INTERNAL Information System Journal
INTERNAL (Information System Journal)
Volume 8 No. 1 | Juni 2025 : 70-84
DOI: 10.32627
http://jurnal.masoemuniveristy.ac.id/index.php/internal
p-ISSN: 2621-9433
e-ISSN: 2656-0259

Klasifikasi Citra Ras Kucing Berbasis CNN dengan Metode MobileNet-V2

Ramadhan Anugrah Hermawan[1], Ichsan Taufik[2], Yana Aditia Gerhana[3]
[1,2,3]Teknik Informatika, UIN Sunan Gunung Djati Bandung, Indonesia
1207050101@student.uinsgd.ac.id

Info Artikel

Sejarah artikel:
Diterima Mei 2025
Direvisi Juni 2025
Disetujui Juni 2025
Diterbitkan Juni 2025

ABSTRACT

This study evaluates the performance of a Convolutional Neural Network (CNN) using the MobileNet-V2 architecture in classifying four cat breeds. The lack of public understanding in distinguishing cat breeds, especially due to the prevalence of mixed breeds, presents a significant challenge in accurate identification. The model was tested across multiple epochs to observe training and validation accuracy, aiming to assess its effectiveness and stability. Experimental results show that the highest validation accuracy of 93.81% was achieved at epoch 90. Although the model performed well, further optimization is needed to address overfitting and improve generalization capability. This research contributes to the development of an automated breed identification system that can be applied in education, adoption processes, and veterinary healthcare.

Keywords **:** *Accuracy; Deep Learning; Epoch; Cat Classification; MobileNet-V2.*

ABSTRAK

Penelitian ini melakukan evaluasi terhadap performa CNN dengan arsitektur *MobileNet-V2* dalam mengklasifikasikan empat jenis ras kucing. Kurangnya pemahaman masyarakat dalam membedakan ras kucing menjadi tantangan tersendiri dan kurangnya pemahaman masyarakat terhadap ciri fisik tiap ras, terutama karena banyaknya ras hasil persilangan. Model diuji pada beberapa epoch untuk mengamati tingkat akurasi pelatihan dan validasi guna menilai efektivitas serta kestabilannya. Hasil percobaan menunjukkan bahwa pada epoch ke-90, model mencapai akurasi validasi tertinggi sebesar 93,81%. Walaupun hasilnya sudah cukup baik, diperlukan upaya optimasi lebih lanjut untuk mengatasi masalah *overfitting* dan meningkatkan kemampuan generalisasi model. Penelitian ini memberikan kontribusi penting dalam pengembangan sistem otomatis untuk identifikasi ras kucing yang dapat digunakan dalam bidang edukasi, proses adopsi, serta kesehatan hewan.

Kata Kunci : Akurasi; *Deep Learning; Epoch*; Klasifikasi Kucing; MobileNet-V2.

PENDAHULUAN

Kucing merupakan salah satu hewan peliharaan yang populer di seluruh dunia. Kurangnya pengetahuan mengenai jenis ras kucing menjadi permasalahan, apalagi banyak kucing saat ini merupakan hasil kawin persilangan. Untuk mengidentifikasi ras kucing secara akurat, diperlukan sistem yang mampu mengenali ciri-ciri fisiknya. Pemahaman ras kucing penting untuk kesehatan dan perawatan. Sistem klasifikasi otomatis berbasis gambar membantu pengambilan keputusan oleh pemilik dan dokter hewan. [1]. Kucing dengan garis keturunan tercatat disebut ras murni (pure breed), dengan sekitar 32 ras yang diakui secara

internasional, seperti Anggora, Persia, Himalaya, dan Kampung [2]. Mengutip secara langsung "Penerapan *Convolutional Neural Network* pada Klasifikasi Jenis Ras Kucing Menggunakan ResNet50V2". Penelitian ini mengklasifikasikan 12 ras kucing menggunakan arsitektur CNN ResNet50 dengan optimasi stochastic gradient descent (SGD). Model mencapai akurasi *training* 99,17% dan akurasi validasi 88,13%, dengan loss masing-masing 0,3760 (training) dan 0,7857 (validasi). Evaluasi menggunakan *confusion matrix, precision, recall, f1-score*, dan *support* menunjukkan rata-rata akurasi 88% [3]. Mengutip secara langsung "Klasifikasi Ras pada Kucing menggunakan Algoritma *Convolutional Neural Network(CNN*)". Penelitian ini menyimpulkan bahwa model *Xception* dengan *Transfer Learning* dan *Fine Tuning* memberikan performa terbaik, mencapai akurasi 93,75%, *precision* 93,74%, *recall* 93,56%, *dan f1-score* 93,64% *[4].*

Dalam beberapa tahun terakhir, kemajuan teknologi kecerdasan buatan (AI) telah membawa perubahan signifikan di berbagai sektor, termasuk di bidang akademik dan pendidikan [5]. *Google Lens* adalah aplikasi inovatif dari Google yang memanfaatkan teknologi pengenalan gambar dan kecerdasan buatan untuk mengenali objek, termasuk tanaman, melalui foto dari kamera ponsel. Sementara itu, *QR Code (Quick Response Code)* adalah kode dua dimensi yang dapat membantu perangkat seluler untuk mengarahkan pengguna ke informasi yang relevan, seperti video tutorial dan artikel tentang manfaat tanaman [6].

