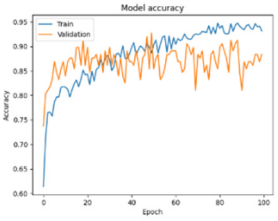


Gambar 4 DenseNet-121 tanpa segmentasi

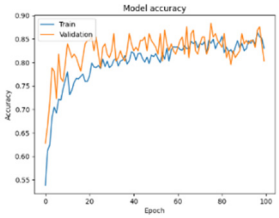
Pada Gambar 5 – 8 menunjukan hasil model akurasi dari setiap arsitektur tanpa penambahan segmentasi, model akurasi arsitektur *MobileNetV2* memberikan hasil yang konsisten dibandingkan arsitektur lain dari data train dan juga data validationnya. Pada arsitektur yang lain menunjukan fluktuasi dan juga memperlihatkan persilangan antara data train dan juga *validation*, walaupun persilangannya tidak terlalu jauh. Tapi hasil ini bukan berarti final ataupun menentukan kinerja model CNN meningkat atau menjadi lebih baik kinerjanya, karena hasil ini belum di evaluasi seperti *Confusion Matrix* yang dapat memperlihat kan model tersebut sudah dapat mengenali data atau belum.



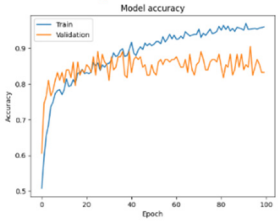
Gambar 5. MobileNetV1 dengan segmentasi



Gambar 6. GoogleNet dengan segmentasi



Gambar 7. MobileNetV2 dengan segmentasi



Gambar 8. DenseNet-121 dengan segmentasi

Pada Gambar 9 - 12 menunjukan hasil model akurasi dengan penambahan segmentasi, pada hasil dari setiap arsitektur yang telah dilakukan eksperimen bahwa *MobileNetV2* menghasilkan akurasi yang konsisten daripada arsitektur yang lainnya. Tingkat akurasi yang dihasilkan arsitektur *MobileNetV2* sebesar 82%, tingkat akurasi tertinggi yang dihasilkan keempat arsitektur ini adalah *MobileNetV1* dengan tingkat akurasi 96%, dengan tingkat akurasi sebesar 96% *MobileNetV1* mununjukan model akurasi yang *overfit* walaupun tidak terlalu signifikan.

Tabel 2. Hasil akurasi per epoch 100 tanpa segmentasi

Arsitektur	**Data *Train***	
	Acc / epoch 100	**Loss / epoch 100**
DenseNet-121	0.9671	0.0805
MobileNet-V1	0.9658	0.8065
MobileNet-V2	0.8229	0.3895
GoogleNet	0.9772	0.1728
Arsitektur	**Data *Testing***	
	Val Acc / epoch 100	**Val Loss / epoch 100**
DenseNet-121	0.8905	0.3744
MobileNet-V1	0.9051	1.0492
MobileNet-V2	0.8102	0.3805
GoogleNet	0.8029	0.5893

Tabel 3. Hasil akurasi per epoch 100 dengan segmentasi

Arsitektur	**Data *Train***	
	Acc / epoch 100	**Loss / epoch 100**
DenseNet-121	0.9634	0.0991
MobileNet-V1	0.9653	0.8476
MobileNet-V2	0.8204	0.3928
GoogleNet	0.9218	0.2015
Arsitektur	**Data *Testing***	
	Val Acc / epoch 100	**Val Loss / epoch 100**
DenseNet-121	0.8321	0.6231
MobileNet-V1	0.8832	1.0960
MobileNet-V2	0.8029	0.3881
GoogleNet	0.8832	0.3072

Dilihat pada tabel 6 dan 7 menunjukan hasil akurasi data *training* dari per *epoch* 100 setiap arsitektur, performa model sebelum dan sesudah segmentasi menunjukkan perbedaan yang signifikan. Sebelum segmentasi, model *MobileNet*-V1 memiliki akurasi validasi tertinggi sebesar 90.51%, namun disertai dengan nilai loss validasi yang cukup tinggi, yaitu 1.0492, yang dapat mengindikasikan potensi *overfitting*. Setelah dilakukan segmentasi, terjadi perubahan performa pada semua model. *GoogleNet* mengalami peningkatan akurasi validasi yang paling signifikan, dari 80.29% menjadi 88.32%, serta penurunan loss validasi dari 0.5893 menjadi 0.3078. Hal ini menunjukkan bahwa segmentasi berkontribusi dalam meningkatkan stabilitas dan keandalan model, terutama untuk *GoogleNet*. Sementara itu, model *DenseNet-121* dan *MobileNet*-V2 mengalami sedikit penurunan akurasi setelah segmentasi, meskipun perubahan *loss* validasi tidak terlalu drastis. Secara keseluruhan, segmentasi terbukti membantu meningkatkan performa *GoogleNet* dalam mendeteksi *Lumpy Skin Disease*, menjadikannya kandidat model yang lebih stabil dan akurat dibandingkan arsitektur lainnya.

