IDENTIFIKASI TANAMAN HIAS MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK DENGAN IMPLEMENTASI TRANSFER LEARNING

TUGAS AKHIR

VIDIAN VITO OKTARIYANTO
311710018

TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS MA CHUNG
MALANG
2023

LEMBAR PENGESAHAN

TUGAS AKHIR

IDENTIFIKASI TANAMAN HIAS MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK DENGAN IMPLEMENTASI TRANSFER LEARNING

Oleh:

VIDIAN VITO OKTARIYANTO

NIM. 311710018

dari:

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS dan TEKNOLOGI

UNIVERSITAS MA CHUNG

Telah dinyatakan lulus dalam melaksanakan Tugas Akhir sebagai syarat kelulusan dan berhak mendapatkan gelar Sarjana Komputer (S.Kom.)

Dosen Pembimbing I,

Dr. Kestrilia Rega Prilianti, M.Si.
NIP. 20120035

Dosen Pembimbing II,

Hendry Setiawan, ST., M.Kom.
NIP. 20100006

Dekan Fakultas Sains dan Teknologi,

Dr. Kestrilia Rega Prilianti, M.Si.
NIP. 20120035

PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi Tugas Akhir ini dengan judul IDENTIFIKASI TANAMAN HIAS MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK DENGAN IMPLEMENTASI TRANSFER LEARNING adalah benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diizinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Malang, 8 Juni 2023

Vidian Vito Oktariyanto
NIM. 311710018

IDENTIFIKASI TANAMAN HIAS DENGAN IMPLEMENTASI TRANSFER LEARNING PADA CONVOLUTIONAL NEURAL NETWORK

Vidian Vito Oktariyanto, Kestrilia Rega Prilianti, Hendry Setiawan

Universtias Ma Chung

Abstrak

Pada saat pandemi covid 19 tren tanaman hias meningkat di kalangan masyarakat. Seiring banyaknya permintaan tanaman hias muncul juga permasalahan yakni informasi-informasi yang kurang tepat bahkan bisa menjadi *hoax*. Berita yang tidak benar dan cenderung menakut-nakuti dapat menambahkan kecemasan bagi masyarakat ditengah pandemi apalagi bagi masyarakat yang awam terhadap tanaman hias. Maka dari itu tujuan penelitian ini adalah membuat aplikasi android yang membantu masyarakat dalam mendapatkan informasi mengenai tanaman hias dengan mudah.

Penelitian ini menggunakan metode *transfer learning* dengan model MobileNetV2 dari Keras ditambah dengan *layer custom* buatan sendiri untuk 9 kelas tanaman hias. Penelitian ini juga melakukan beberapa eksperimen dengan jumlah layer MobileNetV2 dan pemilihan *optimizer*. Kemudian untuk rancang bangun aplikasi android keseluruhannya menggunakan *framework* flutter, python, dan JSON sebagai penyimpanan data.

Dari eksperimen yang dilakukan keputusan yang diambil adalah menggunakan semua layer dari model MobileNetV2 dengan *Adagrad Optimizer* dengan akurasi 88%. Dapat disimpulkan bahwa dengan metode *transfer learning* dapat membantu menghasilkan nilai akurasi sebesar 88% dengan data latih yang terbatas yakni sebesar 300 citra, tentunya juga ditambah dengan data augmentasi.

Kata Kunci : Convolutional Neural Network, Transfer Learning, Adagrad, Flutter, Android

IDENTIFY OF ORNAMENTAL PLANTS WITH TRANSFER LEARNING IMPLEMENTATION IN CONVOLUTIONAL NEURAL NETWORK

Vidian Vito Oktariyanto, Kestrilia Rega Prilianti, Hendry Setiawan

Ma Chung University

Abstract

During the Covid 19 pandemic, the trend of ornamental plants increased among the public. Along with the large number of requests for ornamental plants, problems also arise, namely inaccurate information that can even become a hoax. Many news spread and tends to be frightening can add anxiety to the public in the midst of a pandemic, especially for people who are new with ornamental plants. Therefore, the purpose of this research is to create an android application that can helps people to get information about ornamental plants easily.

This study used the transfer learning method with the MobileNetV2 model from Keras plus custom layers for 9 classes of ornamental plants. This study also conducted several experiments with the number of MobileNetV2 layers and the optimizer selection. Then to design and build android applications that use the Flutter frameworks, Python, and JSON as data storage.

From the experiments carried out the decision was taken to use all layers from the MobileNetV2 model with the Adagrad Optimizer with an accuracy of 88%. It can be concluded that the transfer learning method can help produce an accuracy value of 88% with limited training data of 300 images, of course also added with augmentation data.

Kata Kunci : Convolutional Neural Network, Transfer Learning, Adagrad, Flutter, Android

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa karena atas rahmat dan karunia-Nya sehingga laporan Tugas Akhir (TA) ini dapat diselesaikan. Laporan ini berisi hasil dari pengembangan dan penelitian Tugas Akhir dengan judul "Identifikasi Tanaman Hias Dengan Implementasi Transfer Learning Pada Convolutional Neural Network".

Atas semua dukungan yang telah diberikan dalam pengerjaan penelitian ini, penulis ingin menyampaikan ucapan terima kasih kepada:

1. Orang tua yang telah mendukung pengerjaan Tugas Akhir hingga selesai,
2. Bapak Hendry Setiawan, ST, M.Kom selaku dosen pembimbing sekaligus Ketua Program Studi Teknik Informatika Universitas Ma Chung,
3. Ibu Dr.Kestrilia Rega Prilianti, M.Si selaku dosen pembimbing sekaligus Dekan dari Fakultas Sains dan Teknologi Universitas Ma Chung,
4. Bapak Mochamad Subianto, S.Kom., M.Cs. selaku pembimbing akademik
5. Bapak Agus Soenartono, S.Sos selaku BRIDS Advisor di Gerai BRI Danareksa Sekuritas Universitas Ma Chung yang telah memfasilitasi pengerjaan Tugas Akhir ini.
6. Hagi, Niken, Mifta, Justine, dan teman-teman lainnya yang telah medukung penulis selama pengembangan aplikasi dan penelitian ini berlangsung.

Laporan ini adalah salah satu prasyarat kelulusan. Dalam laporan ini tidak menutup kemungkinan terdapat beberapa kesalahan. Oleh sebab itu kritik dan saran sangat membantu bagi Penulis untuk memperbaiki kekurangan yang ada. Demikian semoga tugas akhir ini bisa bermanfaat bagi semua pihak.

Malang, 5 Januari 2023

Vidian Vito Oktariyanto
NIM. 311710018

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

Bab I
Pendahuluan

1.1 Latar Belakang

Pada masa pandemi Covid-19 saat ini, tren menanam tanaman hias sangat digemari masyarakat. Tujuannya pun berbeda-beda, salah satunya dijadikan sebagai hiasan di rumah. Ada juga masyarakat perkotaan yang tujuannya adalah digunakan sebagai pengimbang polusi udara yang dikarenakan oleh kendaraan bermotor. Selain itu tanaman hias ini juga berperan di dalam pembangunan sektor agrowisata di Indonesia. Sektor agrowisata ini memanfaatkan usaha pertanian sebagai objek wisata bisnis tanaman hias yang bagus untuk dikembangkan (Noviana, et al., 2014).

Tanaman hias saat ini banyak dijual di berbagai toko bunga atau bahkan di beberapa daerah ada yang dijual dengan cara berkeliling menggunakan mobil *pick up*. Harganya pun ekonomis mulai dari Rp 10.000,00 sampai di atas Rp 250.000,00. Hal ini menjadi kesempatan bagi penjual atau pemilik usaha tanaman hias. Namun di sisi lain, bagi pembeli biasanya mereka hanya membeli tanaman hias yang bagus dilihat dan murah, tanpa mengetahui apa manfaat dan bahayanya.

Seiring dengan berkembangnya teknologi di jaman sekarang yang berkembang pesat, informasi sangatlah mudah didapatkan lewat internet, salah satu perangkatnya yaitu mengakses melalui ponsel pintar. Informasi di internet pun beragam, ada informasi mengenai pendidikan, olahraga, berita, kesehatan, dan lain - lain. Namun banyak juga dari beberapa informasi tersebut masih ada yang mengandung hoaks atau informasi palsu. Menurut Pellegrini (2008) mengembangkan definisi hoaks dari McDougall kemudian menjelaskan bahwa hoaks merupakan sebuah kebohongan yang dikarang sedemikian rupa oleh seseorang untuk menutupi atau mengalihkan perhatian dari kebenaran dengan tujuan untuk kepentingan pribadi baik secara intrinsik maupun ekstrinsik.

Salah satu masalah yang timbul adalah beredarnya berita hoaks mengenai tanaman hias, salah satunya tanaman sri rejeki atau *dieffenbachia omoena*. Pada tahun 2019 beredar berita bahwa tumbuhan sri rejeki tersebut mengandung getah racun yang bisa mematikan anak-anak kurang dari 1 menit. Setelah dikonfirmasi, menurut Dr. Susiani Purbaningsih, DEA, dari Fakultas Matematika dan Ilmu

Pengetahuan Alam Universitas Indonesia mengatakan bahwa racun dari tanaman sri rejeki tergolong ringan karena hanya menyebabkan gatal-gatal ringan saja (Kominfo, 2019).

Dari satu berita kebohongan tersebut saja pastinya akan berdampak buruk khususnya bagi penjual tanaman hias sri rejeki. Begitu juga bagi pembeli atau pemelihara bunga sri rejeki, yang seharusnya bunga tersebut menjadi hiasan indah di rumah malah bisa saja tanaman tersebut dipotong dan dibuang.

Penelitian ini bertujuan membantu masyarakat dalam hal mengidentifikasi tanaman hias beserta manfaat dan bahayanya menggunakan aplikasi android. Adapun sumber mengenai manfaat dan bahaya dari tanaman hias diambil dari beberapa *website* di internet yang mempunyai domain jelas seperti .com, .id, .co.id, dan lain - lain. Manfaat dan bahaya dari tanaman hias contohnya seperti tanaman hias lidah buaya (*Aloe vera*), tanaman ini mempunyai segudang manfaat tetapi juga mempunyai bahaya yang perlu diperhatikan. Salah satu manfaatnya yaitu mengatasi kulit kering, karena lidah buaya mengandung vitamin A, C, E yang baik untuk menguatkan lapisan pelindung kulit sehingga kulit tetap lembab (Nareza, 2021). Disisi lain lidah buaya juga mempunyai bahaya efek samping resiko sakit perut, kram, dan diare apabila ekstrak lidah buaya ini dikonsumsi oleh anak dibawah usia 12 tahun (Putri, Tanpa Tahun). Penelitian ini menggunakan pre-trained model dari keras yang menyediakan banyak model deep learning siap pakai dengan mengatur parameter *weights* imagenet. ImageNet sendiri merupakan dataset yang berisikan jutaan gambar berbagai objek, dari tanaman, hewan, sayuran, benda, dan lain-lain. Metode yang dipakai di penelitian ini adalah menggunkan teknik transfer learning pada *Convolutional Neural Network* (CNN).

Transfer learning merupakan teknik yang memanfaatkan model orang lain yang sudah dilatih untuk digunakan mengklasifikasi *dataset* yang baru, sehingga tidak perlu melakukan *training* data dari awal (Tan et al, 2018). Teknik ini bisa diimplementasikan ke beberapa metode, salah satunya metode CNN.

CNN merupakan metode salah satu jenis *neural network* yang bisa digunakan untuk mengenali objek pada suatu citra. CNN termasuk dalam jenis *Deep Neural Network* yang merupakan pengembangan dari *Multiplater Perceptron* (MLP) yang digunakan untuk mengolah data yang berbentuk dua dimensi.

Beberapa penelitian yang menggunakan metode CNN menghasilkan akurasi deteksi objek dengan hasil yang bagus. Penelitian pengenalan *gender* yang dilakukan oleh (Smith & Chen, 2019) menghailkan akurasi 98,56%. Penelitian kedua dilakukan oleh (Kaur & Gandhi, 2019) dengan teknik *transfer learning* pada CNN dengan arsitektur VGG mampu menghasilkan akurasi sampai 100% tentang klasifikasi otak.

1.2 Identifikasi Masalah

Identifikasi masalah pada penelitian ini adalah banyaknya informasi-informasi mengenai tanaman hias yang terkadang kurang tepat atau bahkan *hoax* pada saat pandemi.

1.3 Batasan Masalah

Batasan masalah dari pengerjaan penelitian ini sebagai berikut.

1. Identifikasi tanaman hias berdasarkan daunnya.
2. Klasifikasi menggunakan metode CNN.
3. *Dataset* didapatkan dari internet dan foto sendiri.
4. Penelitian ini dibuat menjadi aplikasi android.
5. Output berupa nama tanaman, manfaat, dan bahaya tanaman hias.

1.4 Perumusan Masalah

Perumusan masalah dalam penelitian ini yaitu bagaimana membantu masyarakat dalam mencari informasi mengenai tanaman hias menggunakan bantuan teknologi berupa aplikasi pada *smartphone*.

1.5 Tujuan Penelitian

Tujuan pengerjaan penelitian ini adalah membuat aplikasi android yang dapat membantu masyarakat untuk menambah wawasan mengenai informasi tanaman hias beserta manfaat dan bahayanya menggunakan metode *transfer learning convolutional neural network* yang kemudian diterapkan ke aplikasi androidnya.

1.6 Luaran

Luaran dari penelitian ini berupa aplikasi android yang mengidentifikasi tanaman hias dengan manfaat dan bahayanya beserta publikasi ilmiah.

1.7 Manfaat

Manfaat yang didapat dari pengerjaan penelitian ini adalah membantu masyarakat mengetahui manfaat dan bahaya dari tanaman hias yang mereka tanam di rumah, dibantu dengan aplikasi android yang dimana saat ini mayoritas masyarakat sudah mempunyai ponsel pintar.

1.8 Sistematika Penulisan

Sistematika penulisan laporan penelitian Tugas Akhir ini ditulis dengan seperti berikut.

Bab 1. Pendahuluan

Pada bab pendahuluan ini, penulis akan menerangkan latar belakang yang akan menjadi dasar penulis membuat penelitian ini. Rumusan masalah dan juga tujuan hingga manfaat dalam penelitian ini.

Bab 2. Tinjaun Pustaka

Bab ini berisikan penjelasan mengenai teori-teori pendukung penelitian yang dibuat. Dimulai dari penjelasan metode convolutional neural network, arsitekturnya, teknik transfer *learning*, android, dan plugin atau library yang digunakan.

Bab 3. Analisis dan Desain Sistem

Pada bab ini berisi tentang tata cara melakukan persiapan *dataset*, pengolahan menjadi model, implementasi *pretrained* model yang sudah disiapkan, purwarupa aplikasi androidnya sampai proses pelatihan dan pengenalan.

Bab 4. Hasil dan Pembahasan

Pada bab ini berisi tentang hasil implementasi pengerjaan dari bab 3, mulai dari ekperimen pembuatan model, ekspor model ke .tflite, membuat data JSON, membuat aplikasi android, dan testing black box aplikasi.

Bab 5. Saran dan Kesimpulan

Pada bab ini berisi tentang kesimpulan dari penelitian secara keseluruhan beserta saran-saran yang berguna bagi penelitian selanjutnya.

Bab II
Tinjauan Pustaka

2.1 Tanaman Hias

Merupakan semua jenis tanaman yang memiliki nilai fungsi sebagai keindahan atau kecantikan hiasan di suatu ruangan atau tempat. Tanaman ini mencakup semua jenis tanaman dari yang merambat, semak-semak, dan juga ada yang pohon. Selain keindahan yang dimunculkan dari warna dan aroma bunganya, ada beberapa bagian juga yang bisa dimanfaatkan dalam tanaman hias, seperti daun, buah, batang, bahkan pepagan yang dapat menjadi produk-produk seperti keranjang, rempah-rempah, bambu, dan lain-lain. Selain itu ada juga tanaman hias yang memiliki bahaya seperti racun. Misalnya tanaman hias lidah mertua, menurut *The American Society for the Prevention of Cruelty to Animals* atau ASPCA dalam *website* mereka, mereka mengkategorikan lidah mertua sebagai tanaman beracun yang mengandung saponins dan berbahaya bagi binatang peliharaan seperti anjing dan kucing (ASPCA, Tanpa Tahun).

Ada berbagai jenis tanaman hias, diantaranya tanaman hias daun, bunga, buah, pohon, dan akar. Pada penelitian ini fokus menggunakan tanaman hias berjenis daun, berikut penjelasan mengenai tanaman hias daun :

2.1.1 Tanaman Hias Daun

Tanaman hias daun merupakan tanaman yang biasanya tidak memiliki bunga. Tanaman ini mengandalkan daunnya sebagai komponen utama yang selain menjadi daya tarik sendiri juga bisa dimanfaatkan sebagai beberapa produk, seperti daun tanaman suplir, daun tanaman tersebut dapat digunakan sebagai obat menetralisir gangguan ginjal. Contoh lain tanaman hias daun adalah keladi red star, aglaonema, dieffenbachia, kuping gajah, paku tanduk rusa, dan saka asparagus plumosus.

Gambar 2.1 Daun Lidah Mertua (Levina, 2022)

Ada 9 tanaman hias daun yang dipilih pada penelitian ini, yakni Andong, Cornus Alba, Krokot Epah Daun Merah, Lidah Buaya, Lidah Mertua, Miana, Pucuk Merah, Puring, dan Sri Rejeki. Berikut adalah detailnya.

1. Andong

   Tanaman andong merupakan jenis tanaman yang biasanya ditanama di pinggir jalan, kuburan, atau dijadikan tanaman pagar. Di Bali tanaman andong ini lebih dikenal dengan nama Endong yang sering ditanam oleh masyarakat sebagai tanaman hias (Suarsana, et al., 2014). Jenis tanaman termasuk dalam kategori monokotil, tinggi tanaman daun andong 2-4 meter, batangnya bulat, keras, bekas daun rontok berbentuk cincin. Daunnya tunggal dengan warna hijau, ada juga yang berwarna merah kecoklatan. Letak daun tersebar pada batang, terutama berkumpul di ujung batang.

Gambar 2.2 Tanaman Andong

2. *Cornus alba*

   Cornus alba atau *dogwood* merupakan tanaman yang tergolong dalam genus *Cornus*. Tanaman ini juga dapat disebut juga corniza tanduk, sanguino, atau ceri liar. Ciri-ciri dari tanaman ini adalah berupa semak atau pohon berukuran sedang sekitar 3 sampai 4 meter yang sangat rimbun (Asep, 2023). Daunnya ada yang berbentuk kebulat-bulatan berwarna hijau di tengahnya kemudian berwarna putih di bagian tepinya.

Gambar 2.3 Tanaman Kornus Alba

3. Krokot Epah Daun Merah

   Tanaman Krokot Epah ini merupakan tanaman yang dijadikan sebagai tanaman hias penutup tanah. Tanaman ini berasal dari Amerika Selatan yang dapat tumbuh hingga 60 sampai 80 cm (Azizah, 2021). Daunnya berwarna campuran hijau merah keunguan. Tanaman ini biasa dapat ditemui di sudut-sudut taman. Krokot epah seperti gambar 2.4 sekilas mirip seperti bayam merah, ini dikarenakan krokot merah dengan bayam merah masih dalam keluarga yang sama.

