# **TUGAS**

# **ARTIFICIAL INTELLIGENCE**

**“Plant Disease Detector (Pladiator)”**

****

**Dususun oleh:**

Mokhamad Wijaya 17090092

Reza Aditya Sam Marhaban 17090147

Kelas 5D

# 

**POLITEKNIK HARAPAN BERSAMA**

**TEGAL**

**2019**

## **BAB I**

**PENDAHULUAN**

## **Latar Belakang**

Indonesia dilalui oleh sirkum mediteranian dan sirkum pasifik sehingga menjadikannya sebagai negara dengan banyaknya gunung berapi yang aktif. Hal ini tentunya berdampak pada ekosistem di Indonesia mengingat dengan adanya gunung berapi yang meletus akan memberikan kehidupan yang lebih baik, yaitu tanah yang subur sehingga dapat ditanami berbagai jenis tumbuhan. Tidak hanya itu, Indonesia juga berlokasi di daerah khatulistiwa sehingga musimnya hanya terdiri dari kemarau dan hujan. Maka tidak aneh bila Indonesia memiliki banyak kawasan hutan hujan dan ekosistem lainnya yang didasari oleh beragam varian spesies tumbuhan.

Dengan memanfaatkan keuntungan geografis, para petani di Indonesia menanam berbagai jenis tumbuhan yang beraneka ragam dan cukup strategis dalam perdagangan, mengingat kebutuhan manusia akan tumbuhan sangat tinggi dan beragam. Terdapat berbagai jenis tumbuhan strategis yang beberapa diantaranya, yaitu tomat, apel, jagung, dan lain-lain. Perihal tersebut menjadikan Indonesia sebagai negara agraris.

Dalam sebuah ekosistem, tentunya selalu ada faktor-faktor yang menjaga keseimbangannya. Faktor-faktor tersebut ada yang berpengaruh baik dan buruk jika dilihat dari sudut pandang manusia, mengingat manusia cenderung membuat atau mengubah kondisi dan situasi ekosistem sesuai pengetahuan manusia untuk membudidayakan tumbuhan strategis dalam perdagangan. Beberapa faktor buruk dalam sebuah ekosistem yang berakibat terhadap tumbuhan diantaranya, yaitu hama dan polusi.

Seringkali para pembudidaya tumbuhan atau petani dibuat resah karena kedua faktor tersebut. Banyak tumbuhan dagang strategis yang terkena penyakit sehingga terjadi gagal panen atau panen sebagian dari lahan yang dikerjakan. Hal ini menyebabkan kerugian yang cukup signifikan pada petani.

Tetapi seiring dengan kemajuan teknologi, sudah banyak solusi yang ditemukan untuk menangani kedua faktor tersebut. Hanya saja tidak semua petani menerapkan solusi yang ada. Hal ini dikarenkan banyak faktor, yang beberapa diantaranya adalah ketidaktahuan, keterbatasan sumber daya, keterbatasan keuangan, dan lain-lain. Oleh karena itu Pladiator hadir untuk menyelesaikan salah satu dari faktor tersebut.

Pladiator adalah aplikasi berbasis gawai yang dapat mendeteksi penyakit pada tumbuhan berdasarkan pada kondisi daun. Dengan mengambil gambar daun dari tumbuhan, maka aplikasi ini akan mendeteksi penyakit apa yang sedang diderita tumbuhan tersebut dan pembudidaya dapat mencari beberapa solusi yang dapat diterapkan untuk menyembuhkannya. Dengan adanya Pladiator, diharapkan dapat membantu para petani atau pembudidaya tumbuhan untuk menambah wawasan dalam menyelesaikan masalah yang berkaitan dengan tumbuhan.

