IMPLEMENTASI MODEL *PRE-TRAINED* PADA CNN BERDASARKAN MULTI INPUT PARAMETER UNTUK IDENTIFIKASI JENIS KOPI BERBASIS GUI

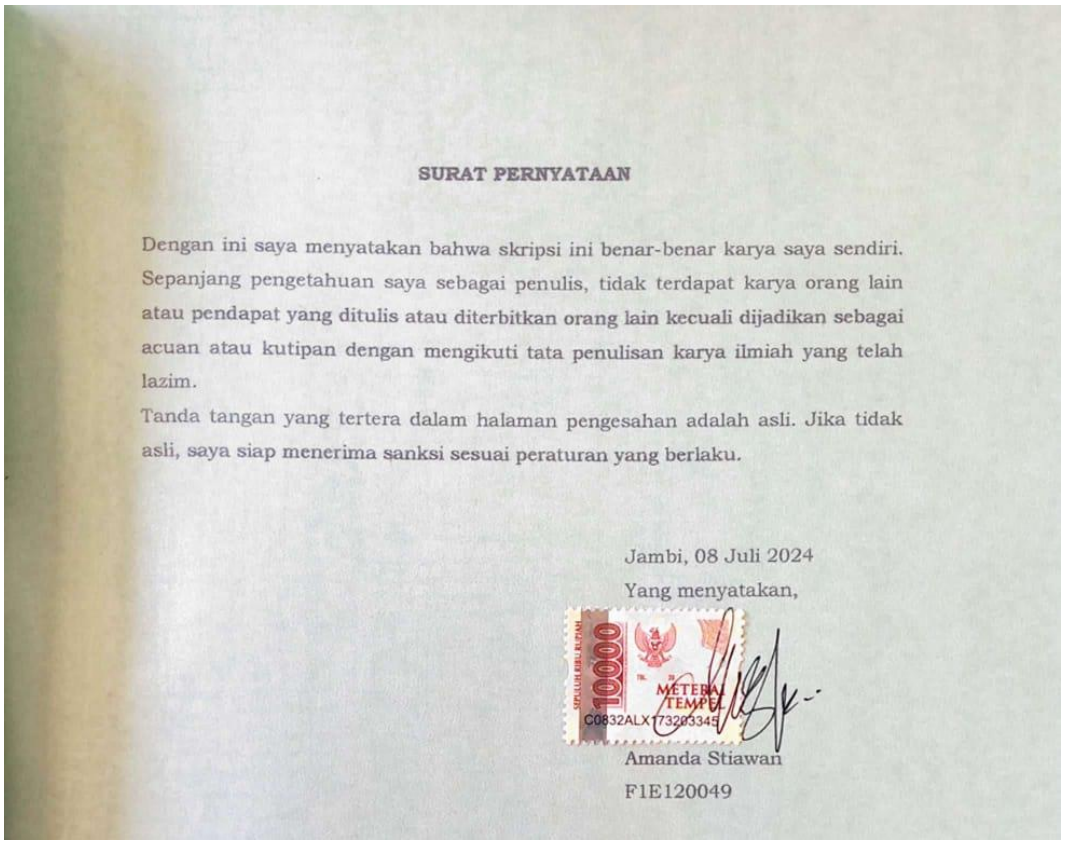
S K R I P S I

AMANDA STIAWAN
F1E120049

PROGRAM STUDI SISTEM INFORMASI
JURUSAN TEKNIK ELEKTRO DAN INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS JAMBI
2024

SURAT PERNYATAAN



SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa skripsi ini benar-benar karya saya sendiri. Sepanjang pengetahuan saya sebagai penulis, tidak terdapat karya orang lain atau pendapat yang ditulis atau diterbitkan orang lain kecuali dijadikan sebagai acuan atau kutipan dengan mengikuti tata penulisan karya ilmiah yang telah lazim.

Tanda tangan yang tertera dalam halaman pengesahan adalah asli. Jika tidak asli, saya siap menerima sanksi sesuai peraturan yang berlaku.

Jambi, 08 Juli 2024

Yang menyatakan,

Amanda Stiawan

F1E120049

IMPLEMENTASI MODEL *PRE-TRAINED* PADA CNN BERDASARKAN MULTI INPUT PARAMETER UNTUK IDENTIFIKASI JENIS KOPI BERBASIS GUI

S K R I P S I

Diajukan sebagai salah satu syarat dalam melakukan penelitian dalam rangka penulisan Skripsi pada program studi Sistem Informasi

AMANDA STIAWAN

F1E120049

PROGRAM STUDI SISTEM INFORMASI

JURUSAN TEKNIK ELEKTRO DAN INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS JAMBI

2024

PENGESAHAN

PENGESAHAN

Skripsi dengan judul "**IMPLEMENTASI MODEL PRE-TRAINED PADA CNN BERDASARKAN MULTI INPUT PARAMETER UNTUK IDENTIFIKASI JENIS KOPI BERBASIS GUI**" yang disusun oleh **AMANDA STIAWAN, NIM: F1E120049** telah dipertahankan di depan penguji pada tanggal 08 Juli 2024 dan dinyatakan Lulus.

Susunan Tim Penguji:

Ketua	:	Pradita Eko Prasetyo Utomo, S.Pd, M.Cs.
Sekretaris	:	Benedika Ferdian Hutabarat, S.Komp., M.Kom.
Anggota	:	1. Ulfa Khaira, S.Komp., M.Kom.
		2. Zainil Abidin, S.T., M.Eng.
		3. Mutia Fadhila Putri, M.Kom.

Disetujui :

Pembimbing Utama,

Pradita Eko Prasetyo Utomo, S.Pd., M.Cs.
NIP. 198710282019031010

Pembimbing Pendamping,

Benedika Ferdian Hutabarat, S.Komp., M.Kom.
NIP. 198702022019031007

Diketahui :

Dekan
Fakultas Sains dan Teknologi,

Drs. Jefri Marzal, M.Sc., D.I.T.
NIP. 196805021993031004

Ketua Jurusan
Teknik Elektro dan Informatika,

Edi Saputra, S.T., M.Sc.
NIP. 198501082015041003

i

RINGKASAN

Indonesia, sebagai negara agraris, memiliki potensi besar dalam sektor pertanian, salah satunya adalah produksi kopi yang menjadi komoditas ekspor penting. Kopi, dengan berbagai varietas seperti robusta, arabika, dan liberika, merupakan tanaman yang sangat diperdagangkan secara global. Provinsi Jambi di Indonesia memiliki beragam jenis kopi yang dibudidayakan, namun identifikasi varietas kopi seringkali mengalami kendala karena ketidakakuratan pengenalan secara kasat mata. Penelitian ini bertujuan untuk mengembangkan sistem identifikasi jenis kopi berbasis teknologi *Artificial Intelligence* (AI) dengan menggunakan metode *Convolutional Neural Network* (CNN) dan model *pretrained* seperti *NasNetMobile*, *ResNet50*, dan *EfficientNetV2B0*.

Hasil penelitian dan pengujian menunjukkan bahwa model *NasNetMobile* memberikan performa terbaik dengan akurasi pelatihan sebesar 97.72%, akurasi validasi 97.40%, *precision* 96%, *recall* 96%, *f1-score* 96%, dan akurasi pengujian 97% setelah 30 *epoch*. Model ini berhasil diimplementasikan pada website identifikasi jenis kopi dengan tingkat akurasi yang memadai, dimana 23 dari 30 gambar kopi berhasil diidentifikasi dengan benar.

RIWAYAT HIDUP

Amanda Stiawan, lahir di Kota Jambi pada tanggal 14 Juni 2003. Penulis merupakan anak kedua dari 3 bersaudara dengan Ayah bernama Nofri Irawan dan Ibu Siti Khatijah. Penulis telah menyelesaikan pendidikan Sekolah Dasar di SD Negeri 139/IV Kota Jambi, pendidikan Sekolah Menengah Pertama di MTss Guppi Kota Jambi, dan Sekolah Menengah Atas di MAN 2 Kota Jambi. Penulis diterima sebagai mahasiswa Universitas Jambi Program Strata Satu (S1) di Program Studi Sistem Informasi, Jurusan Teknik Elektro dan Informatika, Fakultas Sains dan Teknologi, Universitas Jambi melalui jalur SBMPTN pada tahun 2020. Selama menempuh pendidikan jenjang Strata satu, penulis aktif terlibat dalam bidang akademik maupun non-akademik. Pada tahun 2022 - 2023 penulis mendapat kesempatan menjadi anggota Divisi Hubungan Masyarakat Himpunan Mahasiswa Sistem Informasi Universitas Jambi. Selain itu, pada tahun 2023 penulis mengikuti kegiatan Studi Independen Bersertifikat melalui program Merdeka Belajar Kampus Merdeka (MBKM) di PT Dwi Inti Putra (MyEduSolve) dengan fokus pelatihan *Data Analyst.* Pada tahun 2023 penulis mengikuti program magang selama 3 bulan di Badan Pusat Statistik Provinsi Jambi dan ditempatkan di divisi Integrasi Pengolahan dan Diseminasi Statistik (IPDS). Pada semester 7, penulis mengikuti kegiatan Kampus Merdeka yang diselenggarakan oleh kampus, dengan fokus percepatan tugas akhir mahasiswa. Penelitian yang dipilih adalah "Implementasi Model *Pre-Trained* pada CNN Berdasarkan Multi Input Parameter untuk Identifikasi Jenis Kopi Berbasis GUI"

PRAKATA

Bismillahirrahmannirrahiim, Alhamdulillahirabbil'Alamiin, segala puji dan syukur dihaturkan kepada Allah SWT, tidak lupa sholawat dan salam juga dihadiahkan untuk Nabi Muhammad SAW. Karena ridho, rahmat serta karunia-Nya, penulis dapat menyelesaikan skripsi dengan judul "Implementasi Model *Pre-Trained* pada CNN Berdasarkan Multi Input Parameter untuk Identifikasi Jenis Kopi Berbasis GUI" ini dapat terselesaikan atas bimbingan, motivasi, dukungan dan do'a dari berbagai pihak, maka sebagai ungkapan hormat dan penghargaan penulis mengucapkan terima kasih kepada:

1. Bapak Prof. Dr. Helmi., S.H., M.H selaku Rektor Universitas Jambi
2. Drs. Jefri Marzal, M.Sc., D.I.T., selaku Dekan Fakultas Sains dan Teknologi Universitas Jambi.
3. Reni Aryani, S.Kom., M.S.I., selaku Koordinator Program Studi Sistem Informasi Universitas Jambi.
4. Pradita Eko Prasetyo Utomo, S.Pd., M.Cs. dan Benedika Ferdian Hutabarat, S.Komp., M.Kom., selaku Dosen Pembimbing yang telah memberikan arahan dan bimbingan secara langsung kepada penulis dalam proses penyelesaian skripsi.
5. Ulfa Khaira, S.Komp., M.Kom., Mutia Fadhila Putri, M.Kom., Zainil Abidin, S.T., M.Eng., selaku dosen penguji skripsi, yang telah memberikan saran dan masukan untuk penyelesaian skripsi.
6. Dedy Setiawan, S.Kom., M.IT. selaku Dosen Pembimbing Akademik yang telah memberikan pengarahan selama masa studi.
7. Cinta pertama dan panutanku, Ayahanda Nofri irawan dan pintu surga Ibunda Siti Khatijah yang selalu menjadi penyemangat, memberi dukungan moril dan materil, memberi cinta, kasih, serta doa yang tiada henti dan segala bentuk yang telah diberikan sehingga saya mampu menyelesaikan skripsi ini. Terima kasih telah mengajarkan saya untuk tidak pernah menyerah dalam keadaaan terburuk sekalipun. Semoga Allah SWT memberikan keberkahan hidup di dunia serta tempat terbaik di akhirat kelak, karena telah menjadi figur orang tua terbaik bagi penulis.
8. Kepada cinta kasih kedua saudara kandung saya, Putry Sugesti dan Tio Nofrian. Terimakasih atas segala doa, usaha dan support yang telah diberikan kepada saya.
9. Teman-teman tersayang saya Aura, Gusela, Sakinah, Tatak, Widya, Dini, Fintria, Valezza, Jihan, Rifqiyah, Ananda, Puja, dan Putri yang selalu siap membantu saya dalam kesulitan serta sabar mendengarkan apapun cerita saya. Terima kasih sudah menjadi sahabat baik bahkan seperti saudara dan

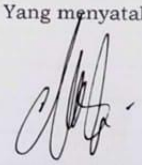
terima kasih selalu memberikan support, motivasi dan meyakinkan saya dalam setiap langkah serta tidak pernah meninggalkan penulis sendiria.

10. Seluruh pihak yang penulis libatkan selama proses penyusunan Skripsi dan tidak dapat penulis sebutkan satu persatu, terima kasih atas segala waktu, motivasi, dukungan dan do'a yang diberikan kepada penulis.
11. Dan yang terakhir, kepada perempuan sederhana namun terkadang sangat sulit dimengerti isi kepalanya, sang penulis sebuah karya tulis ini diri saya sendiri, Amanda Stiawan. Terima kasih sudah bertahan sejauh ini melewati banyaknya rintangan hidup yang tidak tertebak adanya. Terima kasih tetap memilih hidup dan merayakan dirimu sendiri sampai di titik ini, walaupun seringkali merasa putus asa atas apa yang diusahakan dan belum berhasil namun terima kasih tetap menjadi manusia yang selalu mau berusaha dan tidak lelah mencoba. Berbahagialah selalu dimanapun berada, Manda. Rayakan selalu kehadiranmu di dunia semua hal yang membuatmu hidup. Pastikan jiwamu selalu menjadi bagian dari hal baik di alam semesta, semoga selalu dikelilingi hal- hal baik.

Semoga skripsi ini dapat bermanfaat, terutama dalam bidang deep learning. Penulis menyadari bahwa masih banyak ruang untuk perbaikan, dan setiap masukan serta saran yang diberikan akan sangat berarti untuk meningkat kualitas skripsi ini. Semoga skripsi ini dapat memberikan dampak positif kepada pembaca. Terima kasih atas segala perhatiannya.

Jambi, 08 Juli 2024

Yang menyatakan,



Amanda Stiawan

F1E120049

DAFTAR ISI

PENGESAHAN i
RINGKASAN ii
RIWAYAT HIDUP..... iii
PRAKATA.....iv
DAFTAR ISI vi
DAFTAR TABEL. viii
DAFTAR GAMBAR.....ix
DAFTAR LAMPIRAN xi
I. PENDAHULUAN..... 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 5
1.3 Batasan Masalah..... 5
1.4 Tujuan Penelitian 5
1.5 Manfaat Penelitian..... 5
II. TINJAUAN PUSTAKA..... 7
2.1 Kopi 7
2.2 Pengolahan Citra Digital..... 8
2.3 *Artificial Intelligence* 9
2.4 *Machine Learning*..... 9
2.5 *Deep Learning*..... 10
2.6 *Convolutional Neural Network*..... 11
2.7 Multi-input CNN..... 18
2.8 *Transfer Learning*..... 18
2.9 *Pre-trained* Model CNN..... 19
2.10 Pengujian Model..... 23
2.11 Perancangan Graphical User Interface 24
2.12 Penelitian Terdahulu 25
III. METODOLOGI PENELITIAN 30
3.1 Tempat dan Waktu Penelitian..... 30
3.2 Alat dan Bahan Penelitian 30
3.3 Kerangka Metode Penelitian..... 30
3.4 Jadwal penelitian 38
IV. HASIL DAN PEMBAHASAN..... 39
4.1 Pengumpulan Dataset 39
4.2 *Pre-processing* Data 41
4.3 Pembentukan Arsitektur Model CNN..... 48
4.4 Evaluasi dan Perbandingan Model..... 59

4.5 Implementasi dan Deteksi dengan *Website* 71
V. KESIMPULAN DAN SARAN 85
DAFTAR PUSTAKA 87
LAMPIRAN 92

DAFTAR TABEL.

Tabel Halaman

Tabel 1. Penelitian Terdahulu 25
Tabel 2. Confusion Matriks Model..... 35
Tabel 3. Jadwal penelitian 38
Tabel 4. Jumlah Citra..... 41
Tabel 5. Jumlah Data Setelah Oversampling..... 45
Tabel 6. Jumlah sebaran pembagian dataset 48
Tabel 7. Deskripsi Model Summary (NasNetMobile)..... 52
Tabel 8. Deskripsi Model Summary (Resnet50) 54
Tabel 9. Deskripsi Model Summary (EfficientNetV2B0)..... 57
Tabel 10. *Hyperparameter* Model 58
Tabel 11. Hasil akhir pelatihan ketiga model berdasarkan *epoch* 63
Tabel 12. Perbandingan Nilai True Positive 67
Tabel 13. Perbandingan Hasil Testing dengan Data Baru..... 69
Tabel 14. Perbandingan Kinerja Tiga Model Terbaik Masing-Masing Arsitektur70
Tabel 15. Hasil Testing 74
Tabel 16. Confusion Matrix dari Data Uji Baru 82

DAFTAR GAMBAR

Gambar Halaman

Gambar 1. Jumlah produksi Kopi di Indonesia 1
Gambar 2. Metode *Convolutional Neural Network*..... 12
Gambar 3. Contoh Operasi Konvolusi 13
Gambar 4. Operasi Max Pooling 14
Gambar 5. Sigmoid 15
Gambar 6. Tanh..... 16
Gambar 7. ReLU 16
Gambar 8. Arsitektur untuk *ImageNet* dan CIFAR-10..... 20
Gambar 9. Arsitektur *NasNetMobile* 20
Gambar 10. Arsitektur *Resnet50*..... 21
Gambar 11. Arsitektur *EfficientNetV2B0* 23
Gambar 12. *Confusion Matrix*..... 23
Gambar 13. Flowchart Penelitian 31
Gambar 14. Mockup Halaman Utama Website 36
Gambar 15. Mockup Halaman Prediksi Input Citra..... 36
Gambar 16. Mockup Halaman Hasil Prediksi Input Citra 36
Gambar 17. Mockup Halaman Informasi Jenis Kopi Liberika 37
Gambar 18. Mockup Halaman Informasi Jenis Kopi Arabika..... 37
Gambar 19. Mockup Halaman Informasi Jenis Kopi Robusta 38
Gambar 20. Data Citra Kopi Arabika..... 40
Gambar 21. Data Citra Kopi Liberika 40
Gambar 22. Data Citra Kopi Robusta 40
Gambar 23. Kode Program Mengakses dataset 41
Gambar 24. Kode program menghapus latar belakang gambar 42
Gambar 25. Kode program *resizing* gambar..... 42
Gambar 26. Kode program *denoising* 43
Gambar 27. Kode program *sharpening* 43
Gambar 28. Kode program fungsi *ImageDataGenerator* 44
Gambar 29. Contoh data hasil augmentasi 44
Gambar 30. Kode program proses *Oversampling* 46
Gambar 31. Kode program pembagian dataset 47
Gambar 32. Kode program untuk Pembentukan Model 49
Gambar 33. Representasi visual arsitektur model *NasNetMobile*..... 51
Gambar 34. Model *Summary* (*NasNetMobile*) 51
Gambar 35. Representasi visual arsitektur model *Resnet50* 53
Gambar 36. *Model Summary (Resnet50)* 54
Gambar 37. Representasi visual arsitektur model *EfficientNetV2B0*..... 56
Gambar 38. Model *Summary* (*EfficientNetV2B0*) 56
Gambar 39. Kode program untuk konfigurasi *hyperparameter* 59
Gambar 40. Grafik hasill pelatihan NasNetMobile Epoch 10..... 60
Gambar 41. Grafik hasill pelatihan NasNetMobile Epoch 20..... 60
Gambar 42. Grafik hasill pelatihan *NasNetMobile Epoch* 30 60
Gambar 43. Grafik hasill pelatihan Resnet50 Epoch 10 61
Gambar 44. Grafik hasill pelatihan Resnet50 Epoch 20 61
Gambar 45. Grafik hasill pelatihan Resnet50 Epoch 30 61
Gambar 46. Grafik hasill pelatihan EfficientNetV2B0 Epoch 10 62
Gambar 47. Grafik hasill pelatihan EfficientNetV2B0 Epoch 20 62
Gambar 48. Grafik hasill pelatihan EfficientNetV2B0 Epoch 30..... 62
Gambar 49. *Confusion matrix EfficientNetV2-B0 (A) Epoch* 10, *(B) Epoch* 20, *(c) Epoch* 30 66

Gambar 50. Classification Report NasNetMobile (A) Epoch 10, (B) Epoch 20, (c) Epoch 30 67
Gambar 51. Classification Report ResNet50 (A) Epoch 10, (B) Epoch 20, (c) Epoch 30 68
Gambar 52. Classification Report EfficientNetV2B0 (A) Epoch 10, (B) Epoch 20, (c) Epoch 30 69
Gambar 53. Halaman utama website 72
Gambar 54. Halaman Prediksi Input Gambar 72
Gambar 55. Halaman Hasil Prediksi 73
Gambar 56. Halaman Informasi Jenis Kopi 74
Gambar 57. Hasil Prediksi Gambar Diluar Kelas 83

DAFTAR LAMPIRAN

Lampiran Halaman

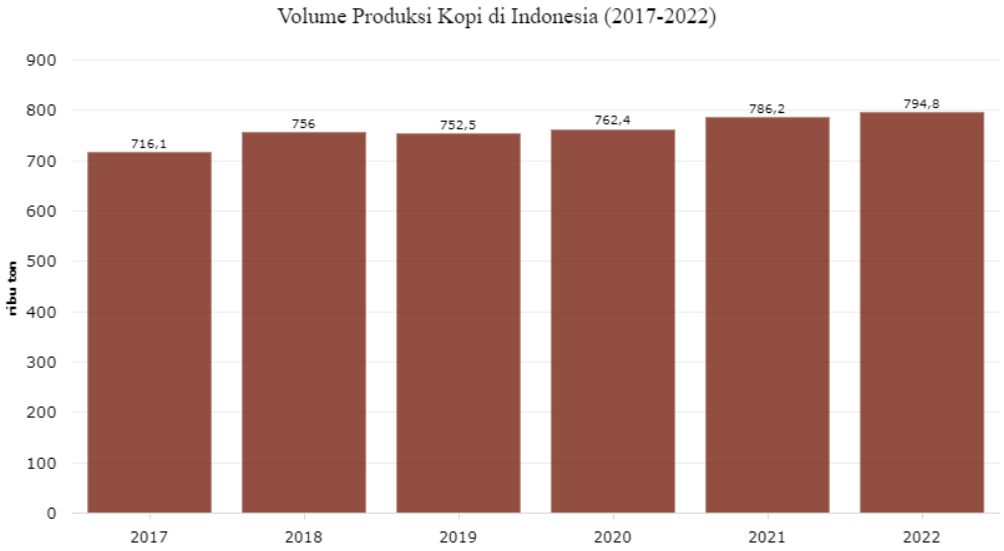
Lampiran 1. Dokumentasi Pengambilan Data Biji, Buah dan Daun Kopi 92
Lampiran 2. Visualisasi Arsitektur Model NasNetMobile 93
Lampiran 3. Visualisasi Arsitektur Model ResNet50..... 93
Lampiran 4. Visualisasi Arsitektur Model EfficientnetV2B0 94
Lampiran 5. Script visualisasi feature maps 94

I. PENDAHULUAN

1.1 Latar Belakang

Indonesia merupakan negara agraris yang sebagian besar lahannya digunakan untuk pertanian atau perkebunan. Hasil pertanian atau panen dapat dijadikan sebagai barang ekspor untuk menghidupkan kembali perekonomian negara. Salah satu tanaman dengan nilai mentah yang tinggi adalah kopi. Kopi merupakan tanaman tropis yang paling banyak diperdagangkan di dunia, dengan sekitar 25 juta rumah tangga petani yang berkontribusi hingga 80% dari produksi dunia (FAO Organizational, 2023). Produksi kopi terutama terpusat di negara-negara berkembang, di mana kopi memberikan sebagian besar keuntungan ekspor dan menjadi sumber utama pendapatan. Produksi kopi global mencapai 174.3 juta karung per 60 kg pada tahun 2022/2023.

Indonesia merupakan negara penghasil kopi terbesar ketiga di dunia setelah Brasil dan Vietnam dengan hasil produksi kopi sebanyak 11.9 juta karung periode tahun 2022/2023 (Foreign Agricultural Service, 2023). Produksi kopi di Indonesia mengalami peningkatan dari tahun sebelumnya yakni sekitar 1.1% dengan produksi kopi mencapai 794.8 ribu ton pada 2022. Pada gambar 1 ditunjukkan jumlah produksi kopi di Indonesia tahun 2017-2022 (Badan Pusat Statistik, 2023).



Gambar 1. Jumlah produksi Kopi di Indonesia

Indonesia memiliki tiga jenis kopi utama yang dibudidayakan di berbagai daerah, yaitu kopi robusta, kopi arabika, dan kopi liberika. Salah satu provinsi di Indonesia yang menjadi produsen kopi dengan tiga varietas tersebut adalah Jambi. Jambi memiliki kopi arabika yang dibudidayakan di Kabupaten Kerinci dan Tebo, Kopi robusta dibudidayakan di Kabupaten Merangin, Bungo, Kerinci

dan Tebo serta kopi liberika dibudidayakan oleh masyarakat Kabupaten Tanjung Jabung Barat (Dinas Perkebunan Provinsi Jambi, 2015).

Sebagai produsen kopi, Provinsi Jambi memiliki potensi yang besar dalam produksi kopi. Beragam varietas kopi yang ada dapat diidentifikasi berdasarkan karakteristik dari kopi tersebut. Masing-masing varietas kopi tersebut memiliki visual yang berbeda-beda misalnya seperti perbedaan warna, bentuk, ataupun tekstur. Sistem identifikasi menjadi penting karena adanya kebutuhan untuk membedakan jenis atau varietas kopi secara akurat. Biasanya, petani ataupun pemilik *coffee shop* dapat mengenali varietas kopi berdasarkan ciri-ciri tertentu seperti biji kopi. Akan tetapi, terkadang para pemilik *coffee shop* dapat melakukan kesalahan dalam mengenali varietas kopi yang dijual ataupun yang dibeli (Sebatubun & Pujiarini, 2018). Identifikasi jenis kopi secara kasat mata dapat tidak akurat atau presisi sehingga diperlukan pengembangan sistem perangkat lunak yang dapat membantu dalam mengidentifikasi jenis-jenis kopi secara visual. Hal tersebut dapat diwujudkan oleh bantuan *Artificial intelligence* (AI) dengan pengolahan citra digital.

Artificial Intelligence (AI) atau kecerdasan buatan adalah era puncak teknologi industri saat ini yang berpotensi meningkatkan produktivitas dan pertumbuhan di berbagai sektor, termasuk pertanian. *Artificial Intelligence* (AI) dapat membantu meningkatkan efisiensi dalam berbagai aspek pertanian, seperti identifikasi jenis-jenis tanaman, irigasi, deteksi tanah dan pemantauan pertumbuhan (Sharma et al., 2022). Dengan memanfaatkan teknologi *Artificiall inteligence* (AI), proses identifikasi jenis-jenis kopi dapat dilakukan secara efisien, biaya rendah dan dengan tingkat akurasi yang dipercaya. Seperti yang dilakukan olehCahyono et al. (2023) yaitu mengklasifikasi jenis biji kopi menggunakan metode *Gray Level Co-occurrence Matrix* (GLCM). Pada penelitian ini bahan utama yang digunakan adalah biji kopi jenis Robusta Tanggul dan Raung, serta Arabika Ijen dan Bali yang telah disangrai. Hasil yang diperoleh dari akurasi klasifikasi jenis biji kopi dengan menggunakan metode GLCM sebesar 99% untuk klasifikasi tipe kernel atau powder, 93% untuk klasifikasi jenis robusta atau arabika, dan 56% untuk klasifikasi daerah asal yaitu (Tanggul, Raung, Ijen, dan Bali) dengan citra data uji sebanyak 80 citra.

Salah satu teknologi yang menerapkan teknik pengolahan citra digital adalah *deep Learning*. *Deep Learning* dan *artificial intelligence* termasuk ke dalam bidang *computer vision* dalam penggalian informasi dari citra. *Deep Learning* menggunakan konsep cara kerja dari jaringan saraf otak manusia dalam menangkap objek gambar (Almryad & Kutucu, 2020). *Deep Learning* merupakan cabang ilmu dari *Machine learning* berbasis jaringan saraf tiruan yang

mengajarkan komputer untuk melakukan suatu tindakan yang dianggap alami oleh manusia. Dalam *Deep Learning*, sebuah komputer belajar mengklasifikasi secara langsung dari gambar, teks atau suara (J. W. G. Putra, 2020).

Salah satu metode *deep Learning* yang paling banyak digunakan dalam melakukan pengolahan citra adalah metode *Convolutional neural network* (CNN). CNN merupakan pengembangan dari *Multi Layer Perceptron* (MLP) dan merupakan salah satu algoritma dari *Deep Learning*. Metode CNN memiliki hasil paling signifikan dalam pengenalan citra, hal tersebut dikarenakan CNN berusaha meniru sistem pengenalan citra pada visual cortex manusia, sehingga memiliki kemampuan mengolah informasi citra (Peryanto et al., 2020). Akan tetapi, pengembangan CNN masih terhitung mahal, mulai dari waktu yang digunakan untuk *training* model dan *hardware* yang dibutuhkan untuk menjalankan proses *training* tersebut. Untuk menghemat proses pengembangan *Convolutional neural network* (CNN), model-model CNN yang sudah mengalami proses *training* disimpan dalam bentuk arsitektur *Pre-trained* model (Pangestu & Bunyamin, 2018).

Pre-trained model biasanya sudah dilatih pada dataset yang besar, sehingga memungkinkan mendapatkan tingkat akurasi yang lebih baik. Seperti penelitian yang dilakukan oleh Fuadi & Suharso (2022) terkait perbandingan kinerja model arsitektur *NasNetMobile* yaitu pada penelitian "Perbandingan Arsitektur Mobilenet Dan *NasNetMobile* Untuk Klasifikasi Penyakit Pada Citra Daun Kentang". Penelitian ini bertujuan untuk membandingkan arsitektur MobileNet dan *NasNetMobile* dalam melakukan deteksi penyakit pada daun tanaman kentang. Kedua arsitektur tersebut digunakan untuk menyelesaikan masalah klasifikasi gambar pada perangkat mobile. Di akhir penelitian ditemukan bahwa pada skema pengujian menggunakan arsitektur *NasNetMobile* dengan perbandingan data latih dan data uji sebesar 90:10 menghasilkan *accuracy* sebesar 90.96%, *precision* sebesar 90.86%, *recall* sebesar 91.11%, dan *f1 score* sebesar 92.93%.

Para peneliti telah mengembangkan solusi *deep Learning* dengan metode CNN untuk pertanian dalam beberapa tahun terakhir dengan mengklasifikasikan ataupun mengidentifikasi jenis produk pertanian. Seperti pada penelitian sebelumnya tentang identifikasi jenis kopi yang dilakukan oleh Murinto, M. (2023), focus pada penelitian ini adalah klasifikasi jenis biji kopi menggunakan *Convolutional neural network* (CNN) dan memanfaatkan model *pre-trained yaitu VGG16 dan MobileNetV2.* Penelitian ini menggunakan 1994 gambar biji kopi arabika, robusta dan liberika sebagai dataset penelitian. Akurasi tertinggi yang dicapai dalam mengklasifikasi jenis kopi dari gambar adalah ketika menggunakan

model MobileNetV2 dengan akurasi 96%. Hasil pengujian menunjukkan bahwa penggunaan transfer learning dapat meningkatkan akurasi klasifikasi.

Selain itu, penelitian terkait yang dilakukan oleh M. T. Hidayat et al. (2024) tentang "Implementation of CNN Trained Model on Coffee Type Detection Android App with Photo Input of Beans, Fruits and Leaves" menggunakan NasNetMobile pada implementasi Android. Dalam penelitian ini, dikembangkan aplikasi menggunakan metode Rapid Application Development (RAD) dengan memanfaatkan framework Flutter dan model Convolutional Neural Network yang telah dilatih. Model pre-train yang digunakan adalah NasNet Mobile dengan dataset 900 foto dan 100 epoch, serta pemanfaatan early-stopping yang berhenti pada epoch ke-55 dengan akurasi 90,67%. Penelitian ini melakukan implementasi model yang telah ada ke dalam aplikasi Android dengan menggunakan framework Flutter. Proses implementasi ini menghasilkan aplikasi yang dapat membantu proses deteksi menggunakan perangkat Android. Hasil implementasi menunjukkan pengujian yang baik dengan skor 0,97.