Gambar 2.4 Tanaman Krokot Epah Daun Merah (Azizah, 2021)

4. Lidah Buaya

   Tanaman Lidah Buaya atau *Aloe vera* merupakan tanaman yang sudah lama dikenal oleh masyarakat Indonesia sebagai salah satu tanaman hias atau sebagai tanaman obat. Selain sebagai obat tanaman ini juga bisa dimanfaatkan untuk kecantikan, atau sumber berbagai macam nutrisi (Zulfia, 2012). Tanaman ini mempunyai pelepah berwarna hijau dengan bunganya yang berwarna oranye. Tanaman ini mempunyai ciri bintik warna putih ketika tanaman ini berusia masih muda.

Gambar 2.5 Tanaman Lidah Buaya

5. Lidah Mertua

   Lidah mertua ini juga populer di kalangan masyarakat Indonesia sebagai tanaman hias. Tanaman ini sering dijumpai di dalam rumah sebagai penghias ruangan. Tanaman ini dapat tumbuh dengan cahaya matahari dengan air yang sedikit sehingga perawatannya tidaklah sulit dan merepotkan. Ciri-ciri dari daun tanaman lidah mertua ini mempunyai daun yang keras, tegak dengan ujung yang runcing, dan mempunyai bermacam-macam corak dengan warna yang dominan hijau (Zulfia, 2012).

Gambar 2.6 Tanaman Lidah Mertua

6. Miana

   Tanaman Miana atau *Coleus scutellarioides* merupakan tanaman hias yang sering dijumpai di berbagai tempat seperti taman-taman perkotaan. Warna pada daun miana ini cukup beragam dan atraktif mulai dari hijau, ungu, ungu ke merah-merahan, atau merah dengan tepi berwarna hijau, dan lain-lain. Selain warnanya yang atraktif daun ini memiliki manfaat untuk kesehatan seperti penetralisir racun, antiseptik, obat hepatitis, batuk, menurunkan demam, influenza, dan lain-lain (Ningsih & Rohmawati, 2019).

Gambar 2.7 Tanaman Miana

7. Pucuk Merah

   Pucuk merah atau *Syzygium oleana* merupakan salah satu tanaman hias yang menarik. Tanaman ini biasanya dapat dijumpai di beberapa taman-taman perkotaan. Pucuk merah memiliki keindahan warna pada daunya, dianyara lain berwarna merah, hijau muda, dan kuning. Tanaman pucuk merah ini selain mempunyai estetika yang cantik pada daunnya, pucuk merah juga memiliki serapan yang cukup terhadap CO2 di udara.

Gambar 2.8 Tanaman Pucuk Merah

8. Puring

   Puring atau *Codiaeum variegatum* adalah tanaman hias yang mempunyai daun beraneka warna. Dulu tanaman ini dijadikan sebagai pendamping makam oleh masyarakat. Namun sekarang tanaman ini dapat dijumpai di berbagai tempat seperti perkantoran, hotel, taman, dan lain-lain. Pola dari daun puring ini bervariasi mulai dari berbentuk mirip seperti jari manusia, keriting spiral, terkadang seperti bentuk huruf “z”.

Gambar 2.9 Tanaman Puring

9. Sri Rejeki

   Di Indonesia tanaman ini juga dikenal sebagai *aglaonema*. Tanaman hias ini banyak ditanam di pekarangan atau sebagai dekorasi ruangan. Karakter dari tanaman ini mempunyai beragam warna dan motif. Warna dari tanaman ini cenderung hijau dengan corak warna putih di tengahnya. Harga dari tanaman sri rejeki ini beragam mulai dari ribuan hingga ratusan ribu.

Gambar 2.10 Tanaman Sri Rejeki

2.2 Machine Learning

Machine Learning merupakan salah satu cabang dari *Artificial Intellegent. Machine Learning* ini banyak digunakan untuk menggantikan tugas manusia dalam menyelesaikan masalah atau bahkan dengan otomatisasi. Seperti dengan namanya, *machine learning* atau pembelajaran mesin ini butuh mempelajarari sesuatu

sebelum menyelesaikan masalah dengan membutuhkan data, kemudian dari sebuah data lalu akan diproses memakai algoritma-algoritma pembelajaran. *Machine Learning* mempunyai ciri khas yaitu adanya proses *training* atau pembelajaran. Maka dari itu peran data sangat penting, karena machine learning tidak akan bisa bekerja apabila tidak ada data (Kusumaningrum, 2018).

Ciri *machine learning* yang pertama adalah proses pelatihan atau *training*. Proses ini menggunakan sebagian data untuk melatih algoritma pembelajaran yang hasilnya nanti akan berupa model, kemudian ciri kedua yaitu data *testing*. Data *testing* ini berfungsi sebagai data ujicoba untuk menilai performa data yang sudah melalui proses pelatihan atau *training*. Dari hasil testing nantinya akan muncul laporan akurasi model berdasarkan data dan algoritma pembelajaran yang sudah ditentukan. Kemudian machine learning ini terbagi menjadi tiga jenis dengan cara pembelajaran yang berbeda, yaitu :

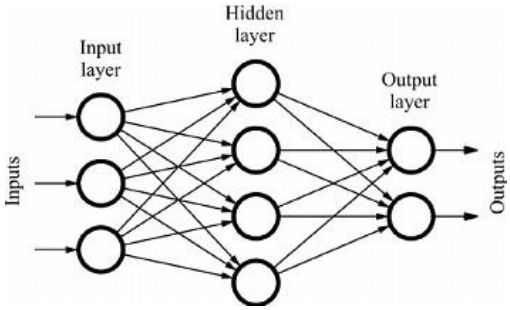
1. *Supervised Learning*, data *training* dengan label output yang sudah ditentukan.
2. *Unsupervised Learning*, data *training* dengan label output yang tidak ditentukan.
3. *Reinforced Learning*, akan ada pemberian hadiah disaat algoritma dijalankan di setiap tindakan yang dilakukan.

Penerapan Machine Learning di bidang teknologi contohnya seperti pengenalan wajah pada beberapa *device*, *handphone* misalnya, *fingerprint recognition*, pengenalan objek tertentu seperti yang ada di *smart car*, dan masih banyak lagi.

2.3 Deep Learning

Deep learning adalah bagian dari *machine learning* yang terinspirasi dari struktur otak manusia. Struktur tersebut dinamakan *Artificial Neural Networks* (ANN). Dalam struktur *deep learning* mempunyai tiga lapisan yang terdiri dari tiga bagian yaitu *input layer*, *hidden layer*, dan *output layer*. Dalam *input layer* berisikan *node-node* yang di setiap nodenya menyimpan nilai input tetap pada tahap *train* dan akan berubah jika diberikan nilai baru. Untuk *hidden layer* biasanya terdiri dari beberapa lapis yang berguna untuk menemukan komposisi algoritma yang tepat supaya bisa meminimalisir *error* pada *output*. Sedangkan untuk output layer

berfungsi untuk menampilkan hasil perhitungan sistem dengan fungsi aktivasi di bagian *hidden layer* berdasarkan *input*. Gambaran dari arsitektur deep learning dengan tiga lapisan tersebut contohnya adalah arsitektur *Multi Layer Perceptron* (MLP).



Gambar 2.11 Multi Layer Perceptron

Gambar 2.11 merupakan arsitektur *Multi Layer Perceptron* sederhana dengan 3 buah *neuron* pada *input layer*, 4 buah *neuron* pada *hidden layer*, dan 2 buah *neuron* padah *output layer*. *Neuron-neuron* pada masing-masing *layer* akan langsung terhubung dengan *neuron* lain pada *layer* setelahnya.

Deep Learning atau *deep structural learning* atau *h irarchial learning* atau *deep neural* juga merupakan metode pembelajaran yang menggunakan multiple linear transformation (Nugroho, et al., 2020). Ada beberapa algoritma yang termasuk dalam kategori deep learning, diantaranya :

1. *Convolutional Neural Network (CNN).*
2. *Long Short Term Memory Network (LSTM).*
3. *Reccurent Neural Network (RNN).*
4. *Self Organizing Maps (SOM).*
5. *Deep Neural Network (DNN).*
6. *Restricted Boltzmann Machine (RBM).*
7. *Deep Belief Network (DBN).*
8. *Stacked Autocoders.*

2.4 Algoritma Backpropagation

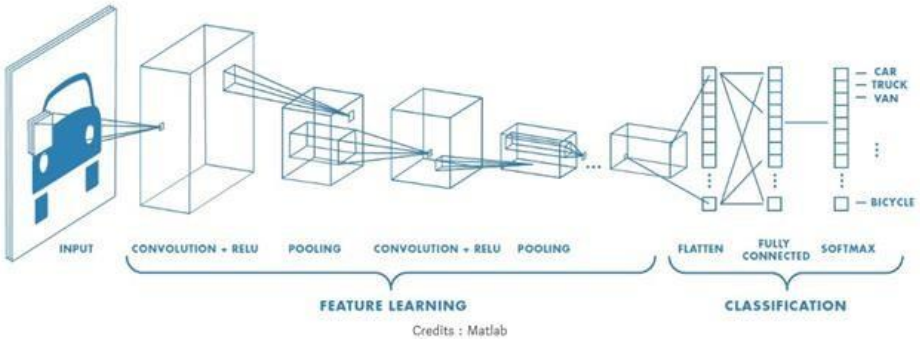
Neural network atau jaringan saraf adalah suatu sistem komputasi yang menyerupai jaringan saraf manusia. Dalam jaringan ini membutuhkan suatu proses pembelajaran yang dimana tujuan dari pembelajaran tersebut adalah untuk

melakukan proses dalam menentukan nilai bobot (*weight*) yang tepat ke setiap inputnya. Salah satu algoritma pembelajaran yang ada di *neural network* adalah *backpropagation*. *Backpropagation* merupakan algoritma pembelajaran yang terawasi yang menggunakan perceptron dengan banyak lapisan untuk mengubah setiap bobot yang terhubung dengan *neuron-neuron* yang ada pada *hidden layer*. Algoritma ini menggunakan *error* bertujuan untuk mengubah nilai bobot dalam arah mundur (*backward*). Ada tahap sebelum mendapatkan *error* tersebut, yakni tahap perambatan maju (*feedforward*).

Ada tiga tahap dalam pelatihan backpropagation, yaitu ada tahap maju, tahap ini menghitung maju tahap layer input sampai layer output. Kemudian tahap kedua yaitu tahap mundur, pada tahap ini selisih antara output jaringan dengan target yang diinginkan merupakan error yang terjadi. *Error* atau kesalahan tersebut akan di propagasi mundur dimulai dari garis yang terhubung langsung dengan setiap unit pada layer output. Terakhir, tahap ketiga yakni tahap memodifikasi nilai bobot, tahap ini menurunkan tingkat *error* yang muncul.

2.5 Convolutional Neural Network

Convolutional Neural Network merupakan salah satu algoritma dari deep learning yang merupakan pengembangan dari *Multi Layer Perceptron* (MLP) yang digunakan untuk mengolah data dua dimensi seperti gambar atau suara. CNN juga termasuk dalam jenis *Deep Neural Network* (DNN) karena mempunyai kedalaman jaringan yang tinggi dan biasa diaplikasikan pada data berjenis citra. CNN memiliki beberapa struktur yang terdiri dari *input*, *feature learning* atau ekstraksi fitur, klasifikasi dan *output*.



Gambar 2.12 Arsitektur CNN (Saha, 2018)

Berikut penjelasan dari setiap proses arsitektur CNN:

a. *Feature Learning*

Proses ini melakukan penerjemahan dari input menjadi features menggunakan lapisan-lapisan berdasarkan ciri dari inputan tersebut. *Output* dari proses ini adalah berupa angka dalam vektor. Ada dua lapisan pada proses *feature learning* ini, antara lain *Convolutional Layer* dan *Pooling Layer*. Berikut adalah proses-proses yang ada di ruang lingkup *feature learning*.

1. *Convolutional Layer*

   Convolutional layer ini terdiri dari beberapa neuron yang tersusun membentuk sebuah filter dengan panjang dan tinggi dalam satuan pixel. Filter ini berisikan bobot yang digunakan untuk mendeteksi karakter dari setiap objek, contohnya seperti tepi, kurva, atau warna. Setiap filter yang terbentuk akan digeser ke seluruh bagian dari suatu matriks citra. Output yang akan dihasilkan nantinya disebut features map yang dihasilkan dari proses "dot" antara input dan nilai dari filter tersebut. Proses "dot" tersebut dilakukan berulang setiap pergeseran terjadi.

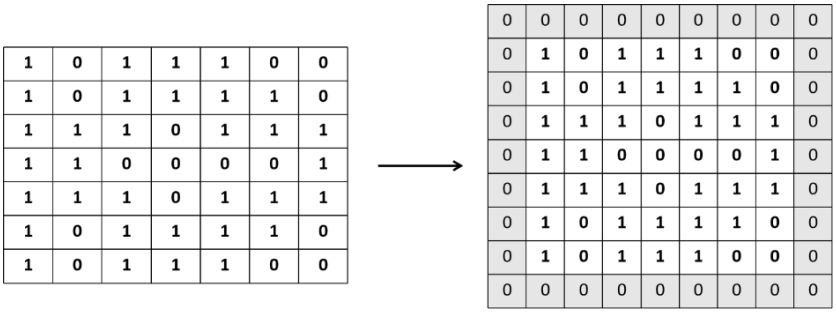
2. *Stride*

   Stride adalah parameter yang digunakan untuk menentukan jumlah pergeseran filter. Contoh, jika nilai stride adalah 1, maka lapisan konvolusi (*Convolution Layer*) akan bergeser sebanyak 1 pixel secara horizontal dan vertikal. Semakin kecil nilai *stride* maka detil informasi yang didapatkan semakin bagus dari input, akan tetapi hal tersebut dapat memakan waktu yang lebih lama dalam melakukan komputasi dibandingkan dengan *stride* yang bernilai besar. Menggunakan nilai *stride* yang kecil juga tidak selalu akan mendapatkan performa yang bagus.

3. *Padding*

   Padding atau *zero padding* merupakan parameter yang digunakan untuk menentukan jumlah *pixel* yang berisikan nilai 0 untuk ditambahkan di setiap sisi dari sebuah input. *Padding* ini bertujuan

untuk memanipulasi dimensi *feature map* yang merupakan *output* dari lapisan konvolusi (*Convolutional Layer*). *Output* dari proses ini berupa data yang akan digunakan sebagai inputan pada lapisan konvolusi selanjutnya. Berikut ilustrasi pemberian *padding* pada suatu citra.



Gambar 2.13 Contoh Pemberian Padding.

4. Fungsi Aktivasi

   Proses fungsi aktivasi ini ada di tahap sebelum melakukan *pooling layer* dan sesudah proses konvolusi. Terdapat beberapa fungsi aktivasi yang digunakan pada CNN, namun umumnya peneliti menggunakan aktivasi ReLU. Aktivasi ReLU menjadi pilihan yang sering dipilih oleh peneliti karena sifatnya yang berfungsi dengan baik. Fungsi ReLU merupakan sebuah nilai dari *output* dari *neuron* yang dinyatakan sebagai 0 jika nilai inputannya berupa negatif. Namun jika positif, maka output dari neuron adalah nilai input aktivasi itu sendiri. Fungsi aktivasi ReLU mengaplikasikan fungsi (x) = max(0,x) artinya apabila $x \leq 0$ maka $x = 0$ dan apabila $x > 0$ maka $x = x$.

5. *Pooling Layer*

   Pooling layer ini terdiri dari filter dengan ukuran dan stride tertentu yang selalu bergeser pada seluruh area *feature map*. Proses ini biasanya dilakukan setelah proses *convolution layer*. Tujuan dari *pooling layer* adalah untuk melakukan downsampling atau mengurangi dimensi pada feature map guna mempercepat komputasi dan mencegah terjadinya *overfitting*. *Overfitting* adalah suatu keadaan dimana hampir semua data yang telah melalui proses

training menghasilkan persentase yang baik, akan tetapi terjadi ketidaksesuaian ketika proses prediksi (Santoso & Ariyanto, 2018)

b. Classification

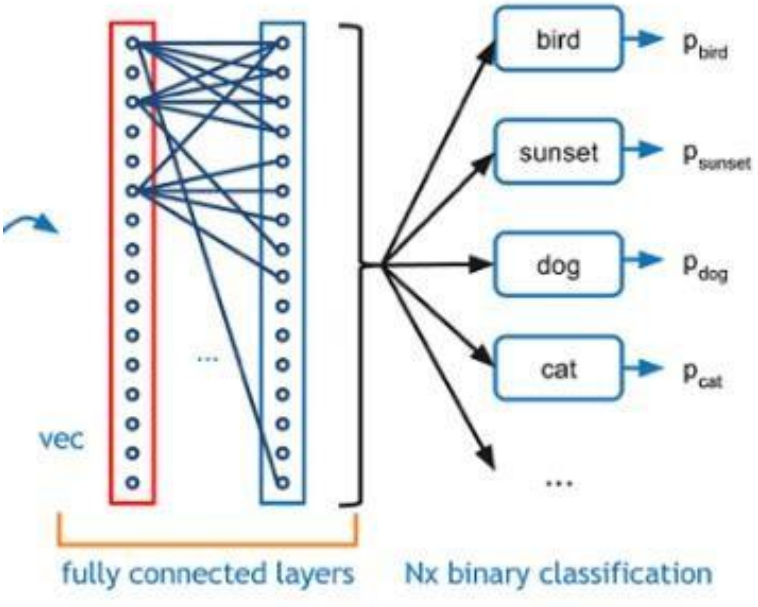
Pada tahap ini, proses klasifikasi dijalankan untuk setiap neuron yang telah melalui tahap sebelumnya yaitu *feature learning* atau pembelajaran fitur. *Classification* mempunyai tiga lapisan yang terdiri dari:

1. *Flatten*

   Pada tahap ini, proses *flatten* atau *reshape* akan menghasilkan sebuah vektor yang akan digunakan sebagai input *Fully Connected Layer* dari fitur yang sudah diproses sebelumnya.

2. *Fully Connected Layer*

   Setelah melalui proses *Flatten*, pada fully connected layer ini semua neuron aktvitas dari lapisan sebelumnya terhubung semua ke neuron lapisan selanjutnya. Pada gambar 2.14 adalah gambaran *fully connected layer* dimana sebelumnya menerima inputan berupa *vector* yang berasal dari proses Flatten, setelah itu proses akan dihubungkan ke layer selanjutnya untuk proses klasifikasi dengan fungsi aktifasi.



Gambar 2.14 *Fully Connected Layer* (Yanuar, 2018)

3. *Softmax Activation*

Softmax merupakan fungsi yang menghitung probabilitas untuk setiap kelas target, kemudian probabilitas kelas yang mempunyai nilai tertinggi akan dijadikan sebagai hasil prediksi (Rizal, et al., 2020). Hasil yang akan didapat dari fungsi *softmax* ini adalah rentang probabilitas output yang bernilai dari 0 hingga 1. Berikut adalah rumus dari *Softmax Activation*.

$$h_t = softmax(W_{hh}h_{t-1} + W_{xh}X_t) \quad (2\text{-}2)$$

Dimana,

W : *weight*

h : *single hidden vector*

W_{hh} : *weight* pada *hidden state* sebelumnya

W_{xh} : *weight* pada *input state* sekarang

h_t : nilai *vector* pada *hidden layer*

Selanjutnya guna mendukung proses identifikasi menggunakan CNN, CNN juga memerlukan komparasi arsitektur di dalamnya. Terdapat berbagai jenis arsitektur yang bisa digunakan, beberapa diantaranya adalah AlexNet, VGGNet, ResNet, DenseNet, Inception, Xception, MobileNet, dan lain-lain.

