KLASIFIKASI SPESIES BUNGA EDELWEIS MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DAN PENGOPTIMAL *ADAPTIVE MOMENT ESTIMATION* (ADAM)

SKRIPSI

Diajukan untuk menempuh Ujian Sarjana
pada Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Padjadjaran

WIBI ANTO
NPM 140110200025

UNIVERSITAS PADJADJARAN
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
PROGRAM STUDI S-1 MATEMATIKA
JATINANGOR
2024

LEMBAR PENGESAHAN

JUDUL : KLASIFIKASI SPESIES BUNGA EDELWEIS MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DAN PENGOPTIMAL *ADAPTIVE MOMENT ESTIMATION* (ADAM)

PENULIS : WIBI ANTO
NPM : 140110200025

Jatinangor, Mei 2024

Menyetujui,

Pembimbing Utama	Pembimbing Pendamping
Herlina Napitupulu M.Sc., Ph.D.	Nurul Gusriani S.Si., M.Si.
NIP 198804082019032015	NIP 197008181998032001

Mengetahui,
Ketua Program Studi S-1 Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Padjadjaran

Edi Kurniadi, S.Si., M.Si., Ph.D
NIP 198104182008121003

ABSTRAK

Wibi Anto
NPM 140110200025

Indonesia adalah negara tropis yang memiliki banyak kekayaan hayati. Pemerintah mengatur perlakuan tumbuhan yang dilindungi untuk menjaga kelestariannya. Salah satu karakteristik hewan dan tumbuhan dilindungi adalah endemik, yaitu suatu organisme hanya dapat ditemukan di wilayah tertentu. *Anaphalis javanica* merupakan salah satu spesies edelweis endemik yang ada di Indonesia. Pemerintah telah melakukan berbagai upaya untuk melestarikan bunga edelweis dengan membuat peraturan perundang-undangan dan menetapkan kawasan konservasi dan pelestarian. Pelestarian bunga edelweis secara khusus dilakukan di Desa Wonokitri dengan membudidayakan berbagai bunga edelweis termasuk beberapa jenis yang tidak dilindungi. Spesies bunga edelweis sangat beragam dan memiliki kemiripan warna dan bentuk yang serupa. Perbedaan perlakuan bunga liar dan hasil budidaya juga menambah keberagamannya. Identifikasi spesies bunga edelweis secara akurat menjadi tantangan tersendiri karena perlunya ketelitian yang tinggi. Permasalahan tersebut dapat diatasi dengan melakukan penelitian klasifikasi edelweis menggunakan *Convolutional Neural Network* dengan pengoptimal Adam. Penelitian dilakukan menggunakan sebanyak 3498 data gambar *train* yang kemudian *dibagi ke da*lam data *train* dan *validation* serta 1050 data gambar test yang terbagi ke dalam tiga kelas. Kelas data dalam penelitian ini merupakan nama-nama spesies edelweis yang ada pada dataset yaitu *Anaphalis javanica, Leontopodium alpinum,* dan *Leucogenes grandiceps.* Penelitian ini bertujuan untuk membangun algoritma CNN dengan pengoptimal Adam, memperoleh hasil klasifikasi dan performa akurasi model, dan memperoleh nilai *learning rate* terbaik yang menghasilkan nilai akurasi tertinggi.

Kata kunci: Klasifikasi Gambar; Bunga Edelweis; CNN; Pengoptimal Adam.

ABSTRACT

Wibi Anto
NPM 140110200025

Indonesia is a tropical country that has a lot of biological wealth. The government regulates the treatment of protected plants to maintain their sustainability. One of the characteristics of protected animals and plants is endemic, which means that an organism can only be found in certain areas. Anaphalis javanica is one of the endemic edelweiss species in Indonesia. The government has made various efforts to conserve edelweiss flowers by making laws and regulations and establishing conservation and preservation areas. edelweiss conservation is specifically carried out in Wonokitri Village by cultivating various edelweiss flowers including several species that are not protected. edelweiss flower species are very diverse and have similarities in color and shape. Differences in the treatment of wild and cultivated flowers also add to the diversity. Accurately identifying edelweiss flower species is a challenge because of the need for high accuracy. To try to overcome this problem, edelweiss classification research will be conducted using Convolutional Neural Network with Adam optimizer. The research is conducted using 3498 train image data which will then be divided into train and validation data and 1050 test image data which are divided into three classes. The data classes in this study are the names of edelweiss species in the dataset, namely Anaphalis javanica, Leontopodium alpinum, and Leucogenes grandiceps. This research aims to build a CNN algorithm with Adam optimizer, obtain classification results and model accuracy performance, and obtain the best learning rate value that produces the highest accuracy value.

Keywords*: Image Classification; Edelweiss flower; CNN; Adam optimizer*.

KATA PANGANTAR

Segala puji dan syukur kepada Allah SWT, karena berkat rahmat, petunjuk, dan karunia-Nya, penulis dapat menyelesaikan skripsi yang berjudul "Klasifikasi Spesies Bunga Edelweis Menggunakan *Convolutional Neural Network* dan Pengoptimal *Adaptive Moment Estimation* (Adam)" Skripsi ini disusun untuk memenuhi salah satu syarat ujian sarjana pada Program Studi S-1 Matematika di Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Padjadjaran.

Skripsi ini tidak dapat selesai tanpa adanya dukungan dan bantuan dari berbagai pihak. Oleh karena itu pada kesempatan ini, penulis mengucapkan terima kasih kepada Ibu Herlina Napitupulu M.Sc., Ph.D., selaku Dosen Pembimbing Utama dan Ibu Nurul Gusriani S.Si., M.Si., selaku Dosen Pembimbing Pendamping yang telah memberikan bimbingan, bantuan, dan dorongan yang sangat berharga kepada penulis dalam proses penyusunan skripsi ini. Selain itu, penulis juga ingin mengucapkan terima kasih kepada:

1. Prof. Dr. Iman Rahayu, S.Si., M.Si., selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Padjadjaran.
2. Dr. Ema Carnia, M.Si., selaku Kepala Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Padjadjaran.
3. Edi Kurniadi, S.Si., M.Si., Ph.D., selaku Ketua Program Studi S-1 Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Padjadjaran.
4. Dr. Alit Kartiwa S.Si., M.Si., selaku Dosen Wali penulis.

5. Seluruh Civitas Akademika Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Padjadjaran.
6. Kedua orang tua dan kakak-kakak penulis yang selalu memberikan dukungan penuh, motivasi serta doa yang tidak pernah terputus kepada penulis.
7. Seluruh pihak yang tidak dapat penulis sebutkan satu persatu yang telah memberikan banyak dukungan dan doa kepada penulis.

Semoga skripsi ini dapat bermanfaat bagi diri penulis sendiri, orang-orang yang membacanya, dan bahkan masyarakat secara luas lewat segala ilmu dan gagasannya.

Jatinangor, Mei 2024

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR NOTASI

a : indeks baris pada suatu filter

b : indeks kolom pada suatu filter

c : indeks saluran warna pada suatu filter

$\overrightarrow{d^{(L)}}$: vektor output *dense layer* pada *layer* ke-L

d_j : elemen output *dense layer* di *layer* ke-j

i : indeks *learning rate*

j : indeks *neuron* pada *layer* saat ini

k : indeks *neuron* pada *layer* sebelumnya

m : banyak data pada suatu dataset

n : banyak kelas pada dataset train

p : indeks data pada dataset

q : indeks baris pada output *feature map*

r : indeks kolom pada output *feature map*

s : indeks saluran warna pada output *feature map*

t : indeks iterasi atau indeks *mini-batch*

$\vec{w}$: vektor kumpulan parameter bobot atau *weight*

$w_{jk}^{(L)}$: bobot penghubung antara *neuron* ke-j pada layer ke-L dan *neuron* ke-k pada layer sebelumnya

$w_{a,b,c}$: elemen tensor filter baris ke-a, kolom ke-b, dan saluran warna ke-c

x_k : nilai input untuk *dense layer* dari *neuron* pada *layer* sebelumnya

$x_{q,r,s}$: nilai input berupa elemen tensor dari gambar dengan baris ke-q, kolom ke-r, dan saluran warna ke-s

$\hat{y}$: hasil prediksi gambar

y : label gambar sebenarnya pada data *train*

bs : ukuran *batch* (*batch size*)

lr : *learning rate*

lr_i : *learning rate* ke-i

$zc^{(L)}$: hasil perhitungan operasi konvolusi pada *layer* ke-L

$\overrightarrow{zf}$: output proses *flatten*

$zd_j^{(L)}$: output proses perhitugan *neuron* ke-j *dense layer* pada *layer* ke-L

$\boldsymbol{A}^{(L)}$: output dari *convolution layer* pada *layer* ke-L

$A_{q,r,s}$: elemen output *convolution layer* dengan indeks baris ke-q, kolom ke-r, dan saluran warna ke-s

L : indeks *layer*

N : banyak neuron pada *dense layer*

$P^{(L)}$: hasil *pooling layer* pada *layer* ke-L

Q : banyak baris pada *feature map*

R : banyak kolom pada *feature map*

S : banyak saluran warna pada *feature map*

T : banyak iterasi

$\boldsymbol{W^{(L)}}$: filter pada operasi konvolusi di layer ke-L

$\boldsymbol{X}$: dataset gambar

$\boldsymbol{X}_p$: data ke-p dari dataset $\boldsymbol{X}$ yang berupa tensor

$\boldsymbol{X}^{\{t\}}$: *mini-batch* ke-t dari dataset $\boldsymbol{X}$

$\boldsymbol{X}_p^{\{t\}}$: data ke-p dari *mini-batch* ke-t pada dataset $\boldsymbol{X}$

$\boldsymbol{Xtn}^{\{t\}}$: *mini-batch* ke-t dari *dataset train*

$\boldsymbol{Xts}^{\{t\}}$: *mini-batch* ke-t dari *dataset test*

$\boldsymbol{Xval}^{\{t\}}$: *mini-batch* ke-t dari *dataset validation*

$Y_{q,r,s}$: hasil operasi konvolusi baris ke-q kolom ke-r dan saluran warna ke-s

$\alpha_{ReLU}(z)$: fungsi aktivasi ReLU

$\alpha_{sm}(z)$: fungsi aktivasi Softmax

$\vec{\beta}$: vektor kumpulan parameter bias

$\beta^{(L)}$: bias filter dari *convolution layer* pada *layer* ke-L

$\beta_j^{(L)}$: bias neuron ke-j *dense layer* pada *layer* ke-L

γ_1 : *decay rate* untuk momentum, nilai adalah 0,9

γ_2 : *decay rate* untuk gradien, nilai adalah 0,999

ϵ : konstanta 10^{-8}

g_t : gradien saat iterasi ke-t

m_t : pembaruan momentum pertama

$\hat{m}_t$: koreksi bias momentum pertama

v_t : pembaruan momentum kedua

$\hat{v}_t$: koreksi bias momentum kedua

θ_t : pembaruan parameter model pada iterasi ke-t

BAB I

PENDAHULUAN

1.1 Latar Belakang

Indonesia adalah negara tropis yang memiliki banyak kekayaan alam termasuk kekayaan hayati. Sebagai upaya pelestarian dari beragamnya kekayaan alam di Indonesia, pemerintah membagi kawasan pelestarian menjadi taman nasional, taman hutan raya, dan taman wisata alam. Taman nasional adalah kawasan pelestarian alam yang mempunyai ekosistem asli, dikelola dengan sistem zonasi yang dimanfaatkan untuk tujuan penelitian, ilmu pengetahuan, pendidikan, menunjang budidaya, pariwisata, dan rekreasi (Presiden RI, 1990).

Setiap taman nasional memiliki karakteristik yang berbeda bergantung pada lokasi geografis. Perbedaan karakteristik tersebut diantaranya adalah ragam keanekaragaman hayati yang juga didukung dengan perbedaan topografi dari taman nasional. Taman nasional dengan topografi pegunungan cenderung memiliki keragaman hayati yang mampu beradaptasi dengan ketinggian dan suhu yang rendah sedangkan Taman nasional dengan topografi pesisir cenderung memiliki karakteristik hayati yang mampu beradaptasi dan berinteraksi dengan ekosistem laut.

Perbedaan karakteristik taman nasional membuat adanya beberapa hayati yang hanya dapat tumbuh pada lokasi tertentu, salah satunya adalah bunga Edelweis. Menurut Direktorat Jenderal Konservasi Sumber Daya Alam dan Ekosistem, bunga edelweis hanya dapat tumbuh di daerah pegunungan dan memerlukan sinar matahari penuh. Pemerintah melalui Kementerian LHK menyatakan salah satu spesies endemik bunga edelweis yaitu *Anaphalis javanica* sebagai jenis tumbuhan yang dilindungi (Menteri LHK, 2018). Bunga edelweis telah memenuhi kriteria sebagai tumbuhan yang dilindungi. Kriteria tumbuhan dan satwa yang dilindungi adalah: mempunyai populasi yang kecil, adanya penurunan yang tajam pada jumlah individu di alam, dan daerah penyebaran yang terbatas atau disebut juga dengan endemik (Presiden RI, 1999).

Upaya pelestarian bunga edelweis secara khusus dilakukan dengan membudidayakannya di Desa Wonokitri. Desa Wonokitri menjadi tempat budidaya bunga edelweis termasuk beberapa jenis yang tidak dilindungi (Anam, 2021; Kartika, 2023). Pengunjung secara khusus dapat membeli bibit bunga edelweis hasil budidaya untuk dapat ditanam maupun buket bunga sebagai hiasan di desa Wonokitri (Riani, 2024). Namun demikian, bunga edelweis liar terutama dengan jenis *Anaphalis javanica*, tetap menjadi tumbuhan yang dilindungi undang-undang.

