Klasifikasi Genus Tanaman Anggrek Menggunakan Metode Convolutional Neural Network (CNN)

M. Raihan Rafiiful Allaam [1], Agung Toto Wibowo[2]

[1,2] Prodi S1 Informatika, Fakultas Informatika, Universitas Telkom, Bandung

[1]raihanprogrammer@students.telkomuniversity.ac.id, [2]agungtoto@telkomuniversity.ac.id

Abstrak

**Anggrek merupakan salah satu tanaman hias yang banyak dibudidayakan. Tiap genus anggrek mempunyai cara budidaya yang berbeda, sehingga para pembudidaya anggrek yang baru memulai perlu mengetahui genus dari anggrek yang akan dibudidayakannya terlebih dahulu. Namun tidak sedikit pemula yang mencoba membudidayakan anggrek tanpa ada pengetahuan dan pengalaman yang cukup, sehingga anggrek yang dibudidayakan tidak tumbuh dan berbunga dengan optimal. Dalam penelitian ini dibangun sebuah sistem yang dapat mengklasifikasi citra genus tanaman anggrek, yaitu genus *Cattleya, Dendrobium, Oncidium, Phalaenopsis* dan *Vanda.* Klasifikasi citra dilakukan dengan menggunakan metode *Convolutional Neural Network* (CNN). Dimana citra tanaman anggrek sebagai data *input* akan dilakukan proses klasifikasi sesuai dengan genusnya. Semua proses klasifikasi ini dilakukan melalui skema *training* dan *testing,* dimana tahap *training* menghasilkan sebuah model CNN beserta bobot (*weight*) yang telah diperbarui (*updated*), lalu tahap *testing* menggunakan model tersebut untuk diujikan terhadap data citra yang baru. *K-Fold Cross Validation* digunakan pada tahap *training,* lalu untuk mengevaluasi model CNN setelah dilakukan *testing,* digunakan *Confusion Matrix*. Selain itu, pada penelitian ini digunakan arsitektur CNN kustom dan MobileNetV2. Akhirnya, dari total model yang dihasilkan, didapat model terbaik dengan *score* akurasi *testing dari lapangan* sebesar 90.44% dan *score* akurasi *testing dari internet* sebesar 80.54%, serta *F1-Score* tertinggi sebesar 98% dari genus *Dendrobium.*

Kata Kunci: ***anggrek, klasifikasi citra, convolutional neural network, k-fold cross validation, confusion matrix.***

Abstract

Orchid is one of the ornamental plants that is widely cultivated. Each genus of orchids has different cultivation methods, so orchid cultivators who are just starting out need to know the genus of orchids they will cultivate first. However, not a few beginners who try to cultivate orchids without sufficient knowledge and experience, so the cultivated orchids do not grow and flower optimally. In this study, a system was built that could classify the image of orchid genera, namely the genus *Cattleya, Dendrobium, Oncidium, Phalaenopsis* and *Vanda*. Image classification is carried out using the *Convolutional Neural Network* (CNN) method. Where the image of the orchid as input data will be carried out according to the genus classification process. All of these classification processes are carried out through a training and testing scheme, where the training stage produces a CNN model and updated weights, then the testing stage uses the model to be tested against new image data. *K-Fold Cross Validation* is used at the training stage, then to evaluate the CNN model after testing, the *Confusion Matrix* is used. In addition, this research uses custom CNN architecture and MobileNetV2. Finally, from the total models produced, the best model is obtained with a testing accuracy score from the field of 90.44% and a testing accuracy score from the internet of 80.54%, and the highest F1-score of 98% from the genus *Dendrobium*.

Keywords: ***orchid, image classification, convolutional neural network, k-fold cross validation, confusion matrix.***

1. Pendahuluan

1.1 Latar Belakang

Anggrek (*Orchidaceae*) merupakan salah satu tanaman hias yang sangat populer dan memiliki pesona yang indah. Nilai penting dari tanaman anggrek terletak pada keindahan bunganya, dan keindahan bunga anggrek akan didapat jika dalam proses pembudidayaannya dilakukan secara tepat semenjak anggrek tersebut masih usia muda. Sehingga keberhasilan pembungaan dalam budidaya anggrek menjadi faktor yang sangat penting. Tiap genus anggrek mempunyai cara budidaya yang berbeda, para pembudidaya anggrek yang baru memulai perlu mengetahui genus dari anggrek yang akan dibudidayakannya terlebih dahulu, agar mendapatkan informasi yang tepat sesuai dengan genus anggreknya. Namun tidak sedikit pemula yang belum dapat memaksimalkan proses budidaya tanaman anggrek akibat pengetahuan dan pengalaman yang kurang, sehingga anggrek yang dibudidayakan tidak dapat tumbuh dan berbunga dengan optimal. Selain itu, masih sedikit sekali pakar anggrek yang dapat ditemui sehari-hari, dan waktu yang terbatas antara pakar dengan pembudidaya anggrek pemula, serta literatur yang tidak mudah dijangkau.

Oleh karena alasan tersebut, dalam penelitian ini dibangun sebuah sistem yang dapat mengklasifikasi citra genus taman anggrek secara otomatis. Genus tanaman anggrek yang dipilih yaitu yang paling umum dibudidayakan, terlebih sebagai tanaman hias, yaitu genus *Cattleya, Dendrobium, Oncidium, Phalaenopsis* dan *Vanda.* Dengan terus berkembangnya teknologi pengolahan citra digital (*digital image processing*) dapat membantu menyelesaikan berbagai permasalahan sehari-hari, salah satunya klasifikasi citra. Dengan memanfaatkan metode *Convolutional Neural Network* (CNN) sebagai salah satu teknologi *Deep Learning*, masalah klasifikasi citra tanaman anggrek akan lebih mudah untuk dilakukan. CNN dipilih karena metode tersebut paling optimal dalam kasus klasifikasi citra, dimana salah satu kelebihannya ialah ekstraksi fitur citra yang dilakukan secara otomatis, sehingga dapat menghemat waktu dan tenaga.

1.2 Topik dan Batasannya

Penelitian tugas akhir ini mengangkat permasalahan terkait bagaimana cara membangun sebuah sistem yang dapat mengklasifikasi citra genus tanaman anggrek dengan menggunakan metode *Convolutional Neural Network* (CNN). Kemudian, menganalisis performansi dari sistem yang dibangun. Dan terakhir, men*deployment* model CNN yang telah dihasilkan kedalam sebuah aplikasi siap pakai berbasis website.

Adapun batasan pada penelitian ini yaitu citra yang dapat diklasifikasi oleh sistem hanya citra lima genus tanaman anggrek saja, yaitu *Cattleya, Dendrobium, Oncidium, Phalaenopsis* dan *Vanda.* Citra genus tanaman anggrek berusia muda dan belum berbunga, bentuk dan warna daun menjadi fokus utama dari citra. Citra memiliki pencahayaan yang baik, sudut yang tidak terlalu atas maupun bawah, dan latar belakang yang tidak terlalu ramai (*noise*).

1.3 Tujuan

Tujuan penelitian ini yaitu untuk membangun sebuah sistem yang dapat mengklasifikasi citra genus tanaman anggrek dengan menggunakan metode *Convolutional Neural Network* (CNN), serta menganalisis performansi dari sistem yang dibangun, dan men*deployment* model CNN yang telah dihasilkan kedalam sebuah aplikasi siap pakai.

1.4 Organisasi Tulisan

Pertama yaitu studi literatur, dimana penulis mengumpulkan data-data yang akurat untuk mendapatkan dasar teori tentang sistem klasifikasi genus tanaman anggrek. Pada tahap ini dilakukan pembelajaran mengenai pengolahan citra digital (*digital image processing*), *Convolutional Neural Network* (CNN), *K-Fold Cross Validation, Confusion Matrix*, Keras & Tensorflow *library*, dan Flask *Framework*. Kedua, perancangan dan realisasi, yaitu merancang alur kerja sistem, algoritma sistem, pengkodean sistem, serta menentukan *tools* maupun perangkat yang dibutuhkan sistem. Ketiga, analisis kinerja sistem, yaitu menganalisis dan mengevaluasi hasil pengujian terdahap model CNN yang dihasilkan, sehingga diketahui performansi dari model CNN tersebut dalam melakukan tugasnya, yaitu klasifikasi citra genus tanaman anggrek, lalu menarik kesimpulan. Dan terakhir, penyusunan laporan, berdasarkan alur kerja sistem dan hasil pengujian yang telah didapat.

2. Studi Terkait

2.1 Tanaman Anggrek

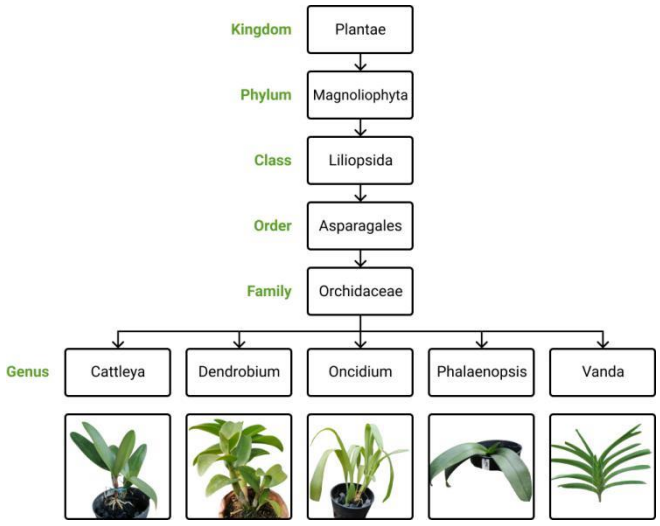
Anggrek (*Orchidaceae*) merupakan salah satu tanaman hias yang paling banyak dibudidayakan. Selain sebagai tanaman hias unggulan nasional, anggrek juga dapat memiliki nilai jual yang tinggi tergantung dari keindahan dan kelangkaannya[2]. Tanaman anggrek mudah beradaptasi dengan lingkungan tumbuhnya sehingga tidak heran apabila dapat tumbuh dan dijumpai hampir di seluruh bagian di dunia. Tempat tumbuhnya juga beragam mulai dari daerah dataran rendah sampai dataran tinggi dan dari bersuhu dingin sampai panas. Anggrek menempati posisi penting dalam industri florikultura — disiplin ilmu yang berurusan dengan pembudidayaan tanaman berbunga dan tanaman hias — di Indonesia[1].

Secara umum anggrek bisa digolongkan menjadi dua, yaitu epifit dan terresterial. Kategori epifit merupakan jenis anggrek yang tumbuhnya menempel pada tanaman lain, namun tidak bersifat parasit atau merugikan tanaman yang ditumpanginya. Sedangkan kategori terresterial adalah anggrek yang tumbuhnya di tanah. Keunggulan anggrek jika ditinjau dari aspek bisnis disebabkan antara lain karena jenisnya yang beraneka ragam, baik itu terkait bentuk dan warna serta ukuran bunganya. Selain itu, juga umumnya memiliki periode fase hidup yang lebih panjang dibandingkan bunga potong lainnya[1].

Sejauh ini telah teridentifikasi sekitar 750 famili, 43.000 spesies, dan 35.000 varietas hibrida anggrek dari seluruh penjuru dunia. Indonesia sekurangnya memiliki 5.000 spesies. Dari jumlah itu, 986 spesies tersebar di hutan-hutan di Pulau Jawa, 971 spesies berada di Pulau Sumatra, 113 spesies tumbuh di Kepulauan Maluku, dan sisanya bisa ditemukan di Sulawesi, Irian Jaya, Nusa Tenggara, dan Kalimantan[1]. Tanaman anggrek hias lebih dikenal dengan nama genusnya (marga), dibandingkan dengan nama jenisnya (spesies), dikarenakan banyak penyilangan antar spesies dan antar genus yang telah berhasil dilakukan manusia[1].

Tiap genus anggrek mempunyai cara budidaya yang berbeda[2], sehingga pembudidaya anggrek yang baru memulai perlu mengetahui genus dari anggrek yang akan dibudidayakan, agar mendapatkan informasi cara budidaya yang tepat sesuai dengan genusnya. Terdapat beberapa cara yang bisa dilakukan untuk membedakan genus anggrek, mulai dari bertanya pada ahli secara langsung, membaca literatur terkait anggrek, atau membedakannya dengan mengetahui ciri-ciri yang terdapat di masing-masing genus anggrek.

Hal ini perlu menjadi perhatian karena tidak sedikit pemula yang mencoba membudidayakan anggrek tanpa ada pengetahuan dan pengalaman yang cukup, sehingga anggrek yang dibudidayakan tidak tumbuh dan berbunga dengan optimal. Dari beragam genus anggrek yang ada, terdapat beberapa yang paling umum dibudidayakan, terlebih sebagai tanaman hias, yaitu diantaranya genus *Cattleya, Dendrobium, Oncidium, Phalaenopsis* dan *Vanda*[1]. Berikut adalah klasifikasi ilmiah tumbuhan anggrek secara umum dari *Kingdom* hingga *Genus:*

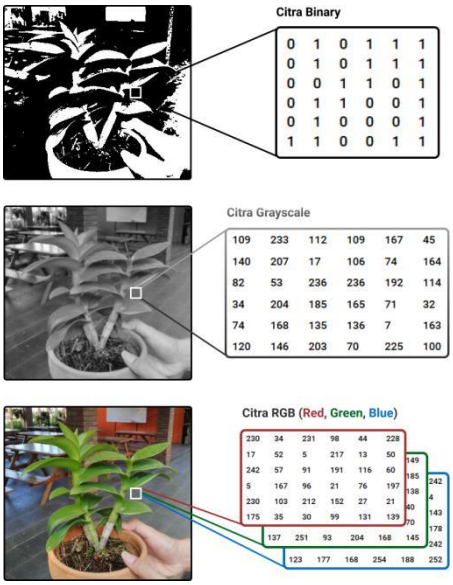


Gambar 1. Diagram klasifikasi anggrek dari Kingdom hingga Genus

2.2 Citra Digital & Tipenya

Citra adalah suatu representasi (gambaran), kemiripan, atau imitasi dari suatu objek. Citra sebagai *output* suatu sistem perekeman data dapat bersifat optik berupa foto, bersifat analog berupa sinyal-sinyal video seperti gambar pada monitor televisi, atau bersifat digital yang dapat langsung disimpan pada suatu media penyimpanan. Citra analog adalah citra yang bersifat kontinu, seperti gambar pada monitor televisi, foto sinar X, foto yang tercetak di kertas foto, lukisan, pemandangan alam, dan lain sebagainya. Citra analog tidak dapat direpresentasikan dalam komputer sehingga tidak bisa diproses di komputer secara langsung — perlu dilakukan proses konversi dari analog ke digital. Sedangkan citra digital ialah citra yang dapat diolah langsung oleh komputer[3]. Sebuah citra digital diwakili oleh sebuah matriks 2 dimensi, yang terdiri dari *M* kolom dan *N* baris yang disebut piksel (*picture element*) sebagai elemen terkecil dari sebuah citra. Setiap piksel menampung sebuah nilai yang kemudian akan merepresentasikan warna yang tampil di layar monitor. Beberapa jenis citra digital yaitu citra *binary*, *grayscale* dan citra warna (RGB). Berikut penjelasannya:

1. Citra *binary* (monokrom)
   Memiliki 2 warna, yaitu hitam dan putih, dimana dibutuhkan 1 bit di memori untuk menyimpan kedua warna ini, bit 0 untuk warna putih dan bit 1 untuk warna hitam.
2. Citra *grayscale* (skala keabuan)
   Disetiap pikselnya mewakili derajat keabuan dengan nilai antara 0 (hitam) sampai 255 (putih), pada jangkauan nilai 0 sampai 255, ini berarti bahwa setiap piksel memiliki ukuran 8 bit atau 1 byte.
3. Citra warna RGB (*true color*)
   Disetiap piksel memiliki 3 komponen warna, yaitu merah (R), hijau (G) dan biru (B). Setiap komponen warna memiliki jangkauan nilai antara 0 sampai 255 (8 bit). Warna pada piksel ditentukan dari kombinasi merah, hijau dan biru. Hal ini akan memberikan kemungkinan total warna sebanyak 255^3=16.581.375. Total ukuran bit untuk setiap piksel adalah 24 bit (8 bit R, 8 bit G dan 8 bit B). Citra seperti ini biasanya juga disebut dengan citra warna 24 bit.

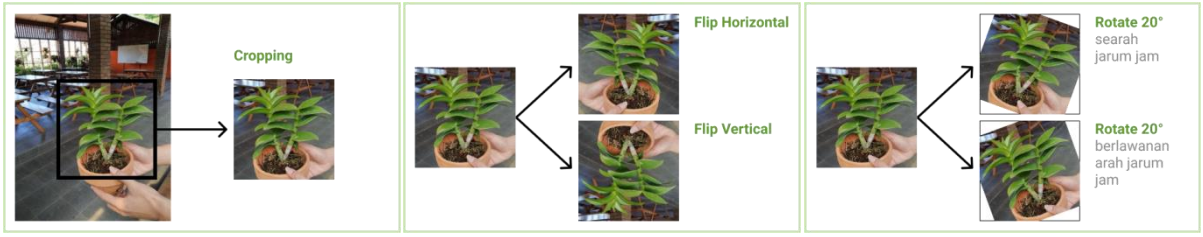


Gambar 2. Citra binary, grayscale dan RGB

2.3 Pengolahan Citra Digital

Pengolahan citra digital (*digital image processing*) atau biasa disingkat dengan PCD merupakan sebuah disimplin ilmu yang mempelajari hal-hal yang berkaitan dengan perbaikan kualitas citra (peningkatan kontras, transformasi warna, restorasi citra), transformasi/augmentasi citra (translasi, skala, transformasi geometri — *flipping*, *rotation*, *cropping*, *scalling*, *zooming*), melakukan pemilihan ciri sebuah citra (*features image*) yang optimal untuk tujuan analisis, melakukan proses penarikan informasi atau deskripsi objek atau pengenalan objek yang terkandung pada citra, dan melakukan kompresi atau reduksi data untuk tujuan penyimpanan data. *Input* dari PCD adalah citra, sedangkan *output*nya ialah citra hasil pengolahan/modifikasi[3].

