JURNAL RISET REKAYASA ELEKTRO
Vol.5, No.2, Desember 2023, Hal. 145~154
P-ISSN: 2685 - 4341 E-ISSN: 2685 - 5313

Identifikasi Kinerja Arsitektur *Transfer Learning* VGG16, ResNet-50, dan Inception-V3 Dalam Pengklasifikasian Citra Penyakit Daun Tomat

Identification of VGG16, ResNet-50, and Inception-V3 Transfer Architecture Performance in Image Classification of Tomato Leaf Diseases

Muhammad Iqbal Fathur Rozi[1], Nelly Oktavia Adiwijaya[2], Dwiretno Istiyadi Swasono[3]
[1,2,3] Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Jember
Jl. Kalimantan Tegalboto No.37, Krajan Timur, Sumbersari, Kec. Sumbersari, Kabupaten Jember, Jawa Timur 68121, Indonesia

Informasi Makalah

Dikirim, 18 Juni 2023
Diterima, 6 Desember 2023
Diterbitkan, 14 Desember 2023

Kata Kunci*:*

Penyakit Daun Tomat,
Convolutional Neural Network,
Transfer Learning.

INTISARI

Tomat merupakan salah satu tumbuhan holtikultura sekaligus tanaman musiman yang banyak dikonsumsi di Indonesia. Produksi tanaman tomat sering kali terancam oleh serangan hama dan penyakit, sehingga diperlukannya campur tangan teknologi dalam pengidentifikasian penyakitnya. Teknologi untuk mengidentifikasi penyakit yang terfokus pada daun tomat ini menggunakan pengolahan citra dengan metode CNN. Penelitian ini diharapkan dapat menghasilkan perbandingan arsitektur CNN yang terbaik secara akurasi. Terdapat tiga arsitektur CNN yang dibandingkan dalam penelitian ini yaitu, arsitektur VGG16, ResNet50 dan Inception-V3. Dalam pengimplementasiannya, ketiga arsitektur tersebut diberikan perlakuan yang sama seperti penggunaan input piksel, penambahan layers model, dan lain sebagainya. Penelitian ini menghasilkan tingakt akurasi yang berbeda beda. Arsitektur Inception-V3 mendapatkan nilai akurasi dan validasi akurasi sebesar 0.9551 dan 0.9544. Arsitektur ResNet50 mendapatkan nilai akurasi sebesar 0.9578 dan nilai validasi akurasi sebesar 0.9467. Dan nilai akurasi tertinggi didapat dengan nilai akurasi sebesar 0.9754 dan nilai validasi akurasi tertinggi pada 0.9778 menggunakan arsitektur VGG16.

Keyword*:*

Tomato Leaves Disesase,
Convolutional Neural Network,
Transfer Learning.

ABSTRACT

Tomato is a horticultural plant as well as a seasonal plant which is widely consumed in Indonesia. Production of tomato plants is often threatened by pests and diseases, so it is necessary to intervene in the identification of the disease. The technology for identifying diseases focused on tomato leaves uses image processing using the CNN method. This research is expected to produce the best comparison of CNN architectures in terms of accuracy. There are three CNN architectures compared in this study, namely the VGG16, ResNet50 and Inception-V3 architectures. In its implementation, the three architectures are given the same treatment as the use of input pixels, the addition of model layers, and so on. This research produces different levels of accuracy. The Inception-V3 architecture obtained accuracy and validation values of 0.9551 and 0.9544. The ResNet50 architecture gets an accuracy value of 0.9578 and an accuracy validation value of 0.9467. And the highest accuracy value is obtained with an accuracy value of 0.9754 and the highest accuracy validation value is 0.9778 using the VGG16 architecture.

Korespondensi Penulis:
Muhammad Iqbal Fathur Rozi
Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Jember
Jl. Kalimantan Tegalboto No.37, Krajan Timur, Sumbersari, Kec. Sumbersari, Kabupaten Jember, Jawa Timur 68121, Indonesia
Email : ibamfathur01@gmail.com

1. PENDAHULUAN

Pembangunan pertanian sangat berperan penting dalam perekonomian nasional, yaitu sebagai pembentukan kapital, penyediaan bahan pangan, bahan baku industri, pakan dan bioenergi, penyerap tenaga kerja, sumber devisa negara, sumber pendapatan serta pelestarian lingkungan melalui praktik usaha tani yang ramah lingkungan [1]. Tomat merupakan salah satu tumbuhan holtikultura dan merupakan tanaman musiman yang banyak dikonsumsi oleh masyarakat Indonesia karena kaya akan air dan mengandung vitamin A dan vitamin C yang sangat bermanfaat bagi tubuh. Namun, dibalik produksi yang harus selalu mengalami peningkatan tomat merupakan tanaman yang sangat rentan terhadap serangan penyakit.

