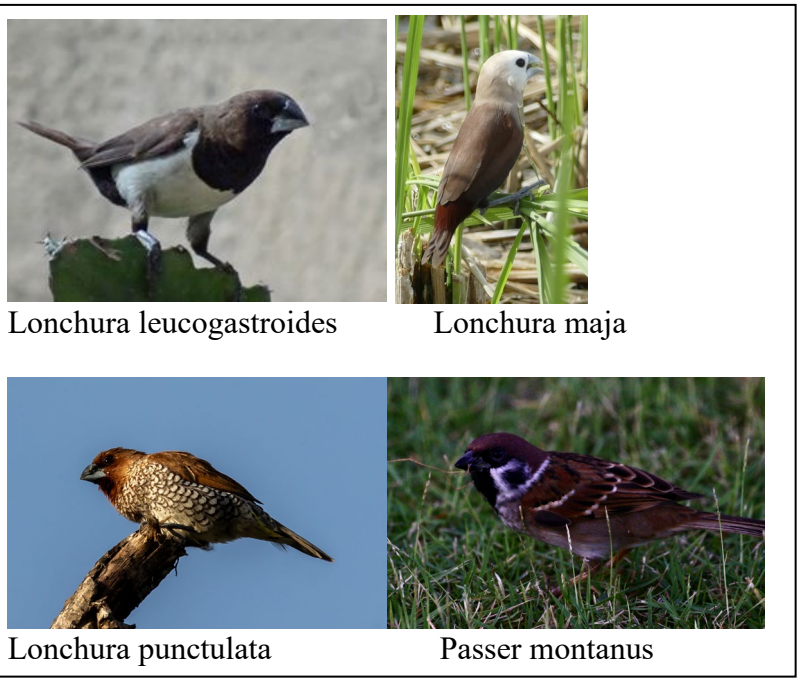
BAB IV
HASIL DAN PEMBAHASAN

4.1. Preprocessing Data

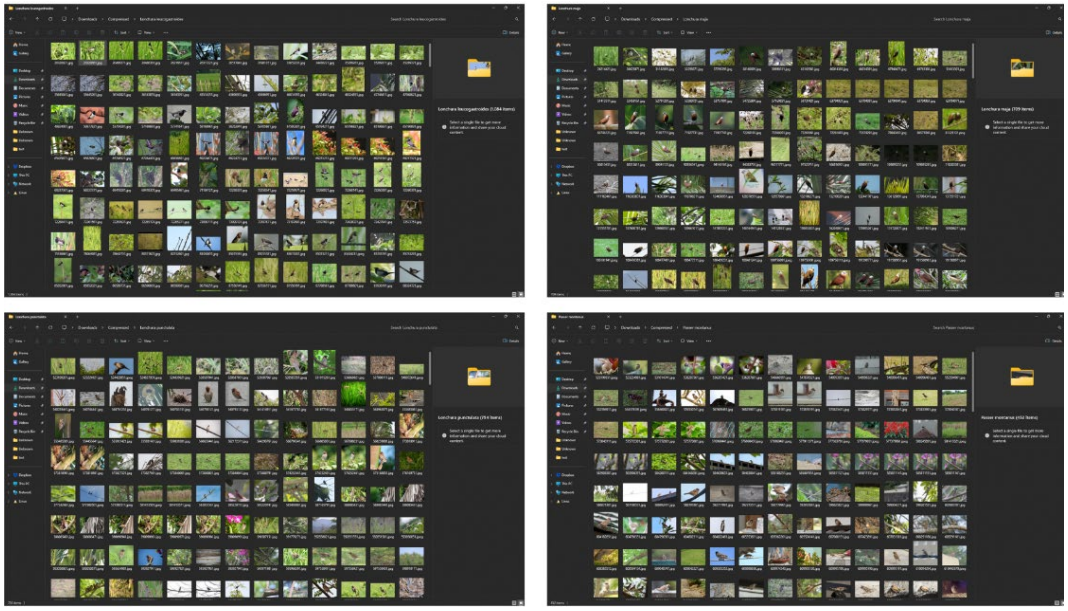
Setelah dilakukan analisis kebutuhan pada pembuatan model maka dilakukan implementasi dari tiap langkah-langkah tersebut yang di mulai dari *data collecting, data cleaning, data labelling* dan *data augmentation,* membagi data ke dalam data uji dan data latih,

4.1.1. Data *Collecting*

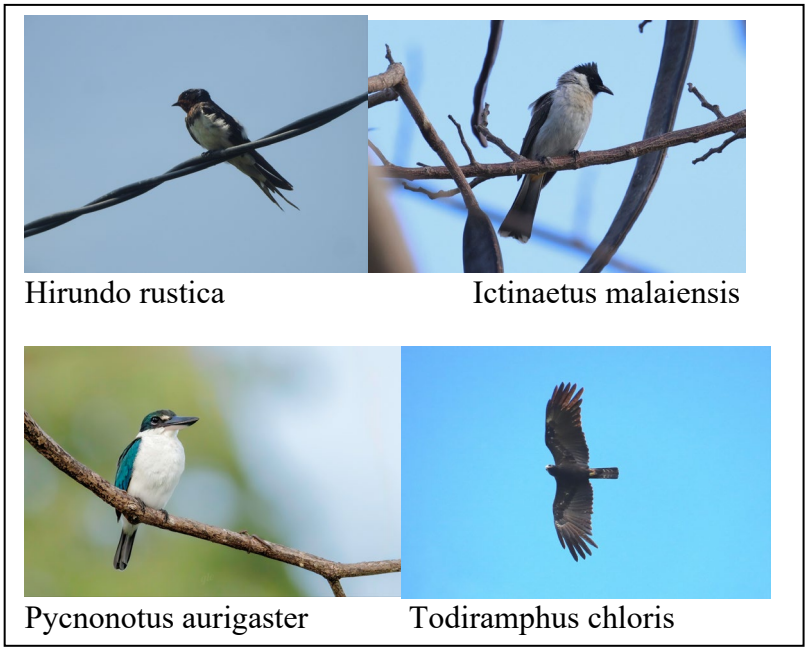
Pengumpulan data dilakukan di website eBird dan untuk data lapangan diambil di lahan padi yang terletak di Kalirejo, Kecamatan Negeri Katon, Kabupaten Pesawaran, Provinsi Lampung. Didapatkan data burung pemakan biji-bijian sebanyak 2.999 gambar yang terdistribusi sebagai berikut: Lonchura leucogastroides 1.084 gambar, Lonchura maja 709 gambar, Lonchura punctulata 754 gambar, dan Passer montanus 452 gambar. Data kelas unknown (selain pemakan biji-bijian) didapatkan sebanyak 635 file yang terdistribusi sebagai berikut: Todiramphus chloris 200 gambar, Pycnonotus aurigaster 200 gambar, Ictinaetus malaiensis 147 gambar, dan Hirundo rustica 88 gambar. Selain itu, terdapat tambahan data lapangan sebanyak 18 gambar. Semua data spesies gambar dari eBird yang disebutkan diunduh dengan filter benua Asia dan negara Indonesia.



