Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

Monkeypox Classification Using Convolutional Neural Networks (CNN) Pruned Residual Network-50 (ResNet-50) Architecture on Flutter Framework

Irfan Priatna*[1], Ipung Permadi [2], Nofiyati [3], Eddy Maryanto[4]

[1,2,3,4]Informatics, Universitas Jenderal Soedirman, Indonesia

Email: [1]irfanpriatna402@gmail.com

Received : Aug 4, 2025; Revised : Aug 21, 2025; Accepted : Aug 21, 2025; Published : Aug 24, 2025

Abstract

The monkeypox outbreak, which was previously only found in Africa, has now spread to other continents, including Asia, causing public concern as it occurred shortly after the COVID-19 pandemic was declared over. This disease has symptoms similar to cowpox, chickenpox, and measles, making early detection based on visual observation difficult. To address this issue, various studies have developed Deep Learning (DL)-based classification models using datasets such as WSI, MSID, MCSI, and MSLD v2, which are also utilized in this research. This study proposes a pruned ResNet-50 model using the Global MP method for pruning and QAT for quantization. These modifications not only maintain the model's performance with an accuracy of 94.44%, precision of 94.12%, recall of 94.71%, and F1-score of 94.16%, but also significantly reduce the model size to just 20.993 MB. As a result, the model can be implemented on Android devices with limited resources, enabling rapid and practical early detection of monkeypox in the field without requiring large-scale servers. Blackbox testing results show that the Flutter-based application utilizing this model performs well, potentially providing tangible support for medical personnel and the public in monitoring the spread of monkeypox in a more efficient and accessible manner.

Keywords : *CNN, Flutter, Monkeypox, Pruning, Quantization, ResNet-50*

1. INTRODUCTION

Kekhawatiran masyarakat kembali mencuat dengan adanya penyebaran wabah cacar monyet (*monkeypox*) dari Afrika, terutama setelah pandemi COVID-19 dinyatakan berakhir pada tahun 2023 lalu [1]. Penyakit ini termasuk ke dalam zoonosis yang disebabkan oleh Virus *Monkeypox*, bagian dari genus *Orthopoxvirus* yang juga mencakup *Virus Variola*, *Vaccinia*, dan *Cowpox* [2]. Beberapa studi klinis telah menunjukkan bahwa penyakit ini memiliki karakteristik epidemiologis dan klinis yang baru [3]. Gejala cacar monyet pada manusia mirip dengan campak (*measles*), cacar sapi (*cowpox*), dan cacar air (*chickenpox*) [4]. Kemiripan tersebut menyebabkan tantangan besar dalam proses diagnosis karena keterbatasan deteksi dan kesulitan membedakan gejala yang serupa [5]. Proses diagnosis penyakit cacar menjadi lebih sulit dan membutuhkan kejelian serta pengalaman dari ahli [6].

Dalam upaya mendukung deteksi dini dan diagnosis yang akurat, pengembangan alat deteksi yang dapat mengenali penyakit cacar monyet menjadi sangat penting. Ditambah lagi ahli medis yang jumlahnya terbatas tidak dapat menjawab seluruh keresahan dari masyarakat dalam mengenali penyakit cacar monyet. Namun dengan adanya alat tersebut, masyarakat dapat dengan mudah melakukan deteksi awal penyebaran cacar monyet. Teknologi kecerdasan buatan *Machine Learning* (ML) menawarkan solusi melalui teknik klasifikasi yang mengelompokkan data berdasarkan kesamaannya dengan label kelas atau kelompok data yang sudah ditentukan [7]. Namun, data berjenis gambar membutuhkan komputasi yang lebih kompleks sehingga teknik ML sederhana kurang efektif dalam mengenali pola

visual yang rumit. Oleh karena itu, *Deep Learning* (DL) menjadi pilihan yang lebih baik, karena model komputasi ini menyerupai cara kerja otak manusia dan mampu mengenali fitur gambar secara otomatis [8]. Salah satu algoritma DL yang banyak digunakan pada pengolahan gambar adalah *Convolutional Neural Networks* (CNN), algoritma ini memiliki kemampuan mempelajari fitur spesifik dari gambar secara otomatis melalui *hidden layer* yang mendalam dan kompleks [9], [10].

Meski model DL dapat menghasilkan akurasi yang lebih tinggi, kompleksitas dan ukuran model yang besar membawa risiko *overfitting* terhadap data pelatihan [11]. Selain itu, pembelajaran dapat terhambat karena masalah lain yaitu gradien yang menyusut secara eksponensial seiring penambahan layer [12]. Karena hal itulah, jaringan *residual* menjadi salah satu solusi dengan menambahkan *skip links* atau koneksi lompat yang memungkinkan *output* dari satu layer diteruskan ke dua hingga tiga layer selanjutnya [13], [14]. Berbagai penelitian mengenai alat deteksi cacar monyet telah dilakukan, salah satunya yang dilakukan oleh [15] dengan meneliti berbagai model klasifikasi CNN untuk mengenali penyakit cacar monyet. Penelitian tersebut menunjukkan hasil yang cukup baik dengan akurasi paling tinggi 82.26% pada model DenseNet-121. Sementara itu, model ResNet-50 dengan ukuran layer kurang dari setengah DenseNet-121 menghasilkan akurasi 76.26%. Hasil penelitian sudah menunjukkan akurasi yang cukup baik dan dapat diimplementasikan dengan optimal pada aplikasi berbasis *website* [16].

Di samping itu, arsitektur CNN terdiri atas jutaan parameter yang harus dilatih. Hal ini dapat menyebabkan *overparameterization* pada model dan berpotensi memperlambat kinerja dari klasifikasi cacar monyet [17]. Sebagai contoh pada arsitektur *Residual Network* dengan 50 layer atau dikenal dengan ResNet-50, memiliki 50 layer dengan 26 juta parameter dan membutuhkan 4.1 miliar *Floating-Point Operations* (FLOPs) [18]. Masalah ini dapat diatasi melalui penerapan *pruning* dan *quantization*, metode ini akan membuat sub-model dari jaringan yang sudah ada dengan melakukan pengurangan parameter [19]. Hal ini membuka kesempatan untuk menerapkan model yang kompleks pada perangkat *mobile* secara langsung [20], [21], dan [22].

Di sisi lain, jumlah pengguna *smartphone* pada tahun 2024 diperkirakan mencapai 4.5 miliar dan akan terus bertambah [23]. Dengan demikian, penerapan model jaringan pada perangkat *mobile* menjadi keharusan agar lebih mudah diakses. Salah satu *framework* yang dapat digunakan untuk membuat aplikasi *mobile* yaitu Flutter. Flutter sendiri merupakan *Software Development Kit* (SDK) *mobile* yang dibangun secara *open source* oleh Google dan menawarkan kemudahan dalam *multi platform* [24], [25].

Penelitian ini akan menerapkan model *pruned* ResNet-50 pada perangkat *mobile* android untuk klasifikasi cacar monyet dan penyakit lain yang sejenis. Model tersebut merupakan modifikasi dari basis ResNet-50 dengan tambahan *pruning* dan *quantization*. Pengaruh dari proses modifikasi tersebut akan dipelajari terkait efektivitasnya dalam mengurangi ukuran model.

2. METHOD

Prosedur kerja yang dilakukan dalam penelitian ini meliputi pengumpulan dan pemilahan data, *preprocessing*, pemisahan data, augmentasi data, pelatihan model, evaluasi model, *pruning*, dan pengembangan aplikasi android. Untuk lebih jelasnya dapat ditunjukkan dengan Figure 1.

2.1. Pengumpulan dan Pemilahan Data

Pada tahap ini, data dikumpulkan berdasarkan empat set data yang tersedia. Kelas yang digunakan dalam penelitian ini yaitu enam dari delapan kelas tersedia. Kelas *acne* akan digabungkan dengan kelas HFMD sedangkan kelas *smallpox* tidak akan digunakan karena keterbatasan kualitas dan jumlah data yang tersedia. Data tersebut kemudian dilakukan pemilahan terhadap duplikasi antar set data, penyesuaian jumlah antar kelas data, dan modifikasi terhadap latar belakang gambar.

