Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.3, June 2025, pp. 1006~1012
e-ISSN: 2548-6861

Application of Convolutional Neural Network (CNN) Algorithm with ResNet-101 Architecture for Monkey Pox Detection in Human

Al Fathir Rizal Januar [1]*, Jamaludin Indra [2]*, Dwi Sulistya Kusumaningrum [3]*, Sutan Faisal [4]*
* Teknik Informatika, Universitas Buana Perjuangan Karawang
if21.aljanuar@mhs.ubpkarawang.ac.id [1], jamaludin.indra@ubpkarawang.ac.id [2], dwi.sulistya@ubpkarawang.ac.id [3], sutan.faisal@ubpkarawang.ac.id [4]

Article Info

Article history:

Received 2025-05-06
Revised 2025-06-15
Accepted 2025-06-17

Keyword:

CNN,
Detection,
Monkey Pox,
ResNet-101.

ABSTRACT

Monkeypox is a zoonotic disease that has spread to various countries, including Indonesia. It is transmitted through direct contact with skin lesions, respiratory droplets, or contaminated objects. Early and accurate detection is crucial to reduce the risk of transmission and improve treatment effectiveness. This study aims to detect monkeypox using a Convolutional Neural Network (CNN) with the ResNet-101 architecture. The pre-processing steps include normalization and resizing of images to 224×224 pixels. The model is trained using the Adam optimizer, categorical crossentropy loss function, and an adaptive learning rate reduction. Evaluation results show that the model achieved an accuracy of 94%, with a precision of 0.92, recall of 0.92, and an F1-score of 0.92. The model is capable of classifying images effectively, although some misclassifications still occur. This system is intended to function as an initial image-based screening tool, but its results should be confirmed through clinical diagnosis and laboratory testing to ensure accuracy.

I. PENDAHULUAN

Cacar monyet atau *Monkeypox* merupakan penyakit *zoonosis* yang berasal dari *genus Orthopoxvirus,* bagian dari keluarga *Poxviridae* [1]. Virus ini awalnya terindikasi pada monyet yang dikirim dari Singapura ke Denmark pada tahun 1958, sementara kasus infeksi pertama pada manusia tercatat di Kongo pada tahun 1970. Awalnya, penyakit ini hanya ditemukan di wilayah endemis Afrika, tetapi sejak 13 Mei 2022, kasusnya mulai dilaporkan di berbagai negara *non*-endemis [2]. Hingga Juni 2024, tercatat sebanyak 99.176 kasus cacar monyet secara global, sementara di Indonesia per Agustus 2024 terdapat 88 kasus dengan DKI Jakarta sebagai wilayah dengan jumlah kasus tertinggi, yakni 59 kasus [3].

Penyebaran cacar monyet dapat terjadi ketika seseorang melakukan kontak fisik langsung dengan lesi pada kulit penderita, membran mukosa, atau cairan tubuh dari penderita. Tak hanya itu, virus juga dapat menyebar melalui droplet pernapasan serta benda yang terkontaminasi virus, seperti pakaian dan peralatan rumah tangga. Faktor risiko utama meliputi berbagi tempat tidur, alat makan, serta interaksi erat dengan penderita [4]. Mengingat karakteristik penularannya yang cukup luas, diperlukan sistem deteksi dini yang akurat untuk mencegah penyebaran yang lebih besar.

Convolutional Neural Network (CNN) adalah metode yang banyak diterapkan di berbagai bidang dalam analisis citra medis karena kemampuannya dalam mengenali pola dan fitur dari gambar [5]. Metode ini dapat diterapkan pada berbagai aplikasi, seperti pengenalan wajah, klasifikasi gambar, dan lain sebagainya [6]. CNN terdiri dari lapisan-lapisan yang meniru mekanisme otak dalam mengenali objek visual, menjadikannya lebih efektif dibandingkan jaringan saraf konvensional [7]. Selain itu, CNN meniru cara kerja *visual cortex* manusia, menjadikannya sangat efektif dalam berbagai aplikasi seperti diagnosis medis dan pengenalan wajah [8]. Menurut [9], banyak penelitian telah membuktikan efektivitas CNN dalam mengenali pola serta melakukan klasifikasi citra dengan akurasi yang tinggi.

CNN pertama kali diperkenalkan dalam bentuk *LeNet* pada tahun 1998 dan berkembang menjadi *AlexNet* pada 2012, diikuti oleh *ResNet* pada 2015 dengan tingkat

kesalahan yang lebih rendah [10]. Salah satu arsitektur CNN yang unggul adalah *ResNet-101*, yang diperkenalkan oleh Kaiming He dkk. Arsitektur ini mengadopsi teknik *skip connections* sebagai solusi untuk mengurangi risiko *vanishing gradient* pada jaringan yang dalam [11]. Keunggulan *ResNet-101* dalam menjaga kestabilan pelatihan menjadikannya pilihan yang tepat untuk mendeteksi pola halus pada gambar lesi kulit cacar monyet, sehingga dapat meningkatkan akurasi deteksi.

Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk mendeteksi cacar monyet dengan berbasis citra kulit manusia menggunakan Algoritma CNN dengan arsitektur *ResNet-101* dan mengetahui hasil akurasinya. Dengan adanya sistem deteksi berbasis CNN ini, diharapkan proses deteksi dapat dilakukan secara otomatis berbasis citra kulit, namun tetap memerlukan dukungan diagnosis klinis dan laboratorium lanjutan.

Implementasi model ini diharapkan dapat membawa dampak positif atau manfaat bagi pihak yang terkait, seperti fasilitas kesehatan dalam membantu mempercepat diagnosis dengan catatan dibantu juga dengan dukungan diagnosis lain. pemerintah dalam pengendalian penyebaran penyakit, serta peneliti dan pengembang teknologi kesehatan dalam meningkatkan metode deteksi penyakit berbasis citra kulit. Selain itu, masyarakat juga dapat memperoleh akses deteksi dini untuk mencegah penyebaran cacar monyet lebih luas.

Penelitian ini memiliki kontribusi baru dibanding studi sebelumnya karena merupakan salah satu studi awal yang menerapkan arsitektur *ResNet-101* untuk deteksi cacar monyet berbasis gambar kulit. Pendekatan ini belum banyak dijumpai dalam literatur, yang umumnya masih menggunakan CNN dasar atau *ResNet-50*.

II. METODE

A. *Convolutional Neural Network* (CNN)

CNN memiliki struktur yang terdiri dari beberapa lapisan utama, yaitu *convolutional layer*, *pooling layer*, dan *fully connected layer* [12]. *Convolutional layer* berperan untuk mengekstraksi fitur dari citra input menggunakan operasi konvolusi. Tahapan ini mencakup penerapan kernel pada citra untuk menghasilkan *feature map*, yang dapat dirumuskan seperti pada rumus (1).

$$s(i,j) = (K * I)(i,j) = \sum\sum I(i-m, j-n)K(m,n) \quad (1)$$

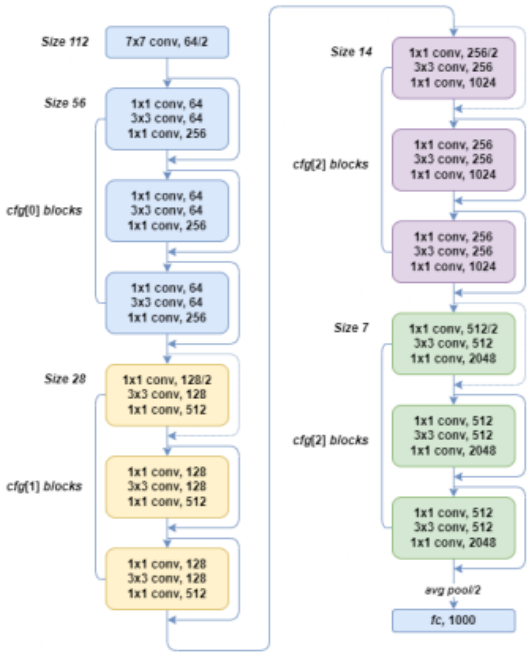
di mana I adalah input gambar, K adalah kernel, dan $s(i, j)$ adalah hasil konvolusi [13]. *Pooling layer* berfungsi untuk mereduksi dimensi *feature map* tanpa kehilangan informasi penting, menggunakan metode seperti *max pooling* dan *average pooling*, yang membantu meningkatkan efisiensi komputasi [14]. Sementara itu, *fully connected layer* menghubungkan seluruh *neuron* dalam jaringan, memungkinkan model melakukan klasifikasi berdasarkan fitur yang telah diekstraksi [13].

Seiring perkembangan teknologi, CNN telah memiliki berbagai arsitektur seperti *LeNet*, *AlexNet*, *VGGNet*, *GoogLeNet*, *dan ResNet*, masing-masing dengan keunggulan dalam menangani kompleksitas data visual serta meningkatkan akurasi dalam berbagai tugas *computer vision* [15],[16]. CNN unggul *dalam feature learning*, memungkinkan model mengekstraksi fitur secara otomatis tanpa memerlukan rekayasa fitur manual [17].