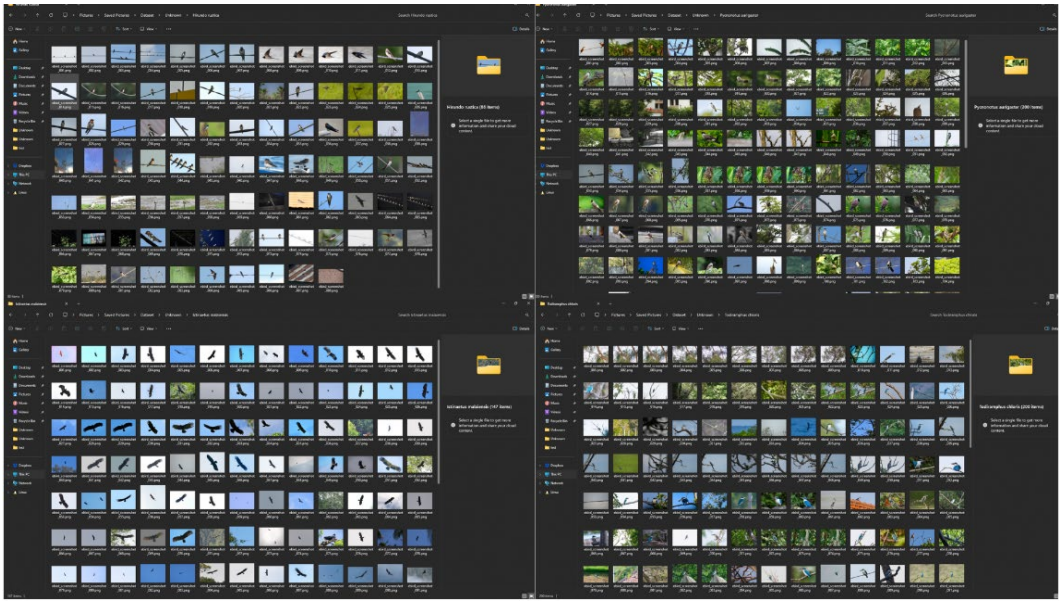
Gambar 11. Spesies Burung Pemakan Biji-bijian



Gambar 12. Hasil Pengumpulan Data Burung Pemakan Biji-Bijian.



Gambar 13. Spesies burung bukan pemakan biji-bijian



Gambar 14. Hasil Pengumpulan Data untuk Kelas Unknown

Gambar 15. Hasil Pengumpulan Gambar Dari Lapangan

Dari hasil pengumpulan gambar burung dari website ebird, dilakukan proses *down sampling* sebelum digunakan. Selanjutnya, gambar disesuaikan ukurannya, kemudian dilakukan pelabelan yang akan diproses pada tahap *data cleaning* dan *data labeling*.

4.1.2. Data *Cleaning*

Tahap data cleaning dilakukan untuk memastikan citra memiliki kualitas yang baik serta ukuran yang seragam. Proses ini meliputi validasi file gambar, penghapusan citra yang rusak (corrupted images), dan penyeragaman ukuran gambar menjadi 224 × 224 piksel. Menggunakan Kode program yang dibuat. Kode berfungsi menyeragamkan ukuran seluruh gambar pada dataset menjadi 224 × 224 piksel. Proses pneyeragaman ukuran gambar dapat di lihat pada gambar 4.3.

Gambar 16. Proses Menyeragamkan Ukuran Gambar

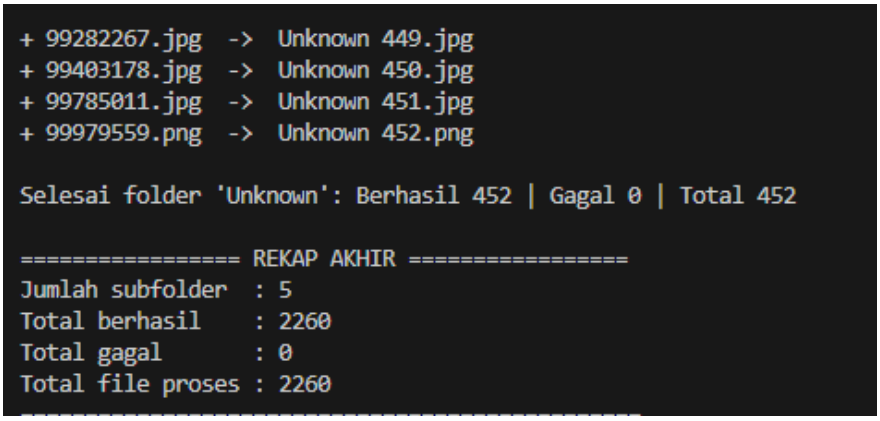
Lalu pada tahap ini juga data di lakukan *downsampling* yaitu proeses dilakukan untuk menyeimbangkan jumlah data pada setiap kelas di dalam dataset. Langkah ini bertujuan agar model tidak bias terhadap kelas dengan jumlah data

yang lebih banyak, sehingga mampu mempelajari pola dari setiap kelas secara seimbang. Dengan distribusi data yang lebih merata yaitu pada setiap kelas data di bagi per kelas 452 gambar, Untuk unknown data digunakan sebanyak 304 ini di lakukan agar model tidak *overfitting* serta mempercepat pelatihan model. Dataset citra yang di ambil dari lapangan di lakukan cleaning yaitu di dapat 20 gambar yang di lakukan downsampling menjadi 18 gambar dikarenakan data terlalu jauh dari kamera. Setelah itu di ubah ukuran nya menjadi 224x224, untuk menyesuaikan model yang dibuat.

4.1.3. Data *Labeling*

Pelabelan data dilakukan untuk mengidentifikasi setiap citra ke dalam kategori burung pemakan biji-bijian yaitu kelas Lonchura leucogastroides, Lonchura maja, Lonchura punctulata, Passer montanus, serta kelas unknown. Meskipun dataset telah dikelompokkan ke dalam folder berdasarkan nama spesies, penamaan berkas belum merefleksikan label spesiesnya. Oleh karena itu, setiap gambar perlu diberi label sesuai spesies (termasuk unknown) agar siap digunakan pada proses klasifikasi.

Implementasi pelabelan tersebut direalisasikan melalui kode program. Dapat dilihat pada hasil output, proses rename dan pelabelan data telah dilakukan secara otomatis berdasarkan struktur direktori yang sudah dikelompokkan sebelumnya. Seluruh file yang berada di dalam subfolder berhasil diubah namanya menjadi format yang sesuai, tanpa ada file yang gagal diproses. Proses ini dilakukan pada total 5 subfolder dengan jumlah keseluruhan 2.112 file, yang seluruhnya berhasil diproses (berhasil: 2.112, gagal: 0). Hasil pada gambar 17.



Gambar 17. Hasil Rename Dataset Dari Folder

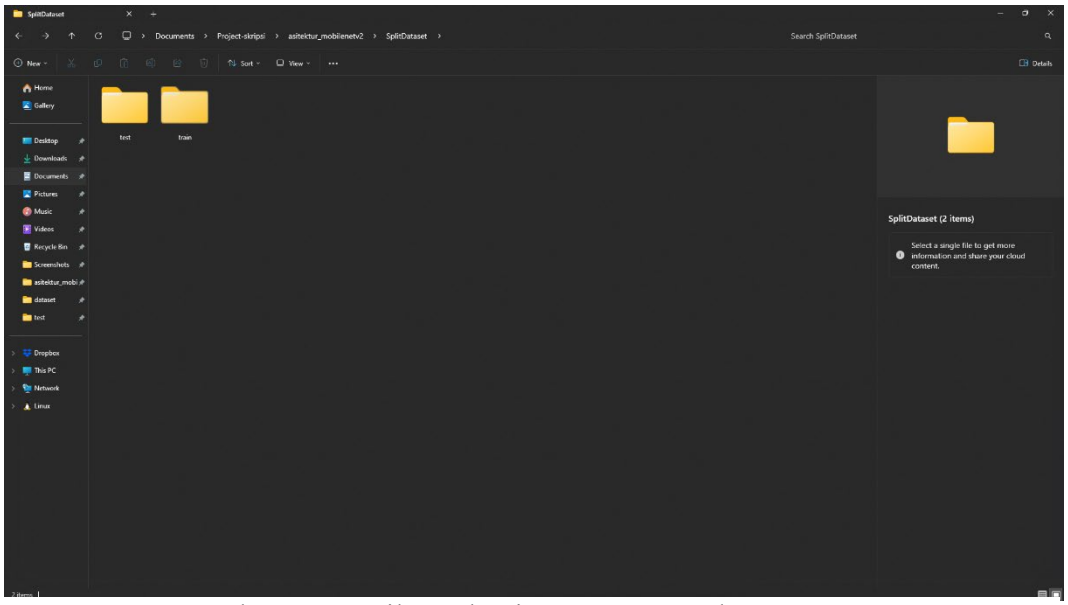
Dataset terdiri dari dua komponen utama:

1. Data training sebanyak 2.112 citra yang dikelompokkan ke dalam lima kelas: *Lonchura leucogastroides*, *Lonchura maja*, *Lonchura punctulata*, *Passer montanus*, serta kelas *unknown*. Struktur folder pelatihan mengikuti masing-masing label tersebut untuk memudahkan proses pembelajaran model yang lalu di split dengan proporsi 70% data latih dan 30% data uji. Pemilihan proporsi 70:30 memberikan jumlah data pelatihan yang cukup besar untuk memungkinkan model CNN mempelajari pola visual dengan baik, sekaligus menyediakan data validasi yang memadai untuk menguji kemampuan generalisasi model terhadap data yang belum pernah dilihat sebelumnya hal ini berdasarkan penelitian (Syaputra dkk., 2022) berikut perhitunganya:

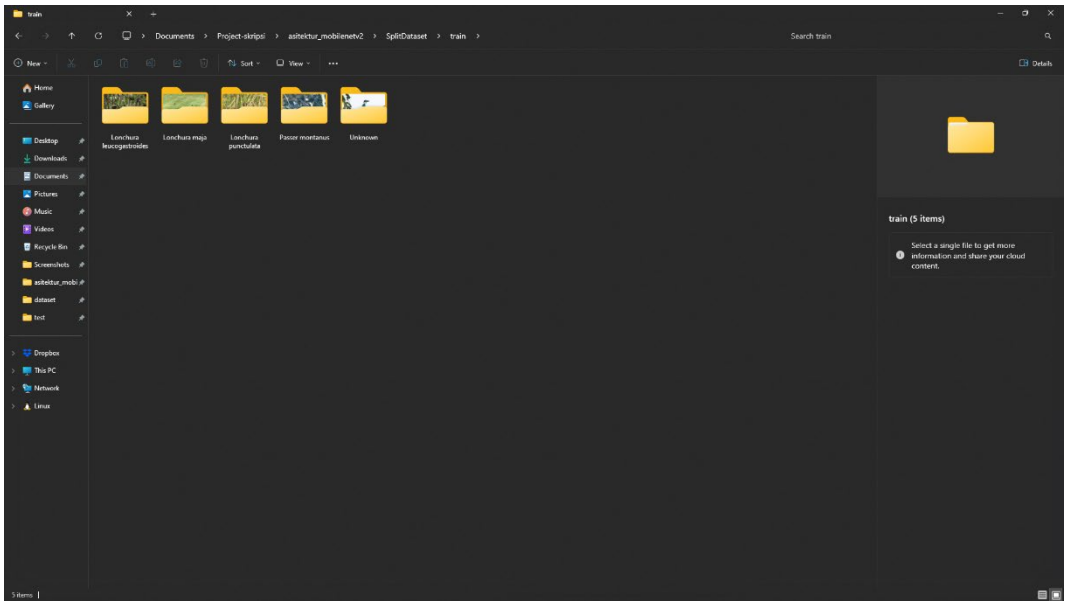
$$\text{Data latih} = 2112 \times 70\% = 1478 \text{ data latih}$$

$$\text{Data uji} = 2112 \times 30\% = 637 \text{ data uji}$$

2. Data testing sebanyak 18 citra yang merupakan data lapangan yang diambil secara langsung di lokasi penelitian. Data test ini diberi nama secara berurutan dari burung01 hingga burung18 dan tidak dikelompokkan ke dalam subfolder berdasarkan spesies, melainkan disimpan dalam satu folder untuk keperluan pengujian model. Visualisasi hasil pengelompokan dapat dilihat pada Gambar 18 dan 19.



Gambar 18. Hasil Pembagian Data *Train* dan Data *Test*



Gambar 19. Kumpulan Kategorisai pada Data *train*

Hasil pengelompokan dataset menghasilkan dua bagian dengan fungsi yang berbeda. Data training telah dikelompokkan ke dalam lima kategori untuk proses pembelajaran model, sedangkan data testing berupa data lapangan dengan penamaan burung01 sampai burung18 digunakan untuk menguji kinerja model dalam kondisi nyata, sehingga dapat diketahui tingkat akurasi yang diperoleh pada data yang belum pernah dilihat oleh model sebelumnya.

4.1.4. Data *Augmentation*

Data augmentation dilakukan dengan tujuan untuk mengurangi *overfitting* pada dataset yang dimiliki dan meningkatkan variasi data pelatihan tanpa menambah data asli secara manual. Teknik augmentasi gambar ini dapat meningkatkan *robustness* model dengan menciptakan variasi dari gambar yang sudah ada. Namun, data hasil augmentasi tersebut hanya digunakan pada tahap pelatihan model, sedangkan untuk validasi dan pengujian menggunakan data asli tanpa augmentasi.

- Parameter Konfigurasi Data *Augmentation*
  Program menggunakan fungsi *ImageDataGenerator* dari Pustaka Keras untuk melakukan augmentasi gambar. Konfigurasi parameter yang digunakan dalam penelitian ini adalah sebagai berikut:

```
# Parameter umum
target_size = (224, 224)
batch_size = 32
num_classes = 5

# Augmentasi untuk data pelatihan
train_datagen = ImageDataGenerator(
  preprocessing_function=preprocess_input,
  rotation_range=15,
  width_shift_range=0.15,
  height_shift_range=0.15,
  zoom_range=0.2,
  shear_range=0.1,
  brightness_range= (0.90, 1.10),
  channel_shift_range=20.0,
  horizontal_flip=True,
  fill_mode="nearest",
  validation_split=0.3
)
```

Parameter-parameter yang digunakan dalam augmentasi data memiliki

fungsi sebagai berikut:

- **preprocessing_function = preprocess_input:** Berfungsi untuk melakukan preprocessing sesuai dengan backbone model yang digunakan, menggantikan fungsi rescale manual untuk normalisasi pixel
- **rotation_range = 15:** Merotasi gambar secara acak dalam rentang ±15 derajat untuk memberikan variasi orientasi objek
- **width_shift_range = 0.15:** Menggeser gambar secara horizontal hingga maksimal 15% dari lebar gambar untuk simulasi perubahan posisi objek
- **height_shift_range = 0.15:** Menggeser gambar secara vertikal hingga maksimal 15% dari tinggi gambar
- **zoom_range = 0.2:** Memperbesar atau memperkecil gambar secara acak hingga 20% untuk simulasi perubahan jarak pengambilan foto
- **shear_range = 0.1:** Memberikan efek memiringkan gambar dari posisi awal yaitu sebesar 0.1 radian untuk variasi perspektif
- **brightness_range = (0.90, 1.10):** Mengubah tingkat kecerahan gambar dalam rentang 90%-110% untuk simulasi kondisi pencahayaan berbeda
- **channel_shift_range = 20.0:** Mengubah intensitas warna antar kanal RGB hingga 20 unit untuk variasi kondisi warna
- **horizontal_flip = True:** Membalik gambar secara horizontal untuk meningkatkan variasi orientasi objek
- **fill_mode = "nearest":** Mengisi pixel kosong hasil transformasi dengan nilai pixel terdekat
- **validation_split = 0.3:** Membagi data menjadi 70% untuk pelatihan dan 30% untuk validasi

Gambar 20. Hasil Data *Augmentation*

Gambar 20 menunjukkan hasil dari proses data *augmentation* yang diterapkan pada dataset burung. Dapat dilihat bahwa augmentasi berhasil menciptakan variasi gambar dengan rotasi, pergeseran, perubahan kecerahan, dan transformasi lainnya sesuai parameter yang telah ditetapkan dalam *training datagen*. Variasi ini membantu model untuk belajar fitur-fitur yang lebih *robust* dan mengurangi kemungkinan *overfitting* pada data pelatihan.

4.2. Pembuatan Model CNN

Pada penelitian ini, lapisan dasar CNN yakni operasi konvolusi, aktivasi non-linier, normalisasi, dan pooling menjadi landasan penggunaan EfficientNetB0 sebagai backbone CNN pra-latih (ImageNet). Inti rancangan adalah memanfaatkan representasi fitur umum dari backbone CNN (konvolusi, normalisasi, aktivasi, dan downsampling) lalu menambahkan head klasifikasi ringan agar spesifik terhadap lima kelas data penelitian.