Data yang digunakan dalam penelitian ini bersumber dari data sekunder yang didapatkan dari beberapa penelitian terdahulu. Penelitian pertama [26] menghasilkan set data *Web-scraped Skin Image*

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

(WSI) yang di dalamnya berjumlah 804 dan terdiri atas 6 jenis gambar kulit yang berbeda yaitu set data *chickenpox* 178 buah, *cowpox* 54 buah, *healthy* 50 buah, *measles* 47 buah, *monkeypox* 117 buah, dan *smallpox* 358 buah.

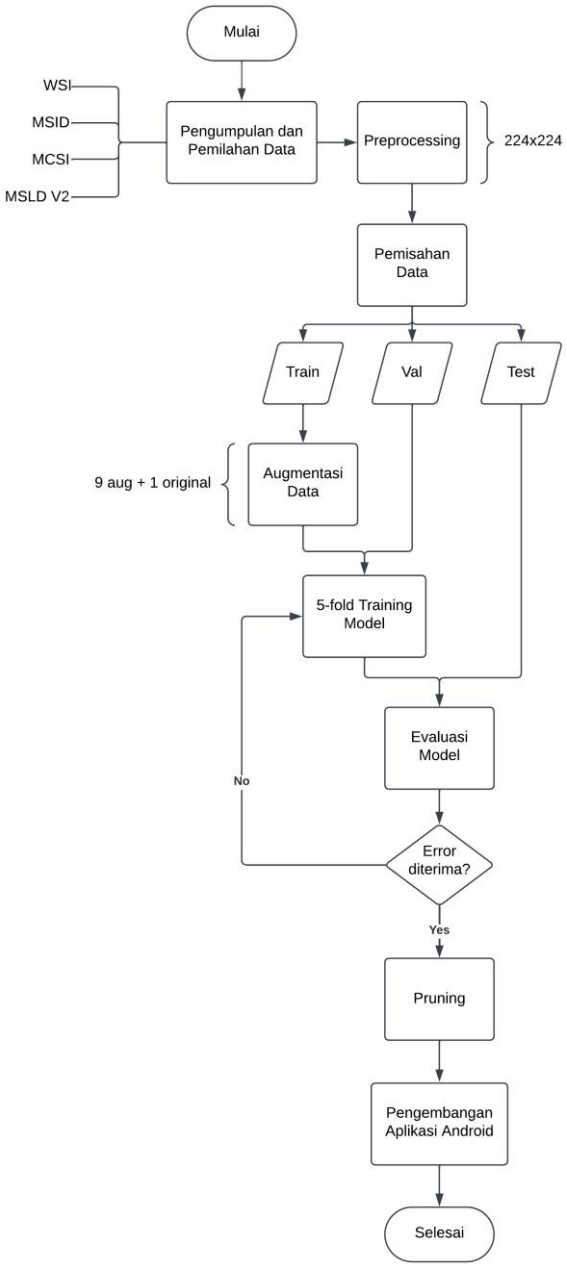

Figure 1. Prosedur Penelitian

Penelitian selanjutnya oleh [27] menghasilkan set data *Monkeypox Skin Images Dataset* (MSID) yang di dalamnya berjumlah 770 dan terdiri atas 4 jenis gambar kulit yang berbeda, yaitu set data *chickenpox* 107 buah, *measles* 91 buah, *monkeypox* 279 buah, dan normal 293 buah.

Penelitian berikutnya oleh [28] menghasilkan set data *Mpox Close Skin Images* (MCSI) yang di dalamnya berjumlah 400 dan terdiri atas 4 jenis gambar kulit yang berbeda, yaitu set data *acne* 100 buah, *chickenpox* 100 buah, *monkeypox* 100 buah, dan normal 100 buah.

Penelitian terakhir oleh [15] menghasilkan set data *Monkeypox Skin Lesion Dataset v2* (MSLD v2) yang di dalamnya berjumlah 755 dan terdiri atas 6 jenis gambar kulit yang berbeda, yaitu set data

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

monkeypox 284 buah, *chickenpox* 75 buah, *measles* 55 buah, *cowpox* 66 buah, *Hand Foot Mouth Disease* (HFMD) 161 buah, dan *healthy* 114 buah.

2.2. *Preprocessing* dan Augmentasi Data

Pada tahap ini, data akan dikelompokkan berdasarkan enam kelas berbeda. Setelah itu setiap data akan melewati tahap penyesuaian rasio dan ukuran gambar. Rasio yang digunakan yaitu 1:1 dengan ukuran 224x224 piksel. Kemudian, data akan dipisahkan menjadi tiga bagian yaitu *training*, *validation*, dan *testing*. Data *training* dan *validation* digunakan selama proses pelatihan berlangsung, sedangkan data *testing* digunakan setelah pelatihan selesai untuk mengevaluasi hasil pelatihan. Perbandingan ukuran data antara ketiganya masing-masing sekitar 7:2:1, dengan mempertimbangkan jumlah data dapat dibagi oleh ukuran *batch* pelatihan yaitu 16.

Setelah rangkaian *preprocessing* tersebut, augmentasi dilakukan terhadap data *training*. Proses ini dilakukan untuk mempercepat masa pelatihan dan meningkatkan generalisasi terhadap gangguan yang mungkin timbul pada tahap evaluasi ataupun implementasinya. Augmentasi terdiri atas sembilan jenis modifikasi dan ditambah satu data asli, sehingga hasil akhir data *training* akan menghasilkan jumlah sepuluh kali lipat dari asalnya. Teknik yang dipakai pada augmentasi data ini meliputi manipulasi gambar seperti *flipping*, *rotation*, *scaling ratio*, *noise injection*, *color space*, *contrast*, *sharpening*, *translation*, dan *cropping*.

2.3. Pelatihan dan Evaluasi Model

Pada tahap ini, pelatihan akan dilakukan sebanyak lima kali dengan pembagian data secara acak dalam setiap pelatihan. Data *validation* akan digunakan untuk menguji model dalam setiap iterasi/*epoch*. Hasil pengujian tersebut menentukan apakah pelatihan dilanjutkan ke *epoch* selanjutnya atau dihentikan secara paksa.

Table 1. Struktur ResNet-50

Layer	*Output*	50 Layer ResNet
Conv1	112×112	7×7, 64 *stride* 2
		3×3 *max pooling*, stride 2
Conv2.x	56×56	$\begin{bmatrix} 1 \times 1 & 64 \\ 3 \times 3 & 64 \\ 1 \times 1 & 256 \end{bmatrix} \times 3$
Conv3.x	28×28	$\begin{bmatrix} 1 \times 1 & 128 \\ 3 \times 3 & 128 \\ 1 \times 1 & 512 \end{bmatrix} \times 4$
Conv4.x	14×14	$\begin{bmatrix} 1 \times 1 & 256 \\ 3 \times 3 & 256 \\ 1 \times 1 & 1024 \end{bmatrix} \times 6$
Conv5.x	7×7	$\begin{bmatrix} 1 \times 1 & 512 \\ 3 \times 3 & 512 \\ 1 \times 1 & 2048 \end{bmatrix} \times 3$
fc	1×1	*Average pooling*, 1000-d fc, *softmax*
FLOPs		3.8×10^9

Model ResNet-50 pada penelitian ini merupakan perkembangan dari model *Visual Geometry Group* (VGG). Blok *convolutional* dari model ini terdiri dari tiga layer *convolutional*, yaitu satu layer berukuran 3x3 dan dua layer ukuran 1x1 yang disimpan sebelum dan setelahnya. Operasi BN dan fungsi aktivasi ReLU dijalankan setelah setiap layer *convolutional*. Selain itu, pada bagian akhir terdapat layer

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

fully connected atau dikenal juga dengan *dense layer*, kemudian fungsi *softmax* dijalankan terakhir sebelum menghasilkan *output* untuk penyelesaian masalah klasifikasi. Struktur ResNet-50 secara lebih lengkap dipaparkan pada Table 1 [29].

Evaluasi dilakukan setiap proses pelatihan selesai. Evaluasi tersebut berdasarkan pada confusion matrix untuk menentukan nilai accuracy, precision, dan recall. Satu model dengan nilai akurasi tertinggi akan digunakan sebagai input untuk proses pruning. Jika hasil evaluasi masih kurang memuaskan maka proses pelatihan akan diulang dengan melakukan beberapa modifikasi pada model [30].

