IMPLEMENTASI METODE TRANSFER LEARNING UNTUK KLASIFIKASI MALARIA MENGGUNAKAN *RESNET-50*

LAPORAN TUGAS AKHIR

Laporan ini disusun guna memenuhi salah satu syarat untuk menyelesaikan program studi S1 Teknik Informatika pada Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang

DISUSUN OLEH :

ANDHI ROHMAN

32602100134

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM SULTAN AGUNG

SEMARANG

2025

IMPLEMENTATION OF TRANSFER LEARNING METHOD FOR MALARIA CLASSIFICATIONUSING RESNET-50

FINAL PROJECT

Proposed to complete the requirement to obtain a bachelor's degree (S1) at Informatics Engineering Departement of Industrial Technology Faculty Sultan Agung Islamic University

ARRANGED BY:

ANDHI ROHMAN

32602100134

MAJORING OF INFORMATICS ENGINEERING

INDUSTRIAL TECHNOLOGY FACULTY

SULTAN AGUNG ISLAMIC UNIVERSITY

SEMARANG

2025

LEMBAR PENGESAHAN TUGAS AKHIR

LEMBAR PENGESAHAN TUGAS AKHIR

IMPLEMENTASI METODE TRANSFER LEARNING UNTUK DETEKSI MALARIA MENGGUNAKAN RESNET-50

ANDHI ROHMAN
NIM 32602100134

Telah dipertahankan di depan tim penguji ujian sarjana tugas akhir
Program Studi Teknik Informatika
Universitas Islam Sultan Agung
Pada tanggal : 1 September 2025

TIM PENGUJI UJIAN SARJANA :

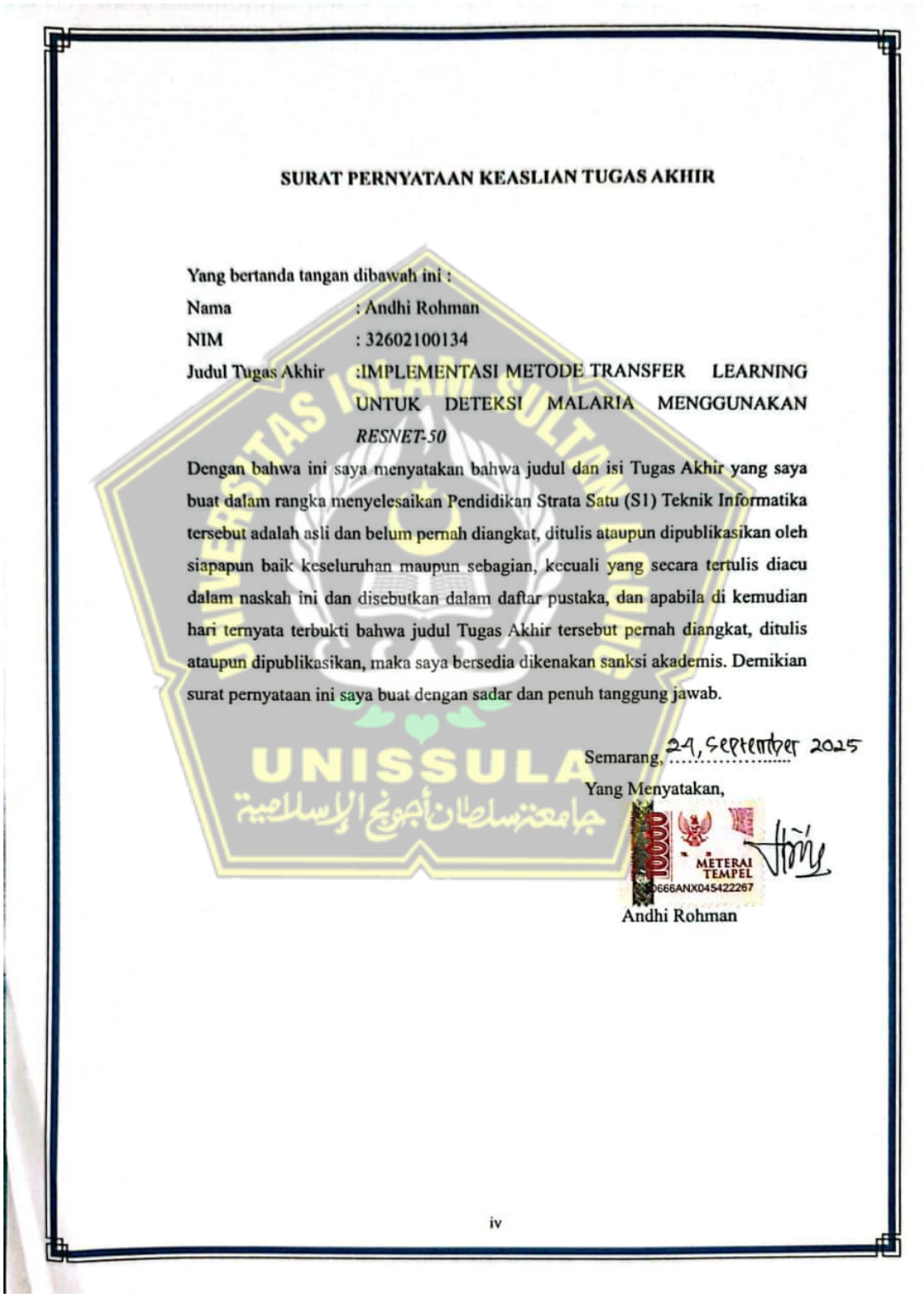
Sam Farisa Chaerul Haviana, ST., M.Kom NIDN. 210615046 (Penguji 1)	.....	09-09-2025
Dedy Kurniadi, ST., M.Kom NIDN. 210615048 (Penguji 2)	.....	10-09-2025
Imam Much Ibnu Subroto, ST., M.Sc., Ph.D NIDN. 210600017 (Pembimbing)	.....	08-09-2025

Semarang, 1 September 2025
Mengetahui,
Kaprodi Teknik Informatika
Universitas Islam Sultan Agung

Moch. Taufik., ST. MIT
NIDN. 0622037502

iii

SURAT PERNYATAAN KEASLIAN TUGAS AKHIR



SURAT PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan dibawah ini :

Nama : Andhi Rohman

NIM : 32602100134

Judul Tugas Akhir :IMPLEMENTASI METODE TRANSFER LEARNING UNTUK DETEKSI MALARIA MENGGUNAKAN *RESNET-50*

Dengan bahwa ini saya menyatakan bahwa judul dan isi Tugas Akhir yang saya buat dalam rangka menyelesaikan Pendidikan Strata Satu (S1) Teknik Informatika tersebut adalah asli dan belum pernah diangkat, ditulis ataupun dipublikasikan oleh siapapun baik keseluruhan maupun sebagian, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka, dan apabila di kemudian hari ternyata terbukti bahwa judul Tugas Akhir tersebut pernah diangkat, ditulis ataupun dipublikasikan, maka saya bersedia dikenakan sanksi akademis. Demikian surat pernyataan ini saya buat dengan sadar dan penuh tanggung jawab.

Semarang, 24 September 2025

Yang Menyatakan,

METERAI TEMPEL

Andhi Rohman

iv

PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH

PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH

Saya yang bertanda tangan dibawah ini :

Nama : Andhi Rohman

NIM : 32602100134

Program Studi : Teknik Informatika

Fakultas : Teknologi industri

Dengan ini menyatakan Karya Ilmiah berupa Tugas akhir dengan Judul : **IMPLEMENTASI METODE TRANSFER LEARNING UNTUK DETEKSI MALARIA MENGGUNAKAN *RESNET-50*.**

Menyetujui menjadi hak milik Universitas Islam Sultan Agung serta memberikan Hak bebas Royalti Non-Eksklusif untuk disimpan, dialihmediakan, dikelola dan pangkalan data dan dipublikasikan diinternet dan media lain untuk kepentingan akademis selama tetap menyantumkan nama penulis sebagai pemilik hak cipta. Pernyataan ini saya buat dengan sungguh-sungguh. Apabila dikemudian hari terbukti ada pelanggaran Hak Cipta/Plagiarisme dalam karya ilmiah ini, maka segala bentuk tuntutan hukum yang timbul akan saya tanggung secara pribadi tanpa melibatkan Universitas Islam Sultan agung.

Semarang, 24 September 2025

Yang menyatakan,

Andhi Rohman

KATA PENGANTAR

Dengan mengucap syukur alhamdulillah atas kehadirat Allah SWT yang telah memberikan rahmat dan karunianya kepada penulis, sehingga dapat menyelesaikan Tugas Akhir dengan judul "Implementasi Metode Transfer Learning Untuk Klasifikasi Malaria Menggunakan Resnet-50" ini untuk memenuhi salah satu syarat menyelesaikan studi serta dalam rangka memperoleh gelar sarjana (S-1) pada Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang.

Tugas Akhir ini disusun dan dibuat dengan adanya bantuan dari berbagai pihak, materi maupun teknis, oleh karena itu saya selaku penulis mengucapkan terima kasih kepada:

1. Rektor UNISSULA Bapak Prof. Dr. H. Gunarto, S.H., M.H yang mengizinkan penulis menimba ilmu di kampus ini.
2. Dekan Fakultas Teknologi Industri Ibu Dr. Novi Marlyana, S.T., M.T.
3. Dosen pembimbing I penulis Imam Much Ibnu Subroto, ST., M.Sc., Ph.D yang telah meluangkan waktu dan memberi ilmu. Serta memberikan banyak nasehat dan saran.
4. Orang tua penulis yang telah mengizinkan untuk menyelesaikan laporan ini,
5. Dan kepada semua pihak yang tidak dapat saya sebutkan satu persatu.

Dengan segala kerendahan hati, penulis menyadari masih terdapat banyak kekurangan dari segi kualitas atau kuantitas maupun dari ilmu pengetahuan dalam penyusunan laporan, sehingga penulis mengharapkan adanya saran dan kritikan yang bersifat membangun demi kesempurnaan laporan ini dan masa mendatang.

Semarang, ……..

Andhi Rohman

DAFTAR ISI

IMPLEMENTASI METODE TRANSFER LEARNING UNTUK KLASIFIKASI MALARIA MENGGUNAKAN *RESNET-50*..... i

***IMPLEMENTATION OF TRANSFER LEARNING METHOD FOR MALARIA CLASSIFICATIONUSING RESNET-50*..... ii**

LEMBAR PENGESAHAN TUGAS AKHIR..... iii

SURAT PERNYATAAN KEASLIAN TUGAS AKHIR..... iv

PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH..... v

KATA PENGANTAR..... vi

DAFTAR ISI..... vii

DAFTAR GAMBAR..... x

DAFTAR TABEL..... xii

ABSTRAK..... xiii

BAB I PENDAHULUAN..... 1

1.1 Latar Belakang..... 1

1.2 Rumusan Masalah..... 3

1.3 Batasan Masalah..... 3

1.4 Tujuan..... 4

1.5 Manfaat..... 4

1.6 Sistematika Penulisan..... 4

BAB II TINJAUAN PUSTAKA DAN DASAR TEORI..... 6

2.1 Tinjauan Pustaka..... 6

2.2 Dasar Teori..... 9

2.2.1 Malaria..... 9

2.2.2 *Deep Learning*..... 9

2.2.3 *Convolutional Neural Network (CNN)*..... 11

2.2.4 *Transfer Learning*..... 14

2.2.5 ResNet 50..... 17

2.2.6 Keras Image Data Generator..... 19

2.2.7 Split Ratio..... 21

2.2.8 Binary Cross-Entropy Loss..... 22

2.2.9 Confussion Matrix 23
2.2.10 *Classification Report* 24
2.2.11 Fungsi Aktivasi 25
2.2.12 Learning Rate 26
2.2.13 Epoch 26
BAB III METODE PENELITIAN 27
3.1 Metode Pengumpulan Data 27
3.1.1 Studi Literatur 28
3.1.2 Pengumpulan Data 28
3.1.3 *Preprocessing Data* 29
3.1.4 Split Data 29
3.1.5 Load Arsitektur Resnet-50 29
3.1.6 Compile Model 30
3.1.7 Training Model 30
3.1.8 Evaluasi Model 30
3.1.9 *Deployment* Model 31
3.2 *Software* Yang Digunakan 32
3.3 Perancangan sistem 34
BAB IV HASIL DAN ANALISIS PENELITIAN 37
4.1 Hasil Pengumpulan Data 37
4.2 Split Data 39
4.3 Augmentasi Data 41
4.4 Load Model 42
4.5 Compile Model 43
4.6 Training Model 44
4.6.1 Partial FineTuned 44
4.6.2 FineTuned Unfreeze 20 Layer 47
4.7 Evaluasi Model 50
4.7.1 Partial FineTuned 51
4.7.2 FineTuned Unfreeze 20 Layer 59
4.8 Visualisasi Hasil Prediksi Gambar 64
4.8.1 Visualisasi Model Data Validasi 64
4.8.2 Visualisasi Model Data Uji 66

4.8.3 Visualisasi Gambar Manual 69
4.8.4 Prediksi Gambar Tunggal dengan Penyimpanan Otomatis ke CSV dan PDF 71
4.9 Hasil Deployment..... 74
BAB V KESIMPULAN DAN SARAN 79
5.1 Kesimpulan 79
5.2 Saran..... 79
DAFTAR PUSTAKA..... 81

DAFTAR GAMBAR

Gambar 2. 1 *CNN Architecture* 11
Gambar 2. 2 Proses Konvolusi 12
Gambar 2. 3 Konsep Transfer Learning 16
Gambar 2. 4 Concept of Transfer Learning 17
Gambar 2. 5 *ResNet 50 Architecture* 18
Gambar 2. 6 *Contoh Representatif Data Augmentasi* 20
Gambar 2. 7 *Augmentation Data Flow* 21
Gambar 3. 1 Flowchart 27
Gambar 3. 2 Flowchart Sistem 35
Gambar 4. 1 Contoh citra dari masing-masing kelas 37
Gambar 4. 2 Model Summary Partial 46
Gambar 4. 3 Model Summary FineTuned Unfreeze 20 layer 49
Gambar 4. 4 Confusion Matrix Partial 52
Gambar 4. 5 Hasil Evaluasi Validasi 53
Gambar 4. 6 Training Validasi accuracy & loss partial 55
Gambar 4. 7 Visualisasi model per epoch partial 56
Gambar 4. 8 ROC Curve Partial 59
Gambar 4. 9 Confusion Matrix FineTuned Unfreeze 20 layer 60
Gambar 4. 10 Visualisasi training accuracy &loss unfreeze 20 layer 61
Gambar 4. 11 ROC Curve unfreeze 20 layer 62
Gambar 4. 12 Accuracy Loss unfreeze 20 layer 63
Gambar 4. 13 Prediksi Data Validasi 65
Gambar 4. 14 Visualisasi Data Uji 68
Gambar 4. 15 Hasil Visualisasi Prediksi Manual 70
Gambar 4. 16 Hasil Visualisasi Prediksi Manual 2 71
Gambar 4. 17 File CSV hasil prediksi 73

Gambar 4. 18 File PDF hasil prediksi 73
Gambar 4. 19 File PDF hasil prediksi 2 74
Gambar 4. 20 Halaman Awal 76
Gambar 4. 21 Halaman Prediksi 77

DAFTAR TABEL

Tabel 2. 1 Studi Literatur 6
Tabel 2. 2 Confusion Matrix 24
Tabel 4. 1 Sebaran Data 40

ABSTRAK

Malaria merupakan penyakit menular yang disebabkan oleh parasit *Plasmodium* dan masih menjadi masalah kesehatan serius di Indonesia, khususnya wilayah timur. Diagnosis malaria yang cepat dan akurat sangat penting untuk mencegah komplikasi serta memutus rantai penularan. Penelitian ini bertujuan mengimplementasikan metode *transfer learning* menggunakan arsitektur Convolutional Neural Network (CNN) ResNet-50 dalam mendeteksi sel darah terinfeksi malaria pada citra mikroskopis. Dataset yang digunakan terdiri atas citra sel darah yang dikategorikan menjadi dua kelas, yaitu terinfeksi (*parasitized*) dan tidak terinfeksi (*uninfected*). Model dilatih dengan pembagian data *training*, *validation*, dan *testing*. Hasil penelitian menunjukkan bahwa ResNet-50 mencapai akurasi sebesar 93,00%, dengan presisi 96,00%, recall 97,00%, F1-score 93,00%, serta nilai AUC 98,00%. Dengan performa tersebut, pendekatan ini berpotensi menjadi solusi pendukung diagnosis malaria yang lebih cepat, objektif, dan efisien, serta dapat diintegrasikan sebagai sistem skrining laboratorium maupun decision support bagi tenaga medis, sekaligus mendukung penelitian lanjutan.

Kata Kunci: Malaria, *Plasmodium*, Deep Learning, CNN, ResNet-50, Transfer Learning.

ABSTRACT

Malaria is an infectious disease caused by Plasmodium parasites and remains a major public health problem in Indonesia, particularly in the eastern regions. Rapid and accurate diagnosis is essential to prevent complications and break the chain of transmission. This study aims to implement a transfer learning approach using the ResNet-50 Convolutional Neural Network (CNN) architecture to detect malaria-infected blood cells from microscopic images. The dataset consists of blood cell images categorized into two classes: infected (parasitized) and uninfected. The model was trained using training, validation, and testing sets. The results show that ResNet-50 achieved an accuracy of 93.00%, with a precision of 96.00%, recall of 97.00%, F1-score of 93.00%, and an AUC value of 98.00%. With this performance, the proposed approach has strong potential to serve as a faster, more objective, and efficient malaria diagnostic tool, and can be integrated as a laboratory screening system as well as a decision support system for medical staff, while also supporting further research.

Keywords: Malaria, Plasmodium, Deep Learning, CNN, ResNet-50, Transfer Learning.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Parasit *Plasmodium* yang dibawa oleh nyamuk *Anopheles* betina menjadi penyebab utama malaria, salah satu penyakit menular yang hingga kini menimbulkan masalah serius di negara-negara tropis, termasuk Indonesia. Pada tahun 2020, WHO mencatat sekitar 241 juta kasus malaria secara global. Angka ini menegaskan bahwa malaria tetap menjadi salah satu penyakit menular dengan prevalensi tertinggi di dunia dengan 627.000 kematian (WHO, 2021). Indonesia sendiri mencatat 418.546 kasus malaria pada 2023, di mana lebih dari 90% kasus berasal dari Papua dan sekitarnya. Pemerintah menargetkan 8 juta tes malaria per tahun sebagai langkah menuju eliminasi malaria nasional pada tahun 2030 (Aisyah *dkk.*, 2024).

Di Indonesia, wilayah timur seperti Papua, Papua Barat, Nusa Tenggara Timur (NTT) dan Kalimantan Timur menjadi daerah dengan tingkat penularan malaria yang paling tinggi. Pada 2023, lebih dari 90% kasus malaria nasional berasal dari Papua dan sekitarnya (369.000 kasus dari total 418.546). Wilayah ini memiliki tantangan besar, yaitu keterbatasan akses ke fasilitas kesehatan dan kekurangan tenaga ahli laboratorium. Meskipun capaian eliminasi malaria telah mendekati 79% kabupaten/kota di Pulau Jawa dan Bali, daerah timur Indonesia masih mengalami endemisitas tinggi yang memerlukan inovasi teknologi diagnosis yang lebih cepat dan akurat (Fransisca *dkk.*, 2025).