Keberagaman spesies bunga edelweis menjadi bertambah dengan adanya perbedaan perlakuan antara bunga liar dan hasil budidaya dengan bentuk dan warna yang serupa (Malasari, 2022). Ketelitian sangat diperlukan dalam mengetahui jenis bunga edelweis agar tidak melanggar undang-undang dan mendapatkan sanksi yang

berat. Hal ini menghadirkan tantangan dalam identifikasi spesies bunga edelweis secara akurat.

Salah satu bidang *Machine Learning* yaitu *Computer Vision* mampu menjawab tantangan tersebut. Salah satu cabang *Computer Vision* yaitu *Image Classification* atau klasifikasi gambar dengan menggunakan *Convolutional Neural Network* (CNN) mampu melakukan pengenalan pola dan fitur visual yang rumit, sehingga memungkinkan untuk membedakan spesies-spesies bunga edelweis dengan lebih tepat.

Klasifikasi gambar menggunakan CNN secara umum memiliki dua buah tahapan pelatihan model yaitu proses ekstraksi fitur dan proses klasifikasi (Géron, 2019). Proses ekstraksi fitur bertujuan untuk memperoleh fitur yang ada pada gambar dan proses tersebut dilakukan dengan menggunakan operasi konvolusi dua dimensi dan operasi *max pooling* dua dimensi. Setelah mendapatkan hasil ekstraksi, selanjutnya dilakukan proses *flatten* untuk mengubah dimensi dari hasil ekstraksi fitur menjadi satu dimensi. Kemudian dilanjutkan dengan proses klasifikasi pada lapisan *fully-connected layer* berupa *dense layer*.

Penelitian mengenai klasifikasi gambar menggunakan CNN telah banyak dilakukan dengan berbagai objek. Alkaff & Prasetiyo (2022) mengaplikasikan CNN untuk klasifikasi penyakit tanaman tomat yang menyerang daun. Model CNN dibuat dengan mengoptimasi *hyperparameter* menggunakan *Hyperband* dan mencapai tingkat akurasi sebesar 95,69% pada proses *train,* 88,5% pada proses *validation,* dan 88,6% pada proses *test.*

Muhammad & Wibowo (2021) menggunakan CNN dengan arsitektur ResNet50v2 untuk mengklasifikasikan empat jenis tanaman Aglaonema menggunakan 1960 gambar. Eksperimen dalam penelitian ini dilakukan dengan mencoba melakukan *training* dengan menghilangkan *background* gambar, mengubah *learning rate,* dan mengubah nilai parameter *dropout* dan menghasilkan akurasi tertinggi pada proses *testing* sebesar 99% menggunakan gambar dengan *background* dan 71% menggunakan gambar tanpa background.

Penelitian klasifikasi bunga edelweis juga telah dilakukan oleh Malau & Mulyana (2022) dengan mengklasifikasikan dua jenis bunga edelweis yaitu *Anaphalis javanica* dan *Leontopodium alpinum* menggunakan metode *Linear Discriminant Analysis* (LDA). Model yang dibuat dalam penelitian tersebut mencapai tingkat akurasi 100% dengan menggunakan 1500 data gambar *train* pada saat proses *training* dan 99,77% dengan menggunakan 450 data gambar *test* pada proses *testing.*

Pada penelitian ini dilakukan klasifikasi gambar edelweis seperti yang dilakukan Malau & Mulyana (2022). Perbedaan penelitian ini dengan Malau & Mulyana (2022) adalah penggunaan metode CNN dengan pengoptimal Adam. Perbedaan penggunaan metode CNN pada penelitian ini dengan penelitian Muhammad & Wibowo (2021) terdapat pada penggunaan arsitektur CNN yang berbeda. Jumlah kelas yang digunakan pada penelitian ini juga lebih banyak dibandingkan dengan Malau & Mulyana (2022) dengan tiga buah kelas berupa jenis bunga edelweis yaitu *Anaphalis javanica, Leontopodium alpinum,* dan *Leucogenes*

grandiceps. Penelitian ini menggunakan 3498 data gambar *train* yang kemudian dibagi ke dalam data *train* dan *validation* serta 1050 data gambar *test*.

1.2 Identifikasi Masalah

Adapun identifikasi masalah pada penelitian ini adalah sebagai berikut:

1. Bagaimana mengimplementasikan (algoritma) CNN dengan pengoptimal *Adaptive Moment Estimation* dalam mengklasifikasikan spesies bunga Edelweis.
2. Bagaimana hasil klasifikasi spesies bunga edelweis menggunakan model *CNN* dengan pengoptimal Adam?
3. Berapa nilai *learning rate* terbaik yang menghasilkan nilai akurasi tertinggi?

1.3 Batasan Masalah

Batasan permasalahan dalam penelitian ini adalah:

1. Klasifikasi spesies bunga edelweis berdasarkan gambar bunga dilakukan menggunakan model *Machine Learning* yaitu *Convolutional Neural Network* dan pengoptimal menggunakan *Adaptive Moment Estimation* (Adam).
2. Besaran parameter *learning rate* yang digunakan adalah $10^{-i}, i = 1, 2, 3, 4$
3. Performa model *Machine Learning* dianalisa menggunakan metrik akurasi.
4. Data yang digunakan dalam penelitian ini adalah gambar-gambar dari bunga edelweis yang diambil dari *Kaggle* yang terdiri dari tiga spesies bunga edelweis yang berbeda.

5. Bahasa pemrograman yang digunakan adalah bahasa pemrograman Python menggunakan *Google Colaboratory*.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut:

1. Membangun algoritma CNN dengan pengoptimal *Adaptive Moment Estimation* untuk mengklasifikasikan spesies bunga Edelweis.
2. Memperoleh hasil klasifikasi dan performa akurasi model.
3. Memperoleh nilai *learning rate* terbaik yang menghasilkan nilai akurasi tertinggi.

1.5 Kegunaan Penelitian

Kegunaan dari penelitian ini adalah sebagai berikut:

1. Model klasifikasi spesies bunga edelweis menggunakan *Convolutional Neural Network* dan pengoptimal Adam dapat digunakan untuk mengetahui spesies bunga edelweis hanya dengan menggunakan gambar.
2. Alat bantu untuk pembaca khususnya para pendaki maupun masyarakat umum dalam mengenali dan mengidentifikasi bunga edelweis guna mendukung pelestarian bunga Edelweis.

1.6 Metodologi Penelitian

Metodologi Penelitian terdiri dari studi literatur dan eksperimental:

1. Studi literatur dilakukan dengan mempelajari teori terkait *Computer Vision,* klasifikasi gambar, algoritma Adam, dan *Convolutional Neural Network*

dari berbagai sumber berupa buku, jurnal, dan artikel yang tersedia secara daring.

2. Studi Eksperimental dilakukan dengan membuat arsitektur model CNN dengan pengoptimal Adam serta mencari nilai parameter *learning rate* terbaik untuk klasifikasi jenis bunga edelweis menggunakan bahasa pemrograman Python pada *Google Colaboratory.*

1.7 Sistematika Penulisan

Sistematika penulisan pada skripsi ini adalah sebagai berikut:

BAB I PENDAHULUAN, pada bab ini dijelaskan mengenai latar belakang, identifikasi masalah, batasan masalah, tujuan penelitian, kegunaan penelitian, metodologi penelitian, dan sistematika penulisan.

BAB II LANDASAN TEORI, pada bab ini dijelaskan mengenai teori-teori yang menjadi acuan dasar dalam penelitian yaitu klasifikasi gambar, pra-pemrosesan gambar*, Convolution Neural Network, convolution layer* dua dimensi*, pooling layer* dua dimensi*, flatten layer, dense layer,* pengoptimal, metrik *accuracy*, *k-fold cross validation,* TensorFlow, Matplotlib, dan *Google Colaboratory.*

BAB III OBJEK DAN METODE PENELITIAN, pada bab ini berisi tentang objek penelitian, metode penelitian, dan alur penelitian.

BAB IV HASIL DAN PEMBAHASAN, pada bab ini berisi pengolahan data gambar dan hasil penelitian yang dilakukan.

BAB V SIMPULAN DAN SARAN, pada bab ini berisi simpulan dan saran dari pembahasan penelitian yang telah dilakukan untuk peneliti selanjutnya.

BAB II

LANDASAN TEORI

2.1 Klasifikasi Gambar

Klasifikasi gambar atau *Image Classification* adalah proses kategorisasi dan pemberian label dari sekumpulan piksel atau vektor dari suatu gambar berdasarkan aturan tertentu (Shinozuka & Mansouri, 2009). Terdapat dua jenis metode klasifikasi yaitu *supervised* dan *unsupervised.* Metode *unsupervised* adalah proses klasifikasi yang sepenuhnya otomatis tanpa menggunakan *training data* sedangkan metode *supervised* adalah proses klasifikasi menggunakan *training data* dan mengelompokkan ke dalam kategori yang telah dipilih sebelumnya (Shinozuka & Mansouri, 2009).

2.2 Pra-pemrosesan gambar

Pra-pemrosesan gambar atau *Image Preprocessing* adalah suatu metode untuk mengubah data gambar mentah yang mengandung *noise* dan nilai yang tidak sesuai menjadi data gambar yang bersih dan siap digunakan (Chaki & Dey, 2018). Chaki & Dey (2018) menyatakan bahwa *noise* ataupun nilai yang tidak sesuai dapat diatasi dengan beberapa cara diantarnya: *image correction*, *image enhancement, image restoration,* dan *image compression*. *Image compression* atau kompresi gambar digunakan untuk mengurangi ukuran gambar baik dengan mengurangi dimensi maupun kualitas gambar. Salah satu cara untuk menurunkan kualitas gambar dan menyeragamkan ukuran piksel adalah dengan menggunakan *library*

PIL. Algoritma untuk melakukan kompresi gambar dengan menggunakan *library* PIL dinyatakan sebagai berikut.

1. Inisiasi lebar gambar baru dalam satuan piksel = 512 px
2. Ambil ukuran lebar dan tinggi gambar asli
3. Hitung rasio gambar asli
4. Cari tinggi gambar baru dengan membagi lebar gambar baru dengan rasio gambar asli
5. Simpan dengan kualitas gambar sebesar 85%

Selain menggunakan *library* PIL, Pra-pemrosesan gambar juga dapat dilakukan menggunakan TensorFlow menggunakan fungsi *ImageDataGenerator* dan beberapa metode dalam fungsi tersebut sesuai dengan kebutuhan seperti *flow_from_directory* (TensorFlow Developer, 2024). Setidaknya terdapat satu pra-pemrosesan gambar dalam *neural network* yaitu mengubah gambar menjadi Tensor. Tensor adalah objek matematika yang menggambarkan hubungan linear antar kumpulan data multidimensi dan merupakan generalisasi dari skalar, vektor, dan matriks (Jain, 2023). Beberapa jenis pra-pemrosesan gambar lainnya yang menggunakan TensorFlow diantaranya adalah penyeragaman dimensi, normalisasi nilai piksel, pembagian data *train* dan *validation,* pembagian *batch,* dll.

Pembagian *batch* menjadi *mini-batch* dilakukan untuk menambah banyak proses iterasi untuk tahap optimasi parameter menggunakan pengoptimal Adam. Ukuran banyak data pada setiap *mini-batch* disebut dengan *batch size*. Misal $\boldsymbol{X}$ adalah dataset dengan jumlah data sebanyak m, $\boldsymbol{X_i}$ data ke-i dari dataset $\boldsymbol{X}$ yang berupa tensor dan *bs* adalah *batch size*.

$$\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_m] \tag{2.1}$$

maka jumlah proses iterasi dapat dihitung menggunakan persamaan sebagai berikut.

$$T = \left\lceil \frac{m}{bs} \right\rceil \tag{2.2}$$

mini-batch dinotasikan dengan $X^{\{t\}}$ dan dapat dinyatakan dengan sebagai berikut.

$$\boldsymbol{X}^{\{t\}} = \left[\boldsymbol{X}_{(bs\cdot(t-1))+1}, \boldsymbol{X}_{(bs\cdot t)+2} \dots, \boldsymbol{X}_{(t)\cdot bs}\right] \tag{2.3}$$

dimana $t = 1, 2, 3, \dots, T$ sehingga dataset X berubah menjadi sebagai berikut.

