**KLASIFIKASI IMAGE TINGGI TANAMAN JAGUNG DENGAN MENGGUNAKAN ALGORITMA CONVOLUTION NEURAL NETWORK (CNN)**

## **IKBAL**

UNIVERSITAS MUHAMMADIYAH MAKASSAR 2024

e-mail: xxxx@xxxx.xxx\*

***Abstract***

*This research implements a Convolutional Neural Network (CNN) algorithm for classifying corn plant images, focusing on identifying plant height. The study aims to assist farmers in early detection of corn plant growth, contributing to more efficient agricultural management. The developed CNN model successfully classified plant height with a maximum accuracy of 88% on a test set of 1,019 images. The use of CNN demonstrated promising results in predicting plant height, indicating positive potential for further development of image-based applications in the agricultural sector.*

*KATA PENGANTAR iv*

*DAFTAR ISI v*

*DAFTAR TABEL vii*

*DAFTAR GAMBAR viii*

*Daftar Istilah x*

*Bab I Pendahuluan 1*

*Latar belakang 1*

*Rumusan Masalah 2*

*Tujuan Penelitian 2*

*Manfaat Penelitian 3*

*Ruang Lingkup Penelitian 3*

*Sistematika Penyusunan 3*

*Keywords: Convolutional Neural Network (CNN), Image Classification, Corn Plant Height, Precision Agriculture, Digital Image Processing, Model Testing, Agricultural Efficiency, Measurement Accuracy, Image Dataset.*

**Abstrak**

Penelitian ini mengimplementasikan algoritma Convolutional Neural Network (CNN) untuk klasifikasi citra tanaman jagung dengan fokus pada pengidentifikasian tinggi tanaman. Penelitian ini bertujuan untuk membantu petani dalam mendeteksi secara dini pertumbuhan tanaman jagung, yang dapat berkontribusi dalam pengelolaan pertanian yang lebih efisien. Model CNN yang dikembangkan berhasil mengklasifikasikan tinggi tanaman dengan akurasi maksimal mencapai 88% pada data uji dengan 1019 citra. Penggunaan CNN menunjukkan hasil yang cukup baik dalam prediksi tinggi tanaman, memberikan indikasi positif untuk pengembangan lebih lanjut dalam aplikasi berbasis citra untuk sektor pertanian.

Kata Kunci: : Convolutional Neural Network (CNN), Klasifikasi Citra, Tinggi Tanaman Jagung, Pertanian Presisi, Pengolahan Citra Digital, Pengujian Model, Efisiensi Pertanian, Akurasi Pengukuran, Dataset Gambar.

**1. Pendahuluan**

Tanaman merupakan salah satu komponen penting dalam ekosistem dan memiliki peran signifikan dalam kehidupan manusia. Tinggi tanaman adalah salah satu parameter yang dapat memberikan informasi penting tentang pertumbuhan dan kesehatan tanaman. Pengukuran tinggi tanaman secara akurat memiliki nilai penting dalam berbagai aplikasi seperti pertanian, kehutanan, ekologi, dan ilmu lingkungan.

Pengukuran tinggi tanaman secara manual memerlukan waktu, tenaga, dan sumber daya yang signifikan. Selain itu, ketepatan pengukuran sering kali bergantung pada faktor subjektif yang dapat mengakibatkan kesalahan pengukuran. Oleh karena itu, pengembangan metode otomatis untuk menghitung tinggi tanaman menjadi sangat penting untuk meningkatkan efisiensi dan akurasi pengukuran.

Dalam konteks ini, pemanfaatan Deep Learning, khususnya dalam bidang image processing atau pengolahan citra digital, menjadi relevan. Sistem image processing dirancang untuk membantu manusia dalam mengenali atau mengklasifikasi objek dengan cepat, tepat, dan dapat memproses banyak data sekaligus.

Pada bidang image processing terdapat beberapa algoritma yang dapat digunakan. Di antaranya adalah Naïve Bayes, Support Vector Macine,dan Neural Network. Salah satu algoritma yang sering digunakan adalah Neural Network. Neural Network dikembangkan berdasarkan cara kerja jaringan saraf pada otak manusia.Sejalan dengan perkembangan teknologi, maka, dikembangkan pula algoritma pengolahan citra digital. Salah satu pengembangan dari deep learning adalah Convolutional Neural Network. Pada tahun 1989, Yan LeCun, dkk mengembangkan model Neural Network dengan melakukan klasifikasi citra kode zip menggunakan kasus khusus dari Feed Forward Neural Network yang kemudian diberi nama Convolutional Neural Network (CNN). Metode Convolutional Neural Network memiliki hasil yang paling signifikan dalam pengenalan citra digital. Hal tersebut dikarenakan CNN diimplementasikan berdasarkan sistem pengenalan citra pada visual cortex manusia.