Diagnosis malaria yang akurat dan cepat sangat penting untuk mencegah komplikasi serius dan memutus rantai penularan. Salah satu prosedur standar dalam deteksi malaria adalah pengamatan apus darah tipis atau tebal di bawah mikroskop setelah dilakukan pewarnaan Giemsa masih menjadi gold standard diagnosis. Metode ini dapat mengidentifikasi spesies parasit dan menghitung parasitemia, tetapi memerlukan 10–30 menit per sampel dan sangat bergantung pada keterampilan analis laboratorium. Kelelahan

analis, jumlah sampel yang banyak, dan kualitas slide dapat menyebabkan kesalahan diagnosis *(human error)*. *Rapid Diagnostic Test (RDT)* lebih cepat (±15 menit), tetapi memiliki keterbatasan akurasi pada parasitemia rendah serta tidak mampu membedakan spesies *Plasmodium* (WHO, 2020).

Studi terbaru menunjukkan keterbatasan metode konvensional ini. Penelitian di Ghana (Opoku Afriyie dkk., 2023) melaporkan sensitivitas mikroskop hanya 39,3%, sedangkan RDT 55,7%, keduanya dibandingkan dengan varATS qPCR sebagai acuan. Penelitian lain di Tanzania dan Ethiopia juga menunjukkan tren serupa, di mana sensitivitas mikroskop jarang melampaui 70% (Tegegne *dkk.*, 2025). Hal ini memperkuat kebutuhan teknologi baru yang lebih konsisten dan cepat.

Perkembangan teknologi kecerdasan buatan (AI) memberikan alternatif inovatif untuk mendukung diagnosis malaria. Model berbasis *Convolutional Neural Network (CNN)*, seperti *ResNet-50*, mampu menganalisis gambar mikroskopis secara otomatis dalam hitungan detik (<5 detik), memberikan hasil yang objektif, konsisten, dan terdokumentasi. Kedepannya, model resnet-50 yang akan dikembangkan dalam penelitian ini dapat dimanfaatkan sebagai alat bantu skrining awal di daerah dengan keterbatasan tenaga medis (Sinha dan Gupta, 2023).

Penelitian terkini mendukung keunggulan AI. AIDMAN, sistem berbasis smartphone dengan kombinasi *YOLOv5* dan *CNN*, mencapai akurasi hingga 98,44%, setara dengan analis mikroskopis ahli (Ruicun Liu dkk., 2023). miLab MAL, platform mikroskop digital dengan AI, bahkan mampu mendeteksi parasitemia rendah (<0,1%) yang sering terlewat pada pemeriksaan manual, dengan sensitivitas dan spesifisitas mendekati 100% (Nagendra *dkk.*, 2024). Studi berbasis ResNet-50 pada dataset NIH juga mencatat akurasi 98,75%, precision 99,3%, dan recall 99,5% (Sinha dan Gupta, 2023).

Berdasarkan kondisi tersebut, penelitian ini memiliki urgensi tinggi untuk mengembangkan sistem deteksi malaria berbasis *transfer learning* menggunakan arsitektur *ResNet-50*. Sistem ini tidak dimaksudkan untuk

menggantikan *RDT*, tetapi untuk mendukung pemeriksaan mikroskopis agar lebih cepat, akurat, dan terdigitalisasi, khususnya di daerah endemis dengan keterbatasan tenaga ahli laboratorium selain itu bisa digunakan untuk sistem deteksi awal atau deteksi dini dan digunakan sebagai decision support tenaga medis serta digunakan sebagai fitur otomatisasi dan pelaporan digital dan bisa digunakan sebagai arsip laboratorium. Teknologi ini diharapkan dapat mendukung upaya eliminasi malaria di Indonesia tahun 2030. Kecepatan dan akurasi.

Dalam Penelitian ini mengimplementasikan model CNN *pre-trained* melalui teknik transfer learning sebagai strategi utama dalam klasifikasi citra malaria dengan arsitektur *ResNet-50* sebagai fitur ekstraktor dalam mengklasifikasi *parasitized* dan *uninfected cells* dengan ini semoga didapatkan hasil dan solusi untuk membantu dalam peningkatan sistem diagnosis penyakit malaria.

1.2 Rumusan Masalah

Bagaimana membangun model klasifikasi citra sel darah malaria menggunakan metode transfer learning dengan arsitektur ResNet-50 sebagai *decision support system* bagi tenaga medis?

1.3 Batasan Masalah

Adapun pembatasan masalah dari penulisan proposal sebagai berikut :

1. Data yang digunakan dalam penelitian ini berasal dari dataset “Cell Images for Detecting Malaria” yang diunduh dari Kaggle, Data terdiri atas gambar sel darah yang dikelompokkan dalam dua label, yakni sel yang terinfeksi parasit (*Parasitized*) dan sel yang tidak terinfeksi (*Uninfected*).
2. Proses klasifikasi hanya dilakukan untuk dua kelas: sel darah terinfeksi malaria dan tidak terinfeksi, sehingga tidak mencakup identifikasi jenis atau spesies parasit (misalnya *Plasmodium falciparum*, *vivax*, dll).
3. Penelitian hanya menggunakan satu arsitektur CNN, yaitu ResNet-50, dan tidak menggunakan arsitektur model lainnya seperti VGG16, Inception, atau DenseNet atau lainnya.

1.4 Tujuan

Tujuan dari penelitian ini adalah untuk membangun dan mengimplementasikan model klasifikasi citra sel darah menggunakan pendekatan transfer learning dengan arsitektur ResNet-50, serta mengevaluasi kinerjanya dalam mendeteksi infeksi malaria berdasarkan metrix akurasi, presisi, recall, dan F1-score, dan AUC.

1.5 Manfaat

Penelitian ini diharapkan dapat memberikan manfaat :

1. Penulis dapat mendapatkan knowledge dalam penelitian mengenai metode transfer learning dalam deteksi Malaria menggunakan ResNet-50, CNN.
2. Penulis dapat memahami bagaimana membuat sebuah penelitian yang baik serta menambah ilmu pengetahuan terutama di bidang machine learning.
3. Mendapatkan ilmu pengetahuan mengenai penyakit malaria.

1.6 Sistematika Penulisan

Sistematika penulisan yang akan digunakan oleh penulis dalam sebuah pembuatan laporan tugas akhir adalah sebagai berikut :

BAB I : PENDAHULUAN

Dalam Bab I menjelaskan tentang latar belakang, perumusan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA DAN DASAR TEORI

Dalam Bab II ini akan dibahas mengenai teori-teori dan hasil penelitian terdahulu yang relevan, termasuk konsep Deep Learning, CNN, Transfer Learning, dan ResNet-50.

BAB III : METODE PENELITIAN

Pada Bab III ini peneliti menjelaskan tentang metode penelitian yang digunakan, termasuk pengumpulan data, preprocessing data, arsitektur model, dan langkah-langkah evaluasi model.

BAB IV : HASIL DAN ANALISIS PENELITIAN

Pada Bab IV ini Menampilkan hasil eksperimen dan pembahasan kinerja model dalam mendeteksi malaria.

BAB V : KESIMPULAN DAN SARAN

Pada Bab V ini Berisi kesimpulan yang diperoleh dari penelitian dan saran untuk pengembangan lebih lanjut.

BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Dalam penelitian yang bertujuan untuk membandingkan arsitektur *CNN* dalam klasifikasi *Plant Diseases*. Penelitian ini membandingkan dan menguji arsitektur *AlexNet, GoogleNet, Inception V3, dan ResNet*. Model yang dibangun berhasil mencapai nilai akurasi sebesar 99.15% Hal ini ditunjukkan pada ditunjukan pada tabel 2.1 (Maeda-Gutiérrez *dkk.*, 2020).

Tabel 2. 1 Studi Literatur

Performance Measures	**AlexNet**	**GoogleNet**	**Inception V3**	**ResNet 18**	**ResNet 50**
Accuracy	98.93	99.39	98.65	99.06	99.15
Precision	98.74	99.29	98.29	98.76	98.94
Sensitivity	98.38	99.12	97.84	98.67	98.72
Specificity	99.88	99.93	99.85	99.89	99.90
F-Score	98.54	99.20	98.05	98.71	98.82
Area under the curve (AUC)	99.28	99.72	99.20	99.20	99.55
Time (min)	94.17	133.15	649.85	147.73	476.78

Penelitian ini menyimpulkan bahwa penggunaan arsitektur *ResNet* memiliki tingkat akurasi yang sangat bagus dalam klasifikasi *Tomato Plant Diseases*.

Penelitian lain yang mengevaluasi strategi pembelajaran *CNN* dalam klasifikasi citra multi-kelas dengan 100 kelas juga menemukan bahwa *ResNet* memiliki performa terbaik dengan akurasi mencapai 75,82%. Temuan ini memperkuat posisi *ResNet* sebagai arsitektur yang andal untuk klasifikasi citra kompleks (Mahmud dkk., 2020).

Dalam ranah kesehatan, penggunaan *Deep Learning* semakin luas. Salah satu studi menggunakan *Deep Belief Network (DBN)* untuk pendeteksian parasit malaria dalam citra sel darah dan berhasil mencapai akurasi sebesar 96,35% (Bibin dkk., 2020).

Penelitian terkait klasifikasi tingkat keparahan penyakit Tuberkulosis (TB) telah memanfaatkan arsitektur *ResNet* untuk melakukan analisis. Pada penelitian tersebut, dataset yang digunakan terdiri dari lima kategori tingkat keparahan TB. Seluruh citra dalam dataset tersebut melalui tahap preprocessing dengan merubah ukuran citra menjadi 128×128 piksel. Hasil dari penelitian ini menunjukkan bahwa model mampu mencapai akurasi rata-rata sebesar 85,29% (Gao *dkk.*, 2023)**.**

Penelitian yang berfokus pada peran *Data Science* dalam pengolahan data biomedis menunjukkan bahwa pendekatan ini memiliki potensi besar dalam menganalisis data biologis serta mendukung berbagai penelitian di bidang medis lainnya. *Data Science* memungkinkan analisis mendalam terhadap pelbagai sumber data, termasuk citra medis dan data genomik, dengan bantuan teknik machine learning dan *deep learning*. Pendekatan ini mampu mendeteksi pola tersembunyi, mempercepat diagnosis, serta mendukung pengembangan terapi dan prediksi kondisi kesehatan pasien, meski tantangan seperti kebutuhan akan data berkualitas, daya komputasi tinggi, dan keterbacaan model masih tetap menjadi hambatan (Alarcón-Soto *dkk.*, 2020).

Penelitian yang dilakukan dengan tujuan untuk membandingkan kinerja beberapa model *pretrained* menunjukkan bahwa DenseNet-201 menjadi arsitektur yang paling unggul. Model ini berhasil mencapai akurasi tertinggi sebesar 98%, mengungguli model lainnya dalam tugas klasifikasi citra yang diuji pada penelitian tersebut. Keunggulan DenseNet-201 terletak pada kemampuannya meminimalkan degradasi gradien dan mengoptimalkan aliran informasi melalui koneksi langsung antar lapisan, sehingga memfasilitasi pembelajaran fitur yang lebih kaya dan representatif. Studi ini memberikan bukti empiris bahwa pemilihan arsitektur *pretrained* yang tepat dapat berdampak signifikan pada kinerja model, khususnya dalam domain yang memiliki kompleksitas visual tinggi seperti citra medis (Rahman *dkk.*, 2020)**.**

Terdapat juga penelitian yang secara khusus mengkaji berbagai strategi dan mekanisme *transfer learning*. Studi ini menyajikan wawasan mendalam mengenai penerapan *transfer learning* dalam berbagai domain, termasuk klasifikasi citra medis. Penelitian tersebut menguraikan beragam pendekatan *transfer learning*, mulai dari *feature extraction* hingga *fine-tuning*, serta faktor-faktor yang memengaruhi keberhasilannya, seperti kesesuaian antara domain sumber dan target, ukuran dataset, serta kompleksitas model. Temuan mereka menunjukkan bahwa *transfer learning* tidak hanya mempercepat proses pelatihan, tetapi juga mampu meningkatkan akurasi, khususnya pada skenario dengan keterbatasan data. Namun demikian, penelitian ini juga menyoroti adanya tantangan, seperti risiko *negative transfer* apabila domain sumber dan target berbeda secara signifikan, yang dapat berdampak negatif pada performa model (Zhuang *dkk.*, 2021).

Adapun Penelitian yang secara khusus mengimplementasikan arsitektur *Convolutional Neural Network ResNet-50* dalam tugas klasifikasi citra, yang relevan dengan penelitian ini. Hasil eksperimen menunjukkan bahwa model tersebut mampu mencapai rata-rata akurasi sebesar 99,80%, sehingga membuktikan efektivitas ResNet-50 sebagai metode yang andal dan akurat dalam pengolahan citra medis (Mukti dan Biswas, 2020).

Selain itu, terdapat kajian yang membahas berbagai aplikasi deep learning dalam analisis citra medis, termasuk deteksi malaria. Studi ini menyoroti tren terbaru dalam penggunaan *deep learning* untuk diagnosis penyakit, sekaligus tantangan yang perlu diatasi di masa depan (Hussain dkk., 2020).

Secara keseluruhan, hasil dari berbagai penelitian ini memperkuat dasar bahwa arsitektur *ResNet-50* sangat potensial untuk diterapkan dalam penelitian ini. Keunggulan *ResNet-50* terletak pada kemampuannya dalam mengekstraksi fitur secara mendalam dan penerapan *transfer learning* yang efektif, terutama untuk data dengan skala terbatas.

2.2 Dasar Teori

2.2.1 Malaria

Malaria merupakan penyakit menular yang disebabkan oleh protozoa dari genus *Plasmodium*. Pada manusia, infeksi umumnya melibatkan empat spesies utama, yaitu *Plasmodium falciparum*, *P. vivax*, *P. ovale*, dan *P. malariae*. Penularan berlangsung melalui gigitan nyamuk *Anopheles* betina yang berperan sebagai vektor. Setelah masuk ke tubuh, parasit akan berkembang biak di dalam eritrosit dan menyebabkan kerusakan sel darah. Penyakit ini dapat menyerang semua kelompok umur, baik laki-laki maupun perempuan, dengan gejala klinis berupa demam periodik, menggigil, keringat berlebih, sakit kepala, mual, hingga muntah (WHO, 2021).

Gejala malaria umumnya muncul setelah masa inkubasi sekitar 10–15 hari pasca gigitan nyamuk yang terinfeksi. Manifestasi awal yang sering dialami penderita meliputi demam, menggigil, keringat berlebihan, sakit kepala, nyeri otot, mual, dan muntah. Gejala tersebut biasanya terjadi secara periodik sesuai siklus pecahnya sel darah merah yang terinfeksi parasit. Infeksi berat, terutama yang disebabkan oleh *P. falciparum*, dapat berkembang menjadi kondisi serius seperti anemia parah atau gangguan organ vital. Malaria dapat menyerang semua kelompok umur tanpa memandang jenis kelamin, sehingga menjadi ancaman kesehatan masyarakat di wilayah endemik (WHO, 2021).

2.2.2 *Deep Learning*

Deep learning merupakan salah satu pendekatan dalam *machine learning* yang meniru cara kerja jaringan saraf biologis melalui penggunaan jaringan saraf tiruan *(artificial neural networks)* dengan susunan lapisan yang banyak. Pendekatan ini memungkinkan komputer mempelajari representasi data secara bertingkat, sehingga dapat mengenali pola yang kompleks. Tujuan utama *deep learning* adalah untuk mengekstraksi representasi fitur yang mendalam dan abstrak dari data yang kompleks, sehingga dapat menghasilkan prediksi atau hasil yang lebih akurat dan kompleks (Rajaraman dkk., 2020).

Deep learning dalam bentuk paling sederhana, mengusulkan kelas fungsi komposisi berikut :

$$\left\{ f\left(x; \theta = WL\sigma L\left(WL - 1 \ldots \sigma 2\left(W2\sigma 1(W1x)\right)\right)\right| \theta = \{W1, \ldots, WL\}\right\}. \quad (1)$$

Di sini, untuk setiap $1 \leq \mathbf{l} \leq L$, $\sigma l(\cdot)$ adalah suatu fungsi nonlinear, dan $\theta = \{W1, ..., WL\}$ terdiri dari matriks-matriks dengan ukuran yang sesuai. Meskipun sederhana, *deep learning* telah mencapai kemajuan signifikan dalam menangani masalah pembelajaran dari data selama dekade terakhir. Secara khusus, *deep learning* telah mencapai kinerja yang mendekati atau bahkan melebihi manusia dalam berbagai tugas penting dalam kecerdasan buatan, termasuk pengenalan gambar, *game*, dan mesin terjemahan. Dengan kapabilitas yang besar, dampak *deep learning* juga berkembang pesat dalam bidang-bidang di luar kecerdasan buatan; contohnya meliputi statistik, matematika terapan, penelitian klinis, dan lain sebagainya. (Rajaraman dkk., 2020).

Dalam pembelajaran mesin (*machine learning*), proses klasifikasi citra maupun pengenalan suara pada awalnya bergantung pada teknik ekstraksi fitur manual sebelum diterapkan pada algoritma pembelajaran. Namun, pendekatan tradisional ini memiliki keterbatasan dalam hal kecepatan, akurasi, dan kemampuan generalisasi. Perkembangan jaringan saraf tiruan berlapis banyak kemudian melahirkan konsep *deep learning*, yang mampu mempelajari representasi data secara otomatis dan hierarkis. Dengan keunggulan pada skalabilitas dan akurasi, *deep learning* kini menjadi metode dominan baik di komunitas riset maupun industri, terutama dalam bidang *computer vision, speech recognition,* dan *natural language processing* (Alzubaidi *dkk.*, 2021).

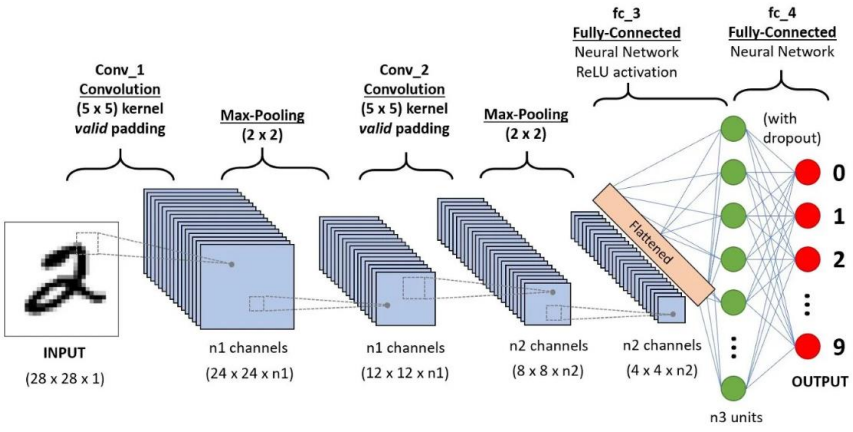
Pada pendekatan *machine learning* tradisional, *feature engineering* berperan penting dalam mengekstraksi pola dari data agar algoritma dapat bekerja secara optimal. Namun, pada *deep learning*, proses representasi fitur dilakukan secara otomatis oleh jaringan saraf tiruan berlapis, sehingga

mengurangi ketergantungan pada feature engineering manual membedakan kelas. *Feature Engineering* juga merupakan teknik yang paling penting untuk mencapai hasil yang baik pada tugas prediksi. Namun, sulit untuk dipelajari dan dikuasai karena kumpulan data dan jenis data yang berbeda memerlukan pedekatan teknik yang berbeda juga. Algoritma yang digunakan pada *Feature Engineering* dapat menemukan pola umum yang penting untuk membedakan antara kelas (Alzubaidi *dkk.*, 2021).