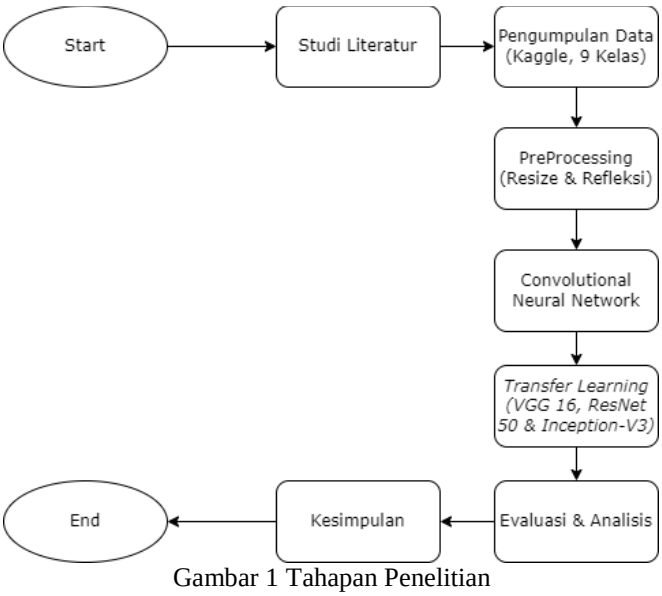
Umumnya, kendala utama menanam tomat adalah serangan hama dan penyakit, banyak varietas tomat yang mudah terserang penyakit busuk pangkal batang dan busuk daun [2]. Dari beberapa penyakit yang menyerang tanaman tomat banyak penyakit yang menyerang pada bagian daunnya. Namun bentuk daun tomat yang beragam tidak mudah bagi manusia untuk mendeteksinya terutama bagi masyarakat petani yang awam yang seringkali dilakukan pengenalan gejala hanya menggunakan mata telanjang saja [3]. Oleh karena itu perlu adanya campur tangan teknologi dan juga para ahli dalam mengidentifikasi penyakit pada daun tomat secara efektif dan akurat. Dalam hal ini teknologi yang dimaksud adalah penggunaan citra yang dapat digunakan untuk mengidentifikasi jenis penyakit pada daun tomat.

Dikarenakan pengidentifikasian penyakit pada daun tomat menggunakan pengolahan data menggunakan citra digital[4]. Metode penelitian yang digunakan adalah machine learning berupa arsitektur *Convolutional Neural Network* (CNN). CNN sendiri merupakan model dari machine learning yang sering digunakan dan dinilai canggih dalam pengklasifikasian citra gambar. Beberapa arsitektur CNN yang dapat digunakan dalam identifikasi objek adalah AlexNet, Visual Geometry Group (VGG) , Residual Network (ResNet) , GoogleNet, Inception-V3, InceptionResNetV2, Squeezenet dan lain lainnya.

Dari berbagai referensi yang telah dikemukakan Penggunaan tiga jenis arsitektur VGG16, ResNet-50 dan Inception-V3 dipilih karena memiliki akurasi yang terbaik dan menghasilkan hasil yang positif dalam pengklasifikasian citra digital pada objek tertentu[5][6][7]. Pada penelitian ini dilakukan identifikasi penyakit pada daun tomat dengan sembilan jenis penyakit, yaitu *Tomato Mosaic Virus, Target Spot, Bacterial Spot, Tomato Yellow Leaf Curl Virus, Late Blight, Leaf Mold, Early Blight, Tomato Healthy, Septoria Lea*f. Penelitian ini juga membandingkan ketiga jenis arsitektur CNN tersebut apabila diterapkan pada jumlah dataset yang terbatas.

2. METODE PENELITIAN

Metode penelitian merupakan bagian yang menjelaskan mengenai proses analisis data, tahapan penelitan yang dilakukan, serta evaluasi dan analisis dari hasil penelitian yang telah dilakukan. Penelitian ini melakuakan analisis dan evaluasi dari serta perbandingan dari model *Transfer Learning* yang digunakan untuk mengidentifikasi penyakit pada citra daun tomat. Berikut beberapa tahapan dari penelitian ini yang ditunjukan pada Gambar 1:



Gambar 1 Tahapan Penelitian

2.1 Studi Literature

Studi literatur merupakan proses yang dilakukan untuk mencari, membaca, mencatat, serta mengumpulkan seferensi teori yang digunakan untuk bahan penelitian. Studi literatur merupakan proses paling awal yang dilakukan dalam penelitian ini. Referensi yang telah dikumpulkan nantinya akan digunakan sebagai dasar teori penelitian.

2.2 Pengumpulan data

Dalam penelitian ini dataset yang digunakan merupakan data yang bersumber dari website penyedia data publik, yaitu www.kaggle.com. Dataset yang diambil memiliki judul *Tomato Leaf Disesase Detection* yang berasal dari author bernama KAUSTUBH B. Dataset tersebut berjumlah 11000 yang terbagi menjadi sembilan kelas penyakit daun tomat dan satu kelas daun tomat yang sehat. Kelas kelas tersebut terbagi menjadi beberapa label, diantaranya adalah *Tomato Mosaic Virus, Target Spot, Bacterial Spot, Tomato Yellow Leaf Curl Virus, Late Blight, Leaf Mold, Early Blight, Septoria Leaf, Spider mites Two spotted spider mite, dan Tomato Healthy*. Setiap kelas memiliki 1000 citra data training dan 100 citra data validation.

