Journal of Applied Informatics and Computing (JAIC)
Vol.8, No.2, December 2024, pp. 443~452
e-ISSN: 2548-6861

Betta Fish Identification System Based On Convolutional Neural Network

Gilang Ardhi Saputra [1]*, I Made Artha Agastya [2]*
*Informatika, Universitas Amikom Yogyakarta
gilangardhis@students.amikom.ac.id [1], artha.agastya@amikom.ac.id [2]

Article Info

Article history:

Received 2024-09-06
Revised 2024-10-09
Accepted 2024-10-14

Keyword:

Betta Splendens,
Convolutional Neural Network (CNN),
Fish Classification,
Image Identification.

ABSTRACT

This study developed an automated identification system based on the Convolutional Neural Network (CNN) to classify Betta Splendens, a fish species with high economic value in Indonesia. The system aims to improve accuracy and efficiency in the identification process. The research was divided into several experiments, where the data was split into 320 images for training, 80 for validation, and 100 for testing. We used two optimizers, Adam and RMSprop. The Adam optimizer experiments conducted two stages with learning rates of 0.0001 and 0.001, each with 100 and 200 epochs. The results showed that a lower learning rate (0.0001) with 200 epochs yielded the best test accuracy of 71%, while a learning rate of 0.001 caused accuracy to stagnate at 66%, indicating potential overfitting. The RMSprop optimizer with a learning rate of 0.00001 demonstrated good stability, though with slightly lower accuracy than Adam. This study highlights the importance of selecting the appropriate learning rate and number of epochs to achieve an optimal balance between training, validation, and testing accuracy, ensuring the model generalizes well to new data.

I. PENDAHULUAN

Betta splendens, atau lebih dikenal sebagai ikan cupang, merupakan salah satu spesies ikan hias yang populer di Indonesia. Menurut Badan Karantina Ikan, Pengendalian Mutu dan Keamanan Hasil Perikanan (BKIPM), pada tahun 2023, ekspor ikan cupang mencapai 10 juta ekor dengan nilai ekonomi sebesar Rp 30 miliar [1]. Tingginya permintaan ini menjadikan identifikasi varietas ikan cupang secara akurat dan efisien sangat penting, terutama bagi orang yang masih awam dalam mengenali ciri fisik ikan cupang, karena jenis-jenis ikan cupang hias memiliki kemiripan pada struktur tubuh, srip, dan ekornya [2]. Identifikasi manual ikan cupang memakan waktu dan rentan terhadap kesalahan, sehingga sistem identifikasi otomatis menggunakan data dan Convolutional Neural Network (CNN) dapat meningkatkan akurasi dan efisiensi.

CNN, yang dirancang untuk pemrosesan data seperti gambar atau suara [3], [4], digunakan dalam pembelajaran terawasi untuk klasifikasi gambar, dengan melibatkan pelatihan menggunakan backpropagation dan klasifikasi menggunakan feedforward [5]. Penelitian terkini menunjukkan kemajuan signifikan dalam klasifikasi berbasis CNN. Sebagai contoh, penelitian [6] mengevaluasi arsitektur CNN Xception, NasNet, dan MobileNet V3L pada 9000 gambar ikan laut, dengan akurasi masing-masing 97%, 92%, dan 99%, meskipun kurang dalam metrik evaluasi seperti presisi dan recall. Penelitian lain [7] menghasilkan hasil serupa tetapi kurang mendetail dalam arsitektur deep learning. Selain itu, penelitian [4] tentang klasifikasi jamur menggunakan CNN mencapai akurasi 89% pada pelatihan dan 82% pada validasi, meskipun pengaruh parameter seperti learning rate tidak dieksplorasi. CNN juga mengungguli K-Nearest Neighbor (K-NN) dalam klasifikasi ikan dengan dataset kecil [8].

Dalam penelitian lain [5], CNN mencapai akurasi 85,31%, presisi 89,92%, dan sensitivitas 86,49% dalam klasifikasi karang, meskipun analisis overfitting kurang mendalam. CNN juga digunakan dalam klasifikasi ikan cupang berbasis mobile dengan model VGG16, mencapai akurasi hingga 94,35% [1], meskipun validasi eksternal tidak dilakukan. Selain itu, CNN digunakan dalam klasifikasi penyakit tanaman berbasis IoT dengan akurasi 98% [9] dan dalam mendeteksi COVID-19 dari gambar X-ray dengan ResNet50, mencapai akurasi hingga 94,7% [10].

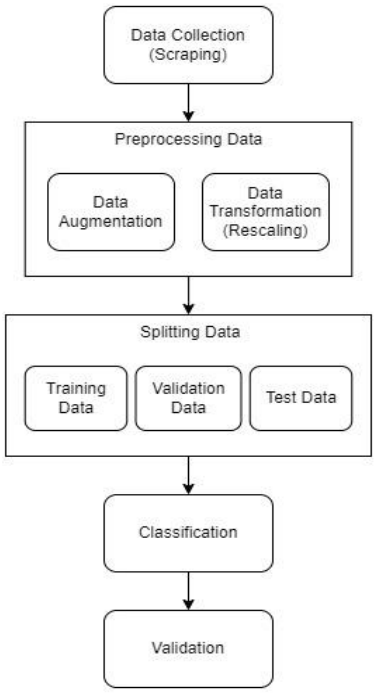
Tinjauan literatur ini menyoroti potensi CNN jenis jaringan neural yang khusus dirancang dan telah menunjukkan kinerja luar biasa dalam berbagai kompetisi yang berkaitan dengan computer vision dan image processing[11], terutama untuk ikan cupang, meskipun diperlukan dataset yang lebih beragam dan teknik augmentasi data. Transfer learning menggunakan VGG16 diusulkan untuk penelitian ini [12]. VGG16 lebih dipilih daripada VGG19 karena ukurannya yang lebih kecil meskipun kinerjanya hampir sama [13]. meskipun kinerja keduanya hampir setara. Namun, VGG19 tetap digunakan sebagai bahan perbandingan guna memperoleh hasil yang paling optimal.

Penelitian sebelumnya [1], [4] telah mengimplementasikan CNN dengan berbagai algoritma pada data dari sumber seperti Kaggle dan pengumpulan gambar langsung. CNN terbukti efektif dalam tugas-tugas pengenalan objek [14], mencapai akurasi hingga 90% bahkan dengan dataset kecil [15], menunjukkan keunggulannya dibandingkan metode lain seperti RNN. Penelitian ini bertujuan mengembangkan sistem yang mampu mengklasifikasikan varietas ikan cupang dengan akurasi tinggi, menjawab kebutuhan yang semakin mendesak akan sistem identifikasi seiring dengan semakin populernya hobi memelihara ikan cupang.

II. METODE PENELITIAN

A. Alur Penelitian

Alur penelitian dapat dilihat pada gambar 1. Penelitian dibagi menjadi beberapa tahap, data collection, preprocessing data, splitting data, classification, dan validation.