Convolutional Neural Network (CNN) merupakan metode jaringan saraf mendalam, yang paling umum diterapkan dalam menganalisis gambar visual. Convolutional Neural Network juga disebut sebagai multilayer perceptron yang setiap neuronnya terhubung ke semua neuron dilapisan berikutnya. Namun disamping itu, Convolutional Neural Network juga mampu menemukan pola hierarkis dalam sebuah data dan mengumpulkan piksel yang lebih kompleks dari piksel yang lebih kecil dan juga lebih sederhana. Oleh karena itu, performa Convolutional Neural Network dalam keterhubungan dan kompleksitas piksel gambar sangat sesuai dan baik (Rasywir et al., 2020).

Berdasatkan latar belakang di atas bahwa penggunaan algoritma Convolution Neural Network dalam sistem klasifikasi image tanaman jagung dapat membantu mengetahui pertumbuhan tanaman jagung itu dengan mengidentifikasi gambar pada tanaman jagung dengan lebih efisien. Ini akan membantu petani jagung dalam menjaga tanaman mereka dan meningkatkan hasil panen mereka.

1. Bagaimana mengimplementasikan Convolutional Neural Network (CNN) untuk klasifikasi gambar tanaman jagung dengan akurasi maksimal?

2. Bagaimana teknologi klasifikasi tinggi berbasis citra, terutama menggunakan algoritma CNN, dapat diterapkan untuk mengidentifikasi secara akurat dalam mengklasifikasi tinggi tanaman menggunakan image?

3. Tujuan Penelitian

4. Mengimplementasikan model Convolutional Neural Network (CNN) untuk klasifikasi gambar tanaman jagung.

5. Mengembangkan dan menguji sistem klasifikasi tinggi tanaman jagung berbasis citra dengan menggunakan algoritma CNN, sehingga dapat memberikan hasil yang akurat.

6. Manfaat Penelitian

7. Dalam penelitian ini diharapkan akan memberikan manfaat sebagai berikut :

Membantu petani jagung dalam mendeteksi tinggi tanaman secara dini, yang dapat berkontribusi dalam pengelolaan pertanian yang lebih baik.

Meningkatkan produktivitas dan keberlanjutan sektor pertanian, khususnya bagi para petani baru.

Penelitian ini difokuskan pada klasifikasi tinggi tanaman jagung menggunakan gambar atau citra digital.

1. Ketersediaan Data :

2. Membatasi penelitian, berdasarkan ketersediaan data yang relevan. Misalnya, penelitian dapat membatasi diri pada dataset tertentu yang mencakup variasi yang cukup untuk melatih dan menguji model CNN.

**2. Metode Penelitian**

Dua faktor yang sangat penting yang harus dipertimbangkan selama proses penelitian adalah lokasi dan Durasi penelitian. Kedua faktor ini berperan penting dalam memastikan validitas dan keakuratan temuan penelitian.

Lokasi penelitian merupakan suatu lokasi atau objek tertentu yang akan dijadikan tempat dilakukannya suatu penelitian. Pemilihan lokasi penelitian merupakan hal yang penting dalam proses penelitian, karena akan memperlancar tugas peneliti. Tempat pengumpulan data untuk penelitian ini, di lakukan di Desa Pakatto, Kec. Bontomarannu, Kab. Gowa.

Adapun waktu penelitian yang akan di selesaikan dimulai pada bulan Mei 2024 sampai proses pengumpulan data selesai.

Diantara alat dan bahan yang digunakan dalam penelitian ini adalah sebagai Berikut :

Persyaratan Perangkat Keras (Hardware)

Laptop Acer Aspire 5 a

RAM 8,00 GB

OS Windows 11

Intel Core i3

HP Vivo Y21

RAM 4,00GB

Tripod

Meteran

Persyaratan Perangkat Lunak (Software)

1. Google Kolaborasi

2. Kumpulan data

3. Operasi Sistem Windows 11

4. Honeyview

Mempersiapkan data mentah untuk pemrosesan sistem adalah tujuan dari proses prapemrosesan. Dua dari tiga kelas foto yang menunjukkan ukuran tanaman jagung dipilih untuk langkah pra-pemrosesan penelitian ini. Setelah terpilih, kumpulan gambar buah-buahan tersebut Akses Google Drive yang berfungsi sebagai media pengembangan Google Collaboratory.

Peneliti menggunakan Confusion Matrix sebagai teknik pengujian untuk menguji akurasi dari model yang dibuat, teknik ini membagi antara total jumlah dari data dengan total jumlah data yang bisa terdeteksi. Perhitungan akurasinya dapat dihtiung menggunakan rumus (1).

TP + TN

Akurasi =

Membuat kesimpulan adalah langkah terakhir dalam proses analitis. Pada titik ini, peneliti mulai menyelidiki hipotesis penelitian, korelasi sebab-akibat, dan arti dari variabel yang digunakan. Untuk mencari solusi atas permasalahan yang dihadapi, data yang dikumpulkan dibandingkan satu sama lain.

**3. Hasil dan diskusi**

Pada bab ini, akan disajikan hasil dan pembahasan mengenai tinggi tanaman jagung menggunakan deteksi citra di Desa Pakatto, Kabupaten Gowa. Untuk hasil eksperimen berupa dataset yang meliputi data testing, dan data training, dengan masing-masing di isi dengan dua penyakit yaitu penyakit kuning dan penyakit layu bakteri. Dimana jumlah dataset gambar yaitu 1019 gambar.

mengukur tinggi tanaman jagung secara manual di salah satu lahan milik petani yang berada di desa pakatto, kabupaten gowa. Pengukuran secara manual ini memiliki akurasi 100% dengan menggunakan alat pengukur tukang bangunan dengan mengukur dengan tinggi tanaman 20 cm, 50 cm dan 110 cm.