Penelitian ini mengusulkan penggunaan model pretrained NasNet Mobile, ResNet50, dan EfficientNetV2B0 dalam arsitektur CNN untuk mengidentifikasi jenis atau varietas kopi berdasarkan citra daun, buah, dan biji. Model-model pretrained ini, yang terbukti efektif dalam tugas klasifikasi dan identifikasi citra, memiliki karakteristik khusus yang mendukung efisiensi dan akurasi. NasNet Mobile memiliki arsitektur ringan dan efisien, cocok untuk perangkat mobile. ResNet50 dengan banyak parameter dan lapisan mampu mengekstrak fitur citra yang kompleks, menghasilkan akurasi tinggi. EfficientNetV2B0 menawarkan keseimbangan antara akurasi dan efisiensi, cocok untuk berbagai platform. Penggunaan model pretrained ini dalam CNN dapat meningkatkan akurasi klasifikasi dengan memanfaatkan pengetahuan dari dataset besar, mengurangi waktu dan biaya pelatihan, serta meningkatkan efisiensi model sehingga dapat digunakan pada platform dengan sumber daya terbatas.

Berdasarkan uraian tersebut, penelitian ini mengimplementasikan algoritma CNN untuk mengidentifikasi jenis atau varietas kopi dengan memanfaatkan arsitektur pretrained yang sudah terlatih. CNN dipilih karena kemampuannya dalam mengekstrak fitur hierarkis dari citra, mirip dengan cara manusia mengenali pola. Kontribusi utama penelitian ini adalah penggunaan tiga parameter berbeda citra daun, buah, dan biji kopi untuk mendapatkan pemodelan terbaik. Diharapkan hasil penelitian ini mampu menghasilkan pemodelan dengan akurasi tinggi untuk mengidentifikasi jenis kopi secara akurat. Selanjutnya, model ini akan diimplementasikan dalam bentuk perangkat lunak berbasis antarmuka grafis (GUI) yang memudahkan pengoperasian model

CNN, meningkatkan aksesibilitas, dan membantu petani, pengamat kopi, ahli, serta masyarakat umum dalam membedakan jenis atau varietas kopi.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang penelitian ini, maka rumusan masalah dalam penelitian ini adalah

1. Bagaimana implementasi model arsitektur *Pre-trained* pada CNN berdasarkan multi input parameter untuk identifikasi jenis kopi berbasis GUI?
2. Bagaimana hasil evaluasi terhadap performa model-model *Pre-trained* yang digunakan dalam identifikasi jenis kopi berdasarkan multi input parameter?

1.3 Batasan Masalah

Batasan masalah yang digunakan dalam penelitian ini adalah sebagai berikut.

1. Dataset yang digunakan adalah citra daun, buah, dan biji kopi yang terdiri dari 3 kategori jenis kopi: jenis kopi arabika, kopi robusta, dan kopi liberika.
2. Pengujian model untuk identifikasi jenis atau varietas kopi pada penelitian ini sampai proses identifikasi jenis kopi berbasis GUI.

1.4 Tujuan Penelitian

Adapun tujuan dari penelitian ini yaitu untuk:

1. Mengimplementasikan model arsitektur *Pre-trained* pada CNN berdasarkan multi input parameter untuk identifikasi jenis kopi berbasis GUI.
2. Mengevaluasi performa model-model *Pre-trained* yang digunakan untuk mengetahui arsitektur mana yang paling baik dalam mengidentifikasi jenis kopi berdasarkan multi input parameter.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat yang signifikan sebagai berikut.

1. Sebagai pedoman dalam menerapkan model *pre-trained* pada CNN untuk identifikasi jenis kopi berdasarkan multi input parameter.
2. Menghasilkan pemodelan dengan nilai akurasi yang baik untuk mengidentifikasi jenis kopi dengan akurat. Model tersebut akan dikembangkan berbasis GUI, memudahkan petani, pengamat kopi, ahli atau peneliti, serta masyarakat umum dalam membedakan jenis atau varietas kopi.

3. Meningkatkan efisiensi dalam identifikasi jenis kopi berdasarkan multi input parameter dengan menggunakan model arsitektur *Pre-trained* pada CNN.
4. Hasil Penelitian ini dapat menjadi pondasi untuk penelitian lanjutan dalam bidang pengenalan citra.

II. TINJAUAN PUSTAKA

2.1 Kopi

Kopi menjadi salah satu komoditas perkebunan yang memegang peranan cukup penting dalam perekonomian Indonesia (Zasari et al., 2023). Sampai saat ini tanaman kopi terus diusahakan penanamannya di Indonesia agar dapat memenuhi kebutuhan konsumsi di dalam negeri maupun luar negeri. Tanaman kopi diklasifikasikan dalam genus *Coffea Sp* termasuk famili *Rubiaceae*. Famili *Rubiaceae* dikenal memiliki sekitar 500 jenis dengan tidak kurang dari 600 species. Genus *Coffee* merupakan salah satu genus penting dengan beberapa *species* yang mempunyai nilai ekonomi dan dikembangkan secara komersial, yaitu kopi arabika (*coffea arabica*), robusta (*coffea canephora var. robusta*), dan kopi liberika (*coffee liberica*) (Siswoputranto, 1993).

Kopi Indonesia 98.14% (Tahun 2021) didominasi oleh kopi yang diusahakan di perkebunan rakyat. Jenis kopi yang diusahakan sepuluh tahun terakhir 71.65% merupakan kopi jenis robusta, sisanya sebesar 28.35% merupakan jenis kopi arabika(Andoko et al., 2023). Selain kopi jenis arabika dan robusta, ada varietas kopi lain yang juga dikenal, yaitu liberika. Namun, produksi kopi jenis liberika masih sangat terbatas di Indonesia.

Kopi Arabika

Kopi arabika adalah salah satu jenis kopi yang paling umum dan populer di dunia. Tanaman kopi arabika (*Coffea arabica*) berasal dari Ethiopia dan merupakan jenis kopi yang paling banyak ditanam di berbagai negara, termasuk Brasil, Kolombia, Kosta Rika, dan banyak lagi. Kopi arabika diduga menghasilkan rasa yang lebih unggul dan aroma lebih baik dibandingkan dengan yang lainnya. Kopi arabika terasa lebih asam, mengandung lebih banyak minyak, mengandung kafein yang lebih rendah (Farhaty, 2016).

Karakter morfologi yang menonjol pada kopi arabika adalah tajuk yang kecil dan ramping, dengan beberapa pohon memiliki cabang yang bersifat ketai dan daun yang berukuran kecil. Biji kopi arabika memiliki beberapa karakteristik khusus dibandingkan biji jenis kopi lainnya, seperti bentuknya yang agak memanjang, bidang cembungnya tidak terlalu tinggi, memiliki kilau yang lebih cerah dibandingkan dengan jenis lainnya, ujung biji mengkilap, dan terdapat lekuk pada bagian datarnya(Panggabean, 2011).

Kopi Robusta

Kopi robusta (Coffea canephora) adalah jenis kopi lain yang populer dan banyak ditemukan di berbagai wilayah, terutama di Afrika, Asia Tenggara, dan Brasil. Kadar kafein biji mentah kopi robusta lebih tinggi dibandingkan biji

mentah kopi arabika, kandungan kafein kopi robusta sekitar 2.2 %(Aditya et al., 2016).

Kopi robusta dikenal dengan kopi yang tahan (*robust*) terhadap berbagai penyakit dan lingkungan yang berubah-ubah, memiliki sifat lebih unggul dan sangat cepat berkembang, oleh karena itu kopi jenis robusta banyak dibudidayakan di Indonesia(Riastuti et al., 2021). Ciri morfologi yang khas pada kopi robusta meliputi tajuk yang lebar, daun yang lebih besar dibandingkan daun kopi arabika, serta pangkal tumpul (Andriani et al., 2022). Secara umum, biji kopi robusta memiliki rendemen yang lebih tinggi dibandingkan kopi arabika. Selain itu, karakteristik yang menonjol yaitu bijinya yang agak bulat, lengkungan bijinya yang lebih tebal dibandingkan kopi arabika, dan garis tengah dari atas ke bawah hampir rata(Panggabean, 2011). Ukuran biji kopi robusta sekitar 20-40% dari ukuran buahnya (Riastuti et al., 2021).

Kopi Liberika

Kopi Liberika (*Coffea liberica*) adalah jenis kopi yang kurang dikenal dan lebih jarang ditemui dibandingkan arabika dan robusta. Tanaman kopi Liberika asli berasal dari Liberia dan tumbuh subur di beberapa negara di Afrika Barat dan Asia Tenggara. Kandungan kafein pada kopi liberika lebih rendah dibandingkan dengan kopi arabika dan robusta. Kadar kafein pada kopi robusta sebesar 2.15%, kopi arabika sebesar 1.77% dan kopi liberika sebesar 1.32%(Aryadi et al., 2020). Meskipun tidak sepopuler arabika dan robusta, kopi Liberika tetap menjadi pilihan unik bagi pecinta kopi yang ingin mencoba sesuatu yang berbeda.

Ciri morfologi tanaman kopi liberika adalah daun besar dan kasar berbentuk elips dengan pinggiran daun bergelombang. Helai daun menonjol di antara urat-uratnya sehingga menimbulkan tekstur daun yang keriput (A Singapore Government Agency Website, 2023). Kopi Liberika memiliki ukuran yang lebih besar dibandingkan dengan jenis kopi lainnya. Bentuk biji membulat oval (panjang 0.83–1.10 cm, lebar 0.61 cm), dengan rendemen rata-rata 9.03% (Haniefan & Basunanda, 2022).

2.2 Pengolahan Citra Digital

Pengolahan citra merupakan bidang ilmu informatika yang mencoba meningkatkan kualitas gambar agar dapat memenuhi kebutuhan (Falah & Purnamasari, 2019). Secara umum pengolahan citra digunakan untuk melakukan proses pengolah citra digital yang meliputi penajaman citra, kompresi citra, penyempurnaan citra, dan klasifikasi citra. Hal ini juga melibatkan

penekanan fitur spesifik dari suatu citra dan bentuk yang tidak sesuai sehingga dapat memperlambat proses pengenalan citra(Muhimmah et al., 2018).

Langkah pertama dalam mengkonversi citra digital pada *low-level image processing techniques* adalah mendigitalkan citra atau citra yang telah ditangkap oleh suatu sensor (seperti kamera), yang kemudian mengurangi *noise* pada citra (*image pre-processing*) dan menyempurnakan beberapa fitur objek yang penting untuk pendeteksian citra seperti melakukan ekstraksi tepi(Sonka et al., 2014).

2.3 *Artificial Intelligence*

A*rtificial Intelligence* (AI) atau kecerdasan buatan merupakan bidang ilmu komputer tertua dan mencakup berbagai aspek dalam meniru fungsi kognitif manusia untuk menyelesaikan masalah dunia nyata. AI juga melibatkan pengembangan sistem yang mampu belajar seperti manusia. Oleh karena itu AI sering disebut sebagai kecerdasan buatan (Sonka et al., 2014).

Tujuan dari *artificial intelligence* (AI) adalah mengembangkan teknik dan sistem yang dapat menyelesaikan masalah-masalah yang sering diselesaikan oleh aktivitas intelektual manusia, seperti pengolahan gambar, perencanaan, ramalan, dan tugas-tugas lainnya, serta untuk meningkatkan fungsionalitas sistem informasi berbasis komputer. Ruang lingkup AI sangat luas dengan beberapa cabang ilmu pengetahuan, yaitu Pengenalan Ucapan (*Speech Recognition*), Pengolahan Bahasa Alami (*Natural Language Processing*), Sistem Pakar (*Expert System*), *Intelligent Computer-Aided Instruction, Computer Vision*, dan lainnya.

Artificial intelligence (AI) dibangun berdasarkan sistem yang memiliki keahlian serupa manusia pada rana tertentu disebut dengan *soft computing. Soft computing* merupakan perubahan baru dalam membangun sistem cerdas yang mampu menyesuaikan dan bekerja lebih baik jika terjadi perubahan pada lingkungan. *Soft computing* juga mengusahakan adanya toleransi terhadap ketidakpastian, ketidaktepatan, dan kebenaran parsial sehingga dapat diselesaikan, dipecahkan dan dikendalikan dengan mudah agar sesuai dengan kenyataan (Susatyono, 2021).

2.4 *Machine Learning*

Machine learning merupakan cabang ilmu bagian dari kecerdasan buatan (*artificial intelligence*) yang berfokus pada penggunaan data pembelajaran dan algoritma untuk meniru cara manusia belajar dan secara bertahap dapat meningkatkan kinerja atau membuat prediksi yang akurat. *Machine learning* digunakan untuk pengembangan sistem yang memungkinkan komputer menjadi cerdas berperilaku seperti manusia dengan memberikan kemampuan memahami secara otomatis melalui pengalaman. *Machine learning* memiliki fokus pada

pengembangan sistem-sistem yang dapat belajar sendiri untuk menyelesaikan tugas-tugas tanpa perlu diprogram berulang kali oleh manusia (Diana et al., 2017).

Dalam *machine learning* terdapat banyak metode dan algoritma yang secara garis besar terbagi menjadi 3 bagian yaitu, *supervised learning, unsupervised learning, dan reinforcement learning* (Id, 2021).

1. *Supervised learning*

   S*upervised learning* merupakan metode pembelajaran yang menggunakan data yang sudah diberi label untuk mengklasifikasikan kelas yang tidak dikenal. Ada beberapa algoritma yang termasuk dalam *supervised learning,* seperti *neural network, naïve bayes classifier, decision tree*, regresi linier berganda, analisis deret waktu dan beberapa algoritma lainya.

2. Uns*upervised learnin*g

   Unsupervised *learning* adalah jenis pembelajaran mesin di mana model mengelompokkan data input berdasarkan struktur atau pola yang ada dalam data tanpa memerlukan label kelas sebelumnya. *Unsupervised learning* sering digunakan untuk pemodelan *clustering.* Model ini digunakan untuk menganalisis suatu data untuk menemukan pola tersembunyi atau mengelompokkan suatu data. Algoritma yang biasa digunakan untuk *unsupervised learning,* seperti *fuzzy c-means, k-means,* dan *hierarchical clustering* dsb.

3. *Reinforcement learning*

   Pada pembelajaran *reinforcement learning* tidak memerlukan pengetahuan sebelumnya dan dapat secara independen mengembangkan kebijakan dengan mencoba-coba dan berinteraksi terus menerus dengan lingkungan yang berubah. Ada beberapa metode dalam *reinforcement learning* seperti metode berbasis model seperti algoritma SARSA, di mana *reinforcement learning* mempelajari pengetahuan model terlebih dahulu dan kemudian menghasilkan strategi optimal dari pengetahuan model tersebut dan metode yang relevan model seperti algoritma *Temporal Difference* dan algoritma *Q-learning,* di mana *reinforcement learning* menghitung strategi optimal secara langsung tanpa memerlukan pengetahuan model.

2.5 *Deep Learning*

Deep Learning merupakan bagian dari *machine learning* dengan banyak lapisan untuk membentuk tumpukan. Lapisan ini adalah algoritma atau metode

yang digunakan untuk memproses data sehingga mendapatkan hasil terbaik (Nurfita & Ariyanto, 2018). Metode ini cocok digunakan ketika menangani masalah yang kompleks atau masalah dengan jumlah data yang besar. Akibatnya, proses pelatihan dalam metode *Deep Learning* akan membutuhkan waktu yang lama. Penerapan *Deep Learning* telah digunakan di beberapa bidang seperti klasifikasi video, klasifikasi gambar, *object detection, object recognition, natural language processing, robotic, text classification,* dan *singing synthesis.*

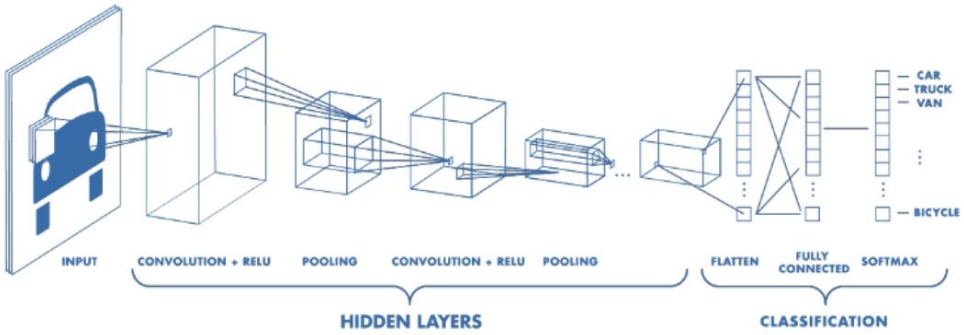
Deep Learning adalah sebuah arsitektur yang memanfaatkan konsep jaringan saraf tiruan yang lebih dalam. Dengan memanfaatkan konsep *neural network* yang lebih kompleks pada algoritma *machine learning*, komputer dapat belajar dengan kecepatan dan akurasi yang lebih tinggi, serta dapat menangani data dalam skala yang lebih besar. Dengan menggunakan metadata sebagai input, *Deep Learning* memproses informasi tersebut melalui *hidden layer*s (lapisan tersembunyi). Lapisan tersembunyi ini melatih fitur-fitur berdasarkan nilai *output* dari jaringan sebelumnya, dan proses ini berulang hingga menghasilkan nilai *output* yang akurat (Wijaya et al., 2021).

Salah satu keunggulan *Deep Learning* terletak pada kemampuan melakukan ekstraksi fitur atau *feature engineering*. Teknik *feature engineering* menjadi kunci penting dalam *Deep Learning*, memungkinkan model mengidentifikasi pola yang signifikan dari data yang kompleks. Algoritma yang digunakan dalam *feature engineering* dapat menemukan pola-pola penting yang membedakan kelas-kelas dalam *Deep Learning*.

2.6 *Convolutional Neural Network*

Convolutional neural network (CNN) merupakan salah satu algoritma dalam *Deep Learning* yang merupakan perkembangan dari *Multi-Layer Perceptron* (MLP). CNN dirancang khusus untuk memproses data dalam bentuk *grid*, seperti citra dua dimensi. Algoritma ini banyak digunakan untuk mengolah data gambar. CNN termasuk dalam kategori *Deep Neural Network* karena memiliki struktur jaringan yang dalam. Meskipun klasifikasi citra dapat dilakukan menggunakan MLP, penggunaan metode tersebut kurang ideal karena tidak dapat menyimpan informasi spasial dari citra. Pendekatan MLP menganggap setiap piksel sebagai fitur independen, sehingga hasil klasifikasi cenderung kurang akurat(I. W. S. E. Putra et al., 2016).

CNN terbukti memiliki kemampuan yang kuat dalam mengklasifikasikan citra dalam jumlah besar. CNN terdiri dari tiga lapisan utama yaitu *convolutional layer, pooling layers dan fully connected layer* (Sun et al., 2017). Gambar 2 adalah arsitektur CNN secara umum.



Gambar 2. Metode *Convolutional Neural Network*

Input layer, hidden layers, dan *output layer* adalah bagian dari sebuah *neural network*. Pada *input layer* data berupa citra akan dimasukkan dan diambil nilai dari tiap *pixel* citra, selanjutnya hasil input tersebut akan diproses pada tumpukan *hidden layers* yang ada pada algoritma CNN dan menghasilkan sebuah *output*. Pada *hidden layers* dalam jaringan saraf terdiri dari lapisan bertumpuk. Lapisan dalam *hidden layer* dapat bervariasi tergantung pada fungsi jaringan saraf dan bobot terkait. *Hidden layers* dapat mencakup *convolutional layer, pooling layers, normalization layer, activation layer, fully connected layer*(Alom et al., 2018).

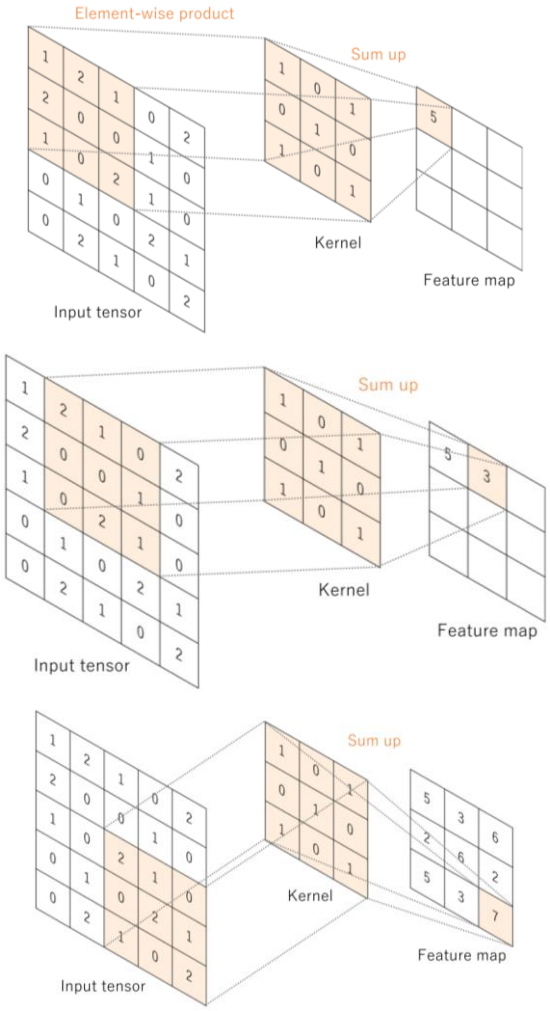
Convolutional Layers

Convolutional layers dalam *Convolutional neural networks* (CNN) memegang peran penting dalam operasi CNN. Lapisan ini menggunakan parameter yang berkaitan dengan penggunaan kernel yang dapat dipelajari. Kernel-kernel ini biasanya memiliki dimensi spasial yang kecil, tetapi menyebar ke seluruh kedalaman (*depth*) input. Ketika data citra masuk pada *convolutional layers*, lapisan ini melakukan konvolusi pada setiap filter yang melintasi dimensi spasial input untuk menghasilkan *output* berupa peta aktivasi 2D atau *feature map* (O'Shea & Nash, 2015).

Operasi konvolusi ini dapat dianggap sebagai hasil dari perkalian matriks antara citra input dan kernel yang dihitung melalui setiap "dot". Selain itu, pengaturan volume *output* juga dapat diatur oleh setiap lapisan melalui *hyperparameters.* Kernel adalah satu-satunya parameter yang dipelajari secara otomatis selama proses pelatihan dalam *convolutional layer*. Sebaliknya, ukuran kernel, jumlah kernel, *padding*, dan *stride* adalah *hyperparameter* yang perlu diatur sebelum proses pelatihan dimulai (Yamashita et al., 2018). Ukuran dari *feature maps* keluaran dapat dirumuskan sebagai berikut:

$$M = \frac{(V - N) + 2P}{S} + 1$$

Di sini, V merujuk pada dimensi-dimensi dari *feature maps* input, N merujuk pada dimensi-dimensi dari filter atau, *M* merujuk pada dimensi-dimensi dari *feature maps output*, P adalah nilai *padding*, dan *S* nilai *stride*. *Padding* biasanya diterapkan selama operasi konvolusi untuk memastikan *feature map* input dan *output* memiliki dimensi yang sama (Mubarok, 2019).



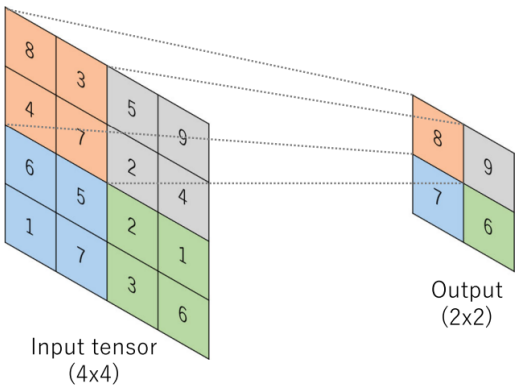
Gambar 3. Contoh Operasi Konvolusi

Sebagai contoh dari proses konvolusi pada *convolutional layer* dengan ukuran kernel 3 × 3, tanpa *padding*, dan *stride* bernilai 1. Sebuah kernel diterapkan melintasi tensor input, dan perkalian *elemen-wise* atau terpisah antara setiap elemen dari kernel dan tensor input dihitung pada setiap lokasi, kemudian dijumlahkan untuk mendapatkan nilai *output* pada posisi yang sesuai di tensor keluaran, yang disebut sebagai *feature map*s.

Pooling layers

Pooling operations digunakan dalam *convolutional neural networks* untuk mengurangi dimensi dari *feature maps* dengan merangkum subregion menggunakan fungsi seperti mengambil nilai maksimum atau nilai rata-rata. Tujuan dari *pooling layer* adalah untuk secara bertahap mengurangi dimensi dari data, yang juga mengurangi jumlah parameter dan kompleksitas model, sehingga mengontrol masalah *overfitting* (Sakib et al., 2019). Proses *pooling* ini menghasilkan *feature maps* yang lebih kecil, yang merupakan ringkasan dari input *feature maps* setelah proses convolutional.

Operasi *pooling* yang paling umum digunakan adalah *max pooling*, di mana nilai maksimum diambil dari setiap area kecil pada *feature maps*. Nilai-nilai lainnya diabaikan. Misalnya, dalam praktiknya sering digunakan *max pooling* dengan filter berukuran 2 × 2 dan *stride* 2, yang mengurangi dimensi *feature maps* sebanyak dua kali lipat dan dimensi kedalaman dari *feature maps* tetap tidak berubah(Yamashita et al., 2018).



Gambar 4. Operasi *Max Pooling*

Contoh operasi *max pooling* pada *pooling layer* dengan ukuran kernel 2x2 tanpa *padding* dan *stride* 2, yang mengekstrak potongan-potongan 2x2 dari tensor input, mengeluarkan nilai maksimum dari setiap potongan, dan mengabaikan nilai-nilai lainnya. Hasilnya adalah pengurangan dimensi bidang pada tensor input sebanyak dua kali lipat.

Fully connected layer

Fully connected layer merupakan lapisan yang berisi *neuron* dari lapisan-lapisan sebelumnya dijadikan satu untuk dilakukan proses klasifikasi menggunakan *neural network*. Setelah proses konvolusi dan *pooling*, data ditransformasikan menjadi vektor satu dimensi. Vektor ini menjadi input yang akan dihubungkan pada semua neuron di *fully-connected layer*. Lapisan ini biasanya digunakan pada metode *multi layer perceptron* (MLP) yang bertujuan untuk mengolah data sehingga dapat diklasifikasikan (Sakib et al., 2019).

Activation Layer

Activation function layer menerapkan *transformasi* non-linier pada *output* yang berasal dari convolutional dan *fully connected layers*. Jika tidak menggunakan fungsi aktivasi dalam *neural network*, *output* yang dikeluarkan akan berupa fungsi linear sederhana. Meskipun *output*nya sederhana dan dapat dengan mudah diselesaikan, namun kompleksitasnya terbatas dan tidak memiliki kemampuan untuk mempelajari dan mengenali pemetaan kompleks dari data. Oleh karena itu, fungsi aktivasi diperlukan karena memberikan jaringan saraf kemampuan untuk memahami dan memodelkan hubungan non-linear dalam data, sehingga memungkinkan untuk memahami pola yang lebih kompleks dan mengambil keputusan yang lebih baik. Dengan menggunakan fungsi aktivasi, jaringan saraf dapat belajar dan mengenali pola yang lebih rumit dalam data, membuatnya efektif dalam tugas-tugas seperti klasifikasi, prediksi, dan pengenalan pola. Beberapa fungsi aktivasi yang biasa digunakan, yaitu Sigmoid, Tanh, ReLU, dan *Softmax* pola (Sharma et al., 2020).

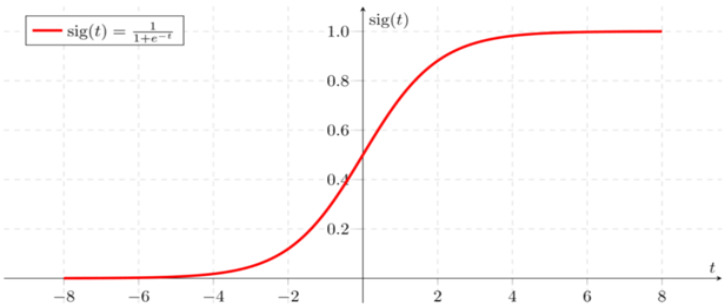
Fungsi sigmoid adalah fungsi aktivasi yang paling umum digunakan dalam jaringan saraf karena merupakan fungsi non-linear. Fungsi sigmoid mentransformasikan nilai dalam rentang 0 hingga 1. Fungsi ini dapat didefinisikan sebagai:

$$f(x) = 1/e^{-x}$$

Fungsi sigmoid kontinu dapat dikalkulasi turunannya dan memiliki bentuk kurva S yang halus. Turunan dari fungsi ini adalah:

$$f'(x) = 1 - sigmoid(x)$$

Fungsi sigmoid tidak simetris terhadap nol, yang berarti bahwa tanda dari semua nilai keluaran neuron akan sama.

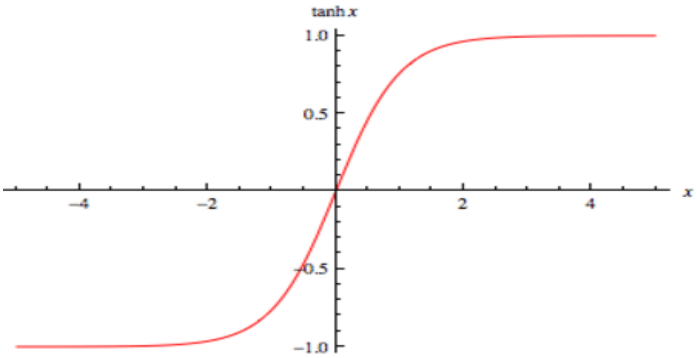


Gambar 5. Sigmoid

Fungsi Tanh (*Hyperbolic Tangent*) adalah fungsi aktivasi yang serupa dengan fungsi sigmoid, namun simetris terhadap asal. Hal ini menghasilkan tanda keluaran yang berbeda dari lapisan sebelumnya yang akan diberikan sebagai input ke lapisan berikutnya. Fungsi Tanh dapat didefinisikan sebagai:

$$f(x) = 2sigmoid(2x) - 1$$

Fungsi Tanh kontinu dan dapat dikalkulasi turunannya, nilai-nilainya berada dalam rentang -1 hingga 1. Dibandingkan dengan fungsi sigmoid, gradien fungsi Tanh lebih curam. Fungsi Tanh lebih disukai daripada fungsi sigmoid karena gradiennya tidak dibatasi untuk bervariasi dalam arah tertentu dan juga, fungsi Tanh memiliki nilai tengah nol.

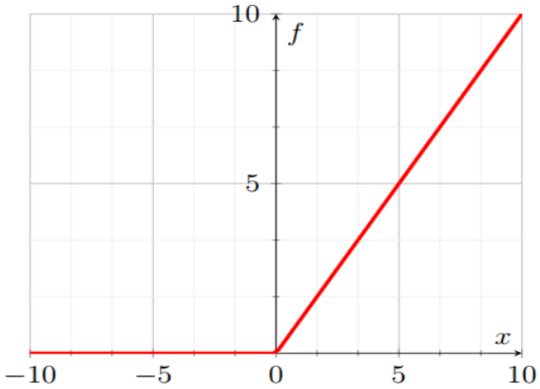


Gambar 6. Tanh

ReLU (*Rectified Linear Unit*) adalah fungsi aktivasi nonlinear yang banyak digunakan dalam jaringan saraf. Keuntungan menggunakan fungsi ReLU adalah bahwa tidak semua *neuron* diaktifkan pada saat yang sama. Ini berarti bahwa neuron akan dinonaktifkan hanya ketika keluaran dari transformasi linear adalah nol. Fungsi ReLU dapat didefinisikan secara matematis sebagai:

$$f(x) = max(0, x)$$

Fungsi ReLU lebih efisien dibandingkan dengan fungsi lain karena hanya sejumlah tertentu *neuron* yang diaktifkan pada satu waktu. ReLU memetakan input (0, x) selama proses. Dengan demikian, input bernilai negatif akan menghasilkan keluaran bernilai 0, sedangkan input bernilai positif akan menghasilkan keluaran yang tidak berubah.