Untuk mewujudkan Pladiator, maka dibutuhkan kecerdasan buatan yang menerapkan algoritma sesuai dengan permasalahan. Algoritma yang cocok adalah algoritma yang bisa mengolah gambar, algoritma pemodelan abstraksi yang didasarkan pada jaringan syaraf tiruan konvolusional dan disusun berlapis-lapis menjadi sebuah model. Model yang digunakan bernama Inception-v3 yang ditambahi dengan beberapa lapis algoritma untuk menghasilkan sekumpulan nilai untuk klasifikasi. Model ini terbilang ringan dan cepat saat melakukan proses inferensi di gawai sehingga sangat cocok untuk Pladiator yang penggunaannya akan lebih banyak berada di lapangan daripada sebuah lab.

## **Tujuan**

Tujuan dibuatnya Pladiator adalah sebagai berikut.

1. Membantu mengenali penyakit tumbuhan berdasarkan kondisi daun secara visual.
2. Membantu pembudidaya tumbuhan mengenali penyakit tumbuhan berdasarkan kondisi daun secara visual.
3. Menambah wawasan pembudidaya tumbuhan dan orang-orang yang tertarik dengan tumbuhan.
4. Membantu pembudidaya tumbuhan dalam membudidayakan tumbuhan agar tidak terjadi gagal panen.

**BAB II**

**METODE YANG DIGUNAKAN**

## **Model**

Model dalam hal ini diartikan sebagai algoritma jaringan syaraf tiruan yang disusun berlapis-lapis dengan struktur tertentu untuk melakukan suatu tugas. Model yang digunakan bernama Inception-v3, model yang sudah dilatih sebelumnya oleh ilmuan pembelajaran dalam Google untuk tugas mengklasifikasikan gambar. Model ini sudah dilatih dengan *dataset* dari ImageNet yang berjumlah 1.331.167 gambar, dibagi menjadi 1.281.167 gambar latihan dan 50.000 gambar validasi. Kelas dalam dataset ini berjumlah 1000. Model ini dilatih ulang dengan *PlantVillage-Dataset*, yang berisikan 38 kelas (jenis) daun tumbuhan baik yang sehat maupun yang berpenyakit.





38

Relu

Flatten

## **Contoh Kasus**

Dalam kasus ini, digunakan gambar daun jagung yang sehat di bawah ini.



Kemudian ukurannya diubah menjadi 299x299 *pixel* dan diubah menjadi sekumpulan *array* dengan dimensi 1x299x299x3. Berikut adalah tampilan ringkas *array*-nya (Selengkapnya ada pada lampiran terpisah).

[[[199, 192, 199], [208, 201, 208], [219, 212, 219], [213, 206, 213],

[201, 194, 201], [199, 192, 199], [195, 188, 195], [210, 203, 210],

[212, 205, 212], [218, 211, 218], [214, 207, 214], [207, 200, 207],

…………………………………..

[170, 162, 173], [172, 164, 175], [163, 155, 166], [155, 147, 158],

[176, 168, 179], [179, 171, 182], [165, 157, 168], [159, 151, 162],

[160, 152, 163], [164, 156, 167], [167, 159, 170], [171, 163, 174]]]

Lalu dilakukan normalisasi, yaitu pengubahan tiap nilai dalam *array* supaya dapat diproses oleh model dengan cepat. Normalisasi yang dilakukan adalah membagi tiap nilai dalam *array* dengan 255 sehingga didapatkan nilai dengan rentang dari 0 sampai 1. Berikut adalah tampilan ringkas *array* setelah normalisasi.

[[[0.78039216, 0.75294118, 0.78039216],

[0.81568627, 0.78823529, 0.81568627],

[0.85882353, 0.83137255, 0.85882353],

……………………...

[0.64313725, 0.61176471, 0.65490196],

[0.65490196, 0.62352941, 0.66666667],

[0.67058824, 0.63921569, 0.68235294]]]

Selanjutnya adalah memasukkan *array* tersebut ke dalam model agar dilakukan inferensi hingga menghasilkan *array* dengan dimensi 1x38. Berikut adalah *array* luaran dari model.

