INFOTECH journal
ISSN : 2460-1861 (Print), 2615-4250 (Online)
Vol. 9 No. 1, pp. 287-298

IDENTIFIKASI KUALITAS BENIH JAHE MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* (CNN)

Hendi Hidayat[1], Freza Riana[2],Ghibtha Fitri Laxmi[3]
[1,2,3]Teknik Informatika, Fakultas Teknik dan Sains, Universitas Ibn Khaldun
Email: hendih016@gmail.com

ABSTRACT
Ginger is one of the export commodities of Indonesian spices. Enhancement demand for ginger has not been balanced with ginger production. Low productivity This ginger is caused, among others, by the low availability of superior ginger seeds and healthy, lack of information related to ginger cultivation techniques and disorders of plant pest organisms (OPT). The seeds in ginger are carriers of natural genetic material in a plant to determine the best yield potential. For To get seeds that are suitable for rhizomes, a sorting process needs to be carried out. Process Sorting ginger seeds that develop in Indonesia is usually still done manually like looking at the seeds one by one, it takes time long enough. From this problem, an algorithm model is made to identify the quality of ginger seeds through digital image processing. Convolutional Neural Network (CNN) is a pattern recognition method that directly studied the characteristics of ginger seeds. CNN can make a ginger seed image classification model to measure the level of accuracy and model performance of digital image classification of ginger seeds. In this study using the CNN method which has an 88% accuracy rate in recognizing digital images of ginger seeds from a total dataset of 1187 images with a training data comparison of 70% (825 images), 20% validation data (240 images), and 10% test data (122 images). So that it can conclude that the designed CNN model can identify the image well.

Keywords: Ginger Seed, Convolutional Neural Network (CNN), Digital Image.

ABSTRAK
Jahe adalah salah satu komoditas ekspor rempah-rempah Indonesia. Peningkatan permintaan jahe belum seimbang dengan produksi jahe. Rendahnya produktivitas jahe ini disebabkan antara lain oleh rendahnya ketersediaan benih jahe yang unggul dan sehat, kurangnya informasi terkait teknik budidaya jahe dan gangguan organisme pengganggu tumbuhan (OPT). Benih pada jahe adalah pembawa sifat genetik pada sebuah tanaman untuk menentukan potensi hasil yang terbaik. Untuk mendapatkan benih yang layak rimpang perlu dilakukan proses pemilahan. Proses pemilahan benih jahe yang berkembang di Indonesia biasanya masih dilakukan secara manual seperti melihat satu per satu benih, hal tersebut merlukan waktu cukup lama. Dari masalah tersebut, maka dibuatlah model algoritma untuk mengidentifikasi kualitas benih jahe melalui sebuah pengolahan citra digital. *Convolutional Neural Network* (CNN) merupakan metode pengenal pola yang secara langsung mempelajari karakteristik dari benih jahe. CNN dapat membuat model klasifikasi citra benih jahe untuk diukur tingkat akurasi dan peforma model klasifkasi citra digital benih jahe. Pada penelitian ini menggunakan metode CNN yang memiliki tingkat akurasi 88% dalam mengenali citra digital benih jahe dari total *dataset* sebanyak 1187 citra dengan proporsi data latih 70% (825 citra), data validasi 20% (240 citra), dan data uji 10% (122 citra). Sehingga dapat disimpulkan bahwa model CNN yang dirancang dapat mengidentifikasikan citra dengan baik.

Kata Kunci: *Benih jahe, Convolutional Neural Network (CNN), Citra Digital.*

Riwayat Artikel :
Tanggal diterima : 27-05-2023
Tanggal revisi : 09-06-2023
Tanggal terbit : 10-06-2023
DOI :
https://doi.org/10.31949/infotech.v9i1.5458

1. PENDAHULUAN

1.1. Latar Belakang

Jahe (zingiber officinale rosc.; ginger) merupakan salah satu produk ekspor rempah indonesia. jahe pun merupakan bahan baku obat tradisional dan jamu serta berperan penting dalam menyerap tenaga kerja dan memperoleh devisa negara (penyakit and bakteri, 2016). Seiring bertambahnya luas penanaman dari tahun ke tahun. namun peningkatan permintaan jahe belum diimbangi dengan peningkatan produksi jahe. penyebab produktivitas jahe antara lain rendahnya ketersediaan benih yang berkualiatas baik dan sehat, teknik budidaya dan perusakan dari organisme pengganggu tanaman (djiwanti, 2015)**.**

Benih pada jahe adalah pembawa sifat genetic pada sebuah tanaman untuk menentukan potensi hasil yang terbaik dan mempengaruhi efektivitas dari *input output* pertanian. Dari banyaknya *input*, benih jahe merupakan sebuah tolak ukur untuk menentukan tingkat keberhasilan budidaya tanaman jahe kurang lebih 40% ditentukan dari kualitas benih. Dalam mencapai hasil yang optimal diperlukan benih yang sehat dan benar seperti benih murni yang sesuai dengan deskripsi varietasnya yang terbebas dari hama dan penyakit (Teknologi & Penelitian, 2011).

Untuk mendapatkan benih yang layak rimpang perlu dilakukan proses pemilahan. Benih yang akan digunakan harus jelas asal usulnya, cukup umurnya (9-10 bulan), bernas, kulit mengkilap, tidak memar, dan tidak tercampur dengan varietas lain. Benih yang sehat adalah berasal dari pertanaman yang sehat dan tidak terserang penyakit (Teknologi & Penelitian, 2011). Tujuannya adalah untuk menjamin stabilitas dan kepastian hasil budidaya pada jahe (Latifah et al., 2014). Proses pemilahan benih jahe yang berkembang di Indonesia biasanya masih dilakukan secara manual seperti melihat satu per satu benih, hal tersebut memerlukan waktu cukup lama. Pemilahan sebuah produk dengan tangan sangat tergantung pada kondisi operator.