JURNAL ELEKTRO & INFORMATIKA
SWADHARMA (JEIS)
VOLUME 05 NOMOR 02 JULI 2025
ISSN 2774 – 5775 | eISSN 2774-5767

Tabel 4. Hasil data uji tanpa segmentasi

Arsitektur	*Acc*	*Loss*
DenseNet-121	0.8523	0.4616
*MobileNet*V1	0.8360	0.4006
*MobileNet*V2	0.8265	0.4169
GoogleNet	0.8597	0.4151

Tabel 5. Hasil data uji dengan segmentasi

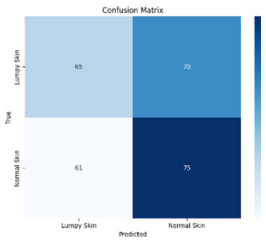
Arsitektur	*Acc*	*Loss*
DenseNet-121	0.8499	0.5471
*MobileNet*V1	0.8600	1.2791
*MobileNet*V2	0.8450	0.3597
GoogleNet	0.8703	0.5242

Berdasarkan hasil evaluasi data uji, penggunaan segmentasi tidak memberikan peningkatan akurasi yang signifikan pada model. Pada Tabel 8, yang menampilkan hasil tanpa segmentasi, model *GoogleNet* memiliki akurasi tertinggi sebesar 0.8597, diikuti oleh *DenseNet-121* (0.8523), *MobileNet*V1 (0.8360), dan *MobileNet*V2 (0.8265). Selain itu, nilai loss tanpa segmentasi lebih rendah dibandingkan dengan model yang menggunakan segmentasi, yang mengindikasikan bahwa model lebih stabil dan memiliki kesalahan yang lebih kecil dalam melakukan prediksi.

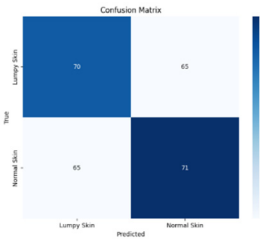
Sementara itu, pada Tabel 9, yang menunjukkan hasil evaluasi dengan segmentasi, *GoogleNet* masih memiliki akurasi tertinggi sebesar 0.8703, namun beberapa model lain mengalami penurunan akurasi, seperti *DenseNet-121* (0.8499) dan *MobileNet*V1 (0.8000). Selain itu, nilai loss secara umum meningkat, terutama pada *MobileNet*V1, yang mengalami kenaikan loss dari 0.4006 menjadi 1.2791, menandakan bahwa model menjadi lebih tidak stabil setelah segmentasi diterapkan.

Dari hasil ini, dapat disimpulkan bahwa penerapan segmentasi tidak selalu meningkatkan performa model. Meskipun ada sedikit peningkatan akurasi pada *GoogleNet* dan *MobileNet*V2, model lain justru mengalami penurunan akurasi dan peningkatan loss, yang menunjukkan bahwa segmentasi dalam bentuk ini mungkin tidak efektif untuk meningkatkan generalisasi model terhadap data uji.

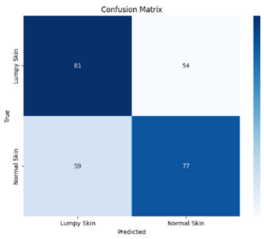
Pada gambar 13 sampai 16 menunjukan hasil evaluasi model CNN dnegan *Confusion Matrix* dari setiap arsitektur yang digunakan dengan tanpa segmentasi. Disini model dapat memahami data dengan baik , berbeda dengan model yang ditambahkan segmentasi. Pada hasil *Confusion Matrix* ini arsitektur yang memberikan hasil yang baik adalah *GoogleNet*.