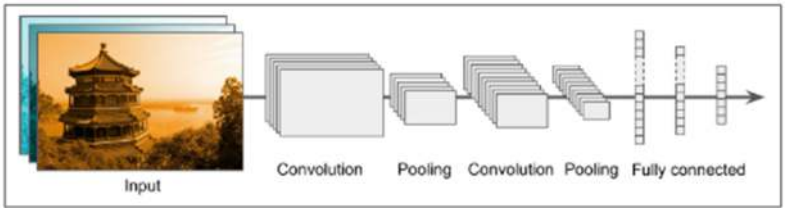
$$\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_{bs}; \boldsymbol{X}_{bs+1}, \dots, \boldsymbol{X}_{2bs}; \boldsymbol{X}_{2bs+1}, \dots, \boldsymbol{X}_{3bs}; \dots; \boldsymbol{X}_{(t-1)bs+1} \dots \boldsymbol{X}_{(m)}] \tag{2.4}$$

$$\boldsymbol{X} = \left[\boldsymbol{X}^{\{1\}}; \boldsymbol{X}^{\{2\}}; \dots; \boldsymbol{X}^{\{T\}}\right] \tag{2.5}$$

2.3 *Convolutional Neural Network*

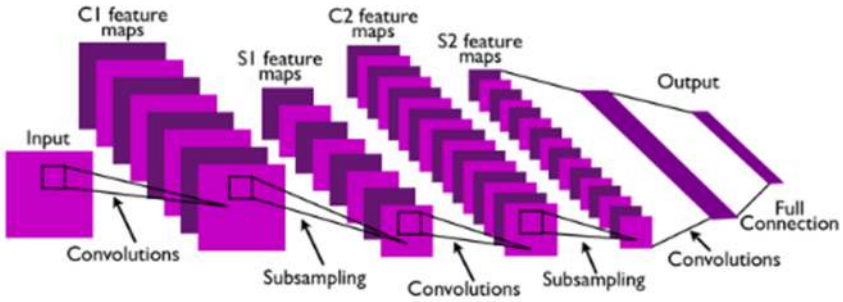
Komputer mengenal sebuah gambar sebagai suatu tensor dengan entri berupa angka yang merepresentasikan warna dari setiap piksel-nya. Pada proses melakukan klasifikasi gambar, angka-angka pada tensor tersebut dipelajari oleh mesin untuk mencari pola tertentu. Salah satu cara untuk mempelajari pola tersebut adalah menggunakan Convolutional Neural Network.

Convolutional Neural Network (CNN) adalah suatu jaringan syaraf tiruan dengan proses konvolusi yang muncul dari studi tentang visual korteks otak dan telah digunakan mulai dari tahun 1980an (Géron, 2019). Menurut Ketkar (2017), proses konvolusi merupakan suatu perhitungan matematika yang digunakan untuk melakukan ekstraksi fitur pada gambar. Secara umum, arsitektur CNN dapat dilihat pada Gambar 2.1.



Gambar 2.1 Arsitektur CNN secara umum (Géron, 2019)

CNN pada umumnya terdiri dari tiga jenis *layer* yaitu *Convolution Layer*, *Pooling Layer*, dan *Fully-connected Layer* (Géron, 2019). Menurut LeCun et al., (2010) *Convolution Network* merupakan arsitektur multitahap yang dapat dilatih dengan input dan output dari setiap tahap berupa sekumpulan *array* yang dinamakan *feature map* dan diikuti oleh modul klasifikasi. Ilustrasi tipe arsitektur CNN menurut LeCun et al., (2010) dapat dilihat pada Gambar 2.2.



Gambar 2.2 Tipe arsitektur CNN dengan dua tahap ekstraksi fitur (LeCun et al., 2010)

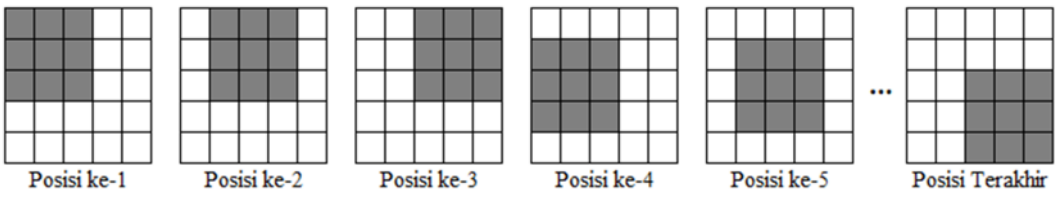
Dalam proses mempelajari data berupa gambar, CNN melakukan dua jenis propagasi yaitu *forward propagation* dan *back propagation*. Proses *forward propagation* memiliki tujuan untuk menghasilkan nilai *loss* atau kesalahan dari hasil klasifikasi terhadap nilai sebenarnya. Nilai *loss* kemudian akan digunakan untuk menghitung turunan parsial dari *loss* terhadap parameter yang akan diperbaharui yaitu *weight* dan *bias.* Pengoptimal kemudian memperbaharui nilai *weight* dan *bias* tersebut dengan tujuan untuk meminimumkan nilai *loss*.

2.4 *Convolution Layer* dua dimensi

Convolution Layer dua dimensi merupakan lapisan neuron dengan operasi konvolusi. Operasi konvolusi dapat dilakukan menggunakan beberapa alat salah satunya adalah menggunakan Python. Pada Python, membuat *convolution layer* dapat menggunakan *library* TensorFlow yang didalamnya telah menyediakan fungsi Conv2D.

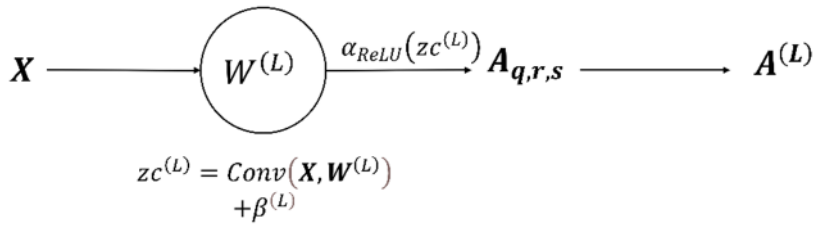
Pada fungsi Conv2D terdapat beberapa parameter yang dapat diubah diantaranya adalah *filters, kernel_size, activation,* dan *strides* (TensorFlow Developer, 2024a). *Filters* merupakan argumen yang menyatakan banyak tensor yang digunakan dalam operasi konvolusi untuk mengekstraksi fitur. Dimensi dari tensor tersebut dinamakan *kernel size*, beberapa *kernel size* yang sering digunakan adalah $(3,3)$, $(5,5)$, dan $(7,7)$.

Fungsi Conv2D memiliki parameter *strides* yang menunjukkan banyak perpindahan piksel setelah operasi konvolusi. Nilai $strides = 1$ berarti setiap operasi konvolusi dilakukan, terjadi perpindahan satu piksel untuk operasi konvolusi selanjutnya. Ilustrasi dari $strides = 1$ dengan filter W yang berukuran $(3,3)$ dapat dilihat pada gambar 2.3.



Gambar 2.3 Ilustrasi perpindahan piksel dengan strides = 1

Proses pada *convolution layer* diawali dengan memproses input *feature map* tensor ($\boldsymbol{X}$). Selanjutnya dilakukan Operasi konvolusi dan menghasilkan hasil $zc^{(L)}$ yang kemudian diproses dengan fungsi aktivasi pada parameter *activation* dan menghasilkan komponen *feature map* output $\boldsymbol{A_{q,r,s}}$. Proses konvolusi dilakukan berulang dengan memindahkan input *feature map* dengan satu *strides* sehingga menghasilkan $A^{(L)}$. Ilustrasi dari proses pada convolution layer dua dimensi dapat dilihat pada Gambar 2.4.



Gambar 2.4 Ilustrasi *convolution layer* dua dimensi

feature map hasil operasi konvolusi pada *layer* ke-L terhadap suatu tensor $\boldsymbol{X}$ dan filter $\boldsymbol{W}$ dengan bias β pada *layer* L dinyatakan sebagai berikut.

$$zc^{(L)} = Conv\left(\boldsymbol{X}, \boldsymbol{W}^{(L)}\right) + \beta^{(L)} \tag{2.6}$$

dimana operasi konvolusi dua dimensi dengan satu filter dan C saluran warna dinyatakan sebagai berikut.

$$Conv(\boldsymbol{X}, \boldsymbol{W}) = Y_{q,r,s} = \sum_{a=1}^{K_1} \sum_{b=1}^{K_2} \sum_{c=1}^{K_3} x_{q+a-1,r+b-1,s+c-1} * w_{a,b,c} \tag{2.7}$$

dimana $y_{q,r,s}$ merupakan hasil konvolusi baris ke-q kolom ke-r dan saluran warna ke-s, $\boldsymbol{X}$ merupakan input *feature map*, dan $\boldsymbol{W}$ merupakan filter dengan dimensi (K_1, K_2, K_3). Filter $\boldsymbol{W}$ didapat dengan menggunakan inisialisasi *glorot uniform*. *Glorot uniform* adalah suatu metode inisialisasi bobot (*weight*) dengan

mendistribusikan nilai-nilai acak dari distribusi *uniform* dengan tujuan untuk mempertahankan keseimbangan nilai-nilai pengali saat *forward propagation* dan *backpropagation* (Glorot & Bengio, 2010).

Rectified Linear Unit (ReLU) merupakan fungsi aktivasi yang sering digunakan dalam *Convolution Layer* (Ketkar, 2017). ReLU memproses nilai zc dengan mengubah setiap nilai yang bernilai negatif menjadi nol

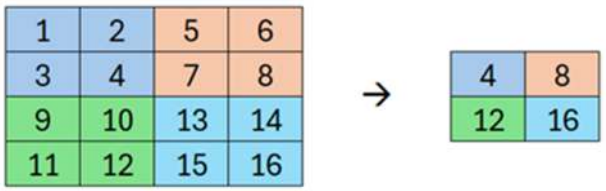
$$A_{q,r,s} = \alpha_{ReLU}\left(zc^{(L)}\right) = \max\left(0, zc^{(L)}\right) \tag{2.8}$$

sehingga output dari *layer* Conv2D ke-L dapat dilihat pada persamaan 2.9.

$$\boldsymbol{A}^{(L)} = \begin{bmatrix} [A_{1,1,1}, A_{1,1,2}, \dots, A_{1,1,s}] & [A_{1,2,1}, a_{1,2,2}, \dots, A_{1,2,s}] & \dots & [A_{1,r,1}, A_{1,r,2}, \dots, a_{1,r,s}] \\ [A_{2,1,1}, A_{2,1,2}, \dots, A_{2,1,s}] & [A_{2,2,1}, A_{2,2,2}, \dots, A_{2,2,s}] & \dots & [A_{2,r,1}, A_{2,r,2}, \dots, A_{2,r,s}] \\ \vdots & \vdots & \ddots & \vdots \\ [A_{q,1,1}, A_{q,1,2}, \dots, A_{q,1,s}] & [A_{q,2,1}, A_{q,2,2}, \dots, A_{q,2,s}] & \dots & [A_{q,r,1}, A_{q,r,2}, \dots, A_{q,r,s}] \end{bmatrix} \tag{2.9}$$

2.5 *Pooling layer* dua dimensi

Pooling Layer adalah suatu lapisan neuron dengan operasi penggabungan atau *pooling*. Operasi *pooling* dilakukan untuk memperkecil dimensi dengan menggabungkan nilai pada area tertentu sesuai dengan ukuran filter. Terdapat dua buah jenis operasi penggabungan yaitu *max pooling* dan *average pooling*. Operasi *pooling* dapat dilakukan dengan bantuan Python pada *library* TensorFlow yang telah menyediakan fungsi *MaxPooling2D* dengan parameter masukan *pool size* berupa *tuple* yang menyatakan dimensi filter. Ilustrasi operasi *max pooling* 2D dapat dilihat pada Gambar 2.5.



Gambar 2.5 Ilustrasi *max pooling* 2D dengan input $(4, 4)$ dan *pool size* $(2, 2)$

secara matematis, ouput dari *pooling layer* ke-L dengan input $\boldsymbol{X}$ dapat dinyatakan sebagai berikut.

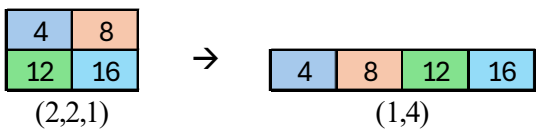
$$\boldsymbol{P}^{(L)} = MaxPooling2D(\boldsymbol{X}) \tag{2.10}$$

2.6 *Flatten layer*

Flatten layer adalah adalah suatu lapisan neuron yang memiliki operasi untuk membuat datar atau *flatten* dimensi dari tensor input yang masuk ke dalam *layer* tersebut. Misalkan terdapat $\boldsymbol{X}$ sebuah tensor dengan dimensi (Q, R, S), maka output dari *flatten layer* dapat dinyatakan sebagai berikut.

$$FL(\boldsymbol{X}) = \overrightarrow{zf} = \left[x_{q,r,s}\right] \tag{2.11}$$

dimana $\overrightarrow{zf}$ merupakan vektor baris, $q = 1, 2, \dots, Q; r = 1,2, \dots, R;$ dan $s = 1,2, \dots, S$. Ilustrasi proses *flatten* pada *flatten layer* dapat dilihat pada Gambar 2.6.