Feature engineering manual pada *machine learning* tradisional membutuhkan banyak waktu dan tenaga karena harus dilakukan secara eksplisit oleh pakar. Sebaliknya, *deep learning* mampu melakukan ekstraksi fitur secara otomatis, sehingga memberikan keunggulan dari segi efisiensi dan akurasi dalam berbagai aplikasi modern (Alzubaidi *dkk.*, 2021).

2.2.3 *Convolutional Neural Network (CNN)*

Jaringan saraf konvolusional *(CNN)* merupakan jenis jaringan saraf maju yang dikhususkan untuk pengolahan gambar. Secara umum, *CNN* cocok untuk analisis data dengan struktur spasial yang menonjol. Komponen pembangun dalam *Convolutional Neural Network (CNN)* dibangun dengan beberapa lapisan, termasuk lapisan konvolusi yang mengekstrak fitur dari gambar, dan lapisan pooling yang mereduksi dimensi data. Kedua komponen ini berperan kunci dalam memungkinkan *CNN* untuk efektif mengidentifikasi pola dan fitur yang kompleks dalam data gambar, seperti ditunjukkan pada gambar 2.1 (Rajaraman dkk., 2020).



Gambar 2. 1 *CNN Architecture*

Konvolusi (convolution) adalah operasi matematika yang digunakan untuk mengekstraksi fitur dari sebuah citra dengan cara mengalikan matriks kecil yang disebut kernel atau filter dengan bagian tertentu dari citra input, kemudian menjumlahkan hasilnya menjadi satu nilai baru. Proses ini dilakukan secara bergeser (sliding) ke seluruh area citra dengan langkah tertentu yang disebut stride, dan terkadang ditambahkan padding di tepi citra untuk mempertahankan ukuran keluaran. Hasil dari proses ini adalah feature map, yaitu representasi baru yang berisi informasi penting seperti tepi, tekstur, pola bentuk, atau objek tertentu. Dalam konteks Convolutional Neural Network (CNN), bobot pada filter tidak ditentukan secara manual, melainkan dipelajari secara otomatis selama proses pelatihan, sehingga lapisan konvolusi dapat mengenali fitur pada berbagai tingkat, mulai dari pola sederhana hingga struktur yang lebih kompleks pada lapisan berikutnya, adapun proses konvolusi dapat diliat pada gambar 2.2 dibawah ini.

Gambar 2. 2 Proses Konvolusi

Penjelasan tiap alurnya :

1. Input
   - Di sisi kiri, terdapat sebuah matriks berwarna biru yang mewakili citra input (biasanya representasi grayscale dalam bentuk piksel).

- Setiap angka adalah nilai piksel (dalam contoh ini berupa 0 atau 1).

2. Filter / Kernel
   - Bagian hijau adalah filter atau kernel (misalnya ukuran 3×3).
   - Filter ini memiliki bobot tertentu (juga angka 0 atau 1 dalam contoh ini).
   - Filter digeser (sliding) melintasi input, melakukan perhitungan pada setiap posisi.
3. Proses Konvolusi
   - Pada tiap langkah, filter melakukan perkalian elemen-per-elemen (element-wise multiplication) antara nilai piksel input dan bobot filter.
   - Hasil penjumlahan ditambahkan, Angka hasil penjumlahan inilah yang menjadi nilai di feature map (matriks di kanan berwarna merah).
4. Sliding Window
   - Filter terus bergeser ke kanan, lalu ke bawah, mengulangi langkah yang sama di setiap lokasi (sliding window).
   - Setiap posisi menghasilkan satu angka output di feature map.
5. Output (Feature Map)
   - Matriks merah di kanan adalah hasil konvolusi, disebut feature map.
   - Feature map berisi informasi fitur yang telah diekstraksi dari citra input oleh filter yang dipakai.

Dengan demikian, proses konvolusi memungkinkan CNN belajar secara otomatis fitur yang relevan dari citra, tanpa memerlukan perancangan fitur manual seperti pada metode tradisional. Berikut adalah lapisan – lapisan penyusunnya :

1. Convolutional Layer (CONV)

Lapisan konvolusi adalah inti dari *CNN*, di mana konvolusi dilakukan pada data gambar untuk mengekstrak fitur-fitur penting. Konvolusi melibatkan menerapkan filter (atau kernel) pada gambar untuk menghitung jumlah bobot tertentu di wilayah kecil gambar. Filter ini bertindak sebagai detektor pola tertentu, seperti tepi, garis, atau bentuk tertentu, yang

membantu dalam pengenalan pola. Selama pelatihan, bobot dalam filter diatur secara otomatis melalui proses pembelajaran untuk mengidentifikasi fitur-fitur yang relevan dalam data. Hasil konvolusi menghasilkan peta fitur yang mencerminkan kehadiran fitur-fitur tersebut dalam gambar. Penggunaan banyak filter di lapisan konvolusi memungkinkan CNN untuk menangkap fitur-fitur yang berbeda secara paralel.

$$x_j^l = f\left(\sum_{i \in M_i} x_j^{l-1} * k_{ij}^l + b_j^l\right) \quad \textbf{(2)}$$

l = *layer* pada CNN

j = *specific convolution feature map*

f = fungsi aktivasi

M_j = *selection of input maps*

k_{ij} = *filter*

b_j = *feature map bias*

2. *Pooling Layer (POOL)*

Pooling layer bertugas mengecilkan ukuran peta fitur yang dihasilkan konvolusi, sehingga model menjadi lebih ringan dan risiko *overfitting* berkurang. Operasi *pooling* (biasanya *max pooling*) diterapkan pada peta fitur dengan tujuan mengambil nilai maksimum dalam kotak atau wilayah kecil. Hasilnya adalah peta fitur yang lebih kecil dengan informasi yang lebih tajam dan penting dari gambar asli. Lapisan *pooling* juga membantu membuat representasi fitur lebih *invariant* terhadap translasi dan perubahan posisi objek dalam gambar.

3. *Full Connected Layer*

Layer ini digunakan untuk mengekstrak *feature* pada *layer* sebelumnya untuk selanjutnya dilakunnya tugas klasifikasi. Hasil dari *convolutional* atau *pooling layer* terakhir akan diberikan ke *fully connected layers* seperti dalam jaringan saraf asli.

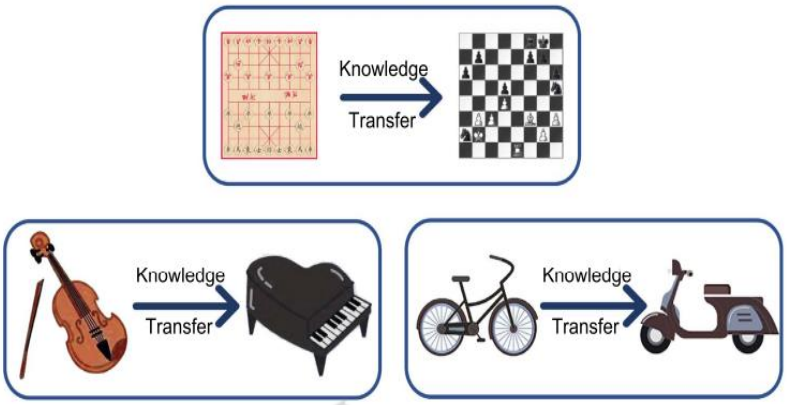
2.2.4 *Transfer Learning*

Transfer learning adalah pendekatan dalam pembelajaran mesin di mana model yang telah dilatih sebelumnya untuk satu tugas dapat digunakan

sebagai dasar untuk memecahkan tugas lain yang berbeda. Pada dasarnya, model yang telah belajar dari satu domain dapat "mentransfer" pengetahuannya untuk membantu memecahkan masalah di domain lain. Pendekatan ini terutama efektif ketika data pelatihan untuk tugas baru terbatas atau sulit diperoleh (Zhuang *dkk.*, 2021).

Transfer learning dapat menjadi berguna dalam aplikasi-aplikasi *CNN* di mana datasetnya tidak besar. Konsep dari *transfer learning* ditunjukkan dalam Gambar 2.2, di mana model yang sebelumnya dilatih dari dataset yang terdiri dari jutaan gambar, seperti halnya *ImageNet*, dapat digunakan untuk aplikasi bersama dataset yang relatif lebih kecil (Rahman *dkk.*, 2020).

Transfer learning, yang berfokus pada transfer pengetahuan antar domain, merupakan metodologi pembelajaran mesin yang ditujukan untuk memecahkan masalah. Konsep tentang *transfer learning* mungkin awalnya berasal dari psikologi pendidikan. Menurut teori generalisasi transfer, yang diusulkan oleh psikolog C. H. Judd, pembelajaran untuk mentransfer adalah hasil dari generalisasi pengalaman. Memungkinkan untuk mentransfer dari satu situasi ke situasi lain, selama seseorang menggeneralisasi pengalamannya. Menurut teori ini, prasyarat dari transfer adalah adanya hubungan antara dua aktivitas pembelajaran. Dalam praktiknya, seseorang yang telah belajar biola dapat belajar piano lebih cepat daripada orang lain, karena baik biola maupun piano adalah alat musik dan mungkin memiliki beberapa kesamaan *knowledge* (Zhuang *dkk.*, 2021).

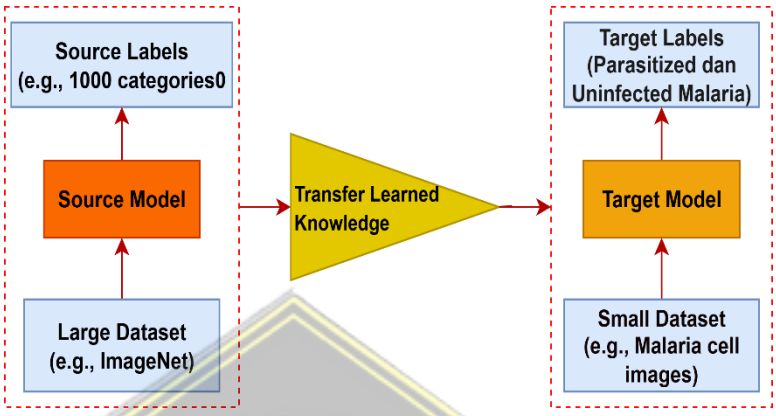


Gambar 2. 3 Konsep Transfer Learning

Gambar 2.3 menunjukkan beberapa contoh intuitif tentang *transfer learning*. Terinspirasi dari kemampuan manusia untuk mentransfer pengetahuan antar domain, *transfer learning* bertujuan untuk memanfaatkan pengetahuan dari domain terkait (disebut domain sumber) untuk meningkatkan kinerja pembelajaran atau mengurangi jumlah contoh berlabel yang diperlukan dalam domain target. Perlu dicatat bahwa pengetahuan yang ditransfer tidak selalu membawa dampak positif pada tugas-tugas baru. Jika ada sedikit kesamaan antara domain, transfer pengetahuan dapat gagal. Sebagai contoh, belajar mengendarai sepeda tidak dapat membantu penulis belajar bermain piano dengan lebih cepat. Selain itu, kesamaan antara domain tidak selalu memfasilitasi pembelajaran, karena terkadang kesamaan dapat menyesatkan. Misalnya, meskipun bahasa Spanyol dan bahasa Prancis memiliki hubungan erat satu sama lain dan keduanya termasuk dalam kelompok bahasa Roman, orang yang belajar bahasa Spanyol mungkin mengalami kesulitan dalam belajar bahasa Prancis, seperti menggunakan kosakata atau konjugasi yang salah. Ini terjadi karena pengalaman sukses sebelumnya dalam bahasa Spanyol dapat mengganggu pembelajaran pembentukan kata, penggunaan, pengucapan, konjugasi, dan sebagainya, dalam bahasa Prancis (Zhuang *dkk.*, 2021).

Transfer learning dapat menjadi berguna dalam aplikasi-aplikasi *CNN* di mana datasetnya tidak besar. Konsep dari *transfer learning* ditunjukkan

dalam Gambar 2.4, di mana model yang telah dilatih dari dataset besar, seperti *ImageNet*, dapat digunakan untuk aplikasi dengan dataset yang relatif lebih kecil (Rahman *dkk.*, 2020).



Gambar 2. 4 Concept of Transfer Learning

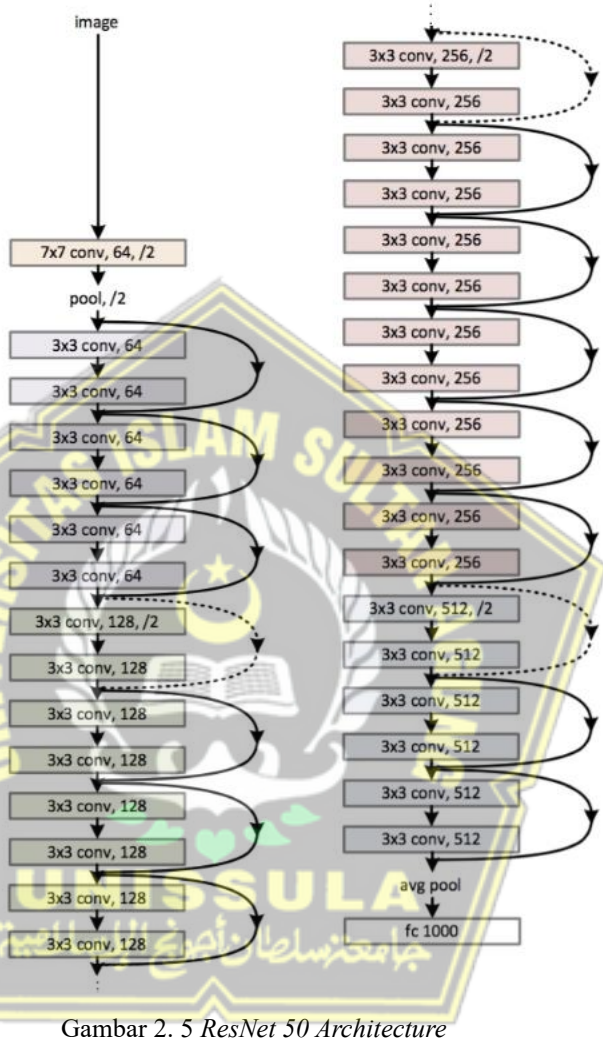
Baru-baru ini, metode *transfer learning* telah berhasil digunakan dalam berbagai aplikasi di berbagai bidang, seperti manufaktur, medis, dan pemeriksaan bagasi. Hal ini menghilangkan kebutuhan untuk memiliki dataset besar dan juga mengurangi periode pelatihan yang panjang, seperti yang dibutuhkan oleh algoritma deep learning saat dikembangkan dari awal (Rahman *dkk.*, 2020).

Pada penelitian ini, pendekatan transfer learning dilakukan dengan *feature extraction dan fine-tuning* pada arsitektur *ResNet-50*. Bagian awal *(convolutional base)* dari *ResNet-50* yang sudah dilatih pada dataset *ImageNet* dipertahankan sebagai ekstraktor fitur dengan freeze 100 layer pertama, sedangkan lapisan *fully-connected* pada bagian akhir dihapus dan diganti dengan lapisan klasifikasi baru (Dense layer dengan aktivasi sigmoid) untuk mendeteksi dua kelas (*Parasitized* dan *Uninfected*). *Fine-tuning* dilakukan pada beberapa lapisan terakhir dari *convolutional base* untuk menyesuaikan bobot agar lebih spesifik pada data malaria.

2.2.5 ResNet 50

ResNet-50 merupakan arsitektur jaringan saraf konvolusional yang terdiri dari 50 lapisan. *ResNet-50* dapat memuat versi jaringan *Pre Trained* pada lebih dari satu juta gambar dari basis data *ImageNet.* Jaringan yang

telah dilatih sebelumnya ini dapat mengklasifikasikan gambar ke dalam 1000 kategori objek, seperti keyboard, mouse, pensil, dan berbagai hewan (Mukti dan Biswas 2020)**.**



Gambar 2. 5 *ResNet 50 Architecture*

Pada gambar 2.5, Jaringan Residual *(ResNet*) membawa ide baru. Ini bermanfaat untuk memecahkan tugas yang kompleks dan juga meningkatkan akurasi deteksi. *ResNet* berusaha untuk mengatasi kesulitan dalam proses pelatihan jaringan *CNN* yang dalam, yaitu jenuhnya akurasi dan degradasi. *ResNet50* memiliki 50 lapisan jaringan residual. Gambar 2.4 menunjukkan arsitektur *ResNet50.*

Struktur *ResNet50* memiliki kelompok-kelompok lapisan identik yang berbeda, seperti yang ditunjukkan dengan warna yang berbeda di Gambar

2.4. Garis kurva mewakili blok identitas yang digunakan untuk menunjukkan penggunaan lapisan sebelumnya dalam lapisan berikutnya. Ini adalah perbedaan kunci dalam *ResNet50* yang menangani masalah gradien yang menghilang atau meledak, masalah degradasi (akurasi pertama jenuh dan kemudian menurun) dalam pelatihan jaringan yang sangat dalam.

Di Gambar 2.4, lapisan pertama memiliki 64 filter dengan ukuran kernel 7×7, yang diikuti oleh lapisan maxpooling dengan ukuran 3×3. Kelompok pertama dari lapisan (seperti yang ditunjukkan dengan warna abu-abu) terdiri dari tiga blok identik. Dengan cara yang sama, kelompok dua, kelompok tiga, dan kelompok empat memiliki 4 blok identik, 4 blok identik, dan 3 blok identik masing-masing. Di antara beberapa kelompok, kurva yang ditpenulisi dengan warna biru, mewakili blok identitas yang menghubungkan dua lapisan dengan ukuran yang berbeda.

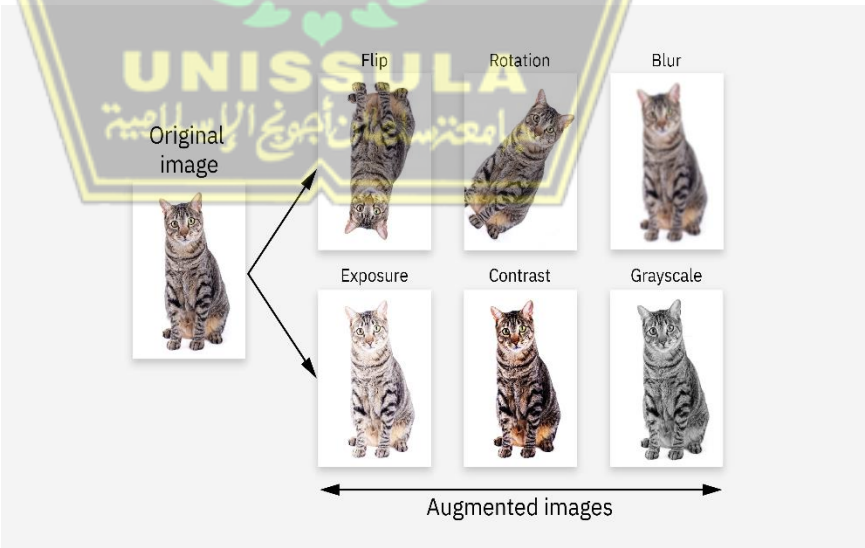
2.2.6 Keras Image Data Generator

Data augmentasi adalah teknik yang digunakan dalam pemrosesan gambar dan data lainnya di bidang *machine learning* dan *deep learning*. Tujuan dari data augmentasi adalah untuk meningkatkan keanekaragaman dataset dengan membuat variasi data baru dari data yang ada. Hal ini dilakukan dengan melakukan berbagai transformasi dan modifikasi pada data asli, seperti memutar, membalikkan, memperbesar, memperkecil, atau menambahkan derau, sehingga menghasilkan data baru yang memiliki variasi dalam tampilan dan sifatnya (Shanqing dkk., 2020).