Dataset yang digunakan dalam penelitian ini merupakan data gambar batang untuk tinggi tanaman yang diperoleh dari lahan pertanian warga Desa Pakatto, Kabupaten Gowa sebanyak 1019 data, dataset ini kemudian di bagi menjadi data testing, dan data training. Citra dari tinggi tanaman berdasarkan batang yang diambil yaitu adalah citra berwarna dan memiliki size 100 dengan pixel 256. Dalam penelitian ini menggunakan 3 kelas, yaitu tinggi tanaman 20cm, 50cm dan 110cm. Adapun hasil pengumpulan data di Desa Pakatto, Kabupaten Gowa antara lain sebagai berikut:

1. Memuat Dataset

2. Pada tahap ini digunakan dalam lingkungan Google Colab untuk menghubungkan Google Drive ke dalam lingkungan tersebut. Ini memungkinkan untuk mengakses file dan folder yang ada di Google Drive langsung dari lingkungan Colab tanpa perlu mengunggah atau mengunduh secara manual.

3. from google.colab import drive

4. drive.mount('/content/drive')

5. Kemudian mengimpor berbagai pustaka dan modul yang digunakan dalam pengembangan model jaringan saraf tiruan (neural network) menggunakan TensorFlow dan Sklearn.

6. from sklearn.model\_selection import train\_test\_split

7. from sklearn.preprocessing import LabelBinarizer

8. from tensorflow.keras.models import Sequential

9. from tensorflow.keras.layers import BatchNormalization

10. from tensorflow.keras.layers import Dense, Dropout, Flatten,Conv2D, MaxPooling2D, Activation

11. from tensorflow.keras import backend as K

12. from tensorflow.keras.preprocessing import image

13. from tensorflow.keras.preprocessing.image import ImageDataGenerator

14. from tensorflow.keras.optimizers import Adam, SGD

15. from tensorflow.keras.preprocessing.image import img\_to\_array

16. from tensorflow.keras.callbacks import EarlyStopping

17. from tensorflow.keras.callbacks import ModelCheckpoint

18. import numpy as np

19. import matplotlib.pyplot as plt

20. import cv2

21. import pickle

22. from os import listdir

23. Pada tahap ini terkait dengan pengaturan parameter dan konfigurasi yang di perlukan untuk melatih sebuah model jaringan saraf tiruan (neural network) dengan dataset gambar.

24. Dalam hal ini epoch dalam pengelolah citra adalah literasi dalam model atau system. Dimana setiap epoch melibatkan proses memasukkan data, pelatihan dalam model, menghitung hasil prediksi, dan membandingkan hasil tersebut dengan data yang ada. Tujuan menggunakan banyak epoch adalah untuk memperbaiki kinerja model atau sistem seiring dengan berjalannya waktu. Dimana pada source code dibawah menggunakan epoch 10 agar prediksi yang diingankan tercapai.

25. Dan batch size mengacu pada jumlah data pelatihan yang diberikan kepada model dalam satu iterasi. Ukuran batch size dapat mempengaruhi waktu pelatihan dan menggunakan batch size yang tepat bisa membantu meningkatkan kecepatam pelatihan dan efisiensi. Dimana dalam source code menggunakan batch size 8 dalam pelatihan model.

EPOCHS = 10

INIT\_LR = 0.001

BS = 8

# BS = 32

# opt = SGD(lr=INIT\_LR, decay=0.0001)

# opt = Adam(learning\_rate=INIT\_LR, decay=INIT\_LR / EPOCHS)

opt = Adam(learning\_rate=INIT\_LR)

mc = ModelCheckpoint(r'/content/Forest Fire CNN/ffs\_cnn.h5', monitor='val\_accuracy', mode='max', verbose=1, save\_best\_only=True)

# mc = ModelCheckpoint(r'/G:\My Drive\Ikbal\Dataset\Training/ffs\_cnn.h5', monitor='val\_accuracy', verbose=1)

default\_image\_size = tuple((256, 256))

image\_size = 0

directory\_root = '/content/drive/MyDrive/Ikbal/Dataset/Training'

width=256

height=256

depth=3

1. Selanjutnya memiliki tujuan untuk membaca, mengonversi, dan menyesuaikan gambar menjadi bentuk array yang siap untuk diolah oleh model jaringan saraf tiruan.

2. def convert\_image\_to\_array(image\_dir):

3. try:

4. image = cv2.imread(image\_dir)

5. if image is not None :

6. image = cv2.resize(image, default\_image\_size)

7. return img\_to\_array(image)

8. else :

9. return np.array([])

10. except Exception as e:

11. print(f"Error : {e}")

12. return None

13. Pada tahap kode ini melakukan beberapa hal terkait dengan memuat data gambar dari direktori tertentu dan mempersiapkannya untuk pelatihan model. Mari kita bahas langkah-langkahnya:

14. Image list, label list = [], [] : Ini adalah inisialisasi dua list kosong yang akan digunakan untuk menyimpan array gambar dan label yang berkaitan.