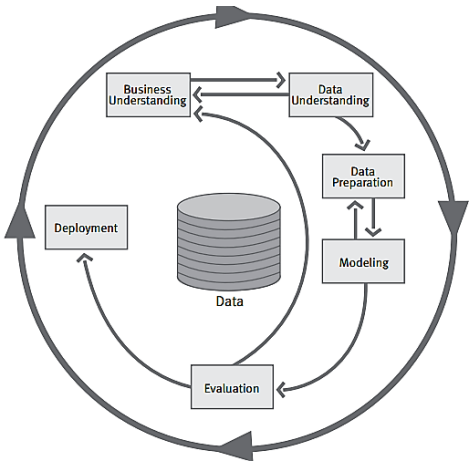
Klasifikasi objek adalah aspek penting dalam Computer Vision, yang fokus pada pengembangan sistem untuk memahami informasi visual [7].*Citra digital* adalah gambar yang dapat diproses oleh komputer.Gambar ini dapat dimodifikasi melalui berbagai metode, seperti pengambilan fitur, perubahan sudut pandang, atau penyesuaian ukuran. Salah satu cara untuk memanfaatkan data gambar adalah melalui klasifikasi menggunakan algoritma *Convolutional Neural Network (CNN*) [8].MobileNet adalah arsitektur CNN yang efisien dan ringan, cocok untuk perangkat dengan keterbatasan daya komputasi [9].*MobileNetV2*, yang dikembangkan oleh peneliti Google, adalah versi lebih lanjut dari MobileNet, menawarkan efisiensi yang lebih tinggi dengan menggunakan blok sisa terbalik dan fitur *bottleneck* untuk mengurangi kompleksitas jaringan [10].

Penelitian ini bertujuan untuk mengembangkan sistem klasifikasi ras kucing berbasis *Convolutional Neural Network (CNN)* dengan menggunakan arsitektur MobileNet-V2. Data dilatih menggunakan bahasa pemograman *Python* di *Google Collaboratory*. *Python* adalah bahasa pemrograman tingkat tinggi yang dikembangkan oleh Guido Van Rossum dan dirilis pada 1991. Bahasa ini populer karena sintaksnya yang sederhana, bersifat open source, serta memiliki library lengkap dan dukungan komunitas yang kuat. *Python* juga multifungsi, termasuk untuk *Machine Learning dan Deep Learning*. Kode *Python* dapat ditulis menggunakan *IDE* seperti *VS Code, Sublime Text, PyCharm,* atau platform online seperti *Jupyter Notebook* dan *Google Collaboratory* [11]. Aplikasi yang dihasilkan akan memudahkan identifikasi ras kucing secara otomatis dengan tingkat akurasi dan efisiensi yang tinggi, terutama pada perangkat *mobile.*

p-ISSN: 2621-9433
e-ISSN: 2656-0259

METODE

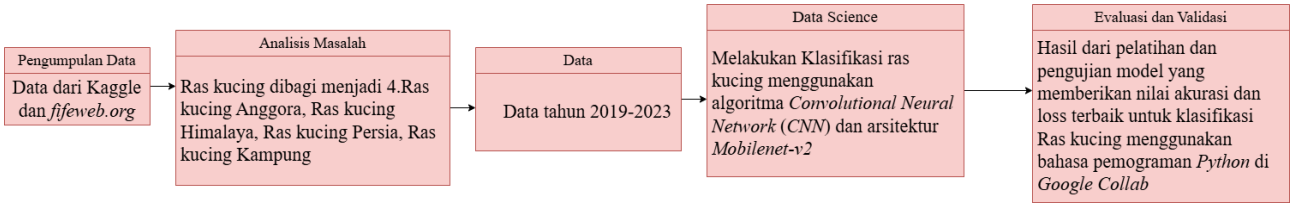
Metode penelitian ini menggunakan metode *Cross-Industry Standard Process for Data Mining (CRISP-DM)* dirancang untuk menyediakan panduan berupa tahapan-tahapan dalam proses pengumpulan data. Tahapan tersebut terdiri dari enam langkah, yaitu pemahaman bisnis, pemahaman data, persiapan data, pemodelan, evaluasi, dan penerapan [12]. Dilihat pada gambar 2.1.



Gambar 1. Tahapan CRISP-DM
Sumber : Salma Navisa, Luqman Hakim ,Aulia Nabilah [12]

Bussines Understanding

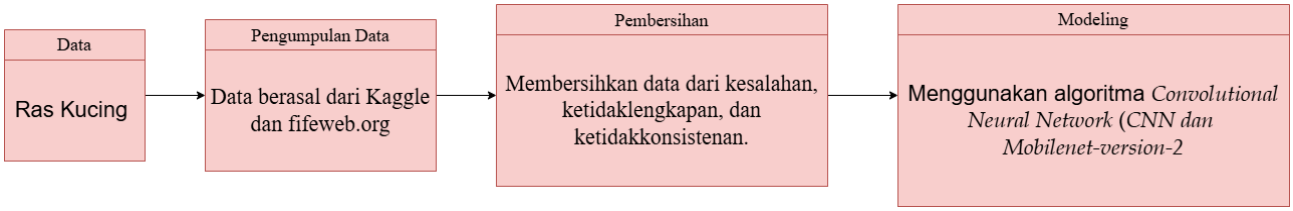
Pada tahap ini, pemahaman tentang inti dari aktivitas data yang akan dilaksanakan dan kebutuhan dari sudut pandang bisnis [13]. Dilihat pada gambar 2.2.



Gambar 2. Tahapan *Bussines Understanding*

Data *Understanding*

Pada tahap ini, fase di mana melibatkan pengumpulan data awal, mempelajari data untuk memahami karakteristiknya, mengidentifikasi masalah [13]. Dilihat pada Gambar 2.3.