Strategi transfer learning yang diterapkan adalah feature extraction, dimana seluruh parameter backbone EfficientNetB0 (4,049,571 parameter) yang telah dilatih pada ImageNet dibekukan (base.trainable=False) dan digunakan sebagai feature extractor tetap, sementara hanya head klasifikasi baru (329,221 parameter) yang dapat dilatih untuk menyesuaikan dengan lima kelas burung target. Parameter

training=False pada pemanggilan backbone memastikan bahwa Batch Normalization layers dalam EfficientNetB0 tetap menggunakan statistik yang telah dipelajari dari ImageNet selama inference, bukan mengupdate running means dan variances berdasarkan batch saat ini. Bagian klasifikasi menggunakan Global Average Pooling dan dense head ringan untuk menekan kompleksitas serta meningkatkan generalisasi. Untuk mengatasi ketidakseimbangan kelas dan over-confidence, digunakan label smoothing pada loss Categorical Cross-Entropy dengan koefisien 0.05, sedangkan optimizer Adam dengan learning rate 1e-4 digunakan untuk mengoptimasi hanya parameter head klasifikasi.

Model: "EffNetB0_wrapper"

Layer (type)	Output Shape	Param #
image (InputLayer)	(None, 224, 224, 3)	0
efficientnetb0 (Functional)	(None, 7, 7, 1280)	4,049,571
gap (GlobalAveragePooling2D)	(None, 1280)	0
dropout	(None, 1280)	0
head_dense (Dense)	(None, 256)	327,936
dropout_1	(None, 256)	0
pred (Dense)	(None, 5)	1,285

Total params (inti model): 4,378,792
Trainable params (tahap-1): 329,221
Non-trainable params: 4,049,571

Pendekatan ini memungkinkan model untuk memanfaatkan representasi fitur yang telah dipelajari dari dataset ImageNet yang besar (1,4 juta gambar) sambil tetap dapat mengadaptasi fitur-fitur tersebut untuk tugas klasifikasi burung yang spesifik, dengan proporsi 92.48% parameter berasal dari backbone pre-trained dan 7.52% parameter dari head klasifikasi yang baru. Model terdiri dari beberapa komponen utama:

- Input Layer: Menerima gambar dengan ukuran (224, 224, 3)
- EfficientNetB0 Base: Pre-trained model dari ImageNet dengan 4,049,571 nparameter
- Global Average Pooling: Mengubah feature map (7×7×1280) menjadi vektor 1D (1280)
- Dropout Layer 1: Regularisasi dengan rate 0.4
- Dense Layer: Fully connected layer dengan 256 unit dan aktivasi ReLU
- Dropout Layer 2: Regularisasi dengan rate 0.3
- Output Layer: Dense layer dengan 5 unit (sesuai jumlah kelas) dan aktivasi

softmax

4.3. Proses Pelatihan Model

Sebelum melakukan pelatihan model diperlukan beberapa pengaturan. Pengaturan ini dilakukan agar model yang dihasilkan baik dan juga mempersingkat proses pelatihan. Pada source code terdapat tiga variabel callback yaitu checkpoint, early_stop, dan reduce_lr, dimana fungsinya adalah sebagai berikut:

- Checkpoint = berguna untuk menyimpan setiap perubahan model latih yang terbaik. Checkpoint akan menyimpan dan membandingkan perubahan data setiap epoch. Pada checkpoint kali ini akan menyimpan model dengan nilai val_loss yang minimum.
- EarlyStopping = EarlyStopping berfungsi untuk menghentikan proses training ketika validation loss tidak mengalami perbaikan dalam periode tertentu (patience=10 epoch). Hal ini bertujuan untuk mencegah overfitting dan menghemat waktu komputasi.
- ReduceLROnPlateau = callback ini berfungsi untuk mengurangi learning rate secara otomatis ketika validation loss tidak mengalami perbaikan. Learning rate akan dikurangi dengan faktor 0.5 setelah 2 epoch tidak ada perbaikan, dengan batas minimum learning rate 1e-6.

Setelah pengaturan callback selesai maka selanjutnya adalah melakukan proses latih. Model yang akan dilatih memiliki parameter sebagai berikut:

- train_generator = merupakan data yang akan dilakukan pelatihan yang sudah melalui proses augmentasi
- validation_data = merupakan data yang akan dilakukan validasi untuk mengukur performa model pada data yang tidak pernah dilihat selama training
- epochs = jumlah iterasi maksimum dalam pelatihan model, ditetapkan sebanyak 30 epoch
- callbacks = berfungsi untuk membantu dalam proses pelatihan dan membantu membuat model menjadi lebih baik serta mencegah terjadinya overfitting

- verbose = mengatur tingkat detail informasi yang ditampilkan selama proses training

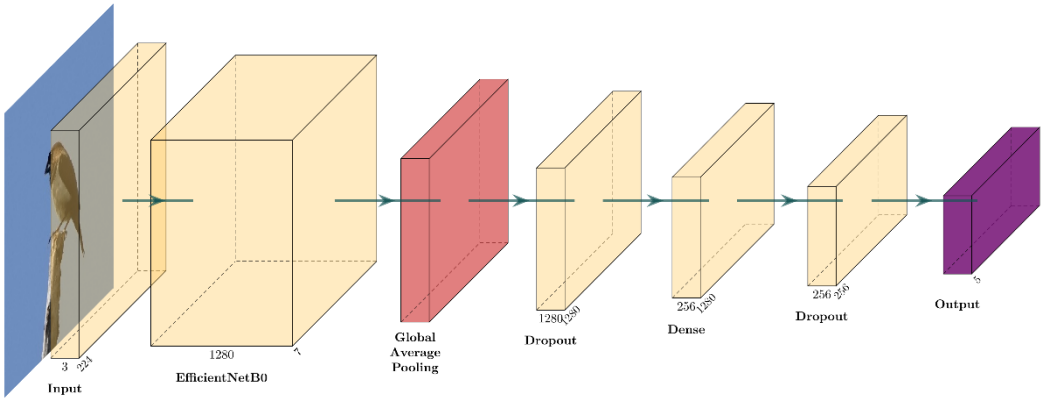
Hasil proses training dapat dilihat di bawah ini:

```
Epoch 1/30
34/34 ━━━━━━━━━━━━━━━━━━━━ 0s 2s/step - accuracy: 0.2578 -
loss: 1.6317
Epoch 1: val_loss improved from inf to 1.32201, saving model
to /content/saved_models/best_efficientnet.keras
34/34 ━━━━━━━━━━━━━━━━━━━━ 165s 2s/step - accuracy: 0.2589 -
loss: 1.6299 - val_accuracy: 0.5256 - val_loss: 1.3220 -
learning_rate: 1.0000e-04
Epoch 2/30
34/34 ━━━━━━━━━━━━━━━━━━━━ 0s 529ms/step - accuracy: 0.5008
- loss: 1.3130
Epoch 2: val_loss improved from 1.32201 to 1.01720, saving
model to /content/saved_models/best_efficientnet.keras
34/34 ━━━━━━━━━━━━━━━━━━━━ 21s 600ms/step - accuracy: 0.5014
- loss: 1.3119 - val_accuracy: 0.6459 - val_loss: 1.0172 -
learning_rate: 1.0000e-04
Epoch 14/30
34/34 ━━━━━━━━━━━━━━━━━━━━ 0s 473ms/step - accuracy: 0.9345
- loss: 0.4040
Epoch 14: val_loss improved from 0.70261 to 0.69148, saving
model to /content/saved_models/best_efficientnet.keras
34/34 ━━━━━━━━━━━━━━━━━━━━ 18s 528ms/step - accuracy: 0.9345
- loss: 0.4043 - val_accuracy: 0.8151 - val_loss: 0.6915 -
learning_rate: 5.0000e-05
Epoch 24/30
34/34 ━━━━━━━━━━━━━━━━━━━━ 0s 468ms/step - accuracy: 0.9624
- loss: 0.3553
Epoch 24: val_loss did not improve from 0.69148

Epoch 24: ReduceLROnPlateau reducing learning rate to
1.56249996052793e-06.
34/34 ━━━━━━━━━━━━━━━━━━━━ 17s 493ms/step - accuracy: 0.9624
- loss: 0.3553 - val_accuracy: 0.8218 - val_loss: 0.7123 -
learning_rate: 3.1250e-06
Epoch 24: early stopping
Restoring model weights from the end of the best epoch: 14.
```

Pada proses pelatihan model EfficientNet yang sudah dirancang, proses latih berhenti pada epoch ke-24 dengan early stopping yang mengembalikan bobot

model terbaik dari epoch ke-14. Hal ini menunjukkan bahwa model telah mencapai performa optimal dan tidak mengalami peningkatan signifikan pada validation loss setelah epoch ke-14. Proses pelatihan menunjukkan peningkatan performa yang konsisten hingga epoch ke-14, dimana akurasi training mencapai 93.45% dan akurasi validasi mencapai 81.51% dengan validation loss sebesar 0.6915.