Pengolahan citra digital diperlukan dengan alasan banyak citra yang belum sesuai dengan harapan — perlu dimodifikasi agar sesuai dengan yang diharapkan. Hal ini dapat terjadi karena beberapa kemungkinan, misalnya terdapat *noise* atau karena lensa kamera tidak bersih. Dalam aplikasinya sehari-hari, misalnya di bidang perfilman, PCD dimanfaatkan untuk menghaluskan gambar, menajamkan gambar, memberi efek terang dan gelap, memberi kesan timbul, dan lain-lain. Di bidang fotografi, PCD dimanfaatkan untuk pengganti kamera filter[3]. Dan di bidang *Data Science* terlebih di *Deep Learning* metode *Convolutional Neural Network* atau yang biasa disebut CNN, PCD banyak dimanfaatkan untuk keperluan augmentasi citra, dengan tujuan menghasilkan citra yang baru sehingga menambah jumlah citra agar lebih bervariasi. Beberapa operasi augmentasi citra yang biasa dilakukan sebagai berikut:

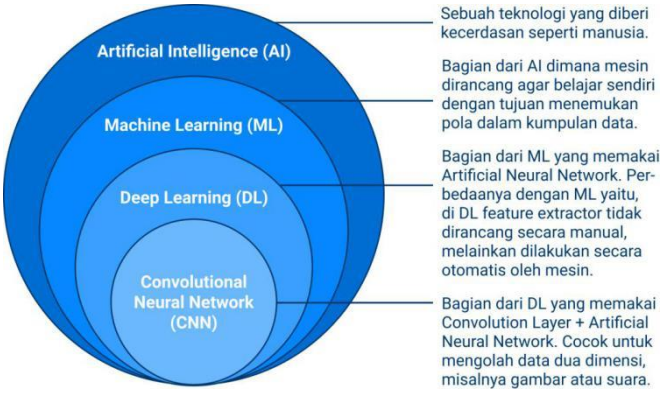


Gambar 3. Operasi Cropping, Flip, Rotate, Resize dan Remove Background

1. Pemotongan (*cropping*)
   Operasi *cropping* merupakan pengolahan citra dengan kegiatan memotong satu bagian dari citra.
2. Pencerminan (*flipping*)
   Operasi *flipping* mengakibatkan adanya perubahan orientasi citra, baik secara horizontal, vertikal, maupun keduanya.
3. Rotasi (*rotating*)
   Operasi *rotating* memutar citra dalam derajat terhadap titik pusatnya, baik searah jarum jam maupun berlawanan arah jarum jam.
4. Mengubah Resolusi (*resizing*)
   Resizing dimaksudkan untuk mengubah resolusi citra sesuai dengan yang diinginkan.
5. Menghapus *Background* (*remove background*)
   Remove Background di beberapa kasus diperlukan dengan tujuan untuk memisahkan citra *foreground* yang menjadi objek utama dari citra, dan menghapus citra *background* sebagai *noise*.

2.4 Convolutional Neural Network (CNN)

Sebelum mengenal apa itu *Convolutional Neural Network* (CNN) ada baiknya dimulai dari mengenal apa itu *Artificial Intelligence* (AI), *Machine Learning* (ML), *Deep Learning* (DL) dan *Artificial Neural Network* (ANN) terlebih dahulu, karena keempat istilah tersebut berkaitan secara langsung dengan CNN dan penting untuk diketahui. Berikut penjelasan secara singkatnya:



Gambar 4. Hubungan AI, ML, DL dan CNN

1. *Artificial Intelligence* (AI)

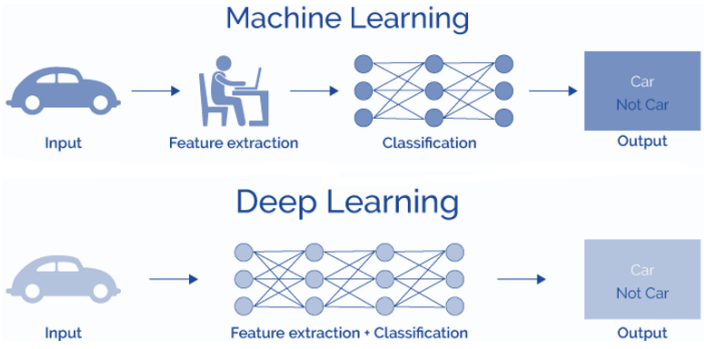
AI atau kecerdasan buatan merupakan bagian dari Ilmu Komputer (*Computer Science*) yang mempelajari cara-cara untuk membangun program dan mesin cerdas yang secara kreatif dapat memecahkan suatu masalah. AI dapat meningkatkan efisiensi dan produktivitas dengan mengotomatiskan tugas yang berulang. Beberapa aplikasi AI misalnya *Self-Driving Cars*, Google Translate, Google Maps, dan Amazon's Alexa[17].

2. *Machine Learning* (ML)
ML atau pembelajaran mesin adalah bagian dari AI dimana program atau mesin menggunakan sekumpulan algoritma yang dirancang sedemikian rupa sehingga mesin mencoba untuk belajar sendiri tanpa diprogram secara eksplisit pada setiap instruksi, dengan tujuan untuk menemukan pola dalam kumpulan data. ML dapat dikategorikan menjadi tiga subset yaitu *Supervised*, *Unsupervised* dan *Reinforcement Learning*. Beberapa aplikasi ML misalnya sistem rekomendasi Amazon, deteksi penipuan dalam transaksi, fitur menyarankan teman di sosial media[17].

Tabel 1. Machine Learning: Supervised, Unsupervised dan Reinforcement

Kriteria	**Supervised Learning**	**Unsupervised Learning**	**Reinforcement Learning**
Definisi	Mesin belajar dengan menggunakan data berlabel	Mesin dilatih pada data tak berlabel, tanpa panduan apapun	Agen melakukan tindakan & belajar dari *errors/rewards*
Jenis Masalah	Regresi & Klasifikasi	Asosiasi & Clustering	*Reward-based*
Jenis Data	Data berlabel	Data tak berlabel	*No Predefined Data*
Training	Diawasi	Tidak diawasi	Tidak diawasi
Pendekatan	Memetakan *input* berlabel ke *output* yang diketahui	Memahami pola & menemukan *output*	Mengikuti metode *trial-and-error*

3. *Deep Learning* (DL)
DL atau pembelajaran mendalam adalah bagian dari ML yang memanfaatkan jaringan saraf tiruan, atau biasa disebut *Artificial Neural Network* (ANN). Disebut sebagai *deep* karena jumlah *hidden layer* maupun *neuron* didalamnya bisa banyak sekali. Algoritma DL dilatih untuk mengidentifikasi pola dan mengklasifikasikan berbagai jenis informasi untuk memberikan *output* yang diinginkan ketika menerima *input* baru. Perbedaan utama dari ML dan DL yaitu di ML *feature extractor* perlu disediakan/dirancang secara manual, yang mana ini memakan banyak sekali waktu dan tenaga. Sedangkan DL akan secara otomatis mengekstrak *features* untuk klasifikasi. Namun di sisi lalin, DL menuntut sejumlah data yang besar untuk melatih algoritmanya. Beberapa aplikasi DL diantaranya meng-*generate* suara ke sebuah video dan klasifikasi citra[17,18].



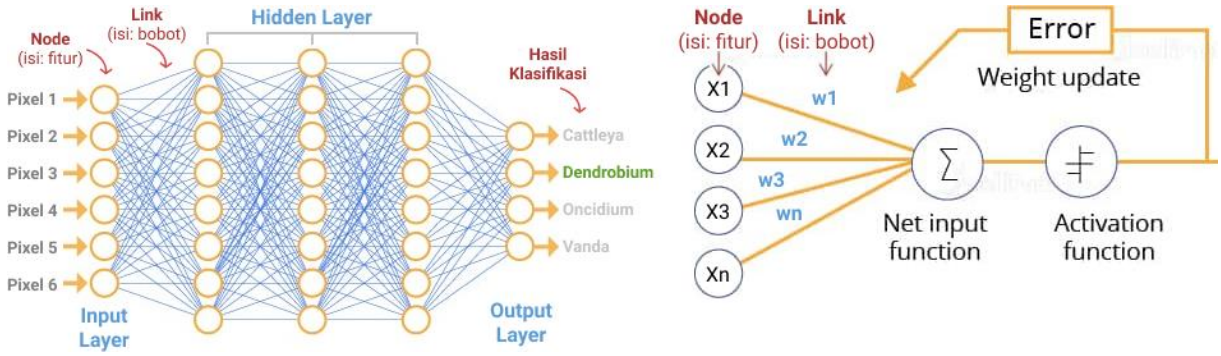
Gambar 5. Machine Learning vs Deep Learning

Sumber Gambar: https://talks.navixy.com/

4. *Artificial Neural Network* (ANN)
ANN atau jaringan saraf tiruan adalah konsep yang terinspirasi oleh jaringan saraf biologis, terdiri dari tiga lapisan (*layer*), yang mana masing-masing *layer* ini terdiri dari *node* yang saling terhubungan dan berinteraksi satu sama lain melalui *link*:

- Input Layer: Digunakan untuk mengambil data masukan dari sumber eksternal, *layer* ini tidak melakukan perhitungan apapun.
- Hidden Layer: Terdiri dari satu, dua atau banyak (*deep*) *hidden layer*. Semua komputasi/perhitungan dilakukan di *hidden layer*.
- Output Layer: Digunakan untuk perhitungan yang memberikan/menghasilkan *output*, misalnya berupa hasil klasifikasi.

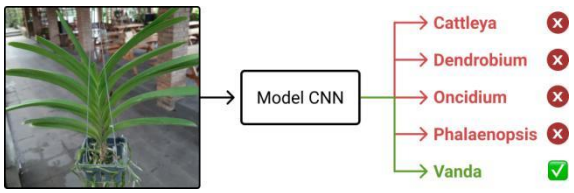
Setiap *node* berisikan fitur (*feature*), *node* yang berada didepan berisi fitur baru yang dihasilkan dari kombinasi fitur-fitur dari *node* sebelumnya. *Link* yang menghubungkan antar *node* berisikan sebuah nilai yang disebut bobot (*weight*), dimana bobot ini pada awalnya diinisialisasi secara acak kemudian akan diperbarui melalui sebuah skema yang disebut *training loop* (yaitu : *feed forward* → hitung *error* → *backpropagation* → *update* bobot dengan *optimizer*) secara bertahap hingga mendapatkan *error* (atau disebut juga *cost*, yakni total dari keseluruhan *loss*)[39] yang kecil. Satu *training loop* terhadap seluruh dataset kemudian disebut sebagai satu *epoch*.



Gambar 6. (kiri) Arsitektur ANN, (kanan) Proses update bobot di ANN

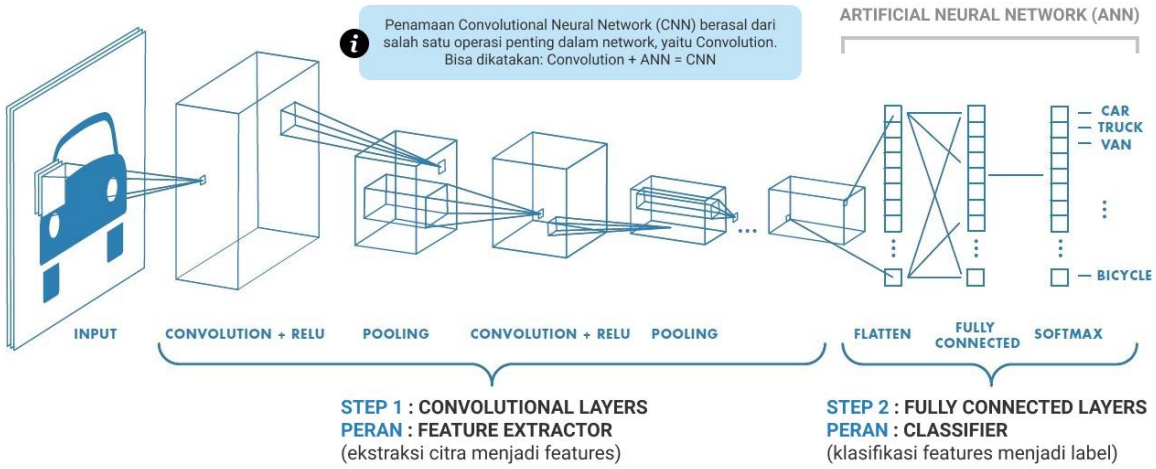
Sumber Gambar: https://intellipaat.com/

Convolutional Neural Network (CNN) merupakah salah satu algoritma DL yang dirancang untuk mengolah data dalam bentuk *grid*/data dua dimensi, misalnya gambar atau suara[7]. CNN digunakan untuk mengklasifikasikan data yang berlabel dengan menggunakan metode *supervised learning*, dimana terdapat data yang dilatih dan terdapat *variable* yang ditargetkan sehingga tujuan dari metode ini adalah mengelompokan suatu data ke data yang sudah ada. CNN sering digunakan untuk mengenali benda atau pemandangan, melakukan deteksi, segmentasi objek dan klasifikasi citra[9].



Gambar 7. Klasifikasi citra dengan CNN

Secara umum tahapan klasifikasi citra di CNN dibagi menjadi dua bagian besar yaitu *feature extractor* dan *classification/fully connected* (ANN). Dimana tahap *feature extractor* berperan melakukan ekstraksi dari sebuah citra (*image*) menjadi sebuah *features* berupa angka-angka yang merepresentasikan citra tersebut, atau dengan kata lain *input* berupa citra dan *output* berupa *features*[7]. Selanjutnya *features* yang dihasilkan dari tahap *feature extractor* ini masih berbentuk *array* multidimensi, sehingga sebelum masuk sebagai *input* ke tahap *classification/fully connected* (ANN) untuk dilakukan klasifikasi, perlu di-*flatten* terlebih dahulu yaitu mengubah bentuk *array* multidimensi tersebut kedalam sebuah *vector* (*array* satu dimensi).



Gambar 8. Tahapan klasifikasi citra di CNN (Arsitektur CNN)

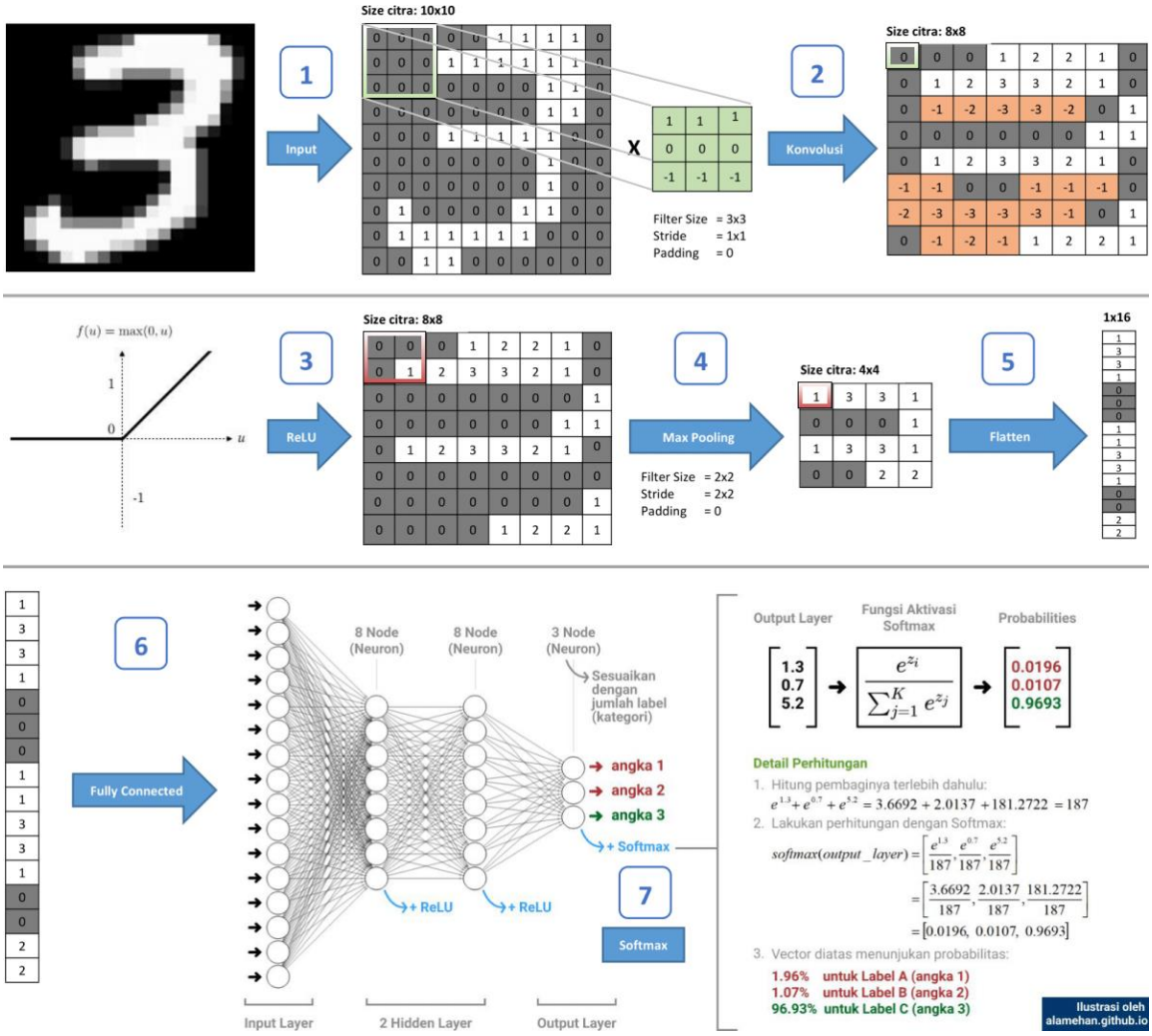
Sumber Gambar: https://towardsdatascience.com/

Proses pembelajaran (atau disebut juga pelatihan) di CNN sama seperti di ANN yaitu melalui skema *training loop* (*feed forward* → hitung *error* → *backpropagation* → *update* bobot dengan *optimizer*), dimana satu *training loop* terhadap seluruh dataset disebut sebagai satu *epoch*. Proses pelatihan (*training*) menghasilkan sebuah model CNN beserta bobot *updated*nya, yang kelak akan digunakan untuk pengujian (*testing*) dengan data citra yang benar-benar baru. Selanjutnya evaluasi akan dilakukan dengan meninjau nilai (*score*) berupa akurasi klasifikasi dari model CNN. Jika *score* akurasi masih belum baik maka perlu dilakukan *fine-tuning* atau penyesuaian di beberapa aspek, misalnya dataset, arsitektur CNN yang digunakan, atau *hyperparameter* yang dipakai. Namun jika *score* akurasi sudah baik, maka *fine-tuning* atau penyesuaian tidak perlu dilakukan, atau bisa saja tetap dilakukan (*experimental*) untuk mendapatkan *score* yang lebih baik lagi.

2.5 Cara Kerja CNN

Secara umum arsitektur CNN terdiri dari beberapa lapis (*layers*), dimulai dari *Input Layer*, *Convolutional Layer*, Fungsi Aktivasi ReLU, *Pooling Layer*, *Flatten Layer*, *Fully Connected Layer* (ANN), Fungsi Aktivasi *Softmax* dan *Dropout*. Masing-masing dari *layer* tersebut memiliki perannya tersendiri dalam klasifikasi citra. Berikut penjelasan mengenai cara kerja CNN (*feed forward*):

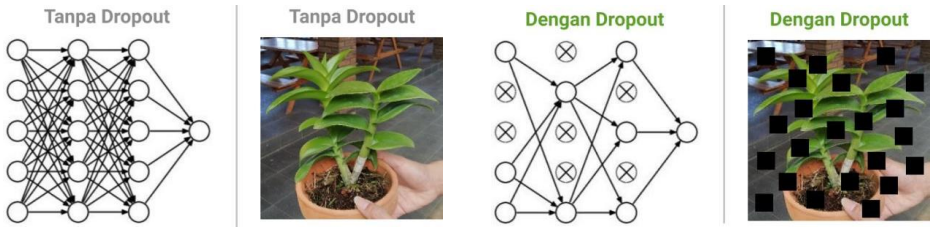
1. Input Layer
   Input Layer mewakili citra masukan ke dalam CNN. Anggap citra yang digunakan sebagai masukan berukuran 240x240 piksel dan berjenis RGB (*Red*, *Green*, *Blue*), maka citra *input* ini berupa *array* multidimensi dengan ukuran 240x240x3 (3 adalah jumlah *channel* yaitu merah, hijau dan biru).
2. Convolutional Layer
   Convolutional Layer merupakan fondasi dari CNN, berupa kernel/filter yang awalnya memiliki bobot acak, yang mana bobot ini akan di perbarui (*update*) pada saat *training*. Filter ini akan dikalikan dan digeser keseluruh bagian citra, bergerak dari sudut kiri atas ke kanan bawah. Tujuan dari *convolutional layer* ialah untuk mengekstraksi fitur dari citra *input* (misalnya tepi, sudut, tekstur, dll). *Output* dari *convolutional layer* biasa disebut *feature map*.



