Model Klasifikasi Makanan Tradisional Berbasis Arsitektur EfficientNetV2

Erin Eka Citra

20917014

Tesis diajukan sebagai syarat untuk meraih gelar Magister Komputer

Konsentrasi Sains Data

Program Studi Informatika Program Magister

Fakultas Teknologi Industri

Universitas Islam Indonesia

2023

Lembar Pengesahan Pembimbing

Model Klasifikasi Makanan Tradisional Berbasis Arsitektur EfficientNetV2

Erin Eka Citra

20917014

Yogyakarta, Juni 2023

Pembimbing 1	Pembimbing 2
Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D	Chandra Kusuma Dewa, S.Kom., M.Cs., Ph.D

Lembar Pengesahan Penguji

Model Klasifikasi Makanan Tradisional Berbasis Arsitektur EfficientNetV2

Erin Eka Citra

20917014

Yogyakarta, Juni 2023

Tim Penguji,

Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D

Ketua

Chandra Kusuma Dewa, S.Kom., M.Cs., Ph.D

Anggota I

Irving Vitra Paputungan, S.T., M.Sc., Ph.D

Anggota II

Mengetahui,

Ketua Program Studi Informatika Program Magister

Universitas Islam Indonesia

Irving Vitra Paputungan, S.T., M.Sc., Ph.D.

Abstrak

Model Klasifikasi Makanan Tradisional Berbasis Arsitektur EfficientNetV2

Indonesia merupakan salah satu negara yang memiliki banyak aneka ragam makanan tradisional dan objek wisata yang mempesona. Banyaknya objek wisata menjadikan masyarakat mempunyai hobi jalan-jalan atau biasanya dikenal dengan sebutan *healing* ke tempat-tempat yang baru, serta mempunyai hobi wisata kuliner untuk mencicipi makanan tradisional dari berbagai daerah. Ketelitian memilih dan mengenali makanan untuk masyarakat yang sedang menjalankan program diet atau yang sedang menjaga pola makan untuk hidup sehat sangat diperlukan. Semakin maraknya penggunaan aplikasi *smartphone* berbasis *Convolutional Neural Network* (CNN) menjadi salah satu solusi dalam membantu memilih dan mengenali gambar makanan secara efisien. EfficientNetV2 merupakan arsitektur pendatang baru dalam mengklasifikasikan gambar dari keluarga *Convolutional Neural Network* (CNN). Penelitian ini bertujuan membuat sebuah sistem aplikasi yang dapat melakukan pengenalan gambar makanan tradisional Indonesia dengan menggunakan arsitektur EfficientNetV2. Model EfficientNetV2 pada penelitian ini menggunakan tiga arsitektur yakni EfficientNetV2-S(21k), EfficientNetV2_M_21k, dan EfficientNetV2_L_21k. Dataset gambar makanan tradisional berasal dari dua kategori sumber data yakni dari *crawling* menggunakan *Google Image*, dan pengambilan gambar secara langsung menggunakan kamera *Smartphone*. Masing-masing dataset mempunyai 18 kelas dengan total 1800 gambar per kategori. Dataset diambil dari dua kategori untuk mendapatkan nilai akurasi terbaik dan membandingkan tingkat akurasi. Hasil penelitian yang telah dilakukan mendapatkan nilai akurasi tertinggi dari model EfficientNetV2_S_21k (sumber data kamera *smartphone*) dengan nilai akurasi pelatihan sebesar 99,8%, dan nilai akurasi validasi sebesar 100%, sedangkan untuk nilai akurasi pengujian tertinggi sebesar 100% dari model EfficientNetV2_S_21k dan EfficientNetV2_M_21k (sumber data kamera *smartphone*). Model yang dipilih dengan nilai akurasi terbaik diimplementasikan ke aplikasi android menggunakan TensorFlow Lite Model Maker, dan Android Studio.

Kata kunci

makanan tradisional, indonesia, convolutional neural network, klasifikasi gambar, efficientnetv2

Abstract

Traditional Food Classification Model Based Efficientnetv2 Architecture

Indonesia is one of the countries that has a wide variety of traditional foods and fascinating tourist objects. The large number of tourist objects makes people have a hobby of traveling or usually known as healing to new places, as well as having a hobby of culinary tourism to taste traditional food from various regions. Accuracy in selecting and recognizing food for people who are running a diet program or who are maintaining a healthy diet is very necessary. The increasingly widespread use of smartphone applications based on Convolutional Neural Network (CNN) is one of the solutions to help select and recognize food images efficiently. EfficientNetV2 is a newcomer architecture in image classification from the Convolutional Neural Network (CNN) family. This study aims to create an application system that can perform image recognition of traditional Indonesian food using the EfficientNetV2 architecture. The EfficientNetV2 model in this study uses three architectures, namely EfficientNetV2-S(21k), EfficientNetV2_M_21k and EfficientNetV2_L_21k. The traditional food image dataset comes from two categories of data sources, namely from crawling using Google Images, and direct image capture using a smartphone camera. Each dataset has 18 classes with a total of 1800 images per category. The dataset is taken from two categories to get the best accuracy value and compare the level of accuracy. The results of the research that has been carried out obtain the highest accuracy value from the EfficientNetV2_S_21k model (smartphone camera data source) with a training accuracy value of 99.8%, and a validation accuracy value of 100%, while for the highest testing accuracy value of 100% from the EfficientNetV2_S_21k and EfficientNetV2_M_21k models (Smartphone camera data source). The selected model with the best accuracy value is implemented into an android application using TensorFlow Lite Model Maker and Android Studio.

Keywords

traditional food, indonesia, convolutional neural network, image classification, efficientnetv2

Pernyataan Keaslian Tulisan

Dengan ini saya menyatakan bahwa tesis ini merupakan tulisan asli dari penulis, dan tidak berisi material yang telah diterbitkan sebelumnya atau tulisan dari penulis lain terkecuali referensi atas material tersebut telah disebutkan dalam tesis. Apabila ada kontribusi dari penulis lain dalam tesis ini, maka penulis lain tersebut secara eksplisit telah disebutkan dalam tesis ini.

Dengan ini saya juga menyatakan bahwa segala kontribusi dari pihak lain terhadap tesis ini, termasuk bantuan analisis statistik, desain survei, analisis data, prosedur teknis yang bersifat signifikan, dan segala bentuk aktivitas penelitian yang dipergunakan atau dilaporkan dalam tesis ini telah secara eksplisit disebutkan dalam tesis ini.

Segala bentuk hak cipta yang terdapat dalam material dokumen tesis ini berada dalam kepemilikan pemilik hak cipta masing-masing. Apabila dibutuhkan, penulis juga telah mendapatkan izin dari pemilik hak cipta untuk menggunakan ulang materialnya dalam tesis ini.

Yogyakarta, Juni 2023

Erin Eka Citra, S.Kom

Daftar Publikasi

Citra, E. E., Fudholi, D. H., & Dewa, C. K. (2023). Implementasi Arsitektur EfficientNetV2 untuk Klasifikasi Gambar Makanan Tradisional Indonesia. Jurnal Media Informatika Budidarma.

Kontributor	Jenis Kontribusi
Erin Eka Citra	Memberikan ide (60%) Mendesain eksperimen (100%) Menulis *paper* (100%)
Dhomas Hatta Fudholi	Memberikan ide (35%) Mereview dan menganalisis *paper* (90%)
Chandra Kusuma Dewa	Memberikan ide (25%) Mereview dan menganalisis *paper* (75%)

Halaman Kontribusi

Terima kasih penulis sampaikan kepada Bapak Dhomas Hatta Fudholi dan Bapak Chandra Kusuma Dewa selaku pembimbing Thesis yang telah membimbing dan memberikan saran kepada penulis dalam melaksanakan penelitian ini.

Halaman Persembahan

Alhamdulillahirobbil'alamin, segala puji bagi Allah Subhanahu wa Ta'ala yang telah memberikan rahmat dan karunia-Nya, sehingga penulis mampu menyelesaikan Thesis ini. Thesis ini saya persembahkan untuk :

1. Kedua orang tua (bapak dan mama) dan keluarga besar yang senantiasa memberikan dukungan dan doa.
2. Kak SS, thesis fighter (mas Akmal dan nina), dan teman-teman yang telah ikut membantu dan terlibat baik secara langsung maupun tidak langsung dalam melaksanakan penelitian ini.

Jazakumullahu Khairan Katsira.

Kata Pengantar

Assalamu'alaikum warahmatullahi wabarakatuh

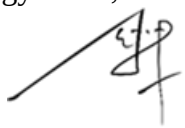
Alhamdulillahirobbil'alamin, segala puji bagi Allah Subhanahu wa Ta'ala yang telah memberikan rahmat dan karunia-Nya, sehingga penulis mampu menyelesaikan Thesis ini. Shalawat serta salam tidak lupa dihaturkan kepada Nabi Agung Muhammad *shalallahu 'alaihi wa sallam* beserta keluarga dan para pengikutnya.

Pada kesempatan ini penulis mengucapkan terima kasih kepada semua pihak yang telah membantu menyelesaikan Thesis ini baik secara langsung maupun tidak langsung. Secara khusus penulis hendak menyampaikan ucapan terima kasih kepada Bapak Dhomas Hatta Fudholi dan Bapak Chandra Kusuma Dewa selaku pembimbing Thesis yang telah membimbing, memberikan saran, diskusi, dan meluangkan waktu selama bimbingan kepada penulis dalam melaksanakan penelitian ini.

Penulis menyadari bahwa Thesis ini jauh dari sempurna, oleh sebab itu kritik, saran dan usulan yang membangun sangat penulis harapkan untuk perbaikan di masa mendatang. Akhir kata semoga Thesis ini dapat memberikan manfaat bagi penulis dan pembaca umumnya.

Wassalamu'alaikum warahmatullahi wabarakatuh

Yogyakarta, Juni 2023



Erin Eka Citra, S.Kom

Daftar Isi

Daftar Tabel

Daftar Gambar

Glosarium

FLOP	- Floating point Operations Per Second
CNN	- Convolutional Neural Network
MBConv	- Mobile Inverted Bottleneck Convolution
Fused-MBConv	- Fused Mobile Inverted Bottleneck Convolution

BAB 1
Pendahuluan

1.1 Latar Belakang

Klasifikasi gambar sudah banyak digunakan dalam mengidentifikasi dan mengenali objek gambar pada berbagai aplikasi (Akter et al., 2018; Kolisnik et al., 2021; Pan et al., 2017). Klasifikasi gambar menjadi penting karena dapat memudahkan serta membantu banyak lini di masyarakat, seperti pada bidang kesehatan (Karthik et al., 2022; Mahaputri et al., 2022; Park et al., 2019; Udayana et al., 2020), bidang industri (Kolisnik et al., 2021; Medus et al., 2021; Yulianti et al., 2021), dan bidang keamanan (Ying et al., 2021). Pada bidang kesehatan, beberapa tahun terakhir klasifikasi gambar dengan dataset makanan untuk mengetahui estimasi kalori yang akan dikonsumsi menjadi pembahasan yang penting terutama untuk program diet (Akter et al., 2018; Mezgec & Seljak, 2017; Park et al., 2019).

Indonesia merupakan salah satu negara yang memiliki banyak aneka ragam makanan terutama makanan tradisional dan objek wisata yang mempesona. Banyaknya objek wisata menjadikan masyarakat mempunyai hobi jalan-jalan atau biasanya dikenal dengan sebutan *healing* ke tempat-tempat yang baru, serta mempunyai hobi wisata kuliner untuk mencicipi makanan tradisional dari berbagai daerah. Namun ketika berwisata kuliner saat menemui beraneka ragam makanan tradisional, membuat mereka harus lebih berhati-hati terutama yang sedang menjalankan program diet atau yang sedang menjaga pola makan untuk hidup sehat. Meningkatnya resiko pada kondisi kesehatan dapat disebabkan karena berbagai factor makanan yang dikonsumsi, terutama makanan tinggi gula (Baequny et al., 2015; Mahaputri et al., 2022).

Keakuratan *deep learning* dalam mengidentifikasi dan mengenali gambar dengan berbagai arsitektur sudah banyak dikenal. Arsitektur dari *deep learning* yang telah banyak digunakan dalam mengindentifikasi dan mengenali gambar yakni *Convolutional Neural Network* (CNN). Penelitian sebelumnya dengan arsitektur CNN menggunakan dataset makanan memiliki berbagai tujuan, seperti untuk pengklasifikasian citra makanan tradisional (Rohim et al., 2019), untuk program diet (Mahaputri et al., 2022; Mezgec & Seljak, 2017; Park et al., 2019), untuk memudahkan dikenali oleh wisatawan asing (Darojat et al., 2021), dan untuk penghitung kalori (Udayana et al., 2020).

EfficientNetV2 adalah pendatang baru dari keluarga *Convolutional Neural Network* (CNN). EfficientNetV2 memiliki keunggulan dalam akselerasi pelatihan dan parameter dengan memadukan *search* dan *scaling* pada *neural architecture* (Tan & Le, 2021). Penelitian yang telah menggunakan EfficientNetV2 yakni (Karthik et al., 2022) untuk mengembangkan system deteksi penyakit kulit, (Sunil et al., 2022) untuk deteksi penyakit tanaman Kapulaga, (Ye et al., 2022) untuk identifikasi penyakit Singkong, dan (Mahaputri et al., 2022) untuk pengenalan makanan Indonesia.

Penelitian ini bertujuan untuk mengklasifikasikan gambar dengan arsitektur EfficientNetV2 dalam mengidentifikasi dan mengenali nama makanan serta mengetahui tingkat akurasinya. Penelitian ini menggunakan dataset makanan tradisional Indonesia yang berasal dari *Google Images* dan pengambilan secara langsung menggunakan kamera *smartphone.* Harapannya klasifikasi gambar yang diaplikasikan dapat berguna dalam mengoptimalkan aplikasi penghitung kalori pada masa mendatang. Model klasifikasi tersebut akan dibangun dengan arsitektur EfficientNetV2 dan akan diujicobakan dengan keluarga arsitektur EfficientNetV2, yakni EfficientNetV2_S_21k, EfficientNetV2_M_21k, dan EfficientNetV2_L_21k. Sebagai bahan evaluasi, untuk menguji perfoma dari model EfficientNetV2 yang dibangun, penelitian ini memvalidasi model menggunakan pengujian akurasi, *confusion matrix, precision, recall,* dan *F1-score*. Model yang memiliki hasil pengujian akurasi terbaik akan dipilih untuk diimplementasikan pada perangkat android menggunakan TensorFlow Lite Model Maker, dan Android Studio.

1.2 Rumusan Masalah

Pada pembuatan klasifikasi gambar untuk mendapatkan nama gambar yang diprediksi dan hasil akurasi, penulis membuat beberapa rumusan masalah diantaranya:

a. Bagaimana melakukan pemodelan dan mengevaluasi model klasifikasi menggunakan arsitektur EfficientNetV2 untuk mendapatkan nilai akurasi terbaik dan mengklasifikasikan gambar?
b. Bagaimana hasil implementasi klasifikasi gambar Makanan Tradisional Indonesia pada pemodelan arsitektur EfficientNetV2?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah yang sudah dirumuskan untuk mengklasifikasikan gambar, maka dapat dibuat tujuan penelitian sebagai berikut:

a. Mengetahui hasil pemodelan dan evaluasi model klasifikasi dari penggunaan arsitektur EfficientNetV2 untuk mendapatkan nilai akurasi terbaik dan mengklasifikasikan gambar.
b. Mengetahui hasil implementasi dari klasifikasi gambar Makanan Tradisional Indonesia menggunakan arsitektur EfficientNetV2.

1.4 Batasan Penelitian

Dari penelitian yang dilakukan untuk klasifikasi gambar, digunakan beberapa Batasan penelitian, yakni:

a. Dataset makanan tradisional Indonesia yang digunakan hanya terdiri dari 18 kelas, yakni: Kue Tat, Pendap, Tempoyak, Lemang Tapai, Bakpia Pathok, Gethuk, Gatot, Tiwul, Gudeg, Lupis, Martabak, Sate Padang, Mie Celor, Pempek, Cireng, Cilok, Nasi Kebuli, dan Ketoprak.
b. Sumber data hanya terdiri dari 2 kategori yakni *crawling* dari Google Image, dan pengambilan secara langsung menggunakan kamera *smartphone*.
c. Model klasifikasi yang digunakan dari keluarga EfficientNetV2 terdiri dari tiga model, yakni: EfficientNetV2_S_21k, EfficientNetV2_M_21k, dan EfficientNetV2_L_21k.
d. Matriks evaluasi yang digunakan yakni: pengujian akurasi, *precision, recall,* dan *F1-score*.
e. Penerapan aplikasi Android hanya dapat berjalan pada perangkat Android dengan spesifikasi minimal Android 6, dan API 23.

1.5 Sistematika Penulisan

Pada penelitian ini, penulis menggunakan sistematika penulisan sebagai berikut.

BAB 1 PENDAHULUAN

Bagian Pendahuluan memuat latar belakang penelitian, rumusan masalah, tujuan penelitian, dan Batasan penelitian.