2.6 Optimizer

Optimizer merupakan algoritma optimisasi yang bertujuan untuk mendapatkan bobot optimal dengan meminimalkan kesalahan dan memaksimalkan akurasi. Dalam proses pelatihan, parameter (bobot) model diubah dengan maksud mencoba dan meminimalkan fungsi kerugian agar mampu melakukan prediksi dengan hasil yang paling akurat. Pada proses pelatihannya, pengubahan-pengubahan parameter tersebut masih belum dapat dipastikan mengenai banyaknya, kapan, serta cara yang tepat. Maka dari itu inilah peran optimizer. *Optimizer* menyatukan fungsi kerugian dan parameter model dengan merubah model dalam menangani output dari fungsi kerugian. Dengan kata lain, pengoptimalisasi membuat model dalam bentuk yang paling akurat memanfaatkan bobotnya.

2.6.1 Adam Optimizer

Adam merupakan algoritma optimalisasi yang dapat mengoptimalkan fungsi *objective stochastic* berdasarkan perkiraan adaptif momen. Algoritma ini pertama kali dikenalkan oleh Diederik Kingma. Dalam penggunaanya, *optimizer* Adam memerlukan penentuan nilai *learning rate* yang akan digunakan dalam proses pelatihan *training dataset*. *Learning rate* merupakan salah satu parameter *training* yang ditetapkan guna menghitung nilai koreksi bobot pada waktu proses *training*. Berikut adalah langkah-langkah pada *optimizer* Adam :

Pertama, menghitung nilai m_t menggunakan persamaan berikut.

$$m_t = \beta_1 * m_{t-1} + (1-\beta_1) * g_t \quad (2\text{-}3)$$

dimana,

m_t : Nilai momentum pertama

β_1 : Konstanta laju peluruhan

m_{t-1} : Inisialisasi vektor momentum pertama

g_t : Nilai gradien bobot

Kedua, menghitung nilai v_t dengan persamaan berikut.

$$v_t = \beta_2 * v_{t-1} + (1-\beta_2) * g_t^2 \quad (2\text{-}4)$$

dimana,

v_t : Nilai momentum kedua

β_2 : Konstanta laju peluruhan

v_{t-1} : Inisialisasi vektor momentum kedua

g_t : Nilai gradien bobot

Ketiga, menghitung nilai $\check{m}_t$ menggunakan persamaan berikut.

$$\check{m}_t = \frac{mt}{\sqrt{1-\beta_1^t}} \quad (2\text{-}5)$$

dimana,

$\check{m}_t$: Nilai momentum pertama dengan pembaruan bias

β_1 : Konstanta laju peluruhan

m_t : Nilai momentum pertama

t : *Epoch*

Keempat, menghitung nilai v menggunakan persamaan berikut.

$$\hat{v}_t = \frac{\hat{v}_t}{\sqrt{1-\beta_2^t}} \quad (2\text{-}6)$$

dimana,

$\hat{v}_t$: Nilai momentum kedua dengan pembaruan bias

β_1 : Konstanta laju peluruhan

v_t : Nilai momentum kedua

t : *Epoch*

Langkah terakhir, memperbarui bobot dengan menggunakan persamaan berikut.

$$\theta_t = \theta_{t-1} - \alpha\left(\frac{mt}{\sqrt{\hat{v}_t + s}}\right) \quad (2\text{-}7)$$

dimana,

θt : Nilai bobot yang telah diperbaharui

θt-1 : Nilai bobot lama

$\check{m}_t$: Nilai momentum pertama dengan pembaruan bias

$\hat{v}_t$: Nilai momentum kedua dengan pembaruan bias

α : *Learning rate*

2.6.2 RMSprop Optimizer

Optimizer ini mirip dengan *Adaprop*, yang merupakan hasil improvisasi dari *Adagrad*. Memiliki gradien kompleks yang mirip seperti jaringan saraf. *RMSprop* adalah fungsi optimasi yang isinya memanfaatkan besarnya gradien terbaru untuk membuat gradien secara normal. *RMSprop* adalah salah satu fungsi optimasi yang membuat nilai rata-rata dan kuadrat gradien pada setiap bobot bertahan. Formulanya seperti di bawah ini.

$$MeanSquare(w,t) = p * MeanSquare(w, t\text{-}1) + 0.1(\partial E / \partial w\ (t))^2 \quad (2\text{-}8)$$

Keterangan:

w = bobot

t = *timestep*

p = 0.9

$\partial E / \partial w$ = gradient

2.6.3 SGD Optimizer

Stochastic Gradient Descent atau SGD adalah optimasi dengan algoritma yang digunakan untuk memperbarui parameter bobot dan bias. Algoritma dalam optimasi ini cukup sederhana, yaitu mengurangi initial weight atau bobot dengan "sebagian" dari nilai gradient yang telah diperoleh (Sitepu & Sigiro, Tanpa Tahun). Berikut adalah rumus *update parameter* menggunakan *Gradient Descent*.

$$\theta = \theta - \eta \,.\nabla_i \,.(\theta) \tag{2-9}$$

dimana,

θ = bobot

η = *learning rate*

∇i = *Gradient Descent*

2.6.4 Adadelta Optimizer

Adaptive Learning Rate Algorithm atau Adadelta merupakan improvisasi dari algoritma SGD. Optimizer sebelumnya yakni SGD, algoritmanya membutuhkan pengaturan *learning rate* secara manual yang dimana hal ini sangat mempengaruhi tingkat presisi dalam hal melakukan prediksi (Qu, et al., 2019). Sedangkan di dalam optimizer Adadelta algoritmanya mampu secara otomatis mengatur *learning rate* sebagai pengoptimalan dari algoritma SGD dan juga mampu meningkatkan presisi dalam melakukan prediksi.

2.6.5 Adagrad Optimizer

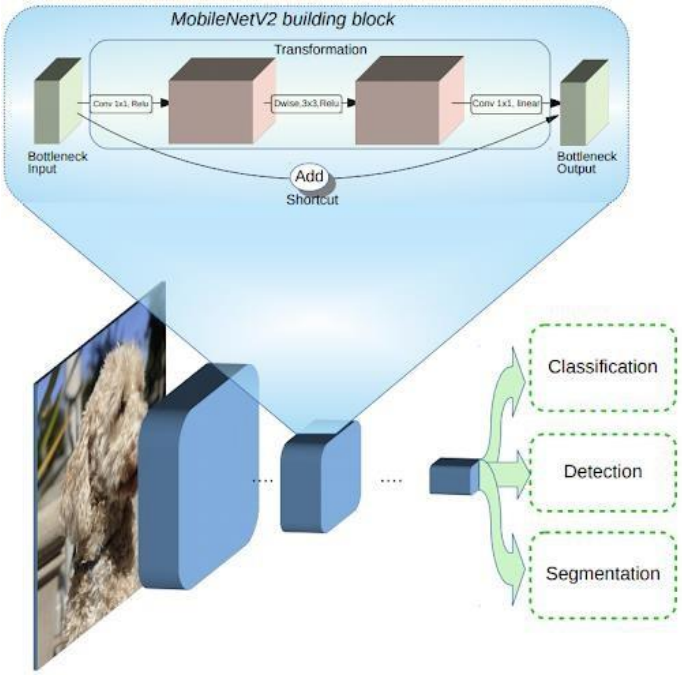
Adagrad merupakan *optimizer* dengan algoritma berbasis *gradient* yang mempunyai variabel *learning rate* di dalam parameter. Dalam algoritma optimasi ini apabila parameternya semakin banyak yang di-*update* maka nilai *learning rate*-nya akan berkurang semakin kecil (Zahara, et al., 2019). Maka nilai *learning rate* yang dihasilkan nantinya akan beradaptasi semakin lama akan mempunyai nilai yang optimum. Optimasi *adagrad* ini merupakan turunan dari SGD yang cocok untuk data skala besar.

2.6.6 Nadam Optimizer

Nadam adalah singkatan dari *Nesterov-accelarated Adaptive Moment Estimation* yang merupakan kombinasi dari *Adam* dan momentum Nesterov. Dalam beberapa *case* optimasi *Nadam* dapat melampaui kinerja *Adam*, ini dilihat secara teoritis dan terkadang empiris momen Nesterov tersebut lebih baik dari momentum konvensional. *Nadam* ini sudah termasuk di salah satu optimasi pada API Keras.

2.7 Model MobileNetV2

MobileNet merupakan salah satu arsitektur CNN (*Convolutional Neural Network*) yang biasanya digunakan untuk membantu sistem dalam keterbatasannya untuk melakukan komputasi dalam menjalankan tugas seperti deteksi objek menggunakan *deep learning* dengan menggunakan *layer* konvolusi *depthwise* dan *pointwise*. MobileNetV2 merupakan model yang dikembangkan dari model versi sebelumnya yaitu MobileNets. Seperti namanya, arsitektur CNN ini dibuat untuk dapat digunakan pada ponsel. Perbedaan yang mendasar antara arsitektur MobileNet dan arsitektur lainnya adalah penggunaan lapisan konvolusi dengan ketebalan filter yang sesuai dengan *input image*. Berikut di bawah ini adalah gambaran arsitektur MobileNetV2.



Gambar 2.15 Arsitektur MobileNetV2 (Okta, 2018)

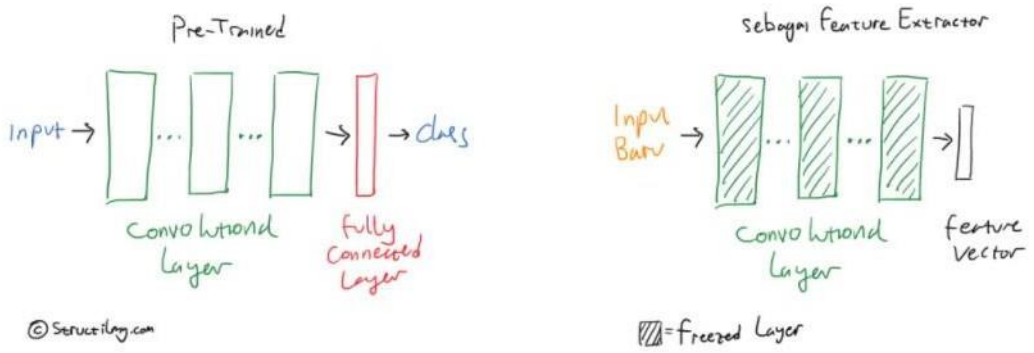
MobileNetV2 ini masih menggunakan *depthwise* dan *pointwise convolution* seperti versi sebelumnya. Kemudan para peneliti menambahkan dua fitur baru yaitu *linear bottleneck* dan *shortcut connections* antar *bottleneck*. Untuk fitur linear bottleneck terdapat input dan output antar model, sedangkan pada lapisan bagian dalamnya melakukan enkapsulasi kemampuan model untuk mengubah input dari konsep tingkat yang lebih rendah ke deskriptor tingkat yang lebih tinggi. Contohnya dari piksel ke kategori gambar. Kelebihan MobileNetV2 ini adalah prosesnya lebih cepat dibandingkan dengan MobileNetV1, ini dikarenakan MobileNetV2 mempunyai dua layer, sedangkan MobileNetV1 mempunyai satu layer. MobileNetV2 menggunakan operasi 2 kali lebih sedikit dan memiliki akurasi yang lebih tinggi, membutuhkan parameter sebesar 30 persen lebih sedikit dan sekitar 30 - 40 persen lebih cepat (Nufus, et al., 2021).

2.8 Transfer Learning

Transfer Learning merupakan metode terkini untuk mempersingkat waktu pada saat proses training data untuk CNN dan mendapatkan performa klasifikasi yang baik. Dikatakan mempersingkat waktu dalam proses *training* karena di metode *transfer learning* ini menggunakan *pre-trained* model (data yang sudah dilatih sebelumnya) guna menyelesaikan masalah yang serupa. Cara penggunaanya adalah menjadikan *pre-trained* model tersebut sebagai *starting point*, lalu dimodifikasi dan di*update* parameternya sesuai dengan kasus permasalahan yang baru. *Transfer learning* juga bisa dijadikan sebuah solusi apabila dataset yang ada masih kurang, maka dari itu untuk mendukung kekurangannya bisa mencari model yang sudah di train oleh peneliti atau pengembang lainnya yang mempunyai kasus serupa dengan masalah yang baru.

Secara umum ada dua tipe *transfer learning* di dalam ranah *deep learning* yaitu *transfer learning* untuk ekstraksi fitur dan *transfer learning* dengan *fine-tune* (Darmatasia, 2020). Apabila digunakan untuk ekstraksi fitur, maka pada saat di awal, *classifier* akan dilatih pada layer paling atas dari *pre-trained* model. *Layer* awal pada *pre-trained* model biasanya digunakan untuk mengekstrak fitur umum seperti citra garis tepi dengan tujuan agar tidak melatih ulang seluruh model, hal ini

dapat menghemat waktu. Berikut adalah ilustrasi proses *ekstraksi fitur* atau istilah lainnya *fixed feature extractor*.



Gambar 2.16 Ilustrasi Proses Ekstraksi Fitur (Adam, 2021)

Pembuatan *pre-trained* model diilustrasikan pada gambar di sebelah kiri. Model CNN biasanya terdiri dari *Convolutional Layer* (hijau) dan *Fully Connected Layer* (merah). Kemudian pada ilustrasi sebelah kanan, lapisan yang digunakan hanya lapisan konvolusinya saja untuk dibekukan (*freezed layer*), sehingga hasil keluarannya masih berupa vektor.

Kemudian pada transfer learning pada *fine-tune* akan dilakukan sebuah tuning parameter dengan klasifikasi yang baru ditambahkan. Tujuan dari *fine-tune* ini untuk menyesuaikan fitur spesifik berdasarkan data yang diperoleh sehingga proses *learning* menjadi lebih akurat dan cepat. Pada gambar 2.16 dimana gambar tersebut mengilustrasikan proses ekstraksi fitur atau *extraction features*. Perbedaan dari ilustrasi tersebut dengan *fine tuning* adalah pada proses *fine tuning* lapisan konvolusinya tidak dibekukan, akan tetapi akan dilatih ulang sesuai dengan kebutuhan kasus yang baru. Sebagai contoh misal didapatkan *pre-trained* model yang bisa membedakan antara mobil dan truk, kemudian dari model tersebut akan digunakan untuk mendeteksi jenis kendaraan lain seperti motor, sepeda, becak, dan lain-lain. Dari model sebelumnya, ada pengetahuan yang bisa dimanfaatkan untuk proses mendeteksi kasus yang baru, seperti model tersebut sudah bisa memahami mana objek mana latar belakang, mana pohon mana yang bukan, dan sebagainya. Dari pengetahuan tersebut nantinya akan dilatih lagi untuk memahami jenis-jenis objek pada kasus yang baru.

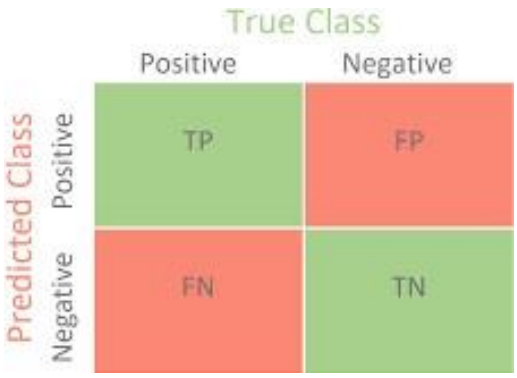
2.9 Python

Python merupakan bahasa pemrograman yang bersifat *open source* yang dibuat dan dikembangangkan oleh Guido Van Rossum pada tahun 1990. Nama python diambil dari acara televisi yang cukup terkenal bernama Mothy Python Flying Circus, dimana acara tersebut merupakan acara sirkus favorit Guido Van Rossum. Lima tahun kemudian python dikembangkan lagi oleh Guido Van Rossum agar lebih kompatible pada teknologi di jamannya. Sampai pada tahun 2000, terdapat pembaharuan versi python sampai versi 3 hingga sekarang. Python adalah salah satu bahasa yang dapat mengeksekusi beberapa instruksi multi guna secara langsung (interperatif) dengan basis orientasi objek (Objek Oriented Programming).

Python juga dikenal sebagai bahasa yang mempunyai kemampuan menggabungkan kapabilitas dan sintaksis kode yang jelas dilengkapi dengan bermacam - macam *library* yang mempunyai fungsionalitas masing-masing. Bahasa pemrograman python tergolong sebagai bahasa pemrograman tingkat tinggi, walaupun begitu python dirancang dengan sedemikian rupa sehingga sintaks kode masih menjadi lebih mudah dipahami dan dipelajari. Beberapa fitur-fitur python yang menarik diantaranya adalah python memiliki sistem pengelolaan data dan memori otomatis, modul pada python selalu di-*update*, mempunyai banyak sekali *library* yang mempunyai fungsionalitas khusus guna membantu pengembang dalam membuat suatu program, dan lain-lain (Baktikominfo, 2019). Python banyak diaplikasikan di berbagai sistem operasi terkenal seperti Linux, Windows, Mac OS, Android, Symbian OS, Palm, Amiga, dan lain-lain. Untuk pengembangan di ranah *machine learning*, kemudian *computer vision*, python menyediakan library yang membantu di ranah tersebut, diantaranya adalah OpenCV, Keras, Matplotlib, Tensorflow, PyTorch, Scikit-learn, Numpy, Pandas, dan lain-lain.

2.10 Confusion Matrix

Confusion matrix ini fungsinya untuk menunjukkan nilai akurasi, jumlah keberhasilan dan kegagalan model dalam menebak citra. Dibawah ini adalah ilustrasi dan penjelasan dari *confusion matrix*.



Gambar 2.17 Confusion Matrix

Pada *confusion matrix* ini apabila kategori atau kelas output pada model hanya mempunyai 2, maka *confusion matrix* hanya mempunyai 2 predikat yakni *positive* dan *negative* saja. Namun jika lebih dari 2 kelas atau kategori, maka di dalam confusion matrix akan mempunyai beberapa predikat seperti *True Positive*, *True Negative*, *False Positive*, *False Negative*. Berikut adalah penjelasan rinciannya.

1. *True Positive* : Total prediksi model yang dimana citra kelas positif diprediksi positif.
2. *True Negatife* : Total prediksi model yang dimana citra kelas negatif diprediksi negatif.
3. *False Positive* : Total prediksi model yang dimana citra kelas negatif salah diprediksi menjadi positif.
4. *False Negative* : Total prediksi model yang dimana citra kelas positif salah diprediksi menjadi negatif.