Data augmentasi membantu mencegah overfitting, meningkatkan generalisasi model, dan meningkatkan performa model pada data uji yang belum pernah dilihat sebelumnya. Teknik augmentasi data menjadi sangat berguna ketika dataset terbatas atau tidak seimbang, karena dengan menghasilkan lebih banyak data dari data asli, model dapat mempelajari pola yang lebih banyak dan dapat menjadi lebih baik dalam tugas-tugas seperti klasifikasi, deteksi objek, dan segmentasi gambar (Shanqing dkk., 2020).

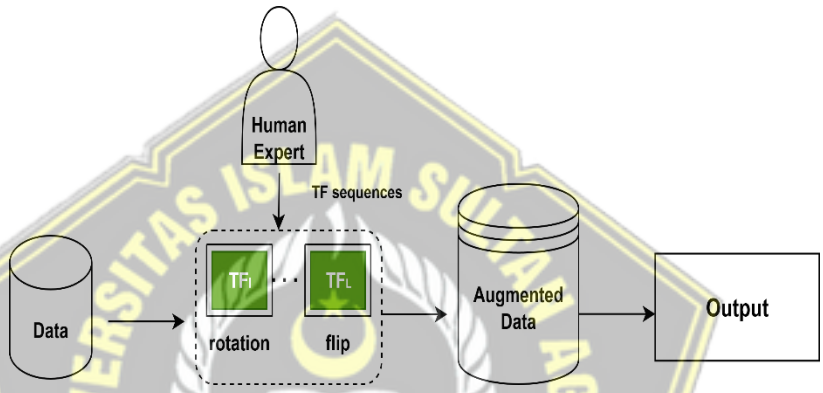
Untuk pra-pemrosesan gambar, sebuah gambar diwakili dengan sebuah array piksel menggunakan skala abu-abu atau nilai RGB. Pada gambar 2.6 Data gambar harus di-skalakan dengan normalisasi min-max untuk meningkatkan kecepatan pembelajaran. Fitur kategori warna dapat diubah menjadi vektor dari tiga nilai numerik dengan *one-hot encoding*. Augmentasi data merupakan salah satu teknik pra-pemrosesan gambar penting yang dapat dilakukan secara offline atau online. Teknik augmentasi offline digunakan untuk meningkatkan ukuran dataset kecil, sementara teknik augmentasi online biasanya diterapkan pada dataset besar (Shanqing dkk., 2020)**.**

Teknik umum untuk menghasilkan gambar baru adalah: (1) membalikkan secara horizontal atau vertikal; (2) memutar pada beberapa derajat; (3) memperbesar atau memperkecil skala; (4) memotong secara acak; (5) mentranslasi; (6) menambahkan derau Gaussian untuk mencegah overfitting dan meningkatkan kemampuan pembelajaran. Selain itu, juga ada metode augmentasi lanjutan dengan menggunakan jaringan *adversarial generatif kondisional* (GANs). Contoh-contoh representatif ditunjukkan dalam Gambar 2.4 (Shanqing et al., 2020)**.**



Gambar 2. 6 *Contoh Representatif Data Augmentasi*

Keras ImageDataGenerator adalah suatu kelas di *library* Keras, yang merupakan *library open-source* untuk deep learning yang dibangun di atas *framework TensorFlow*. *ImageDataGenerator* berfungsi sebagai alat untuk melakukan augmentasi data pada gambar dalam proses pelatihan model deep learning. Augmentasi data adalah teknik yang digunakan untuk menciptakan variasi dalam data pelatihan dengan melakukan transformasi pada gambar secara acak, seperti rotasi, pemotongan, pembalikan, dan zoom, untuk tahapannya bisa diliat pada gambar 2.7 (Shanqing dkk., 2020)**.**



Gambar 2. 7 *Augmentation Data Flow*

Penerapan *ImageDataGenerator* menjadi sangat berguna ketika dataset yang penulis miliki terbatas atau tidak seimbang, karena dengan menghasilkan lebih banyak variasi dari data asli, model dapat mempelajari pola yang lebih banyak dan menjadi lebih baik dalam menggeneralisasi pada data uji. Dengan demikian, *ImageDataGenerator* adalah alat yang penting dan efektif dalam memperkuat kualitas pelatihan model *deep learning* dan meningkatkan kinerja model pada tugas-tugas seperti klasifikasi gambar, deteksi objek, dan segmentasi gambar. Studi literatur tentang *ImageDataGenerator* akan memberikan wawasan yang mendalam tentang bagaimana teknik augmentasi data berkontribusi pada efektivitas dan efisiensi proses pelatihan model *deep learning* dalam pengolahan gambar (Shanqing dkk,, 2020)**.**

2.2.7 Split Ratio

Split ratio atau rasio pembagian adalah proporsi persentase data yang dialokasikan untuk setiap bagian utama dari dataset, yaitu training set dan

testing set. Rasio ini menentukan seberapa besar data yang digunakan untuk melatih model dibandingkan dengan data yang digunakan untuk menguji atau mengevaluasi model. Biasanya, *split ratio* dinyatakan dalam bentuk persentase, di mana jumlah total persentase harus sama dengan 100% (Thabtah *dkk.*, 2020)**.**

Rasio kelas yang dipertimbangkan adalah 10%:90%, 20%:80%, 30%:70%, 40%:60%, 50%:50%, 60%:40%, 70%:30%, 80%:20%, dan 90%:10%. Rasio-rasio ini dihasilkan berdasarkan pengacakan eksperimental dari benih acak yang berbeda untuk setiap dataset yang diberikan dengan stratifikasi. Rasio pembagian harus dipilih dengan hati-hati berdasarkan ukuran total dataset dan karakteristiknya. Sebagai aturan umum, semakin besar *training set*, semakin baik model dapat belajar dan beradaptasi dengan pola dalam data. Namun, *testing set* yang mencakup proporsi yang cukup besar dari dataset juga penting untuk mengukur kemampuan model dalam menggeneralisasi pada data baru yang belum pernah dilihat sebelumnya. Terlebih lagi, pemilihan *training set* dan *testing set* juga harus menyesuaikan kepada limitasi komputasi pada penelitian (Thabtah *dkk.*, 2020)**.**

2.2.8 Binary Cross-Entropy Loss

Binary Cross Entropy Loss (BCE Loss) atau juga dikenal sebagai *Log Loss*, adalah fungsi kerugian yang digunakan dalam tugas klasifikasi biner pada *machine learning* dan *deep learning*. Fungsi ini digunakan untuk mengukur perbedaan antara probabilitas yang diprediksi oleh model dengan label kelas yang sebenarnya pada masalah klasifikasi biner (Ho dan Wookey, 2020)**.**

Misalkan penulis memiliki dataset pelatihan yang terdiri dari pasangan input (x) dan label kelas yang sebenarnya (y). Pada tugas klasifikasi biner, label kelas y akan bernilai 0 atau 1, yang mewakili kelas negatif dan positif. Model *deep learning* yang digunakan akan menghasilkan probabilitas prediksi (p) untuk kelas positif (1) pada setiap sampel data (Ho dan Wookey, 2020).

BCE Loss dihitung dengan menggunakan fungsi logaritma natural (ln) dari probabilitas prediksi (p) untuk kelas positif (y = 1) ketika label sebenarnya adalah 1, dan logaritma natural dari (1 - p) untuk kelas negatif (y = 0) ketika label sebenarnya adalah 0. Fungsi ini menggambarkan perbedaan antara probabilitas prediksi dan nilai yang sebenarnya (Ho dan Wookey, 2020).

Secara Formal, BCE Loss didefinisikan sebagai berikut :

$$BCE\ Loss = -\ (\ y * log(p) + (1 - y) * log(1 - p)\ ) \quad \textbf{(3)}$$

Di mana :

- y adalah label kelas yang sebenarnya (0 atau 1).
- p adalah probabilitas prediksi untuk kelas positif (1).
- 1 - p adalah probabilitas prediksi untuk kelas negatif (0).

Fungsi BCE Loss sangat penting dalam tugas-tugas seperti deteksi objek, klasifikasi medis, dan identifikasi spam email. Penggunaannya dalam kombinasi dengan fungsi aktivasi Sigmoid pada lapisan output, memungkinkan model untuk menghasilkan probabilitas yang tepat dan meningkatkan performa klasifikasi pada masalah biner. Studi literatur tentang BCE Loss menyediakan pemahaman mendalam tentang peran dan kegunaannya dalam pengembangan model deep learning untuk tugas klasifikasi biner (Ho dan Wookey, 2020).

2.2.9 Confussion Matrix

Confusion matrix adalah alat evaluasi yang umum digunakan dalam pemodelan klasifikasi pada machine learning. Matriks ini memberikan gambaran komprehensif tentang kinerja model dengan membandingkan hasil prediksi model dengan keadaaan sebenarnya dari data yang diuji. *Confusion matrix* umumnya digunakan untuk tugas klasifikasi biner (dua kelas), tetapi dapat diadaptasi untuk tugas klasifikasi multikelas, bisa diliat seperti tabel 2.2 (Zeng, 2020).

Tabel 2. 2 Confusion Matrix

	Predicted Class	
True Class	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Confusion Matrix terdiri dari empat sel utama :

1. *True Positive (TP)*

   Ini adalah jumlah data yang secara benar diprediksi sebagai positif oleh model. Dalam konteks klasifikasi, ini berarti model dengan benar mengidentifikasi contoh yang termasuk dalam kelas positif.

2. *True Negative (TN)*

   Ini adalah jumlah data yang secara benar diprediksi sebagai negatif oleh model. Model dengan benar mengidentifikasi contoh yang termasuk dalam kelas negative.

3. *False Positive (FP)*

   Juga dikenal sebagai Type I Error, ini adalah jumlah data yang keliru diprediksi sebagai positif oleh model, padahal seharusnya termasuk dalam kelas negative.

4. *False Negative (FN)*

   Juga dikenal sebagai Type II Error, ini adalah jumlah data yang keliru diprediksi sebagai negatif oleh model, padahal seharusnya termasuk dalam kelas positif

2.2.10 *Classification Report*

Classification report adalah suatu bentuk ringkasan yang memberikan informasi rinci tentang kinerja model klasifikasi. Biasanya, classification report mencakup metrik-metrik evaluasi utama seperti presisi (*precision*), *recall* (sensitivitas), *F1-score*, dan dukungan (*support*) untuk setiap kelas dalam tugas klasifikasi (Adoma dkk., 2020)**.**

1. Accuracy

Akurasi mengukur seberapa tepat model secara keseluruhan. Dihitung dengan rumus:

$$Accuracy = \frac{(TP + TN)}{TP + FP + TN + FN} \quad \textbf{(4)}$$

2. *Precision*

Presisi mengukur seberapa akurat model dalam memprediksi suatu kelas tertentu. Dihitung dengan rumus:

$$Precision = \frac{TP}{(TP + FP)} \quad \textbf{(5)}$$

3. *Recall*

Recall mengukur sejauh mana model dapat menangkap semua contoh positif dalam dataset. Dihitung dengan rumus:

$$Recall = \frac{TP}{(TP + FN)} \quad \textbf{(6)}$$

4. *F1-Score*

F1-score adalah *harmonic mean* dari *presisi* dan *recall*. Berguna ketika kita perlu mencapai keseimbangan antara *presisi* dan *recall*. Dihitung dengan rumus:

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad \textbf{(7)}$$

2.2.11 Fungsi Aktivasi

Fungsi aktivasi adalah fungsi *non linear* yang memungkinkan sebuah *neural network* untuk dapat mentransformasikan data input menjadi dimensi yang lebih tinggi sehingga dapat dilakukan *hyperlane* sederhana yang memungkinkannya dilakukan klasifikasi (Tilasefana dan Putra, 2023)**.**

Setiap fungsi aktivasi mengambil sebuah nilai dan melakukan operasi matematika. Pada arsitektur CNN, fungsi aktivasi terletak pada perhitungan akhir output *feature map* atau sesudah proses operasi *convolution* atau *pooling* untuk menghasilkan suatu pola fitur (Zufar, 2020)**.**

Beberapa macam fungsi aktivasi yang sering digunakan dalam penelitian CNN antara lain fungsi sigmoid, tanh, *Rectified Liniear Unit* (ReLU), *Leaky* ReLU (LReLU), SGD dan SoftMax.

2.2.12 Learning Rate

Learning rate adalah salah satu *hyper parameter* yang sangat mempengaruhi performa suatu model CNN. Dalam melakukan pencarian *learning rate* yang tepat, digunakan suatu metode yang bernama *cyclical learning rates* (Wu dan Liu, 2023)**.**

Dalam metode ini, akan dilakukan beberapa *epochs training*, dengan *learning rate* yang dimulai dari nilai yang kecil, dan setiap iterasi akan diperbesar *learning rate*-nya, setiap iterasi akan dilihat *loss* yang didapat, dan jika *loss* yang didapat meningkat cukup drastis maka proses pencarian akan diberhentikan. Berdasarkan data *loss* yang didapat, maka dipilih *learning rate* sesaat sebelum *loss* memiliki nilai terendah (Subramanian, Ganapathiraman dan Barrett, 2024).

2.2.13 Epoch

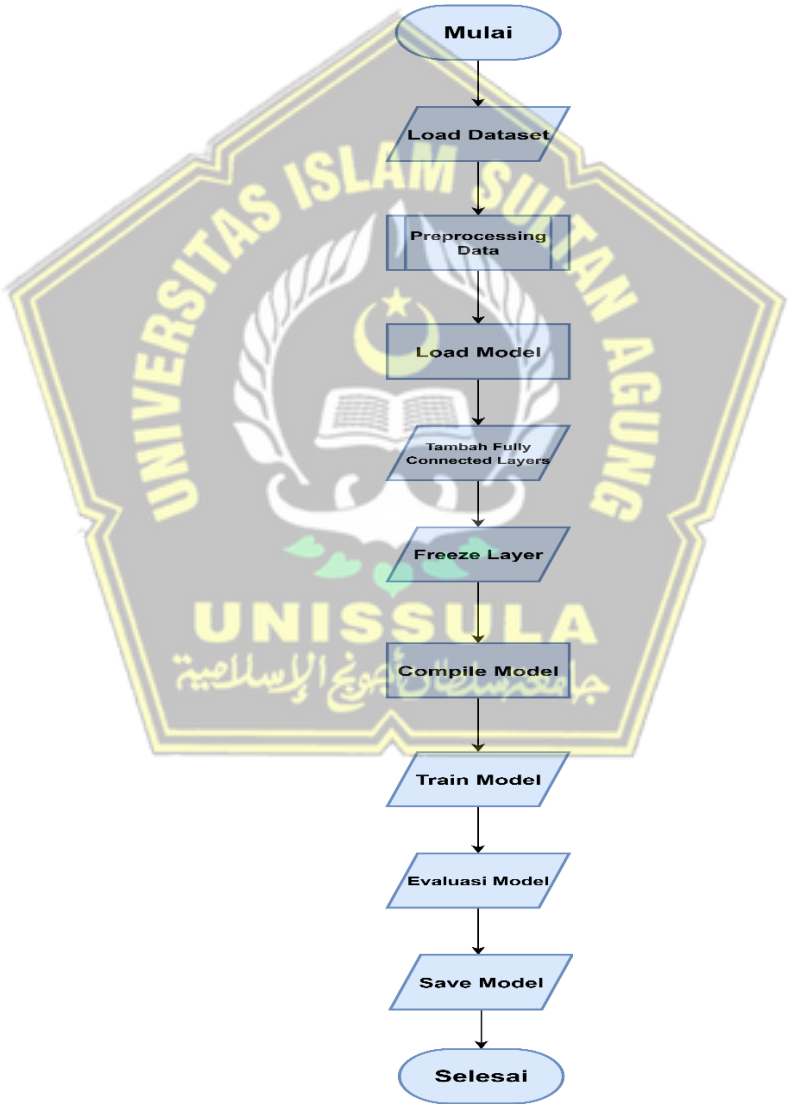
Epoch adalah ketika seluruh dataset sudah melalui proses training pada *Neural Netwok* sampai dikembalikan ke awal untuk sekali putaran, karena satu Epoch terlalu besar untuk dimasukkan (*feeding*) kedalam komputer maka dari itu kita perlu membaginya kedalam satuan kecil (*batches*). Didalam epoch inilah tiap bobot/ *weight* ditambahkan (Ramadhan dan Riwurohi, 2024).

Jumlah epoch dan ukuran batch merupakan hyperparameter penting yang berpengaruh terhadap kinerja dan akurasi model. Semakin banyak epoch yang digunakan, model akan semakin sering mempelajari data, namun penggunaan epoch yang terlalu besar dapat menyebabkan *overfitting*. Sebaliknya, batch size menentukan seberapa banyak data yang diproses dalam satu iterasi, sehingga batch size yang terlalu kecil dapat memperlambat pelatihan, sedangkan batch size yang terlalu besar dapat menurunkan kemampuan generalisasi model, dengan itu menunjukkan bahwa pemilihan kombinasi epoch, batch size, dan optimizer yang tepat mampu meningkatkan performa model secara signifikan (Ramadhan dan Riwurohi, 2024).

BAB III
METODE PENELITIAN

3.1 Metode Pengumpulan Data

Pada penelitian ini penulis mengumpulkan informasi dan data yang dapat menjadi penunjang dan referensi penulis dalam melakukan penelitian. Berikut merupakan proses dan pengumpulan data ditunjukkan pada gambar flowchart 3.1.



Gambar 3. 1 Flowchart diagram alir model deteksi penyakit malaria

3.1.1 Studi Literatur

Salah satu metode pengumpulan data yang penulis lakukan pada pembuatan sistem ini adalah studi pustaka. Studi pustaka dilakukan dengan cara membaca, mengutip, mengolah informasi yang berkaitan dengan pembuatan sistem, serta membuat catatan yang bersumber pada bahan-bahan pustaka yang mendukung dan berkaitan dengan pembuatan sistem ini. Sumber-sumber yang penulis kumpulkan dan pelajari berupa data dan informasi definisi yang terdapat pada beberapa jurnal dan buku yang terkait dengan penelitian dan pembuatan sistem ini serta dari beberapa situs online yang terpercaya. Sumber-sumber tersebut diperlukan penulis untuk membantu penyelesaian pada bab pendahuluan, landasan teori dan metode penelitian.

3.1.2 Pengumpulan Data

Dataset yang digunakan adalah dataset gambar sel darah yang sudah dilabeli (terinfeksi / tidak terinfeksi:

- Total Gambar 27.558
- Kelas :
  - Parasitized (terinfeksi malaria).
  - Uninfected (tidak terinfeksi).

Data yang digunakan dalam penelitian ini berasal dari **NIH Malaria Dataset** (National Institutes of Health, USA) yang berisi 27.558 citra sediaan darah tipis berwarna Giemsa, terdiri dari dua kelas yaitu parasit terinfeksi (*parasitized*) dan tidak terinfeksi (*uninfected*). Dataset ini merupakan **dataset sekunder** yang bersifat open-access. Setiap citra telah diberi label oleh para ahli mikroskopis. Dataset ini sudah dipublikasikan di jurnal PeerJ (Rajaraman dkk., 2018) dan sudah dipakai di banyak penelitian internasional. Dataset ini sangat kredibel karena mengikuti prosedur laboratorium dengan pewarnaan Giemsa dan verifikasi ahli. Selain itu untuk sediaan darah sampel dataset ini diambil dari Bangladesh, diambil dari 150 pasien terinfeksi dan 50 pasien sehat.