BAB 2 TINJAUAN PUSTAKA

Bagian Tinjauan Pustaka memuat landasan teori yang terdiri dari teori pendukung klasifikasi gambar dan kajian Pustaka terhadap penelitian-penelitian sebelumnya yang berkaitan dengan klasifikasi gambar.

BAB 3 METODOLOGI PENELITIAN

Bagian Metodologi Penelitian menyajikan metode atau Langkah-langkah yang digunakan pada penelitian yang memuat penjelasan mengenai dataset, *preprocessing* data, pemodelan, matriks yang digunakan untuk evaluasi model, implementasi model menggunakan aplikasi Android, dan skenario eksperimen.

BAB 4 HASIL DAN PEMBAHASAN

Bagian Hasil dan Pembahasan menyajikan hasil dan pembahasan dari pemodelan EfficientNetV2 untuk klasifikasi gambar. Pada bagian ini penulis menjelaskan hasil dari pengambilan data, *preprocessing* data, parameter penelitian, pemodelan EfficientNetV2 secara rinci dengan memasukkan *code* yang digunakan untuk mendapatkan data gambar, penerapan model klasifikasi gambar, hasil dari matriks evaluasi, dan implementasi model pada *smartphone* android.

BAB 5 PENUTUP

Bagian Penutup memuat kesimpulan serta saran untuk perbaikan kedepannya yang dirangkum penulis dari penelitian yang sudah dilakukan di bidang klasifikasi gambar makanan tradisional Indonesia.

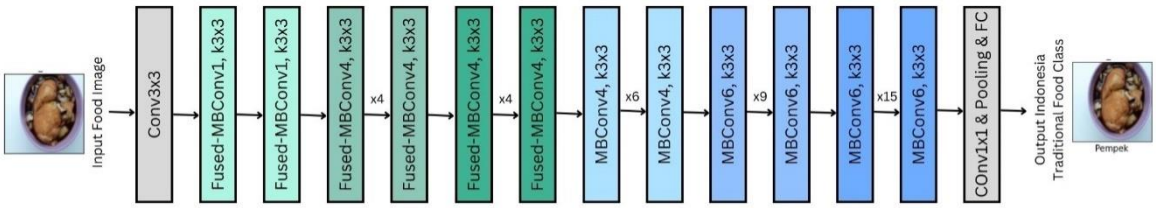
BAB 2
Tinjauan Pustaka

2.1 Landasan Teori

2.1.1 Image Classification

Image Classification adalah sebuah proses untuk menemukan insight dari gambar secara otomatis menggunakan kecerdasan buatan. *Image Classification* merupakan salah satu teknik pengolahan untuk mentransformasikan *input* yang berupa gambar dan menghasilkan *output* yang berupa gambar dengan pengolahan sinyal. *Image Classification* mempunyai beberapa macam proses, seperti proses *sampling*, kuantisasi, dan *noise.* Proses *sampling* adalah suatu proses yang bertujuan untuk mendapatkan warna *pixel* yang spesifik dari gambar. Kuantisasi adalah suatu proses yang bertujuan untuk mendapatkan warna secara keseluruhan dengan standar warna tertentu. *Noise* adalah sebuah gambar yang tidak diperlukan dari proses klasifikasi, dan dianggap hanya mengganggu kualitas gambar (Ramdani et al., 2020).

Penelitian *Image classification* belakangan ini banyak yang menggunakan model berbasis *deep learning* dan mampu mendapatkan hasil yang dapat mengklasifikasikan gambar sesuai dengan tujuan penelitian ini (Tan & Le, 2021). Gambar 2.1 merupakan contoh proses *image classification* menggunakan model EfficientNetV2_S. Gambar 2.1 menunjukkan proses *input* gambar berupa gambar makanan pempek yang dapat menghasilkan klasifikasi gambar dengan memunculkan nama dari kelas dataset yang bernama Pempek. Proses klasifikasi gambar menggunakan arsitektur EfficientNetV2_S yang terjadi pada Gambar 2.1 melalui beberapa tahapan layer, yakni *convolutional layer* dengan kernel berukuran 3x3, Fused-MBConv1 dengan kernel berukuran 3x3 sebanyak 2 layer, Fused-MBConv4 dengan kernel berukuran 3x3 sebanyak 8 layer, MBConv4 dengan kernel berukuran 3x3 sebanyak 6 layer, MBConv6 dengan kernel berukuran 3x3 sebanyak 24 layer, *convolutional layer* dengan kernel berukuran 1x1, *Pooling*, dan *FC (Fully Connected).*



Gambar 2.1 Contoh Proses Klasifikasi Gambar menggunakan Arsitektur EfficientNetV2_S

2.1.2 Convolutional Neural Network (CNN)

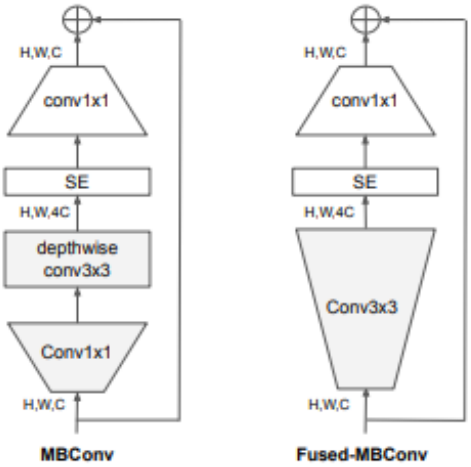
Convolutional Neural Network (CNN) merupakan satu dari beberapa arsitektur *deep learning* yang biasa digunakan untuk mengidentifikasi dan mengenali gambar. CNN memiliki fitur utama yang dapat mengetahui prediksi informasi dari gambar walaupun berada diposisi manapun pada *input.* CNN mempunyai dua bagian pokok yakni *classification* dan *feature learning. Feature learning* berfungsi untuk melakukan suatu proses *encoding* dari gambar menjadi bentuk numerik untuk memberikan gambaran objek. *Feature learning* memiliki beberapa *layer* yang berkolaborasi dalam memperoleh gambar, yakni *convolution layer, activation layer,* dan *pooling layer. Convolution layer* adalah layer yang memiliki fungsi untuk menemukan pola pada gambar dengan proses konvolusi (Ramdani et al., 2020). *Convolution layer* mempelajari pola visual dari data seperti tinggi, ketebalan, dan lebar (Mahaputri et al., 2022). *Activation layer* adalah layer yang memiliki fungsi meneruskan fitur utama ke layer selanjutnya. *Pooling layer* adalah layer yang memiliki fungsi untuk mengurangi bentuk dari *feature map* dan meningkatkan komputasi agar mengurangi terjadinya *overfitting. Overfitting* dapat terjadi apabila data latih tidak dapat digeneralisasi dengan baik, nilai *loss* berkurang, dan validasi nilai *loss* meningkat atau tidak berubah. Sedangkan *classification* berfungsi untuk mengklasifikasikan *neuron* yang dihasilkan pada proses *feature learning* dengan masing-masing *layer* yang saling terhubung. *Classification* mempunyai beberapa fungsi yakni *flatten* dan *fully connected layer. Flatten* memiliki fungsi sebagai masukan dari *fully connected layer* dengan mengaktivasi kembali *feature map* yang mengubah nilai array menjadi vektor (Mahaputri et al., 2022). *Fully connected layer* adalah layer terakhir pada proses klasifikasi (Ramdani et al., 2020). *Fully connected layer* dapat berfungsi dalam klasifikasi data secara linier dengan mengubah ukuran dimensi data menjadi satu dimensi. *Fully connected layer* terdiri dari beberapa *layer* yakni *hidden layer, activation function, output layer,* dan *loss function* (Mahaputri et al., 2022).

2.1.3 EfficientNet

EfficientNet pertama kali dikenalkan oleh Tan et al (Tan & Le, 2019). EfficientNet adalah pendatang baru dari *neural architecture* yang memiliki keunggulan dalam akurasi dan efisiensi dari *Convolutional Neural Network (CNN)* dengan metode *scalling* yang sederhana. Metode *scalling* yang digunakan pada penelitian ini yakni *depth, width,* dan *resolution.* EfficientNet menggunakan parameter dan FLOPs yang lebih sedikit dibandingkan *Convolutional Neural Network (CNN)* (Tan & Le, 2019).

2.1.4 EfficientNetV2

EfficientNetV2 pertama kali dikenalkan oleh Tan et al (Tan & Le, 2021). EfficientNetV2 adalah pendatang baru dari keluarga *convolutional network* yang memiliki keunggulan dalam akselerasi pelatihan dan parameter dengan memadukan *search* dan *scaling* pada *neural architecture.* Arsitektur EfficientNetV2 memiliki beberapa *layer* tambahan diawal yang merupakan ciri khas dari EfficientNetV2, yakni MBConv dan Fused-MBConv. *Layer* tambahan pada EfficientNetV2 diusulkan karena pada EfficientNet masih berfokus pada efisiensi parameter dan FLOP, sehingga dengan adanya *layer* tambahan dapat meningkatkan fungsi akselerator agar lebih optimal. *Layer* MBConv pada arsitektur EfficientNetV2 mempunyai ekspansi rasio yang lebih kecil untuk mengurangi *overhead* akses memori. Struktur MBConv dan Fused-MBConv memiliki sedikit perbedaan, yakni MBConv menggunakan depthwise conv3x3 dan Conv1x1, sedangkan Fused-MBConv menggunakan Conv3x3. Fused-MBConv berfungsi meningkatkan efisiensi komputasi dengan cara menggabungkan beberapa proses konvolusi dengan kernel 3x3. Arsitektur EfficientNetV2 juga mampu mendapatkan nilai akurasi yang lebih tinggi dengan parameter dan FLOP yang lebih sedikit dibandingkan arsitektur EfficientNet (Tan & Le, 2021). Gambar 2.2 menunjukkan struktur MBConv dan Fused-MBConv dari arsitektur EfficientNetV2.



Gambar 2.2 Struktur MBConv dan Fused-MBConv (Tan & Le, 2021)

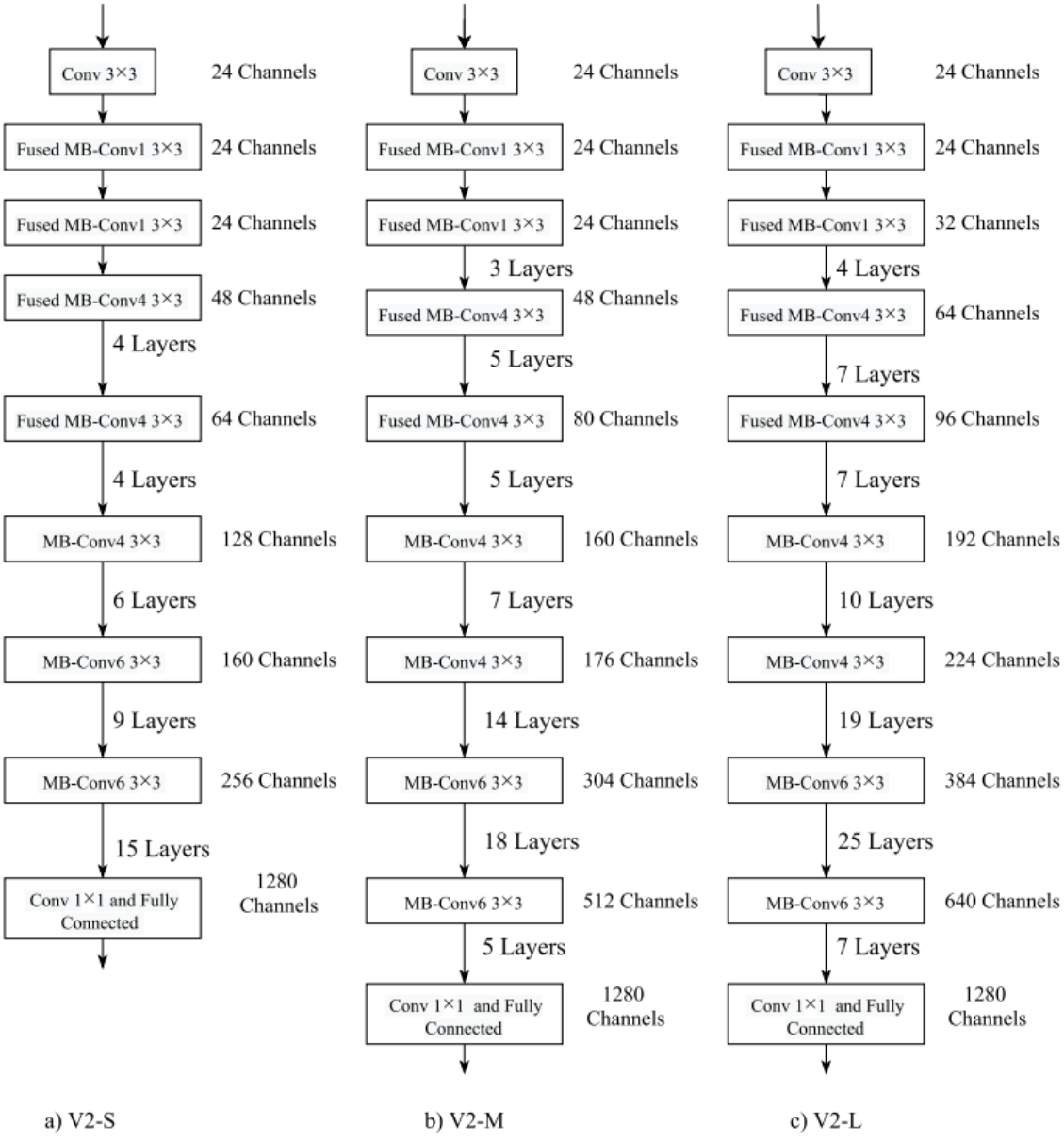
Pemodelan EfficientNetV2 pada penelitian ini mempunyai tiga arsitektur dasar, yakni EfficientNetV2_S_21k, EfficientNetV2_M_21k, dan EfficientNetV2_L_21k. Tabel 2.1 merupakan blok-blok arsitektur dari EfficientNetV2_S. Arsitektur EfficientNetV2_S menggunakan Fused-MBConv dan MBConv dengan rasio ekspansi {1, 4, dan 6} serta ukuran kernel 3x3. Blok-blok arsitektur pada Tabel 2.1 terdiri dari beberapa layer yakni, *convolutional layer* dengan kernel 3x3, Fused-MBConv dengan kernel 3x3, MBConv dengan kernel 3x3, *convolutional layer* dengan kernel 1x1, *pooling layer*, dan *FC (Fully Connected Layer)* (Tan & Le, 2021).

Tabel 2.1 Arsitektur EfficientNetV2-S (Tan & Le, 2021)

Stage	Operator	Stride	#Channels	Layers
0	Conv3x3	2	24	1
1	Fused-MBConv1, k3x3	1	24	2
2	Fused-MBConv4, k3x3	2	48	4
3	Fused-MBConv4, k3x3	2	64	4
4	MBConv4, k3x3, SE0.25	2	128	6
5	MBConv6, k3x3, SE0.25	1	160	9
6	MBConv6, k3x3, SE0.25	2	256	15
7	Conv1x1 & Pooling & FC	-	1280	1

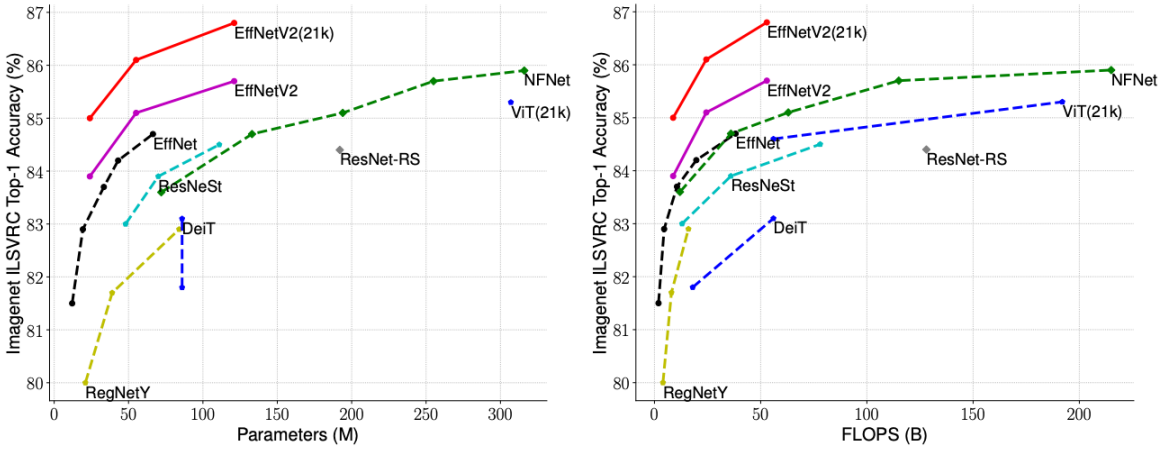
Gambar 2.3 menunjukkan struktur dari masing-masing arsitektur EfficientNetV2-S, EfficientNetV2-M, dan EfficientNetV2-L yang diimplementasikan saat pengklasifikasian

gambar. Arsitektur EfficientNetV2-S, EfficientNetV2-M, dan EfficientNetV2-L memiliki beberapa perbedaan, seperti jumlah *channels* dan layer yang digunakan, EfficientNetV2-S memiliki jumlah *channels* dan jumlah layer yang lebih sedikit dibandingkan EfficientNetV2-M dan EfficientNetV2-L. Ukuran EfficientNetV2 berbanding lurus dengan jumlah *channels* dan layer yang digunakan, semakin besar ukuran EfficientNetV2 maka semakin banyak jumlah *channels* dan layer yang digunakan. Total layer dari arsitektur EfficientNetV2_S berjumlah 42 layer, EfficientNetV2_M berjumlah 60 layer, dan EfficientNetV2L berjumlah 82 layer (Tan & Le, 2021).