Gambar 7. ReLU

Fungsi *Softmax* adalah kombinasi dari beberapa fungsi sigmoid. Seperti yang kita ketahui, fungsi sigmoid mengembalikan nilai dalam rentang 0 hingga 1, yang dapat dianggap sebagai probabilitas dari titik data kelas tertentu. Fungsi

Softmax, berbeda dengan fungsi sigmoid yang digunakan untuk klasifikasi biner, dapat digunakan untuk masalah klasifikasi multi kelas. Fungsi ini mengembalikan probabilitas untuk setiap titik data dari semua kelas individu. Fungsi *Softmax* dapat dinyatakan sebagai:

$$f(x_i) = \frac{exp(x_i)}{\sum_i exp(x_i)}$$

Keterangan:

x = nilai input lapisan sebelumnya

i = indeks unit

j = lapisan

Ketika kita membangun jaringan atau model untuk klasifikasi multi kelas, maka lapisan *output* dari jaringan akan memiliki jumlah *neuron* yang sama dengan jumlah kelas pada target. Fungsi *Softmax* digunakan sebagai fungsi aktivasi pada lapisan *output* dalam jaringan saraf tiruan dan dapat digunakan untuk memetakan nilai input ke dalam probabilitas untuk setiap kelas.

Batch Normalization

Batch Normalization merupakan salah satu teknik normalisasi yang sangat umum digunakan dalam *neural network*. Cara kerja *Batch Normalization* adalah dengan menyesuaikan dan menstabilkan distribusi input dari setiap lapisan dalam jaringan sehingga dapat mempercepat pelatihan. Hal ini dilakukan dengan menghitung *mean* dan *variance* dari setiap *batch* data selama pelatihan, kemudian menyesuaikan input dengan *mean* dan *variance* tersebut. Proses *Batch Normalization* terdiri dari beberapa langkah, yaitu menghitung *mean* dan *variance* dari setiap *batch*, normalisasi input, dan menghitung *scale & shift* Dengan demikian, *Batch Normalization* memastikan bahwa distribusi input tetap stabil dan tidak berubah secara signifikan selama pelatihan. Selain itu, *Batch Normalization* membantu menjaga konsistensi skala parameter jaringan, memungkinkan gradien bergerak lebih stabil melalui lapisan-lapisan selama proses pelatihan, sehingga memungkinkan penggunaan *learning rate* yang lebih tinggi tanpa risiko divergensi. Penggunaan *Batch Normalization layer* dilakukan setelah lapisan konvolusi dan sebelum fungsi aktivasi(Norhikmah et al., 2022).

Flattening Layer

Flatten berfungsi untuk membentuk atau mengubah objek menjadi vektor yang dapat digunakan sebagai nilai masukan pada *Fully-connected*(Wijaya et al., 2021). Setelah dilakukan proses konvolusi dan *average pooling, output* masih berbentuk *multidimensional array* sehingga belum dapat dijadikan input pada *neural network*. Input pada *neural network* harus berbentuk vektor satu dimensi. Oleh karena itu, fungsi *flatten* digunakan untuk mengubah *multidimensional*

array menjadi vektor tunggal atau *array* satu dimensi. *Output* dari fungsi *flatten* akan menghasilkan *array* satu dimensi yang dapat dijadikan input pada *fully connected layer*. Fungsi *flatten* digunakan pada jaringan saraf tiruan untuk mengubah data menjadi vektor satu dimensi sehingga dapat diolah lebih lanjut (Alwanda et al., 2020).

Dropout

Overfitting adalah masalah umum dalam *machine learning* yang terjadi ketika model terlalu fokus pada data pelatihan dan tidak dapat digeneralisasi dengan baik pada data uji. Salah satu cara untuk menghindari *overfitting* adalah dengan menggunakan teknik regularisasi, yang membatasi koefisien ke nol. *Dropout* adalah salah satu teknik regularisasi yang digunakan dalam jaringan saraf tiruan untuk menghindari *overfitting* (Yamashita et al., 2018).

Dropout adalah teknik regularisasi yang paling populer dan efektif dalam praktik *machine learning.* Teknik ini secara acak menghapus sejumlah *neuron* dalam jaringan pada setiap iterasi pelatihan, sehingga menghasilkan subnet yang lebih kecil dan mengurangi *overfitting*. Tanpa *Dropout*, pelatihan jaringan cenderung stagnan dan tingkat kesalahan pada data uji bahkan meningkat dan fluktuatif sebelum turunnya tingkat pembelajaran. Dengan *Dropout*, pelatihan menjadi lebih stabil dan jaringan terus meningkat untuk kesalahan yang lebih rendah. *Dropout* dapat diintegrasikan ke dalam blok bangunan CNN dan diimplementasikan dalam platform *Deep Learning* yang ada. *Dropout* adalah teknik regularisasi yang efektif dalam mengurangi *overfitting* pada CNN dan dapat mempercepat konvergensi jaringan selama pelatihan (Cai et al., 2019).

2.7 Multi-input CNN

Arsitektur Multi-input CNN digunakan ketika data input memiliki banyak parameter. Dalam hal ini, kernel konvolusi harus dibuat dengan jumlah saluran masukan yang sama dengan data masukan, sehingga dapat melakukan korelasi silang dengan data masukan. Arsitektur CNN dengan satu input tidak cocok untuk penelitian yang menggambarkan setiap sampel dengan mengambil setidaknya semua gambar, karena satu parameter tidak dapat mencakup seluruh data. Oleh karena itu, arsitektur Multi-input CNN digunakan (Sun et al., 2017).

2.8 *Transfer Learning*

Metode *transfer* learning menggunakan model yang telah dilatih sebelumnya terhadap suatu dataset untuk menyelesaikan permasalahan serupa sehingga tidak perlu melakukan pembelajaran dari awal. Metode ini memulai dengan menggunakan model tersebut sebagai *starting point,* kemudian

memodifikasi dan memperbaharui parameternya sehingga sesuai dengan *dataset* pada masalah yang baru (O. Saputra et al., 2022).

Salah satu cara yang paling umum untuk menerapkan *transfer learning* adalah melalui tahapan berikut. Tahap pertama mengambil lapisan (lapisan) dari model *pret-trained*, kemudian, melakukan pembekuan (*freeze*) pada lapisan model untuk mencegah informasi hilang selama pelatihan, setelah itu menambahkan lapisan baru di atas lapisan yang dibekukan dan yang terakhir, melatih *layer* baru pada dataset (Wijaya et al., 2021).

2.9 *Pre-trained* Model CNN

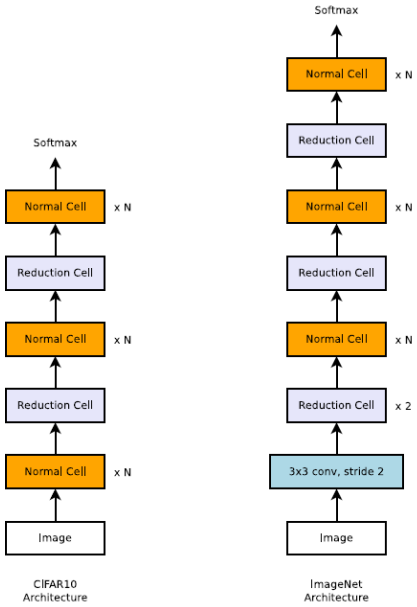
Mengembangkan Convolutional *neural network* (CNN) dapat menjadi mahal dalam hal waktu dan persyaratan *hardware* yang digunakan. Dengan menggunakan model yang telah dilatih sebelumnya, pengembang dapat menghemat waktu dan sumber daya dalam melatih model dari awal. Model-model *Convolutional neural network* (CNN) yang sudah mengalami proses *training* pada *dataset ImageNet challenge* disimpan dalam bentuk *Pre-trained* model (Pangestu & Bunyamin, 2018). Model yang telah dilatih sebelumnya juga dapat mencapai ekstraksi fitur dan pemodelan bahasa yang lebih baik(Chen et al., 2020). Banyak macam model *Pre-trained* CNN yang populer untuk digunakan, *NasNetMobile*, *Resnet50*, EfficienNetV2 dan arsitektur yang lainnya.

NasNetMobile

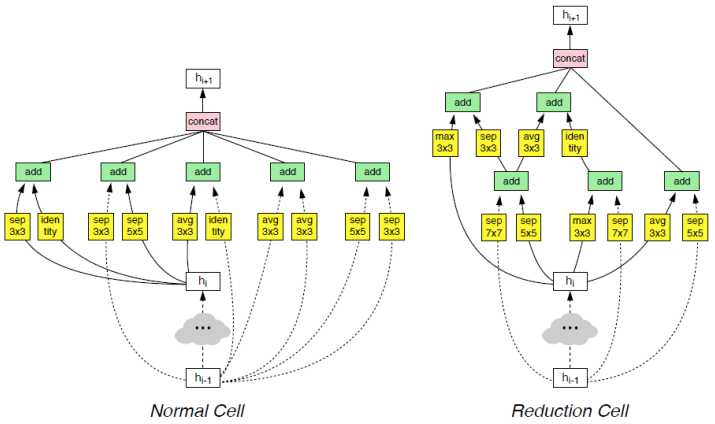
NasNetMobile merupakan salah satu varian arsitektur NasNet yang memiliki performa yang baik dengan kebutuhan komputasi yang lebih rendah, sehingga arsitektur ini dirancang khusus untuk aplikasi mobile. NASNet (*Neural Architecture Search Network*) adahal *neural network* yang dikembangkan Oleh Google Brain. NASNet adalah arsitektur *Deep Learning* yang dikembangkan menggunakan teknik *Neural Architecture Search* (NAS) yang merupakan pendekatan untuk mencari arsitektur jaringan saraf tiruan yang optimal dengan menggunakan *reinforcement learning search method*.

NAS adalah proses yang mengotomatisasi desain jaringan saraf, memungkinkan pengembangan arsitektur yang lebih baik daripada yang dibuat oleh ahli manusia. Tujuannya adalah untuk menemukan arsitektur jaringan saraf yang optimal untuk tugas tertentu. Dalam NAS, algoritma pencarian digunakan untuk mengeksplorasi ruang pencarian arsitektur jaringan saraf dan mencari arsitektur yang paling cocok untuk tugas yang diberikan (Dishar & Muhammed, 2023). NAS mulanya, digunakan sebagai metode search untuk mencari arsitektur *convolutional* terbaik pada dataset kecil seperti pada CIFAR10, dan kemudian mentransfer arsitektur tersebut ke dataset yang lebih besar,

seperti *ImageNet*. Proses pencarian ini dilakukan dengan merancang *search space* baru yang disebut NASNet *search space* yang memungkinkan transferabilitas arsitektur.



Gambar 8. Arsitektur untuk *ImageNet* dan CIFAR-10



Gambar 9. Arsitektur *NasNetMobile*

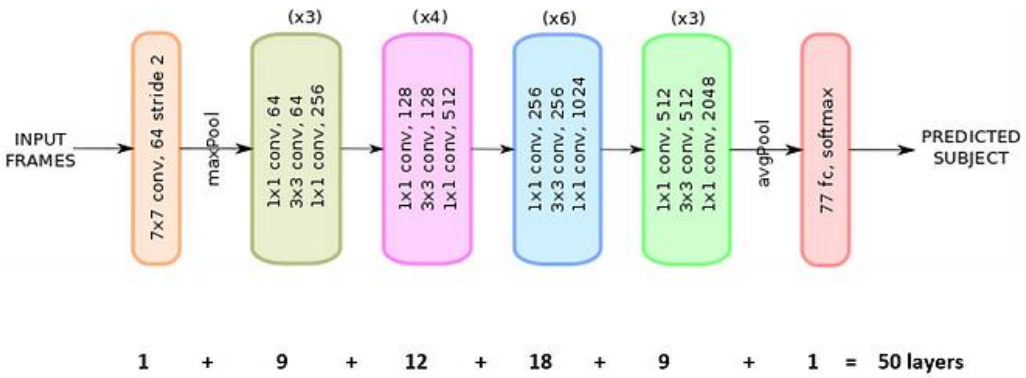
Dalam arsitektur NasNet, jumlah sel atau arsitektur tidak ditentukan dari awal melainkan dicari menggunakan metode pencarian *reinforcement learning*. Namun, ada dua jenis sel yang digunakan dalam arsitektur ini: *normal cell* dan *reduction cell*. *Normal cell* digunakan untuk *feature extraction* dan memiliki input dimensi *feature map* dan output yang sama, sedangkan *reduction cell* berfungsi untuk menghasilkan *output feature map* yang lebih kecil dengan mengurangi tinggi dan lebar masukan sebanyak dua kali lipat. Dalam hal ini, NasNet Menggunakan Pengendalian RNN untuk mencari struktur atau desain yang optimal untuk *normal cell* dan *reduction cell* dengan menentukan berapa banyak variasi pengulangan (N) dan jumlah filter konvolusi yang digunakan pada lapisan

awal sebagai parameter bebas(Zoph et al., 2017). Arsitektur *NasNetMobile* dapat dilihat pada Gambar 9.

Resnet50

ResNet-50 adalah arsitektur *convolutional neural network* yang terdiri dari 50 lapisan yang memuat lapisan yang telah dilatih sebelumnya menggunakan *ImageNet* (Hidayat et al., 2023). ResNet-50 merupakan salah satu varian dari arsitektur ResNet yang dianggap sebagai salah satu arsitektur jaringan saraf konvolusional paling populer. ResNet-50 memiliki 48 lapisan konvolusi, satu *MaxPool layer*, dan satu *average pool layer*. ResNet-50 dibagi menjadi enam tahap, arsitektur Resnet dimulai dengan lapisan konvolusi yang membentuk kernel dengan ukuran 7x7, 64 filter, dan stride 2. Setelah tahap ini, gambar memasuki lapisan *pooling* dengan *stride* 2, yang digunakan untuk merangkum informasi yang dihasilkan oleh konvolusi dan mengurangi dimensi input, serta mengurangi noise pada gambar. Ada dua jenis *pooling*: max *pooling* dan average *pooling*.

Konvolusi berikutnya terdiri dari kernel 1x1 dengan 64 filter, diikuti oleh kernel 3x3 dengan 64 filter, dan kernel 1x1 dengan 256 filter, lapisan ini diulang tiga kali, menghasilkan 9 lapisan baru. Konvolusi berikutnya melibatkan kernel 1x1 dengan 128 filter, diikuti oleh kernel 3x3 dengan 128 filter, dan akhirnya kernel 1x1 dengan 512 filter. Langkah ini diulang empat kali, menghasilkan 12 lapisan baru. Konvolusi berikutnya melibatkan kernel 1x1 dengan 256 filter, dua kernel 3x3 dengan 256 filter, dan kernel 1x1 dengan 1024 filter. Ini diulang enam kali, menghasilkan total 18 lapisan. Konvolusi berikutnya memperkenalkan kernel 1x1 dengan 512 filter, dua kernel 3x3 dengan 512 filter, dan kernel 1x1 dengan 2048 filter. Ini diulang tiga kali, menghasilkan 9 lapisan baru. Hasil dari setiap filter akan melewati average *pooling* dan masuk ke dalam jaringan fully connected dengan fungsi *Softmax* untuk mengetahui hasil klasifikasi (Tanjung et al., 2022). Arsitektur untuk ResNet-50 dapat dilihat pada gambar 8.



Gambar 10. Arsitektur *Resnet50*

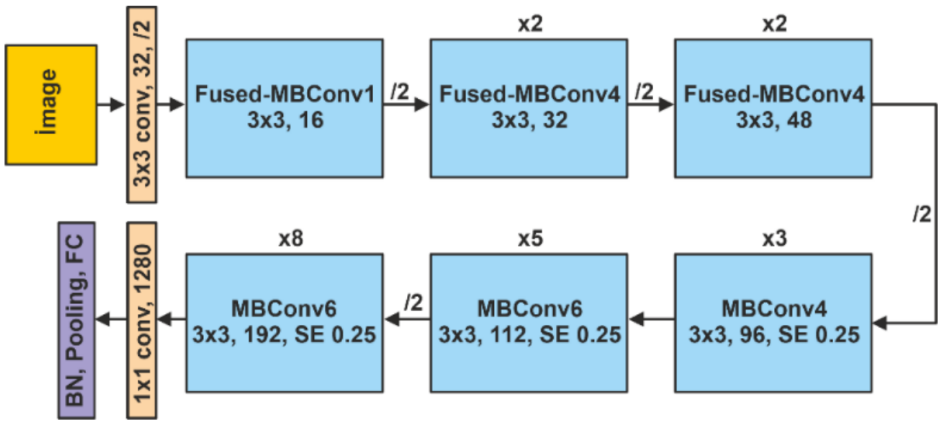
ResNet-50 adalah salah satu jaringan saraf yang paling populer dan telah digunakan dalam berbagai aplikasi *computer vision,* seperti pengenalan objek, deteksi objek, dan segmentasi gambar. Salah satu alasan menggunakan Resnet 50 yaitu meraih juara 1 dalam kompetisi klasifikasi ILSVRC 2015 dengan tingkat kesalahan top-5 sebesar 3,57%. Memenangkan Juara 1 dalam kompetisi ILSVRC dan COCO 2015 di *ImageNet Detection, ImageNet localization*, *Coco detection dan Coco segmentation* (Tsang, 2021).

EfficientNetV2B0

EfficientNetV2 adalah bagian model jaringan konvolusional yang memiliki kecepatan pembelajaran yang lebih cepat serta efisiensi parameter yang lebih baik dibandingkan model sebelumnya. Model ini direkomendasikan karena menggunakan *Neural Architecture Search* (NAS) yang mengatur pelatihan dan penskalaan untuk dapat mengoptimalkan akurasi model, kecepatan pelatihan, dan ukuran parameter. EfficientNetV2 dianggap lebih unggul karena dapat mempelajari model hingga empat kali lebih cepat daripada model sebelumnya sekaligus memiliki ukuran parameter enam kali lebih kecil (Tan & Le, 2021).

EfficientNetV2 adalah versi terbaru dari model arsitektur convolutional neural network (CNN) yang telah ditingkatkan untuk tugas klasifikasi gambar. EfficientNetV2 memiliki beberapa versi, yaitu *EfficientNetV2B0*, EfficientNetV2B1, EfficientNetV2B2, EfficientNetV2B3, EfficientNetV2L, EfficientNetV2M, dan EfficientNetV2S. *EfficientNetV2B0* adalah salah satu versi dari EfficientNetV2 yang memiliki dari susunan lapisan tersusun dari blok-blok MBConv dan Fused-MBConv yang meningkatkan efisiensi pelatihan sambil mengurangi ukuran model(Shabrina et al., 2023).

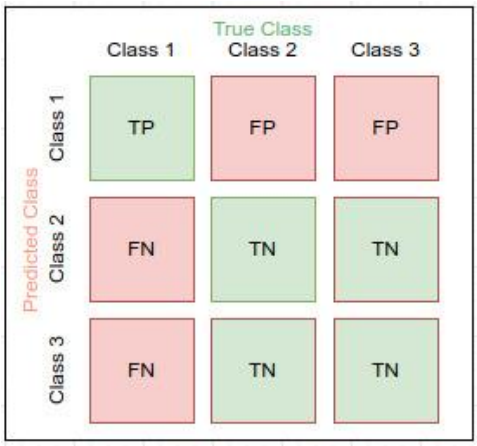
EfficientNetV2 memanfaatkan blok Convolution Mobile Inverted Bottleneck (MBConv) dengan rasio ekspansi yang lebih rendah dan Fused-MBConv yang diintegrasikan pada lapisan awal menggunakan kernel berukuran 3x3 dengan beberapa lapisan. Terdapat perbedaan yang menonjol dalam penggunaan lapisan antara MBConv dan Fused-MBConv dalam EfficientNetV2. MBConv mengadopsi depthwise conv3x3 dan Conv1x1, sedangkan Fused-MBConv mengimplementasikan Conv3x3. Struktur blok dalam arsitektur *EfficientNetV2B0* dapat terlihat dalam **Gambar 11**, yang terdiri dari *layer* Conv3x3 (*layer* konvolusi dengan kernel berukuran 3x3), Fused-MBConv 3x3 (kernel berukuran 3x3) dengan rasio ekspansi {1 dan 4} MBConv 3x3 (kernel berukuran 3x3) dengan rasio ekspansi {4 dan 6}, serta Conv1x1 (*layer* konvolusi dengan kernel berukuran 1x1) dan *Batch Normalization* diikuti oleh operasi Pooling dan Fully Connected *layer*.



Gambar 11. Arsitektur *EfficientNetV2B0*

2.10 Pengujian Model

Pengujian model dilakukan dengan menilai akurasi, *precision*, *F1 Score*, dan *recall* dari suatu model dengan menggunakan metode *Confusions matrix* untuk mengukur kinerja model. *Confusion matrix* digunakan untuk melakukan klasifikasi biner (klasifikasi 2 kelas) maupun klasifikasi multi kelas. Meskipun jumlah kelas berbeda, *Confusion matrix* memiliki konsep yang sama untuk klasifikasi biner dan multi kelas (Amin, 2023). Dalam tabel *Confusion matrix* menggambarkan kinerja model klasifikasi kelas-n dengan n baris dan n kolom yang merupakan sampel kelas sebenarnya dari input klasifikasi, dan hasil prediksi kelas yang merupakan *output* klasifikasi. Visualisasi dari *Confusion matrix* dapat dilihat pada gambar 9.



Gambar 12. *Confusion Matrix*

Pada umumnya Terdapat empat istilah yang merepresentasikan hasil proses klasifikasi pada *confusion matriks*, yaitu:

- *True positive (TP)* merupakan kondisi jika model memprediksi dengan benar dimana kelas sebenarnya adalah *positive,* dan hasil prediksi kelas sebagai kelas *positive.*

- *True negative (TN)* merupakan kondisi jika model memprediksi dengan benar dimana kelas sebenarnya adalah *negative,* dan hasil prediksi kelas sebagai kelas *negative.*
- *False positive (FP)* merupakan kondisi jika model memprediksi dengan salah dimana kelas sebenarnya adalah *negative,* dan hasil prediksi kelas sebagai kelas *positive.*
- *False negative (FN)* merupakan kondisi jika model memprediksi dengan salah dimana kelas sebenarnya adalah *positive,* dan hasil prediksi kelas sebagai kelas *negative.*

Nilai-nilai pada *confusion matrik*s dapat digunakan untuk mengevaluasi performa dari suatu model dari nilai akurasi, *precision, recall* dan *f1-score* yang dinyatakan sebagai:

$$Akurasi = \frac{TPall}{TP+TN+FP+FN} \quad \text{..... (1)}$$

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

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

$$F1 - score = 2 \times \frac{Recall \times Precision}{Recall+Precision} \quad \text{..... (4)}$$

2.11 Perancangan Graphical User Interface

Perancangan *Graphical User Interface (*GUI*)* adalah salah satu aspek penting dalam pengembangan aplikasi web. GUI adalah tampilan visual sebuah produk yang menghubungkan sistem dengan pengguna. *Flask* adalah sebuah mini Web *Framework* yang ditulis dalam bahasa pemrograman *Python*, dapat digunakan untuk membangun aplikasi web dengan GUI yang baik. *Flask* adalah sebuah *micro framework Python* yang menyediakan fungsionalitas dasar dari sebuah web *framework* dan memungkinkan penambahan *plug-in* untuk memperluas fungsionalitas dan fitur. *Flask* disebut sebagai *micro framework Python* karena membuat fungsionalitas inti menjadi sederhana dan mudah diperluas dalam pengembangan. *Flask* dapat digunakan untuk mengembangkan aplikasi web dengan mudah dan dapat menghemat waktu dalam pembuatan aplikasi web (Aslam et al., 2015).

2.12 Penelitian Terdahulu

Tabel 1. Penelitian Terdahulu

No	Judul Penelitian	Penulis dan Tahun	Isi	Metode	Hasil Penelitian
1	Klasifikasi Jenis Mangga Mengguna kan Algoritma CNN	(Yati et al., 2023)	Penelitian ini menggunakan algoritma CNN untuk membedakan 5 jenis mangga berdasarkan bentuknya. Data yang digunakan pada penelitian kali ini adalah 5 jenis mangga yaitu Mangga Apel, Mangga Arumanis, Mangga Gedong Gincu, Mangga Golek, dan Mangga Manalagi. Digunakan 375 citra mangga sebagai data set. Data set sebelum masuk ke dalam proses *train*ing sebelumnya dilakukan tahap *pre-processing* yang meliputi proses augmentasi dan resize.	Algoritma *Convolutional neural network* (CNN)	Dari hasil pengujian dengan *Confusion matrix* nilai akurasi terbaik dihasilkan dari *hyperparameter* input *size* 100x100, *epoch* 100, *learning rate* 0,001 dan *batch size* 15 dengan nilai *accuracy* 99,56%, *precision* 100%, *recall* 100%, dan f1 score 100%.
2	Analisis Performa Inception 3 Convolutio nal Network pada Klasifikasi Varietas Daun Grapevine	(Huda et al., 2023)	Penelitian ini memanfaatkan metode *machine learning* untuk mengklasifikasikan 5 jenis daun *Grapevine* menggunakan arsitektur Inception V3 yang merupakan salah satu arsitektur CNN. Dataset yang digunakan terdiri dari 500 gambar yang kemudian dibagi menjadi data *training* dan *testing* masing masing 80% dan 20%. Selanjutnya dilakukan *pre-processing* gambar dengan augmentasi. Hasil augmentasi digunakan untuk uji coba model	Arsitektur Inception V3 CNN.	Model yang diusulkan dalam penelitian ini berhasil mencapai akurasi 99.5% dengan menggunakan model Inception V3

No	Judul Penelitian	Penulis dan Tahun	Isi	Metode	Hasil Penelitian
3	Klasifikasi Varietas Kopi Berdasarkan *Green Bean Coffee* Menggunakan Metode *Machine learning*	(Nugroho & Sebatubun, 2020)	Penelitian ini melakukan proses identifikasi pada tiga jenis varietas kopi arabika yaitu *Red Bourbon,* Catimor dan Sigararutang dengan menggunakan salah satu metode *machine learning* yaitu *Deep Learning*. Penelitianan ini diawali dengan proses pengambilan data, kemudian tahap *pre-processing*, dan data citra akan langsung diklasifikasi menggunakan CNN untuk menguji kemampuan metode dalam mengenali varietas kopi.	Metode *Deep Learning*	Pada hasil *Confusion matrix Google autoML* dapat mendeteksi varietas ciwangi *red bourbon* sebesar 71.4%, ciwangi catimor 85.7%, dan rasuna sigararutang 80%
4	*Detection Brain Tumor Disease Using a Combination of Xception and NasNetMobile*	(Dishar & Muhammed, 2023)	Penelitian ini fokus pada pengembangan model CNN yang menggunakan *transfer learning* dengan menggabungkan dua arsitektur CNN, yaitu Xception dan *NasNetMobile* untuk mengklasifikasi gambar-gambar medis MRI otak untuk mendeteksi tingkat keparahan dari tumor otak,	Algoritma CNN, *Transfer learning* (Xception dan *NasNetMobile*)	Hasil penelitian menunjukkan bahwa penggunaan kombinasi arsitektur Xception dan *NasNetMobile* dengan teknik *transfer learning* dapat meningkatkan akurasi deteksi tumor otak hingga 99.83%. Penelitian ini menunjukkan bahwa penggunaan teknik *Deep Learning* dan *transfer learning* dapat meningkatkan akurasi deteksi tumor otak dan dapat menjadi solusi yang efektif untuk diagnosis dini dan

No	Judul Penelitian	Penulis dan Tahun	Isi	Metode	Hasil Penelitian
					pengobatan penyakit ini
5	Perbandingan Arsitektur Mobilenet dan *NasNetMobile* Untuk Klasifikasi Penyakit Pada Citra Daun Kentang	(Fuadi & Suharso, 2022)	Fokus pada penelitian ini adalah membandingkan arsitektur MobileNet dan *NasNetMobile* dalam melakukan deteksi penyakit pada daun tanaman kentang. Penelitian ini menggunakan beberapa skema pemisahan data latih dan data uji. Data yang digunakan pada penelitian ini terbagi menjadi gambar daun kentang sehat, gambar daun kentang terinfeksi *Early Blight*, dan gambar daun kentang terinfeksi *Late Blight*	Arsitektur MobileNet dan *NasNetMobile*	Skema pengujian menggunakan arsitektur *NasNetMobile* dengan perbandingan data latih dan data uji sebesar 90:10 menghasilkan *accuracy* sebesar 90.96%, *precision* sebesar 90.86%, *recall* sebesar 91.11%, dan f1 score sebesar 92.93%.
6	*Multi-Input Convolutional neural network For Breast Cancer Detection Using Thermal Images And Clinical Data*	(Sánchez-Cauce et al., 2021)	Penelitian ini mengembangkan pendekatan baru untuk deteksi dini kanker payudara dengan menggabungkan gambar termal dari berbagai sudut pandang data pribadi klinis (pasien), dengan membangun model klasifikasi multi-input yang memanfaatkan CNN untuk analisis gambar.	Metode *Deep Learning*, CNN	Hasil penelitian menunjukkan bahwa model terbaik mencapai akurasi 97% dan area di bawah kurva ROC sebesar 0,99, dengan spesifisitas 100% dan sensitivitas 83%
7	Identifikasi Mutu Biji Kopi Arabika Berdasarkan Cacat Dengan Teknik *Convolutional Neural Network*	(M. Saputra et al., 2020)	Penelitian ini menerapkan CNN untuk mengetahui informasi kecacatan biji kopi arabika. Input yang digunakan dalam penelitian ini adalah gambar biji kopi arabika dengan proses penguraian yang telah	Algoritma CNN.	Hasil percobaan menunjukkan bahwa akurasi terbaik yang diperoleh untuk model 2 kelas adalah 82,46 % dengan menggunakan

No	Judul Penelitian	Penulis dan Tahun	Isi	Metode	Hasil Penelitian
			dikeringkan.klasifikasi dilakukan oleh CNN, untuk mendapatkan akurasi model yang terbaik. Parameter yang ada harus diuji dan dievaluasi. Pengujian dilakukan untuk dua jenis model, model 2 kelas dan model 4 kelas.		tingkat pembelajaran 0.0001, konvolusi lapisan tunggal dengan lima belas filter dan 100 neuron pada lapisan tersembunyi. Ukuran filter adalah 3x3x3. Sedangkan model 4-kelas memperoleh akurasi terbaik 70.73% dengan dua lapisan konvolusional. Jumlah filter di setiap lapisan adalah 6 filter dengan ukuran 3x5x5 di lapisan pertama dan 18 filter dengan ukuran 6x3x3 di lapisan kedua
8	Analisis Perbandingan Optimizer pada Arsitektur *NasNetMobile Convolutional neural network* untuk Klasifikasi Ras Kucing	(Cahyo et al., 2023)	Fokus dari penelitian ini adalah menentukan optimizer yang optimal pada klasifikasi ras kucing. Dataset berisi 840 gambar yang dibagi menjadi 4 kelas dan dibagi menjadi 588 data *train*ing, 168 data *test*ing, dan 84 data validasi	Algoritma CNN dengan arsitektur *NasNetMobile*.	untuk optimizer RMSprop dengan learning rate 0.0001 mendapatkan akurasi 89.88% hasil ini merupakan yang tertinggi diantara yang lainnya. Sedangkan untuk optimizer SGD mendapatkan akurasi 78.57& hasil ini merupakan yang terendah. Sehingga dapat disimpulkan

No	Judul Penelitian	Penulis dan Tahun	Isi	Metode	Hasil Penelitian
					arsitektur dan optimizer sangat penting dan berpengaruh untuk meningkatkan kinerja model.
9	*Multi-Input Convolutional neural network for Flower Grading*	(Sun et al., 2017)	Penelitian pada jurnal tersebut membahas tentang penggunaan multi-input *convolutional neural network* (CNN) untuk melakukan *grading* atau klasifikasi bunga. Penelitian ini memperkenalkan dataset baru yang disebut BjfuGloxinia, yang berisi tiga tingkatan kualitas bunga.	Multi-input *convolutional neural network* (CNN)	Hasil penelitian menunjukkan bahwa multi-input *convolutional neural network* (CNN) yang diusulkan mencapai akurasi yang memuaskan sebesar 89.6% pada dataset BjfuGloxinia setelah augmentasi data. Dibandingkan dengan single-input CNN, CNN multi-input meningkatkan akurasi rata-rata sebesar 5%, menunjukkan efektivitasnya dalam penilaian bunga

III. METODOLOGI PENELITIAN

3.1 Tempat dan Waktu Penelitian

Data penelitian yang pertama adalah data kopi jenis Liberika yang diambil di perkebunan Kopi Liberika Mekar jaya yang berlokasi Kelurahan Mekar Jaya, kecamatan Betara Kabupaten Tanjung Jabung Barat yang menjadi lokasi budidaya kopi liberika di kecamatan Betara kabupaten Tanjung Jabung Barat, Provinsi Jambi. Data penelitian selanjutnya yaitu biji kopi arabika dan robusta yang diambil di Motokoffie Jambi yang berlokasi di Sungai Putri, Kec. Telanaipura, Kota Jambi. Untuk data berupa buah dan daun dari kopi jenis arabika dan robusta di Balai Pelatihan Pertanian (BPP) yang berlokasi Jl. Raya Palembang - Jambi No.16, Tj. Pauh Km. 32, Kec. Mestong, Kabupaten Muaro Jambi. Waktu penelitian Penelitian ini dilakukan pada bulan Juli 2023 hingga bulan februari 2024.