Dari *Dataset* mentah yang awalnya berjumlah 11000 citra gambar dari 10 kelas selanjutnya akan dipilah menjadi 9000 citra gambar saja dari 9 kelas. Pemilahan tersebut dibagi menjadi 1000 citra gambar setiap kelas dengan 900 citra gambar data training dan 100 citra gambar validation. Penghilangan pada salah satu kelas juga dilakukan pada penelitian ini. Pengilangan salah satu kelas disebabkan oleh kurangnya literasi terkait penyakit daun tomat *Spider mites Two-spotted spider mite*. Pemilahan data ini juga disebabkan untuk mempermudah proses split data dan juga untuk mendapatkan tingkat performa pemodelan yang baik dengan hanyak membatasi 1000 gambar saja.

2.3 Preprocessing

Preprocessing merupakan salah satu proses dari pengolahan citra yang bertujuan untuk mempermudah pengolahan citra dengan mendapatkan ekstraksi data. Pada penelitian ini, Preprocessing yang digunakan berupa resize, refleksi dan rotasi. Preprocessing refleksi atau flipping digunakan untuk menseragamkan arah citra. Pada penelitian ini refleksi yang digunakan adalah horizontal dan rotasi yang dilakukan dengan memutar citra gambar dengan sudut sudut yang ditentukan yaitu 30^o, 45^o, 60^o, 90^o. Sedangkan untuk *resize* yang digunakan adalah dengan memperkecil piksel dari 256 x 256 menjadi 224 x 224[8]. Hal ini dilakukan guna mempermudah dan meringankan proses perhitungan.

2.4 Metode Convolutional Neural Network

Pada tahapan ini citra yang sebelumnya telah di Preprocessing mulai dilakukannya identifikasi menggunakan algoritma Machine Learning, yaitu *Convolutional Neural Network* (CNN)[9]. Tahapan yang dilakukan dalam proses *Convolutional Neural Network* (CNN) ini dimulai dari tahapan *Convolutional Layer, Pooling Layer, dan Fully Connected Layer*[10]. Dari tahapan ini selanjutnya dilakukan identifikasi lanjutan dengan arsitektur *Transfer Learning* .

2.5 *Transfer Learning*

Penelitian ini menggunakan *Transfer Learning* yang diterapkan pada setiap percobaan yang ada guna mengetahui performa terbaik dari penggunaan arsitektur transfer leraning yang diberikan. Dikarenakan penerapan *Transfer Learning* , citra yang sudah dilatih sebelumnya kemudian diambil beberapa bagian untuk digunakan kembali atau pre-trained dalam identifikasi model yang baru[11]. Model CNN pre - trained dilatih menggunakan ImageNet yang merupakan kumpulan data publik besar yang berisi 9000 gambar untuk diklasifikasikan ke dalam 9 kelas. Penelitian ini mengusung tiga model pre-trained yang digunakan, yaitu VGG16, ResNet50 dan Inception-V3 yang masing – masing modelnya diujikan menggunakan dataset yang sama untuk diuji arsitektur mana yang terbaik. Berikut metode *Transfer Learning* yang digunakan pada penelitian kali ini adalah sebagai berikut:

a. VGG16

VGG16 atau kepanjangan dari *Visual Geometric Group* dengan 16 layer merupakan salah satu arsitektur dari CNN yang memiliki 16 layer berupa *convolutional layer* dan 3 *fully-connected layer*. Keunggunlan dari arsitektur ini adalah memiliki arsitektur yang terdiri dari layer convolutional 3 x 3 dan pooling 2 x 2 yang dinilai lebih akurat jika dibandingkan dengan arsitektur CNN sebelumnya. Keakuratan model ini diambil dari arsitektur CNN *state-of-the-art*, yaitu tolak ukur keakuratan gambar ini diambil dari ketiga model tersebut [12].

b. ResNet50

Residual Network50 atau yang biasa disebut dengan ResNet50 merupakan salah satu arsitektur CNN yang memiliki layer sebanyak 50. ResNet memiliki sistem skip yang bertujuan untuk melewati sistem komputasi perkalian. Konsep ini disebut dengan shortcut connections dalam konsep ini memanfaatkan input layer sebelumnya, dijadikan sebagai input terhadap output dari layer. Arsitektur ini berfungsi untuk menghindari gambar dari kehilangan informasi sekaligus mengatasi venish gradient [13]. Sesuai dengan arsitekturnya, ResNet50 memiliki konsep melewati 3 layer dan terdapat hanya 1 x 1 *convolutional Layer*.

c. Inception-V3

Inception-V3 merupakan salah satu model CNN yang memiliki 42 layer yang dinilai lebih efisien dalam segi kedalaman arsitektur, dan *error rate* yang lebih kecil. Inception-V3 mengusung sistem yang berbeda, yaitu menggunakan sistem pemfaktoran *convolutional layer* menjadi multi - layers dengan ukuran kernel yang terbilang lebih kecil. Arsitektur ini memiliki parameter yanglebih tepat karena mampu mengurangi nilai parameternya dengan membagikan bobotnya menjadi beberapa multi-layer [14].

2.6 Evaluasi & Analisis

Setelah dilakukannya identifikasi dengan 3 arsitektur *Transfer Learning* yang berbeda, selanjutnya merupakan Evaluasi dan Analisis. Pada tahapan ini dilakukannya uji akurasi dan analisis dari model *Convolutional Neural Network* (CNN) yang telah dilakukan. Uji akurasi ini digunakan untuk mengetahui performa dan efektifitas dari model *Convolutional Neural Network* (CNN) yang telah dilakukan. Setelah dilakukan analsisi dan uji akurasi selanjutnya dilakukan evaluasi perbadingan antara arsitektur transfer leraning untuk mencari arsitektur yang terbaik.