Gambar 1. Alur Penelitian

B. Data Collection

Pada tahap awal penelitian ini, dataset yang berhasil dikumpulkan berjumlah 500 citra ikan cupang dari 20 jenis yang berbeda menggunakan metode pengumpulan data secara otomatis melalui teknik scraping. Namun, analisis awal terhadap dataset menunjukkan adanya beberapa kendala kualitas, seperti ukuran citra yang terlalu kecil dan terjadinya duplikasi citra antar jenis ikan. Kendala-kendala ini mengindikasikan bahwa data yang dikumpulkan tidak cukup representatif dan tidak memenuhi standar kualitas yang diharapkan untuk melatih model dengan akurasi yang tinggi. Table 1 menunjukkan hasil proses penyederhanaan label dataset, di mana bagian kiri menampilkan citra data yang didapat dari porses scaping secara otomatis yang menghasilkan beberapa data yang memiliki kemiripan bentuk yang signifikan, yang dapat berpotensi meningkatkan risiko overfitting pada proses pelatihan model, terutama jika data tidak cukup beragam atau tidak diolah dengan teknik augmentasi yang tepat, sedangkan bagian kanan menampilkan citra data setelah penghapusan atau penggabungan citra data yang digunakan.

Untuk mengatasi masalah tersebut, pengumpulan data dilakukan kembali secara manual dengan teknik scraping yang lebih terarah dari sumber-sumber terpercaya di internet. Proses ini menghasilkan dataset final yang terdiri dari 10 jenis ikan cupang yang dianggap paling relevan berdasarkan tujuan penelitian. Jenis-jenis tersebut meliputi coccina, crown tail, double tail, halfmoon, halfsun, paradise, plakat, snakehead, spade tail, dan veil tail. Setiap jenis ikan cupang diwakili oleh 50 citra yang berkualitas tinggi, sehingga total dataset terdiri dari 500 citra.

TABEL 1
PERUBAHAN DATASET

Label Citra Original	Label Citra yang Disederhanakan
 HPMK	 Plakat
 Fancy	
 Giant	


Emas


Surga


Halfmoon


Overhalfmoon


Emas


Slayer


Veil tail


Emas


Double tail

Citra-citra yang terkumpul telah diseleksi dengan cermat untuk memastikan tidak adanya duplikasi dan keseragaman dalam dimensi gambar. Citra-citra ini kemudian akan digunakan sebagai dasar dalam proses pelatihan model.

C. Preprocessing Data

Untuk representasikan kondisi sebenarnya kami menggunakan augmentasi. Teknik augmentasi ini mencakup rotasi acak hingga maksimal 40 derajat, yang memungkinkan model untuk mengenali objek dalam berbagai orientasi. Selain itu, diterapkan juga pergeseran horizontal dan vertikal dengan nilai maksimum sebesar 20% dari dimensi gambar, yang membantu model untuk menangani variasi posisi objek dalam gambar. Transformasi geser (shear) dengan intensitas maksimal 0.2 digunakan untuk menambah variasi bentuk objek dalam dataset, sementara skala acak dengan maksimal 0.2 membantu model dalam mengenali objek dalam berbagai ukuran. Teknik augmentasi ini dilengkapi dengan flipping horizontal atau vertikal untuk lebih memperkaya variasi data, Sehingga model dapat lebih baik terhadap perubahan orientasi objek.

Model yang digunakan dalam penelitian ini adalah Convolutional Neural Network (CNN) dengan arsitektur VGG16 dan VGG19 yang telah dilatih sebelumnya (pre-trained). VGG16 dan VGG19 dipilih karena kemampuannya dalam mengekstraksi fitur-fitur penting dari gambar, sehingga mempercepat proses pelatihan dan meningkatkan kinerja model secara keseluruhan. Penggunaan model yang telah dilatih sebelumnya sangat bermanfaat dalam konteks ini, terutama mengingat keterbatasan jumlah dataset yang tersedia. Dengan memanfaatkan bobot-bobot yang telah dipelajari pada dataset besar, model dapat memulai pelatihan dengan pemahaman awal yang kuat tentang pola-pola dasar dalam gambar, yang kemudian disesuaikan lebih lanjut dengan dataset spesifik yang digunakan dalam penelitian ini.

D. Splitting Data

Dataset yang telah dikumpulkan kemudian dibagi menjadi beberapa percobaan pembagian data untuk evaluasi kinerja model secara acak. Data dibagi secara acak untuk memastikan distribusi yang merata antara pelatihan, validasi, dan pengujian. Sebanyak 320 citra dialokasikan untuk pelatihan (train data), yang digunakan untuk melatih model agar dapat mengenali pola-pola dalam data. Sebanyak 80 citra digunakan sebagai data validasi (validation data), yang berfungsi untuk memantau kinerja model selama pelatihan dan membantu mengatur parameter hiper (hyperparameter tuning) tanpa mempengaruhi model secara langsung. Selain itu, 100 citra lainnya disisihkan sebagai data pengujian (test data), yang digunakan untuk mengevaluasi kinerja model secara keseluruhan setelah proses pelatihan selesai.

Sebelum proses pelatihan dimulai, seluruh data menjalani serangkaian transformasi dan augmentasi. Transformasi ini mencakup normalisasi dan penyesuaian ukuran citra agar sesuai dengan input yang diperlukan oleh model. Augmentasi data diterapkan dengan tujuan utama untuk mencegah overfitting, yaitu situasi di mana model terlalu menyesuaikan diri dengan data pelatihan sehingga kehilangan kemampuannya untuk menggeneralisasi pada data baru. Teknik augmentasi seperti rotasi, flipping, pergeseran, dan skala acak digunakan untuk menambah variasi pada data pelatihan, sehingga model dapat belajar dari berbagai representasi yang lebih luas.

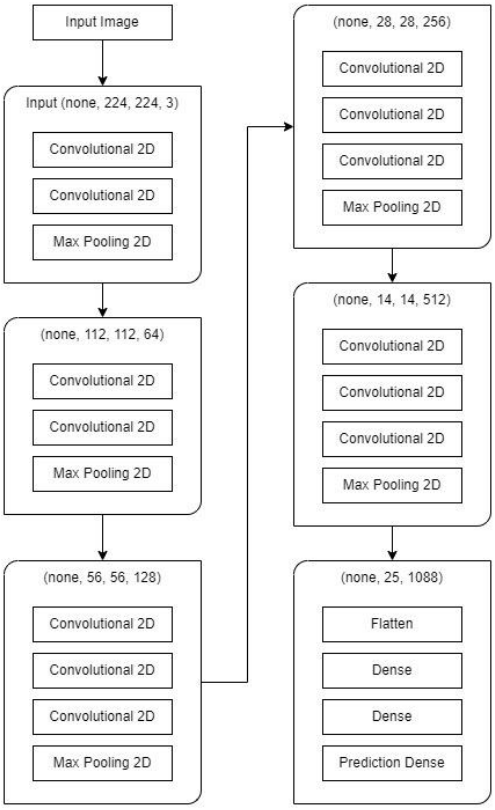
E. Classification

Dataset yang telah dibagi menjadi data pelatihan dan data validasi kemudian akan digunakan untuk melatih model Convolutional Neural Network (CNN) dengan memanfaatkan

TensorFlow, pustaka dari python yang digunakan untuk melakukan perhitungan aritmatika dengan cepat. TensorFlow dapat digunakan untuk membuat model pembelajaran mendalam (Deep Learning) secara langsung atau dengan menggunakan pustaka pembungkus yang dibangun di atas TensorFlow. TensorFlow dapat berjalan pada sistem single-CPU, GPU, serta pada perangkat seluler dan sistem terdistribusi berskala besar yang terdiri dari ratusan mesin [9].

