Perbandingan Arsitektur Convolutional Neural Network, ResNet50, dan EfficientNetB0 untuk Klasifikasi Gambar Hewan

Galang Dwiwana Thabrani[1], Irma Dwiyanti[1], Irma Rohmatillah[1]
[1]Department of Informatics Engineering, UIN Sunan Gunung Djati Bandung

Article Info

Article history:

Received month dd, 2025
Revised month dd, yyyy
Accepted month dd, yyyy

Keywords:

Klasifikasi Gambar
CNN
ResNet50
EfficientNetB0
Transfer Learning

ABSTRACT

Penelitian ini bertujuan untuk membandingkan performa tiga arsitektur CNN, yaitu Custom CNN, ResNet50, dan EfficientNetB0, dalam klasifikasi gambar tiga jenis hewan: kucing, anjing, dan ular. Dataset yang digunakan berasal dari Kaggle dan terdiri dari 3.000 gambar JPG beresolusi 256x256 piksel dalam tiga kelas seimbang. Data dibagi ke dalam subset pelatihan (70%), validasi (15%), dan pengujian (15%). Model Custom CNN dibangun dari awal, sementara ResNet50 dan EfficientNetB0 menggunakan pendekatan transfer learning. Proses pelatihan dilakukan selama maksimal 25 epoch, menggunakan data augmentasi pada subset pelatihan dan evaluasi dilakukan berdasarkan akurasi uji, jumlah epoch, dan waktu pelatihan. Hasil penelitian menunjukkan bahwa Custom CNN memberikan akurasi tertinggi sebesar 74%, diikuti oleh ResNet50 sebesar 65,33%, dan EfficientNetB0 sebesar 41,78%. Custom CNN juga menunjukkan efisiensi waktu pelatihan yang baik. Temuan ini menunjukkan bahwa model sederhana yang dirancang khusus dapat mengungguli model pretrained dalam kondisi dataset terbatas. Oleh karena itu, pemilihan arsitektur sebaiknya disesuaikan dengan karakteristik data dan tujuan aplikatif.

Corresponding Author:

Galang Dwiwana Thabrani
Department of Informatics Engineering, UIN Sunan Gunung Djati Bandung
Email: galangdwi@gmail.com

1. PENDAHULUAN

Kemampuan komputer dalam mengenali dan mengklasifikasikan gambar secara otomatis menjadi pondasi penting dalam berbagai aplikasi teknologi, seperti pengawasan keamanan, sistem pertanian presisi, dan konservasi satwa liar. Salah satu pendekatan utama dalam bidang ini adalah penggunaan Convolutional Neural Network (CNN), yang telah terbukti mampu mengenali pola visual kompleks dengan presisi tinggi [1]. Akan tetapi, kompleksitas morfologi dan kemiripan visual antar spesies hewan menjadi tantangan tersendiri dalam tugas klasifikasi, sehingga dibutuhkan arsitektur model yang tidak hanya akurat, tetapi juga efisien dan mampu menangani variasi data secara optimal.

Salah satu arsitektur lanjutan dari CNN adalah ResNet50, yang memanfaatkan konsep *residual learning* untuk mengatasi masalah vanishing gradient pada jaringan dalam. Beberapa penelitian membuktikan bahwa ResNet50 unggul dibandingkan CNN konvensional, baik dari sisi akurasi maupun kestabilan pelatihan, khususnya pada dataset besar dan kompleks [2], [3], [4]. Penggunaan ResNet50 dalam berbagai domain seperti klasifikasi daun, kanker kulit, dan analisis medis lainnya berhasil mencapai akurasi di atas 95% [5], [6].

Selain ResNet50, model EfficientNetB0 juga menarik perhatian peneliti karena kemampuannya mengoptimalkan akurasi dan efisiensi komputasi melalui *compound scaling*. Dalam studi klasifikasi burung, EfficientNetB0 mencapai akurasi lebih tinggi daripada MobileNetV2 [7]. Pada klasifikasi retina dan deteksi

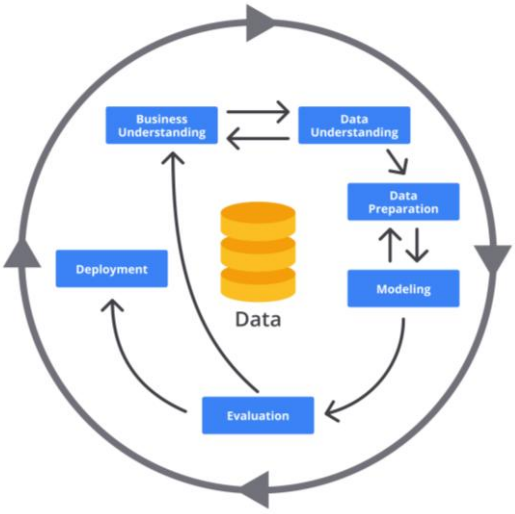
deepfake, EfficientNetB0 terbukti unggul dengan akurasi hingga 91% [8], [9]. Model ini juga digunakan sebagai fitur extractor dalam sistem captioning gambar dan menunjukkan stabilitas performa yang baik [10].