2.7 Kesimpulan

Pada bagian kesimpulan merupakan bagian paling akhir sekaligus menjadi rangkuman dari semua penelitian yang telah dilakukan. Tahapan ini dilakukan untuk memastikan apakah penelitian yang dilakukan telah memenuhi kriteria dari tujuan dan manfaat penelitian.

3. HASIL DAN PEMBAHASAN

Pada Bab ini dibahas mengenai proses pengklasifikasian penyakit daun tomat mengunakan metode *Convolutional Neural Network* dengan tiga arsitektur *Transfer Learning* yaitu: VGG16, ResNet50, dan juga Inception-V3. Dari ketiga arsitektur tersebut kemudian dilakukan uji akurasi dengan memberikan perlakuan yang sama antara satu arsitektur dengan arsitektur lainnya.

3.1 Pengumpulan Data

Dataset yang digunakan berupa dataset yang berasal dari *website* penyedia jasa data publik yaitu kaggle. *Dataset* ini berupa citra gambar penyakit daun tomat dan juga daun tomat sehat. Dataset yang digunakan dalam penelitian ini berasal dari website kaggle dengan judul *Tomato Leaf Disesase Detection* yang berasal dari author bernama KAUSTUBH B. Dataset yang ditemukan berukuran 183.164 KB yang berisi 9 jenis penyakit citra daun tomat dan 1 citra daun tomat yang sehat. Masing masing kelas pada dataset memiliki data training 1000 citra gambar dan 100 citra gambar validation. *Dataset* tersebut terbagi menjadi beberapa kelas diantaranya adalah : *Tomato Mosaic Virus, Target Spot, Bacterial Spot, Tomato Yellow Leaf Curl Virus, Late Blight, Leaf Mold, Early Blight, Septoria Leaf Spot*, dan *Spider mites Two-spotted spider mite*. Table 1 berikut menunjukan persebaran kelas pada *Dataset* yang digunakan mulai dari kelas dengan nama setiap penyakit serta pembagian dataset mentah yang akan diolah:

Table 1 *Dataset* awal kaggel

Kelas	Data Latih	Data Uji
Tomato Mosaic Virus	1000	100
Target Spot	1000	100
Bacterial Spot	1000	100
Tomato Yellow Leaf Curl Virus	1000	100
Late Blight	1000	100
Leaf Mold	1000	100
Early Blight	1000	100
Septoria Leaf Spot	1000	100
Spider mites Two spotted spider mite	1000	100
Healty Tomato Leaf	1000	100
Total	10000	1000

3.2 Preprocessing Data

Setelah menemukan dataset yang cocok untuk penelitian ini, selanjutnya dilakukan persiapan pada dataset sebelum dilakukan pelatihan model. Proses tersebut berupa pemilahan dataset mentah menjadi dataset yang sesuai dengan matode dan penelitian ini. Pemilahan tersebut berupa pengurangan beberapa citra gambar dari masing masing kelas dan juga penghilangan pada salah satu kelas.

a. Pemilahan Dataset

Dari total 11000 citra gambar yang dibagi menjadi 10 kelas kemudian diambil sebanyak 1000 gambar saja pada masing masing kelasnya. Hal ini dilakukan untuk mempermudah dalam melakukan split data dan juga untuk mendapatkan tingkat performa pemodelan yang baik dengan hanyak membatasi 1000 gambar saja. Alasan yang mendasar adalah pada rujukan challange yang dilakukan oleh ImageNet yaitu 1000 data untuk masing masing kategori kelas. Proses pemilihan ini dilakukan secara acak dengan mengurangi pada folder trainning sebanyak 100 citra gambar yang sesuai dengan folder validation sebanyak 100 citra gambar sehingga didapatkan citra gambar sebanyak 1000 pada setiap kelasnya. Penghilangan salah satu kelas juga dilakukan dikarenakan kurangnya literasi terkait penyakit daun tomat yang terdapat pada dataset tersebut yaitu dataset pada penyakit *Spider mites Two-spotted spider mite*. Pemilahan tersebut terinci pada Tabel 2 yang berisi kelas setiap dataset dengan pengurangan salah satu kelas dan pembagian data latih dan data uji yang telah dikelolah:

Tabel 2 Pemilahan dataset

Kelas	Data Latih	Data Uji
Tomato Mosaic Virus	900	100
Target Spot	900	100
Bacterial Spot	900	100
Tomato Yellow Leaf Curl Virus	900	100
Late Blight	900	100
Leaf Mold	900	100
Early Blight	900	100
Septoria Leaf Spot	900	100
Healty Tomato Leaf	900	100
Total	8100	900

b. Augmentasi Data

Dataset yang telah dilakukan pemilahan secara manual, selanjutnya dilakukan proses augmentasi data. Augmentasi data yang diterapkan 7 terdapat jenis pemrosesan diantranya adalah: rotation_range, width_shift_range, height_shift_range, shear_range, zoom_range, horizontal_flip, vertical_flip. Hal ini dilakukan untuk mengurangi overfitting dengan upaya yang seminimum mungkin[15]. Gambar 2 menjelaskan tentang penggunaan augmentasi data dengan berbagai jenis pemrosesan augmentasinya:

```
[ ]  1 train_datagen = ImageDataGenerator(
     2     rotation_range=40,
     3     width_shift_range=0.2,
     4     height_shift_range=0.2,
     5     shear_range=0.2,
     6     zoom_range=0.2,
     7     horizontal_flip=True,
     8     vertical_flip=True,
     9     rescale = 1./255.,
    10     fill_mode='nearest')
```