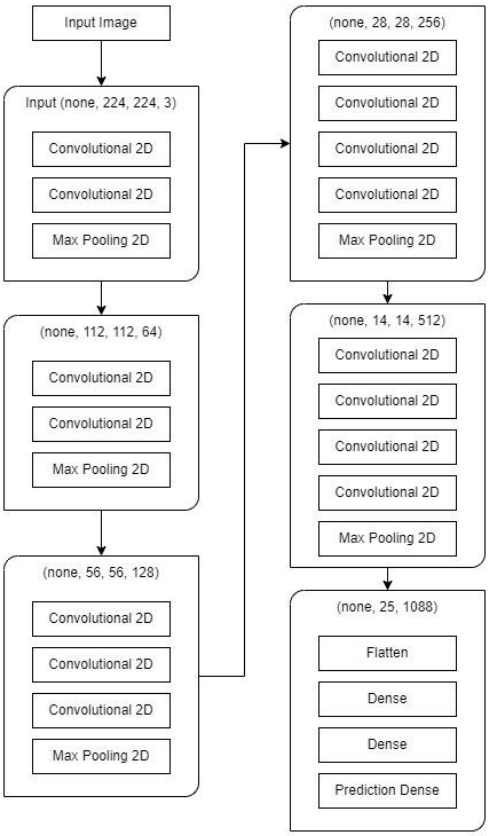
Dalam penelitian ini, digunakan arsitektur model Convolutional Neural Network (CNN) berbasis transfer learning dengan memanfaatkan model dasar (base model) VGG16 yang telah dilatih sebelumnya pada dataset ImageNet. Arsitektur ini memanfaatkan lapisan konvolusi yang sudah terdapat pada base model untuk ekstraksi fitur, kemudian dilanjutkan dengan lapisan fully connected untuk klasifikasi gambar spesifik yang digunakan dalam penelitian.

Base Model (VGG16) adalah model CNN dengan 13 lapisan konvolusi yang dilengkapi dengan fungsi aktivasi ReLU, diikuti oleh MaxPooling layers untuk mengurangi dimensi data. Setiap lapisan konvolusi menggunakan filter berukuran 3x3, dan jumlah filter bertambah secara bertahap di setiap blok konvolusi, dari 64 filter pada blok pertama hingga 512 filter pada blok akhir. Fungsi dari lapisan konvolusi ini adalah mengekstraksi fitur dari gambar, seperti tepi, tekstur, dan pola, yang relevan dengan tugas klasifikasi.



Gambar 2. Arsitektur vgg16

Sebagai perbandingan, VGG19 adalah versi yang lebih dalam dari VGG16, dengan 16 lapisan konvolusi dan 3 lapisan fully connected, sehingga total memiliki 19 lapisan. VGG19 menggunakan konfigurasi yang sama dengan VGG16, yaitu filter 3x3 pada setiap lapisan konvolusi dan fungsi aktivasi ReLU. Namun, dengan penambahan tiga lapisan konvolusi ekstra, VGG19 diharapkan mampu mengekstraksi fitur yang lebih kompleks dari gambar.



Gambar 3. Arsitektur vgg19

Pembekuan base model, seluruh lapisan pada base model dibekukan selama proses pelatihan, yang berarti bobot-bobot pre-trained pada lapisan-lapisan konvolusi tersebut tidak akan diperbarui. Hal ini bertujuan agar model dapat memanfaatkan fitur-fitur yang telah dipelajari dari dataset ImageNet tanpa perlu melatih ulang lapisan-lapisan konvolusi tersebut. Teknik ini sangat membantu dalam mengurangi overfitting dan mempercepat proses pelatihan, terutama ketika dataset yang digunakan relatif kecil.

Setelah base model, ditambahkan beberapa lapisan tambahan (fully connected layers) untuk menyesuaikan model dengan tugas klasifikasi baru, yaitu klasifikasi gambar ikan cupang. Berikut adalah detail dari lapisan tambahan flatten Layer, digunakan untuk meratakan keluaran dari base model yang berbentuk tensor 3D menjadi vektor 1D. Dense Layer (512 units, ReLU activation), lapisan fully connected ini berfungsi untuk belajar pola dari fitur yang telah diekstraksi oleh lapisan konvolusi pada base model. Aktivasi yang digunakan adalah ReLU. Dropout Layer (rate = 0.5), untuk mengurangi overfitting, Dropout dengan rate 50%

digunakan untuk secara acak menonaktifkan separuh neuron selama pelatihan. Output Layer (10 units, Softmax activation): Lapisan ini terdiri dari 10 neuron, sesuai dengan jumlah kelas yang ingin diklasifikasikan. Fungsi aktivasi Softmax digunakan untuk menghasilkan distribusi probabilitas untuk setiap kelas.

Penelitian ini juga melibatkan beberapa pustaka pendukung, termasuk Matplotlib, yang digunakan untuk memplot hasil presentasi terkait dengan prediksi model dari gambar system [9], NumPy, berperan untuk menyediakan koleksi besar fungsi matematika tingkat tinggi untuk mendukung pemrosesan array dan matriks multi-dimensi yang besar. OpenCV, yang dimanfaatkan untuk digunakan untuk pengeditan gambar (misalnya, untuk melakukan konversi ke skala abu-abu). Ini adalah pustaka Python yang dapat digunakan untuk menyelesaikan masalah visi mekanik. OpenCV mendukung berbagai bahasa pemrograman seperti C++, Python, Java, dan lainnya. OpenCV juga didukung oleh berbagai platform, termasuk Windows, Linux, dan macOS. OpenCV Python hanyalah sebuah kelas pembungkus untuk pustaka asli C++ agar dapat digunakan dengan Python. Dengan menggunakan pustaka ini, semua struktur tabel OpenCV diubah dari/ke tabel NumPy [9].

III. HASIL DAN PEMBAHASAN

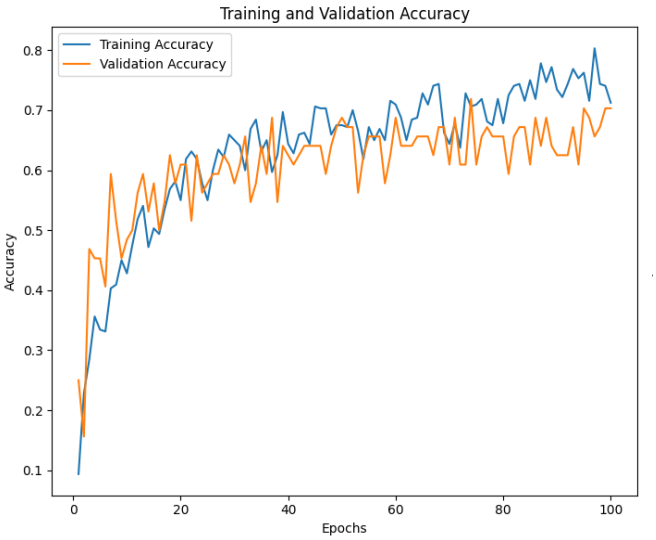
Penelitian ini dibagi ke dalam beberapa percobaan dengan pembagian data yang terdiri dari 320 citra untuk pelatihan, 80 citra untuk validasi, dan 100 citra untuk pengujian. Dalam upaya meningkatkan akurasi model yang digunakan, mengingat keterbatasan ukuran dataset, arsitektur transfer learning VGG16 dan VGG19 diterapkan. Pemilihan kedua arsitektur ini bertujuan untuk membandingkan performa model dan menentukan pendekatan yang paling optimal dalam menghasilkan hasil klasifikasi yang akurat. Selain itu, optimizer Adam dipilih sebagai algoritma optimasi utama karena kemampuannya dalam mengadaptasi learning rate secara individual untuk setiap parameter model, sehingga memungkinkan proses pelatihan menjadi lebih efisien dan konvergen. Selain itu, Adam juga sering digunakan sebagai pengganti metode pengurangan gradien stokastik tradisional dan memiliki kemiripan dengan RMSprop [16].