2.4. *Pruning*

Pada tahap ini, metode *pruning* Global MP akan diterapkan pada model secara bertahap. Model akan mengalami pemangkasan dengan *threshold* yang kecil. Setelah itu, *fine tuning* akan dilakukan dengan melatih ulang model menggunakan data awal yang sesuai. Proses ini diulang seterusnya sampai penurunan akurasi yang signifikan terjadi. Setelah itu model akan melakukan proses *quantization* dengan metode QAT untuk optimalisasi pada perangkat *mobile* android.

Pruning dapat dikelompokkan menjadi dua kategori, *pruning* terstruktur dan tidak terstruktur. *Pruning* tidak terstruktur bertujuan untuk menghapus neuron yang redundan tanpa struktur tertentu/acak. Namun, hal ini dapat menghasilkan operasi matriks yang rumit yang sulit dipercepat menggunakan GPU. Untuk mengatasi ini, *pruning* terstruktur menghilangkan kelompok neuron yang redundan secara terstruktur sehingga operasi matriks masih dapat dilakukan dengan efisien, seperti menargetkan neuron pada layer tertentu [31].

Global Magnitude Pruning (Global MP) merupakan salah satu metode *pruning* terstruktur. Metode ini bekerja dengan mengurutkan seluruh bobot dari model yang menjadi target *pruning* berdasarkan nilai absolutnya. Kemudian melalui suatu minimum *threshold* bobot yang bersesuaian akan dipangkas. Operasi ini akan menghilangkan neuron dengan bobot mendekati nol. Setelah itu, *fine tuning* dapat dilakukan untuk mengembalikan model mendekati kondisi awal dan operasi Global MP dapat diulang sesuai kebutuhan [32].

2.5. Pengembangan Aplikasi Android

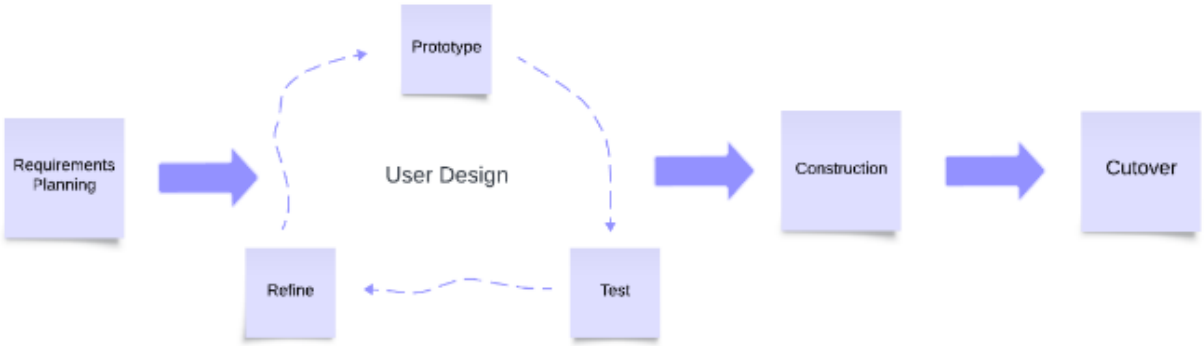

Figure 2. Tahapan RAD

Pada tahap ini, model hasil proses *pruning* akan diimplementasikan pada aplikasi android menggunakan Flutter. Metode pengembangan yang digunakan yaitu berdasarkan pada RAD seperti Figure 2 dan penggambaran struktur aplikasi dibuat menggunakan standar UML. Setelah itu, aplikasi akan dirilis pada format APK untuk dilakukan percobaan pada perangkat android langsung [33], [34].

3. RESULT

Tahapan hasil dan pembahasan disusun berdasarkan prosedur penelitian yang telah dilakukan sebelumnya.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

3.1. Pengumpulan dan Pemilahan Data

Pengumpulan dan pemilahan data dilakukan berdasarkan empat set data yang sudah tersedia dari penelitian sebelumnya. MSLD v2 menjadi dasar dari penyusunan set data pada penelitian ini dengan alasan waktu perilisan yang paling terbaru. Ketiga set data lainnya berperan sebagai data tambahan terutama pada kelas dengan jumlah data yang sedikit seperti *chickenpox*, *cowpox*, dan *measles*. Pada Table 2 menunjukkan secara detail susunan dari data penelitian. Kelas *smallpox* pada data WSI tidak akan digunakan dengan alasan keterbatasan kualitas data dan juga ketersediaannya pada set data lain sebagai pembanding. Selain itu, beberapa data kelas *acne* juga akan dimasukkan ke dalam kelompok kelas HFMD demi menambah variasi dari kelas tersebut. Bentuk pemilahan dari gambar yaitu dengan melakukan penghapusan duplikasi, *cropping* pada bagian kulit, penghapusan data dengan gejala kurang terlihat, dan penyesuaian terhadap jumlah keseluruhan data.

Table 2. Data Penelitian

Kelas/label	Set Data				Total
	MSLD v2	MCSI	MSID	WSI	
chickenpox	64	91	67	-	**222**
cowpox	54	-	-	40	**94**
HFMD	152	87	-	-	**239**
healthy	104	92	235	-	**431**
measles	52	-	48	-	**100**
monkeypox	238	24	156	-	**418**
Total	**664**	**294**	**506**	**40**	**1504**

3.2. *Preprocessing* dan Augmentasi Data

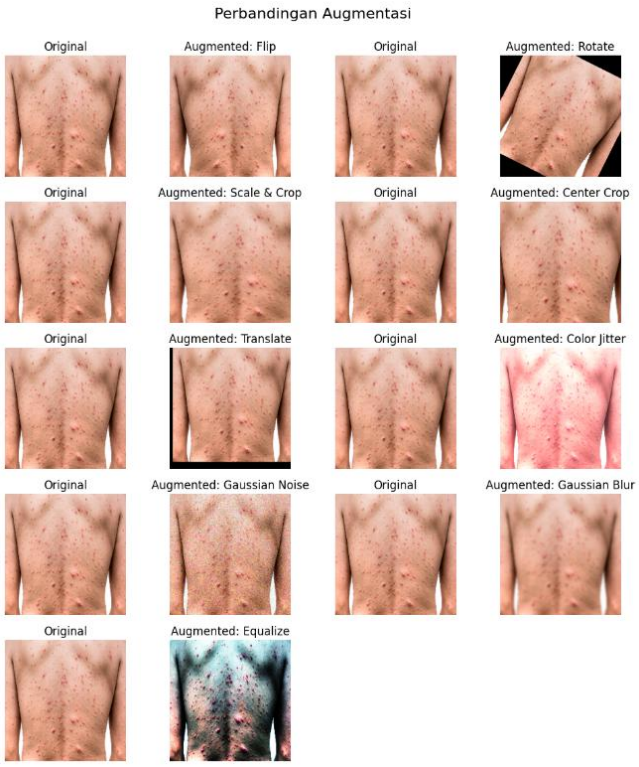

Figure 3. Augmentasi Data

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

Set data yang sudah dipilah pada tahap sebelumnya kemudian dilakukan pembagian menjadi tiga bagian. Pembagian tersebut memperhatikan kelipatan bilangan 16 yang sesuai dengan ukuran *batch* pelatihan. Dari 1504 data dengan pembagian kira-kira 7:2:1 didapatkan 1056 data *training*, 304 data *validation*, dan 144 data *testing*. Untuk mencapai jumlah yang sesuai dengan kelipatan 16 tersebut, pembagian antar kelas diatur secara pasti meskipun sedikit melenceng dari rasio. Data *training* yang berjumlah 1056 kemudian akan dilakukan augmentasi data dengan sembilan macam augmentasi dan satu gambar aslinya. Dengan demikian, data *training* yang digunakan dalam model DL akan berjumlah 10560. Setelah itu, data disamakan ukurannya menjadi 224x224 piksel, pada Figure 3 menunjukkan hasil dari augmentasi.

3.3. Pelatihan dan Evaluasi Model

Setiap bagian data pada tahap sebelumnya kemudian dikelompokkan masing-masing berjumlah 16 gambar yang secara bersamaan dimasukkan pada model. Pelatihan model akan menggunakan *pretrained* model atau model yang sudah dilatih sebelumnya dengan bobot yang berasal dari *ImageNet*. Model dengan bobot awal ini akan lebih cepat beradaptasi terhadap data *training* daripada membangun model dari awal.

