



**KLASIFIKASI *FINE-GRAINED* DAUN MANGROVE  
MENGGUNAKAN RES4NET-CBAM DI DESA EKOWISATA  
CUKU NYI NYI**

**NASKAH SKRIPSI**

**Abdurrahman Al-atsary  
121450128**

**PROGRAM STUDI SAINS DATA  
FAKULTAS SAINS  
INSTITUT TEKNOLOGI SUMATERA  
LAMPUNG SELATAN**

**2025**



**KLASIFIKASI *FINE-GRAINED* DAUN MANGROVE  
MENGGUNAKAN RES4NET-CBAM DI DESA EKOWISATA  
CUKU NYI NYI**

**NASKAH SKRIPSI**  
**Diajukan sebagai syarat maju sidang tugas akhir**

**Abdurrahman Al-atsary**  
**121450128**

**PROGRAM STUDI SAINS DATA  
FAKULTAS SAINS  
INSTITUT TEKNOLOGI SUMATERA  
LAMPUNG SELATAN**

**2025**

## **HALAMAN PENGESAHAN**

Naskah Skripsi untuk Sidang Akhirdengan judul "**Klasifikasi Fine-grained Daun Mangrove Menggunakan Res4Net-CBAM di Desa Ekowisata Cuku Nyi Nyi**" adalah benar dibuat oleh saya sendiri dan belum pernah dibuat dan diserahkan sebelumnya, baik sebagian ataupun seluruhnya, baik oleh saya ataupun orang lain, baik di Institut Teknologi Sumatera maupun di institusi pendidikan lainnya.

Lampung Selatan, .... 2025

Penulis,



**Abdurrahman Al-atsary**  
**NIM. 121450128**

Diperiksa dan disetujui oleh,

Pembimbing I

Pembimbing II

**Mika Alvionita S, M.Si**  
**NRK. 1993050920212258**

**Luluk Muthoharoh, M.Si**  
**NIP. 199504112022032014**

Disahkan oleh,

Koordinator Program Studi Sains Data  
Fakultas Sains  
Institut Teknologi Sumatera

**Tirta Setiawan, S.Pd., M.Si**  
**NIP. 199008222022031003**

Sidang Tugas Akhir: 16 Juni 2025

Penguji I : Christyan Tamaro Nadeak, M.Si

Penguji II : Triyana Muliawati, S.Si., M.Si.

## **HALAMAN PERNYATAAN ORISINALITAS**

**Skripsi ini adalah karya saya sendiri dan semua sumber baik yang dikutip maupun yang dirujuk telah saya nyatakan benar.**

**Nama : Abdurrahman Al-atsary**

**NIM : 121450128**

**Tanda tangan :**

**Tanggal : .... 2025**

## **HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI UNTUK KEPENTINGAN AKADEMIS**

Sebagai civitas akademik Institut Teknologi Sumatera, saya yang bertanda tangan di bawah ini:

Nama : Abdurrahman Al-atsary  
NIM : 121450128  
Program Studi : Sains Data  
Fakultas : Sains  
Jenis karya : Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan Hak Bebas Royalti Non-eksklusif (*Non-Exclusive Royalty Free Right*) kepada Institut Teknologi Sumatera atas karya ilmiah saya yang berjudul:

### **Klasifikasi *Fine-grained* Daun Mangrove Menggunakan Res4Net-CBAM di Desa Ekowisata Cuku Nyi Nyi**

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Institut Teknologi Sumatera berhak menyimpan, mengalih media/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Lampung Selatan  
Pada tanggal : .... 2025

Yang menyatakan

**Abdurrahman Al-atsary**

## ABSTRAK

### **Klasifikasi Fine-grained Daun Mangrove Menggunakan Res4Net-CBAM di Desa Ekowisata Cuku Nyi Nyi**

Abdurrahman Al-atsary (121450128)

Pembimbing I: Mika Alvionita S, M.Si

Pembimbing II: Luluk Muthoharoh, M.Si

*Identifikasi spesies mangrove secara akurat sangat penting untuk mendukung konservasi dan edukasi di kawasan ekowisata Desa Cuku Nyi Nyi, Lampung Selatan. Tantangan utama dalam klasifikasi mangrove terletak pada kemiripan morfologi antar spesies, terutama pada struktur daun yang sulit dibedakan secara visual. Model CNN konvensional sering gagal menangkap perbedaan halus antar objek sejenis, karena kurang mampu memfokuskan perhatian pada fitur lokal penting. Penelitian ini bertujuan melakukan pendekatan berbasis attention mechanism untuk meningkatkan akurasi dan ketepatan dalam klasifikasi mangrove dan mengusulkan sebuah model Res4Net-CBAM, yaitu arsitektur Res4Net yang diintegrasikan dengan Convolutional Block Attention Module (CBAM), untuk klasifikasi fine-grained tiga spesies mangrove: Avicennia alba, Rhizophora apiculata, dan Rhizophora stylosa. Sebanyak 300 citra daun dikumpulkan menggunakan kamera smartphone beresolusi 50 MP dan diproses melalui tahapan pre-processing, augmentasi. Model dilatih menggunakan Adam optimizer dengan learning rate  $10^{-4}$  memberikan hasil terbaik dari beberapa kombinasi yang dilatih dengan optimizer Adam dan SGD dan learning rate dengan rentang  $10^{-3}$  sampai  $10^{-5}$  dan model terbaik yang dihasilkan mencapai akurasi, recall, serta F1-score sebesar 96%. Hasil ini menunjukkan efektivitas CBAM dalam meningkatkan fokus ekstraksi fitur dan akurasi klasifikasi spesies mangrove. Penelitian ini juga dapat dikembangkan untuk studi-studi lanjutan yang menangani kasus-kasus unik dengan pola data yang memiliki perbedaan kecil dalam karakteristik objek*

**Kata kunci :** CBAM, Deep Learning, Klasifikasi fine-grained, Mangrove, Res4Net.

## ***ABSTRACT***

### ***Fine-grained Classification of Mangrove Leaves Using Res4Net-CBAM in Cuku NyiNyi Ecotourism Village***

Abdurrahman Al-atsary (121450128)

*Advisor I : Mika Alvionita S, M.Si*

*Advisor II: Luluk Muthoharoh, M.Si*

*The identification of mangrove species accurately is very important to support conservation and education in the ecotourism area of Cuku Nyi Nyi Village, South Lampung. The primary challenge in mangrove classification lies in the morphological similarities between species, especially in leaf structures that are difficult to distinguish visually. Traditional CNN models often fail to capture subtle differences between similar objects, as they lack the ability to focus attention on important local features. This research proposes an attention mechanism-based approach to improve accuracy and precision in mangrove classification and proposes a Res4Net-CBAM model, which is a ResNet architecture integrated with a Convolutional Block Attention Module (CBAM), for the classification of three mangrove species: Avicennia alba, Rhizophora apiculata, and Rhizophora stylosa. A total of 300 leaf images were collected using a 50 MP resolution camera and processed through the stages of pre-processing, augmentation. The model trained using the Adam optimizer with learning rate  $10^{-4}$  gave the best results from several combinations trained with Adam and SGD optimizers and learning rates with a range of  $10^{-3}$  to  $10^{-5}$  and the resulting best model achieved accuracy, recall, and F1-score of 96%. These outcomes demonstrate the effectiveness of CBAM in improving the focus of feature extraction and classification accuracy of mangrove species. This research can also be extended to further studies that deal with unique cases with data patterns that have small differences in object characteristics.*

***Keywords : CBAM, Deep Learning, Fine-grained Classification, Mangrove, Res4Net.***

## **MOTTO**

*The best research is research that can make us realize how great God's power is over this universe.*

## **HALAMAN PERSEMBAHAN**

*Untuk semua yang berkontribusi pada penelitian ini, saya mengucapkan terima kasih sebesar-besarnya*

## **KATA PENGANTAR**

Puji syukur penulis panjatkan ke hadirat Allah SWT atas berkah dan rahmat-Nya sehingga skripsi ini dapat terselesaikan dengan baik. Skripsi ini dibuat untuk menyelesaikan pendidikan jenjang sarjana pada Institut Teknologi Sumatera. Penyusunan skripsi ini banyak mendapat bantuan dan dukungan dari berbagai pihak sehingga dalam kesempatan ini, dengan penuh kerendahan hati, penulis mengucapkan terima kasih kepada:

1. Ibu Mika Alvionita S, M.Si selaku Dosen Pembimbing pertama saya yang selalu memberikan masukan dan arahan dalam menyelesaikan dan perancangan skripsi ini.
2. Ibu Luluk Muthoharoh, M.Si selaku Dosen Pembimbing kedua saya yang selalu memberikan masukan, diskusi dan memberikan masukan mengenai penulisan yang sangat membantu saya dalam menyelesaikan skripsi ini.
3. Bapak Christyan Tamara Nadeak, M.Si, dan Ibu Triyana Muliawati M.Si, selaku Dosen Penguji saya yang telah menguji saya dalam skripsi ini dan memberikan masukan yang berharga sebagai arahan revisi dari metode dan penjelasan yang saya paparkan pada skripsi ini.
4. Kedua orang tua dan saudara-saudara saya yang selalu mendukung saya setiap melangkah lebih jauh serta menuntun saya untuk lebih semangat dalam pengerjaan skripsi ini.
5. Partner belajar bersama Miftahul Huda, Husni Na'fa Mubarok dan Sasa Rahma Lia yang selalu bisa mendengarkan dan mengajak diskusi pribadi dan bertukar insight yang didapatkan selama belajar.

Penulis menyadari bahwa penyusunan skripsi ini jauh dari sempurna. Akhir kata penulis mohon maaf yang sebesar-besarnya apabila ada kekeliruan di dalam penulisan skripsi ini.

Lampung Selatan, .... 2025

**Abdurrahman Al-atsary**

## DAFTAR ISI

<b>HALAMAN JUDUL</b> . . . . .	i
<b>HALAMAN PENGESAHAN</b> . . . . .	ii
<b>HALAMAN PERNYATAAN ORISINALITAS</b> . . . . .	iii
<b>HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI</b> . .	iv
<b>ABSTRAK</b> . . . . .	v
<b>ABSTRACT</b> . . . . .	vi
<b>MOTTO</b> . . . . .	vii
<b>HALAMAN PERSEMBAHAN</b> . . . . .	viii
<b>KATA PENGANTAR</b> . . . . .	ix
<b>DAFTAR ISI</b> . . . . .	x
<b>DAFTAR GAMBAR</b> . . . . .	xii
<b>DAFTAR TABEL</b> . . . . .	xiii
<b>DAFTAR SINGKATAN</b> . . . . .	xv
<b>I Pendahuluan</b> . . . . .	1
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	3
1.3 Tujuan . . . . .	3
1.4 Batasan Masalah . . . . .	3
<b>II Tinjauan Pustaka</b> . . . . .	4
2.1 Penelitian Terdahulu . . . . .	4
2.2 Morfologi Mangrove . . . . .	5
2.3 CNN ( <i>Convolutional Neural Network</i> ) . . . . .	6
2.3.1 <i>Convolutional Layer</i> . . . . .	6
2.3.2 <i>Pooling Layer</i> . . . . .	7
2.3.3 <i>Fully Connected Layer</i> . . . . .	8
2.4 CBAM ( <i>Convolutional Block Attention Module</i> ) . . . . .	8
2.4.1 Mekanisme Atensi CBAM . . . . .	8
2.4.2 <i>Channel Attention Module</i> (CAM) . . . . .	10
2.4.3 <i>Spatial Attention Module</i> (SAM) . . . . .	11
2.5 Arsitektur Res4Net-CBAM . . . . .	12
2.6 Fungsi Aktivasi . . . . .	12

2.6.1	ReLU . . . . .	12
2.6.2	Leaky ReLU . . . . .	13
2.6.3	Sigmoid . . . . .	13
2.6.4	Softmax . . . . .	13
2.7	Optimizer . . . . .	14
2.7.1	Adam . . . . .	14
2.7.2	<i>SGD (Stochastic Gradient Descent)</i> . . . . .	14
2.8	<i>Learning Rate</i> . . . . .	14
2.9	<i>Loss Function</i> . . . . .	15
2.10	Interpretasi Grad-CAM . . . . .	15
2.11	<i>Confussion Matrix</i> . . . . .	16
2.12	Kurva ROC-AUC . . . . .	17
<b>III</b>	<b>Metode Penelitian</b> . . . . .	<b>18</b>
3.1	Deskripsi Data . . . . .	18
3.2	Metode . . . . .	18
3.2.1	Flowchart . . . . .	18
3.2.2	Pseudocode . . . . .	20
3.2.3	Augmentasi Gambar . . . . .	20
3.2.4	Lingkungan Pelatihan Model . . . . .	21
<b>IV</b>	<b>HASIL DAN PEMBAHASAN</b> . . . . .	<b>22</b>
4.1	Praproses Data . . . . .	22
4.1.1	Identifikasi Resolusi Gambar . . . . .	22
4.1.2	Pembagian Dataset . . . . .	22
4.2	Augmentasi Gambar . . . . .	23
4.3	Setup Arsitektur Res4Net-CBAM . . . . .	24
4.4	Pelatihan Model . . . . .	24
4.5	Evaluasi Model . . . . .	26
4.5.1	<i>Confusion Matrix</i> . . . . .	26
4.5.2	Kurva ROC/AUC . . . . .	30
4.6	Interpretasi Grad-CAM . . . . .	31
4.7	Pengujian Model <i>On-Site</i> . . . . .	33
<b>V</b>	<b>KESIMPULAN DAN SARAN</b> . . . . .	<b>34</b>
5.1	Kesimpulan . . . . .	34
5.2	Saran . . . . .	34
<b>DAFTAR PUSTAKA</b>	. . . . .	<b>35</b>
<b>LAMPIRAN</b>	. . . . .	<b>40</b>
<b>A</b>	<b>Surat Pengambilan Data</b> . . . . .	<b>41</b>

<b>B</b>	<b>Perhitungan Manual Modular CBAM . . . . .</b>	<b>42</b>
B.0.1	<i>Channel Attention Module (CAM) . . . . .</i>	42
B.0.2	<i>Spatial Attention Module (SAM) . . . . .</i>	45
B.0.3	Hasil Perhitungan CBAM (CAM + SAM) . . . . .	48
<b>C</b>	<b>Struktur Arsitektur Res4Net-CBAM . . . . .</b>	<b>50</b>

## DAFTAR GAMBAR

Gambar 2.1	Pohon dan daun <i>Avicennia alba</i> . . . . .	5
Gambar 2.2	Pohon dan daun <i>Rhizophora apiculata</i> . . . . .	5
Gambar 2.3	Pohon dan daun <i>Rhizophora stylosa</i> . . . . .	6
Gambar 2.4	Proses dalam Layer Konvolusi . . . . .	6
Gambar 2.5	Perbandingan <i>Max Pooling</i> dan <i>Average Pooling</i>	7
Gambar 2.6	Ilustrasi Proses <i>Flatten</i> dan <i>Fully Connected Layer</i> . . . . .	8
Gambar 2.7	Mekanisme <i>attention</i> di CBAM . . . . .	9
Gambar 2.8	Sub-proses <i>channel attention</i> CBAM . . . . .	10
Gambar 2.9	Sub-proses <i>spatial attention</i> CBAM . . . . .	11
Gambar 2.10	Res4Net Architecture . . . . .	12
Gambar 2.11	Res4Net-CBAM . . . . .	12
Gambar 2.12	Hasil Representasi Grad-CAM . . . . .	16
Gambar 2.13	<i>Confusion Matrix</i> . . . . .	16
Gambar 2.14	Kurva ROC dan AUC . . . . .	17
Gambar 3.1	Data daun yang dikumpulkan di Cuku NyiNyi .	18
Gambar 3.2	Flowchart Penelitian . . . . .	19
Gambar 4.1	Augmentasi Gambar . . . . .	23
Gambar 4.2	Pelatihan Model Res4Net-CBAM . . . . .	26
Gambar 4.3	Hasil <i>Confusion Matrix</i> - Res4Net-CBAM . . .	28
Gambar 4.4	Kurva ROC/AUC - Res4Net-CBAM . . . . .	31
Gambar 4.5	Hasil Grad-CAM Res4Net-CBAM . . . . .	32

## **DAFTAR TABEL**

Tabel 2.1	Rumus Evaluasi Model . . . . .	16
Tabel 3.1	Teknik Augmentasi pada Gambar untuk Lingkungan Mangrove . . . . .	21
Tabel 3.2	Spesifikasi Lingkungan Pelatihan pada Platform Kaggle . . . . .	21
Tabel 4.1	Evaluasi Confusion Matrix Model Res4Net-CBAM . . . . .	29
Tabel 4.2	Rekapitulasi Hasil Deteksi Daun Mangrove dengan Res4Net-CBAM . . . . .	33
Tabel C.1	Ringkasan Arsitektur Res4Net-CBAM . . . . .	50

## DAFTAR SINGKATAN

### **A**

AvgPool      *Average Pooling* (Pooling Rataan)

### **F**

F                Fitur Gambar (Keseluruhan)

f                Fitur hasil transformasi salah satu proses CBAM

$F^s$             Fitur Spasial (Proses SAM)

$F^c$             Fitur Kanal/*Channel* (Proses CAM)

### **M**

$M_c$             *Map Channel* (Pemetaan Channel)

$M_s$             *Map Spatial* (Pemetaan Spatial)

MLP            *Multi-layer Preceptron*

MaxPool      *Maximum Pooling* (Pooling Maksimal)

### **P**

px              piksel (satuan ukuran dalam gambar)

P                Probabilitas softmax (output dari hasil klasifikasi)

### **W**

$W_0$             Bobot layer *Shared MLP* laten dari *input* → *hidden* (CAM)

$W_1$             Bobot di layer *Shared MLP* dari *hidden* → *output* (CAM)

## BAB I

### Pendahuluan

#### 1.1 Latar Belakang

Indonesia dikenal sebagai negara maritim, dengan sekitar 2/3 atau 70% wilayahnya terdiri dari lautan yang berbatasan langsung dengan daratan[1]. Sumber daya alam yang melimpah baik dari laut maupun daratan menjadikan Indonesia memiliki banyak potensi. Salah satu daya tarik yang paling populer adalah wilayah pesisir pantai, yang menjadi destinasi wisata bagi wisatawan domestik maupun mancanegara [2]. Provinsi Lampung yang kurang lebih memiliki garis pantai sepanjang 1.105 km [3], berbatasan langsung dengan Pulau Jawa melalui Selat Sunda dan juga dengan Samudra Hindia. Lokasi strategis ini membuat Lampung menjadi salah satu destinasi wisata pesisir yang populer didatangi wisatawan setiap tahunnya[2]. Salah satu potensi pesisir di Indonesia terdapat di Desa Ekowisata Cuku Nyinyi, yang merupakan kawasan konservasi mangrove terbesar di Provinsi Lampung. Desa ini dikelola bersama oleh elemen masyarakat dan PT Bukit Asam Tbk., yang bekerja sama untuk melestarikan ekosistem mangrove serta mengembangkan destinasi wisata bernilai ekonomis dan berbasis lingkungan di Lampung[4].