3.2 Alat dan Bahan Penelitian

Data yang diolah sebagai bahan penelitian ini berupa dataset yang terdiri dari citra daun, buah, dan biji kopi dari tiga jenis kopi yaitu kopi arabika, kopi robusta, dan kopi liberika. Alat yang dibutuhkan dalam penelitian ini, terdiri dari perangkat keras dan perangkat lunak sebagai berikut:

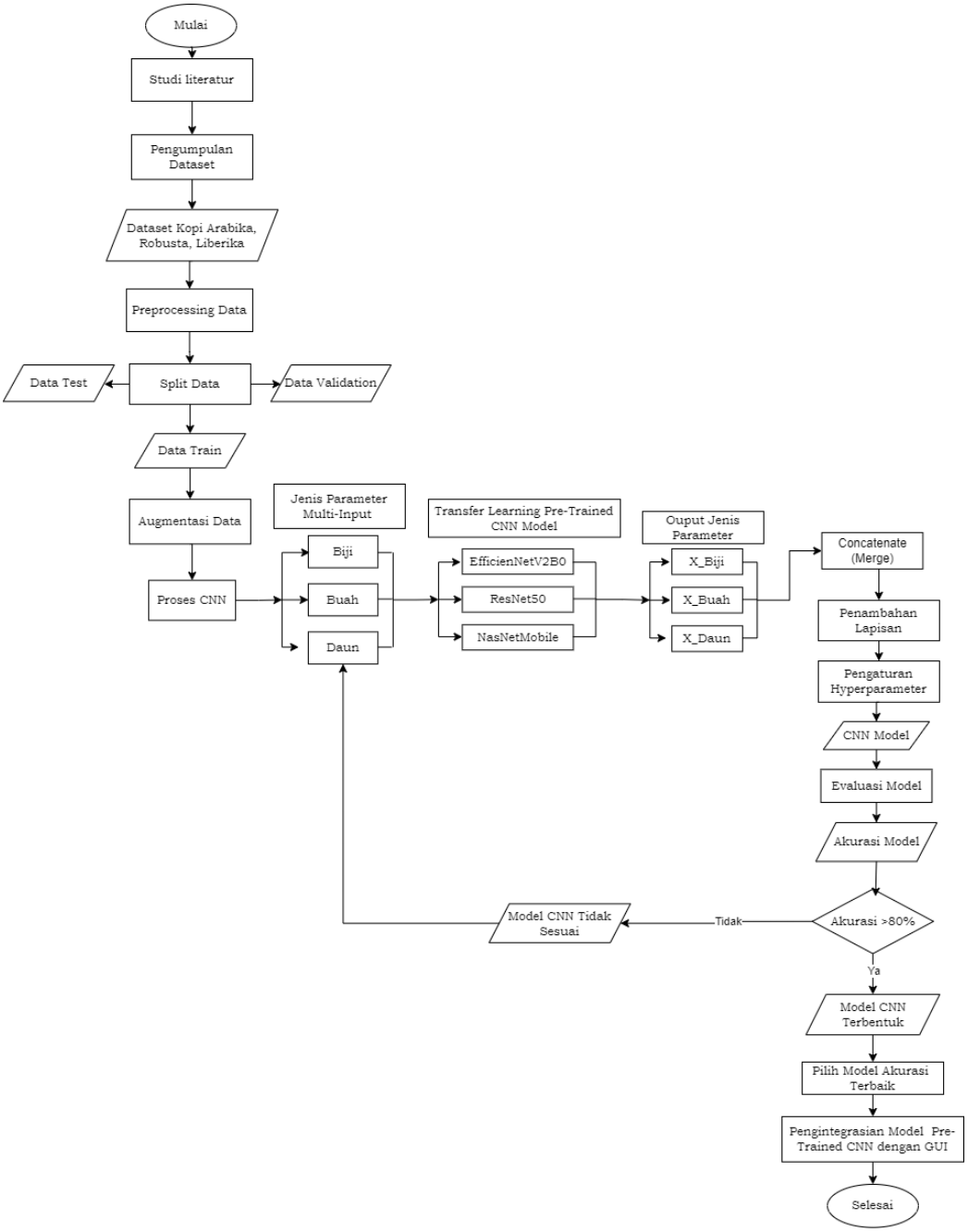
1. Perangkat keras
   a. Laptop dengan *processor* Intel(R) Core(TM) i5-M540, RAM 4 GB
   b. *Smartphone* Android Redmi Note 10 Pro dengan RAM 8+5 GB dan memori 128 GB
2. Perangkat lunak
   a. Sistem Operasi *Windows*
   b. Bahasa pemrograman *Python* 3.10
   c. Pengolahan data *Google colaboratory*
   d. *Framework Tensorflow, Keras,* dan *Flask*
   e. *Visual Studio Code* 1.82.3.

3.3 Kerangka Metode Penelitian

Studi Literatur

Pada tahap ini adalah tahapan dalam mengumpulkan berbagai informasi sebagai landasan teori dalam penelitian yang dilakukan. Literatur bersumber dari buku, Jurnal, artikel internet, dan sumber lain yang berkaitan dengan penelitian yang dilakukan.

Tahapan Penelitian



Gambar 13. Flowchart Penelitian

Pengumpulan Data

Pengumpulan dataset dengan jenis data yang digunakan pada penelitian ini merupakan jenis data primer dan data sekunder yang diperoleh dari beberapa sumber. Untuk pengumpulan data sekunder berasal dari situs Roboflow yang dilakukan secara manual dengan cara memilah dan menyeleksi gambar yang sesuai dengan kelas yang telah ditentukan. Sedangkan pengumpulan data primer dapat menggunakan beberapa device seperti kamera digital ataupun *smartphone*,

dengan minimal *requirement* yang perlu diperhatikan. Pertama, cahaya yang digunakan harus cukup untuk menghasilkan gambar yang jelas dan tidak terlalu gelap atau terang. Kedua, memastikan resolusi gambar yang digunakan memenuhi standar yang ditetapan. Ketiga, kondisi pengambilan gambar yang baik meliputi pengaturan fokus dengan memastikan objek yang difoto berada pada fokus yang tepat.

Pada penelitian ini dataset citra diambil menggunakan kamera *Smartphone* sesuai dengan kelas masing-masing dengan dimensi rata-rata sekitar 3000x3000 piksel. Saat pengambilan gambar dataset, pencahayaan dijaga agar relatif seragam dengan memanfaatkan cahaya matahari tidak langsung pada rentang waktu pukul 10 pagi hingga 2 sore dalam cuaca cerah untuk menjaga konsistensi kondisi pencahayaan yang mempengaruhi hasil citra kopi. Selain itu, dalam upaya memastikan kualitas dataset, setiap pengambilan gambar dilakukan hanya pada objek kopi tanpa adanya objek lain dalam frame. Hal ini bertujuan untuk fokus pada satu objek, baik itu citra daun, buah, atau biji kopi, dengan tujuan memudahkan proses *pre-processing* data pada tahap selanjutnya. Latar belakang dalam setiap citra juga dijaga agar bersih dan minim gangguan.

Pemilihan kamera smartphone dalam penelitian ini didasari oleh pertimbangan praktis dan ekonomis. Kamera smartphone memiliki portabilitas tinggi karena ukurannya yang kecil, ringan, dan mudah dibawa. Selain itu, kemajuan teknologi kamera smartphone juga mendukung pengambilan gambar dataset dengan kualitas yang cukup baik untuk penelitian ini. Dengan jumlah dataset yang digunakan adalah 900 dataset yang dibagi menjadi tiga kategori jenis kopi, yaitu kopi arabika, kopi robusta, dan kopi liberika. Setiap kategori terdiri dari 300 dataset, dengan pembagian 100 dataset untuk masing-masing parameter, meliputi citra daun, buah, dan biji kopi.

***Pre-processing* Data**

Tujuan *pre-processing* dataset untuk memastikan data yang digunakan pada model *pre-trained* CNN adalah data yang valid dan akurat. Tahapan awal dalam proses ini adalah melakukan *labeling* pada setiap gambar dalam dataset. Setiap gambar diberi label sesuai dengan kelasnya masing-masing dan disimpan ke dalam folder yang sesuai dengan kelas tersebut. Setelah itu, dilakukan penghapusan latar belakang gambar pada dataset menggunakan *library Rembg*. Proses ini bertujuan untuk memisahkan objek dari latar belakang sehingga model dapat lebih fokus pada objek terkait. Selanjutnya, gambar-gambar diubah menjadi ukuran seragam menggunakan metode *resizing. Resizing* merupakan tahap umum yang dilakukan untuk memastikan bahwa semua gambar dalam dataset memiliki dimensi yang konsisten sebelum diproses oleh model

Convolutional Neural Network (CNN), memastikan data memenuhi ukuran input yang diperlukan untuk model tersebut. Tahapan berikutnya adalah *denoising*, yang bertujuan untuk meningkatkan kualitas gambar dan memastikan bahwa data yang digunakan untuk pelatihan model tidak terganggu oleh *noise* yang tidak diinginkan.

Setelah dilakukan *denoising* pada data, *sharpening* kemudian dilakukan untuk meningkatkan kejelasan dan ketajaman gambar dengan menekankan tepi dan detail penting dalam gambar, membantu model dalam mengenali fitur-fitur utama dari objek yang ada dalam gambar. Setelah *sharpening*, dilakukan augmentasi dataset menggunakan *ImageDataGenerator*. Augmentasi ini menciptakan variasi dataset melalui transformasi seperti rotasi, pergeseran lebar dan tinggi, zoom, *flipping horizontal* dan vertikal, serta *fill mode*, memungkinkan model mengenali berbagai pola dari berbagai bentuk, orientasi, dan posisi objek dalam gambar. Setelah augmentasi, dilakukan *oversampling* untuk menyamakan jumlah gambar dalam setiap kategori (biji, buah, dan daun) yang tidak seimbang, memastikan model memiliki data yang seimbang dan dapat mempelajari representasi yang lebih baik dari setiap kelas. Terakhir, dataset dibagi menjadi tiga bagian utama untuk keperluan pelatihan, pengujian, dan validasi model secara proporsional: 70% untuk pelatihan (*train*), 15% untuk pengujian (*test*), dan 15% untuk validasi (*validation*). Pembagian ini mendukung pengukuran performa model secara objektif dan memastikan bahwa model dapat diuji dan divalidasi dengan data yang tidak pernah dilihat selama pelatihan.

Model Selection

Setelah itu data input hasil *pre-processing* dilakukan proses *training* terlebih dahulu dengan *Pre-trained* dari model-model CNN yang dipilih dengan melakukan uji coba dan evaluasi performa pada masing-masing *pre-trained* model. Model-model tersebut adalah *NasNetMobile*, *Resnet50*, EfficienNetV2B0. Proses *transfer learning* yang dilakukan pada ketiga model tersebut dilakukan dengan dataset jenis kopi untuk dapat membandingkan model *Pre-trained NasNetMobile* dengan model lainnya, dan menentukan model yang memiliki nilai akurasi yang tinggi. Proses *transfer learning* ini diperlukan karena model-model *pre-trained* belum pernah dilatih dengan dataset yang digunakan. Proses *transfer learning* dilakukan pada ketiga model tersebut untuk memanfaatkan pengetahuan yang telah diperoleh oleh model-model tersebut dari dataset *ImageNet* sebelumnya. Proses ini juga bertujuan untuk meningkatkan nilai akurasi yang diperoleh dari model.

Dalam proses transfer *learning* menggunakan model *pre-trained*, langkah pertama adalah memilih model *pre-trained* yang dijadikan *base model*. Model ini disesuaikan dengan bentuk input yang telah ditentukan sebelumnya selama proses augmentasi dataset. Setelah itu, lapisan-lapisan dari *base model* dibekukan (*freeze*) mencegah pembobotan pada *layer* tertentu agar tidak diperbarui selama pelatihan. Hal ini sangat penting karena memastikan pengetahuan yang telah ada pada *base model* tetap terjaga. Kemudian menggunakan *base model* dengan lapisan yang telah dibekukan ini untuk memproses setiap jenis input parameter (buah, biji, dan daun) dan menyimpan fitur-fiturnya. Dalam pembuatan multi-input, penulis menggunakan *concatenate* untuk menggabungkan output dari tiga jenis data (buah, biji, dan daun) setelah melewati model dasar (*base model*) yang sama. Dengan menggunakan *concatenate* dapat menggabungkan representasi-fitur pada setiap input menjadi satu tensor tunggal (x), yang kemudian menjadi input untuk lapisan selanjutnya.

Selanjutnya, menambahkan beberapa lapisan (*layers*) baru di atas lapisan output sehingga model belajar mengubah fitur lama menjadi kemampuan prediksi pada dataset yang baru. Pada penelitian ini penulis menambahkan *layer flatten, dense* dan droput. Pada lapisan terakhir, sebuah lapisan *dense* dengan 3 unit (sesuai dengan jumlah kelas yang ada: Arabika, Robusta, dan Liberika) dan fungsi aktivasi *Softmax* untuk menghasilkan *output* probabilitas untuk masing-masing kelas. *Flatten* memiliki peran krusial dalam melakukan transformasi terhadap hasil feature map yang diperoleh dari lapisan konvolusional, menjadikannya array satu dimensi. Hal ini menjadi penting karena sebelum memasuki lapisan *fully connected* hasil *feature m*ap masih berada dalam format array multidimensi. Fungsi *Dense* berperan dalam menghubungkan setiap neuron di suatu lapisan dengan neuron-neuron yang terdapat dalam lapisan berikutnya. Sementara itu, *Dropout* memegang peran penting dalam upaya untuk mengurangi kemungkinan terjadinya overfitting.

Langkah selanjutnya adalah pengaturan *hyperparameter* yang digunakan pada model seperti *batch size, epoch, learning rate, loss, dan optimizer.* Setelah model disusun dan lapisan-lapisan yang diperlukan ditambahkan, proses pelatihan dimulai menggunakan data *training*. Setelah proses pelatihan selesai, model CNN yang telah dibuat memiliki parameter-parameter yang dioptimalkan untuk dataset yang dimiliki.

Pengujian model

Pada tahap ini model yang telah dibangun akan diuji menggunakan data *testing* dan bertujuan untuk mendapatkan informasi detail mengenai akurasi prediksi pada seluruh data *testing*. Hal ini ditujukan untuk mengukur kinerja

model-model *pre-trained* CNN yang telah di bangun untuk kemudian menentukan apakah nilai akurasi yang didapatkan dari hasil pengujian model-model *pre-trained* dengan data *test* sudah cukup baik atau belum. Dari pengujian ini didapatkan informasi mengenai akurasi, *confusion matrix*, termasuk nilai *precision*, *recall* dan f1-score.

Confusion *matriks* untuk pengujian model yang telah dibuat dapat dilihat pada tabel (n) berikut.

Tabel 2. *Confusion Matriks* Model

Kelas Sebenarnya	Hasil Prediksi		
	Kopi Arabika	Kopi Liberika	Kopi Robusta
Kopi Arabika	TN	FP	TN
Kopi Liberika	FN	TP	FN
Kopi Robusta	TN	FP	TN

Pengembangan GUI dan pengintegrasian Model

Selanjutnya, setelah model *pre-trained* CNN berhasil dibangun, langkah berikutnya adalah mengimplementasikannya dalam bentuk *Graphical user interfac*e (GUI). Proses ini melibatkan pengembangan GUI menggunakan Flask, sebuah *framework web* untuk *python*. Flask digunakan untuk mengintegrasikan UI dengan sistem pemrosesan citra menggunakan Keras. Dalam tampilan antarmuka web yang dikembangkan, digunakan bahasa pemrograman HTML untuk elemen dasar, CSS, dan *Bootstrap* untuk memberikan gaya pada tampilan.

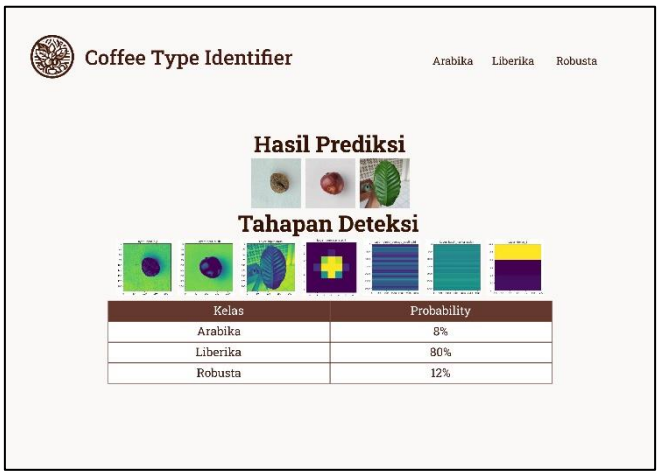
Alur pengujian pada sistem yang dibangun, dimulai dengan pengguna input citra buah, biji, atau daun kopi ke dalam aplikasi. Citra tersebut kemudian diproses dan diklasifikasi menggunakan model CNN yang telah tertanam dalam sistem. Model ini memberikan prediksi terhadap citra tersebut dalam bentuk klasifikasi jenis kopi. Hasil prediksi ini ditampilkan kepada pengguna melalui antarmuka grafis, memungkinkan pengguna untuk dengan mudah mengidentifikasi jenis kopi berdasarkan citra yang diunggah.

Halaman *website* yang dibangun pada penelitian ini antara lain halaman utama sistem, halaman prediksi dan halaman hasil prediksi input citra. Adapun *mockup* atau rancangan *user interface* yang dibangun pada penelitian ini. Halaman utama *website* berisi informasi berupa Logo, Nama *Website*, Animasi *Icon Coffee Recognition*, Deskripsi singkat mengenai informasi kegunaan *website*, menu Halaman jenis kopi dan *Button* untuk mulai menggunakan sistem.

Gambar 14. *Mockup* Halaman Utama *Website*

Gambar 15. *Mockup* Halaman Prediksi Input Citra

Pada halaman prediksi input citra digunakan untuk pengujian gambar. Dalam proses prediksi terdapat *button* "Pilih Gambar" yang digunakan untuk memilih gambar atau citra dari daun, buah, atau biji kopi yang tersimpan pada folder penyimpanan. Tahap selanjutnya prediksi gambar dilakukan dengan menekan *button* "Prediksi Gambar".



Gambar 16. *Mockup* Halaman Hasil Prediksi Input Citra

Halaman hasil prediksi input citra berisikan hasil prediksi jenis tanaman kopi yang telah diproses dengan hasil probabilitas dari tiga kelas dengan nilai probabilitas tertinggi merupakan kelas sesungguhnya. Pada halaman ini user dapat melihat bagaimana gambar diproses selama prediksi.

Gambar 17. *Mockup* Halaman Informasi Jenis Kopi Liberika

Halaman ini berisikan informasi jenis kopi hasil prediksi beserta gambar citra daun, buah dan biji dari jenis kopi tersebut. Pada setiap halaman terdapat menu navbar jenis-jenis kopi yang menampilkan halaman informasi mengenai jenis kopi lainnya.

Gambar 18. *Mockup* Halaman Informasi Jenis Kopi Arabika

Gambar 19. *Mockup* Halaman Informasi Jenis Kopi Robusta

3.4 Jadwal penelitian

Berikut ini adalah jadwal penelitian yang akan dilaksanakan.

Tabel 3. Jadwal penelitian

No	Kegiatan	Bulan											
		Jul	Agu	Sep	Okt	Nov	Des	Jan	Feb	Mar	Apr	Mei	Jun
1	Literature review												
2	Penyusunan proposal dan bimbingan												
3	Collecting data												
4	*Pre-processing* Data												
5	Rancangan Model												
6	Pengujian												
7	Menyusun laporan akhir												

IV. HASIL DAN PEMBAHASAN

4.1 Pengumpulan Dataset

Dalam tahap awal penelitian, penulis mengumpulkan dataset Citra Jenis Kopi berupa data primer dan data sekunder yang diperoleh dari beberapa sumber, yaitu Perkebunan Kopi Liberika Mekar Jaya, Motokoffie Jambi, Balai Pelatihan Pertanian dan situs Roboflow. Jumlah keseluruhan data citra yang berhasil dikumpulkan oleh penulis sebanyak 1.188 data citra yang dibagi kedalam 3 kelas yaitu arabika, liberika, dan robusta. Setiap kelas memiliki parameter masing-masing yang meliputi buah, biji, dan daun daun.

Dataset primer diperoleh langsung oleh penulis dengan menggunakan kamera *smartphone* untuk mengambil gambar secara langsung dari sumber-sumber tersebut. Sedangkan dataset sekunder berasal dari situs Roboflow, yang menyediakan gambar buah kopi Arabika. Penggunaan dataset sekunder dari Roboflow diperlukan karena jumlah data primer yang terkumpul tidak mencukupi atau tidak proporsional untuk kelas tertentu, dalam hal ini buah kopi arabika. Pengumpulan data gambar buah kopi arabika dari situs Roboflow dilakukan secara manual dengan cara memilah dan menyeleksi gambar yang sesuai dengan kelas yang telah ditentukan. Penggunaan kedua jenis dataset ini penting untuk memastikan kelengkapan dan representativitas data yang digunakan dalam penelitian, sehingga analisis dan hasil yang diperoleh dapat lebih akurat dan reliabel. Dalam proses pengumpulan dataset citra secara langsung, penulis menerapkan standar dan kriteria untuk memastikan kualitas dan representativitas data citra yang dikumpulkan.

Standar pengambilan gambar yang diterapkan mencakup berbagai aspek teknis. Pengambilan gambar biji kopi dilakukan secara individual, memastikan bahwa biji kopi yang difoto merupakan biji kopi mentah yang belum disangrai. Buah kopi dipotret per buah, bukan bergerombol, dengan jarak 20 cm dari objek dan zoom 5x. Daun kopi robusta dan liberika diambil pada jarak 20 cm tanpa zoom, sementara daun kopi arabika memerlukan pengaturan berbeda, yaitu diambil dari jarak 15 cm dengan zoom 2x. Penyesuaian ini disebabkan oleh ukuran daun arabika yang lebih kecil, sehingga memerlukan peningkatan zoom untuk mendapatkan detail yang cukup. Penyesuaian ini mencerminkan upaya untuk memastikan bahwa setiap jenis daun kopi direpresentasikan dengan baik dalam dataset, memperhatikan perbedaan ukuran dan detail morfologisnya. Dalam proses ini, penting untuk memastikan bahwa warna, tekstur, bentuk, dan pencahayaan objek terlihat jelas dan optimal.

Selain jarak, aspek lain dari standar pengambilan gambar meliputi resolusi, pencahayaan, sudut pengambilan, dan kualitas gambar. Resolusi ditetapkan pada 3000x3000 piksel untuk memastikan citra memiliki kejelasan yang memadai. Pencahayaan yang cukup terang dan merata diperlukan untuk menghindari bayangan yang tidak diinginkan dan memastikan representasi warna yang akurat. Sudut pengambilan dipilih untuk menampilkan objek secara jelas dan utuh, sementara kualitas gambar harus bebas dari *blur* dan *noise*.

Dalam proses pengumpulan dataset citra kopi arabika, liberika, dan robusta, penting untuk memahami perbedaan visual yang mencolok antara buah, biji, dan daun dari setiap jenis kopi tersebut. Dari visual gambar yang dihasilkan, dapat diamati bahwa buah kopi arabika memiliki bentuk yang lebih lonjong, dengan kulit buah yang tipis, sementara buah kopi robusta cenderung lebih bulat dan memiliki kulit buah yang lebih tebal dan buah kopi liberika memiliki bentuk yang lebih besar dan lebih gemuk dibandingkan dengan arabika dan robusta. Perbedaan ini tercermin dalam proses pengambilan gambar, di mana kriteria yang ketat diterapkan untuk memastikan representasi yang akurat dari karakteristik unik masing-masing jenis kopi.

Gambar 20. Data Citra Kopi Arabika

Gambar 21. Data Citra Kopi Liberika

Gambar 22. Data Citra Kopi Robusta

Selain itu, perbedaan dalam bentuk, ukuran, dan warna biji kopi juga menjadi fokus dalam pengambilan gambar. Misalnya, biji kopi arabika memiliki bentuk lonjong pipih dengan lekukan di tengah yang lebih halus, sementara biji kopi robusta cenderung lebih bulat dengan lekukan di tengah yang dangkal, dan

biji kopi liberika juga memiliki ciri khas tertentu, seperti bentuk lonjong pipih dengan lekukan di tengah yang lebih tajam dan permukaan yang lebih kasar.

Di sisi lain, daun kopi arabika memiliki bentuk oval memanjang dengan ujung yang runcing, tepi yang bergerigi halus, dan warna hijau tua yang mengkilap. Liberika memiliki bentuk oval besar dengan ujung yang runcing, tepi yang bergerigi kasar, dan warna hijau tua. Sementara itu, daun kopi robusta memiliki bentuk oval dengan ujung yang tumpul, tepi yang bergerigi halus, dan warna hijau tua yang mengkilap.

4.2 *Pre-processing* Data

Dataset yang telah terkumpul mengalami *pre-processing* sebelum dilakukan tahap pemodelan. *Pre-processing* merupakan tahap di mana data gambar yang telah dikumpulkan disiapkan dan dibersihkan sebelum digunakan dalam pemodelan.

Labeling

Proses *labeling* merupakan langkah pertama dalam tahap *pre-processing*, di mana setiap gambar diberi label sesuai dengan kelasnya masing-masing dengan memastikan rasio dan ketajamannya optimal. Diprioritaskan untuk memilih gambar yang merepresentasikan masing-masing kelas. Melalui proses ini, setiap gambar disimpan ke dalam folder yang sesuai dengan kelasnya. Dengan menyimpan gambar ke dalam folder kelas yang sesuai, dataset menjadi terstruktur dengan baik, memudahkan pengelolaan dan penggunaan data dalam proses selanjutnya. Proses *labeling* ini menghasilkan 100 gambar untuk setiap kelas.

Setelah proses *labeling* selesai, dataset diunggah ke *Google Drive*. Hal ini bertujuan agar saat diproses menggunakan *Google Colab, Google Colab* dapat terhubung dengan *Google Drive* yang berisi dataset tersebut. Di bawah ini tercantum kode program yang dapat digunakan untuk menghubungkan *Google Drive* dengan *Google Colab*:

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

Gambar 23. Kode Program Mengakses dataset

Jumlah keseluruhan citra yang terkumpul dari berbagai sumber setelah dilakukan *labeling* ditunjukkan pada tabel 4.

Tabel 4. Jumlah Citra

Data	**Sumber**	**Jenis Kopi**	**Parameter**	**Jumlah citra**
Data Sekunder	Roboflow	Arabika	Buah	100
Data primer	Perkebunan Kopi Liberika Mekar Jaya	Liberika	Biji	100
			Buah	100
			Daun	100

Data	Sumber	Jenis Kopi	Parameter	Jumlah citra
	Motokoffie Jambi	Arabika	Biji	100
		Robusta	Biji	100
	Balai Pelatihan Pertanian	Arabika	daun	100
		Robusta	Buah	100
			Daun	100
Total				**900**

Remove Background

Tahap *pre-processing* selanjutnya adalah proses *remove background*. Metode yang digunakan melibatkan penggunaan *library Rembg*. Proses ini difokuskan untuk memisahkan objek dari latar belakang serta mengubahnya menjadi warna putih, sehingga memungkinkan model untuk lebih berfokus pada pengenalan objek yang relevan dalam dataset. Dengan demikian, penghapusan latar belakang menjadi penting dalam mempersiapkan dataset untuk proses selanjutnya, karena hal ini membantu meningkatkan kualitas pemrosesan dan analisis yang dilakukan oleh model, dengan menghilangkan gangguan yang disebabkan oleh latar belakang yang tidak relevan.

```
# Menghapus latar belakang menggunakan Rembg
output_img = rembg.remove(img_file.read())
# Ubah gambar menjadi format PIL
img = Image.open(io.BytesIO(output_img))
# Buat latar belakang putih
new_img = Image.new("RGB", img.size, (255, 255, 255))
new_img.paste(img, (0, 0), img)
# Simpan gambar dengan latar belakang putih di folder output
new img.save(output filepath)
```

Gambar 24. Kode program menghapus latar belakang gambar

Resizing

Proses berikutnya setelah *remove background* pada data adalah proses *resizing* gambar. Dalam proses ini, dilakukan perubahan ukuran gambar dengan mempertahankan rotasi yang sudah ada tanpa mengorbankan informasi visual yang penting. Proses *resize* gambar dilakukan dengan melibatkan penggunaan *library Python Imaging Library (PIL)* atau *Pillow*. Tujuan dari proses ini adalah untuk mengubah ukuran gambar menggunakan metode *'resize()'* sesuai dengan parameter yang ditentukan, dalam hal ini, *'new_size'*. Proses ini penting karena dapat menghasilkan gambar dengan dimensi yang konsisten dan sesuai dengan kebutuhan model yang dilatih.

```
new_size = (224, 224)
with Image.open(input_file_path) as img:
  # Ubah ukuran gambar dengan mempertahankan rotasi
  resized_img = img.resize(new_size, resample=Image.BICUBIC)
  # Simpan gambar yang sudah diresize ke direktori output
  resized_img.save(output_file_path)
```

Gambar 25. Kode program *resizing* gambar

Denoising

Proses *denoising* gambar dilakukan menggunakan *library OpenCV (cv2)* dengan menggunakan filter median. Filter median digunakan karena efektif dalam menghilangkan *noise* sambil mempertahankan tepi gambar. Tujuan dari proses ini adalah untuk mengurangi *noise* atau gangguan yang mungkin muncul setelah proses *resize. Noise* dapat muncul dalam bentuk kecil yang menyebabkan gambar terlihat kabur atau tidak jelas yang terjadi karena adanya ketidaksempurnaan dalam proses pengambilan atau manipulasi gambar, sehingga menghasilkan gambar yang kurang jelas atau terdistorsi. Proses *denoising* gambar ini merupakan langkah penting dalam *pra-processing* data karena membantu meningkatkan kualitas gambar dan memastikan bahwa data yang digunakan untuk pelatihan model tidak terganggu oleh gangguan atau *noise* yang tidak diinginkan.

```
img = cv2.imread(input_file_path)
# Hapus noise menggunakan filter median
denoised_img = cv2.medianBlur(img, kernel_size)
# Simpan gambar yang sudah diproses ke direktori output
cv2.imwrite(output_file_path, denoised_img)
# Panggil fungsi remove_noise
remove noise(input directory, output directory)
```

Gambar 26. Kode program *denoising*

Sharpening

Proses *sharpening* gambar dilakukan dengan menggunakan *library OpenCV (cv2)* dalam bahasa pemrograman *Python*. Tujuan dari proses ini adalah untuk meningkatkan kejelasan dan ketajaman gambar dengan menekankan tepi dan detail penting dalam gambar. Proses *sharpening* ini dilakukan setelah proses *denoising* karena menghilangkan *noise* terlebih dahulu dapat membantu mengoptimalkan hasil dari proses *sharpening*. Proses *sharpening* diterapkan pada gambar menggunakan kernel yang telah ditentukan dengan menggunakan fungsi *'cv2.filter2D()'*.

```
def sharpen_image(image):
    # Matriks kernel untuk filter sharpening
    kernel = np.array([[-1, -1, -1],
                       [-1,  9, -1],
                       [-1, -1, -1]])
    # Aplikasikan filter sharpening
    sharpened_image = cv2.filter2D(image, -1, kernel)
    return sharpened_image
        image = cv2.imread(input_file_path)
        image = cv2.resize(image, None, fx=0.9, fy=0.9)
      # sharpening gambar
      sharpened_image = sharpen_image(image)
        # Simpan hasil pada folder output
        cv2.imwrite(output_file_path, sharpened_image)
sharpen_gambar(input_directory, output_directory)
```

Gambar 27. Kode program *sharpening*

Augmentasi

Setelah melakukan *sharpening* pada dataset, tahapan berikutnya yang dijalankan adalah augmentasi dataset. Tujuannya adalah untuk memperkaya variasi gambar yang digunakan oleh model. Variasi ini memungkinkan model untuk mengenali pola-pola yang berbeda dari berbagai bentuk, orientasi, dan posisi objek dalam gambar. Proses augmentasi mengunakan fungsi *ImageDataGenerator* dengan menerapkan beberapa teknik augmentasi. Pertama, dilakukan pengaturan ulang ukuran gambar (*resize*) menjadi 224x224 piksel sebagai langkah awal. Selanjutnya, berbagai teknik yang digunakan mencakup *rotation, width & height shift, share, zoom, flipping horizontal & vertical, fill mode.* Pada gambar 28 ditunjuukan kode program untuk pembuatan fungsi *ImageDataGenerator* yang digunakan pada proses augmentasi pada data set.

```
# Define augmentasi data
datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
  fill_mode='nearest')
#Generate augmented images and save them to the
directory
for image_path in image_files:
  # Load the image
  image = load_img(image_path)
  x = img_to_array(image)
  x = x.reshape((1,) + x.shape)
    i = 0
    for batch in datagen.flow(x, batch_size=1,
    save_to_dir=augmented_dir, save_prefix='aug',
    save_format='png'):
      i += 1
      if i >= 5:
         break
```

Gambar 28. Kode program fungsi *ImageDataGenerator*

Setelah menerapkan augmentasi, hasil augmentasi disimpan ke dalam folder direktori yang sesuai dengan label dan parameter dataset yaitu direktori *'augmented_dir'*, dengan menyimpan setiap gambar augmentasi dengan awalan *'aug'* dan format PNG. Contoh data yang telah melalui proses augmentasi yang disimpan kedalam direktori dataset dapat dilihat pada Gambar 29.