Namun, meskipun kedua arsitektur tersebut menjanjikan, masih ada keterbatasan dan gap dalam literatur. ResNet50 diketahui cenderung overfitting pada dataset kecil tanpa augmentasi atau regularisasi yang memadai [11], sementara EfficientNetB0 tidak selalu unggul dalam klasifikasi kompleks seperti penyakit kanker atau klasifikasi satwa liar di kondisi alami [12], [13]. Bahkan dalam beberapa kasus, model sederhana seperti CNN custom justru menunjukkan performa yang kompetitif jika dikombinasikan dengan augmentasi dan tuning yang tepat [14].

Penelitian ini bertujuan untuk membandingkan secara langsung tiga arsitektur CNN yaitu CNN konvensional, ResNet50, dan EfficientNetB0 dalam konteks klasifikasi gambar hewan. Penelitian ini dilakukan secara sistematis melalui pelatihan dan evaluasi model pada dataset yang sama, menggunakan metrik standar seperti akurasi, presisi, recall, dan F1-score. Kontribusi utama dari studi ini adalah memberikan analisis empiris tentang performa relatif ketiga model tersebut.

2. METODE

Pada bagian ini dijelaskan bagaimana metode yang digunakan untuk melakukan penelitian dalam Membandingan Arsitektur Convolutional Neural Network, ResNet50, dan EfficientNetB0 untuk Klasifikasi Gambar Hewan. Metode yang digunakan untuk penelitian ini menggunakan metode bernama CRIPS-DM dengan beberapa tahapan yang ada sebagai berikut:



Gambar 1. Metode CRISP-DM

(https://www.dicoding.com/blog/crisp-dm-tahapan-studi-kasus-kelebihan-dan-kekurangan/)

Penelitian ini menggunakan pendekatan CRISP-DM (Cross Industry Standard Process for Data Mining) sebagai kerangka kerja untuk proses klasifikasi gambar hewan. Tahapan CRISP-DM yang diterapkan dalam penelitian ini dijelaskan sebagai berikut:

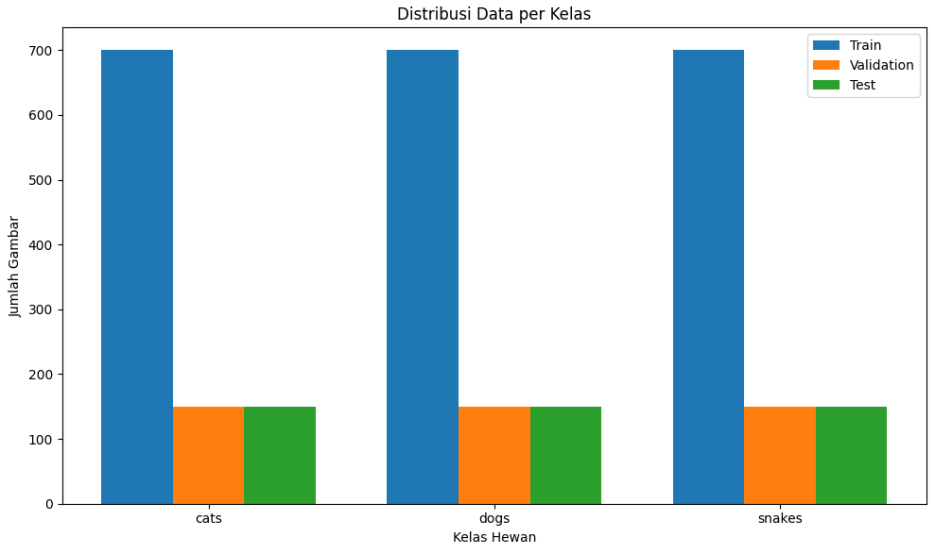
1) Business Understanding

Tujuan utama dari penelitian ini adalah untuk mengevaluasi dan membandingkan performa tiga arsitektur deep learning, yaitu CNN, ResNet50, dan EfficientNetB0, dalam mengklasifikasikan gambar hewan ke dalam beberapa kategori. Penelitian ini ingin mengetahui arsitektur mana yang paling efektif dan efisien untuk digunakan pada tugas klasifikasi citra, khususnya dalam konteks klasifikasi hewan, berdasarkan metrik evaluasi seperti akurasi, presisi, recall, dan F1-score.