3.1.3 *Preprocessing Data*

Tahapan ini bertujuan menyiapkan data agar kompatibel dengan arsitektur ResNet50 dan memperbaiki kualitas data input :

- Resize Gambar : Ukuran gambar disesuaikan menjadi 224x224 piksel, sesuai input standar ResNet50.
- Normalisasi piksel : Nilai piksel dikonversi ke skala [0–1] dengan membagi dengan 255 untuk mempercepat konvergensi saat training.
- Augementasi Gambar :
  - Rotasi acak (0–30°).
  - Flip horizontal/vertical.
  - Zoom.
  - Shear

3.1.4 Split Data

Dataset dibagi menjadi :

- Data Training 80%.
- Data Validation 10%.
- Data Testing 10%.

3.1.5 Load Arsitektur Resnet-50

- Load Resnet-50
  - Menggunakan pretrained model ResNet50 dari pustaka seperti TensorFlow atau Keras.
  - Opsi include_top=False digunakan agar bagian klasifikasi bawaan ImageNet dihapus.
- Freeze Layer
  - Layer awal dari ResNet50 dapat “dibekukan” (non-trainable) agar tidak dilatih ulang.
  - Hanya layer akhir (fully connected) yang dilatih ulang pada dataset malaria.
- Tambahkan Custom Classification Layer
  Flatten → Dense(128) → Dropout → Dense(1, activation='sigmoid') untuk klasifikasi biner.

3.1.6 Compile Model

Model dikompilasi dengan parameter berikut :

- Loss Function: binary_crossentropy.
- Optimizer: Adam (dengan learning rate 1e-4 atau disesuaikan).
- Metrics: accuracy, precision, recall

3.1.7 Training Model

- Model dilatih selama sejumlah epoch (misalnya: 10–50).
- Selama training, nilai loss dan akurasi pada training dan validation set dicatat.

3.1.8 Evaluasi Model

Setelah proses pelatihan model selesai, tahap evaluasi model dilakukan untuk mengukur kinerja dan generalisasi model dalam mendeteksi klasifikasi malaria. Parameter evaluasi utama yang digunakan meliputi nilai akurasi, *loss* (fungsi kerugian), serta analisis lebih mendalam melalui *confusion matrix* dan *classification report*. *Confusion matrix* digunakan untuk melihat distribusi prediksi model terhadap kelas yang sebenarnya, sehingga dapat dianalisis sejauh mana model berhasil mengklasifikasikan. Selain itu, *classification report* menghasilkan metrik seperti *precision*, *recall*, dan *F1-score* untuk masing-masing kelas, yang sangat penting untuk memahami kekuatan dan kelemahan model terhadap setiap kategori.

Untuk mengukur *Acuraccy, Precision, Recall*, dan *F1-Score* berdasarkan data *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN), digunakan metode *confusion matrix*. Persamaan pertama digunakan untuk menghitung permutasi *acuraccy*, persamaan kedua untuk menghitung permutasi *precision*, persamaan ketiga untuk menghitung permutasi *recall,* dan persamaan keempat untuk menghitung permutasi *F1-Score* dengan lebih dari dua kelas keluaran.

$$\text{Acuraccy} = \frac{TP+TN}{TP+FP+FN+TN} \text{X } 100\% \quad (1)$$

$$\text{Precision} = \frac{TP}{TP+FP} \text{X } 100\% \quad (2)$$

$$\text{Recall} = \frac{TP}{TP+FN} \text{ X } 100\% \quad (3)$$

$$\text{F1-Score} = \frac{2 \text{ x } Precision \text{ x } Recall}{Precision+Recall} \text{ X } 100\% \quad (4)$$

Keterangan :

TP : Jumlah data *actual* yang sebenarnya benar dan diprediksi benar

TN : Jumlah data *actual* yang sebenarnya salah dan diprediksi salah

FP : Jumlah data *actual* yang sebenarnya benar dan diprediksi salah

FN : Jumlah data *actual* yang sebenarnya salah dan diprediksi benar

Sedangkan tiga jenis *Confusion matrix* yang digunakan dalam menghitung prediksi antara lain :

1. *Precision* adalah rasio prediksi benar dengan jumlah hasil prediksi positif dan negatif.
2. *Recall* adalah rasio prediksi benar dengan jumlah nilai aktual positif atau negatif.
3. *F1 Score* adalah perbandingan rata – rata *precision* dan *recall* yang dibobotkan.

3.1.9 *Deployment* Model

Setelah model CNN berbasis *EfficientNetB3* berhasil dilatih dan di evaluasi, tahap selanjutnya adalah *deployment model* ke dalam bentuk aplikasi interaktif berbasis web. Proses ini dilakukan menggunakan *framework* Streamlit, yang memungkinkan integrasi antara antarmuka pengguna dan logika prediksi model secara efisien. Aplikasi dibangun sedemikian rupa agar pengguna dapat mengunggah citra sel darah terinfeksi dan non infeksi malaria, yang kemudian diproses dan dianalisis oleh model untuk mengidentifikasi sel darah manusia terinfeksi atau tidak. Gambar yang diunggah akan melalui proses *preprocessing* serupa seperti pada tahap pelatihan model, kemudian hasil prediksi ditampilkan secara *real-time.* Hal ini bertujuan untuk meningkatkan transparansi dan kepercayaan pengguna terhadap sistem yang dikembangkan. Seluruh proses implementasi aplikasi dilakukan secara modular, sehingga mudah dipelihara dan dapat dikembangkan lebih lanjut untuk kebutuhan diagnosis awal malaria.

3.2 *Software* Yang Digunakan

1. Bahasa Pemrograman
   a. Python

      Python merupakan bahasa pemrograman utama yang digunakan dalam pengembangan sistem ini. Python dipilih karena memiliki sintaks yang sederhana dan didukung oleh berbagai pustaka *Deep Learning* yang sangat powerful. Dengan fitur-fitur seperti kemudahan dalam pemrograman berorientasi objek, dukungan untuk pengolahan data, serta kemampuan untuk bekerja dengan data besar, Python adalah bahasa yang sangat ideal untuk pengembangan sistem berbasis *Deep Learning*.

2. Perangkat Lunak (*software)*
   a. *Google Colab*

      Google Colaboratory (Colab) dimanfaatkan sebagai lingkungan pemrograman interaktif yang berbasis *cloud* untuk menulis, menjalankan, dan menguji kode secara langsung. *Platform* ini sangat mendukung proses pengembangan dan eksperimen dalam penelitian karena memungkinkan integrasi dengan berbagai pustaka Python serta 32 penyimpanan di *Google Drive*. Dengan antarmuka yang ramah pengguna dan dukungan terhadap eksekusi berbasis GPU, *Colab* mempermudah peneliti dalam memodifikasi kode, mencatat hasil eksperimen, serta mendokumentasikan proses pengujian secara terstruktur dalam satu dokumen yang interaktif.

   b. *Visual Studio Code*

      Visual Studio Code dipilih sebagai text editor pada pengembangan aplikasi dalam penelitian ini, *Visual Studio Code* dipilih dikarenakan mendukung banyak bahasa pemrograman dan *framework*, multi platform, performa yang sangat cepat, dan mempunyai banyak *extensions* yang dapat mempermudah proses pengembangan *website*.

   c. *Streamlit*

Streamlit adalah *framework Python* yang sangat berguna untuk membangun aplikasi web interaktif secara cepat, terutama dalam proyek *machine learning* dan *deep learning* seperti CNN (*Convolutional Neural Network*). Dengan menggunakan *EfficientNet* untuk klasifikasi gambar, *Streamlit* memungkinkan untuk membuat antarmuka sederhana.

3. *Library* dan *Framework*

   a. *TensorFlow/Keras*

      TensorFlow merupakan *framework* utama untuk *deep learning* dalam program ini. Submodul Keras dari *TensorFlow* digunakan untuk membangun dan melatih model CNN. Misalnya, *EfficientNetB3* digunakan sebagai model dasar *pre-trained*, lalu ditambahkan layer tambahan untuk klasifikasi. Komponen seperti *ImageDataGenerator, ModelCheckpoint, EarlyStopping,* dan *ReduceLROnPlateau* sangat penting dalam pelatihan karena membantu memperkuat performa model dan mencegah *overfitting*.

   b. *Pandas*

      Pandas digunakan untuk memuat dan memproses file CSV (train.csv) yang berisi data label gambar. *Library* ini juga digunakan untuk manipulasi *DataFrame*, seperti menggabungkan *path* gambar dengan nama *file* dan menangani *undersampling* data agar distribusi kelas menjadi seimbang.

   c. *OpenCV*

      OpenCV digunakan untuk membaca, menampilkan, dan memproses gambar. Fungsinya termasuk membaca gambar dari direktori, mengubah format warna, dan melakukan *resize* sebelum digunakan untuk visualisasi.

   d. *NumPy*

      NumPy digunakan untuk operasi *array* numerik, termasuk normalisasi dan konversi label. Selain itu, *library* ini juga berperan

dalam pengolahan label untuk menghitung *class weight* agar pelatihan model lebih adil terhadap kelas minoritas.

e. *Matplotlib & Seaborn*

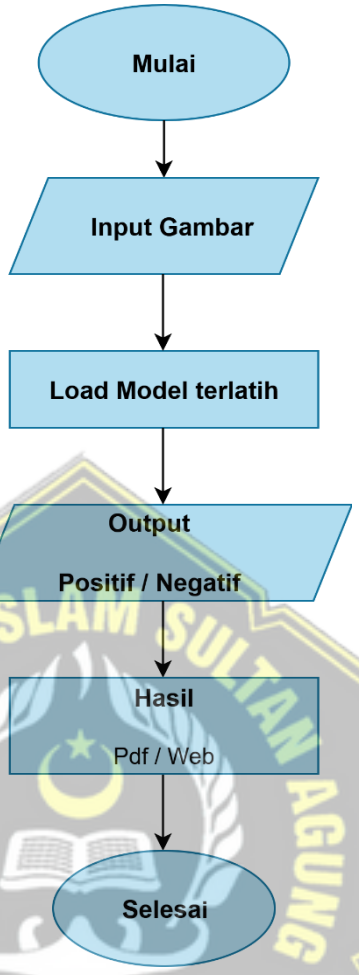
Kedua *library* visualisasi ini digunakan untuk menampilkan berbagai grafik, termasuk distribusi kelas (dengan *Seaborn*), contoh gambar tiap kelas, grafik akurasi dan *loss* selama pelatihan, serta *confusion matrix* dan visualisasi prediksi hasil model.

f. *Scikit-learn*

Library ini digunakan dalam beberapa bagian penting, seperti `train_test_split` untuk membagi data, `resample` untuk *undersampling*, `compute_class_weight` untuk menangani ketidakseimbangan kelas, dan `classification_report` serta `confusion_matrix` untuk mengevaluasi performa model.

3.3 Perancangan sistem

Pada tahap ini akan menentukan alur kerja system berupa *flowchart* yang akan memberikan gambaran dari alur sistem nantinya. Di mana metode ini nantinya akan melakukan klasifikasi Malaria, dan ada beberapa tahapan yang perlu dilakukan dalam penellitian ini, seperti yang terlihat pada gambar 3.2 :



Gambar 3. 2 Flowchart Implementasi Model

Gambar 3.2 merupakan *flowchart* streamlit yang menerangkan alur kerja aplikasi web dalam mengklasifikasikan gambar sel darah yang terinfeksi/tidak. Tahapan alur kerja streamlit tersebut dijelaskan sebagai berikut :

a) Input Gambar
   - Sediaan darah (thin/thick smear) diwarnai dengan Giemsa seperti prosedur laboratorium standar.
   - Mikroskop yang dilengkapi kamera (bisa kamera smartphone dengan adaptor atau kamera digital mikroskop) digunakan untuk mengambil foto slide darah.
   - Foto diunggah ke sistem (aplikasi web/mobile atau lokal PC.

b) Load Model

- Foto yang masuk akan melewati CNN ResNet-50 untuk analisis.
- Model menghasilkan prediksi: *terinfeksi* atau *tidak terinfeksi*, dan bisa juga mengklasifikasi jenis parasit jika dilatih multikelas.

c) Output

Hasil Dapat berupa: Deteksi positif/negatif malaria.

d) Hasil Penyimpanan

- Data hasil analisis disimpan dalam database.
- Hasil bisa diunduh dalam bentuk laporan PDF (gambar + hasil prediksi) ataupun bisa diunduh dalam bentuk file csv.
- Laporan bisa dibagikan untuk telemedicine atau arsip laboratorium.

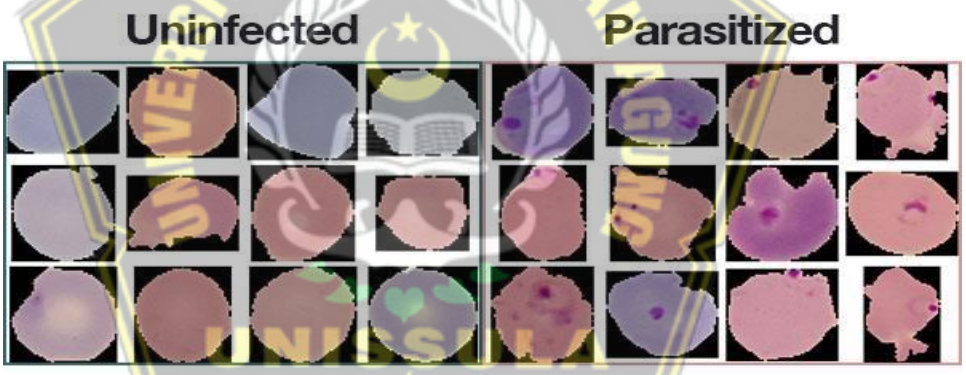
e) Validasi Tenaga Medis

- Hasil model tetap perlu diverifikasi oleh analis laboratorium atau dokter sebelum menjadi hasil diagnosis final.
- Model di sini berfungsi sebagai pendukung keputusan (*decision support*).

BAB IV
HASIL DAN ANALISIS PENELITIAN

4.1 Hasil Pengumpulan Data

Data yang digunakan dalam penelitian ini merupakan dataset citra mikroskopis sel darah merah yang terdiri dari dua kelas, yaitu *Parasitized* (sel darah yang terinfeksi malaria) dan *Uninfected* (sel darah yang sehat/tidak terinfeksi). Dataset ini diperoleh dari sumber publik dan telah melalui proses ekstraksi serta pengelompokan ulang agar sesuai dengan kebutuhan penelitian. Sebelum dilakukan proses pelatihan model, data terlebih dahulu disusun dalam struktur direktori yang terorganisir, sehingga dapat dengan mudah diolah menggunakan *data generator*. Gambar berikut menampilkan contoh data yang digunakan dalam penelitian ini.



Gambar 4. 1 Contoh citra dari masing-masing kelas

Pada gambar 4.1 berikut merupakan dataset citra mikroskopis sel darah merah manusia yang terdiri dari sel darah terinfeksi dan sel darah sehat. Pengumpulan data dan informasi yang penulis lakukan yaitu dengan menggunakan studi pustaka dan mendapatkan hasil berupa *dataset* yang diambil dari *National Library of Medicine* yang dikembangkan oleh Stefan Jaeger pada tahun 2018 ftp://lhcftp.nlm.nih.gov/Open-Access-Datasets/Malaria/cell_images.zip yang dapat diakses bebas untuk kepentingan pengembangan aplikasi dan pembelajaran. Dataset ini nantinya akan digunakan sebagai data input sistem prediksi parasit malaria.

Dataset terdiri dari 27.558 gambar sel yang terbagi kedalam dua kategori yaitu *parasitized* dan *uninfected*. Setiap kategori terdiri dari 13.779 gambar sel, dataset yang telah diunduh akan dimasukan ke dalam Google Drive milik penulis yang nantinya akan dilakukan *pulling data* dengan *python* pada Google Colab.

kode "drive.mount('/content/gdrive')" merupakan langkah awal untuk menghubungkan Google Drive dengan *environment* Google Colab. Dengan menjalankan kode ini, sistem akan meminta izin untuk mengakses Google Drive pengguna. Setelah izin diberikan, Google Drive dapat diakses dan digunakan secara langsung dari dalam Google Colab. Hal ini memungkinkan pengguna untuk menyimpan dan memuat data, model, atau *file* lainnya secara langsung dari Google Drive ke lingkungan kerja Google Colab, serta sebaliknya. Dengan demikian, kode ini membuka kemungkinan kolaborasi yang lebih baik antara *platform* Google Drive dan Google Colab dalam mengelola dan menganalisis data serta menjalankan kode.

untuk mengekstrak file ZIP yang berisi dataset dari Google Drive ke dalam environment Google Colab. Terlebih dahulu, jalur (path) menuju file dataset dalam format ZIP yang berada di Google Drive didefinisikan dalam variabel zip_path. Kemudian jalur tujuan tempat ekstraksi dataset didefinisikan dalam variabel extract_path. Langkah selanjutnya adalah melakukan pengecekan apakah folder tujuan sudah ada dengan fungsi os.path.exists(). Jika folder belum ada, maka file ZIP dibuka menggunakan modul zipfile dan seluruh isinya diekstrak ke dalam folder extract_path menggunakan fungsi extractall(). Jika folder tujuan sudah tersedia, proses ekstraksi dilewati. Terakhir, kode menampilkan pesan status proses, apakah dataset berhasil diekstrak atau dilewati. Dengan demikian, dataset yang sebelumnya tersimpan di Google Drive dapat diakses dan digunakan di dalam lingkungan Google Colab dalam format yang sudah siap untuk tahap pemrosesan berikutnya.

4.2 Data Preparation

Tahap preprocessing dilakukan sebelum citra dimasukkan ke dalam model, untuk memastikan konsistensi bentuk data, pengurangan noise yang tidak diinginkan, serta menyesuaikan citra agar nilai pixelnya terstandarisasi, preprocessing membantu memperjelas informasi penting yang akan dipelajari model untuk melakukan pelatihan.

Selanjutnya, untuk membagi dataset gambar sel darah malaria menjadi tiga subset, yaitu data pelatihan (*training*), validasi (*validation*), dan pengujian (*testing*) dengan rasio 80% : 10% : 10%. Terlebih dahulu, jalur (path) menuju folder dataset asli didefinisikan dalam variabel input_folder, sedangkan jalur folder tujuan hasil pembagian dataset didefinisikan dalam variabel output_dir. Proses pembagian dataset dilakukan menggunakan fungsi splitfolders.ratio() dari library Python split-folders dengan parameter seed=42 untuk memastikan proses pembagian dapat direproduksi secara konsisten. Metode ini membagi data secara *stratified per kelas*, sehingga proporsi jumlah sampel di setiap kelas tetap seimbang pada ketiga subset. Hasil pembagian disimpan di dalam folder /content/malaria_split_80 yang secara otomatis memiliki subdirektori train/, val/, dan test/, masing-masing berisi subfolder Parasitized/ dan Uninfected/. Jumlah sampel pada setiap subset kemudian dicatat untuk digunakan pada tahap pelatihan dan evaluasi model.