15. Root dir = listdir(directory root) : ): Ini mendapatkan daftar file dan folder dalam directory\_root, yang merupakan lokasi utama tempat gambar-gambar dataset ditempatkan.

16. Loop pertama (for directory in root dir): Loop ini digunakan untuk mengakses setiap folder yang mewakili kategori gambar, seperti berbagai jenis penyakit tanaman.

17. Loop kedua (for plant disease folder in plant disease folder list:): Loop ini digunakan untuk mengakses setiap folder yang mewakili kategori gambar, seperti berbagai jenis penyakit tanaman.

18. Loop ketiga (for image in plant\_disease\_image\_list:): Loop ini digunakan untuk mengakses setiap gambar dalam subfolder penyakit tanaman. Ini adalah bagian yang memuat dan memproses gambar.

19. Image\_directory=f"{directory\_root}/{plant\_folder}/{plant\_disease\_folder}/{image}": Ini adalah jalur lengkap ke gambar yang sedang diproses.

20. Kondisi if image\_directory.endswith(".jpg") == Trueor image\_directory.endswith(".JPG") = True or image\_directory.endswith(".jpeg") == True: Kondisi ini memastikan bahwa hanya berkas gambar dengan ekstensi .jpg, .JPG, atau .jpeg yang akan diproses.

21. Label\_list.append(plant\_disease\_folder): Label yang sesuai dengan jenis penyakit dari subfolder akan ditambahkan ke label\_list.

22. Akhir dari semua loop menunjukkan bahwa proses memuat gambar dan label selesai. Kegunaan dari kode ini adalah untuk memuat gambar-gambar dari berbagai kategori (misalnya, jenis penyakit tanaman) dalam dataset, mengubahnya menjadi bentuk array yang sesuai dengan format input yang dibutuhkan oleh model neural network, dan menghubungkannya dengan label yang sesuai. Hal ini diperlukan dalam tahap persiapan data sebelum memulai pelatihan model.

image\_list, label\_list = [], []

# try:

print("[INFO] Loading images ...")

root\_dir = listdir(directory\_root)

# root\_dir = directory\_root

for directory in root\_dir :

# remove .DS\_Store from list

if directory == ".DS\_Store" :

root\_dir.remove(directory)

for plant\_folder in root\_dir :

plant\_disease\_folder\_list = listdir(f"{directory\_root}/{plant\_folder}")

for disease\_folder in plant\_disease\_folder\_list :

# remove .DS\_Store from list

if disease\_folder == ".DS\_Store" :

plant\_disease\_folder\_list.remove(disease\_folder)

for plant\_disease\_folder in plant\_disease\_folder\_list:

print(f"[INFO] Processing {plant\_disease\_folder} ...")

plant\_disease\_image\_list = listdir(f"{directory\_root}/{plant\_folder}/{plant\_disease\_folder}/")

for single\_plant\_disease\_image in plant\_disease\_image\_list :

if single\_plant\_disease\_image == ".DS\_Store" :

plant\_disease\_image\_list.remove(single\_plant\_disease\_image)

for image in plant\_disease\_image\_list:

image\_directory = f"{directory\_root}/{plant\_folder}/{plant\_disease\_folder}/{image}"

if image\_directory.endswith(".jpg") == True or image\_directory.endswith(".JPG") == True or image\_directory.endswith(".jpeg") == True:

image\_list.append(convert\_image\_to\_array(image\_directory))

label\_list.append(plant\_disease\_folder)

print("[INFO] Image loading completed")

# except Exception as e:

#     print(f"Error : {e}")

1. Tahap ini memiliki tujuan untuk menghitung jumlah total gambar yang telah dimuat ke dalam image\_list dan kemudian menyimpan jumlah tersebut dalam variabel image\_size. Dan juga, langkah ini adalah untuk mendapatkan informasi tentang berapa banyak data gambar yang telah dimuat dan siap untuk diproses oleh model neural network.

2. image\_size = len(image\_list)

3. image\_size

4. Tahap ini berkaitan dengan pemrosesan label yang berkaitan dengan kategori (kelas) gambar yang dimuat. Dan juga, dari kode ini adalah untuk mengonversi label kategori menjadi bentuk biner yang sesuai untuk proses pelatihan, dan juga untuk menyimpan objek LabelBinarizer yang digunakan dalam format pustaka pickle. Mari kita bahas langkah-langkahnya:

5. Label\_binarizer = LabelBinarizer(): Ini adalah objek LabelBinarizer yang digunakan untuk mengonversi label kategori menjadi bentuk biner. Objek ini digunakan untuk mentransformasi label kategori menjadi bentuk yang dapat diolah oleh model, khususnya dalam kasus klasifikasi dengan beberapa kelas.