Gambar 2.6 Ilustrasi *flatten layer* dengan dimensi input $(2, 2, 1)$

2.7 *Dense layer*

Dense layer dalam arsitektur CNN berfungsi sebagai modul pengklasifikasi hasil konvolusi, serangkaian dense *layer* yang saling terhubung disebut dengan *fully-connected layer* (LeCun et al., 2010). TensorFlow telah menyediakan fungsi Dense untuk membuat *fully-connected layer.* Terdapat dua parameter dalam fungsi Dense yang sering diubah dalam pembuatan model CNN yaitu *units* dan *activation. Units* merujuk pada banyaknya sel neuron yang terdapat pada satu *layer* tersebut dan *activation* merujuk pada fungsi aktivasi yang digunakan. Secara matematis, *output* dari setiap *dense layer* dinyatakan sebagai berikut.

$$zd_j^{(L)} = \sum_{k=1}^{N} w_{jk}^{(L)} \times x_k^{(L-1)} + \beta_j^{(L)} \tag{2.12}$$

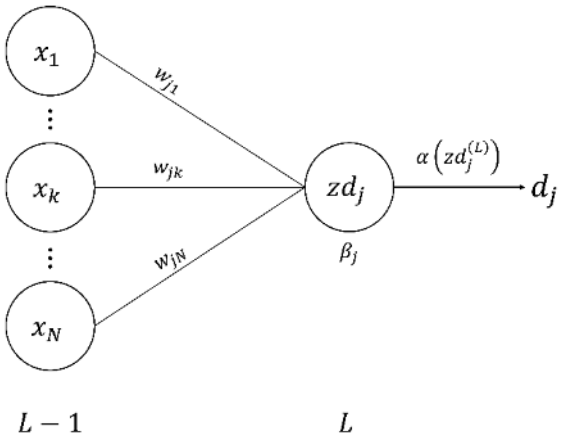
dimana $zd_j^{(L)}$ adalah nilai *neuron* ke-j pada *layer* ke-L, w_{jk} adalah bobot penghubung antara *neuron* ke-j pada *layer* L dan neuron ke-k pada *layer* sebelumnya, $x_j^{(L-1)}$ adalah nilai *neuron* ke-k dari output *layer* sebelumnya, N adalah jumlah banyaknya neuron pada *layer* sebelumnya, dan $\beta_j^{(L)}$ adalah bias pada *neuron* ke-j pada *layer* L. Secara umum, persamaan fungsi aktivasi dinyatakan sebagai berikut.

$$d_j = \alpha\left(zd_j^{(L)}\right) \tag{2.13}$$

sehingga output *dense layer* ke-L dengan fungsi aktivasi α dan memiliki sebanyak N *neuron* dapat dinyatakan sebagai vektor baris berikut.

$$\left(\vec{d}\right)^{(L)} = \left[d_1, d_2, \dots, d_j, \dots, d_N\right] \tag{2.14}$$

Ilustrasi dari proses perhitungan pada *dense layer* dapat dilihat pada Gambar 2.7 dimana terdapat nilai input dari layer sebelumnya yang dinotasikan dengan x dan diproses di dalam *dense layer* sehingga diperoleh zd_j. Nilai zd_j kemudian ditambahkan dengan bias β_j dan diproses dengan *activation function* sehingga diperoleh d_j.



Gambar 2.7 Ilustrasi *neuron* di *dense layer*

1. **Fungsi Aktivasi *Rectified Linear Unit***

Fungsi aktivasi dalam CNN adalah fungsi matematika yang diterapkan pada keluaran setiap neuron dalam jaringan. Fungsi ini menentukan apakah neuron harus diaktifkan atau tidak berdasarkan masukan yang diterimanya. Tujuan dari fungsi aktivasi adalah untuk memperkenalkan non-linearitas ke dalam keluaran neuron, sehingga memungkinkan jaringan untuk belajar dan memodelkan pola kompleks dalam data. Non-linearitas ini sangat penting agar jaringan dapat belajar dan membuat prediksi yang akurat. Fungsi aktivasi yang umum digunakan dalam CNN meliputi *Rectified Linear Unit* (ReLU) atau unit linear tereduksi, yang banyak digunakan karena kesederhanaan komputasinya dan efektivitas dalam pelatihan, dan fungsi aktivasi linear. Notasi matematis

dari *dense layer* dengan fungsi aktivasi *Rectified Linear Unit* (ReLU) dinyatakan sebagai berikut.

$$\alpha_{ReLU}\left(zd_j^{(L)}\right) = \max\left(0, zd_j^{(L)}\right) \tag{2.15}$$

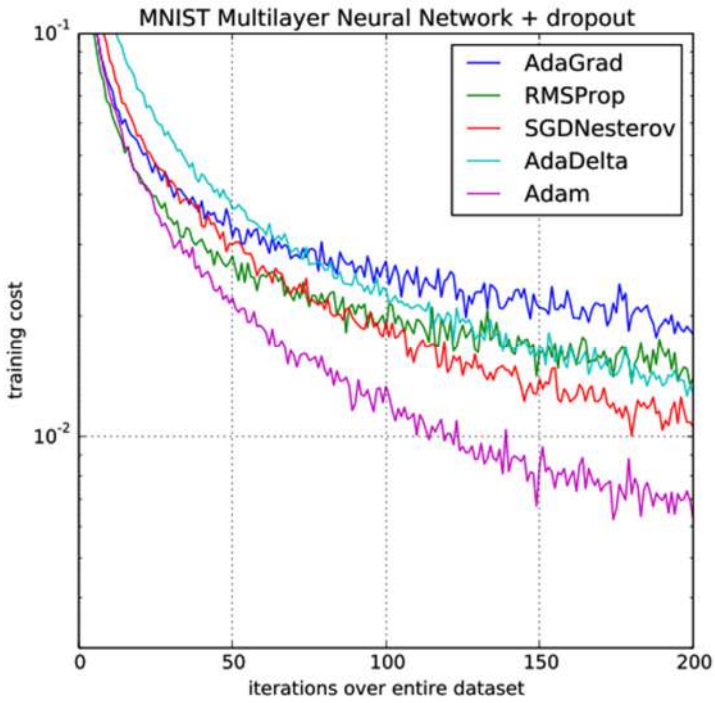
2. **Fungsi Aktivasi Softmax**

Fungsi aktivasi Softmax adalah suatu fungsi aktivasi *layer* pada *neural network* dengan menggunakan regresi Softmax. Regresi Softmax adalah perluasan dari regresi logistik dengan kemungkinan nilai output sebanyak *n* nilai. Pada kasus *klasifikasi gambar*, regresi Softmax digunakan untuk melakukan klasifikasi dengan jumlah kelas lebih dari dua (*n* kelas). Notasi matematis dari fungsi aktivasi Softmax dinyatakan sebagai berikut.

$$\alpha_{sm}\left(zd_j^{(L)}\right) = \frac{e^{z_j^{(L)}}}{1+e^{-z_j^{(L)}}} \tag{2.16}$$

2.8 Pengoptimal

Pengoptimal merupakan sebuah fungsi dengan tujuan meminimumkan *loss function* dengan mengatur parameter bias dan bobot. Terdapat beberapa pengoptimal yang sering digunakan dalam *neural network* diantaranya adalah *Stochastic Gradient Descend* (SGD), *Root Mean Square Propagation (RMSprop), Adaptive Gradient (AdaGrad), Adaptive Moment Estimation (Adam),* dll (Chauchan, 2020). Kingma & Ba (2017) menyatakan bahwa *Adam* kuat dan cocok untuk berbagai macam masalah optimasi non-konveks di bidang *machine learning* dengan keandalannya yang mampu meminimumkan nilai dari fungsi biaya lebih cepat dibandingkan pengoptimal lainnya (Kingma & Ba, 2017). Perbandingan performa Adam dengan pengoptimal lainnya dapat dilihat pada Gambar 2.8.



Gambar 2.8 Perbandingan Adam dengan pengoptimal lainnya (Kingma & Ba, 2017)

1. *Loss function*

Categorical cross-entropy adalah fungsi *loss* yang digunakan untuk klasifikasi multi-kelas. Misal $\hat{\boldsymbol{y}}$ adalah hasil prediksi dari model serta $\boldsymbol{y}$ adalah nilai dari label yang sebenarnya pada data *train*, maka *loss function* dari *categorical cross-entropy* dinyatakan sebagai berikut.

$$Loss(\hat{y}, y) = -\Sigma_{j=1}^{n} y_j . \log(\hat{y}_j) \tag{2.17}$$

dimana $j = 1, 2, \dots, n$ dengan n merupakan banyak kelas dari data *train* yang digunakan.

2. *Cost Function*

Cost function digunakan untuk mencari nilai *loss* rata-rata dengan menghitung setiap nilai *loss* yang diperoleh dari setiap hasil prediksi dan

tiap label $\boldsymbol{y}$ pada data *train*. Misal terdapat sebanyak $\boldsymbol{m}$ hasil prediksi dan label $\boldsymbol{y}$, maka *Cost function* dapat dinyatakan sebagai berikut.

$$J = \frac{1}{m}\sum_{p=1}^{m} Loss\left(\hat{y}_j^{(p)}, y_j^{(p)}\right) \tag{2.18}$$

dimana $p = 1, 2, \ldots, m$.

3. Algoritma *Adaptive Moment Estimation*

Adaptive Moment Estimation atau Adam merupakan suatu algoritma pengoptimal berbasis stokastik gradien yang efisien karena hanya membutuhkan order pertama dari gradien dengan memakan memori yang minimal (Kingma & Ba, 2017). Misal terdapat fungsi $J(\theta)$ yang merupakan *cost function* yang diferensiabel terhadap parameter model θ dan nilai awal $m_t = v_t = 0$, maka aturan pembaruan adam adalah sebagai berikut.

$$g_t = \frac{d}{d\theta}J(\theta) \tag{2.19}$$

$$m_t = \gamma_1 \cdot m_{t-1} + (1-\gamma_1)\cdot g_t \tag{2.20}$$

$$v_t = \gamma_2 \cdot v_{t-1} + (1-\gamma_2)\cdot g_t^2 \tag{2.21}$$

$$\widehat{m}_t = \frac{m_t}{1-\gamma_1^t} \tag{2.22}$$

$$\hat{v}_t = \frac{v_t}{1-\gamma_2^t} \tag{2.23}$$

$$\theta_t = \theta_{t-1} - lr_i \cdot \frac{\widehat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} \tag{2.24}$$

dimana:

t : iterasi ke-t

lr_i : *learning rate* ke-*i*

γ_1 : *decay rate* untuk momentum, nilai adalah 0,9

γ_2 : *decay rate* untuk gradien, nilai adalah 0,999

ϵ : konstanta 10^{-8}

g_t : gradien saat iterasi ke-t

m_t : pembaruan momentum pertama

$\hat{m}_t$: koreksi bias momentum pertama

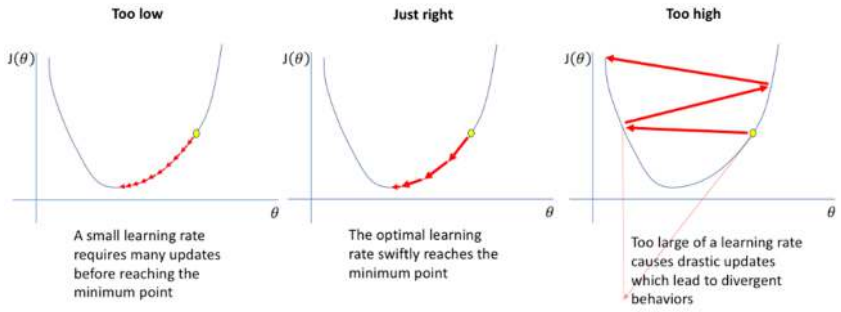
v_t : pembaruan momentum kedua

$\hat{v}_t$: koreksi bias momentum kedua

θ_t : pembaruan parameter model pada iterasi ke-t

4. *Learning Rate*

Learning rate (lr) adalah salah satu parameter dalam pengoptimal yang mengontrol seberapa besar langkah yang diambil dalam penyesuaian bobot model. Pemilihan nilai *learning rate* sangat mempengaruhi hasil model karena ketika nilai *learning rate* terlalu besar, maka melewatkan nilai *loss* yang minimum sedangkan ketika nilai *learning rate* terlalu kecil, maka membutuhkan waktu yang sangat banyak dalam menuju titik konvergensi dan mendapatkan nilai *loss* minimum (Jordan, 2018). Perbandingan pemilihan nilai *learning rate* dapat dilihat pada Gambar 2.9.



Gambar 2.9 Perbandingan pemilihan *learning rate* (Jordan, 2018)

2.9 Metrik *Accuracy*

Evaluasi dilakukan dengan menguji model menggunakan dataset *test* dengan menghitung banyak prediksi yang benar dibagi dengan seluruh prediksi.

$$Acc = \frac{True\ Positive + True\ Negative}{All\ Prediction} \tag{2.25}$$

2.10 *Python*

Bahasa *Python* merupakan suatu bahasa pemrograman buatan Guido van Rossum yang pertama kali dirilis pada tahun 1991 dan merupakan sebuah bahasa pemrograman tingkat tinggi yang populer digunakan berkat kemudahan dalam membaca dan memahami sintaknya (Python Software Foundation, 2012). Bahasa ini mendukung beragam paradigma pemrograman, mencakup pemrograman berorientasi objek, pemrograman fungsional, serta pemrograman prosedural (Python Software Foundation, 2012). Kode menggunakan bahasa Python dapat ditulis dan dijalankan dengan menggunakan berbagai *platform*. Salah satu *platform* yang paling banyak digunakan adalah Google Colaboratory yang dapat diakses secara *online*. Kelebihan *Python* terletak pada koleksi modul dan pustaka yang kuat, menjadikannya pilihan yang serbaguna untuk berbagai keperluan, mulai dari pengembangan web, analisis data, hingga pengaplikasian kecerdasan buatan. Beberapa modul dan pustaka yang digunakan dalam penelitian ini diantaranya adalah sebagai berikut.

1. TensorFlow

   TensorFlow adalah suatu *end-to-end platform* yang dibuat ramah untuk pemula untuk membuat model *Machine Learning* baik pada *platform desktop, mobile, website*, maupun *cloud* (TensorFlow Developer, 2023). Kenton (2022) menyatakan bahwa *end-to-end platform* berarti *platform* atau program yang menyediakan alat untuk memproses sesuatu dari awal hingga akhir dan memberikan solusi fungsional yang lengkap tanpa menggunakan pihak ketiga.