Gambar 2. Code Augmentasi data

c. Resize Citra Gambar

Tahapan ini bertujuan untuk menyesuaikan ukuran piksel *input*-an data citra gambar dengan metode CNN yang akan digunakan. Data yang digunakan berasal dari website penyedia data publik kaggle. Setiap citra digital pada dataset memiliki ukuran yang sama yaitu 256 x 256. Selanjutnya proses yang dilakukan dalam tahapan *Preprocessing* yaitu melakukan resize menjadi ukuran 224 x 224. Pada Gambar 3 melakukan proses resize dengan mengambil input data sebesar 224 x 224, batch size sebesar 128 dan mode class berupa categorical:

```
[ ]  1 train_generator = train_datagen.flow_from_directory(train_dir,
     2                                                     target_size = (224, 224),
     3                                                     batch_size = 128,
     4                                                     class_mode = 'categorical')
```

Gambar 3. Code Resize Citra Gambar

d. Pembagian Dataset

Setelah dilakukannya tahapan Preprocessing dan juga resize dataset, selanjutnya dilakukannya tahapan split dataset menjadi 2 folder yaitu folder trainning dan folder validation secara mandiri. Slpit data ini dilakukan oleh author penyedia dataset pada kaggle dengan jumlah file 1000 citra gambar untuk trainning dan 100 citra gambar untuk validation pada masing masing kelas dataset. Dataset yang telah dilakukan split data selanjutnya diambil dan dimasukkan menjadi direktori yang bersumber dari Google Drive. Gambar 4 ini berfungsi sebagai

pengambilan data yang sebelumnya telah disimpan di google drive yang kemudian dimasukkan kedalam masing masing direktori:

```
[ ]  1 from google.colab import drive
     2 drive.mount('/content/gdrive')

     Mounted at /content/gdrive

[ ]  1 train_dir = '/content/gdrive/My Drive/SkripsiKu/Dataset/tomato/train'
     2 valid_dir = '/content/gdrive/My Drive/SkripsiKu/Dataset/tomato/val'
```

Gambar 4. *Code* Pembagian Dataset

3.3 Pemodelan *Transfer Learning*

Setelah pada tahapan sebelumnya telah dilakukan pre-processing, pada tahapan selanjutnya adalah melakukan pemodelan sesuai dengan arsitektur *Convolutional Neural Network* yang diterapkan. Pada pemodelan ini dilakukan 3 arsitektur yaitu VGG16, ResNet50, dan Inception-V3. Berikut beberapa perancangan model CNN terebut:

a. Perancangan Model VGG16

Pada perancangan model ini digunakan pemodelan VGG16 yang diambil dari API Keras dengan wight dari ImageNet dan CNN. Dari Gambar 5 dilakukannya pemodelan VGG16 untuk mendapatkan model CNN yang diinginkan:

```
[ ]  1 vgg = VGG16(input_shape=IMAGE_SIZE + [3], weights ='imagenet' , include_top=False)
```

Gambar 5. Code Pemodelan VGG16

Dari pemodelan dasar menggunakan *Transfer Learning* VGG16 tadi, selanjutnya perancangan model dilakukan penambahan layer berupa Average Pooling, Dense, dan juga DropOut. Pada layer yang ditambahkan terdapat dense layer / fully connected layer sebanyak 2 layer dengan jumlah neuron sebanyak 4096 pada keduanya. Selain itu juga ditambahkan penggunaan metode dropout yang diberikan pada setia layer dense dengan probabilitas 0.5. VGG 16 merupakan pemodelan dengan jumlah parameter paling sedikit jika dibandingkan dengan 2 metode lainnya yaitu sebesar 33,634,12, perancangan permodelan tersebut dituangkan pada codingan pada Gambar 6 berikut:

```
[ ]  1 global_average_layer = tf.keras.layers.GlobalAveragePooling2D()
     2 prediction = Dense(len(folders), activation='softmax')
     3
     4 model = tf.keras.Sequential([
     5     vgg,
     6     global_average_layer,
     7     tf.keras.layers.Dense(4096, activation='relu'),
     8     tf.keras.layers.Dropout(0.5),
     9     tf.keras.layers.Dense(4096, activation='relu'),
    10     tf.keras.layers.Dropout(0.5),
    11     prediction
    12 ])
    13
    14 model.summary()
```

Gambar 6. Code Pemodelan VGG16 Lanjutan

b. Perancangan Model ResNet50

Pada perancangan model ResNet50 ini juga menggunakan pemodelan yang diambil dari API Keras berupa ResNet50 dengan wight dari ImageNet dan CNN. Dari pemodelan pada Gambar 7 tersebut didapatkan model CNN dengan parameter ResNet50 sebagai berikut:

```
[ ]  1 resnet = ResNet50(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
```