Selanjutnya untuk perhitungan akurasi menggunakan confusion matrix mempunyai sebuah rumus persamaan, persamaan tersebut dapat dilihat pada persamaan (2-10).

$$Akurasi = \frac{TP + TN}{TP+TN+FP+FN} \times 100\% \qquad (2\text{-}10)$$

Keterangan :

TP : *True Positive*

TN : *True Negative*

FP : *False Positve*

FN : *False Negative*

2.11 Android

Android adalah salah satu sistem operasi yang berbasis mobile yang populer digunakan pada saat ini, terutama pada perangkat ponsel pintar atau tablet. Android dirancang oleh Google berbasis kernel linux untuk mendukung kinerja perangkat layar sentuh. Sehingga penggunaan android saat ini yang tertanam di beberapa perangkat, semuanya berinteraksi dengan sentuhan, gesekan atau ketukan jari tangan pada layar perangkat tersebut. Android pertama kali dikenalkan pada tahun 2007 dengan beberapa versi. Google menamai versi android dengan nama-nama *desserts* (makanan penutup) seperti Cupcake, Donut, Éclair, Froyo, Gingerbread, dan lain-lain sampai yang terakhir android versi Pie pada tahun 2018. Pada tahun selanjutnya google menamai versi android menggunakan angka, android 10 pada tahun 2019, kemudian android 11 pada tahun 2020 sampai penelitian ini ditulis android 11 masih menjadi versi android yang terbaru.

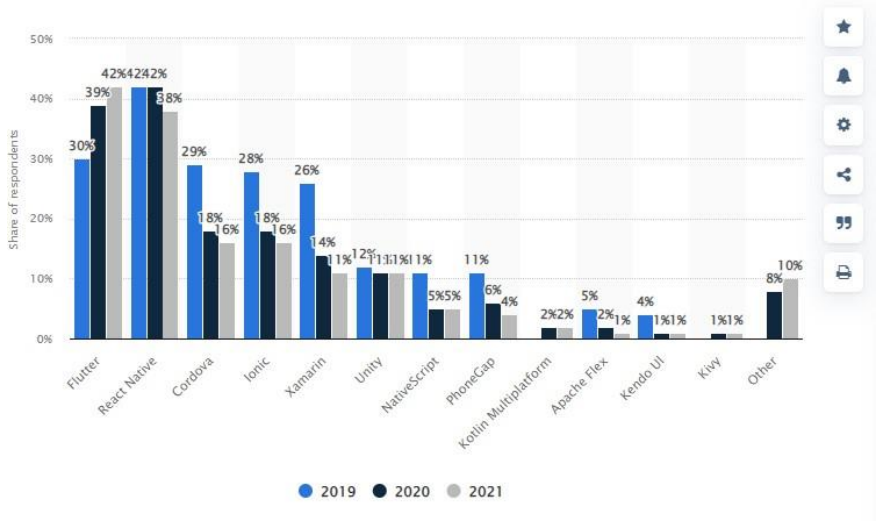
Pada tahun 2010, android mengalami masa kejayaannya, dimana untuk pertama kalinya android mengambil alih pangsa pasar yang sebelumnya dikuasai Apple dan iPhone, dan masih terus berlanjut hingga sekarang. Salah satu alasannya yang juga menjadi kelebihan dari android adalah jumlah ketersediaan *smartphone* dan harganya. Berbeda dengan iPhone yang harus menggunakan gawai khusus dan ketersediaannya terbatas satu merk saja, sehingga iPhone masih terkategori sebagai barang yang eksklusif. Selain ketersediaan gawai dan harga, kelebihan lain dari android adalah sistem operasinya bersifat *open source*, yang dimana dengan sifat *open source* ini banyak pengembang yang dimudahkan untuk mengembangkan dan mengoptimalkan sistem operasi android yang mereka tanam pada *smartphone* buatan mereka.

2.12 Flutter

Flutter merupakan sebuah *platform* yang digunakan oleh pengembang untuk membuat aplikasi *multiplatform* dengan satu basis *coding* (codebase). Dengan begitu, aplikasi yang dihasilkan menggunakan flutter dapat diaplikasikan di berbagai platform seperti android, IOS, web, bahkan desktop. Flutter ini bersifat open source buatan Google. Basis dari platform flutter ini menggunakan bahasa pemrograman dart sebagai front-endnya, berbeda dari *framework front-end* yang pada umumnya menggunakan javascript. Bahasa pemrograman dart merupakan

buatan Google pada tahun 2011 lalu dan masih dikembangkan hingga kini sehingga popularitasnya naik selama beberapa tahun terakhir. Flutter memiliki dua komponen penting yaitu SDK (Software Development Kit) dan *Framework UI (User Interface)*. SDK sendiri merupakan kumpulan beberapa tools yang berfungsi sebagai hasil dari aplikasi bisa dijalankan di multiplatform. Sedangkan Framework UI merupakan komponen UI yang berisikan teks, navigasi, tombol, teks input, tombol, dan lain-lain yang semuanya dapat di kustomisasi sesuai kebutuhan.

Flutter resmi diluncurkan pada bulan Desember tahun 2018. Flutter pada awalnya mulai dikembangkan oleh Google dari tahun 2015. Kemudian pada tahun 2019 popularitas flutter kian meningkat dan banyak para pengembang akhirnya berpindah menggunakan flutter sebagai platform membuat aplikasi android salah satunya. Berikut adalah statistik pengguna flutter pada tahun 2019-2021.



Gambar 2.18 Statistik pengguna flutter (Liu, 2021)

Pada grafik diatas menunjukkan bahwa flutter setiap tahunnya mengalami kenaikan jumlah pengguna dari tahun 2019 sampai 2021 yakni 30% pada tahun 2019, kemudian naik 9% menjadi 39% pada tahun 2020, lalu naik lagi sebesar 3% menjadi 42% di tahun 2021. Statistik juga menunjukkan bahwa selain flutter ada React Native besutan facebook (sekarang meta) yang tidak kalah populer dari flutter, karena keduanya memiliki fungsi yang sama yaitu menciptakan aplikasi multiplatform. Akan tetapi react native mengalami penurunan user di setiap tahunnya, ada kemungkinan bahwa user react bermigrasi ke flutter.

2.13 Google Colab

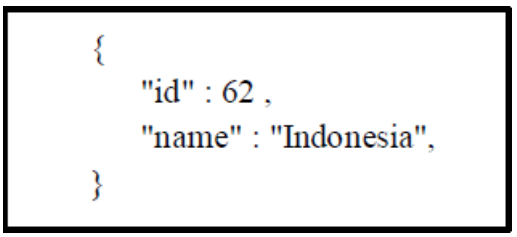
Google Colab atau nama lainnya Google *Colaboratory* merupakan sebuah program *executable document* yang bisa digunakan untuk menulis, menyimpan, menjalankan, serta membagikan program yang nantinya akan disimpan di google drive. *Software* ini dapat diakses secara gratis yang dijalankan di browser, bisa menggunakan Mozilla Firefox dan Google Chrome (Oliver, 2021). Google Colab memungkinkan penggunanya untuk bisa menulis, menyimpan, dan menjalankan program python secara mudah di *browser*. Dikatakan mudah karena pada google colab sudah disediakan *setting*, *adjustment*, *library* yang sudah terinstall dan penyimpanannya berbasis cloud, sehingga pengembang hanya perlu meng-*import library*-nya saja tanpa perlu setup atau *install* terlebih dahulu. Sehingga ini sangat memberikan kemudahan bagi para pengembang yang mempunyai spek komputer *standard*, pengembang hanya perlu koneksi internet yang stabil saja. Ibarat google colab seperti google sedang meminjamkan komputernya yang sudah berisikan berbagai keperluan untuk coding python disana, sehingga para pengembang tinggal menggunakannya saja untuk keperluan membuat program python.

Google colab juga menyediakan fitur untuk menghubungkan ke jupyter notebook pada komputer pengembang (*local runtime*), dan bisa juga menghubungkannya ke akun github pribadi pengembang. Kemudian tak lupa google colab sesuai namanya google *colaboratory* (kolaborasi) juga bisa dimanfaatkan oleh pengembang untuk bisa saling kolaborasi atau share coding dari satu pengembang ke pengembang lain. Hal ini juga memudahkan pengembang untuk bekerja sama secara tim dalam membangun atau membuat sebuah program.

2.14 JSON

JSON atau *JavaScript Object Notation* merupakan format pertukaran data ringan yang memudahkan manusia untuk dibaca dan ditulis, serta mudah untuk di-*parse* (terjemahkan) dan di-*generate* (dibuat) oleh komputer. (Sahrial, et al., 2022) Penulisan pada format JSON tidak tergantung pada satu jenis bahasa pemrograman saja, sehingga ini memudahkan penggunaan JSON sebagai bahasa pertukaran data antar bahasa pemrograman. Struktur datanya bersifat *universal* yang meliputi kumpulan nilai (*value*) atau *object* dan larik (*array*).

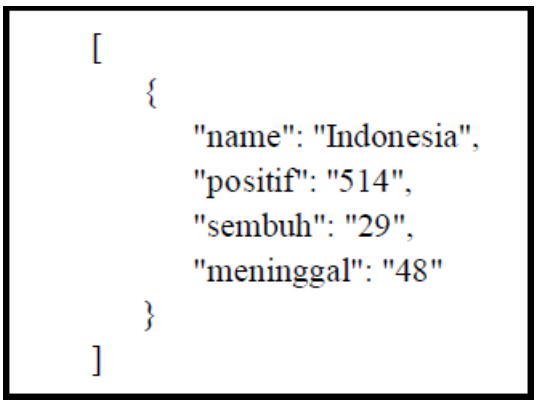
Objek pada JSON dituliskan dengan simbol { diakhiri dengan simbol }. Contoh penulisan struktur objek pada JSON dalam kode seperti gambar 2.19 di bawah ini.



```
{
    "id" : 62 ,
    "name" : "Indonesia",
}
```

Gambar 2.19 Struktur penulisan objek JSON

Objek diatas biasanya dimuat dalam larik atau array. *Array* merupakan data yang terurutkan dimulai dengan simbol [kemudian diakhiri dengan simbol], serta koma (,) untuk memisahkan setiap *value*. Contoh penulisannya seperti gambar 2.19 di bawah ini.



```
[
    {
        "name": "Indonesia",
        "positif": "514",
        "sembuh": "29",
        "meninggal": "48"
    }
]
```

Gambar 2.20 Struktur penulisan objek JSON di dalam *array*

2.15 Black Box Testing

Black Box atau *Black Box* testing dapat disebut juga *Behavioral Testing* adalah pengujian yang dilakukan untuk mengamati apakah hasil input dan output dari sebuah perangkat lunak dapat berjalan dengan baik tanpa mengetahui struktur kode (Setiawan, 2021). Dalam melakukan pengujian, penguji tidak harus mengetahui kode program suatu perangkat lunak, sehingga pengujian dapat dilakukan oleh siapa saja. Ada beberapa jenis pada Black Box yang biasanya digunakan untuk menguji perangkat lunak. Dibawah ini ada 3 jenis pengujian Black Box Testing, yakni:

1. *Functional* Testing

   Functional testing adalah jenis proses pengujian pada fitur spesifik pada *software*. Tujuannya adalah untuk memastikan bahwa aplikasi berjalan sesuai yang diharapkan pengembang.

2. *Non-functional* Testing

   Non-functional testing ini dilakukan untuk pengujian yang dilakukan dengan aspek tambahan non-functional. Tujuan dari testing ini dilakukan untuk mengetahui bagaimana *software* mampu menjalankan suatu tugas.

3. *Regression* Testing

   Jenis Regression Testing ini dilakukan untuk menguji apakah terjadi suatu kemunduran atau regresi saat aplikasi di-*upgrade*. Contohnya seperti pengujian performa aplikasi sudah di-*upgrade*, apakah performa aplikasinya menurun atau meningkat.

2.16 Skala *Likert*

Skala *likert* merupakan suatu skala psikometrik yang banyak digunakan dalam pengisian angket dan juga merupakan skala yang sering digunakan untuk kebutuhan riset berupa survey. Pada penggunaan skala ini disediakan pernyataan kepada responden, kemudian responden menentukan keputusan tingkat persetujuan mereka terhadap pernyataan yang telah diberikan dengan memilih salah satu dari pilihan yang tersedia. Skala likert biasanya disediakan lima pilihan dengan format seperti sangat setuju, setuju, kurang setuju, tidak setuju, sangat tidak setuju. Tujuan dari skala *likert* adalah untuk mengevaluasi suatu program guna mengukur persepsi, pendapat, dan sikap seseorang atau sekelompok orang (Ukkas, 2017). Berikut adalah contoh tabel dari skala *likert*.

Tabel 2.1 Contoh Angket Skala Likert

Pernyataan	STS	TS	KS	S	SS
UI aplikasi menarik				√	
Aplikasi mudah digunakan					√

(Keterangan : STS = Sangat Tidak Setuju, TS = Tidak Setuju, KS = Kurang Setuju, S = Setuju, SS = Sangat Setuju)

2.17 Penelitian Terdahulu

Terdapat penelitian sebelumnya yang mengimplementasikan CNN untuk mendeteksi tanaman maupun objek lain. Penelitian ini mempunyai rentang waktu lima tahun terakhir. Penelitian pertama diambil dari penelitian yang dilakukan oleh (Ilahiyah & Nilogiri, 2018). Pada penelitian ini, para peneliti ingin mengimplementasikan deep learning untuk identifikasi jenis tumbuhan berdasarkan citra daun menggunakan *Convolutional Neural Network* (CNN). Hasil dari penelitian ini adalah fitur terbaik diperoleh dari arsitektur Efficient B3 dengan nilai akurasi pelatihan yang tinggi dan akurasi yang rendah sebesar 99% dan 0,012. Kemudian akurasi pelatihan yang lebih rendah didapatkan dari arsitektur MobileNet V3 sebesar 57% dengan loss 0.007. Kedua arsitektur tersebut menggunakan metode transfer learning untuk model pelatihannya. Kedua arsitektur diuji dengan citra yang tidak dilatihkan sebelumnya untuk mengklasifikasi penyakit padi *brown spot* dan *bacterial leaf* secara acak. Hasil menunjukkan bahwa arsitektur Efficient B3 lebih tinggi daripada arsitektur MobileNet V3 yaitu sebesar 79,53%, ini membuktikan bahwa arsitektur Efficient B3 lebih baik dalam mengklasifikasikan penyakit padi dengan citra daun padi daripada arsitektur MobileNet V3.

Penelitian kedua dilakukan oleh (Anggiratih, et al., 2021). Para peneliti tersebut ingin mendeteksi jenis penyakit tanaman padi berdasarkan citra daun menggunakan model CNN arsitektur Efficientnet B3 dengan *transfer learning*. Penelitian ini memanfaatkan dataset dari peneliti lain yang berjumlah 857 citra penyakit padi yang terdiri dari *brown spot* (bercak coklat) dan *bacterial leaf* (hawar daun). Citra diambil menggunakan kamera digital SLR NIKON D90 di desa Sherta, India. Semua citra berukuran 2848x4288 dengan latar belakang putih dan diambil ketika waktu siang hari. Hasil dari penelitian ini adalah fitur terbaik diperoleh dari arsitektur Efficient B3 dengan nilai akurasi pelatihan yang tinggi dan akurasi yang rendah sebesar 99% dan 0,012. Kemudian akurasi pelatihan yang lebih rendah didapatkan dari arsitektur MobileNet V3 sebesar 57% dengan loss 0.007 seperti yang ditunjukkan pada gambar 2.17. Kedua arsitektur tersebut menggunakan metode transfer learning untuk model pelatihannya. Kedua arsitektur diuji dengan citra yang tidak dilatihkan sebelumnya untuk mengklasifikasi penyakit padi *brown*

spot dan *bacterial leaf* secara acak. Hasil menunjukkan bahwa arsitektur Efficient B3 lebih tinggi daripada arsitektur MobileNet V3 yaitu sebesar 79,53%, ini membuktikan bahwa arsitektur Efficient B3 lebih baik dalam mengklasifikasikan penyakit padi dengan citra daun padi daripada arsitektur MobileNet V3.

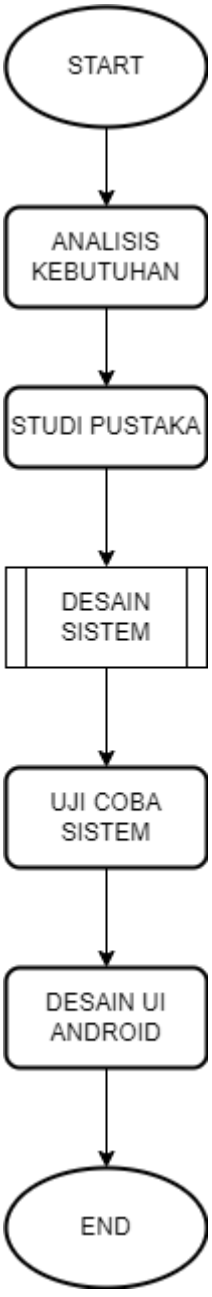
Penelitian yang ketiga dilakukan oleh (Rochman & Junaedi, 2020). Penelitian ini ingin mengklasifikasikan ordo tumbuhan menggunakan citra daun sebagai input. Metode yang digunakan adalah menggunakan *transfer learning* pada *convolutional neural network*. Keluaran dari penelitian ini adalah sistem dapat mengklasifikasikan dalam delapan kelas ordo tumbuhan. Delapan kelas ordo diantaranya adalah Alismatales, Asparagales, Cucurbitales. Poales, Ranuncucales, Sapindales, Solanales, dan Zingiberales. Hasil dari penelitian ini adalah arsitektur model ResNet50 dan Inception-V3 memiliki akurasi sebesar 85% untuk mengklasifikasi ordo tanaman. Dengan nilai akurasi tersebut sudah bisa dikatakan cukup besar, ditambah nilai akurasi tersebut muncul di iterasi kurang dari 10 dan stabil diantara 83% - 88%. Sedangkan arsitektur model VGG16 memiliki nilai akurasi dibawah 50% di setiap iterasi. Maka dapat disimpulkan bahwa arsitektur model ResNet50 dan Inception-V3 performanya lebih baik daripada VGG16 untuk mengklasifikasi ordo tumbuhan pada penelitian ini.

Bab III

Analisis dan Perancangan Sistem

3.1 Alur Penelitian

Penelitian ini ingin melakukan identifikasi tanaman hias menggunakan metode *Convolutional Neural Network* dengan citra daun. Penelitian ini mempunyai beberapa tahapan seperti gambar 3.1 di bawah ini.



Gambar 3.1 Diagram Alir Penelitian

3.2 Analisis Kebutuhan

Penelitian ini dikerjakan menggunakan hardware laptop Dell Latitude 3510 dengan spesifikasi RAM 16GB, *Processor* Intel Core i5. Dalam pengembangannya, penelitian ini memfokuskan kepada kinerja aplikasi sesuai dengan yang diharapkan yakni mengidentifikasi tanaman hias. Berikut adalah beberapa poin-poinnya:

1. Identitas dari aplikasi agar dapat dikenali sesuai dengan fungsinya
2. Terdapat halaman petunjuk cara menggunakan aplikasi
3. Pada halaman kamera aplikasi terdapat kotak beserta petunjuk untuk memposisikan objek tanaman hias yang akan diidentifikasi
4. Hasil dari identifikasi akan berupa nama umum tanaman hias di Indonesia, nama latin, suhu, periode penyiraman, nilai *confidence*, deskripsi, manfaat, bahaya, dan persentase nilai *confidence* aplikasi dalam mendeteksi tanaman hias.
5. Terdapat fitur untuk mencari informasi lainnya apabila hasil dari aplikasi dirasa kurang lengkap
6. Proses pengenalan tidak membutuhkan koneksi internet jadi *user* dapat menggunakannya kapan saja tanpa menggunakan paket data internet. Semua data mengenai tanaman hias sudah disimpan di aplikasi tersebut.