[[4.6510851e-07, 2.2118666e-08, 7.2860371e-09, 2.5625669e-03,

2.5936744e-12, 4.0713604e-07, 6.0860458e-07, 1.8846893e-11,

2.4848626e-12, 3.5132546e-11, 9.9740797e-01, 1.5751841e-09,

6.1362911e-12, 2.5331942e-13, 2.0301982e-09, 4.9445212e-08,

1.1052828e-09, 1.2400291e-10, 1.1722995e-10, 2.8861045e-09,

2.2245803e-12, 4.9718119e-07, 4.3223460e-07, 5.1047064e-09,

2.2830574e-05, 5.6818165e-11, 4.2907910e-11, 1.7561041e-07,

7.1636391e-10, 6.8038450e-09, 2.9787718e-06, 2.9060745e-08,

2.4517002e-09, 4.8934942e-07, 3.7937795e-08, 7.4077233e-09,

2.7334131e-11, 4.8361602e-07]]

Kemudian menentukan indeks pada *array* yang memiliki nilai paling tinggi. Dalam hal ini, nilai tertinggi adalah 9.9740797e-01 yang berarti indeksnya adalah 10. Indeks yang sudah ditemukan kemudian dijadikan acuan pada array yang berisikan jenis-jenis tumbuhan baik yang berpenyakit maupun tidak. Berikut adalah array jenis-jenis tumbuhan.

[apple apple scab, apple black rot, apple cedar apple rust, apple healthy, blueberry healthy, cherry including sour powdery mildew, cherry including sour healthy, corn maize cercospora leaf spot gray leaf spot, corn maize common rust, corn maize northern leaf blight, corn maize healthy, grape black rot,  
grape esca black measles, grape leaf blight isariopsis leaf spot.   
grape healthy, orange haunglongbing citrus greening, peach bacterial spot,  
peach healthy, pepper bell bacterial spot, pepper bell healthy ,potato early blight,  
potato late blight, potato healthy, raspberry healthy, soybean healthy,  
squash powdery mildew, strawberry leaf scorch, strawberry healthy,  
tomato bacterial spot, tomato early blight, tomato late blight, tomato leaf mold,  
tomato septoria leaf spot, tomato spider mites two spotted spider mite,  
tomato target spot, tomato tomato yellow leaf curl virus,  
tomato tomato mosaic virus, tomato healthy]

Dalam array ini, indeks 10 berisikan Corn maize healty yang berarti jagung yang sehat. Maka didapatkan hasil berupa jagung yang sehat dengan *confidence score* 9.9740797e-01.

**BAB III**

**IMPLEMENTASI ALGORITMA**

## **Tahap Pengembangan Sistem**

Untuk mengembangkan Pladiator, dibutuhkan beberapa tahap pengembangan, diantaranya:

1. Pembuatan dan Pelatihan Model

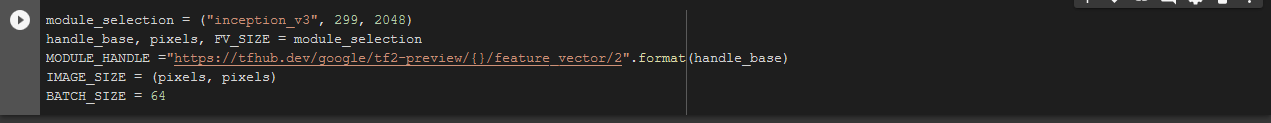
Pada tahap ini, model dibuat dengan menggunakan pustaka pembelajaran dalam Tensorflow dan Tensorflow-Hub. Tensorflow adalah pustaka pembelajaran mesin yang di dalamnya terdapat pustaka Keras untuk pembelajaran mendalam. Tensorflow-Hub adalah pustaka untuk mengunduh model-model yang disediakan oleh Google, yang salah satunya adalah Inception-v3.