Mangrove, atau yang dikenal sebagai tumbuhan bakau, merupakan tanaman khas yang hidup di kawasan pantai dan memiliki kemampuan adaptasi terhadap kadar garam yang tinggi [5][6]. Secara keseluruhan terdapat sekitar 20-25 genus dan 80-100 spesies mangrove di dunia [7][8]. Di Desa Ekowisata Cuku Nyinyi, hanya terdapat dua jenis mangrove yang ditemukan, yaitu dari genus *Avicennia* sp. dan *Rhizophora* sp.. Perbedaan antar spesies seringkali sulit dikenali oleh wisatawan, sehingga mendatangkan tantangan ketika jumlah pengunjung meningkat. Sementara hanya sebagian kecil masyarakat yang memiliki pengetahuan memadai tentang jenis-jenis mangrove. Padahal, dengan mengenali spesies mangrove secara langsung, wisatawan tidak hanya memperoleh pengetahuan baru, tetapi juga dapat memahami lebih dalam pentingnya pelestarian ekosistem pesisir serta peran aktif masyarakat lokal dalam menjaga kelestariannya. Oleh karena itu, dibutuhkan inovasi teknologi berbasis perangkat lunak (*software*) yang mengintegrasikan algoritma pemrosesan citra guna

membantu pengunjung dalam mengidentifikasi dan memperoleh informasi mengenai berbagai jenis mangrove yang terdapat di kawasan Ekowisata Cuku Nyinyi.

Perkembangan pemrosesan gambar berkembang pesat ketika ditemukan cara untuk mengekstraksi fitur dari gambar menjadi data numerik digital, yang kemudian dimodelkan secara matematis melalui metode CNN (*Convolutional Neural Network*) [9]. Kemajuan CNN mulai dikenal luas sejak tahun 2010, terutama melalui ajang ILSVRC (*ImageNet Large Scale Visual Recognition Challenge*), yang berlangsung hingga 2017. Perlombaan ini berfokus pada tiga aspek utama dalam arsitektur CNN: kedalaman lapisan (*depth*), lebar lapisan (*width*), dan kardinalitas (*cardinality*) [9]. Meskipun pendekatan ini menghasilkan model dengan tingkat galat rendah, kompleksitas komputasinya sangat tinggi, terutama karena banyaknya lapisan yang digunakan untuk mengekstraksi fitur. Hal ini menyebabkan waktu pemrosesan yang lama dan penggunaan daya komputasi yang besar [10]. Untuk mengatasi masalah tersebut, pada tahun-tahun berikutnya, muncul metodologi baru yang lebih efisien dalam mengekstraksi fitur penting dari gambar, salah satunya adalah CBAM (*Convolutional Block Attention Module*).

CBAM bekerja dengan menggunakan blok *attention* yang memfokuskan model pada bagian paling relevan dari gambar dan CBAM juga dikenal blok yang sangat ringan (*lightweight*) sehingga dapat mempercepat proses pemrosesan gambar dan meningkatkan akurasi [11]. Selain itu, CBAM terbukti efektif dalam meningkatkan performa model pada berbagai tugas pengenalan gambar termasuk dalam *general dataset* "PascalVOC". Hasil dari penelitian Cao et al. (2024) menunjukkan bahwa arsitekur CNN dengan penambahan blok CBAM mampu memberikan hasil akurasi 92.15% dan performansi yang terbaik dalam ekstraksi fitur berdasarkan visualisasi Grad-CAM ketika dibandingkan tanpa penambahan blok CBAM dalam mengelompokan spesies kalelawar tapal kuda (*horseshoe bats*)[12]. Berdasarkan eksperimen pada penelitian sebelumnya, penelitian kali ini akan berfokus pada penggunaan arsitektur Res4Net-CBAM untuk diimplementasikan dalam mengenali morfologi daun dan mengidentifikasi spesies mangrove.

## **1.2 Rumusan Masalah**

Bagaimana performa dan hasil ekstraksi fitur model Res4Net-CBAM dalam mengklasifikasikan setiap spesies daun mangrove?

## **1.3 Tujuan**

Evaluasi performa dan hasil ekstraksi fitur model Res4Net-CBAM dalam membedakan spesies mangrove berdasarkan morfologi daun.

## **1.4 Batasan Masalah**

Penelitian ini dibatasi pada data gambar daun yang berasal dari pohon mangrove yang ada di lingkup desa Ekowisata Cuku NyiNyi.

## BAB II

### Tinjauan Pustaka

#### 2.1 Penelitian Terdahulu

Judul Penelitian	Peneliti, Tahun	Data, Metode	Hasil Penelitian
Identification of Mangrove Tree Species Using Deep Learning Method	Paranita Asnur, Rifki Kosasih, Sarifuddin Madenda, Dewi A. Rahayu, 2023 [13]	Dataset citra daun dari 4 spesies mangrove ( <i>Avicenia marina</i> , <i>A. officinalis</i> , <i>Rhizophora apiculata</i> , <i>Sonneratia caseolaris</i> ), menggunakan CNN dengan 4 lapisan konvolusi + ReLU + max-pooling 2x2, dua <i>fully connected layers</i> , batch size 32, input 128x128 piksel	Akurasi model pada data uji mencapai 97,50%, sedangkan pada data validasi sebesar 81,25%
Deep Learning-Based Identification of Mangrove Species Using Leaf Images and Transfer Learning	Arif Nugraha, Sari Hartati, Puji Santosa, Tommy Hidayat, 2025 [14]	Citra daun dari 11 spesies mangrove dikumpulkan di pulau Bali, menggunakan deep learning dengan metode transfer learning berbasis MobileNetV2	Model mampu mengidentifikasi spesies mangrove dengan tingkat akurasi 96.49%

## 2.2 Morfologi Mangrove

Hutan mangrove merupakan ekosistem unik yang tumbuh di zona pasang surut pesisir tropis dan subtropis [15][6]. Tiga spesies mangrove yang umum ditemukan di kawasan Ekowisata Cuku Nyinyi adalah *Avicennia alba*, *Rhizophora apiculata*, dan *Rhizophora stylosa*. Masing-masing spesies ini memiliki karakteristik morfologi daun yang khas yang dapat dibedakan dari satu dengan lainnya. *Avicennia alba*, yang dikenal juga sebagai api-api putih, memiliki daun berbentuk elips hingga lanset dengan ujung yang meruncing. Daunnya berwarna hijau cerah di bagian atas dan putih keperakan di bagian bawah, serta terkadang tampak kekuningan saat masih muda, dengan panjang sekitar 7–10 cm dan lebar 3–5 cm [15][16][8].



Gambar 2.1 Pohon dan daun *Avicennia alba*

*Rhizophora apiculata*, atau bakau minyak, memiliki daun yang lebih besar dibandingkan *A. alba*. Daunnya berbentuk elips dengan ujung meruncing dan memiliki panjang sekitar 7-19 cm dan lebar 3-8 cm. Daun *R. apiculata* berwarna hijau tua mengkilap di bagian atas dan hijau kekuningan di bagian bawah. Salah satu ciri khas daun *R. apiculata* adalah adanya duri kecil (*mucro*) di ujung daun[8][15][17].



Gambar 2.2 Pohon dan daun *Rhizophora apiculata*

*Rhizophora stylosa*, yang sering disebut bakau kurap, memiliki morfologi daun yang mirip dengan *R. apiculata*, namun dengan beberapa perbedaan penting. Daun *R. stylosa* berbentuk *obovate* hingga elips dengan ujung yang tumpul atau sedikit meruncing. Ukurannya sedikit lebih kecil dibandingkan *R. apiculata*, dengan

panjang 6-12 cm dan lebar 3-7 cm. Daun *R. stylosa* memiliki warna hijau gelap di bagian atas dan hijau kekuningan di bagian bawah[15][17].



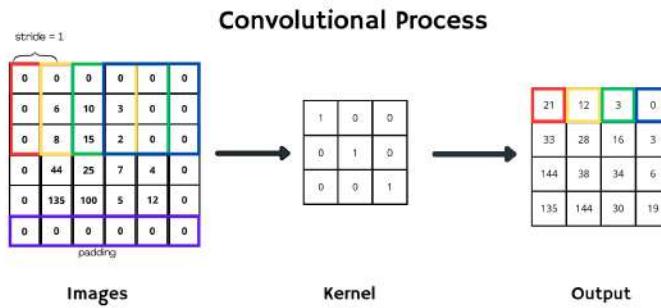
**Gambar 2.3** Pohon dan daun *Rhizophora stylosa*

### 2.3 CNN (*Convolutional Neural Network*)

*Convolutional Neural Networks* (CNN) adalah jaringan saraf tiruan yang terinspirasi oleh sistem saraf biologis, khususnya untuk pengenalan gambar. CNN terdiri dari lapisan input (*input layer*), lapisan tersembunyi (*hidden layer*), dan lapisan penyatuhan atau hasil (*output layer*) untuk mengekstrak fitur penting dari data visual dan menghasilkan suatu prediksi[18][19].

### 2.3.1 *Convolutional Layer*

Lapisan konvolusi dalam CNN berfungsi untuk mengekstraksi fitur dari data berdimensi, seperti gambar, video dan dalam format bentuk yang serupa. Di lapisan ini, filter atau kernel yang bisa dipelajari (*learnable*) akan bergerak melintasi gambar dan melakukan operasi konvolusi. Hasilnya adalah peta fitur 2D yang menangkap informasi penting, seperti tepi dan pola[18][20].



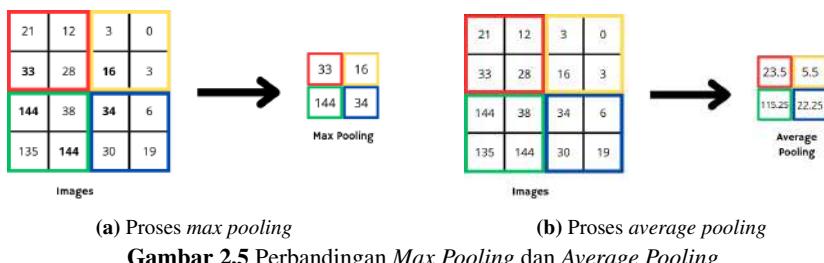
**Gambar 2.4** Proses dalam Layer Konvolusi

Proses konvolusi ditunjukkan pada Gambar 2.4, diawali dengan citra

yang telah dikonversi menjadi representasi numerik dalam bentuk *digital number* dengan disesuaikan dengan rentang warna dari 0-255, kemudian dikenai operasi perhitungan menggunakan sebuah kernel, yaitu matriks bobot kecil yang juga disebut filter, yang berfungsi untuk mengekstraksi fitur tertentu seperti tepi, tekstur, atau pola dari citra[21]. Kernel ini bergerak melintasi citra input berdasarkan ukuran langkah (*stride*) dan pemberian *padding* sebagai penyesuaian ukuran. Pada setiap perpindahan kernel, dilakukan perkalian elemen-per-elemen (*element-wise multiplication*) antara nilai pada kernel terhadap nilai piksel citra, kemudian dijumlahkan untuk menghasilkan satu nilai fitur[21]. Hasil dari setiap pergerakan kernel membentuk peta fitur (*feature map*) yang mewakili informasi penting dari citra input menjadi output.

### 2.3.2 Pooling Layer

*Pooling* merupakan proses mengurangi ukuran spasial (lebar dan tinggi) peta fitur, sehingga mengurangi jumlah parameter dan komputasi yang membantu mempercepat pelatihan[18]. Dua jenis *pooling* yang umum digunakan adalah *max pooling* dan *average pooling*.



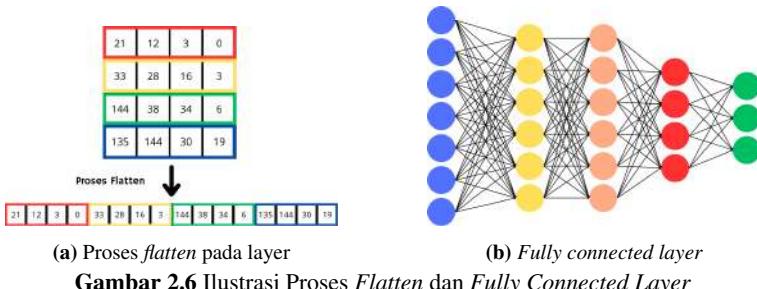
Gambar 2.5 Perbandingan Max Pooling dan Average Pooling

Perbedaan proses antara *average pooling* dan *max pooling* ditunjukkan pada Gambar 2.5, yang memperlihatkan perbedaan hasil perhitungan dari setiap wilayah atau area yang dilalui oleh filter. Pada *average pooling*, nilai yang dihasilkan merupakan rata-rata dari seluruh piksel dalam satu area filter, sehingga informasi yang diperoleh bersifat menyeluruh dan lebih halus. Sebaliknya, pada *max pooling*, nilai maksimum dari area tersebut yang diambil, sehingga hanya fitur paling menonjol yang dipertahankan. Dengan demikian, *max pooling* cenderung lebih efektif dalam mempertahankan fitur-fitur penting seperti tepi atau pola dominan, sementara *average pooling* memberikan

representasi yang lebih merata namun kurang menonjolkan detail tertentu[18].

### 2.3.3 Fully Connected Layer

*Fully Connected Layer* (FCL) adalah lapisan dalam jaringan saraf di mana setiap neuron pada lapisan ini terhubung dengan semua neuron di lapisan sebelumnya. *Fully Connected* memungkinkan jaringan untuk mempelajari pola dan representasi yang lebih kompleks dari data. Sebelum FCL, biasanya digunakan lapisan *flatten*, yang berfungsi mengubah data multidimensi menjadi vektor satu dimensi, agar bisa diproses oleh FCL [22][21].



(a) Proses flatten pada layer

(b) Fully connected layer

Gambar 2.6 Ilustrasi Proses Flatten dan Fully Connected Layer

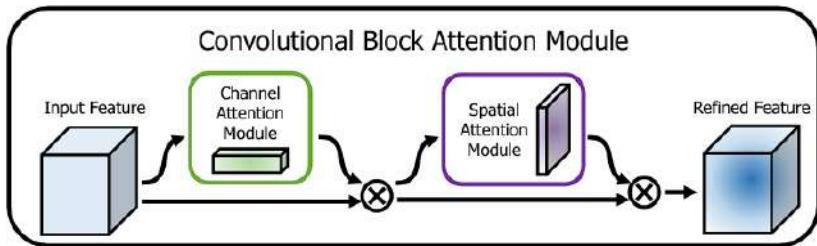
## 2.4 CBAM (*Convolutional Block Attention Module*)

*Convolutional Block Attention Module* (CBAM) adalah mekanisme blok perhatian (*attention*) yang dirancang ringan *lightweight*, adaptif dan efektif untuk meningkatkan kinerja jaringan saraf konvolusional (CNN) dengan memperbaiki peta fitur melalui perhatian saluran (*channel attention*) dan spasial (*spatial attention*) secara berurutan. CBAM bekerja dengan pertama-tama menghasilkan peta *channel attention* yang berfokus pada "apa" fitur yang penting, diikuti oleh peta *spatial attention* yang menekankan "di mana" harus fokus dalam peta fitur[11].