3.3 Studi Pustaka

Pada tahap studi pustaka ini, terdapat beberapa hal yang menjadi fokus utama yaitu metode *deep learning*, *Convolutional Neural Network*, *transfer learning*, dan *platform* Flutter. Semuanya merupakan topik yang paling penting dalam perancangan, pembuatan aplikasi pada penelitian ini. Materi-materi dari topik tersebut bersumber dari berbagai jurnal, *website*, forum pengembang, youtube, buku, dan lain-lain.

3.4 Pengumpulan Dataset

Dataset yang digunakan dalam penelitian ini bersumber dari gambar di internet dan pengambilan foto menggunakan kamera *smartphone* Infinix Smart 3 Plus, Model X627 dengan kamera belakang 13 *MegaPixels*. Pengambilan gambar dataset pada penelitian ini difokuskan ke objek daun dengan intensitas cahaya dari

500 - 700 lux, background bebas dan dengan jarak kurang lebih 30 - 50 cm dari kamera, contoh seperti gambar 3.2 berikut.

Gambar 3.2 Contoh gambar dataset, daun lidah mertua

Selanjutnya untuk mempermudahkan penelitian dalam hal pengujian secara langsung, penelitian ini menggunakan jenis tanaman hias yang berada di sekitar lingkungan penelitian. Total ada 10 kategori atau kelas tanaman hias dan di setiap kelasnya berisikan 30 citra. Berikut adalah 10 kategori tanaman hias pada penelitian ini, yaitu:

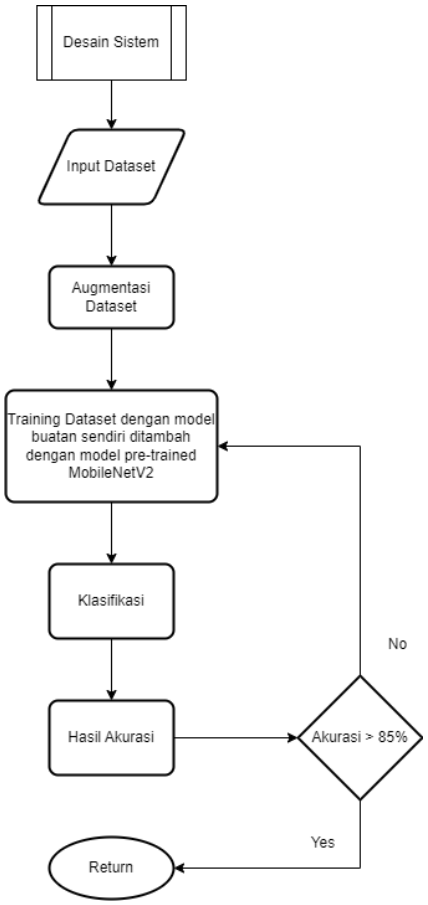
1. Andong (Cordyline fruticosa)
2. Kornus (Cornus alba)
3. Krokot Epah Daun Merah (Alternanthera brasiliana)
4. Lidah Buaya (Aloe vera)
5. Lidah Mertua (Sansevieria trifasciata)
6. Miana (Coleus scutellarioides)
7. Pucuk Merah (Syzygium oleana)
8. Puring (Codiaeum variegatum)
9. Sri Rejeki (Agloenema commutatum)
10. Tidak Diketahui (*Random Images*)

Struktur dataset pada penelitian ini dibagi menjadi dua folder, yang pertama *folder train* dan yang kedua *folder validation*. Untuk *folder train* berisikan 10

kategori tanaman hias yang sudah dipaparkan diatas. Setiap kelas atau kategori berisikan 30 citra yang bersumber dari internet atau dari pengambilan foto menggunakan *smartphone*. Sedangkan *folder validation* berisikan 5 citra tanaman hias per kategori sehingga jika ditotal ada sebanyak 50 citra pada *folder* data *validation.*

3.5 Desain Sistem

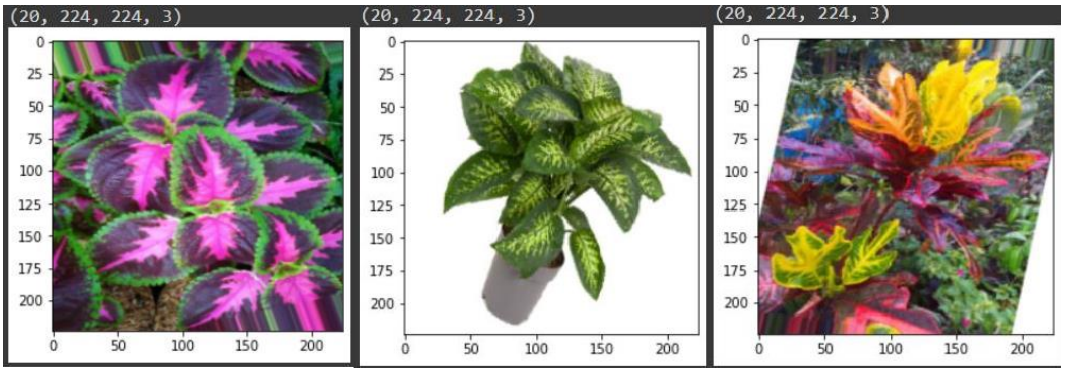
Pada desain sistem penelitian ini dilakukan proses menambahkan *knowledge* baru atau *transfer learning* menggunakan model MobilenetV2 dari keras (https://keras.io/api/applications/mobilenet/) dengan parameter *pre-trained* bobot dari ImageNet. ImageNet ini merupakan dataset yang berisikan 1.000 *class*, di setiap kelas berisikan 1.200 gambar, total 1.200.000 gambar. Dataset tersebut bisa digunakan untuk berbagai kebutuhan, seperti prediksi, *feature extraction*, dan *fine tuning*. Berikut diagram alir desain sistem pada penelitian ini.



Gambar 3.3 Diagram alir Desain Sistem

3.5.1 Augmentasi Dataset

Pada tahap ini, citra dataset akan di proses dahulu sebelum memasuki proses *training*. Tahap augmentasi mempunyai banyak variasi pada proses memodifikasi citra, umumnya seperti menyamakan semua ukuran, menskala ulang, menggeser citra, memperbesar citra, mengacak citra, dan lain-lain. Pada penelitian ini yang digunakan adalah mengubah ukuran citra menjadi 224x224 piksel, kemudian dilakukan skala ulang per piksel pada citra, sehingga nilainya menjadi diantara rentang dari 0 sampai 1. Selanjutnya dilakukan proses menggeser citra, memperbesar citra (*zoom*) sebesar 0.2, terakhir membalik citra secara horizontal. Alasan mengubah ukuran citra menjadi 224 x 224 piksel adalah umumnya suatu inputan citra pada CNN dengan arsitektur MobileNetV2 menggunakan ukuran tersebut. Setelah itu citra akan dilakukan proses transformasi yang dimana hasil tersebut berupa matrix yang akan digunakan sebagai input untuk proses berikutnya. Untuk contoh hasil dari proses augmentasi ini ada pada gambar 3.4 di bawah ini.



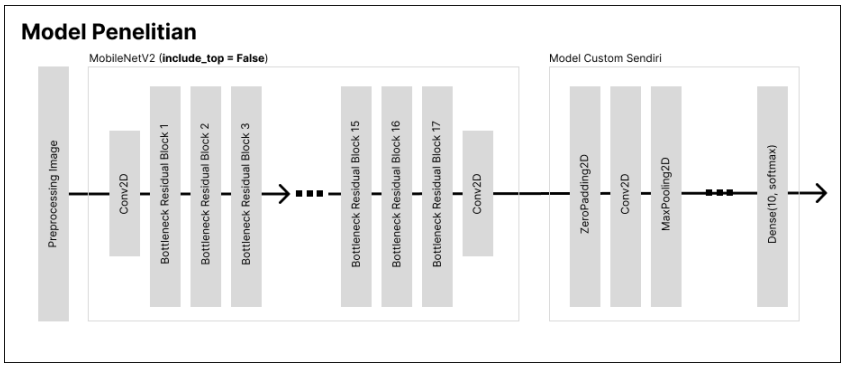
Gambar 3.4 Beberapa Hasil *Preprocessing Dataset*

3.5.2 Training Model

Tahap ini akan dilakukan proses training pada dataset yang sudah melalui augmentasi citra sebelumnya. Proses training dilakukan pada model MobileNetV2 yang sudah memiliki *weight* atau bobot dari dataset ImageNet yang berisikan 1000 kelas. Tapi sebelum dilakukan transfer learning, pada penelitian ini akan dilakukan eksperimen tambahan di awal untuk mengetahui besar akurasi model apabila dataset pada penelitian ini dilatih tanpa *transfer learning* atau dengan *full layer* buatan sendiri. Penelitian ini akan dilakukan *train* dengan 100 *epoch* per-*optimizers*. Ada 6 *optimizers* yang di uji coba pada penelitian ini yakni, Nadam, RMSProp, Adam, SGD, Adadelta, dan Adagrad. Untuk proses *transfer learning* model dari

MobileNetV2 akan ditambahkan dengan model *custom layer* buatan sendiri, berikut di bawah ini adalah strukturnya :

1. Membuat layer *input* dengan ukuran gambar 224x224 RGB
2. Melakukan proses augmentasi pada citra input seperti normalisasi layer input atau *Rescale* dengan 1./255, *vertical flip*, *zoom range*, *validation split=0.2*, data dan lain-lain
3. Memakai semua *layer* MobileNetV2 kecuali layer *output* yang sudah dilatih dan tidak akan dilatih ulang nantinya (*trainable = False*)
4. Menambahkan beberapa layer *custom* buatan sendiri dengan mencoba beberapa variasi layer, untuk custom layer pada penelitian ini ditambahkan layer seperti di bawah ini beserta ilustrasinya,
    a. *ZeroPadding2D(padding=2,2)*
    b. *Conv2D 64, filter(3x3), strides=3,3, relu*
    c. *MaxPooling2D(pool_size=(3,3))*
    d. *ZeroPadding2D(padding=2,2)*
    e. *Conv2D 128, filter(3x3),strides=(2,2), relu*
    f. *MaxPooling2D(pool_size=(2,2))*
    g. *ZeroPadding2D(padding=1,1)*
    h. *Conv2D 256, filter(3x3), relu*
    i. *Flatten()*
    j. *Dropout(0.2)*
    k. *Dense(512), relu*
    l. *Dense(10), softmax*



Gambar 3.5 Ilustrasi Model Penelitian

Tujuan dari penambahan atau *custom layer* model tersebut adalah *knowledge* dari model MobileNetV2 akan ditambahkan lagi *knowledge* untuk mengatasi kasus baru, yakni pada penelitian ini adalah mengidentifikasi tanaman hias. Penambahan *layer* tersebut ditambahkan setelah layer *output* pada *layer*

bawaan MobileNetV2. Setelah proses penambahan *layer*, akan dilakukan proses *fine tuning*, *Fine tuning* ini adalah proses *training* menggunakan dataset kasus baru yakni tanaman hias. Kemudian ada beberapa layer dari *pre-trained* model MobileNetV2 yang harus dibekukan agar pengetahuan dari dataset ImageNet sebelumnya tidak hilang dan masih ada. Layer yang dibekukan tidak akan dilakukan proses training, sehingga weight dari bawaan model masih tetap terjaga. Jadi dengan kata lain pengetahuan dari ImageNet akan digabung dengan pengetahuan baru untuk kasus identifikasi tanaman hias pada penelitian ini. Tidak ada patokan khusus untuk jumlah pembekuan layer pada proses *fine tuning* tersebut.

3.6 Uji Coba (Akurasi)

Setelah proses *training* selesai, model dari MobileNetV2 tersebut akan digunakan untuk uji coba data *validation*. Awalnya sebagai percobaan model akan diuji dengan citra *validation* yang sudah tersedia sebelumnya. Apabila dalam uji data *validation* model menghasilkan nilai akurasi yang tinggi, maka model akan dicoba juga untuk mengidentifikasi menggunakan citra lain dari dataset. Namun jika ternyata hasil dari model menghasilkan nilai akurasi yang rendah, maka harus ada perbaikan pada model tersebut sampai menemukan nilai akurasi yang tinggi. Maka dari itu dibutuhkan alat untuk melihat tingkat akurasi model. *Library* yang digunakan untuk menghitung nilai akurasi model pada penelitian ini adalah Sklearn dengan *tools*-nya yang bernama *confusion matrix*. *Confusion matrix* ini akan menunjukkan nilai akurasi, jumlah keberhasilan dan kegagalan model dalam menebak citra. Dibawah ini pada tabel 3.1 adalah ilustrasi dan penjelasan dari *confusion matrix* dari penelitian ini, yakni sebanyak 10 *class*.

Tabel 3.1 *Confusion Matrix*

5	0	0	0	0	0	0	0	0	0
0	5	0	0	0	0	0	0	0	0
0	0	5	0	0	0	0	0	0	0
0	0	0	3	0	0	0	0	1	1
0	0	0	0	5	0	0	0	0	0
0	0	0	0	0	4	1	0	0	0
0	0	0	0	0	0	5	0	0	0
0	0	0	0	1	0	0	4	0	0
0	0	0	0	0	0	0	0	4	1
0	0	0	0	0	0	0	0	1	4

Pada confusion *matrix* di atas ada 10 *output class* atau kategori dari model pada penelitian ini. Cara membaca dari hasil *Confusion Matrix* tersebut adalah kolom dari yang paling kiri menunjukkan class yang ke 1 sampai ke 10. Begitupun sebaliknya, untuk baris, baris dari yang paling atas sampai bawah merupakan class yang ke 1 sampai ke 10. Misalnya pada baris pertama kolom pertama merupakan class yang ke 1 menunjukkan hasil sebanyak 5, itu artinya ke 5 citra yang diujikan semuanya menunjukkan hasil atau *output class* 1 yang berarti prediksinya benar semua. Kemudian baris ke 4 kolom ke 4 yang merupakan *class* ke 4, hasilnya menunjukkan dari 5 citra yang diuji hanya 3 citra yang menunjukkan class ke 4, sisanya terindentifikasi di *class* 9 dan 10.

3.7 Menyimpan Model

Setelah melakukan uji coba dan akurasi sudah cukup baik, maka hasil model yang sudah dilatih sebelumnya akan disimpan menggunakan fitur dari tensorflow menjadi *file* model berekstensi .tflite. *File* berekstensi .tflite inilah yang nantinya akan di impor ke dalam *project* android yakni menggunakan Flutter. Pada *project* Flutter nantinya akan membutuhkan *library* tensorflow juga, yaitu tflite. Pada *project* android di penelitian ini juga dipasang sebuah JSON untuk menyimpan nama-nama tanaman hias, manfaat, dan bahayanya. Jadi aplikasi android pada penelitian ini nantinya akan menerima input parameter berupa gambar yang diambil oleh *user* menggunakan kamera, kemudian akan di klasifikasikan menggunakan model yang sudah diimpor sebelumnya lalu akan muncul data-datanya yang diambil dari JSON.

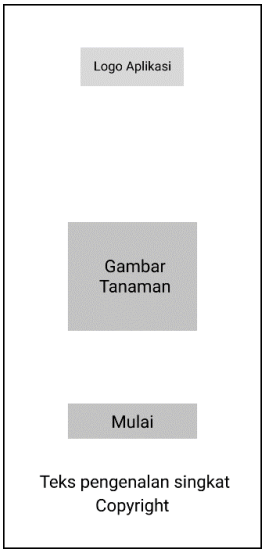
3.8 Desain UI Aplikasi Android

Pengaplikasian model pada penelitian ini adalah menggunakan perangkat android. Maka dari itu pada tahap ini akan dilakukan pembuatan aplikasi android. Sebelum membuat aplikasi android, diperlukan rancangan desain tampilan antarmuka (*User Interface*) yang nantinya akan diaplikasikan dalam proses pembuatan. Rancangan antarmuka aplikasi android untuk penelitian ini mempunyai 3 halaman utama dan 1 halaman splash screen atau halaman yang muncul saat

pertama kali aplikasi android dijalankan. Berikut ini adalah rincian dari desain antarmuka aplikasi android.

Gambar 3.6 *Splash Screen*

Pada gambar 3.6 tersebut merupakan halaman splash screen yang akan muncul beberapa detik ketika pertama kali aplikasi dijalankan. Pada halaman tersebut hanya mengandung logo aplikasi beserta *background*.



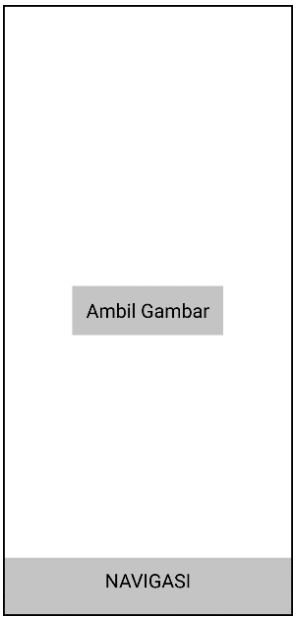
Gambar 3.7 *Intro Page*

Pada gambar 3.8 merupakan halaman intro atau pengenalan aplikasi disaat user pertama kali membuka aplikasi setelah *splash page*. Halaman ini bertujuan untuk

melakukan pengenalan singkat mengenai aplikasi penelitian ini atau Plantis (*Plant Identification System*) serta *copyright* pengembang (*developer*).

Gambar 3.8 Halaman Utama

Pada gambar 3.8 diatas merupakan halaman utama ketika *user* menekan tombol mulai pada halaman *intro*. Pada halaman ini berisikan ilustrasi beserta penjelasan mengenai cara penggunaan aplikasi tersebut. Dan juga ada navigasi di bagian bawah aplikasi berisikan tombol untuk menuju ke halaman-halaman lainnya.



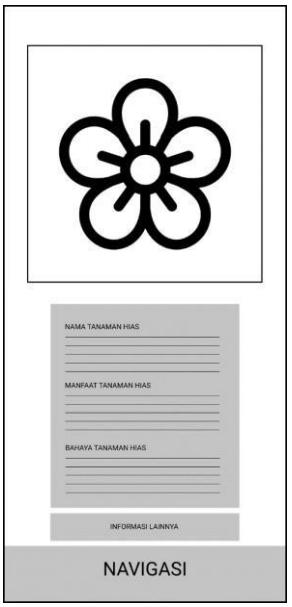
Gambar 3.9 Halaman penghubung menuju halaman kamera

Pada halaman ini atau seperti yang ada di gambar 3.9 merupakan halaman penghubung menuju halaman pengambilan citra atau gambar yang langsung

terhubung dengan kamera *smartphone user*. Ada tombol ambil gambar yang fungsinya untuk mengarah ke halaman kamera.

Gambar 3.10 Halaman kamera

Pada halaman ini *user* akan terhubung dengan kamera *smartphone* bagian belakang yang dimana pada halaman tersebut tertera kotak dengan panduan agar dapat memposisikan kamera ke daun tanaman hias yang akan diidentifikasi bisa sesuai atau pas dengan kotak tersebut. Halaman ini mirip seperti halaman kamera *smartphone* biasa, setelah menekan tombol kamera di bagian bawah, sistem akan mengarahkan ke halaman hasil identifikasi.