Gambar 7. Code Pemodelan ResNet50

Dari pemodelan dasar menggunakan *Transfer Learning* ResNet50 tersebut, selanjutnya perancangan model ini juga diberlakukan penambahan layer berupa Average Pooling. 2 layer berupa dense layer / fully connected layer dengan jumlah pada masing masing neuron sebanyak 4096. Metode dropout juga diberikan pada setia layer dense dengan probabilitas 0.5, perancangan permodelan tersebut dituangkan pada Gambar 8 codingan dan perancangan berikut:

```
[ ]  1 global_average_layer = tf.keras.layers.GlobalAveragePooling2D()
     2 prediction = Dense(len(folders), activation='softmax')
     3
     4 model = tf.keras.Sequential([
     5     resnet,
     6     global_average_layer,
     7     tf.keras.layers.Dense(4096, activation='relu'),
     8     tf.keras.layers.Dropout(0.5),
     9     tf.keras.layers.Dense(4096, activation='relu'),
    10     tf.keras.layers.Dropout(0.5),
    11     prediction
    12 ])
    13
    14 model.summary()
```

Gambar 8. Code ResNet50 Lanjutan

c. Perancangan Model Inception-V3

Sama dengan dua perancangan model sebelumnya, pada perancangan Inception-V3 ini juga menggunakan pemodelan yang diambil dari API Keras Inception-V3 dengan wight dari ImageNet dan CNN. Sama seperti pemodelan lainnya pada Inception-V3 juga dilakukan pemodelan yang dituangkan pada Gambar 9:

```
[ ]  1 inception = InceptionV3(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
```

Gambar 9. Code Pemodelan Inception-V3

Dari pemodelan *Transfer Learning* Inception-V3 tersebut, juga ditambahkan beberapa layer yaitu Average Pooling dari tensorflow keras. 2 layer berupa *dense layer / fully connected layer* dengan jumlah pada masing masing neuron sebanyak 4096. Metode dropout juga diberikan pada setia layer dense dengan probabilitas 0.5, Perancangan permodelan tersebut dituangkan pada Gambar 10 berikut:

```
[ ]  1 global_average_layer = tf.keras.layers.GlobalAveragePooling2D()
     2 prediction = Dense(len(folders), activation='softmax')
     3
     4 model = tf.keras.Sequential([
     5     inception,
     6     global_average_layer,
     7     tf.keras.layers.Dense(4096, activation='relu'),
     8     tf.keras.layers.Dropout(0.5),
     9     tf.keras.layers.Dense(4096, activation='relu'),
    10     tf.keras.layers.Dropout(0.5),
    11     prediction
    12 ])
    13
    14 model.summary()
```

Gambar 10. Code Inception-V3 Lanjutan

3.4 Pelatihan *Transfer Learning*

Setelah dilakukannya perancangan model CNN sesuai dengan *Transfer Learning* yang digunakan. Selanjutnya dilakukan pelatihan model yang disusun menggunakan loss function berupa Categorical Crossentropy, penggunaan metode tersebut dilakukan karena jumlah kelas / kategori yang terdapat pada dataset yang digunakan lebih banyak atau sama dengan 3 kelas. Pada pelatihan ini juga diberikan Optimizer berupa Adam Optimizer yang menggunakan Accuracy untuk melakukan metrics pengukuran performanya. Penggunan codingan pada ketiga pelatihan tersebut dicantumkan sebagai berikut:

a. Pelatihan Model VGG16

Gambar 11 menunjukan mengenai pelatihan model VGG16 mulai dari trainable, penggunaan optimizer, penetapan matric, model fit, jumlah epochs dan lain sebagainya.

```
[ ]  1 vgg.trainable = False
     2 model.compile(loss='categorical_crossentropy',
     3               optimizer='adam',
     4               metrics=['accuracy'])
     5
     6 vgg_model = model.fit(
     7   train_generator,
     8   validation_data=valid_generator,
     9   epochs=20,
    10   steps_per_epoch=len(train_generator),
    11   validation_steps=len(valid_generator),
    12   verbose = 1
    13 )
```

Gambar 11 Code Pelatihan VGG16

b. Pelatihan Model ResNet50

Gambar 12 menunjukan mengenai pelatihan model ResNet50 mulai dari trainable, penggunaan optimizer, penetapan matric, model fit, jumlah epochs dan lain sebagainya.

```
[ ]  1 resnet.trainable = False
     2 model.compile(loss='categorical_crossentropy',
     3               optimizer='adam',
     4               metrics=['accuracy'])
     5
     6 resnet_model = model.fit(
     7   train_generator,
     8   validation_data=valid_generator,
     9   epochs=20,
    10   steps_per_epoch=len(train_generator),
    11   validation_steps=len(valid_generator),
    12   verbose = 1
    13 )
```

Gambar 12 Code Pelatihan ResNet50

c. Pelatihan Model Inception-V3

Sama seperti dua arsitektur sebelumnya Gambar 13 menunjukan mengenai pelatihan model Inception-V3 mulai dari trainable, penggunaan optimizer, penetapan matric, model fit, jumlah epochs dan lain sebagainya.

```
[ ]  1 inception.trainable = False
     2 model.compile(loss='categorical_crossentropy',
     3               optimizer='adam',
     4               metrics=['accuracy'])
     5
     6 inceptionV3_model = model.fit(
     7   train_generator,
     8   validation_data=valid_generator,
     9   epochs=20,
    10   steps_per_epoch=len(train_generator),
    11   validation_steps=len(valid_generator),
    12   verbose = 1
    13 )
```