   TensorFlow terdiri dari berbagai macam modul pemrograman dengan fungsi yang beragam. Beberapa modul TensorFlow digunakan untuk pra-pemrosesan gambar dan membuat model CNN. Pra-pemrosesan gambar menggunakan modul *keras.preprocessing* sedangkan Model CNN dibuat dengan menggunakan modul Keras yang terdiri dari *models.sequential* dan *layers* dengan jenis Conv2D, MaxPooling2D, Flatten, dan Dense.

2. Matplotlib

   Matplotlib adalah salah satu modul atau *library* pada Python untuk melakukan visualisasi data menggunakan Python (Tineges & Davita, 2021). Pada penelitian ini, Matplotlib digunakan untuk memvisualisasikan performa model yang didapat oleh model.

3. Python Imaging Library (PIL)

   Python Imaging Library (PIL) atau yang biasa dikenal dengan Pillow adalah suatu modul atau *library* pada Python yang digunakan untuk

Pra-pemrosesan gambar yang dibuat oleh Jeffrey A. Clark. Modul ini digunakan untuk pengarsipan, pemprosesan *batch,* menampilkan, dan memproses gambar (Clark, 2024).

4. Modul *csv*

Format CSV adalah format impor dan ekspor data yang paling umum digunakan untuk data file spreadsheet maupun database. Modul *csv* pada Python memungkinkan *interpreter* untuk melakukan penulisan dan pembacaan file dengan format CSV (Python Software Foundation, 2024).

BAB III

OBJEK DAN METODE PENELITIAN

3.1 Objek Penelitian

Objek pada penelitian ini adalah dataset gambar spesies bunga edelweis yang terdiri dari data *train* dan data *test.* Secara berurutan, data untuk *training* dan data untuk *test* terdiri dari 3498 dan 1050 gambar yang tersebar ke dalam tiga buah kelompok sesuai spesiesnya yaitu *Anaphalis javanica, Leontopodium alpinum,* dan *Leucogenes grandiceps.* Persebaran data *training* dan data *test* secara berturut-turut sebanyak 1666 dan 350 per spesies. Data yang digunakan dalam penelitian ini merupakan data sekunder yang diperoleh dari *website* Kaggle (Malau, 2022).

3.2 Metode Penelitian

Metode yang digunakan dalam penelitian ini adalah *Convolutional Neural Network* dan menggunakan algoritma Adam sebagai pengoptimal untuk klasifikasi spesies edelweis menggunakan gambar dengan bahasa pemrograman Python. Pada penelitian ini, data diklasifikasikan ke dalam tiga kelas sesuai dengan jenis spesies pada dataset. Secara garis besar, langkah-langkah dalam penelitian ini terbagi ke dalam beberapa tahap dengan rincian sebagai berikut.

1. Pra-pemrosesan gambar

   Pra-pemrosesan gambar dilakukan dengan melakukan kompresi gambar dengan mengecilkan ukuran *piksel* menjadi lebar sebesar 512px dan tinggi menyesuaikan dengan rasio tiap gambar serta mengurangi kualitas gambar

menjadi 85%. Kompresi dilakukan dengan tujuan untuk mempercepat proses komputasi dan menyeragamkan ukuran gambar. Kemudian pra-pemrosesan gambar dilanjutkan dengan membuat *image data generator*. Pada tahap ini, komponen warna pada gambar diubah menjadi angka dan dinormalisasi. Selanjutnya data dibagi menjadi data *train* dan data *validation*. Rasio perbandingan antara data *train* dan data *validation* sebesar 8:2 dengan metode pemisahan yang dilakukan secara acak sehingga total data *train* sebanyak 2799 dan data *validation* sebanyak 699. Tahap terakhir dalam pra-pemrosesan gambar adalah dengan menyeragamkan ukuran gambar menjadi persegi dan membagi data *train* ke dalam *mini-batch.* Diketahui bahwa total data *train* sebanyak 2799 dan *batch size* sebesar 32, sehingga dengan menggunakan persamaan 2.2 diperoleh:

$$T = \left\lceil \frac{2799}{32} \right\rceil = 88 \tag{3.1}$$

dengan menggunakan persamaan 2.3, diperoleh:

$$Xtn^{\{1\}} = [\boldsymbol{X_1}, \boldsymbol{X_2}, \dots, \boldsymbol{X_{32}}] \tag{3.2}$$

$$Xtn^{\{2\}} = [\boldsymbol{X_{33}}, \boldsymbol{X_{34}}, \dots, \boldsymbol{X_{64}}] \tag{3.3}$$

$$\vdots$$

$$Xtn^{\{t\}} = \left[\boldsymbol{X}_{(32\cdot(t-1))+1}, \boldsymbol{X}_{(32\cdot t)+2} \dots, \boldsymbol{X}_{(t)\cdot 32}\right]; t = 1, 2, \dots, T \tag{3.4}$$

2. Inisiasi nilai *learning rate* sebesar 10^{-i}, dengan $i = 1$
3. *Training* Model: *forward propagation*

   Proses *Training* model diawali dengan tahap *forward propagation* yang bertujuan untuk mendapatkan hasil klasifikasi beserta nilai *Loss* dan *Cost function*. Proses *forward propagation* diawali dengan memasukkan data

gambar pertama dari *mini-batch* pertama dari persamaan 3.2 $\left(\boldsymbol{Xtn}_1^{\{1\}}\right)$. Substitusi $\boldsymbol{Xtn}_1^{\{1\}}$ ke persamaan 2.6 sehingga diperoleh:

$$zc^{(1)} = Conv\left(\boldsymbol{Xtn}_1^{\{1\}}, \boldsymbol{W}^{(1)}\right) + \beta^{(1)} \tag{3.5}$$

dimana dengan menggunakan persamaan 2.7 dan filter $\boldsymbol{W}$ yang berukuran $(3, 3, 16)$ maka diperoleh:

$$Conv\left(\boldsymbol{Xtn}_1^{\{1\}}, \boldsymbol{W}^{(1)}\right) = \sum_{a=1}^{3}\sum_{b=1}^{3}\sum_{c=1}^{16}\left(x_1^{\{1\}}\right)_{q+a-1,r+b-1,s+c-1} * w_{a,b,c}^{(1)} \tag{3.6}$$

substitusikan $zc^{(1)}$ ke fungsi aktivasi ReLU menggunakan persamaan 2.8 sehingga diperoleh:

$$A_{q,r,s} = \alpha_{ReLU}\left(zc^{(1)}\right) = \max\left(0, zc^{(1)}\right) \tag{3.7}$$

pindahkan *feature map* $\boldsymbol{Xts}_1^{\{1\}}$ sejauh 1 piksel kemudian ulangi langkah 3.5 sampai dengan langkah 3.7 sehingga diperoleh:

$$\boldsymbol{A}^{(1)} = \begin{bmatrix} [A_{1,1,1}, A_{1,1,2}, \dots, A_{1,1,s}] & [A_{1,2,1}, a_{1,2,2}, \dots, A_{1,2,s}] & \dots & [A_{1,r,1}, A_{1,r,2}, \dots, a_{1,r,s}] \\ [A_{2,1,1}, A_{2,1,2}, \dots, A_{2,1,s}] & [A_{2,2,1}, A_{2,2,2}, \dots, A_{2,2,s}] & \dots & [A_{2,r,1}, A_{2,r,2}, \dots, A_{2,r,s}] \\ \vdots & \vdots & \ddots & \vdots \\ [A_{q,1,1}, A_{q,1,2}, \dots, A_{q,1,s}] & [A_{q,2,1}, A_{q,2,2}, \dots, A_{q,2,s}] & \dots & [A_{q,r,1}, A_{q,r,2}, \dots, A_{q,r,s}] \end{bmatrix} \tag{3.8}$$

dengan dimensi output sebesar $(254, 254, 16)$. Substitusi $A^{(1)}$ ke persamaan 2.10 sehingga diperoleh:

$$\boldsymbol{P}^{(2)} = MaxPooling2D\left(\boldsymbol{A}^{(1)}\right) \tag{3.9}$$

dengan dimensi output sebesar $(127, 127, 16)$ dan akan menjadi input untuk *layer* selanjutnya sehingga substitusi $\boldsymbol{P}^{(2)}$ ke persamaan 2.6, diperoleh:

$$zc^{(3)} = Conv\left(\boldsymbol{P}^{(2)}, \boldsymbol{W}^{(3)}\right) + \beta^{(3)} \tag{3.10}$$

dimana dengan menggunakan persamaan 2.7 dan filter $\boldsymbol{W}$ yang berukuran $(3, 3, 32)$ maka diperoleh:

$$Conv\left(\boldsymbol{P}^{(2)}, \boldsymbol{W}^{(3)}\right) = \sum_{a=1}^{3}\sum_{b=1}^{3}\sum_{c=1}^{32}\left(\boldsymbol{P}^{(2)}\right)_{q+a,r+b,s+c} * w_{a,b,c}^{(3)} \qquad (3.11)$$

substitusikan $zc^{(3)}$ ke fungsi aktivasi ReLU menggunakan persamaan 2.8 sehingga diperoleh:

$$A_{q,r,s} = \alpha_{ReLU}\left(zc^{(3)}\right) = \max\left(0, zc^{(3)}\right) \qquad (3.12)$$

pindahkan *feature map* $\boldsymbol{P}^{(2)}$ sejauh 1 piksel kemudian ulangi langkah 3.10 sampai dengan langkah 3.12 sehingga diperoleh:

$$\boldsymbol{A}^{(3)} = \begin{bmatrix} [A_{1,1,1}, A_{1,1,2}, \dots, A_{1,1,s}] & [A_{1,2,1}, a_{1,2,2}, \dots, A_{1,2,s}] & \dots & [A_{1,r,1}, A_{1,r,2}, \dots, a_{1,r,s}] \\ [A_{2,1,1}, A_{2,1,2}, \dots, A_{2,1,s}] & [A_{2,2,1}, A_{2,2,2}, \dots, A_{2,2,s}] & \dots & [A_{2,r,1}, A_{2,r,2}, \dots, A_{2,r,s}] \\ \vdots & \vdots & \ddots & \vdots \\ [A_{q,1,1}, A_{q,1,2}, \dots, A_{q,1,s}] & [A_{q,2,1}, A_{q,2,2}, \dots, A_{q,2,s}] & \dots & [A_{q,r,1}, A_{q,r,2}, \dots, A_{q,r,s}] \end{bmatrix} \qquad (3.13)$$

dengan dimensi output sebesar $(125, 125, 32)$. Substitusi $\boldsymbol{A}^{(3)}$ ke persamaan 2.10 sehingga diperoleh:

$$\boldsymbol{P}^{(4)} = MaxPooling2D\left(\boldsymbol{A}^{(3)}\right) \qquad (3.14)$$

dengan dimensi output sebesar $(62, 62, 32)$ dan akan menjadi input untuk *layer* selanjutnya sehingga substitusi $\boldsymbol{P}^{(4)}$ ke persamaan 2.6, diperoleh:

$$zc^{(5)} = Conv\left(\boldsymbol{P}^{(4)}, \boldsymbol{W}^{(5)}\right) + \beta^{(5)} \qquad (3.15)$$

dimana dengan menggunakan persamaan 2.7 dan filter $\boldsymbol{W}$ yang berukuran $(3, 3, 64)$ maka diperoleh:

$$Conv\left(\boldsymbol{P}^{(4)}, \boldsymbol{W}^{(5)}\right) = \sum_{a=1}^{3}\sum_{b=1}^{3}\sum_{c=1}^{64}\left(\boldsymbol{P}^{(4)}\right)_{q+a-1,r+b-1,s+c-1} * w_{a,b,c}^{(5)} \qquad (3.16)$$

substitusikan $zc^{(5)}$ ke fungsi aktivasi ReLU menggunakan persamaan 2.8 sehingga diperoleh:

$$A_{q,r,s} = \alpha_{ReLU}\left(zc^{(5)}\right) = \max\left(0, zc^{(5)}\right) \tag{3.17}$$

pindahkan *feature map* $\boldsymbol{P}^{(4)}$ sejauh 1 piksel kemudian ulangi langkah 3.15 sampai dengan langkah 3.17 sehingga diperoleh:

$$\boldsymbol{A}^{(5)} = \begin{bmatrix} [A_{1,1,1}, A_{1,1,2}, \dots, A_{1,1,s}] & [A_{1,2,1}, a_{1,2,2}, \dots, A_{1,2,s}] & \dots & [A_{1,r,1}, A_{1,r,2}, \dots, a_{1,r,s}] \\ [A_{2,1,1}, A_{2,1,2}, \dots, A_{2,1,s}] & [A_{2,2,1}, A_{2,2,2}, \dots, A_{2,2,s}] & \dots & [A_{2,r,1}, A_{2,r,2}, \dots, A_{2,r,s}] \\ \vdots & \vdots & \ddots & \vdots \\ [A_{q,1,1}, A_{q,1,2}, \dots, A_{q,1,s}] & [A_{q,2,1}, A_{q,2,2}, \dots, A_{q,2,s}] & \dots & [A_{q,r,1}, A_{q,r,2}, \dots, A_{q,r,s}] \end{bmatrix} \tag{3.18}$$

dengan dimensi output sebesar $(60, 60, 64)$. Substitusi $\boldsymbol{A}^{(5)}$ ke persamaan 2.10 sehingga diperoleh:

$$\boldsymbol{P}^{(6)} = MaxPooling2D\left(\boldsymbol{A}^{(5)}\right) \tag{3.19}$$