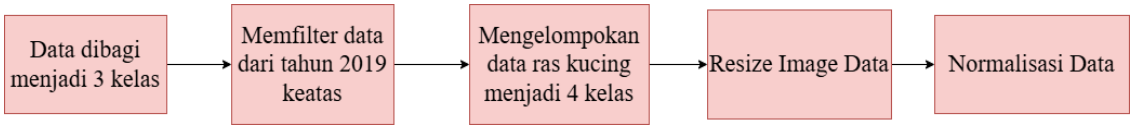


Gambar 1. Tahapan Data *Understanding*

p-ISSN: 2621-9433
e-ISSN: 2656-0259

Data *Preparation*

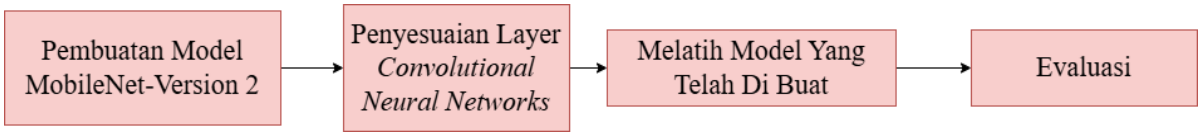
Pada tahap ini, sering disebut sebagai fase yang membutuhkan banyak tenaga kerja [13]. Aktivitas yang dilakukan melibatkan pembersihan, transformasi, dan pemilihan data yang akan digunakan untuk pelatihan. Dilihat pada Gambar 2.4.



Gambar 2. Tahapan Data Preparation

Modeling

Pada tahap ini, fase menentukan tehnik yang digunakan [13]. Dilihat pada Gambar 2.5.



Gambar 3. Modeling

Evaluation

Pada tahap ini, mencakup analisis data yang dihasilkan dari proses pemodelan yang telah dilakukan pada fase sebelumnya [13]. Dilihat pada Gambar 2.6.

Gambar 4. *Evaluation*

Melakukan *Resize Image*

Proses Resize Image dilakukan dengan menyesuaikan dimensinya menjadi 224 × 224 piksel [17]. Diubah secara otomatis menggunakan *Source Code* yang diuji dengan *Google Collaboratory*. Dengan begitu, seluruh gambar dalam dataset akan memiliki ukuran dimensi piksel yang sama, sehingga mempermudah tahap pemrosesan data dan analisis berikutnya. *Source Code* dapat dilihat dibawah ini.

Source Code :

```
    for i, class_name in enumerate ( classes ) :
class_path = os.path.join ( dataset_dir, class_name )
    for file_name in os.listdir ( class_path ) :
image_path = os.path.join ( class_path, file_name )
image = Image.open ( image_path ).convert ( RGB )
image = image.resize (( 224,224 )
x_data.append ( np.array( image ))
y_data.append ( i )
```

p-ISSN: 2621-9433
e-ISSN: 2656-0259

Melakukan Normalisasi Data

Normalisasi data dilakukan dengan mengubah nilai X dan Y. Nilai X disesuaikan agar berada dalam rentang 0 hingga 1, sementara nilai Y dikonversi menjadi angka dari 0 hingga 7 sesuai dengan jumlah kelas yang tersedia [17]. *Source Code* dapat dilihat dibawah ini.

Source Code :

```
x_train = preprocess_input ( x_train )
x_test = preprocess_input ( x_test )
y_train = to_categorical ( y_train, 4 )
y_test = to_categorical ( y_test, 4 )
print(f"y_train shape: { y_train.shape }")
print(f"y_test shape: { y_test.shape }")
```

Membangun Model *MobileNet - Version 2*

Dalam pengujian penelitian ini, model dikembangkan menggunakan arsitektur *MobileNetV2* di *Google Collaboratory*. Model ini digunakan sebagai *feature extractor* dalam sistem *deep learning* untuk klasifikasi gambar.

Source Code :

```
input_shape = (224, 224, 3)
base_model = MobileNetV2 ( include_top=False, weights='imagenet',
input_shape=input_shape )
base_model.summary ()
```

Penyesuaian *Layer Convolutional Neural Networks*

Setelah membangun model menggunakan arsitektur *MobileNetV2*, langkah selanjutnya adalah menyesuaikan layer *Convolutional Neural Networks (CNN)* untuk meningkatkan performa dalam klasifikasi gambar. Kode berikut digunakan untuk membangun model yang telah dimodifikasi dengan penyesuaian layer agar lebih optimal.

Source Code :

```
inputs = Input ( input_shape )
x = base_model ( inputs, training = False )
x = Flatten() ( x )
x = Dense ( 256, activation = 'relu',
kernel_regularizer=tf.keras.regularizers.l2 ( 0.001 )) ( x )
x = Dropout ( 0.5 ) ( x )
outputs = Dense ( 4, activation = 'softmax' ) ( x )
model = Model ( inputs, outputs )
model.summary()
```

Melatih Model MobileNet - Version 2

Setelah melakukan *input layer* pada *Convolutional Neural Network (CNN)* sesuai kebutuhan, langkah berikutnya adalah melatih model *MobileNetV2* yang telah dibangun sebelumnya. Proses pelatihan ini menggunakan berbagai teknik optimasi untuk meningkatkan akurasi dan menekan nilai loss.

Source Code :

```
model.compile ( loss = 'categorical_crossentropy',
        optimizer=tf.keras.optimizers.Adam (l earning_rate = 0.0001 ),
        metrics = ['accuracy'])
# Callbacks
early_stop = EarlyStopping ( monitor = 'val_loss', patience = 20, mode =
'min')
reduce_lr = ReduceLROnPlateau ( monitor ='val_loss', factor = 0.2, patience
= 10, min_lr = 0.00001, mode = 'min' )
# Train the model
history = model.fit (
   datagen.flow ( x_train, y_train, batch_size = 32),
   epochs = 1,
   validation_data = ( x_test, y_test ),
   callbacks = [ early_stop, reduce_lr ],
   verbose = 1)
```

Deployment

Pada tahap ini, menyajikan hasil dari model yang telah diterapkan dalam proses data mining, termasuk penyampaian informasi yang diperoleh selama analisis agar mudah dipahami oleh pengguna [12]. Tahap ini mencakup pembuatan laporan setelah mengevaluasi penelitian terkait model klasifikasi Ras kucing.