Model tersebut memiliki layer *dropout* pada bagian akhir layer *fully connected* dengan nilai probabilitas 50%. Dengan demikian, keluaran dari neuron memiliki kemungkinan 50% untuk tidak aktif selama fase pelatihan. Perhitungan *loss* dari pelatihan model menggunakan *cross entropy* dengan memperhatikan bobot yang berbeda untuk setiap kelas. Kelas dengan data yang tinggi akan mendapat bobot yang lebih rendah dan sebaliknya. Optimalisasi model dilakukan dengan algoritma *AdamW* yang merupakan modifikasi dengan menambahkan regularisasi melalui *weight decay* untuk menghindari *overfitting*. Penjadwalan *learning rate* juga dilakukan dengan metode *cossine annealing*, yaitu penyesuaian *learning rate* berdasarkan kurva kosinus. Table 3 menunjukkan *hyperparameter* pelatihan.

Table 3. *Hyperparameter* Pelatihan

No.	***Hyperparameter***	**Nilai**
1.	*Epoch* Maksimal	100
2.	Ukuran *Batch*	16
3.	*Learning Rate*	8e-6
4.	*Weight Decay*	3e-7
5.	*Learning Rate* Minimal	8e-9
6.	*Patience* (*early stopper*)	10

Pelatihan model DL menghasilkan keluaran model, akurasi, dan *loss*. Figure 4 menunjukkan plot akurasi dan *loss* model pada *fold* terbaik. *Early stopping* dengan *patience* bernilai 10 diterapkan pada pelatihan model. Dengan demikian, *epoch* dari setiap *fold* berbeda-beda tergantung pada perkembangan pelatihan dari setiap *fold*. Jika *loss* tidak menunjukkan penurunan dalam 10 *epoch* terakhir, maka pelatihan akan dihentikan. Bobot model akan disimpan berdasarkan dua kondisi yaitu akurasi dan/atau *loss* terbaik pada fase *validation*.

Setiap model pada lima *fold* kemudian disandingkan melalui data testing. Fase ini akan menentukan akurasi yang sebenarnya dari model terhadap data yang asing dan belum pernah ditemui selama pelatihan. Figure 5 menunjukkan *confusion matrix* dari model terbaik. Diagonal utama pada matriks tersebut menunjukkan prediksi model yang tepat terhadap data testing atau disebut *True Positive* (TP). Kemudian, nilai selain diagonal utama pada satu kolom yang sama menunjukkan kesalahan model dalam memprediksi suatu kelas padahal bukan kelas yang sesuai atau disebut *False Positive* (FP). Selanjutnya, nilai selain diagonal utama pada satu baris yang sama menunjukkan kesalahan model dalam

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

mengenali suatu kelas dengan memprediksi sebagai kelas lainnya atau disebut *False Negative* (FN). Sementara itu, nilai *True Negative* (TN) pada model dengan kelas yang banyak ditentukan dengan menjumlahkan nilai pada matriks selain dari satu baris dan kolom suatu kelas yang akan diuji. Table 4 menunjukkan perhitungan akurasi dari setiap *fold* pelatihan.

Model fold2 dengan kondisi *acc*/akurasi fase *validation* tertinggi menjadi model dengan akurasi terbaik pada tahap testing. Nilai ini dicapai pada *epoch* 12 dari 33 *epoch* yang dilakukan. Model kemudian disimpan untuk menjalani proses *pruning* pada tahap selanjutnya. Berdasarkan *confusion matrix* pada Figure 5, nilai *precision*, *recall*, dan *f1-score* dari model ini yaitu masing-masing 93.42%, 94.63%, dan 93.91%.

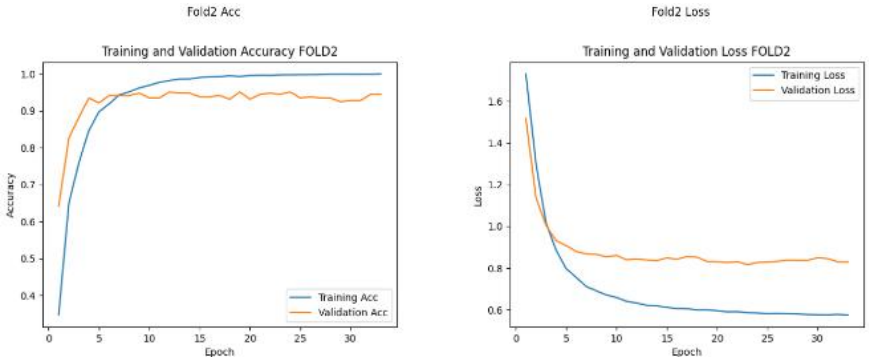

Figure 4. *Plotting Loss* dan Akurasi *Fold* Pelatihan Terbaik

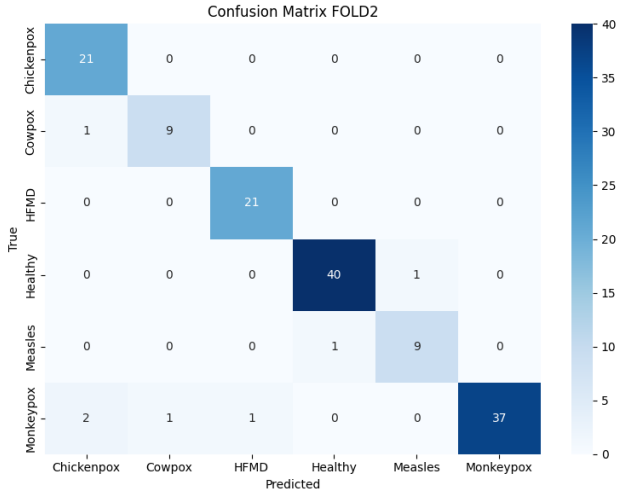

Figure 5. *Confusion Matrix* Model Terbaik

Table 4. Hasil Pelatihan Model Setiap *Fold* Pelatihan

No.	Pelatihan	*Epoch*	Waktu	*Train*		*Val*		Tes Akurasi (%)	
				Acc(%)	*Loss*	*Acc(%)*	*Loss*	*By Acc*	*By Loss*
1	*Fold1*	20	00:58:32	99.10	0.61	91.12	0.88	90.97	90.97
2	***Fold2***	**33**	**01:35:39**	**98.12**	**0.63**	**95.07**	**0.84**	**95.14**	**93.75**
3	*Fold3*	47	02:15:47	97.55	0.65	93.09	0.88	93.75	91.67
4	*Fold4*	43	02:03:31	99.91	0.57	92.11	0.86	88.89	89.58
5	*Fold5*	25	01:12:12	99.33	0.60	91.78	0.89	88.19	87.50

3.4. *Pruning*

Pada tahap ini model *fold* 2 dengan perhitungan berdasarkan *acc* akan dilakukan *pruning*. *Pruning* dengan metode Global MP akan dilakukan dengan menggunakan *library pytorch_prune*. *Threshold*

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

awal yang dipakai dalam proses ini yaitu 0.05 kemudian akan dilakukan *fine-tuning* dengan melakukan pelatihan model kembali seperti pada tahap sebelumnya.

Pruning dilakukan secara bertahap dengan diatur oleh sebuah *looping* sebanyak lima kali. Setiap model mengalami *pruning* pada satu iterasi, model tersebut akan dilatih ulang untuk mengembalikan performa awal sebelum *pruning*. Pelatihan tersebut dilakukan dengan tahapan yang sama pada pelatihan awal model tetapi *hyperparameter* yang lebih besar. Sementara itu, nilai probabilitas *importance* yang digunakan yaitu 1 dan bersifat global. Hal ini berarti bobot model dari semua layer akan diurutkan berdasarkan nilai absolut dan dipangkas sebanyak 5% berdasarkan rasio *pruning* dengan probabilitas 100%. Table 5 menunjukkan hasil akhir proses *pruning* Global MP.