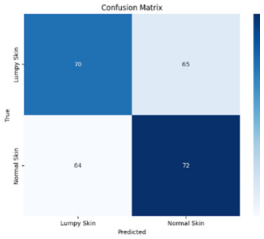
Gambar 9. Confusion matrix arsitektur DenseNet-121 tanpa segmentasi



Gambar 10. Confusion matrix arsitektur MobileNetV1 tanpa segmentasi

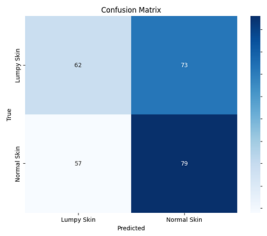


Gambar 12. Confusion matrix arsitektur GoogleNet tanpa segmentasi

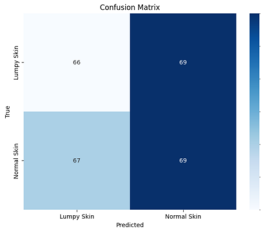


Gambar 11. Confusion matrix arsitektur MobileNetV2 tanpa segmentasi

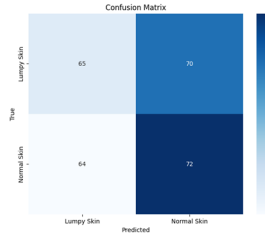
Hasil evaluasi *Confusion Matrix* pada gambar 17– 20 menunjukan hasil yang kurang baik, evaluasi *Confusion Matrix* ini diambil dari tes yang telah kita bagi pada tahap split data. Data tes yang dibagi kedua *class* tersebut sebanyak 271 data yang terdiri dari 135 data tes *Lumpy Skin* dan 136 data tes *normal skin,* terlihat bahwa model tidak dapat menklasifikasi data dengan baik.



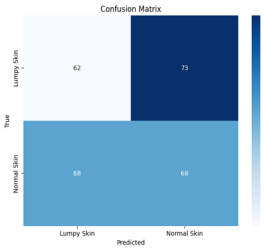
Gambar 13. Confusion matrix MobileNetV1 dengan segmentasi



Gambar 14. Confusion matrix DenseNet-121 dengan segmentasi



Gambar 16. Confusion matrix MobileNetV2 dengan segmentasi



Gambar 15. Confusion matrix GoogleNet dengan segmentasi

Tabel 6. Hasil evaluasi *Confusion Matrix* tanpa segmentasi

Model	*Lumpy Skin*		*Normal Skin*	
	TP	**FN**	**FP**	**TN**
DenseNet-121	65	70	61	75
*MobileNet*V1	70	65	65	70
*MobileNet*V2	70	64	65	74
GoogleNet	81	54	59	77

Tabel 7. Hasil evaluasi *Confusion Matrix* dengan segmentasi

Arsitektur	*Lumpy Skin*		*Normal Skin*	
	TP	**FN**	**FP**	**TN**
DenseNet-121	66	69	67	69
*MobileNet*V1	62	73	57	79
*MobileNet*V2	65	70	64	72
GoogleNet	62	73	68	68

Berdasarkan hasil *Confusion Matrix* pada Tabel 10 dan Tabel 11, terlihat bahwa penggunaan segmentasi berpengaruh terhadap performa model dalam mengklasifikasikan *Lumpy Skin Disease*. Sebelum segmentasi, model *GoogleNet* menunjukkan performa terbaik dengan *True Positive* (TP) sebanyak 81 dan *True Negative* (TN) sebanyak 77, yang berarti model mampu mengenali kasus *Lumpy Skin* dan *Normal Skin* dengan cukup baik. Namun, setelah segmentasi, TP pada semua model mengalami penurunan, di mana *GoogleNet* turun dari 81 menjadi 62, yang berarti model menjadi kurang sensitif terhadap *Lumpy Skin* setelah segmentasi.

Selain itu, *False Negative* (FN) meningkat pada hampir semua model setelah segmentasi. Sebagai contoh, pada *MobileNet*V1, FN meningkat dari 65 menjadi 73, yang berarti lebih banyak kasus *Lumpy Skin* yang salah diklasifikasikan sebagai *Normal Skin*. Hal ini menunjukkan bahwa setelah segmentasi, model menjadi lebih sulit mengenali *Lumpy Skin*, yang dapat menyebabkan risiko kesalahan diagnosis meningkat.

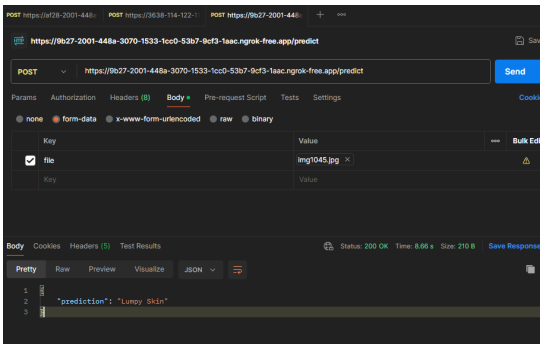
Namun, ada sedikit perbaikan pada *False Positive* (FP) untuk beberapa model setelah segmentasi. Sebagai contoh, *GoogleNet* mengalami sedikit peningkatan FP dari 59 menjadi 68, yang berarti model lebih berhati-hati dalam mengklasifikasikan *Normal Skin* sebagai *Lumpy Skin*. Akan tetapi, peningkatan FP ini tidak cukup signifikan untuk mengimbangi peningkatan FN, sehingga secara keseluruhan, model mengalami penurunan performa dalam mendeteksi *Lumpy Skin Disease*.