Gambar 21. Visualisasi Arsitektur

Dalam proses pelatihan model, sebuah gambar akan melewati beberapa tahapan utama dalam arsitektur EfficientNet, yaitu:

1. Preprocessing dan Input Layer

   Pada tahapan pertama, data gambar akan masuk ke input layer dengan ukuran yang telah ditentukan (224x224x3 untuk EfficientNet-B0). Gambar akan dinormalisasi dengan nilai pixel dalam rentang 0-1 melalui pembagian dengan 255, hal ini telah di lakuan ditahap Data Augmentasi.

2. Feature Extraction dengan EfficientNet Backbone

   Model EfficientNet yang telah di-pretrain pada ImageNet akan mengekstrak fitur-fitur penting dari gambar input. EfficientNet menggunakan Mobile Inverted Bottleneck Convolution (MBConv) blocks yang efisien dalam mengekstrak fitur dengan parameter yang relatif sedikit.

3. Global Average Pooling

   Setelah feature extraction, dilakukan Global Average Pooling untuk mengurangi dimensi spatial dari feature maps menjadi vector 1D.

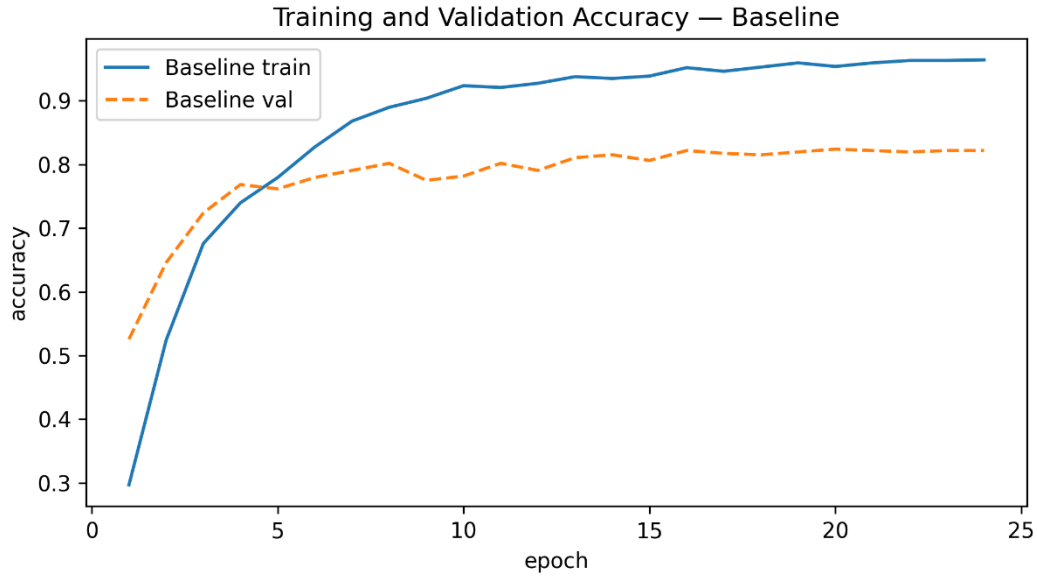
4. Fully Connected Layer

   Tahap terakhir adalah fully connected layer dengan jumlah neuron sesuai dengan jumlah kelas yang akan diprediksi. Pada layer ini ditambahkan fungsi aktivasi

softmax untuk menghasilkan probabilitas klasifikasi.

4.4. Hasil Akurasi dan Loss

Berikut adalah hasil akurasi dan loss yang diperoleh dari proses pelatihan model. Metrik-metrik ini digunakan untuk mengevaluasi efektivitas dan konvergensi proses pembelajaran.



Gambar 22. Hasil Akurasi Dan Validasi Training

Hasil pelatihan model CNN dengan arsitektur EfficientNet-B0 menunjukkan progres pembelajaran yang konsisten selama 24 epoch sebelum dihentikan oleh early stopping. Gambar 19 menampilkan kurva akurasi training dan validasi yang mencerminkan pola pembelajaran yang sehat. Akurasi training menunjukkan peningkatan yang konsisten dari 25.9% pada epoch pertama hingga mencapai puncak 96.2% pada epoch ke-24. Kenaikan akurasi training yang paling signifikan terjadi pada 5 epoch pertama, dimana akurasi melonjak dari 25.9% menjadi 89.7%, menunjukkan kemampuan model dalam mempelajari pola dasar dari dataset burung hama.

Sementara itu, akurasi validasi menunjukkan tren peningkatan yang lebih moderat dan stabil. Dimulai dari 52.6% pada epoch pertama, akurasi validasi mencapai performa terbaik sebesar 82.2% pada epoch ke-24. Stabilitas akurasi validasi di

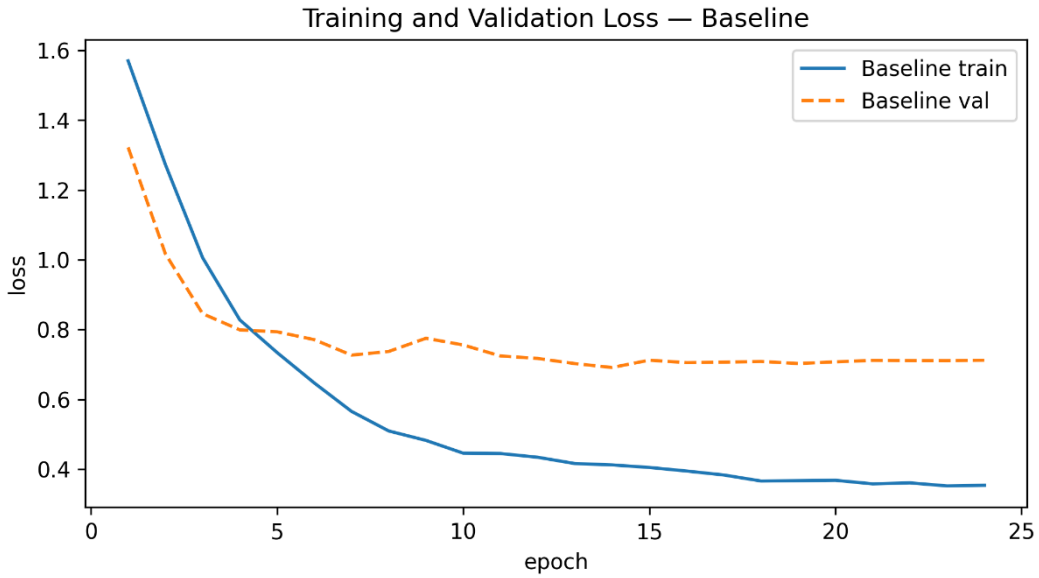
kisaran 80-82% sejak epoch ke-8 menunjukkan bahwa model telah mencapai kemampuan generalisasi yang optimal.

Tabel 3. Hasil Training dan Validation Akurasi

Epoch	Training	Validation
1	0.2969	0.5256
2	0.5240	0.6459
3	0.6758	0.7238
4	0.7399	0.7684
5	0.7795	0.7617
6	0.8275	0.7795
7	0.8680	0.7906
8	0.8897	0.8018
9	0.9039	0.7751
10	0.9237	0.7817
11	0.9208	0.8018
12	0.9274	0.7906
13	0.9378	0.8107
14	0.9350	0.8151
15	0.9387	0.8062
16	0.9519	0.8218
17	0.9463	0.8174
18	0.9529	0.8151
19	0.9595	0.8196
20	0.9538	0.8241
21	0.9595	0.8218
22	0.9632	0.8196
23	0.9632	0.8218
24	0.9642	0.8218

Terdapat gap akurasi antara training dan validasi yang berkisar 10-15% pada epoch-epoch akhir pelatihan. Gap ini masih dalam batas wajar untuk model deep learning dan tidak menunjukkan indikasi overfitting yang parah. Early stopping yang terjadi

pada epoch ke-24 menunjukkan bahwa sistem callback berhasil mencegah degradasi performa lebih lanjut.