B. *ResNet-101*

Residual Network atau *ResNet* adalah arsitektur dari *Convolutional neural network* yang dikemukakan oleh Kaiming He et al pada tahun 2015 untuk menangani masalah *vanishing gradient* pada jaringan yang sangat dalam. *ResNet* memperkenalkan *skip connections* atau *shortcut connections*, yang memungkinkan *gradien* melewati beberapa *layer* tanpa mengalami degradasi, sehingga mempercepat dan menstabilkan proses pelatihan.

Terdapat beberapa varian *ResNet* seperti *ResNet-18*, *ResNet-34*, *ResNet-50*, *ResNet-101*, dan *ResNet-152*, yang dibedakan berdasarkan jumlah lapisannya. *ResNet-101* memiliki 33 *residual block*, menjadikannya cocok untuk analisis fitur yang lebih kompleks pada dataset berskala besar.

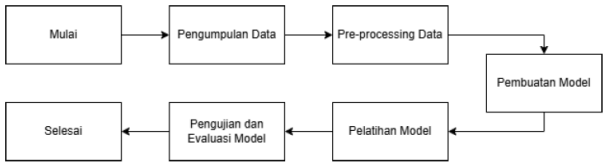


Gambar 1. Arsitektur *resnet-101*

Gambar 1, menunjukkan struktur utama *ResNet-101*, yang menggunakan *residual block* dengan *shortcut connection* sebagai elemen utama. Setiap *residual block* terdiri dari tiga lapisan konvolusi berukuran 1x1, 3x3, dan 1x1, yang dikenal sebagai *Deeper Bottleneck Architecture*. Pendekatan ini meningkatkan efisiensi komputasi dengan mengurangi jumlah parameter tanpa mengorbankan kualitas representasi fitur. Selain itu, *ResNet-101* menggunakan *parameter-free identity shortcut*, yang mempertahankan operasi langsung tanpa penambahan parameter tambahan,

sehingga lebih efisien dalam implementasi jaringan skala besar [18].

C. Rancangan Penelitian



Gambar 2. Rancangan Penelitian

Gambar 2, dapat dilihat penelitian ini akan dilakukan melewati tahapan-tahapan yang sistematis dan berurutan, dimulai dari pengumpulan data, *pre-processing* data, pembuatan model, pelatihan model, hingga pengujian dan evaluasi model.

D. Pengumpulan Data

(*Monkey Pox*) (*Others*)

Gambar 3. Dataset

Gambar 3, data yang akan digunakan dalam penelitian adalah citra kulit yang menunjukkan tanda-tanda cacar monyet pada manusia. Dataset yang digunakan berasal dari kaggle berjumlah 316 gambar yang terdiri dari 2 folder yaitu folder *Monkey Pox* (234 gambar) dan *Others* (82 gambar). Folder *Monkey Pox* berisi gambar lesi kulit yang terindikasi cacar monyet dan folder *Others* berisi gambar lesi kulit yang tidak terindikasi cacar monyet.

Karena keterbatasan ukuran dan sumber dataset yang hanya berasal dari satu sumber, terdapat potensi bias dan keterbatasan dalam representasi keragaman kondisi kulit pasien nyata. Distribusi kelas ini menunjukkan adanya ketidakseimbangan data (*data imbalance*), karena jumlah gambar pada kelas *Monkey Pox* jauh lebih banyak dibandingkan dengan kelas *Others*. Ketidakseimbangan ini dapat memengaruhi performa model, khususnya dalam mengenali kelas dengan jumlah data lebih sedikit, dan perlu menjadi perhatian dalam proses pelatihan dan evaluasi model.

E. Pre-processing Data

Data yang telah dikumpulkan akan melalui tahap *pre-processing* yang meliputi langkah-langkah penting seperti augmentasi, pembagian data dan sebagainya untuk memastikan data yang digunakan dalam pelatihan model memenuhi standar kualitas yang diperlukan. Proses *pre-processing* ini bertujuan untuk mempersiapkan data sehingga model dapat diproses dengan baik. Berikut merupakan langkah-langkah yang digunakan:

1) Flip :

Pseudocode :

```
Mulai
// Baca gambar dari file atau dataset
gambar = BacaGambar("path/to/gambar")

// Balik gambar secara horizontal
gambarHorizontal = FlipHorizontal(gambar)

// Balik gambar secara vertical
gambarVertical = FlipVertical(gambar)

// Tampilkan gambar hasil flipping
TampilkanGambar(gambarHorizontalVertical)
Selesai
```

Hasil :

(gambar asli) (Horizontal) (Vertical)

Gambar 4. *Flip* gambar

Gambar 4 setiap gambar dalam dataset, posisinya dibalik secara *horizontal* dan *vertical* untuk meningkatkan variasi data latih.