### 2.4.1 Mekanisme Atensi CBAM

Proses mekanisme yang menjadi ciri khas dari CBAM dalam proses menemukan fitur atensi merupakan proses kombinasi antara *channel attention module* yang dipusatkan untuk menemukan pada bagian dari setiap kanal manakah yang akan mendapatkan atensi dan *spatial attention module* yang bertujuan untuk menemukan di area manakah

dari suatu gambar yang akan mendapatkan atensi akan tetapi kedua atensi tersebut saling terhubung satu dengan lainnya sebagai proses berurutan dengan mekanisme hasil dari pemetaan secara kanal (*channel attention module*) dan dipetakan kembali secara spasial (*spatial attention module*) sehingga didapatkan fitur yang menjadi pusat ekstraksi model secara keseluruhan[11].



Gambar 2.7 Mekanisme *attention* di CBAM

Berdasarkan ilustrasi diatas, mekanisme dari CBAM secara keseluruhan dapat dijelaskan proses matematisnya. Proses pertama  $\mathbf{F} \in \mathbb{R}^{C \times H \times W}$  sebagai input yang merupakan representasi dari fitur yang dimiliki oleh gambar diwakili oleh  $\mathbf{F}$  yang didalamnya tersusun dari elemen bilangan real yang berupa matriks dengan ketentuan  $C$  (*channel*) merupakan jumlah dari *channel*,  $H$  (*height*) untuk tinggi gambar dan  $W$  (*width*) merupakan lebar dari gambar yang diukur berdasarkan inputan yang sudah ditentukan. CBAM secara berurutan menginferensikan peta atensi kanal (*channel attention*) untuk menghasilkan peta fitur berupa 1D  $\mathbf{M}_C \in \mathbb{R}^{C \times 1 \times 1}$ , dapat diartikan sebagai  $M_c$  (*map channel*) yang tersusun dari matriks dengan ukuran  $1 \times 1$  dari setiap *channel* yang hasilnya akan di *broadcast* kedalam setiap *channel*. Sementara itu hasil dari peta atensi spasial (*spatial attention module*) yang berupa 2D  $\mathbf{M}_S \in \mathbb{R}^{1 \times H \times W}$  yang tersusun dari  $M_c$  (*map channel*) dengan 1 *kanal* dari matriks berukuran  $W$  (*width*)  $\times$   $H$  (*height*) yang disesuaikan inputan hasil dari *channel attention*. Berdasarkan yang diilustrasikan pada Gambar 2.7. Proses matematis *attention* secara keseluruhan dapat kalkulasikan sebagai:

$$\mathbf{F}' = \mathbf{M}_C(\mathbf{F}) \otimes \mathbf{F}, \quad (2.1)$$

$$\mathbf{F}'' = \mathbf{M}_S(\mathbf{F}') \otimes \mathbf{F}', \quad (2.2)$$

di mana  $\otimes$  menunjukkan perkalian elemen-per-elemen (*element wise*

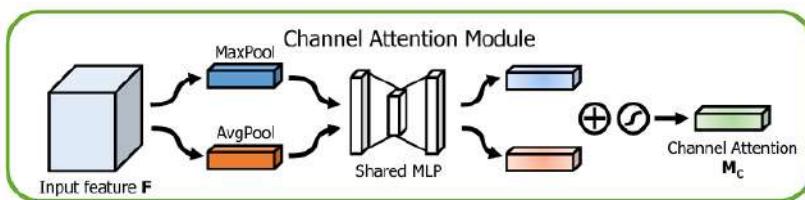
*multiplication*). Selama proses perkalian, nilai *attention* pada sub-proses *channel attention* akan dikalikan dengan hasil dari  $M_s$  yang merupakan hasil dari kalkulasi dengan fitur inputan  $\mathbf{F}$  kemudian dilanjutkan dengan nilai *channel attention* ( $\mathbf{F}'$ ) jalankan sepanjang dimensi spasial. Hasil akhirnya menghasilkan  $\mathbf{F}''$  yang merupakan hasil keluaran akhir (*output CBAM*) sebagai representasi fitur yang telah disempurnakan dan mendapatkan atensi (*refined feature*)[11].

#### 2.4.2 Channel Attention Module (CAM)

Sub-proses *channel attention* prosesnya diawali dengan menggabungkan fitur *average pooling* dan *max pooling* dari setiap *channel* dari peta fitur ( $\mathbf{F}$ ) untuk menghasilkan dua deskriptor yang dinotasikan sebagai ( $\mathbf{F}_{\text{avg}}^c$  dan  $\mathbf{F}_{\text{max}}^c$ ). Keduanya diproses melalui *shared MLP* yang merupakan *multi-layer perceptron* (MLP) dengan 1 *hidden layer*, proses didalamnya dari 2 deskriptor yang sudah diekstrak dari setiap kanal akan dikenakan sebuah bobot ( $\mathbf{W}$ ) dan bias ( $\mathbf{b}$ ) yang sama dan menghasilkan 2 deskriptor yang nantinya akan dilakukan *element-wise summation* atau penambahan secara elemen-per-elemen dan proses selanjutnya dinormalisasi dengan fungsi sigmoid ( $\sigma$ ) untuk membentuk peta *channel attention*[11]. Pada proses CAM juga terdapat mekanisme strategi pengurangan parameter yang masuk kedalam MLP dengan tujuan agar mengurangi parameter agar tidak besar (*overhead parameter*) yang dinotasikan sebagai  $r$  (*reduction ratio*) yang ditentukan diawal[11].

$$\mathbf{M}_c(\mathbf{F}) = \sigma (\text{MLP}(\text{AvgPool}(\mathbf{F})) + \text{MLP}(\text{MaxPool}(\mathbf{F}))) \quad (2.3)$$

$$= \sigma \left( \mathbf{W}_1 \left( \mathbf{W}_0 \mathbf{F}_{\text{avg}}^c \right) + \mathbf{W}_1 \left( \mathbf{W}_0 \mathbf{F}_{\text{max}}^c \right) \right) \quad (2.4)$$



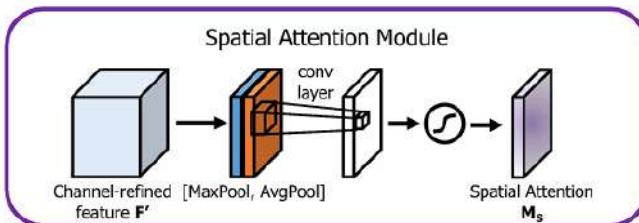
Gambar 2.8 Sub-proses *channel attention* CBAM

### 2.4.3 Spatial Attention Module (SAM)

Sub-proses *spatial attention* pada CBAM berfungsi untuk menyoroti bagian-bagian penting dari peta fitur dengan menghitung peta *spatial attention*,  $M_s(F)$  dimaknai sebagai (*map spatial*) yang menunjukkan lokasi mana yang perlu ditekankan atau diabaikan. Prosesnya awalnya melibatkan pengambilan 2 deskriptor juga yakni *channel-wise max pooling* dan *channel-wise average pooling* yang prosesnya sendiri mengambil setiap indeks dari setiap *channel* yang kemudian dilakukan mekanisme statistik dalam pengambilan nilai maksimal dan nilai rataan yang melibatkan seluruh indeks dari setiap *channel*. Proses selanjutnya setelah mendapatkan kedua deskriptor tersebut dalam bentuk 2 *channel* yang berupa matriks, selanjutnya dilakukan penggabungan dengan metode *concat* antar hasil dari *channel-wise max pooling* dan *channel-wise average pooling* yang kemudian dikonvolusi dengan matriks dengan ukuran  $7 \times 7$  sebelum dilakukan konvolusi harus melalui proses *padding* sehingga bentuk input dan output selaras dalam satu ukuran, matriks  $7 \times 7$  akan berjalan dari setiap *channel-wise* lalu proses dilanjutkan dengan penggabungan (*element-wise summation*) dari hasil konvolusi yang dilakukan sebelumnya dan menghasilkan pemetaan fitur dengan 1 *channel* yang nantinya dilakukan proses normalisasi dengan fungsi aktivasi sigmoid dan menghasilkan peta atensi spasial (*spatial attention*)[11]. Notasi persamaan untuk mendapatkan pemetaan spasial atau *spatial attention module* dituliskan sebagai:

$$M_s(F) = \sigma(f^{7 \times 7}([\text{AvgPool}(F); \text{MaxPool}(F)])) \quad (2.5)$$

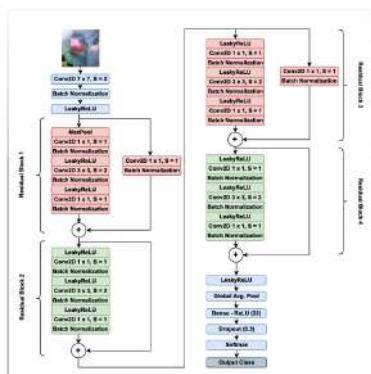
$$= \sigma(f^{7 \times 7}([F_{\text{avg}}^s; F_{\text{max}}^s])) \quad (2.6)$$



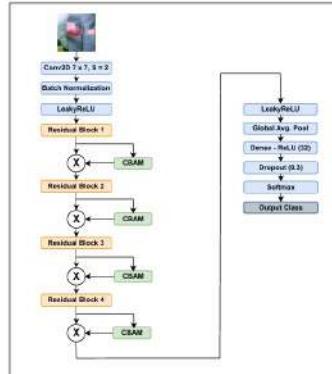
Gambar 2.9 Sub-proses *spatial attention* CBAM

## 2.5 Arsitektur Res4Net-CBAM

Arsitektur Res4Net terdiri dari empat blok residual yang diawali dengan layer konvolusi berukuran  $7 \times 7$  dengan stride 2, diikuti oleh *BatchNormalization* dan LeakyReLU sebagai transisi untuk mengontrol input dan mempercepat pelatihan [23][24]. Sebelum masuk ke blok residual pertama, ditambahkan *max pooling* untuk mereduksi dimensi spasial. Setiap blok residual dibangun dari susunan layer konvolusi, *BatchNormalization*, dan LeakyReLU, dengan tambahan *skip connection* khusus pada blok ke-1 dan ke-3. *Skip connection* merupakan mekanisme penggabungan gradien yang didapatkan layer sebelumnya dengan gradient yang didapatkan setelahnya[25]. Kemudian, di setiap akhir blok residual, ditambahkan CBAM untuk menghasilkan *attention map* terhadap fitur yang dihasilkan. Setelah seluruh blok residual, output distandarisasi dengan LeakyReLU sebelum masuk ke *Global Average Pooling* (GAP), dilanjutkan dengan *Fully Connected Layer* dan *Dropout*, lalu diakhiri oleh fungsi aktivasi softmax untuk proses klasifikasi. Ilustrasi lengkap arsitektur Res4Net-CBAM ditunjukkan pada Gambar 2.10 dan 2.11.



Gambar 2.10 Res4Net Architecture



Gambar 2.11 Res4Net-CBAM

## 2.6 Fungsi Aktivasi

### 2.6.1 ReLU

Fungsi aktivasi *Rectified Linear Unit* (ReLU) merupakan salah satu fungsi aktivasi paling umum digunakan dalam jaringan saraf tiruan (*neural network*) [26]. ReLU menghasilkan nilai nol untuk input negatif dan mengembalikan nilai input itu sendiri untuk input positif.

Kelemahan utama ReLU adalah potensi kosongnya nilai neuron saat nilai input terus-menerus menghasilkan negatif [27]. Secara matematis, fungsi ReLU dirumuskan sebagai:

$$f(x) = \max(0, x) \quad (2.7)$$

### 2.6.2 Leaky ReLU

Leaky ReLU merupakan varian dari ReLU yang memperkenalkan gradien kecil untuk nilai negatif guna mengatasi masalah kosongnya nilai neuron saat input negatif. Fungsi ini memperkenalkan output bernilai  $\alpha x$  (*slope*) untuk input negatif dengan  $\alpha$  biasanya kecil, contohnya 0.01 [28]. Hal ini menjaga agar neuron tetap dapat belajar meskipun menerima nilai negatif. Fungsi ini dirumuskan sebagai:

$$f(x) = \begin{cases} x, & \text{jika } x \geq 0 \\ \alpha x, & \text{jika } x < 0 \end{cases} \quad (2.8)$$

### 2.6.3 Sigmoid

Fungsi sigmoid banyak digunakan pada lapisan output untuk klasifikasi biner karena mengonversi input menjadi rentang nilai antara 0 dan 1, menyerupai probabilitas [29]. Namun, fungsi ini cenderung mengalami saturasi untuk nilai input yang besar dan menghasilkan gradien yang sangat kecil, yang menyulitkan pelatihan jaringan yang terlalu dalam [29]. Rumus matematis sigmoid adalah:

$$f(x) = \frac{1}{1 + e^{-x}} \quad (2.9)$$

### 2.6.4 Softmax

Fungsi *softmax* digunakan pada lapisan output untuk klasifikasi multikelas karena mengubah vektor menjadi distribusi probabilitas yang jumlahnya 1 [30]. Fungsi ini menekankan nilai terbesar dalam vektor, sehingga cocok untuk mengambil keputusan berdasarkan probabilitas tertinggi. Fungsi ini dirumuskan sebagai:

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^K e^{x_j}} \quad (2.10)$$

## 2.7 Optimizer

### 2.7.1 Adam

Adam (*Adaptive Moment Estimation*) mengombinasikan keuntungan dari *Momentum* dan RMSProp dengan memanfaatkan rata-rata bergerak dari gradien dan kuadrat gradien [31]. Adam menyesuaikan laju pembelajaran untuk setiap parameter secara adaptif, menjadikannya optimizer yang sangat efisien dan banyak digunakan di dalam pelatihan model *deep learning*. Langkah-langkah perhitungan Adam adalah sebagai berikut:

$$g_t = \nabla_{\theta} J(\theta_{t-1}) \quad (2.11)$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (2.12)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (2.13)$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (2.14)$$

$$\theta_t = \theta_{t-1} - \frac{\eta \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \quad (2.15)$$

### 2.7.2 SGD (*Stochastic Gradient Descent*)

*Stochastic Gradient Descent* (SGD) memperbarui parameter model menggunakan sebagian kecil data (*mini batch*) atau satu contoh secara acak, sehingga proses pelatihan menjadi lebih cepat dan efisien. Pendekatan ini sangat sesuai untuk *online learning*, di mana data diterima secara bertahap. Meskipun prosesnya dapat mengalami fluktuasi, penyesuaian *learning rate* yang tepat dapat membantu SGD mencapai konvergensi yang stabil[32]. Rumus dasarnya adalah:

$$\theta_t = \theta_{t-1} - \eta \nabla_{\theta} J(\theta_{t-1}; x^{(i)}, y^{(i)}) \quad (2.16)$$

## 2.8 Learning Rate

*Learning rate* merupakan salah satu hyperparameter yang esensial dalam proses pelatihan model pembelajaran mesin karena berperan dalam mengatur laju pembaruan parameter model terhadap fungsi kerugian (*loss function*). Pemilihan serta penyesuaian *learning rate* yang tepat sangat menentukan efisiensi proses pelatihan dan stabilitas konvergensi, sehingga model dapat mencapai performa prediktif yang optimal[21].

## 2.9 Loss Function

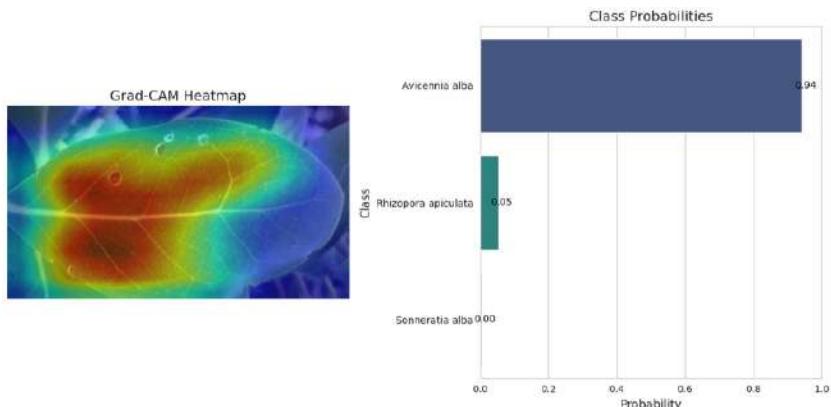
*Categorical Cross-Entropy* (CCE) merupakan salah satu fungsi loss yang paling umum diterapkan dalam permasalahan klasifikasi multi-kelas, khususnya ketika label target direpresentasikan dalam bentuk *one-hot encoding* dan keluaran model berupa distribusi probabilitas yang dihasilkan melalui fungsi aktivasi seperti *softmax*. Fungsi ini secara matematis mengukur tingkat dissimilaritas antara distribusi probabilitas prediksi model  $\hat{y}$  dan distribusi target sebenarnya  $y$  yang dirumuskan sebagai berikut:

$$L = - \sum_{i=1}^C y_i \log(\hat{y}_i),$$

dengan  $C$  menyatakan jumlah total kelas,  $y_i$  merupakan nilai aktual untuk kelas ke- $i$ , dan  $\hat{y}_i$  merupakan probabilitas yang diprediksi oleh model untuk kelas tersebut. Nilai *loss* akan semakin tinggi apabila probabilitas yang diberikan pada kelas yang benar semakin kecil, menjadikan CCE sangat sensitif terhadap kesalahan prediksi dengan probabilitas rendah. CCE secara luas diadopsi dalam pengembangan model-model *deep learning* karena kemampuannya dalam merepresentasikan kesalahan klasifikasi secara probabilistik dengan presisi yang tinggi [22].