Dalam percobaan ini, arsitektur VGG16 digunakan untuk mengevaluasi kinerja optimizer Adam melalui dua tahap pengujian dengan variasi learning rate 0,001 dan 0,0001, serta jumlah epoch 100 dan 200. Variasi learning rate tersebut bertujuan untuk menemukan nilai optimal yang dapat mengurangi kesalahan generalisasi pada model. Selain itu, optimizer RMSprop juga diuji pada arsitektur VGG16 dengan learning rate 0,00001 dan jumlah epoch 100 serta 200. Pemilihan learning rate yang lebih kecil pada RMSprop didasarkan pada sensitivitas algoritma terhadap nilai tersebut.

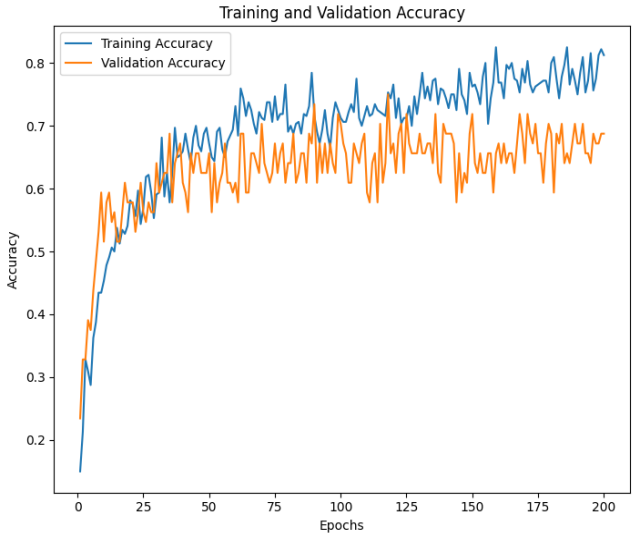
Pada percobaan optimizer adam dengan learning rate 0,001 dengan jumlah epoch 100 hingga 200 dapat dilihat pada gambar 4 dan gambar 5. Peningkatan jumlah epoch dari 100 ke 200 sedikit meningkatkan akurasi validasi, namun akurasi pada data test tetap stagnan di 66%. Hal ini menunjukkan bahwa meskipun model terus belajar lebih baik pada data training, performa pada data test tidak mengalami peningkatan. Ini bisa menjadi indikasi bahwa model mungkin mulai mengalami overfitting, di mana peningkatan akurasi pada validasi tidak diterjemahkan ke performa yang lebih baik pada data baru (test).

Untuk mengatasi masalah overfitting pada model, beberapa strategi telah diterapkan seperti membekukan (freeze) layer pada base model dan menambahkan layer dropout dengan nilai 0.5 untuk mengurangi ketergantungan model pada neuron tertentu. Pembekuan layer pada model dasar, yang sudah dilatih pada dataset besar seperti VGG16, membatasi kapasitas model untuk belajar terlalu banyak dari dataset kecil, sehingga mengurangi risiko overfitting. Selain itu, penggunaan dropout membantu meningkatkan generalisasi dengan secara acak mengabaikan sebagian neuron selama pelatihan.

Namun, penggunaan learning rate yang lebih tinggi (0.001) dengan optimizer Adam tampaknya menyebabkan overfitting, di mana akurasi validasi tidak meningkat secara konsisten. Untuk memperbaiki hal ini, menurunkan learning rate (misalnya menjadi 0.0001) bisa membuat proses pembelajaran lebih stabil. Selain itu, penggunaan ModelCheckpoint untuk menyimpan model terbaik berdasarkan nilai val_loss membantu memastikan model yang paling optimal digunakan meskipun overfitting terjadi setelah beberapa epoch.

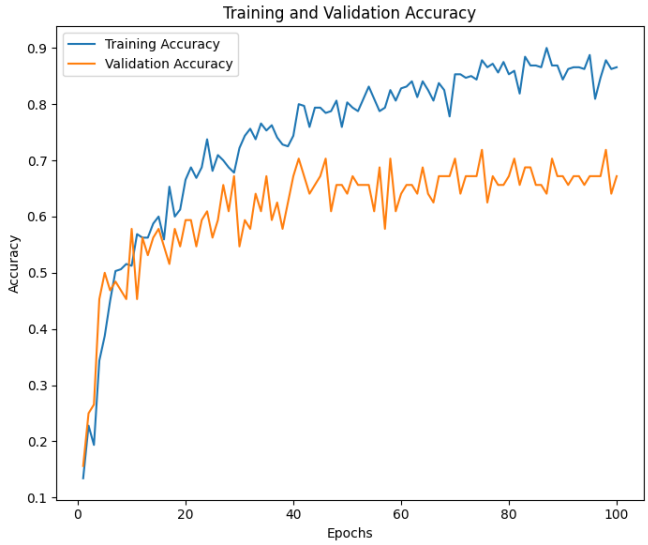


Gambar 4. Adam learning rate 0,001 epoch 100

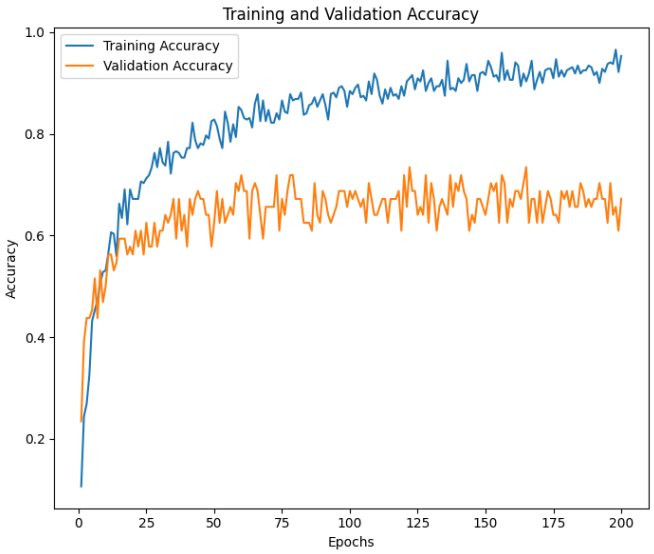


Gambar 5. Adam learning rate 0,001 epoch 200

Pada percobaan optimizer adam dengan learning rate 0,0001 dengan jumlah epoch 100 hingga 200 dapat dilihat pada gambar 6 dan gambar 7. Learning rate yang lebih rendah (0,0001) menghasilkan peningkatan yang lebih baik pada akurasi test dibandingkan dengan learning rate 0,001. Penambahan epoch dari 100 ke 200 meningkatkan akurasi validasi dan test, meskipun tidak secara signifikan. Ini menunjukkan bahwa model dengan learning rate yang lebih rendah dapat beradaptasi lebih baik dan menunjukkan generalisasi yang lebih baik pada data test, dibandingkan dengan learning rate yang lebih tinggi.