Gambar 29. Contoh data hasil augmentasi

Oversampling

Data yang telah mengalami proses augmentasi memiliki jumlah gambar dalam setiap parameter, yaitu biji, buah, dan daun tidak seimbang, pada parameter biji terdapat 1474 gambar, pada buah terdapat 1466 gambar, dan pada daun terdapat 1450 gambar, menghasilkan ketidakseimbangan dalam data yang digunakan untuk pelatihan model. Untuk mengatasi hal ini, dilakukan proses *oversampling*. Proses *oversampling* dilakukan dengan menggunakan modul *os* dan *shutil* dalam bahasa pemrograman *Python*. Tujuan utama dari proses ini adalah untuk menyamakan jumlah data setiap jenis input agar model yang dibangun memiliki data yang seimbang dan dapat mempelajari representasi yang lebih baik dari setiap kelas. Fokus utama dalam proses *oversampling* adalah pada penambahan data baru yang serupa dengan data yang sudah ada.

Setelah proses *oversampling*, jumlah data dalam setiap jenis telah diimbangi. Setiap parameter kini memiliki jumlah data yang sama, yaitu jumlah maksimum sampel per label yang telah ditentukan sebelumnya. Setelah proses *oversampling*, masing-masing jenis input memiliki 492 gambar untuk label 'ARABIKA', 493 gambar untuk label 'LIBERIKA', dan 489 gambar untuk label 'ROBUSTA'. Dengan demikian, total jumlah data setiap jenis input telah disamakan, memastikan bahwa model yang dibangun memiliki representasi yang seimbang dari setiap kelas. Hasil dari proses *Oversampling* data ditunjukan pada tabel 5 dibawah ini:

Tabel 5. Jumlah Data Setelah *Oversampling*

Kelas	**Jenis Input**	**Jumlah Gambar**
Arabika	Biji	492
	Buah	492
	Daun	492
Liberika	Biji	493
	Buah	493
	Daun	443
Robusta	Biji	489
	Buah	489
	Daun	489
Total		**4422**

```
# Maximum number of samples per label
max_samples_per_label = {'ARABIKA': 492, 'LIBERIKA': 493, 'ROBUSTA': 489}
def oversample_data(aug_dir, max_samples_per_label):
  total_data = 0
  for label in max_samples_per_label:
    label_dir = os.path.join(aug_dir, label)
    images = os.listdir(label_dir)
    current_count = len(images)
    if current_count < max_samples_per_label[label]:
       oversample_count = max_samples_per_label[label] - current_count
        for i in range(oversample_count):
            # Copy existing images to oversample
            src_image = os.path.join(label_dir, images[i % current_count])
            dst_image = os.path.join(label_dir, f"oversample_{i}.jpg")
            shutil.copyfile(src_image, dst_image)
       total_data += oversample_count + current_count
       else:
         total_data += current_count
  return total_data
# Perform oversampling
total_data_biji = oversample_data(biji_dir, max_samples_per_label)
total_data_buah = oversample_data(buah_dir, max_samples_per_label)
total_data_daun = oversample_data(daun_dir, max_samples_per_label)
```

Gambar 30. Kode program proses *Oversampling*

Pembagian *Dataset*

Setelah melakukan *oversampling* pada dataset, tahapan berikutnya yang dijalankan adalah pembagian dataset untuk keperluan pelatihan, pengujian, dan validasi model. Proses ini dilakukan dengan membagi dataset menjadi tiga bagian utama: set data latih (*train*), set data uji (*test*), dan set data validasi (*validation*), yang masing-masing memiliki proporsi tertentu dari total dataset. Pembagian dataset merupakan bagian penting dari tahapan pra-pemrosesan, yang bertujuan untuk memastikan kualitas data, membangun konsistensi evaluasi, dan meningkatkan objektivitas dalam menilai performa model yang dikembangkan.

Pembagian dataset dilakukan secara proporsional, di mana 70% dari dataset digunakan untuk pelatihan (*train*), 15% digunakan untuk pengujian (*test*), dan 15% digunakan untuk validasi (*validation*). Proses pembagian dataset dilakukan untuk setiap label dan parameter yang ada dalam dataset, yakni "ARABIKA", "LIBERIKA", dan "ROBUSTA", serta "buah", "biji", dan "daun".

Dalam implementasinya, setiap bagian dataset dibuat menjadi direktori tersendiri sesuai dengan label dan parameter yang bersangkutan. Kemudian, gambar-gambar dari dataset awal disalin ke direktori masing-masing set data (*train, test,* dan *validation*) sesuai dengan proporsi yang telah ditentukan secara

acak. Kode program dan tabel sebaran jumlah data setiap kelas dalam pembagian dataset pada folder direktori data adalah sebagai berikut:

```
# Define path to original data directory
data_dir = r'/content/drive/MyDrive/data primer/sharpening'
# Define path to directories for train, test, and validation data
output_dir = r'/content/drive/MyDrive/data primer/Data'
# labels and parameters
labels = ["ARABIKA", "LIBERIKA", "ROBUSTA"]
parameters = ['biji', 'buah', 'daun']
# Set the percentage of data to be used for training, testing, and validation
train_ratio = 0.7
test_ratio = 0.15
val_ratio = 0.15

# Loop through each label and parameter
for label in labels:
    for parameter in parameters:
        # Filter DataFrame berdasarkan label dan parameter saat ini
        subset_df = df[(df['labels'] == label) &
(df['filepaths'].str.contains(parameter))]
        # Bagi subset DataFrame menjadi set pelatihan, pengujian, dan validasi
        train_subset_df, dummy_subset_df = train_test_split(subset_df,
train_size=train_ratio, shuffle=True, random_state=123)
        test_subset_df, val_subset_df = train_test_split(dummy_subset_df,
test_size=test_ratio / (test_ratio + val_ratio), shuffle=True, random_state=123)
        # Tambahkan subset ke DataFrame yang sesuai berdasarkan parameter
        if parameter == 'biji':
            train_biji_df = pd.concat([train_biji_df, train_subset_df],
ignore_index=True)
            test_biji_df = pd.concat([test_biji_df, test_subset_df], ignore_index=True)
            val_biji_df = pd.concat([val_biji_df, val_subset_df], ignore_index=True)
        elif parameter == 'buah':
            train_buah_df = pd.concat([train_buah_df, train_subset_df],
ignore_index=True)
            test_buah_df = pd.concat([test_buah_df, test_subset_df], ignore_index=True)
            val_buah_df = pd.concat([val_buah_df, val_subset_df], ignore_index=True)
        elif parameter == 'daun':
            train_daun_df = pd.concat([train_daun_df, train_subset_df],
ignore_index=True)
            test_daun_df = pd.concat([test_daun_df, test_subset_df], ignore_index=True)
            val_daun_df = pd.concat([val_daun_df, val_subset_df], ignore_index=True)
```

Gambar 31. Kode program pembagian dataset

Berikut merupakan tabel sebaran jumlah data citra tiap class pada folder direktori data *train*ing dan data *validation*.

Tabel 6. Jumlah sebaran pembagian dataset

No	Kelas	Parameter	*Train*	*Validation*	*Test*	Jumlah
1	Arabika	Biji	344	74	74	492
		Buah	344	74	74	492
		Daun	344	74	74	492
2	Liberika	Biji	342	74	74	490
		Buah	342	74	74	490
		Daun	342	74	74	490
3	Robusta	Biji	345	74	73	492
		Buah	345	74	73	492
		Daun	345	74	73	492
Jumlah			3093	666	663	4422

4.3 Pembentukan Arsitektur Model CNN

Setelah melewati serangkaian tahapan *pre-processing*, langkah selanjutnya adalah melakukan pembentukan arsitektur model CNN dengan menggunakan tiga model yang berbeda, yaitu *NasNetMobile*, *Resnet50*, dan *EfficientNetV2B0*. Model-model tersebut memperoleh pengetahuan dari model *pre-trained* yang telah ada sebelumnya. Dalam proses pembangunan model, bobot atau weight dari *pre-trained* model tidak dilatih kembali untuk mempertahankan pengetahuan yang telah diperoleh pada pelatihan sebelumnya. Namun, setelah memasukkan *layer* input dari *pre-trained* model, lapisan tambahan ditambahkan untuk menyesuaikan pengetahuan yang sudah ada dengan tugas baru, yaitu identifikasi jenis kopi. Hal ini termasuk penambahan beberapa lapisan seperti *Global Average Pooling 2D Layer*, *Batch Normalization Layer*, *Dense Layer*, *Dropout Layer*, dan *Dense Layer* (Output *Layer*).

Pada penelitian ini, data gambar yang dimasukkan memiliki dimensi 224x224x3 untuk setiap jenisnya, termasuk input biji, buah, dan daun. Ketiga input tersebut diwakili oleh tiga lapisan input terpisah (biji, buah, dan daun) dengan ukuran yang sama, yaitu (224, 224, 3) masing-masing. Selanjutnya, lapisan output yang asli dari tiap model *pre-trained* yang digunakan sebagai dasar (*base model*) tidak dimasukkan. Proses ini melibatkan penerapan tiga versi dari model *pre-trained* yang sama untuk setiap jenis input, yang masing-masing menerima inputnya sendiri dan mengahasilkan fitur-fitur yang sesuai dengan jenis input yang diberikan. Setelah fitur-fitur tersebut dihasilkan, langkah selanjutnya adalah menggabungkan keluaran dari tiga parameter tersebut. Ini dilakukan dengan menggunakan lapisan *concatenate* dari TensorFlow, di mana fitur-fitur yang dihasilkan dari setiap jenis input (buah, biji, dan daun) digabungkan menjadi satu tensor tunggal. Hal ini memungkinkan informasi dari ketiga jenis input untuk disatukan dan diproses secara bersamaan oleh lapisan-lapisan selanjutnya dalam arsitektur model.

```
# Buat lapisan input terpisah untuk setiap parameter
input_buah = Input(shape=input_shape_buah, name='input_buah')
input_biji = Input(shape=input_shape_biji, name='input_biji')
input_daun = Input(shape=input_shape_daun, name='input_daun')
# Buat model dasar untuk setiap parameter
base_model = ResNet50(include_top=False, weights="imagenet",
input_shape=(224, 224, 3))
# base_model = NASNetMobile(include_top=False, weights="imagenet",
input_shape=(224, 224, 3))
# base_model = EfficientNetV2B0(include_top=False, weights="imagenet",
input_shape=(224, 224, 3))
# Freeze the weights of the pre-trained layers
for layer in base_model.layers:
    layer.trainable = False
# Define feature extraction layers for each type of input
x_buah = base_model(input_buah)
x_biji = base_model(input_biji)
x_daun = base_model(input_daun)
# Gabungkan keluaran dari tiga parameter
x = tf.keras.layers.concatenate([x_buah, x_biji, x_daun])
# Gabungkan keluaran dari tiga parameter
x = tf.keras.layers.concatenate([x_buah, x_biji, x_daun])
# Add custom head for classification
x = GlobalAveragePooling2D()(x)
x = BatchNormalization()(x)
x = Dense(128, activation='relu')(x)
x = Dropout(rate=0.5)(x)
output = Dense(num_classes, activation='softmax')(x)
# Create the final model
model = Model(inputs=[input_buah, input_biji, input_daun], outputs=output)
```

Gambar 32. Kode program untuk Pembentukan Model

Dalam pemrosesan data multi-input menggunakan model CNN, penggunaan lapisan *concatenate* sangat penting untuk menggabungkan informasi dari berbagai jenis input. Lapisan *concatenate* berfungsi untuk menyatukan fitur-fitur yang dihasilkan oleh model *pre-trained* dari tiap jenis input (biji, buah, dan daun) menjadi satu representasi tensor tunggal. Proses ini memungkinkan model untuk mempertimbangkan dan menganalisis informasi dari semua sumber input secara holistik.

Cara kerja lapisan *concatenate* adalah dengan mengambil tensor-tensor output dari setiap model *pre-trained* yang menerima jenis input yang berbeda dan kemudian menyatukan tensor-tensor tersebut sepanjang sumbu yang ditentukan. Dalam hal ini, tensor-tensor dari input biji, buah, dan daun digabungkan sehingga menghasilkan tensor baru yang mengandung informasi gabungan dari ketiga sumber tersebut. Penggabungan ini dilakukan setelah

lapisan-lapisan *pre-trained* menghasilkan fitur-fitur masing-masing, memastikan bahwa informasi yang kaya dan bervariasi dari setiap jenis input dipertahankan dan digunakan secara efektif dalam lapisan-lapisan selanjutnya dalam arsitektur model.

Dengan menggabungkan informasi dari berbagai jenis input, model dapat mengeksploitasi korelasi dan hubungan antara fitur-fitur yang dihasilkan dari tiap jenis input, yang berpotensi meningkatkan kinerja dan akurasi prediksi model. Penggunaan lapisan *concatenate* menjadi esensial dalam menciptakan model yang lebih robust dan mampu menangani kompleksitas data multi-input dengan lebih efektif.

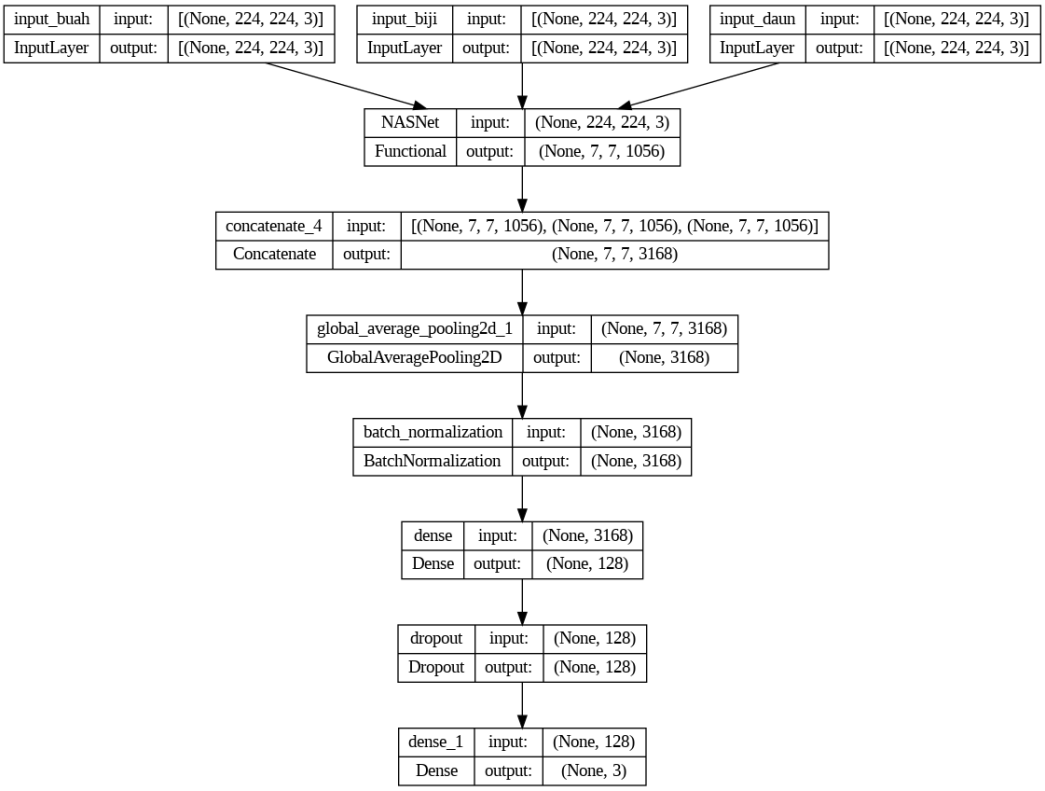
Selanjutnya, lapisan *GlobalAveragePooling2D* digunakan sebagai langkah untuk mereduksi dimensi spasial dari fitur yang diperoleh dari *pre-trained* model. *Layer* ini mengambil rata-rata dari fitur yang dihasilkan oleh lapisan sebelumnya dalam tiap saluran, menghasilkan vektor fitur global dari seluruh gambar. Setelah lapisan *GlobalAveragePooling2D*, terdapat lapisan BatchNormalization yang diterapkan untuk normalisasi aktivasi dari lapisan sebelumnya, memastikan bahwa model tetap stabil dan mempercepat proses pembelajaran.

Langkah berikutnya adalah proses d*ense* dilakukan dengan menambahkan *fully-connect*ed *layer* yang terdiri dari 128 unit dengan fungsi aktivasi ReLU pada *hidden layer*nya. Tujuan dari langkah ini adalah untuk mengekstraksi fitur yang lebih abstrak dari fitur-fitur yang telah diperoleh sebelumnya. Selanjutnya, lapisan *Dropout* diterapkan dengan tingkat *Dropout* sebesar 0.5 sebagai probabilitas pada setiap neuron yang diharapkan dapat mencegah atau mengurangi overfitting sekaligus mempercepat pembelajaran Ini berfungsi dengan mengatur nilai nol pada sejumlah unit secara acak selama proses pelatihan, sehingga mengurangi ketergantungan pada kumpulan data pelatihan tertentu.

Terakhir, lapisan output ditambahkan dengan tiga unit output sesuai dengan jumlah kelas atau label pada penelitian ini, yaitu 'arabika', 'liberika', dan 'robusta'. Karena ini adalah tugas multi-kelas, fungsi aktivasi yang digunakan pada lapisan output adalah softmax, yang mengembalikan probabilitas dari setiap kelas, di mana kelas target memiliki nilai probabilitas yang lebih tinggi daripada kelas lainnya.

NasNetMobile

Berikut adalah ringkasan model dan representasi visual dari arsitektur model yang menggunakan *pre-trained* model *NasNetMobile*:



Gambar 33. Representasi visual arsitektur model *NasNetMobile*

Layer (type)	Output Shape	Param #
input_buah (InputLayer)	[(None, 224, 224, 3)]	0
input_biji (InputLayer)	[(None, 224, 224, 3)]	0
input_daun (InputLayer)	[(None, 224, 224, 3)]	0
NASNet (Functional)	(None, 7, 7, 1056)	4269716
concatenate_4 (Concatenate)	(None, 7, 7, 3168)	0
global_average_pooling2d_1 *GlobalAveragePooling2D*)	(None, 3168)	0
batch_normalization (Batch Normalization)	(None, 3168)	12672
dense (Dense)	(None, 128)	405632
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 3)	387

Total params: 4688407 (17.88 MB)
Trainable params: 412355 (1.57 MB)
Non-trainable params: 4276052 (16.31 MB)

Gambar 34. Model *Summary* (*NasNetMobile*)

Ringkasan dari model yang telah dibangun mencakup detail dari setiap lapisan dalam model, termasuk *output shape* serta jumlah parameter yang terlibat dalam proses pelatihan untuk setiap lapisannya. Berikut adalah hasil ringkasan model menggunakan *NasNetMobile*:

Tabel 7. Deskripsi Model *Summary* (*NasNetMobile*)

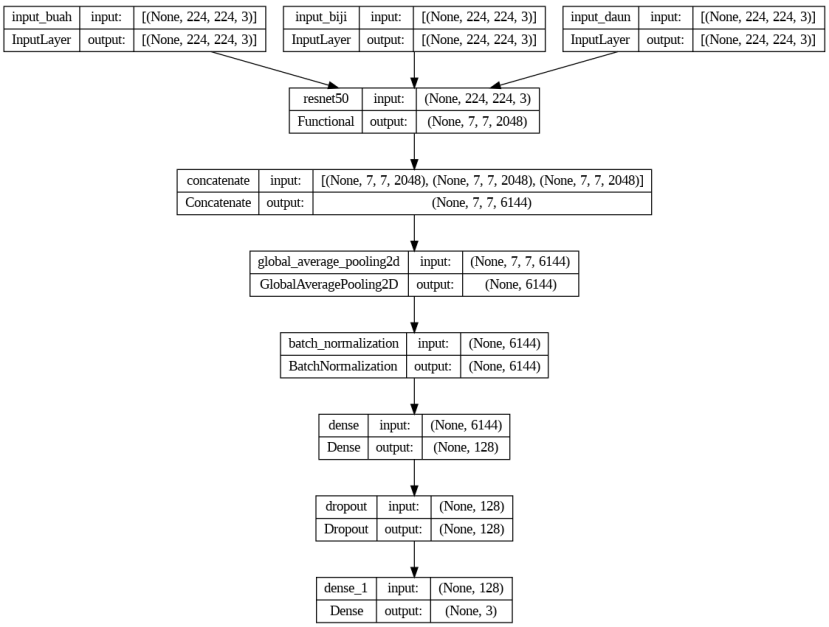
Layer (type)	**Deskripsi**	
	Output shape	**Parameter**
Input_buah	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
Input_biji	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
Input_daun	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
NASNet	(None, 7, 7, 1056)	*Layer* ini adalah model *pre-trained* NASNet yang digunakan sebagai ekstraktor fitur. Model ini menghasilkan output dengan ukuran 7x7 piksel dan 1056 saluran. Jumlah parameter yang besar (4269716) mencakup bobot dan bias yang telah dilatih sebelumnya untuk menghitung output.
*Concatenate*_4	(None, 7,7, 3168)	*Layer* ini menggabungkan output dari beberapa *layer* sebelumnya. Dalam hal ini, menggabungkan tiga keluaran dari NASNet, masing-masing dengan 1056 saluran, menjadi satu tensor dengan 3168 saluran. *Layer* ini tidak memiliki parameter yang dapat dilatih.
Global_average_ pooling2d_1	(None, 3168)	*Layer* ini melakukan pooling global rata-rata pada tensor 7x7, mereduksi tiap saluran menjadi satu nilai. Outputnya adalah vektor dengan 3168 elemen, tanpa parameter yang dapat dilatih.
Batch_Normalization	(None, 3168)	*Layer* ini menormalkan aktivasi dari *layer* sebelumnya, membantu mempercepat pelatihan dan stabilitas. Jumlah parameter 12672 mencakup bobot dan bias yang telah dilatih sebelumnya.
Dense	(None, 128)	Jumlah parameter 405632 mencakup bobot dan bias yang telah dilatih sebelumnya untuk menghitung input dan output dengan 128 neuron.
Dropout	(None, 128)	*Layer* ini digunakan untuk mencegah overfitting dengan mengabaikan unit-unit tertentu

Layer (type)	**Deskripsi**	
	Output shape	**Parameter**
		secara acak selama pelatihan. *Layer* ini tidak memiliki parameter yang dapat dilatih.
Dense _1	(None, 3)	*Layer* fully connected terakhir dengan 3 unit, sesuai dengan jumlah kelas output. Jumlah parameternya terdiri dari 128 input x 3 unit = 384 bobot, ditambah 3 bias, totalnya 387.

Dari total parameter sebanyak 4.688.407, sebanyak 412.355 parameter adalah yang dapat dilatih selama proses pelatihan model. Parameter ini termasuk bobot dan bias yang disesuaikan oleh algoritma pembelajaran mesin untuk meminimalkan kesalahan prediksi. Sementara itu, sebanyak 4.276.052 parameter lainnya adalah parameter yang tidak dilatih. Parameter ini termasuk parameter dari model *pre-trained* seperti NASNet, yang telah dilatih pada dataset besar seperti *ImageNet*, serta parameter statistik dari lapisan *Batch Normalization*. Meskipun parameter non-*train*able ini tidak diubah selama pelatihan model, mereka tetap penting untuk menyediakan informasi yang diperlukan bagi model dalam melakukan inferensi yang akurat. Dengan demikian, walaupun mayoritas parameter dalam model tidak dapat diubah selama pelatihan, mereka tetap berperan penting dalam memastikan kinerja model yang optimal.

Resnet50

Berikut adalah ringkasan model dan representasi visual dari arsitektur model yang menggunakan *pre-trained* model *Resnet50*:



Gambar 35. Representasi visual arsitektur model *Resnet50*

```
Layer (type)                 Output Shape              Param #
=================================================================
input_buah (InputLayer)      [(None, 224, 224, 3)]     0
input_biji (InputLayer)      [(None, 224, 224, 3)]     0
input_daun (InputLayer)      [(None, 224, 224, 3)]     0
resnet50 (Functional)        (None, 7, 7, 2048)        23587712
concatenate (Concatenate)    (None, 7, 7, 6144)        0
global_average_pooling2d     (None, 6144)              0
GlobalAveragePooling2D)
batch_normalization (Batch   (None, 6144)              24576
Normalization)
Dense (Dense)                (None, 128)               786560
Dropout (Dropout)       (None, 128)                  0
dense_1 (Dense)              (None, 3)                 387
=================================================================
Total params: 24399235 (93.08 MB)
Trainable params: 799235 (3.05 MB)
Non-trainable params: 23600000 (90.03 MB)
```

Gambar 36. *Model Summary (Resnet50)*

Ringkasan dari model yang telah dibangun mencakup detail dari setiap lapisan dalam model, termasuk *output shape* serta jumlah parameter yang terlibat dalam proses pelatihan untuk setiap lapisannya. Berikut adalah hasil ringkasan model menggunakan *Resnet50*:

Tabel 8. Deskripsi Model *Summary* (*Resnet50*)

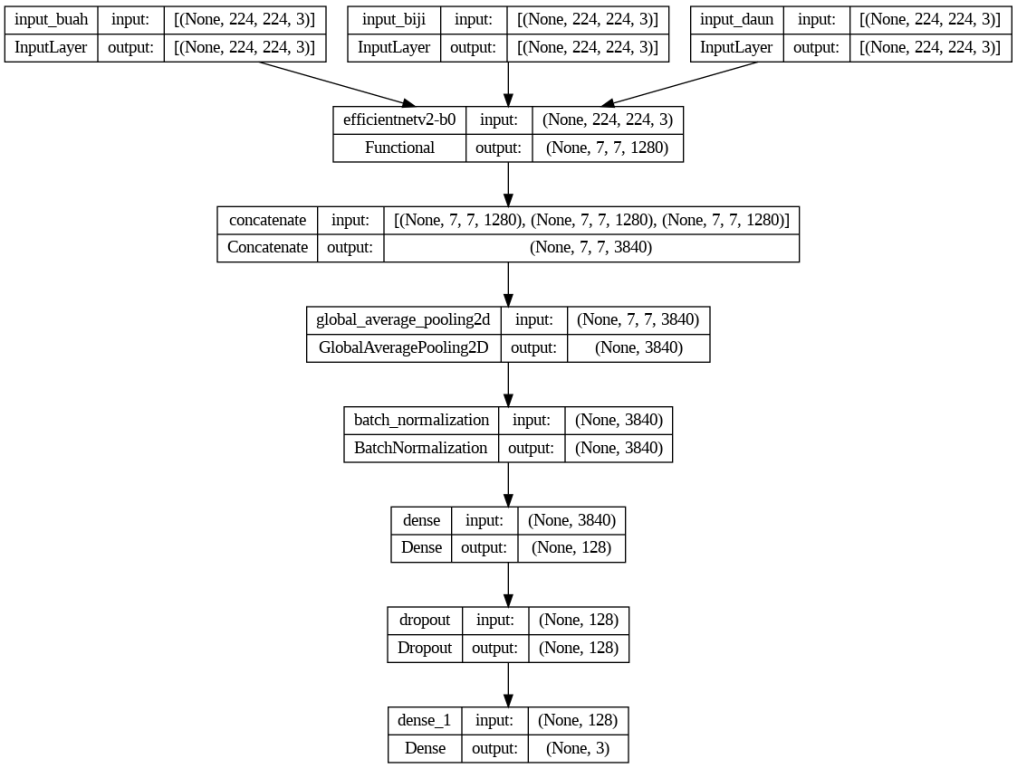
Layer (type)	**Deskripsi**	
	Output shape	**Parameter**
Input_buah	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
Input_biji	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
Input_daun	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
Resnet50	(None, 7, 7, 2048)	*Layer* ini adalah model *pre-trained Resnet50* yang digunakan sebagai ekstraktor fitur. Model ini menghasilkan output dengan ukuran 7x7 piksel dan 2048 saluran. Jumlah parameter yang besar (23587712) mencakup bobot dan bias yang telah dilatih sebelumnya untuk menghitung output.
Concatenate	(None, 7,7, 6144)	*Layer* ini menggabungkan output dari beberapa *layer* sebelumnya. Dalam hal ini, menggabungkan tiga keluaran dari *Resnet50,* masing-

Layer (type)	**Deskripsi**	
	Output shape	**Parameter**
		masing dengan 2048 saluran, menjadi satu tensor dengan 6144 saluran. *Layer* ini tidak memiliki parameter yang dapat dilatih.
Global_average_ pooling2d	(None, 6144)	*Layer* ini melakukan pooling global rata-rata pada tensor 7x7, mereduksi tiap saluran menjadi satu nilai. Outputnya adalah vektor dengan 6144 elemen, tanpa parameter yang dapat dilatih.
Batch_Normalization	(None, 6144)	*Layer* ini menormalkan aktivasi dari *layer* sebelumnya, membantu mempercepat pelatihan dan stabilitas. Jumlah parameter 24576 mencakup bobot dan bias yang telah dilatih sebelumnya.
Dense	(None, 128)	Jumlah parameter 786560 mencakup bobot dan bias yang telah dilatih sebelumnya untuk menghitung input dan output dengan 128 neuron.
Dropout	(None, 128)	*Layer* ini digunakan untuk mencegah overfitting dengan mengabaikan unit-unit tertentu secara acak selama pelatihan. *Layer* ini tidak memiliki parameter yang dapat dilatih.
Dense _1	(None, 3)	*Layer* fully connected terakhir dengan 3 unit, sesuai dengan jumlah kelas output. Jumlah parameternya terdiri dari 128 input x 3 unit = 384 bobot, ditambah 3 bias, totalnya 387..