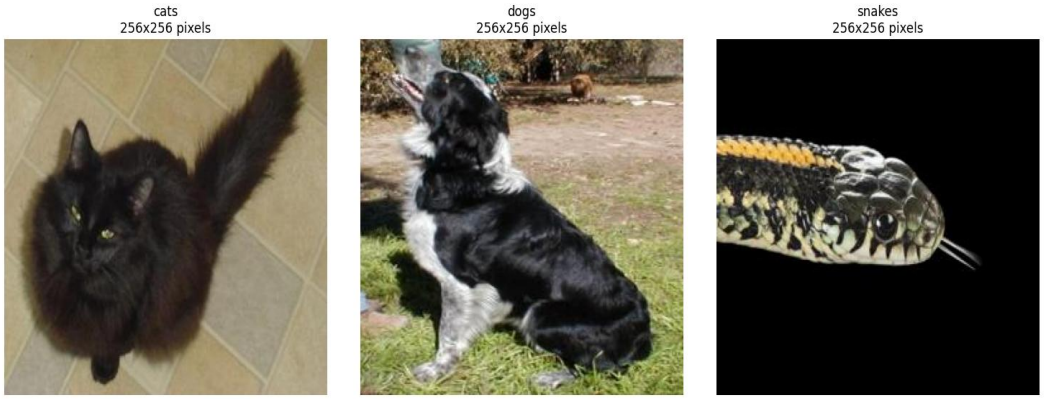
2) Data Understanding

Dataset yang digunakan dalam penelitian ini berasal dari situs Kaggle (Dataset) berisi 3.000 gambar JPG beresolusi 256x256 piksel dalam format RGB. Dataset terbagi secara seimbang ke dalam tiga kelas: kucing, anjing, dan ular, masing-masing berjumlah 1.000 gambar yang mewakili beragam spesies, postur, dan lingkungan. Dataset diunduh dan diidentifikasi secara otomatis menggunakan skrip Python untuk menemukan folder bernama "Animals" dan mengakses subfolder kelas.

Setelah ditemukan, dataset dibagi menjadi tiga subset: 70% untuk pelatihan, 15% validasi, dan 15% pengujian, menggunakan metode pembagian acak terkontrol. Data kemudian disusun ulang ke dalam struktur direktori yang sesuai untuk pelatihan model. Distribusi jumlah gambar per kelas divisualisasikan untuk memastikan keseimbangan antar subset, dan contoh gambar ditampilkan untuk meninjau kualitas visual. Tahapan ini memastikan bahwa data telah dipahami secara menyeluruh sebelum proses pemodelan dilakukan.



Gambar 2. Grafik Distribusi Data PerKelas



Gambar 3. Contoh gambar dataset

3) Data Preparation

Pada tahap ini, data yang telah dibagi ke dalam folder pelatihan, validasi, dan pengujian diproses menggunakan ImageDataGenerator untuk menyiapkan input yang sesuai bagi model CNN. Seluruh gambar diubah ukurannya menjadi 224x224 piksel, dan diproses dalam batch berukuran 32 gambar per iterasi.

Untuk subset pelatihan, diterapkan augmentasi data guna meningkatkan kemampuan generalisasi model terhadap variasi citra. Teknik augmentasi yang digunakan meliputi rotasi acak hingga 20 derajat, pergeseran horizontal dan vertikal hingga 15%, *shear*, zoom, dan *horizontal flip*. Semua gambar juga dinormalisasi dengan membagi nilai pikselnya dengan 255 agar berada dalam rentang [0, 1].

Sementara itu, subset validasi dan pengujian hanya diproses dengan **rescaling**, tanpa augmentasi, untuk menjaga kemurnian data uji dan validasi. Proses ini dilakukan melalui generator

flow_from_directory() yang secara otomatis membaca gambar dari folder, memberi label sesuai nama subdirektori kelas (kucing, anjing, ular), dan menghasilkan array input serta target dalam format *categorical*.

```
img_size = (224, 224)
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.15,
    height_shift_range=0.15,
    shear_range=0.15,
    zoom_range=0.15,
    horizontal_flip=True,
    fill_mode='nearest'
)

val_test_datagen = ImageDataGenerator(rescale=1./255)
```