Dikarenakan keterbatasan sumber daya lokal, maka pelatihan model dilakukan dengan menggunakan *Google Colabolatory* atau bisa disebut colab, sebuah layanan dari Google yang menyediakan akses *cloud computer* secara gratis berbasis sistem operasi linux dan juga *interpreter* berbahasa pemrograman python untuk melakukan berbagai keperluan, yang khususnya adalah untuk pembelajatan mesin dan pembelajaran mendalam. Colab tidak memiliki antarmuka seperti sistem operasi linux biasanya karena layanan ini berjalan melalui peramban. Antarmuka colab seperti tampilan antarmuka *jupyter notebook*.

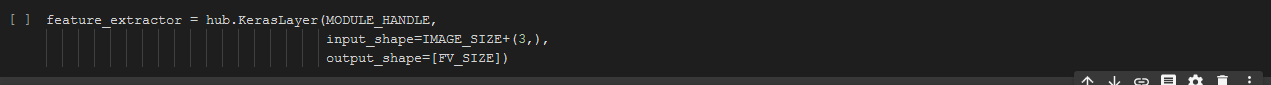
Sebelum memulai pelatihan model, dilakukan pengunduhan *dataset* dan *preprocessing* *dataset*. *Dataset* yang digunakan berasal dari Kaggle, yaitu *New Plant Village Dataset*. Dataset ini terdiri dari 38 kelas yang berisikan 14 jenis tumbuhan dengan kondisinya baik kondisi sehat dan kondisi berpenyakit. Dataset yang berjumlah 10861 ini dibagi menjadi gambar data latih dan gambar data validasi.

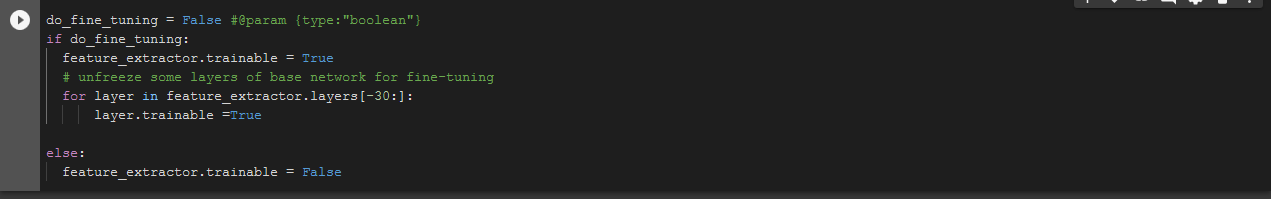
Setelah memisahkan data latih dan data validasi, dilakukan proses normalisasi dengan mengubah semua data ke bentuk *array*. Kemudian tiap nilai dalam *array* dibagi 255.

Setelah selesai mempersiapkan data, selanjutnya adalah mengunduh model Inception-v3 dengan pustaka bawaan Tensorflow-Hub.

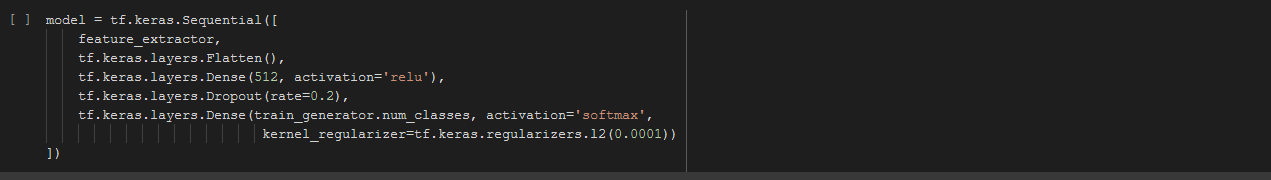


Berikut adalah parameter yang disiapkan sebelum mengunduh model. Mulai dari nama model “inception\_v3”, kemudian ukuran masukan gambar adalah 299x299 pixel, dan dimensi layer terakhir (bisa dianggap sebagai dimensi luaran) adalah 2048. *Batch Size* (sejumlah gambar yang sekaligus dimasukkan ke dalam model) yang ditentukan adalah 64 gambar, karena kita menggunakan perangkat keras dari Google yaitu NVIDIA Tesla K80 dengan memori sebesar 12 GB, sehingga tidak perlu khawatir akan terjadinya kekurangan memori saat terjadinya proses pelatihan model.