Table 5. Hasil *Pruning* Model dengan Metode Global MP

No.	Iterasi	Parameter(juta)	FLOPs(miliar)	Penyimpanan(MB)	Akurasi(%)
1.	*Pruning* 1	21.58	7.16	82.623	94.44
2.	*Pruning* 2	19.95	6.21	76.421	90.28
3.	*Pruning* 3	18.48	5.42	70.799	86.81
4.	*Pruning* 4	17.21	4.73	65.967	86.81
5.	*Pruning* 5	15.99	4.14	61.284	82.64

Basis model yang sudah dilatih sebelumnya memiliki parameter 23.52 juta, FLOPs 8.26 miliar, penyimpanan 90.06 MB, dan akurasi 95.14%. Iterasi pertama dengan penurunan parameter 8.25%, FLOPs 13.32%, dan penyimpanan 8.26% menjadi *pruning* dengan akurasi tertinggi. Model tersebut didapat setelah melalui pelatihan ulang dengan maksimal 100 *epoch* kemudian berhenti pada *epoch* 30 melalui *early stopping*. Kemudian untuk memaksimalkan implementasi pada perangkat android, proses *Quantization Aware Training* (QAT) akan dilakukan pada model *Pruning* 1.

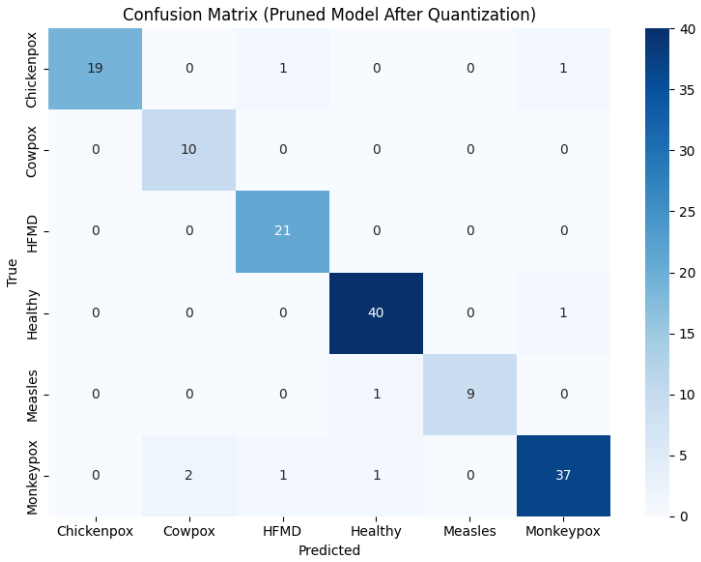

Figure 6. *Confusion Matrix* Model Setelah *Pruning* dan *Quantization*

Proses QAT akan melakukan konversi terhadap bobot dalam model yang sebelumnya bertipe data *float* 32 bit menjadi integer 8 bit. Sebagai konteks, tipe data float32 menyimpan data bersifat *floating point* atau desimal dengan pembagian 1 bit untuk menentukan positif atau negatif, 8 bit untuk menentukan eksponensial, dan 23 bit yang menyimpan nilai *mantissa*. Kemudian konfigurasi *default* dalam QAT menggunakan jenis *qnnpack* yang dirancang untuk arsitektur ARM, konfigurasi ini mengombinasikan *symmetric quantization* untuk bobot dan *asymmetric quantization* untuk aktivasi.

Akurasi dari model akan berkurang setelah melalui proses *quantization*. Oleh karena itu, jenis *quantization* dengan tambahan pelatihan/QAT menjadi opsi terbaik untuk memaksa model terbiasa dengan proses konversi. Selain itu dalam jenis *quantization* ini, model mengalami modifikasi terlebih dahulu dengan tambahan *observer* yang disebut dengan *fake quantization*. Dengan demikian, pelatihan dalam tahap ini berjalan lebih lambat karena terdapat layer tambahan yang menyimpan nilai terbesar dan terkecil dalam setiap layer. Nilai-nilai tersebut nantinya akan digunakan untuk mereduksi jangkauan dari bobot, sehingga bobot akhir setelah konversi akan berjenis int8.

Pada Figure 6 menunjukkan *confussion matrix* dari model akhir setelah melewati proses *pruning* dan *quantization*. Nilai akurasi, *precision*, *recall*, dan *f1-score* masing-masing adalah 94.44%, 94.12%, 94.71%, dan 94.16%. Sementara itu, *pruning* dan QAT mengurangi ukuran penyimpanan model menjadi 20.993 MB.

3.5. Pengembangan Aplikasi Android

Aplikasi *mobile* pendeteksi cacar monyet yang diberi nama *Classier: Monkeypox Classifier*, dikembangkan berdasarkan metode RAD.

3.5.1. *Requirements Planning*

Tahap *Requirements Planning* dilakukan dengan analisis mandiri terhadap segala kebutuhan dari pengguna terhadap sistem. Analisis tersebut didasarkan pada kemungkinan masalah pengguna yang dapat diselesaikan dengan sistem. Kebutuhan tersebut yaitu sebagai berikut,

1. Pengguna dapat melakukan masukan gambar yang bersumber dari kamera, media penyimpanan, dan *request* URL.
2. Pengguna dapat mengetahui jenis penyakit/kelas yang didukung oleh model DL pada aplikasi *Classier: Monkeypox Classifier*.
3. Pengguna dapat mengetahui hasil klasifikasi dari model DL beserta nilai persentasenya pada aplikasi *Classier: Monkeypox Classifier*.
4. Pengguna dapat menggunakan aplikasi *Classier: Monkeypox Classifier* tanpa terhubung dalam jaringan.
5. Pengguna dapat menyimpan riwayat hasil klasifikasi pada aplikasi *Classier: Monkeypox Classifier*.

3.5.2. *User Design*

Struktur statis dari aplikasi *Classier: Monkeypox Classifier* dapat ditunjukkan pada Figure 7. Aplikasi ini memiliki tujuh kelas yang menggambarkan obyek aplikasi. Kelas pertama autentikasi berhubungan dengan utilitas pengguna saat *login* ke dalam aplikasi. Kelas kedua terdapat pengguna yang menyimpan informasi pengguna yang sudah *login* ataupun menyediakan nilai dasar kosong pada pengguna yang tidak *login*. Kelas ketiga terdapat riwayat yang memuat riwayat klasifikasi yang dilakukan pengguna setelah *login*. Kelas keempat terdapat hasil yang memuat hasil klasifikasi dari gambar yang dimasukkan pengguna dengan berisi informasi jenis penyakit, persentase klasifikasi, dan waktu eksekusi model DL. Kelas kelima terdapat media yang memuat fungsi-fungsi penyedia masukan gambar dari pengguna. Kelas keenam terdapat model yang menyimpan model klasifikasi dan fungsi-fungsi terkaitnya. Kemudian kelas ketujuh terdapat kelas yang memuat jenis penyakit atau kelas yang didukung oleh model DL.

Pada Figure 8 dapat diketahui bahwa aplikasi *Classier: Monkeypox Classifier* memiliki tiga fungsi utama yang dapat diakses oleh pengguna. Fungsi pertama yakni klasifikasi gambar, sebelumnya pengguna diharuskan menyediakan gambar terlebih dahulu melalui salah satu dari tiga sumber yang disediakan yaitu, kamera, media penyimpanan, dan URL. Setelah itu, pengguna juga diharuskan melakukan *preprocessing* dasar dengan memilih bagian kulit yang akan diklasifikasi dari gambar

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

dengan rasio 1:1. Fungsi kedua terdapat jenis penyakit yang didukung oleh model DL, aplikasi akan menyediakan contoh gambar dan penjelasan melalui *hyperlink* pencarian Google. Fungsi ketiga yaitu mengakses riwayat klasifikasi, pengguna diharuskan *login* terlebih dahulu melalui *plugin* Google. Setelah itu, pengguna dapat mengelola riwayat klasifikasi yang didapatkan dari proses klasifikasi gambar setelah *login*.