2) Rotation :

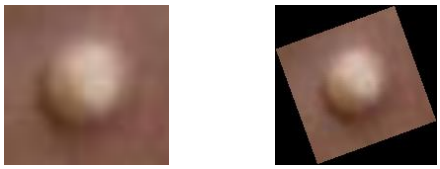
Pseudocode :

```
Mulai
// Baca gambar dari file atau dataset
gambar = BacaGambar("path/to/gambar")

// Rotasi gambar sebesar 20 derajat
gambarRotation = RotasiGambar(gambar)

// Tampilkan gambar hasil Rotation
TampilkanGambar(gambarRotation)
Selesai
```

Hasil :



(gambar asli) (Rotation 20°)

Gambar 5. *Rotation* gambar

Gambar 5, dapat dilihat setiap gambar dalam dataset, posisinya diputar 20 derajat untuk meningkatkan keragaman data latih.

3) Zoom :
Pseudocode :

```
Mulai
// Baca gambar dari file atau dataset
gambar = BacaGambar("path/to/gambar")

// Perbesar gambar sebesar 1.5x
gambarZoom = ZoomGambar(gambar)

// Tampilkan gambar hasil Zoom
TampilkanGambar(gambarZoom)
Selesai
```

Hasil :

(gambar asli) (*Zoom* 1.5x)

Gambar 6. *Zoom* gambar

Gambar 6, setiap gambar dalam dataset, dipotong bagian tengah dan diperbesar kembali sebesar 1.5x untuk menciptakan efek pembesaran objek gambar agar terfokus pada objek yang akan diolah.

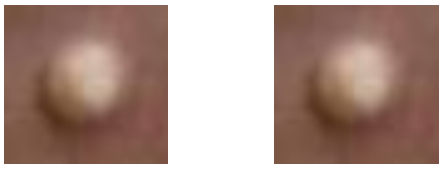
4) Resize :
Pseudocode :

```
Mulai
// Baca gambar dari file atau dataset
gambar = BacaGambar("path/to/gambar")

// Ubah ukuran gambar ke 224x224 piksel
gambarResize = UbahUkuran(gambar, 224, 224)

// Tampilkan gambar hasil resize
TampilkanGambar(gambarResize)
Selesai
```

Hasil :



(gambar asli) (224x224)

Gambar 7. *Resize* gambar

Pada Gambar 7, dapat dilihat setiap gambar dalam dataset, ukurannya diubah ke 224x224 piksel agar memiliki kesesuaian dengan arsitektur model *ResNet-101*.

5) Normalization :
Pseudocode :

```
Mulai

// Baca gambar dari file atau dataset
gambar = BacaGambar("path/to/gambar")

// Normalisasi gambar dengan membagi setiap piksel dengan
255.0
gambarNormalisasi = NormalisasiGambar(gambar, 255.0)

// Tampilkan gambar setelah normalisasi
TampilkanGambar(gambarNormalisasi)
Selesai
```

Hasil :

(gambar asli) (*Normalization*)

Gambar 8. Normalization gambar

Gambar 8, dapat dilihat seluruh gambar dalam dataset menjalani proses *normalization* atau normalisasi dengan membagi nilai pikselnya dengan 255, sehingga nilainya berada dalam kisaran 0 sampai 1.

6) Penetapan Parameter Acak : Untuk memastikan eksperimen yang dilakukan memiliki hasil yang replikatif, *seed* acak ditetapkan dengan nilai 1 pada *TensorFlow* dan *NumPy*.

7) Batch Size : Digunakan *batch size* sebesar 32 untuk mempercepat proses pelatihan, meningkatkan efisiensi komputasi, dan mengoptimalkan penggunaan memori *GPU* selama proses *training*.

8) Pembagian Data : Dataset yang berjumlah 316 gambar, dilakukan pembagian dengan proporsi 80% sebagai data latih dan 20% sebagai data validasi, sementara data *test* menggunakan subset yang sama dengan data validasi. Sehingga menghasilkan pembagian data sebanyak 285 gambar untuk data latih, 62 gambar untuk data validasi, dan 31 gambar dari data validasi juga digunakan sebagai data *test.* Meskipun total jumlah data tampak melebihi jumlah awal, data *test* diambil sebagai subset dari data validasi, sehingga tidak menambah total data secara keseluruhan. Strategi ini diterapkan untuk memaksimalkan pemanfaatan dataset yang terbatas.

F. Pembuatan Model

ResNet-101 adalah arsitektur yang digunakan pada penelitian ini, yang sudah dilatih sebelumnya menggunakan dataset *ImageNet* dan digunakan sebagai fitur ekstraktor. Parameter yang digunakan dalam pembuatan model adalah:

1) Base Model: ResNet-101 tanpa lapisan fully connected (include_top=False) dan menggunakan bobot pre-trained dari ImageNet.