## 2.10 Interpretasi Grad-CAM

Grad-CAM (*Gradient-weighted Class Activation Mapping*) merupakan sebuah metode visualisasi dan interpretasi yang digunakan untuk mengidentifikasi area penting pada citra yang menjadi fokus perhatian model CNN dalam proses pengambilan keputusan. Teknik ini memanfaatkan informasi gradien yang diperoleh selama proses *backpropagation* pada lapisan konvolusi terakhir guna mengevaluasi kontribusi setiap piksel terhadap prediksi kelas tertentu. Grad-CAM sangat relevan untuk diterapkan dalam konteks klasifikasi citra, karena memungkinkan identifikasi area yang signifikan atau menjadi fokus model dalam menentukan prediksi. Mekanisme kerja Grad-CAM dimulai dengan menghitung gradien dari skor kelas target terhadap aktivasi fitur di lapisan konvolusi akhir, kemudian diakumulasikan untuk membentuk peta panas (*heatmap*) yang merepresentasikan wilayah dengan pengaruh paling besar terhadap keputusan klasifikasi model [33].



**Gambar 2.12** Hasil Representasi Grad-CAM

## 2.11 Confusion Matrix

*Confusion matrix* merupakan metrik evaluasi penting dalam mengukur performa model klasifikasi, dengan membandingkan hasil prediksi terhadap label aktual melalui empat kategori: *true positive* (TP), *false positive* (FP), *true negative* (TN), dan *false negative* (FN) [34]. Metrik ini menjadi dasar perhitungan indikator lain seperti *accuracy*, *precision*, *recall*, dan *F1-score*, serta memudahkan analisis keberhasilan dan kesalahan model. Perhitungan masing-masing metrik serta ilustrasi *confusion matrix* disajikan pada Gambar 2.13 dan tabel 2.1.

		Prediksi		
		A	B	C
Aktual	A	TP	FP	FP
	B	FN	TP	FP
	C	FN	FN	TP

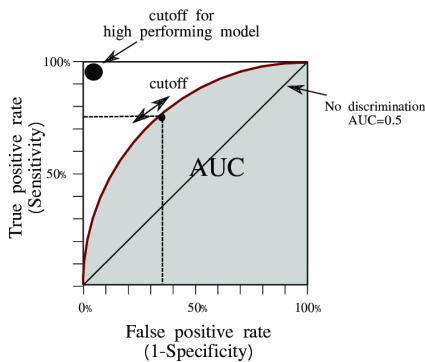
**Gambar 2.13** Confusion Matrix

**Tabel 2.1** Rumus Evaluasi Model

Metric	Formula
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F1-Score	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

## 2.12 Kurva ROC-AUC

ROC (*Receiver Operating Characteristic*) dan AUC (*Area Under the Curve*) merupakan metrik evaluasi yang digunakan untuk mengukur kinerja model klasifikasi. Kurva ROC menggambarkan hubungan antara *True Positive Rate* (TPR), yang juga dikenal sebagai *recall* atau *sensitivity*, dengan *False Positive Rate* (FPR) pada berbagai nilai ambang keputusan (*threshold*) yang digunakan oleh model. Secara umum, TPR didefinisikan sebagai proporsi data positif yang berhasil diklasifikasikan dengan benar, sementara FPR merupakan proporsi data negatif yang secara keliru diklasifikasikan sebagai positif[35]. Kurva ROC memiliki domain dan rentang dalam rentang [0, 1], di mana semakin dekat nilai TPR terhadap 1 dengan FPR yang rendah menunjukkan performa model yang semakin baik. Nilai AUC mengkuantifikasi luas area di bawah kurva ROC, yang secara numerik mencerminkan probabilitas bahwa model akan memberikan skor prediksi yang lebih tinggi terhadap sampel positif dibandingkan sampel negatif secara acak[36].



$$\text{TPR (Sensitivity)} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{FPR (1 - Specificity)} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

Gambar 2.14 Kurva ROC dan AUC

## BAB III

### Metode Penelitian

#### 3.1 Deskripsi Data

Data yang digunakan dalam penelitian ini merupakan data primer yang diambil dari Desa Ekowisata Cuku NyiNyi yang terletak di Kelurahan Sidodadi, Kecamatan Teluk Pandan, Kabupaten Pesawaran, Lampung. Data ini mencakup gambar daun dari 3 spesies mangrove, seperti pada sampel gambar berikut :



Gambar 3.1 Data daun yang dikumpulkan di Cuku NyiNyi

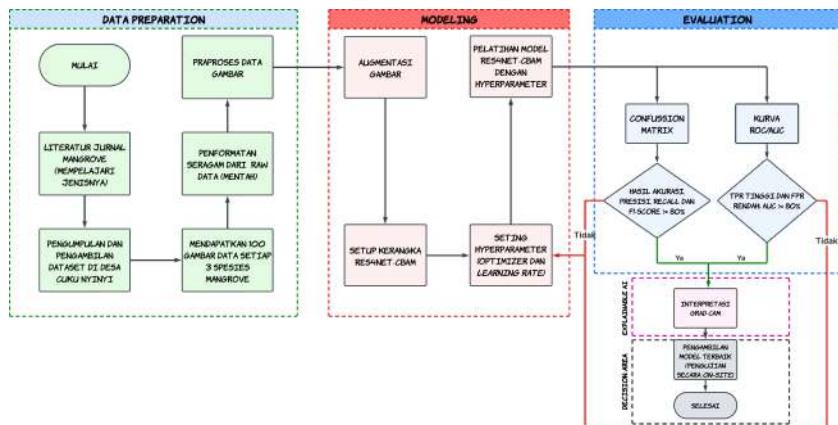
Pengambilan data gambar dilakukan menggunakan kamera smartphone Vivo Y35 dan Infinix Note 40, yang masing-masing memiliki resolusi kamera sebesar 50 megapiksel. Untuk setiap spesies, dikumpulkan sebanyak 100 gambar dengan metode pengambilan pada jarak sekitar 25 cm dari kamera. Seluruh gambar yang diperoleh disimpan dalam format .jpg dan dikelompokkan ke dalam folder berdasarkan spesiesnya.

#### 3.2 Metode

##### 3.2.1 Flowchart

Pada penelitian ini dapat dirancang sebuah diagram alir proses penelitian yang akan dilakukan. Alur penelitian klasifikasi mangrove menggunakan model Res4Net-CBAM dijelaskan dalam Gambar 3.2. Tahap pertama, yaitu *Data Preparation*, dimulai dengan studi literatur mengenai mangrove dan pengumpulan dataset dari Desa Cuku Nyinyi, dilanjutkan dengan tahap *Pre-processing* Data Gambar. Pada tahap ini, dilakukan pemformatan seragam serta pengambilan 100 gambar untuk masing-masing dari tiga spesies mangrove. Setelah itu, pada tahap *modeling*, dilakukan augmentasi gambar, setup kerangka model

Res4Net-CBAM disertai pengaturan *hyperparameter* seperti *learning rate* dan *optimizer* sebelum melatih model. Tahap selanjutnya, yaitu *Evaluation*, mengevaluasi performa model menggunakan metrik dalam *confusion matrix*, termasuk *accuracy*, *recall*, dan *precision* dengan ambang batas minimal 80%, serta analisis ROC/AUC untuk memastikan *True Positive Rate* (TPR) tinggi dan *False Positive Rate* (FPR) rendah. Jika hasil evaluasi tidak memenuhi kriteria, dilakukan penyetelan *learning rate* dalam optimizer. Jika model memenuhi kriteria evaluasi, tahap *Explainable AI* dilakukan untuk interpretasi menggunakan Grad-CAM, sebelum akhirnya model terbaik dipilih pada tahap *Decision Area*.



Gambar 3.2 Flowchart Penelitian

### 3.2.2 Pseudocode

---

#### Algoritma 1 Convolutional Block Attention Module (CBAM)

---

```
1: Input: Feature Map  $F \in \mathbb{R}^{C \times H \times W}$ 
2: Output: Enhanced Feature Map  $F'' \in \mathbb{R}^{C \times H \times W}$ 
3: function CBAM( $F$ )
4:    $M_c \leftarrow \text{ChannelAttention}(F)$             $\triangleright$  Calculate channel attention map
5:    $F' \leftarrow M_c \otimes F$                        $\triangleright$  Apply channel attention
6:    $M_s \leftarrow \text{SpatialAttention}(F')$          $\triangleright$  Calculate spatial attention map
7:    $F'' \leftarrow M_s \otimes F'$                     $\triangleright$  Apply spatial attention
8:   return  $F''$                                  $\triangleright$  Return enhanced feature map
9: end function
10: function CHANNELATTENTION( $F$ )
11:   AvgPool  $\leftarrow \text{GlobalAveragePooling}(F)$      $\triangleright$  Global average pooling
12:   MaxPool  $\leftarrow \text{GlobalMaxPooling}(F)$         $\triangleright$  Global max pooling
13:    $MLP(\text{AvgPool}) \leftarrow \sigma(W_1(\delta(W_0(\text{AvgPool}))))$    $\triangleright$  MLP with activation
    for AvgPool
14:    $MLP(\text{MaxPool}) \leftarrow \sigma(W_1(\delta(W_0(\text{MaxPool}))))$    $\triangleright$  MLP with activation
    for MaxPool
15:    $M_c \leftarrow \sigma(MLP(\text{AvgPool}) + MLP(\text{MaxPool}))$      $\triangleright$  Combine and apply
    sigmoid
16:   return  $M_c$                                  $\triangleright$  Return channel attention map
17: end function
18: function SPATIALATTENTION( $F'$ )
19:   AvgPool  $\leftarrow \text{AveragePoolingAlongChannels}(F')$      $\triangleright$  Average along
    channels
20:   MaxPool  $\leftarrow \text{MaxPoolingAlongChannels}(F')$      $\triangleright$  Max along channels
21:   Concat  $\leftarrow \text{Concatenate}(\text{AvgPool}, \text{MaxPool})$      $\triangleright$  Concatenate pools
22:    $M_s \leftarrow \sigma(\text{Conv}(\text{Concat}))$            $\triangleright$  Apply 1x1 convolution and sigmoid
23:   return  $M_s$                                  $\triangleright$  Return spatial attention map
24: end function
```

---

### 3.2.3 Augmentasi Gambar

Proses augmentasi gambar dilakukan sebagai bagian dari tahap pre-processing untuk meningkatkan keragaman data pelatihan dan mengurangi risiko *overfitting* pada model. Dalam penelitian ini, augmentasi dilakukan menggunakan library Albumentations[37]. Penerapan variasi dari augmentasi bertujuan untuk mensimulasikan kondisi lingkungan nyata, sehingga model dapat belajar dari citra dengan karakteristik visual yang beragam. Rincian teknik augmentasi yang diterapkan beserta tujuannya dalam penelitian ini disajikan pada Tabel 3.1.

**Tabel 3.1** Teknik Augmentasi pada Gambar untuk Lingkungan Mangrove

Teknik Augmentasi	Tujuan
<i>Gaussian Blur</i>	Menambahkan efek buram pada gambar untuk mensimulasikan data dari kamera berkualitas rendah atau fokus yang kurang tajam.
<i>Rotate</i>	Memutar gambar dengan sudut tertentu pada gambar daun.
<i>Horizontal Flip</i>	Membalik gambar secara horizontal untuk meningkatkan variasi orientasi gambar.
<i>Vertical Flip</i>	Membalik gambar secara vertikal untuk meningkatkan variasi orientasi gambar.
<i>Bright and Contrast</i>	Memberikan efek penyetelan dari kecerahan dan kontras dari suatu gambar.
<i>Hue and Saturation</i>	Memberikan efek kejemuhan gambar dan saturasi dari gambar.

### 3.2.4 Lingkungan Pelatihan Model

Pelatihan model dilakukan menggunakan platform Kaggle sebagai lingkungan komputasi berbasis cloud yang menyediakan sumber daya GPU. Pemanfaatan GPU dalam proses pelatihan bertujuan untuk mempercepat konvergensi model dan memungkinkan eksplorasi berbagai kombinasi *hyperparameter* dalam waktu yang lebih singkat. Spesifikasi detail lingkungan pelatihan disajikan pada Tabel 3.2.

**Tabel 3.2** Spesifikasi Lingkungan Pelatihan pada Platform Kaggle

Spesifikasi	Keterangan
Platform	Kaggle Notebook (Cloud-based)
GPU	NVIDIA Tesla P100
Kapasitas GPU	16 GB VRAM
CPU	Intel Xeon (2 vCPU)
RAM	13 GB
Library Utama	PyTorch, NumPy, Matplotlib, Albumentations, Scikit-learn
Jumlah <i>Epoch</i> Pelatihan	100
Batch Size	32
Metode Simpan Model	<i>ModelCheckpoint</i>

## BAB IV

### HASIL DAN PEMBAHASAN

#### 4.1 Praproses Data

##### 4.1.1 Identifikasi Resolusi Gambar

Praproses data dalam penelitian ini bertujuan untuk mengetahui ukuran gambar, yaitu tinggi(H) dan lebar(W) dalam satuan piksel, berdasarkan citra hasil pemotretan. Rata-rata resolusi gambar yang dihasilkan dari dua kamera selama proses pengambilan data dihitung berdasarkan rata-rata ukuran setiap kelas. Identifikasi ini dilakukan untuk memastikan bahwa perbedaan ukuran tinggi(H) dan lebar(W) antar kelas tidak terlalu jauh. Dataset terdiri dari tiga kelas, masing-masing berisi 100 gambar, dengan rata-rata ukuran tinggi dan lebar sebagai berikut:

- Rataan Resolusi ( $H \times W$ ) *Avicennia alba*: 2136.56 px  $\times$  1576.42 px
- Rataan Resolusi ( $H \times W$ ) *Rhizophora apiculata*: 2057.99 px  $\times$  1213.73 px
- Rataan Resolusi ( $H \times W$ ) *Rhizophora stylosa*: 2392.49 px  $\times$  1626.71 px

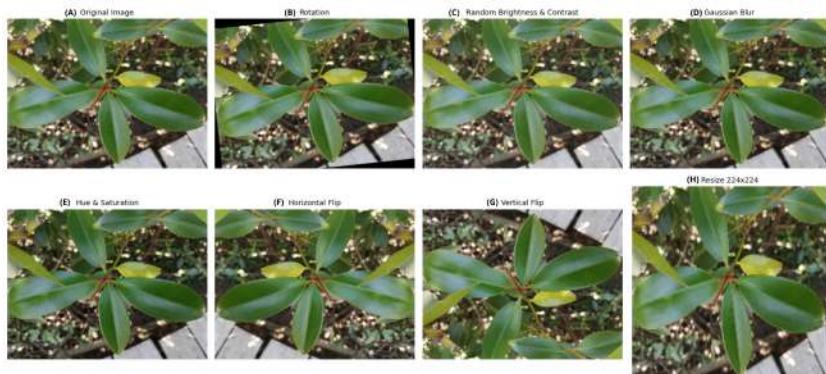
Hasil pengukuran menunjukkan bahwa rata-rata panjang dan lebar gambar bervariasi antar kelas. Karena variasi ukuran dan jarak pengambilan gambar masih cenderung dekat, proses pemotongan (*crop*) sebaiknya dihindari. Hal ini bertujuan untuk mencegah hilangnya fitur penting pada bentuk daun yang dapat memengaruhi identifikasi karakteristik spesifik tiap kelas.