Gambar 13 Code Pelatihan Inception-V3

Dari ketiga pelatihan tersebut diterapkan metode yang sama yaitu penggunaan Epoch sebanyak 20, steps_per_Epoch sebanyak panjang train_generator yaitu 8100, dan validation_steps sebanyak panjang dari valid_generator yaitu 900. Setelah dilakukan pelatihan tersebut, selanjutnya diberikan pelatihan menggunakan metode Fine Tuning yang juga merupakan konsep dari *Transfer Learning* . Metode ini merupakan pelatihan model dalam *Transfer Learning* yang dilatih ulang menggunakan layer yang tidak dibekuka. Metode ini digunakan karena dinilai dapat meningkatkan performa yang sudah terlatih menjadi lebih baik daripada penyusunan model secara manual. Metode ini perlu dilalukan learning rate yang kecil agar tidak merubah / merusak paramater yang telah dilatih oleh model. Berikut merupakan pengcodingan *Transfer Learning* pada ketiga arsitektur yang digunakan:

d. Pelatihan Model VGG16

Gambar 14 menunjukan pelatihan lanjutan dengan penambahan pembekuan pada setiap layer, penggunaan jumlah epochs keseluruhan dan juga penambahan metode *learning rate*.

```
[ ]  1 vgg.trainable = True
     2 model.compile(loss=tf.keras.losses.CategoricalCrossentropy(),
     3               optimizer=tf.keras.optimizers.Adam(1e-5),
     4               metrics=['accuracy'])
     5
     6 vgg_model_fine = model.fit(
     7   train_generator,
     8   validation_data=valid_generator,
     9   epochs=40,
    10   steps_per_epoch=len(train_generator),
    11   validation_steps=len(valid_generator),
    12   verbose = 1,
    13   initial_epoch= vgg_model.epoch[-1]
    14 )
```

Gambar 14 Code Pelatihan VGG16 TL

e. Pelatihan Model ResNet50

Gambar 15 menunjukan pelatihan lanjutan dengan penambahan pembekuan pada setiap layer, penggunaan jumlah epochs keseluruhan dan juga penambahan metode *learning rate*.

```
[ ]  resnet.trainable = True
     model.compile(loss=tf.keras.losses.CategoricalCrossentropy(),
                   optimizer=tf.keras.optimizers.Adam(1e-5),
                   metrics=['accuracy'])

     resnet_model_fine = model.fit(
       train_generator,
       validation_data=valid_generator,
       epochs=40,
       steps_per_epoch=len(train_generator),
       validation_steps=len(valid_generator),
       verbose = 1,
       initial_epoch= resnet_model.epoch[-1]
     )
```

Gambar 15 Code Pelatihan ResNet50 TL

f. Pelatihan Model Inception-V3

Sama seperti sebelum sebelumnya Gambar 16 menunjukan pelatihan lanjutan dengan penambahan pembekuan pada setiap layer, penggunaan jumlah epochs keseluruhan dan juga penambahan metode *learning rate*.

```
[ ]  inception.trainable = True
     model.compile(loss=tf.keras.losses.CategoricalCrossentropy(),
                   optimizer=tf.keras.optimizers.Adam(1e-5),
                   metrics=['accuracy'])

     inceptionV3_model_fine = model.fit(
       train_generator,
       validation_data=valid_generator,
       epochs=40,
       steps_per_epoch=len(train_generator),
       validation_steps=len(valid_generator),
       verbose = 1,
       initial_epoch= inceptionV3_model.epoch[-1]
     )
```

Gambar 16 Code Pelatihan Inception-V3 TL

3.5 Analisis Hasil

Pada pembahasan subbab ini merupakan analisis hasil dari penelitian yang telah dilakukan. Analisa ini dilakukan berdasarkan hasil penelitan dari pengklasifikasian penyakit daun tomat menggunakan metode CNN. Metode CNN yang dibiberlakukan dalam penelitian ini menggunakan tiga algoritma yang berbeda, diantaranya adalah VGG16, ResNet50 dan juga Inception-V3. Setiap arsitektur mendapatkan perlakuan yang sama antara satu arsitektur dengan arsitektur lainnya. Setelah dilakukan penerapan perancangan model CNN sesuai dengan *Transfer Learning* yang digunakan didapatkan hasil yang berbeda-beda. Ketiga arsitektur ini dinilai cukup mendapatkan hasil akurasi yang cukup tinggi pada masing masing arsitekturnya. VGG16 terlihat memiliki keuntungan karena melibatkan representasi fitur dan kegunaan arsitektur yang sederhana dan konsisten yang dinilai cocok dalam penelitian ini. Hasil uji akurasi dari ketiga arsitektur CNN terbseut ditunjukkan pada Tabel 3 berikut:

Tabel 3 Analisis Hasil Akurasi

Arsitektur	Resolusi	Parameter	Akurasi
VGG16	224 x 224	33,634,121	97,78%
ResNet50	224 x 224	48,798,601	94,67%
Inception-V3	224 x 224	47,013,673	95,44%

Setiap arsitektur tersebut menunjukkan hasil pelatihan yang selalu meningkat dari awal hingga akhir baik secara drastis maupun secara bertahap.