Berdasarkan hasil *Confusion Matrix*, penggunaan segmentasi pada arsitektur model ini tampaknya tidak memberikan peningkatan performa yang signifikan. Sebaliknya, segmentasi justru menyebabkan penurunan sensitivitas model terhadap *Lumpy Skin Disease*, seperti yang terlihat dari meningkatnya *False Negative* (FN) pada semua arsitektur setelah segmentasi

JURNAL ELEKTRO & INFORMATIKA
SWADHARMA (JEIS)
VOLUME 05 NOMOR 02 JULI 2025
ISSN 2774 – 5775 | eISSN 2774-5767

diterapkan. Arsitektur yang digunakan dalam penelitian ini kurang cocok jika dikombinasikan dengan segmentasi dalam bentuk yang digunakan saat ini. *Dataset* juga memperngaruhi dalam segmentasi yang digunakan. Hal ini dikarenakan segmentasi dapat menghilangkan detail penting dalam gambar yang dibutuhkan oleh model untuk membedakan *Lumpy Skin* dari *Normal Skin*.

Penerapan



Gambar 17. Penggunaa API machine

Setelah mengetahui model mana yang lebih baik, langkah selanjutnya adalah melakukan *deployment* menggunakan *FastAPI*. API ini akan berjalan di lingkungan lokal dengan bantuan ngrok untuk membuatnya dapat diakses secara *publik*. Model yang telah dilatih harus disimpan dalam format .h5 agar dapat dimuat kembali saat API dijalankan. Selanjutnya, dibuat endpoint */predict* yang berfungsi untuk menerima input gambar, melakukan *Preprocessing*, dan mengirimkannya ke model untuk mendapatkan hasil prediksi. Setelah itu, API ini akan dihubungkan dengan runtime ngrok sehingga dapat diakses melalui URL (*Uniform Resource Locator*) yang diberikan oleh ngrok. API yang telah dibuat lalu dijalankan dan akan menampilkan localhost yang berjalan di port 8000, port ini akan dihubungkan dengan runtime dari *ngrok* .URL (*Uniform Resource Locator)* yang dihasilkan *ngrok* setelah dihubungkan dengan port dari API akan seperti berikut https://734c-180-243-189-83.ngrok-free.app/predict.

D. PENUTUP

Optimasi algoritma *Convolutional Neural Network* (CNN) untuk klasifikasi *Lumpy Skin Disease* dilakukan dengan membandingkan arsitektur DenseNet-121, MobileNetV1, MobileNetV2, dan GoogleNet, di mana GoogleNet menunjukkan performa terbaik dengan akurasi 85,97% tanpa segmentasi dan 87,03% dengan segmentasi. DenseNet-121 memiliki akurasi 85,23% tanpa segmentasi dan 84,99% dengan segmentasi, sementara MobileNetV1 mengalami penurunan dari 83,60% menjadi 80,00% setelah segmentasi. Di sisi lain, MobileNetV2 justru mengalami peningkatan akurasi dari 82,65% menjadi 84,50% setelah segmentasi. Meskipun segmentasi diharapkan meningkatkan akurasi, hasil evaluasi menunjukkan bahwa tidak semua arsitektur mengalami peningkatan performa, bahkan beberapa seperti DenseNet-121 dan MobileNetV1 mengalami penurunan akurasi serta peningkatan loss. Keunggulan GoogleNet didukung oleh arsitekturnya yang lebih dalam dan efisien dalam mengekstrak fitur dari citra penyakit, namun optimasi lebih lanjut melalui pengaturan hyperparameter atau teknik pelatihan yang lebih baik tetap diperlukan untuk meningkatkan stabilitas dan akurasi model dalam klasifikasi *Lumpy Skin Disease*. Selain itu, penelitian lebih lanjut mengenai teknik segmentasi yang lebih optimal serta eksplorasi arsitektur CNN yang lebih ringan namun tetap akurat dapat dilakukan untuk meningkatkan efisiensi dan performa model secara keseluruhan.