##### 4.1.2 Pembagian Dataset

Setelah dilakukan identifikasi resolusi dari panjang dan lebar gambar, tahapan selanjutnya adalah pembagian dataset menggunakan metode *stratified split* dengan proporsi 80:20 untuk setiap kelas. Dari total 300 gambar, masing-masing kelas terdiri atas 100 gambar yang kemudian dibagi menjadi 80 gambar untuk data pelatihan (*train*) dan 20 gambar untuk data pengujian (*test*). Dengan demikian, diperoleh total 240 gambar untuk pelatihan dan 60 gambar untuk pengujian. Pembagian *stratified* menjaga distribusi kelas tetap seimbang dan model dapat belajar secara optimal dan adil dari ketiga spesies mangrove *Avicennia alba*, *Rhizophora apiculata*, dan *Rhizophora stylosa* tanpa adanya dominasi kelas tertentu.

## 4.2 Augmentasi Gambar

Proses augmentasi gambar, atau yang dapat disebut sebagai penambahan noise maupun pola khusus pada data, dilakukan pada data latih (*training set*) untuk memperkaya variasi pola selama pelatihan model. Teknik ini tidak hanya meningkatkan akurasi dalam mengenali kondisi sekitar, tetapi juga membuat model lebih *robust*, yaitu mampu mempertahankan kinerjanya meskipun dihadapkan pada kondisi yang berbeda atau tidak ideal dibandingkan dengan data pelatihan, seperti perubahan pencahayaan, sudut pandang, atau adanya gangguan visual.



Gambar 4.1 Augmentasi Gambar

Pada Gambar 4.1 merupakan hasil dari augmentasi gambar yang diterapkan pada gambar daun mangrove(A), seperti *rotate*(B) yaitu mengubah rotasi daun sebesar  $20^\circ$  searah atau berlawanan arah jarum jam, *brightness contrast*(C) mengubah kecerahan dan kontras menjadi lebih cerah dengan proporsi perubahan  $\pm 20\%$  dari gambar aslinya, *Gaussian Blur*(D) bertujuan untuk mengubah bentuk gambar daun menjadi bentuk yang sedikit buram menyerupai kamera lama dengan kernel blur yang diterapkan sebesar  $3 \times 3$  piksel, *hue-saturation*(E) untuk mengubah warna gambar berdasarkan tiga komponen warna yaitu *hue* (nada warna), *saturation* (kejemuhan warna), dan *value* (kecerahan) dengan pergeseran dari kecerahan dan kejemuhan sebesar  $\pm 10$  unit transformasi ini bertujuan untuk meniru kondisi pencahayaan dan sensor kamera yang berbeda, *FlipH*(F) dan *FlipV*(G) merupakan kedua augmentasi yang memberikan efek cermin atau pembalikan dengan posisi yang berbeda secara horizontal (kesamping) dan vertikal (keatas), *Resize 224 × 224*(H) merupakan representasi dari penyesuaian

ukuran input terhadap data gambar yang akan di proses melewati model yang dibangun, proses *resize* diterapkan pada data latih (*train*) maupun data uji (*test*). Selanjutnya, gambar yang sudah melewati proses augmentasi akan dipakai untuk tahapan pemodelan menggunakan arsitektur Res4Net-CBAM secara *on-the-fly augmentation* atau dapat disebut juga pelatihan model disertai augmentasi secara langsung.

### 4.3 Setup Arsitektur Res4Net-CBAM

Arsitektur Res4Net-CBAM yang direkonstruksi pada penelitian ini memiliki penyusun utama dalam ilustrasi berikut:

Untuk lebih jelas struktur dari arsitekturnya dapat dilihat pada Tabel lampiran C.1

Arsitektur Res4Net-CBAM yang direkonstruksi memiliki total parameter sebanyak 7.371.395, seluruhnya bersifat *trainable*, tanpa parameter *non-trainable*, menunjukkan bahwa semua bagian model dapat belajar selama pelatihan. Ukuran model diperkirakan sekitar 28.12 MB, yang berasal dari representasi parameter dalam float32 (4 byte per parameter). Selama proses training, total estimasi memori yang digunakan per batch mencapai sekitar 77.47 MB, termasuk input, *forward/backward pass*, dan parameter.

### 4.4 Pelatihan Model

Proses pelatihan model Res4Net-CBAM dilakukan dengan menggunakan dua jenis *optimizer*, yaitu Adam dan SGD (*Stochastic Gradient Descent*) dengan momentum sebesar 0,9. Setiap *optimizer* dikombinasikan dengan tiga nilai *learning rate* yang berbeda, yaitu  $10^{-3}$ ,  $10^{-4}$ , dan  $10^{-5}$ . Pelatihan dilakukan sebanyak 100 *epoch* untuk setiap kombinasi antara *optimizer* dan *learning rate*. Selama proses pelatihan, metode yang digunakan yakni *ModelCheckpoint* dengan mekanisme prosesnya menyimpan bobot secara otomatis untuk model terbaik berdasarkan nilai *loss* terkecil dan akurasi tertinggi pada setiap *epoch*.

Berdasarkan Grafik Gambar 4.2, yang terdiri dari grafik *Training Loss* (A), *Validation Loss* (B), *Training Accuracy* (C), dan *Validation Accuracy* (D), terlihat bahwa *optimizer* Adam menunjukkan performa pelatihan yang lebih unggul dan konsisten dibandingkan SGD pada berbagai nilai *learning rate*.

Pada grafik (A), Adam dengan *learning rate*  $10^{-3}$  (garis hijau) menunjukkan penurunan *training loss* yang sangat cepat sejak awal pelatihan, dari sekitar 1.0 ke bawah 0.1 hanya dalam 20 *epoch* meskipun polanya fluktuatif. Adam dengan *learning rate*  $10^{-4}$  dan  $10^{-5}$  menunjukkan tren menurun yang stabil. Sebaliknya, SGD dengan *learning rate*  $10^{-4}$  dan  $10^{-5}$  (garis biru dan ungu) memperlihatkan *loss* yang hampir stagnan sepanjang 100 *epoch*, berada di atas 1.0. Meskipun SGD dengan *learning rate*  $10^{-3}$  (garis oranye) mengalami penurunan *loss*, kurvanya tidak sehalus Adam dan cenderung fluktuatif.

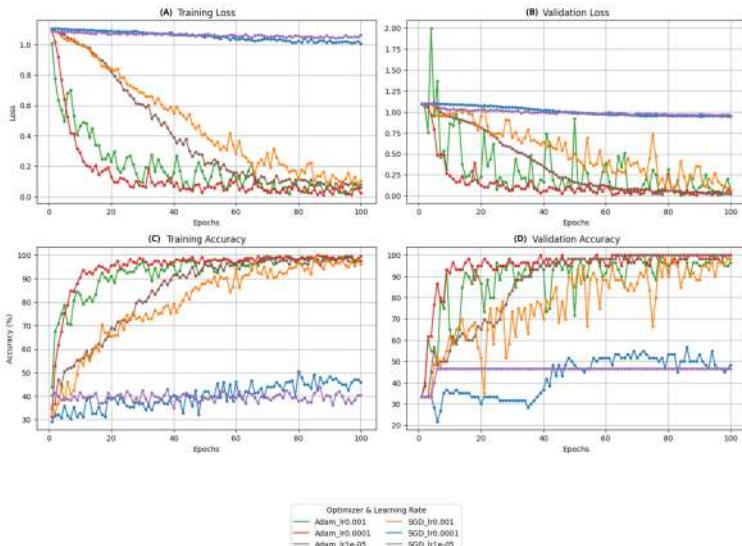
Hal serupa tampak pada grafik (B) *Validation Loss*, di mana Optimizer Adam (terutama  $10^{-3}$  dan  $10^{-4}$ ) kembali menunjukkan penurunan *loss* yang signifikan hingga di bawah 0.1, sementara SGD dengan  $10^{-4}$  dan  $10^{-5}$  stagnan di sekitar nilai awal, dan SGD  $10^{-3}$  memiliki pola fluktuatif yang kurang stabil.

Pada grafik (C), akurasi pelatihan dengan Adam  $10^{-3}$  mencapai lebih dari 95% hanya dalam 20 *epoch* dan terus meningkat hingga mendekati 100%. SGD  $10^{-3}$  juga menunjukkan kenaikan, tetapi jauh lebih lambat dan tidak setinggi Adam. SGD dengan *learning rate*  $10^{-4}$  dan  $10^{-5}$  menunjukkan peningkatan akurasi yang sangat terbatas (sekitar 30–45% saja).

Grafik (D) menunjukkan tren akurasi validasi yang konsisten. Dengan optimizer Adam  $10^{-3}$  dan  $10^{-4}$  berhasil mencapai akurasi validasi di atas 95%, menunjukkan kemampuan generalisasi yang baik. Berbanding terbalik dengan optimizer SGD  $10^{-3}$  terjadi fluktuasi tinggi dengan akurasi yang tidak stabil, sementara dua konfigurasi SGD pada  $10^{-4}$  dan  $10^{-5}$  menunjukkan performa validasi yang lemah (kurang dari 60%).

Secara keseluruhan, Adam menunjukkan performa pelatihan dan validasi yang jauh lebih baik dan stabil dibandingkan SGD. Keunggulan Adam berasal dari kemampuannya melakukan penyesuaian adaptif terhadap *learning rate* per parameter, menggabungkan elemen momentum dan RMSProp dalam optimisasi. Hal ini sangat berguna pada arsitektur kompleks seperti Res4Net-CBAM yang memiliki banyak parameter dan dinamika gradien yang bervariasi. Selain mempercepat konvergensi, Adam juga lebih efisien dalam menghindari jebakan *local minima* dan pelatihan yang lambat akibat gradien kecil [31].

Temuan ini sejalan dengan hasil penelitian oleh Ketut et al. [38], yang menunjukkan bahwa Adam memiliki keunggulan dalam hal kecepatan konvergensi dan kestabilan pelatihan pada arsitektur CNN, dibandingkan dengan SGD maupun SGD dengan momentum. Selain itu, penelitian Nugraha et al. [14] yang menerapkan metode transfer learning untuk klasifikasi daun mangrove juga mengadopsi Adam sebagai *default optimizer* dan mencatat akurasi tinggi serta proses pelatihan yang stabil, meskipun tidak melakukan perbandingan eksplisit dengan SGD.



**Gambar 4.2** Pelatihan Model Res4Net-CBAM

Untuk memastikan kualitas prediksi model secara menyeluruh dan memilih kombinasi *optimizer* dan *learning rate* terbaik. Metrik evaluasi seperti *confusion matrix* dan kurva ROC/AUC masing-masing memberikan gambaran lebih komprehensif terhadap kemampuan klasifikasi model pada data uji yang dilakukan terhadap model yang sudah dilatih sebelumnya.

## 4.5 Evaluasi Model

### 4.5.1 Confusion Matrix

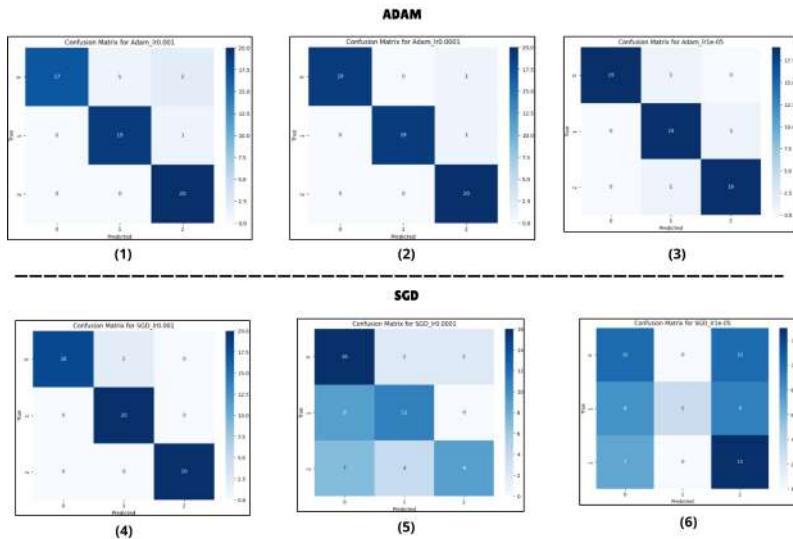
Berdasarkan Gambar 4.3 menyajikan hasil *confusion matrix* dari model klasifikasi tiga kelas, yang memperlihatkan performa unggul dari

model dengan *optimizer* Adam, ditunjukkan oleh dominasi warna diagonal. Model dengan *optimizer* SGD juga menunjukkan hasil yang cukup baik, khususnya pada *learning rate*  $10^{-3}$  (4), dengan sebagian data *test* berhasil diklasifikasikan secara akurat. Evaluasi lanjutan menggunakan metrik seperti *accuracy*, *precision*, *recall*, *specificity*, dan *F1-score* disajikan pada Tabel 4.1.

Secara keseluruhan, penggunaan *optimizer* Adam menunjukkan performa yang lebih stabil dan konsisten pada berbagai nilai *learning rate*, yang ditampilkan pada Gambar 4.3(a)–(c). Misalnya, pada *learning rate*  $10^{-4}$ (b), Adam menghasilkan klasifikasi yang hampir sempurna dengan hanya dua kesalahan minor pada prediksi kelas 0 (*Avicennia alba*) dan 1 (*Rhizophora apiculata*). Hal ini tercermin dari metrik evaluasi yang sangat tinggi: akurasi 96,67%, *precision* 96,97%, *recall* 96,67%, *specificity* 98,33%, dan *F1-score* 96,70%. Penilaian ini menunjukkan distribusi prediksi yang seimbang untuk ketiga kelas, mengindikasikan bahwa model mampu membedakan antar kelas secara efektif. Bahkan ketika *learning rate* diubah ke  $10^{-3}$ (a) atau  $10^{-5}$ (c), performa model tetap berada pada tingkat tinggi, masing-masing dengan akurasi 93,33% dan 95,00%. Hal ini menunjukkan bahwa Adam memiliki karakteristik *robustness* terhadap perubahan skala pembelajaran.

Berbeda dengan Adam, optimizer SGD menunjukkan performa yang jauh lebih bergantung pada nilai *learning rate*, seperti yang terlihat pada Gambar 4.3(d)–(f). Ketika digunakan dengan *learning rate*  $10^{-3}$ (d), SGD mampu menyamai performa terbaik Adam, dengan akurasi 96,67% dan *F1-score* 96,66%. Pada matrix ini, model berhasil mengklasifikasikan hampir seluruh sampel dengan benar, menunjukkan bahwa dengan nilai pembelajaran yang tepat, SGD dapat mencapai performa optimal. Namun, ketika *learning rate* diturunkan menjadi  $10^{-4}$ (e), performa model menurun drastis; terjadi banyak kesalahan klasifikasi terutama pada kelas 1 dan 2. Hal ini berlanjut lebih buruk pada *learning rate*  $10^{-5}$ (f), di mana terlihat ketidakseimbangan klasifikasi dengan tingginya prediksi salah antara kelas 0 (*Avicennia alba*) dan kelas 2 (*Rhizophora stylosa*). Akurasi menurun drastis menjadi 43,33%, dan *F1-score* turun ke 40,18%. Penurunan ini menunjukkan fenomena *underfitting*, di mana model gagal belajar karena pembaruan bobot terlalu kecil untuk menangkap pola dalam data.

Dengan mempertimbangkan keseluruhan hasil, dapat disimpulkan bahwa model yang dilatih dengan optimizer Adam lebih unggul dalam stabilitas performa model Res4Net-CBAM. Model yang diuji berhasil mempertahankan kinerja tinggi bahkan ketika nilai *learning rate* diubah-ubah, seperti dibuktikan oleh keseragaman warna biru tua diagonal dalam *confusion matrix* pada Gambar 4.3(a)–(c). Sebaliknya, SGD cenderung sensitif terhadap perubahan *learning rate*, sehingga hanya memberikan performa optimal pada pengaturan tertentu, yakni  $10^{-3}$ (d) saja. Oleh karena itu, dalam konteks tugas klasifikasi banyak kelas seperti ini, Adam memberikan fleksibilitas dan keandalan lebih besar untuk proses pelatihan model yang stabil dan memberikan generalisasi terhadap data yang dilatih.