Gambar 9. Cara kerja CNN (feed forward)

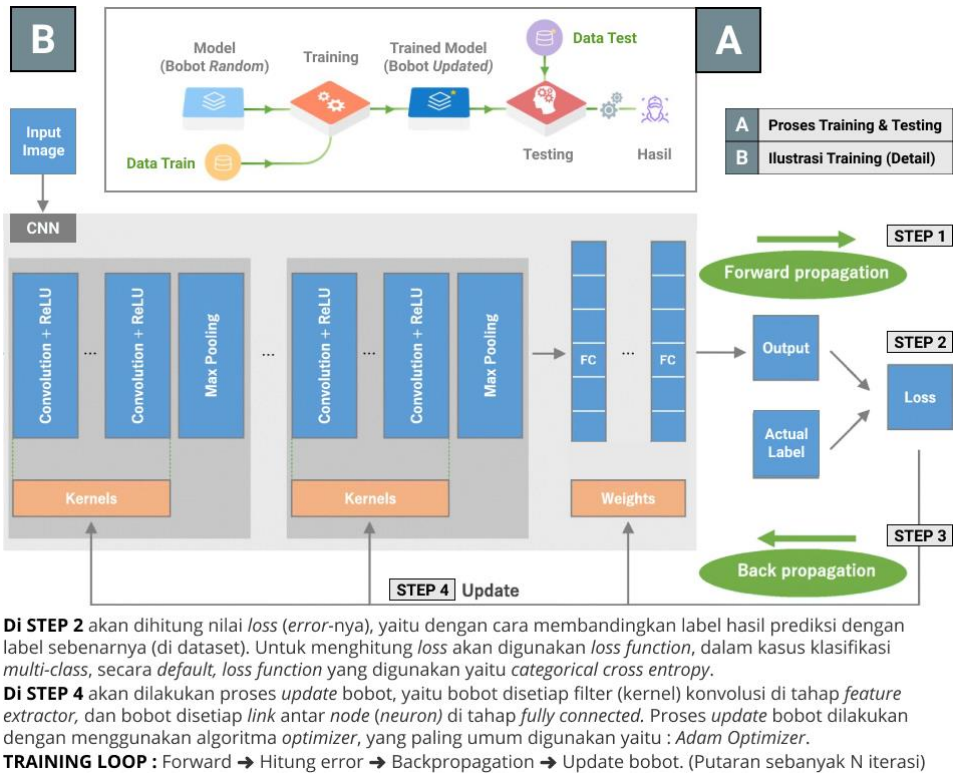
3. Fungsi Aktivasi: ReLU
   Fungsi Aktivasi ReLU (*Rectified Linear Unit*) dipakai setelah melakukan proses konvolusi/sebelum melakukan *pooling* (di tahap *feature extractor*) dan dipakai pula di setiap *node* di *hidden layer* (di tahap *fully connected*). ReLU merupakan lapisan aktivasi dengan fungsi *f*(x) = max(0,x) yang membuat seluruh nilai piksel yang bernilai kurang dari nol pada suatu citra akan dijadikan 0. Tujuan ReLU ialah mengurangi linearitas yang terjadi dari proses konvolusi sehingga CNN lebih mudah mencapai nilai optimum[37].
4. Pooling Layer
   Pooling Layer ialah sebuah filter dengan ukuran tertentu yang bergeser pada seluruh area *feature map*. *Pooling* yang biasa digunakan adalah *Max Pooling* (mengekstrak fitur yang paling penting) dan *Average Pooling*. Tujuan dari *pooling layer* ialah mengurangi dimensi/mereduksi (*downsampling*) *feature map*, sehingga mempercepat komputasi karena bobot yang harus di *update* semakin sedikit.
5. Flatten Layer
   Flatten Layer adalah tahapan sebelum memasuki *Fully Connected Layer*. *Feature map* yang dihasilkan dari tahap *feature extractor* masih berbentuk *array* multidimensi, oleh karena itu perlu dilakukan *Flatten*, yaitu membentuk ulang fitur (*reshape feature map*) menjadi sebuah *vector* (*array* satu dimensi) agar bisa digunakan sebagai *input* ke tahap *fully connected/classification*.
6. Fully Connected Layer
   Fully Connected Layer terdiri dari *Input Layer*, *Hidden Layer* dan *Output Layer*, jika diperhatikan *layer-layer* ini sama seperti di ANN. Oleh karena itu beberapa literatur menyebut *fully connected layer* ini sebagai ANN. Di *fully connected layer* semua *node* (*neuron*) dari *layer* sebelumnya terhubung menyeluruh dengan *node* di *layer* selanjutnya. Disetiap *hidden layer* terdapat fungsi aktivasi, yang umum digunakan yaitu ReLU, dan begitu pula di *output layer*, juga tedapat fungsi aktivasi, dalam kasus klasifikasi (yang lebih dari 2 label/kelas/kategori) yang umum digunakan yaitu *Softmax*. Tujuan utama dari *fully connected layer* ialah mengolah data sehingga bisa dilakukan klasifikasi. *Output* dari *fully connected layer* yaitu probabilitas terhadap kategori (jika menggunakan *Softmax*).

7. Fungsi Aktivasi: Softmax
   Fungsi Aktivasi *Softmax* dipakai dalam kasus klasifikasi lebih dari 2 kelas (*multi-class*), posisinya berada di *output layer* di tahap *fully connected/classification*. Tujuan penggunaan *Softmax* yaitu menghitung probabilitas dari setiap kelas target (kategori) atas semua kelas target yang memungkinkan, dan akan membantu untuk menentukan kelas target untuk *input* citra yang diberikan. Kelebihan menggunakan *Softmax* yaitu rentang probabilitas *output* dengan nilai 0 hingga 1, dan jumlah semua probabilitas akan sama dengan satu.
8. Dropout Regularization
   Dropout adalah teknik regularisasi — sebuah teknik untuk mengurangi *overfitting* — *neural network* dimana beberapa *neuron* akan dipilih secara acak dan tidak dipakai selama *training* (dengan kata lain: membuang fitur/mengurangi variansi). *Neuron-neuron* ini dapat dibilang dibuang secara acak. Hal ini berarti bahwa kontribusi *neuron* yang dibuang akan diberhentikan sementara *network* dan bobot baru juga tidak diterapkan pada *neuron* pada saat melakukan *backpropagation.* Selain mengurangi *overfitting*, penggunaan *Dropout* pun mempercepat proses pelatihan (*training*)[14].



Gambar 10. Ilustrasi penggunaan Dropout

2.6 Training & Testing di CNN



Di STEP 2 akan dihitung nilai *loss* (*error*-nya), yaitu dengan cara membandingkan label hasil prediksi dengan label sebenarnya (di dataset). Untuk menghitung *loss* akan digunakan *loss function*, dalam kasus klasifikasi *multi-class*, secara *default, loss function* yang digunakan yaitu *categorical cross entropy*.
Di STEP 4 akan dilakukan proses *update* bobot, yaitu bobot disetiap filter (kernel) konvolusi di tahap *feature extractor*, dan bobot disetiap *link* antar *node* (*neuron)* di tahap *fully connected.* Proses *update* bobot dilakukan dengan menggunakan algoritma *optimizer*, yang paling umum digunakan yaitu : *Adam Optimizer*.
TRAINING LOOP : Forward ➜ Hitung error ➜ Backpropagation ➜ Update bobot. (Putaran sebanyak N iterasi)

Gambar 11. (A) Proses training & testing, (B) Ilustrasi training (detail)
Sumber Gambar: blog.mellanox.com, insightsimaging.springeropen.com

Proses pembelajaran, atau biasa disebut juga sebagai pelatihan (*training*) terhadap dataset citra di CNN mempunyai siklus (*loop*) yaitu: *feed forward* → hitung *error* → *backpropagation* → *update* bobot dengan *optimizer*. Hasil akhir dari tahap *training* yaitu model CNN dengan bobot yang sudah di-*update*. Ingat bahwa diawal bobot ini diinisialisasi secara acak, yang berarti model CNN masih buruk dalam melakukan klasifikasi, kemudian setelah dilakukan *training*, bobot ini akan diperbarui dengan kombinasi nilai yang sedemikian rupa

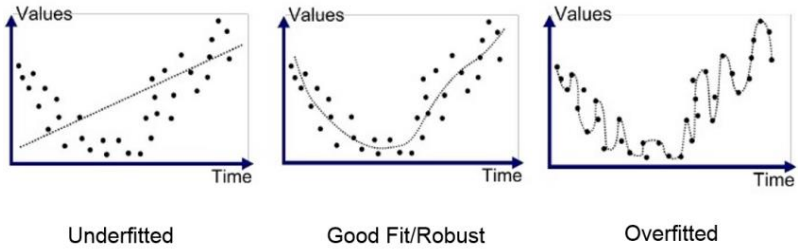
sehingga pada saat model CNN melakukan klasifikasi lagi, akan memberikan hasil prediksi yang lebih baik, acuannya yaitu *score* berupa akurasi.

Sedangkan proses pengujian (*testing*) terhadap data citra baru di CNN hanya melalui tahapan *feed forward* → hitung *error* saja[39], lalu mengeluarkan hasil prediksinya untuk kemudian dilakukan evaluasi. Atau dengan kata lain, pada tahap *testing* ini, model CNN beserta bobotnya yang dihasilkan dari tahap *training*, akan digunakan untuk diujikan terhadap data baru, yaitu data *testing*. Hasil akhir dari tahap *testing* ini yaitu bahan evaluasi berupa *testing accuracy* untuk dilihat seberapa baik model CNN dalam melakukan klasifikasi terhadap data baru. Sebagai tambahan, *loss function* yang digunakan untuk menghitung *loss* dalam masalah klasifikasi yang menggunakan lebih dari 2 kelas (*multi-class*) seperti dalam kasus penelitian ini, secara *default* yaitu *categorical crossentropy*.

2.7 Overfitting & Hyperparameter

Underfitting dan *Overfitting* model adalah hal yang bisa terjadi ketika membuat model CNN. Model yang *underfitting* atau *overfitting* tidak akan melakukan prediksi dengan benar[14]. Berikut penjelasan:

1. Underfitting
   Terjadi ketika model tidak bisa melihat logika dibelakang data, hingga tidak bisa melakukan prediksi dengan tepat, baik untuk dataset *training* maupun dataset *testing*. *Underfitting* model akan memiliki *loss* tinggi dan akurasi rendah.
2. Overfitting
   Terjadi karena model yang dibuat terlalu fokus pada *training* dataset tertentu, hingga tidak bisa melakukan prediksi dengan tepat jika diberikan dataset *testing*. *Overfitting* biasanya akan menangkap data *noise* yang seharusnya diabaikan. *Overfitting* model akan memiliki *loss* rendah dan akurasi rendah.



Gambar 12. Underfitting, Good Fitting dan Overfitting
Sumber Gambar: https://medium.com/greyatom/

Model yang baik adalah model yang bisa menjelaskan data tanpa terpengaruh oleh data *noise*. Model tidak akan fit terhadap tiap data, namun mampu menjelaskan *trend* atau kelompok data. Model yang baik akan memiliki *loss* rendah dan akurasi tinggi. Namun demikian, dalam aplikasinya di CNN, *Overfitting* biasa terjadi, oleh karena itu terdapat beberapa *fine-tuning* atau penyesuain yang bisa dilakukan, salah satunya yaitu penyesuaian *Hyperparameter*.

Hyperparameter adalah sebuah parameter yang nilainya ditentukan oleh perancang model CNN, digunakan untuk mengontrol proses pelatihan (*training*). *Hyperparameter* perlu dilakukan penyesuaian (*tuning*) — *experimental* — untuk mendapatkan hasil *score* akurasi yang baik. Berikut beberapa *hypermarameter* yang bisa disesuaikan oleh perancang model CNN:

1. Input Shape
   Bisa saja model CNN tidak mempelajari *features* di citra dengan baik akibat ukuran citra yang terlalu kecil, sehingga ukuran citra sebagai *input* perlu diperhatikan. Beberapa arsitektur CNN yang terkenal menyarankan ukuran citra (*shape*) yaitu 224x224x3 piksel (3 adalah *channel* RGB).
2. Batch Size
   Pelatihan data (*training*) secara keseluruhan sekaligus itu berat secara komputasi, terlebih jika *dataset training* yang digunakan banyak — misalnya hingga ribuan — dan akan semakin berat lagi jika data berupa data citra. Oleh karena itu *batch size* berperan untuk memecah dataset kedalam beberapa bagian kecil. Sebagai contoh, jika dataset berjumlah 3200 dan *batch size* yang digunakan 64, maka terdapat 50 iterasi untuk satu putaran pelatihan (50x64=3200). Penggunaan *batch size* juga dapat menimimalisir *optimizer* (*gradient*) agar tidak terjebak di *local optimum*, hal ini dikarenakan bobot akan di *update* (melalui skema *training loop*: *feed forward* → hitung *error* → *backpropagation* → *update* bobot dengan *optimizer*) per iterasi bukan per keseluruhan dataset (*epoch*)[37]. Terdapat beberapa nilai yang umum digunakan sebagai *batch size*, yaitu 16/32/64/128/256.
3. Epoch

Satu *epoch* berarti satu putaran penuh pelatihan (*training*) terhadap seluruh *dataset*. Tidak ada keterangan yang pasti terkait berapa jumlah *epoch* yang optimal, karena akan berbeda untuk kumpulan data yang berbeda[26]. Namun sebagai acuan, hentikan *epoch* saat nilai *loss* sudah *converge* (tidak turun lagi) atau nilai akurasi sudah tidak mengalami kenaikan yang signifikan.

4. Jumlah Filter
   Jumlah filter di *Convolutional Layer* bisa disesuaikan dan tidak ada keterangan yang pasti terkait berapa jumlah filter yang optimal, namun sebagai acuan secara umum banyak para peneliti yang menggunakan nilai 2^n (misalnya: 8/16/32/64/128/256).
5. Filter Size
   Ukuran filter di *Convolutional Layer* dan *Pooling Layer* bisa disesuaikan, namun secara umum ukuran filter *Convolutional* yaitu 3x3 atau 5x5 dengan stride=1 atau stride=2, sedangkan untuk *Pooling* yaitu 2x2 dengan stride=2 (ini akan mereduksi/mengecilkan citra sebesar 2x dari ukuran citra aslinya).
6. Stride
   Parameter yang menentukan berapa jumlah pergeseran filter. Jika nilai stride=1, maka filter akan bergeser sebanyak 1 piksel secara horizontal lalu vertikal hingga seluruh area terkena efek filter.
7. Padding
   Parameter yang menentukan jumlah piksel (berisi nilai 0) yang akan ditambahkan di setiap sisi dari *input*. Tujuannya untuk memanipulasi dimensi *output* dari proses *convolutional* atau *pooling*. Dengan menggunakan *padding*, dimensi *output* dapat diukur agar tetap sama seperti dimensi *input* (di *library* Keras memakai: padding=same) atau setidaknya tidak berkurang secara drastis (padding=valid).
8. Optimizer
   Sebuah algoritma atau metode yang digunakan untuk meng-*update* bobot (*weight*) dengan tujuan untuk menurunkan nilai *loss* sehingga menghasilkan *score* akurasi yang baik. Salah satu *optimizer* terbaik dan paling sering digunakan dikalangan para peneliti yaitu *Adam Optimizer*, dianggap sebagai *optimizer* yang cepat dalam dalam mencapai *loss* minimum (*converge*)[27].
9. Learning Rate
   Parameter yang mengontrol seberapa cepat atau lambat model mempelajari masalah pada saat pelatihan (*training*). Jika *learning rate* terlalu kecil, mungkin diperlukan waktu yang lama untuk mencapai bobot yang optimal. Jika terlalu besar, mungkin bobot yang optimal terlewatkan begitu saja — baca mengenai analogi "*guiding the ball to the hole*"[6]. Terdapat beberapa nilai *learning rate* yang umum digunakan, yaitu 0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3 (lihat: https://playground.tensorflow.org/).
10. Jumlah Hidden Layer
    Jumlah *Hidden Layer* di tahap *Fully Connected* dapat disesuaikan dan tidak ada keterangan yang pasti terkait berapa jumlah yang optimal, namun sebagai acuan mulai dari 2 *hidden layer*[25].
11. Jumlah Node/Neuron
    Jumlah *neuron* di setiap *Hidden Layer* dapat disesuaikan, sebagai acuan gunakan nilai 2^n (misalnya: 8/16/32/64/128/256)[25]. Perlu diperhatikan bahwa semakin banyak jumlah *neuron* yang digunakan, maka semakin besar pula *model size* CNN yang dihasilkan (bahkan bisa sampai ratusan MB, seperti *VGG*).

2.8 Arsitektur CNN: MobileNetV2

Dalam beberapa tahun terakhir muncul berbagai macam arsitektur CNN, masing-masing dari arsitektur ini bersaing untuk mendapatkan *score* akurasi tertinggi yang dilatih terhadap ImageNet. ImageNet itu sendiri ialah sekumpulan *dataset* gambar yang berjumlah sangat besar, yaitu lebih dari 14 juta gambar. Dirancang oleh akademisi yang ditunjukan untuk penelitian *computer vision*. Dari beragam arsitektur CNN yang ada, terdapat beberapa yang populer dan bisa digunakan untuk kebutuhkan penelitian dengan studi kasus *dataset* baru.

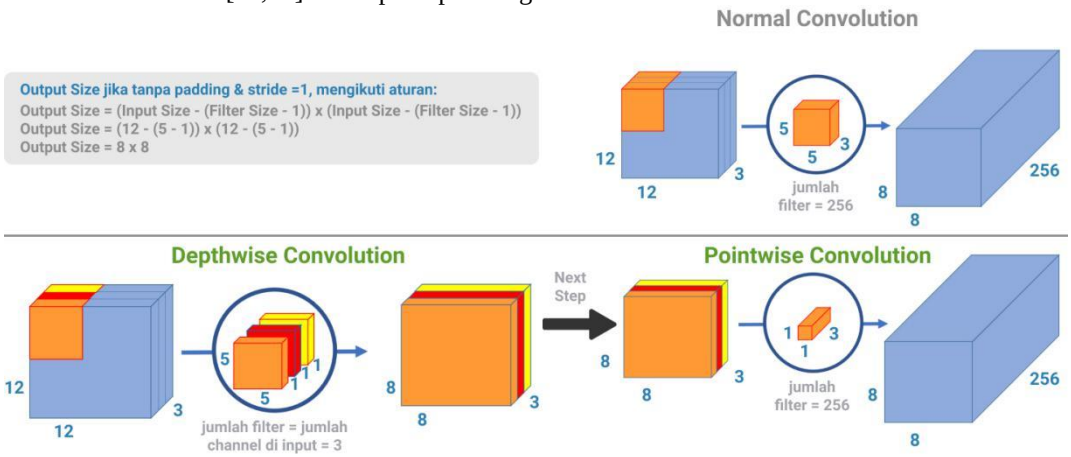
Dalam penelitian ini, penulis memilih arsitektur CNN MobileNetV2. Alasannya, selain memang *score* akurasinya cukup tinggi, juga yang menjadi keunggulan utamanya ialah jumlah *training parameters* yang kecil dibandingkan dengan arsitektur CNN lainnya, sehingga kebutuhan akan komputasinya lebih ringan. Selain itu *model size* MobileNetV2 juga kecil, hanya 14MB saja, namun tentunya dengan performansi yang baik. Sehingga kedepannya jika model tersebut akan di *deploy* kedalam sebuah *real app*, misal aplikasi android ataupun aplikasi berbasis *website* akan ringan dan berukuran kecil.