Proses pemuatan data dilakukan menggunakan fungsi flow_from_directory() dari ImageDataGenerator, yang membaca gambar langsung dari direktori berdasarkan struktur folder sebagai label kelas. Generator ini menghasilkan batch data citra berukuran 224x224 piksel dengan batch size 32 dan format label *categorical*. Untuk data pelatihan, gambar diacak menggunakan shuffle=True, sementara data validasi dan pengujian tidak diacak (shuffle=False) untuk menjaga konsistensi urutan saat evaluasi. Pendekatan ini memastikan pemrosesan data yang efisien dan terstruktur untuk setiap tahap pelatihan dan pengujian model.

```
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=img_size,
    batch_size=32,
    class_mode='categorical',
    shuffle=True
)

val_generator = val_test_datagen.flow_from_directory(
    val_dir,
    target_size=img_size,
    batch_size=32,
    class_mode='categorical',
    shuffle=False
)

test_generator = val_test_datagen.flow_from_directory(
    test_dir,
    target_size=img_size,
    batch_size=32,
    class_mode='categorical',
    shuffle=False
)
```

4) Modelling

Pada tahap pemodelan, tiga arsitektur CNN dikembangkan dan dibandingkan, yaitu Custom CNN, ResNet50, dan EfficientNetB0. Model Custom CNN dibangun dari awal menggunakan beberapa lapisan konvolusi yang semakin dalam, masing-masing diikuti oleh *max pooling* dan dua *dense layer* dengan dropout untuk mencegah overfitting. Lapisan akhir menggunakan *softmax* untuk klasifikasi multi-kelas.

```
def create_cnn_model():
    model = Sequential([
        Conv2D(32, (3,3), activation='relu', padding='same', input_shape=(224,224,3)),
        MaxPooling2D(2,2),
        Conv2D(64, (3,3), activation='relu', padding='same'),
        MaxPooling2D(2,2),
        Conv2D(128, (3,3), activation='relu', padding='same'),
        MaxPooling2D(2,2),
        Conv2D(256, (3,3), activation='relu', padding='same'),
        MaxPooling2D(2,2),
        Flatten(),
        Dense(512, activation='relu'),
        Dropout(0.5),
        Dense(256, activation='relu'),
        Dropout(0.3),
        Dense(num_classes, activation='softmax')
    ])
    model.compile(optimizer=Adam(learning_rate=0.001),
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])
    return model
```

Sementara itu, model ResNet50 dan EfficientNetB0 dibangun dengan pendekatan transfer learning menggunakan bobot pretrained dari ImageNet. Lapisan konvolusional dasar dari kedua model dibekukan untuk mempertahankan fitur umum, dan ditambahkan lapisan pooling serta *dense* agar dapat disesuaikan dengan tugas klasifikasi hewan.

```
def create_resnet_model():
    base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(224,224,3))
    base_model.trainable = False
    model = Sequential([
        base_model,
        tf.keras.layers.GlobalAveragePooling2D(),
        Dense(256, activation='relu'),
        Dropout(0.4),
        Dense(128, activation='relu'),
        Dropout(0.3),
        Dense(num_classes, activation='softmax')
    ])
    model.compile(optimizer=Adam(learning_rate=0.0001),
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])
    return model

def create_efficientnet_model():
    base_model = EfficientNetB0(weights='imagenet', include_top=False,
input_shape=(224,224,3))
    base_model.trainable = False
    model = Sequential([
        base_model,
        tf.keras.layers.GlobalAveragePooling2D(),
        Dense(256, activation='relu'),
        Dropout(0.3),
        Dense(128, activation='relu'),
        Dropout(0.2),
```

```
    Dense(num_classes, activation='softmax')
  ])
  model.compile(optimizer=Adam(learning_rate=0.0001),
                loss='categorical_crossentropy',
                metrics=['accuracy'])
  return model
```

Setelah ketiga model diinisialisasi, proses pelatihan dilakukan dengan maksimum 25 epoch dan menggunakan callback EarlyStopping serta ReduceLROnPlateau untuk menghindari overfitting dan menyesuaikan learning rate secara dinamis.

```
models = {
  "Custom CNN": create_cnn_model(),
  "ResNet50": create_resnet_model(),
  "EfficientNetB0": create_efficientnet_model()
}

callbacks = [
  EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True, verbose=1),
  ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, min_lr=1e-7, verbose=1)
]
```

Setiap model kemudian dilatih menggunakan data yang telah disiapkan dengan train_generator dan val_generator. Waktu pelatihan serta jumlah epoch aktif dicatat untuk masing-masing model sebagai bagian dari evaluasi performa.

```
histories = {}
training_times = {}
epoch_counts = {}

for name, model in models.items():
  print(f"\n{'='*50}")
  print(f"Training model {name}")
  print(f"{'='*50}")

  model.summary()

  start_time = time.time()
  epochs = 25

  history = model.fit(
    train_generator,
    epochs=epochs,
    validation_data=val_generator,
    callbacks=callbacks,
    verbose=1
  )

  training_time = time.time() - start_time
  training_times[name] = training_time
  histories[name] = history
  epoch_counts[name] = len(history.history['loss'])

  print(f"\nTraining {name} selesai!")
  print(f"Waktu training: {training_time/60:.2f} menit")
```

```
print(f"Jumlah epoch: {epoch_counts[name]}")
```