6. Image\_labels = label\_binarizer.fit\_transform(label\_list): Di sini, objek LabelBinarizer yang telah dibuat digunakan untuk mengonversi label\_list (daftar label kategori) menjadi bentuk biner. Setiap kategori akan diwakili dalam bentuk biner, di mana setiap kelas akan memiliki nilai 0 atau 1 untuk setiap sampel gambar.

7. Pickle.dump(label\_binarizer,open(r'forest\_fire\_smoke\_transform.pkl', 'wb')): Ini adalah langkah yang sangat penting. Objek label\_binarizer disimpan dalam format biner menggunakan modul pickle. Ini memungkinkan untuk menggunakan transformasi yang sama pada data uji atau data baru yang Anda ingin klasifikasikan di masa mendatang.

8. n\_classes = len(label\_binarizer.classes\_): Ini menghitung jumlah total kelas atau kategori dalam dataset. Hal ini digunakan untuk menentukan jumlah unit output pada lapisan akhir dari model .

9. label\_binarizer = LabelBinarizer()

10. image\_labels = label\_binarizer.fit\_transform(label\_list)

11. pickle.dump(label\_binarizer,open(r'forest\_fire\_smoke\_transform.pkl', 'wb'))

12. n\_classes = len(label\_binarizer.classes\_)

13. Kemudian dalam tahap ini digunakan untuk mencetak daftar kelas atau kategori yang telah diidentifikasi oleh objek LabelBinarizer. Setiap kelas akan dicetak dalam urutan yang sesuai.

14. print(label\_binarizer.classes\_)

15. Dalam tahap ini memiliki tujuan untuk mengubah dan menormalkan array gambar yang ada dalam image\_list. Jadi, baris ini secara efektif mengubah semua array gambar dalam image\_list menjadi array float16 yang dinormalisasi antara 0 dan 1.

16. image\_list = np.array(image\_list, dtype=np.float16) / 225.0

17. Selanjutnya tahap ini digunakan untuk membagi data gambar dan label menjadi set data latih dan data uji. dari baris ini adalah untuk membagi dataset gambar dan label menjadi dua set yang terpisah yaitu satu untuk pelatihan dan satu untuk pengujian (validasi).

18. x\_train, x\_test, y\_train, y\_test = train\_test\_split(image\_list, image\_labels, test\_size=0.1, random\_state=42)

19. Pada tahap ini digunakan untuk mengimpor modul "garbage collection" (gc) yang ada dalam Python. Garbage collection adalah proses otomatis pemulihan memori yang tidak terpakai agar bisa digunakan kembali.

20. import gc

21. gc.collect()

22. Kemudian dari tahap ini digunakan untuk menghapus variabel image\_list dan image\_labels dari memori. Ini adalah tindakan yang disebut "penghapusan variabel" atau "dealokasi variabel". Langkah ini juga untuk membebaskan memori yang ditempati oleh variabel-variabel ini setelah selesai menggunakannya.

23. del(image\_list)

24. del(image\_labels)

25. Tahap ini digunakan untuk menghasilkan variasi dari dataset gambar yang ada dengan melakukan transformasi pada gambar-gambar tersebut. Dan juga, dari kode ini adalah untuk membuat objek ImageDataGenerator dengan konfigurasi transformasi gambar yang dapat digunakan selama pelatihan model.

26. aug = ImageDataGenerator(

27. rotation\_range=25, width\_shift\_range=0.1,

28. height\_shift\_range=0.1, shear\_range=0.2,

29. zoom\_range=0.2,horizontal\_flip=True,

30. fill\_mode="nearest")

31. Tahap ini digunakan untuk memanggil gc.collect() adalah perintah yang digunakan untuk secara manual memicu proses pemulihan memori yang tidak terpakai oleh kolektor sampah (garbage collector) dalam Python. Kegunaan dari gc.collect() adalah untuk membebaskan memori yang tidak terpakai segera, alih-alih hanya mengandalkan garbage collector untuk menjalankan proses pemulihan memori ketika diperlukan

gc.collect()

Convolutional Neural Network adalah metode jaringan saraf mendalam, yang paling umum diterapkan untuk menganalisis gambar visual. Convolutional Neural Network juga disebut sebagai multilayer perceptron yang setiap neuronnya terhubung ke semua neuron dilapisan berikutnya. Pada tahap ini yaitu pembuatan model Convolutional Neural Network pada sistem yang akan dibuat.

def fit\_model(n\_nodes, idx):

###################

#define model arch

model = Sequential()

inputShape = (height, width, depth)

chanDim = -1

if K.image\_data\_format() == "channels\_first":

inputShape = (depth, height, width)

chanDim = 1

### 1

model.add(Conv2D(n\_nodes[idx][0], (3, 3), padding="same",input\_shape=inputShape))

model.add(Activation("relu"))

# model.add(BatchNormalization(axis=chanDim))

model.add(MaxPooling2D(pool\_size=(3, 3)))

# model.add(Dropout(0.25))

### 2

model.add(Conv2D(n\_nodes[idx][1], (3, 3), padding="same"))

model.add(Activation("relu"))

# model.add(BatchNormalization(axis=chanDim))

model.add(MaxPooling2D(pool\_size=(2, 2)))