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
MobileNetV2	14 MB	0.713	0.901	3,538,984	88

Gambar 13. MobileNetV2 di laman website Keras

MobileNet dibuat oleh Google, mempunyai *layer* khusus yang disebut dengan *depthwise separable convolution*, merupakan sebuah blok yang terdiri dari *depthwise convolution* dan *pointwise convolution*. Tujuan dari *layer* ini ialah mereduksi komputasi (agar lebih sedikit parameter) sehingga menghasilkan ukuran model

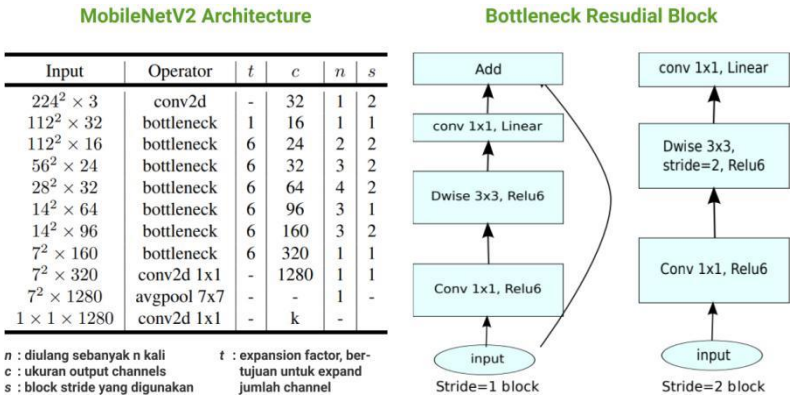
yang lebih kecil[12,13]. Untuk memahami *depthwise separable convolution*, berikut ilustrasi perbandingannya dengan *normal convolution*[30,31]. Terdapat input image berukuran 12x12x3 dan 256 kernel berukuran 5x5x3:



Gambar 14. Normal Convolution vs Depthwise Separable Convolution
Sumber Gambar: https://hackmd.io/

Dalam *normal convolution*, filter melakukan konvolusi terhadap *input image* dan menghasilkan *feature map* (*output image*). Sedangkan dalam *depthwise convolution*, jumlah filter akan sama dengan jumlah *channel* di *input image*, setiap filter akan melakukan konvolusi terhadap masing-masing *channel* di *input image* (atau dengan kata lain, tidak langsung melakukan konvolusi terhadap seluruh *channel* di *input image* seperti yang dilakukan pada *normal convolution*), lalu menghasilkan *feature map* yang memiliki jumlah *channel* yang sama dengan jumlah filter. Tahap selanjutnya, agar *output image* dengan *depthwise convolution* sama dengan *output image* yang dihasilkan dengan *normal convolution*, maka akan dilakukan *pointwise convolution*, dimana hasil dari tahap *depthwise convolution* akan dikonvolusi lagi dengan filter 1x1, filter ini memiliki kedalaman sama dengan jumlah *channel* di *output image* sebelumnya. Berikut perbandingan jumlah komputasi antara *normal convolution* versus *depthwise separable convolution* berdasarkan contoh gambar diatas:

1. Normal Convolution
   Terdapat 256 filter 5x5x3 yang bergeser sebanyak 8x8 kali (8x8 yaitu jumlah pergeseran filter 5x5 dari kiri atas ke kanan bawah terhadap *input image* 12x12). Ini berarti 256x5x5x3x8x8 = 1.228.800 total perkalian yang dilakukan saat proses konvolusi.
2. Depthwise Separable Convolution
   Pertama, pada tahap *depthwise convolution* terdapat 3 filter 5x5x1 yang bergeser sebanyak 8x8 kali. Ini berarti 3x5x5x1x8x8 = 4.800. Selanjutnya, pada tahap *pointwise convolution* terdapat 256 filter 1x1x3 yang bergeser sebanyak 8x8 kali. Ini berarti 256x1x1x3x8x8 = 49.152. Maka total perkalian yang dilakukan saat proses konvolusi di *depthwise separable convolution* ini yaitu 4.800 + 49.152 = 52.952. Dapat dilihat bahwa, jumlah 52.952 hanya kurang lebih 22,7% nya saja dari 1.228.800 di *normal convolution*.



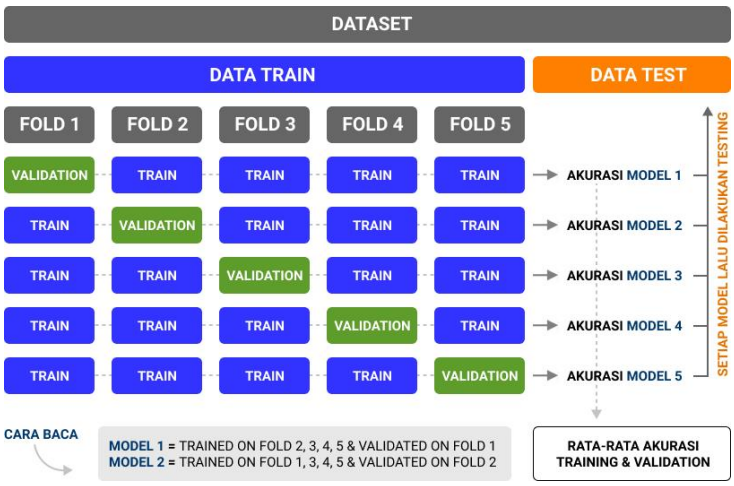
Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Gambar 15. Arsitektur MobileNetV2

Perkembangan lebih lanjut dari MobileNet yaitu MobileNetV2, dimana ditambahkan fitur baru yaitu *bottleneck* yang merupakan sebuah blok dalam arsitektur yang didalamnya menggunakan *depthwise separable convolution*.

2.9 K-Fold Cross Validation

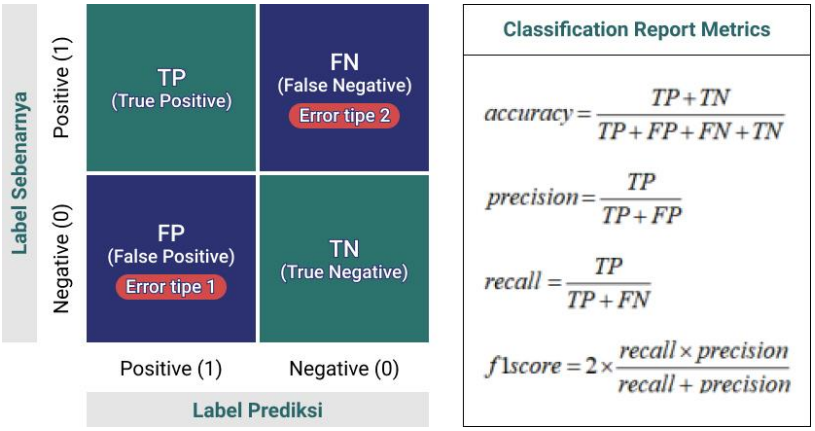
K-Fold Cross Validation merupakan sebuah metode statistik yang dapat digunakan untuk mengevaluasi model dimana data *train* akan di bagi menjadi 2 subset yaitu data *train* dan data *validation*. Model atau algoritma akan dilatih dengan menggunakan data *train* dan akan di validasi dengan menggunakan data *validation*. Teknik ini biasanya dilakukan untuk melakukan prediksi dari model yang akan dilatih agar dapat dilihat seberapa akurat model yang akan digunakan. *K-Folds Cross Validation* membagi subset sebanyak K dan dilakukan perulangan sebanyak K, sehingga model yang dihasilkan dari tahap *training & validation* ini berjumlah K model. Pilihan K biasanya 5 atau 10, tetapi tidak ada aturan formal [34]. Berikut ilustrasi *K-Folds Cross Validation* dengan K=5.



Gambar 16. Ilustrasi 5-Fold Cross Validation
Sumber Gambar: https://satishgunjal.com/kfold/

2.10 Confusion Matrix & Classification Report

Dalam ML/DL, klasifikasi merupakan bagian dari *supervised learning*. Langkah penting dalam *life cycle* model ML/DL adalah evaluasi performanya. Terdapat dua teknik yang dapat digunakan untuk mengevaluasi model klasifikasi yaitu *Confusion Matrix* dan *Classification Report.* Berikut penjelasan lebih detailnya:



Gambar 17. (kiri) Confusion Matrix, (kanan) Rumus metrics di Classification Report

1. Confusion Matrix

Merupakan tabel N x N (di mana N adalah jumlah kelas/label/ kategori) yang berisi jumlah prediksi yang benar dan salah dari model klasifikasi; tujuannya yaitu membandingkan nilai aktual dengan nilai prediksi. Baris matriks mewakili kelas yang sebenarnya, sedangkan kolom mewakili kelas yang diprediksi. Nilai yang dikembalikan oleh *Confusion Matrix* dibagi ke dalam 4 kategori:

- True Positive (TP): Prediksi positif & nilai sebenarnya positif.
- True Negative (TN): Prediksi negatif & nilai sebenarnya negatif.
- False Positive (FP): Prediksi positif & nilai sebenarnya negatif.
- False Negative (FN): Prediksi negatif & nilai sebenarnya positif.

2. Classification Report

Meski hasil yang ditampilkan *Confusion Matrix* sudah detail, namun sebenarnya masih sulit untuk memahami seberapa baik kinerja model dalam melakukan klasifikasi. Oleh karena itu, data dari *Confusion Matrix* dapat digunakan untuk menghitung *metrics* yang dapat mengukur kinerja model, yang kemudian disebut sebagai *Classification Report*[35]. Berikut *metrics* yang digunakan:

- Accuracy: Menggambarkan seberapa akurat model dalam mengklasifikasikan dengan benar.
- Precision: Menggambarkan akurasi antara data yang diminta dengan hasil prediksi model.
- Recall: Menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi.
- F1-Score: Menggambarkan perbandingan rata-rata *Precision* dan *Recall* yang dibobotkan.

Accuracy digunakan ketika TP dan TN lebih penting, sementara *F1-Score* digunakan ketika FN dan FP sangat penting. *Accuracy* dapat digunakan jika distribusi kelas serupa, sedangkan *F1-Score* adalah *metrics* yang lebih baik jika ada kelas yang tidak seimbang. Dalam sebagian besar masalah klasifikasi di kehidupan nyata, terdapat distribusi kelas yang tidak seimbang dan dengan demikian *F1-Score* adalah *metrics* yang lebih baik untuk mengevaluasi model[36].

2.11 Python Untuk CNN & Google Colaboratory

Untuk mengimplementasikan proyek *Deep Learning*, yang mana salah satunya yaitu metode *Convolutional Neural Network* (CNN) diperlukan sebuah bahasa pemrograman yang stabil, fleksibel dan memiliki sekumpulan *tools* yang tersedia, baik itu berupa pustaka (*library)* maupun kerangka kerja (*framework*). Tujuannya agar saat proses implementasi lebih mudah dan cepat, sehingga tidak memakan banyak waktu maupun tenaga. Saat ini, terdapat bahasa pemrograman yang menawarkan itu semua, yaitu Python.

Python merupakan salah satu bahasa pemrograman yang banyak digunakan oleh akademisi maupun umum, dengan beragam tujuan, mulai dari membangun aplikasi berbasis desktop atau *website*, membuat permainan (*games*), hingga untuk keperluan statistik, IoT (*Internet of Things)*, analisis, mengolah dan visualisasi data, serta saat ini umum digunakan untuk membangun proyek AI, salah satunya CNN, dengan dukungan *library* maupun *framework* yang memadai. Berikut beberapa *library* Python yang umum digunakan di CNN:

1. Tensorflow

Merupakan *library* untuk ML/DL, dikembangkan dan dikelola oleh Google. Menawarkan pemrograman yang melakukan berbagai macam tugas ML/DL (regresi, klasifikasi, dll). Dibangun untuk berjalan pada CPU, GPU atau bahkan OS *mobile*. Tensorflow memiliki tingkat fleksibilitas yang tinggi, artinya banyak hal yang dapat dimodifikasi sesuai kebutuhan, namun kurang cocok untuk pemula karena penggunaannya yang tidak mudah.

2. Keras

Merupakan *high-level* API dari Tensorflow, dikembangkan oleh François Chollet, seorang *engineer* di Google. Dirancang untuk manusia, bukan mesin (lihat: https://keras.io/). Artinya mudah dipelajari dan digunakan. Sehingga para peneliti bisa lebih fokus pada eksperimen yang dilakukan bukan pada bagaimana cara menggunakan *library* tersebut. Dengan kata lain, Keras membungkus Tensorflow menjadi blok-blok yang lebih mudah digunakan, terlebih untuk para pemula. Sehingga sesuai dengan jargon Keras yang berbunyi "Mampu beralih dari ide ke hasil secepat mungkin adalah kunci untuk melakukan penelitian yang baik". Contoh pengguannya di CNN yaitu: *one-hot-encode labels* citra, membangun arsitektur CNN, men-*load* arsitektur CNN populer yang tersedia (siap pakai), mem-*plot* model CNN, men-*load* dataset citra kedalam arsitektur CNN, melakukan *training* dan *testing* model CNN sehingga menghasilkan bahan evaluasi berupa *accuracy* dan *loss*, serta menyimpan (*save*) dan mengakses (*load*) model CNN dalam format h5.

3. cv2 (OpenCV)

Digunakan untuk tujuan pengolahan citra digital (*digital image processing*) dan *computer vision*. Contoh pengguannya di CNN yaitu membaca data citra (jpg, png, dll), mengubah data citra menjadi sebuah *array*

agar bisa diolah, mengkonversi citra (misal dari RGB ke *grayscale*), dan *resize* citra (misal dari 500x500 piksel diubah menjadi 224x224 piksel).

4. Sklearn
   Digunakan sebagal *tools* ML dan model statistik berupa *classification, regression, clustering* dan *dimensionality reduction.* Contoh pengguannya di CNN yaitu mengaplikasikan *K-Fold Cross Validation* terhadap dataset, membuat *Confusion Matrix* dan *Classification Report*.
5. Numpy
   Digunakan untuk memproses/mengolah beragam keperluan *array* maupun matriks. Contoh pengguannya di CNN yaitu *reshape* dimensi citra, menghitung rata-rata maupun standar deviasi akurasi model CNN, dan pengolahan berbasis *array*/matriks lainnya.
6. Pandas
   Digunakan untuk memanipulasi dan analisis data. Contoh pengguannya di CNN yaitu menyimpan *history* pelatihan (berupa *accuracy* dan *loss*) kedalam sebuah format tabel CSV.
7. Matplotlib
   Digunakan untuk membuat visualisasi data. Contoh pengguannya di CNN yaitu: mem-*plot* dataset beserta labelnya dan mem-*plot* hasil *training* maupun *testing* model CNN kedalam sebuah grafik.
8. Seaborn
   Digunakan untuk membuat visualisasi data yang lebih menarik dan informatif berdasarkan Matplotlib. Contoh pengguannya di CNN yaitu membuat *heatmap* untuk *Confusion Matrix*, tujuannya agar lebih informatif sehingga mudah untuk dilakukan analisis.

Untuk keperluan *deploy* model CNN kedalam sebuah *real app* berupa aplikasi *website*, dengan tujuan agar bisa digunakan secara luas, mudah dan praktis, Flask bisa menjadi salah satu solusinya. Flask merupakan sebuah *web framework* yang ditulis dengan bahasa pemrograman Python. Dengan demikian semua lingkungan kerja (*environment*) dan *library* Python untuk CNN yang telah diuraikan sebelumnya (Tensorflow, Keras, dst) bisa digunakan kembali di Flask, seperti *load* model CNN dan *testing*. Atau dengan kata lain kegunaan Flask pada penelitian ini yaitu untuk *deploy* model CNN (format .h5) yang sebelumnya sudah dihasilkan, kedalam sebuah aplikasi berbasis *website* dengan tampilan yang *user-friendly* agar bisa digunakan secara luas, mudah dan praktis. Terdapat sebuah *open source project* — sebuah proyek yang bebas digunakan, dimodifikasi, dipublikasikan untuk beragam tujuan — yaitu *image classifier app template* berbasis Flask yang dikembangkan oleh seorang pengguna *platform* Github bernama Xin Fu (lihat: https://github.com/mtobeiyf/) yang dapat digunakan untuk proses *deploy* model CNN ke sebuah *website*. Dengan menggunakan *template* dari Xin Fu ini, proses *deploy* model CNN akan mudah untuk dilakukan, karena hanya perlu sedikit modifikasi sesuai kebutuhan, dan *load* model CNN yang dihasilkan pada tahap penelitian.

Google Colaboratory atau kemudian biasa disingkat Colab adalah lingkungan pengembangan Jupyter Notebook (Python) berbasis *cloud* gratis yang memungkinkan para pelajar, ilmuwan data (*data scientist*), atau peneliti AI (*AI researcher*) untuk membangun, melatih dan menguji model *Machine Learning* dan *Deep Learning* di CPU, GPU, dan TPU. Colab Memiliki 3 keunggulan utama, yaitu tidak perlunya konfigurasi (*zero configuration required*), akses gratis ke GPU (*free access to GPUs*) dan dapat berbagi ke sesama rekan penelitian atau siapapun dengan mudah (*easy sharing*). Dengan keunggulan tersebut, pertama yang diperlukan untuk memulai hanyalah akun Google dan *internet browser*, lalu tanpa konfigurasi apapun, Colab bisa langsung digunakan dengan mudah. Kedua, kebutuhan akan komputasi yang tinggi dari DL dapat teratasi karena adanya GPU yang disediakan Google secara gratis. Dalam pilihan tanpa berbayar/gratis, Colab memberikan maksimal 12 jam/hari waktu penggunaan *notebook*, dengan 12GB RAM dan 100GB Disk. Ketiga, berbagi dengan mudah.