Gambar 2.3 Arsitektur EfficientNetV2: a)EfficientNetV2-S, b)EfficientNetV2-M, c)EfficientNetV2-L (Sunil et al., 2022; Tan & Le, 2021)

Gambar 2.4 merupakan grafik perbandingan ukuran model, dan *Floating Points Performance Per Second* (FLOPs) EfficientNetV2(21k), EfficientNetV2, EfficientNet, dan arsitektur lainnya. Model yang memiliki tanda 21k mempunyai makna bahwa sebelumnya model telah dilatih menggunakan dataset ImageNet21k, untuk mengambil serta mengenali ciri dari gambar (Tan & Le, 2021). Gambar 2.4 menunjukkan bahwa EfficientNetV2(21k) mampu mencapai akurasi yang lebih tinggi dari model lainnya.



(a) Parameters (b) FLOPs

Gambar 2.4 Ukuran model dan FLOPs (Tan & Le, 2021)

2.1.5 Matriks Evaluasi

Matriks Evaluasi berfungsi untuk melihat hasil prediksi dari model secara menyeluruh sebagai bahan evaluasi dari hasil pengujian model. Penelitian ini menggunakan lima matriks evaluasi yang terdiri dari *accuracy*, *precision*, *recall*, dan *F1score*. Persamaan (1) merupakan rumus dari *accuracy* dan memperlihatkan cara dalam memperoleh nilai *accuracy*. Persamaan (2), (3), dan (4) memperlihatkan cara dalam memperoleh nilai *precision*, *recall*, dan *F1-score* (Karthik et al., 2022). Nilai akurasi (*accuracy*) bertujuan untuk menunjukkan tingkat korelasi hasil prediksi dengan hasil *real*. *Recall* bertujuan untuk menunjukkan tingkat pencapaian model dalam mengenali kembali sebuah objek. *Precision* bertujuan untuk menunjukkan tingkat ketepatan prediksi model dengan informasi yang diinginkan. *F1score* bertujuan untuk menunjukkan skor dari perbandingan rata-rata *precision* dan *recall* (Anam et al., 2021).

Istilah pada matriks evaluasi yang digunakan yakni *True Positive* (TP), *True Negative* (TN), *False Negative* (FN), dan *False Positive* (FP). *True Positive* (TP) adalah gambar spesimen yang dikategorikan dengan benar pada kelas tertentu. *True Negative* (TN) adalah

total kolom dan baris yang masih ada, tanpa kolom dan baris yang benar. *False Negative* (FN) adalah total nilai pada baris yang benar, tanpa TP. *False Positive* (FP) adalah total kolom yang benar, tanpa TP (Padilla et al., 2019). Berikut matriks evaluasi yang diaplikasikan pada penelitian ini :

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$F1score = \frac{2*(precision*recall)}{precision+recall} \quad (4)$$

2.1.6 Kajian Pustaka

Penelitian mengenai klasifikasi gambar makanan tradisional Indonesia beberapa tahun terakhir telah banyak dipublikasikan dengan berbagai model. Makanan tradisional memang menjadi salah satu tujuan wisata bagi sebagian masyarakat baik skala regional maupun Internasional dalam rangka mencoba hal yang baru bagi mereka atau dalam rangka mencintai produk lokal. Beberapa peneliti menggunakan makanan tradisional Indonesia sebagai dataset agar memudahkan dalam mengetahui jumlah kalori untuk program diet (Mahaputri et al., 2022; Udayana et al., 2020), dan untuk membantu agar lebih mudah dikenali oleh wisatawan asing (Darojat et al., 2021).

Penelitian (Rohim et al., 2019) mengaplikasikan makanan tradisional Indonesia untuk pengklasifikasian citra makanan menggunakan arsitektur CNN. Hasil evaluasi yang didapatkan yaitu 73% nilai presisi, 69% *recall*, dan 69% *F1score*. Kemudian (Kurnia et al., 2021) menggunakan dataset makanan Indonesia yaitu kue tradisional Indonesia yang menggunakan model CNN dan mendapatkan hasil akurasi sebesar 65%. Penelitian ini bertujuan untuk mengklasifikasikan kue tradisional Indonesia berbasis *computer vision.* (Darojat et al., 2021) mengaplikasikan dataset makanan tradisional Indonesia dalam rangka untuk klasifikasi citra agar lebih mudah dikenali oleh wisatawan asing. Penelitian ini memperoleh nilai akurasi sebesar 91% menggunakan arsitektur CNN. (Udayana et al., 2020) menggunakan dataset makanan Indonesia untuk mengenali gambar makanan sehingga dapat digunakan untuk pengembangan aplikasi penghitung kalori kedepannya. Penelitian ini mendapatkan hasil akurasi sebesar 88% dengan model CNN. Kemudian (Mahaputri et al., 2022) juga menggunakan dataset makanan Indonesia sebagai tahap awal untuk program diet.

Penelitian ini mendapatkan hasil akurasi sebesar 64% dengan model EfficientNetV2M, dan 59% dengan model EfficientNetB6.

Tabel 2.2 menampilkan perbandingan yang telah dilakukan pada penelitian ini dengan beberapa penelitian sebelumnya yang menggunakan dataset Makanan Indonesia. Perbandingan yang ditampilkan pada Tabel 2.2 terdiri dari model yang digunakan, total data gambar yang memuat jumlah kelas dan jumlah gambar, dan hasil akurasi dari penelitian sebelumnya. Tabel 2.2 menunjukkan bahwa total data dan penggunaan model yang digunakan pada penelitian memiliki pengaruh pada hasil akurasi yang didapatkan. Penelitian sebelumnya mampu mendapatkan nilai akurasi tertinggi sebesar 91% menggunakan model CNN, dengan total data 9 kelas yang berisi 900 gambar (Darojat et al., 2021).

Tabel 2.2 *State of the Art* penelitian

Referensi	**Dataset**	**Model**	**Total Data**	**Hasil Akurasi**
(Kurnia et al., 2021)	Kue Tradisional	CNN	8 kelas, 1.676 gambar	65%.
(Mahaputri et al., 2022)	Makanan Indonesia	Deep Convolutional Neural Network (DCNN) model EfficientNetB6 dan EfficientNetV2M	20 kelas, 1202 gambar	64% EfficientNetV2M, dan 59% EfficientNetB6
(Rohim et al., 2019)	Makanan Tradisional Indonesia	CNN	20 kelas, 380 gambar	73% presisi, 69% recall dan 69% Fscore.
(Darojat et al., 2021)	Makanan Khas Indonesia	CNN	9 kelas, 900 gambar	91%
(Udayana et al., 2020)	Makanan Umum Indonesia	CNN	10 kelas, 10.500 gambar	88%

Penelitian yang diusulkan	Makanan Tradisional Indonesia	EfficientNetV2_S_21k	18 kelas, 1.800 gambar	99.8%

EfficientNet adalah salah satu keluarga model baru dari CNN yang memiliki nilai akurasi dan tingkat efisiensi yang lebih baik (Tan & Le, 2019). Pada tahun 2021 (Tan & Le, 2021) mengeluarkan arsitektur baru dari EfficientNet, dikenalkan sebagai salah satu model yang memiliki waktu pelatihan yang lebih cepat dan penggunaan model yang lebih kecil.

Beberapa penelitian yang telah mengaplikasikan penggunaan model EfficientNetV2 yakni (Fan et al., 2021) menggunakan dataset BreakHis dengan model EfficientNetV2-S. Penelitian ini membuat system yang bertujuan untuk mengenali citra patologis kanker payudara dengan menambahkan chatbot *pra*-pelatihan. Model yang diusulkan diberi nama EfficientNetV2-SA. Hasil yang didapatkan sebesar 84,71%, sehingga mampu meningkatkan kinerja akurasi model. (Karthik et al., 2022) menerapkan model EfficientNetV2 dengan mengganti blok Squeeze and Excitation (SE) pada arsitektur EfficientNetV2 dengan blok Efficient Channel Attention (ECA), kemudian diberi nama Eff2Net**.** Penelitian ini membuat system yang bertujuan untuk mendeteksi penyakit kulit. Hasil akurasi yang didapatkan sebesar 84,70% menggunakan Eff2Net, dan 80,38% menggunakan EfficientNetV2.

(Ye et al., 2022) mengaplikasikan arsitektur EfficientNetV2 dengan menambahkan metode *transfer learning,* kemudian diberi nama PDRNet. Penelitian ini bertujuan untuk mengidentifikasi penyakit pada tanaman singkong. Hasil akurasi yang diperoleh sebesar 99,56% untuk PDRNet, dan 98,53% untuk EfficientNetV2. Model PDRNet mampu mengungguli model EfficientNetV2 untuk hasil akurasi yang didapatkan, namun membutuhkan waktu pelatihan yang lebih banyak. (Sunil et al., 2022) menerapkan EfficientNetV2 dengan dataset Cardamon dan PlantVillage. Penelitian ini bertujuan untuk mendeteksi penyakit pada tanaman kapulaga. Hasil akurasi yang didapatkan sebesar 98,26% menggunakan EfficientNetV2-L, dan 98,28% menggunakan EfficientNetV2-S.

Tabel 2.3 merupakan beberapa hasil penelitian sebelumnya yang telah menerapkan penggunaan model EfficientNetV2 dengan berbagai tujuan penelitian dan dataset yang berbeda. Perbandingan yang ditampilkan pada Tabel 2.3 terdiri dari model yang digunakan, total data gambar yang memuat jumlah kelas dan jumlah gambar, dan hasil akurasi dari penelitian sebelumnya. Tabel 2.3 menunjukkan bahwa EfficientNetV2 mampu menghasilkan nilai akurasi yang baik dalam mengklasifikasikan gambar dengan berbagai

tujuan dan dataset. Nilai tertinggi pada penelitian sebelumnya yang didapat menggunakan arsitektur EfficientNetV2 adalah 99.56% dengan dataset Daun Singkong (Ye et al., 2022). Penelitian yang diusulkan pada penelitian ini mampu mendapatkan nilai akurasi diatas nilai akurasi penelitian sebelumnya yakni 99.8% menggunakan dataset makanan tradisional Indonesia.

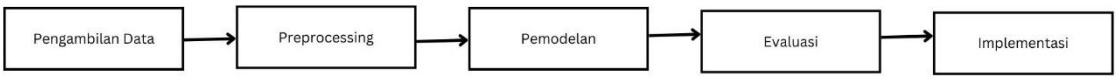
Tabel 2.3 Hasil implementasi penelitian menggunakan EfficientNetV2

Referensi	**Dataset**	**Model**	**Total Data**	**Hasil Akurasi**
(Fan et al., 2021)	Breast Cancer	EfficientNetV2-SA	8 kelas, 7.909 gambar	84.71%
(Karthik et al., 2022)	Penyakit Kulit	Eff2Net, EfficientNetV2	4 kelas, 13.864 gambar	84.70 Eff2Net, 80.38 EfficientNetV2
(Ye et al., 2022)	Daun Singkong	EfficientNetV2, PDRNet	5 kelas, 21.397 gambar	98.53% EfficientNetV2, 99.56% PDRNet
(Sunil et al., 2022)	Cardamon & Grape Plant	EfficientNetV2S, EfficientNetV2M, EfficientNetV2L, EfficientNet, CNN	7 kelas, 56.008 gambar	Hasil tertinggi yakni: 98.26% EfficientNetV2L, 98.28% EfficientNetV2S
Penelitian yang diusulkan	Makanan Tradisional Indonesia	EfficientNetV2_S_21k	18 kelas, 1.800 gambar	99.8%

BAB 3
Metodologi

3.1 Langkah Penelitian

Bab ini akan menguraikan metodologi penelitian yang akan digunakan pada *image classification* makanan tradisional Indonesia. Terdapat 5 langkah penelitian yakni, pengambilan data, *preprocessing*, pemodelan, evaluasi model, dan implementasi penelitian. Langkah penelitian tersebut dapat dilihat Gambar 3.1.



Gambar 3.1 Metodologi Penelitian

3.2 Uraian Metodologi

3.2.1 Pengambilan Data

Dataset pada penelitian *image classification* didapatkan dari pengambilan secara langsung menggunakan kamera *smartphone* dan data hasil *crawling* dari *google image.* Data yang diambil merupakan data gambar makanan tradisional Indonesia. Digunakan 18 kelas makanan tradisional yang terdiri dari 100 gambar setiap kelasnya. Total dataset yang digunakan berjumlah 1800 gambar untuk masing-masing kategori pengambilan gambar. Contoh dataset nama makanan dan asal daerah yang digunakan pada penelitian ini dapat dilihat dari Tabel 3.1. Data gambar yang berasal dari pengambilan langsung menggunakan kamera *smartphone* mengaplikasikan teknik pengambilan gambar dari berbagai *angle*, seperti pencahayaan dari dalam dan luar ruangan, dari letak pengambilan gambar yang berbeda-beda, serta beberapa dari dalam dan luar kemasan.

Tabel 3.1 Contoh dari Dataset Makanan Tradisional Indonesia

No	Gambar Makanan	Nama Makanan	Asal Makanan	Sumber Data
1		Kue Tat	Bengkulu	Kamera Smartphone

2		Pendap	Bengkulu	Kamera Smartphone
3		Tempoyak	Bengkulu	Google Image
4		Lemang Tapai	Bengkulu	Google Image
5		Bakpia Pathok	Yogyakarta	Google Image
6		Gethuk	Yogyakarta	Google Image
7		Gatot	Yogyakarta	Google Image
8		Tiwul	Yogyakarta	Google Image
9		Gudeg	Yogyakarta	Kamera Smartphone
10		Lupis	Yogyakarta	Kamera Smartphone
11		Martabak	Bangka	Kamera Smartphone
12		Sate Padang	Padang	Kamera Smartphone
13		Mie Celor	Palembang	Kamera Smartphone

14		Pempek	Palembang	Google Image
15		Cireng	Bandung	Google Image
16		Cilok	Bandung	Google Image
17		Nasi Kebuli	Lombok	Google Image
18		Ketoprak	Jakarta	Google Image

3.2.2 Preprocessing

Setelah dilakukan tahapan pengumpulan data gambar, selanjutnya dilakukan tahapan *preprocessing* data sebelum pengaplikasian ke model klasifikasi gambar. Tahap ini dilakukan *preprocessing* data yakni melakukan filterisasi gambar hasil *crawling* dari *google image*, melakukan *cropping* gambar, dan mengubah ukuran gambar menjadi standar berukuran 480x480. Tahapan *preprocessing* data sangat penting dalam meningkatkan nilai akurasi pada klasifikasi gambar (Akter et al., 2018). Gambar 3.2 merupakan contoh gambar sebelum dilakukan *preprocessing* dari dataset Google Image yang memiliki ukuran gambar belum standar. Gambar 3.3 merupakan dataset yang sudah dilakukan *preprocessing* dari dataset Google Image yang memiliki ukuran gambar sudah sesuai standar.

Gambar 3.2 Before preprocessing (dataset Google Image)

Gambar 3.3 After preprocessing (dataset Google Image)

3.2.3 Pemodelan

Pada tahap pemodelan data menggunakan arsitektur EfficientNetV2. Sebelum tahapan pemodelan dilakukan, terdapat tahap distribusi data dan *feature extraction*. Data yang telah dikumpulkan serta sudah melalui tahapan *preprocessing* data, kemudian dilakukan tahapan distribusi data yang terdiri dari data latih, data validasi, dan data uji. Dataset dibagi menjadi 90% data latih, serta 10% untuk data uji dan data validasi. Pembagian dataset memiliki pengaruh pada hasil akurasi yang akan didapatkan, seperti (Karthik et al., 2022) yang menerapkan pembagian dataset menjadi 90:10, dan mampu mendapatkan nilai akurasi sebesar 84,70%. Tabel 3.2 merupakan proses distribusi data yang telah dilakukan pada penelitian ini.

Tabel 3.2 Distribusi data

Sumber Data	Jumlah Kelas	Jumlah Data	Data Latih	Data Validasi	Data Uji
Google Image	18	1800	1620	72	108
Kamera	18	1800	1620	72	108

Tahapan selanjutnya yakni *feature extraction* menggunakan model EfficientNetV2_21k dengan dataset ImageNet21k. Pada tahap ini model klasifikasi dilatih terlebih dahulu menggunakan ImageNet21k. ImageNet21k mempunyai sekitar 21.841 kelas serta 13 juta gambar pelatihan (Tan & Le, 2021). Penggunaan *Feature Extraction* berfungsi dalam menangkap karakter dasar gambar seperti tepi gambar serta gumpalan sejak susunan awal jaringan (Akter et al., 2018).