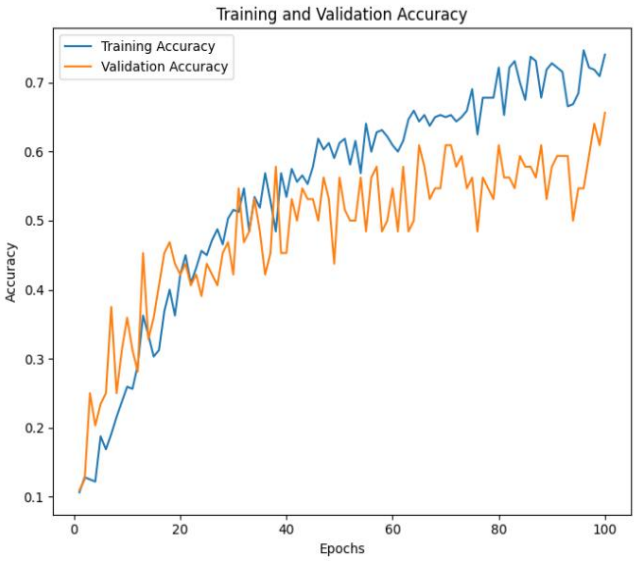


Gambar 6. Adam learning rate 0,0001 epoch 100

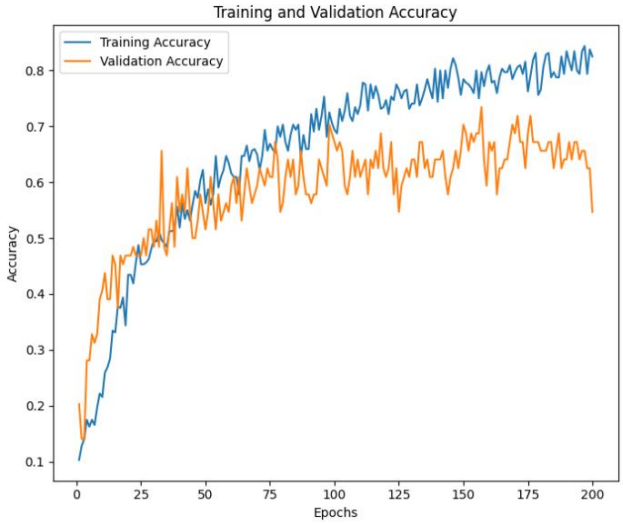


Gambar 7. Adam learning rate 0,0001 epoch 200

Pada percobaan optimizer RMSprop dengan learning rate 0,00001 dengan jumlah epoch 100 hingga 200 dapat dilihat pada gambar 8 dan gambar 9. RMSprop dengan learning rate yang sangat rendah (0,00001) menunjukkan peningkatan akurasi validasi dan test yang signifikan saat jumlah epoch ditingkatkan dari 100 ke 200. Model dengan optimizer ini menunjukkan stabilitas yang baik, meskipun akurasi validasi dan test sedikit lebih rendah dibandingkan dengan optimizer Adam pada learning rate yang sama. Ini menunjukkan bahwa RMSprop mampu mencapai performa yang baik, meskipun tidak setinggi Adam, dan dapat mengurangi risiko overfitting yang lebih besar.

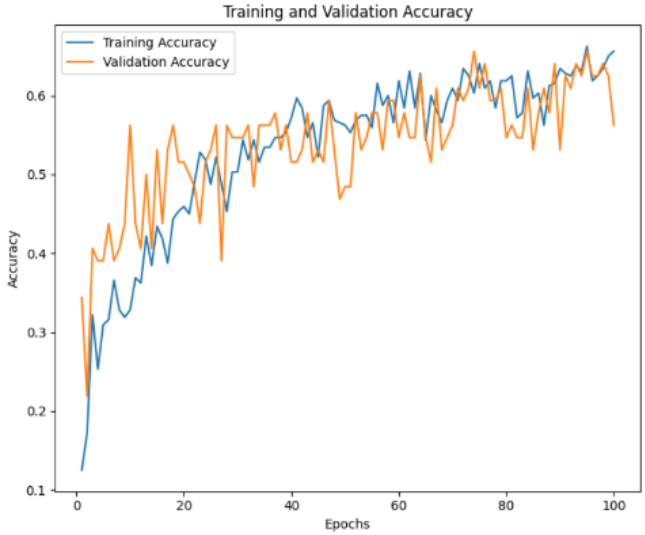


Gambar 8. RMSprop learning rate 0,00001 epoch 100



Gambar 9. RMSprop learning rate 0,00001 epoch 200

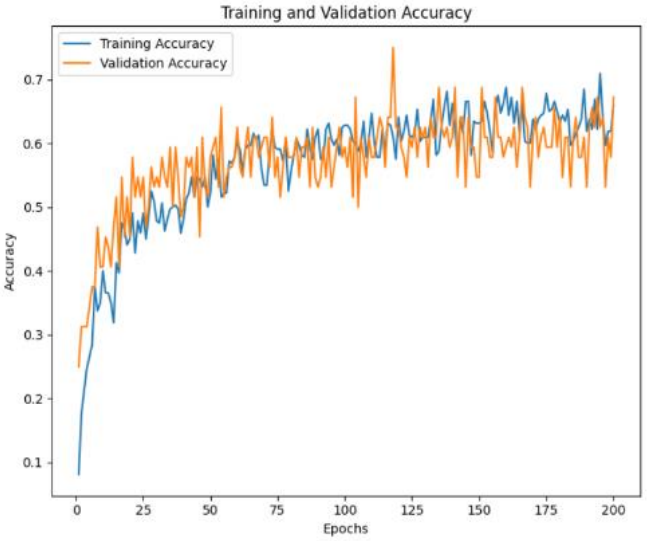
Dalam percobaan selanjutnya, arsitektur VGG19 digunakan untuk mengevaluasi kinerja optimizer Adam melalui dua tahap pengujian dengan variasi learning rate 0,001 dan 0,0001, serta jumlah epoch 100 dan 200. Variasi learning rate tersebut bertujuan untuk menemukan nilai optimal yang dapat mengurangi kesalahan generalisasi pada model. Selain itu, optimizer RMSprop juga diuji pada arsitektur VGG16 dengan learning rate 0,00001 dan jumlah epoch 100 serta 200. Pemilihan learning rate yang lebih kecil pada RMSprop didasarkan pada sensitivitas algoritma terhadap nilai tersebut.



Gambar 10. Adam learning rate 0,001 epoch 100

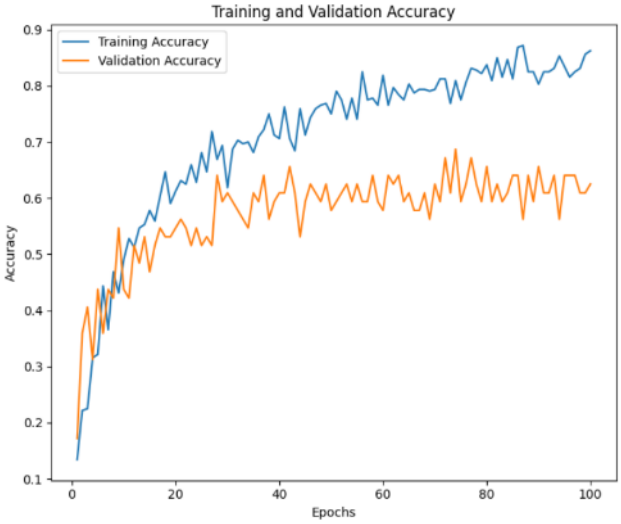
Pada percobaan optimizer adam dengan learning rate 0,001 dengan jumlah epoch 100 hingga 200 dapat dilihat pada gambar 10 dan gambar 11. Pada learning rate 0,001 dengan 100 epoch, akurasi training dan validasi cukup dekat, menunjukkan model mampu belajar dengan baik tanpa terlalu overfitting. Namun, akurasi testing dan validasi yang relatif serupa (65%) mengindikasikan bahwa model masih dalam tahap awal pembelajaran. Performanya stabil di ketiga dataset, tetapi masih ada potensi peningkatan jika model diberi lebih banyak waktu untuk berlatih.