# model.add(Dropout(0.25))

### 3

model.add(Conv2D(n\_nodes[idx][2], (3, 3), padding="same"))

model.add(Activation("relu"))

# model.add(BatchNormalization(axis=chanDim))

model.add(MaxPooling2D(pool\_size=(2, 2)))

# model.add(Dropout(0.25))

#     ## 4

#     model.add(Conv2D(n\_nodes[idx][3], (3, 3), padding="same"))

#     model.add(Activation("relu"))

#     model.add(BatchNormalization(axis=chanDim))

#     model.add(MaxPooling2D(pool\_size=(2, 2)))

# #     model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(n\_nodes[idx][3]))

model.add(Activation("relu"))

# model.add(BatchNormalization())

# model.add(Dropout(0.5))

model.add(Dense(3))

model.add(Activation("softmax"))

#model.add(Activation("sigmoid"))

###################

#compile model & fit

# model.compile(loss="binary\_crossentropy", optimizer=opt,metrics=["accuracy"])

# model.compile(loss="binary\_crossentropy", metrics=["accuracy"])

model.compile(loss="categorical\_crossentropy", metrics=["accuracy"])

print(f'Training {idx+1}st Arch!')

history = model.fit(

aug.flow(x\_train, y\_train, batch\_size=BS),

validation\_data=(x\_test, y\_test),

steps\_per\_epoch=len(x\_train) // BS,

epochs=EPOCHS, verbose=1, callbacks=[mc]

# epochs=EPOCHS, verbose=1

)

###################

#plot learning curves

acc = history.history['accuracy']

epoch = range(1, len(acc) + 1)

plt.subplot(len(n\_nodes),2,(idx+1)\*2-1)

plt.plot(epoch, history.history['accuracy'], label='train acc')

plt.plot(epoch, history.history['val\_accuracy'], label='val acc')

plt.title(f'Accuracy, node {n\_nodes[idx]}')

plt.legend()

plt.subplot(len(n\_nodes),2,(idx+1)\*2)

plt.plot(epoch, history.history['loss'], label='train loss')

plt.plot(epoch, history.history['val\_loss'], label='val loss')

plt.title(f'Loss, node {n\_nodes[idx]}')

plt.legend()

Pada tahap ini kegunaan dari fungsi ini adalah untuk merancang, mengkompilasi, melatih, dan memvisualisasikan hasil dari beberapa arsitektur model yang berbeda (ditentukan oleh n\_nodes). Fungsi ini memungkinkan untuk menguji berbagai konfigurasi arsitektur dan melihat bagaimana model bereaksi terhadap data pelatihan dan validasi.

Fungsi fit\_model (n\_nodes, idx) digunakan untuk mengatur, mengkompilasi, dan melatih model jaringan saraf tiruan (neural network) dengan arsitektur yang telah ditentukan. Mari kita bahas langkah-langkah yang dilakukan oleh fungsi ini:

1. Definisi Model: Fungsi ini memulai dengan mendefinisikan arsitektur model menggunakan modul Sequential dari TensorFlow Keras. Setiap layer model, termasuk layer konvolusi, aktivasi, max pooling, dan fully connected, ditambahkan sesuai dengan arsitektur yang telah ditentukan.

2. Compile Model: Setelah definisi model selesai, model dikompilasi dengan fungsi kerugian (loss function) 'binary\_crossentropy' dan metrik evaluasi 'accuracy'. Ini adalah langkah penting sebelum model dapat dilatih.

3. Pelatihan Model: Model dilatih menggunakan fungsi fit. Data latih diproses dengan augmentasi yang ditentukan oleh objek ImageDataGenerator aug dan validasi dilakukan pada data uji x\_test dan y\_test. Epochs, ukuran batch (BS), dan parameter lainnya ditentukan.

4. Visualisasi Learning Curves: Setelah pelatihan selesai, nilai akurasi dan kerugian selama pelatihan dan validasi disimpan dalam variabel history. Fungsi ini mencetak kurva pembelajaran (learning curves) dalam bentuk grafik menggunakan matplotlib.

Fungsi ini menerima dua argumen:

n\_nodes: Ini adalah daftar (list) yang berisi tiga angka yang mewakili jumlah node (filter) pada setiap lapisan konvolusi dalam arsitektur model.

idx: Ini adalah indeks yang mengacu pada urutan iterasi saat memproses arsitektur dari n\_nodes.

Dalam pemakaian fungsi sigmoid pada saat pemodelan sangat membantu untuk meningkatkan akurasi dikarenakan fungsi sigmoid menghasilkan probabilitas output yang berkisar antara 0 dan 1. Dan itu menyebabkan tingkat akurasi meningkat dengan stabil dalam kurva akurasi.

Dan pengujian yang dipakai yaitu pengujian faktual dimana pengujian faktual adalah tahap dimana pengelola citra di uji menggunakan data yang sebenarnya. Tujuannya untuk memverifikasi keakuratan dan kinerja model dalam memberikan solusi dalam penyelesaian masalah dengan metode Convolutional Neural Network .