**Gambar 4.3** Hasil *Confusion Matrix* - Res4Net-CBAM

**Tabel 4.1** Evaluasi Confusion Matrix Model Res4Net-CBAM

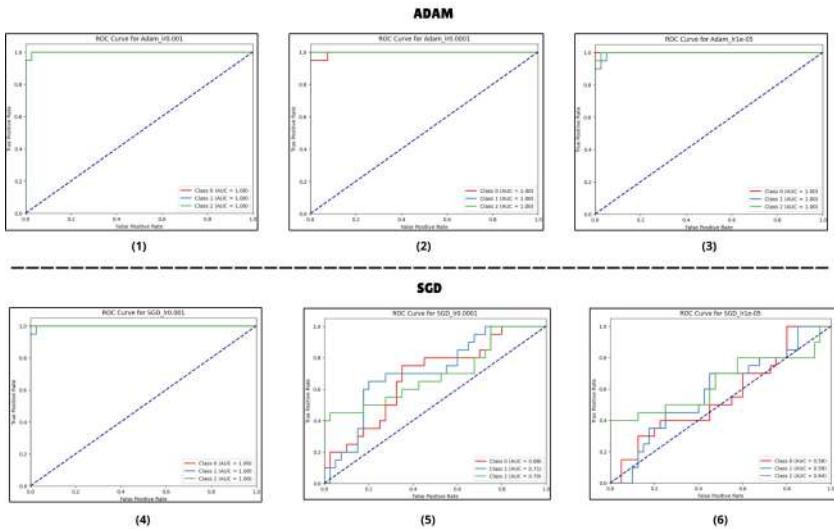
<b>Optimizer</b>	<b><i>Learning Rate</i></b>	<b>Accuracy(%)</b>	<b>Precision(%)</b>	<b>Recall(%)</b>	<b>Specificity(%)</b>	<b>F1-Score(%)</b>
Adam	$10^{-3}$	93.33	93.99	93.33	96.56	93.31
	$10^{-4}$	96.67	96.97	96.67	98.33	96.70
	$10^{-5}$	95.00	95.16	95.00	97.48	95.04
SGD	$10^{-3}$	96.67	96.97	96.67	98.33	96.66
	$10^{-4}$	60.00	65.51	60.00	76.43	59.69
	$10^{-5}$	43.33	60.21	43.33	64.08	40.18

#### 4.5.2 Kurva ROC/AUC

Hasil metrik ROC/AUC yang ditampilkan pada Gambar 4.4 menunjukkan performa model Res4Net-CBAM yang telah dilatih sebelumnya dalam klasifikasi tiga spesies daun mangrove. Grafik (1), (2) dan (3) menunjukkan hasil dari model yang dilatih dengan menggunakan optimizer Adam dengan *learning rate* yang diatur secara berurutan  $10^{-3}$ ,  $10^{-4}$ , dan  $10^{-5}$ . Pada setiap grafik yang ditujukan model yang dibangun untuk optimizer Adam dengan kombinasi 3 *learning rate* dapat memisahkan antar kelas dengan sempurna dengan indikasi dari nilai AUC dari setiap kelas menunjukkan 1.00 dan menunjukkan performa klasifikasi yang sangat tinggi dan konsisten untuk setiap kelas. Secara khusus untuk grafik (1), (2) dan (3) seperti indikator warna garis tidak terlalu tampak di grafik sebab indikator warna garis menyatu dengan garis koordinat sumbu-y yang menyebabkan indikator garis tidak terlihat secara eksplisit.

Sebaliknya, grafik (4), (5), dan (6) menunjukkan performa model saat menggunakan optimizer SGD. Pada *learning rate*  $10^{-3}$  Grafik (4), model masih mampu menghasilkan AUC sempurna (1.00) untuk semua kelas. Namun, penurunan *learning rate* menjadi  $10^{-4}$  Grafik (5) menyebabkan penurunan performa yang signifikan, dengan AUC berkisar antara 0.68 hingga 0.71. Penurunan juga terjadi pada *learning rate*  $10^{-5}$  Grafik (6), di mana nilai AUC berada dalam rentang 0.56 hingga 0.64 dan kurva ROC mendekati garis diagonal, yang menunjukkan performa klasifikasi yang lemah dan mendekati tebakan acak (*random guess*) yang ditandai dengan garis putus-putus pada diagonal grafik.

Hasil secara keseluruhan, hasil metrik ROC/AUC menunjukkan bahwa penggunaan optimizer Adam memberikan hasil klasifikasi yang lebih stabil dan konsisten dalam memisahkan antar kelas dan unggul dibandingkan dengan SGD di beberapa variasi *learning rate*.



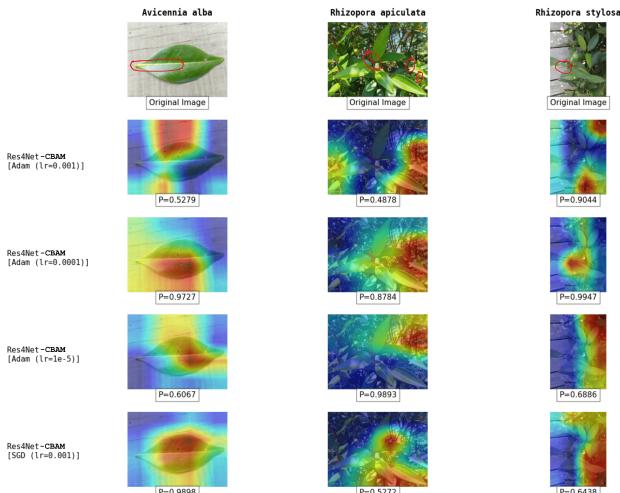
**Gambar 4.4** Kurva ROC/AUC - Res4Net-CBAM

#### 4.6 Interpretasi Grad-CAM

Berdasarkan hasil evaluasi menggunakan *confusion matrix* dan kurva ROC/AUC, model Res4Net-CBAM yang memenuhi kriteria akurasi, presisi, recall, F1-score, serta nilai AUC di atas 80% untuk setiap kelas, yakni dengan menggunakan *optimizer* Adam dengan variasi *learning rate*  $10^{-3}$ ,  $10^{-4}$ , dan  $10^{-5}$ , serta satu model yang dilatih menggunakan *optimizer* SGD dengan *learning rate*  $10^{-3}$ . Selanjutnya, model yang terpilih akan dianalisis menggunakan metode Grad-CAM untuk mengidentifikasi area morfologis spesifik yang berkontribusi terhadap keputusan klasifikasi, sehingga memberikan pemahaman mendalam terhadap pola perhatian (*attention*) pada setiap kelas spesies mangrove.

Visualisasi Grad-CAM pada Gambar 4.5 menunjukkan hasil model *Res4Net-CBAM* terhadap citra daun dari tiga spesies berbeda. Kombinasi arsitektur dengan *optimizer* Adam dan *learning rate*  $10^{-4}$  memberikan hasil interpretasi yang paling konsisten dan relevan terhadap fitur morfologis utama. Area atensi model secara jelas terpusat pada bagian penting daun—seperti ujung, tulang, dan pangkal daun—yang telah diidentifikasi sebelumnya sebagai penanda morfologi khas (ditandai dengan lingkaran merah). Kombinasi ini juga menghasilkan nilai *confidence score* (*P*) tinggi dan stabil: 0,9727 untuk *Avicennia alba*, 0,8784 untuk *Rhizophora apiculata*, dan 0,9947 untuk *Rhizophora stylosa*.

Sebaliknya, penggunaan *learning rate* yang terlalu tinggi ( $10^{-3}$ ) atau terlalu rendah ( $10^{-5}$ ) dengan Adam menghasilkan distribusi atensi yang menyebar dan kurang fokus pada area morfologis penting, disertai skor prediksi yang lebih rendah (misalnya,  $P = 0,5279$  dan  $0,6067$  untuk *Avicennia alba*). Sementara itu, penggunaan *optimizer* SGD menunjukkan hasil yang lebih bervariasi. Meskipun kombinasi SGD– $10^{-3}$  memberikan pemetaan fitur yang baik dan nilai  $P$  tinggi ( $0,9898$ ) pada *Avicennia alba*, performa pada spesies lain tidak stabil, dengan fokus atensi yang kurang tepat dan nilai  $P$  menurun ( $P = 0,5272$  dan  $0,6438$ ).



**Gambar 4.5** Hasil Grad-CAM Res4Net-CBAM

Variabilitas hasil mengindikasikan bahwa keberhasilan interpretasi model sangat dipengaruhi oleh pemilihan *hyperparameter* pelatihan, meskipun arsitektur telah dilengkapi dengan modul perhatian seperti CBAM. Oleh karena itu, visualisasi Grad-CAM berperan penting tidak hanya dalam menilai efektivitas klasifikasi, tetapi juga dalam mengungkap sejauh mana model mampu mengidentifikasi fitur spasial yang bermakna secara biologis. Di sisi lain, metode interpretabilitas ini masih jarang digunakan dalam studi sejenis, seperti yang terlihat pada penelitian Farhan et al. [39] yang hanya menyoroti performa model CNN dengan evaluasi *confusion matrix* saja dan penelitian Asnur et al. [13] yang lebih berfokus pada hasil pelatihan tanpa membahas aspek transparansi model secara mendalam dan tidak adanya metrik khusus yang menjadi tolak ukur keberhasilan model.

## 4.7 Pengujian Model *On-Site*

Proses pengujian model secara *on-site* diperlukan untuk memastikan model cocok dengan lingkungan mangrove di Cuku NyiNyi. Pengujian dilakukan dengan menggunakan aplikasi berbasis Android yang diintegrasikan dengan model yang sudah dipilih sebelumnya berdasarkan hasil evaluasi. Tabel 4.2 merupakan hasil dari rekapitulasi pengujian model terhadap 15 gambar dengan masing-masing spesies diwakili sebanyak 5 gambar.

Berdasarkan hasil rekapitulasi model pada Tabel 4.2, model Res4Net-CBAM yang dilatih menggunakan *optimizer* Adam dengan *learning rate* sebesar  $10^{-4}$  menunjukkan kinerja terbaik, dengan jumlah prediksi benar sebanyak 14 dari 15 sampel gambar, dibandingkan dengan kombinasi *learning rate* lainnya. Model ini juga menunjukkan distribusi prediksi yang merata pada ketiga kelas mangrove selama pengujian *on-site*, yang mengindikasikan kemampuannya dalam mengenali karakteristik unik dari spesies *Avicennia alba*, *Rhizophora apiculata*, dan *Rhizophora stylosa* secara akurat. Model Res4Net-CBAM yang awalnya direkonstruksi untuk mendeteksi penyakit pada daun teh [24] berhasil diimplementasikan untuk klasifikasi daun mangrove dengan pola yang kompleks, serta menunjukkan performa optimal dari berbagai pengujian.

**Tabel 4.2** Rekapitulasi Hasil Deteksi Daun Mangrove dengan Res4Net-CBAM

Model	Learning Rate	<i>Avicennia alba</i>	<i>Rhizophora apiculata</i>	<i>Rhizophora stylosa</i>	Jumlah
Res4Net-CBAM (Adam)	$10^{-3}$	2	3	1	6
	$10^{-4}$	5	5	4	14
	$10^{-5}$	3	3	4	10
Res4Net-CBAM (SGD)	$10^{-3}$	1	2	2	5

Pendekatan menggunakan pengujian *on-site* sangat berbeda dengan beberapa penelitian sebelumnya yang umumnya hanya mengevaluasi performa model pada data uji di lingkungan dataset publik, seperti pada studi oleh Farhan et al. [39] dan Asnur et al. [13], yang tidak melakukan pengujian langsung di lapangan. Oleh karena itu, pengujian *on-site* yang dilakukan dalam penelitian ini memberikan kontribusi signifikan dalam mengevaluasi kestabilan dan kemampuan generalisasi model terhadap kondisi nyata di lapangan serta menjadikannya lebih aplikatif untuk diterapkan dalam sistem klasifikasi mangrove di Desa Ekowisata Cuku NyiNyi.

## **BAB V**

### **KESIMPULAN DAN SARAN**

#### **5.1 Kesimpulan**

Arsitektur Res4Net-CBAM menunjukkan kinerja yang sangat baik dalam mengklasifikasi tiga spesies daun mangrove: *Avicennia alba*, *Rhizophora apiculata*, dan *Rhizophora stylosa*. Kombinasi *optimizer* Adam dengan *learning rate*  $10^{-4}$  memberikan hasil terbaik, mencapai akurasi sekitar 96% dan skor *ROC/AUC* sempurna (1.00) untuk semua kelas. Visualisasi menggunakan Grad-CAM memperkuat bahwa model secara efektif mengenali area-area penting pada daun serta menghasilkan nilai prediksi yang akurat untuk masing-masing kelas. Selanjutnya, uji coba secara langsung di lokasi mangrove Cuku NyiNyi membuktikan bahwa model tetap akurat dan stabil, menjadikan konfigurasi ini paling cocok untuk sistem klasifikasi daun mangrove di Desa Ekowisata Cuku NyiNyi.

#### **5.2 Saran**

Untuk penelitian selanjutnya, eksplorasi lebih lanjut dan pengambilan dataset yang lebih besar dari gambar mangrove di Desa Ekowisata Cuku NyiNyi akan dilakukan guna memperoleh hasil yang lebih optimal serta memastikan model Res4Net-CBAM tetap *sustainable* terhadap pola-pola baru pada bentuk daun mangrove yang berubah-ubah setiap pertumbuhannya.

## DAFTAR PUSTAKA

- [1] Badan-Riset-dan-Inovasi-Nasional, *Jaga ekosistem hutan dan laut, periset brin gagas kolaborasi co-management*, Accessed: 23 September 2024, 2023. sumber: <https://www.brin.go.id/news/116909/jaga-ekosistem-hutan-dan-laut-periset-brin-gagas-kolaborasi-co-management>.
- [2] BINUS-University, *Melihat keindahan pantai nan indah di indonesia*, Accessed: 23 September 2024, 2024. sumber: <https://binus.ac.id/alumni/news/4678/melihat-keindahan-pantai-nan-indah-di-indonesia/>.
- [3] Universitas-Lampung, *Wilayah kerja program mitra bahari*, Accessed: 23 September 2024, 2024. sumber: <https://pik.fp.unila.ac.id/program-mitra-bahari/wilayah-kerja/>.
- [4] PT-Bukit-Asam, *Tanam mangrove, masyarakat desa sidodadi bersama bukit asam (ptba) menanam kebaikan*, Accessed: 23 September 2024, 2024. sumber: <https://www.ptba.co.id/berita/tanam-mangrove-masyarakat-desa-sidodadi-bersama-bukit-asam-ptba-menanam-kebaikan-1919>.
- [5] X. Feng, S. Xu, J. Li, dkk., “Molecular adaptation to salinity fluctuation in tropical intertidal environments of a mangrove tree \*sonneratia alba\*”, *BMC Plant Biology*, vol. 20, no. 1, hlmn. 178, 2020. sumber: <https://doi.org/10.1186/s12870-020-02395-3>.
- [6] M. E. B. Gerona-Daga dan S. G. Salmo, “A systematic review of mangrove restoration studies in southeast asia: Challenges and opportunities for the united nation’s decade on ecosystem restoration”, *Frontiers in Marine Science*, vol. 9, 2022. sumber: <https://www.frontiersin.org/journals/marine-science/articles/10.3389/fmars.2022.987737>.
- [7] B. A. Polidoro, K. E. Carpenter, L. Collins, dkk., “The loss of species: Mangrove extinction risk and geographic areas of global concern”, *PLoS ONE*, vol. 5, no. 4, e10095, Apr. 2010. sumber: <https://doi.org/10.1371/journal.pone.0010095>.
- [8] D. R. Bellwood dan C. P. Meyer, “Searching for heat in a marine biodiversity hotspot”, *Journal of Biogeography*, vol. 36, no. 4, hlmn. 569–576, 2009. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1365-2699.2008.02029>.

- x. sumber: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2699.2008.02029.x>.
- [9] O. Russakovsky, J. Deng, H. Su, dkk., *Imagenet large scale visual recognition challenge*, 2015. arXiv: 1409.0575 [cs.CV]. sumber: <https://arxiv.org/abs/1409.0575>.
- [10] S. Xie, R. Girshick, P. Dollár, Z. Tu, dan K. He, *Aggregated residual transformations for deep neural networks*, 2017. arXiv: 1611.05431 [cs.CV]. sumber: <https://arxiv.org/abs/1611.05431>.
- [11] S. Woo, J. Park, J.-Y. Lee, dan I. S. Kweon, *Cbam: Convolutional block attention module*, 2018. arXiv: 1807.06521 [cs.CV]. sumber: <https://arxiv.org/abs/1807.06521>.
- [12] Z. Cao, K. Wang, J. Wen, dkk., “Fine-grained image classification on bats using vgg16-cbam: A practical example with 7 horseshoe bats taxa (chiroptera: Rhinolophidae: Rhinolophus) from southern china”, *Frontiers in Zoology*, vol. 21, no. 10, 2024. sumber: <https://doi.org/10.1186/s12983-024-00531-5>.
- [13] P. Asnur, R. Kosasih, S. Madenda, dan D. A. Rahayu, “Identification of mangrove tree species using deep learning method”, *International Journal of Advances in Applied Sciences (IJAAS)*, vol. 12, no. 2, hlmn. 163–170, 2023. sumber: <https://doi.org/10.11591/ijaas.v12.i2.pp163-170>.
- [14] A. R. Nugraha, N. Widagti, R. Wardoyo, dkk., “Mangrove species identification using convolutional neural network”, *Taiwania*, vol. 70, hlmn. 2102–2102, 3 2025. sumber: <https://taiwania.ntu.edu.tw/abstract/2102>.
- [15] P. B. Tomlinson, *The Botany of Mangroves*. Cambridge: Cambridge University Press, 2016.
- [16] T. C. Goulding dan B. Dayrat, “The coral triangle and strait of malacca are two distinct hotspots of mangrove biodiversity”, *Scientific Reports*, vol. 13, no. 1, hlmn. 15 793, 2023. sumber: <https://doi.org/10.1038/s41598-023-42057-6>.
- [17] N. Duke, “Indo-west pacific stilt mangrove. rhizophora apiculata, r. mucronata, r. stylosa, r. x annamalai, r. x lamarckii”, In: *Traditional Trees of Pacific Islands: Their Culture, Environment, and Use*, hlmn. 641–660, Nov. 2006.
- [18] K. O’Shea dan R. Nash, *An introduction to convolutional neural networks*, 2015. arXiv: 1511.08458 [cs.NE]. sumber: <https://arxiv.org/abs/1511.08458>.