2) Lapisan Tambahan : Global Average Pooling 2D untuk mereduksi dimensi fitur. *Batch Normalization* setelah setiap lapisan *dense* untuk meningkatkan stabilitas pelatihan. Lapisan *dense* dengan jumlah *neuron* bertingkat 1024, 512, dan 128 dengan fungsi aktivasi *ReLU*. *Dropout* dengan nilai 0.2 pada setiap lapisan *dense*. *Dropout* ini secara acak menonaktifkan sejumlah *neuron* selama pelatihan guna

meningkatkan kemampuan generalisasi model untuk menghindari *overfitting*. Lapisan dari *Output* terdapat 2 *neuron* dengan aktivasi *softmax* untuk klasifikasi dua kelas (cacar monyet dan bukan cacar monyet).

3) Optimasi : Optimizer Adam dengan *learning rate* awal 5e-4. *Loss function categorical crossentropy*. Metode evaluasi berdasar dari akurasi.

G. Pelatihan Model

Proses Pelatihan model dilaksanakan selama 100 *epoch* dengan mekanisme pengurangan *learning rate* sebesar 0.2 jika validasi *loss* tidak mengalami peningkatan selama 3 *epoch*.

Beberapa *callback* yang diterapkan dalam pelatihan model adalah:

1) Model Checkpoint : Menyimpan model terbaik secara otomatis berdasarkan akurasi validasi tertinggi yang diperoleh selama pelatihan.

2) Reduce Learning Rate on Plateau : Mengurangi laju pembelajaran secara bertahap untuk meningkatkan konvergensi model saat tidak terjadi peningkatan akurasi.

3) Early Stopping : Selain itu, diterapkan juga teknik *early stopping* dengan patokan *patience* 15 *epoch*. Yang berarti pelatihan akan dihentikan secara otomatis jika tidak terjadi peningkatan pada *validation loss* selama 15 *epoch* berturut-turut. Teknik ini bertujuan untuk mencegah *overfitting* dan menghemat waktu pelatihan.

Setelah pelatihan, dilakukan visualisasi hasil pelatihan dengan menampilkan grafik akurasi dan *loss* pada setiap *epoch* untuk memantau performa model secara keseluruhan.

H. Pengujian dan Evaluasi Model

Pengujian model dilakukan menggunakan data uji yang diproses dengan metode yang sama seperti data latih. Evaluasi dilakukan dengan:

1) Classification Report : Menggunakan metrik *precision*, recall, F1-score, dan akurasi untuk mengetahui kinerja model.

2) Confusion Matrix : Membandingkan prediksi model dengan label asli untuk mengukur jumlah prediksi yang benar dan salah.

3) Visualisasi Prediksi : Menampilkan beberapa contoh hasil klasifikasi model dalam bentuk gambar dengan label asli dan prediksi.

I. Pengujian Pada Data Eksternal

Selain menggunakan dataset utama, model diuji dengan data atau gambar eksternal yang mewakili kondisi kulit yang berbeda. Setiap gambar diuji dengan model yang telah dilatih. Gambar yang diuji diubah ukurannya menjadi 224x224 piksel, dinormalisasi, lalu dimasukkan ke dalam model untuk memperoleh prediksi probabilitas.

Hasil prediksi ditampilkan dalam bentuk gambar dengan label yang menunjukkan apakah gambar tersebut diklasifikasikan sebagai cacar monyet atau bukan cacar monyet, dengan warna merah untuk cacar monyet dan hijau untuk bukan cacar monyet. Selain label warna, pada gambar yang diklasifikasikan sebagai cacar monyet, juga ditampilkan *bounding box* yang menandai area lesi atau bagian kulit yang menjadi fokus klasifikasi model. Hal ini dilakukan untuk memperkuat interpretabilitas dan membantu pengguna dalam mengidentifikasi lokasi yang relevan dari hasil deteksi.

III. HASIL DAN PEMBAHASAN

A. Hasil Pelatihan Model

Proses pelatihan model dilaksanakan selama 100 epoch. Grafik hasil pelatihan model ditunjukkan pada Gambar 9, yang memperlihatkan tren akurasi dan *loss* selama proses pelatihan.

Gambar 9. Grafik hasil pelatihan model

Berdasarkan Gambar 9, terlihat bahwa model mengalami peningkatan akurasi seiring dengan bertambahnya jumlah *epoch*. Akurasi tertinggi yang diperoleh pada data validasi mencapai 94%, menunjukkan model memiliki performa yang cukup baik dalam mengenali pola dari data latih.