Gambar 3.11 Halaman Hasil Identifikasi

Setelah citra diproses akan muncul hasil identifikasi pada halaman hasil identifikasi. Seperti gambar 3.9, pada halaman tersebut berisikan citra yang ditangkap kemudian dibawahnya ada hasil nama tanaman hias beserta deskripsi tanaman hias, manfaat tanaman hias, hasil *confidence* identifikasi citra daun, bahaya tanaman hias jika ada, dan tombol yang mengarah ke halaman pencarian google agar dapat mencari informasi lainnya mengenai tanaman tersebut.

3.9 Uji Black Box

Uji black box pada penelitian ini dirancang dengan beberapa poin untuk menguji fungsionalitas aplikasi sesuai dengan tujuan dan desain *User Interface*. Ada 8 poin yang akan diujikan yakni:

1. Aplikasi akan memunculkan halaman cepat atau splash dengan logo ‘Plantis’ saat pertama kali dibuka.
2. Aplikasi memunculkan halaman pengenalan atau intro dengan logo, teks, gambar background tanaman, teks penjelasan singkat, tombol mulai, dan *copyright* pengembang aplikasi.
3. Aplikasi memunculkan halaman guide dengan 4 buah halaman *carousel* yang bisa di geser ke kanan atau ke kiri, diakses dari tombol ‘Mulai’ pada halaman *intro*.
4. Aplikasi memunculkan halaman *Capture* dengan teks dan tombol ‘Start’ di tengah halaman yang diakses dari tombol navigasi bagian tengah atau kedua dengan *icon* kamera.
5. Aplikasi memunculkan tampilan kamera bagian belakang dengan box dan teks di tengah serta tombol kamera di bawah.
6. Aplikasi memunculkan hasil identifikasi dari halaman *camera* melalui tombol dengan *icon* kamera bagian bawah.
7. Aplikasi mengeluarkan hasil indentifikasi kurang dari 2 detik.
8. Aplikasi memunculkan halaman peramban pencarian google dengan kata kunci ‘manfaat dan bahaya tanaman x’ (x = nama tanaman hias).

Dari poin-poin yang dibuat di atas nantinya akan di uji apakah aplikasi mengeluarkan hasil sesuai yang diharapkan dari poin-poin di atas atau tidak.

3.10 Uji Coba Aplikasi Dengan Parameter *Confidence*

Pada aplikasi di penelitian ini ada nilai yang namanya *confidence*. Ini adalah nilai persentasi aplikasi dalam mengenali objek daun tanaman hias yang difoto. Fungsi nilai ini adalah apabila aplikasi mengeluarkan nilai *confidence* yang rendah maka hasil dari identifikasi akan langsung mengeluarkan *output* 'Tidak Diketahui' artinya aplikasi tidak mengenali apa objek yang difoto. Untuk itu perlu adanya uji coba berapa nilai *confidence* yang pas untuk aplikasi berhasil mengenail tanaman hias dengan benar. Secara teknis untuk melakukan uji ini adalah dengan merubah parameter nilai *confidence* pada *if else* yang ada di program. Misal jika nilai *confidence* kurang dari 70 persen maka hasil dari identifikasinya adalah tanaman tersebut 'Tidak Diketahui'. Selanjutnya ada 3 nilai *confidence* yang akan diuji yakni < 70%, < 80%, < 90%.

Cara untuk melakukan uji coba ini adalah mencoba memfoto semua jenis tanaman hias lalu dihitung berapa percobaan sampai menghasilkan *output* tanaman hias yang benar. Dari percobaan tersebut dihitung juga berapa jumlah aplikasi salah dalam melakukan identifikasi. Selanjutnya akan dinilai berapa persen kesalahan dari jumlah percobaan yang dilakukan di setiap nilai parameter *confidence* yang diujikan. Setelah itu dilihat nilai persentase kesalahan yang paling kecil dari ketiga nilai *confidence* tersebut, nilai *confidence* itulah yang akan menjadi parameter untuk aplikasi pada penelitian ini.

3.11 Uji Coba Aplikasi Skala Likert

Uji coba pada penelitian ini menggunakan skala likert. Pengujian dilakukan kepada 10 koresponden dengan cara mengisi kuisioner di google *form*. Tata cara melakukan uji aplikasi kepada user pada penelitian ini adalah dengan meminta koresponden atau *user* mengakses link *google form* lalu membaca petunjuk selanjutnya. Kemudian setelah itu koresponden akan diminta mengunduh aplikasi dan memasangnya dari gawai masing-masing. Setelah memasang aplikasi, koresponden akan diminta untuk mengoperasikan aplikasi sesuai *guide* yang ada pada aplikasi. Kemudian untuk percobaannya koresponden dapat memfoto salah satu dari 9 tanaman hias yang ada pada penilitian ini, objek dapat berupa tanaman asli jika punya, apabila tidak koresponden dapat memfoto gambar tanaman hias

pada layar *desktop* atau sejenisnya. Pada tahap pengoperasian aplikasi ini, koresponden diminta untuk mengingat berapa kali koresponden melakukan percobaan sampai aplikasi benar mengenali tanaman hias yang difoto. Setelah itu, koresponden diminta untuk mengisi kuisioner pada google *form*.

Koresponden untuk pengujian ini tidak bersyarat usia, semua orang yang mempunyai gawai android dan dapat mengoperasikannya dapat ikut untuk mengisi kuisioner pada penelitian ini.

BAB IV
Hasil dan Pembahasan

4.1 Hasil Eksperimen Model

Untuk hasil desain model pada penelitian ini terbagi menjadi beberapa eksperimen. Eksperimen dilakukan di *Google Colab* menggunakan bahasa python. Eksperimen ini dilakukan bertujuan untuk mengetahui model mana yang mempunyai nilai akurasi tertinggi yang nantinya akan dipakai untuk aplikasi android pada tahap selanjutnya. Berikut adalah hasil dari eksperimen yang dilakukan.

1. Eksperimen Pertama

Eksperimen ini membandingkan model tanpa transfer learning buatan sendiri dengan model MobileNetV2 yang masing-masing menggunakan *optimizer* Adam. Hasilnya seperti tabel di bawah ini.

Tabel 4.1 Hasil akurasi eksperimen pertama

No	Model	Optimizer	Akurasi
1	Tanpa Transfer Learning	Adam	70%
2	Tanpa Transfer Learning	RMSprop	66%
3	Tanpa Transfer Learning	Adagrad	40%
4	Transfer Learning	Adam	84%
5	Transfer Learning	RMSprop	88%
6	Transfer Learning	Adagrad	88%

Dari hasil tabel 4.1 diketahui bahwa nilai akurasi tertinggi tanpa *transfer learning* hanya sebesar 70% dengan optimizer Adam, sedangkan akurasi dengan *transfer learning* tertinggi sebesar 88% dengan optimizer RMSprop dan Adagrad. Perbedaan nilai akurasi ini dikarenakan model dengan *transfer learning* sudah membawa bobot dari model *MobileNetV2* yang sudah dilatih dengan ribuan gambar yang mempunyai 1000 *output* atau kelas dibuat oleh pengembang Imagenet sebelumnya. Setelah dari hasil akurasi tersebut maka penelitian ini akan dilanjutkan dengan eksperimen menggunakan *transfer learning* menggunakan model *MobileNetV2*.

2. Eksperimen Kedua

Eksperimen ini melakukan perbandingan nilai akurasi mengenai komposisi model *transfer learning* dari *layer* MobileNetV2 yang dipakai bersama layer buatan sendiri. Komposisi yang dimaksudkan adalah jumlah layer yang di-*freeze* dan tidak di-*freeze*. *Freeze layer* adalah *layer* yang tidak dilatih ulang bersama *layer* buatan sendiri. Sedangkan *layer* yang tidak di *freeze* akan dilatih ulang bersama dengan *layer* buatan sendiri. Total keseluruhan *layer* pada MobileNetV2 berjumlah 154 layer. Ada 3 macam komposisi dalam eksperimen ini yang akan diujikan dengan beberapa *optimizers* yakni komposisi 80% (123 *layer*) yang akan di-*freeze*, 90% (139 *layer*) yang akan di-*freeze*, dan 100% (154 *layer*) yang akan *di-freeze* dari *layer* MobileNetV2. Berikut adalah hasil dari eksperimen kedua.

Tabel 4.2 Hasil Akurasi eksperimen kedua

No	Optimizers	Komposisi Layer MobileNetV2		
		123 Layer	139 Layer	154 Layer
1	Adam	68%	18%	80%
2	RMSprop	60%	10%	88%
3	Adagrad	82%	72%	88%

Setelah melihat hasil akurasi dari tiga *optimizers* pertama, terlihat bahwa hasil akurasi dari ketiga komposisi tersebut nilai tertingginya ada pada kompisi 100%. Maka keputusannya untuk penelitian ini menggunakan komposisi transfer learning dengan 100% *layer* dari MobileNetV2 ditambah dengan layer buatan sendiri.

3. Eksperimen Ketiga

Eksperimen ini membandingkan model dengan *transfer learning* mengunakan 6 *optimizers* yang berbeda. Setiap proses latih dilakukan dengan 100 epoch masing-masing *optimizers*. Ada 6 optimizers yang dicoba pada eksperimen ini yakni, Nadam, RMSProp, Adam, SGD, Adadelta, dan Adagrad. Di bawah ini adalah hasil eksperimen dari masing-masing model.

Tabel 4.3 Hasil akurasi dari 6 *Optimizers*

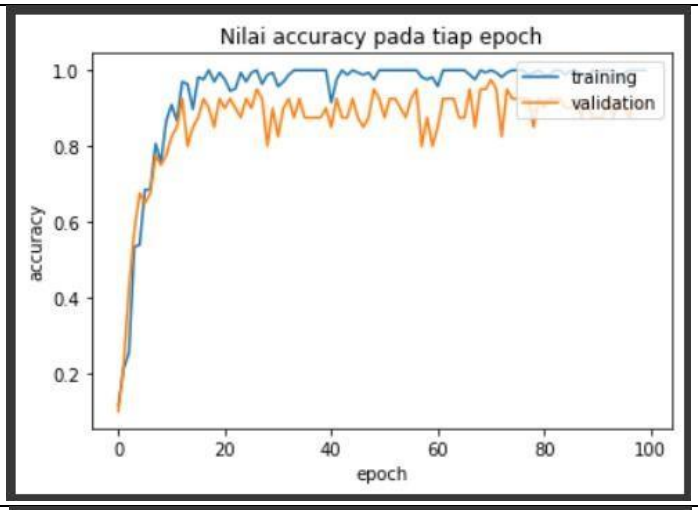
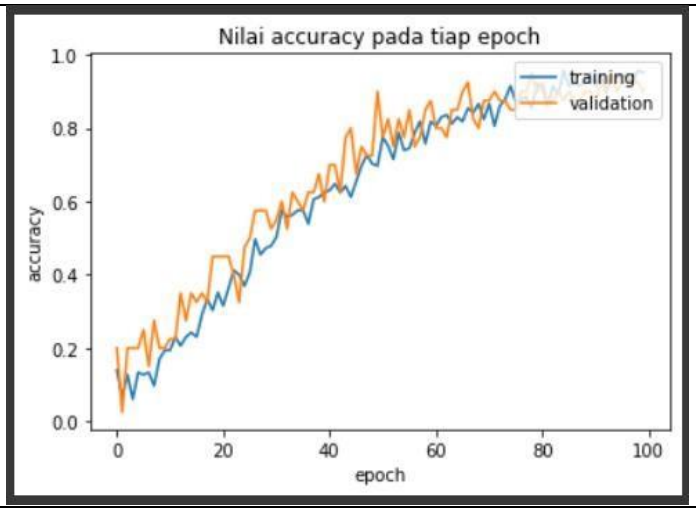
No	Optimizers	Akurasi
1	Adagrad	88%
2	RMSProp	88%
3	SGD	84%
4	Nadam	84%

Tabel 4.3 Hasil akurasi dari 6 *Optimizers*

No	Optimizers	Akurasi
5	Adam	80%
6	Adadelta	14%

Dari 6 proses train yang dilakukan menggunakan *optimizers* yang berbeda, pada tabel 4.2 terlihat nilai akurasi tertinggi yakni 88% menggunakan optimizers RMSProp dan Adagrad. Proses train ini dilakukan beberapa kali dengan hasil yang sama yakni RMSProp dan Adagrad yang tertinggi. Untuk itu perlu dilihat aspek lain untuk menentukan optimizer mana yang akan digunakan untuk aplikasi android nantinya. Aspek yang digunakan yaitu melihat hasil grafik dan *Confusion Matrix* dari kedua *optimizer*, berikut di bawah ini adalah hasil grafiknya.

Tabel 4.4 Hasil grafik model rmsprop dan adagrad

No	Optimizers	Grafik
1	RMSProp	
2	Adagrad	

Tabel 4.5 Hasil *confusion matrix* model rmsprop dan adagrad

No	Optimizer	Confusion Matrix
1	RMSProp	[[5 0 0 0 0 0 0 0 0 0] [0 5 0 0 0 0 0 0 0 0] [0 0 5 0 0 0 0 0 0 0] [0 0 0 3 1 0 0 0 1 0] [0 0 0 0 5 0 0 0 0 0] [0 0 0 0 0 5 0 0 0 0] [0 0 0 0 0 0 5 0 0 0] [0 0 0 0 1 0 0 4 0 0] [0 0 0 0 0 0 0 0 4 1] [0 0 0 0 0 0 1 0 1 3]]
2	Adagrad	[[4 0 0 0 0 0 1 0 0 0] [0 5 0 0 0 0 0 0 0 0] [0 0 5 0 0 0 0 0 0 0] [0 0 0 3 1 0 0 0 1 0] [0 0 0 0 5 0 0 0 0 0] [0 0 0 0 0 5 0 0 0 0] [0 0 0 0 0 0 5 0 0 0] [0 0 0 0 0 0 0 4 0 1] [0 0 0 0 0 0 0 0 5 0] [0 0 0 0 0 0 1 0 1 3]]

Dari aspek grafik, tabel 4.4 grafik Adagrad menunjukkan jarak garis antara *training* dan *validation* saling berdekatan, sedangkan RMSProp jarak antara kedua garis tersebut mulai merenggang ini menunjukkan adanya sedikit *overfitting* karena nilai *training* dan *validation* mempunyai jarak yang cukup jauh. Kemudian untuk aspek *Confusion Matrix* yang ditunjukkan oleh tabel 4.4, dari 10 kelas yang diujikan dengan data *validation* yang berjumlah masing-masing 5 citra per kelas, menunjukkan hasil beberapa yang *miss*. Model dengan *optimizer RMSProp* menunjukkan hasil ada 4 kelas yang tidak semuanya teridentifikasi yakni pada kelas 4, 8, 9, dan 10. Sedangkan model dengan *optimizer* Adagrad menunjukkan hasil yang sama yakni 4 kelas yang *miss* yaitu pada kelas 1, 4, 8, dan 10.

Karena aspek *confusion matrix* menunjukkan jumlah miss yang sama dan nilai akurasi kedua model sama maka keputusan pemilihan *optimizer* akan dilihat dari grafik saja. Maka keputusan yang diambil untuk pemilihan *optimizer* pada penelitian ini adalah *optimizer Adagrad*. Dengan alasan grafik *RMSProp* menunjukkan adanya kerenggangan jarak kedua garis, sedangkan grafik Adagrad masih lebih bagus karena kedua garis masih saling berdekatan. Kemudian pada model ini *loss function*-nya menggunakan *Categorical Cross Entrophy*. *Categorical Cross Entropy*

adalah *loss function* yang paling cocok karena pada penelitian ini menggunakan multi kelas yakni ada 9 jenis tanaman hias atau 9 *output*.

4.2 Ekspor Model Menjadi TFLite

Setelah penentuan metode optimasi menggunakan Adagrad, maka selanjutnya model tersebut akan di ekspor menjadi tflite beserta labels.txt yang didapat dari nama direktori setiap kelas tanaman hias. Di bawah ini adalah source code yang membuat labels.txt dari kelas tanaman hias serta source code ekspor model kedalam bentuk ekstensi .tflite yang disajikan dengan gambar 4.1 dan 4.2 di bawah ini.

```
1. print(train_datagen.class_indices)
2. labels = '\n'.join(sorted(
3.       train_datagen.class_indices.keys()
4. ))
5. with open('labels.txt', 'w') as f:
6.     f.write(labels)
```

Gambar 4.1 *Source code* membuat labels.txt

```
1. saved_model_dir = ''
2. tf.saved_model.save(model, saved_model_dir)
3.
4. converter =
   tf.lite.TFLiteConverter.from_saved_model(saved_model_dir)
5. tflite_model = converter.convert()
6.
7. with open('model.tflite', 'wb') as f:
8.   f.write(tflite_model)
```

Gambar 4.2 *Source code* ekspor model menjadi .tflite

Pada gambar 4.1 kode pada baris 2 dan 3 merupakan struktur isi data dengan variabel ‘labels’ yang diisi dengan nama kelas berdasarkan ‘train_datagen’ kemudian ditambahkan ‘\n’ yang merupakan pemisah berupa baris baru setiap nama kelas. Variabel train_datagen adalah alamat direktori yang mengarah ke dataset *train* pada penelitian ini, kemudian pada variabel train_datagen ditambahkan ‘.class_indices.keys()’ yang berarti setiap *folder* yang ada pada direktori *train* diambil namanya lalu dijadikan menjadi sebuah *class*. Setelah itu pada baris kode 5 dan 6 di atas merupakan kode untuk menyimpan nama-nama kelas dari direktori *train* disimpan menjadi *file* berekstensi .txt atau teks dengan nama *file* ‘labels.txt’.

Kemudian penjelasan kode pada gambar 4.2 adalah baris 1 merupakan direktori tujuan yang nantinya akan menyimpan model tflite dari penelitian ini. Variabel ‘saved_model_dir’ berisikan ‘ ’ atau kosong yang berarti file model nanti akan disimpan di direktori yang sama dengan kode python penelitian ini ditulis. Baris 2 fungsi tf.saved_model.save(model, saved_model_dir) merupakan fungsi untuk menyimpan model kita ke direktori yang baru saja buat yakni pada variabel saved_model_dir. Kemudian pada baris 5 variabel converter berisikan fungsi untuk menyimpan direktori dari model yang akan dikonversi menjadi tflite, setelah itu di baris 6 ada variabel tflite_model yang berisikan fungsi *convert()* yakni untuk mengkonversi model yang sudah disimpan di saved_model_dir pada variabel converter menjadi tflite. Kemudian pada baris 7 model tflite disimpan dengan nama ‘model.tflite’. Berikut adalah output dari kedua *source code* di atas.

Gambar 4.3 Output ekspor model ke tflite serta labels.txt

Setelah menghasilkan 2 *file* tersebut tahap selanjutnya adalah pembuatan aplikasi android menggunakan flutter.

4.3 Membuat Data JSON

Tahap selanjutnya yaitu membuat data JSON yang berisikan data-data tanaman hias. Data ini digunakan sebagai data *output* yang akan ditampilkan nanti pada aplikasi Android. Data JSON pada penelitian ini dibuat dengan struktur sebagai berikut.