Pengolahan citra digital dapat digunakan untuk melakukan proses pemilahan secara non destruktif tanpa merusak produk. Pemanfaatan *image processing* dalam melakukan identifikasi dapat membuat otomatisasi dalam pemilahan benih jahe secara efektif dan efisien (Rozaqi et al., 2021). Citra digital yang diperoleh *machine vision* akan dijadikan *input* pada *machine learning*. *Machine learning* ini akan dibangun untuk model identifikasi kualitas benih jahe berbasis klasifikasi.

Dalam penelitian ini algoritma model yang akan digunakan adalah *Convolutional Neural Network* (CNN). CNN merupakan metode pengenal pola yang secara langsung mempelajari karakteristik dari benih jahe dan mengurangi intervensi manual. Metode ini merupakan metode yang sedang banyak digunakan untuk pengenalan citra. Maka dari itu penggunaan CNN memiliki nilai yang berpotensi tinggi dalam hal kecepatan identifikasi, serta sangat luas untuk selalu dikembangkan dan diteliti (You, 2021).

Metode CNN dapat melakukan identifikasi beberapa jenis objek dengan cukup baik seperti, identifikasi ikan, daun, bunga yang menghasilkan nilai akurasi sebesar 80-97% (Arkadia et al., 2021; Dwi Fitriana Sari & Swanjaya, 2020; Fauzi et al., 2019; Felix et al., 2019; Zainuri & Pamungkas, 2020). Metode CNN menunjukan nilai akurasi yang cukup baik dalam mengidentifikasikan sebuah objek.

Penelitian ini bertujuan untuk membuat model dalam evaluasi kualitas benih jahe secara non destruktif menggunakan pengolahan citra digital. Sumber informasi dari pengolahan citra digital ini diharapkan dapat meningkatkan keberhasilan dalam identifikasi kualitas benih jahe.

1.2. Tinjuan Pustaka

1. Jahe

Jahe (Zingiber officinale Rosc.) merupakan komoditas tanaman obat yang permintaannya kuat di dalam dan luar negeri, selain dapat menambah devisa negara, juga dapat meningkatkan pendapatan petani. Usaha pengembangan lahan semakin meningkat dari tahun ke tahun, sehingga permintaan benih juga meningkat dari tahun ke tahun. Namun peningkatan permintaan jahe belum dibarengi dengan peningkatan produksi jahe. Rendahnya hasil jahe antara lain karena rendahnya ketersediaan benih yang berkualitas baik dan sehat, teknik budidaya, dan gangguan organisme pengganggu tanaman (Djiwanti, 2015).

2. Citra digital

Citra digital adalah representasi dari citra yang terbentuk berdasarkan sampling dan kuantisasi (jumlah citra yang akan diproses) yang diperoleh dari sebuah mesin. Sampel menggambarkan matriks baris dan kolom. Dengan kata lain, pengambilan sampel gambar mewakili ukuran piksel (titik) pada gambar, kuantisasi mewakili nilai tingkat kecerahan skala abu-abu yang sesuai dengan bilangan biner, dan kuantisasi teks pada gambar lain mewakili jumlah gambar berwarna. (Arkadia et al., 2021).

3. Pengolahan Citra Digital

Pengolahan citra adalah proses pengolahan citra dengan cara tertentu untuk mencapai tujuan yang diinginkan. Dalam perkembangan selanjutnya, image processing dan computer vision digunakan untuk menggantikan mata manusia sebagai alat pemeroleh citra, seperti kamera dan scanner sebagai mata, dan mesin komputer (beserta program perhitungannya) sebagai otak atau pusat informasi untuk penyesuaian. dan pemrosesan (Arkadia et al., 2021).

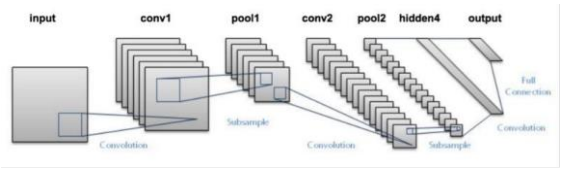
4. *Deep Learning*

Deep Learning (DL) adalah teknik di *Neural Networks* (NN) yang menggunakan teknik seperti *Restricted Boltzmann Machines* (RBM) untuk mempercepat pembelajaran di *Neural Networks* yang menggunakan proses multi layer atau lebih.

Dengan DL, pelatihan akan memakan waktu lebih sedikit karena masalah kehilangan gradien di backpropagation akan lebih sedikit. Beberapa jenis DL antara lain *Deep Autoencoder*, *Deep Belief Networks*, *Convolutional Neural Networks* (Ahmad Hania, 2017).

5. *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) adalah algoritma jaringan saraf mendalam, yang paling umum diterapkan untuk menganalisis gambar visual. CNN adalah multilayer perceptron yang setiap neuronnya terhubung ke semua neuron di lapisan berikutnya. Namun, CNN mampu menemukan pola hierarkis dalam data dan mengumpulkan *pixel* yang lebih kompleks dari piksel yang lebih kecil dan lebih sederhana. Oleh karena itu, performa CNN dalam keterhubungan dan kompleksitas *pixel* yang sangat baik. Arsitektur CNN yang khas terdiri dari *convolution layers*, *pooling layers*, *dan full connection layer* (Rasywir et al., 2020; Sholihin, 2021). Arsitektur pada CNN dapat dilihat pada Gambar 1.



Gambar 1. Arsitektur CNN

5.1 *Convolutional Layer*

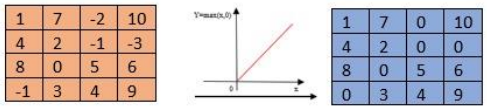
Convolution Layer adalah lapisan pada CNN yang bertugas untuk melakukan proses konvolusi dari citra digital *input* menggunakan *filter* telah ditentukan untuk mendapatkan *feature map. Filter* yang digunakan pada proses *Convolutional Layer* adalah berukuran 2x2 dengan *Stride* yang bernilai 1 (Arkadia et al., 2021).