B. Hasil Evaluasi Model

Setelah melalui tahap pelatihan dan pengujian, model dievaluasi menggunakan *classification report* dengan metrik *precision*, *recall*, *F1-score*, dan akurasi. Hasil evaluasi model disajikan dalam Gambar 10.

Classification Report:

	precision	recall	f1-score	support
Monkey Pox	0.96	0.96	0.96	23
Others	0.88	0.88	0.88	8
accuracy			0.94	31
macro avg	0.92	0.92	0.92	31
weighted avg	0.94	0.94	0.94	31

Gambar 10. *Clasification report*

Gambar 10, dapat dilihat hasil evaluasi, model mencapai akurasi sebesar 94%, yang menunjukkan bahwa model mampu mengklasifikasikan sebagian besar gambar dengan

benar. Namun, performa model untuk masing-masing kelas menunjukkan adanya perbedaan yang signifikan.

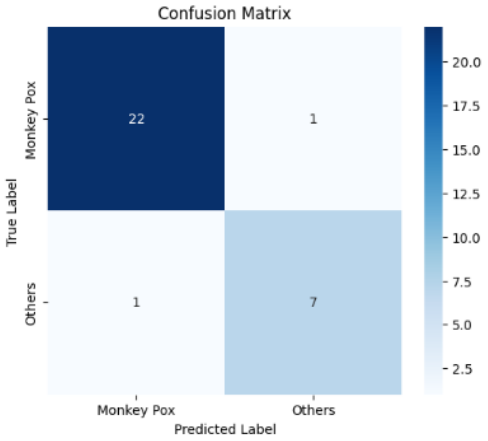
Pada kelas *Monkey Pox*, model memiliki *precision* sebesar 0.96, yang berarti hampir semua gambar yang diklasifikasikan sebagai *Monkey Pox* memang benar-benar termasuk dalam kategori tersebut. *Recall* pada kelas ini mencapai 0.96, yang mengindikasikan bahwa dari seluruh gambar *Monkey Pox* yang ada, 96% yang berhasil dikenali dengan benar oleh model, sementara sisanya (4%) salah diklasifikasikan sebagai kelas *Others* (*false negative*). Hal ini membuat *F1-score* pada kelas *Monkey Pox* mencapai 0.96, yang lebih tinggi dibandingkan dengan kelas lainnya.

Sementara itu, pada kelas *Others*, model menunjukkan performa yang lebih buruk dengan *precision* sebesar 0.88 dan *recall* sebesar 0.88, yang berarti bahwa dari semua prediksi kelas *Others*, 88% benar, dan dari seluruh data kelas *Others* yang sebenarnya ada, 88% berhasil dikenali dengan benar. Hal ini menghasilkan *F1-score* sebesar 0.88, menunjukkan bahwa model lebih unggul dalam mengenali kelas *Monkey Pox* dibandingkan kelas *Others*.

Jika dilihat dari *macro average*, *precision* mencapai 0.92, *recall* 0.92, dan *F1-score* 0.92. Sementara itu, *weighted average* menunjukkan *precision* sebesar 0.94, *recall* 0.94, dan *F1-score* 0.94. Nilai rata-rata berbobot ini mempertimbangkan jumlah sampel di masing-masing kelas, dan menunjukkan bahwa performa keseluruhan model cukup baik, meskipun cenderung lebih optimal dalam mengenali kelas *Monkey Pox*.

C. Analisis Confusion Matrix

Confusion Matrix yang diperoleh dari hasil pengujian model dapat dilihat pada Gambar 11.



Gambar 11. *Confusion matrix*

Pada Gambar 11, memperlihatkan *Confusion Matrix* di atas, dari 31 gambar pada data uji dapat diketahui bahwa model berhasil mengklasifikasikan 22 sampel *Monkey Pox* dengan benar, tetapi terdapat 1 sampel *Monkey Pox* yang salah diklasifikasikan sebagai *Others* (*false negative*). Model memiliki *false positive*, karena ada 1 sampel dari kelas *Others* yang salah diklasifikasikan sebagai *Monkey Pox*. Sebanyak 7 sampel dari kelas *Others* diklasifikasikan dengan benar.

Dari hasil ini, menyimpulkan bahwa model memiliki performa yang sangat baik dalam mengenali kelas *Monkey Pox*, tetapi masih mengalami sedikit kesalahan dalam mendeteksi *Others*. Kesalahan ini terlihat dari adanya *false positive*, di mana satu sampel *Others* justru diklasifikasikan sebagai *Monkey Pox*. Kesalahan semacam ini cukup krusial karena dapat menyebabkan keterlambatan dalam diagnosis penyakit cacar monyet dan potensi salah penanganan jika individu yang sehat diklasifikasikan sebagai penderita.