1. id

   Id ini digunakan sebagai pembeda antar data kelas masing-masing tanaman hias.
2. casualName

   casualName ini diisi dengan nama tanaman hias yang umum dikenal masyarakat Indonesia.
3. realName

   realName ini diisi dengan nama latin tanaman hias.

4. temperature

   temperature ini diisi dengan data suhu tempat tanaman hias disimpan.

5. watering

   watering ini diisi dengan data penyiraman tanaman hias dalam jangka beberapa hari atau minggu sekali.

6. description

   description ini diisi dengan data tanaman hias mulai dari ciri-ciri, dimana saja dia tumbuh, dan lain-lain.

7. pros

   pros ini diisi dengan data manfaat dari masing-masing tanaman hias.

8. cons

   cons ini diisi dengan data bahaya dari masing-masing tanaman hias.

9. link

   link ini diisi dengan data berupa alamat tautan pencarian pada google dengan kata kunci ‘manfaat dan bahaya tanaman hias x’ (x = nama umum tanaman hias).

Data-data yang akan diisikan ke JSON ini didapat dari berbagai macam sumber di internet, bisa dari jurnal, website, atau wikipedia. Semua data di atas dimasukkan kedalam array ‘data’ seperti contoh pada gambar 4.4 di bawah ini.


```
{
    "data":[
        {
            "id" : 1,
            "casualName" :"Andong",
            "realName" : "Cordyline fruticosa",
            "temperature" : "15'C - 25'C",
            "watering" : "1 Hari Sekali",
             "description" : "Tanaman daun andong adalah tanaman hias dari keluarga Asparagaceae, berasal dari Austronesia, Asia Tenggara, Oseania dan tersebar secara luas di daerah Indonesia. Jenis tanaman termasuk dalam kategori monokotil, tinggi tanaman daun andong 2-4 meter, batangnya bulat, keras, bekas daun rontok berbentuk cincin. Daunnya tunggal dengan warna hijau, ada juga yang berwarna merah kecoklatan. Letak daun tersebar pada batang, terutama berkumpul di ujung batang. Helaian berbentuk lanset dengan panjang 20-60 cm dan lebar 5-13 cm. Ujung dan pangkalnya runcing, tepinya rata, pertulangannya menyirip dan tangkai daunnya berbentuk talang.",
            "pros" : "tanaman Andong mempunyai salah satu manfaat bagi kesehatan yakni:\n\n1. Untuk mengatasi batuk darah, urine berdarah, dan haid terlalu banyak Bagian yang digunakan yaitu daun atau akar, cara pembuatan ramuannya yaitu dengan merebus daun andong segar ±90 gr, atau akar andong kering ±50 gr dengan tiga gelas air sampai tersisa 1 gelas. Kemudian disaring dan diminum pagi dan sore sebanyak ½ gelas.\n\n2. Untuk obat diare / disentri Bagian yang digunakan adalah daun atau akar tanaman, cara nya yaitu merebus bagian daun ±90 gr, atau akar andong kering ± 15 gr menggunakan tiga gelas air. Rebus hingga tersisa 1 gelas, kemudian di saring dan di bagikan menkadi 3 gelas ukuran sama. Diminum 3 x sehari, pagi, siang, dan sore.\n\n3. Mengatasi wasir Bahan yang digunakan adalah daun andong 3 helai + daun sungu 7 helai, kemudian direbus dengan tiga gelas air sampai tersisa 1 gelas. Disaring, dan minum secara rutin setiap hari.\n\n4. Obati luka akibat sengatan binatang beracun Bagian yang digunakan adalah daun andong, haluskan beberapa lembar daun andong segar, kemudian panaskan diatas api. Dalam kondisi hangat , diusapkan pada bagian tubuh yang terluka akibat sengatan dan di balut dengan kain. Ganti 2-3 kali sehari.\n\n5. Mengobati radang gusi Kulit andong di kikis secukupnya, tambahkan sedikit garam, aduk sampai rata. Kemudian oleskan ramuan pada bagian gusi yang meradang.",
            "cons" : "atau efek samping bagi manusia sampai saat ini belum ada, tetapi tanaman Andong ini mengandung saponin yang beracun bagi hewan berdarah dingin seperti Ular, Kadal, Buaya, Komodo, Kura-kura, dan lain - lain.",
```


Gambar 4.4 Data JSON

4.4 Aplikasi Android

Aplikasi android pada penelitian ini didesain menjadi 6 halaman utama ditambah dengan 1 halaman *web view*. Desain mockup untuk aplikasi penelitian ini sudah dipaparkan pada bab 3.8. Bahasa pemrograman yang digunakan adalah bahasa Dart menggunakan Flutter. Aplikasi pada penelitian ini dinamakan PLANTIS singkatan dari *Plant Identification System*. Ada 13 *file* Dart utama pada aplikasi Plantis yang menghasilkan 6 halaman utama ditambah 1 halaman web view. Berikut adalah hasilnya.

1. Halaman Splash Screen

Gambar 4.5 *Splash Screen*

Halaman ini adalah halaman yang pertama muncul apabila aplikasi Plantis dibuka. Halaman ini berisikan logo aplikasi bertuliskan Plantis. Logo Plantis ini diilustrasikan dengan huruf P pada logo Plantis terlihat seperti kaca pembesar dengan daun ditengahnya. Ini mengilustrasikan seperti kaca pembesar sebagai sedang mencari sesuatu dari daun tersebut. Lalu ditambahkan variasi titik pada huruf ‘i’ dengan *icon* daun yang sesuai dengan tema fungsi aplikasi Plantis yakni identifikasi tanaman hias menggunakan objek daun.

2. Halaman Intro

Gambar 4.6 *Intro Page*

Halaman *intro* tersebut muncul setelah halaman splash sebelumnya. Halaman ini berisikan logo di atas beserta kepanjangannya yang berada di bawah logo. Kemudian ada ilustrasi yang menggambarkan aplikasi Plantis ini dengan foto tanaman dengan kotak yang berada di daunnya beserta tulisan dari nama latin tanaman tersebut. Kemudian ada penjelasan singkat di bawah foto tersebut beserta tombol mulai untuk mengarah ke halaman utama aplikasi. Terakhir, paling bawah ada teks *copyright* pengembang aplikasi Plantis.

3. Halaman Guide

Gambar 4.7 *Guide Page*

Halaman pada gambar 4.7 itu muncul setelah user menekan tombol ‘mulai’ pada halaman *intro* sebelumnya. Pada halaman ini berisikan 4 guide yang akan memandu user untuk penggunaan aplikasi Plantis ini.

4. Halaman Capture

Gambar 4.8 *Capture Page*

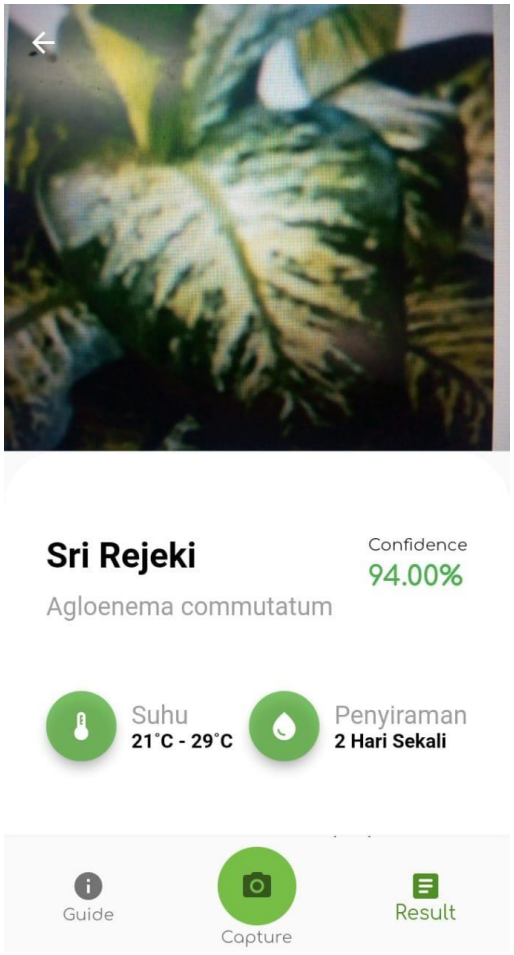
Ini adalah halaman *capture* yang berada di tengah atau indeks kedua dari navigasi yang berada di bawah. Halaman ini hanya berisikan tombol ‘Start’ di tengah untuk menuju halaman kamera guna mengambil citra daun.

5. Halaman Camera

Gambar 4.9 *Camera Page*

Halaman kamera ini berisikan 1 tombol dengan *icon* kamera, dan kotak serta panduan di bawahnya. *User* akan dipandu sesuai teks yang tertera di bawah kotak tersebut guna memastikan citra daun yang diambil agar pas dalam kotak tersebut.

6. Halaman Result



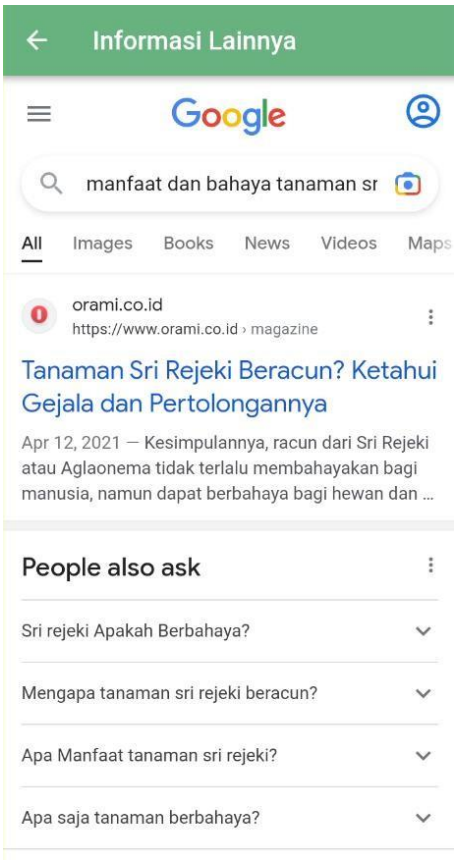
Gambar 4.10 *Result Page*

Halaman *result* atau hasil adalah halaman *output* dari aplikasi Plantis. Halaman ini berisikan nama umum yang dikenal masyarakat, nama latin dari tanaman hias tersebut, kemudian ada informasi suhu, penyiraman, deskripsi, manfaat, bahaya, dan tombol Informasi Lainnya.

Untuk bagian *confidence*, pada aplikasi ini diatur apabila citra yang diambil oleh user mempunyai nilai *confidence* dibawah 70%, maka halaman *result* akan langsung mengeluarkan hasil tidak diketahui dan tombol Informasi Lainnya akan di *disabled*.

Nilai *confidence* hanya akan muncul apabila nilainya lebih dari 70%. Untuk pewarnaan teks *confidence*, nilai pada rentang antara 70% - 89% akan berwarna oranye. Kemudian jika nilai *confidence* diatas 89% maka nilai *confidence* akan berwarna hijau, selain itu nilai *confidence* akan berwarna merah karena dibawah 70% dan langsung diberi nilai 0.0% oleh sistem.

7. Halaman Web View



Gambar 4.11 *Web View* informasi lainnya

Setelah itu pada aplikasi ini menyediakan fitur halaman Informasi Lainnya. Fitur ini berupa halaman *web view* yang isinya seperti menampilkan peramban dengan alamat tautan google. Fitur ini dapat diakses dengan tombol Informasi lainnya yang tertera di bagian paling bawah halaman *result*. Fitur ini ada karena untuk membantu *user* mencari informasi lainnya apabila informasi yang disampaikan oleh aplikasi Plantis dirasa kurang membantu.

4.5 Pengujian Black Box

Tahap selanjutnya adalah memasuki tahap pengujian Black Box. Tahap ini bertujuan untuk menguji fungsionalitas aplikasi apakah sudah berjalan seperti yang diharapkan atau tidak. Berikut adalah tabel pengujian *black box* pada aplikasi Plantis.

Tabel 4.6 Uji *Black Box*

No	Komponen Uji	Hasil yang Diharapkan	Keterangan
1	Halaman *Splash Screen*	Aplikasi akan memunculkan halaman cepat atau splash dengan logo 'Plantis' saat pertama kali dibuka.	Sukses
2	Halaman *Introduction*	Aplikasi memunculkan halaman pengenalan atau intro dengan logo, teks, gambar background tanaman, teks penjelasan singkat, tombol mulai, dan *copyright* pengembang aplikasi.	Sukses
3	Halaman *Guide*	Aplikasi memunculkan halaman guide dengan 4 buah halaman *carousel* yang bisa di geser ke kanan atau ke kiri, diakses dari tombol 'Mulai' pada halaman *intro*.	Sukses

Tabel 4.7 Uji *Black Box*

No	Komponen Uji	Hasil yang Diharapkan	Keterangan
4	Halaman *Capture*	Aplikasi memunculkan halaman *Capture* dengan teks dan tombol 'Start' di tengah halaman yang diakses dari tombol navigasi bagian tengah atau kedua dengan *icon* kamera.	Sukses
5	Halaman *Camera*	Aplikasi memunculkan tampilan kamera bagian belakang dengan box dan teks di tengah serta tombol kamera di bawah.	Sukses
6	Halaman *Result*	Aplikasi memunculkan hasil identifikasi dari halaman *camera* melalui tombol dengan *icon* kamera bagian bawah.	Sukses
7	Response halaman *Result* muncul kurang dari 2 detik	Aplikasi mengeluarkan hasil indentifikasi kurang dari 2 detik.	Sukses
8	Halaman *Web View*	Aplikasi memunculkan halaman peramban pencarian google dengan kata kunci 'manfaat dan bahaya tanaman x' (x = nama tanaman hias)	Sukses

4.6 Pengujian Parameter *Confidence* Aplikasi

Pengujian ini melakukan eksperimen dengan nilai persentase *confidence* pada aplikasi. Ada 3 nilai yang diujikan yakni, 70%, 80%, dan 90%. Apabila aplikasi mengenali tanaman hias dibawah dari salah satu ketiga nilai tersebut maka aplikasi akan memunculkan hasil 'Tidak Diketahui'. Pengujian dilakukan dengan mencoba aplikasi ke 9 jenis tanaman hias yang ada pada penelitian ini kemudian dihitung berapa kali aplikasi bisa mengenali tanaman hias sampai benar. Kemudian juga dihitung jumlah kesalahan identifikasi yang dilakukan oleh aplikasi. Berikut adalah rekap hasil pengujian ini.

Tabel 4.8 Hasil Pengujian Nilai Parameter *Confidence* Aplikasi

Jenis Tanaman Hias	Confidence < 70%		Confidence < 80%		Confidence < 90%	
	Percobaan	Salah	Percobaan	Salah	Percobaan	Salah
Andong	1	-	2	-	1	-
Kornus	1	-	1	-	3	-
Krokot Epah Daun Merah	2	-	2	-	6	1
Lidah Buaya	1	-	1	-	1	-
Lidah Mertua	1	-	3	-	1	-
Miana	3	2	2	1	1	-
Pucuk Merah	1	-	1	-	1	-
Puring	2	1	2	-	1	-
Sri Rejeki	1	-	2	-	1	-
Total	13	3	16	1	16	1

Dari ketiga hasil tabel 4.8 di atas dapat dilihat bahwa confidence dengan nilai 80% dan 90% memiliki kesamaan yakni total ada 16 percobaan dengan 1 kesalahan identifikasi. Dari pengujian ini diketahui juga bahwa ada jenis tanaman hias yang sulit dikenali oleh aplikasi ini, yakni tanaman Krokot, Miana, dan Puring. Ada sejumlah faktor yang mempengaruhi aplikasi butuh beberapa kali untuk melakukan identifkasi tanaman hias dengan benar, yakni:

1. Jarak pengambilan citra. Ketika aplikasi memfoto hanya satu daun secara dekat, aplikasi cenderung tidak mengetahuinya, tapi apabila jarak pengambilan gambar sedikit lebih jauh (menangkap gambar daun dengan lengkap) aplikasi baru bisa mengidentifkasi.

2. Kurangnya citra satu daun dengan utuh pada dataset. Pada dataset penelitian ini cenderung memang foto dari jenis-jenis tanaman menangkap banyak daun dalam satu *frame*. Maka dari itu apabila aplikasi menangkap foto satu daun tanaman hias full dalam satu *frame* hasilnya akan sulit mengenali atau 'Tidak Diketahui'. Tapi walaupun begitu dari percobaan yang dilakukan, jenis-jenis tanaman selain krokot, miana, dan puring dapat dikenali dengan baik walaupun pada saat pengambilan gambar hanya satu daun saja dalam satu *frame*.
3. Ketiga jenis daun tanaman hias tersebut mirip dan banyak varian. Dari ketiga jenis tersebut, aplikasi terkadang salah mengenali dikarenakan daunnya mempunyai banyak varian walaupun di spesies yang sama. Dan juga daunnya terkadang mirip-mirip, misalnya tanaman miana pernah diidentifikasi oleh aplikasi sebagai puring.

Dari pengujian ini maka akan dilakukan sedikit penambahan intruksi pada halaman guide di aplikasi, yakni jika aplikasi tidak dapat mengidentifikasi tanaman hias, maka user harus sedikit mundur dalam proses pengambilan fotonya. Kemudian keputusan untuk penentuan nilai parameter *confidence* akan menggunakan nilai yang 80%, dengan alasan pada nilai 80% dari 16 percobaan hanya 1 saja yang salah, jumlah ini sama dengan nilai 90%.

Sedangkan jika menggunakan nilai 90% pada jenis krokot epah, ada 6 kali percobaan dalam satu jenis tanaman saja, hal ini paling banyak diantara jenis-jenis lainnya. Maka dengan faktor tersebut keputusan yang paling baik adalah memilih nilai parameter 80% saja, karena dikhawatirkan user nantinya akan mengulang-ulang proses pengambilan gambar jika nilai parameter dipasang sebesar 90%.