5.2 ReLu *Layer*

Retrified Linear Unit (ReLu) *layer* ini mengambil suatu nilai input dan membatasi nilainya sampai dengan nol yaitu mengubah nilai negative menjadi nol (Arkadia et al., 2021). Adapun persamaan yang digunakan pada ReLu menggunakan persamaan (1)

$$y = \max(x, 0) \quad (1)$$

Proses pada ReLu *Layer* dapat dilihat pada Gambar 2.



Gambar 2. Proses ReLu

5.3 Pooling *Layer*

Pooling Layer merupakan lapisan yang dimasukkan diantara lapisan konvolusi secara berturut-turut dalam arsitektur CNN. Lapisan *Pooling* yang dimasukkan diantara lapisan konvolusi secara berturut-turut dalam arsitektur model CNN dapat secara progresif mengurangi ukuran volume output pada *Feature Map*, sehingga mengurangi jumlah parameter dan perhitungan di jaringan, dan untuk mengendalikan *Overfitting* (Nour, 2018).

5.4 *Flatten layer*

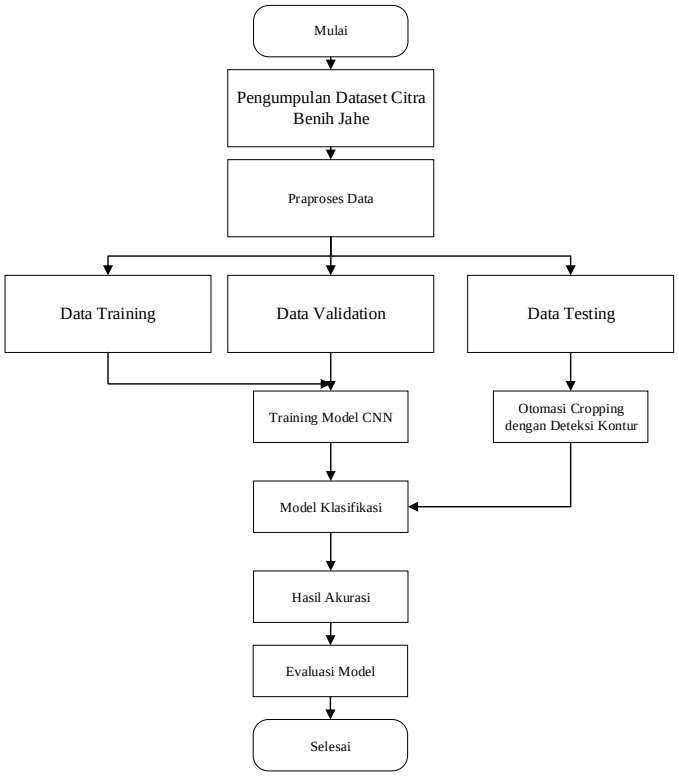
Lapisan membentuk suatu nilai vektor yangterbentuk dari nilai *matrix* dari hasil proses *max pooling feature map*. Hasil *flatten layer* ini kemudian menjadi nilai input yang akan dimasukan pada *Fully connected Layer* untuk mendapatkan hasil klasifikasi (Arkadia et al., 2021).

5.5 *Fully Connected Layer*

Pada lapisan ini, ia memiliki bentuk yang sama persis dengan beberapa lapisan Perceptron di mana setiap input terhubung ke *neuron* di lapisan tersembunyi Dapatkan hasil keluaran berdasarkan nilai bobot dan bias (Arkadia et al., 2021)

1.3. Metodologi Penelitian

Tahapan Penelitian yang dilakukan dapat dilihat pada Gambar 3.



Gambar 3. Tahapan Penelitian

1. Pengumpulan Dataset Citra Benih Jahe

Peneliti melakukan pengumpulan benih jahe untuk mendapatkan dataset citra. Dataset yang didapat untuk penelitian ini yaitu 1187 citra benih jahe yang terbagi kedalam dua kelas yaitu benih jahe yang

layak sebanyak 749 dataset dan benih jahe tidak layak sebanyak 438 dataset.

2. Praproses Data

Praproses merupakan tahapan sebelum melakukan proses klasifikasi meliputi beberapa proses yaitu :

2.1 *Data Augmentation*

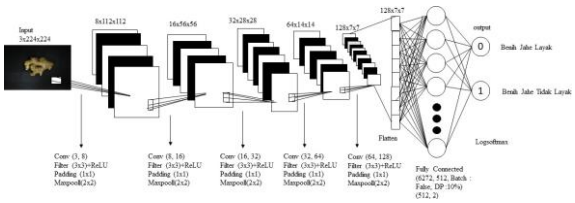
a. *Random Rotation* yaitu memutar secara acak citra digital sebanyak 15˚.
b. *Random Resize Crop* yaitu pemotongan citra menjadi ukuran 224x224 pixel.
c. *Transformasi to tensor* yaitu mengubah citra menjadi bentuk *tensor* agar dapat dibaca pada proses pembuatan model.

2.2 *Data Splitting*

Pengelompokkan *dataset* yaitu mengelompokkan *dataset* citra benih jahe yang berjumlah 1187 citra ke dalam data latih, data validasi, dan data uji dengan proporsi 70% (825 citra) data latih, 20% (240 citra) data validasi dan 10% (122 citra) data uji.

3. Arsitektur CNN

Pada tahap ini akan dilakukan pembuatan arsitektur model CNN. Adapun ilustrasi arsitektur CNN yang akan dibangun dapat dilihat pada Gambar 4.