HASIL DAN PEMBAHASAN

Bussines Understanding

Pada tahap ini, tujuan, persyaratan, dan batasan penelitian ditetapkan melalui analisis mendalam guna merumuskan permasalahan yang dapat diukur [14]. Banyak orang mengalami kesulitan dalam membedakan berbagai ras kucing yang ada karena jumlahnya yang sangat beragam. Dengan memanfaatkan algoritma *Convolutional Neural Network*, diharapkan sistem ini dapat menyajikan informasi yang akurat mengenai jenis kucing berdasarkan data yang telah diuji, sehingga membantu masyarakat dalam mengenali ras kucing dengan lebih mudah.

Data *Understanding*

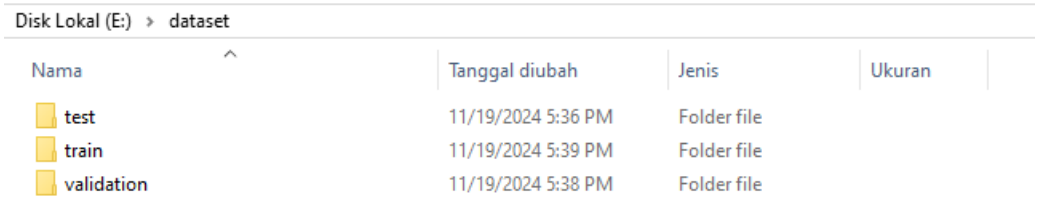
Tahap ini dimulai dengan pengumpulan data, deskripsi data, serta evaluasi kualitasnya [15]. Dalam penelitian ini, data diperoleh dari platform *Kaggle* berjumlah 1.500 Gambar dan situs *fifeweb.org* berjumlah 1.203 Gambar. Data yang digunakan merupakan kumpulan informasi yang relevan, dengan fokus pada data yang dirilis sejak tahun 2019 dan seterusnya, guna memastikan kesesuaian serta kualitas dalam proses analisis.

Data *Preparation*

Fase Data Preparation mencakup seluruh aktivitas yang diperlukan untuk membangun dataset yang akan digunakan sebagai input dalam pemodelan aplikasi dari data mentah [16]. Proses pembersihan data dilakukan agar data siap diolah pada tahap implementasi. Langkah-langkah yang dilakukan meliputi:

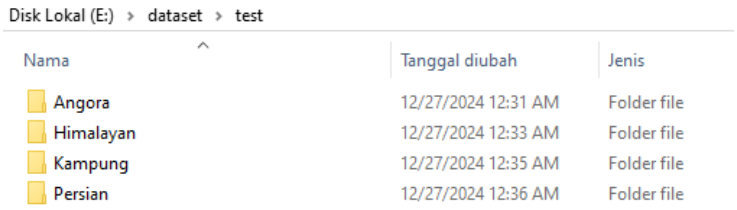
p-ISSN: 2621-9433
e-ISSN: 2656-0259

1. Membagi Data
   Membagi Data ke dalam tiga kategori, yaitu Data *Train,* Data *Test,* dan Data *Validation.* Dilihat pada Gambar 3.1.



Gambar 7. Membagi Data Ke Dalam Tiga Kategori

2. Mengelompokkan Jenis Kucing
   Mengelompokkan Jenis Kucing menjadi empat ras, yaitu kucing Anggota, kucing Persia, kucing Himalayan, dan kucing Kampung. Dilihat pada Gambar 3.2.



Gambar 8. Mengelompokan Kucing Menjadi Empat Ras

3. Data Pengujian
   Data Yang Dipakai 2.703 data yang digunakan untuk pengujian.
4. Melakukan *Resize Image*
   Proses Resize Image dilakukan dengan menyesuaikan dimensinya menjadi 224 × 224 piksel [17]. Diubah secara otomatis menggunakan *Source Code* yang diuji dengan *Google Collaboratory*. Dengan begitu, seluruh gambar dalam dataset akan memiliki ukuran dimensi piksel yang sama, sehingga mempermudah tahap pemrosesan data dan analisis berikutnya.
5. Melakukan Normalisasi Data
   Normalisasi data dilakukan dengan mengubah nilai X dan Y. Nilai X disesuaikan agar berada dalam rentang 0 hingga 1, sementara nilai Y dikonversi menjadi angka dari 0 hingga 7 sesuai dengan jumlah kelas yang tersedia [17].

Modelling

MobileNetV2 merupakan arsitektur yang dikembangkan untuk meningkatkan efisiensi dalam pemrosesan citra dengan memanfaatkan *inverted residual block* dan *bottleneck layer.* Komponen ini memungkinkan pengurangan jumlah perhitungan dibandingkan dengan versi sebelumnya, sehingga membuat model lebih ringan dan cepat dalam melakukan klasifikasi gambar [18]. Kemampuan ini membuat model sangat cocok digunakan pada perangkat dengan keterbatasan daya komputasi, seperti ponsel atau perangkat *IoT*. Dilihat pada Gambar 3.3.

p-ISSN: 2621-9433
e-ISSN: 2656-0259

Membangun Model *MobileNet - Version 2*

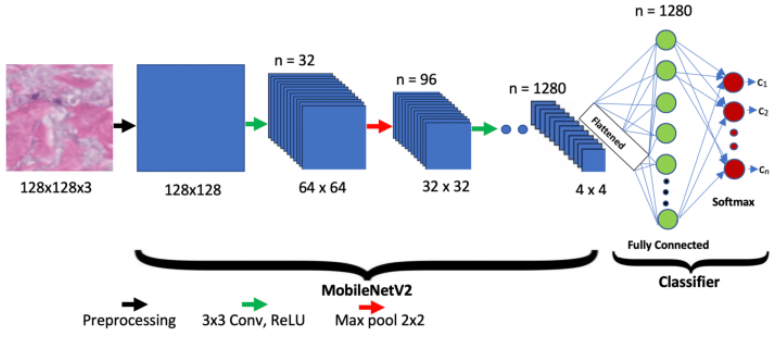
Dalam pengujian penelitian ini, model dikembangkan menggunakan arsitektur *MobileNetV2* di *Google Collaboratory*. Model ini digunakan sebagai *feature extractor* dalam sistem *deep learning* untuk klasifikasi gambar.