Ketika jumlah epoch ditingkatkan menjadi 200, akurasi training meningkat menjadi 70,94%, menunjukkan bahwa model semakin baik dalam belajar dari data training. Namun, ada gap yang jelas antara akurasi training dan testing (65%), yang mengindikasikan overfitting. Akurasi validasi meningkat signifikan menjadi 75%, tetapi stagnansi di akurasi testing menunjukkan bahwa model terlalu fokus pada data validasi, kehilangan kemampuan generalisasi yang baik terhadap data baru (testing).



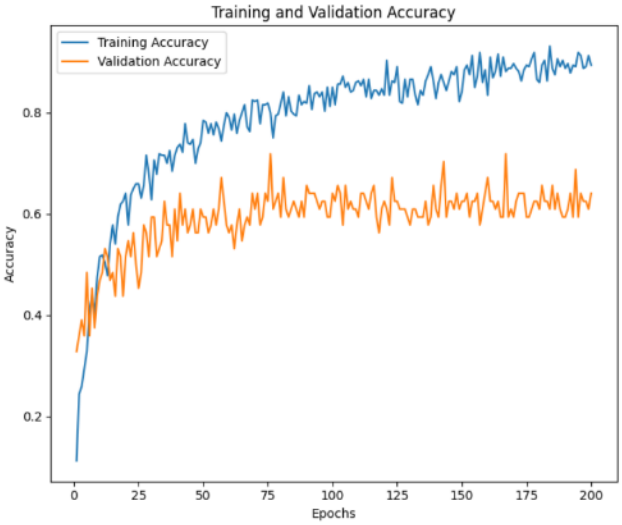
Gambar 11. Adam learning rate 0,001 epoch 200

Pada percobaan optimizer adam dengan learning rate 0,0001 dengan jumlah epoch 100 hingga 200 dapat dilihat pada gambar 12 dan gambar 13. Dengan learning rate yang lebih rendah (0,0001) dan 100 epoch, akurasi training melonjak menjadi 87,19%. Ini menunjukkan bahwa model belajar dengan sangat baik pada data training. Namun, kesenjangan yang lebih besar antara akurasi training dan validasi/testing mengindikasikan potensi overfitting. Meskipun begitu, akurasi testing meningkat menjadi 70%, yang lebih tinggi dari hasil pada learning rate 0,001. Ini menunjukkan bahwa learning rate rendah membuat model belajar lebih lambat namun lebih akurat.



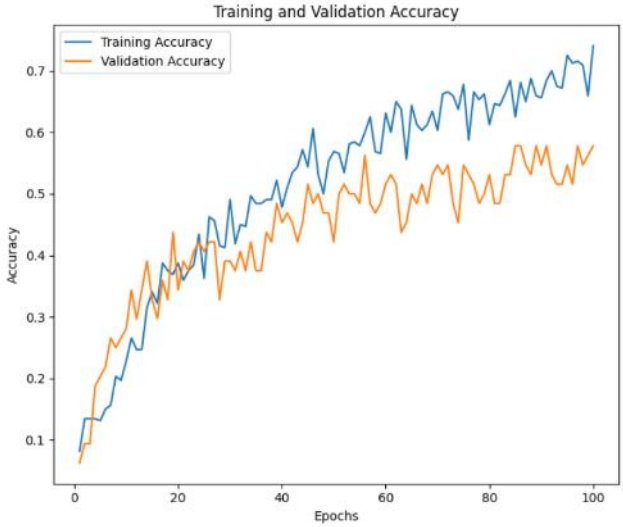
Gambar 12. Adam learning rate 0,0001 epoch 100

Pada 200 epoch, akurasi training mencapai 91,87%, yang menandakan model sangat mahir mengenali pola pada data training. Namun, gap yang cukup besar dengan akurasi validasi (71,88%) dan testing (70%) tetap ada, menunjukkan overfitting. Meski begitu, performa model pada testing tetap stabil di 70%, menandakan learning rate 0,0001 dengan 200 epoch memberikan hasil yang relatif optimal, meskipun overfitting perlu diwaspadai.



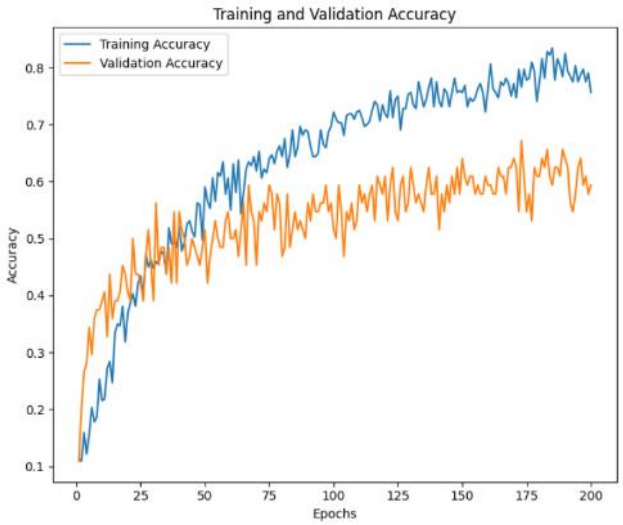
Gambar 13. Adam learning rate 0,0001 epoch 200

Pada percobaan optimizer RMSprop dengan learning rate 0,00001 dengan jumlah epoch 100 hingga 200 dapat dilihat pada gambar 14 dan gambar 15. Pada learning rate yang sangat rendah (0,00001) dengan 100 epoch, akurasi training hanya mencapai 74,04%, yang menunjukkan bahwa model masih belum sepenuhnya belajar dari data. Selain itu, akurasi validasi yang jauh lebih rendah (57,81%) menunjukkan underfitting, di mana model kesulitan mengenali pola dari data. Akurasi testing juga rendah (63%), mengindikasikan bahwa model tidak mampu menggeneralisasi dengan baik pada data baru.



Gambar 14. RMSprop learning rate 0,00001 epoch 100

Setelah jumlah epoch dinaikkan menjadi 200, akurasi training meningkat signifikan menjadi 83,44%, yang menandakan bahwa model belajar lebih baik dari data. Akurasi validasi dan testing juga mengalami peningkatan (67,19% dan 72% masing-masing), menunjukkan bahwa model dengan learning rate rendah ini mampu meningkatkan performa seiring waktu. Namun, akurasi validasi tetap lebih rendah daripada akurasi testing, yang mungkin menunjukkan bahwa model lebih unggul pada data testing daripada data validasi. Learning rate yang sangat kecil memungkinkan model untuk belajar dengan lambat dan bertahap, namun ini membutuhkan epoch yang lebih banyak untuk performa optimal.