Pada tahap ini yaitu pengujian metode Convolutional Neural Network untuk menampilkan tingkat akurasi dan gambar tinggi pada tanaman jagung.

1. Pengujian Tingkat Akurasi

n\_nodes = [

[256, 128, 64, 32]

]

plt.figure(figsize=(13, 4\*len(n\_nodes)))

for x in range(len(n\_nodes)):

fit\_model(n\_nodes, x)

print(f'{x+1}st Arch finished !')

# show learning curves

plt.show()

1. Penggunaan variabel n\_nodes untuk menguji berbagai arsitektur jaringan saraf tiruan dengan jumlah node (filter) yang berbeda pada setiap lapisan konvolusi. Mari kita bahas langkah-langkahnya:

2. n\_nodes: Ini adalah variabel yang berisi daftar arsitektur yang ingin diuji. Setiap sublist dalam n\_nodes menunjukkan jumlah filter yang akan digunakan dalam lapisan konvolusi pada setiap arsitektur.

3. plt.figure(figsize=(13, 4\*len(n\_nodes))) : Ini adalah perintah untuk membuat gambar plot dengan ukuran yang diatur sesuai dengan banyaknya arsitektur yang ingin diuji. Setiap arsitektur akan memiliki subplot dalam gambar.

4. Looping melalui setiap arsitektur dalam n\_nodes:

5. Memanggil fungsi fit\_model(n\_nodes, x) untuk melatih dan menguji model dengan arsitektur yang sesuai.

6. Mencetak pesan yang menunjukkan bahwa arsitektur ke-x telah selesai dilatih dan diuji.

7. plt.show(): Ini adalah perintah untuk menampilkan semua plot hasil dari proses pelatihan dan pengujian model dengan berbagai arsitektur.

8. Jadi, dalam kode ini melakukan iterasi melalui daftar arsitektur dalam n\_nodes, melatih dan menguji model menggunakan setiap arsitektur, dan akhirnya menampilkan plot hasil pelatihan dan pengujian untuk semua arsitektur yang diuji.

Dari gambar diatas dapat dilihat bahwa kondisi kurva akurasi meningkat dengan epoch 10 dan memperoleh nilai akurasi training mencapai kestabilan 80%. Pada saat epoch ke-5 sudah mencapai 0.8730 hingga epoch ke-10 dalam mendeteksi tinggi tanaman jagung. Akurasi adalah perbandingan antara data yang ingin diprediksi apakah benar sesuai dengan kelas target yang ingin dicapai dengan keseluruhan data oleh model klasifikasi. Akurasi juga digunakan untuk mengukur seberapa baik data diprediksi dengan benar.

Dari gambar diatas dapat dilihat bahwa kondisi kurva loss menurun dengan memperoleh nilai training loss 0.2308 merupakan suatu fungsi dalam mengukur seberapa baik sebuah prediksi atau diagnosa dengan cara mengukur jarak hasil dari output prediksi dengan nilai target yang ingin dicapai. Loss juga menghasilkan nilai kesalahan atau eror antara output training dan target yang ingin dicapai.

Dari gambar diatas dapat dilihat bahwa gambar yang terdeteksi adalah tinggi tanaman jagung dengan title “20 cm” yang sesuai dengan dataset validasi. Dan dalam pengujian identifikasi citra tinggi tanaman jagung untuk mengetahui tinggi tanaman tersebut termasuk jenis kelas tinggi tanaman yang mana.

Dari gambar diatas dapat dilihat bahwa gambar yang terdeteksi adalah tinggi tanaman jagung bakteri dengan title “50 cm” yang sesuai dengan dataset validasi. Dan dalam pengujian identifikasi citra penyakit untuk mengetahui tinggi tanaman tersebut termasuk jenis kelas yang mana.

Dan dari hasil pengujian di atas mulai dari kurva akurasi dan loss, pengujian penyakit pada masing-masing kelas dengan metode Convolutional Neural Network (CNN) faktanya dapat membantu kinerja pengelola citra dalam mendeteksi tinggi pada tanaman jagung.

1. Adapun dataset gambar yang tersedia yaitu dataset gambar tinggi tanaman jagung dengan tiga kelas,. Berikut hasil diagnose dari masing- masing dataset gambar.

2. Table Hasil Pengujian

Pada tabel diatas adalah hasil pengujian antara dataset gambar tinggi tanaman 20cm, 50cm dan 110cm pada tanaman jagung. Dimana pada dataset gambar tinggi tanaman, akurasi yang di dapat mencapai 80% untuk 5 kali percobaan testing yang dilakukan dengan hasil diagnosa adalah gambar tinggi pada tanaman jagung yaitu 20 cm, 50 cm dan 110 cm.

Pada tabel diatas adalah hasil pengukuran manual tinggi tanaman 20cm, 50cm dan 110cm pada tanaman jagung. Dimana pada pengukuran manual tinggi tanaman, akurasi yang di dapat mencapai 100% untuk beberapa kali percobaan yang dilakukan dengan hasil diagnosa tinggi pada tanaman jagung yaitu 20 cm, 50 cm dan 110 cm.