Gambar 23. Hasil Training Dan Validation Loss.

Gambar 20 menampilkan evolusi fungsi loss selama proses pelatihan, yang memberikan insight mendalam tentang proses konvergensi model. Training loss menunjukkan penurunan yang konsisten dan smooth dari 1.630 pada epoch pertama hingga 0.355 pada epoch terakhir. Pola penurunan yang eksponensial pada epoch-epoch awal (epoch 1-5) kemudian melambat dan stabil menunjukkan proses optimasi yang berjalan dengan baik. Tidak terdapat fluktuasi yang signifikan, mengindikasikan stabilitas proses pembelajaran.

Validation loss dimulai dari 1.322 dan mencapai nilai minimum 0.691 pada epoch ke-14. Setelah epoch ke-14, validation loss cenderung stabil dengan sedikit fluktuasi, yang menjadi pemicu aktivasi early stopping. Pola ini menunjukkan bahwa model telah mencapai titik optimal generalisasi dan mulai menunjukkan tanda-tanda diminishing returns.

Tabel 4. Hasil Training Dan Validasi Loss

Epoch	Training	Validation
1	1.5698	1.3220
2	1.2730	1.0172
3	1.0061	0.8457

4.5. Pengujian Data *Testing*

Setelah proses pelatihan (training) selesai dilakukan dan diperoleh model CNN terbaik, tahap selanjutnya yaitu melakukan pengujian (testing) menggunakan data lapangan. Pengujian ini bertujuan untuk mengukur kemampuan model dalam mengenali objek secara akurat pada data yang belum pernah dilihat sebelumnya. Pengujian dilakukan menggunakan 18 citra uji lapangan yang terdiri dari beberapa kelas burung pemakan padi seperti *Lonchura leucogastroides*, *Lonchura punctulata*, *Lonchura maja*, dan *Passer montanus*. Hasil prediksi yang diperoleh dari model terhadap citra-citra uji tersebut dapat dilihat pada Gambar 24, di mana setiap citra menampilkan hasil prediksi model beserta tingkat kepercayaannya (confidence score).

Gambar 24. Hasil Prediksi Data Testing

Dari hasil pengujian model menggunakan 18 data uji, diperoleh tingkat prediksi yang bervariasi dengan nilai *confidence* yang berbeda-beda pada setiap citra. Beberapa prediksi menunjukkan tingkat kepercayaan yang tinggi di atas 80%, sedangkan sebagian lainnya memiliki nilai confidence di bawah 50%, menandakan ketidakpastian model terhadap kelas yang diprediksi. Untuk melihat hasil pengujian model secara lebih jelas, dapat dilihat pada Tabel 5 yang berisi hasil prediksi model CNN terhadap 18 data uji lapangan.

Table 5. Hasil Pengujian Model CNN

No	**Result**	**Prediction**
1	0.6260	Lonchura leucogastroides
2	0.9197	Lonchura leucogastroides
3	0.5778	Lonchura leucogastroides
4	0.3396	Lonchura punctulata
5	0.8797	Lonchura leucogastroides
6	0.4603	Lonchura punctulata
7	0.3461	Lonchura punctulata
8	0.8515	Lonchura leucogastroides
9	0.3865	Lonchura leucogastroides
10	0.5938	Lonchura leucogastroides
11	0.5495	Lonchura maja
12	0.5922	Lonchura maja
13	0.5142	Lonchura punctulata
14	0.4030	Lonchura maja
15	0.3741	Lonchura leucogastroides
16	0.4196	Lonchura punctulata
17	0.3158	Lonchura punctulata
18	0.3621	Lonchura punctulata

Dari tabel 5 di dapat dilihat bahwa sebagian besar hasil prediksi menunjukkan bahwa model cenderung mengklasifikasikan objek ke dalam kelas *Lonchura*

leucogastroides dan *Lonchura punctulata*, yang menandakan bahwa kedua kelas tersebut memiliki karakteristik visual yang lebih dominan dikenali oleh model.

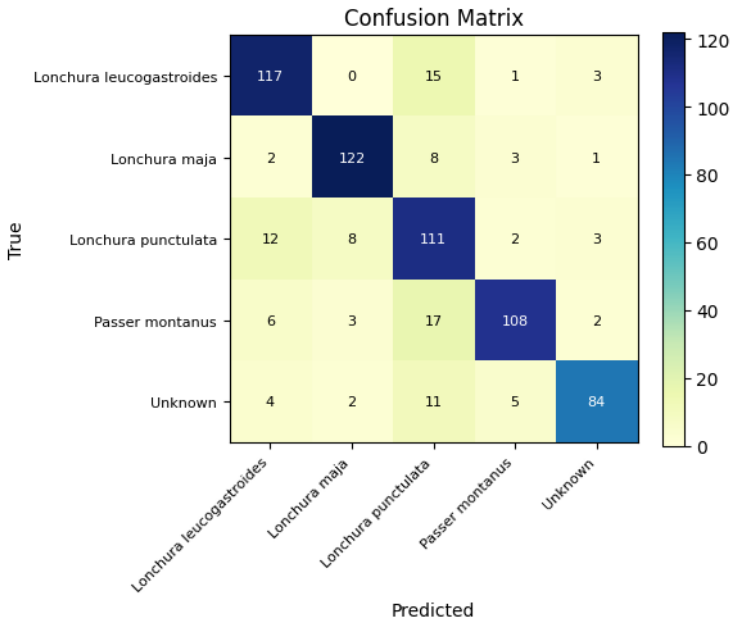
Beberapa hasil prediksi dengan nilai *confidence* di bawah 50% mengindikasikan bahwa model masih mengalami kesulitan dalam membedakan fitur antar kelas akibat kondisi citra lapangan yang memiliki latar belakang bervariasi dan citra terlalu jauh sehingga ketika citra di kecilkan resolusinya menjadi 224x224 pixel menjadi pecah. Secara keseluruhan, model CNN mampu melakukan prediksi terhadap data lapangan. Akurasi total dapat dihitung dari rumus akurasi (4) yang di dapat dari model sebagai berikut:

$$\text{akurasi} = \frac{9.1577}{18} = 0.5088$$

Hasil pengujian menunjukkan bahwa dari 18 data uji di dapat akurasi 50%, model berhasil memprediksi dengan benar pada beberapa citra dengan *confidence* tinggi, sedangkan sebagian lainnya masih perlu dilakukan peningkatan akurasi melalui penambahan data mentah yang bersih dari noise gambar dan memiliki kualitas gambar yang lebih tinggi.

4.6. Confusion Matrix

Berikut adalah hasil dari evaluasi *matrix* menggunakan *confusion matrix*.



Gambar 25. Hasil Confusion Matrix Data Training

Gambar 25 memperlihatkan confusion matrix hasil evaluasi model CNN berbasis EfficientNetB0 untuk klasifikasi lima kelas, yaitu *Lonchura leucogastroides*, *Lonchura maja*, *Lonchura punctulata*, *Passer montanus*, serta kelas *unknown*. Dari gambar dapat diamati beberapa temuan penting. Pada indeks ke-0 (*Lonchura leucogastroides*), model berhasil mengklasifikasikan dengan benar 117 data dari total keseluruhan citra uji. Namun, masih terdapat kesalahan klasifikasi terhadap 15 data yang diprediksi sebagai *Lonchura punctulata*. Selanjutnya, pada indeks ke-1 (*Lonchura maja*), model mampu mengenali 122 data secara tepat, dengan sejumlah kecil kesalahan prediksi yang tersebar pada kelas *Lonchura punctulata* dan *Passer montanus*. Pada indeks ke-2 (*Lonchura punctulata*), model berhasil mengklasifikasikan 111 data dengan benar. Akan tetapi, terdapat beberapa kebingungan klasifikasi dengan kelas *Lonchura leucogastroides* dan *Lonchura maja*, yang mengindikasikan adanya kemiripan fitur visual antarspesies bondol. Indeks ke-3 (*Passer montanus*) menunjukkan performa yang baik dengan 108 data yang diklasifikasikan secara benar. Meski demikian, terdapat 17 data yang salah terklasifikasi sebagai *Lonchura punctulata*, yang dapat dipengaruhi oleh kesamaan pola warna tubuh antara pipit dengan bondol. Sementara itu, pada indeks ke-4 (kelas *unknown*), model mampu mengidentifikasi 84 data dengan benar, namun masih terjadi kesalahan pada sejumlah kecil citra yang terklasifikasi sebagai kelas burung pemakan biji, khususnya *Lonchura punctulata*.