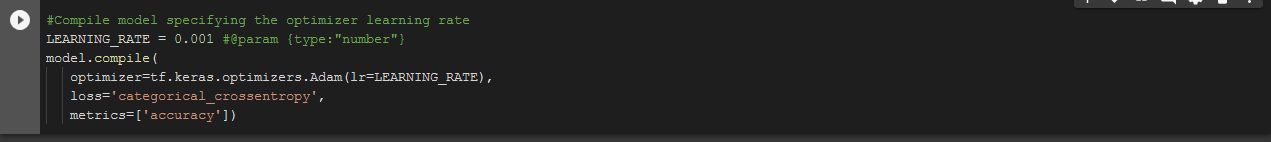
Kemudian dilakukan proses pengunduhan menggunakan sintaks pada gambar di atas. Kata “hub” mewakili pustaka Tensorflow-Hub dan fungsi “KerasLayer” adalah fungsi yang menampung model yang akan diunduh dengan parameter berupa nama model, dimensi masukan dan dimensi luaran. Nama model diisi sesuai dengan variabel MODULE\_HANDLE, dan dimensi masukan diisi variabel IMAGE\_SIZE+(3,) yang mengartikan dimensinya berukuran (299x299x3), dan dimensi luarannya berukuran 2048. Semua itu ditampung oleh variabel feature\_extractor.



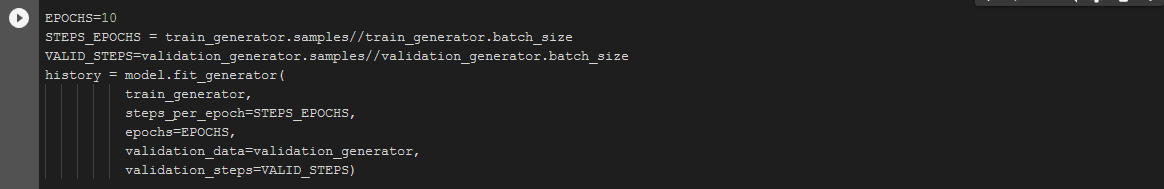
feature\_extractor diberi pengaturan supaya tidak dilakukan *fine-tuning* model karena feature\_extractor ini akan diberi *layer* tambahan lalu dilatih.