Gambar 4. Arsitektur CNN

Setelah arsitektur model CNN dibuat, maka tahap selanjutnya adalah melakukan tuning parameter. Tuning parameter berguna untuk konfigurasi beberapa parameter yang digunakan pada model yang akan dibangun. Beberapa parameter tersebut berfungsi untuk memperoleh model dengan tingkat akurasi yang terbaik pada model CNN tersebut antara lain : Input citra dengan ukuran 224x224 *pixel*, *Convolutional Layer* dengan 5 *layer* konvolusi, *Max Pooling* dengan ukuran 2x2, *Kernel* terdapat 3 macam kernel yang dipakai peneliti yaitu 3x3,5x5,dan 7x7, *Stride* yang digunakan yaitu *stride* 1, *Learning Rate* yang digunakan yaitu 0,001, *Loss Function* yang digunakan yaitu NLLLOS.

4. Training Model CNN

Setelah melakukan *preprocessing* data, membuat arsitektur serta melakukan persiapan training. Maka langkah selanjutnya yaitu melakukan pelatihan dari model yang telah dirancang pada tahap sebelumnya. Tahapan ini merupakan proses pelatihan data citra benih jahe dalam mendeteksi atau mengklasifikasikan benih jahe layak dan tidak layak untuk algoritma yang digunakan yaitu *Convolutional Neural Network* (CNN).

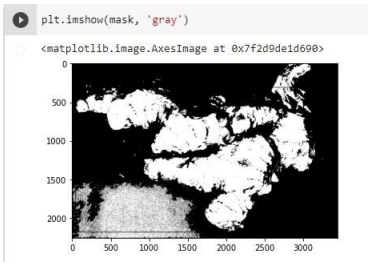
5. Model Prediksi CNN

Setelah melakukan proses pelatihan data citra benih jahe menggunakan CNN, maka diperoleh hasil yaitu berupa sebuah model untuk klasifikasi untuk selanjutnya akan diukur nilai keakuratannya dengan memasukkan data testing citra jahe ke dalam model tersebut.

6. Testing Model CNN

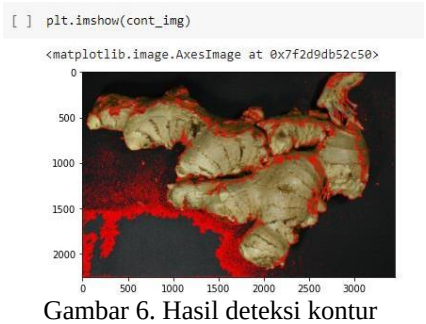
Pada testing, dilakukan proses otomasi pada cropping dengan mendeteksi contur pada citra. Proses ini dilakukan agar mendeteksi benih jahe yang lebih akurat, karena pada data testing terdapat beberapa citra yang background-nya cukup besar dibandingkan benih jahe tersebut. Terdapat beberapa tahap dalam mendeteksi contur pada citra, yaitu:

6.1 Melakukan *Masking* terhadap gambar yang akan di testing Masking pada gambar testing bertujuan untuk menyeleksi objek pada gambar *testing* dan merubahnya menjadi *grayscale*. Proses *masking* ditunjukkan pada Gambar 5.



Gambar 5. Proses *masking*

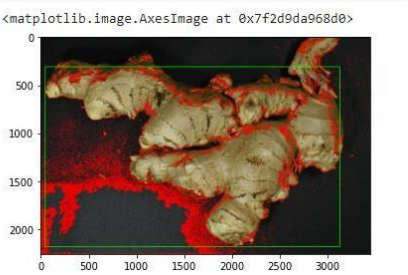
6.2 Mendeteksi kontur dari hasil mask gambar Hasil masking tersebut digunakan untuk mendeteksi contour dengan bantuan library dari cv2. Ilustrasi deteksi kontur ditunjukkan pada Gambar 6.



Gambar 6. Hasil deteksi kontur

6.3 Setelah contour terdeteksi dari gambar, langkah selanjutnya *rectangle* sesuai dengan kontur yang telah didapat pada gambar

tersebut. Ilustrasi pembuatan *rectangle* ditunjukkan pada Gambar 7.



Gambar 7. Hasil *Rectangle* pada citra

6.4 *Cropping* ini dilakukan dengan memotong gambar sesuai bentuk contour yang didapat. *Cropping* citra ditunjukkan pada gambar 8.

Gambar 8. Hasil Cropping

7. Evaluation

Klasifikasi adalah proses pengelompokan hasil terhadap model CNN berdasarkan hyperparameter yang telah ditentukan.untuk mendapatkan hasil akurasi dari model klasifikasi yang telah dibuat. Klasifikasi dalam penelitian dibagi kedalam 2 kelas, yakni layak dan tidak layak. Terdapat beberapa Pada tahap Evaluation yaitu

a. F1-Score

F1 score merupakan hasil perhitungan dari rata-rata *precision* dan *recall*. Nilai terbaik dari *F1 score* adalah 1 dan nilai terburuknya adalah 0. Adapun bentukpersamaan untuk menghitung *F1 score* dapat dilihat pada Persamaan 1

$$F1 - Score = \frac{1}{2}\left(\frac{1}{Precission} + \frac{1}{Recall}\right) \quad (1)$$

b. Recall

Recall atau *True Positive Rate (TPR)* dihitung dengan membagi prediksi positif yang benar dengan keseluruhan dari jumlah kelas yang positif. Nilai terbaik dari *Recall* adalah 1 dan nilai terburuknya adalah 0. Adapun bentuk persamaan untuk menghitung *Recall* dapat dilihat pada Persamaan 2

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

c. Precission

Precision merupakan hasil perhitungan dari keseluruhan jumlah prediksipositif yang benar dibagi dengan keseluruhan jumlah kelas yang benar. Nilai terbaikdari *Precision* adalah 1 dan nilai terburuknya adalah 0. Adapun bentuk persamaan untuk menghitung *Precision* dapat dilihat pada persamaan 3

$$Precission = \frac{TP}{TP + FP} \quad (3)$$

d. Accuration

Akurasi dihitung dengan membagi semua prediksi yang benar dengan seluruh data. Nilai akurasi terbaik 1 dan nilai akurasi terburuk adalah 0 (rentang 0-1). Adapun untuk menghitung akurasi dapat dilihat pada Persamaan 4

$$Accuration = \frac{TP + TN}{TP + TN + FN + FP} \quad (4)$$

2. PEMBAHASAN

Pada tahap ini dijelaskan urutan proses klasifikasi citra kelayakan benih jahe menggunakan metode CNN, mengukur hasil akurasi dari model yang dibuat serta mengukur performa model yang dibangun. Proses pembuatan model klasifikasi CNN dibantu dengan library deep learning python yaitu Pytorch.