1. *input_shape* = (224, 224, 3) : Menentukan ukuran input gambar (224 × 224, 3 channel warna (RGB)).
2. *base_model* = *MobileNetV2* (include_top = False, weights = imagenet, input_shape = input_shape) :
   a. *MobileNetV2* : Memuat arsitektur model.
   b. *include_top* = *False* : Menghapus lapisan *fully connected* (FC) agar model berfungsi sebagai ekstraktor fitur.
   c. *weights* = *imagenet* : Menggunakan bobot pralatih dari ImageNet.
   d. *input_shape* = *input_shape* : Menyesuaikan ukuran input.
   e. *base_model.summary* () : Menampilkan detail arsitektur model.

Penyesuaian *Layer Convolutional Neural Networks*

Setelah membangun model menggunakan arsitektur *MobileNetV2*, langkah selanjutnya adalah menyesuaikan layer *Convolutional Neural Networks (CNN)* untuk meningkatkan performa dalam klasifikasi gambar. Kode berikut digunakan untuk membangun model yang telah dimodifikasi dengan penyesuaian layer agar lebih optimal.

1. *Input Layer* : Menerima gambar berwarna (224 × 224, RGB).
2. *Base Model* : MobileNetV2 digunakan sebagai ekstraktor fitur dengan bobot dibekukan (training = False).
3. *Flatten* : Mengubah feature map 2D menjadi vektor 1D untuk proses klasifikasi.
4. *Dense Layer* : Dense (256, activation = relu) Belajar pola dari fitur yang diekstrak.
5. *kernel_regularizer* = tf.keras.regularizers.l2 (0.001) Mencegah overfitting dengan L2 regularization.
6. *Dropout* : Dropout (0.5) Menonaktifkan 50% neuron secara acak untuk meningkatkan generalisasi.
7. *Output Layer* : Dense (4, activation = softmax) Menghasilkan probabilitas dari 4 kelas.
8. Model Akhir : Menggabungkan seluruh layer dan menampilkan arsitektur dengan summary ().



Gambar 9. Layer MobileNet-V2

Melatih Model *MobileNet - Version 2*

Setelah melakukan *input layer* pada *Convolutional Neural Network (CNN)* sesuai kebutuhan, langkah berikutnya adalah melatih model *MobileNetV2* yang telah dibangun sebelumnya. Proses pelatihan ini menggunakan berbagai teknik optimasi untuk meningkatkan akurasi dan menekan nilai *loss*.

1. Mengompilasi Model
   Model dikompilasi menggunakan *loss categorical_crossentropy* untuk klasifikasi multi-kelas, dengan *optimizer* Adam (learning rate 0.0001) dan metrik evaluasi akurasi.
2. *Callbacks* (Penghentian Dini dan Penyesuaian Learning Rate)
   EarlyStopping menghentikan pelatihan jika val_loss tidak membaik selama 20 epoch berturut-turut, sementara *ReduceLROnPlateau* menurunkan *learning rate* jika val_loss stagnan selama 10 epoch.
3. Pelatihan Model
4. Model dilatih menggunakan *data augmentation* dengan *datagen.flow*(), hanya dalam **1 epoch** sehingga belajar sekali dari seluruh dataset. Validasi dilakukan dengan (x_test, y_test), serta menggunakan *callbacks early_stop* dan *reduce_lr.*

Evaluation

Tahap ini bertujuan untuk mengevaluasi hasil dari tahap sebelumnya, yaitu *Modelling*. Evaluasi dilakukan untuk menyesuaikan model agar lebih optimal dan sesuai dengan target yang ingin dicapai [19]. Pengujian dilakukan menggunakan *optimizer Adam, loss CategoricalCrossentropy,* dan *metrics Accuracy* pada epoch 10–100 untuk menentukan model terbaik berdasarkan akurasi *validasi dan loss*. Epoch 70 mencapai akurasi training 99,98%, validasi 91,15%, dengan loss training 37,27% dan loss validasi 94,53%. Epoch 100 menunjukkan akurasi training 99,93%, validasi 91,59%, serta *loss training* 38,27%dan *loss validasi* 93,03%.Epoch 90 terbukti paling optimal, dengan akurasi *training* 99,45%, validasi 93,81%,*loss training* 38,30%, dan loss validasi 71,08%. Evaluasi menggunakan confusion matrix,precision, recall, *f1-score*, dan *support* menunjukkan bahwa epoch 90 menghasilkan model paling stabil dengan akurasi tertinggi (93,81%) dan *overfitting* minimal dibandingkan epoch 70 dan 100.

Tabel 1. Hasil Pengujian Arsitektur MobileNet-V2

Epoch	Validasi Akurasi (%)	Class	Precision (%)	Recall (%)	F1-Score (%)	Support
90	93,81	0	90,48	96,61	93,44	59
		1	98,11	92,86	95,41	56
		2	91,07	92,73	91,89	55
		3	96,30	92,86	94,55	56
		Macro Avg	93,99	93,76	93,82	226
		Weighted Avg	93,96	93,81	93,83	226
70	91,15	0	88,02	100,00	89,39	59
		1	94,64	94,64	94,64	56
		2	94,12	87,27	90,57	55
		3	100,00	82,14	90,22	56
		Macro Avg	92,40	91,01	91,28	226

Epoch	Validasi Akurasi (%)	Class	Precision (%)	Recall (%)	F1-Score (%)	Support
		Weighted Avg	92,23	91,15	91,18	226
100	91,59	0	85,07	96,61	90,48	59
		1	96,43	96,43	96,43	56
		2	91,23	94,55	92,86	55
		3	97,83	80,36	88,24	56
		Macro Avg	92,64	91,99	92,08	226
		Weighted Avg	92,55	91,59	91,98	226

Tabel 2. Data Hasil Pengujian Model

Metrik	Epoch 70	Epoch 90	Epoch 100
Akurasi Training	99,98%	99,45%	99,93%
Loss Training	37,27%	38,30%	38,27%
Akurasi Validasi	91,15%	93,81%	91,59%
Loss Validasi	94,53%	71,08%	93,03%

Analisis Perbandingan

Analisis perbandingan dilakukan pada epoch 70, 90, dan 100 berdasarkan akurasi validasi, loss validasi, dan rata-rata akurasi. Epoch 90 terbukti paling optimal dengan akurasi validasi tertinggi (93,81%) dan loss validasi terendah (0,7108), menunjukkan kestabilan model serta minimnya *overfitting*.