4. KESIMPULAN

Berdasarkan hasil yang didapatkan dari pengklasifikasian penyakit daun tomat menggunakan metode *Convolutional Neural Network* (CNN) pada beberapa arsitketur *Transfer Learning* VGG16, ResNet-50, dan Inception V-3 dapat diambil kesimpulan dari beberapa percobaan yang telah dilakukan bahwa arsitektur VGG16 merupakan arsitektur yang terbaik dibandingkan dari 2 arsitektur lainnya dengan nilai akurasi tertinggi adalah 0.9726 dan nilai validasi akurasi tertinggi sebesar 0.9778. Arsitektur ini mendapatkan penambahan skenario yaitu penambahan 2 layer Dense (4096) dan 2 layer dropout dengan nilai 0.5.Penggunaan fine tuning dalam pelatihan model dinilai lebih efektif dan efisien dibandingkan dengan pelatihan model secara manual.

DAFTAR PUSTAKA

[1] N. Heriani, W. Abbas Zakaria, and S. Achdiansyah, "Analisis Keuntungan dan Risiko Usahatani Tomat di Kecamatan Sumberejo Kabupaten Tanggamus," *Jiia*, vol. 1, no. 2, pp. 169–173, 2013.

[2] R. Wulandari, "Respon pertumbuhan tanaman t o m a t," *RESPON PERTUMBUHAN Tanam. TOMAT {Lycopersicum esculentum L.) DENGAN PENAMBAHAN PUPUK ORGANIK BAYAM {Amaranthus sp L.) SERTA PENGAJARANNYA DI MADRASAH ALIYAH NEGERI 1 PALEMBANG*, no. November, 2015.

[3] L. Sahrani, "Klasifikasi Penyakit Daun Tomat Berdasarkan Ekstraksi Tekstur Daun Menggunakan Gabor Filter Dan Algoritma Support Vector Machine," 2021, [Online]. Available: http://repository.uinsu.ac.id/13431/

[4] N. Z. Munantri, H. Sofyan, and M. Yanu, "Aplikasi Pengolahan Citra Digital Untuk Mendeteksi Umur Pohon," *Telematika*, vol. 16, no. 2, pp. 97–104, 2019.

[5] M. Agarwal, S. K. Gupta, and K. K. Biswas, "Development of Efficient CNN model for Tomato crop disease identification," *Sustain. Comput. Informatics Syst.*, vol. 28, p. 100407, Dec. 2020, doi: 10.1016/j.suscom.2020.100407.

[6] Stephen, Raymond, and H. Santoso, "APLIKASI CONVOLUTION NEURAL NETWORK UNTUK MENDETEKSI JENIS-JENIS SAMPAH," *Explor. ± J. Sist. Inf. dan Telemat.*, 2013.

[7] C. Wang *et al.*, "Pulmonary image classification based on inception-v3 *Transfer Learning* model," *IEEE Access*, vol. 7, pp. 146533–146541, 2019, doi: 10.1109/ACCESS.2019.2946000.

[8] N. Khasanah, "Komparasi Arsitektur RESNET50 dan VGG16 untuk Klasifikasi Citra Tanda Tangan," *J. Sist. Inf.*, vol. 14, no. 1, pp. 2611–2621, 2022.

[9] M. Zufar and S. Setiyono, Budi Si, "Convolutional Neural Networks for Real-Time Face Recognition," pp. 1–137, 2016.

[10] A. Patil and M. Rane, "Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition," *Smart Innov. Syst. Technol.*, vol. 195, pp. 21–30, 2021, doi: 10.1007/978-981-15-7078-0_3.

[11] B. K. Umri, E. Utami, and M. P. Kurniawan, "Tinjauan Literatur Sistematik tentang Deteksi Covid-19 menggunakan Convolutional Neural Networks," *Creat. Inf. Technol. J.*, vol. 8, no. 1, p. 9, 2021, doi: 10.24076/citec.2021v8i1.261.

[12] M. A. Pangestu and H. Bunyamin, "Analisis Performa dan Pengembangan Sistem Deteksi Ras Anjing pada Gambar dengan Menggunakan Pre-Trained CNN Model," *J. Tek. Inform. dan Sist. Inf.*, vol. 4, pp. 337–344, 2018.

[13] D. Theckedath and R. R. Sedamkar, "Detecting Affect States Using VGG16, ResNet50 and SE-ResNet50 Networks," *SN Computer Science*, vol. 1, no. 2. 2020. doi: 10.1007/s42979-020-0114-9.

[14] M. R. Kapa, "Klasifikasi Citra Penyakit Leukemia Menggunakan *Convolutional Neural Network* Dengan Arsitektur Inception-V3," p. 129, 2022.

[15] S. GC *et al.*, "Using Deep Learning Neural Network in Artificial Intelligence Technology to Classify Beef Cuts," *Front. Sensors*, vol. 2, no. June, pp. 1–12, 2021, doi: 10.3389/fsens.2021.654357.