Penelitian ini mengaplikasikan arsitektur EfficientNetV2 seperti penelitian aslinya (Tan & Le, 2021), tanpa mengganti model arsitektur yang substansial. Implementasi penelitian ini melengkapi model sebelumnya dengan mengubah hyperparameter agar memperoleh nilai akurasi dalam pengklasifikasian gambar yang lebih baik.

3.2.4 Evaluasi

Untuk proses evaluasi, digunakan empat matriks evaluasi untuk melihat performa model klasifikasi gambar yang sudah dilatih. Matriks evaluasi yang dilakukan pada klasifikasi gambar yakni pengujian akurasi, *precision, recall,* dan *F1-score.* Semakin besar nilai akurasi yang diperoleh, maka semakin baik proses klasifikasi gambar. Nilai akurasi memperlihatkan perbandingan proses klasifikasi gambar dengan benar (Ying et al., 2021).

3.2.5 Implementasi

Untuk proses implementasi, digunakan TensorFlow Lite Model Maker dalam *export* model menjadi format TfLite, kemudian dijalankan pada Android Studio. Hasil implementasi diterapkan pada perangkat Android dengan minimal spesifikasi Android 6, dan API 23.

3.2.6 Skenario Eksperimen

Image classification pada arsitektur EfficientNetV2 penelitian ini menerapkan pengujian parameter, yakni *learning rate, dropout rate,* dan epoch, untuk mendapatkan nilai akurasi terbaik. Dalam menentukan nilai epoch, penelitian ini mengaplikasikan arsitektur EfficientNetV2_S_21k dengan nilai epoch 20, dan *dropout* 0.5 dari dataset kamera *smartphone*. Nilai epoch didapatkan dari nilai akurasi terbaik berdasarkan nilai epoch 5, 10,

dan 20. Sedangkan untuk menentukan nilai *learning rate,* penelitian ini menggunakan nilai learning rate 0.005, dan dropout 0.5 dari arsitektur EfficientNetV2_S_21k dataset kamera *smartphone*. Nilai *learning rate* didapatkan dari nilai akurasi terbaik berdasarkan nilai *learning rate* 0.01, 0.05, 0.001, 0.005, dan 0.0001. *Dropout* memiliki fungsi dalam mengurangi terjadinya *overfitting* (Srivastava et al., 2014).

BAB 4
Hasil dan Pembahasan

Bab ini akan membahas mengenai hasil penelitian, tahapan dan proses eksperimen, evaluasi dan implementasi model. Hasil penelitian, tahapan dan proses eksperimen berisi proses pengumpulan dataset, preprocessing data, pengujian parameter epoch dan learning rate, dan klasifikasi gambar.

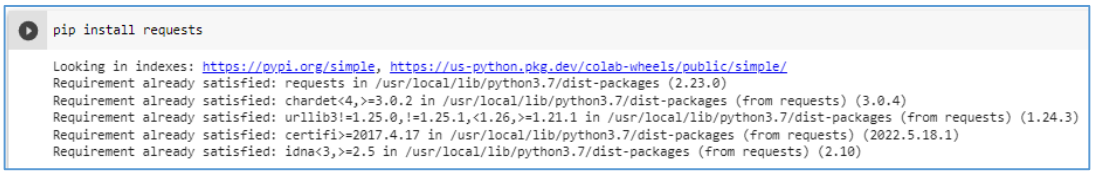
4.1 Hasil Pengambilan Data

4.1.1 Hasil Crawling Dataset Gambar dari Google Images

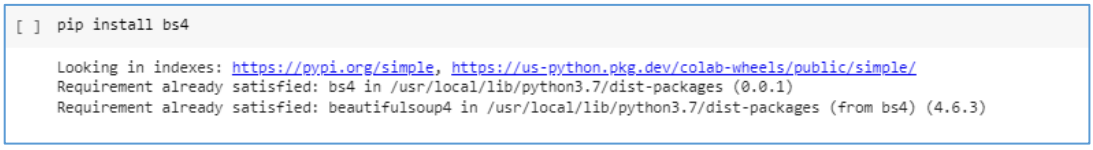
Proses *crawling* dataset memiliki beberapa tahapan, yakni:

1. Install dan Import Library

   Gambar 4.1 merupakan source code untuk melakukan instalasi library request. Gambar 4.2 merupakan source code untuk melakukan instalasi library bs4. Gambar 4.3 merupakan source code untuk melakukan instalasi library proxycrawl, dan Gambar 4.4 merupakan source code untuk melakukan import library request, BeautifulSoup, dan ProxyCrawlAPI.



Gambar 4.1 Install Library Request



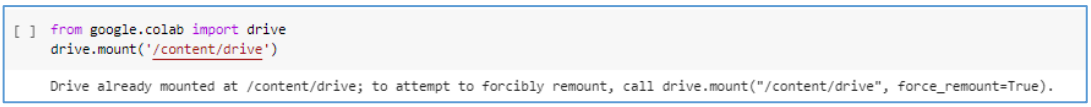
Gambar 4.2 Install Library bs4

Gambar 4.3 Install Library proxycrawl

Gambar 4.4 Import Library request, BeautifulSoup dari bs4, dan ProxyCrawlAPI dari library proxycrawl

2. Import Google Drive untuk penyimpanan dataset gambar

Gambar 4.5 merupakan source code untuk melakukan import Google Drive. Google Drive digunakan untuk penyimpanan dataset gambar agar lebih mudah untuk diakses.

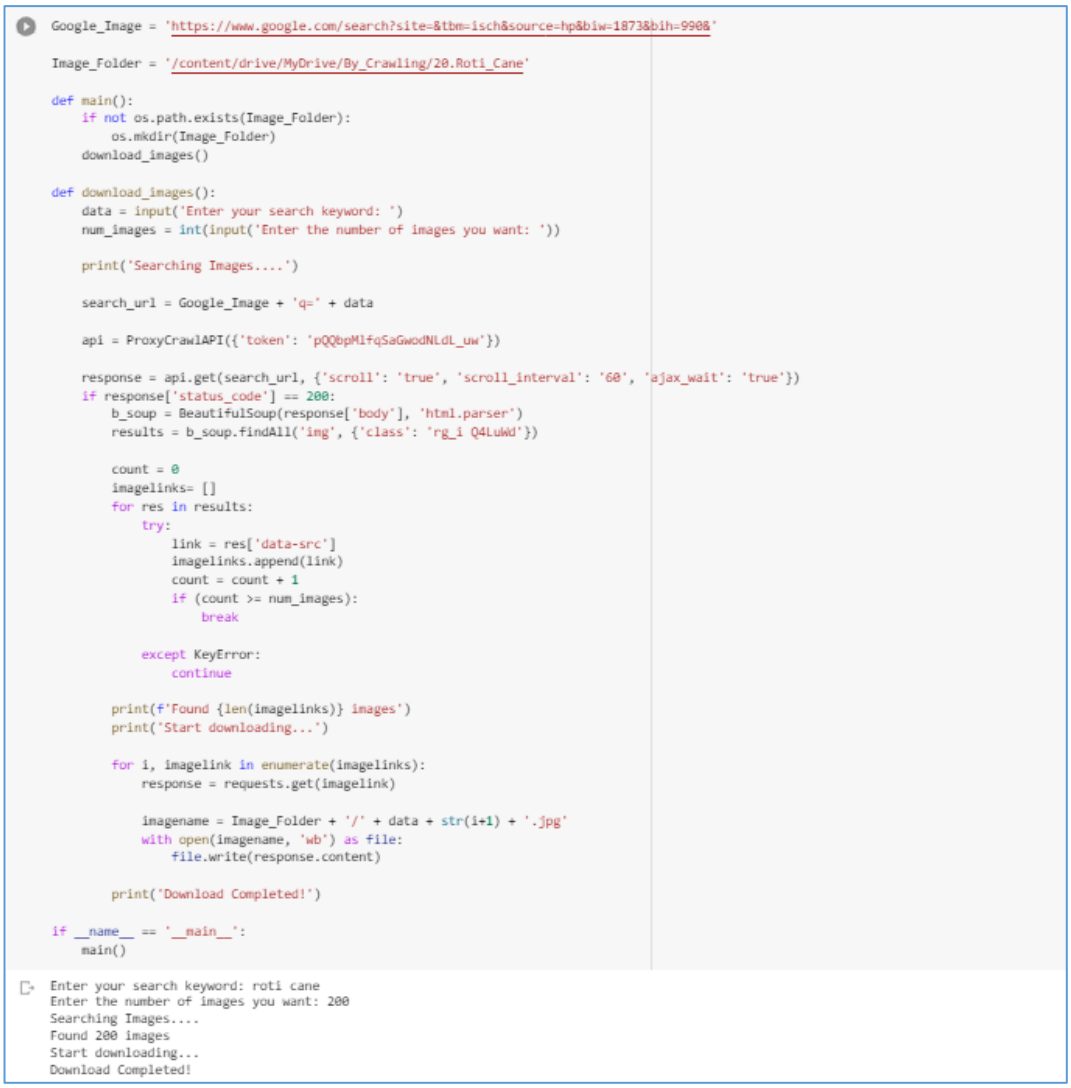


Gambar 4.5 Import Google Drive

3. Proses Crawling Dataset

Gambar 4.6 merupakan source code proses crawling. Pada proses crawling dataset gambar, terdapat beberapa tahapan yakni:

a. Download gambar dari Google Search menggunakan URL
b. Membuat folder di Google Drive serta masukkan link Drive tempat penyimpanan dataset
c. Membuat Token API pada ProxyCrawl.com, kemudian masukkan proxycrawl javascript token yang sudah dibuat pada source code
d. Setelah source code berhasil di compile, masukkan keyword yang hendak dicari dan masukkan jumlah gambar yang hendak didownload. Pada penelitian ini menggunakan 18 keyword nama makanan



```
Google_Image = 'https://www.google.com/search?site=&tbm=isch&source=hp&biw=1873&bih=990&'

Image_Folder = '/content/drive/MyDrive/By_Crawling/20.Roti_Cane'

def main():
    if not os.path.exists(Image_Folder):
        os.mkdir(Image_Folder)
    download_images()

def download_images():
    data = input('Enter your search keyword: ')
    num_images = int(input('Enter the number of images you want: '))

    print('Searching Images....')

    search_url = Google_Image + 'q=' + data

    api = ProxyCrawlAPI({'token': 'pQQbpMlfqSaGwodNLdL_uw'})

    response = api.get(search_url, {'scroll': 'true', 'scroll_interval': '60', 'ajax_wait': 'true'})
    if response['status_code'] == 200:
        b_soup = BeautifulSoup(response['body'], 'html.parser')
        results = b_soup.findAll('img', {'class': 'rg_i Q4LuWd'})

        count = 0
        imagelinks= []
        for res in results:
            try:
                link = res['data-src']
                imagelinks.append(link)
                count = count + 1
                if (count >= num_images):
                    break

            except KeyError:
                continue

        print(f'Found {len(imagelinks)} images')
        print('Start downloading...')

        for i, imagelink in enumerate(imagelinks):
            response = requests.get(imagelink)

            imagename = Image_Folder + '/' + data + str(i+1) + '.jpg'
            with open(imagename, 'wb') as file:
                file.write(response.content)

        print('Download Completed!')

if __name__ == '__main__':
    main()
```

```
Enter your search keyword: roti cane
Enter the number of images you want: 200
Searching Images....
Found 200 images
Start downloading...
Download Completed!
```

Gambar 4.6 Proses Crawling Gambar

4. Hasil Crawling Dataset

Gambar 4.7 merupakan contoh hasil crawling dari Google Images pada kelas dataset Kue Tat. Gambar 4.7 menunjukkan contoh 15 gambar Kue Tat dari total 18 kelas dataset yang berasal dari hasil crawling melalui Google Images. Gambar Kue Tat yang didapatkan dari hasil crawling masih belum dilakukan preprocessing data, sehingga masih memiliki ukuran yang belum standar dan beberapa gambar tidak sesuai dengan keyword yang dimasukkan.

Gambar 4.7 Contoh Hasil Crawling Kue Tat

4.1.2 Hasil Dataset Gambar dari Kamera Smartphone

Gambar 4.8 merupakan contoh hasil dataset pada kelas Kue Tat dari pengambilan secara langsung menggunakan kamera Smartphone. Gambar 4.8 menunjukkan bahwa hasil pengambilan secara langsung menggunakan kamera smartphone masih perlu dilakukan preprocessing data, dikarenakan hasil gambar yang diperoleh belum memiliki ukuran yang standar. Contoh hasil dataset pada kelas Kue Tat menunjukkan bahwa Teknik pengambilan gambar menggunakan kamera *smartphone* diambil dari berbagai *angle* (sudut pengambilan gambar), dari dalam dan luar ruangan, serta ada yang dari dalam dan luar kemasan.

Gambar 4.8 Contoh Dataset Kue Tat dari Kamera Smartphone

4.2 Hasil Preprocessing Data

Tahap ini dilakukan *preprocessing* data yakni melakukan filterisasi gambar hasil *crawling* dari *google image*, melakukan *cropping* gambar, dan mengubah ukuran gambar menjadi standar berukuran 480x480. Tahap filterisasi gambar dilakukan secara manual dengan menghapus gambar yang tidak sesuai keyword dataset. Tahap *cropping* juga dilakukan secara manual dengan memotong bagian gambar yang tidak sesuai dengan ciri/bagian fitur gambar.

Proses *crawling* dataset gambar dari *Google Images* dan dataset gambar dari kamera smartphone memiliki ukuran yang masih berbeda-beda, sehingga perlu dilakukan *resize images* pada *Google Colaboratory.* Tahapan *resize images* yang dilakukan, yakni:

1. Import Library Glob

Gambar 4.9 merupakan source code untuk melakukan import library Glob.

```
[1]  from PIL import Image
     import glob
```

Gambar 4.9 Import Library Glob

2. Import Google Drive untuk tempat penyimpanan dataset gambar

Gambar 4.10 merupakan source code untuk melakukan import Google Drive. Google drive digunakan untuk mengakses gambar yang akan dilakukan *resize images* dan untuk melakukan penyimpanan dataset gambar hasil *resize images.*

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

Gambar 4.10 Import Google Drive

3. Cek Image Path dan masukkan link untuk mengakses Image Path dari Google Drive

Gambar 4.11 merupakan source code untuk mengakses dataset pada Google Drive. Gambar 4.11 menunjukkan bahwa dataset disimpan didalam folder Google drive dengan nama By_Crawling, dan kelas Bakpia_Pathok. Gambar 4.12 merupakan source code untuk mengakses image path. Gambar 4.12 menunjukkan bahwa gambar pada folder By_crawling mempunyai ukuran 225x225 dengan *image mode* RGB.

```
[3]  img_path = r'/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja1.jpg'
```

Gambar 4.11 Mengakses Image Path

```
[4]  im=Image.open(img_path)
     print('{}'.format(im.format))
     print('size: {}'.format(im.size))
     print('image_mode: {}'.format(im.mode))
     im.show()
```

```
JPEG
size: (225, 225)
image_mode: RGB
```

Gambar 4.12 Image Path dari data hasil crawling yang menampilkan size asli gambar sebelum dilakukan resize images dan image mode

4. Melakukan proses resize images

Gambar 4.13 merupakan tampilan list dataset gambar sebelum dilakukan proses resize images. Contoh dataset yang akan dilakukan resize images pada Gambar 4.13 berasal dari Google drive dengan nama folder By_Crawling yang berarti dataset berasal dari crawling melalui Google Images, dan nama kelas Bakpia_Pathok. Gambar 4.14 merupakan proses resize images serta ukuran yang hendak distandarkan menjadi 480x480.

```
[5]  image_list = []
     resized_images = []
```

```
for filename in glob.glob(r'/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/*.jpg'):
    print(filename)
    img = Image.open(filename)
    image_list.append(img)
```

```
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja81.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja80.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja83.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja44.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja47.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja34.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja69.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja43.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja39.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja38.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja57.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja64.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja50.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja74.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja62.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja31.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja54.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja52.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja75.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja55.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja78.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja40.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja87.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja72.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja76.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja61.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja5.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja24.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja13.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja170.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja11.jpg
/content/drive/MyDrive/By_Crawling/Bakpia_Pathok/bakpia pathok jogja21.jpg
```

Gambar 4.13 List dataset sebelum resize images

```
[8]  for image in image_list:
         image = image.resize((480,480))
         resized_images.append(image)
```

Gambar 4.14 Proses resize images

5. Masukkan link folder untuk menyimpan dataset gambar yang sudah dilakukan resize images

Gambar 4.15 merupakan source code untuk menyimpan dataset gambar yang sudah dilakukan resize. Gambar 4.15 menunjukkan bahwa hasil resize images disimpan pada Google Drive dengan nama folder Resize_Crawling, dan nama kelas dataset Bakpia_Pathok. Nama masing-masing gambar hasil resize images akan berurutan dari angka 1 sampai 100 dengan nama kelas didepannya, seperti Bakpia_Pathok1, dan Bakpia_Pathok100.

```
[9]  for (i, new) in enumerate(resized_images):
         new.save('{}{}{}'.format(r'/content/drive/MyDrive/Resize_crawling/Bakpia_Pathok/Bakpia_Pathok',i+1,'.jpg'))
```

Gambar 4.15 Proses menyimpan dataset yang sudah dilakukan resize images

6. Hasil Resize Images

Gambar 4.16 merupakan contoh hasil resize images dataset Bakpia Pathok dari hasil crawling. Contoh dataset Bakpia Pathok pada Gambar 4.16 sudah memiliki ukuran yang standar yakni 480x480.