5) Evaluation

Setelah pelatihan selesai, ketiga model dievaluasi untuk mengukur performa pada data uji. Evaluasi dilakukan melalui tiga aspek utama: visualisasi performa pelatihan, pengujian terhadap data uji, serta analisis klasifikasi menggunakan confusion matrix dan classification report. Fungsi evaluate_model() dirancang untuk menampilkan kurva akurasi dan loss selama pelatihan, menghitung akurasi pada data uji, dan menyajikan ringkasan klasifikasi yang lengkap.

```
def evaluate_model(name, model, history, test_generator):
    # Visualisasi akurasi dan loss per epoch
    # Evaluasi terhadap test set
    # Classification report dan confusion matrix
    ...
```

Dalam proses ini, setiap model diuji terhadap test_generator, dan akurasi serta loss dicatat. Hasil prediksi dikonversi menjadi kelas dan dibandingkan dengan label sebenarnya untuk menghasilkan classification report dan confusion matrix. Evaluasi dilakukan untuk semua model dan hasilnya disimpan dalam dictionary results.

```
results = {}
for name in models:
    test_acc = evaluate_model(name, models[name], histories[name], test_generator)
    results[name] = {
        'accuracy': test_acc,
        'training_time': training_times[name],
        'epochs': epoch_counts[name]
    }
```

Untuk memperjelas perbandingan performa antar model, dibuat dua visualisasi tambahan. Visualisasi pertama menunjukkan perbandingan akurasi akhir dari ketiga model dalam bentuk grafik batang.

```
# Grafik akurasi uji
plt.figure(figsize=(8, 6))
bars = plt.bar(model_names, acc_values, color=colors)
plt.title('Perbandingan Akurasi Model', fontsize=14)
...
plt.show()
```

Visualisasi kedua menampilkan perbandingan waktu pelatihan masing-masing model dalam satuan menit, untuk menunjukkan efisiensi komputasi.

```
# Grafik waktu pelatihan
plt.figure(figsize=(10, 6))
plt.bar(model_names, time_values, color=colors)
plt.title('Perbandingan Waktu Training', fontsize=14)
...
plt.show()
```

Dari hasil ini, pengguna dapat menilai trade-off antara akurasi dan efisiensi waktu dari masing-masing model. Proses evaluasi ini menjadi dasar dalam menentukan model terbaik untuk tugas klasifikasi gambar hewan berdasarkan kinerja aktual dan efisiensi pelatihan.

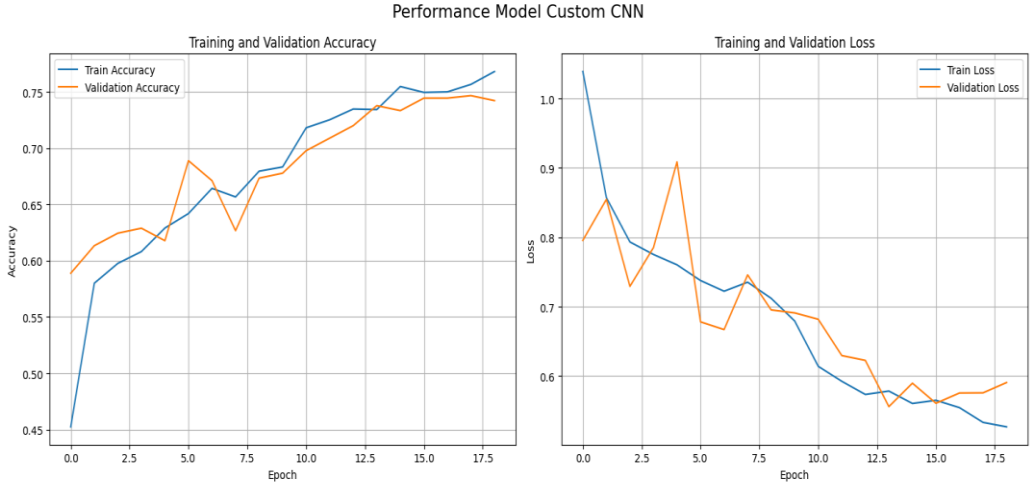
6) Deployment

Pada tahapan terakhir metode CRIPS-DM adalah deployment dimana perencanaan untuk deployment dimulai dari fase business understanding dan harus menggabungkan tidak hanya untuk menghasilkan nilai model, tetapi juga cara mengonversi skor keputusan dan penggabungan keputusan tersebut. Namun pada penelitian ini tahapan deployment tidak digunakan, karena penelitian ini berfokus hanya pada evaluasi perbandingan ketiga algoritma yang digunakan yaitu CNN (Custom), ResNet50, dan EfficientNetB0.