Dari hasil *confusion matrix,* dilakukan perhitungan metrik evaluasi *precision, recall, dan f1*-score pada setiap kelas, sebagaimana ditampilkan pada Tabel 6.

Tabel 6. Perbandingan *Score* Klasifikasi (*Classficiatoin Report*)

Classification	Precision	Recall	F1- Score	Support
Lonchura punctulata	0.6852	0.8162	0.7450	136
Unknown	0.9032	0.7925	0.8442	106
Lonchura leucogastroides	0.8298	0.8603	0.8448	136
Passer montanus	0.9076	0.7941	0.8471	136
Lonchura maja	0.9037	0.8971	0.9004	136

Dari tabel 6 klasifikasi burung dengan score tertinggi adalah Lonchura maja sedangkan score terkecil Lonchura puntulata.

Selain evaluasi metrik kuantitatif, analisis lebih lanjut dilakukan terhadap 20 pasangan kebingungan terbesar yang muncul pada confusion matrix. Analisis ini bertujuan untuk mengidentifikasi pola kesalahan klasifikasi model dan mencari penyebab biologis maupun teknis.

Tabel 7. 20 Pasangan Kebingungan Terbesar

True	Prediksi	count
Passer montanus	Lonchura punctulata	17
Lonchura leucogastroides	Lonchura punctulata	15
Lonchura punctulata	Lonchura leucogastroides	12
Unknown	Lonchura punctulata	11
Lonchura maja	Lonchura punctulata	8
Lonchura punctulata	Lonchura maja	8
Passer montanus	Lonchura leucogastroides	6
Unknown	Passer montanus	5
Unknown	Lonchura leucogastroides	4
Lonchura leucogastroides	Unknown	3
Lonchura maja	Passer montanus	3
Lonchura punctulata	Unknown	3
Passer montanus	Lonchura maja	3
Unknown	Lonchura maja	2
Lonchura maja	Lonchura leucogastroides	2
Lonchura punctulata	Passer montanus	2
Passer montanus	Unknown	2
Lonchura leucogastroides	Passer montanus	1
Lonchura maja	Unknown	1

Pada tabel 7 diatas kesalahan prediksi paling banyak terjadi antarspesies bondol (*Lonchura leucogastroides*, *L. maja*, dan *L. punctulata*). Tingginya tingkat kebingungan di antara kelas ini dipengaruhi oleh kemiripan morfologi yang sangat tipis antarspesies dalam genus *Lonchura*, terutama pada pola warna bulu bagian dada dan kepala. Selain itu, ditemukan pula kebingungan antara *Lonchura*

punctulata dengan *Passer montanus*. Hal ini dapat dijelaskan karena keduanya memiliki pola warna dominan coklat keabu-abuan yang sulit dibedakan oleh model, terutama bila citra diambil pada jarak jauh. Kelas *unknown* relatif stabil, tetapi masih menimbulkan kesalahan prediksi terhadap kelas *Lonchura punctulata* dan *L. leucogastroides*. Berikut adalah contoh gambar dengan spesies berbeda tetapi mirip karena masih satu genus:

Gambar 26. Lonchura punctulata dan L. leucogastroides

Hal ini menunjukkan bahwa variasi latar belakang pada kelas *unknown* (misalnya burung dengan vegetasi padat di sekitarnya) dapat memunculkan fitur visual yang tumpang tindih dengan kelas target. Selain kemiripan biologis antarspesies, kualitas citra juga menjadi faktor penting. Banyak citra dalam dataset merupakan hasil pengambilan gambar dari jarak jauh, dengan ukuran burung relatif kecil di dalam frame. Kondisi ini membuat fitur visual yang seharusnya menjadi pembeda antarspesies tidak terbaca dengan baik oleh model. Lebih lanjut, adanya noise visual seperti dedaunan, batang padi, atau objek lain di sekitar burung, juga memperbesar potensi kesalahan klasifikasi. Noise ini berfungsi sebagai *distractor* bagi CNN, sehingga model cenderung salah fokus pada pola teskstur lingkungan ketimbang karakteristik morfologi burung itu sendiri.

4.7. Cross Validation

Untuk memastikan robustness dan kemampuan generalisasi model CNN, dilakukan evaluasi menggunakan teknik Stratified K-Fold Cross Validation dengan k=5 fold. Berbeda dengan pendekatan evaluasi sederhana, implementasi cross validation

pada penelitian ini menggunakan pendekatan full training and evaluation, dimana model dilatih dari awal pada setiap fold untuk memastikan estimasi performa yang tidak bias dan representatif. Proses cross validation dilakukan dengan langkah-langkah berikut:

1. Seluruh dataset (training dan validation) digabungkan untuk memastikan pembagian yang konsisten data di ambil dari proses pelatihan model.
2. Dataset dibagi menjadi 5 fold menggunakan StratifiedKFold dengan shuffle=True dan random_state=42 untuk mempertahankan proporsi kelas yang seimbang di setiap fold dan juga sama seperti proses pelatihan
3. Untuk setiap fold, dibuat generator terpisah dengan augmentasi data untuk training dan tanpa augmentasi untuk validation
4. Model EfficientNet di ambil dari model pelatihan sebelumnya dan dilatih dari awal pada setiap fold dengan konfigurasi yang sama pada pelatihan
5. Model terbaik dari setiap fold dievaluasi menggunakan berbagai metrix Accuracy, Macro F1, Log Loss, AUC Macro
6. Hasil dari semua fold diagregasi untuk mendapatkan estimasi performa yang robust

Setiap fold menggunakan konfigurasi training yang identic dengan proses training:

- Epochs: Maksimal 30 epoch dengan early stopping
- Callbacks: ModelCheckpoint (save best model), EarlyStopping (patience=10), dan ReduceLROnPlateau (factor=0.5, patience=2)
- Data Augmentation: Rotasi (15°), translasi (15%), zoom (20%), shear (10%), brightness adjustment (90-110%), channel shift, dan horizontal flip
- Monitor Metric: Validation loss untuk early stopping dan learning rate reduction

Berdasarkan implementasi Stratified K-Fold Cross Validation, diperoleh hasil evaluasi performa model pada 5 fold dengan distribusi yang seimbang per fold. Setiap fold menjalankan training lengkap hingga konvergensi atau mencapai batas maksimum 30 epoch.