Tabel 3. Analisis Perbandingan

Epoch	Perbedaan (Loss Validasi - Loss Training)	Persentase Perbedaan
70	94,53−37,27= 57,26	$\frac{37,27}{57,26} x100\%$ = 153,71%
100	93,03−38,27 = 54,76	$\frac{38,27}{54,76} x100\%$ = 143,14%
90	71,08−38,30 = 32,78	$\frac{38,30}{32,78} x100\%$ = 85,61%

Menghitung Persentase *Loss Training* dan *Loss* Validasi

Tahap ini menghitung perbedaan antara *Loss Training* dan *Loss* Validasi setiap epoch menunjukkan bahwa epoch 70 dan 100 memiliki selisih *loss* yang besar, masing-masing 153,71% dan 143,14%, mengindikasikan potensi *overfitting*. Sebaliknya, epoch 90 memiliki perbedaan terkecil (85,61%), menandakan model yang lebih stabil dan seimbang. Perhitungan ini dilakukan menggunakan rumus berikut:

$$\textbf{Persentase Perbedaan} = \frac{Loss\ Validasi - Loss\ Training}{Loss\ Training} X100\%$$

Tabel 4. Menghitung Persentase Loss Training dan Loss Validasi

Epoch	Akurasi Validasi	Loss Validasi	Rata-rata Akurasi
70	91,15%	94,53%	91,15%
100	91,59%	93,03%	91,59%
90	93,81%	71,08%	93,81%

p-ISSN: 2621-9433
e-ISSN: 2656-0259

	Perbedaan (Akurasi Training - Akurasi Validasi)	Persentase Perbedaan
70	99,98%− 91,15%= 8,83%	$\frac{8,83}{99,98} x100\% = 8,83\%$
100	99,93%− 91,59%= 8,34%	$\frac{8,34}{99,93} x100\% = 8,34\%$
90	99,45%− 93,81% = 5,64%	$\frac{5,64\%}{99,45} x100\% = 5,64\%$

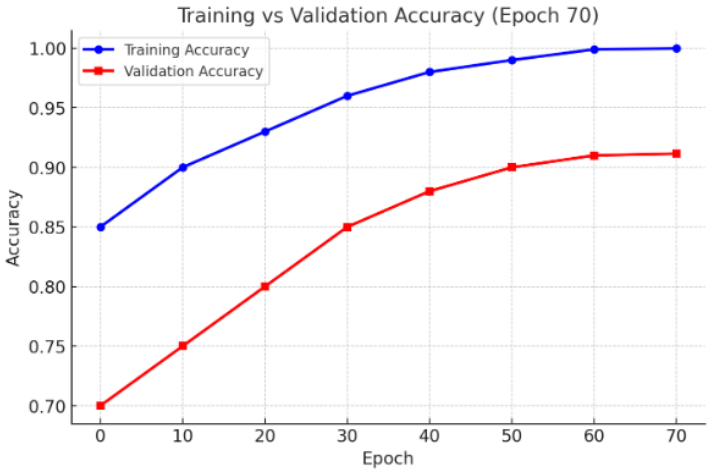
Menghitung Persentase Akurasi *Training* dan Akurasi Validasi

Tahap selanjutnya menghitung persentase perbedaan antara Akurasi *Training* dan Akurasi Validasi setiap epoch menggunakan rumus :

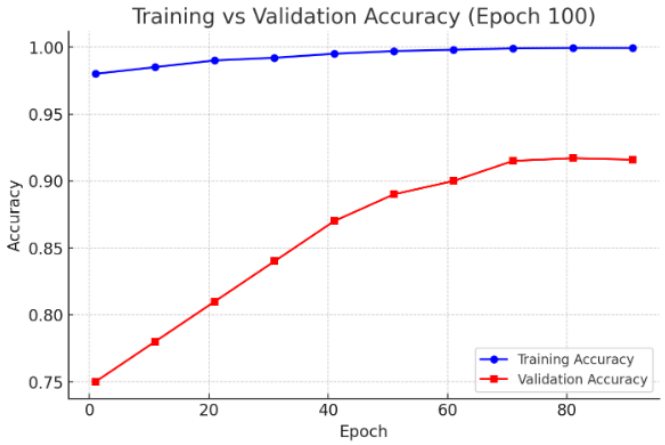
$$\text{Persentase Perbedaan} = \left(\frac{\text{Akurasi Training} - \text{Akurasi Validasi}}{\text{Akurasi Training}}\right) x100$$

Grafik *Accuracy Training* Dan Validasi

Akurasi adalah metrik yang mengukur persentase prediksi yang benar dari total prediksi yang dilakukan [20]. Pengujian pada epoch 70 dapat dilihat pada Gambar 10, sementara epoch 100 ditampilkan pada Gambar 11, dan epoch 90 pada Gambar 12.