2.1 Hasil Percobaan Skenario Model CNN

Pada evaluasi model ini, peneliti akan melihat dari Confusion Matrix dan performa dari Sembilan skenario yang telah didapat yakni f1-score, recall, precision, dan accuration. Peneliti akan membagi nya sesuai epoch yang dilakukan pada proses training yakni 50,75, dan 100.

2.2 Evaluasi Performa Skenario dengan *Epoch* 50

Dalam mengevaluasi percobaan yang telah dilakukan pada skenario 50 dengan kernel 3x3, 5x5, 7x7 dan Padding 1,2,3. Untuk melihat performa dari model yang didapat dilakukan evaluasi pada model tersebut. Adapun evaluasi yang digunakan adalah *recall*, f1 *score*, *precision*. Untuk menghitungnilai *recall*, *f1 score* dan *precision* dapat menggunakan persamaan 2., persamaan 3. dan persamaan 5. yang telah dibahas pada bab sebelumnyamengenai *confusion matrix*. Tabel 2. merupakan hasil dari *recall*,*f1 score* dan precision pada data testing.

Skenario Dengan *Epoch* 100							
Kernel, Padding	*Precision*		*Recall*		*F1-Score*		*Accuracy*
	Benih Jahe Layak	Benih Jahe Tidak Layak	Benih Jahe Layak	Benih Jahe Tidak Layak	Benih Jahe Layak	Benih Jahe Tidak Layak	
3x3, 1	0.89	0.85	0.92	0.80	0.92	0.82	0.88
5x5, 2	0.82	0.90	0.96	0.61	0.88	0.73	0.84
7x7, 3	0.94	0.73	0.81	0.91	0.87	0.81	0.84

Gambar 9. Evaluasi Skenario 3x3, 5x5 dan 7x7 dengan *epoch* 50

Dari Gambar 9. Didapat hasil skenario dengan *epoch* 50, dari hasil skenario tersebut terdapat tiga model klasifikasi untuk identifikasi kualitas benih jahe dengan tingkat akurasi yang cukup baik yaitu berkisar 80-88%. Pada kernel 3x3 dengan *epoch* 50 cukup baik untuk mengidentifikasikan kualitas benih jahe dengan tingkat akurasi sebesar 88%. *Kernel*, *padding*, dan *epoch* sangat berpengaruh pada tingkat akurasi, belum tentu nilai *epoch* yang besar akan menghasilkan nilai akurasi yang tinggi.

2.3 Evaluasi Performa Skenario dengan *Epoch* 75

Dalam mengevaluasi percobaan yang telah dilakukan pada skenario 75 dengan kernel 3x3, 5x5, 7x7 dan Padding 1,2,3. Untuk melihat performa dari model yang didapat dilakukan evaluasi pada model tersebut. Adapun evaluasi yang digunakan adalah *recall*, *f1 score*, *precision*. Untuk menghitungnilai *recall*, *f1 score* dan *precision* dapat menggunakan persamaan 2., persamaan 3. dan persamaan 5. yang telah dibahas pada bab sebelumnya mengenai *confusion matrix*. Tabel 4. merupakan hasil dari *recall*,*f1 score* dan precision pada data *testing*.

Skenario Dengan *Epoch* 75							
Kernel, Padding	*Precision*		*Recall*		*F1-Score*		*Accuracy*
	Benih Jahe Layak	Benih Jahe Tidak Layak	Benih Jahe Layak	Benih Jahe Tidak Layak	Benih Jahe Layak	Benih Jahe Tidak Layak	
3x3, 1	0.89	0.69	0.79	0.82	0.84	0.75	0.80
5x5, 2	0.85	0.89	0.95	0.70	0.90	0.78	0.86
7x7, 3	0.90	0.82	0.90	0.82	0.90	0.82	0.87

Gambar 10. Evaluasi Skenario 3x3, 5x5, dan 7x7 dengan *epoch* 75

Dari Gambar 10. Didapat hasil skenario dengan *epoch* 75, dari hasil skenario tersebut terdapat tiga model klasifikasi untuk identifikasi kualitas benih jahe dengan tingkat akurasi yang cukup baik yaitu berkisar 80-87%. Pada kernel 7x7 dengan *epoch* 75 mendapat nilai akurasi terbaiknya pada *epoch* 75 yaitu dengan tingkat akurasi sebesar 87%. Pada kernel 5x5 mencapai nilai akurasi terbaiknya pula yaitu sebesar 86%, tetapi pada kernel 3x3 mengalami penurunan yang cukup signifikan menjadi 80%. Hal tersebut dikarenakan adanya informasi yang menjadi bias pada sebuah citra.