Pada penentuan rasio pembagian dataset ini, untuk rasio pembagian data training dan data testing menggunakan metode *Simple Hold-Out Validation* (Chollet, 2020), metode ini menjadi acuan standar dalam pembagian data training dan testing dengan rasio 80:20 dan untuk rasio pembagian data training dan data validation berdasarkan penelitian (Akromunnisa dkk., 2020) yang membandingkan antara nilai split 6:4, 7:3, 8:2 dan 9:1. Dari penelitian tersebut disimpulkan bahwa rasio 9:1 menghasilkan nilai akurasi yang lebih besar dari rasio yang lainnya.

untuk menghitung dan menampilkan jumlah gambar pada setiap kelas di subset *train*, *val*, dan *test,* proses dilakukan dengan membaca isi folder

tiap subset, menghitung jumlah file pada masing-masing kelas, lalu menampilkannya ke layar. Langkah ini berfungsi sebagai verifikasi bahwa pembagian dataset telah sesuai dengan rasio yang ditentukan

Pembagian dataset menggunakan *split data* dengan persentase 80% untuk *data training,* 10 % untuk *data validation* dan *data testing* 10%.

Semua data pada folder *training, testing* dan *validation* dilabelkan dengan dua kategori yaitu "0" untuk *parasitized* dan "1" untuk *uninfected.*

Berikut adalah sebaran data yang terdapat pada folder training, validation dan testing :

Tabel 4. 1 Sebaran Data

No.	Nama folder	Kategori		Jumlah data
		Parasitized	Uninfected	
1.	cell_images(dataset)	13779	13779	27558
2.	training	11024	11024	22048
3.	validation	1378	1378	2756
4.	testing	1378	1378	2756

Pada tabel 4.1 dapat diliat bahwa Dataset yang digunakan pada penelitian ini terdiri dari 27.558 citra sel darah yang diambil dari *Cell Images Dataset*. Data ini terbagi secara seimbang menjadi dua kelas, yaitu Parasitized (sel darah yang terinfeksi malaria) dan Uninfected (sel darah yang tidak terinfeksi malaria), masing-masing sebanyak 13.779 citra.

Sebelum dilakukan proses pelatihan, dataset dibagi menjadi tiga subset utama, yaitu training set, validation set, dan testing set. Pembagian ini dilakukan secara proporsional sehingga distribusi jumlah citra pada kedua kelas tetap seimbang di setiap subset. Training set digunakan untuk melatih model agar dapat mempelajari pola pada data, validation set digunakan untuk memantau dan mengevaluasi performa model selama proses pelatihan, sedangkan testing set digunakan untuk mengukur kemampuan generalisasi model terhadap data baru yang belum pernah dilihat sebelumnya.

Penjelasan rincian jumlah data pada masing-masing subset adalah sebagai berikut :

- Training set berjumlah 22.048 citra, terdiri dari 11.024 citra kelas *Parasitized* dan 11.024 citra kelas *Uninfected*. Data pada subset ini akan mengalami proses augmentasi untuk memperluas variasi data pelatihan.
- Validation set berjumlah 2.756 citra, terdiri dari 1.378 citra kelas *Parasitized* dan 1.378 citra kelas *Uninfected*. Data ini digunakan sebagai pembanding selama pelatihan untuk mencegah overfitting.
- Testing set berjumlah 2.756 citra, terdiri dari 1.378 citra kelas *Parasitized* dan 1.378 citra kelas *Uninfected*. Data ini digunakan pada tahap akhir penelitian untuk mengevaluasi kinerja model secara objektif.

Dengan pembagian seperti ini, penelitian dapat memastikan bahwa proses pelatihan, validasi, dan pengujian dilakukan secara adil dan representatif, sehingga hasil yang diperoleh dapat dipercaya dan memiliki kemampuan generalisasi yang baik.

4.3 Augmentasi Data

Pada tahap ini, data yang telah dibagi menjadi *folder Train, validation*, dan *test* diproses menggunakan modul Keras ImageDataGenerator Untuk data latih (*train*) dilakukan proses augmentasi guna memperbanyak variasi data dan mengurangi risiko *overfitting*.

Augmentasi yang diterapkan meliputi :

- Normalisasi nilai piksel ke rentang 0–1 dengan rescale=1./255
- Rotasi acak hingga 20 derajat.
- Zoom acak hingga 20%**.**
- Pergeseran *horizontal* dan *vertikal* hingga 20%.
- Pembalikan (flip) gambar secara *horizontal*.

Untuk data validasi (val) dan pengujian (*test*), hanya dilakukan normalisasi piksel (rescale=1./255) tanpa augmentasi lain. Hal ini dilakukan agar data validasi dan pengujian tetap merepresentasikan data asli yang akan digunakan untuk mengevaluasi performa model.

Seluruh gambar dibaca dalam format RGB secara default, diubah ukurannya menjadi 224×224 piksel, dan diberi label secara otomatis berdasarkan nama folder kelas (*Parasitized* dan *Uninfected).*

untuk membentuk *data generator* yang akan memuat dan memproses dataset gambar malaria sebelum digunakan pada proses pelatihan model. Proses ini memanfaatkan kelas ImageDataGenerator dari library Keras.

Terlebih dahulu, dilakukan inisialisasi *data generator* untuk data pelatihan melalui variabel train_datagen. Pada generator ini diterapkan *data augmentation* berupa normalisasi nilai piksel ke rentang 0–1 (rescale=1./255), rotasi acak hingga 20 derajat, *zoom* acak sebesar 20%, pergeseran lebar (*width shift*) dan tinggi (*height shift*) masing-masing hingga 20%, serta pembalikan horizontal (*horizontal flip*). Tujuan augmentasi ini adalah memperkaya variasi data latih dan mengurangi risiko *overfitting.*

Untuk data validasi dan pengujian, digunakan generator val_test_datagen yang hanya melakukan normalisasi piksel (rescale=1./255) tanpa augmentasi, agar data tetap merepresentasikan distribusi asli.

Selanjutnya, masing-masing generator digunakan untuk memuat data dari direktori hasil pembagian dataset menggunakan fungsi flow_from_directory(). Parameter target_size=(224, 224) menyesuaikan ukuran gambar dengan input model ResNet-50, batch_size=32 menentukan jumlah sampel per *batch*, dan class_mode='binary' digunakan karena kasus ini merupakan klasifikasi dua kelas (*Parasitized* dan *Uninfected).*

Untuk *test generator*, parameter tambahan shuffle=False digunakan agar urutan data uji tetap sama, sehingga memudahkan pencocokan hasil prediksi dengan label asli saat evaluasi.

4.4 Load Model

Tahap ini bertujuan membangun model deteksi malaria menggunakan pendekatan *transfer learning* dengan arsitektur ResNet-50. Pemilihan ResNet-50 didasarkan pada kemampuannya mengekstraksi fitur citra secara efektif melalui *residual connections* yang mampu mengatasi permasalahan *vanishing gradient* pada jaringan saraf yang sangat dalam. Dengan

memanfaatkan bobot awal (*pretrained weights*) dari dataset ImageNet, proses pelatihan dapat dilakukan lebih cepat, meminimalkan kebutuhan data latih dalam jumlah besar, serta meningkatkan akurasi model pada data yang belum pernah dilihat sebelumnya.

Untuk memuat tahapan implementasi *transfer learning* menggunakan arsitektur ResNet-50. Bagian awal memanggil model ResNet-50 dengan bobot awal (*weights*) dari ImageNet, tanpa bagian *fully connected layer* asli (*include_top=False*) karena akan diganti dengan lapisan klasifikasi baru sesuai kebutuhan deteksi malaria. Lapisan awal hingga ke-100 dibekukan (*freeze*) agar parameter yang sudah terlatih pada ImageNet tetap dipertahankan dan berfungsi sebagai *feature extractor,* selain *itu* sebagai perbandingan juga akan dilakukan fine tuning dengan meng unfreeze 20 layer terakhir dari layer yang sebelumnya dibekukan .

Setelah itu, ditambahkan lapisan baru berupa *Global Average Pooling* untuk mereduksi dimensi fitur, diikuti *Batch Normalization* untuk menstabilkan distribusi data pada tiap lapisan. Selanjutnya digunakan *Dropout* sebesar 50% untuk mengurangi risiko *overfitting*. Lapisan *Dense* dengan 256 neuron dan aktivasi *ReLU* digunakan sebagai lapisan tersembunyi, kemudian diakhiri dengan *Dense* satu neuron beraktivasi *sigmoid* untuk klasifikasi biner (*Parasitized* dan *Uninfected*). Model dikompilasi menggunakan optimizer Adam dengan *learning rate* 1e-4, fungsi *loss* binary_crossentropy, dan *metric* akurasi.

4.5 Compile Model

Tahap ini bertujuan mengatasi potensi ketidakseimbangan jumlah data pada masing-masing kelas serta mengoptimalkan proses pelatihan model melalui penerapan *class weight* dan berbagai *callbacks*. *Class weight* membantu model memberi perhatian lebih pada kelas minoritas, sedangkan *callbacks* memantau, menyesuaikan, dan menyimpan hasil pelatihan agar performa optimal dapat dicapai.

langkah pertama adalah menghitung *class weight* secara otomatis menggunakan fungsi compute_class_weight dari scikit-learn. Nilai bobot

ini akan digunakan saat pelatihan untuk menyeimbangkan kontribusi masing-masing kelas terhadap pembaruan bobot model, sehingga mengurangi bias terhadap kelas yang memiliki jumlah sampel lebih banyak.

Selanjutnya, beberapa callback digunakan untuk mengontrol dan memantau proses pelatihan :

1. EarlyStopping

   Menghentikan pelatihan lebih awal jika *val_loss* tidak membaik selama 5 *epoch*, dengan mengembalikan bobot model pada kondisi terbaik.

2. ReduceLROnPlateau

   Mengurangi *learning rate* sebesar faktor 0,2 jika tidak ada peningkatan *val_loss* selama 3 *epoch* berturut-turut.

3. ModelCheckpoint

   Menyimpan model terbaik berdasarkan *val_loss* dengan nama file best_model_80split.h5.

4. CSVLogger

   Menyimpan riwayat pelatihan ke dalam file training_log_80split.csv untuk analisis lebih lanjut.

Dengan kombinasi ini, pelatihan dapat berlangsung lebih adaptif, menghindari *overfitting*, dan menghasilkan model yang lebih baik dan handal.

4.6 Training Model

4.6.1 Partial FineTuned

Partial fine-tuned adalah metode transfer learning di mana hanya sebagian lapisan (*layer*) dari model *pretrained* yang dilatih ulang, sedangkan lapisan lainnya tetap dibekukan dengan bobot awal. Pendekatan ini memungkinkan model tetap efisien karena memanfaatkan fitur umum dari model awal, namun tetap mampu menyesuaikan diri dengan karakteristik dataset malaria. Dengan demikian, partial fine-tuned menjadi solusi tengah antara *feature extraction* dan *full fine-tuning.*

Tahap ini merupakan inti dari proses pengembangan sistem deteksi malaria, yaitu melatih model berbasis ResNet-50 menggunakan data yang

telah dibagi dengan rasio 80% data latih, 10% data validasi, dan 10% data uji. Proses pelatihan ini memanfaatkan *class weight* untuk menangani potensi ketidakseimbangan data dan *callbacks* untuk memastikan pelatihan berjalan efisien, adaptif, serta menghasilkan model terbaik.

Menggunakan fungsi model.fit() digunakan untuk memulai proses pelatihan model, dengan train_generator sebagai sumber data latih yang telah diproses dan diaugmentasi, serta val_generator sebagai data validasi untuk memantau performa model pada data yang belum pernah dilihat sebelumnya. Jumlah *epoch* ditetapkan sebanyak 50, meskipun pelatihan dapat berhenti lebih awal berkat *EarlyStopping* yang ada pada daftar callbacks. Daftar *callbacks* ini juga mencakup *ReduceLROnPlateau* untuk menurunkan *learning rate* secara adaptif, *ModelCheckpoint* untuk menyimpan model terbaik, dan *CSVLogger* untuk merekam riwayat pelatihan. Selain itu, parameter class_weight diberikan agar model memberi perhatian lebih pada kelas dengan jumlah data yang relatif lebih sedikit, sehingga mengurangi bias prediksi. Parameter verbose=1 digunakan agar proses pelatihan menampilkan *progress bar* dan metrik pelatihan di setiap *epoch.*

Dengan pengaturan ini, proses pelatihan model menjadi lebih terkontrol, mengurangi risiko *overfitting*, serta membantu model mencapai konvergensi lebih cepat dan stabil.

Berikut merupakan model summary hasil pelatihan partial finetuned dapat ditunjukkan pada gambar dibawah ini :

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 224, 224, 3)	0	-
conv1_pad (ZeroPadding2D)	(None, 230, 230, 3)	0	input_layer[0][0]
conv1_conv (Conv2D)	(None, 112, 112, 64)	9,472	conv1_pad[0][0]
conv1_bn (BatchNormalizatio…	(None, 112, 112, 64)	256	conv1_conv[0][0]
conv1_relu (Activation)	(None, 112, 112, 64)	0	conv1_bn[0][0]
pool1_pad (ZeroPadding2D)	(None, 114, 114, 64)	0	conv1_relu[0][0]
pool1_pool (MaxPooling2D)	(None, 56, 56, 64)	0	pool1_pad[0][0]
conv2_block1_1_conv (Conv2D)	(None, 56, 56, 64)	4,160	pool1_pool[0][0]
conv2_block1_1_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block1_1_c…
conv2_block1_1_relu (Activation)	(None, 56, 56, 64)	0	conv2_block1_1_b…
conv2_block1_2_conv (Conv2D)	(None, 56, 56, 64)	36,928	conv2_block1_1_r…

....

conv5_block3_2_bn (BatchNormalizatio…	(None, 7, 7, 512)	2,048	conv5_block3_2_c…
conv5_block3_2_relu (Activation)	(None, 7, 7, 512)	0	conv5_block3_2_b…
conv5_block3_3_conv (Conv2D)	(None, 7, 7, 2048)	1,050,624	conv5_block3_2_r…
conv5_block3_3_bn (BatchNormalizatio…	(None, 7, 7, 2048)	8,192	conv5_block3_3_c…
conv5_block3_add (Add)	(None, 7, 7, 2048)	0	conv5_block2_out… conv5_block3_3_b…
conv5_block3_out (Activation)	(None, 7, 7, 2048)	0	conv5_block3_add…
global_average_poo… (GlobalAveragePool…	(None, 2048)	0	conv5_block3_out…
batch_normalization (BatchNormalizatio…	(None, 2048)	8,192	global_average_p…
dropout (Dropout)	(None, 2048)	0	batch_normalizat…
dense (Dense)	(None, 256)	524,544	dropout[0][0]
dropout_1 (Dropout)	(None, 256)	0	dense[0][0]
dense_1 (Dense)	(None, 1)	257	dropout_1[0][0]

Total params: 64,084,357 (244.46 MB)
Trainable params: 19,981,825 (76.22 MB)
Non-trainable params: 4,138,880 (15.79 MB)
Optimizer params: 39,963,652 (152.45 MB)

Gambar 4. 2 Model Summary Partial

Pada gambar 4.2, Model deteksi malaria dibangun menggunakan arsitektur ResNet-50, salah satu bentuk *deep convolutional neural networks* (CNN) yang banyak digunakan pada berbagai tugas klasifikasi citra. ResNet-50 memanfaatkan konsep *residual learning* melalui mekanisme *skip connection*, yang memungkinkan aliran informasi melewati satu atau lebih lapisan tanpa mengalami degradasi. Pendekatan ini membantu mengatasi masalah *vanishing gradient* yang sering muncul pada jaringan yang sangat dalam, sehingga memperbaiki kinerja pelatihan dan akurasi model.

Dalam implementasinya, ResNet-50 digunakan sebagai *base model* yang telah melalui proses *pretraining* pada dataset ImageNet. Sebanyak 100 lapisan awal dibekukan (*frozen*) untuk mempertahankan bobot hasil *pretraining*, sementara bagian akhir dimodifikasi agar sesuai dengan tugas klasifikasi biner (*Parasitized* dan *Uninfected*). Penyesuaian dilakukan dengan menambahkan lapisan GlobalAveragePooling2D untuk mereduksi dimensi fitur, BatchNormalization untuk menstabilkan distribusi aktivasi, Dropout untuk mengurangi *overfitting*, dan Dense layer dengan aktivasi sigmoid sebagai keluaran akhir.

Model menerima masukan berupa citra RGB berukuran 224×224×3 dan menghasilkan keluaran berupa probabilitas kelas tunggal (0 atau 1). Berdasarkan hasil model.summary(), model ini memiliki total 64.084.357 parameter (244,46 MB), dengan 19.981.825 parameter dapat dilatih (*trainable*, 76,22 MB) dan 4.138.880 parameter dibekukan (*non-trainable*, 15,79 MB). Selain itu, terdapat 39.963.652 parameter yang digunakan oleh optimizer (152,45 MB) selama proses pelatihan. Jumlah parameter yang besar ini memberikan kapasitas yang cukup untuk mempelajari perbedaan morfologi sel darah yang terinfeksi malaria dan yang sehat.

4.6.2 FineTuned Unfreeze 20 Layer

Tahap ini merupakan proses *fine-tuning* model, yaitu pelatihan lanjutan setelah pelatihan awal (*initial training*) pada ResNet-50. Berbeda dengan tahap sebelumnya, pada *fine-tuning* sebagian lapisan *feature extractor* yang sebelumnya dibekukan kini dibuka (*unfreeze*) sehingga

bobotnya dapat diperbarui. Pendekatan ini bertujuan agar model tidak hanya mengandalkan fitur umum hasil pra-pelatihan dari ImageNet, tetapi juga mampu mempelajari pola visual yang lebih spesifik terhadap citra malaria.

Proses *fine-tuning* dilakukan dengan memanggil kembali fungsi model.fit() menggunakan data latih (train_generator) dan data validasi (val_generator). Model dilatih selama 20 *epoch* menggunakan *callback* yang telah disesuaikan (callbacks_ft) untuk mengontrol jalannya pelatihan, seperti *early stopping*, penyesuaian *learning rate*, dan penyimpanan model terbaik. Parameter class_weight tetap digunakan untuk mengatasi ketidakseimbangan kelas. Dengan membuka sebagian lapisan awal ResNet-50, model dapat memperbarui representasi fitur secara lebih mendalam, sehingga meningkatkan kemampuannya dalam mengenali perbedaan halus antara sel terinfeksi dan tidak terinfeksi.

Hasil dari tahap *fine-tuning* ini diharapkan memberikan peningkatan akurasi dan generalisasi model, karena bobot lapisan awal kini telah menyesuaikan secara lebih optimal terhadap karakteristik citra malaria yang digunakan.