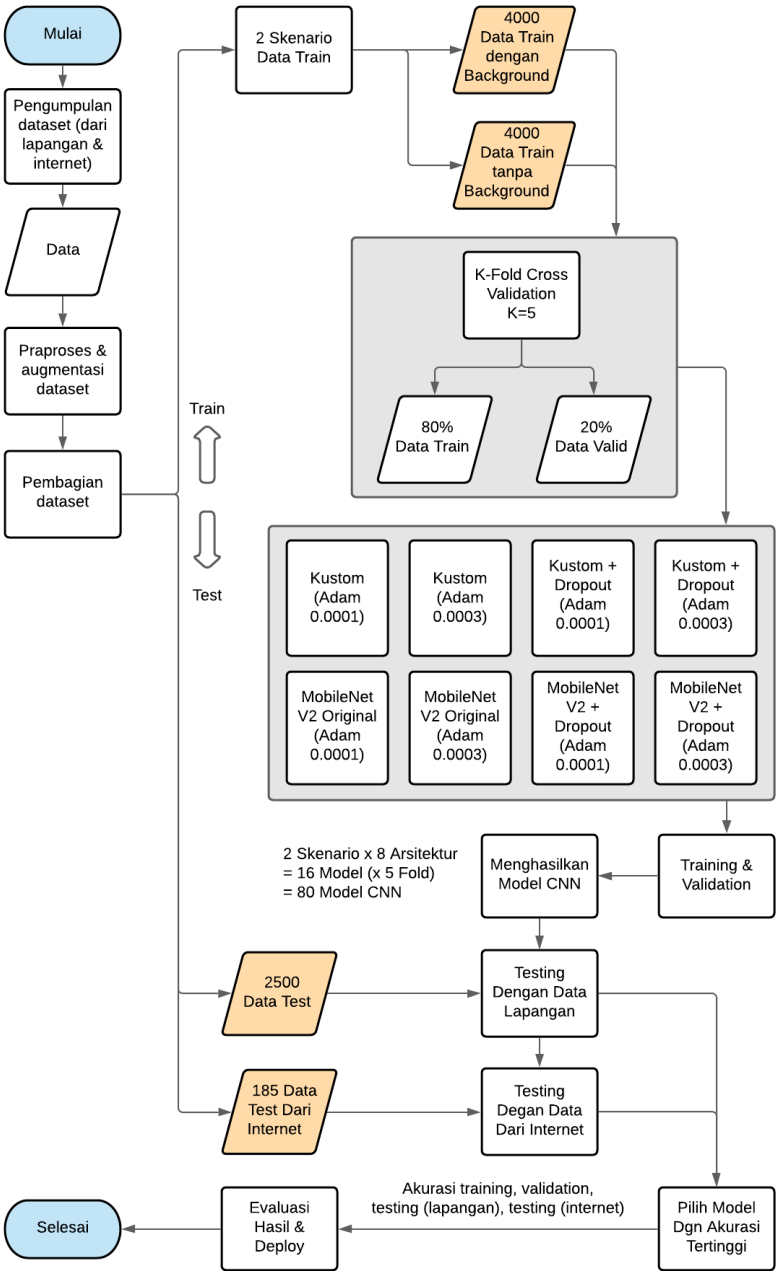
3. Sistem yang Dibangun

3.1 Deskripsi Umum

Sistem ini dirancang dengan tujuan untuk mengklasifikasi genus tanaman anggrek berdasarkan citra *input* menggunakan teknologi *Deep Learning* metode *Convolutional Neural Network* (CNN). Genus tanaman anggrek yang akan diklasifikasikan terdiri dari 5 kelas yaitu *Cattleya*, *Dendrobium*, *Oncidium*, *Phalaenopsis* dan *Vanda*.

Secara keseluruhan, proses klasifikasi citra tanaman anggrek ini melalui beberapa tahapan. Pertama, dilakukan pengumpulan dataset citra tanaman anggrek baik secara langsung dari lapangan maupun mengunduh melalui internet. Kedua, akan dilakukan praproses dan augmentasi dataset agar data citra sesuai dengan kriteria standar saat memasuki arsitektur CNN. Proses augmentasi citra dilakukan untuk menambah jumlah citra (menambah variansi). Selain itu terdapat skenario dataset (*train)* orisinal yaitu dengan *background noise*, dan skenario dataset (*train*) tanpa *background noise.*

Praproses dan augmentasi dataset ini dilakukan dengan harapan menghasilkan model CNN baik. Tahap selanjutnya, yang ketiga, dataset akan dipecah menjadi *data train* dan *data test*, dimana *data train* nantinya akan digunakan untuk melatih model dan *data test* nantinya akan digunakan untuk menguji model. Keempat, *data train* akan dipecah lagi menjadi 80% *data train* dan 20% *data validation.* Dimana pemecahan *data train* ini menggunakan *K-Fold Cross Validation* dengan nilai K=5, tujuannya untuk mengevaluasi model. Kelima, akan dilakukan perancangan arsitektur CNN. Pada penelitian ini akan digunakan 4 tipe arsitektur CNN, yaitu arsitektur kustom, arsitektur kustom dengan tambahan *dropout*, arsitektur MobileNetV2 orisinal, dan arsitektur MobileNetV2 dengan tambahan *dropout*. Setiap arsitektur akan dilatih (*training & validation*) terhadap skenario dataset dengan *background noise* dan skenario dataset tanpa *background noise.* Selain itu, juga akan diterapkan *hyperparameter* berupa *learning rate* dengan nilai 0.0001 dan 0.0003, sehingga *output* yang dihasilkan nanti berjumlah 16 model untuk setiap *fold* (16 x 5 *fold* = 80 total model CNN), yang selanjutnya masing-masing model ini akan diujikan (*testing*) terhadap *data test* lalu mengevaluasi hasilnya, memilih model terbaik, dan terakhir men-*deploy*nya ke aplikasi siap pakai.



Gambar 18. Flowchart sistem klasifikasi genus tanaman anggrek

3.2 Pengumpulan Dataset

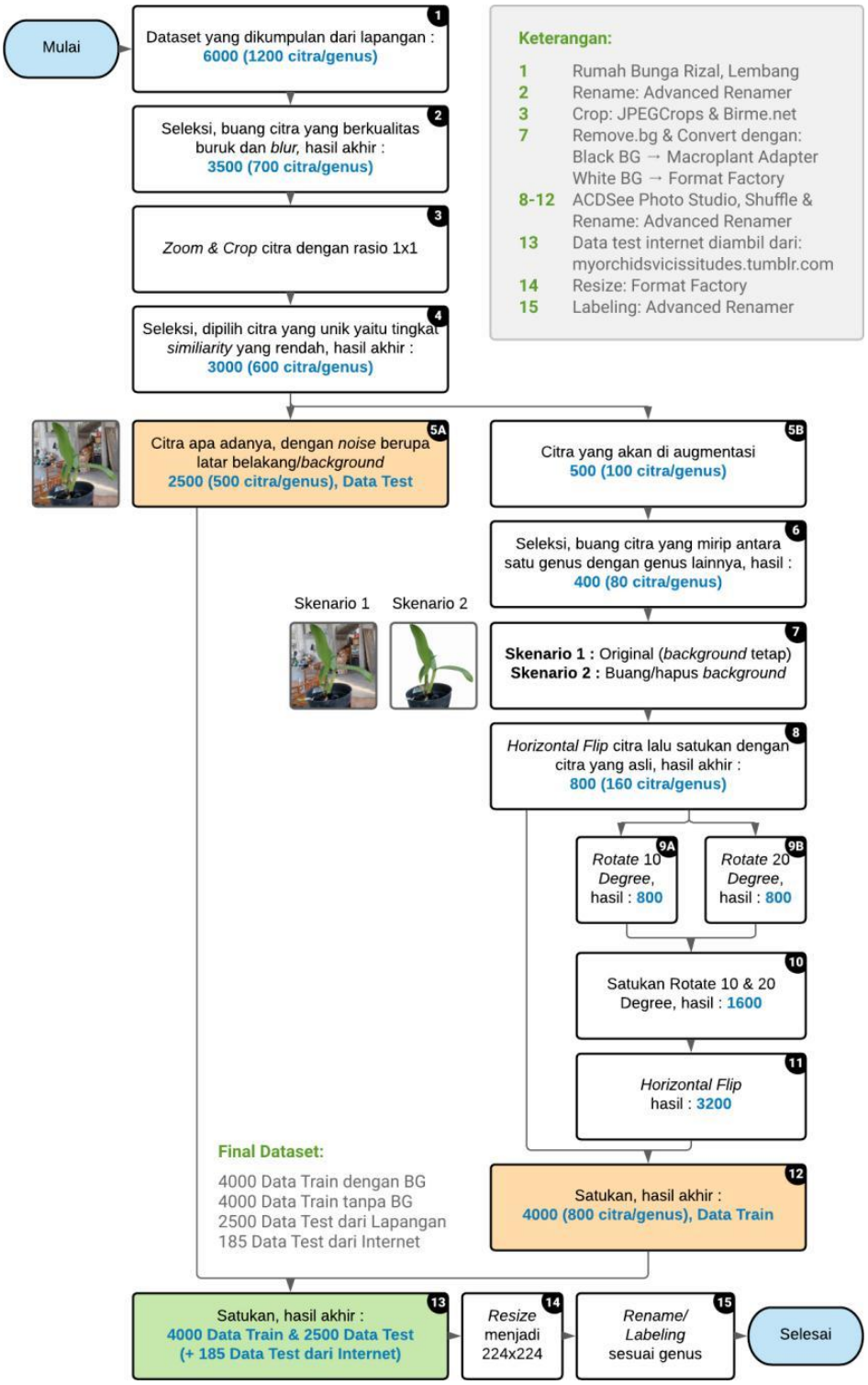
Dataset yang digunakan dalam penelitian ini berupa gambar/citra dari lima genus tanaman anggrek yaitu *Cattleya*, *Dendrobium*, *Oncidium*, *Phalaenopsis* dan *Vanda*. Dataset diperoleh dengan dua cara yaitu:

1. Mengambil dataset secara langsung di lapangan
   Penulis ditemani dosen pembimbing dan beberapa rekan penelitian mengunjungi tempat budidaya tanaman anggrek yaitu Rumah Bunga Rizal Lembang untuk mengambil citra tanaman anggrek secara langsung. Hasilnya diperoleh total 6000 citra, yang terdiri dari 1200 citra untuk setiap genus (*Cattleya*, *Dendrobium*, *Oncidium*, *Phalaenopsis* dan *Vanda*). Selanjutnya data yang berhasil dikumpulkan ini akan digunakan sebagai *data train* dan *data test dari lapangan*.
2. Mengunduh dataset melalui internet
   Penulis mengunjungi beberapa *website* yang menyediakan citra tanaman anggrek seperti https://myorchidsvicissitudes.tumblr.com/, kemudian mengunduhnya. Hasilnya diperoleh total 185 citra,

yang terdiri dari 25 citra *Cattleya*, 40 citra *Dendrobium*, 40 citra *Oncidium*, 40 citra *Phalaenopsis* dan 40 citra *Vanda*. Selanjutnya data yang berhasil dikumpulkan dari internet ini akan digunakan sebagai *data test dari internet*, ini berarti data ini benar-benar baru, diluar dari dataset yang dikumpulkan secara langsung di lapangan.

3.3 Praproses dan Augmentasi Dataset

Sebelum dilakukan pembagian dataset menjadi *data train* dan *data test*, terlebih dahulu akan dilakukan praproses dan augmentasi dataset. Tahapannya yaitu pertama-tama dataset yang berjumlah 6000 (1200 citra/genus) akan diseleksi, untuk citra yang berkualitas buruk dan *blur* akan dibuang karena khawatir akan menggangu proses pembelajaran model CNN nantinya. Hasilnya menjadi 3500 (700 citra/genus). Kedua, citra akan di *zoom* dan *crop* ke bagian fitur inti dari citra tanaman anggrek dengan rasio 1x1. Ketiga, akan dilakukan proses seleksi citra lagi, dipilih citra yang unik yaitu tingkat *similiarity* rendah terhadap citra lainnya yang hampir serupa, hasilnya menjadi 3000 (600 citra/genus).



Gambar 19. Tahapan praproses dan augmentasi dataset

Sampai di tahap ini, dari 3000 (600 citra/genus) akan dipecah menjadi 2500 citra yang tetap apa adanya (akan digunakan nanti sebagai *data test dari lapangan*), dan 500 citra sisanya akan diterapkan proses augmentasi (akan digunakan nanti sebagai *data train*). Selanjutnya, tahap ke empat, dari 500 (100 citra/genus) yang akan diaugmentasi, terlebih dahulu akan diseleksi lagi, beberapa citra yang memiliki tingkat *similiarity* yang cukup tinggi antara satu genus dengan genus lainnya akan dibuang. Hal ini dilakukan agar pada saat proses *training* nanti model tidak keliru antara genus yang satu dengan genus lainnya hanya karena kemiripan fitur. Hasilnya menjadi 400 (80 citra/genus) yang benar-benar unik. Kelima, akan dibuat dua skenario yaitu citra yang apa adanya (terdapat *background noise*) dan citra yang akan dihapus latar belakang/*background noise*nya. Tujuan penghapusan *background noise* ini agar citra yang tampak benar-benar berisi fitur dari tanaman anggrek itu

sendiri, tanpa *noise*. *Tool*s yang digunakan yaitu https://www.remove.bg/. Keenam, dari dua skenario tadi masing-masing akan diterapkan augmentasi yaitu *horizontal flip*, *rotate* 10 *degree*, dan *rotate* 20 *degree*. Hasil akhir diperoleh total 4000 (800 citra/genus) yang unik.

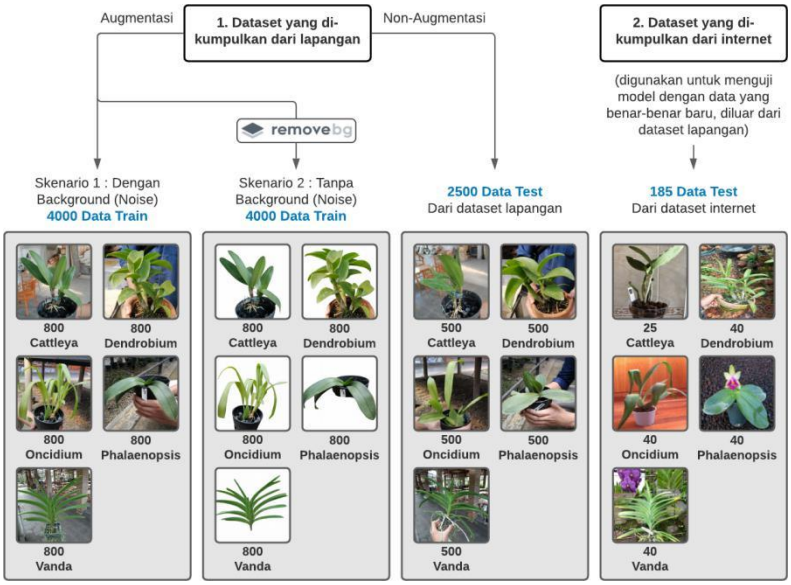
Sampai di tahap ini, 4000 (800 citra/genus) unik hasil augmentasi akan digunakan sebagai *data train*, lalu 2500 (500 citra/genus) yang tidak diterapkan augmentasi di tahap sebelumnya akan digunakan sebagai *data test dari lapangan*. Kemudian *data train* yang berjumlah 4000 dan *data test dari lapangan* yang berjumlah 2500 ini, akan disatukan dengan *data test dari internet* yang berjumlah 185. Kini hasil akhir diperoleh total 4000 + 2500 + 185 = 6685 citra. Selanjutnya, tahap ketujuh, semua citra tadi akan dikecilkan ukurannya (*resize*) menjadi 224x224 piksel. Hal ini dilakukan untuk menyesuaikan dengan *input shape* di arsitektur CNN. Kedelapan, sebagai langkah terakhir dari praproses dataset, semua citra akan dinamai ulang (*rename/labeling*) sesuai dengan nama kelas/genusnya dengan pola sebagai berikut:

- C001.jpg, C002.jpg, C003.jpg dan seterusnya untuk genus *Cattleya.*
- D001.jpg, D002.jpg, D003.jpg dan seterusnya untuk genus *Dendrobium.*
- O001.jpg, O002.jpg, O003.jpg dan seterusnya untuk genus *Oncidium.*
- P001.jpg, P002.jpg, P003.jpg dan seterusnya untuk genus *Phalaenopsis.*
- V001.jpg, V002.jpg, V003.jpg dan seterusnya untuk genus *Vanda*.

3.4 Pembagian Dataset Menjadi Train dan Test

Dalam proses klasifikasi citra di CNN, dataset dibagi menjadi dua yaitu data latih (*train*) dan data uji (*test*). Merujuk pada uraian sebelumnya, total dataset yang berhasil terkumpul setelah melalui praproses dan augmentasi yaitu 6685 citra, dimana terdiri dari:

- 4000 *data train* dengan *background noise* (800 citra/genus); Skenario 1
- 4000 *data train* tanpa *background noise* (800 citra/genus); Skenario 2
- 2500 *data test dari lapangan* (500 citra/genus)
- 185 *data test dari internet* (25 *Cattleya*, 40 untuk genus lainnya)



Gambar 20. Pembagian data train dan data test

Data *train* digunakan untuk proses pelatihan (*training*) sehingga menghasilkan sebuah model CNN beserta bobot *updated*nya. Sedangkan data *test* digunakan untuk menguji (*testing*) model CNN yang dihasilkan tersebut, untuk dievaluasi. Namun sebelum melakukan *training* dan *testing*, data *train* maupun data *test* ini perlu dipraproses terlebih dahulu, diubah kedalam bentuk *array*, sehingga dapat diolah sesuai kebutuhan. Lebih rincinya diuraikan di subbab berikutnya.

3.5 Praproses Citra Input

Praproses citra *input* dilakukan pada saat implementasi dalam penulisan kode program (*coding*) dengan bahasa pemrograman Python. Tujuan praproses citra *input* yaitu untuk menyesuaikan format citra agar pada saat citra memasuki arsitektur CNN dapat terbaca dengan baik. Tahapannya sebagai berikut:

1. Mengubah data citra (.jpg) kedalam bentuk *array*.
2. Mendapatkan fitur (X) dan label (y) citra, hasilnya menjadi *variable* berupa X_train, y_train, X_test, y_test.
3. Normalisasi (*feture scaling*) untuk X_train dan X_test, yaitu mengubah rentang nilai 0-225 menjadi 0-1.
4. *One-hot-encodin*g untuk y_train dan y_test, yaitu mengubah setiap nilai di dalam kolom menjadi kolom baru dan mengisinya dengan nilai biner yaitu 0 dan 1 (biasa digunakan untuk data kategorikal).
5. Hasil akhir berupa X_train_norm (X_train yang telah di normalisasi), y_train_encode (y_train yang telah di *encoding*), X_test_norm (X_test yang telah di normalisasi) dan y_test_encode (y_test yang telah di *encoding*).
6. Selanjutnya *variable* yang merepresentasikan citra tersebut sudah siap diolah kedalam algoritma CNN untuk dilakukan proses *training* kemudian *testing*.