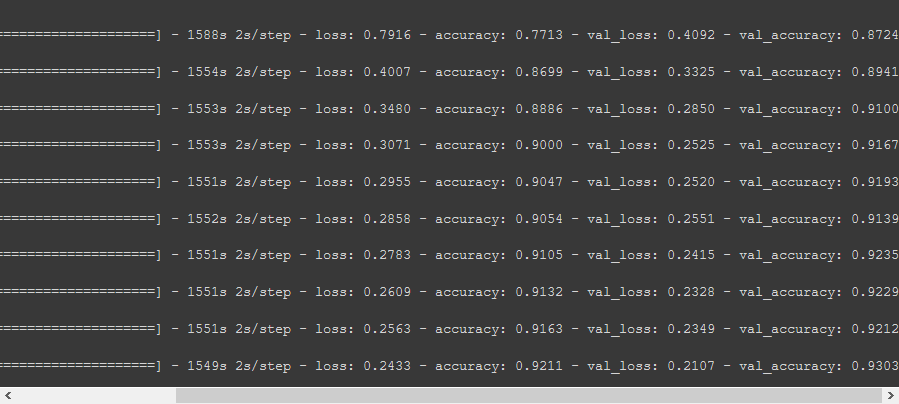
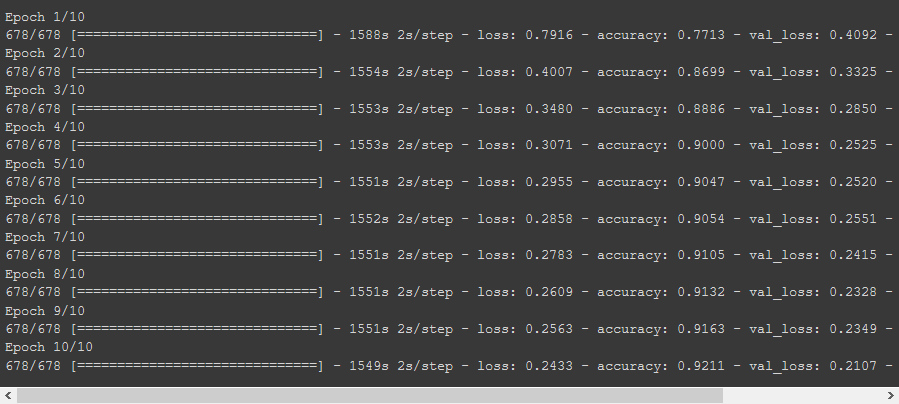
Pada sintaks di atas, adalah penyusunan model Inception-v3 dengan beberapa layer tambahan. Dimulai dari feature\_extractor yang berisikan model Inception-v3 dari Google, kemudian dilanjut dengan layer Flatten yang berfungsi untuk mengubah ukuran dimensi *array* menjadi 1 dimensi *array*. Lalu *layer* Dense, berupa *layer* jaringan syaraf tiruan dengan banyak *perceptron* 512 unit dengan fungsi aktivasinya menggunakan ReLU. Kemudian layer Dropout, berupa layer yang mengeliminasi parameter latih (yang cenderung tidak digunakan untuk latihan parameter dari model) milik layer sebelumnya dengan rate (persentase parameter yang dieliminasi) sebesar 20 %. Dan terakhir adalah layer Dense, dengan jumlah *perceptron* sama dengan jumlah kelas pada dataset, yang fungsi aktivasinya menggunakan *softmax,* dan lost function diregularisasi menggunakan metode L2 dengan *rate* 0,0001.



Pada sintaks di atas, model dikompilasi menggunakan *optimizer* bernama Adam dengan *learning rate* sebesar 0,001, fungsi untuk menghitung eror bernama *Categorical Cross Entropy*, dan *metric* (informasi yang akan ditampilkan tiap *epoch* berakhir) berupa akurasi dari model yang akan dites menggunakan data validasi.



Pada sintaks di atas, dimulai proses pelatihan model dengan konfigurasi *epochs* (berapa kali perulangan model dilatih menggunakan semua data dalam *dataset*) 10, *steps\_per\_epoch* (berapa kali model melakukan inferensi dan *backward propagation* pada data latih tiap satu kali *epoch*) dengan rumus jumlah data latih [43444] modulus *batch size* data latih [64] sama dengan 678 kali, dan validation\_steps (berapa kali model melakukan inferensi dan *backward propagation* pada data validasi tiap satu kali *epoch*) dengan rumus jumlah data validasi [10861] modulus *batch size* data validasi [64] sama dengan 169 kali. Kemudian menunggu proses pelatihan model dengan waktu sekitar 3 jam 30 menit.



Setelah itu, model sudah dapat melakukan inferensi dengan dengan akurasi sebesar 92,11% pada data latih dan 93,02% pada data validasi. Model setelah dilatih masih belum dapat berjalan di gawai, dan karena itu model disimpan dalam format .tflite supaya dapat berjalan di gawai.