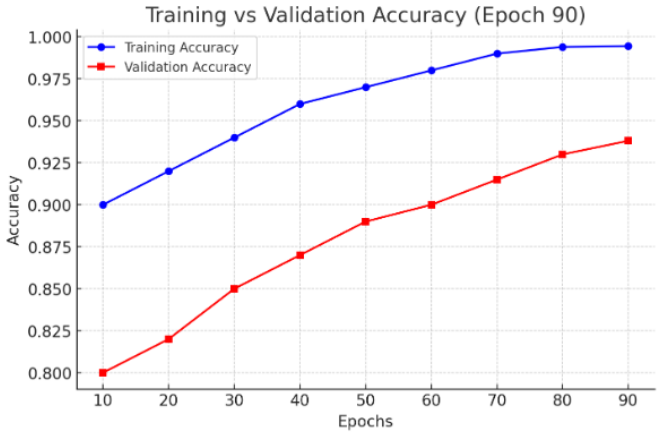


Gambar 10. Grafik Accuracy Epoch 70



Gambar 11. Grafik *Accuracy Epoch* 100

p-ISSN: 2621-9433
e-ISSN: 2656-0259

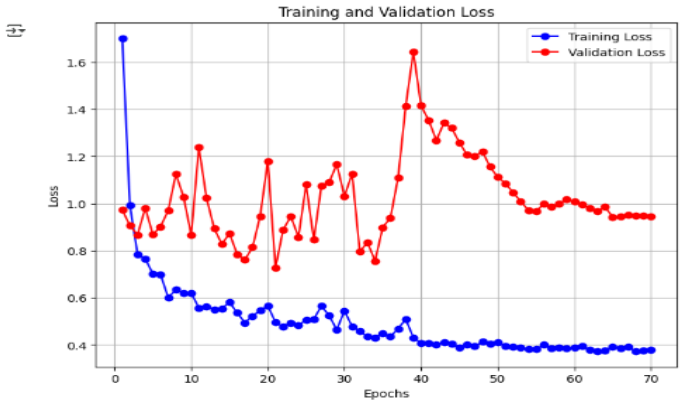


Gambar 12. Grafik *Accuracy Epoch* 90

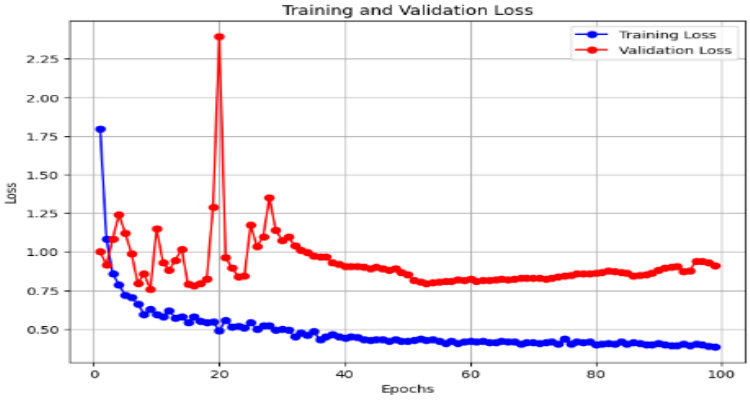
Grafik ini membandingkan akurasi training (biru) dan validasi (merah). Pada epoch 70, akurasi training mencapai 99,98% dan validasi 91,15%. Epoch 90 menunjukkan akurasi training 99,45% dan validasi 93,81%,sementara epoch 100 mencatat training 99,93% dan validasi 91,59%.

Grafik *Loss Training* dan Validasi

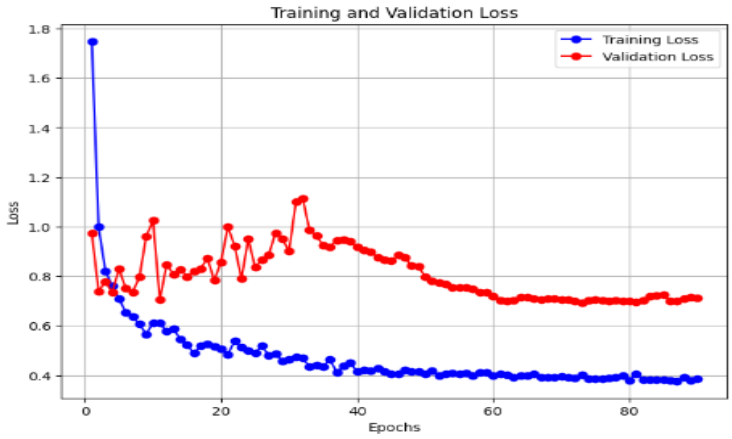
Loss adalah metrik yang digunakan untuk menilai sejauh mana perbedaan antara hasil prediksi model dan nilai sebenarnya [20]. Pengujian pada epoch 70 dapat dilihat pada Gambar 13, sementara epoch 100 ditampilkan pada Gambar 14, dan epoch 90 pada Gambar 15.



Gambar 13. Grafik *Loss Epoch* 70



Gambar 14. Grafik *Loss Epoch* 100



Gambar 15. Grafik *Loss Epoch* 90

Grafik membandingkan training loss (biru) dan validation loss (merah) pada berbagai epoch. Epoch 70 dan 100 menunjukkan penurunan *training loss*, namun validation loss fluktuatif, menandakan *overfitting*. Sebaliknya, epoch 90 memiliki perbedaan loss lebih kecil dan stabil, menghasilkan model yang lebih seimbang dan minim *overfitting*.

Deployment

Pada tahap ini, menyajikan hasil dari model yang telah diterapkan dalam proses data mining, termasuk penyampaian informasi yang diperoleh selama analisis agar mudah dipahami oleh pengguna [12]. Tahap ini mencakup pembuatan laporan setelah mengevaluasi penelitian terkait model klasifikasi Ras kucing.

PENUTUP

Penelitian ini membuktikan bahwa *CNN* dengan *MobileNet-V2* mampu mengklasifikasikan ras kucing dengan akurasi tinggi. Epoch 90 memberikan performa optimal dengan keseimbangan terbaik antara akurasi pelatihan dan validasi. Namun, tantangan seperti overfitting dan generalisasi masih ada. Pengembangan selanjutnya dapat mencakup regulasi tambahan seperti dropout, data augmentation, atau model yang lebih kompleks. Pengujian dengan dataset lebih besar juga diperlukan untuk memastikan kinerja model dalam berbagai kondisi nyata, sehingga lebih efektif dalam sistem pengenalan gambar berbasis *AI*.