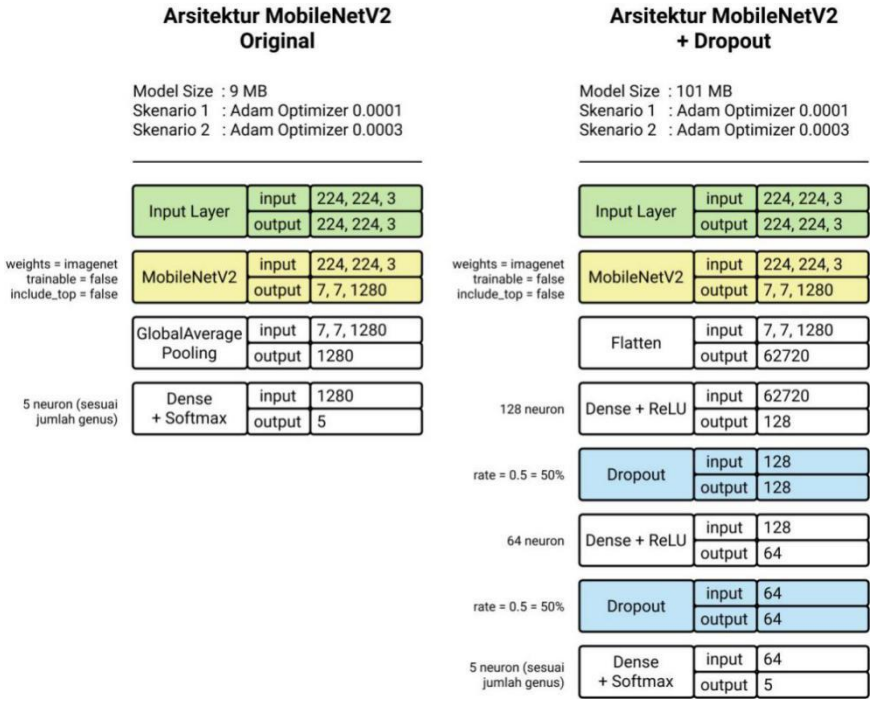
3.6 Perancangan K-Fold Cross Validation

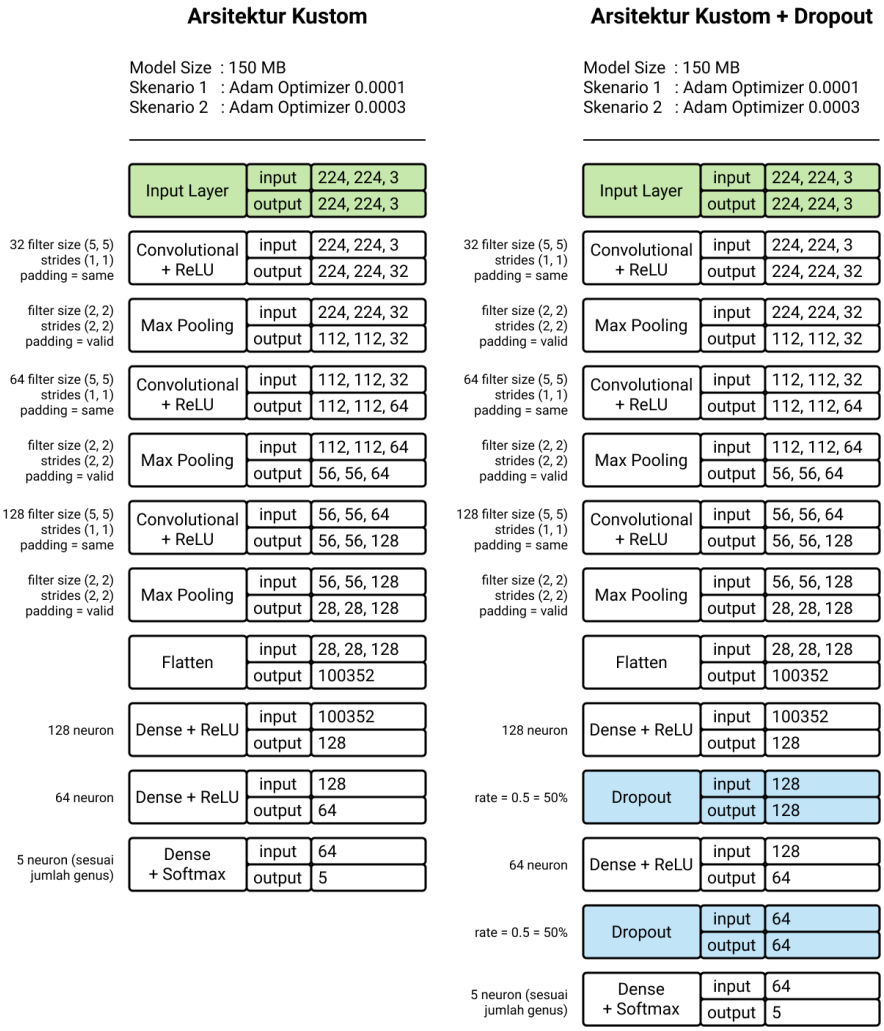
Untuk melihat performansi model CNN dalam pelatihan (*training*) yang akan dijalankan nanti, digunakanlah *K-Fold Cross Validation*. Dalam penelitian ini akan digunakan nilai K=5, ini berarti 80% data *train* dan 20% data *validation*.

3.6 Perancangan Arsitektur CNN

Pada penelitian ini akan digunakan 4 tipe arsitektur CNN, yaitu arsitektur kustom (penulis merancang sendiri arsitektur CNN dengan berpedoman pada *rule-of-thumb*), arsitektur kustom dengan tambahan *dropout*, arsitektur MobileNetV2 orisinal, dan arsitektur MobileNetV2 dengan tambahan *dropout*. Setiap arsitektur akan dilatih (*training & validation*) terhadap skenario dataset dengan *background noise* dan skenario dataset tanpa *background noise.* Selain itu, juga akan diterapkan *hyperparameter* berupa *learning rate* dengan nilai 0.0001 dan 0.0003, sehingga *output* yang dihasilkan nanti berjumlah 16 model untuk setiap *fold* (16 x 5 *fold* = 80 total model CNN), yang selanjutnya masing-masing model ini akan diujikan (*testing*) terhadap *data test* lalu mengevaluasi hasilnya, memilih model terbaik, dan terakhir men-*deploy*nya ke aplikasi siap pakai.

Berikut rancangan arsitektur kustom, arsitektur kustom dengan tambahan *dropout*, MobileNetV2 orisinal dan MobileNetV2 dengan tambahan *dropout* yang akan digunakan dalam penelitian ini, lengkap dengan keterangan *model size*nya dan *hyperparameter* yang dipakai.





Gambar 21. Rancangan 4 Tipe Arsitektur CNN

4. Evaluasi

4.1 Hasil Pelatihan & Pengujian

Proses pelatihan (*training*) dilakukan dengan menggunakan dua skenario data *train*, yang sebelumnya telah diolah pada tahap praproses dan augmentasi dataset, kedua skenario data *train* yaitu sebagai berikut:

- Skenario 1 → 4000 *data train* dengan *background noise* (800 citra/genus)
- Skenario 2 → 4000 *data train* tanpa *background noise* (800 citra/genus)

Selain itu, *training* dilakukan menggunakan teknik *K-Fold Cross Validation* dengan nilai K=5, ini berarti 80% data *train* (3200) dan 20% data *validation* (800). Juga digunakan beberapa *hyperparameter* sebagai berikut:

- Input Shape Citra : 224x224x3 (RGB *channel*)
- Batch Size : 64 (50 iterasi dalam satu *epoch*)
- Epoch : 10 (Hasil sudah cukup optimal)
- Optimizer : Adam (Salah satu *optimizer* terbaik)
- Learning Rate : 0.0001, 0.0003 (2 Skenario *Learning Rate*)

Kemudian, arsitektur CNN yang digunakan untuk *training* yaitu sebagai berikut:

- Arsitektur kustom
- Arsitektur kustom + *Dropout*
- Arsitektur MobileNetV2 orisinal
- Arsitektur MobileNetV2 + *Dropout*

Karena proses *training* dalam penelitian ini menggunakan dua skenario data *train* (dengan dan tanpa *background noise*) dan dua skenario *learning rate* (0.0001 & 0.0003) yang kemudian diterapkan ke 4 tipe arsitektur CNN, maka *output* keseluruhannya menghasilkan 16 (2 x 2 x 4 = 16) model untuk setiap *fold* (16 x 5 *fold* = 80 total

model CNN). Semua model yang dihasilkan dari proses *training* tersebut dilakukan pengujian (*testing*) terhadap dua tipe data *test*:

- 2500 *data test dari lapangan* (500 citra/genus)
- 185 *data test dari internet* (25 *Cattleya*, 40 untuk genus lainnya)

Terdapat total 80 model CNN lengkap beserta *score* akurasi *training*, *validation*, *testing dari lapangan* dan *testing dari internet* dalam bentuk tabel. Selengkapnya dapat dilihat di Lampiran 5-9. Namun, agar menghemat ruang dan agar lebih mudah dilakukan analisis, berikut hanya akan ditampilkan rata-rata *score* akurasi dari 5 *fold* untuk setiap model:

Tabel 2. Rata-rata score akurasi dari 5 fold untuk setiap model

Nama Model		Ukuran Model	Rata-Rata Akurasi 5 Fold			
			Training	Validation	Testing (Lapangan)	Testing (Internet)
Dengan BG Noise	kustom (adam 0.0001)	150 MB	99.02 %	98.04 %	97.88 %	40.32 %
	kustom (adam 0.0003)	150 MB	99.99 %	99.39 %	98.56 %	43.13 %
	kustom + dr (adam 0.0001)	150 MB	91.34 %	98.22 %	97.80 %	42.70 %
	kustom + dr (adam 0.0003)	150 MB	95.29 %	98.17 %	97.37 %	41.94 %
	mobilenetv2 ori (adam 0.0001)	9 MB	98.45 %	98.52 %	96.45 %	61.61 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.83 %	99.72 %	98.63 %	62.91 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	94.78 %	99.94 %	99.65 %	73.61 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	94.33 %	99.97 %	99.68 %	68.21 %
Tanpa BG Noise	kustom (adam 0.0001)	150 MB	99.93 %	98.19 %	41.43 %	42.26 %
	kustom (adam 0.0003)	150 MB	100 %	98.59 %	44.90 %	47.34 %
	kustom + dr (adam 0.0001)	150 MB	87.49 %	97.29 %	59.15 %	49.29 %
	kustom + dr (adam 0.0003)	150 MB	96.89 %	98.17 %	64.71 %	48.64 %
	mobilenetv2 ori (adam 0.0001)	9 MB	97.32 %	96.94 %	71.16 %	70.05 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.57 %	99.34 %	77.96 %	76.42 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	95.62 %	99.85 %	83.32 %	80.26 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	95.82 %	99.87 %	84.08 %	75.34 %

Rata-rata *score* akurasi dari 5 *fold* untuk setiap model pada tabel diatas menunjukkan seberapa baik model tersebut dalam melakukan klasifikasi secara umum. Dapat dilihat bahwa MobileNetV2 + *Dropout* dengan *hyperparameter* berupa *Adam Optimizer* dan *Learning Rate* 0.0001 menjadi model dengan *score* akurasi tertinggi, baik itu dalam skenario data *train* dengan *background noise* maupun dalam skenario data *train* tanpa *background noise*. Namun, jika tidak diambil nilai rata-rata akurasi modelnya, setiap model (total 80 model CNN) tentunya memiliki nilai akurasinya masing-masing, dan berikut ditampilkan 10 model CNN (masing-masing 5 model skenario data *train* dengan dan tanpa *background noise*) terbaik dengan *score* akurasi tertinggi:

Tabel 3. 10 model (dalam 2 skenario) terbaik dengan score akurasi tertinggi

Fold	Nama Model	Ukuran Model	Akurasi			
			Training	Validation	Testing (Lapangan)	Testing (Internet)
1	mobilenetv2 + dr (adam 0.0003) train dengan bg noise	101 MB	95.78 %	100 %	99.56 %	74.05 %
	mobilenetv2 + dr (adam 0.0003) train tanpa bg noise	101 MB	95.78 %	100 %	84.64 %	74.59 %
2	mobilenetv2 + dr (adam 0.0001) train dengan bg noise	101 MB	95.50 %	100 %	99.68 %	71.35 %
	mobilenetv2 + dr (adam 0.0001) train tanpa bg noise	101 MB	95.09 %	99.50 %	90.44 %	80.54 %
3	mobilenetv2 + dr (adam 0.0001) train dengan bg noise	101 MB	94.62 %	100 %	99.68 %	74.59 %
	mobilenetv2 + dr (adam 0.0003) train tanpa bg noise	101 MB	97.43 %	100 %	84.08 %	77.83 %
4	mobilenetv2 + dr (adam 0.0001) train dengan bg noise	101 MB	94.75 %	99.87 %	99.68 %	75.67 %
	mobilenetv2 + dr (adam 0.0001) train tanpa bg noise	101 MB	95.65 %	100 %	85.48 %	72.97 %
5	mobilenetv2 + dr (adam 0.0001)	101 MB	94.90 %	99.87 %	99.64 %	74.05 %

	train dengan bg noise					
	mobilenetv2 ori (adam 0.0003) train tanpa bg noise	9 MB	99.50 %	99.37 %	83.12 %	80.00 %

Dari tabel diatas dapat dilihat terdapat dua model CNN dengan hasil *score* akurasi tertinggi, yaitu model:

- MobileNetV2 + *Dropout* dengan *hyperparameter* berupa *Adam Optimizer* dan *Learning Rate* 0.0001 (dalam skenario data *train* tanpa *background noise*) di *fold* ke-2. Menghasilkan *score* akurasi *testing dari lapangan* sebesar 90.44% dan *testing dari internet* sebesar 80.54% (Ukuran model: 101 MB).
- MobileNetV2 orisinal dengan *hyperparameter* berupa *Adam Optimizer* dan *Learning Rate* 0.0003 (dalam skenario data *train* tanpa *background noise*) di *fold* ke-5. Menghasilkan *score* akurasi *testing dari lapangan* sebesar 83.12% dan *testing dari internet* sebesar 80.00% (Ukuran model: 9 MB).

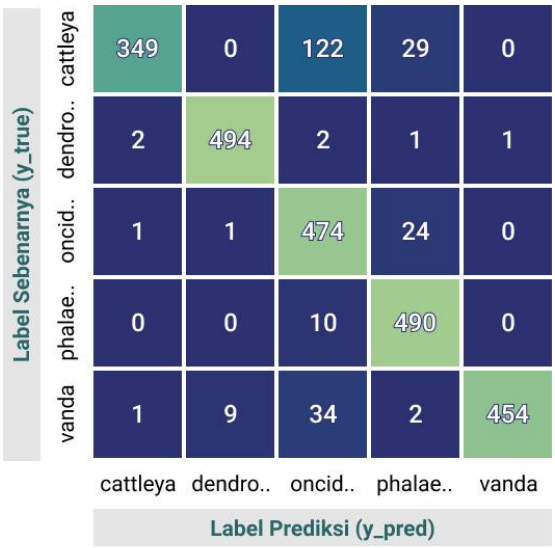
Berdasarkan informasi pada tabel 2 dan tabel 3, berikut beberapa hal yang dapat disimpulkan (dalam studi kasus klasifikasi citra genus tanaman anggrek):

1. Penggunaan *Dropout* meningkatkan *score* akurasi *testing* di banyak model.
2. Pemakaian *hyperparameter* berupa *Adam Optimizer* + *Learning Rate* 0.0001 menghasilkan *score* akurasi *testing* yang lebih tinggi di banyak model.
3. Skenario data *train* tanpa *background noise* secara umum menghasilkan *score* akurasi *testing* yang lebih tinggi di banyak model, jika dibandingkan dengan skenario data *train* dengan *background noise*.

4.2 Analisis Model CNN Terbaik

Dari hasil pelatihan (*training*) dan pengujian (*testing*) pada subbab sebelumnya, didapat dua model CNN dengan *score* akurasi tertinggi. Pada tahap ini akan dilakukan analisis terhadap kedua model tersebut dengan menggunakan *Confusion Matrix* dan *Classification Report*. Berikut detail analisisnya untuk kedua model.

Pertama, untuk MobileNetV2 + *Dropout* (*Adam* 0.0001) dengan skenario data *train* tanpa *background noise* di *fold* ke-2 (TOP 1 model CNN) menghasilkan *Confusion Matrix* dan *Classification Report* sebagai berikut:



TOP 1 Model CNN (101 MB)
Hasil Testing dengan Data Test Lapangan

MobileNetV2 + Dropout | Adam 0.0001 | Fold ke-2
(Skenario: Data Train tanpa Background Noise)

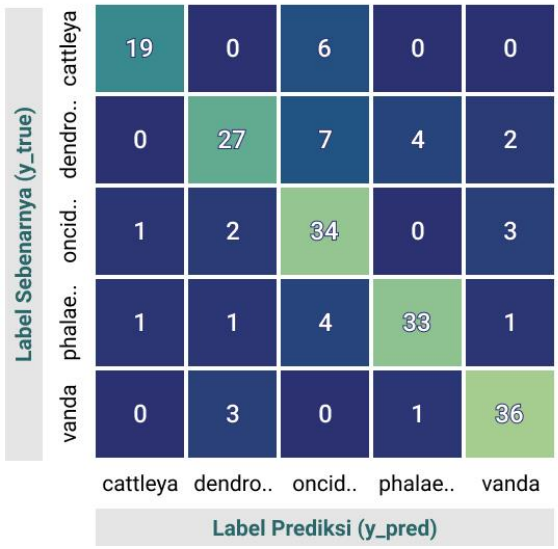
```
Accuracy : 90.440%
Loss     : 0.297

Classification Report :

              precision    recall  f1-score

    cattleya       0.99      0.70      0.82
  dendrobium       0.98      0.99      0.98
    oncidium       0.74      0.95      0.83
phalaenopsis       0.90      0.98      0.94
       vanda       1.00      0.91      0.95

    accuracy                           0.90
   macro avg       0.92      0.90      0.90
weighted avg       0.92      0.90      0.90
```



TOP 1 Model CNN (101 MB)
Hasil Testing dengan Data Test Internet

MobileNetV2 + Dropout | Adam 0.0001 | Fold ke-2
(Skenario: Data Train tanpa Background Noise)

```
Accuracy : 80.541%
Loss     : 0.609

Classification Report :

              precision    recall  f1-score

    cattleya       0.90      0.76      0.83
  dendrobium       0.82      0.68      0.74
    oncidium       0.67      0.85      0.75
phalaenopsis       0.87      0.82      0.85
       vanda       0.86      0.90      0.88

    accuracy                           0.81
   macro avg       0.82      0.80      0.81
weighted avg       0.82      0.81      0.81
```

Gambar 22. Confusion Matrix dan Classification Report model CNN terbaik ke-1

Berdasarkan informasi pada *Confusion Matrix* dan *Classification Report* diatas, serta berdasarkan Lampiran 10 yang digunakan sebagai acuan bentuk dan warna daun dari setiap genus, berikut beberapa hal yang dapat disimpulkan:

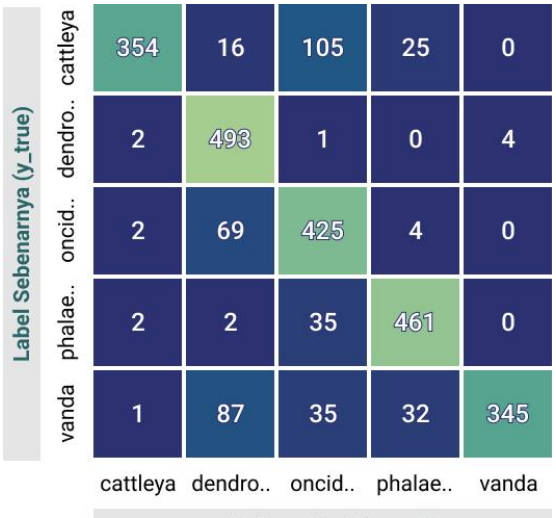
1. Hasil *testing* dengan data *test lapangan* menghasilkan paling banyak salah prediksi yaitu pada genus *Cattleya*. Dimana dari total 500 citra *Cattleya*, 112 diantaranya diprediksi salah oleh model sebagai *Oncidium*. Setelah dilakukan penelusuran, kemungkinan besar kesalahan prediksi disebabkan oleh kemiripan beberapa citra *Cattleya* dan *Oncidium* di dataset. Contohnya (kiri *Cattleya* dari data *train* dan kanan *Oncidium* dari data *test lapangan*):

Kemudian, paling banyak salah prediksi di posisi ke-2, yaitu dari total 500 citra *Vanda*, 34 diantaranya diprediksi salah oleh model sebagai *Oncidium*. Alasan yang sama, yaitu kemungkinan akan kemiripan citra menjadi penyebab kesalah prediksi. Contohnya (kiri *Vanda* dari data *train* dan kanan *Oncidium* dari data *test lapangan*):

2. Hasil *testing* dengan data *test internet* menghasilkan paling banyak salah prediksi yaitu pada genus *Dendrobium*. Dari total 50 citra *Dendrobium*, 7 diantaranya diprediksi salah oleh model sebagai *Oncidium.* Kemungkinan akan kemiripan citra menjadi penyebab kesalah prediksi. Contohnya (kiri *Dendrobium* dari data *train* dan kanan *Oncidium* dari data *test internet*):

3. Model MobileNetV2 + *Dropout* (*Adam* 0.0001) dengan skenario data *train* tanpa *background noise* di *fold* ke-2 secara keseluruhan menghasilkan *score* akurasi dan *f1-score* yang cukup baik.