D. Visualisasi Prediksi Model

Selain menggunakan metrik numerik seperti *classification report* dan *confusion matrix*, model juga dievaluasi dengan visualisasi prediksi terhadap sampel gambar dalam dataset uji. Gambar 12 menunjukkan beberapa hasil klasifikasi yang dilakukan oleh model CNN dengan arsitektur *ResNet-101*.

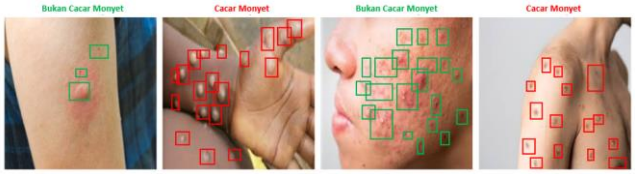
Gambar 12. Visualisasi Prediksi

Gambar 12, dapat dilihat setiap sampel diberikan label *True* (label asli) dan *Pred* (prediksi model). Analisis hasil prediksi menunjukkan bahwa model mampu mengklasifikasikan sebagian besar gambar dengan benar. Dari lima sampel yang ditampilkan, tiga gambar terklasifikasi dengan benar sesuai dengan label aslinya. Dua gambar lainnya termasuk dalam kategori "bukan cacar monyet" (label 1) dan diprediksi dengan benar sebagai bukan cacar monyet.

Hasil ini mencerminkan bahwa model memiliki performa yang cukup baik dalam membedakan antara cacar monyet dan kondisi kulit lainnya.

E. Hasil Pengujian dengan Data Eksternal

Untuk menguji kemampuan generalisasi model, dilakukan pengujian terhadap 4 gambar eksternal yang tidak termasuk dalam dataset pelatihan, validasi, maupun pengujian. Gambar 13 menunjukkan hasil klasifikasi model, di mana label hijau ("Bukan Cacar Monyet") menunjukkan bahwa model mendeteksi kondisi kulit lain seperti jerawat atau eksim, sedangkan label merah ("Cacar Monyet") menandakan model mengenali lesi khas cacar monyet. Selain label warna, gambar juga dilengkapi dengan *bounding box* yang menandai area kulit yang menjadi fokus klasifikasi model. Pada gambar yang diklasifikasikan sebagai cacar monyet, *bounding box* berwarna merah ditampilkan di area lesi yang terindikasi cacar monyet, sementara pada gambar yang diklasifikasikan sebagai bukan cacar monyet, *bounding box* berwarna hijau menunjukkan area yang terindikasi bukan cacar monyet.



Gambar 13. Pengujian data eksternal

Berdasarkan Gambar 13, dapat dilihat hasil pengujian menunjukkan bahwa model berhasil mengklasifikasikan seluruh gambar dengan benar, menegaskan kemampuannya dalam membedakan cacar monyet dan bukan cacar monyet. Performa ini mencerminkan efektivitas arsitektur *ResNet-101* dalam menangkap fitur penting pada gambar lesi kulit. Meskipun hasilnya menunjukkan akurasi penuh, jumlah gambar eksternal masih terbatas sehingga generalisasi model perlu diuji lebih lanjut dengan dataset nyata dari institusi medis.

IV. KESIMPULAN

Penelitian ini berhasil menerapkan *Convolutional Neural Network* (CNN) dengan arsitektur *ResNet-101* untuk mendeteksi cacar monyet pada manusia, dengan akurasi 94% berdasarkan evaluasi menggunakan *classification report* dan *confusion matrix*. Performa model menunjukkan hasil yang sangat baik dalam mengenali kelas *Monkey Pox*, namun masih terdapat *false positive*, yaitu ketika model salah mengklasifikasikan satu sampel dari kelas *Others* sebagai *Monkey* Pox, yang dapat berdampak pada kesalahan deteksi. Pengujian dengan data eksternal juga menunjukkan hasil yang konsisten, membuktikan bahwa model memiliki kemampuan generalisasi yang baik.

Penelitian berikutnya, disarankan untuk menggunakan dataset yang lebih besar dan beragam, serta eksplorasi teknik *balancing* data dan *fine-tuning* model agar dapat meningkatkan akurasi dan mengurangi kesalahan deteksi. Penelitian ini menggunakan satu model utama yaitu *ResNet-101*. Ke depan, diperlukan perbandingan dengan baseline arsitektur lain seperti *ResNet-50, MobileNet, atau VGG-16* untuk memastikan keunggulan arsitektur yang digunakan.

DAFTAR PUSTAKA

[1] L. Hilmi Marisah, I. Laily, and Salman, "Studi dan Tatalaksana Terkait Penyakit Cacar Monyet (Monkeypox) yang Menginfeksi Manusia," *Jurnal Farmasetis*, vol. 11, no. 3, 2022.