Gambar 4.16 Contoh hasil resize images Bakpia Pathok

4.3 Hasil Eksperimen Learning Rate

Tabel 4.1 merupakan table untuk menentukan nilai *learning rate*. Parameter yang digunakan untuk menentukan nilai *learning rate* menggunakan pengujian nilai akurasi dan nilai loss. Berdasarkan Tabel 4.1 dapat dilihat bahwa nilai *learning rate* terbaik adalah 0.005 dengan nilai akurasi tertinggi dan nilai *loss* terendah. Nilai akurasi tertinggi yang didapatkan *learning rate* 0.005 yakni 95.8%, dengan nilai *loss* terendah yakni 0.8786, diantara nilai *learning rate* 0.01, 0.05, 0.001, dan 0.0001.

Tabel 4.1 Learning Rate

Learning Rate	Accuracy	Loss
0.01	95%	0.9922
0.05	84.9%	6.0510
0.001	91.6%	0.9726
0.005	95.8%	0.8786
0.0001	45.1%	2.1331

4.4 Hasil Eksperimen Epoch

Tabel 4.2 merupakan tabel untuk menentukan nilai epoch. Parameter yang digunakan untuk menentukan nilai epoch menggunakan pengujian nilai akurasi dan nilai loss. Tabel 4.2 menunjukkan bahwa nilai epoch terbaik adalah epoch 20, dengan nilai akurasi tertinggi dan nilai *loss* terendah. Nilai akurasi tertinggi yang didapatkan yakni 99.8% dan nilai *loss* terendah yakni 0.7088 diantara nilai epoch 5 dan 10.

Tabel 4.2 Epoch

Learning rate 0,005		
Epoch	Accuracy	Loss
5	95.8%	0.8786
10	98.4%	0.7768
20	99.8%	0.7088

4.5 Proses Klasifikasi Gambar

Klasifikasi gambar yang dilakukan pada penelitian ini melalui beberapa tahapan. Contoh arsitektur yang digunakan pada proses klasifikasi gambar ini yakni arsitektur EfficientNetV2_L_21k dari dataset kamera *smartphone*. Tahapan yang dilakukan yakni :

1. Install dan import Library

   Gambar 4.17 merupakan source code untuk menginstal library yang digunakan, dan Gambar 4.18 merupakan source code untuk import library. Library yang digunakan yakni seedir, tensorflow, tflite_model_maker, matplotlib, dan sklearn.

```
[1]  !pip install seedir
     !pip install tensorflow
     !pip install tflite_model_maker
     !pip install matplotlib
     !pip install sklearn
```

Gambar 4.17 Install Library

```
[2]  import os
     import seedir as sd

[3]  import tensorflow as tf
     assert tf.__version__.startswith('2')

[4]  from tflite_model_maker import image_classifier
     from tflite_model_maker.image_classifier import DataLoader
     from tflite_model_maker.config import QuantizationConfig
     import matplotlib.pyplot as plt
     from sklearn.metrics import confusion_matrix, classification_report
     from google.colab import drive
```

Gambar 4.18 Import Library

2. Buka image path pada Google Drive untuk mengakses dataset

Gambar 4.19 merupakan source code untuk mendapatkan akses dataset pada Google Drive. Folder penyimpanan dataset gambar yang diakses melalui google drive yakni folder Resize_camera. Folder resize_camera merupakan folder yang berisi dataset hasil resize gambar dari kamera *smartphone.*

```
[5]  drive.mount('/content/drive')
     image_path = r'drive/MyDrive/Resize_camera'
     !pwd

     Mounted at /content/drive
     /content
```

Gambar 4.19 Akses Google Drive

3. Split dataset menjadi data latih, data validasi, dan data uji

Gambar 4.20 merupakan source code untuk membagi label menjadi all_label, samples_origin_label, dan predict_label, kemudian membagi dataset menjadi data latih (train_data), data uji (test_data), dan data validasi (validation_data). Dataset dibagi menjadi 90% data latih, sedangkan 10% nya untuk data uji (0,6) dan data validasi (0,4).

```
[6]  all_label = []
     samples_origin_label = []
     predicts_label = []

[7]  data = DataLoader.from_folder(image_path)
     train_data, split_data = data.split(0.9)
     test_data, validation_data = split_data.split(0.6)
```

```
[8]  train_data_length = len(train_data)
     validation_data_length = len(validation_data)
     test_data_length = len(test_data)
```

```
[9]  print('Train Data : '+ str(train_data_length))
     print('Validation Data : '+ str(validation_data_length))
     print('Test Data : '+ str(test_data_length))

     Train Data : 1620
     Validation Data : 72
     Test Data : 108
```

Gambar 4.20 Split Dataset

4. Menampilkan preview dataset

Gambar 4.21 merupakan source code untuk menampilkan contoh gambar dari dataset. Sample_preview merupakan contoh dari dataset yang akan coba ditampilkan. Contoh dari dataset yang akan ditampilkan berjumlah 25 gambar. Samples berisi jumlah dataset yang akan dilakukan percobaan proses klasifikasi gambar, dan berjumlah 100 gambar.

```
[10] samples_preview = test_data.gen_dataset().unbatch().take(25)
     samples = test_data.gen_dataset().unbatch().take(100)
```

```
[11] print('Showing 25 sample images...')
     plt.figure(figsize=(10,10))
     plt.title("Sample Images")
     for i, (image, label) in enumerate(samples_preview):
         plt.subplot(5,5,i+1)
         plt.xticks([])
         plt.yticks([])
         plt.grid(False)
         plt.imshow(image.numpy(), cmap=plt.cm.gray)
         plt.xlabel(test_data.index_to_label[label.numpy()])
     plt.show()
```

Gambar 4.21 Menampilkan contoh gambar dataset

5. Masukkan image classifier dari arsitektur EfficientNetV2

Gambar 4.22 merupakan source code image classifier. Uri_model_spec merupakan *code* yang berisi url untuk mengakses model EfficientNetV2. Model_spec.input_image_shape merupakan ukuran dari gambar yang akan diuji cobakan.

```
[12] uri_model_spec = 'https://tfhub.dev/google/imagenet/efficientnet_v2_imagenet21k_s/feature_vector/2'
     model_spec = image_classifier.ModelSpec(
         uri=uri_model_spec)
     model_spec.input_image_shape = [480, 480]
     print('Model spec loaded')

     Model spec loaded
```

Gambar 4.22 Image Classifier

6. Tentukan nilai hyperparameter

Gambar 4.23 merupakan source code untuk menentukan parameter. Parameter yang digunakan terdiri dari tiga parameter, yakni epoch, dropout_rate, dan learning_rate. Nilai epoch yang digunakan adalah 20. Dropout_rate yang digunakan adalah 0.5. Learning_rate yang digunakan adalah 0.005.

```
epochs = 20
dropout_rate = 0.5
learning_rate = 0.005
```

Gambar 4.23 Parameter penelitian

7. Pemodelan

Gambar 4.24 merupakan source code untuk melakukan pemodelan menggunakan arsitektur EfficientNetV2. Source code model yang digunakan berisi image classifier yang akan dibuat dengan parameter yang sudah ditentukan sebelumnya. Hasil akhir nilai akurasi dan loss dari data latih dan data validasi akan muncul setelah *code* epoch ke 20 berhasil *running* serta muncul tulisan “Training finished”.

```
[14] model = image_classifier.create(train_data, model_spec=model_spec,
                                     epochs=epochs, dropout_rate=dropout_rate, learning_rate=learning_rate,
                                     validation_data=validation_data)
     history = model.history
     print("Training finished")
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 hub_keras_layer_v1v2 (HubKe  (None, 1280)             20331360
 rasLayerV1V2)

 dropout (Dropout)           (None, 1280)              0

 dense (Dense)               (None, 18)                23058

=================================================================
Total params: 20,354,418
Trainable params: 23,058
Non-trainable params: 20,331,360
```

```
Epoch 1/20
50/50 [==============================] - 68s 922ms/step - loss: 2.0113 - accuracy: 0.6531 - val_loss: 1.1045 - val_accuracy: 0.9444
Epoch 2/20
50/50 [==============================] - 28s 557ms/step - loss: 1.0951 - accuracy: 0.9350 - val_loss: 0.8374 - val_accuracy: 1.0000
Epoch 3/20
50/50 [==============================] - 28s 552ms/step - loss: 0.9913 - accuracy: 0.9388 - val_loss: 0.7786 - val_accuracy: 1.0000
Epoch 4/20
50/50 [==============================] - 29s 566ms/step - loss: 0.9065 - accuracy: 0.9556 - val_loss: 0.7358 - val_accuracy: 1.0000
Epoch 5/20
50/50 [==============================] - 28s 556ms/step - loss: 0.8567 - accuracy: 0.9688 - val_loss: 0.7247 - val_accuracy: 1.0000
Epoch 6/20
50/50 [==============================] - 29s 586ms/step - loss: 0.8395 - accuracy: 0.9631 - val_loss: 0.7040 - val_accuracy: 1.0000
Epoch 7/20
50/50 [==============================] - 28s 559ms/step - loss: 0.8103 - accuracy: 0.9750 - val_loss: 0.6848 - val_accuracy: 1.0000
Epoch 8/20
50/50 [==============================] - 30s 589ms/step - loss: 0.8056 - accuracy: 0.9781 - val_loss: 0.6931 - val_accuracy: 1.0000
Epoch 9/20
50/50 [==============================] - 28s 558ms/step - loss: 0.7753 - accuracy: 0.9900 - val_loss: 0.6721 - val_accuracy: 1.0000
Epoch 10/20
50/50 [==============================] - 30s 591ms/step - loss: 0.7763 - accuracy: 0.9850 - val_loss: 0.6726 - val_accuracy: 1.0000
Epoch 11/20
50/50 [==============================] - 28s 561ms/step - loss: 0.7643 - accuracy: 0.9862 - val_loss: 0.6626 - val_accuracy: 1.0000
Epoch 12/20
50/50 [==============================] - 30s 591ms/step - loss: 0.7521 - accuracy: 0.9856 - val_loss: 0.6633 - val_accuracy: 1.0000
Epoch 13/20
50/50 [==============================] - 28s 563ms/step - loss: 0.7421 - accuracy: 0.9894 - val_loss: 0.6575 - val_accuracy: 1.0000
Epoch 14/20
50/50 [==============================] - 30s 588ms/step - loss: 0.7491 - accuracy: 0.9837 - val_loss: 0.6664 - val_accuracy: 1.0000
Epoch 15/20
50/50 [==============================] - 28s 564ms/step - loss: 0.7272 - accuracy: 0.9919 - val_loss: 0.6487 - val_accuracy: 1.0000
Epoch 16/20
50/50 [==============================] - 30s 591ms/step - loss: 0.7262 - accuracy: 0.9962 - val_loss: 0.6544 - val_accuracy: 1.0000
Epoch 17/20
50/50 [==============================] - 28s 562ms/step - loss: 0.7240 - accuracy: 0.9919 - val_loss: 0.6501 - val_accuracy: 1.0000
Epoch 18/20
50/50 [==============================] - 30s 592ms/step - loss: 0.7144 - accuracy: 0.9956 - val_loss: 0.6382 - val_accuracy: 1.0000
Epoch 19/20
50/50 [==============================] - 29s 573ms/step - loss: 0.7112 - accuracy: 0.9956 - val_loss: 0.6460 - val_accuracy: 1.0000
Epoch 20/20
50/50 [==============================] - 28s 559ms/step - loss: 0.7050 - accuracy: 0.9956 - val_loss: 0.6506 - val_accuracy: 1.0000
Training finished
```

Gambar 4.24 Pemodelan EfficientNetV2

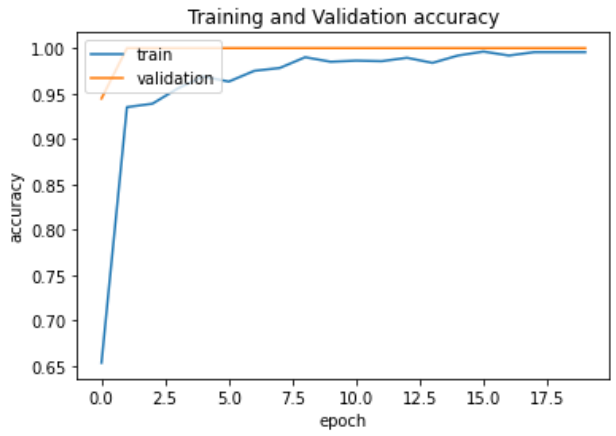
8. Visualisasi pemodelan

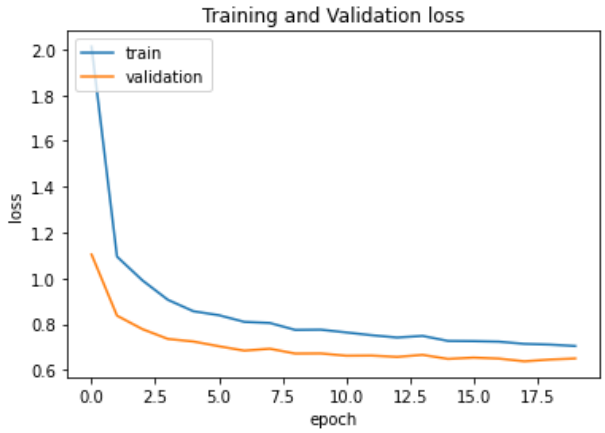
Gambar 4.25 merupakan source code untuk melakukan visualiasi hasil dari pemodelan yang sudah dibuat. Visualisasi pemodelan yang dibuat menggunakan table plot. Plot yang dibuat menampilkan visualisasi dari hasil epoch 20 serta hasil akurasi data latih dan data validasi. Garis pada hasil visualisasi yang berwarna biru menunjukkan hasil dari data latih, sedangkan garis yang berwarna merah menunjukkan hasil dari data validasi.

```python
def plot_history(history, key):
    epochs_range = range(epochs)
    key_val = history.history[key]
    val = history.history['val_'+key]
    plt.plot(epochs_range, key_val, label='Training '+key)
    plt.plot(epochs_range, val, label='Validation '+key)
    plt.title("Training and Validation "+key)
    plt.ylabel(key)
    plt.xlabel("epoch")
    plt.legend(["train", "validation"], loc="upper left")
    plt.show()
```

```python
def get_label_color(val1, val2):
  if val1 == val2:
    return 'black'
  else:
    return 'red'
```

```python
plot_history(history, 'accuracy') #do plot history accuracy
plot_history(history, 'loss') #do plot history loss
```





Gambar 4.25 Visualisasi hasil pemodelan

9. Pengujian akurasi dari pemodelan EfficientNetV2

Gambar 4.26 merupakan source code untuk melakukan pengujian akurasi dari hasil pemodelan. Hasil pengujian menampilkan nilai akurasi dan nilai loss yang akan

digunakan untuk evaluasi model. Nilai akurasi yang didapatkan sebesar 1.0000 dan nilai loss sebesar 0.6499.

```
[18] print("Evaluating Accuracy and Loss...")
     loss, accuracy = model.evaluate(test_data)

     Evaluating Accuracy and Loss...
     4/4 [==============================] - 5s 719ms/step - loss: 0.6499 - accuracy: 1.0000
```

Gambar 4.26 Pengujian EfficientNetV2

10. Uji coba model image classification yang telah dibuat

Gambar 4.27 merupakan source code untuk melakukan uji coba prediksi image classification yang telah dibuat menggunakan arsitektur EfficientNetV2. Uji coba prediksi gambar yang dilakukan akan ditampilkan dengan ukuran gambar 30x30. Hasil uji coba akan menampilkan gambar yang diprediksi beserta keterangan yang berisi *predicted* dan *real. Predicted* merupakan nama gambar yang diprediksi, serta *real* merupakan nama asli dari gambar yang diprediksi. Keterangan yang berisi nama asli dari gambar digunakan untuk membantu mengenali apabila terjadi kesalahan dalam memprediksi gambar.

```
[19] print("Generating prediction food figure...")
     plt.figure(figsize=(30, 30))
     predicts = model.predict_top_k(test_data)
     for i, (image, label) in enumerate(samples):
         ax = plt.subplot(10, 10, i+1)
         plt.xticks([])
         plt.yticks([])
         plt.grid(False)
         plt.imshow(image.numpy(), cmap=plt.cm.gray)
         predict_label = predicts[i][0][0]
         predicts_label.append(predict_label)
         test_label = test_data.index_to_label[label.numpy()]
         samples_origin_label.append(test_label)
         color = get_label_color(predict_label, test_label)
         ax.xaxis.label.set_color(color)
         plt.xlabel('Predicted: %s' % predict_label+'\nReal: %s' % test_label)
     plt.show()
```

Gambar 4.27 Proses klasifikasi gambar

11. Melihat semua daftar label dari kelas dataset

Gambar 4.28 merupakan source code untuk melihat label dari kelas yang ada pada dataset gambar. *Code* All_label merupakan *code* yang berisi daftar/*list* dari contoh label asli pada dataset.

```
[20] all_label = list(dict.fromkeys(samples_origin_label))
     print(all_label)
```