Kedua, untuk MobileNetV2 orisinal (*Adam* 0.0003) dengan skenario data *train* tanpa *background noise* di *fold* ke-5 (TOP 2 model CNN) menghasilkan *Confusion Matrix* dan *Classification Report* sebagai berikut:



TOP 2 Model CNN (9 MB)
Hasil Testing dengan Data Test Lapangan

MobileNetV2 Orisinal | Adam 0.0003 | Fold ke-5
(Skenario: Data Train tanpa Background Noise)

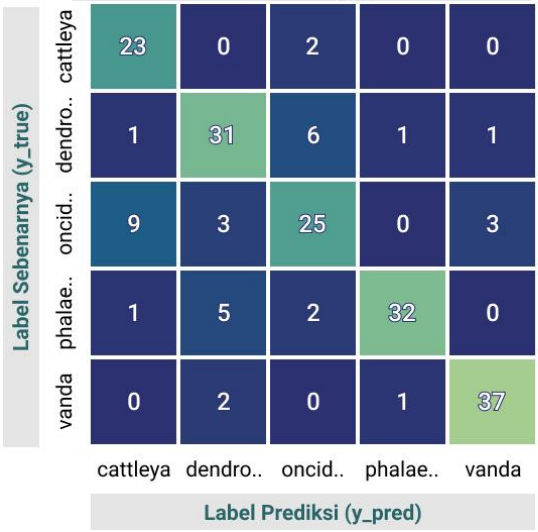
```
Accuracy : 83.120%
Loss     : 0.459

Classification Report :

              precision    recall  f1-score

    cattleya       0.98      0.71      0.82
  dendrobium       0.74      0.99      0.84
    oncidium       0.71      0.85      0.77
phalaenopsis       0.88      0.92      0.90
       vanda       0.99      0.69      0.81

    accuracy                           0.83
   macro avg       0.86      0.83      0.83
weighted avg       0.86      0.83      0.83
```



TOP 2 Model CNN (9 MB)
Hasil Testing dengan Data Test Internet

MobileNetV2 Orisinal | Adam 0.0003 | Fold ke-5
(Skenario: Data Train tanpa Background Noise)

```
Accuracy : 80.000%
Loss     : 0.646

Classification Report :

              precision    recall  f1-score

    cattleya       0.68      0.92      0.78
  dendrobium       0.76      0.78      0.77
    oncidium       0.71      0.62      0.67
phalaenopsis       0.94      0.80      0.86
       vanda       0.90      0.93      0.91

    accuracy                           0.80
   macro avg       0.80      0.81      0.80
weighted avg       0.81      0.80      0.80
```

Gambar 23. Confusion Matrix dan Classification Report model CNN terbaik ke-2

Berdasarkan informasi *Confusion Matrix* dan *Classification Report* diatas, serta berdasarkan Lampiran 10, beberapa hal yang dapat disimpulkan yaitu:

1. Hasil *testing* dengan data *test lapangan* menghasilkan paling banyak salah prediksi yaitu pada genus *Cattleya*. Dimana dari total 500 citra *Cattleya,* 105 diantaranya diprediksi salah oleh model sebagai *Oncidium.* Jika diperhatikan, kesalahan prediksi pada model ini sama dengan kesalahan prediksi paling banyak di TOP 1 model CNN sebelumnya. Kemudian, paling banyak salah prediksi di posisi ke-2, yaitu dari total 500 citra *Vanda*, 87 diantaranya diprediksi salah oleh model sebagai *Dendrobium*. Dengan alasan yang sama, yaitu kemungkian akan kemiripan citra menjadi penyebab kesalah prediksi.
2. Hasil *testing* dengan data *test internet* menghasilkan paling banyak salah prediksi yaitu pada genus *Oncidium*. Dari total 50 citra *Oncidium,* 9 diantaranya diprediksi salah oleh model sebagai *Cattleya.* Hal yang menarik terjadi disini, kesalahan prediksi yang seharusnya *Oncidium* malah diprediksi *Cattleya*, sama saja dengan kesalahan prediksi dipembahasan sebelumnya, yaitu yang seharusnya *Cattleya* malah diprediksi *Oncidium.* Sehingga antara *Cattleya & Oncidium,* merupakan genus dengan kesalahan prediksi terbanyak.
3. Model MobileNetV2 orisinal (*Adam* 0.0003) dengan skenario data *train* tanpa *background noise* di *fold* ke-5 secara keseluruhan menghasilkan *score* akurasi dan *f1-score* yang cukup baik.

4.3 Deploy model CNN ke Website

Dari dua model terbaik yang telah dihasikan, dipilih TOP 2 model CNN yaitu MobileNetV2 orisinal (*Adam* 0.0003) dengan skenario data *train* tanpa *background noise* di *fold* ke-5 yang akan di *deploy* ke aplikasi website. Alasannya karena TOP 2 model CNN ini berukuran hanya 9 MB, sehingga jauh lebih kecil dan ringan jika dibandingkan TOP 1 model CNN yang berukuran 101 MB, terlebih karena proses *deploy* dilakukan ke aplikasi website yang memerlukan jaringan internet untuk dapat mengaksesnya. Proses *deploy* menggunakan Flask dan *image classifier app template* oleh Fing di Github (lihat: https://github.com/mtobeiyf/), serta *hosting* di layanan bernama Heroku. Untuk *screenshot* tampilan dan proses prediksi citra di website, dapat lihat Lampiran 11. Aplikasi dapat diakses di: https://anggrek-classifier.herokuapp.com/.

5. Kesimpulan

Penelitian ini menghasilkan sebuah sistem pengklasifikasi genus tanaman anggrek yang paling umum dibudidayakan (*Cattleya*, *Dendrobium*, *Oncidium*, *Phalaenopsis* dan *Vanda*) berdasarkan citra input yang menggunakan teknologi *Deep Learning* dengan metode *Convolutional Neural Network* (CNN) berbasis aplikasi website. Dihasilkan dua model CNN terbaik dengan *score* akurasi *testing* yang cukup tinggi, masing-masing model beserta *score* akurasi *testing*nya yaitu:

- MobileNetV2 + *Dropout* dengan *hyperparameter* berupa *Adam Optimizer* dan *Learning Rate* 0.0001 (dalam skenario data *train* tanpa *background noise*) di *fold* ke-2. Menghasilkan *score* akurasi *testing dari lapangan* sebesar 90.44% dan *testing dari internet* sebesar 80.54% (Ukuran model: 101 MB).
- MobileNetV2 orisinal dengan *hyperparameter* berupa *Adam Optimizer* dan *Learning Rate* 0.0003 (dalam skenario data *train* tanpa *background noise*) di *fold* ke-5. Menghasilkan *score* akurasi *testing dari lapangan* sebesar 83.12% dan *testing dari internet* sebesar 80.00% (Ukuran model: 9 MB).

Berikut beberapa hal yang ditarik menjadi kesimpulan dalam penelitan ini (dalam studi kasus klasifikasi citra genus tanaman anggrek dengan CNN):

1. Arsitektur MobileNetV2 teruji menghasilkan *score* akurasi *testing* yang tinggi.
2. Penggunaan *Dropout* dapat mengurangi *overfitting* pada model CNN yang dibangun, ditunjukkan dengan *score* akurasi *testing* yang lebih tinggi.
3. Pemakaian *hyperparameter* berupa *Adam Optimizer* + *Learning Rate* 0.0001 menghasilkan *score* akurasi *testing* yang lebih tinggi di banyak model CNN.
4. Skenario data *train* tanpa *background noise* secara umum menghasilkan *score* akurasi *testing* yang lebih tinggi di banyak model CNN, jika dibandingkan dengan skenario data *train* dengan *background noise*.
5. Augmentasi terhadap data *train* di dataset teruji menghasilkan *score* akurasi *testing* yang tinggi, jika digunakan ke arsitektur CNN beserta nilai *hyperparameter* yang tepat (*experimental*).
6. Kesalahan prediksi banyak terjadi akibat kemiripan citra antar genus, terlebih karena dalam penelitian ini semua dataset berupa citra daun anggrek, yang mana baik dari segi bentuk maupun warna (hijau) hampir serupa. Dua genus dengan kesalahan prediksi terbanyak yaitu antara *Cattleya* dengan *Oncidium.*

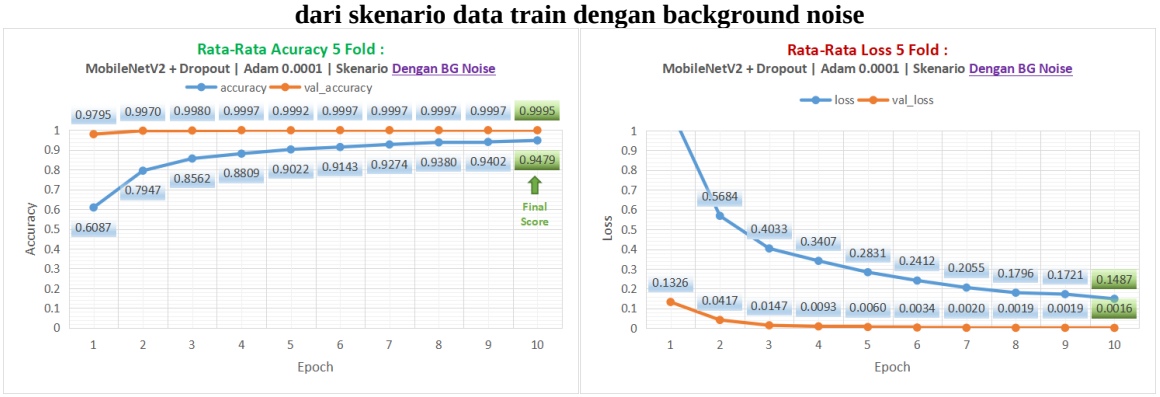
Referensi

[1] Adisarwanto, Titis. et al. 2012. Anggrek Species Indonesia. Jakarta: Direktorat Perbenihan Hortikultura.
[2] Assagaf, Mazna Hashim. 2012. 1001 Spesies Anggrek Yang Dapat Berbunga di Indonesia. Jakarta: Kataelha.
[3] Sutoyo, T. et al. 2009. Teori Pengolahan Citra Digital. Yogyakarta: Andi.
[4] Ng Annalyn dan Soo Kenneth. 2017. Numsense! Data Science for the Layman, No Math Added. Cambridge: Annalyn Ng & Kenneth Soo.
[5] French, Alan. 2018. Neural Networks Without the Math. Hong Kong: Joyously Aware Media.
[6] Kelleher, John D. 2019. Deep Learning. Cambridge: The MIT Press Essential Knowledge Series.
[7] Khan, Salman. et al. 2018. A Guide to Convolutional Neural Networks for Computer Vision. California: Morgan & Claypool.
[8] Putra, Jan Wira Gotama. 2020. Pengenalan Konsep Pembelajaran Mesin dan Deep Learning. Tokyo: Jan Wira Gotama Putra.
[9] Krizhevsky, Alex. Sutskever Ilya dan E Hinton Geoffrey. 2012. ImageNet Classification With Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems 25 (NIPS 2012).
[10] Murugan, Pushparaja. 2017. Feed Forward and Backward Run in Deep Convolution Neural Network. arXiv:1711.03278 [cs.CV] 9 Nov 2017.
[11] Khan, Asifullah. Sohail, Anabia. et al. 2020. A Survey of the Recent Architectures of Deep Convolutional Neural Networks. arXiv:1901.06032 [cs.CV] 10 May 2020.
[12] Howard, Andrew G. Zhu, Menglong. et al. 2017. MobileNets Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861v1 [cs.CV] 17 Apr 2017.
[13] Sandler, Mark. Howard, Andrew. et al. 2019. MobileNetV2 Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381 [cs.CV] 21 Mar 2019.
[14] NarasingaRao, Dr.M.R., Prasad V Venkatesh. et al. 2018. Survey on Prevention of Overfitting in Convolution Neural Networks Using ML Techniques. International Journal of Engineering & Technology, 7 (2.32) (2018) 177-180.
[15] Xiao, Han. Rasul, Kashif dan Vollgraf Roland. 2017. Fashion-MNIST a Novel Image Dataset for Benchmarking Machine Learning Algorithms. arXiv:1708.07747 [cs.LG] 15 Sep 2017.
[16] Rajnoha, Martin. Burget, Radim dan Povoda Lukas. 2018. Image Background Noise Impact on Convolutional Neural Network Training. 2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT).
[17] Intellipaat. 2021. AI vs ML vs DL. [Online] Available at: https://intellipaat.com/blog/tutorial/artificial-intelligence-tutorial/ai-vs-ml-vs-dl/
[18] Intellipaat. 2019. Supervised Learning vs Unsupervised Learning vs Reinforcement Learning. [Online] Available at: https://intellipaat.com/blog/supervised-learning-vs-unsupervised-learning-vs-reinforcement-learning/
[19] Kulshreshtha, Ankur. 2019. Brief History of Deep Learning from 1943-2019 Timeline. [Online] Available at: https://machinelearningknowledge.ai/brief-history-of-deep-learning/
[20] Kumar, Praveen dan Singh Nilesh. 2019. Introduction to Deep Learning With Computer Vision — Kernels, Channels & Neural Architecture. [Online] Available at: https://medium.com/hitchhikers-guide-to-deep-learning/5-introduction-to-deep-learning-with-computer-vision-kernels-channels-neural-architecture-41b6bc4befa7
[21] Deshpande, Adit. 2016. A Beginner's Guide To Understanding Convolutional Neural Networks. [Online] Available at: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/
[22] Karn, Ujjwal. 2016. An Intuitive Explanation of Convolutional Neural Networks. [Online] Available at: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/
[23] Sood, Devashish. 2018. Backpropagation Concept Explained in 5 Levels of Difficulty. [Online] Available at: https://medium.com/coinmonks/backpropagation-concept-explained-in-5-levels-of-difficulty-8b220a939db5
[24] Hmkcode. 2019. Backpropagation Step by Step. [Online] Available at: https://hmkcode.com/ai/backpropagation-step-by-step/
[25] Ranjan, Chitta. 2019. Rules-of-thumb for Building a Neural Network. [Online] Available at: https://towardsdatascience.com/17-rules-of-thumb-for-building-a-neural-network-93356f9930af
[26] Sharma, Sagar. 2017. Epoch vs Batch Size vs Iterations. [Online] Available at: https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9

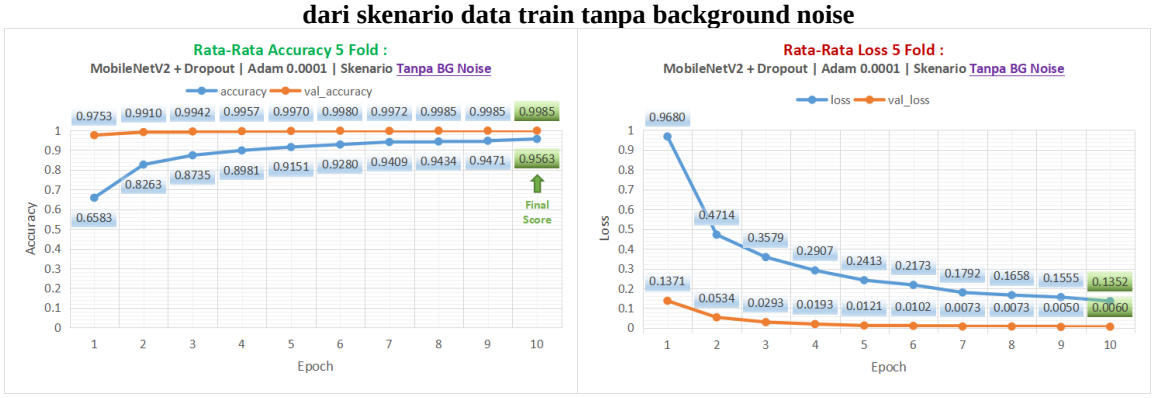
[27] Kumar, Satyam. 2020. Overview of Various Optimizers in Neural Networks. [Online] Available at: https://towardsdatascience.com/overview-of-various-optimizers-in-neural-networks-17c1be2df6d5
[28] Ruder, Sebastian. 2020. An Overview of Gradient Descent Optimization Algorithms. [Online] Available at: https://ruder.io/optimizing-gradient-descent/index.html
[29] Bai, Kunlun. 2019. A Comprehensive Introduction to Different Types of Convolutions in Deep Learning. [Online] Available at: https://towardsdatascience.com/a-comprehensive-introduction-to-different-types-of-convolutions-in-deep-learning-669281e58215
[30] Wang, Chi-Feng. 2018. A Basic Introduction to Separable Convolutions. [Online] Available at: https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728
[31] Bouteille. 2019. MobileNet-V2: Summary and Implementation. [Online] Available at: https://hackmd.io/@bouteille/ryaDuxe5L
[32] Tsang, Sik-Ho. 2019. Review MobileNetV2 - Light Weight Model (Image Classification). [Online] Available at: https://towardsdatascience.com/review-mobilenetv2-light-weight-model-image-classification-8febb490e61c
[33] Hollemans, Matthijs. 2018. MobileNet Version 2. [Online] Available at: https://machinethink.net/blog/mobilenet-v2/
[34] Brownlee, Jason. 2018. A Gentle Introduction to K-Fold Cross-Validation. [Online] Available at: https://machinelearningmastery.com/k-fold-cross-validation/
[35] Laurenti, Giulio. 2020. Confusion Matrix and Classification Report. [Online] Available at: https://medium.com/swlh/confusion-matrix-and-classification-report-88105288d48f
[36] Huilgol, Purva. 2019. Accuracy vs. F1-Score. [Online] Available at: https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2
[37] Herlambang, Mega Bagus. 2019. Deep Learning: Artificial Neural Networks & Convolutional Neural Network. [Online] Available at: https://www.megabagus.id/deep-learning-artificial-neural-networks/ (ANN), https://www.megabagus.id/deep-learning-convolutional-neural-networks/ (CNN)
[38] Skillplus. 2019. Komponen Artificial Neural Network. [Online] Available at: https://skillplus.web.id/komponen-artificial-neural-network/
[39] Putra, Wira Dharma Kencana. 2020. Course 5 Transisi ke Deep Learning: Training Loop. [Online] Available at: https://youtu.be/KYqTelQHOQ4?t=301 (Cost), https://youtu.be/U4_nXVj_cpk?t=1578 (Testing)
[40] Ilyas, Ridwan. 2021. Machine Learning 101: Backpropagation. [Online] Available at: https://www.youtube.com/playlist?list=PLo6nZTcpSz2p5oKKkg6ZWHx4Pw7ToYVtD