1. Pembuatan Aplikasi

Dalam pembuatan aplikasi, digunakan Android Studio dengan bahasa pemrograman Kotlin. Pengembangan aplikasi merujuk pada sumber kode dari *repository* Github milik obeshor (<https://github.com/obeshor/Plant-Diseases-Detector>). Dengan rujukan tersebut, untuk pembutan aplikasi berbasis seluler Pladiator hanya membutuhkan proses *debugging* pada sumber kode yang sudah ada.

Sumber kode yang dirujuk sudah sampai tahap inferensi baik menggunakan foto dari penyimpanan internal atau dari foto hasil tangkapan kamera dalam satu *Activity* utama. Namun pada *repository* terujuk terdapat kesalahan. Kesalahan tersebut adalah ukuran *resize* gambar yang akan dimasukkan ke dalam model untuk dilakukan inferensi sebesar 224. Sedangkan saat pelatihan model, ukuran gambar yang dimasukkan sebesar 299. Oleh karena itu diubah variabel yang menampung nilai tersebut (INPUT\_SIZE pada file Classifier.kt) menjadi 299.

Selain itu, ada satu kesalahan lagi yaitu saat mengubah format gambar dari kelas *Bitmap* ke kelas *ByteBuffer*. Disiapkan sebuah variabel bernama byteBuffer dan diisi menggunakan fungsi *allocateDirect()* dari kelas *ByteBuffer* dengan parameter kapasitasnya yang diisi 4 \* INPUT\_SIZE \* INPUT\_SIZE \* PIXEL\_SIZE (PIXEL\_SIZE adalah jumlah *channel* warna pada gambar yaitu 3 karena gambar memiliki *channel* warna RGB). Isi parameter tersebut salah karena dimensi yang sesungguhnya dari gambar saat akan masuk ke dalam model saat pelatihan adalah None x 299 x 299 x 3 [Jumlah data x panjang x lebar x jumlah *channel* warna] dan dalam kasus inferensi di gawai, jumlah data untuk dimensi gambar tetap harus ada dan berjumlah 1, sehingga koreksi isi dari parameternya adalah 4 \* BATCH\_SIZE \* INPUT\_SIZE \* INPUT\_SIZE \* PIXEL\_SIZE, dengan BATCH\_SIZE bernilai 1.

Setelah mengoreksi 2 kesalahan di atas, model yang sudah berformat .tflite ditaruh di dalam folder *assets*. Lalu dilakukan beberapa modifikasi tampilan seperti menambahkan gambar latar dan memberi warna tombol serta penyesuaian warna teks dan jadilah Pladiator.

**BAB IV**

**EVALUASI**

Untuk mengetahui sejauh mana aplikasi Pladiator dapat digunakan, dilakukan sebuah tes akurasi model dengan melakukan inferensi model dalam gawai pada 5 gambar yang di dapat dari beberapa alamat web dan tidak ada pada *dataset*. Gambar tersebut terdiri dari 5 kelas yang berbeda yang kelas tersebut tidak termasuk dalam kelas yang digunakan untuk pelatihan model. Berikut adalah tabel pengetesannya.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Gambar dan Alamat web | Jenis tumbuhan terdeteksi dan *confidence score* | Jenis tumbuhan sebenarnya | Nilai Prediksi (1 – *confidence score*) |
| 1. | Masakapaya.com | Daun Strawberry | Daun Singkong | 0.8491752 |
| 2. | Dinimon.com | Daun Kedelai | Daun Sirih | 0.9994849 |
| 3. | Belajartani.com | Daun Tomat | Daun Cabai | 0.9996277 |
| 4. | Tiaragatzu.com  C:\Users\Reza Aditya\Desktop\daun pare.jpg | Daun Tomat | Daun Pare | 0.6568232 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Dari hasil nilai eror di atas, dapat disimpulkan bahwa model yang berjalan digawai masih belum dapat diandalkan dalam pembudidayaan tumbuhan yang sebenarnya dan masih membutuhkan data yang lebih banyak lagi dan lebih bervariasi.