Gambar 4.28 List semua label

12. Melakukan evaluasi menggunakan matriks evaluasi

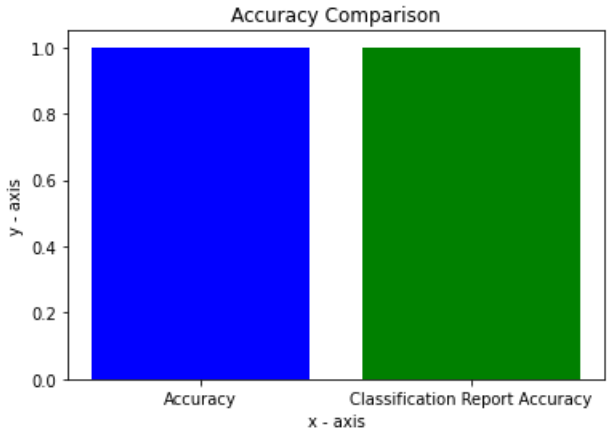
Gambar 4.29 merupakan source code untuk membuat dan melakukan evaluasi pemodelan menggunakan matriks evaluasi. Matriks evaluasi yang digunakan yakni *accuracy, precision, recall,* dan F1-score. Visualisasi *accuracy comparison* merupakan hasil visualisasi yang menampilkan perbandingan nilai akurasi pada data uji dan nilai akurasi pada matriks evaluasi. Hasil visualiasi nilai akurasi menunjukkan bahwa nilai akurasi pada data uji dan matriks evaluasi tidak memiliki perbedaan atau *error rate*.

```
[22] print("All Label : ")
     print(all_label)
     print("Origin Label : ")
     print(samples_origin_label)
     print("Predicted Label : ")
     print(predicts_label)
```

```
[23] cr = classification_report(samples_origin_label, predicts_label, target_names=all_label)
     print(cr)
     cr_with_dict = classification_report(samples_origin_label, predicts_label, target_names=all_label, output_dict=True)
     cr_accuracy = cr_with_dict['accuracy']
```

	precision	recall	f1-score	support
Lemang_Tapai	1.00	1.00	1.00	2
Tiwul	1.00	1.00	1.00	2
Bakpia_Pathok	1.00	1.00	1.00	2
Lupis	1.00	1.00	1.00	5
Gatot	1.00	1.00	1.00	3
Pempek	1.00	1.00	1.00	9
Mie_Celor	1.00	1.00	1.00	5
Kue_Tat	1.00	1.00	1.00	11
Cilok	1.00	1.00	1.00	8
Ketoprak	1.00	1.00	1.00	2
Cireng	1.00	1.00	1.00	7
Gethuk	1.00	1.00	1.00	7
Sate_Padang	1.00	1.00	1.00	6
Pendap	1.00	1.00	1.00	10
Martabak	1.00	1.00	1.00	6
Gudeg	1.00	1.00	1.00	4
Nasi_Kebuli	1.00	1.00	1.00	3
Tempoyak	1.00	1.00	1.00	8
accuracy			1.00	100
macro avg	1.00	1.00	1.00	100
weighted avg	1.00	1.00	1.00	100

```
[24] left = [1, 2]
     height = [accuracy, cr_accuracy]
     tick_label = ['Accuracy', 'Classification Report Accuracy']
     plt.bar(left, height, tick_label=tick_label, width=0.8, color=['blue', 'green'])
     plt.xlabel('x - axis')
     plt.ylabel('y - axis')
     plt.title('Accuracy Comparison')
     plt.show()
```



Gambar 4.29 Matriks Evaluasi

4.6 Hasil Eksperimen Nilai Akurasi Model

Tabel 4.3 merupakan perbandingan hasil nilai akurasi dan nilai *loss* dari masing-masing arsitektur EfficientNetV2 berdasarkan tiga pengklasifikasi model. Berdasakan Tabel 4.3 hasil pelatihan EfficientNetV2_S_21k dari dataset kamera mampu mencapai nilai akurasi tertinggi yakni sebesar 99,8% dengan nilai *loss* terendah yakni 0,71, dan hasil validasi mampu mencapai nilai akurasi sebesar 100% dengan nilai *loss* yakni 0,63. Sedangkan untuk hasil pengujian akurasi, EfficientNetV2_M_21k dari dataset kamera mendapatkan nilai sebesar 100% dengan nilai *loss* terendah yakni 0,64.

EfficientNetV2_S_21k memiliki jumlah layer dan parameter yang lebih kecil, dibandingkan EfficientNetV2_M_21k dan EfficientNetV2_L_21k. Jumlah layer dan parameter dari model yang digunakan menjadi salah satu penentu dalam menentukan kecocokan model dengan dataset. Selain itu, jumlah dataset, kompleksitas masalah, sumber daya komputasi serta karakteristik dari dataset makanan tradisional yang digunakan pada penelitian ini juga menentukan nilai akurasi serta kecocokan model (Tan & Le, 2021). Berdasarkan Tabel 4.3 nilai akurasi terbaik didapatkan dari model EfficientNetV2_S_21k, sehingga menunjukkan bahwa dengan jumlah dataset dan karakteristik makanan tradisional

yang digunakan pada penelitian ini lebih cocok dengan arsitektur pada model EfficientNetV2_S_21k.

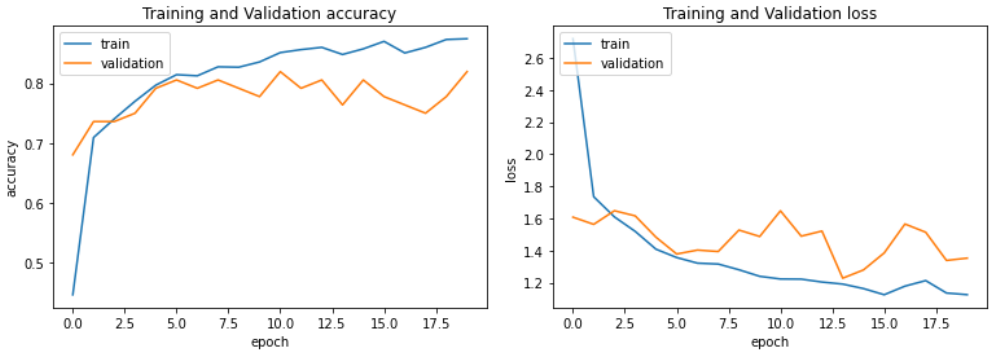
Tabel 4.3 Hasil nilai akurasi dan nilai loss menggunakan EfficientNetV2

Model	Dataset	Data Latih		Data Validasi		Data Uji	
		Loss	Acc (%)	Loss	Acc (%)	Loss	Acc (%)
EffNetV2_S_21k	Kamera	0.71	99.8	0.63	100	0.65	100
	Google Images	1.13	87.4	1.35	81.9	1.32	79.6
EffNetV2_M_21k	Kamera	0.71	99.7	0.63	100	0.64	100
	Google Images	1.08	88.4	1.43	76.4	1.47	85.2
EffNetV2_L_21k	Kamera	0.71	99.7	0.68	98.6	0.67	99.1
	Google Images	1.11	88.6	1.74	72.2	1.47	0.79

Gambar 4.30 merupakan gambar visualisasi dari hasil pemodelan EfficientNetV2_S_21k. Gambar 4.31 merupakan gambar visualisasi dari hasil pemodelan EfficientNetV2_M_21k. Gambar 4.32 merupakan gambar visualisasi dari hasil pemodelan EfficientNetV2_L_21k. Dataset yang digunakan untuk visualiasi pada Gambar 4.30, 4.31, dan 4.32 berasal dari Google Image yang merupakan hasil dari Tabel 4.3. Gambar 4.30, 4.31, dan 4.32 menunjukkan grafik nilai akurasi dan *loss* dari epoch 1 hingga epoch 20. Grafik yang ditampilkan menunjukkan bahwa nilai akurasi data latih dari model EfficientNetV2_S_21k, EfficientNetV2_M_21k, dan EfficientNetV2_L_21k cenderung naik serta nilai *loss* cenderung menurun pada setiap *epoch*, sedangkan nilai akurasi dan nilai *loss* dari data validasi cenderung fluktuatif.

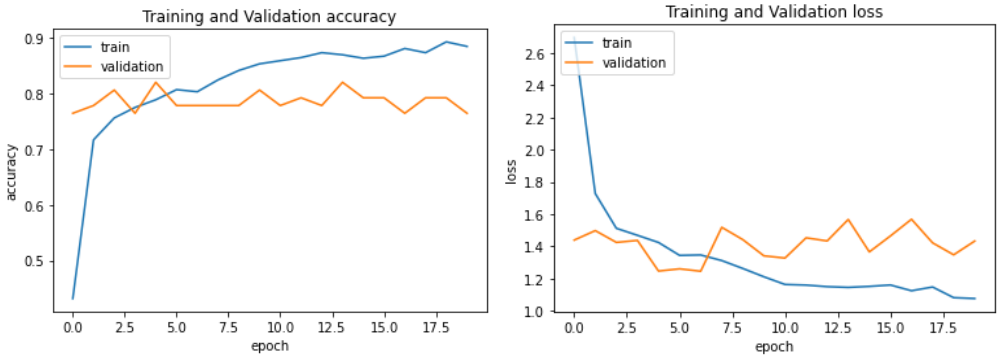
Gambar 4.30, 4.31, dan 4.32 juga menunjukkan perbandingan grafik nilai akurasi data latih yang lebih tinggi dibandingkan data validasi, serta nilai *loss* data latih yang lebih rendah dibandingkan data validasi. Hal ini menunjukkan bahwa model mengalami *overfitting*. Masalah *overfitting* dapat terjadi dikarenakan model terlalu spesifik pada data latih sehingga menyebabkan kurang baik dalam mengeneralisasi dan mengenali data baru dengan lebih baik.

a. EfficientNetV2_S_21k



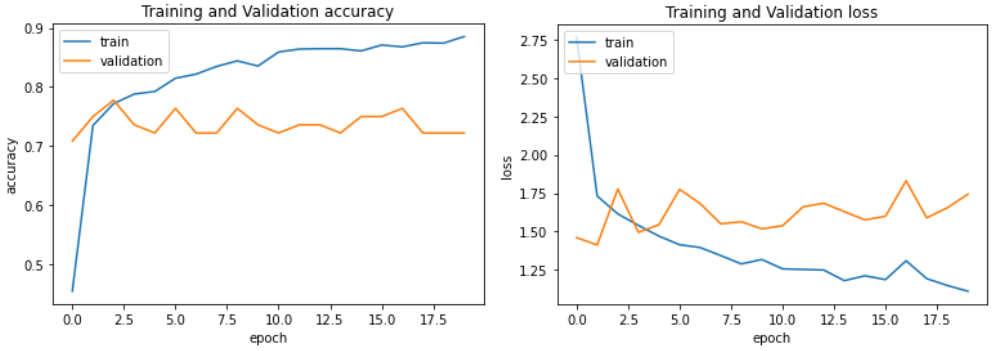
Gambar 4.30 Grafik nilai akurasi dan nilai loss data latih dan data validasi EfficientNetV2_S_21k (dataset google image)

b. EfficientNetV2_M_21k



Gambar 4.31 Grafik nilai akurasi dan nilai loss data latih dan data validasi EfficientNetV2_M_21k (dataset google image)

c. EfficientNetV2_L_21k

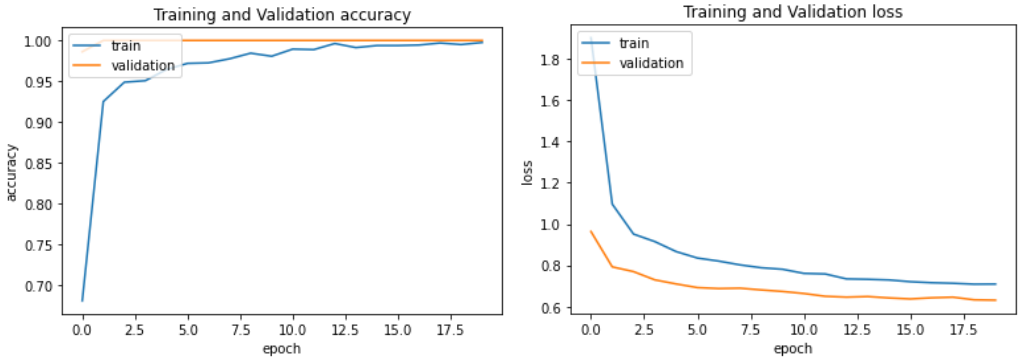


Gambar 4.32 Grafik nilai akurasi dan nilai loss data latih dan data validasi EfficientNetV2_L_21k (dataset google image)

Gambar 4.33 merupakan gambar visualisasi dari hasil pemodelan EfficientNetV2_S_21k. Gambar 4.34 merupakan gambar visualisasi dari hasil pemodelan EfficientNetV2_M_21k. Gambar 4.35 merupakan gambar visualisasi dari hasil pemodelan EfficientNetV2_L_21k. Dataset yang digunakan untuk visualiasi pada Gambar 4.33, 4.34, dan 4.35 berasal dari kamera *smartphone* yang merupakan hasil dari Tabel 4.3. Gambar 4.33, 4.34, dan 4.35 menunjukkan grafik nilai akurasi dan *loss* dari epoch 1 hingga epoch 20. Grafik yang ditampilkan menunjukkan bahwa nilai akurasi data latih dari model EfficientNetV2_S_21k, EfficientNetV2_M_21k, dan EfficientNetV2_L_21k cenderung naik serta nilai *loss* cenderung menurun, sedangkan nilai akurasi dari data validasi cenderung stabil dan nilai *loss* cenderung menurun.

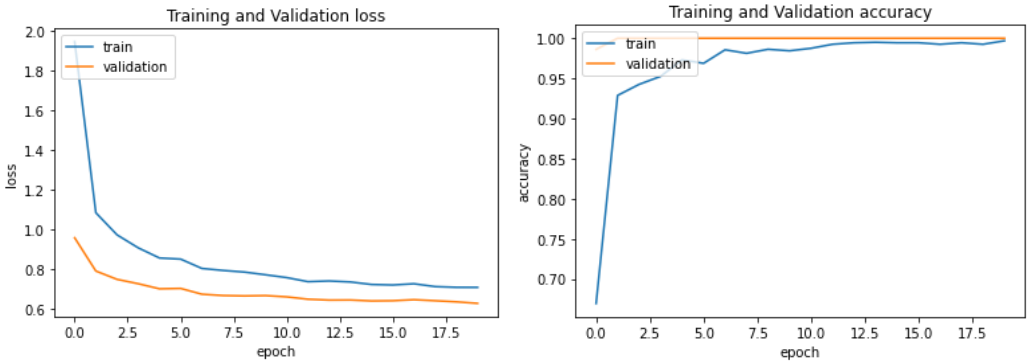
Gambar 4.33 dan 4.34 menunjukkan perbandingan grafik nilai akurasi data latih dan data validasi yang memiliki sedikit perbedaan, yakni nilai akurasi pada data validasi lebih tinggi dibandingkan data latih, namun tidak terlalu signifikan. Gambar 4.35 menunjukkan perbandingan grafik nilai akurasi data latih dan data validasi yang memiliki sedikit perbedaan, yakni pada *epoch* ke 10 nilai data latih cenderung naik melebihi data validasi sehingga menunjukkan adanya sedikit *overfitting*. Sedangkan pada Gambar 4.33, 4.34, dan 4.35, grafik nilai *loss* pada data latih dan data validasi cenderung menurun hingga akhir *epoch*, serta nilai *loss* pada data validasi cenderung lebih rendah dibandingkan data latih. Hal ini menunjukkan bahwa model mengalami *underfitting*. Masalah *underfitting* dapat terjadi dikarenakan model tidak dapat mempelajari dengan baik data latih (Abdulfattah et al., 2021), dan karena terbatasnya variasi pada data latih.

a. EfficientNetV2_S_21k



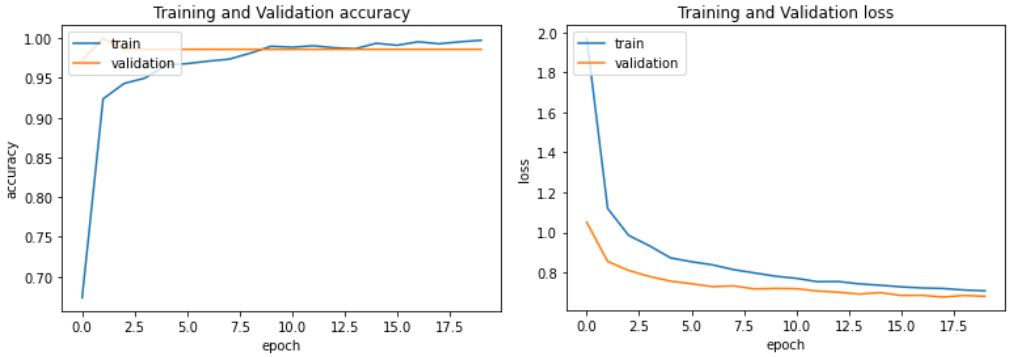
Gambar 4.33 Grafik nilai akurasi dan nilai loss data latih dan data validasi EfficientNetV2_S_21k (dataset kamera *smartphone*)

b. EfficientNetV2_M_21k



Gambar 4.34 Grafik nilai akurasi dan nilai loss data latih dan data validasi EfficientNetV2_M_21k (dataset kamera *smartphone*)

c. EfficientNetV2_L_21k



Gambar 4.35 Grafik nilai akurasi dan nilai loss data latih dan data validasi EfficientNetV2_L_21k (dataset kamera *smartphone*)

4.7 Hasil Klasifikasi Gambar

Gambar 4.36 merupakan hasil pengujian *image classification* EfficientNetV2_S_21k (dataset Kamera *Smartphone*) pada Google Colaboratory sebelum diimplementasikan pada perangkat Android. Pengujian Gambar 4.36 menunjukkan bahwa hasil klasifikasi dari arsitektur EfficientNetV2_S_21k mampu menampilkan nama makanan yang berhasil diprediksi. Hasil prediksi menampilkan gambar makanan yang diprediksi keterangan yang berisi *predicted* dan *real name*. *Predicted* adalah nama hasil prediksi, sedangkan *real* adalah nama asli dari makanan yang diprediksi.