Lampiran

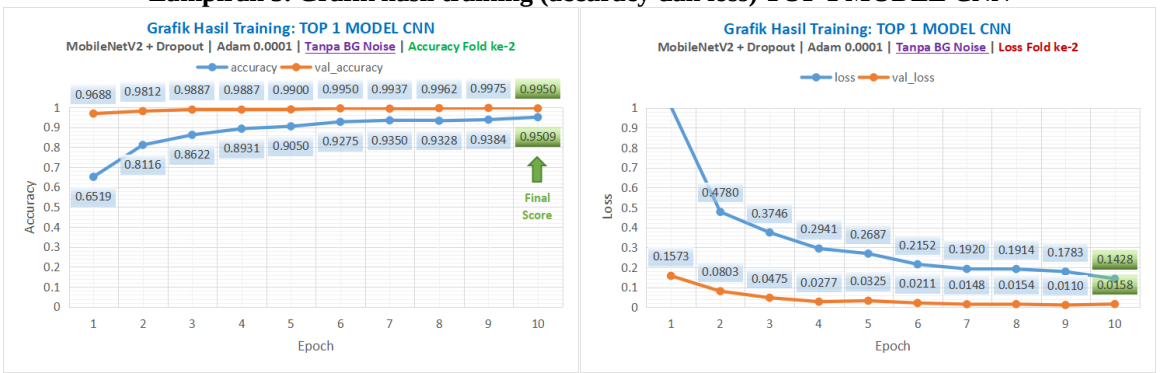
Lampiran 1. Grafik rata-rata skor accuracy 5 fold tertinggi beserta grafik lossnya, dari skenario data train dengan background noise



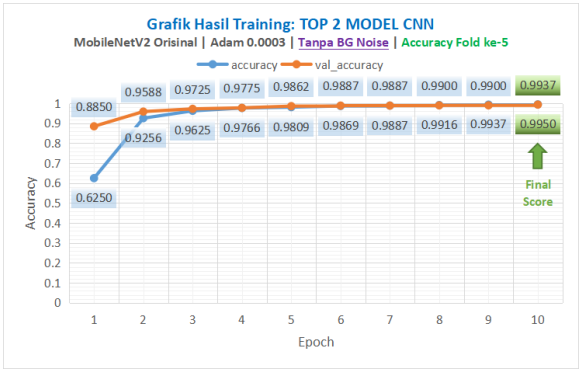
Lampiran 2. Grafik rata-rata skor accuracy 5 fold tertinggi beserta grafik lossnya, dari skenario data train tanpa background noise

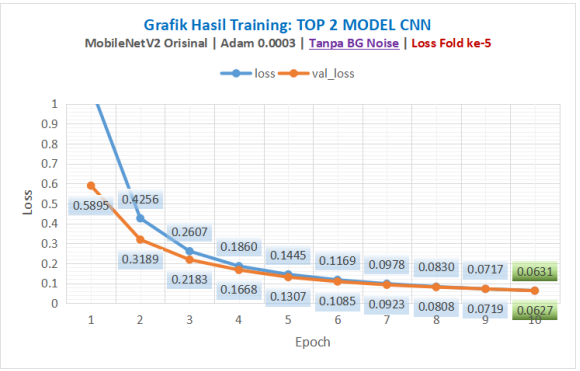


Lampiran 3. Grafik hasil training (accuracy dan loss) TOP 1 MODEL CNN



Lampiran 4. Grafik hasil training (accuracy dan loss) TOP 2 MODEL CNN


Grafik Hasil Training: TOP 2 MODEL CNN
MobileNetV2 Orisinal | Adam 0.0003 | Tanpa BG Noise | Accuracy Fold ke-5
accuracy
val_accuracy
0.8850
0.9588
0.9725
0.9775
0.9862
0.9887
0.9887
0.9900
0.9900
0.9937
0.6250
0.9256
0.9625
0.9766
0.9809
0.9869
0.9887
0.9916
0.9937
0.9950
Final Score
Accuracy
Epoch


Grafik Hasil Training: TOP 2 MODEL CNN
MobileNetV2 Orisinal | Adam 0.0003 | Tanpa BG Noise | Loss Fold ke-5
loss
val_loss
0.5895
0.4256
0.2607
0.1860
0.1445
0.1169
0.0978
0.0830
0.0717
0.0631
0.3189
0.2183
0.1668
0.1307
0.1085
0.0923
0.0808
0.0719
0.0627
Loss
Epoch

Lampiran 5. Tabel hasil training fold 1 (16 model CNN) beserta hasil testingnya

Nama Model		Ukuran Model	Akurasi Fold 1			
			Training	Validation	Testing	Testing
Dengan BG Noise	kustom (adam 0.0001)	150 MB	99.87 %	98.25 %	98.28 %	41.08 %
	kustom (adam 0.0003)	150 MB	100 %	99.25 %	98.56 %	43.24 %
	kustom + dr (adam 0.0001)	150 MB	90.21 %	98.37 %	98.00 %	41.62 %
	kustom + dr (adam 0.0003)	150 MB	95.81 %	98.25 %	95.40 %	39.45 %
	mobilenetv2 ori (adam 0.0001)	9 MB	98.34 %	98.62 %	95.96 %	66.48 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.87 %	99.87 %	98.32 %	62.16 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	94.15 %	100 %	99.60 %	72.43 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	95.78 %	100 %	99.56 %	74.05 %
Tanpa BG Noise	kustom (adam 0.0001)	150 MB	99.78 %	99.00 %	36.16 %	43.78 %
	kustom (adam 0.0003)	150 MB	100 %	99.37 %	44.08 %	44.32 %
	kustom + dr (adam 0.0001)	150 MB	87.09 %	98.00 %	52.56 %	55.67 %
	kustom + dr (adam 0.0003)	150 MB	96.65 %	99.12 %	57.20 %	45.40 %
	mobilenetv2 ori (adam 0.0001)	9 MB	98.00 %	97.25 %	70.24 %	68.64 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.53 %	99.37 %	74.56 %	75.67 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	96.31 %	100 %	82.72 %	75.13 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	95.78 %	100 %	84.64 %	74.59 %

Lampiran 6. Tabel hasil training fold 2 (16 model CNN) beserta hasil testingnya

Nama Model		Ukuran Model	Akurasi Fold 2			
			Training	Validation	Testing	Testing
Dengan BG Noise	kustom (adam 0.0001)	150 MB	99.90 %	98.75 %	98.20 %	41.62 %
	kustom (adam 0.0003)	150 MB	100 %	99.50 %	98.28 %	41.62 %
	kustom + dr (adam 0.0001)	150 MB	92.87 %	97.50 %	98.12 %	41.08 %
	kustom + dr (adam 0.0003)	150 MB	93.31 %	98.37 %	98.40 %	40.54 %
	mobilenetv2 ori (adam 0.0001)	9 MB	98.37 %	98.75 %	96.32 %	55.13 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.81 %	99.87 %	98.40 %	61.08 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	95.50 %	100 %	99.68 %	71.35 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	93.56 %	100 %	99.64 %	63.24 %
Tanpa BG Noise	kustom (adam 0.0001)	150 MB	100 %	98.00 %	39.76 %	38.91 %
	kustom (adam 0.0003)	150 MB	100 %	98.62 %	43.96 %	47.02 %
	kustom + dr (adam 0.0001)	150 MB	84.06 %	97.12 %	62.36 %	45.94 %
	kustom + dr (adam 0.0003)	150 MB	97.09 %	96.87 %	70.80 %	49.73 %
	mobilenetv2 ori (adam 0.0001)	9 MB	96.71 %	95.87 %	71.72 %	71.35 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.71 %	98.75 %	78.28 %	77.29 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	95.09 %	99.50 %	90.44 %	80.54 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	94.46 %	99.87 %	84.72 %	72.97 %

Lampiran 7. Tabel hasil training fold 3 (16 model CNN) beserta hasil testingnya

Nama Model		Ukuran Model	Akurasi Fold 3			
			Training	Validation	Testing	Testing
Dengan BG Noise	kustom (adam 0.0001)	150 MB	100 %	98.62 %	98.88 %	40.54 %
	kustom (adam 0.0003)	150 MB	99.96 %	99.50 %	98.60 %	40.00 %
	kustom + dr (adam 0.0001)	150 MB	91.43 %	98.75 %	97.08 %	44.86 %
	kustom + dr (adam 0.0003)	150 MB	96.59 %	98.37 %	97.12 %	42.16 %
	mobilenetv2 ori (adam 0.0001)	9 MB	98.37 %	98.37 %	96.52 %	63.24 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.78 %	99.75 %	98.68 %	60.00 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	94.62 %	100 %	99.68 %	74.59 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	94.03 %	100 %	99.64 %	73.51 %
Tanpa BG Noise	kustom (adam 0.0001)	150 MB	99.96 %	98.62 %	44.96 %	41.62 %
	kustom (adam 0.0003)	150 MB	100 %	98.62 %	47.44 %	47.02 %
	kustom + dr (adam 0.0001)	150 MB	89.81 %	98.00 %	60.08 %	48.10 %
	kustom + dr (adam 0.0003)	150 MB	97.28 %	98.62 %	65.92 %	50.27 %
	mobilenetv2 ori (adam 0.0001)	9 MB	97.56 %	96.75 %	64.16 %	70.27 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.50 %	99.50 %	75.28 %	74.59 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	95.87 %	99.75 %	80.68 %	76.21 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	97.43 %	100 %	84.08 %	77.83 %

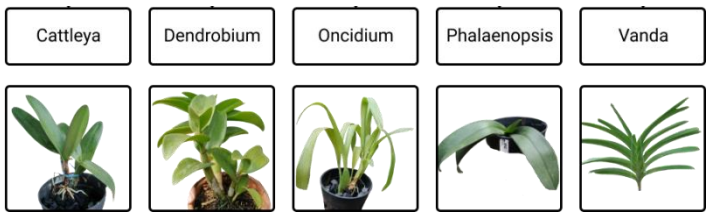
Lampiran 8. Tabel hasil training fold 4 (16 model CNN) beserta hasil testingnya

	Nama Model	Ukuran Model	Akurasi Fold 4			
			Training	Validation	Testing	Testing
Dengan BG Noise	kustom (adam 0.0001)	150 MB	100 %	99.25 %	98.36 %	40.54 %
	kustom (adam 0.0003)	150 MB	100 %	99.62 %	98.60 %	45.40 %
	kustom + dr (adam 0.0001)	150 MB	88.93 %	98.25 %	96.88 %	44.32 %
	kustom + dr (adam 0.0003)	150 MB	95.53 %	98.62 %	98.76 %	42.70 %
	mobilenetv2 ori (adam 0.0001)	9 MB	98.53 %	98.62 %	96.12 %	57.83 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.90 %	99.75 %	99.20 %	64.32 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	94.75 %	99.87 %	99.68 %	75.67 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	95.53 %	100 %	99.80 %	65.40 %
Tanpa BG Noise	kustom (adam 0.0001)	150 MB	99.93 %	97.25 %	37.80 %	44.32 %
	kustom (adam 0.0003)	150 MB	100 %	98.62 %	35.56 %	49.18 %
	kustom + dr (adam 0.0001)	150 MB	87.75 %	97.12 %	59.88 %	45.94 %
	kustom + dr (adam 0.0003)	150 MB	96.93 %	98.25 %	66.52 %	49.73 %
	mobilenetv2 ori (adam 0.0001)	9 MB	97.31 %	97.75 %	71.60 %	68.64 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.65 %	99.75 %	78.60 %	74.59 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	95.65 %	100 %	85.48 %	72.97 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	95.78 %	99.87 %	81.56 %	76.21 %

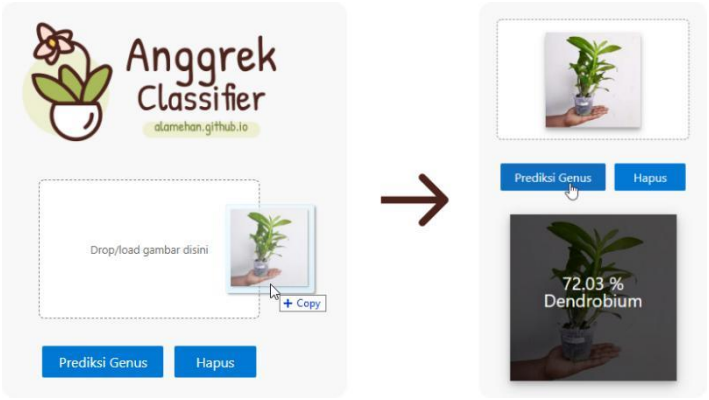
Lampiran 9. Tabel hasil training fold 5 (16 model CNN) beserta hasil testingnya

	Nama Model	Ukuran Model	Akurasi Fold 5			
			Training	Validation	Testing	Testing
Dengan BG Noise	kustom (adam 0.0001)	150 MB	95.37 %	95.37 %	95.68 %	37.83 %
	kustom (adam 0.0003)	150 MB	100 %	99.12 %	98.76 %	45.40 %
	kustom + dr (adam 0.0001)	150 MB	93.28 %	98.25 %	98.96 %	41.62 %
	kustom + dr (adam 0.0003)	150 MB	95.25 %	97.25 %	97.20 %	44.86 %
	mobilenetv2 ori (adam 0.0001)	9 MB	98.68 %	98.25 %	97.36 %	65.40 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.81 %	99.62 %	98.56 %	67.02 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	94.90 %	99.87 %	99.64 %	74.05 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	92.75 %	99.87 %	99.76 %	64.86 %
Tanpa BG Noise	kustom (adam 0.0001)	150 MB	100 %	98.12 %	48.48 %	42.70 %
	kustom (adam 0.0003)	150 MB	100 %	97.75 %	53.48 %	49.18 %
	kustom + dr (adam 0.0001)	150 MB	88.75 %	96.25 %	60.88 %	50.81 %
	kustom + dr (adam 0.0003)	150 MB	96.53 %	98.00 %	63.12 %	48.10 %
	mobilenetv2 ori (adam 0.0001)	9 MB	97.06 %	97.12 %	78.12 %	71.35 %
	mobilenetv2 ori (adam 0.0003)	9 MB	99.50 %	99.37 %	83.12 %	80.00 %
	mobilenetv2 + dr (adam 0.0001)	101 MB	95.21 %	100 %	77.28 %	75.67 %
	mobilenetv2 + dr (adam 0.0003)	101 MB	95.68 %	99.62 %	85.40 %	75.13 %

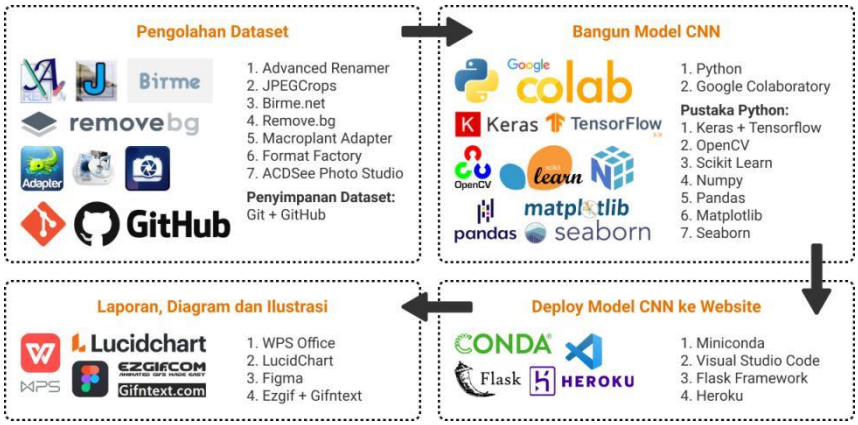
Lampiran 10. Gambar bentuk dan warna daun dari masing-masing lima genus anggrek



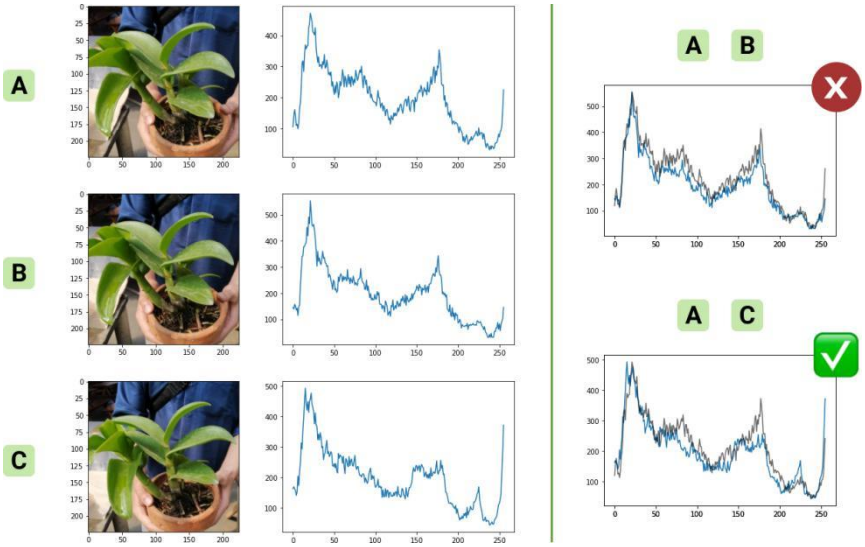
Lampiran 11. Screenshot tampilan dan proses prediksi citra di website



Lampiran 12. Teknologi dan tools yang digunakan dalam penelitian



Lampiran 13. Toleransi tingkat kemiripan citra (pada tahap seleksi citra)



Lampiran 14. Dokumentasi penelitian: foto saat pengambilan

dataset citra tanaman Anggrek di Rumah Bunga Rizal, Lembang

Lampiran 15. Link penunjang penelitian

- https://github.com/alamehan/skripsi-cnn-anggrek/ : Berisi dataset lima genus Anggrek, *source code* Python Google Colaboratory (bangun model CNN) & Python Flask Framework (*deploy* model CNN ke *website*), dokumentasi hasil *training* dan *testing*, serta aset berupa gambar (.jpg), ikon (.svg) dan animasi (.gif) yang digunakan dalam laporan penelitian.
- https://github.com/alamehan/deep-learning-cnn-simplified/ : Berisi referensi *tools* serta aset lainnya sebagai penunjang penelitian.
- https://anggrek-classifier.herokuapp.com/ : *Website* Anggrek Classifier (hasil akhir penelitian).