dengan dimensi output sebesar $(30, 30, 64)$ dan akan menjadi input untuk *layer* selanjutnya sehingga substitusi $\boldsymbol{P}^{(6)}$ ke persamaan 2.6, diperoleh:

$$zc^{(7)} = Conv\left(\boldsymbol{P}^{(6)}, \boldsymbol{W}^{(7)}\right) + \beta^{(7)} \tag{3.20}$$

dimana dengan menggunakan persamaan 2.7 dan filter W yang berukuran $(3, 3, 128)$ maka diperoleh:

$$Conv\left(\boldsymbol{P}^{(6)}, \boldsymbol{W}^{(7)}\right) = \sum_{a=1}^{3}\sum_{b=1}^{3}\sum_{c=1}^{128}\left(\boldsymbol{P}^{(6)}\right)_{q+a-1,r+b-1,s+c-1} * w_{a,b,c}^{(7)} \tag{3.21}$$

substitusikan $zc^{(7)}$ ke fungsi aktivasi ReLU menggunakan persamaan 2.8 sehingga diperoleh:

$$A_{q,r,s} = \alpha_{ReLU}\left(zc^{(7)}\right) = \max\left(0, zc^{(7)}\right) \tag{3.22}$$

pindahkan *feature map* $\boldsymbol{P}^{(6)}$ sejauh 1 piksel kemudian ulangi langkah 3.20 sampai dengan langkah 3.22 sehingga diperoleh:

$$\boldsymbol{A}^{(7)} = \begin{bmatrix} [A_{1,1,1}, A_{1,1,2}, \dots, A_{1,1,s}] & [A_{1,2,1}, a_{1,2,2}, \dots, A_{1,2,s}] & \dots & [A_{1,r,1}, A_{1,r,2}, \dots, a_{1,r,s}] \\ [A_{2,1,1}, A_{2,1,2}, \dots, A_{2,1,s}] & [A_{2,2,1}, A_{2,2,2}, \dots, A_{2,2,s}] & \dots & [A_{2,r,1}, A_{2,r,2}, \dots, A_{2,r,s}] \\ \vdots & \vdots & \ddots & \vdots \\ [A_{q,1,1}, A_{q,1,2}, \dots, A_{q,1,s}] & [A_{q,2,1}, A_{q,2,2}, \dots, A_{q,2,s}] & \dots & [A_{q,r,1}, A_{q,r,2}, \dots, A_{q,r,s}] \end{bmatrix} \tag{3.23}$$

dengan dimensi output sebesar $(28, 28, 128)$. Substitusi $\boldsymbol{A}^{(7)}$ ke persamaan 2.10 sehingga diperoleh:

$$\boldsymbol{P}^{(8)} = MaxPooling2D\left(\boldsymbol{A}^{(7)}\right) \tag{3.24}$$

dengan dimensi output sebesar $(14, 14, 128)$ dan akan menjadi input untuk *layer* selanjutnya sehingga substitusi $\boldsymbol{P}^{(8)}$ ke persamaan 2.6, diperoleh:

$$zc^{(9)} = Conv\left(\boldsymbol{P}^{(8)}, \boldsymbol{W}^{(9)}\right) + \beta^{(9)} \tag{3.25}$$

dimana dengan menggunakan persamaan 2.7 dan filter $\boldsymbol{W}$ yang berukuran $(3, 3, 256)$ maka diperoleh:

$$Conv\left(\boldsymbol{P}^{(8)}, \boldsymbol{W}^{(9)}\right) = \sum_{a=1}^{3} \sum_{b=1}^{3} \sum_{c=1}^{256} \left(\boldsymbol{P}^{(8)}\right)_{q+a-1, r+b-1, s+c-1} * w_{a,b,c}^{(9)} \tag{3.26}$$

substitusikan $zc^{(9)}$ ke fungsi aktivasi ReLU menggunakan persamaan 2.8 sehingga diperoleh:

$$A_{q,r,s} = \alpha_{ReLU}\left(zc^{(9)}\right) = \max\left(0, zc^{(9)}\right) \tag{3.27}$$

pindahkan *feature map* $\boldsymbol{P}^{(8)}$ sejauh 1 piksel kemudian ulangi langkah 3.25 sampai dengan langkah 3.27 sehingga diperoleh:

$$\boldsymbol{A}^{(9)} = \begin{bmatrix} [A_{1,1,1}, A_{1,1,2}, \dots, A_{1,1,s}] & [A_{1,2,1}, a_{1,2,2}, \dots, A_{1,2,s}] & \dots & [A_{1,r,1}, A_{1,r,2}, \dots, a_{1,r,s}] \\ [A_{2,1,1}, A_{2,1,2}, \dots, A_{2,1,s}] & [A_{2,2,1}, A_{2,2,2}, \dots, A_{2,2,s}] & \dots & [A_{2,r,1}, A_{2,r,2}, \dots, A_{2,r,s}] \\ \vdots & \vdots & \ddots & \vdots \\ [A_{q,1,1}, A_{q,1,2}, \dots, A_{q,1,s}] & [A_{q,2,1}, A_{q,2,2}, \dots, A_{q,2,s}] & \dots & [A_{q,r,1}, A_{q,r,2}, \dots, A_{q,r,s}] \end{bmatrix}$$

(3.28)

dengan dimensi output sebesar $(12, 12, 256)$. Substitusi $\boldsymbol{A}^{(9)}$ ke persamaan 2.10 sehingga diperoleh:

$$\boldsymbol{P}^{(10)} = MaxPooling2D\left(\boldsymbol{A}^{(9)}\right) \tag{3.29}$$

dengan dimensi output sebesar $(6, 6, 256)$ dan akan menjadi input untuk *layer* selanjutnya sehingga substitusi $\boldsymbol{P}^{(10)}$ ke persamaan 2.6, diperoleh:

$$zc^{(11)} = Conv\left(\boldsymbol{P}^{(10)}, \boldsymbol{W}^{(11)}\right) + \beta^{(11)} \tag{3.30}$$

dimana dengan menggunakan persamaan 2.7 dan filter $\boldsymbol{W}$ yang berukuran $(3, 3, 256)$ maka diperoleh:

$$Conv\left(\boldsymbol{P}^{(10)}, \boldsymbol{W}^{(11)}\right) = \sum_{a=1}^{3} \sum_{b=1}^{3} \sum_{c=1}^{256} \left(\boldsymbol{P}^{(10)}\right)_{q+a-1, r+b-1, s+c-1} * w_{a,b,c}^{(11)}$$

(3.31)

Substitusikan $zc^{(11)}$ ke fungsi aktivasi ReLU menggunakan persamaan 2.8 sehingga diperoleh:

$$A_{q,r,s} = \alpha_{ReLU}\left(zc^{(11)}\right) = \max\left(0, zc^{(11)}\right) \tag{3.32}$$

pindahkan *feature map* $\boldsymbol{P}^{(10)}$ sejauh 1 piksel kemudian ulangi langkah 3.30 sampai dengan langkah 3.32 sehingga diperoleh:

$$\boldsymbol{A}^{(11)} = \begin{bmatrix} [A_{1,1,1}, A_{1,1,2}, \dots, A_{1,1,s}] & [A_{1,2,1}, a_{1,2,2}, \dots, A_{1,2,s}] & \dots & [A_{1,r,1}, A_{1,r,2}, \dots, a_{1,r,s}] \\ [A_{2,1,1}, A_{2,1,2}, \dots, A_{2,1,s}] & [A_{2,2,1}, A_{2,2,2}, \dots, A_{2,2,s}] & \dots & [A_{2,r,1}, A_{2,r,2}, \dots, A_{2,r,s}] \\ \vdots & \vdots & \ddots & \vdots \\ [A_{q,1,1}, A_{q,1,2}, \dots, A_{q,1,s}] & [A_{q,2,1}, A_{q,2,2}, \dots, A_{q,2,s}] & \dots & [A_{q,r,1}, A_{q,r,2}, \dots, A_{q,r,s}] \end{bmatrix}$$

(3.33)

dengan dimensi output sebesar $(4, 4, 256)$. Substitusi $\boldsymbol{A}^{(11)}$ ke persamaan 2.10 sehingga diperoleh:

$$\boldsymbol{P}^{(12)} = MaxPooling2D\left(\boldsymbol{A}^{(11)}\right) \tag{3.34}$$

dengan dimensi output sebesar $(2, 2, 256)$ dan akan menjadi input untuk *layer* selanjutnya sehingga substitusi $\boldsymbol{P}^{(12)}$ ke persamaan 2.11 diperoleh:

$$FL\left(P^{(12)}\right) = \overrightarrow{zf} = \left[P_{1,1,1}^{(12)}, P_{1,2,1}^{(12)}, P_{2,1,1}^{(12)}, P_{2,2,1}^{(12)}, \dots, P_{2,2,256}^{(12)}\right] \tag{3.35}$$

karena *layer flatten* tidak memiliki parameter yang dapat dilatih maka output dari *dense layer* diperoleh sebagai berikut:

$$zd_j^{(14)} = \overrightarrow{zf}_j, j = 1, 2, \dots, 1024 \tag{3.36}$$

substitusi $zd_j^{(14)}$ ke fungsi aktivasi ReLU menggunakan persamaan 2.13 dan persamaan 2.15 sehingga diperoleh.

$$d_j = \alpha_{ReLU}\left(zd_j^{(14)}\right) = \max\left(0, zd_j^{(14)}\right) \tag{3.37}$$

sehingga diperoleh output dari *dense layer* pada *layer* ke-14 adalah sebagai berikut

$$\left(\vec{d}\right)^{(14)} = [d_1, d_2, \dots, d_{1024}] \tag{3.38}$$

substitusi $\left(\vec{d}\right)^{(14)}$ ke persamaan 2.12 sehingga diperoleh.

$$zd_j^{(15)} = \sum_{k=1}^{1024} w_{jk}^{(15)} \times d_k^{(14)} + \beta_j^{(15)} \tag{3.39}$$

substitusi $zd_j^{(15)}$ ke persamaan 2.13 dan 2.16 sehingga diperoleh.

$$d_j = \alpha_{sm}\left(zd_j^{(15)}\right) = \frac{e^{z_j^{(15)}}}{1+e^{-z_j^{(15)}}} \tag{3.40}$$

sehingga diperoleh output dari *dense layer* pada *layer* ke 15 adalah sebagai berikut.

$$\left(\vec{d}\right)^{(15)} = [d_1, d_2, d_3] \tag{3.41}$$

substitusi d_j ke *loss function* pada persamaan 2.17 diperoleh:

$$Loss\left(d_j, y_j\right) = -\sum_{j=1}^{3} y_j . \log\left(d_j\right) \tag{3.42}$$

lakukan langkah 3.4 hingga 3.42 menggunakan $X_p^{\{1\}}$ dimana $p = 2, 3, \ldots, 28$ sehingga diperoleh:

$$Loss^{(p)}\left(d_j^{(p)}, y_j^{(p)}\right) = -\sum_{j=1}^{3} y_j^{(p)} . \log\left(d_j^{(p)}\right) \tag{3.43}$$

Hitung *cost function* dengan substitusi $Loss^{(p)}, p = 1,2, \ldots,32$ ke persamaan 2.18 diperoleh:

$$J = \frac{1}{28}\sum_{p=1}^{32} Loss\left(d_j{}^{(p)}, y_j^{(p)}\right) \tag{3.44}$$

$$J = -\frac{1}{28}\sum_{p=1}^{32}\sum_{j=1}^{3} y_j^{(p)} \cdot \log\left(d_j^{(p)}\right) \tag{3.45}$$

4. *Training* model: *backpropagation*

Proses *backpropagation* bertujuan untuk mencari turunan dari cost function terhadap parameter model. Setelah mendapatkan nilai *cost function,* hitung gradien J terhadap parameter model yaitu *weight* dan *bias.* Dari proses *forward propagation* dengan memperhatikan persamaan 2.6 dan persamaan 2.12 didapat:

$$zc^{(1)} = Conv\left(\boldsymbol{X}_1^{\{1\}}, \boldsymbol{W}^{(1)}\right) + \beta^{(1)} \tag{3.46}$$

$$zc^{(3)} = Conv\left(\boldsymbol{P}^{(2)}, \boldsymbol{W}^{(3)}\right) + \beta^{(3)} \tag{3.47}$$

$$zc^{(5)} = Conv\left(\boldsymbol{P}^{(4)}, \boldsymbol{W}^{(5)}\right) + \beta^{(5)} \tag{3.48}$$

$$zc^{(7)} = Conv\left(\boldsymbol{P}^{(6)}, \boldsymbol{W}^{(7)}\right) + \beta^{(7)} \tag{3.49}$$

$$zc^{(9)} = Conv\left(\boldsymbol{P}^{(8)}, \boldsymbol{W}^{(9)}\right) + \beta^{(9)} \tag{3.50}$$

$$zc^{(11)} = Conv\left(\boldsymbol{P}^{(10)}, \boldsymbol{W}^{(11)}\right) + \beta^{(11)} \tag{3.51}$$

$$zd_j^{(15)} = \sum_{k=1}^{1024} w_{jk}^{(15)} \times d_k^{(14)} + \beta_j^{(15)} \tag{3.52}$$

dari persamaan 3.46 hingga persamaan 3.52 didapat parameter-parameter dari model yang dapat dilatih adalah sebagai berikut:

$$\vec{w} = \left[\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(3)}, \boldsymbol{W}^{(5)}, \boldsymbol{W}^{(7)}, \boldsymbol{W}^{(9)}, \boldsymbol{W}^{(11)}, w_{jk}^{(15)}\right] \tag{3.54}$$

$$\vec{\beta} = \left[\beta^{(1)}, \beta^{(3)}, \beta^{(5)}, \beta^{(7)}, \beta^{(9)}, \beta^{(11)}, \beta_{jk}^{(15)}\right] \tag{3.55}$$