Gambar 4.36 Hasil Prediksi *Image Classification* pada Google Colab. Tulisan predicted dibawah gambar menunjukkan hasil prediksi nama makanan, dan tulisan real menunjukkan nama asli dari nama makanan yang diprediksi.

4.8 Evaluasi Klasifikasi Gambar

Tabel 4.4 adalah hasil evaluasi nilai *accuracy, precision, recall,* dan F1Score dari data uji menggunakan arsitektur EfficientNetV2_S_21k (dataset kamera). Tabel 4.4 menunjukkan bahwa secara keseluruhan hasil evaluasi nilai *accuracy, precision, recall,* dan F1Score yang didapatkan sebesar 100%. Support merupakan jumlah data gambar yang dilakukan uji evaluasi antar kelas dataset. Total gambar yang dilakukan evaluasi yakni 100 gambar yang terbagi kedalam 18 kelas dataset.

Nilai rata-rata *precision, recall,* dan F1score pada hasil evaluasi data uji EfficientNetV2_S_21k mendapatkan nilai 1,00. Hal ini menunjukkan bahwa masing-masing kelas memiliki nilai akurasi yang bagus dengan nilai prediksi yang dapat diklasifikasikan dengan benar. Nilai rata-rata 1,00 didapatkan karena dataset yang digunakan pada model EfficientNetV2_S_21k memiliki kompleksitas dan keseimbangan jumlah gambar pada masing-masing kelas, tetapi memiliki variasi data yang minimum sehingga nilai akurasi yang didapatkan lebih tinggi.

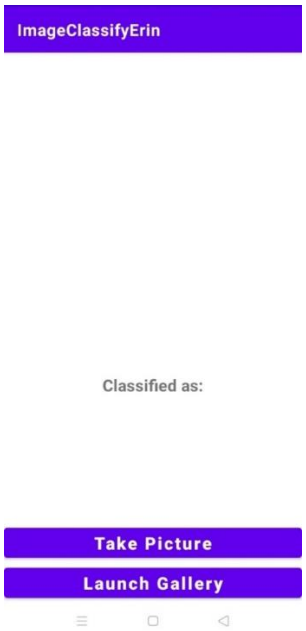
Tabel 4.4 Hasil evaluasi data uji EfficientNetV2_S_21k (dataset kamera)

Nama Makanan	Precision	Recall	F1-Score	Support
Lemang_Tapai	1.00	1.00	1.00	2
Tiwul	1.00	1.00	1.00	2
Bakpia_Pathok	1.00	1.00	1.00	2
Lupis	1.00	1.00	1.00	5
Gatot	1.00	1.00	1.00	3
Pempek	1.00	1.00	1.00	9
Mie_Celor	1.00	1.00	1.00	5

Kue_Tat	1.00	1.00	1.00	11
Cilok	1.00	1.00	1.00	8
Ketoprak	1.00	1.00	1.00	2
Cireng	1.00	1.00	1.00	7
Gethuk	1.00	1.00	1.00	7
Sate_Padang	1.00	1.00	1.00	6
Pendap	1.00	1.00	1.00	10
Martabak	1.00	1.00	1.00	6
Gudeg	1.00	1.00	1.00	4
Nasi_Kebuli	1.00	1.00	1.00	3
Tempoyak	1.00	1.00	1.00	8
Accuracy			1.00	100
Macro Avg	1.00	1.00	1.00	100
Weighted Avg	1.00	1.00	1.00	100

4.9 Implementasi

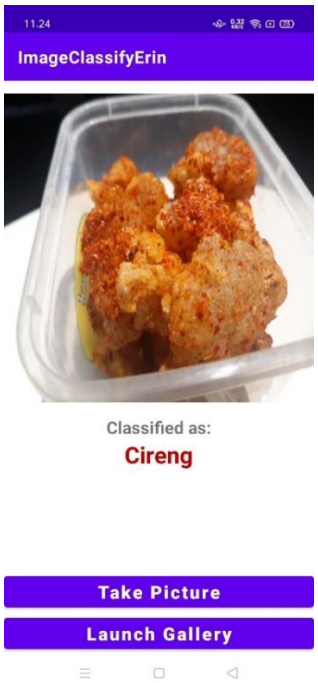
Gambar 4.37 dan 4.38 merupakan tampilan implementasi model dari arsitektur EfficientNetV2_S_21k dari dataset kamera *smartphone* pada perangkat Android dengan spesifikasi Android 10, Merk Oppo, dan Ram 3,00 GB. Gambar 4.37 menunjukkan tampilan awal aplikasi klasifikasi gambar yang berisi tombol menu *take picture* dan *launch gallery.* Tombol menu *take picture* berfungsi apabila hendak mengambil gambar secara langsung menggunakan kamera *smartphone,* sedangkan tombol menu *launch gallery* berfungsi apabila hendak mengambil gambar melalui penyimpanan galeri. Gambar 4.38 merupakan tampilan aplikasi apabila hendak menggunakan menu *take picture.* Gambar 4.39 merupakan tampilan aplikasi yang sudah *running* dan menampilkan hasil prediksi gambar nama makanan tradisional Indonesia (kelas Pempek) menggunakan menu Take Picture. Gambar 4.40 merupakan tampilan aplikasi yang sudah *running* dan menampilkan hasil prediksi gambar nama makanan tradisional Indonesia (kelas Cireng) menggunakan menu Launch Gallery.



Gambar 4.37 Tampilan Awal Aplikasi Image Classification

Gambar 4.38 Tampilan menggunakan Take Picture

Gambar 4.39 Tampilan Hasil Klasifikasi Gambar Makanan (Kelas Pempek) menggunakan Take Picture

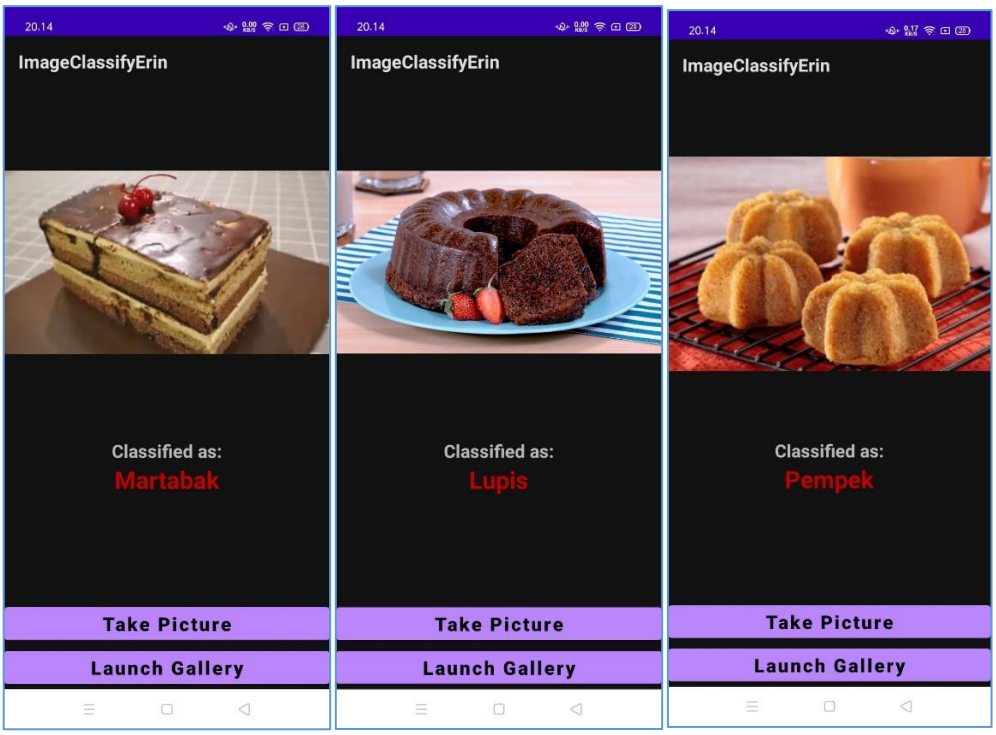


Gambar 4.40 Tampilan Hasil Klasifikasi Gambar Makanan (Kelas Cireng) menggunakan Launch Gallery

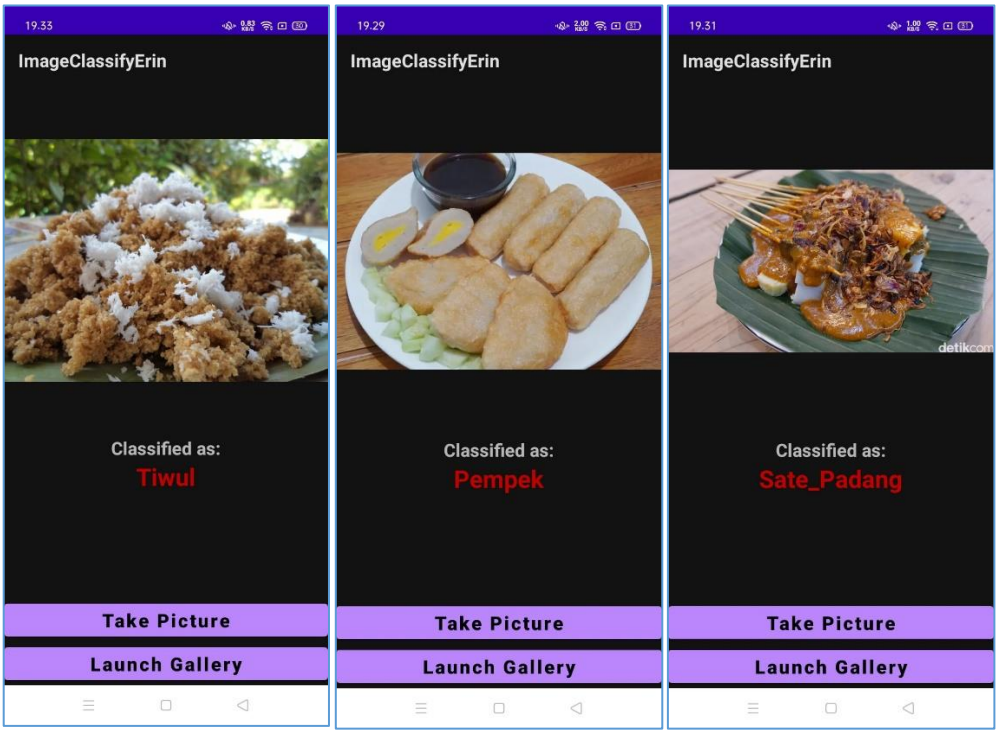
Implementasi pengujian pada aplikasi Android dilakukan secara beberapa kali menggunakan dataset dari *google images* dan pengambilan secara langsung menggunakan

menu take picture. Dataset dari *google images* diuji sebanyak 60 kali, terdiri dari 5 gambar yang tidak ada pada kelas dataset, dan 55 gambar yang berasal dari 18 kelas dataset. Saat pengujian menggunakan dataset dari *google images,* data gambar yang memiliki karakteristik berbeda dari gambar pada kelas dataset mendapatkan hasil klasifikasi yang tidak sesuai dengan nama asli makanan, yakni dengan nama makanan yang dipilih secara acak berdasarkan ciri paling mendekati dari karakteristik yang ada pada kelas dataset. Sedangkan saat data gambar yang dimasukkan tidak terdapat pada kelas dataset, hasil klasifikasi yang didapatkan juga tidak sesuai dengan nama asli dari nama makanan yang diklasifikasikan. Hasil prediksi yang tidak sesuai terjadi karena model yang digunakan pada aplikasi Android merupakan model yang berasal dari kamera *smartphone,* yakni EfficientNetV2_S_21k. Dataset yang berasal dari kamera *smartphone* pada Model EfficientNetV2_S_21k hanya berfokus pada sudut pandang pengambilan gambar sehingga memiliki sedikit variasi objek.

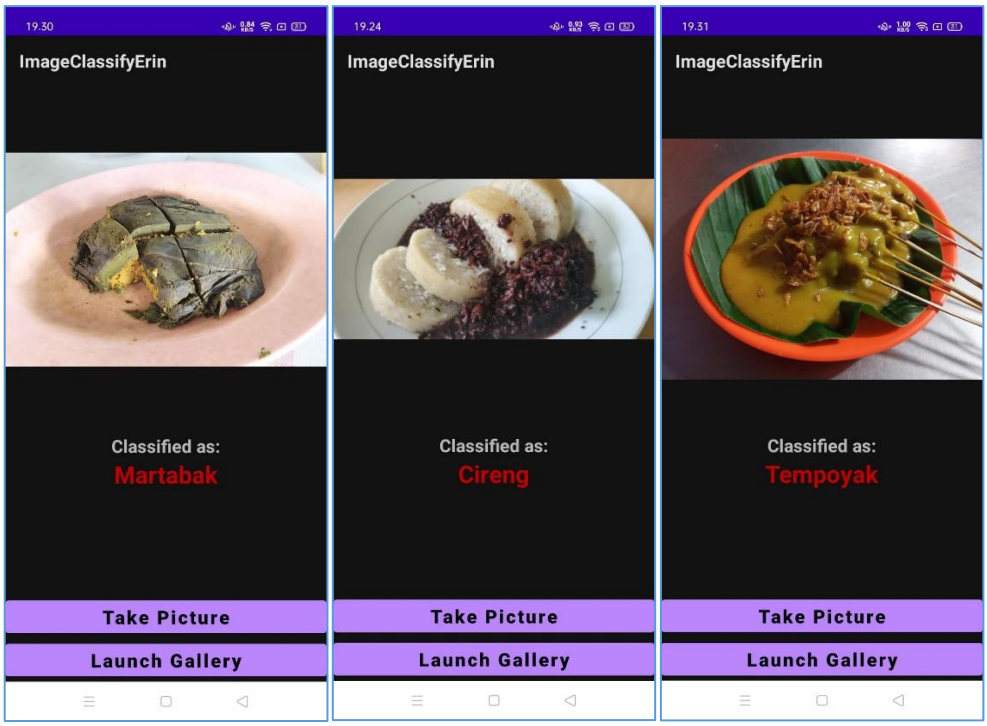
Gambar 4.41 merupakan contoh hasil pengujian menggunakan kelas selain yang ada pada dataset. Data gambar yang berasal dari *google images,* kemudian diuji menggunakan menu *launch gallery.* Gambar 4.41 mendapatkan hasil klasifikasi dengan nama “martabak”, “lupis”, dan “pempek” yang diambil secara acak dari nama pada dataset dengan karakteristik yang paling mendekati dari karakteristik yang ada pada dataset. Gambar 4.42 menggunakan dataset yang diambil dari *google images* dan mendapatkan hasil klasifikasi dengan nama “tiwul”, “pempek”, dan “sate padang” yang sesuai dengan nama aslinya, dikarenakan gambar memiliki karakteristik yang hampir sama dengan nama kelas yang ada pada dataset. Gambar 4.43 merupakan contoh hasil klasifikasi yang menggunakan kelas yang ada pada dataset tetapi mengalami salah prediksi. Gambar 4.43 mengklasifikasikan kelas “pendap” menjadi “martabak”, kelas “lemang tapai” menjadi “cireng”, dan kelas “sate padang” menjadi “tempoyak”. Hasil klasifikasi yang mengalami salah prediksi dikarenakan gambar memiliki karakteristik yang sulit dikenali oleh model, meskipun menggunakan gambar yang ada pada kelas dataset. Hal ini dapat terjadi dikarenakan model yang digunakan memiliki variasi dataset yang masih terbatas. Berdasarkan pengujian yang telah dilakukan sebanyak 60 kali, hasil yang didapatkan yakni 16 gambar yang ada pada kelas dataset mengalami salah prediksi, 5 gambar yang tidak ada pada kelas dataset mengalami salah prediksi, dan 39 gambar yang ada pada kelas dataset berhasil diprediksi dengan benar.