[2] C. S. Kuncoro, "Monkeypox: Manifestasi dan Diagnosis," 2023.

[3] *GoodStats Indonesia, "Jumlah Kasus Cacar Monyet di Indonesia," GoodStats, 2024. [Online]. Available: https://goodstats.id/article/jumlah-kasus-cacar-monyet-di-indonesia-PHI8p. [Accessed: Apr. 28, 2025].*

[4] L. Budiyarto, A. A. Sabila, and H. C. Putri, "Infeksi Cacar Monyet (Monkeypox)," *Jurnal Medikah Utama*, 2023. [Online]. Available: http://jurnalmedikahutama.com. [Accessed: Apr. 28, 2025].

[5] Tamba, "Jumlah Kasus Cacar Monyet," *Circle Archive*, 2024. [Online]. Available: https://circle-archive.com/index.php/carc/article/view/290. [Accessed: Apr. 28, 2025].

[6] A. Antoni, T. Rohana, and A. R. Pratama, "Implementasi Algoritma Convolutional Neural Network untuk Klasifikasi Citra Kemasan Kardus Defect dan No Defect," *Building of Informatics, Technology and Science (BITS)*, vol. 4, no. 4, 2023. [Online]. Available: https://doi.org/10.47065/bits.v4i4.3270.

[7] A. R. Juwita, T. Al Mudzakir, A. R. Pratama, P. Husodo, and R. Sulaiman, "Identifikasi Citra Batik dengan Metode Convolutional Neural Network," Buana Ilmu, vol. 6, no. 1, pp. 192–208, 2021. [Online]. Available: https://doi.org/10.36805/bi.v6i1.1996.

[8] D. Aprillia, T. Rohana, T. Al Mudzakir, and D. Wahiddin, "Deteksi Nominal Mata Uang Rupiah Menggunakan Metode Convolutional Neural Network dan Feedforward Neural Network," *Kajian Ilmiah Informatika dan Komputer (KLIK)*, vol. 4, no. 4, 2024.

[9] A. Kirana and H. Hikmayanti, "Pengenalan Pola Aksara Sunda dengan Metode Convolutional Neural Network," *Jurnal Informatika*, vol. 1, no. 2, 2020.

[10] I. N. Pratama, T. Rohana, T. Al Mudzakir, and P. Karawang, "Pengenalan Sampah Plastik dengan Model Convolutional Neural Network," dalam *Seminar Nasional Hasil Riset Prefix-RTR*, 2020.

[11] N. Hanun, M. Sarosa, and R. A. Asmara, "Pemanfaatan Algoritma Faster R-CNN ResNet-101 untuk Deteksi Potongan Tubuh Manusia," *Jurnal Elektronika dan Otomasi Industri*, vol. 10, no. 1, pp. 94–103, 2023. [Online]. Available: https://doi.org/10.33795/elkolind.v10i1.2754.

[12] Febriyanti, F. A. (2024). Image Processing Dengan Metode Convolutional Neural Network (Cnn) Untuk Deteksi Penyakit Kulit Pada Manusia. https://ejournal.warunayama.org/kohesi.

[13] Mahmud, "Implementasi Deep Learning dengan Menggunakan Algoritma Convolutional Neural Network untuk Mengidentifikasi Jenis Ikan Laut," 2021.

[14] Ivan, "Pooling Layer," *BINUS School of Computer Science*, Oct. 7, 2021. [Online]. Available: https://socs.binus.ac.id/2021/10/07/pooling-layer/. [Accessed: Apr. 28, 2025].

[15] A. Kumar, "Different Types of CNN Architectures Explained with Examples," *Vitalflux*, 2023. [Online]. Available: https://vitalflux.com/different-types-of-cnn-architectures-explained-examples/. [Accessed: Apr. 28, 2025].

[16] A. Pamungkas, "Jenis-jenis Arsitektur Convolutional Neural Network (CNN) untuk Image Recognition dan Computer Vision," *Pemrograman MATLAB*, Jul. 23, 2023. https://pemrogramanmatlab.com/2023/07/23/jenis-jenis-arsitektur-convolutional-neural-network-cnn-untuk-image-recognition-dan-computer-vision/. [Accessed: Apr. 28, 2025].

[17] I. Maulana et al., "Deteksi Bentuk Wajah Menggunakan Convolutional Neural Network (CNN)," *Jurnal Mahasiswa Teknik Informatika*, vol. 7, no. 6, 2023.

[18] E. Chodry, "Implementasi Arsitektur ResNet50 dan ResNet101 pada Sistem Kehadiran Berbasis Face Recognition," 2024.