DAFTAR PUSTAKA

[1] R. Gunawan, D. M. I. Hanafie, and A. Elanda, "Klasifikasi Jenis Ras Kucing Dengan Gambar Menggunakan Convolutional Neural Network (CNN)," *Jurnal Interkom: Jurnal Publikasi Ilmiah Bidang Teknologi Informasi dan Komunikasi*, vol. 18, no. 4, pp. 1–8, Jan. 2024, doi: 10.35969/interkom.v18i4.318.

[2] A. Nur Ramadhayani, V. Lusiana, U. Stikubank Semarang JlTri Lomba Juang No, K. Semarang Selatan, K. Semarang, and J. Tengah, "Klasifikasi Jenis Kucing Menggunakan Algoritma Principal Component Analysis dan K-Nearest Neighbor *Jurnal informasi dan Komputer*, vol. 10, no. 2, 2022.

[3] C. Agusniar and D. Adelia, "Penerapan Convolutional Neural Network pada Klasifikasi Jenis Ras Kucing Menggunakan ResNet50V2," 2024.
[4] J. Tugas, A. Fakultas Informatika, M. Afif, A. Fawwaz, K. N. Ramadhani, and F. Sthevanie, "Klasifikasi Ras pada Kucing menggunakan Algoritma Convolutional Neural Network(CNN)," 2020.
[5] S. Diantama, "Pemanfaatan Artificial Intelegent (AI) dalam Dunia Pendidikan," 2023.
[6] P. Annisa, "Penerapan Teknologi Google Lens dan QR Code pada Tanaman Pertanian," *dst*, vol. 3, no. 2, pp. 240–245, Nov. 2023, doi: 10.47709/dst.v3i2.3130.
[7] A. S. Riyadi, I. P. Wardhani, D. S. Widayati, and K. Kunci, "Klasifikasi Citra Anjing dan Kucing Menggunakan Metode Convolutional Neural Network (CNN)," *Universitas Gunadarma Jl. Margonda Raya No*, vol. 5, no. 1, p. 12140, 2021.
[8] A. Aziz, R. Reyhan Zhafari, and M. M. Santoni, *Klasifikasi 10 Spesies Monyet Berdasarkan Citra Menggunakan Convolutional Neural Network*. 2021.
[9] D. Ramayanti, D. Asri, and L. Lionie, "Implementasi Model Arsitektur VGG16 dan MobileNetV2 Untuk Klasifikasi Citra Kupu-Kupu Article Info ABSTRAK," *JSAI: Journal Scientific and Applied Informatics*, vol. 5, no. 3, 2022, doi: 10.36085.
[10] O. Virgantara Putra, M. Zaim Mustaqim, D. Muriatmoko, J. Teknik Informatika, and F. Sains dan Teknologi, "Transfer Learning untuk Klasifikasi Penyakit dan Hama Padi Menggunakan MobileNetV2 Transfer Learning for Rice Disease and Pest Classification using MobileNetV2," 2023.
[11] M. Riziq sirfatullah Alfarizi, M. Zidan Al-farish, M. Taufiqurrahman, G. Ardiansah, and M. Elgar, "Penggunaan Python sebagai Bahasa Pemograman untuk Machine Learning dan Deep Learning," 2023.
[12] S. N. Luqman *et al.*, "Komparasi Algoritma Klasifikasi Genre Musik pada Spotify Menggunakan CRISP-DM," 2021.
[13] I. Budiman, T. Prahasto, and Y. Christyono, "Data Clustering Menggunakan Metodologi CRISP-DM untuk Pengenalan Pola Proporsi Pelaksanaan Tridharma," 2012.
[14] A. Abhista Hibatullah and W. Apriandari, "Klasifikasi Kualitas Jenis Kopi Halus Robusta Menggunakan Convolutional Neural Network (CNN) dan Mobilenet-V2," 2024.
[15] M. A. Hasanah, S. Soim, and A. S. Handayani, "Implementasi CRISP-DM Model Menggunakan Metode Decision Tree dengan Algoritma CART untuk Prediksi Curah Hujan Berpotensi Banjir," 2021. [Online]. Available: http://jurnal.polibatam.ac.id/index.php/JAIC
[16] N. Alfiah, "Klasifikasi Penerima Bantuan Sosial Program Keluarga Harapan Menggunakan Metode Naive Bayes".
[17] Y. P. Astuti, E. R. Subhiyakto, I. Wardatunizza, and E. Kartikadarma, "Implementasi Algoritma Convolutional Neural Network (CNN) Untuk Klasifikasi Jenis Tanah Berbasis Android," *Jurnal Informatika: Jurnal*

Pengembangan IT, vol. 8, no. 3, pp. 220–225, Sep. 2023, doi: 10.30591/jpit.v8i3.5026.

[18] W. Hastomo, E. Hadiyanto, and D. Sutarno, "Klasifikasi Covid-19 CHEST X-RAY denagn Tiga Arsitektur CNN (Resnet-152, Inceptionresnet -V2, Mobilenet-V2)," 2021.

[19] M. Y. Titimeidara and W. Hadikurniawati, "Monica Yoshe Titimeidara Implementasi Metode Naive Bayes Implementasi Metode Naive Bayes Classifier Untuk Klasifikasi Status Gizi Stunting Pada Balita."

[20] Z. I. Nugraha, Arnita, Kana Saputra S, A. Setiawan, R. Maharani, and F. Zaharani, "Impelementasi Algoritma CNN dalam Pengembangan website untuk Klasifikasi Sampah Organik dan Non-Organik," *Jurnal Manajemen Informatika dan Sistem Informasi*, vol. 8, no. 1, pp. 90–101, Jan. 2025, doi: 10.36595/misi.v8i1.1355.