Dari total parameter sebanyak 24.399.235, sebanyak 799.235 parameter adalah yang dapat dilatih selama proses pelatihan model. Parameter ini termasuk bobot dan bias yang disesuaikan oleh algoritma pembelajaran mesin untuk meminimalkan kesalahan prediksi. Sementara itu, sebanyak 23.600.000 parameter lainnya adalah parameter yang tidak dilatih. Parameter ini termasuk parameter dari model *pre-trained* seperti *Resnet50*, yang telah dilatih pada dataset besar seperti *ImageNet*, serta parameter statistik dari lapisan *Batch Normalization*. Meskipun parameter non-*train*able ini tidak diubah selama pelatihan model, mereka tetap penting untuk menyediakan informasi yang diperlukan bagi model dalam melakukan inferensi yang akurat. Dengan demikian, walaupun mayoritas parameter dalam model tidak dapat diubah selama pelatihan, mereka tetap berperan penting dalam memastikan kinerja model yang optimal.

EfficientNetV2B0

Berikut adalah ringkasan model dan representasi visual dari arsitektur model yang menggunakan *pre-trained* model *EfficientNetV2B0*:



Gambar 37. Representasi visual arsitektur model *EfficientNetV2B0*

```
Layer (type)                 Output Shape              Param #
=================================================================
input_buah (InputLayer)      [(None, 224, 224, 3)]     0
input_biji (InputLayer)      [(None, 224, 224, 3)]     0
input_daun (InputLayer)      [(None, 224, 224, 3)]     0
Efficientnetv2-b0            (None, 7, 7, 1280)        5919312
(Functional)
Concatenate (Concatenate)    (None, 7, 7, 3840)        0
global_average_pooling2d     (None, 3840)              0
(GlobalAveragePooling2D)
batch_normalization_3 (Batch (None, 3840)              15360
Normalization)
Dense (Dense)                (None, 128)               491648
Dropout (Dropout)            (None, 128)               0
dense_1 (Dense)              (None, 3)                 387
=================================================================
Total params: 6426707 (24.52 MB)
Trainable params: 499715 (1.91 MB)
Non-trainable params: 5926992 (22.61 MB)
```

Gambar 38. Model *Summary* (*EfficientNetV2B0*)

Ringkasan dari model yang telah dibangun mencakup detail dari setiap lapisan dalam model, termasuk *output shape* serta jumlah parameter yang terlibat dalam proses pelatihan untuk setiap lapisannya. Berikut adalah hasil ringkasan model menggunakan *EfficientNetV2B0*:

Tabel 9. Deskripsi Model *Summary* (*EfficientNetV2B0*)

Layer (type)	**Deskripsi**	
	Output shape	**Parameter**
Input_buah	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
Input_biji	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
Input_daun	(None, 224, 224, 3)	Nilai 0 pada parameter ini menunjukkan bahwa *layer* input tidak memiliki parameter yang dapat dilatih, karena hanya bertugas untuk menerima data input.
EfficientNetV2B0	(None, 7, 7, 1280)	*Layer* ini adalah model *pre-trained EfficientNetV2B0* yang digunakan sebagai ekstraktor fitur. Model ini menghasilkan output dengan ukuran 7x7 piksel dan 1280 saluran. Jumlah parameter yang besar (5919312) mencakup bobot dan bias yang telah dilatih sebelumnya untuk menghitung output.
*Concatenate*_4	(None, 7,7, 3840)	*Layer* ini menggabungkan output dari beberapa *layer* sebelumnya. Dalam hal ini, menggabungkan tiga keluaran dari *EfficientNetV2B0*, masing-masing dengan 1280 saluran, menjadi satu tensor dengan 3840 saluran. *Layer* ini tidak memiliki parameter yang dapat dilatih.
Global_average_ pooling2d_1	(None, 3840)	*Layer* ini melakukan pooling global rata-rata pada tensor 7x7, mereduksi tiap saluran menjadi satu nilai. Outputnya adalah vektor dengan 3840 elemen, tanpa parameter yang dapat dilatih.
Batch_Normalization	(None, 3840)	*Layer* ini menormalkan aktivasi dari *layer* sebelumnya, membantu mempercepat pelatihan dan stabilitas. Jumlah parameter 15360 mencakup bobot dan bias yang telah dilatih sebelumnya.
Dense	(None, 128)	Jumlah parameter 491648 mencakup bobot dan bias yang telah dilatih sebelumnya untuk menghitung input dan output dengan 128 neuron.
Dropout	(None, 128)	*Layer* ini digunakan untuk mencegah overfitting dengan

Layer (type)	**Deskripsi**	
	Output shape	**Parameter**
		mengabaikan unit-unit tertentu secara acak selama pelatihan. *Layer* ini tidak memiliki parameter yang dapat dilatih.
Dense _1	(None, 3)	*Layer* fully connected terakhir dengan 3 unit, sesuai dengan jumlah kelas output. Jumlah parameternya terdiri dari 128 input x 3 unit = 384 bobot, ditambah 3 bias, totalnya 387.

Dari total parameter sebanyak 6.426.707, sebanyak 499.715 parameter adalah yang dapat dilatih selama proses pelatihan model. Parameter ini termasuk bobot dan bias yang disesuaikan oleh algoritma pembelajaran mesin untuk meminimalkan kesalahan prediksi. Sementara itu, sebanyak 5.926.992 parameter lainnya adalah parameter yang tidak dilatih. Parameter ini termasuk parameter dari model *pre-trained* seperti *EfficientNetV2B0*, yang telah dilatih pada dataset besar seperti *ImageNet*, serta parameter statistik dari lapisan *Batch Normalization*. Meskipun parameter non-*train*able ini tidak diubah selama pelatihan model, mereka tetap penting untuk menyediakan informasi yang diperlukan bagi model dalam melakukan inferensi yang akurat. Dengan demikian, walaupun mayoritas parameter dalam model tidak dapat diubah selama pelatihan, mereka tetap berperan penting dalam memastikan kinerja model yang optimal.

Arsitektur pada masing-maing model dibangun dengan hyperparameter sebagai berikut, proses trial dan eror pada nilai-nilai hyperparameter berkontribusi pada penemuan arsitektur dengan tingkat akurasi yang optimal. Berikut merupakan konfigurasi hyperparameter yang digunakan pada model.

Tabel 10. *Hyperparameter* Model

No	**Hyperparameter**	**Nilai**
1	*Optimizer*	Adam
2	*Learning Rate*	0.001
3	*Loss Function*	*Categorical Cross Entropy*
4	*Epoch*	10, 20,30
5	*Batch Size*	64
6	*Learning Rate minimum*	0.00005

Selama pelatihan, *callback ReduceLR0nPlateu* digunakan untuk mengontrol *learning rate* dalam model. Metrik 'val_loss' dipantau untuk menentukan apakah *learning rate* perlu disesuaikan. *Monitoring val_loss* memungkinkan evaluasi kinerja model terhadap data validasi, yang merupakan indikator penting dari kemampuan generalisasi model terhadap data baru. Ketika nilai 'val_loss' tidak menunjukkan peningkatan dalam beberapa *epoch* berturut-

turut, yang diatur oleh parameter *'patience'*, penyesuaian terhadap learning rate diperlukan. Dalam kasus ini, *'patience'* diatur menjadi 6, sehingga model menunggu selama 6 *epoch* tanpa adanya peningkatan dalam val_loss sebelum learning rate dikurangi.

Penurunan learning rate bertujuan untuk membantu model mencapai konvergensi yang lebih baik dengan memperlambat laju pembelajaran saat sudah mendekati minimum lokal. *Learning rate* minimum diatur ke 0.00005 untuk memastikan bahwa proses pembelajaran tidak melambat lebih dari nilai tersebut, sehingga model tetap dapat mengakses informasi yang relevan dari data pelatihan tanpa risiko konvergensi yang terlalu lambat atau terjebak dalam minimum lokal yang tidak optimal. Hasil konfigurasi ini disimpan dalam objek *ReduceLROnPlateau* di dalam model, yang memungkinkan pengaturan dan penggunaan kembali selama pelatihan model.

```
from tensorflow.keras.callbacks import ReduceLROnPlateau
# Compile the model
learning_rate = 0.001
optimizer = Adam(learning_rate=learning_rate)
model.compile(optimizer=optimizer, loss='categorical_crossentropy',
metrics=['accuracy'])
# Define the callback for reducing learning rate on plateau
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.6,
patience=6, min_lr=0.00005)
```

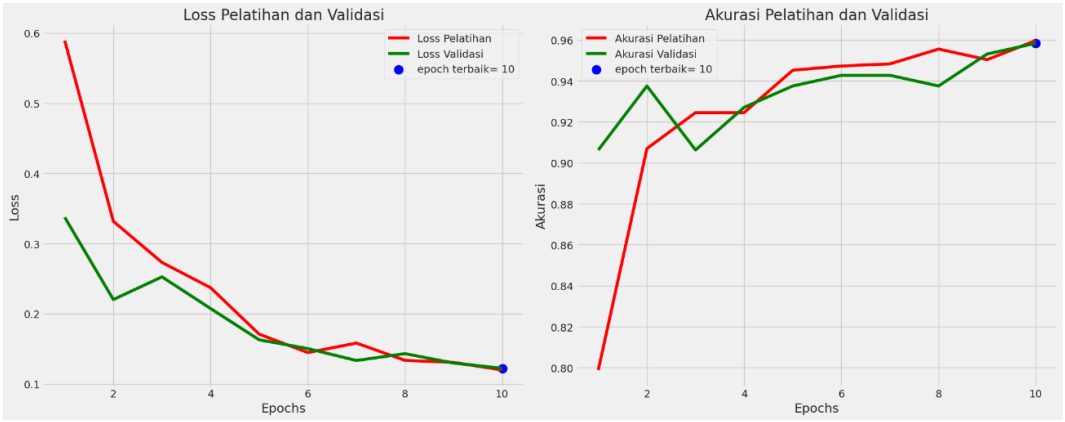
Gambar 39. Kode program untuk konfigurasi *hyperparameter*

4.4 Evaluasi dan Perbandingan Model

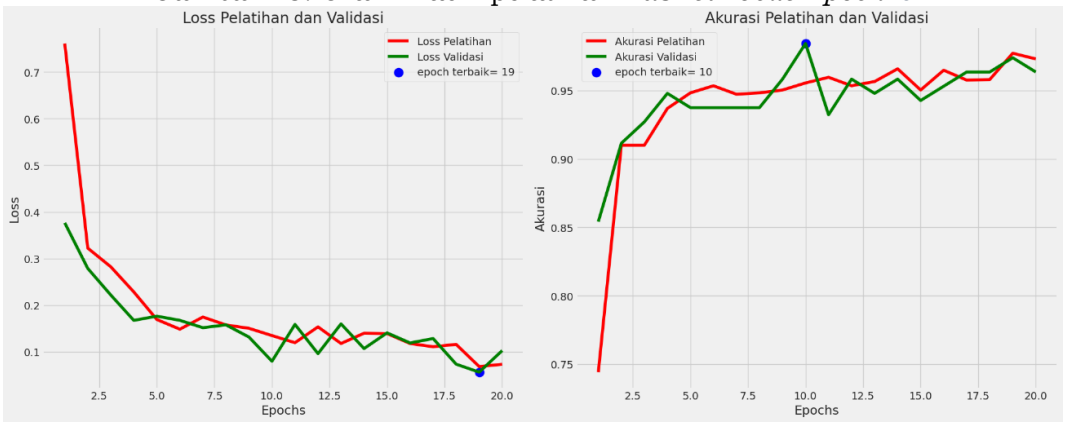
Setelah menyelesaikan penyusunan arsitektur model dan melakukan pengujian terhadap hyperparameter untuk menemukan model yang optimal, langkah berikutnya adalah menjalankan proses pelatihan pada ketiga model tersebut. Penelitian ini menggunakan hyparameter *epoch* sebagai acuan dalam perbandingan, dimana perbandingan dilakukan dengan cara memanipulasi jumlah *epoch* pada setiap arsitektur, masing-masing model dijalankan dengan *epoch* 10, 15, dan 30. Tujuan dilakukannya perbandingan jumlah *epoch* adalah agar dapat mengevaluasi stabilitas dan konsistensi pada model sehingga dapat mengetahui bagaimana *epoch* dapat mempengaruhi kinerja dan tingkat pembelajaran pada model. Pemilihan jumlah epoch ini mengacu pada penelitian sebelumnya yang menggunakan epoch 10, 20, dan 30 pada model pretrain NasNetMobile, Resnet50, dan EfficientNetV2B0, yang menunjukkan akurasi yang memuaskan (Ariawan et al., 2022; Putra et al., 2023;R. P. S. Putra et al., 2024). Pendekatan ini mempertimbangkan temuan dari studi-studi tersebut.

Model dengan masing-masing arsitektur diperlakukan sama yaitu dengan menggunakan batch size, optimizer, learning rate, dan loss function yang sama. Tujuan dari dilakukannya manipulasi *epoch* ini adalah untuk mengetahui apakah *epoch* dapat mempengaruhi akurasi dan loss dari model. Hasil dari pelatihan ini digunakan untuk pengujian lebih lanjut. Berikut adalah perbandingan grafik hasil pelatihan.

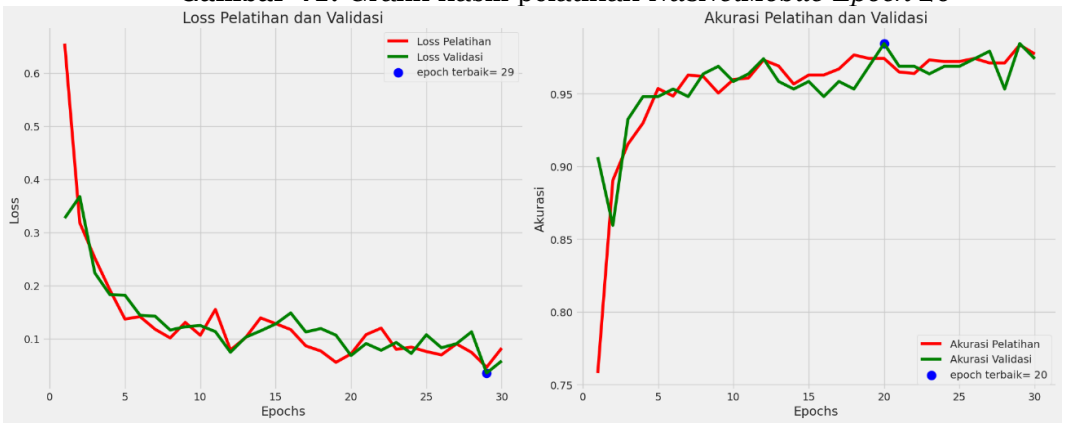
NasNetMobile



Gambar 40. Grafik hasill pelatihan *NasNetMobile Epoch 10*



Gambar 41. Grafik hasill pelatihan *NasNetMobile Epoch* 20

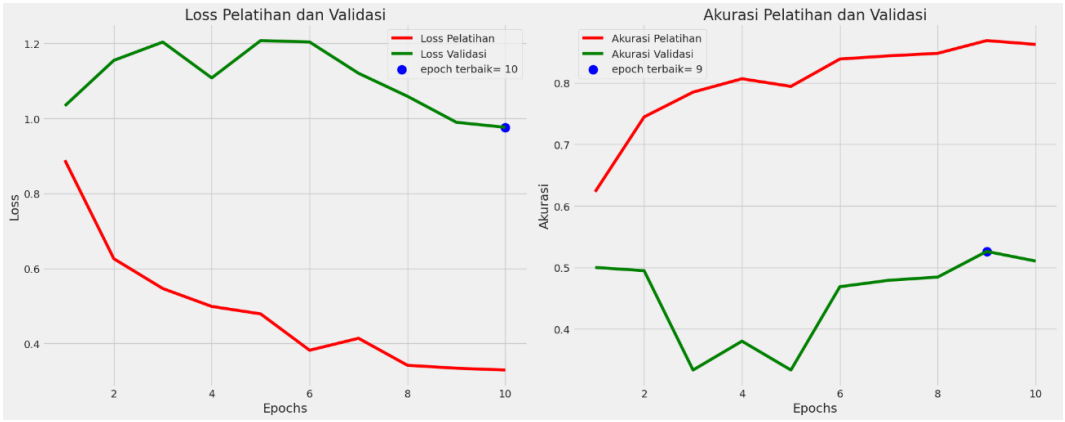


Gambar 42. Grafik hasill pelatihan *NasNetMobile Epoch* 30

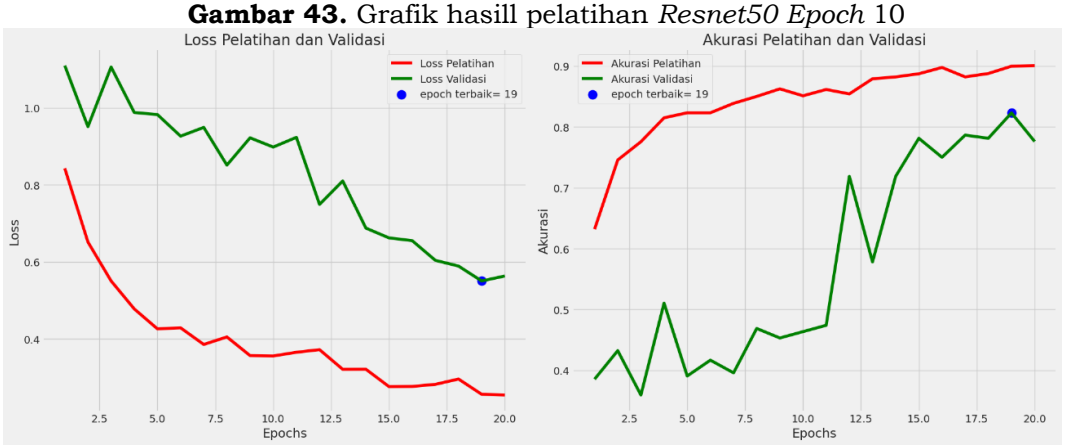
Dari Grafik pelatihan pada data latih terlihat bahwa dari segi kestabilan, grafik terbaik diperoleh oleh model *NasNetMobile* dengan *Epoch* 30 dengan

akurasi 97.72%, sementara model dengan *Epoch* 20 menunjukkan kestabilan yang sama yaitu 97.31%, diikuti oleh *epoch* 10 dengan akurasi 95.97%. sementara Pada data validasi, grafik paling stabil diperoleh oleh model *NasNetMobile* dengan *epoch* 30 dengan akurasi 96.88%, diikuti oleh *Epoch* 20 dengan akurasi 96.35% dan *Epoch* 10 di akurasi 95.97%. Perbandingan visualisasi akurasi pelatihan dapat dilihat pada Gambar 40.

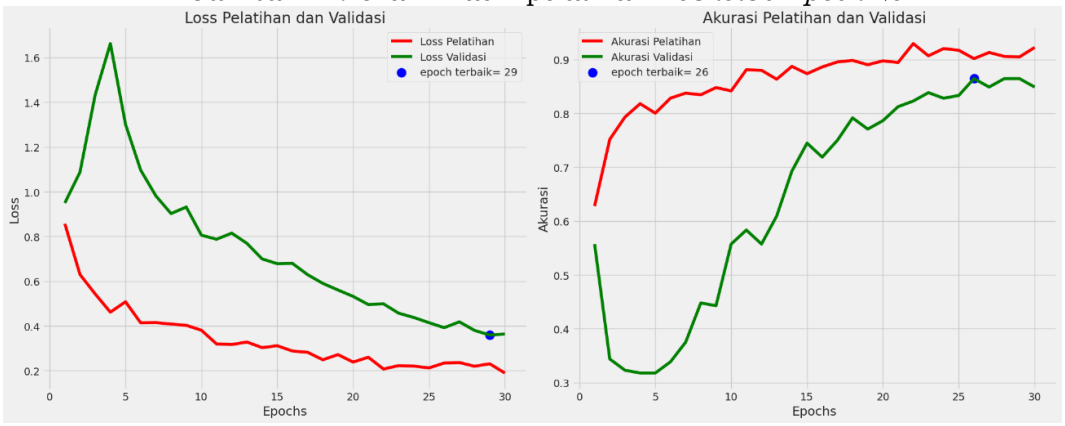
Resnet50



Gambar 43. Grafik hasill pelatihan *Resnet50 Epoch* 10



Gambar 44. Grafik hasill pelatihan *Resnet50 Epoch* 20

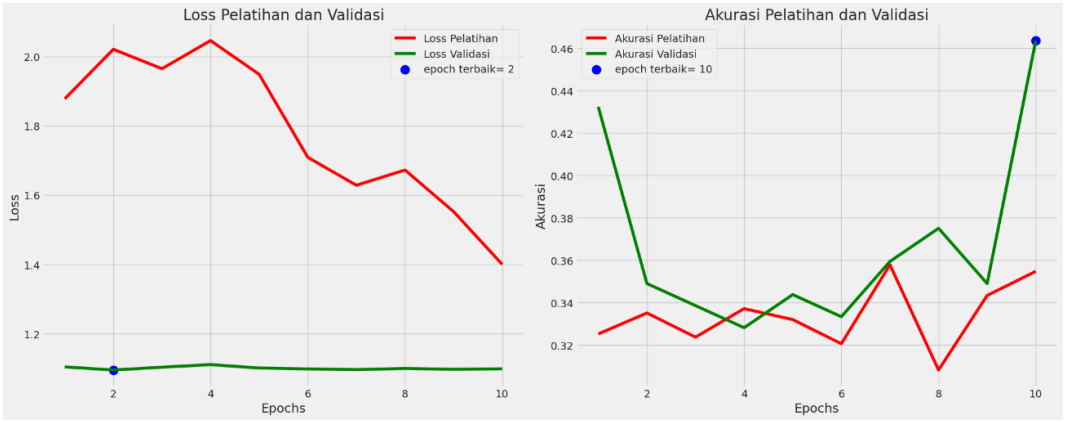


Gambar 45. Grafik hasill pelatihan *Resnet50 Epoch* 30

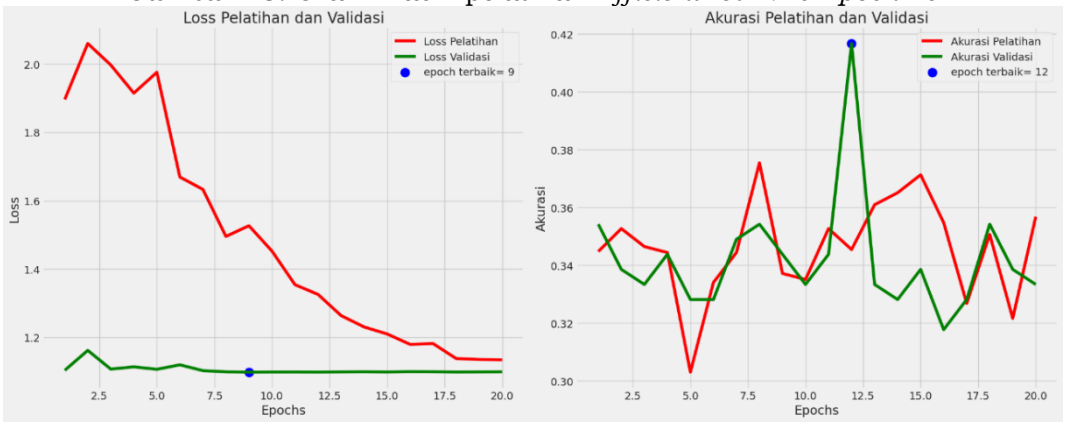
Dari Grafik pelatihan pada data latih terlihat bahwa dari segi kestabilan, grafik terbaik diperoleh oleh model *Resnet50* dengan *epoch* 30 dengan akurasi

mencapai 92.24%, sementara model dengan *epoch* 20 mencapai akuarasi 90.07% dan diikuti oleh model dengan *epoch* 10 hanya mencapai akurasi. 86.25%. sementara Pada data validasi, grafik paling stabil diperoleh oleh model *Resnet50* dengan *epoch* 30 dengan akurasi 84.90%, diikuti oleh *Epoch* 20 dengan akurasi 77.60% dan *Epoch* 10 di akurasi 51.04%. Perbandingan visualisasi akurasi pelatihan dapat dilihat pada Gambar 40.

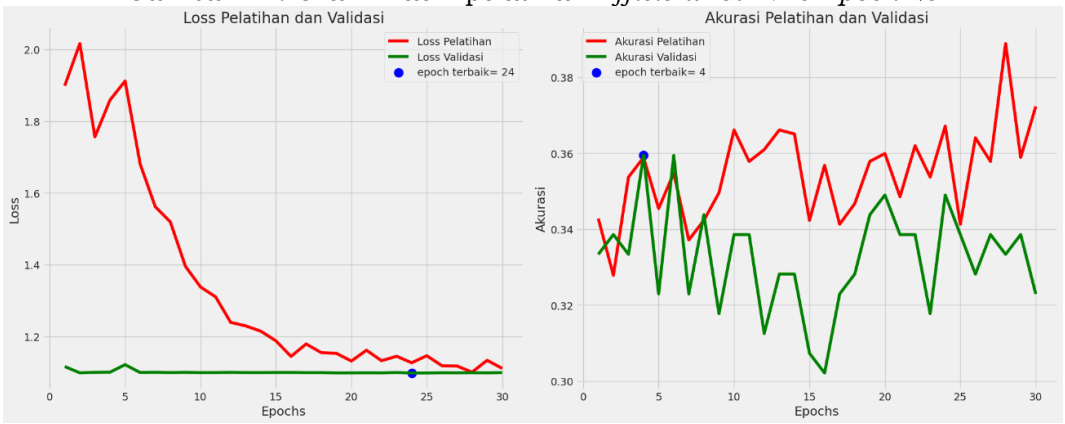
EfficientNetV2B0



Gambar 46. Grafik hasill pelatihan *EfficientNetV2B0 Epoch* 10



Gambar 47. Grafik hasill pelatihan *EfficientNetV2B0 Epoch* 20



Gambar 48. Grafik hasill pelatihan *EfficientNetV2B0 Epoch* 30

Dari Grafik pelatihan pada data latih terlihat bahwa dari segi kestabilan, grafik terbaik diperoleh oleh model *EfficientNetV2B0* dengan *epoch* 30 dengan

akurasi mencapai 37.23%, sementara model dengan *epoch* 20 mencapai akuarasi 35.68% dan diikuti oleh model dengan *epoch* 10 hanya mencapai akurasi. 35.47%. Sementara pada data validasi, grafik paling stabil diperoleh oleh model *EfficientNetV2B0* dengan *epoch* 10 dengan akurasi 46.35%, diikuti oleh *Epoch* 20 dengan akurasi 33.33% dan *Epoch* 10 di akurasi 32.29%. Perbandingan visualisasi akurasi pelatihan dapat dilihat pada Gambar 40.

Dari tabel tersebut, model dengan jumlah *epoch* pelatihan terkecil yang menunjukkan bahwa hanya diperlukan 10 *epoch* pelatihan untuk mendapatkan hasil terbaik adalah model dengan transfer learning EfficientNetV2-B0. Selanjutnya, nilai akurasi pelatihan dan validasi tertinggi diperoleh dari model dengan transfer learning *NasNetMobile*, dengan masing-masing nilai akurasi pelatihan dan validasi sebesar 97.72% dan 97.40%. Meskipun nilai *validation* loss terendah diperoleh dari model dengan transfer learning *NasNetMobile*, yaitu 5.83%, nilai *train*ing loss terendah juga diperoleh oleh model dengan transfer learning *NasNetMobile*, yaitu 7.33% pada 20 *epoch*.

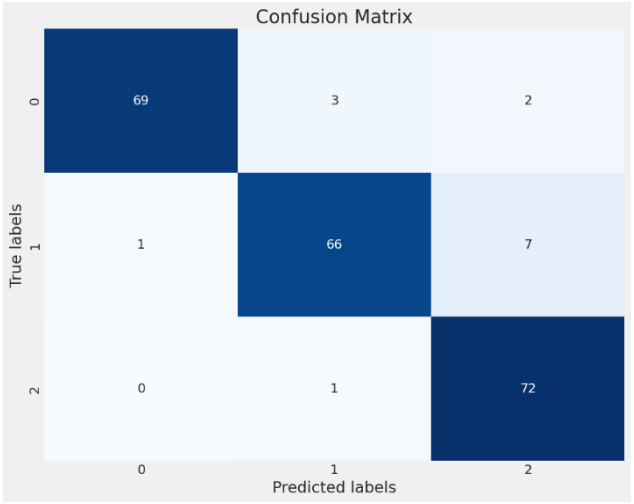
Tabel 11. Hasil akhir pelatihan ketiga model berdasarkan *epoch*

***Pre-trained* model**	***Epoch***	***Train***		***Validation***	
		Acc	***Loss***	***Acc***	***Loss***
NasNetMobile	10	95.97%	11.96%	95.83%	12.21%
	20	97.31%	7.33%	96.35%	10.26%
	30	97.72%	8.24%	97.40	5.83
Resnet50	10	86.25%	32.91%	51.04%	97.63%
	20	90.07%	25.43%	77.60%	56.33%
	30	92.24%	18.93%	84.90%	36.40%
EfficientNetV2-B0	10	35.47%	140.01%	46.38%	109.86%
	20	35.68%	113.41%	33.33%	109.87%
	30	37.23%	111.07%	32.29%	109.87%

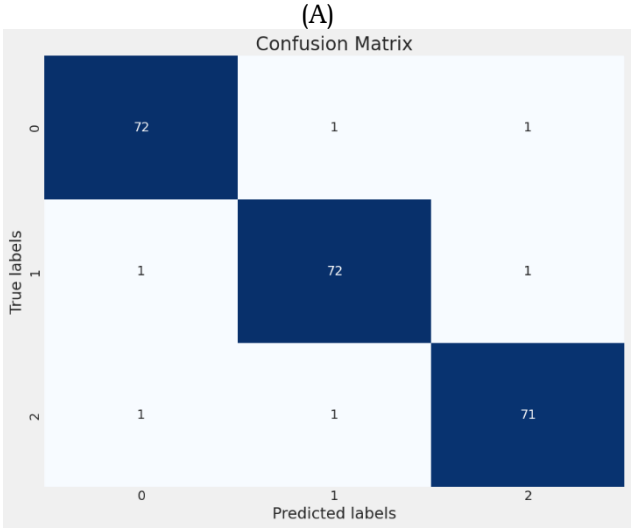
Setelah dilakukan pembangunan model CNN pada masing-masing *base model* (*NasNetMobile, Resnet50*, *EfficientNetV2B0*) pada *epoch* 10, 20, dan 30. Didapatkan model terbaik dari masing masing arsitektur adalah model *NasNetMobile* dengan *epoch* 30, *Resnet50* dengan *epoch* 30 dan *EfficientNetV2B0* dengan *epoch* 10. Langkah selanjutnya adalah evaluasi model tersebut menggunakan *Confusion matrix* dan *classification report*. *Confusion matrix* digunakan untuk menampilkan jumlah prediksi yang benar dan salah untuk setiap kelas. Sementara itu, *classification report* digunakan untuk menghasilkan laporan klasifikasi yang mencakup perhitungan akurasi, *presisi, recall,* dan skor F1. Hasil evaluasi dari masing-masing model ditunjukkan pada *confusion matrix* dan *classification report* dari ketiga model tersebut.