- [19] A. G. Howard, *Some improvements on deep convolutional neural network based image classification*. sumber: <http://code.google.com/p/cuda-convnet>.
- [20] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, dan J. Schmidhuber, “High-performance neural networks for visual object classification”, Feb. 2011. sumber: <http://arxiv.org/abs/1102.0183>.
- [21] A. Geron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. O'Reilly Media, 2019.
- [22] I. Goodfellow, Y. Bengio, dan A. Courville, *Deep Learning*. MIT Press, 2016. sumber: <https://www.deeplearningbook.org/>.
- [23] S. Ioffe dan C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, 2015. arXiv: 1502.03167 [cs.LG]. sumber: <https://arxiv.org/abs/1502.03167>.
- [24] P. Bhuyan, P. K. Singh, dan S. K. Das, “Res4net-cbam: A deep cnn with convolution block attention module for tea leaf disease diagnosis”, *Multimedia Tools and Applications*, vol. 83, no. 16, hlmn. 48 925–48 947, 2024. sumber: <https://doi.org/10.1007/s11042-023-17472-6>.
- [25] K. He, X. Zhang, S. Ren, dan J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV]. sumber: <https://arxiv.org/abs/1512.03385>.
- [26] X. Glorot, A. Bordes, dan Y. Bengio, “Deep sparse rectifier neural networks”, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, vol. 15, hlmn. 315–323, 2011.
- [27] A. F. Agarap, “Deep learning using rectified linear units (relu)”, *arXiv preprint arXiv:1803.08375*, 2018.
- [28] B. Xu, N. Wang, T. Chen, dan M. Li, “Empirical evaluation of rectified activations in convolutional network”, di dalam *International Conference on Machine Learning (ICML)*, 2015.
- [29] Y. Han, B. Lakshminarayanan, dan M. Zeiler, “Sigmoid-weighted linear units for neural network function approximation in reinforcement learning”, *Springer Neural Computing and Applications*, 2020.

- [30] J. S. Bridle, “Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition”, di dalam *Neurocomputing*, 1990, hlmn. 227–236.
- [31] D. P. Kingma dan J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412 . 6980 [cs.LG]. sumber: <https://arxiv.org/abs/1412.6980>.
- [32] L. Bottou, “Large-scale machine learning with stochastic gradient descent”, di dalam *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010)*, Y. Lechevallier dan G. Saporta, timed., Paris, France: Springer, Agt. 2010, hlmn. 177–187. sumber: <http://leon.bottou.org/papers/bottou-2010>.
- [33] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, dan D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization”, *International Journal of Computer Vision*, vol. 128, no. 2, hlmn. 336–359, Okt. 2019. sumber: <http://dx.doi.org/10.1007/s11263-019-01228-7>.
- [34] I. Düntsch dan G. Gediga, “Confusion matrices and rough set data analysis”, *arXiv preprint arXiv:1902.01487*, 2019. sumber: <http://arxiv.org/abs/1902.01487v1>.
- [35] J. Davis dan M. Goadrich, “The relationship between precision-recall and roc curves”, vol. 06, Juni 2006. sumber: <https://doi.org/10.1145/1143844.1143874>.
- [36] P. M. Gopych, *Roc curves within the framework of neural network assembly memory model: Some analytic results*, 2003. arXiv: cs/0309007 [cs.AI]. sumber: <https://arxiv.org/abs/cs/0309007>.
- [37] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, dan A. A. Kalinin, “Albumentations: Fast and flexible image augmentations”, *Information*, vol. 11, no. 2, 2020. sumber: <https://www.mdpi.com/2078-2489/11/2/125>.
- [38] Ketut, I, Wirayasa, dkk., “Comparison of convolutional neural networks model using different optimizers for image classification”, *International Journal of Sciences: Basic and Applied Research (IJSBAR)*, vol. Vol 60 No 2 (2021), hlmn. 116–126, Sept. 2021.

- [39] F. Farhan, C. A. Sari, E. H. Rachmawanto, dan N. R. D. Cahyo, “Mangrove tree species classification based on leaf, stem, and seed characteristics using convolutional neural networks with k-folds cross validation optimization”, *Advance Sustainable Science, Engineering and Technology*, vol. 5, no. 3, hlmn. 02 303 011, 2023. sumber: <http://journal.upgris.ac.id/index.php/asset/article/view/17188>.

# LAMPIRAN

# LAMPIRAN A

## Surat Pengambilan Data



KEMENTERIAN PENDIDIKAN TINGGI, SAINS,  
DAN TEKNOLOGI  
INSTITUT TEKNOLOGI SUMATERA  
LEMBAGA PENELITIAN DAN PENGABDIAN KEPADA MASYARAKAT  
Jalan Terusan Ryaedau Way Hui, Kecamatan Jati Agung, Lampung Selatan 35365  
Telepon: (0721) 8030188  
Email: [ippem@itera.ac.id](mailto:ippem@itera.ac.id), Website: <http://ippem.itera.ac.id>

### SURAT KETERANGAN

Nomor: 4626/TT9.2.1/ PT.01.03/2024

Yang bertanda tangan di bawah ini:

Nama : Yahdi Zain  
NRK : 1951030920221421  
Jabatan : Kepala Lembaga Penelitian dan Pengabdian kepada Masyarakat

dengan ini mecerangkan bahwa,

No	Nama	NIDN/K	Program Studi	Keterangan
1.	Mika Alvionita S. S.Si., M.Si.	0009059302	Sains Data	Ketua
2.	Eristia Arfi, S.Si., M.Si.	0018119002	Matematika	Anggota
3.	Abdurrakhman Al Atsary	121450128	Sains Data	Mahasiswa
4.	Miftahul Huda	121450125	Sains Data	Mahasiswa

Telah melaksanakan kegiatan "Pengambilan Data Penelitian" skema Penelitian Dosen Pemula Dana PNBP Institut Teknologi Sumatera Tahun Anggaran 2024 pada tanggal 16 November 2024 dengan judul "Identifikasi Spesies Mangrove berdasarkan Morfologi Daun menggunakan Convolutional Neural Network pada Kawasan Ekowisata Cuku Nyinyi", di Desa Cuku Nyinyi, Kabupaten Pesawaran, Lampung.

Demikian surat keterangan ini dibuat, untuk digunakan sebagaimana mestinya.



Tentusai :

1. Wakil Rektor Bidang Akademik dan Kemahasiswaan.
2. Dekan Fakultas Sains.
3. Koordinator Prodi Sains Data.
4. Koordinator Prodi Matematika.
5. Arsitek.

## LAMPIRAN B

### Perhitungan Manual Modular CBAM

Berdasarkan Tinjauan Pustaka yang telah dituliskan didalam Bab 2 dapat dilakukan perhitungan dari CBAM dengan melalui 2 tahapan yakni CAM (*Channel Attention Module*) dan SAM (*Spatial Attention Module*).

#### B.0.1 *Channel Attention Module (CAM)*

Perhitungan kali ini dimulai dengan membangkitkan matrix secara acak dari 3 kanal (RGB) dengan ukuran  $5 \times 5$ . Inputan matriks sebagai berikut :

$$\text{Input} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \\ \hline 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \\ 32 & 34 & 36 & 38 & 40 \\ 42 & 44 & 46 & 48 & 50 \\ \hline 3 & 6 & 9 & 12 & 15 \\ 18 & 21 & 24 & 27 & 30 \\ 33 & 36 & 39 & 42 & 45 \\ 48 & 51 & 54 & 57 & 60 \\ 63 & 66 & 69 & 72 & 75 \end{bmatrix}$$

Kemudian dari 3 channel yang sudah dibangkitkan dicari nilai *max pooling* dan *avarage pooling* dari masing-masing channel. Perhitungannya sebagai berikut:

Untuk *max-pooling* dari masing-masing fitur dapat dijabarkan sebagai berikut :

- $F_{\max(\text{red})} = 25$
- $F_{\max(\text{green})} = 50$
- $F_{\max(\text{blue})} = 75$

Kemudian dilanjutkan dengan perhitungan mencari nilai rata-rata dari setiap channel untuk (*average pooling*) sebagai berikut :

- $F_{avg(red)} = \frac{1+2+3+4+5+6+\dots+25}{25} = 13$
- $F_{avg(green)} = \frac{2+4+6+8+10+\dots+50}{25} = 26$
- $F_{avg(blue)} = \frac{3+6+9+12+15+\dots+75}{25} = 39$

Setelah dicari dari *average pooling* dan *max pooling* dari setiap channel maka akan masuk kedalam proses *shared MLP* yang merupakan gabungan dari *MLP* (*Multi-layer Perceptron*) dengan 1 *hidden layer*, untuk proses matematisnya. Dalam hal ini, dapat di insiasi terlebih dahulu bobot (W) dan bias (b) untuk *MLP*,

$$\text{Matriks Bobot (3x3): } \begin{pmatrix} 0.5 & 0.5 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.6 & 0.7 \end{pmatrix}$$

$$\text{Matriks Bias (3x1): } \begin{pmatrix} 10 \\ 20 \\ 30 \end{pmatrix}$$

Hasil yang sudah didapatkan dalam max-pooling dan average pooling diatas digabungkan antar setiap channel menghasilkan matrix :

$$F_{max} = \begin{bmatrix} [25] \\ [50] \\ [75] \end{bmatrix}, \quad F_{avg} = \begin{bmatrix} [13] \\ [26] \\ [39] \end{bmatrix}$$

Proses didalam kaidah *shared MLP* mengikuti :

#### Average Pool:

$$\begin{aligned} F'_{avg} &= W^{(1)} \cdot F_{avg} + b_1, & F'_{max} &= W^{(1)} \cdot F_{max} + b_1, \\ F''_{avg} &= \text{ReLU}(F'_{avg}), & F''_{max} &= \text{ReLU}(F'_{max}), \\ F_{avg(out)} &= W^{(2)} \cdot F''_{avg} + b_2. & F_{max(out)} &= W^{(2)} \cdot F''_{max} + b_2. \end{aligned}$$

#### Max Pool:

Maka dapat dihitung sebagai :

Untuk **Max Pooling** :

$$F'_{max} = W^{(1)} \cdot F_{max} + b_1$$

$$F'_{max} = \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.6 & 0.7 \end{bmatrix} \cdot \begin{bmatrix} 25 \\ 50 \\ 75 \end{bmatrix} + \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix} = \begin{bmatrix} 85 \\ 35 \\ 117.5 \end{bmatrix}$$

Kemudian lakukan aktivasi ReLU terhadap hasil dari  $F'_{max}$  untuk menghasilkan  $F''_{max}$ :

$$F''_{max} : \text{ReLU}(0,x) \begin{bmatrix} 85 \\ 35 \\ 117.5 \end{bmatrix} = \begin{bmatrix} 85 \\ 35 \\ 117.5 \end{bmatrix}$$

Setelah didapatkan hasil dari  $F''_{max}$ , maka dilanjutkan dengan menghitung hasil dari  $F_{max(out)}$ . Misalkan kita samakan saja bobot dan bias yang ada dengan inisialisasi awal :

$$F_{max(out)} = W^{(2)} \cdot F''_{max} + b_2$$

$$F_{max(out)} = \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.6 & 0.7 \end{bmatrix} \cdot \begin{bmatrix} 85 \\ 35 \\ 117.5 \end{bmatrix} + \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix} = \begin{bmatrix} 128.75 \\ 43.75 \\ 150.25 \end{bmatrix}$$

Untuk **Average Pooling**

Kemudian dilakukan hal yang sama terhadap perhitungan untuk menghasilkan  $F'_{avg(out)}$  :

$$F'_{avg} = W^{(1)} \cdot F_{avg} + b_1$$

$$F'_{avg} = \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.6 & 0.7 \end{bmatrix} \cdot \begin{bmatrix} 13 \\ 26 \\ 39 \end{bmatrix} + \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix} = \begin{bmatrix} 49 \\ 27.8 \\ 75.5 \end{bmatrix}$$

Kemudian lakukan aktivasi ReLU terhadap hasil dari  $F'_{avg}$  untuk menghasilkan  $F''_{avg}$ :

$$F''_{avg} : \text{ReLU}(0,x) \begin{bmatrix} 49 \\ 27.8 \\ 75.5 \end{bmatrix} = \begin{bmatrix} 49 \\ 27.8 \\ 75.5 \end{bmatrix}$$

Setelah didapatkan hasil dari  $F''_{avg}$ , maka dilanjutkan dengan menghitung hasil dari  $F_{avg(out)}$ . Misalkan kita samakan saja bobot dan bias yang ada dengan inisialisasi awal :

$$F_{avg(out)} = W^{(2)} \cdot F''_{avg} + b_2$$

$$F_{avg(out)} = \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0.1 & 0.1 & 0.1 \\ 0.2 & 0.6 & 0.7 \end{bmatrix} \cdot \begin{bmatrix} 49 \\ 27.8 \\ 75.5 \end{bmatrix} + \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix} = \begin{bmatrix} 86.15 \\ 35.23 \\ 109.33 \end{bmatrix}$$

Setelah mendapatkan hasil dari *shared MLP* dari  $F_{max(out)}$  dan  $F_{avg(out)}$ , kedua matriks dijumlahkan dengan metode *element-wise summation* (penambahan elemen-per-elemen) kemudian dikenakan fungsi aktivasi sigmoid, sebagai berikut:

$$M_c(F) = \sigma((AvgPool(\mathbf{F})) + (MaxPool(\mathbf{F}))$$

$$(AvgPool(\mathbf{F}) + MaxPool(\mathbf{F})) = \begin{bmatrix} 86.15 \\ 35.23 \\ 109.33 \end{bmatrix} + \begin{bmatrix} 128.75 \\ 43.75 \\ 150.25 \end{bmatrix} = \begin{bmatrix} 214.9 \\ 78.98 \\ 259.58 \end{bmatrix}$$

$$M_c(F) = \sigma \begin{bmatrix} 214.9 \\ 78.98 \\ 259.58 \end{bmatrix} \approx \begin{bmatrix} [1] \\ [1] \\ [1] \end{bmatrix}$$

### B.0.2 Spatial Attention Module (SAM)

Misalkan inputan data disamakan dengan proses dari CAM (*Channel Attention Module*) yang merupakan inputan yang terdiri dari 3 Kanal (RGB):

$$\text{Input} = \left[ \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \\ \hline 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \\ 32 & 34 & 36 & 38 & 40 \\ 42 & 44 & 46 & 48 & 50 \\ \hline 3 & 6 & 9 & 12 & 15 \\ 18 & 21 & 24 & 27 & 30 \\ 33 & 36 & 39 & 42 & 45 \\ 48 & 51 & 54 & 57 & 60 \\ \hline 63 & 66 & 69 & 72 & 75 \end{array} \right]$$

Dari 3 Kanal diatas, tujuan dari SAM yakni mencari *Channel Wise*

*Average Pooling* dan *Channel Wise Max Pooling* dengan penjabaran rumus sebagai berikut :