3. HASIL PENELITIAN

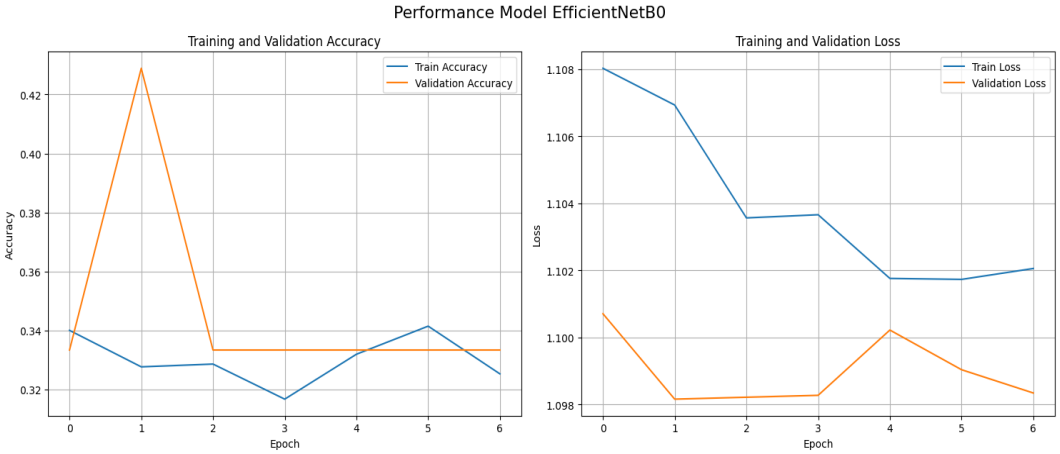
Penelitian ini mengevaluasi tiga arsitektur CNN, yaitu Custom CNN, ResNet50, dan EfficientNetB0, dalam tugas klasifikasi gambar hewan. Ketiga model dilatih menggunakan dataset yang sama dengan proporsi 70% untuk pelatihan, 15% untuk validasi, dan 15% untuk pengujian. Pelatihan dilakukan selama maksimal 25 epoch dengan penerapan *early stopping* dan *learning rate scheduling*. Setiap model diuji terhadap data uji menggunakan metrik akurasi, dan hasil pelatihan dikaji lebih lanjut melalui grafik kurva pelatihan serta evaluasi akhir.

Hasil pelatihan tiap model divisualisasikan melalui grafik akurasi dan loss terhadap epoch, yang memberikan gambaran proses konvergensi masing-masing model. Grafik ini menunjukkan bahwa Custom CNN mengalami peningkatan akurasi secara stabil tanpa overfitting yang signifikan, sementara ResNet50 dan EfficientNetB0 mengalami stagnasi lebih awal.



Gambar 4. Grafik Performance Model Custom CNN

Gambar 5. Grafik Performance Model ResNet50



Gambar 6. Grafik Performance Model EfficientNet80

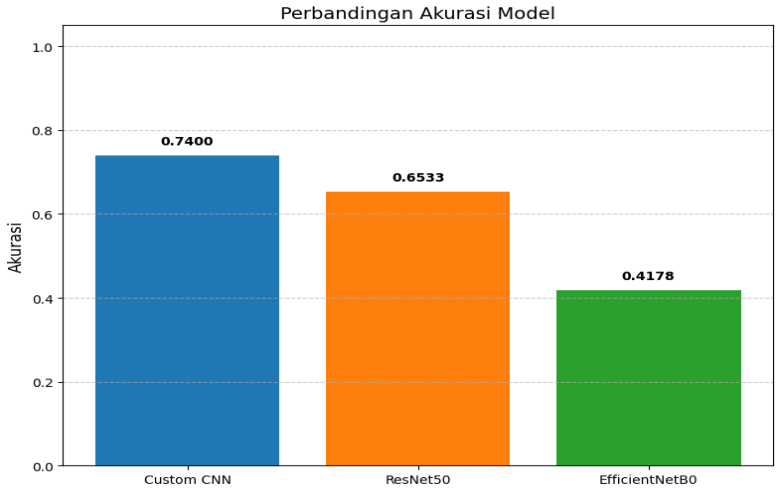
Setelah proses pelatihan, ketiga model dievaluasi menggunakan data uji. Tabel berikut merangkum hasil akhir akurasi uji, waktu pelatihan, dan jumlah epoch untuk masing-masing model:

Tabel 1. Hasil Training Model

Model	Akurasi Uji	Waktu Latih (menit)	Jumlah Epoch
Custom CNN	0.7400	8.88 min	19
ResNet50	0.6533	11.78 min	25
EfficientNetB0	0.4178	3.60 min	7

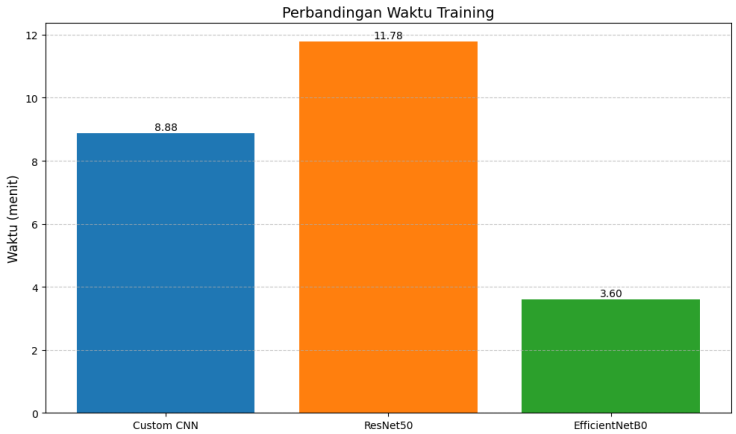
Custom CNN menunjukkan akurasi tertinggi sebesar 74%, menjadikannya model paling efektif dalam klasifikasi gambar hewan pada eksperimen ini. ResNet50 mencatat akurasi 65,33%, sedangkan EfficientNetB0 hanya mencapai 41,78%, meskipun memiliki waktu latihan paling cepat.

Untuk memperjelas perbandingan performa antar model, grafik akurasi uji disajikan sebagai berikut:



Gambar 7. Grafik Perbandingan Akurasi Model

Selain itu, perbandingan efisiensi pelatihan ditampilkan dalam grafik waktu pelatihan masing-masing model:



Gambar 8. Grafik Perbandingan Waktu Training Model

Model Custom CNN berhasil mengungguli model pretrained dalam eksperimen ini, yang menunjukkan bahwa arsitektur yang dirancang khusus dan dioptimalkan sesuai dataset dapat memberikan hasil lebih baik dibanding transfer learning, terutama dalam konteks data terbatas. Sementara ResNet50 dan EfficientNetB0 memerlukan penyesuaian lanjutan seperti fine-tuning lebih dalam dan penambahan epoch untuk mencapai hasil optimal.

4. CONCLUSION

Penelitian ini bertujuan untuk membandingkan performa tiga arsitektur CNN yaitu Custom CNN, ResNet50, dan EfficientNetB0 dalam tugas klasifikasi gambar hewan. Berdasarkan hasil eksperimen, model Custom CNN menunjukkan akurasi uji tertinggi sebesar 74%, dengan waktu pelatihan yang relatif singkat dan jumlah epoch yang efisien. Model ResNet50 mencatat akurasi 65,33% dengan waktu pelatihan paling lama, sedangkan EfficientNetB0, meskipun efisien secara komputasi, hanya mencapai akurasi 41,78% akibat jumlah epoch yang terlalu sedikit. Temuan ini menegaskan bahwa kompleksitas arsitektur tidak selalu menjamin performa terbaik, dan bahwa model sederhana dapat bekerja lebih efektif jika dikembangkan dan dikonfigurasi secara tepat sesuai dengan karakteristik data.

Penelitian selanjutnya disarankan untuk mengoptimalkan proses fine-tuning pada model pretrained, seperti menambah jumlah epoch dan menyesuaikan *learning rate*. Penggunaan dataset yang lebih besar serta teknik augmentasi lanjutan juga penting untuk meningkatkan generalisasi. Selain itu, eksplorasi model alternatif seperti DenseNet atau MobileNet, serta penggunaan matrik evaluasi tambahan seperti precision dan F1-score, dapat memberikan analisis yang lebih komprehensif. Terakhir, efisiensi komputasi perlu dipertimbangkan agar model dapat diterapkan secara praktis, terutama pada perangkat dengan sumber daya terbatas.