Dari persamaan 3.45 dan persamaan 3.54, substitusi $(\vec{w})_l$ dimana $l = 1,2,\ldots,7$ ke persamaan 2.19 sehingga didapat:

$$g_t = \frac{d}{d\vec{w}_l} J(\vec{w}_l) \tag{3.56}$$

kemudian gunakan persamaan 2.19 hingga persamaan 2.24 untuk memperbaharui nilai $weight_i$ sehingga didapat:

$$(\vec{w}_l)_t = (\vec{w}_l)_{t-1} - lr_i \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} \tag{3.57}$$

Dari persamaan 3.45 dan persamaan 3.55, substitusi $\vec{\beta}_l$ dimana $l = 1,2,\ldots,7$ ke persamaan 2.19 sehingga didapat:

$$g_t = \frac{d}{d\vec{\beta}_l} J\left(\vec{\beta}_l\right) \tag{3.58}$$

kemudian gunakan persamaan 2.22 hingga persamaan 2.26 untuk memperbaharui nilai $\vec{\beta}_l$ sehingga didapat:

$$\left(\vec{\beta}_l\right)_t = \left(\vec{\beta}_l\right)_{t-1} - lr_i \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} \tag{3.59}$$

Proses pembaharuan parameter w dan b dilakukan di setiap *mini-batch.* Setelah seluruh mini-batch telah selesai, maka proses training model dilanjutkan dengan menguji model dengan menggunakan data *validation.* Proses pengujian menggunakan data *validation* dilakukan hanya hingga proses *forward propagation* dengan hasil akhir berupa nilai *loss. Dataset validation* memiliki total data sebanyak 699 gambar dengan *batch size* 32, substitusi ke persamaan 2.2 diperoleh.

$$T = \left\lceil \frac{699}{32} \right\rceil = 22 \tag{3.60}$$

sehingga mini-batch dari *dataset test* dapat dinyatakan sebagai berikut.

$$\boldsymbol{Xval} = \left[\boldsymbol{X}^{\{t\}}\right]; t = 1,2, \dots, T \tag{3.61}$$

dimana

$$\boldsymbol{Xval}^{\{t\}} = \left[\boldsymbol{X}_{(32\cdot(t-1))+1}, \boldsymbol{X}_{(32\cdot t)+2} \dots, \boldsymbol{X}_{(t)\cdot 32}\right] \tag{3.62}$$

dari $\boldsymbol{Xval}^{\{1\}}, t = 1,2, \dots T$ dengan substitusi nilai $\boldsymbol{Xtn} = \boldsymbol{Xval}$ pada persamaan 3.5 diperoleh.

$$zc^{(1)} = Conv\left(\boldsymbol{Xval}_1^{\{1\}}, \boldsymbol{W}^{(1)}\right) + \beta^{(1)} \tag{3.63}$$

cari nilai *Cost function* dengan melanjutkan perhitungan hingga langkah ke 3.45. Kemudian cari nilai akurasi dengan menggunakan persamaan 2.25.

Kecil nya nilai *loss* dan tinggi nya *accuracy* menjadi penanda bahwa performa model baik. Proses *training* hingga pengujian menggunakan data *validation* dilakukan secara berulang hingga 20 *epoch*.

5. *Testing* model

Setelah diperoleh model terbaik dari proses *training* sebanyak 20 *epoch.* Model yang disimpan pada *epoch* terakhir kemudian diuji menggunakan dataset *test*. *Dataset test* diubah menjadi mini-batch dan dimasukkan ke dalam model untuk dilakukan proses klasifikasi. *Dataset test* memiliki total data sebanyak 1050 gambar dengan *batch size* 32, substitusi ke persamaan 2.2 diperoleh.

$$T = \left\lceil \frac{1050}{32} \right\rceil = 33 \tag{3.64}$$

sehingga mini-batch dari *dataset test* dapat dinyatakan sebagai berikut.

$$\boldsymbol{Xts} = \left[\boldsymbol{X}^{\{t\}}\right]; t = 1,2, \dots, T \tag{3.65}$$

dimana

$$\boldsymbol{Xts}^{\{t\}} = \left[\boldsymbol{X}_{(32\cdot(t-1))+1}, \boldsymbol{X}_{(32\cdot t)+2} \dots, \boldsymbol{X}_{(t)\cdot 32}\right] \tag{3.66}$$

dari $\boldsymbol{Xts}^{\{1\}}, t = 1,2, \dots T$ dengan substitusi nilai $\boldsymbol{Xtn} = \boldsymbol{Xts}$ pada persamaan 3.5 diperoleh.

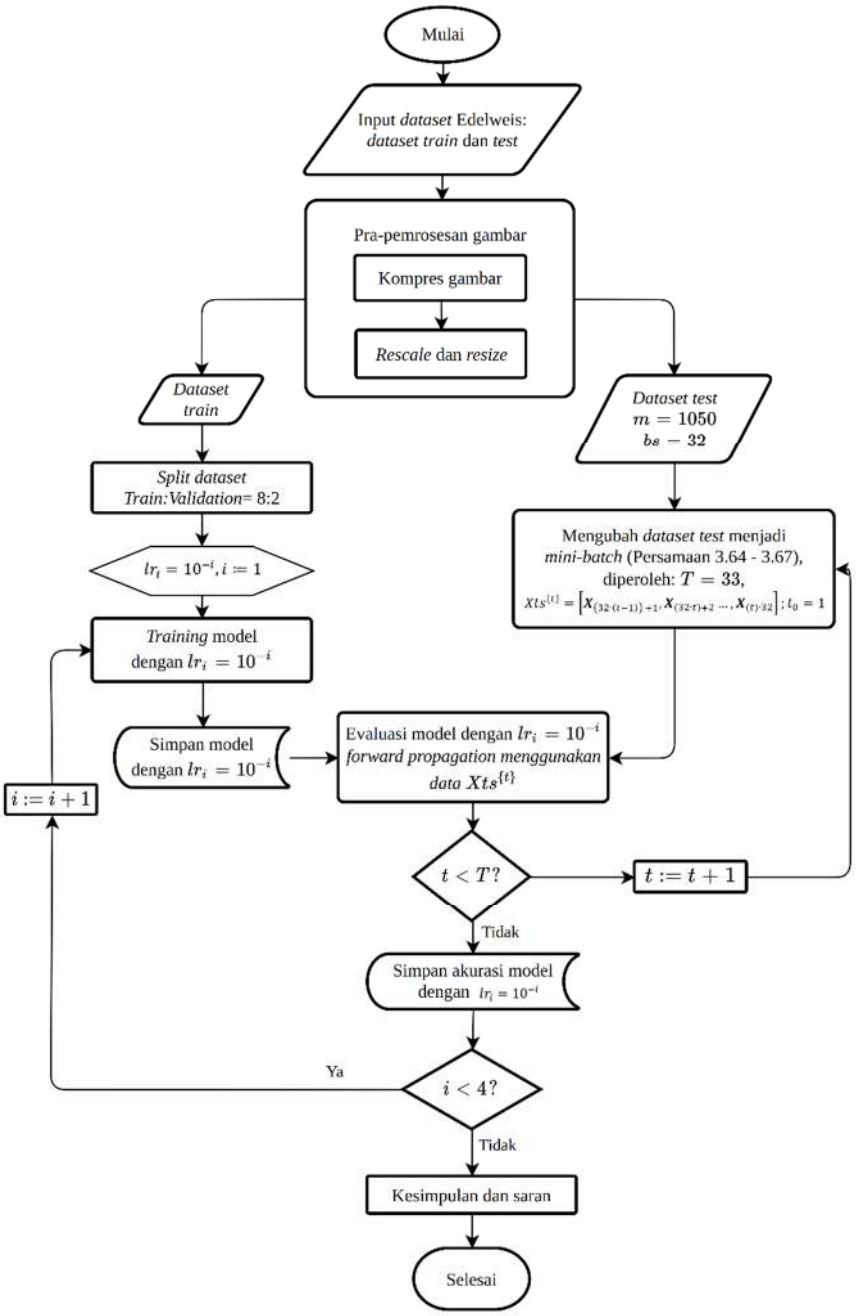
$$zc^{(1)} = Conv\left(\boldsymbol{Xts}_1^{\{1\}}, \boldsymbol{W}^{(1)}\right) + \beta^{(1)} \tag{3.67}$$

cari nilai *Cost function* dengan melanjutkan perhitungan hingga langkah ke 3.45. Kemudian cari nilai akurasi dengan menggunakan persamaan 2.25.

6. Perbarui nilai i pada *learning rate* sehingga $i = i + 1$.
7. Ketika $i > 4$, proses *training* model berhenti.

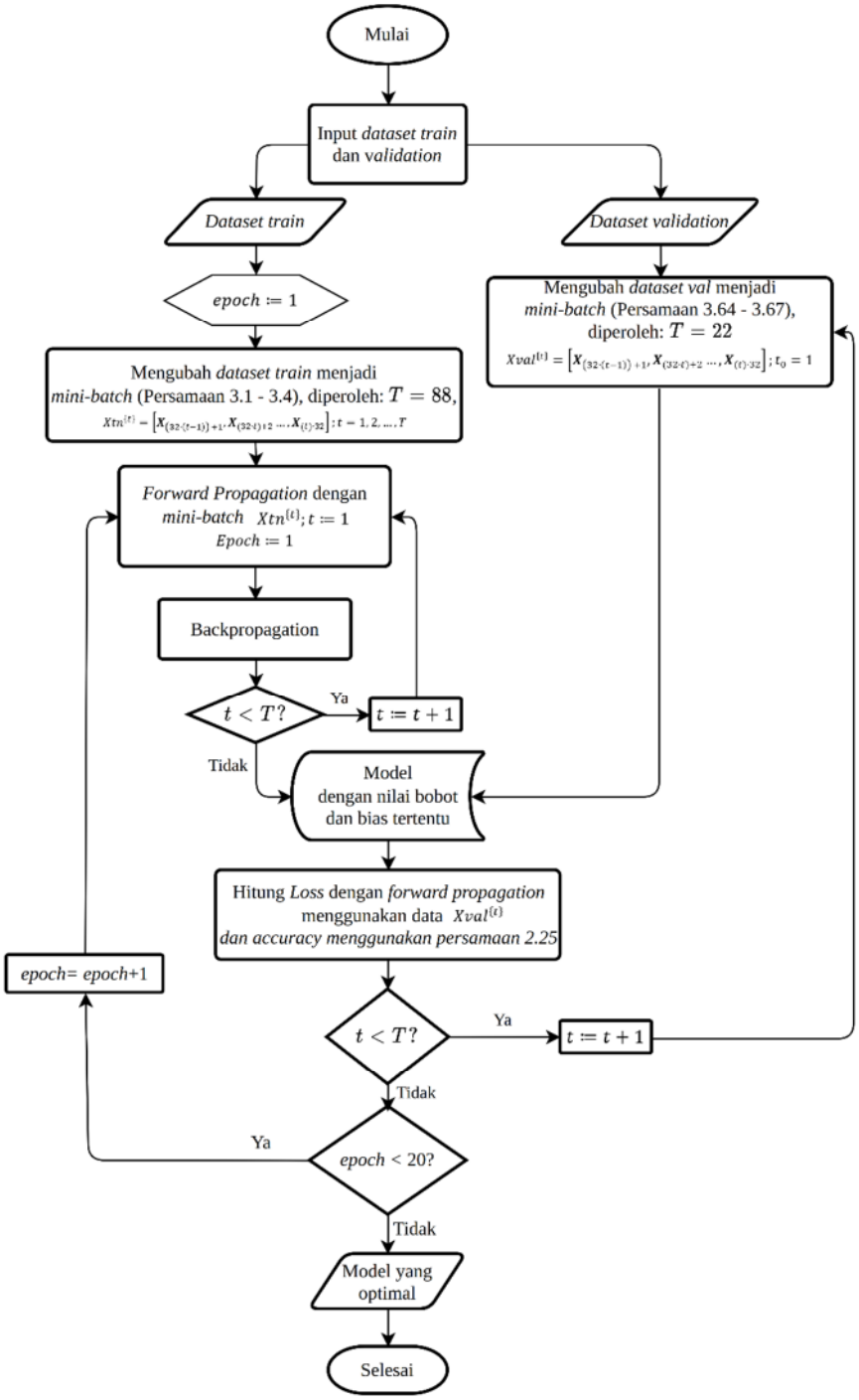
3.3 Diagram Alir Penelitian

Diagram alir penelitian klasifikasi spesies bunga edelweis menggunakan *Convolutional Neural Network* dan pengoptimal Adam terdapat pada Gambar 3.1.