Tabel 8. Hasil 5 K-Fold Cross Validation Model

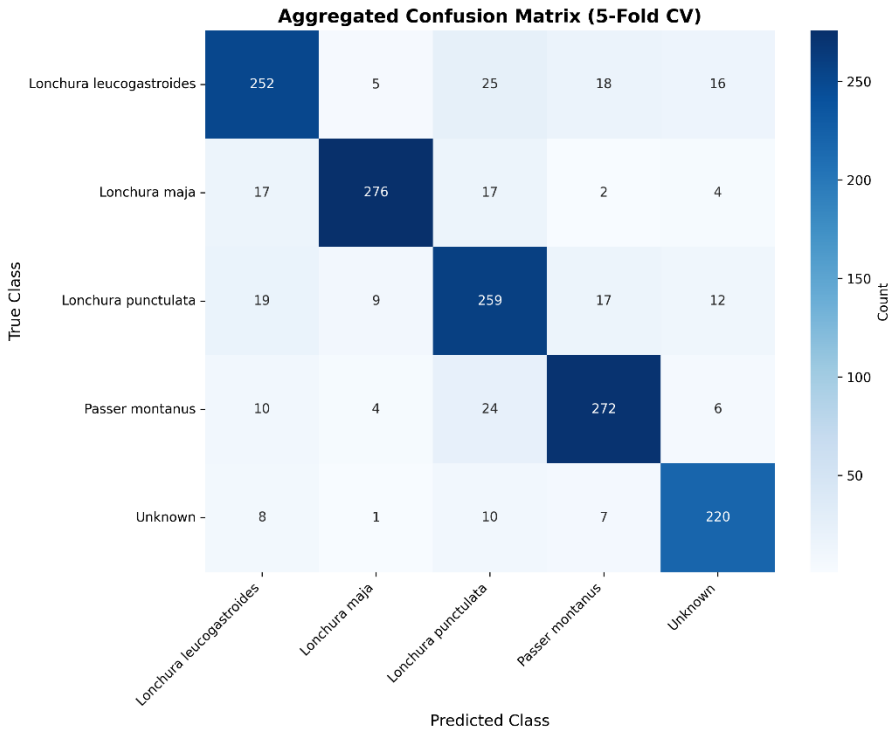
Fold	Data Train	Data Val	Akurasi	Macro F1	Log Loss	AUC Macro	Val Loss Terbaik	Epoch
1	1208	302	0.8377	0.8384	0.4791	0.9710	0.6351	30
2	1208	302	0.8510	0.8527	0.3932	0.9813	0.5560	30
3	1208	302	0.8709	0.8716	0.4371	0.9758	0.5900	30
4	1208	302	0.8311	0.8326	0.4508	0.9760	0.6015	30
5	1208	302	0.8444	0.8469	0.5057	0.9672	0.6539	30

Tabel 9. Ringkasan Statistik Cross Validation

Metrik	**Mean ± Std**
Accuracy	0.8470 ± 0.0152
Macro F1	0.8484 ± 0.0151
Log Loss	0.4532 ± 0.0427
AUC Macro	0.9742 ± 0.0054

Analisis karakteristik training menunjukkan bahwa semua fold mencapai batas maksimum 30 epoch, mengindikasikan bahwa model masih dalam proses pembelajaran dan belum mencapai konvergensi penuh. Variasi best validation loss antar fold (0.5560 - 0.6539) menunjukkan adanya perbedaan kompleksitas data antar fold, dengan Fold 2 menunjukkan konvergensi terbaik (val_loss: 0.5560) dan Fold 5 menunjukkan learning yang paling challenging (val_loss: 0.6539).

- Fold terbaik: Fold 3 dengan accuracy 87.09%
- Fold terendah: Fold 4 dengan accuracy 83.11%
- Range variasi: 3.98%, yang masih dalam batas toleransi yang baik untuk variabilitas natural antar fold



Gambar 27. Aggregate Confusion Matrix Cross Validation

Untuk memberikan gambaran yang lebih komprehensif tentang performa model, dilakukan agregasi confusion matrix dari seluruh fold cross validation. Gambar 27 menunjukkan aggregated confusion matrix yang merepresentasikan total prediksi dari 1510 sampel validasi (302 sampel × 5 fold).

Performa Per Kelas (Class-wise Accuracy):

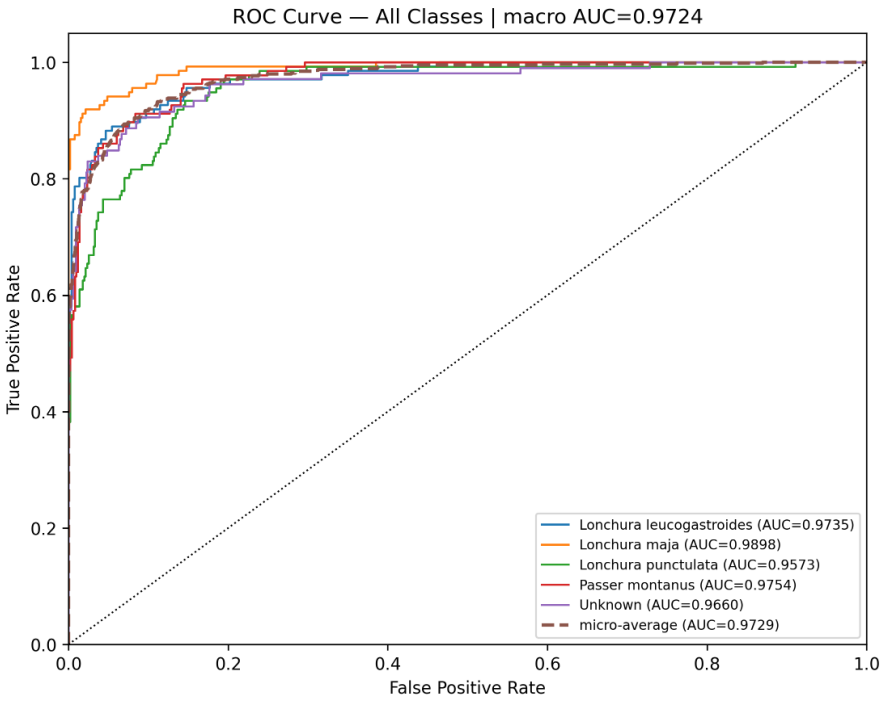
- Lonchura leucogastroides: 252/316 = 79.7% accuracy
- Lonchura maja: 276/316 = 87.3% accuracy
- Lonchura punctulata: 259/316 = 82.0% accuracy
- Passer montanus: 272/316 = 86.1% accuracy
- Unknown: 220/246 = 89.4% accuracy

Distribusi kesalahan yang relatif merata antar kelas menunjukkan bahwa model tidak mengalami bias yang signifikan terhadap kelas tertentu, meskipun terdapat sedikit variasi dalam jumlah sampel per kelas.

4.8. ROC & AUC

klasifikasi multi-class dengan lima kelas (Lonchura leucogastroides, Lonchura maja, Lonchura punctulata, Passer montanus, dan Unknown), implementasi ROC-

AUC menggunakan pendekatan One-vs-Rest (OvR). Setiap kelas diperlakukan sebagai kelas positif terhadap gabungan semua kelas lainnya sebagai kelas negatif, sehingga menghasilkan kurva ROC individual untuk masing-masing kelas.



Gambar 28. Kurva ROC untuk semua kelas

Berdasarkan gambar 23 kurva ROC yang dihasilkan, model menunjukkan performa yang sangat baik dengan macro AUC mencapai 0.9724. Hasil ini mengkonfirmasi tingkat kemampuan deskriminasi yang tinggi dari model CNN berbasis EfficientNetB0. Evaluasi AUC individual untuk setiap kelas menunjukkan performa yang konsisten tinggi:

Tabel 10. Nilai Auc Per Kelas

Class	AUC Score
Lonchura maja	0.9898
Passer montanus	0.9754
Lonchura leucogastroides	0.9735
Micro-average	0.9729
Lonchura punctulata	0.9573
Unknown	0.9660

Nilai macro AUC sebesar 0.9724 mengindikasikan kemampuan diskriminasi yang

sangat baik. Nilai ini menunjukkan bahwa model memiliki probabilitas 97.24% untuk memberikan skor yang lebih tinggi kepada sampel positif dibandingkan sampel negatif secara acak. Semua kelas menunjukkan nilai AUC di atas 0.95, mengindikasikan tidak ada kelas yang secara signifikan lebih sulit dibedakan dibandingkan yang lain. Rentang variasi AUC yang sempit (0.9573 - 0.9898) menunjukkan keseimbangan performa model disemua kelas. Kelas Lonchura maja mencapai AUC tertinggi (0.9898), hampir mendekati performa pengklasifikasi sempurna. Hal ini konsisten dengan hasil confusion matrix yang menunjukkan akurasi klasifikasi terbaik untuk spesies ini. Meskipun masih dalam kategori bagus sekali (AUC = 0.9573), kelas Lonchura punctulata menunjukkan nilai AUC terendah. Hal ini sejalan dengan temuan pada confusion matrix dimana terdapat beberapa konfusi dengan spesies Lonchura lainnya.

Model CNN yang dikembangkan memiliki kemampuan diskriminasi yang sangat baik untuk mengidentifikasi spesies burung hama pemakan biji-bijian, dengan performa yang konsisten dan reliable untuk implementasi praktis dalam sistem smart farming.