Berikut merupakan Model Summary dari pelatihan model unfreeze 20 layer dapat dilihat pada gambar dibawah ini:

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 224, 224, 3)	0	-
conv1_pad (ZeroPadding2D)	(None, 230, 230, 3)	0	input_layer[0][0]
conv1_conv (Conv2D)	(None, 112, 112, 64)	9,472	conv1_pad[0][0]
conv1_bn (BatchNormalizatio…	(None, 112, 112, 64)	256	conv1_conv[0][0]
conv1_relu (Activation)	(None, 112, 112, 64)	0	conv1_bn[0][0]
pool1_pad (ZeroPadding2D)	(None, 114, 114, 64)	0	conv1_relu[0][0]
pool1_pool (MaxPooling2D)	(None, 56, 56, 64)	0	pool1_pad[0][0]
conv2_block1_1_conv (Conv2D)	(None, 56, 56, 64)	4,160	pool1_pool[0][0]
conv2_block1_1_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block1_1_c…
conv2_block1_1_relu (Activation)	(None, 56, 56, 64)	0	conv2_block1_1_b…
conv2_block1_2_conv (Conv2D)	(None, 56, 56, 64)	36,928	conv2_block1_1_r…
conv2_block1_2_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block1_2_c…
conv2_block1_2_relu (Activation)	(None, 56, 56, 64)	0	conv2_block1_2_b…

...

conv5_block3_2_bn (BatchNormalizatio…	(None, 7, 7, 512)	2,048	conv5_block3_2_c…
conv5_block3_2_relu (Activation)	(None, 7, 7, 512)	0	conv5_block3_2_b…
conv5_block3_3_conv (Conv2D)	(None, 7, 7, 2048)	1,050,624	conv5_block3_2_r…
conv5_block3_3_bn (BatchNormalizatio…	(None, 7, 7, 2048)	8,192	conv5_block3_3_c…
conv5_block3_add (Add)	(None, 7, 7, 2048)	0	conv5_block2_out… conv5_block3_3_b…
conv5_block3_out (Activation)	(None, 7, 7, 2048)	0	conv5_block3_add…
global_average_poo… (GlobalAveragePool…	(None, 2048)	0	conv5_block3_out…
batch_normalization (BatchNormalizatio…	(None, 2048)	8,192	global_average_p…
dropout (Dropout)	(None, 2048)	0	batch_normalizat…
dense (Dense)	(None, 256)	524,544	dropout[0][0]
dropout_1 (Dropout)	(None, 256)	0	dense[0][0]
dense_1 (Dense)	(None, 1)	257	dropout_1[0][0]

Total params: 64,084,357 (244.46 MB)
Trainable params: 19,981,825 (76.22 MB)
Non-trainable params: 4,138,880 (15.79 MB)
Optimizer params: 39,963,652 (152.45 MB)

Gambar 4. 3 Model Summary FineTuned Unfreeze 20 layer

Hasil ringkasan arsitektur menunjukkan bahwa model berbasis ResNet-50 dengan modifikasi *custom classifier* telah berhasil dibangun. Total parameter yang dimiliki model adalah 64.084.357 parameter (±244,46 MB). Dari jumlah tersebut, sebanyak 39.963.652 parameter (±152,45 MB) termasuk dalam kategori optimizer params, sedangkan 4.138.880 parameter (±15,79 MB) merupakan non-trainable params yang tetap dibekukan agar pengetahuan dasar dari *pretrained weights ImageNet* tetap terjaga. Adapun parameter yang benar-benar dilatih (*trainable params*) berjumlah 19.981.825 parameter (±76,22 MB).

Jumlah parameter yang cukup besar ini mencerminkan kapasitas model yang tinggi dalam mengekstraksi dan mempelajari representasi fitur citra malaria. Penggunaan strategi fine-tuning memungkinkan sebagian besar lapisan ResNet-50 tetap berperan sebagai *feature extractor*, sementara lapisan atas (dense dan dropout) dioptimalkan kembali agar lebih adaptif terhadap dataset malaria. Dengan demikian, model dapat menyeimbangkan antara pemanfaatan *knowledge transfer* dari ImageNet dan kemampuan adaptasi terhadap domain medis yang lebih spesifik.

Meskipun jumlah parameter tidak mengalami perubahan, perbedaan utama dari penerapan *fine-tuning* terletak pada proses pelatihan bobot internal dari beberapa layer awal ResNet-50 yang sebelumnya dibekukan. Dengan membuka sebagian layer tersebut, diharapkan model dapat menyesuaikan representasi fitur yang lebih relevan terhadap karakteristik citra malaria. Oleh karena itu, evaluasi kinerja model menjadi tahap penting untuk mengetahui sejauh mana strategi *fine-tuning* ini dapat memberikan peningkatan performa dibandingkan pelatihan awal (*feature extraction* saja).

4.7 Evaluasi Model

Tahap evaluasi model bertujuan untuk menilai kinerja sistem deteksi malaria yang telah dibangun setelah melalui proses pelatihan. Evaluasi dilakukan dengan menerapkan berbagai metode pengukuran performa untuk

memperoleh gambaran menyeluruh mengenai kemampuan model dalam mengenali dan membedakan citra yang terinfeksi malaria dan yang sehat.

Pendekatan ini memberikan pemahaman yang lebih komprehensif, tidak hanya terkait ketepatan prediksi secara keseluruhan, tetapi juga kualitas model dalam menangani ketidakseimbangan data dan tingkat kesalahan pada tiap kelas.

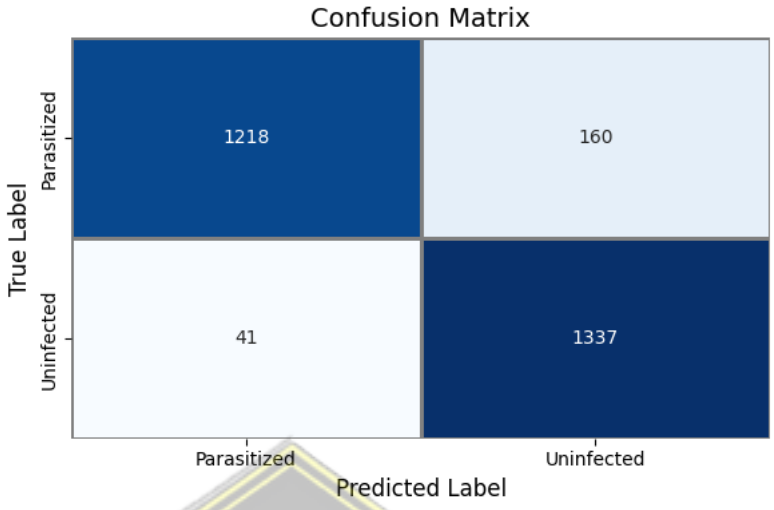
4.7.1 Partial FineTuned

Partial fine-tuned adalah metode transfer learning di mana hanya sebagian lapisan (*layer*) dari model *pretrained* yang dilatih ulang, sedangkan lapisan lainnya tetap dibekukan dengan bobot awal. Pendekatan ini memungkinkan model tetap efisien karena memanfaatkan fitur umum dari model awal, namun tetap mampu menyesuaikan diri dengan karakteristik dataset malaria. Dengan demikian, partial fine-tuned menjadi solusi tengah antara *feature extraction* dan *full fine-tuning.*

1. Confusion Matrix

   Tahap ini bertujuan untuk mengevaluasi kinerja model deteksi malaria dengan menggunakan *confusion matrix* dan *classification report*. Evaluasi dilakukan untuk melihat sejauh mana model mampu membedakan dua kelas, yaitu Parasitized dan Uninfected, setelah proses pelatihan selesai. *Confusion matrix* memberikan gambaran jumlah prediksi benar dan salah pada masing-masing kelas, sedangkan *classification report* memberikan metrik evaluasi yang lebih detail seperti *precision*, *recall*, dan *f1-score.*

   Diawali dengan melakukan *reset* pada test_generator untuk menghindari masalah urutan batch. Selanjutnya, model digunakan untuk memprediksi probabilitas kelas pada data uji, lalu hasilnya diubah menjadi label biner berdasarkan *threshold* 0,5. Nilai prediksi (y_pred) kemudian dibandingkan dengan label sebenarnya (y_true) untuk menghasilkan *confusion matrix* dan *classification report*. *Confusion matrix* menampilkan jumlah prediksi benar dan salah pada tiap kelas, sementara *classification report* memberikan metrik evaluasi secara rinci.



Classification Report:

	precision	recall	f1-score	support
Parasitized	0.97	0.88	0.92	1378
Uninfected	0.89	0.97	0.93	1378
accuracy			0.93	2756
macro avg	0.93	0.93	0.93	2756
weighted avg	0.93	0.93	0.93	2756

Gambar 4. 4 Confusion Matrix Partial

Berdasarkan gambar 4.4, *confusion matrix* yang dihasilkan, model berhasil mengklasifikasikan 1.218 citra *Parasitized* dengan benar, namun masih terdapat 160 citra *Parasitized* yang salah diklasifikasikan sebagai *Uninfected*. Untuk kelas *Uninfected*, model berhasil mengklasifikasikan 1.337 citra dengan benar, sedangkan 41 citra salah diklasifikasikan sebagai *Parasitized.*

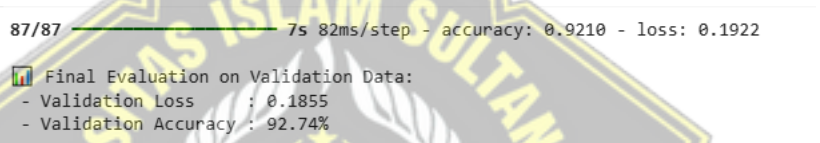
Dari *classification report*, diperoleh bahwa :

- Kelas Parasitized memiliki *precision* sebesar 0,97, yang menunjukkan bahwa sebagian besar prediksi positif untuk kelas ini benar. *Recall*-nya sebesar 0,88 menandakan bahwa masih ada sekitar 12% citra *Parasitized* yang tidak terdeteksi.
- Kelas Uninfected memiliki *precision* sebesar 0,89 dan *recall* sebesar 0,97, yang menunjukkan model sangat jarang melewatkan citra *Uninfected*, walaupun ada sedikit kesalahan prediksi.

Secara keseluruhan, akurasi model mencapai 93%, dengan *macro average* dan *weighted average* yang sama-sama berada pada angka 0,93. Hasil ini membuktikan bahwa model memiliki performa yang baik dan relatif seimbang pada kedua kelas, meskipun masih dapat ditingkatkan terutama pada deteksi kelas *Parasitized.*

2. Evaluasi Data Validasi

Selain evaluasi pada data uji, pengujian model juga dilakukan pada data validasi untuk memastikan bahwa model memiliki kemampuan generalisasi yang baik terhadap data yang tidak digunakan selama pelatihan. Langkah ini membantu mendeteksi adanya potensi *overfitting* atau *underfitting* pada model.



Gambar 4. 5 Hasil Evaluasi Validasi

Pada gambar 4.5, dengan menggunakan fungsi model.evaluate() untuk menghitung nilai *loss* dan *accuracy* pada *validation set*. Parameter verbose=1 memastikan proses evaluasi menampilkan progres per batch. Nilai *loss* memberikan indikasi seberapa besar kesalahan prediksi model terhadap data validasi, sedangkan nilai *accuracy* menunjukkan persentase prediksi yang benar. Hasil evaluasi ini menjadi acuan untuk melihat apakah model tetap konsisten dalam performa prediksi ketika diuji pada data yang belum pernah dilihat selama pelatihan.

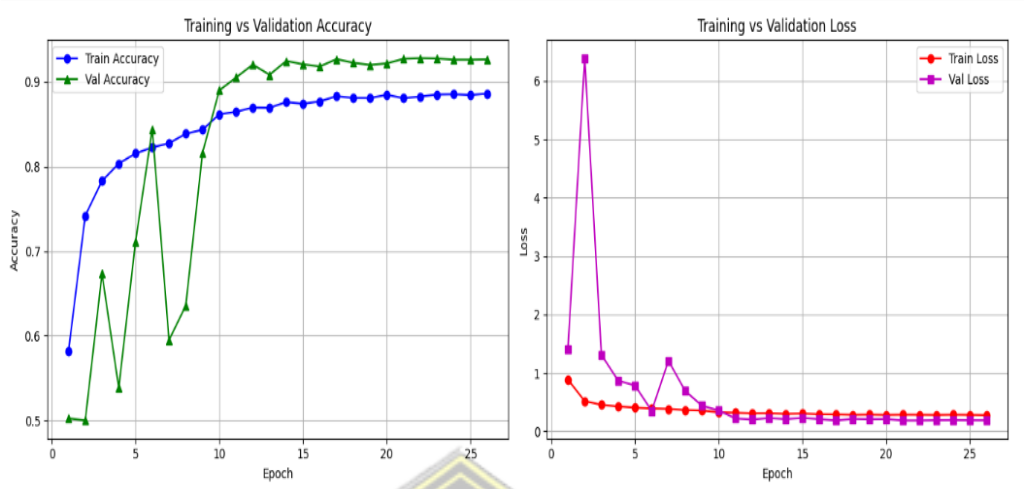
Model yang telah dilatih kemudian dievaluasi menggunakan data validasi untuk mengukur performanya dalam mengenali pola yang belum pernah dilihat selama proses pelatihan. Dari hasil evaluasi diperoleh nilai *validation loss* sebesar 0.1855, yang menunjukkan bahwa rata-rata kesalahan prediksi model pada data validasi cukup rendah. Nilai loss yang kecil ini menandakan model mampu menyesuaikan diri dengan pola data secara efektif tanpa mengalami overfitting yang berlebihan.

Selanjutnya, akurasi validasi mencapai 92.74%, yang berarti sekitar 92 dari 100 sampel data validasi berhasil diklasifikasikan dengan benar oleh model. Angka ini menggambarkan kemampuan model dalam mengenali karakteristik fitur gambar malaria dengan tingkat ketelitian yang tinggi.

3. Perbandingan accuracy dan loss training & validation

Pada tahap ini, performa model selama proses pelatihan dan validasi dianalisis dengan menggunakan metrik akurasi dan loss. Visualisasi grafik akurasi dan loss pada data training dan validasi ditampilkan untuk memberikan gambaran mengenai kemampuan model dalam mempelajari pola data serta mengidentifikasi potensi overfitting atau underfitting.

Dengan menggunakan kode yang berfungsi untuk memvisualisasikan perkembangan nilai akurasi dan loss selama proses pelatihan model. Pertama, nilai akurasi dan loss pada data latih (accuracy, loss) serta data validasi (val_accuracy, val_loss) diambil dari objek history yang dihasilkan oleh fungsi model.fit(). Selanjutnya, dibuat rentang *epoch* untuk sumbu-x sesuai jumlah iterasi pelatihan. Grafik dibagi menjadi dua bagian, yaitu grafik akurasi pada sisi kiri dan grafik loss pada sisi kanan. Pada grafik akurasi, ditampilkan perbandingan antara akurasi data latih (garis biru) dan akurasi data validasi (garis hijau), sedangkan pada grafik loss ditampilkan perbandingan antara loss data latih (garis merah) dan loss data validasi (garis magenta). Setiap grafik dilengkapi dengan judul, label sumbu, legenda, dan grid untuk memudahkan pembacaan. Fungsi ini dipanggil setelah pelatihan selesai untuk membantu mengevaluasi kinerja model serta mendeteksi kemungkinan terjadinya *overfitting* atau *underfitting* melalui pola perbedaan nilai antara data latih dan validasi.



Gambar 4. 6 Training Validasi accuracy & loss partial

Berdasarkan Gambar 4.6, terlihat bahwa nilai akurasi pada data latih (*Train Accuracy*) meningkat secara bertahap dari epoch pertama hingga sekitar epoch ke-10, kemudian cenderung stabil di kisaran ±88%. Sementara itu, akurasi pada data validasi (*Validation Accuracy*) mengalami fluktuasi cukup signifikan pada beberapa epoch awal, namun setelah epoch ke-10 mulai stabil di kisaran ±91–92%, sedikit lebih tinggi dibandingkan akurasi data latih.

Pada grafik loss, nilai *Train Loss* menunjukkan tren penurunan yang konsisten sejak awal pelatihan, menandakan bahwa model semakin mampu meminimalkan kesalahan pada data latih. Nilai *Validation Loss* awalnya cukup tinggi dan sempat mengalami lonjakan pada epoch ke-2, namun kemudian menurun drastis dan stabil mendekati nilai *Train Loss* setelah epoch ke-10. Pola ini mengindikasikan bahwa model tidak mengalami *overfitting* yang signifikan, karena perbedaan antara *Train Loss* dan *Validation Loss* relatif kecil pada tahap akhir pelatihan.

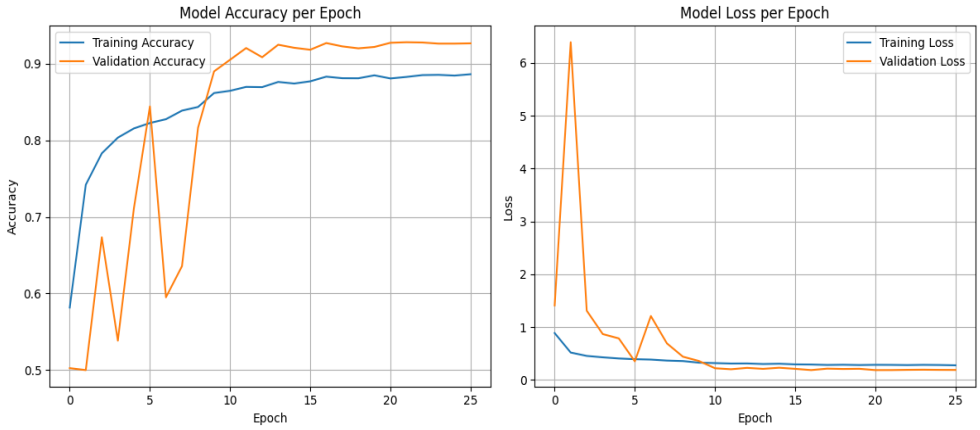
Secara keseluruhan, kurva akurasi dan loss ini menunjukkan bahwa proses pelatihan ResNet-50 dengan metode transfer learning berhasil mencapai performa yang baik, di mana model mampu mempertahankan akurasi tinggi pada data validasi dengan kesalahan yang rendah.

4. Metrik Kinerja Model per Epoch

Pada tahap ini dilakukan proses visualisasi metrik kinerja model selama proses pelatihan dengan memanfaatkan data log yang disimpan dalam format CSV. Visualisasi ini bertujuan untuk memantau perkembangan akurasi dan loss pada data latih (training) serta data validasi (validation) di setiap epoch. Dengan adanya grafik ini, peneliti dapat mengevaluasi apakah model mengalami peningkatan kinerja secara konsisten, mendeteksi adanya gejala overfitting atau underfitting, serta mengidentifikasi titik optimal pelatihan.

Kode program dimulai dengan membaca file CSV berisi log pelatihan menggunakan library Pandas. CSV ini memuat informasi seperti nomor epoch, nilai akurasi, nilai akurasi validasi, nilai loss, dan nilai loss validasi. Setelah data dimuat ke dalam DataFrame log_df, dilakukan pembuatan dua grafik menggunakan Matplotlib. Grafik pertama menampilkan hubungan antara epoch dengan akurasi pada data latih (accuracy) dan validasi (val_accuracy), sementara grafik kedua menunjukkan hubungan antara epoch dengan loss pada data latih (loss) dan validasi (val_loss).

Proses plotting menggunakan plt.subplot() untuk menempatkan dua grafik dalam satu kanvas secara berdampingan, sehingga memudahkan perbandingan langsung antara tren akurasi dan tren loss. Setiap grafik dilengkapi judul, label sumbu, legenda, dan grid agar lebih informatif dan mudah dibaca. Penggunaan plt.tight_layout() memastikan tata letak grafik rapi tanpa saling tumpang tindih.



Gambar 4. 7 Visualisasi model per epoch partial

Berdasarkan grafik Model Accuracy per Epoch, akurasi pelatihan (Training Accuracy) mengalami peningkatan yang konsisten dari sekitar 58% pada epoch pertama hingga mencapai ±88% pada epoch ke-25. Sementara itu, akurasi validasi (Validation Accuracy) menunjukkan fluktuasi cukup besar pada 7 epoch pertama, yang kemungkinan disebabkan oleh proses adaptasi awal model terhadap data yang belum pernah dilihat. Setelah epoch ke-8, akurasi validasi stabil di kisaran 91–92%, menunjukkan bahwa model mampu melakukan generalisasi dengan baik pada data uji validasi.