Gambar 15. RMSprop learning rate 0,00001 epoch 200

Pada percobaan ini, digunakan learning rate sebesar 0.00001 untuk optimizer RMSprop. Learning rate yang sangat kecil ini dipilih karena karakteristik RMSprop yang dirancang untuk mengatur langkah optimasi berdasarkan nilai rata-rata kuadrat gradien sebelumnya. RMSprop adalah salah satu optimizer yang secara adaptif mengubah ukuran langkah (step

size) untuk setiap parameter, sehingga lebih efektif dalam menangani gradien yang tidak stabil, terutama pada dataset yang lebih kecil atau kompleks.

Alasan penggunaan learning rate yang kecil (0.00001) pada RMSprop adalah untuk memastikan stabilitas dan kehalusan konvergensi selama pelatihan. Learning rate yang terlalu besar pada RMSprop (misalnya 0.001 atau 0.0001) dapat menyebabkan model melompat-lompat di sekitar nilai minimum lokal, sehingga menghambat tercapainya solusi optimal. Sebaliknya, learning rate yang lebih kecil memungkinkan model untuk mengambil langkah-langkah kecil menuju solusi yang optimal, yang lebih sesuai untuk menghindari overshoot (langkah terlalu besar melampaui minimum) dan memastikan model belajar dengan stabil.

Dibandingkan dengan optimizer lain, seperti Adam, yang menggunakan momentum dan adaptasi learning rate, RMSprop lebih cocok untuk penggunaan pada learning rate yang lebih kecil. Hal ini disebabkan oleh pendekatan adaptif RMSprop yang fokus pada normalisasi gradien, sehingga learning rate kecil seperti 0.00001 tetap efektif dalam mencapai konvergensi yang lebih stabil, terutama pada masalah yang melibatkan dataset kecil dan kompleksitas tinggi seperti dalam penelitian ini.

TABEL II.
HASIL MODEL PERCOBAAN PERTAMA TRANSFER LEARNING VGG16

Optimizer	Learning Rate	Epoch	Akurasi Validasi	Akurasi Testing
Adam	0,001	100	71,88%	66%
Adam	0,001	200	73,44%	66%
Adam	0,0001	100	71,88%	70%
Adam	0,0001	200	73,44%	71%
RMSprop	0,00001	100	65,62%	66%
RMSprop	0,00001	200	71,88%	70%

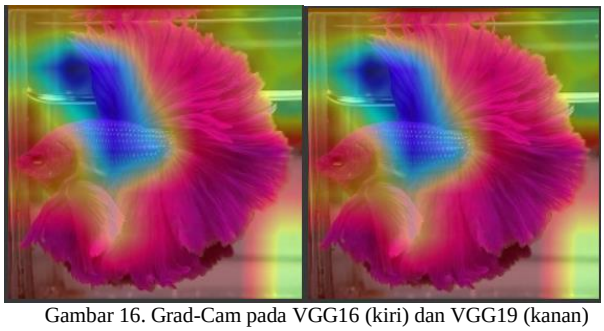
Berdasarkan tabel 2, penggunaan VGG16 dengan optimizer Adam dengan learning rate 0,001 menunjukkan peningkatan akurasi validasi ketika jumlah epoch ditingkatkan dari 100 menjadi 200, namun akurasi pada data uji tetap stagnan di angka 66%. Fenomena ini mengindikasikan bahwa meskipun model mengalami peningkatan performa pada data validasi, kemampuan generalisasi terhadap data baru tidak mengalami perbaikan yang berarti, yang dapat menjadi indikasi terjadinya overfitting. Sebaliknya, penurunan learning rate menjadi 0,0001 menghasilkan peningkatan yang lebih konsisten pada akurasi validasi dan uji, dengan akurasi uji mencapai 71% setelah 200 epoch. Hal ini menunjukkan bahwa model yang dilatih dengan learning rate yang lebih rendah memiliki kemampuan generalisasi yang lebih baik terhadap data uji, meskipun ada peningkatan risiko overfitting pada data pelatihan.

Selain itu, penggunaan optimizer RMSprop dengan learning rate yang sangat rendah (0,00001) menunjukkan hasil yang stabil, dengan peningkatan signifikan pada akurasi validasi dan uji seiring dengan penambahan jumlah epoch dari 100 menjadi 200. Meskipun akurasi validasi dan uji sedikit lebih rendah dibandingkan dengan Adam, RMSprop menunjukkan kemampuan untuk menghasilkan model yang lebih stabil dan kurang rentan terhadap overfitting. Hasil ini menekankan pentingnya pemilihan learning rate dan jumlah epoch yang tepat untuk mencapai keseimbangan optimal antara akurasi pelatihan, validasi, dan uji, serta untuk memastikan bahwa model mampu menggeneralisasi dengan baik pada data yang tidak terlihat sebelumnya. Temuan ini memberikan panduan yang berharga bagi penelitian di masa depan terkait optimalisasi hyperparameter dalam pengembangan model pembelajaran mesin.

Berdasarkan tabel 3, penggunaan VGG19 dengan optimizer Adam dengan learning rate 0,001 menghasilkan performa yang cukup stabil, tetapi terdapat indikasi overfitting pada 200 epoch di mana akurasi validasi meningkat signifikan namun akurasi testing tetap stagnan. Learning Rate 0,0001 memberikan performa yang lebih baik dengan akurasi testing yang lebih tinggi (70%) dan lebih stabil. Namun, overfitting juga terjadi karena akurasi training yang sangat tinggi dibandingkan validasi/testing. Selain itu, penggunaan optimizer RMSprop dengan learning rate 0,00001, model belajar lebih lambat dan membutuhkan lebih banyak epoch untuk mencapai performa yang lebih baik. Pada 200 epoch, akurasi testing mencapai 72%, yang lebih tinggi dibandingkan Adam pada 200 epoch, tetapi akurasi validasi masih lebih rendah. Ini menunjukkan bahwa RMSprop bisa efektif jika diberikan waktu lebih untuk belajar.

TABEL III.
HASIL MODEL PERCOBAAN PERTAMA TRANSFER LEARNING VGG19

Optimizer	Learning Rate	Epoch	Akurasi Validasi	Akurasi Testing
Adam	0,001	100	65,62%	65%
Adam	0,001	200	75%	65%
Adam	0,0001	100	68,75%	70%
Adam	0,0001	200	71,88%	70%
RMSprop	0,00001	100	57,81%	63%
RMSprop	0,00001	200	67,19%	72%



Gambar 16. Grad-Cam pada VGG16 (kiri) dan VGG19 (kanan)

Visualisasi Grad-CAM digunakan untuk memahami area yang mendapat perhatian utama dalam proses identifikasi ikan cupang, seperti yang ditunjukkan pada Gambar X dan Gambar Y. Grad-CAM menghasilkan peta lokalisasi yang menyoroti bagian penting dalam gambar yang berkontribusi pada prediksi model, dengan menggunakan gradien dari

lapisan konvolusional terakhir untuk memetakan area yang signifikan [17].

Hasil visualisasi untuk model VGG16 dan VGG19 menunjukkan pola perhatian yang serupa, dengan fokus utama pada tubuh dan sirip ikan cupang, yang merupakan fitur utama dalam membedakan jenis ikan. Hal ini menunjukkan bahwa kedua model menggunakan fitur visual yang sama dan relevan dalam pengambilan keputusan. Dengan demikian, baik VGG16 maupun VGG19 secara konsisten mengekstraksi informasi penting yang mendukung proses klasifikasi, yang mengindikasikan bahwa arsitektur yang berbeda tetap mampu mengenali karakteristik visual utama ikan cupang secara efektif.