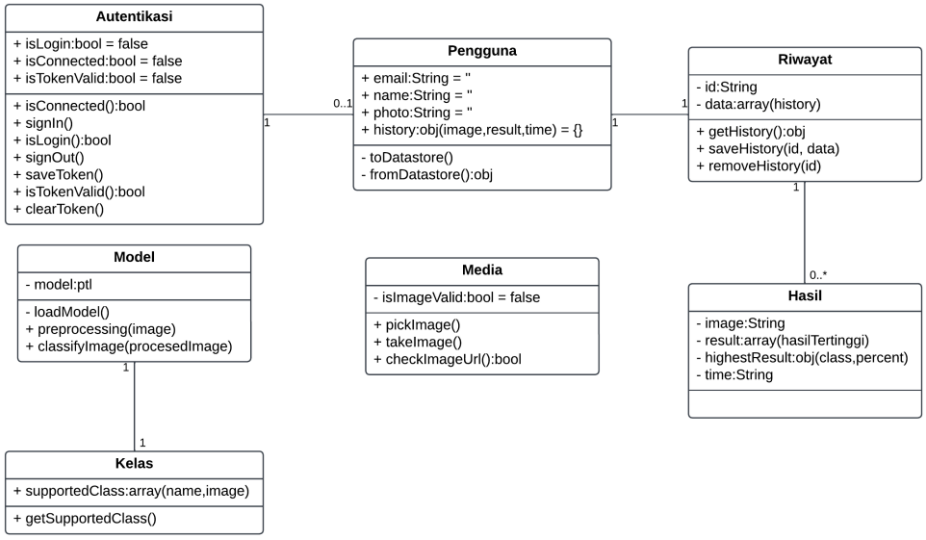

Figure 7. *Class Diagram*

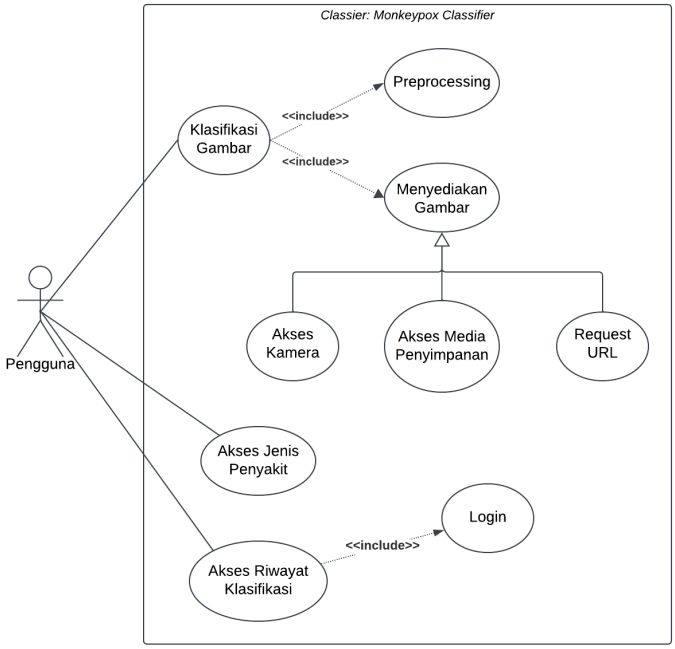

Figure 8. *Use Case Diagram*

3.5.3. *Construction*

Tampilan utama dari aplikasi *Classier: Monkeypox Classifier* seperti pada Figure 9, terdapat empat menu utama yaitu jenis penyakit yang didukung, klasifikasi gambar melalui akses kamera, klasifikasi gambar melalui *request* URL, dan klasifikasi gambar melalui akses media penyimpanan.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

Figure 9. Tampilan Utama Aplikasi

Pertama, klasifikasi gambar melalui akses kamera dapat dilakukan dengan klik menu kamera kemudian aplikasi akan memberikan tampilan kamera perangkat bawaan dan pengguna dapat melakukan penangkapan gambar. Jika menu dibuka pertama kali maka aplikasi akan meminta izin akses kamera. Contoh tangkapan kamera pada aplikasi ini seperti pada Figure 10, pengguna dapat mengkonfirmasi penggunaan gambar tersebut atau mengambil ulang.

Figure 10. Tampilan Akses Kamera

Kedua, pengguna dapat melakukan klasifikasi gambar melalui *request* URL dengan klik menu *web image*, aplikasi akan memberikan *form* yang berisikan *input* URL dan tombol cari. Kemudian aplikasi akan melakukan pemeriksaan apakah URL yang dimasukkan memuat format gambar yang sesuai dan dapat digunakan dalam klasifikasi. Contoh pemeriksaan tersebut seperti pada Figure 11. Jika URL sudah sesuai, pengguna dapat meneruskan gambar yang dimaksud ke dalam model DL, sedangkan jika gambar tidak sesuai maka pesan *error* akan ditampilkan.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

Figure 11. Tampilan *Request* URL

Ketiga, klasifikasi gambar melalui media penyimpanan dapat dilakukan dengan klik tombol galeri. Kemudian pengguna dapat memilih satu gambar yang akan dilakukan klasifikasi. Sumber media yang didukung oleh aplikasi yaitu seperti penyimpanan internal, eksternal, maupun *cloud*. Contoh pemilihan gambar melalui media internal seperti pada Figure 12.

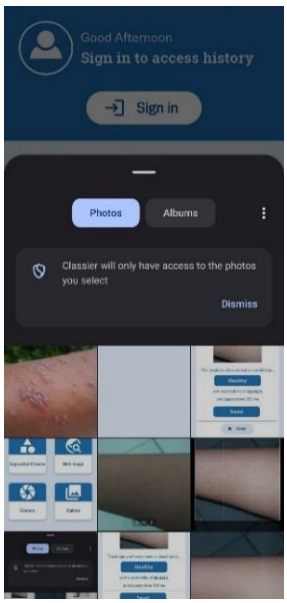

Figure 12. Tampilan Akses Media Penyimpanan

Setelah memilih salah satu dari tiga cara yang sudah dibahas, pengguna dapat melanjutkan proses klasifikasi gambar pada tahap *preprocessing*. *Preprocessing* berupa *cropping* gambar ke bidang 1:1 dilakukan dengan *library image cropper*. Pengguna harus melakukan seleksi bidang gambar yang benar-benar menunjukkan bagian kulit untuk meningkatkan kualitas klasifikasi gambar. Pada Figure 13 menunjukkan tampilan *cropping* yang bisa diatur pengguna dengan melakukan *zoom* in ke target klasifikasi.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

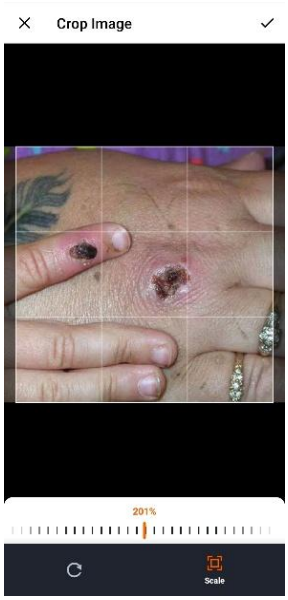

Figure 13. Tampilan *Cropping*

Hasil dari *preprocessing* tersebut akan dijadikan masukan untuk model DL. Kemudian keluaran dari klasifikasi tersebut akan ditunjukkan pada tampilan hasil seperti Figure 14. Hasil klasifikasi yang didapat menunjukkan kelas *monkeypox*/cacar monyet sesuai dengan label gambarnya dengan probabilitas 78.106%. Klasifikasi seluruhnya juga ditampilkan pada detail klasifikasi dengan kemungkinan kedua yaitu *cowpox*/cacar sapi.

Figure 14. Tampilan Hasil Klasifikasi

Akses jenis penyakit disediakan pada tampilan utama dengan melakukan klik pada menu *supported class*/kelas yang didukung. Seperti pada Figure 15, aplikasi menyediakan contoh gambar beserta labelnya. Penjelasan dari gambar tersebut dapat diakses pengguna dengan klik salah satu dari gambar, lalu pengguna akan diarahkan ke pencarian Google dengan *query* yang sesuai dengan gambar tersebut. *Library* yang digunakan pada bagian penjelasan tersebut yaitu URL *launcher*, *library* ini memungkinkan pengguna mengakses browser bawaan dengan URL yang sesuai tetapi masih di dalam aplikasi *Classier: Monkeypox Classifier*.