4.7 Pengujian Skala Likert

Terakhir ini merupakan pengujian aplikasi kepada user lalu disediakan beberapa pernyataan untuk mengetahui penilaian secara langsung. Pengujian dilakukan menggunakan google form disertai dengan *link download* aplikasi Plantis. Penilaian akan dilakukan dengan 3 aspek yakni aspek desain *user interface*, kemudahan penggunaan aplikasi, dan kinerja prediksi. Aspek user interface bertujuan untuk mengetahui penilaian koresponden terhadap tampilan aplikasi

secara keseluruhan mulai dari halaman intro sampai *result*. Untuk aspek kemudahan penggunaan aplikasi ini bertujuan untuk mengetahui penilaian sesorang terhadap jalannya aplikasi Plantis tersebut, mulai dari berjalannya aplikasi, data-data yang muncul, sampai selesai melakukan identifikasi. Terakhir aspek ketiga yakni kinerja prediksi, aspek ini bertujuan untuk mengetahui penilaian user terhadap kinerja aplikasi saat melakukan identifikasi tanaman hias sampai keluar hasil datanya. Untuk aspek *user interface* disediakan 4 pernyataan, yakni sebagai berikut:

1. User dapat memahami fungsi kegunaan aplikasi dari halaman intro pada awal aplikasi dijalankan
2. Petunjuk penggunaan aplikasi yang tersedia jelas mudah dipahami
3. Tata letak teks, gambar, dan simbol rapi
4. Desain aplikasi secara keseluruhan (font, warna, logo, dan gambar) menarik

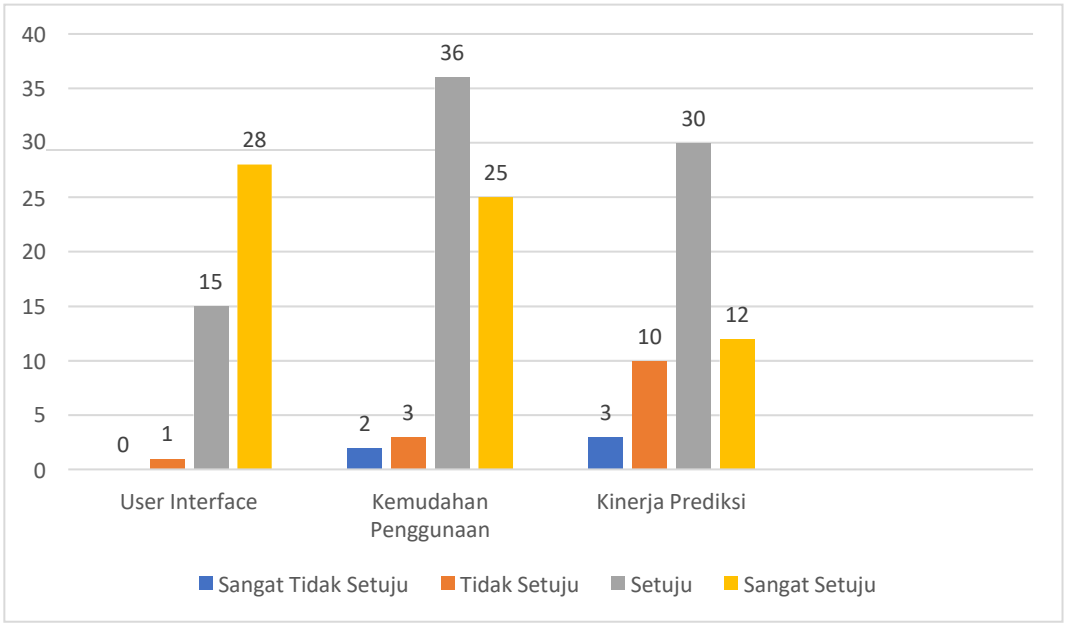
Kemudian untuk aspek kemudahan penggunaan aplikasi disediakan 5 pernyataan sebagai berikut:

1. Aplikasi berjalan dengan lancar tanpa bug atau error
2. Aplikasi terasa ringan saat dijalankan
3. Navigasi perpindahan halaman pada aplikasi tidak membingungkan
4. Data atau informasi pada halaman result dari aplikasi Plantis cukup informatif
5. Data dari hasil identifikasi tanaman hias pada halaman result jelas dan mudah dipahami
6. Fitur "Informasi Lainnya" pada halaman Result sangat berguna apabila data pada Plantis dirasa kurang informatif

Terakhir adalah aspek kinerja prediksi, ada 5 pernyataan yang diberikan, sebagai berikut:

1. Nilai *confidence* prediksi >80% dan diprediksi benar
2. Hasil identifikasi benar dan dilakukan dengan sekali percobaan
3. Perlu <= 3 kali percobaan sampai tanaman diidentifikasi dengan benar
4. Perlu > 3 kali percobaan sampai tanaman diidentifikasi dengan benar
5. Proses identifikasi dari pengambilan gambar sampai halaman result memakan waktu yang cepat

Dari aspek yang telah dipaparkan sebelumnya, berikut adalah hasil dari pengujian menggunakan skala likert memakai 4 skala likert yakni Sangat Tidak Setuju, Tidak Setuju, Setuju, Sangat Setuju dengan jumlah koresponden 11 orang.



Gambar 4.12 Grafik Hasil Uji Aplikasi ke User

Keterangan untuk pengujiannya terbagi menjadi 3 aspek yang masing-masing mempunyai jumlah pernyataan yang berbeda. Pada aspek *User Interface* ada 4 pernyataan, Kemudahan Penggunaan Aplikasi ada 6 pernyataan, dan Kinerja Prediksi ada 5 pernyataan. Setiap koresponden harus menjawab total 15 pernyataan yang diajukan. Dari gambar 4.12 terlihat hasil dari uji aplikasi ke 11 koresponden yakni dari aspek *User Interface* respon koresponden cukup positif dengan hasil 15 Setuju dan 28 Sangat Setuju dapat disimpulkan bahwa performa User Interface pada aplikasi Plantis cukup positif walaupun ada respon dengan 1 Sangat Tidak Setuju. Selanjutnya pada aspek Kemudahan Penggunaan terlihat hasilnya juga cukup positif yakni hasil 2 Sangat Tidak Setuju, 3 Tidak Setuju, 36 Setuju, dan 25 Sangat Setuju, dapat disimpulkan bahwa aplikasi Plantis pada aspek Kemudahan Penggunaannya cukup positif walaupun ada 2 dan 3 respon yang negatif, ini dikarenakan fitur Informasi Lainnya pada aplikasi Plantis tidak jalan di beberapa *smartphone*. Terakhir pada aspek Kinerja Prediksi hasilnya Sangat Tidak Setuju sebesar 3, 10 Tidak Setuju, 30 Setuju, dan 12 Sangat Setuju. Ada respon negatif pada aspek ini karena beberapa koresponden yang merespon pernyataan poin ke 2 dan 3 dengan Sangat Tidak Setuju dan Tidak Setuju. Artinya aplikasi Plantis

terkadang memerlukan 2 sampai 3 kali bahkan lebih untuk dapat mengidentifikasikan tanaman hias secara benar. Hal tersebut disebabkan oleh beberapa faktor, kemungkinannya adalah kurangnya data latih, kurang dekatnya objek daun tanaman ke kamera, atau bahkan jaraknya terlalu dekat.

Kesimpulan keseluruhan dari 3 aspek yang diujikan ke aplikasi Plantis responnya cukup positif terlihat dari 2 aspek *User Interface* dan Kemudahan Penggunaan walaupun aplikasi Plantis terkadang memerlukan beberapa kali percobaan untuk mendapatkan hasil yang diharapkan.

Bab V
Simpulan dan Saran

5.1 Simpulan

Berdasarkan hasil pengembangan aplikasi Plantis yang dilakukan mulai dari beberapa eksperimen pembuatan model, maka dapat diambil kesimpulan seperti berikut.

1. Pada eksperimen pertama dimana model *transfer learning* dibandingkan dengan model tanpa *transfer learning*. Nilai akurasi tertinggi dari kedua model yakni adalah model dengan transfer learning. Ini dibuktikan dengan dilakukannya percobaan menggunakan *epoch*, *optimizer*, augmentasi *image* yang sama di kedua model kemudian hasilnya nilai akurasi tertinggi terdapat pada model dengan *transfer learning*.
2. Kemudian pada eksperimen kedua dan ketiga menunjukkan bahwa apabila transfer learning dilakukan dengan semua *layer* model MobileNetV2 atau 100% dapat membantu peningkatan nilai akurasi pada model. Tinggal langkah selanjutnya dilakukan variasi percobaan-percobaan seperti menggunakan *optimizers*.

Dua kesimpulan di atas sekaligus membuktikan bahwa dengan adanya metode *transfer learning* sangat membantu para peneliti dengan dataset terbatas dapat melakukan identifikasi berbagai objek. Seperti pada penelitian ini hanya menggunakan 30 citra per kelas sudah mendapatkan nilai akurasi sebesar 88% menggunakan optimizer Adagrad dan RMSprop.

Setelah dari percobaan-percobaan di atas maka hasil model tersebut sudah bisa dipakai untuk aplikasi androidnya. Untuk aplikasinya dari 11 koresponden saat uji aplikasi tidak ada *bug* atau *error* saat pemasangan. Maka dapat disimpulkan bahwa luaran pada penelitian ini sudah terpenuhi yakni membuat aplikasi android tentang identifikasi tanaman hias guna membantu memudahkan masyarakat untuk mendapatkan informasi mengenai tanaman hias.

5.2 Saran

Berdasarkan simpulan yang diperoleh, berbagai saran dan perbaikan dalam penelitian ini adalah sebagai berikut.

1. Melakukan penambahan kelas atau jenis tanaman hias, agar aplikasi ini dapat lebih banyak mengenali berbagai jenis tanaman hias lain.
2. Melakukan penambahan fitur pada aplikasi Plantis seperti penyimapanan data tanaman hias setelah diidentifikasi ke dalam *database*.
3. Perlu melakukan perubahan UI yang lebih bagus dan *modern* lagi agar memungkinkan user menjadi lebih tertarik lagi menggunakan aplikasi ini
4. Penambahan citra pada dataset sangat disarankan apabila mempunyai *source* yang lebih tentang citra tanaman hias.
5. Melakukan percobaan menggunakan model yang lain selain MobileNetV2, lalu diuji akurasinya.

Daftar Pustaka

Adam, R., 2021. *STRUCTILMY.* [Online] Available at: https://structilmy.com/2021/01/transfer-learning-solusi-deep-learning-dengan-data-sedikit/ [Accessed 25 November 2021].

Anggiratih, E., Siswanti, S., Octaviani, S. K. & A., 2021. Klasifikasi Penyakit Tanaman Padi Menggunakan Model Deep Learning Efficientnet B3 Dengan Transfer Learning. *Jurnal Ilmiah Sinus (JIS),* 19(1), pp. 75-82.

Asep, 2023. *Dogwood atau genus Cornus.* [Online] Available at: https://artikelkeren.com/dogwood-atau-genus-cornus.html [Accessed 30 01 2023].

ASPCA, Tanpa Tahun. *Snake Plant.* [Online] Available at: https://www.aspca.org/pet-care/animal-poison-control/toxic-and-non-toxic-plants/snake-plant [Accessed 29 April 2022].

Azizah, A. N. I., 2021. *Krokot Epah Daun Merah.* [Online] Available at: https://eco.sdmupat.sch.id/krokot-epah-daun-merah/ [Accessed 30 Januari 2023].

Baktikominfo, 2019. *Bakti.* [Online] Available at: https://www.baktikominfo.id/id/informasi/pengetahuan/bahasa_pemrograman_python_pengertian_sejarah_kelebihan_dan_kekurangannya-954 [Accessed 28 November 2021].

Darmatasia, 2020. Deteksi Penggunaan Masker Menggunakan Xception Transfer Learning. *Jurnal Instek : Informatika Sains Dan Teknologi,* 5(2), p. 284.

Ilahiyah, S. & Nilogiri, A., 2018. Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. *JUSTINDO (Jurnal Sistem & Teknologi Informasi Indonesia),* 3(2), pp. 52-55.

Kaur, T., & Gandhi, T. K. 2019. *Automated Brain Image Classification Based On VGG-16 and Transfer Learning. Proceedings-2019 International Conference on Information Technology, ICIT 2019,* 94–98.

Kominfo. 2019. *[DISINFORMASI] Tanaman Hias Beracun Paling Mematikan.* 2019. Diakses pada 15 September 2021, dari https://www.kominfo.go.id/content/detail/17256/disinformasi-tanaman-hias-beracun-paling-mematikan/0/laporan_isu_hoaks.

Kusumaningrum, T. F., 2018. *Implementasi Convolutional Neural Network (CNN) Untuk Klasifikasi Jamur Konsumsi Di Indonesia Menggunakan Keras*. Yogyakarta: Universitas Islam Indonesia Yogyakarta.

Levina, 2022. *8 Macam Tanaman Hias Daun untuk Teras Rumah.* [Online] Available at: https://www.ruparupa.com/blog/macam-macam-tanaman-hias-daun/
[Accessed 26 January 2023].

Liu, S., 2021. *Statista.* [Online] Available at: https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/
[Accessed 27 November 2021].

Nareza, d. M., 2021. *Alodokter: 10 Manfaat Lidah Buaya untuk Kesehatan.* [Online] Available at: https://www.alodokter.com/khasiat-lidah-buaya-yang-sebenarnya [Accessed 18 Mei 2022].

Ningsih, E. P. & Rohmawati, I., 2019. Respon Stek Pucuk Tanaman Miana (Coleus atropurpureus(L.) Benth) terhadap Pemberian Zat Pengatur Tumbuh. *Jurnal Biologi Tropis,* 19(2), p. 278.

Noviana, A., Indriani, Y., Situmorang, S., 2014. *Perilaku Konsumen Dalam Pembelian Tanaman Hias Di Kecamatan Pekalongan Kabupaten Lampung Timur, 2(1),* 77.

Nugroho, P. A., Fenriana, I., Arijanto, Rudy., 2020. *Implementasi Deep Learningmenggunakan Convolutional Neural Network (Cnn)pada Ekspresi Manusia.* 2(1), 13.

Okta, R. E., 2018. *MobileNet: Deteksi Objek pada Platform Mobile.* [Online] Available at: https://medium.com/nodeflux/mobilenet-deteksi-objek-pada-platform-mobile-bbbf3806e4b3
[Accessed 25 November 2021].

Oliver, A., 2021. *glints.* [Online] Available at: https://glints.com/id/lowongan/google-colab-adalah/#.YOlSd-gzZEY [Accessed 26 November 2021].

Pellegrini, L.A. 2008. *An Argument For Criminal Hoax. Disertasi. University of Southern California*.

Putra, 2019. *Salamadian Muda & Berilmu.* [Online] Available at: https://salamadian.com/pengertian-android/ [Accessed 25 November 2021].

Putri, F. A. W., Tanpa Tahun. *theAsianParent: Hati-Hati! 5 Efek Samping Lidah Buaya yang Bisa Timbul Bagi Kesehatan.* [Online] Available at: https://id.theasianparent.com/efek-samping-lidah-buaya#:~:text=Sebab%2C%20lidah%20buaya%20dapat%20menimbulkan,berat%20badan%2C%20dan%20gangguan%20jantung. [Accessed 18 Mei 2022].

Qu, Z. et al., 2019. Genetic Optimization Method of Pantograph and Catenary Comprehensive Monitor Status Prediction Model Based on Adadelta Deep Neural Network. *IEEEAccess,* 7(10), p. 23215.

Rizal, S. et al., 2020. Deep Learning untuk Klasifikasi Diabetic Retinopathy menggunakan Model EfficientNet. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika,* 8(3), p. 698.

Rochman, F. & Junaedi, H., 2020. Implementasi Transfer Learning Untuk Identifikasi Ordo Tumbuhan Melalui Daun. *Jurnal Syntax Admiration,* 1(6), pp. 672-677.

Rosha, P. T., Fitriyana, M. N., Ulfa, S. F. & D., 2013. Pemanfaatan Sansevieria Tanaman Hias Penyerap Polutan Sebagai Upaya Mengurangi Pencemaran Udara Di Kota Semarang. *Jurnal Ilmiah Mahasiswa,* III(1), p. 6.

Saha, S., 2018. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way.* [Online] Available at: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 [Accessed 25 November 2021].

Santoso, A. & Ariyanto, G., 2018. Implementasi Deep Learning Berbasis Keras Untuk Pengenalan Wajah. *Jurnal Teknik Elektro,* 18(01), p. 17.

Sitepu, A. C. & Sigiro, M., Tanpa Tahun. Analisis Fungsi Aktivasi Relu Dan Sigmoid Menggunakan Optimizer Sgd Dengan Representasi Mse Pada Model Backpropagation. *JUTISAL (Jurnal Teknik Informatika Komputer Universal),* Tanpa Volume(Tanpa Nomor), p. 18.

Smith, P., & Chen, C. 2019. *Transfer Learning with Deep CNNs for Gender Recognition and Age Estimation. Proceedings-2018 IEEE International Conference on Big Data, Big Data 2018,* 2564–2571.

Suarsana, I. N., Kumbara, A. N. A. & Satriawan, I. K., 2014. *Tanaman Obat Sembuhkan Penyakit Untuk Sehat.* 1st ed. Bali: Swasta Nulus.

Tan, C., Sun, F., Kong, T. 2018. *A Survey on Deep Transfer Learning.*

Ukkas, M. I., 2017. Implementasi Skala Likert Pada Metode Perbandingan Eksponensial Untuk Menentukan Pilihan Asuransi. *SESINDO,* -(-), p. 101.

Witanto, K. S. et al., 2022. Implementasi LSTM pada Analisis Sentimen Review Film Menggunakan Adam dan RMSprop Optimizer. *Jurnal Elektronik Ilmu Komputer Udayana,* 10(4), p. 356.

Yanuar, A., 2018. *Fully-Connected Layer CNN dan Implementasinya.* [Online] Available at: https://machinelearning.mipa.ugm.ac.id/2018/06/25/fully-connected-layer-cnn-dan-implementasinya/
[Accessed 31 Januari 2023].

Zulfia, D., 2012. Kajian Fisiologi Tanaman Lidah Buaya Dengan Pemotongan Ujung Pelepah Pada Kondisi Cekaman Kekeringan. *Jurnal Perkebunan & Lahan Tropika,* 2(1), p. 7.

Lampiran A

Hasil Uji Aplikasi ke User

Tabel Hasil Uji

Aspek	Sangat Tidak Setuju	Tidak Setuju	Setuju	Sangat Setuju	Total
User Interface	-	1	15	28	44
Kemudahan Penggunaan	2	3	36	25	66
Kinerja Prediksi	3	10	30	12	55
Total	5	14	71	76	165

Lampiran B

Pesan Dari Beberapa User Untuk Aplikasi Plantis

Perbanyak jenis tanaman

jumlah tanaman dilebih banyakan terutama tanaman asing & tanaman diseluruh dunia, text jangan buram, fitur dilebihbanyakan contoh history

sudah oke kok. mantapp

Apakah aplikasi dibuat agar bisa dijalankan dengan posisi landscape atau tidak? Jika iya, tampilan perlu dibuat agar responsive khusus landscape. Jika tidak, yang perlu dibuat responsive adalah bagian guide karena pada device yang memiliki ukuran lebih kecil tidak bisa melihat secara keseluruhan guide hingga bawah. Sarannya bisa dibuat agar bisa di-scroll secara vertikal. Pengambilan foto dan setelahnya pada bagian hasil kurang sama, perlu diperhatikan lagi. Bagian informasi lainnya masih error.
Stacks: Saat setelah pengambilan foto, user dipindah ke halaman hasil. Saat mengambil foto lagi, kemudian dipindah ke halaman hasil kembali. Saat user 'back', perlu melakukan 'back' sebanyak pengambilan foto yang sudah dilakukan

Tombol resultnya server not found bang, perbaiki ya atau saya baru sekali coba ini mungkin

Nganu , jangan lupa Navigator pop , dicari2 mana sekiranya bisa pake itu. Sebener e tadi sempet ada lagi halaman yg numpuk tapi bukan di halaman result.

Terus bagian deskripsi tanaman itu walaupun sepele tapi perlu sedikit di perhatikan bagian (Aku nyoba di bagian Puring, itu agak rancu sih)

UI nya damn son , lumayan cantik loh. Boleh lah

Sebaiknya pada bagian result diberikan history hasil-hasil foto yang pernah diambil oleh pengguna, sehingga pengguna bisa melihat kembali gambar apa saja yang pernah diambilnya.
Mungkin juga bisa diberitahukan jenis tanaman hias apa saja yang dapat dikenali oleh aplikasi Plantis di bagian guide.

Aplikasi berjalan dgn ringan dan responnya cepat

Lampiran C

Tangkapan Layar Dari Beberapa User