Gambar 4.41 Tampilan Hasil Klasifikasi Gambar Makanan selain Kelas Dataset menggunakan Launch Gallery

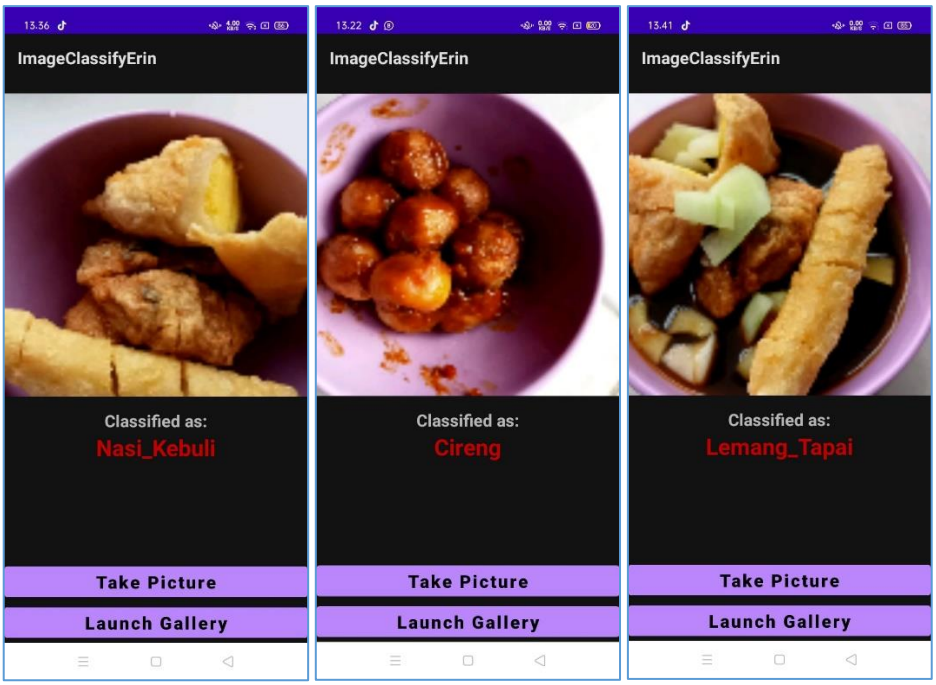


Gambar 4.42 Tampilan Hasil Klasifikasi Gambar Makanan dari Kelas Dataset menggunakan Launch Gallery

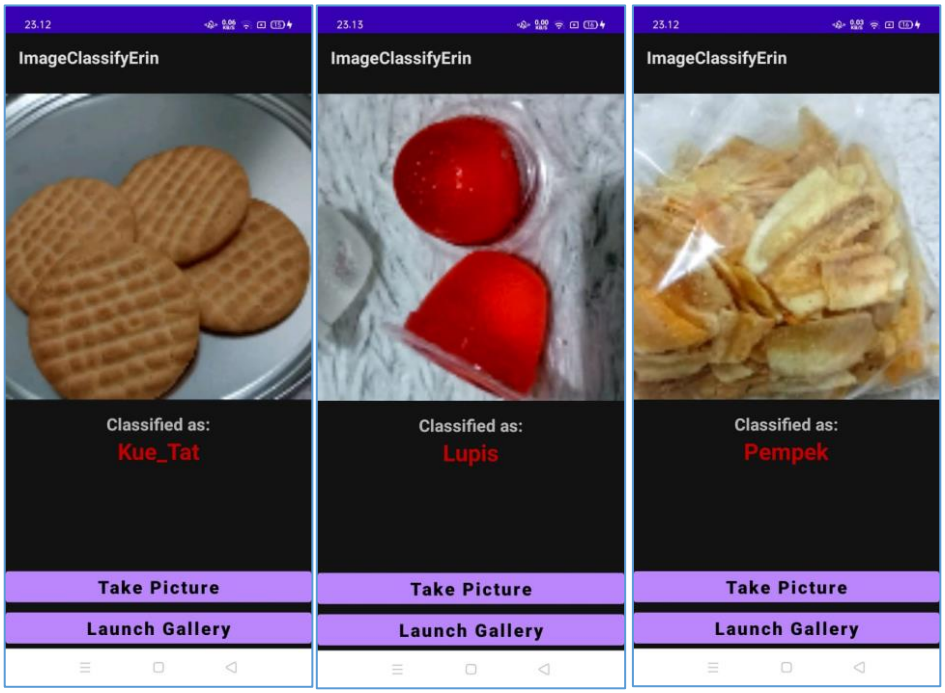


Gambar 4.43 Tampilan Hasil Klasifikasi Gambar Makanan dari Kelas Dataset menggunakan Launch Gallery yang mengalami “Salah Prediksi”

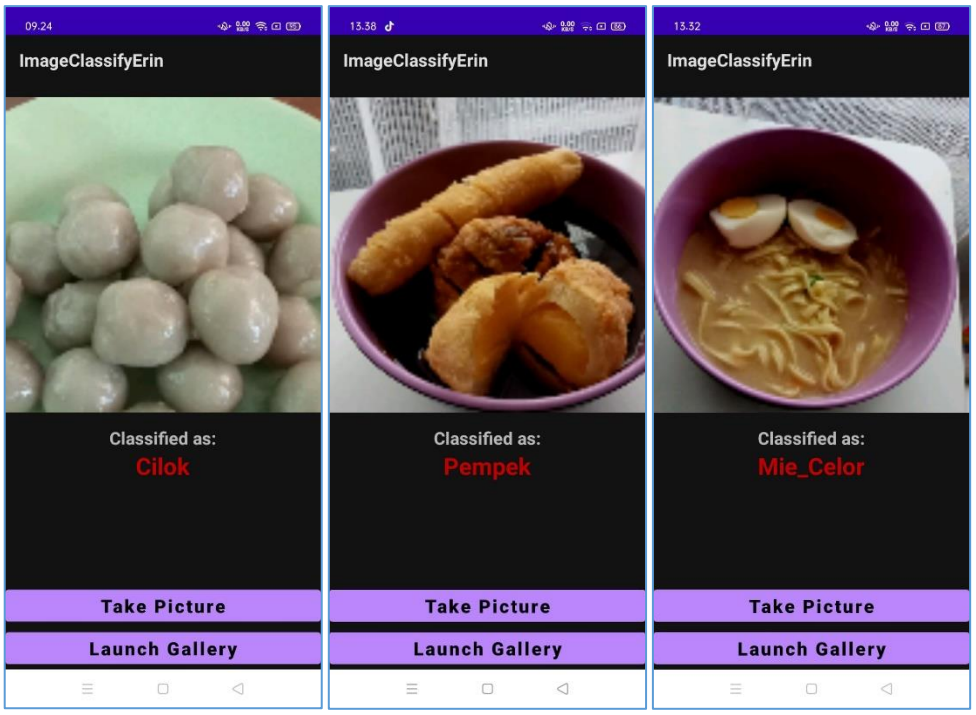
Pengujian yang berasal dari dataset kamera *smartphone* menggunakan menu *take picture* dilakukan sebanyak 60 kali. Pengujian menggunakan 5 gambar yang tidak ada pada kelas dataset, dan 55 gambar yang berasal dari 18 kelas dataset. Gambar 4.44 merupakan hasil pengujian yang menggunakan menu *take picture*. Gambar 4.44 mendapatkan hasil klasifikasi yang tidak sesuai dengan nama asli makanan yang diprediksi, dikarenakan data gambar yang dimasukkan memiliki sudut pengambilan atau karakteristik yang berbeda dari gambar pada kelas dataset aslinya. Gambar 4.45 merupakan gambar makanan yang tidak ada pada kelas dataset dan mendapatkan hasil klasifikasi yang tidak sesuai atau salah prediksi. Gambar 4.46 merupakan gambar makanan yang ada pada kelas dataset dengan hasil klasifikasi yang sesuai. Berdasarkan pengujian yang telah dilakukan sebanyak 60 kali mendapatkan hasil yakni, 8 gambar yang ada pada kelas dataset mengalami salah prediksi, 5 gambar yang tidak ada pada kelas dataset mengalami salah prediksi, dan 47 gambar yang ada pada kelas dataset berhasil diprediksi dengan benar.



Gambar 4.44 Tampilan Hasil Klasifikasi Gambar Makanan dari Kelas Dataset menggunakan Take Picture yang mengalami “Salah Prediksi”



Gambar 4.45 Tampilan Hasil Klasifikasi Gambar Makanan selain Kelas Dataset menggunakan Take Picture yang mengalami “Salah Prediksi”



Gambar 4.46 Tampilan Hasil Klasifikasi Gambar Makanan dari Kelas Dataset menggunakan Take Picture dengan Hasil Prediksi yang Benar

BAB 5
Kesimpulan dan Saran

5.1 Kesimpulan

Setelah dibangun model menggunakan EfficientNetV2 yakni EfficientNetV2_S_21k, EfficientNetV2_M_21k, dan EfficientNetV2_L_21k untuk mengklasifikasikan gambar dengan dataset makanan tradisional Indonesia yang dibagi menjadi 18 kelas dengan jumlah 1800 gambar dari dua kategori sumber data, yang terdiri dari 1620 data latih, 72 data validasi, 108 data uji untuk masing-masing kategori. Maka didapatkan kesimpulan sebagai berikut:

a. Penelitian ini menggunakan pengaturan parameter yakni *learning rate* 0.005, *dropout* 0.5, dan epoch 20.
b. Telah didapatkan nilai akurasi pelatihan terbaik sebesar 99,8% dari model EfficientNetV2_S_21k (sumber data kamera *smartphone*), dan nilai akurasi validasi sebesar 100%, sedangkan untuk nilai akurasi pengujian terbaik sebesar 100% dari model EfficientNetV2_S_21k dan EfficientNetV2_M_21k (sumber data kamera *smartphone*).
c. Model EfficientNetV2 pada penelitian ini mampu mengklasifikasikan gambar makanan tradisional Indonesia dengan baik berdasarkan hasil prediksi dan real dari nama dataset gambar makanan.
d. Model mampu mendapatkan hasil evaluasi yang baik pada matriks evaluasi berdasarkan nilai akurasi, *precision, recall,* dan *F1score.*
e. Model dengan nilai akurasi terbaik berhasil diuji dan diimplementasikan pada perangkat *smartphone* Android untuk klasifikasi gambar makanan tradisional Indonesia.

5.2 Saran

Penelitian klasifikasi gambar makanan tradisional Indonesia ini menggunakan dataset yang terbilang sedikit jika dibandingkan dengan beragam serta banyaknya jenis makanan tradisional Indonesia. Penelitian selanjutnya dapat menggunakan dataset gambar makanan yang lebih beragam dan banyak lagi, agar dapat meningkatkan nilai akurasi pada pengujian model serta memudahkan dalam mengenali jenis nama makanan tradisional Indonesia. Dalam implementasi model juga masih terbatas pada nama makanan saja, sehingga

kedepannya dapat dikembangkan dengan menambahkan bahan-bahan yang terkandung atau informasi lebih detail mengenai makanan yang diklasifikasikan.

Daftar Pustaka

Abdulfattah, M. E., Novamizanti, L., & Rizal, S. (2021). Super Resolution pada Citra Udara menggunakan Convolutional Neural Network. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, *9*(1), 71. https://doi.org/10.26760/elkomika.v9i1.71

Akter, F., Khatun, T., & Uddin, M. S. (2018). Recognition and Classification of Fast Food Images. In *Article in Global Journal of Computer Science and Technology* (Vol. 5). https://www.researchgate.net/publication/350845216

Anam, D. A., Novamizanti, L., & Rizal, S. (2021). *Classifying Retinal Pathology Using OCT Retinal Imaging With Convolutional Neural Network*. https://www.kaggle.com/paultimothymooney/kermany2018

Baequny, A., Sri, A., Elsye, H., Jurusan, R., Poltekkes, K., Semarang, K., & Tirto Agung, J. (2015). Pengaruh Pola Makan Tinggi Kalori terhadap Peningkatan Kadar Gula Darah pada Penderita Diabetes Mellitus Tipe 2. In *Jurnal Riset Kesehatan* (Vol. 4, Issue 1).

Darojat, M. D., Sari, Y. A., & Wihandika, R. C. (2021). *Convolutional Neural Network untuk Klasifikasi Citra Makanan Khas Indonesia* (Vol. 5, Issue 11). http://j-ptiik.ub.ac.id

Fan, S., Xu, R., & Yan, Z. (2021). *A Medical Pre-Diagnosis System for Histopathological Image of Breast Cancer*. https://doi.org/10.1109/CISP-BMEI53629.2021.9624252

Karthik, R., Vaichole, T. S., Kulkarni, S. K., Yadav, O., & Khan, F. (2022). Eff2Net: An Efficient Channel Attention-based Convolutional Neural Network for Skin Disease Classification. *Biomedical Signal Processing and Control*, *73*. https://doi.org/10.1016/j.bspc.2021.103406

Kolisnik, B., Hogan, I., & Zulkernine, F. (2021). Condition-CNN: A Hierarchical Multilabel Fashion Image Classification Model. *Expert Systems with Applications*, *182*. https://doi.org/10.1016/j.eswa.2021.115195

Kurnia, D. A., Setiawan, A., Amalia, D. R., Arifin, R. W., & Setiyadi, D. (2021). Image Processing Identification for Indonesian Cake Cuisine using CNN Classification Technique. *Journal of Physics: Conference Series*, *1783*(1). https://doi.org/10.1088/1742-6596/1783/1/012047

Mahaputri, C., Kristian, Y., & Setyati, E. (2022). *Pengenalan Makanan Tradisional Indonesia Beserta Bahan-bahannya dengan Memanfaatkan DCNN Transfer Learning*. https://doi.org/10.52985/insyst.v4i2.252

Medus, L. D., Saban, M., Francés-Víllora, J. V., Bataller-Mompeán, M., & Rosado-Muñoz, A. (2021). Hyperspectral Image Classification using CNN: Application to Industrial Food Packaging. *Food Control*, *125*. https://doi.org/10.1016/j.foodcont.2021.107962

Mezgec, S., & Seljak, B. K. (2017). Nutrinet: A Deep Learning Food and Drink Image Recognition System for Dietary Assessment. *Nutrients*, *9*(7). https://doi.org/10.3390/nu9070657

Padilla, D., Yumang, A., Diaz, A. L., & Inlong, G. (2019). *Differentiating Atopic Dermatitis and Psoriasis Chronic Plaque using Convolutional Neural Network MobilNet Architecture*.

Pan, L., Pouyanfar, S., Chen, H., Qin, J., & Chen, S. C. (2017). DeepFood: Automatic Multi-Class Classification of Food Ingredients Using Deep Learning. *Proceedings - 2017 IEEE 3rd International Conference on Collaboration and Internet Computing, CIC 2017*, *2017-January*, 181–189. https://doi.org/10.1109/CIC.2017.00033

Park, S. J., Palvanov, A., Lee, C. H., Jeong, N., Cho, Y. I., & Lee, H. J. (2019). The Development of Food Image Detection and Recognition Model of Korean Food for Mobile Dietary Management. *Nutrition Research and Practice*, *13*(6), 521–528. https://doi.org/10.4162/nrp.2019.13.6.521

Ramdani, A., Virgono, A., & Setianingsih, C. (2020). *Food Detection with Image Processing using Convolutional Neural Network (CNN) Method*.

Rohim, A., Sari, Y. A., & Tibyani. (2019). *Convolution Neural Network (CNN) untuk Pengklasifikasian Citra Makanan Tradisional* (Vol. 3, Issue 7). http://j-ptiik.ub.ac.id

Srivastava, N., Hinton, G., Krizhevsky, A., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. In *Journal of Machine Learning Research* (Vol. 15).

Sunil, C. K., Jaidhar, C. D., & Patil, N. (2022). Cardamom Plant Disease Detection Approach Using EfficientNetV2. *IEEE Access*, *10*, 789–804. https://doi.org/10.1109/ACCESS.2021.3138920

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. http://arxiv.org/abs/1905.11946

Tan, M., & Le, Q. V. (2021). *EfficientNetV2: Smaller Models and Faster Training*. http://arxiv.org/abs/2104.00298

Udayana, I. P. A. E., Sudarma, M., & Nugraha, P. G. S. C. (2020). Implementation of Convolutional Neural Networks to Recognize Images of Common Indonesian Food.

IOP Conference Series: Materials Science and Engineering, *846*(1). https://doi.org/10.1088/1757-899X/846/1/012023

Ye, Y., Zhou, H., Yu, H., Hu, H., Zhang, G., Hu, J., & He, T. (2022). An Improved EfficientNetV2 Model Based on Visual Attention Mechanism: Application to Identification of Cassava Disease. *Computational Intelligence and Neuroscience*, *2022*. https://doi.org/10.1155/2022/1569911

Ying, L., Nan, Z. Q., Ping, W. F., Kiang, C. T., Pang, L. K., Chang, Z. H., Lu, C., Jun, L. G., & Nam, L. (2021). Adaptive Weights Learning in CNN Feature Fusion for Crime Scene Investigation Image Classification. *Connection Science*, *33*(3), 719–734. https://doi.org/10.1080/09540091.2021.1875987

Yulianti, N. S., Seminar, K. B., Hermanianto, J., & Wahjuni, S. (2021). *Identifikasi Kemurnian Daging berbasis Analisis Citra*. *8*(4), 643–650. https://doi.org/10.25126/jtiik.202183307