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

Figure 15. Tampilan Jenis Penyakit

Pengguna dapat mengakses riwayat klasifikasi setelah melakukan *login* terlebih dahulu melalui *plugin* Google. Riwayat klasifikasi menyediakan informasi klasifikasi gambar yang sebelumnya sudah dilakukan setelah *login* beserta waktu pengambilan gambar klasifikasinya. Seperti pada Figure 16, percobaan-percobaan sebelumnya dilakukan ulang setelah *login* dan hasilnya akan tersimpan pada tampilan riwayat ini.

Figure 16. Tampilan Riwayat Klasifikasi

3.5.4. *Cutover*

Penyelesaian dari aplikasi *Classier: Monkeypox Classifier* akan dilakukan perilisan format APK dan pengujian dalam perangkat android langsung. Perilisan APK pada Flutter dapat dilakukan dengan menjalankan perintah *flutter build apk –release –split-per-abi*. Perintah ini akan menghasilkan tiga keluaran untuk arsitektur android yang berbeda seperti ARM 32-bit, ARM 64-bit, dan x86 64-bit. Pengujian akan dilakukan pada perangkat *smartphone* dengan seri Samsung M23 5G dengan arsitektur ARM64 menggunakan metode *blackbox* testing. Hasil pengujian ditunjukkan pada Table 6.

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

Table 6. Hasil Pengujian Aplikasi

No.	Menu	Deskripsi Pengujian	Hasil Yang Diharapkan	Hasil Pengujian
1	Klasifikasi Gambar	Pengguna melakukan klasifikasi gambar melalui akses kamera.	Aplikasi menunjukkan tampilan hasil klasifikasi yang memuat gambar, hasil klasifikasi tertinggi, hasil klasifikasi lengkap, dan waktu eksekusi.	Valid
2	Klasifikasi Gambar	Pengguna melakukan klasifikasi gambar melalui akses media penyimpanan.	Aplikasi menunjukkan tampilan hasil klasifikasi yang memuat gambar, hasil klasifikasi tertinggi, hasil klasifikasi lengkap, dan waktu eksekusi.	Valid
3	Klasifikasi Gambar	Pengguna melakukan klasifikasi gambar melalui *request* URL.	Aplikasi menunjukkan gambar yang termuat dalam URL, kemudian pengguna dapat melakukan konfirmasi proses klasifikasi.	Valid
4	Jenis Penyakit	Pengguna mengakses jenis penyakit/kelas yang didukung.	Aplikasi menampilkan enam daftar penyakit/kelas yang didukung oleh model DL.	Valid
5	Jenis Penyakit	Pengguna mengakses *hyperlink* penjelasan jenis penyakit/kelas yang didukung.	Aplikasi menampilkan informasi seputar kelas yang dimaksud melalui pencarian Google.	Valid
6	Riwayat Klasifikasi	Pengguna melakukan *login* ke dalam aplikasi menggunakan *plugin* Google.	Aplikasi menampilkan informasi pengguna seperti nama lengkap dan foto profil pada tampilan utama.	Valid
7	Riwayat Klasifikasi	Pengguna melakukan klasifikasi gambar setelah *login*.	Aplikasi akan menyimpan hasil klasifikasi secara otomatis pada riwayat klasifikasi pengguna dan menampilkan *snackbar* notifikasi.	Valid
8	Riwayat Klasifikasi	Pengguna mengakses riwayat klasifikasi setelah *login*.	Aplikasi menampilkan seluruh riwayat klasifikasi yang dilakukan pengguna setelah *login*.	Valid
9	Riwayat Klasifikasi	Pengguna menggeser ke kiri/kanan salah satu riwayat klasifikasi setelah *login*.	Aplikasi akan menghapus riwayat klasifikasi pengguna dan menampilkan *snackbar* notifikasi.	Valid
10	Kesalahan	Pengguna mematikan koneksi internet pada saat berinteraksi dengan aplikasi setelah *login*.	Aplikasi menampilkan dialog notifikasi *reload* halaman atau *logout* untuk melanjutkan operasi.	Valid

4. DISCUSSIONS

Pada penelitian sebelumnya yang menghasilkan set data MSLD v2, akurasi tertinggi didapat oleh model DenseNet-121 sebesar 82.26%. Sementara itu, penelitian ini menambahkan data dari penelitian lainnya seperti set data WSI, MSID, dan MCSI serta melakukan pemilahan terhadap kombinasi data. Hasil akurasi yang didapat mencapai 95.14%. Tidak hanya itu, *pruning* dan *quantization* juga berhasil diterapkan dengan baik hingga ukuran model berkurang sekitar 76.69% menjadi 20.993 MB dan parameter yang berkurang sekitar 8.25% menjadi 21.58 juta. Akurasi akhir hanya mengalami penurunan ±0.7% menjadi 94.44% dan dapat diterapkan pada perangkat android dengan lebih efisien. Selain itu, terdapat keterbatasan dari model yang sudah dilatih pada penelitian ini, di antaranya yaitu, belum mampu membedakan dengan baik antara obyek klasifikasi dengan *noise* berupa latar belakang atau obyek lain yang ikut termuat dalam gambar dan terjadi pembiasan hasil menjadi kelas HFMD untuk obyek klasifikasi yang menunjukkan bagian telapak tangan dikarenakan keterbatasan data yang memperlihatkan telapak tangan pada kelas lain selain HFMD yang kurang bahkan tidak ada.

5. CONCLUSION

Model ResNet-50 yang digunakan dalam penelitian ini dilatih menggunakan kombinasi dari set data penelitian sebelumnya yaitu, WSI, MSID, MCSI, dan MSLD v2. Kombinasi dari set data tersebut menghasilkan data sejumlah 1504 buah dengan 6 kelas yang berbeda meliputi *chickenpox*, *cowpox*, HFMD, *healthy*, *measles*, dan *monkeypox*. Nilai akurasi, *precision*, *recall*, dan *f1-score* awal model masing-masing yaitu, 95.14%, 93.42%, 94.63%, dan 93.91%.

Model *pruned* ResNet-50 sendiri merupakan modifikasi dari basis model ResNet-50 dengan bobot awal *ImageNet*. Model tersebut melewati tahap *pruning* dengan metode Global MP dan *quantization* dengan metode QAT. Sementara itu, pengurangan parameter dan FLOPs yang didapat yaitu sekitar 10% menjadi 21.58 juta parameter dan 7.16 miliar FLOPs. Selain itu, ukuran penyimpanan dari model sendiri berkurang 76.69% menjadi 20.993 MB dan disimpan dalam format *.ptl* (PyTorch *Lite*). Di samping itu, nilai akurasi, *precision*, *recall*, dan f1-score akhir model hanya mengalami pengurangan minor masing-masing yaitu, 94.44%, 94.12%, 94.71%, dan 94.16%.

Aplikasi *Classier: Monkeypox Classifier* berhasil mengimplementasikan model *pruned* ResNet-50 pada perangkat *mobile* android dengan menggunakan *framework* Flutter. Menu klasifikasi, jenis penyakit, riwayat klasifikasi, dan eksepsi kesalahan dapat berfungsi dengan baik setelah melewati pengujian *blackbox*. Saran pengembangan selanjutnya penelitian ini yaitu menerapkan model DL lain yang mampu melakukan deteksi obyek seperti YOLO. Model tersebut akan mengatur fase *preprocessing* dengan mendeteksi target klasifikasi dan melakukan *cropping*. Hal ini menjadi perhatian karena model klasifikasi seperti ResNet-50 masih belum mampu beradaptasi membedakan obyek klasifikasi dengan obyek lain di sekitarnya.

CONFLICT OF INTEREST

Para penulis dengan ini menyatakan bahwa tidak terdapat konflik kepentingan, baik di antara para penulis maupun dengan objek penelitian yang dimasukkan dalam artikel ini.

ACKNOWLEDGEMENT

Terima kasih kepada peneliti-peneliti sebelumnya [15], [26]-[28] yang telah menyediakan akses secara publik terhadap data penelitian seperti WSI, MSID, MCSI, dan MSLD v2. Karena dengan adanya data-data tersebut, penelitian ini dapat terlaksana dan mampu memberikan perkembangan penelitian dengan topik terkait cacar monyet.

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

REFERENCES

[1] X. Peng *et al.*, "Perceptions and worries about monkeypox, and attitudes towards monkeypox vaccination among medical workers in China: A cross-sectional survey," *J Infect Public Health*, vol. 16, no. 3, 2023, doi: 10.1016/j.jiph.2023.01.010.