**Channel-wise Average Pool:**

$$F_{\text{avg(spatial)}}(i, j) = \frac{1}{C} \sum_{c=1}^C F(c, i, j)$$

$$F_{\text{avg\_spatial}}(1, 1) = \frac{1}{3}(1 + 2 + 3) = 2,$$

$$F_{\text{avg\_spatial}}(1, 2) = \frac{1}{3}(2 + 4 + 6) = 4,$$

$$F_{\text{avg\_spatial}}(1, 3) = \frac{1}{3}(3 + 6 + 9) = 6,$$

$$F_{\text{avg\_spatial}}(1, 4) = \frac{1}{3}(4 + 8 + 12) = 8,$$

$$F_{\text{avg\_spatial}}(1, 5) = \frac{1}{3}(5 + 10 + 15) = 10,$$

:

$$F_{\text{avg\_spatial}}(5, 1) = \frac{1}{3}(21 + 42 + 63) = 42,$$

$$F_{\text{avg\_spatial}}(5, 2) = \frac{1}{3}(22 + 44 + 66) = 44,$$

$$F_{\text{avg\_spatial}}(5, 3) = \frac{1}{3}(23 + 46 + 69) = 46,$$

$$F_{\text{avg\_spatial}}(5, 4) = \frac{1}{3}(24 + 48 + 72) = 48,$$

$$F_{\text{avg\_spatial}}(5, 5) = \frac{1}{3}(25 + 50 + 75) = 50,$$

**Channel-wise Max Pool:**

$$F_{\text{max(spatial)}}(i, j) = \max(F_R(i, j), F_G(i, j), F_B(i, j))$$

$$F_{\text{max\_spatial}}(1, 1) = \max(1, 2, 3) = 3$$

$$F_{\text{max\_spatial}}(1, 2) = \max(2, 4, 6) = 6$$

$$F_{\text{max\_spatial}}(1, 3) = \max(3, 6, 9) = 9$$

$$F_{\text{max\_spatial}}(1, 4) = \max(4, 8, 12) = 12$$

$$F_{\text{max\_spatial}}(1, 5) = \max(5, 10, 15) = 15$$

:

$$F_{\text{max\_spatial}}(5, 1) = \max(21, 42, 63) = 63$$

$$F_{\text{max\_spatial}}(5, 2) = \max(22, 44, 66) = 66$$

$$F_{\text{max\_spatial}}(5, 3) = \max(23, 46, 69) = 69$$

$$F_{\text{max\_spatial}}(5, 4) = \max(24, 48, 72) = 72$$

$$F_{\text{max\_spatial}}(5, 5) = \max(25, 50, 75) = 75$$

Hasil dari matriks yang dihasilkan dengan **Channel-wise Average Pooling**:

$$F_{\text{avg\_spatial}} = \begin{bmatrix} 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \\ 32 & 34 & 36 & 38 & 40 \\ 42 & 44 & 46 & 48 & 50 \end{bmatrix}$$

Hasil dari matriks yang dihasilkan dengan **Channel-wise Max Pooling**:

$$F_{\text{max\_spatial}} = \begin{bmatrix} 3 & 6 & 9 & 12 & 15 \\ 18 & 21 & 24 & 27 & 30 \\ 33 & 36 & 39 & 42 & 45 \\ 48 & 51 & 54 & 57 & 60 \\ 63 & 66 & 69 & 72 & 75 \end{bmatrix}$$

Setelah didapatkan hasil dari *Channel-wise Max Pooling* dan *Channel-wise Average Pooling*, proses selanjutnya menggabungkan (*concatenate*) 2 hasil tersebut dan melakukan konvolusi dengan standard konvolusi dengan ukuran  $7 \times 7$ , untuk prosesp perhitungan matematisnya sebagai berikut :

Misalkan inisiasi 2 kernel untuk 2 channel yang tersusun dari konvolusi ukuran  $7 \times 7$ , karna channel yang dihasilkan harus bentuknya sama seperti ukuran input, maka dilakukan penyetelan padding "same" agar hasil setelah konvolusi sama dengan input.

$$\begin{bmatrix} -3 & -2 & -1 & 0 & 1 & 2 & 3 \\ -4 & -3 & -2 & 0 & 2 & 3 & 4 \\ -5 & -4 & -3 & 0 & 3 & 4 & 5 \\ -6 & -5 & -4 & 0 & 4 & 5 & 6 \\ -4 & -4 & -3 & 0 & 3 & 4 & 4 \\ -4 & -3 & -2 & 0 & 2 & 3 & 4 \\ -3 & -2 & -1 & 0 & 1 & 2 & 3 \end{bmatrix}$$

Matriks hasil *concatenate* dari *Channel-wise Max Pooling* dan *Channel-Wise Average Pooling* :

$$F_{\text{concat}} = [F_{\text{avg\_spatial}}, F_{\text{max\_spatial}}] = \left[ \begin{bmatrix} 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \\ 32 & 34 & 36 & 38 & 40 \\ 42 & 44 & 46 & 48 & 50 \end{bmatrix}, \begin{bmatrix} 3 & 6 & 9 & 12 & 15 \\ 18 & 21 & 24 & 27 & 30 \\ 33 & 36 & 39 & 42 & 45 \\ 48 & 51 & 54 & 57 & 60 \\ 63 & 66 & 69 & 72 & 75 \end{bmatrix} \right]$$

Hasil *Concencation* berupa tensor akan dikonvolusi dengan kernel  $7 \times 7$ , tapi sebelum itu dilakukan padding, hasil matriks tensor yang terbentuk sebagai berikut:

$$\begin{bmatrix} 1825., & 1730., & 380., & -1050., & -1755. \\ 3040., & 2740., & 490., & -1860., & -2950. \\ 3660., & 3170., & 520., & -2240., & -3570. \\ 3890., & 3290., & 490., & -2410., & -3800. \\ 3670., & 3080., & 380., & -2380., & -3590. \end{bmatrix}$$

Kemudian dari setiap *value* didalam produk tensor yang sudah melewati konvolusi diterapkan fungsi aktivasi sigmoid, matriks tensor akan berubah menjadi:

$$\begin{bmatrix} 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \end{bmatrix}$$

Dan ini merupakan hasil dari SAM (*Spatial Attention Module*) yang akan dilanjutkan operasinya pada setiap fitur atensi kanal.k

### B.0.3 Hasil Perhitungan CBAM (CAM + SAM)

Dengan mengikuti rumus berdasarkan rumus yang ada di bab 2 untuk mencari fitur yang *refined* ( $F''$ ) dapat dihitung sebagai berikut :

$$F' = M_c(F) \otimes F, \quad (B.1)$$

$$F'' = M_s(F') \otimes F' \quad (B.2)$$

Diketahui hasil dari  $M_c(\mathbf{F})$  dan  $M_s(\mathbf{F})$  berdasarkan perhitungan sebelumnya secara berurutan sebagai berikut :

$$M_c = \begin{bmatrix} [1] & [1] & [1] \end{bmatrix}$$

$$M_s(F) = \begin{bmatrix} 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \end{bmatrix}$$

Maka dapat dihitung, untuk  $F' = M_c(\mathbf{F}) \otimes \mathbf{F}$  :

Hasil dari  $M_c(\mathbf{F})$  di *broadcast* kedalam setiap index matriks fitur ( $\mathbf{F}$ ) dengan metode pengkalian secara elemen-per-elemen (*element-wise multiplication*).

$$F'_{red} = [1] \otimes \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{bmatrix}$$

$$F'_{green} = [1] \otimes \begin{bmatrix} 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \\ 32 & 34 & 36 & 38 & 40 \\ 42 & 44 & 46 & 48 & 50 \end{bmatrix} = \begin{bmatrix} 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \\ 32 & 34 & 36 & 38 & 40 \\ 42 & 44 & 46 & 48 & 50 \end{bmatrix}$$

$$F'_{blue} = [1] \otimes \begin{bmatrix} 3 & 6 & 9 & 12 & 15 \\ 18 & 21 & 24 & 27 & 30 \\ 33 & 36 & 39 & 42 & 45 \\ 48 & 51 & 54 & 57 & 60 \\ 63 & 66 & 69 & 72 & 75 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 9 & 12 & 15 \\ 18 & 21 & 24 & 27 & 30 \\ 33 & 36 & 39 & 42 & 45 \\ 48 & 51 & 54 & 57 & 60 \\ 63 & 66 & 69 & 72 & 75 \end{bmatrix}$$

Hasil dari seluruh channel diatas akan dilanjutkan pada proses perhitungan untuk mendapatkan area spasial  $F'' = M_s(F') \otimes F'$  sebagai berikut:

Perhitungan dilakukan setelah mendapatkan seluruh hasil dari  $F'$  di setiap channel. Hasil dari  $M_s(F')$  sudah dihitung sebelumnya akan diterapkan didalam setiap channel dengan cara pengkalian secara elemen-per-elemen (*element-wise multiplication*) untuk mendapatkan  $F''$  yang merupakan (*refined feature*):

$$F''_{red} = \begin{bmatrix} 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \end{bmatrix} \otimes \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 6 & 7 & 8 & 0 & 0 \\ 11 & 12 & 13 & 0 & 0 \\ 16 & 17 & 18 & 0 & 0 \\ 21 & 22 & 23 & 0 & 0 \end{bmatrix}$$

$$F''_{green} = \begin{bmatrix} 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \end{bmatrix} \otimes \begin{bmatrix} 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \\ 32 & 34 & 36 & 38 & 40 \\ 42 & 44 & 46 & 48 & 50 \end{bmatrix} = \begin{bmatrix} 2 & 4 & 6 & 0 & 0 \\ 12 & 14 & 16 & 0 & 0 \\ 22 & 24 & 26 & 0 & 0 \\ 32 & 34 & 36 & 0 & 0 \\ 42 & 44 & 46 & 0 & 0 \end{bmatrix}$$

$$F''_{blue} = \begin{bmatrix} 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \\ 1. & 1. & 1. & 0. & 0. \end{bmatrix} \otimes \begin{bmatrix} 3 & 6 & 9 & 12 & 15 \\ 18 & 21 & 24 & 27 & 30 \\ 33 & 36 & 39 & 42 & 45 \\ 48 & 51 & 54 & 57 & 60 \\ 63 & 66 & 69 & 72 & 75 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 9 & 0 & 0 \\ 18 & 21 & 24 & 0 & 0 \\ 33 & 36 & 39 & 0 & 0 \\ 48 & 51 & 54 & 0 & 0 \\ 63 & 66 & 69 & 0 & 0 \end{bmatrix}$$

## LAMPIRAN C

### Struktur Arsitektur Res4Net-CBAM

**Tabel C.1** Ringkasan Arsitektur Res4Net-CBAM

<b>Layer (type)</b>	<b>Output Shape</b>	<b>Param #</b>
Conv2d-1	[ -1, 64, 112, 112]	9,408
BatchNorm2d-2	[ -1, 64, 112, 112]	128
LeakyReLU-4	[ -1, 64, 112, 112]	0
MaxPool2d-3	[ -1, 64, 56, 56]	0
Conv2d-5	[ -1, 128, 56, 56]	8,192
BatchNorm2d-6	[ -1, 128, 56, 56]	256
LeakyReLU-7	[ -1, 128, 56, 56]	0
Conv2d-8	[ -1, 128, 28, 28]	147,456
BatchNorm2d-9	[ -1, 128, 28, 28]	256
LeakyReLU-10	[ -1, 128, 28, 28]	0
Conv2d-11	[ -1, 128, 28, 28]	16,384
BatchNorm2d-12	[ -1, 128, 28, 28]	256
Conv2d-13	[ -1, 128, 28, 28]	8,192
BatchNorm2d-14	[ -1, 128, 28, 28]	256
ResidualBlock-15	[ -1, 128, 28, 28]	0
Flatten-16	[ -1, 128]	0
Linear-17	[ -1, 8]	1,032
ReLU-18	[ -1, 8]	0
Linear-19	[ -1, 128]	1,152
Flatten-20	[ -1, 128]	0
Linear-21	[ -1, 8]	1,032
ReLU-22	[ -1, 8]	0
Linear-23	[ -1, 128]	1,152
ChannelGate-24	[ -1, 128, 28, 28]	0
ChannelPool-25	[ -1, 2, 28, 28]	0
Conv2d-26	[ -1, 1, 28, 28]	98
BatchNorm2d-27	[ -1, 1, 28, 28]	2
BasicConv-28	[ -1, 1, 28, 28]	0
SpatialGate-29	[ -1, 128, 28, 28]	0
CBAM-30	[ -1, 128, 28, 28]	0
LeakyReLU-31	[ -1, 128, 28, 28]	0
Conv2d-32	[ -1, 256, 28, 28]	32,768
BatchNorm2d-33	[ -1, 256, 28, 28]	512
LeakyReLU-34	[ -1, 256, 28, 28]	0
Conv2d-35	[ -1, 256, 14, 14]	589,824
BatchNorm2d-36	[ -1, 256, 14, 14]	512
LeakyReLU-37	[ -1, 256, 14, 14]	0
Conv2d-38	[ -1, 256, 14, 14]	65,536
BatchNorm2d-39	[ -1, 256, 14, 14]	512
Conv2d-40	[ -1, 256, 14, 14]	32,768

<b>Layer (type)</b>	<b>Output Shape</b>	<b>Param #</b>
BatchNorm2d-41	[ -1, 256, 14, 14]	512
ResidualBlock-42	[ -1, 256, 14, 14]	0
Flatten-43	[ -1, 256]	0
Linear-44	[ -1, 16]	4,112
ReLU-45	[ -1, 16]	0
Linear-46	[ -1, 256]	4,352
Flatten-47	[ -1, 256]	0
Linear-48	[ -1, 16]	4,112
ReLU-49	[ -1, 16]	0
Linear-50	[ -1, 256]	4,352
ChannelGate-51	[ -1, 256, 14, 14]	0
ChannelPool-52	[ -1, 2, 14, 14]	0
Conv2d-53	[ -1, 1, 14, 14]	98
BatchNorm2d-54	[ -1, 1, 14, 14]	2
BasicConv-55	[ -1, 1, 14, 14]	0
SpatialGate-56	[ -1, 256, 14, 14]	0
CBAM-57	[ -1, 256, 14, 14]	0
LeakyReLU-58	[ -1, 256, 14, 14]	0
Conv2d-59	[ -1, 512, 14, 14]	131,072
BatchNorm2d-60	[ -1, 512, 14, 14]	1,024
LeakyReLU-61	[ -1, 512, 14, 14]	0
Conv2d-62	[ -1, 512, 7, 7]	2,359,296
BatchNorm2d-63	[ -1, 512, 7, 7]	1,024
LeakyReLU-64	[ -1, 512, 7, 7]	0
Conv2d-65	[ -1, 512, 7, 7]	262,144
BatchNorm2d-66	[ -1, 512, 7, 7]	1,024
Conv2d-67	[ -1, 512, 7, 7]	131,072
BatchNorm2d-68	[ -1, 512, 7, 7]	1,024
ResidualBlock-69	[ -1, 512, 7, 7]	0
Flatten-70	[ -1, 512]	0
Linear-71	[ -1, 32]	16,416
ReLU-72	[ -1, 32]	0
Linear-73	[ -1, 512]	16,896
Flatten-74	[ -1, 512]	0
Linear-75	[ -1, 32]	16,416
ReLU-76	[ -1, 32]	0
Linear-77	[ -1, 512]	16,896
ChannelGate-78	[ -1, 512, 7, 7]	0
ChannelPool-79	[ -1, 2, 7, 7]	0
Conv2d-80	[ -1, 1, 7, 7]	98
BatchNorm2d-81	[ -1, 1, 7, 7]	2
BasicConv-82	[ -1, 1, 7, 7]	0
SpatialGate-83	[ -1, 512, 7, 7]	0
CBAM-84	[ -1, 512, 7, 7]	0
LeakyReLU-85	[ -1, 512, 7, 7]	0
Conv2d-86	[ -1, 512, 7, 7]	262,144
BatchNorm2d-87	[ -1, 512, 7, 7]	1,024

<b>Layer (type)</b>	<b>Output Shape</b>	<b>Param #</b>
LeakyReLU-88	[ -1, 512, 7, 7 ]	0
Conv2d-89	[ -1, 512, 4, 4 ]	2,359,296
BatchNorm2d-90	[ -1, 512, 4, 4 ]	1,024
LeakyReLU-91	[ -1, 512, 4, 4 ]	0
Conv2d-92	[ -1, 512, 4, 4 ]	262,144
BatchNorm2d-93	[ -1, 512, 4, 4 ]	1,024
ResidualBlock-94	[ -1, 512, 4, 4 ]	0
Flatten-95	[ -1, 512 ]	0
Linear-96	[ -1, 32 ]	16,416
ReLU-97	[ -1, 32 ]	0
Linear-98	[ -1, 512 ]	16,896
Flatten-99	[ -1, 512 ]	0
Linear-100	[ -1, 32 ]	16,416
ReLU-101	[ -1, 32 ]	0
Linear-102	[ -1, 512 ]	16,896
ChannelGate-103	[ -1, 512, 4, 4 ]	0
ChannelPool-104	[ -1, 2, 4, 4 ]	0
Conv2d-105	[ -1, 1, 4, 4 ]	98
BatchNorm2d-106	[ -1, 1, 4, 4 ]	2
BasicConv-107	[ -1, 1, 4, 4 ]	0
SpatialGate-108	[ -1, 512, 4, 4 ]	0
CBAM-109	[ -1, 512, 4, 4 ]	0
LeakyReLU-110	[ -1, 512, 4, 4 ]	0
AdaptiveAvgPool2d-111	[ -1, 512, 1, 1 ]	0
Linear-112	[ -1, 1024 ]	525,312
Dropout-113	[ -1, 1024 ]	0
Linear-114	[ -1, 3 ]	3,075
<b>Total params</b>	7,371,395	
<b>Trainable params</b>	7,371,395	
<b>Non-trainable params</b>	0	
<b>Input size (MB)</b>	0.57	
<b>Forward/backward pass size (MB)</b>	48.77	
<b>Params size (MB)</b>	28.12	
<b>Estimated Total Size (MB)</b>	77.47	