**4. Kesimpulan**

1. Kesimpulan 47

2. Saram 47

DAFTAR PUSTAKA 48

Bab terakhir ini berisi rangkuman dan kesimpulan dari hasil penelitian, serta memberikan rekomendasi sesuai dengan keterbatasan yang ada dalam sistem yang dikembangkan.

Dari hasil penelitian yang dilakukan, Adapun kesimpulan yaitu sebagai berikut:

Penelitian ini berhasil mengimplementasikan algoritma Convolutional Neural Network (CNN) untuk klasifikasi gambar tinggi tanaman jagung. CNN terbukti efektif dalam memproses dan mengenali pola visual dari citra tanaman jagung, yang dapat dihubungkan dengan tinggi tanaman secara akurat. Hasil pengujian menunjukkan bahwa model CNN yang dikembangkan mampu memberikan akurasi yang baik dalam mengklasifikasikan tinggi tanaman jagung, sehingga dapat digunakan sebagai alat bantu dalam pengelolaan pertanian, khususnya dalam pemantauan pertumbuhan tanaman.

Dari hasil penelitian yang dilakukan, dapat disimpulkan beberapa saran yaitu sebagai berikut:

Penelitian ini dapat dikembangkan lebih lanjut dengan menggunakan dataset yang lebih besar dan lebih beragam untuk meningkatkan akurasi dan generalisasi model CNN. Selain itu, penerapan teknik augmentasi data juga dapat dieksplorasi untuk memperkaya dataset yang ada.

**Referensi**

[1] VBai et al. (2018). Deep Convolutional Neural Network Optimization for Extensive Image Retrieval. Pages 60–67 in Volume 303 of Neurocomputing.

[2] In 2017, Guo, T., Dong, J., Li, H., and Gao, Y. Convolutional Neural Network for Image Classification: A Basic Approach. IEEE Beijing.

[3] In 2018, Bennamoun, M.; Khan, S.; Rahmani, H.; Shah, S. A. A. Morgan & Claypool Publishers, s.l., A Guide to Convolutional Neural Networks for Computer Vision.

[4] Li, J., et al. (2020). "Judul Jurnal Terkait." Nama Jurnal, Volume(Issue), Halaman- Halaman. DOI: [DOI Jurnal]

[5] In 2019, Mohamed, O., Brahim, A., Khalid, E. A., and Mohamed, O. Convolutional Neural Networks for Content-Based Image Retrieval. pages 463–476).

[6] S. Naik. (2017). Fruit Grading and Classification Using Machine Vision. Volume 170 of the International Journal of Computer Application.

[7] In 2017, Naik, S. and Patel, B. Fruit classification based on machine vision. Journal of Computer Applications International, 170(9).

[8] Wang, Q., & Zhang, L. (2022). Judul Buku Terkait. Penerbit Buku. ISBN: [Nomor ISBN Buku].

[9] Mahdiyah, U. (2023). KLASIFIKASI KUALITAS CITRA CABAI DENGAN MENGGUNAKAN ALGORITMA GRADIEN BOOSTING. JAMI: Jurnal Ahli Muda Indonesia, 4(1), 61–69.

[10] Sely Wita, D., & Yanti Liliana, D. (2022). Klasifikasi Identitas Dengan Citra Telapak Tangan Menggunakan Convolutional Neural Network (CNN). JURTI, 6(1).

[11] Sri Rahmadhani, U., & Lysbetti Marpaung, N. (2023). Klasifikasi Jamur Berdasarkan Genus Dengan Menggunakan Metode CNN. 8(2).

[12] Teknik Elektro, J., & Wega Intyanto, G. (n.d.). Klasifikasi Citra Bunga dengan Menggunakan Deep Learning: CNN (Convolution Neural Network).

[13] Wolas, N., and M. H. Endah (undated). Convolutional Neural Network (CNN) Algorithm is Used by Deep Learning Applications to Classify Burung Familiar Accipitridae.In 2021, Telkom, U., Novamizanti, L., Rizal, S., and Zakiya, P. N. Classification of the pathology of macula retina by OCT image usage (KLASIFIKASI PATOLOGI MAKULA RETINA MELALUI CITRA OCT USING CONVOLUTIONAL NEURAL NETWORK DENGAN, 8(5), 5072– 5082).

[14] Sun, Y., Nanehkaran, Y. A., Suzauddola, M., Zhang, D., & Chen, J. (2021). Squeeze- and-excitation MobileNet model with twice transfer learning for plant disease picture identification. 15 (5) IET Image Processing, 1115–1127. The publication 10.1049/ipr2.12090

[15] Wang, T., Warden, P., Regev, S., Rhodes, R., David, R., Duke, J., Jain, A., Reddi, V. J., Jeffries, N., Li, J., Kreeger, N., Nappier, I., and Natraj, M. (2020). Embedded Machine Learning on TinyML Systems using TensorFlow Lite Micro. 2010.08678 can be found at arxiv.org.

[16] Susanti Diana, Yeni Astuti, and Dora Fatma Nurshanti. (2019). PRODUKSI JAGUNG MANIS DAN PENGARUH PEMBERIAN AIR TERHADP PERTUMBUHAN (Zea mays). 2579–5171.