DAFTAR PUSTAKA

[1] L. M. K. Sheikh, A. Shaikh, A. Sandupatla, R. Pudale, A. Bakare, and Prof. M. Chavan, “Classification of Simple CNN Model and ResNet50,” *Int J Res Appl Sci Eng Technol*, vol. 12, no. 4, pp. 4606–4610, Apr. 2024, doi: 10.22214/ijraset.2024.60677.

[2] R. Bharti, V. Srivastava, A. Bajpai, and S. Sahu, “Comparative Analysis of Potato Leaf Disease Classification Using CNN and ResNet50,” in *2024 International Conference on Data Science and Its Applications (ICoDSA)*, 2024, pp. 87–91. doi: 10.1109/ICoDSA62899.2024.10651649.

[3] Y. Xu, “Image Classification Based on ResNet Models,” *Science and Technology of Engineering, Chemistry and Environmental Protection*, vol. 1, no. 10, Dec. 2024, doi: 10.61173/c5xnwg67.

[4] P. K. Das and S. S. Rupa, “ResNet for Leaf-based Disease Classification in Strawberry Plant,” *International Journal of Applied Methods in Electronics and Computers*, Sep. 2023, doi: 10.58190/ijamec.2023.42.

[5] C.-C. Peng and B.-R. Lee, “Enhancing Colorectal Cancer Histological Image Classification Using Transfer Learning and ResNet50 CNN Model,” in *2023 IEEE 5th Eurasia Conference on Biomedical*

Engineering, Healthcare and Sustainability (ECBIOS), IEEE, Jun. 2023, pp. 36–40. doi: 10.1109/ECBIOS57802.2023.10218590.

[6] N. Behar and M. Shrivastava, "ResNet50-Based Effective Model for Breast Cancer Classification Using Histopathology Images," *Computer Modeling in Engineering & Sciences*, vol. 130, no. 2, pp. 823–839, 2022, doi: 10.32604/cmes.2022.017030.

[7] A. C. Sitepu, C.-M. Liu, M. Sigiro, J. Panjaitan, and V. Copa, "convolutional neural network bird's classification using north American bird images," *Int J Health Sci (Qassim)*, pp. 15067–15080, Jun. 2022, doi: 10.53730/ijhs.v6nS2.8988.

[8] P. Rastogi, J. Jain, P. Jain, K. Arjun, and R. Aggarwal, "Analysis of CNN Models for Eye Disease Classification Using Retinal Images," in *2024 1st International Conference on Advanced Computing and Emerging Technologies (ACET)*, IEEE, Aug. 2024, pp. 1–6. doi: 10.1109/ACET61898.2024.10730305.

[9] R. N. Bharath Reddy, T. V Naga Siva, B. S. Ram, K. N. Ramya Sree, and B. Suvarna, "Enhanced Deep Fake Image Detection via Feature Fusion of EfficientNet, Xception, and ResNet Models," in *2025 6th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI)*, IEEE, Jan. 2025, pp. 1547–1552. doi: 10.1109/ICMCSI64620.2025.10883356.

[10] R. Mulyawan, A. Sunyoto, and A. H. Muhammad, "Pre-Trained CNN Architecture Analysis for Transformer-Based Indonesian Image Caption Generation Model," *JOIV : International Journal on Informatics Visualization*, 2023, [Online]. Available: https://api.semanticscholar.org/CorpusID:259631323

[11] P. Sarikabuta and S. Supratid, "Impacts of Layer Sizes in Deep Residual-Learning Convolutional Neural Network on Flower Image Classification with Different class sizes," in *2022 International Electrical Engineering Congress (iEECON)*, IEEE, Mar. 2022, pp. 1–4. doi: 10.1109/iEECON53204.2022.9741662.

[12] V. K. D. and M. G., "Optimized Deep Learning Approaches for Lung and Colon Cancer Classification using Histopathological Images," in *2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS)*, IEEE, Dec. 2024, pp. 1665–1669. doi: 10.1109/ICACRS62842.2024.10841547.

[13] A. H. Aslamiah, A. H. Rangkuti, A. Ayuliana, V. H. Athala, N. F. Lutffi, and S. V. Aditama, "Improved Accuracy of Animal Skin Pattern retrieval with CNN Model and Distance Metrics," *International Journal of Emerging Technology and Advanced Engineering*, vol. 12, no. 2, pp. 145–152, Feb. 2022, doi: 10.46338/ijetae0222_17.

[14] Aris Wahyu Murdiyanto, D. H. Zulfikar, B. S. Waluyo Poetro, and A. C. Siregar, "Performance Comparison of CNN and ResNet50 for Skin Cancer Classification Using U-Net Segmented Images," *Indonesian Journal of Data and Science*, vol. 5, no. 3, pp. 198–205, Dec. 2024, doi: 10.56705/ijodas.v5i3.200.