Confusion Matrix

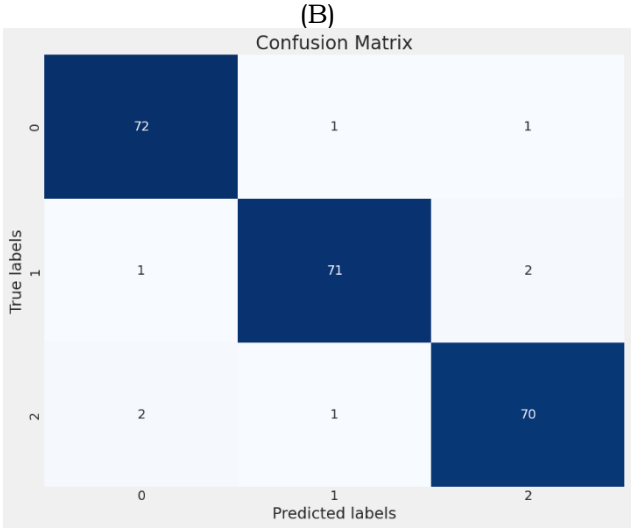
a. NasNetMobile



(A)



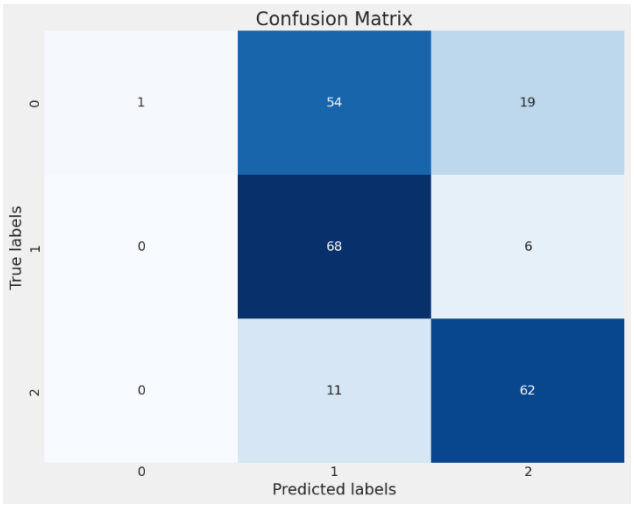
(B)



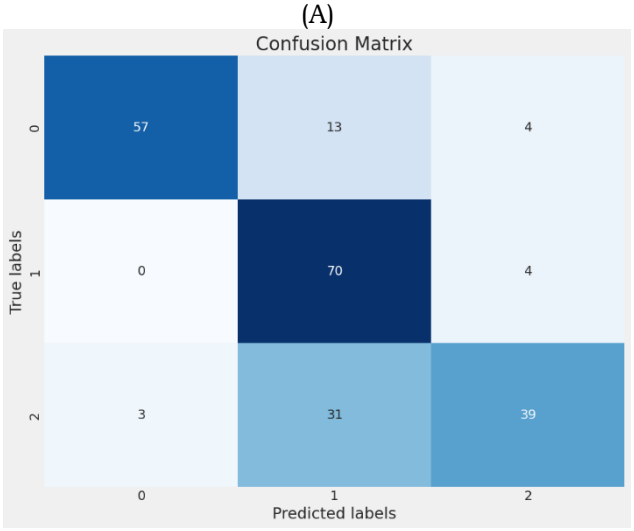
(C)

Gambar 49. *Confusion matrix NasNetMobile (A) Epoch* 10, *(B) Epoch* 20, *(c) Epoch* 30

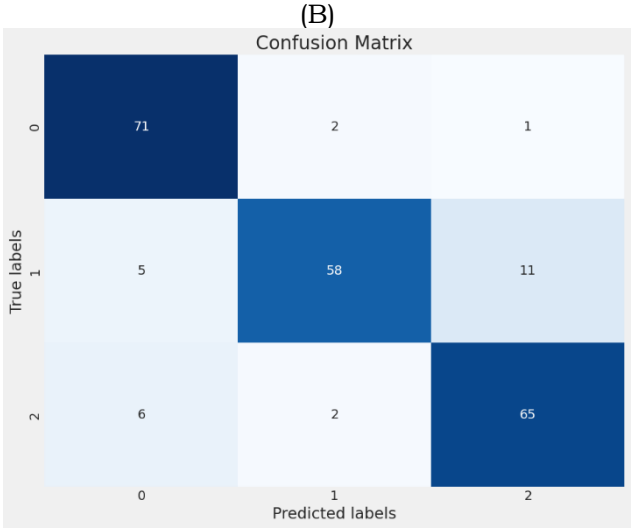
b. Resnet50



(A)



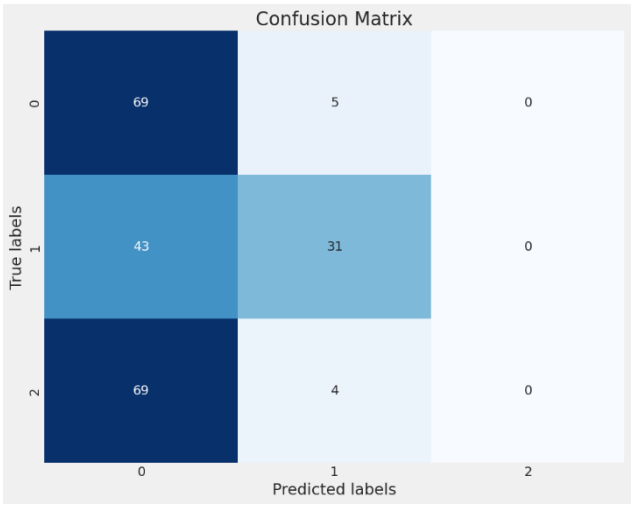
(B)



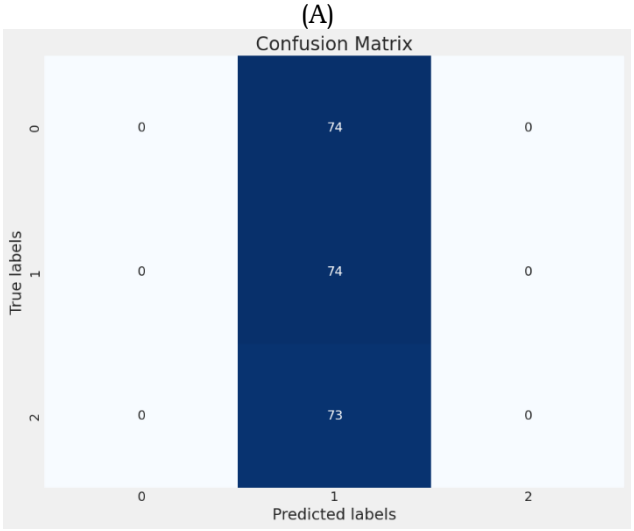
(C)

Gambar 50. *Confusion matrix Resnet50 (A) Epoch* 10, *(B) Epoch* 20, *(c) Epoch* 30

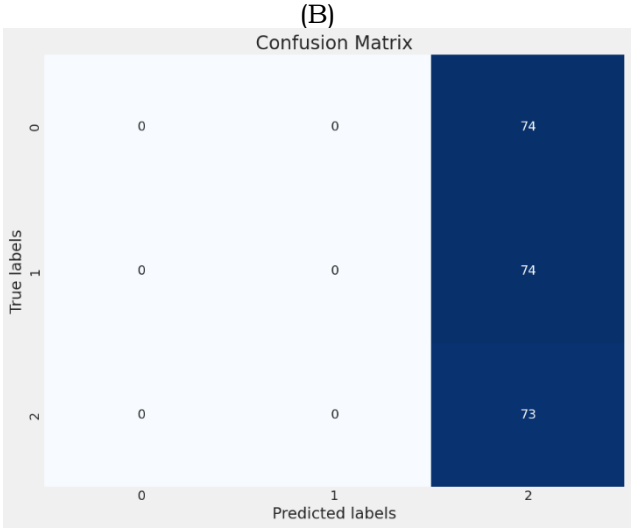
c. *EfficientNetV2*-B0



(A)



(B)



(C)

Gambar 49. *Confusion matrix EfficientNetV2-B0 (A) Epoch* 10, *(B) Epoch* 20, *(c) Epoch* 30

Setiap model diuji menggunakan 221 data. Berdasarkan hasil pengujian pada *heatmap Confusion matrix* dari ketiga model, diperoleh nilai matriks perbandingan true positive pada setiap kelas dari model yang diuji sebagai berikut:

Tabel 12. Perbandingan Nilai *True Positive*

Pre-trained Model	***Epoch***	**Data yang Dipediksi Benar (Bernilai *True Positive*)**			**Total**
		Arabika	***Liberika***	***Robusta***	
NasNetMobile	30	72	71	70	213
Resnet50	30	71	58	65	194
EfficienttV2NeB0	10	69	31	0	100

Classification Report

a. NasNetMobile

```
Classification Report:
----------------------
              precision    recall  f1-score   support

     ARABIKA       0.99      0.93      0.96        74
    LIBERIKA       0.94      0.89      0.92        74
     ROBUSTA       0.89      0.99      0.94        73

    accuracy                           0.94       221
   macro avg       0.94      0.94      0.94       221
weighted avg       0.94      0.94      0.94       221
```

(A)

```
Classification Report:
----------------------
              precision    recall  f1-score   support

     ARABIKA       0.97      0.97      0.97        74
    LIBERIKA       0.97      0.97      0.97        74
     ROBUSTA       0.97      0.97      0.97        73

    accuracy                           0.97       221
   macro avg       0.97      0.97      0.97       221
weighted avg       0.97      0.97      0.97       221
```

(B)

```
Classification Report:
----------------------
              precision    recall  f1-score   support

     ARABIKA       0.96      0.97      0.97        74
    LIBERIKA       0.97      0.96      0.97        74
     ROBUSTA       0.96      0.96      0.96        73

    accuracy                           0.96       221
   macro avg       0.96      0.96      0.96       221
weighted avg       0.96      0.96      0.96       221
```

(C)

Gambar 50. *Classification Report NasNetMobile (A) Epoch 10, (B) Epoch 20, (c) Epoch 30*

b. Resnet50

Classification Report:

	precision	recall	f1-score	support
ARABIKA	1.00	0.01	0.03	74
LIBERIKA	0.51	0.92	0.66	74
ROBUSTA	0.71	0.85	0.77	73
accuracy			0.59	221
macro avg	0.74	0.59	0.49	221
weighted avg	0.74	0.59	0.48	221

(A)

Classification Report:

	precision	recall	f1-score	support
ARABIKA	0.95	0.77	0.85	74
LIBERIKA	0.61	0.95	0.74	74
ROBUSTA	0.83	0.53	0.65	73
accuracy			0.75	221
macro avg	0.80	0.75	0.75	221
weighted avg	0.80	0.75	0.75	221

(B)

Classification Report:

	precision	recall	f1-score	support
ARABIKA	0.87	0.96	0.91	74
LIBERIKA	0.94	0.78	0.85	74
ROBUSTA	0.84	0.89	0.87	73
accuracy			0.88	221
macro avg	0.88	0.88	0.88	221
weighted avg	0.88	0.88	0.88	221

(C)

Gambar 51. *Classification Report ResNet50 (A) Epoch 10, (B) Epoch 20, (c) Epoch 30*

c. EfficientNetV2B0

Classification Report:

	precision	recall	f1-score	support
ARABIKA	0.38	0.93	0.54	74
LIBERIKA	0.78	0.42	0.54	74
ROBUSTA	0.00	0.00	0.00	73
accuracy			0.45	221
macro avg	0.39	0.45	0.36	221
weighted avg	0.39	0.45	0.36	221

(A)

Classification Report:

	precision	recall	f1-score	support
ARABIKA	0.00	0.00	0.00	74
LIBERIKA	0.33	1.00	0.50	74
ROBUSTA	0.00	0.00	0.00	73
accuracy			0.33	221
macro avg	0.11	0.33	0.17	221
weighted avg	0.11	0.33	0.17	221

(B)

Classification Report:

	precision	recall	f1-score	support
ARABIKA	0.00	0.00	0.00	74
LIBERIKA	0.00	0.00	0.00	74
ROBUSTA	0.33	1.00	0.50	73
accuracy			0.33	221
macro avg	0.11	0.33	0.17	221
weighted avg	0.11	0.33	0.16	221

(C)

Gambar 52. *Classification Report EfficientNetV2B0 (A) Epoch 10, (B) Epoch 20, (c) Epoch 30*

Berdasarkan perbandingan, dapat disimpulkan bahwa model *NasNetMobile* dapat memprediksi 211 data *test* dari setiap jeni input (biji, buah, dan daun) dengan sangat baik. Jika dilihat dari perbandingan *classification report,* Model *NasNetMobile* dengan 30 *epoch* memberikan akurasi yang paling tinggi. Berdasarkan perbandingan nilai *true positive,* model *NasNetMobile* dengan 30 *epoch* lebih unggul jika dibandingkan dengan *Resnet50* dengan *epoch* 30 dan *EfficientNetV2B0* 10 *epoch.* Berikut merupakan perbandingan *testing* masing-masing model dengan dataset sebanyak 30 data yang sama dimana terdapat 10 data perkelasnya untuk setiap jenis input (biji, buah, dan daun).

Tabel 13. Perbandingan Hasil *Testing* dengan Data Baru

Pre-trained Model	***Epoch***	**Data Baru di Perdiksi Benar**			**Total**
		Arabika	**Liberika**	**Robusta**	
NasNetMobile	30	5	9	8	22
Resnet50	30	10	0	0	10
EfficienttV2NeB0	10	0	0	10	10

Dari hasil evaluasi menggunakan *Confusion matrix* dan *classification report*, model terbaik dapat disimpulkan berdasarkan kinerja keseluruhan dalam memprediksi kelas target. Berikut adalah perbandingan hasil evaluasi model dari seluruh model yang dibuat:

Tabel 14. Perbandingan Kinerja Tiga Model Terbaik Masing-Masing Arsitektur

Arsitektur	***Epoch***	***Val acc***	***Val Loss***	***Classification report***			
				Precision	***Recall***	***F1-score***	***Acc***
NasNetMobile	30	97.40%	5.83%	96%	96%	96%	96%
Resnet50	30	84.90%	36.40%	88%	88%	88%	88%
EfficientNet V2B0	10	46.38%	109.86%	39%	45%	36%	45%

Berdasarkan hasil perbandingan kinerja tiga arsitektur *NasNetMobile*, *Resnet50*, dan *EfficientNetV2B0* dengan jumlah *epoch* yang berbeda (10, 20, dan 30), dapat disimpulkan bahwa *NasNetMobile dengan* 30 *epoch* memiliki performa terbaik. *NasNetMobile* tidak hanya mencapai akurasi yang tinggi, tetapi berdasarkan grafik akurasi dan *loss* pada data validasi, model ini merupakan model dengan grafik yang paling stabil, dan model tersebut juga mencapai akurasi tertinggi pada *testing* sebesar 99% hal ini menandakan kestabilan model. Oleh karena itu, model *NasNetMobile dengan* 30 *epoch* dipilih sebagai model terbaik yang digunakan dalam *website* yang dikembangkan menggunakan *framework Flask*. Model *NasNetMobile dengan* 30 *epoch* disimpan dalam bentuk file ".h5" untuk digunakan dalam *website.* Penyimpanan file dalam bentuk ".h5" dikarenakan format .h5 memungkinkan penyimpanan model yang efisien dan mendukung penyimpanan arsitektur. Selain itu dapat memudahkan proses deployment dan integrasi ke dalam aplikasi berbasis web.

NasNetMobile menggunakan blok seluler yang terdiri dari operasi konvolusi, pooling, dan skip connection. Struktur ini membantu model untuk mengekstrak fitur secara efisien dari gambar kopi, bahkan dengan detail yang halus. Penggunaan aktivasi ReLU juga membantu model untuk fokus pada fitur yang paling relevan dalam gambar kopi, meningkatkan kemampuannya untuk membedakan antara varietas yang berbeda. Pooling global rata-rata menggabungkan informasi dari seluruh gambar kopi menjadi representasi tunggal, sehingga model dapat mengenali pola global yang membedakan varietas kopi. NasNetMobile menggunakan metode NAS untuk mencari arsitektur jaringan saraf yang optimal untuk tugas deteksi kopi. Dengan proses ini model menemukan struktur yang paling cocok untuk mengekstrak fitur dan membedakan antara varietas kopi yang berbeda, bahkan dengan variasi visual yang rendah. Dengan penyesuaian parameter seperti jumlah variasi pengulangan dan jumlah filter konvolusi, NasNetMobile dapat dioptimalkan untuk dataset gambar kopi tertentu, meningkatkan akurasi deteksi (Zoph et al., 2017).

ResNet-50, dengan arsitekturnya yang mendalam dan kompleks, menawarkan beberapa karakteristik yang mendukung akurasi tinggi dalam deteksi jenis kopi dari buah, biji, dan daun, meskipun variasi visual rendah. Arsitektur ResNet yang mendalam menjadikan model untuk menangkap hierarki fitur yang kaya dari gambar kopi, mulai dari detail halus hingga pola global. Salah satu fitur utama ResNet-50 adalah penggunaan koneksi residual yang menghubungkan input setiap blok konvolusi ke outputnya. Koneksi ini memungkinkan aliran informasi langsung dan mencegah hilangnya informasi penting selama propagasi forward, membantu mengatasi masalah vanishing gradient, dan meningkatkan kemampuan model untuk belajar lebih baik dan mencapai akurasi yang lebih tinggi, bahkan dengan dataset yang kompleks. Fungsi aktivasi ReLU pada neuron ResNet-50 membantu mencegah overfitting, yang dapat terjadi pada model yang kompleks seperti ini. Karakteristik arsitekturnya yang membantu model untuk mempelajari fitur yang kompleks dan abstrak dari gambar kopi, menghasilkan identifikasi yang andal

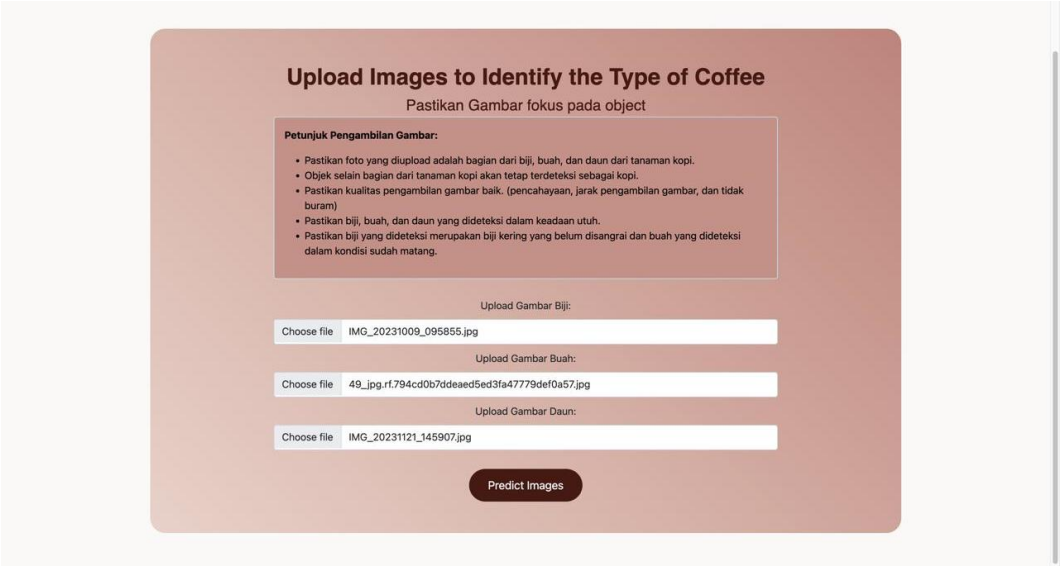
Disisi lain, EfficientNet dirancang untuk mengekstrak fitur dari data dengan variasi yang tinggi menggunakan teknik seperti "compound scaling" dan "squeeze-and-excite". Namun, pada data dengan variasi yang minim, teknik ini bisa menjadi tidak efektif sehingga model ini tidak berkinerja optimal pada dataset dengan variasi rendah (Tan & Le, 2019). Pada data kopi yang minim variasi, seperti yang dijelaskan sebelumnya, terdapat kemiripan signifikan antara biji, buah, dan daun kopi Arabika, Robusta, dan Liberika, yang dapat mempersulit model untuk mengidentifikasi fitur-fitur pembeda antar kelas. Selain itu, variasi ukuran objek yang tidak konsisten atau terlalu kecil dapat memperburuk masalah kemiripan antar kelas, sehingga model kesulitan mengenali detail penting yang membedakan jenis kopi. Oleh karena itu, penggunaan EfficientNet untuk klasifikasi jenis kopi perlu dipertimbangkan dengan seksama.

4.5 Implementasi dan Deteksi dengan *Website*

Pada tahap ini dilakukan pengujian dalam pendeteksian gambar dengan aplikasi berbasis *website* yang telah terintegrasi dengan Model. *Website* ini dikembangkan dengan menggunakan sebuah microframework *Python* yang bernama Flask. Dalam penelitian ini, Hasil deteksi kemudian ditampilkan ke pengguna dalam bentuk GUI (Graphical User Interface) yang mudah dipahami sehingga informasi dapat mudah tersampaikan. Berikut merupakan halaman utama *Website* dapat dilihat pada gambar 56.

Gambar 53. Halaman utama *website*

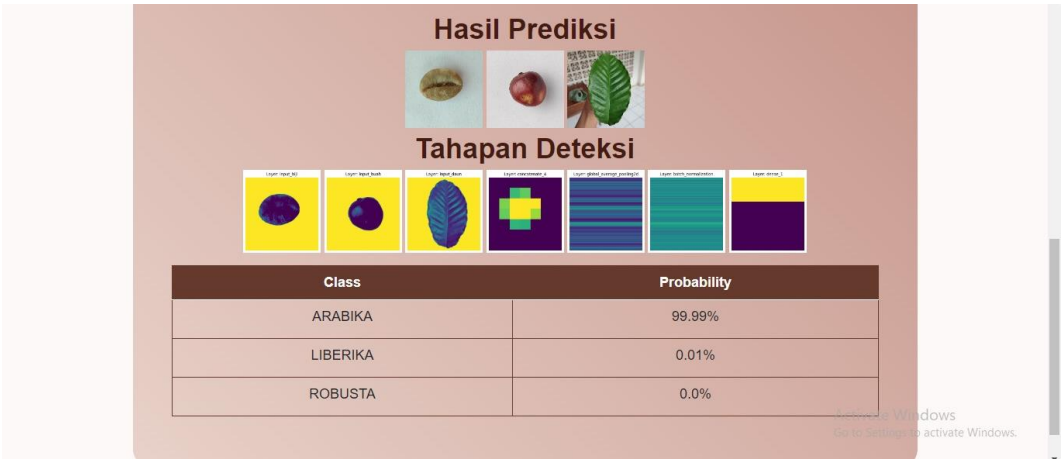
Pada halaman *website* berisi informasi berupa Logo, Nama *Website*, Animasi *Icon Coffee Recognition*, Deskripsi singkat mengenai informasi kegunaan *website*, menu halaman jenis kopi dan *button* untuk mulai menggunakan sistem. *Website* ini diberi nama *Coffe Type Identifier* sebagai identitas dari *website.* yang menggambarkan tujuan utama dari pembuatan *website* ini. Selain itu, pada halaman utama terdapat tombol Mulai Identifikasi gambar yang mengarahkan langsung ke halaman Prediksi Input Gambar. Berikut merupakan halaman Prediksi Input Gambar:



Gambar 54. Halaman Prediksi Input Gambar

Pada halaman prediksi input gambar terdapat petunjuk pengambilan gambar yang akan memberi arahan kepada *user* mengenai gambar yang akan diupload serta terdapat tombol "*choose file"* pada setiap form dari jenis gambar untuk memudahkan pengguna dapat memilih dan mengunggah gambar biji, buah, dan daun kopi yang ingin di deteksi. Setelah pengguna memilih gambar biji, buah, dan daun kopi yang ingin di deteksi, pengguna dapat menekan tombol

"*Predict Image*" untuk dapat diproses oleh model yang telah dilatih sebelumnya. Berikut merupakan halaman Hasil Prediksi:



Gambar 55. Halaman Hasil Prediksi

Halaman Hasil Prediksi menampilkan hasil deteksi pada gambar biji, buah, dan daun kopi yang telah diunggah oleh pengguna dan gambar tersebut dilakukan preprocessing. Hasil deteksi menampilkan persentase setiap kelasnya dan halaman hasil prediksi juga menyertakan visualisasi dari feature maps atau activation maps dari hasil preses prediksi gambar yang disimpan dalam bentuk gambar (*image*).

Feature map merupakan output dari proses konvolusi, yaitu proses perkalian antara nilai input dengan filter. Tujuan memvisualisasikan feature map adalah agar dapat memahami fitur-fitur dari input yang terdeteksi atau dipertahankan oleh model. Feature maps adalah gambar-gambar yang menunjukkan bagian-bagian dari gambar asli yang dianggap penting oleh model untuk membuat prediksi dan memahami bagaimana model mengenali dan mengidentifikasi berbagai objek dalam gambar. Aktivasi menunjukkan seberapa kuat fitur tertentu terdeteksi di berbagai bagian gambar, nilai tinggi dalam aktivasi berarti fitur tersebut sangat kuat terdeteksi di bagian tersebut. Nilai aktivasi ini dinormalisasi dan diubah menjadi gambar yang ditampilkan dengan warna-warna untuk memudahkan interpretasi. Umumnya, semakin dalam lapisan dalam model, semakin umum fitur yang diambil, sementara pada lapisan pertama setelah input, fitur yang diambil akan sangat detail.

Selain itu, website ini juga memberikan informasi detail mengenai masing-masing Jenis Kopi. Informasi mengenai masing-masing jenis kopi dapat dilihat dengan cara menekan masing-masing nama jenis kopi pada menu. Gambar 59 merupakan tampilan halaman detail informasi jenis kopi.

Gambar 56. Halaman Informasi Jenis Kopi

Halaman informasi Jenis berisikan tentang penjelasan mengenai karakteristik dari masing masing Jenis Kopi meliputi biji, buah dan daun kopi. selain itu, pada halaman ini juga bersi informasi mengenai aroma, rasa dari setiap jenis kopi. Informasi-informasi yang diberikan ditujukan agar pengguna dapat memahami dengan baik mengenai jenis - jenis kopi. Dengan informasi tersebut diharapkan pengguna dapat memiliki pemahaman yang mendalam tentang jenis-jenis kopi yang ada. Dengan pemahaman yang baik tentang karakteristik dan cita rasa kopi, pengguna dapat membuat pilihan yang lebih tepat sesuai dengan preferensi pribadi mereka. Hal ini juga dapat membantu pengguna dalam mengeksplorasi berbagai jenis kopi baru dan meningkatkan pengalaman mereka dalam menikmati kopi.

Website yang telah terintegrasi dengan model prediksi Jenis Kopi dengan 3 jenis input (biji, buah, dan daun) kemudian digunakan untuk menguji 30 data baru dari masing masing jenis input (biji, buah, dan daun) yang tidak termasuk dalam data yang telah digunakan untuk melatih model CNN. Hasil prediksi tersebut ditunjukkan pada Tabel dibawah ini:

Tabel 15. Hasil *Test*ing

No.	Gambar	True Label	Predict Label	Keterangan
1		Arabika	Arabika	Benar

No.	Gambar	True Label	Predict Label	Keterangan
2		Arabika	Arabika	Benar
3		Arabika	Arabika	Benar
4		Arabika	Robusta	Salah
5		Arabika	Arabika	Benar

No.	Gambar	True Label	Predict Label	Keterangan
6		Arabika	Robusta	Salah
7		Arabika	Robusta	Salah
8		Arabika	Arabika	Benar
9		Arabika	Liberika	Salah

No.	Gambar	True Label	Predict Label	Keterangan
10		Arabika	Arabika	Benar
11		Liberika	Liberika	Benar
12		Liberika	Liberika	Benar
13		Liberika	Liberika	Benar

No.	Gambar	True Label	Predict Label	Keterangan
14		Liberika	Robusta	Salah
15		Liberika	Liberika	Benar
16		Liberika	Liberika	Benar
17		Liberika	Liberika	Benar

No.	Gambar	True Label	Predict Label	Keterangan
18		Liberika	Liberika	Benar
19		Liberika	Liberika	Benar
20		Liberika	Liberika	Benar
21		Robusta	Arabika	Salah

No.	Gambar	True Label	Predict Label	Keterangan
22		Robusta	Robusta	Benar
23		Robusta	Robusta	Benar
24		Robusta	Arabika	Salah
25		Robusta	Robusta	Benar

No.	Gambar	True Label	Predict Label	Keterangan
26		Robusta	Robusta	Benar
27		Robusta	Robusta	Benar
28		Robusta	Robusta	Benar
29		Robusta	Robusta	Benar

No.	Gambar	True Label	Predict Label	Keterangan
30		Robusta	Robusta	Benar

Berdasarkan Tabel 15, model berhasil memprediksi kelas 'Liberika' dan 'Robusta' dengan sempurna, nanun kurang baik unutk kelas 'Arabika'. Terdapat perbedaan daam prediksi kelas 'Arabika' dimana hanya 6 dari 10 gambar yang dibaca dengan benar, sementara 3 gambar lainnya dibaca 'Robusta' dan 1 lain nya dibaca 'Liberika'.

Perbedaan tersebut disebabkan oleh biji, buah, dan daun Arabika yang memiliki kemiripan dengan kelas lainnya. Misalnya, biji Arabika seringkali memiliki ukuran dan bentuk yang mirip dengan biji robusta ataupun Liberika, sementara tepi daun Arabika menyerupai daun Robusta. Faktor lain yang mempengaruhi adalah ukuran objek pada gambar, dimana variasi ukuran dapat mempengaruhi cara model mengidentifikasi fitur-fitur karakteristik setiap jenis kopi. Ukuran objek yang tidak konsisten atau terlalu kecil dapat menyebabkan model kesulitan untuk mengenali detail-detail penting yang membedakan Arabika dari jenis kopi lainnya. Hal ini menunjukkan perlunya penanganan lebih lanjut seperti penyesuaian parameter model untuk meningkatkan akurasi dalam mengklasifikasikan jenis kopi Arabika.

Hasil dari pengujian menggunakan data uji baru dirangkum dalam bentuk tabel *confusion matrix* berikut:

Tabel 16. *Confusion Matrix* dari Data Uji Baru

		Kelas Sebenarnya		
		Arabika	Liberika	Robusta
Kelas Prediksi	Arabika	6	1	3
	Liberika	0	9	1
	Robusta	2	0	8

Berdasarkan tabel 16, prediksi menggunakan model dalam menguji data baru untuk jenis kopi Arabika diklasifikasi 6 dari 10 gambar yang diprediksi

benar, prediksi terhadapt jenis kopi Liberika 9 dari 10 gambar yang diprediksi benar, dan Robusta 8 dari 10 gambar yang diprediksi benar. Secara keseluruhan, model berhasil mengklasifikasikan 23 dari 30 gambar dengan benar, perhitungan akurasi matrix tersebut adalah sebagai berikut:

$$Akurasi = \frac{\text{TPall}}{TP + TN + FP + FN} = \frac{23}{30} = 0.76667$$

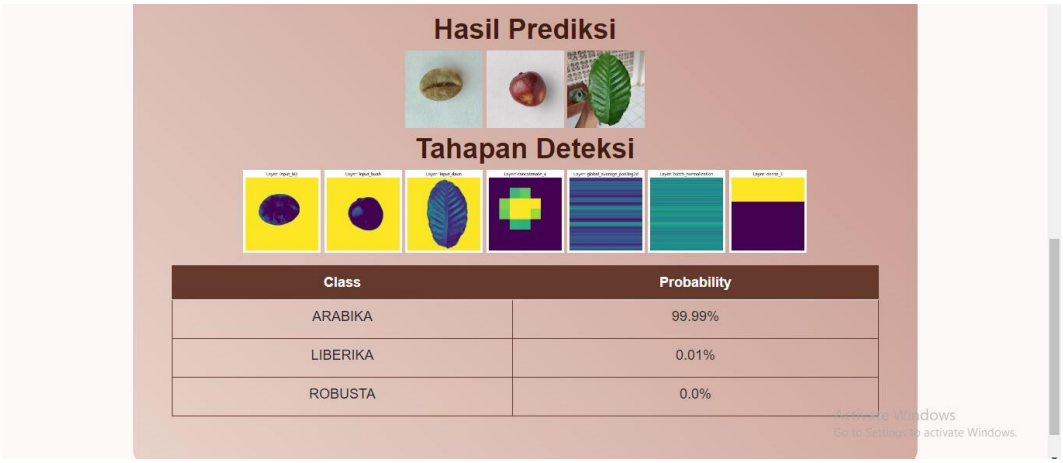
Dari perhitungan akurasi diatas didapatkan akurasi model dengan jumlah data test 10 gambar dari masing masing kelas dan jenis input yaitu sebesar 76.667%. Selain nilai akurasi dari confusion matrix, metrik evaluasi lainnya yang sering digunakan adalah *precision, recall,* dan *F1-score*. Berikut perhitungan unutk mendapatkan nilai *precision, recall,* dan *F1-score* pada salah satu kelas yaitu 'liberika':

$$Precision = \frac{TP}{TP + FP} = \frac{9}{9 + 1} = 0.9$$

$$Recall = \frac{TP}{TP + FN} = \frac{9}{9 + 1} = 0.9$$

$$F1 - score = 2 \times \frac{Recall \times Precision}{Recall + Precision} = 2 \times \frac{0.9 \times 0.9}{0.9 + 0.9} = 2 \times \frac{0.81}{1.8} = 0.9$$

Jadi untuk kelas liberika *Precision* adalah 0.9, *Recall* adalah 0.9, *F1-score* 0.9. Selain pengujian dengan kelas yang telah dilatih, website prediksi jenis kopi ketika digunakan untuk memprediksi gambar diluar kelas yang telah dilatih atau gambar random, maka model tetap akan memetakan gambar tersebut ke salah satu dari tiga kelas yang dikenalnya secara acak. Hal ini terjadi karena jaringan saraf modern, seperti CNN, seringkali tidak mampu mengenali kapan prediksinya mungkin salah karena mereka memiliki kelemahan dalam menghasilkan penilaian keyakinan yang tepat terhadap prediksi yang dibuatnya. Kelemahan ini menyebabkan jaringan tidak dapat mengidentifikasi input yang berbeda dari yang diamati selama pelatihan (DeVries & Taylor, 2018).