2.4 Evaluasi Performa Skenario dengan *Epoch* 100

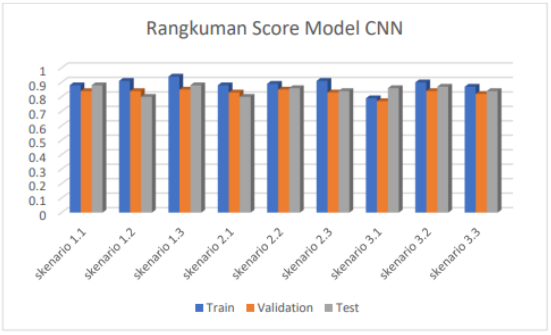
Dalam mengevaluasi percobaan yang telah dilakukan pada skenario 100 dengan kernel 3x3, 5x5, 7x7 dan Padding 1,2,3. Untuk melihat performa dari model yang didapat dilakukan evaluasi pada model tersebut. Adapun evaluasi yang digunakan adalah *recall*, *f1 score*, *precision*. Untuk menghitungnilai *recall*, *f1 score* dan *precision* dapat menggunakan persamaan 2., persamaan 3. dan persamaan 5. yang telah dibahas pada bab sebelumnya mengenai *confusion matrix*. Tabel 6. merupakan hasil dari *recall*,*f1 score* dan precision pada data *testing*.

Skenario Dengan *Epoch* 100							
Kernel, Padding	*Precision*		*Recall*		*F1-Score*		*Accuracy*
	Benih Jahe Layak	Benih Jahe Tidak Layak	Benih Jahe Layak	Benih Jahe Tidak Layak	Benih Jahe Layak	Benih Jahe Tidak Layak	
3x3, 1	0.89	0.85	0.92	0.80	0.92	0.82	0.88
5x5, 2	0.82	0.90	0.96	0.61	0.88	0.73	0.84
7x7, 3	0.94	0.73	0.81	0.91	0.87	0.81	0.84

Gambar 11. Evaluasi Skenario 3x3, 5x5 dan 7x7 dengan *epoch* 100

Dari Gambar 11. Didapat hasil skenario dengan *epoch* 100, dari hasil skenario tersebut terdapat tiga model klasifikasi untuk identifikasi kualitas benih jahe dengan tingkat akurasi yang cukup baik yaitu berkisar 84-88%. Pada kernel 3x3 dengan *epoch* 100 cukup baik untuk mengidentifikasikan kualitas benih jahe dengan tingkat akurasi sebesar 88%. *Kernel*, *padding*, dan *epoch* sangat berpengaruh pada tingkat akurasi, belum tentu nilai *epoch* yang besar akan menghasilkan nilai akurasi yang tinggi.

2.5 Rangkuman Evaluasi Model



Gambar 12. Rangkuman Evaluasi Model

Dari Gambar 12. terdapat Sembilan skenario yang terdapat tiga perbedaan dalam setiap skenarionya yakni *epoch*, *kernel* dan *padding*. Didapat kesimpulan dari Gambar 5. bahwa pada skenario 1.1 dan 1.3 merupakan model terbaik yang mendapat nilai akurasi 88%.