E. DAFTAR PUSTAKA

Abutarbush, S. M. (2017). *Lumpy Skin Disease* (Knopvelsiekte, Pseudo-Urticaria, Neethling Virus Disease, Exanthema Nodularis Bovis). In *Emerging and Re-emerging Infectious Diseases of Livestock* (pp. 309–326). Cham, Switzerland : Springer.

https://doi.org/10.1007/978-3-319-47426-7_14

Ainur Rahman, & Suroyo, H. (2021). Analisis Data Produk Elektronik Di E-Commerce Dengan Metode Algoritma K-Means Menggunakan Python. *Journal of Advances in Information and Industrial Technology*, *3*(2), 11–18. https://doi.org/10.52435/jaiit.v3i2.158

Alfiansyah, N. S., & Litanianda, Y. (2024). Identifikasi *Lumpy Skin Disease* Menggunakan Tensorflow Dengan Metode Convolutional Neuron Network. *JATI : Jurnal Mahasiswa Teknik Informatika*, *8*(4), 7330–7336. https://doi.org/10.36040/jati.v8i4.10238

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, *8*, 53. https://doi.org/10.1186/s40537-021-00444-8

Febriyanti, F. A. (2024). Image Processing Dengan Metode *Convolutional Neural Network* (Cnn) Untuk Deteksi Penyakit Kulit Pada Manusia. *Kohesi: Jurnal Sains Dan Teknologi*, *3*(10), 21–30. https://doi.org/10.3785/kohesi.v3i10.4088

Gupta, T., Patial, V., Bali, D., Angaria, S., Sharma, M., & Chahota, R. (2020). A review: *Lumpy Skin Disease* and its emergence in India. *Veterinary Research Communications*, *44*, 111–118. https://doi.org/10.1007/s11259-020-09780-1

Li, H., Zhang, X., Liu, Y., Zhang, Y., Wang, Q., Zhou, X., Liu, J., Wu, H., & Wang, H. (2019). D-net: A simple framework for improving the generalization of machine reading comprehension. *MRQA@EMNLP 2019 - Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, 212–219.

Martins, D., & Diesel, L. (2024). *Developing a Convolutional Neural Network (CNN) Model for Facial Expression Recognition (FER)*. https://doi.org/10.63227/956.493.45

Musfiroh, D., Khaira, U., Utomo, P. E. P., & Suratno, T. (2021). Analisis Sentimen terhadap Perkuliahan Daring di Indonesia dari Twitter Dataset Menggunakan InSet Lexicon. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, *1*(1), 24–33. https://doi.org/10.57152/malcom.v1i1.20

Namazi, F., & Tafti, A. K. (2021). *Lumpy Skin Disease*, an emerging transboundary viral disease: A review. *Veterinary Medicine and Science*, *7*(3), 888–896. https://doi.org/10.1002/vms3.434

Putri, R. A., Satyawan, A. S., Prihantono, J. A., Linggi, R. S., Paramita, I. G. A. P. S., Iswarawati, N. K. E., Akbar, F., & Utomo, P. A. (2024). Model Deep Learning untuk Klasifikasi Objek pada Gambar Fisheye. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *11*(3), 519–528. https://doi.org/10.25126/jtiik.938047

Raharjo, B. (2022). *Deep Learning dengan Python*. Semarang : Yayasan Prima Agus Teknik.

Ratniasih, N. L., Jayanti, N. W. N., & ... (2023). Klasifikasi Kepuasan Mahasiswa Menggunakan Algoritma Support Vector Machine dan Metode Stemming Sastrawi. *Prosiding CORISINDO*, 373–378.

Santoso, D. P., & Wibowo, W. (2022). Analisis Sentimen Ulasan Aplikasi Buzzbreak Menggunakan Metode Naïve Bayes Classifier pada Situs Google Play Store. *Jurnal Sains Dan Seni ITS*, *11*(2). https://doi.org/10.12962/j23373520.v11i

2.72534

Sentoso, T., Ardiansyah, F., Tamuntuan, V., Wangsa, S. S., Kusrini, & Kusnawi. (2025). Identification of *Lumpy Skin Disease* in Cattle with Image Classification using the *Convolutional Neural Network* Method. *Sistemasi: Jurnal Sistem Informasi*, *14*(5), 864–873. https://doi.org/10.32520/stmsi.v13i3.2569

Susanto, R. I., & Tinaliah. (2024). Klasifikasi Penyakit Cacar Menggunakan Arsitektur AlexNet. *Jurnal Algoritme*, *5*(1), 47–56. https://doi.org/10.35957/algoritme.v5i1.9045