Gambar 57. Hasil Prediksi Gambar Diluar Kelas

Untuk mengatasi masalah ini, beberapa pendekatan yang dapat diambil termasuk menambahkan kelas "lainnya" atau "tidak dikenal" selama pelatihan, atau menggunakan teknik deteksi anomali untuk mengidentifikasi gambar yang tidak cocok dengan kelas yang telah dilatih. Selain itu, pendekatan lainnya melibatkan penekanan pada validasi input saat pengguna mengunggah gambar, memastikan bahwa gambar yang diinput sesuai dengan kelas yang telah dilatih. Dengan cara ini, pengguna dapat diberikan panduan untuk memastikan gambar yang diunggah relevan dengan konteks model, sehingga mengurangi kemungkinan kesalahan prediksi.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang dilakukan, adapun kesimpulan dari penelitian ini adalah sebagai berikut:

1. Penelitian untuk deteksi Jenis kopi ini berhasil dilakukan menggunakan metode *transfer learning* dengan tiga *pre-trained* yaitu *NasNetMobile, ResNet50,* dan *EfficientNetV2B0*. Penelitian ini dilakukan dengan cara menambahkan beberapa lapisan seperti *Global Average Pooling 2D Layer*, *Batch Normalization Layer*, *Dense Layer*, *Dropout Layer*, dan *Dense Layer* (Output *Layer*).
2. Penelitian ini melakukan perbandingan *pre-trained* dengan jumlah *epoch* sebagai pembandingnya, yaitu 10, 20, dan 30 *epoch*. Berdasarkan hasil penelitian, *NasNetMobile* dan *ResNet50* menunjukkan akurasi terbaik dan stabil pada 30 *epoch*, sementara *EfficientNetV2B0* mencapai akurasi optimalnya pada 10 *epoch*. Penelitian ini memperoleh model terbaik yaitu model menggunakan arsitektur *NasNEtMobile* dengan 30 *epoch* dengan nilai akurasi *train*= 97.72%, akurasi validasi= 97.40%, *precision* =96%, *recall= 96%*, *f1-score*= 96%, dan akurasi uji = 97% yang berhasil diimplementasikan pada *website* Identifikasi Jenis Kopi.
3. Model deteksi jenis kopi ini menghadapi tantangan dalam mengidentifikasi gambar di luar kelas yang telah dilatih, karena jaringan saraf modern seperti CNN sering kali tidak dapat menilai keyakinan prediksinya dengan tepat. Untuk mengatasi hal ini, disarankan untuk menambahkan kelas "lainnya" atau "tidak dikenal," menerapkan teknik deteksi anomali, dan menegakkan validasi input guna memastikan gambar yang diunggah relevan dengan kelas yang telah dilatih, sehingga mengurangi kemungkinan kesalahan prediksi.
4. Kesulitan pada penggunaan arsitektur NASNetMobile mencakup kompleksitas dalam penyesuaian parameter yang memerlukan waktu dan sumber daya lebih, sementara pada arsitektur ResNet50, kompleksitas model menyebabkan waktu pelatihan yang lebih lama dan membutuhkan lebih banyak sumber daya komputasi dibandingkan model yang lebih ringan, serta pada arsitektur EfficientNetV2B0, tidak optimal untuk dataset dengan variasi rendah sehingga kesulitan dalam mengenali fitur pembeda pada data yang sangat mirip.

5.2 Saran

Berdasarkan penelitian yang dilakukan, adapun saran dari penelitian ini adalah sebagai berikut:

1. Mengeksplorasi penambahan beberapa lapisan tambahan pada arsitektur model untuk meningkatkan performa dan akurasi prediksi.
2. Menambahkan beberapa *hyperparameter* sebagai perbandingan guna memperoleh arsitektur CNN yang mampu menghasilkan akurasi yang lebih baik.
3. Pada penelitian selanjutnya, diperlukan peningkatan kualitas dataset menggunakan dataset yang lebih bersih dan konsisten. Hal ini dapat dicapai dengan memperhatikan proses pengumpulan data, termasuk spesifikasi jenis kopi yang lebih terperinci. Mengingat bahwa kopi Arabika, Liberika, dan Robusta memiliki berbagai varietas.
4. Mengoptimalkan pengambilan gambar saat pengumpulan dataset harus lebih diperhatikan untuk memastikan konsistensi dan kualitas gambar termasuk penggunaan pencahayaan yang seragam, sudut pengambilan gambar yang konsisten, serta resolusi gambar yang memadai. Ukuran objek dalam gambar juga perlu dijaga konsistensinya agar model dapat mengenali fitur-fitur karakteristik dengan lebih baik.
5. Mengeksplorasi teknik regularisasi tambahan untuk lebih meningkatkan performa, seperti menguji arsitektur pretrained lain yang mungkin menawarkan representasi fitur yang lebih kuat atau lebih sesuai dengan dataset yang digunakan.

DAFTAR PUSTAKA

Aditya, I. W., Nocianitri, K. A., & Yusasrini, N. L. A. (2016). Kajian Kandungan Kafein Kopi Bubuk, Nilai Ph Dan Karakteristik Aroma Dan Rasa Seduhan Kopi Jantan (Pea Berry Coffee) Dan Betina (Flat Beans Coffee) Jenis Arabika dan Robusta. *Itepa: Jurnal Ilmu Teknologi Pangan*, *5(1)*, 1–12.

Almryad, A. S., & Kutucu, H. (2020). Automatic identification for field butterflies by convolutional neural networks. *Engineering Science and Technology, an International Journal*, *23*(1), 189–195. https://doi.org/10.1016/j.jestch.2020.01.006

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Van Esesn, B. C., Awwal, A. A. S., & Asari, V. K. (2018). The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches. *ArXiv*. http://arxiv.org/abs/1803.01164

Alwanda, M. R., Ramadhan, R. P. K., & Alamsyah, D. (2020). Implementasi Metode Convolutional Neural Network Menggunakan Arsitektur LeNet-5 untuk Pengenalan Doodle. *Jurnal Algoritme*, *1*(1), 45–56. https://doi.org/10.35957/algoritme.v1i1.434

Amin, M. F. (2023). Confusion Matrix in Three-class Classification Problems: A Step-by-Step Tutorial. *Journal of Engineering Research*, *7*(1), 0–0. https://doi.org/10.21608/erjeng.2023.296718

Andoko, E., Rafani, I., & Pambudy, R. (2023). Accelerating Coffee Productivity through Farm, Farmers, and Agribusiness Developments in Indonesia. *FFTC Agricultural Policy Platform (FFTC-AP)*. https://ap.fftc.org.tw/article/3465

Andriani, D., Sugianto, S., & Karim, A. (2022). Pemetaan Kebun Kopi Rakyat dan Potensi Pengembangannya di Kabupaten Nagan Raya (Mapping the People's Plantation and its Development Potential in Nagan Raya Districts). *Jurnal Ilmiah Mahasiswa Pertanian*, *7*(4). www.jim.unsyiah.ac.id/JFP

Ariawan, I., Arifin, W. A., Rosalia, A. A., & Tufailah, N. (2022). Klasifikasi Tiga Genus Ikan Karang Menggunakan Convolution Neural Network. J. Ilmu Dan Teknologi Kelautan Tropis, 14(2). https://doi.org/https://doi.org/10.29244/jitkt.v14i1.33633

Aryadi, M. I., Arfi, F., & Harahap, M. R. (2020). Literature Review: Perbandingan Kadar Kafein Dalam Kopi Robusta (Coffea Canephora), Kopi Arabika (Coffea Arabica) Dan Kopi Liberika (Coffea Liberica) Dengan Metode Spektrofotometri Uv-Vis. *AMINA, Ar-Raniry Chemistry Journal*, *2(2)*, 64–70. https://doi.org/10.22373/AMINA.V2I2.700

Aslam, F. A., Gulamgaus, H. N. M. J. M. M. M. M. A., & Lokhande, P. P. S. (2015). Efficient Way Of Web Development Using Python And Flask. *International Journal of Advanced Research in Computer Science*, *6*(2), 54–57. https://doi.org/https://doi.org/10.26483/ijarcs.v6i2.2434

Badan Pusat Statistik. (2023). *Statistik Indonesia 2023*. BPS Statistics Indonesia.

Cahyo, D. D. N., Fauzi, M. A., Nugroho, J. T. K., & Kusrini. (2023). Analisis Perbandingan Optimizer pada Arsitektur NASNetMobile Convolutional Neural Network untuk Klasifikasi Ras Kucing. *Jurnal Teknologi*, *15*(2), 171–177. https://doi.org/10.34151/jurtek.v15i2.4025

Cahyono, B. E., Nugroho, A. T., & Maulinida, I. W. (2023). Klasifikasi Jenis Biji Kopi dengan Menggunakan Metode Gray Level Co-occurrence Matrix (GLCM). *TEKNOTAN*, *16*(3), 191. https://doi.org/10.24198/jt.vol16n3.9

Cai, S., Shu, Y., Chen, G., Ooi, B. C., Wang, W., & Zhang, M. (2019). Effective and Efficient Dropout for Deep Convolutional Neural Networks. *ArXiv*. http://arxiv.org/abs/1904.03392

Chen, K., Wu, Y., Wang, Z., Zhang, X., Nian, F., Li, S., & Shao, X. (2020). Audio Captioning Based On Transformer and Pre-Trained Cnn. *Detection and Classification of Acoustic Scenes and Events*. https://github.com/lukewys/dcase_2020_T6.

DeVries, T., & Taylor, G. W. (2018). *Learning Confidence for Out-of-Distribution Detection in Neural Networks*. http://arxiv.org/abs/1802.04865

Diana, R., Warni, H., & Sutabri, T. (2017). Penggunaan Teknologi Machine Learning Untuk Pelayanan Monitoring Kegiatan Belajar Mengajar pada Smk Bina Sriwijaya Palembang. *JURNAL TEKNIK INFORMATIKA JUTEKIN*, *5*(1). https://doi.org/10.51530/jutekin.v11i1.709

Dishar, H. K., & Muhammed, L. A. (2023). Detection Brain Tumor Disease Using a Combination of Xception and NASNetMobile. *International Journal of Advances in Soft Computing and Its Applications*, *15*(2), 325–336. https://doi.org/10.15849/IJASCA.230720.22

Falah, H. N. Al, & Purnamasari, K. K. (2019). *Implementasi Convolutional Neural Network Pada Pengenalan Tulisan Tangan*.

FAO Organizational. (2023). *Market and Trade Coffe*. Food and Agriculture Organization of the United Nations. https://www.fao.org/markets-and-trade/commodities/coffee/en/

Farhaty, N. (2016). Tinjauan Kimia Dan Aspek Farmakologi Senyawa Asam Klorogenat Pada Biji Kopi : Review. *Farmaka Jurnal Ilmiah Farmasi Indonesia*, *14(1)*, 214–227.

Foreign Agricultural Service. (2023). *Coffee: World Markets and Trade Mexico's Production Recovering from Coffee Leaf Rust*. United States Department of Agriculture. https://public.govdelivery.com/accounts/USDAFAS/subscriber/new

Fuadi, A., & Suharso, A. (2022). Perbandingan Arsitektur Mobilenet Dan Nasnetmobile Untuk Klasifikasi Penyakit Pada Citra Daun Kentang. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *7*(3), 701–710. https://doi.org/10.29100/jipi.v7i3.3026

Haniefan, N., & Basunanda, P. (2022). Eksplorasi dan Identifikasi Tanaman Kopi Liberika di Kecamatan Sukorejo, Kabupaten Kendal. *Vegetalika*, *11*(1), 11–18. https://doi.org/10.22146/veg.44325

Hidayat, M. T., Utomo, P. E. P., & Hutabarat, B. F. (2024). Implementasi Model Terlatih CNN pada Aplikasi Android Pendeteksi Jenis Kopi dengan Input Foto Biji, Buah dan Daun. *Digital Zone: Jurnal Teknologi Informasi Dan Komunikasi*, *15(1)*. https://doi.org/https://doi.org/10.31849/digitalzone.v15i1.19563

Hidayat, R. F., Irawan, B., & Narsun, M. (2023). Implementasi Klasifikasi Gambar Untuk Industri Pakaian Menggunakan Image Search Engine Berbasis Website. *EProceedings of Engineering*, *10(1)*, 365–362.

Huda, N., Mahiruna, A., Sulistijanti, W., & Santi, R. C. N. (2023). ANALISIS PERFORMA INCEPTIONV3 CONVOLUTIONAL NETWORK PADA KLASIFIKASI VARIETAS DAUN GRAPEVINE. *Jurnal Sains Komputer Dan Teknologi Informasi*, *5*(2), 47–53. https://doi.org/10.33084/jsakti.v5i2.5022

Id, I. D. (2021). *MACHINE LEARNING: Teori, Studi Kasus dan Implementasi Menggunakan Python*.

Mubarok, H. (2019). Identifikasi Ekspresi Wajah Berbasis Citra Menggunakan Algoritma Convolutional Neural Network *(CNN)*.

Muhimmah, I., Muchlis, N. F., & Kurniawardhani, A. (2018). *Deteksi Kemerahan pada Kulit Wajah dengan Teknik Pengolahan Citra*.

Norhikmah, Lutfhi, A., & Rumini. (2022). The Effect of Layer Batch Normalization and Droupout of CNN model Performance on Facial Expression Classification. *JOIV : International Journal on Informatics Visualization*, *6*(2–2), 481. https://doi.org/10.30630/joiv.6.2-2.921

Nugroho, M. A., & Sebatubun, M. M. (2020). Klasifikasi Varietas Kopi Berdasarkan Green Bean Coffee Menggunakan Metode Machine Learning. *Journal of Information System Management (JOISM)*, *1*(2), 1–5. https://doi.org/10.24076/JOISM.2020v1i2.24

Nurfita, R. D., & Ariyanto, G. (2018). Implementasi Deep Learning berbasis Tensorflow untuk Pengenalan Sidik Jari. *Emitor: Jurnal Teknik Elektro*, *18*(1), 22–27. https://doi.org/10.23917/emitor.v18i01.6236

O'Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks. *ArXiv*. http://arxiv.org/abs/1511.08458

Pangestu, M. A., & Bunyamin, H. (2018). Analisis Performa dan Pengembangan Sistem Deteksi Ras Anjing pada Gambar dengan Menggunakan Pre-Trained CNN Model. *Jurnal Teknik Informatika Dan Sistem Informasi (JuTISI)*, *4*, 2443–2229. https://doi.org/10.28932/jutisi.v4i2.828

Panggabean, E. (2011). *Buku PiIntar Kopi*. PT. AgroMedia Pustaka.

Peryanto, A., Yudhana, A., & Umar, R. (2020). Klasifikasi Citra Menggunakan Convolutional Neural Network dan K Fold Cross Validation. *Journal of Applied Informatics and Computing (JAIC)*, *4*(1), 45. http://jurnal.polibatam.ac.id/index.php/JAIC

Putra, I. W. S. E., WIjaya, A. Y., & Soleaiman, R. (2016). Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101.

Jurnal Teknik ITS, *5*(1). https://doi.org/10.12962/j23373539.v5i1.15696

Putra, J. W. G. (2020). *Pengenalan Pembelajaran Mesin dan Deep Learning* (1.4).

Putra, M. D. A., Winanto, T. S., Hendrowati, R., Primajaya, A., & Adhinata, F. D. (2023). A Comparative Analysis of Transfer Learning Architecture Performance on Convolutional Neural Network Models with Diverse Datasets. Komputika : Jurnal Sistem Komputer, 12(1), 1–11. https://doi.org/10.34010/komputika.v12i1.8626

Putra, R. P. S., Aditya, C. S. K., & Wicaksono, G. W. (2024). Herbal Leaf Classification Using Deep Learning Model Efficientnetv2b0. JITK (Jurnal Ilmu Pengetahuan Dan Teknologi Komputer), 9(2), 301–307. https://doi.org/10.33480/jitk.v9i2.5119

Riastuti, A. D., Komarayanti, S., & Utomo, A. P. (2021). Karakteristik Morfologi Biji Kopi Robusta (Coffea Canephora) Pascapanen Di Kawasan Lereng Meru Betiri Sebagai Sumber Belajar Smk Dalam Bentuk E-Modul.

Sakib, S., Ashrafi, Z., & Abu Bakr Sidique, M. (2019). Implementation of Fruits Recognition Classifier using Convolutional Neural Network Algorithm for Observation of Accuracies for Various Hidden Layers. *ArXiv E-Journal*.

Sánchez-Cauce, R., Pérez-Martín, J., & Luque, M. (2021). Multi-input convolutional neural network for breast cancer detection using thermal images and clinical data. *Computer Methods and Programs in Biomedicine*, *204*, 106045. https://doi.org/10.1016/j.cmpb.2021.106045

Saputra, M., Kusrini, K., & Kurniawan, M. P. (2020). Identifikasi Mutu Bij Kopi Arabika Berdasarkan Cacat dengan Teknik Convolutional Neural Network. *Inspiration: Jurnal Teknologi Informasi Dan Komunikasi*, *10*(1), 27. https://doi.org/10.35585/inspir.v10i1.2533

Saputra, O., Mulyana, D. I., & Yel, M. B. (2022). Implementasi Algoritma Convolutional Neural Network (CNN) Untuk Klasifikasi Senjata Tradisional Di Jawa Tengah Dengan Metode Transfer Learning. *Jurnal SISKOM-KB (Sistem Komputer Dan Kecerdasan Buatan)*, *5*(2), 45–52. https://doi.org/10.47970/siskom-kb.v5i2.282

Sebatubun, M. M., & Pujiarini, E. H. (2018). Pengenalan Varietas Kopi Arabika Berdasarkan Fitur Bentuk. *Jurnal Informatika Dan Komputer (JIKO) –*, *3 (2)*.

Shabrina, N. H., Indarti, S., Lika, R. A., & Maharani, R. (2023). A comparative analysis of convolutional neural networks approaches for phytoparasitic nematode identification. *Communications in Mathematical Biology and Neuroscience*, *2023*. https://doi.org/10.28919/cmbn/7993

Sharma, S., Sharma, S., & Athaiya, A. (2020). Activation Functions In Neural Networks. *International Journal of Engineering Applied Sciences and Technology*, *04*(12), 310–316. https://doi.org/10.33564/IJEAST.2020.v04i12.054

Sharma, S., Verma, K., & Hardaha, P. (2022). Implementation of Artificial Intelligence in Agriculture. *Journal of Computational and Cognitive Engineering*. https://doi.org/10.47852/bonviewJCCE2202174

Siswoputranto, P. S. (1993). *Kopi internasional dan Indonesia* (1st ed.). Yogyakarta : Kanisius.

Sonka, M., Hlavac, V., & Boyle, R. (2014). *Image Processing, Analysis, and Machine Vision* (4th ed.). Cengage Learning.

Sun, Y., Zhu, L., Wang, G., & Zhao, F. (2017). Multi-Input Convolutional Neural Network for Flower Grading. *Journal of Electrical and Computer Engineering*, *2017*. https://doi.org/10.1155/2017/9240407

Susatyono, J. D. (2021). *Kecerdasan Buatan Kajian Konsep dan Penerapan*. Yayasan Prima Agus Teknik.

Tanjung, K. R., Liliana, & Juwiantho, H. (2022). *Klasifikasi Benda Organik dan Anorganik Dengan Metode YOLOv3 dan ResNet50*.

Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. https://doi.org/https://doi.org/10.48550/arXiv.1905.11946

Tan, M., & Le, Q. V. (2021). EfficientNetV2: Smaller Models and Faster Training. *International Conference on Machine Learning*. https://doi.org/https://doi.org/10.48550/arXiv.2104.00298

Tsang, S.-H. (2021). *Review: ResNeXt — 1st Runner Up in ILSVRC 2016 (Image Classification)*. Towards Data Science. https://towardsdatascience.com/review-resnext-1st-runner-up-of-ilsvrc-2016-image-classification-15d7f17b42ac

Wijaya, A. E., Swastika, W., & Kelana, O. H. (2021). Implementasi Transfer Learning Pada Convolutional Neural Network untuk Diagnosis Covid-19 Dan Pneumonia Pada Citra X-Ray. In *Sainsbertek Jurnal Ilmiah Sains & Teknologi* (Vol. 2).

Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. In *Insights into Imaging* (Vol. 9, Issue 4, pp. 611–629). Springer Verlag. https://doi.org/10.1007/s13244-018-0639-9

Yati, R., Rohana, T., & Rizky Pratama, A. (2023). Klasifikasi Jenis Mangga Menggunakan Algoritma Convolutional Neural Network. *JURNAL MEDIA INFORMATIKA BUDIDARMA*, *7*(3), 1265–1275. https://doi.org/10.30865/mib.v7i3.6445

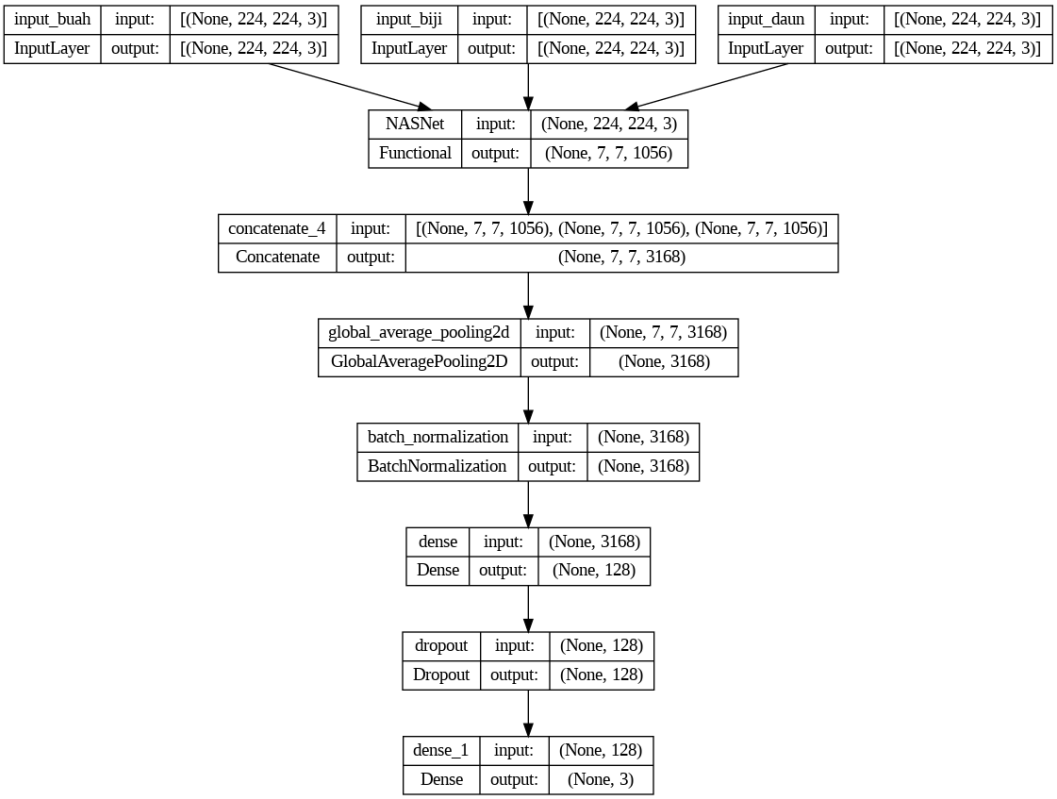
Zasari, M., Kartika, K., & Altin, D. (2023). Eksplorasi-Karakterisasi Morfologi Kopi Robusta Lokal di Pulau Bangka. *Agrikultura*, *34*(2), 200. https://doi.org/10.24198/agrikultura.v34i2.43179

Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2017). *Learning Transferable Architectures for Scalable Image Recognition*. http://arxiv.org/abs/1707.07012

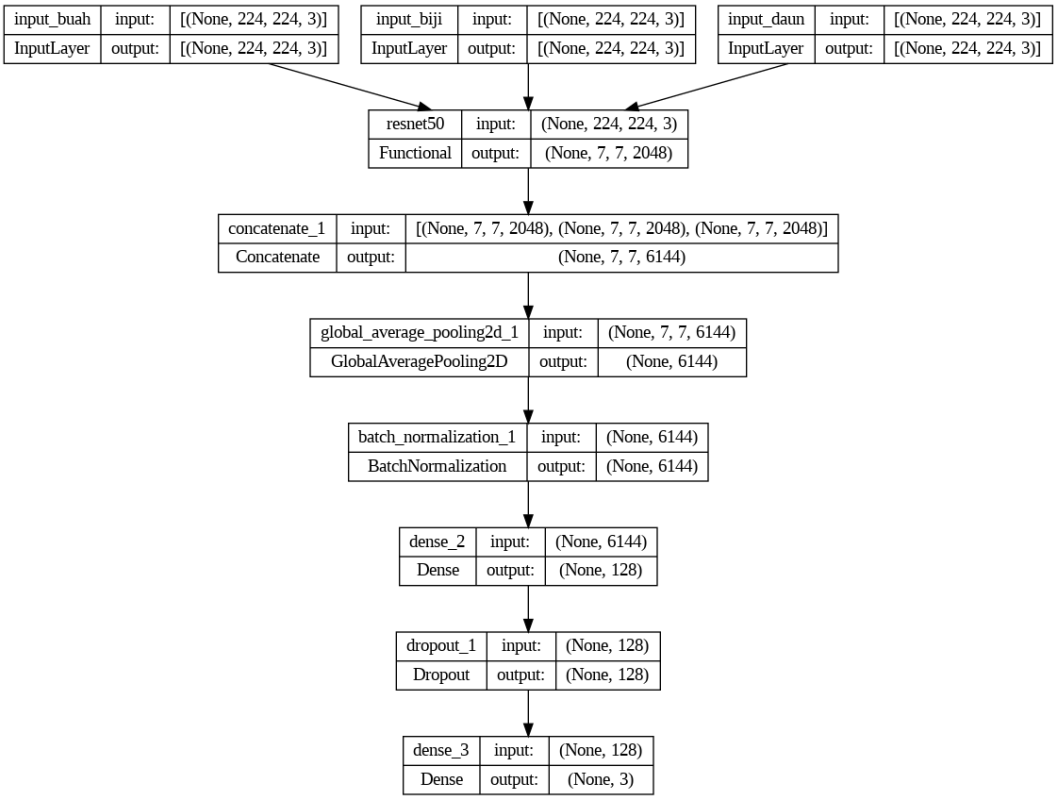
LAMPIRAN

Lampiran 1. Dokumentasi Pengambilan Data Biji, Buah dan Daun Kopi

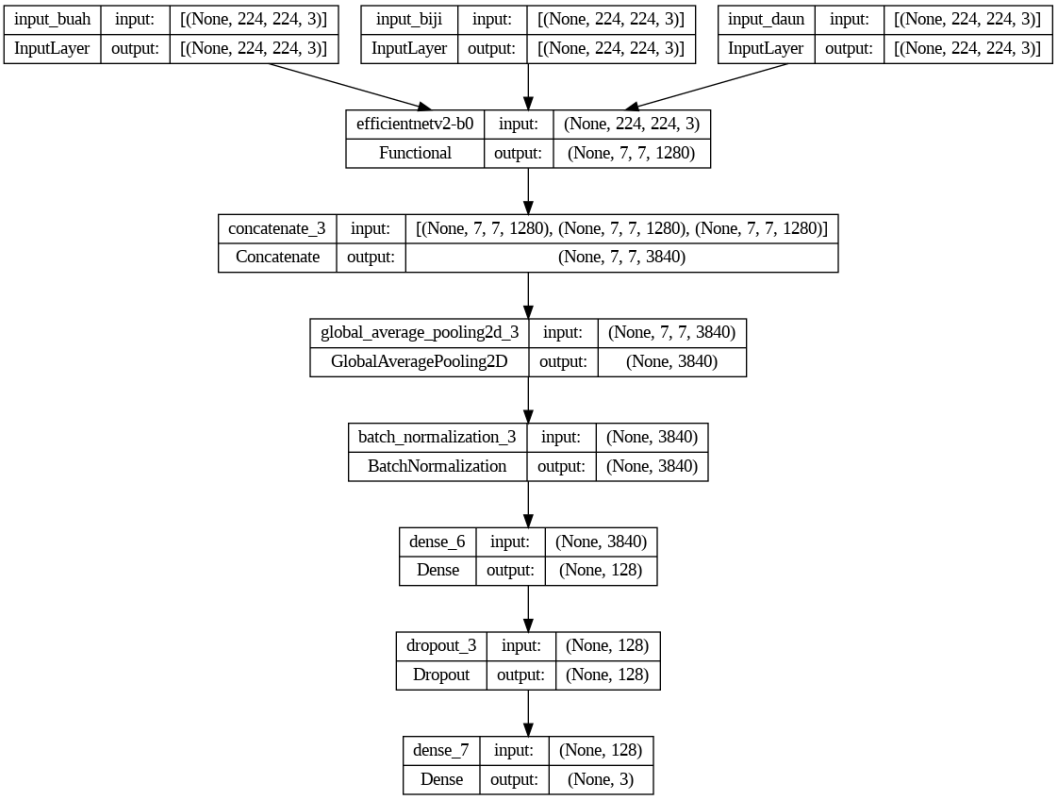
Lampiran 2. Visualisasi Arsitektur Model NasNetMobile



Lampiran 3. Visualisasi Arsitektur Model ResNet50



Lampiran 4. Visualisasi Arsitektur Model EfficientnetV2B0



Lampiran 5. Script visualisasi feature maps

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
def visualize_single_feature_map(activation, layer_name,
file_name):
    if len(activation.shape) == 4:
        n_features = activation.shape[-1]
        size = activation.shape[1]
        selected_channel = np.random.randint(n_features)
        channel_image = activation[0, :, :,
selected_channel].numpy()
    else:
        size = activation.shape[1]
        channel_image = activation[0, :].numpy()
    channel_image -= channel_image.mean()
    if channel_image.std() != 0:
        channel_image /= channel_image.std()
    channel_image *= 64
    channel_image += 128
    channel_image = np.clip(channel_image, 0, 255).astype('uint8')
    channel_image = cv2.resize(channel_image, (size, size))
    plt.figure(figsize=(4, 4))
    plt.title(f'Layer: {layer_name}')
    plt.grid(False)
    plt.imshow(channel_image, cmap='viridis')
    plt.axis('off')
    plt.savefig(file_name, bbox_inches='tight')
    plt.close()
```