3. ALGORITMA ATAU PROGRAM

```
from tensorflow import keras
from keras.applications import
densenet
from keras.preprocessing import
image
from keras.applications.densenet
import preprocess_input,
decode_predictions
from keras.layers import
Dense,Dropout,
GlobalAveragePooling2D,Conv2D,Max
Pooling2D
from keras.callbacks import
ModelCheckpoint
import json
from keras import regularizers
from google.colab import files
import h5py
import pickle
from
tensorflow.keras.applications.inc
eption_v3 import InceptionV3
from
tensorflow.keras.preprocessing
import image
from tensorflow.keras.models
import Model, load_model
```

```
from tensorflow.keras.callbacks
import ModelCheckpoint,
ReduceLROnPlateau
from tensorflow.keras.layers
import Dense,
GlobalAveragePooling2D, Dropout
from tensorflow.keras.layers
import BatchNormalization,
Activation
from tensorflow.keras.optimizers
import Adam, RMSprop
from
tensorflow.keras.preprocessing.im
age import ImageDataGenerator
from sklearn.metrics import
classification_report,confusion_m
atrix
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from numpy import loadtxt
import tensorflow.keras
from PIL import Image, ImageOps
%matplotlib inline
import pandas as pd
!pip install jcopdl
!pip install luwiji
import torch
from torch import nn, optim
import matplotlib.pyplot as plt
from sklearn.datasets import
make_classification
from sklearn.metrics import
plot_confusion_matrix
from torch.utils.data import
DataLoader
from torchvision import datasets,
transforms
from jcopdl.callback import
Callback, set_config
from google.colab import drive
drive.mount('/content/drive')
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import
confusion_matrix,
classification_report
import torch
from torch import nn, optim
from jcopdl.callback import
Callback, set_config
from skimage.util import
random_noise
device = torch.device("cuda:0" if
torch.cuda.is_available() else
"cpu")
device

from
torchvision.transforms.transforms
import RandomHorizontalFlip
from torchvision import datasets,
transforms
from torch.utils.data import
DataLoader

bs = 32
crop_size = 224
# epochs = 50

train_transform =
transforms.Compose([

    #
transforms.CenterCrop(crop_size),
    transforms.RandomRotation(15)
,
    transforms.RandomResizedCrop(
crop_size, scale=(0.8, 1.0)),
    transforms.RandomHorizontalFl
ip(),
    transforms.ToTensor(),
    #
transforms.Normalize([0.485,
0.456, 0.406], [0.229, 0.224,
0.225]),
])

test_transform =
transforms.Compose([
    transforms.Resize(232),
    # transforms.ToPILImage(),
    transforms.CenterCrop(crop_si
ze),
    transforms.ToTensor(),
    #
transforms.Normalize([0.485,
0.456, 0.406], [0.229, 0.224,
0.225]),
])
```

```
validation_transform =
transforms.Compose([
    transforms.CenterCrop(crop_si
ze),
    #
transforms.RandomHorizontalFlip()
,
    #
transforms.RandomRotation(10),
    transforms.ToTensor(),
])

train_set =
datasets.ImageFolder('/content/dr
ive/MyDrive/dataset
benih/train/',
transform=train_transform)
trainloader = DataLoader
(train_set, batch_size=bs,
shuffle=True, num_workers=2)

test_set =
datasets.ImageFolder('/content/dr
ive/MyDrive/dataset benih/test/',
transform=test_transform)
testloader = DataLoader
(test_set, batch_size=bs,
shuffle=True)

validation_set =
datasets.ImageFolder('/content/dr
ive/MyDrive/dataset
benih/valid/',
transform=test_transform)
validationloader = DataLoader
(validation_set, batch_size=bs,
shuffle=True)
label2cat = train_set.classes
label2cat
loader = iter(trainloader)
images, labels = next(loader)
images.shape
fig, axes = plt.subplots(4, 4,
figsize=(18, 18))
for img, label, ax in zip(images,
labels, axes.flatten()):
    ax.imshow(img.permute(1,2,0).
cpu())
    # ax.imshow(img.view(360,
360).cpu(), cmap='gray')
    label =
label2cat[label.item()]
    ax.set_title(f"{label}")
    ax.axis('off');
from jcopdl.layers import
conv_block, linear_block
class CNN(nn.Module):
    def __init__(self):
        super().__init__()

        self.convolutional =
nn.Sequential(
         #jumlah layer #fitur
extraction
            conv_block(3, 8),
            conv_block(8, 16),
            conv_block(16, 32),
            conv_block(32, 64),
            conv_block(64,128),
            nn.Flatten()

        )

        self.fc = nn.Sequential(
            #lapisan NN
            linear_block(6272,
512, dropout=0.10),
            linear_block(512, 2,
activation='lsoftmax')
        )


    def forward(self, x):
        x = self.convolutional(x)
        x = self.fc(x)
        return x
config = set_config ({

    "batch_size" : bs,
    "crop_size" : crop_size,
    #  "batch_norm" : "False",
    "author" : "Hendi"

})
#Training
trainn_cost, vall_cost,
trainn_score, vall_score=
[],[],[],[] #buat report loss
from tqdm.auto import tqdm
epochs=50
```

```
# def loop_fn(mode, dataset,
dataloader, model, criterion,
optimizer, device):
#   if mode == "train" :
#     model.train()
#   elif mode == "test" :
#     model.eval()
#   cost = correct = 0
#   for feature, target in
tqdm(dataloader,
desc=mode.title()):
#       feature, target =
feature.to(device),
target.to(device)
#       output = model(feature)
#       loss = criterion(output,
target)
#       if mode == "train":
#         loss.backward()
#         optimizer.step()
#         optimizer.zero_grad()

#       cost += loss.item() *
feature.shape[0]
#       correct +=
(output.argmax[1] ==
target).sum().item()
#     cost = cost / len(dataset)
#     acc = correct /
len(dataset)
#     return cost, acc
#     while True:
#   train_cost, train_score =
loop_fn("train", train_set,
trainloader, model, criterion,
optimizer, device)
#   with torch.no_grad():
#     test_cost, test_score =
loop_fn("test", test_set,
testloader, model, criterion,
optimizer, device)

#     #logging
#     callback.log(train_cost,
test_cost, train_score,
test_score)

#     #checkpoint
#     callback.cost_runtime_plott
ing()
#     callback.score_runtime_plot
ting()

#     #early stopping
#     if
callback.early_stopping(model,
monitor= "test_score"):
#       callback.plot_cost()
#       callback.plot_score()
#       break

for i in range(epochs): #infinite
looping dan akan di handle oleh
callback sampai early stop
    model.train()

    cost = 0
    correct = 0 #score

    for feature, target in
tqdm(trainloader, desc='<<
Training >> '):
        feature, target =
feature.to(device),
target.to(device) #pindahkan ke
device

        output = model(feature)
#masukan data ke dalam model
        loss = criterion(output,
target) #perbedaan output dgn
data
        loss.backward()
#backpropagation

        optimizer.step() #update
weight
        optimizer.zero_grad()
#reset gradiennya kembali ke 0

        cost += loss.item()
*feature.shape[0] #hitung cost
per minibatch
        correct +=
(output.argmax(1) ==
target).sum().item() #output
dibandingkan dgn labels (berapa
banyak yg benar)
```

```
    train_cost = (cost /
len(train_set)) #reporting cost
rata-rata
    train_score = correct /
len(train_set) #reporting score
rata-rata

    trainn_cost.append(train_cost
) #reporting
    trainn_score.append(train_sco
re)

    #Testing
    with torch.no_grad(): #tdk
simpan komputasinya pada testing
(menghemat memori)
        model.eval()

        cost = 0
        correct = 0 #score

        for feature, target in
tqdm(validationloader, desc="<<
Validation >> "):
            feature, target =
feature.to(device),
target.to(device) #pindahkan ke
device

            output =
model(feature) #masukan data ke
dalam model
            loss =
criterion(output,target)
#perbedaan output dgn data

            cost += loss.item()
*feature.shape[0] #hitung cost
per minibatch
            correct +=
(output.argmax(1) ==
target).sum().item() #output
dibandingkan dgn labels (berapa
banyak yg benar)

    val_cost = (cost /
len(validation_set)) #reporting
cost rata-rata
    val_score = correct /
len(validation_set) #reporting
score

    vall_cost.append(val_cost)
#reporting
    vall_score.append(val_score)
    # callback.log(train_cost,
test_cost, train_score,
test_score)
    # callback.log(train_cost,
test_cost, train_score,
test_score)

    #  # Checkpoint #sesuai
default diatas, save stiap 50
epoch
    # callback.save_checkpoint()

    # # Runtime Plotting #setiap
20 epoch di plot
    #
callback.cost_runtime_plotting()
    #
callback.score_runtime_plotting()


    print(f"\rEpoch:
{i+1:4}/{epochs:2} | Train Loss :
{trainn_cost[-1]:7.4f} |
Validation Loss : {vall_cost[-
1]:7.4f} | Train score :
{trainn_score[-1]:7.4f} |
Validation Score : {vall_score[-
1]:7.4f}",end="")
model = CNN().to(device)
criterion = nn.NLLLoss()
optimizer =
optim.AdamW(model.parameters(),
lr=0.001)
callback = Callback(model,
config, outdir="model",
plot_every=1)
plt.figure(figsize=(17,7))

plt.subplot(121)
plt.plot(trainn_cost, 'b',
label="Train Cost")
plt.plot(vall_cost, 'r',
label="Validasi Cost")
plt.xlabel("Epoch");
plt.ylabel("Cost"); plt.legend();
```

```
plt.savefig('/content/drive/MyDrive/skripsi hendi/skenario 1/skenario epoch 50/cost')

plt.subplot(122)
plt.plot(trainn_score, 'b', label="Train Score")
plt.plot(vall_score, 'r', label="Validation Score")
plt.xlabel("Epoch");
plt.ylabel("Score");
plt.legend();
plt.savefig('/content/drive/MyDrive/skripsi hendi/skenario 1/skenario epoch 50/score')
```