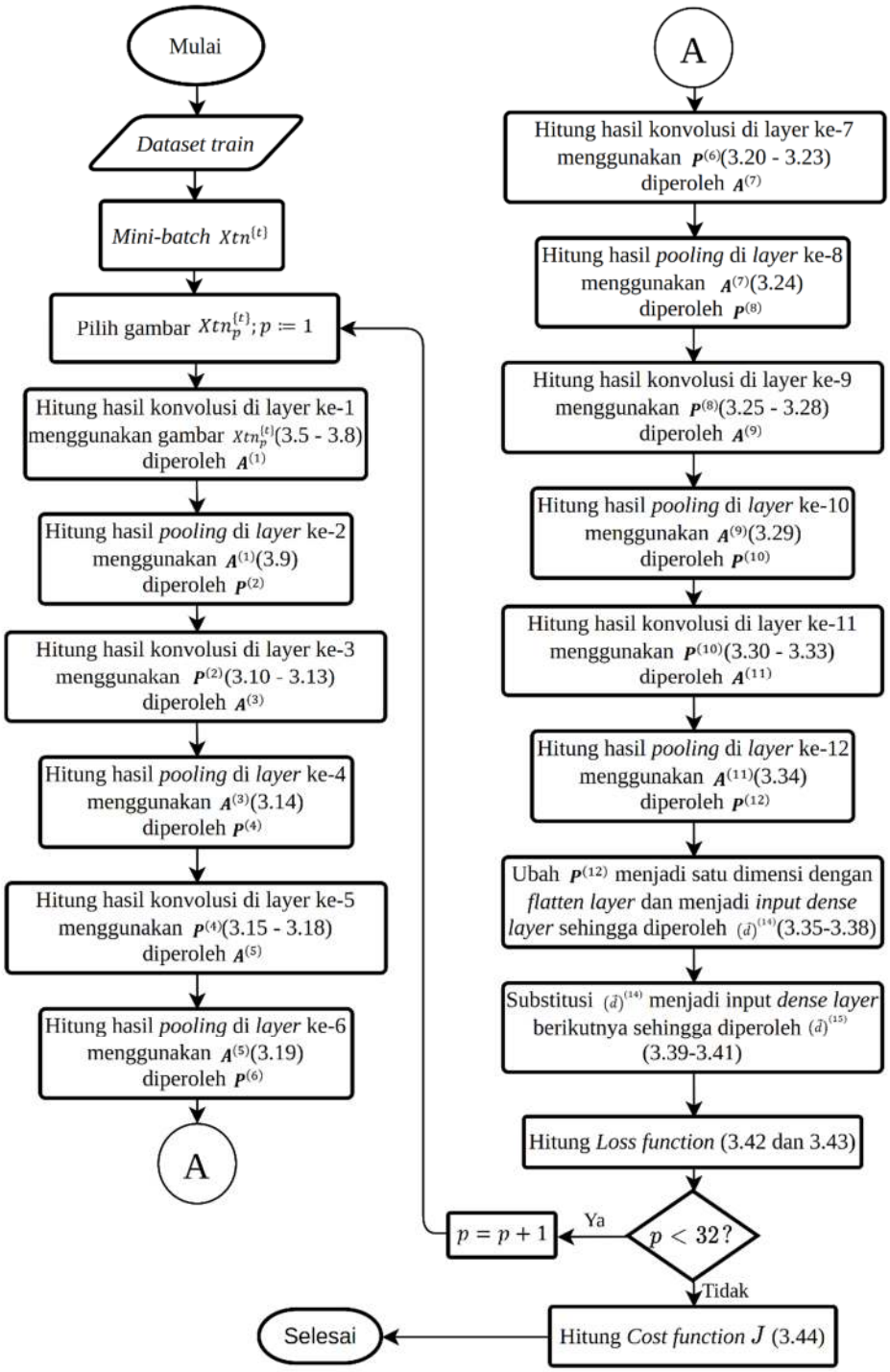
Gambar 3.1 Diagram alir penelitian

Diagram alir untuk *training* model dapat dilihat pada Gambar 3.2.



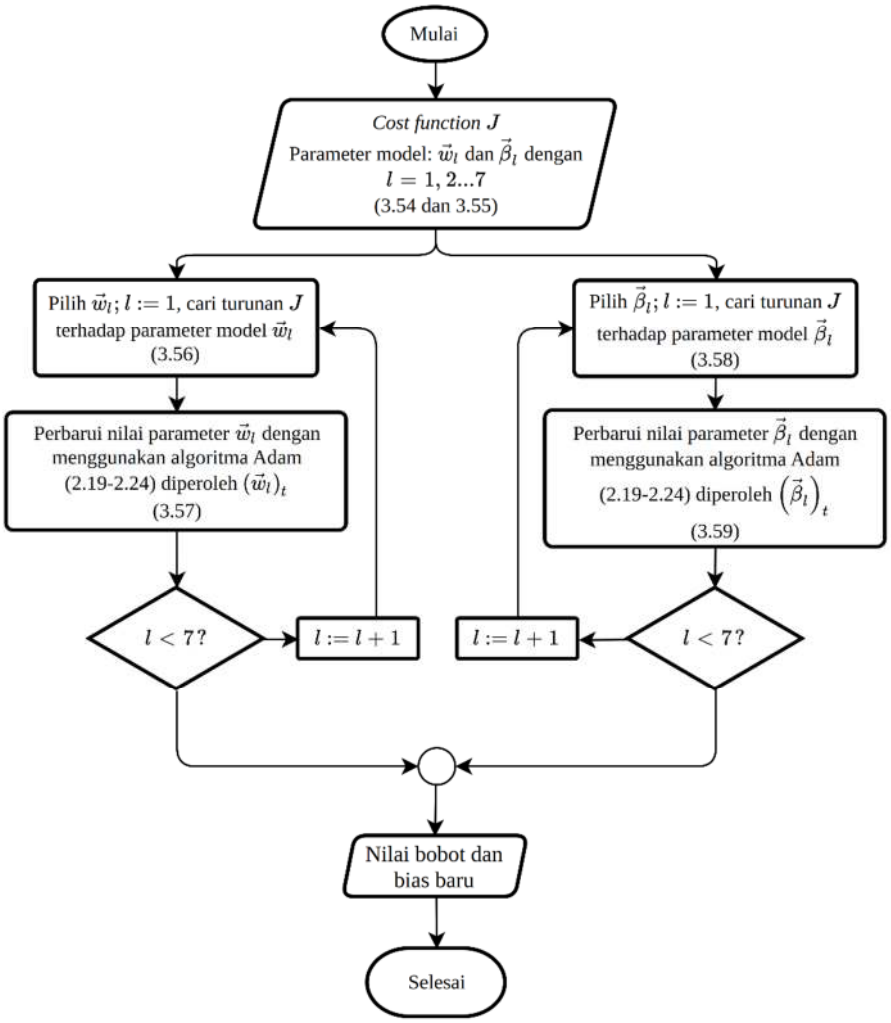
Gambar 3.2 Diagram alir tahap *training* model

Diagram alir pada tahap *forward propagation* dapat dilihat pada Gambar 3.3.



Gambar 3.3 Diagram alir pada tahap *forward propagation*

Diagram alir pada tahap *backpropagation* dapat dilihat pada Gambar 3.4.



Gambar 3.4 Diagram alir pada tahap *backpropagation*

DAFTAR PUSTAKA

Alkaff, A. K., & Prasetiyo, B. (2022). Hyperparameter Optimization on CNN Using Hyperband on Tomato Leaf Disease Classification. *2022 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom)*, 479–483. https://doi.org/10.1109/CyberneticsCom55287.2022.9865317

Anam, C. (2021). *Menengok Desa Wisata Edelweis Wonokitri Pasuruan, Surganya Bunga Abadi*. Solopos Bisnis. https://bisnis.solopos.com/menengok-desa-wisata-edelweis-wonokitri-pasuruan-surganya-bunga-abadi-1191018#sp-sharing

Chaki, J., & Dey, N. (2018). *A beginner's guide to image preprocessing techniques*. CRC Press.

Chauchan, N. S. (2020). *Optimization Algorithms in Neural Networks*. KDnugget. https://www.kdnuggets.com/2020/12/optimization-algorithms-neural-networks.html

Clark, J. (2024). *Pillow (PIL Fork) 10.3.0 documentation*. Python Software Foundation. https://pillow.readthedocs.io/en/stable/about.html

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow* (N. Tache, Ed.; 2nd ed.). O'Reilly Media, Inc. https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh & M. Titterington (Eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (Vol. 9, pp. 249–256). PMLR. https://proceedings.mlr.press/v9/glorot10a.html

Jain, J. (2023). *What are Tensors*. Medium. https://medium.com/@jayeshjain_246/what-are-tensors-495cf37c18e6

Jordan, J. (2018). *Setting the learning rate of your neural network*. https://www.jeremyjordan.me/nn-learning-rate/

Kartika, N. (2023). *Hulun Hyang Menabur Benih Edelweiss, Menuai Cinta yang Abadi*. Dirjen KSDAE MENLHK. https://ksdae.menlhk.go.id/artikel/12214/Hulun-Hyang-Menabur-Benih-Edelweiss-Menuai-Cinta-yang-Abadi.html

Kenton, W. (2022). *What Is End-To-End? A Full Process, From Start to Finish*. https://www.investopedia.com/terms/e/end-to-end.asp

Ketkar, N. (2017). Convolutional Neural Networks. In *Deep Learning with Python: A Hands-on Introduction* (pp. 63–78). Apress. https://doi.org/10.1007/978-1-4842-2766-4_5

Kingma, D. P., & Ba, J. L. (2017). Adam: A Method for Stochastic Optimization. *The 3rd International Conference for Learning Representations*.

LeCun, Y., Kavukcuoglu, K., & Farabet, C. (2010). Convolutional networks and applications in vision. *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, 253–256. https://doi.org/10.1109/ISCAS.2010.5537907

Malasari, T. (2022). *Perbedaan Bunga Edelweis yang Tumbuh Liar dan Dibudidaya*. Sariagri. https://hortikultura.sariagri.id/96735/perbedaan-bunga-edelweis-yang-tumbuh-liar-dan-dibudidaya

Malau, F. R. (2022). *Edelweiss Flower Dataset*. Kaggle. https://www.kaggle.com/datasets/ndomalau/edelweis-flower

Malau, F. R., & Mulyana, D. I. (2022). Classification of Edelweiss Flowers Using Data Augmentation and Linear Discriminant Analysis Methods. *Journal of Applied Engineering and Technological Science*, *4*(1), 139–148. https://doi.org/10.37385/jaets.v4i1.960

Menteri LHK. (2018). Peraturan Menteri Lingkungan Hidup dan Kehutanan Republik Indonesia Nomor P.106/Menlhk/Setjen/Kum.1/12/2018 Tentang Perubahan Kedua Atas Peraturan Menteri Lingkungan Hidup dan kehutanan Nomor P.20/Menlhk/Setjen/Kum.1/6/2018 Tentang Jenis Tumbuhan dan Satwa Yang Dilindungi. *Kementrian Lingkungan Hidup Dan Kehutanan Republik Indonesia, Jakarta*.

Muhammad, S., & Wibowo, A. T. (2021). Klasifikasi Tanaman Aglaonema Berdasarkan Citra Daun Menggunakan Metode Convolutional Neural Network (CNN). *EProceeding of Engineering*, 10621–10636.

Presiden RI. (1990). Undang Undang No. 5 Tahun 1990 Tentang: Konservasi Sumberdaya Alam Hayati Dan Ekosistemnya. *Jakarta: Dephut*.

Presiden RI. (1999). Peraturan Pemerintah No. 7 Tahun 1999 Tentang: Pengawetan Jenis Tumbuhan Dan Satwa. *Menteri Negara Sekretaris Negara Republik Indonesia, Jakarta*.

Python Software Foundation. (2012). *About Python*. https://web.archive.org/web/20120420010049/http://www.python.org/about/

Python Software Foundation. (2024). *CSV FIle Reading and Writing*. https://docs.python.org/3/library/csv.html

Riani, A. (2024). *Beda dari Gunung Lain, Mengapa Bunga Edelweiss di Bromo Dijual ke Wisatawan?* Liputan 6. https://www.liputan6.com/lifestyle/read/5509148/beda-dari-gunung-lain-mengapa-bunga-edelweis-di-bromo-dijual-ke-wisatawan?page=4

Shinozuka, M., & Mansouri, B. (2009). 4 - Synthetic aperture radar and remote sensing technologies for structural health monitoring of civil infrastructure systems. In V. M. Karbhari & F. Ansari (Eds.), *Structural Health Monitoring of Civil Infrastructure Systems* (pp. 113–151). Woodhead Publishing. https://doi.org/10.1533/9781845696825.1.114

TensorFlow Developer. (2023). *Introduction to TensorFlow*. https://www.tensorflow.org/learn

TensorFlow Developer. (2024a). *Conv2D*. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D

TensorFlow Developer. (2024b). *Preprocessing Image: ImageDataGenerator*. https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator

Tineges, R., & Davita, A. W. (2021). *Mengenal Matplotlib untuk Visualisasi Data dengan Python*. https://dqlab.id/mengenal-matplotlib-untuk-visualisasi-data-dengan-python

RIWAYAT HIDUP PENULIS

DATA PRIBADI

Nama Lengkap	: Wibi Anto
NPM	: 140110200025
Tempat, Tanggal Lahir	: Cirebon, 16 Mei 2003
Jenis Kelamin	: Laki-laki
Agama	: Islam
Alamat	: Blok Karang Rame, Desa Gamel, Kec. Plered, Kabupaten Cirebon, 45154
Email	: wibi20001@mail.unpad.ac.id

RIWAYAT PENDIDIKAN

2009 – 2015	SD Negeri 1 Gamel
2015 – 2018	SMP Negeri 1 Sumber
2018 – 2020	SMA Negeri 2 Cirebon
2020 – 2024	Program Studi S-1 Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Padjadjaran