[2] G. Kampf, "Efficacy of heat against the vaccinia virus, variola virus and monkeypox virus," 2022. doi: 10.1016/j.jhin.2022.06.008.

[3] Y. Şahin, H. Yüce, S. Ünüvar, and O. Çiftçi, "Current Pandemic in the World: Monkeypox from Past to Present," *An Acad Bras Cienc*, vol. 95, no. 1, 2023, doi: 10.1590/0001-3765202320220767.

[4] L. Budiyarto, A. A. Sabila, and H. C. Putri, "Infeksi Cacar Monyet (Monkeypox)," *Jurnal Medika Hutama*, vol. 4, no. 2, pp. 3225–3236, 2023, [Online]. Available: http://jurnalmedikahutama.com

[5] S. Surati, H. Trivedi, B. Shrimali, C. Bhatt, and C. M. Travieso-González, "An Enhanced Diagnosis of Monkeypox Disease Using Deep Learning and a Novel Attention Model Senet on Diversified Dataset," *Multimodal Technologies and Interaction*, vol. 7, no. 8, 2023, doi: 10.3390/mti7080075.

[6] O. Mitjà *et al.*, "Monkeypox," *The Lancet*, vol. 401, no. 10370, pp. 60–74, Jan. 2023, doi: 10.1016/S0140-6736(22)02075-X.

[7] R. Doshi, K. K. Hiran, R. K. Jain, and K. Lakhwani, *Machine Learning: Master Supervised and Unsupervised Learning Algorithms with Real Examples*, 1st ed. BPB Publications, 2022.

[8] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput Sci Rev*, vol. 40, 2021, doi: 10.1016/j.cosrev.2021.100379.

[9] J. Zhang, X. Yu, X. Lei, and C. Wu, "A Novel Deep LeNet-5 Convolutional Neural Network Model for Image Recognition," *Computer Science and Information Systems*, vol. 19, no. 3, 2022, doi: 10.2298/CSIS220120036Z.

[10] N. Ketkar and J. Moolayil, *Deep learning with python: Learn Best Practices of Deep Learning Models with PyTorch*. 2021. doi: 10.1007/978-1-4842-5364-9.

[11] A. Mumuni and F. Mumuni, "Data augmentation: A comprehensive survey of modern approaches," 2022. doi: 10.1016/j.array.2022.100258.

[12] R. L. Kumar, J. Kakarla, B. V. Isunuri, and M. Singh, "Multi-class brain tumor classification using residual network and global average pooling," *Multimed Tools Appl*, vol. 80, no. 9, 2021, doi: 10.1007/s11042-020-10335-4.

[13] S. A. Agrawal, V. D. Rewaskar, R. A. Agrawal, S. S. Chaudhari, Y. Patil, and N. S. Agrawal, "Advancements in NSFW Content Detection: A Comprehensive Review of ResNet-50 Based Approaches," *Original Research Paper International Journal of Intelligent Systems and Applications in Engineering IJISAE*, vol. 11, no. 4, pp. 41–45, 2023, [Online]. Available: www.ijisae.org

[14] E. H. I. Eliwa, A. M. El Koshiry, T. Abd El-Hafeez, and H. M. Farghaly, "Utilizing convolutional neural networks to classify monkeypox skin lesions," *Sci Rep*, vol. 13, no. 1, 2023, doi: 10.1038/s41598-023-41545-z.

[15] S. N. Ali *et al.*, "A web-based mpox skin lesion detection system using state-of-the-art deep learning models considering racial diversity," *Biomed Signal Process Control*, vol. 98, 2024, doi: 10.1016/j.bspc.2024.106742.

[16] S. Saha, T. Chakraborty, R. Bin Sulaiman, and T. Paul, "A Comparative Analysis of CNN-Based Pretrained Models for the Detection and Prediction of Monkeypox," *arXiv:2302.10277*, 2023.

[17] L. Oneto, S. Ridella, and D. Anguita, "Do we really need a new theory to understand over-parameterization?," *Neurocomputing*, vol. 543, 2023, doi: 10.1016/j.neucom.2023.126227.

[18] S. Li *et al.*, "Searching for Fast Model Families on Datacenter Accelerators," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021. doi: 10.1109/CVPR46437.2021.00799.

[19] A. M. Almars, "DeepGenMon: A Novel Framework for Monkeypox Classification Integrating Lightweight Attention-Based Deep Learning and a Genetic Algorithm," *Diagnostics*, vol. 15, no. 2, Jan. 2025, doi: 10.3390/diagnostics15020130.

Vol. 6, No. 4, August 2025, Page. 2434-2452
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.5232

[20] C. Morikawa *et al.*, "Image and video processing on mobile devices: a survey," *Visual Computer*, vol. 37, no. 12, 2021, doi: 10.1007/s00371-021-02200-8.
[21] A. Tashildar, N. Shah, R. Gala, T. Giri, and P. Chavhan, "Application Development Using Flutter," *International Research Journal of Modernization in Engineering Technology and Science @International Research Journal of Modernization in Engineering*, vol. 2, no. 8, 2020.
[22] F. J. Dihan *et al.*, "MpoxSLDNet: A Novel CNN Model for Detecting Monkeypox Lesions and Performance Comparison with Pre-trained Models," *arXiv:2405.21016v1*, May 2024, [Online]. Available: http://arxiv.org/abs/2405.21016
[23] Statista, "Number of smartphone users worldwide from 2014 to 2029," Statista.com. Accessed: Oct. 25, 2024. [Online]. Available: https://www.statista.com/forecasts/1143723/smartphone-users-in-the-world
[24] E. Windmill, *Flutter in Action*, 1st ed. Manning, 2019.
[25] P. E. N. Taruno, G. S. Nugraha, R. Dwiyansaputra, and F. Bimantoro, "Monkeypox Classification based on Skin Images using CNN: EfficientNet-B0," in *E3S Web of Conferences*, 2023. doi: 10.1051/e3sconf/202346502031.
[26] T. Islam, M. A. Hussain, F. U. H. Chowdhury, and B. M. R. Islam, "A Web-scraped Skin Image Database of Monkeypox, Chickenpox, Smallpox, Cowpox, and Measles," *bioRxiv*, 2022, doi: 10.1101/2022.08.01.502199.
[27] D. Bala *et al.*, "MonkeyNet: A robust deep convolutional neural network for monkeypox disease detection and classification," *Neural Networks*, vol. 161, 2023, doi: 10.1016/j.neunet.2023.02.022.
[28] M. G. Campana, M. Colussi, F. Delmastro, S. Mascetti, and E. Pagani, "A Transfer Learning and Explainable Solution to Detect mpox from Smartphones images," *Pervasive Mob Comput*, vol. 98, 2024, doi: 10.1016/j.pmcj.2023.101874.
[29] L. Zhang, Y. Bian, P. Jiang, and F. Zhang, "A Transfer Residual Neural Network Based on ResNet-50 for Detection of Steel Surface Defects," *Applied Sciences (Switzerland)*, vol. 13, no. 9, May 2023, doi: 10.3390/app13095260.
[30] D. Valero-Carreras, J. Alcaraz, and M. Landete, "Comparing two SVM models through different metrics based on the confusion matrix," *Comput Oper Res*, vol. 152, 2023, doi: 10.1016/j.cor.2022.106131.
[31] J. Chong, M. Gupta, and L. Chen, "Resource Efficient Neural Networks Using Hessian Based Pruning," Jun. 2023, [Online]. Available: http://arxiv.org/abs/2306.07030
[32] M. Gupta *et al.*, "Is Complexity Required for Neural Network Pruning? A Case Study on Global Magnitude Pruning," *ArXiv*, Jan. 2024, [Online]. Available: http://arxiv.org/abs/2209.14624
[33] L. A. Flowers, "Testing educational digital games," *Commun ACM*, vol. 64, no. 9, pp. 38–40, Sep. 2021, doi: 10.1145/3450758.
[34] J. Martin, *Rapid application development*, 3rd ed. New York : Macmillan Pub. Co. ; Toronto : Collier Macmillan Canada ; New York : Maxwell Macmillan International, 1991.