4. KESIMPULAN

Model klasifikasi CNN yang didapat oleh peneliti yaitu terdapat Sembilan model untuk mengidentifikasikan kualitas benih jahe dengan tingkat akurasi terbaik yaitu pada skenario 1.1 dan 1.3 dengan tingkat akurasi sebesar 88%.

Tingkat akurasi model klasifikasi dapat ditingkatkan dengan beberapa cara antara lain, menambahkan dataset untuk menyeimbangkan data antar kelasnya dan melakukan segmentasi citra sebelum melakukan klasifikasi.

PUSTAKA

Ahmad Hania, A. (2017). Mengenal Artificial Intelligence, Machine Learning, & Deep Learning. *Jurnal Teknologi Indonesia*, *1*(June), 1–6. https://amt-it.com/mengenal-perbedaan-artificial-inteligence-machine-learning-deep-learning/

Arkadia, A., Damayanti, S. A., & Prasvita, D. S. (2021). Klasifikasi Buah Mangga Badami Untuk Menentukan Tingkat Kematangan dengan Metode CNN. *Seminar Nasional Mahasiswa Ilmu Komputer Dan Aplikasinya (SENAMIKA)*, *September*, 158–165.

Djiwanti, S. R. (2015). Perlakuan Benih Air Panas, Ekstrak Mimba Dan Jarak Kepyar Untuk Mengendalikan Nematoda (Meloidogyne spp.) Terbawa Rimpang Jahe. Effectiveness of several seed treatment methods to control rhizome seed-borne nematode Meloidogyne spp. ofginger. *Bul. Littro*, *26*(1), 55–62.

Dwi Fitriana Sari, & Swanjaya, D. (2020). Implementasi Convolutional Neural Network Untuk Identifikasi Penyakit Daun Gambas. *Seminar Nasional Inovasi Teknologi*, *04*(03), 137–142.

Fauzi, S., Eosina, P., & Laxmi, G. F. (2019). Implementasi Convolutional Neural Network Untuk Identifikasi Ikan Air Tawar. *Seminar Nasional Teknologi Informasi*, 163–167.

Felix, Faisal, S., Butarbutar, T. F. M., & Sirait, P. (2019). Implementasi CNN dan SVM untuk Identifikasi Penyakit Tomat via Daun. *Issn 2622-8130*, *20*(2), 117–134.

Latifah, K. ., Jauhari, E., Januwati, M., Rizal, M., D.Wardana, H., Hendani, N., Listyorini, Baswasiati, Hartoyo, B., Purwanto, Nurwidodo, Supriyadi, Elnizar, Hikmat, A., & Lina. (2014). Budidaya Jahe (Zingiber officinale). *Paper Knowledge . Toward a Media History of Documents*.

Nour, E. (2018). *IMPLEMENTASI METODE CONVOLUTIONAL NEURAL NETWORK UNTUK KLASIFIKASI TANAMAN PADA CITRA RESOLUSI TINGGI (The Implementation of Convolutional Neural Network Method for Agricultural Plant Classification in High Resolution Imagery)*. 61–68.

Penyakit, P., & Bakteri, L. (2016). *Otih Rostiana*. *January 2007*, 76–100.

Rasywir, E., Sinaga, R., & Pratama, Y. (2020). Analisis dan Implementasi Diagnosis Penyakit Sawit dengan Metode Convolutional Neural Network (CNN). *Paradigma - Jurnal Komputer Dan Informatika*, *22*(2), 117–123. https://doi.org/10.31294/p.v22i2.8907

Rozaqi, A. J., Sunyoto, A., & Arief, M. rudyanto. (2021). Deteksi Penyakit Pada Daun Kentang Menggunakan Pengolahan Citra dengan Metode Convolutional Neural Network. *Creative Information Technology Journal*, *8*(1), 22. https://doi.org/10.24076/citec.2021v8i1.263

Sholihin, M. (2021). Identifikasi Kesegaran Ikan Berdasarkan Citra Insang dengan Metode Convolution Neural Network. *JATISI (Jurnal Teknik Informatika Dan Sistem Informasi)*, *8*(3), 1352–1360. https://doi.org/10.35957/jatisi.v8i3.939

Teknologi, S., & Penelitian, H. (2011). Zingiber officinale Rosc. In *SpringerReference*. https://doi.org/10.1007/springerreference_69739

You, J. (2021). *Leaf Image Classification Using Deep Learning Network*. *4*(3), 109–115. https://doi.org/10.25236/AJCIS.2021.040317

Zainuri, M., & Pamungkas, D. P. (2020). Implementasi Metode Convolutional Neural Network (CNN) Untuk Klasifikasi Jenis Bunga Anggrek. In *Seminar Nasional Inovasi Teknologi* (pp. 87–92).