IV. KESIMPULAN

Berdasarkan percobaan yang telah dilakukan, diketahui bahwa pemilihan learning rate dan jumlah epoch yang tepat sangat krusial dalam meningkatkan kinerja model. Pada percobaan dengan arsitektur VGG16 dan VGG19, penggunaan optimizer Adam dengan learning rate yang lebih rendah (0,0001) menghasilkan peningkatan akurasi yang konsisten, terutama pada data testing, di mana akurasi mencapai 71% pada VGG16 dan 70% pada VGG19 setelah 200 epoch. Ini menunjukkan bahwa Adam dengan learning rate yang lebih rendah mampu meningkatkan kemampuan generalisasi model secara efektif, sekaligus mengurangi risiko overfitting yang lebih tinggi pada learning rate yang lebih besar.

Sementara itu, optimizer RMSprop dengan learning rate 0,00001 menunjukkan stabilitas yang baik dalam peningkatan akurasi seiring bertambahnya epoch. Pada VGG19, akurasi data uji mencapai 72% setelah 200 epoch, yang menandakan bahwa RMSprop dapat menghasilkan performa optimal dengan durasi pelatihan yang lebih lama. Secara keseluruhan, kombinasi learning rate yang sesuai dan jumlah epoch yang tepat terbukti meningkatkan performa model, memastikan generalisasi yang lebih baik, dan meminimalkan overfitting.

Untuk penelitian selanjutnya, disarankan untuk mengeksplorasi teknik regularisasi seperti dropout atau L1/L2 regularization guna mengurangi overfitting dan meningkatkan generalisasi model. Eksperimen dengan variasi hyperparameter seperti learning rate, jumlah epoch, serta penggunaan optimizer lain seperti Nadam atau AdaGrad juga penting untuk menemukan konfigurasi optimal. Menguji model dengan data yang lebih beragam dan menerapkan cross-validation akan membantu memastikan generalisasi yang lebih kuat. Terakhir, eksplorasi arsitektur model yang lebih kompleks dapat berkontribusi dalam meningkatkan performa model secara keseluruhan.

DAFTAR PUSTAKA

[1] M. T. A. Syech Ahmad and B. Sugiarto, "Implementasi Convolutional Neural Network (CNN) Untuk Klasifikasi Ikan Cupang Berbasis Mobile," *Digital Transformation Technology*, vol. 3, no. 2, pp. 712–723, Dec. 2023, doi: 10.47709/digitech.v3i2.3245.

[2] W. D. Setyawan, A. Nilogiri, and Q. A'yun, "Implementasi Convolution Neural Network (CNN) Untuk Klasifikasi Pada Citra Ikan Cupang Hias," *JTIK (Jurnal Teknik Informatika Kaputama)*, vol. 7, no. 1, 2023, doi: 10.59697/jtik.v7i1.45.

[3] Y. Tian, "Artificial Intelligence Image Recognition Method Based On Convolutional Neural Network Algorithm," *IEEE Access*, vol. 8, 2020, doi: 10.1109/ACCESS.2020.3006097.

[4] U. Sri Rahmadhani and N. Lysbetti Marpaung, "Klasifikasi Jamur Berdasarkan Genus Dengan Menggunakan Metode CNN," *Jurnal Informatika: Jurnal pengembangan IT*, vol. 8, no. 2, pp. 169–173, 2023.

[5] I. Ariawan, W. Aprizal Arifin, A. Armelita Rosalia, and N. Tufailah, "Klasifikasi Tiga Genus Ikan Karang Menggunakan Convolution Neural Network," *Jurnal Ilmu dan Teknologi Kelautan Tropis*, vol. 14, no. 2, pp. 205–216, 2024, doi: 10.29244/jitkt.v14i1.33633.

[6] I. Anugrah, A. Cendekia Siregar, and B. C. Octariadi, "Perbandingan Model Arsitektur Cnn Dengan Metode Transfer Learning Untuk Klasifikasi Spesies Ikan Laut," *Progresif: Jurnal Ilmiah Komputer*, vol. 20, no. 1, pp. 444–453, 2024, doi: 10.35889/progresif.v20i1.1834.

[7] A. Azis, "Identifikasi Jenis Ikan Menggunakan Model Hybrid Deep Learning Dan Algoritma Klasifikasi," *SEBATIK*, pp. 201–206, 2020, Accessed: Aug. 23, 2024. [Online]. Available: https://jurnal.wicida.ac.id/index.php/sebatik/article/view/1057

[8] N. Abdurrahman, B. Rahmat, and A. N. Sihananto, "Perbandingan Performa Klasifikasi Citra Ikan Menggunakan Metode K-Nearest Neighbor (K-NN) Dan Convolutional Neural Network (CNN)," *Jurnal Sistem Informasi dan Informatika (JUSIFOR)*, vol. 2, no. 2, pp. 84–93, Dec. 2023, doi: 10.33379/jusifor.v2i2.3728.

[9] G. Papastergiou, V. C. Gerogiannis, A. Xenakis, G. Papastergiou, and G. Stamoulis, "Applying A Convolutional Neural Network In An Iot Robotic System For Plant Disease Diagnosis," 2023. [Online]. Available: https://www.researchgate.net/publication/343386581

[10] A. M. Ismael and A. Şengür, "Deep Learning Approaches For COVID-19 Detection Based On Chest X-Ray Images," *Expert Syst Appl*, vol. 164, p. 114054, Feb. 2021, doi: 10.1016/j.eswa.2020.114054.

[11] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A Survey Of The Recent Architectures Of Deep Convolutional Neural Networks," *Artif Intell Rev*, vol. 53, no. 8, 2020, doi: 10.1007/s10462-020-09825-6.

[12] F. A. Mohammed, K. K. Tune, B. G. Assefa, M. Jett, and S. Muhie, "Medical Image Classifications Using Convolutional Neural Networks: A Survey Of Current Methods And Statistical Modeling Of The Literature," Mar. 01, 2024, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/make6010033.

[13] A. Saber, M. Sakr, O. M. Abo-Seida, A. Keshk, and H. Chen, "A Novel Deep-Learning Model For Automatic Detection And Classification Of Breast Cancer Using The Transfer-Learning Technique," *IEEE Access*, vol. 9, 2021, doi: 10.1109/ACCESS.2021.3079204.

[14] D. R. Sarvamangala and R. V. Kulkarni, "Convolutional Neural Networks In Medical Image Understanding: A Survey," *Evol Intell*, vol. 15, no. 1, pp. 1–22, Mar. 2022, doi: 10.1007/s12065-020-00540-3.

[15] L. Alzubaidi *et al*., "Review Of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions," *J Big Data*, vol. 8, no. 1, 2021, doi: 10.1186/s40537-021-00444-8.

[16] M. Reyad, A. M. Sarhan, and M. Arafa, "A Modified Adam Algorithm For Deep Neural Network Optimization," *Neural Comput Appl*, vol. 35, no. 23, 2023, doi: 10.1007/s00521-023-08568-z.

[17] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations From Deep Networks Via Gradient-Based Localization," *Int J Comput Vis*, vol. 128, no. 2, 2020, doi: 10.1007/s11263-019-01228-7.