Pada grafik Model Loss per Epoch, nilai loss pelatihan turun drastis dari sekitar 0,8 di awal menjadi ±0,2 pada epoch ke-25. Loss validasi juga menunjukkan tren penurunan yang signifikan, meskipun di awal sempat melonjak pada epoch ke-2 dan ke-6. Setelah memasuki epoch ke-8, nilai loss validasi relatif stabil di bawah 0,2, sejalan dengan kestabilan akurasi validasi.

Fenomena ini mengindikasikan bahwa model ResNet-50 hasil transfer learning berhasil beradaptasi dengan baik terhadap dataset malaria. Tidak terlihat tanda overfitting yang mencolok, karena akurasi validasi justru sedikit lebih tinggi dibanding akurasi pelatihan pada beberapa epoch terakhir. Hal ini bisa disebabkan oleh penggunaan teknik augmentasi data dan regularisasi yang membantu mencegah model menghafal data latih.

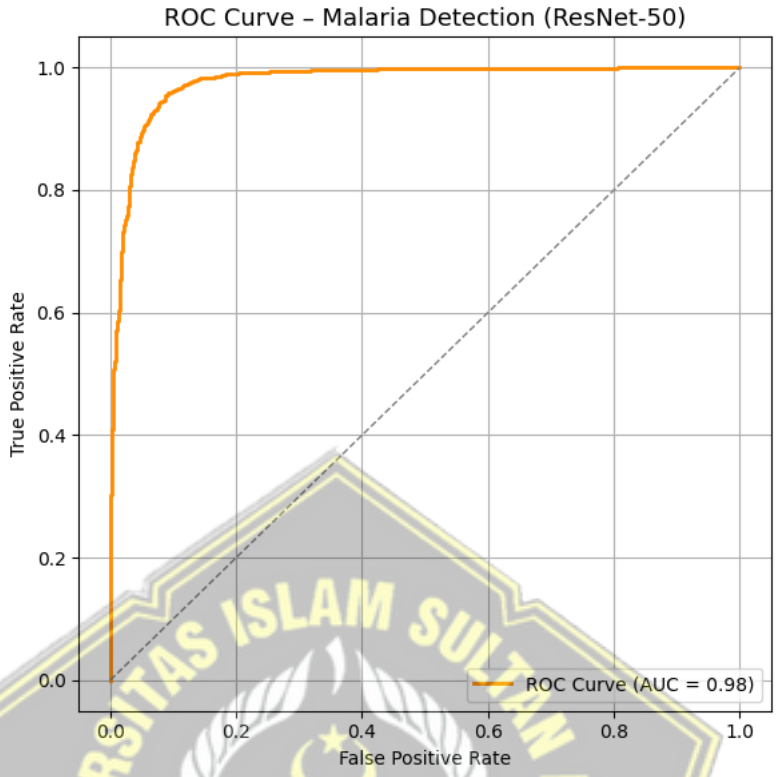
Visualisasi akurasi dan loss ini menjadi salah satu alat penting untuk menganalisis kinerja model ResNet-50 hasil transfer learning. Dari grafik akurasi, peneliti dapat melihat sejauh mana model mampu meningkatkan kemampuan klasifikasinya seiring bertambahnya epoch. Sementara itu, grafik loss membantu memahami bagaimana error pada model berkurang selama proses pelatihan. Apabila terdapat selisih besar antara nilai validasi dan nilai pelatihan pada kedua grafik, hal ini dapat menjadi indikasi overfitting yang memerlukan penyesuaian strategi, seperti penambahan regularisasi atau penerapan early stopping. Dengan demikian, visualisasi ini

tidak hanya menjadi dokumentasi kinerja, tetapi juga berperan sebagai panduan pengambilan keputusan dalam optimasi model.

5. ROC Curve

Selain menggunakan metrik akurasi, performa model juga dievaluasi dengan *Receiver Operating Characteristic* (ROC) Curve dan *Area Under Curve* (AUC). ROC Curve menggambarkan hubungan antara True Positive Rate (TPR) dan False Positive Rate (FPR) pada berbagai nilai ambang (*threshold*), sedangkan AUC mengukur luas area di bawah kurva ROC. Semakin mendekati nilai 1, semakin baik kemampuan model dalam membedakan antara kelas positif dan negative.

Dengan kode yang diawali dengan melakukan *reset* pada test_generator untuk memastikan prediksi dimulai dari awal. Model kemudian menghasilkan probabilitas prediksi untuk setiap sampel pada data uji (y_probs), sedangkan label aslinya (y_true) diambil dari atribut classes. Fungsi roc_curve() digunakan untuk menghitung nilai FPR dan TPR pada berbagai *threshold*, kemudian fungsi auc() menghitung luas area di bawah kurva tersebut. Hasilnya divisualisasikan menggunakan *matplotlib*, dengan garis ROC berwarna oranye dan garis diagonal abu-abu sebagai pembanding model acak. Nilai AUC ditampilkan pada legenda grafik untuk memudahkan interpretasi.



Gambar 4. 8 ROC Curve Partial

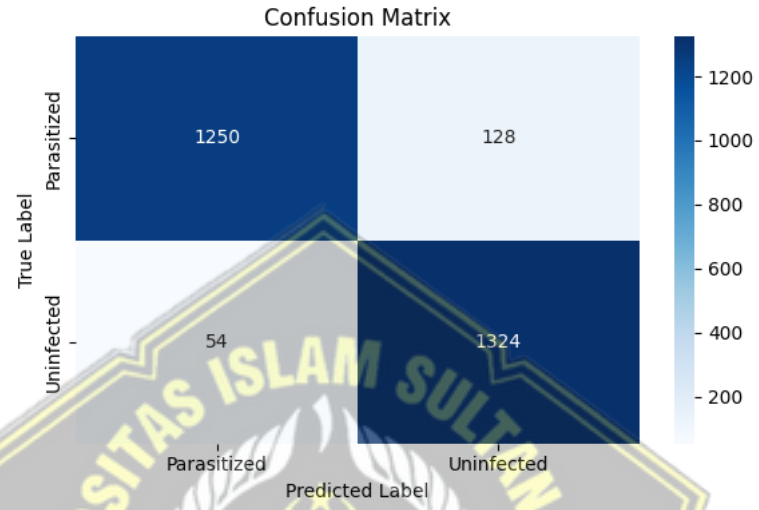
Berdasarkan Gambar 4.8, kurva ROC yang dihasilkan berada jauh di atas garis diagonal acak (warna abu-abu), yang menandakan performa klasifikasi yang sangat baik. Nilai AUC yang diperoleh adalah 0.98, mendekati nilai maksimum 1.0. Hal ini menunjukkan bahwa model ResNet-50 dengan metode transfer learning mampu membedakan citra sel darah yang terinfeksi malaria (*Parasitized*) dan tidak terinfeksi (*Uninfected*) dengan tingkat akurasi yang sangat tinggi pada berbagai nilai ambang keputusan (*threshold*). Bentuk kurva yang mendekati sudut kiri atas grafik juga mengindikasikan bahwa model memiliki True Positive Rate yang tinggi sekaligus False Positive Rate yang rendah, sehingga potensi kesalahan klasifikasi relatif kecil. Secara keseluruhan, hasil ini memperkuat temuan dari evaluasi akurasi dan loss, bahwa model yang dibangun memiliki performa prediksi yang optimal dan andal.

4.7.2 FineTuned Unfreeze 20 Layer

1. Confusion Matrix

Classification Report:

	precision	recall	f1-score	support
Parasitized	0.96	0.91	0.93	1378
Uninfected	0.91	0.96	0.94	1378
accuracy			0.93	2756
macro avg	0.94	0.93	0.93	2756
weighted avg	0.94	0.93	0.93	2756



Gambar 4. 9 Confusion Matrix FineTuned Unfreeze 20 layer

Evaluasi kinerja model dilakukan baik sebelum maupun sesudah proses *fine-tuning* untuk melihat sejauh mana pembukaan sebagian layer awal ResNet-50 berpengaruh terhadap performa klasifikasi. Pada tahap awal, model dijalankan dengan kondisi *frozen layers* (100 layer pertama dibekukan) dan menghasilkan akurasi sebesar 93%. Setelah dilakukan *fine-tuning* dengan membuka 20 layer awal agar dapat ikut belajar, model juga menghasilkan akurasi keseluruhan sebesar 93%. Hal ini menunjukkan bahwa proses *fine-tuning* tidak memberikan peningkatan yang signifikan pada akurasi total.

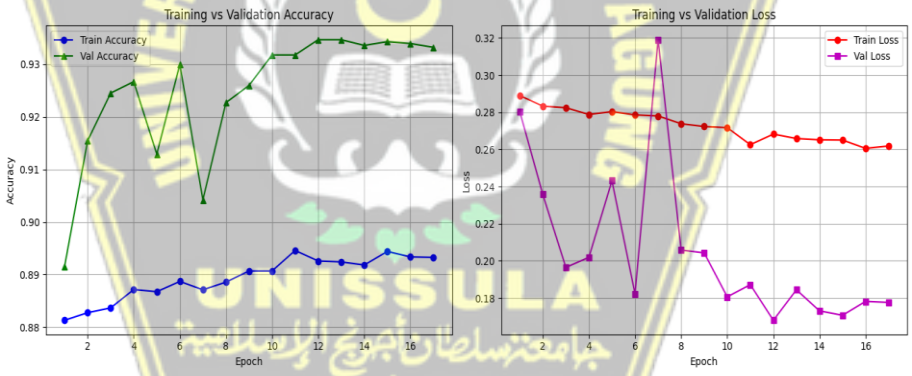
Namun, apabila ditinjau lebih lanjut pada metrik evaluasi per kelas, terlihat adanya perbaikan pada kualitas prediksi. Untuk kelas Parasitized, nilai *recall* meningkat dari 0.88 menjadi 0.91, yang berarti model lebih sensitif dalam mendeteksi sel yang benar-benar terinfeksi malaria. Meskipun demikian, terjadi sedikit penurunan pada *precision* (dari 0.97 menjadi 0.96). Sebaliknya, pada kelas Uninfected, nilai *precision* meningkat dari 0.89 menjadi 0.91, sehingga model lebih tepat dalam mengenali sel

sehat tanpa banyak salah klasifikasi, meskipun nilai *recall* sedikit menurun (dari 0.97 menjadi 0.96).

Perubahan ini juga tercermin pada hasil *confusion matrix*. Sebelum *fine-tuning*, model salah mengklasifikasikan 160 citra Parasitized dan 41 citra Uninfected. Setelah *fine-tuning*, kesalahan pada kelas Parasitized berkurang menjadi 128 citra, sedangkan kesalahan pada kelas Uninfected sedikit meningkat menjadi 54 citra.

Secara keseluruhan, dapat disimpulkan bahwa *fine-tuning* pada model ResNet-50 tidak meningkatkan akurasi secara keseluruhan, tetapi mampu menghasilkan keseimbangan yang lebih baik antar kelas. Model menjadi lebih sensitif dalam mendeteksi sel terinfeksi malaria dan lebih tepat dalam mengenali sel sehat, sehingga kinerjanya lebih seimbang meskipun akurasi total tetap sama.

2. Accuracy & Loss Training



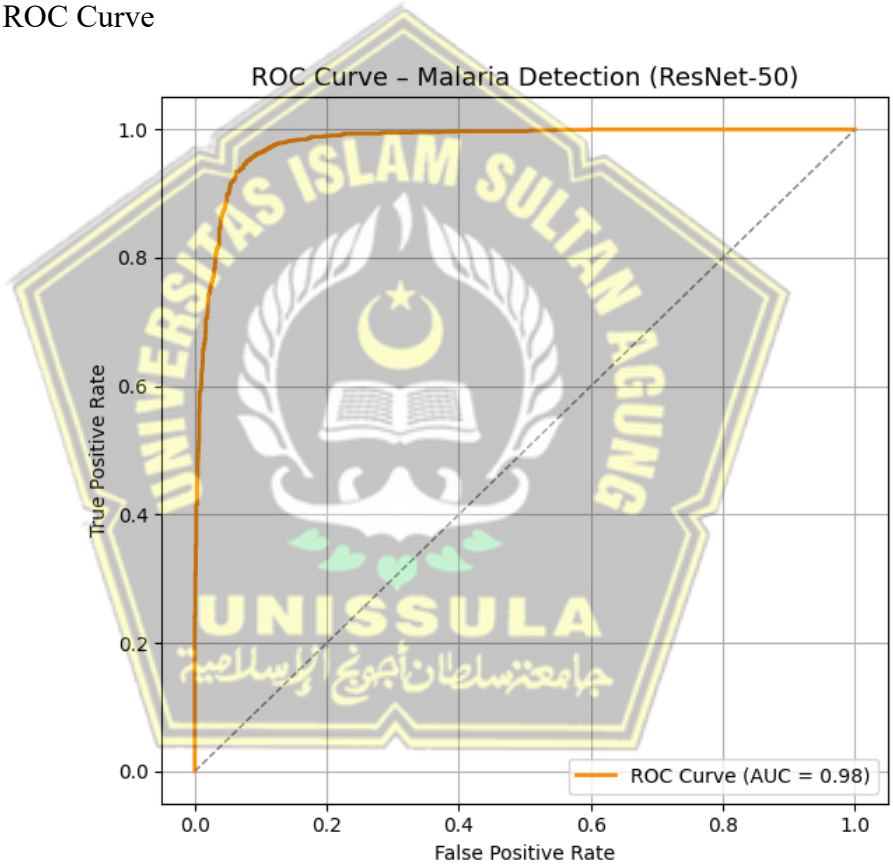
Gambar 4. 10 Visualisasi training accuracy &loss unfreeze 20 layer

Pada gambar 4.10, Setelah fine-tuning, akurasi data latih stabil di kisaran ±89%, sedangkan akurasi validasi lebih tinggi di kisaran ±91–93% dengan fluktuasi kecil. Train Loss relatif stabil, sementara Validation Loss menurun dan bertahan lebih rendah dibandingkan Train Loss. Hal ini menunjukkan bahwa fine-tuning membuat model lebih stabil, meningkatkan generalisasi pada data validasi, serta mengurangi risiko overfitting.

Berdasarkan hasil perbandingan sebelum dan sesudah dilakukan fine-tuning, dapat disimpulkan bahwa proses fine-tuning mampu memberikan kontribusi positif terhadap performa model. Meskipun peningkatan akurasi

secara numerik relatif kecil, stabilitas pelatihan dan konsistensi akurasi pada data validasi menunjukkan perbaikan yang signifikan. Hal ini mengindikasikan bahwa model dengan lapisan tambahan yang dibuka melalui fine-tuning dapat lebih optimal dalam menyesuaikan representasi fitur terhadap karakteristik citra malaria, tanpa menimbulkan overfitting yang berarti. Dengan demikian, fine-tuning terbukti sebagai strategi yang efektif dalam meningkatkan kemampuan generalisasi model ResNet-50 pada tugas klasifikasi citra malaria.

3. ROC Curve

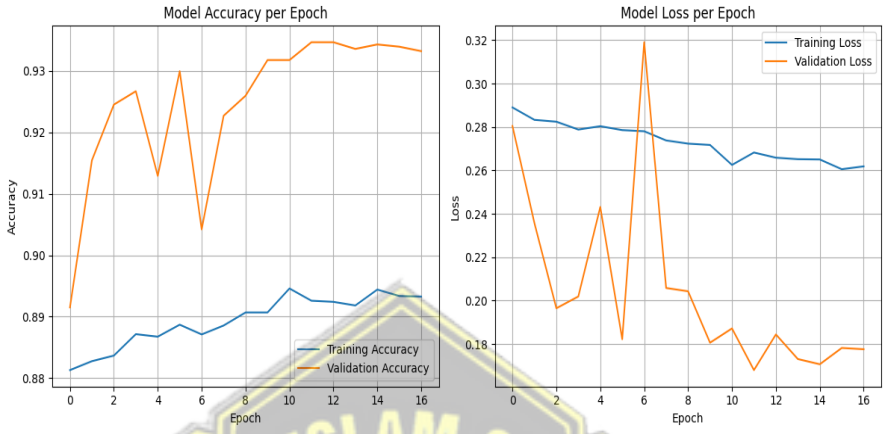


Gambar 4. 11 ROC Curve unfreeze 20 layer

Pada Gambar 4.11, untuk hasilnya tetap sama di 0.98 Berdasarkan hasil ROC Curve, baik sebelum maupun sesudah fine-tuning model ResNet-50 sama-sama memperoleh nilai AUC sebesar 0.98, yang menandakan performa klasifikasi sudah sangat baik. Namun, setelah dilakukan fine-tuning, bentuk kurva ROC terlihat lebih stabil dan mulus dibandingkan sebelum fine-tuning, sehingga meskipun tidak ada peningkatan signifikan

pada nilai AUC, fine-tuning tetap berkontribusi dalam memperkuat konsistensi dan keandalan model dalam mendeteksi malaria.

4. Loss Accuracy Epoch



Gambar 4. 12 Accuracy Loss unfreeze 20 layer

Pada gambar 4.12 grafik akurasi, Train Accuracy terlihat relatif stabil sejak awal pelatihan, berada di kisaran 88–89% dan hanya mengalami peningkatan kecil. Sementara itu, Validation Accuracy mengalami fluktuasi pada epoch awal, namun setelah itu meningkat signifikan hingga mencapai kisaran 93%, lebih tinggi dibandingkan akurasi data latih. Kondisi ini menunjukkan bahwa model dapat melakukan generalisasi dengan baik pada data validasi setelah fine-tuning.

Pada grafik loss, Train Loss mulai dari sekitar 0.29 dan menurun perlahan hingga mencapai 0.26 pada epoch terakhir. Sementara Validation Loss berfluktuasi cukup tajam pada beberapa epoch awal, namun kemudian menurun dan stabil pada kisaran 0.17–0.18, yang justru lebih rendah dibandingkan nilai Train Loss. Pola ini mengindikasikan bahwa model tidak hanya menghindari overfitting, tetapi juga mampu menyesuaikan representasi fitur dengan lebih baik setelah dilakukan fine-tuning.

Secara keseluruhan, hasil fine-tuning menunjukkan bahwa membuka **20** layer awal dari ResNet-50 memberikan dampak positif terhadap performa model, ditandai dengan peningkatan akurasi validasi serta penurunan validation loss. Hal ini membuktikan bahwa penyesuaian bobot

pretrained pada layer awal mampu meningkatkan kemampuan model dalam mendeteksi malaria pada citra mikroskopis.

Perbandingan hasil sebelum dan sesudah fine-tuning menunjukkan bahwa meskipun akurasi keseluruhan tidak berubah drastis, fine-tuning memberikan perbaikan dalam stabilitas dan generalisasi model. Sebelum fine-tuning, akurasi validasi memang sudah tinggi namun masih fluktuatif dengan nilai loss yang cukup besar pada beberapa epoch awal. Setelah fine-tuning, akurasi validasi menjadi lebih konsisten di kisaran ±93% dengan nilai loss yang lebih rendah dan stabil, bahkan cenderung lebih baik daripada train loss. Hal ini mengindikasikan bahwa fine-tuning membantu model ResNet-50 menyesuaikan representasi fitur dengan lebih baik terhadap dataset malaria, sehingga meningkatkan kemampuan generalisasi dan performa deteksi secara keseluruhan.

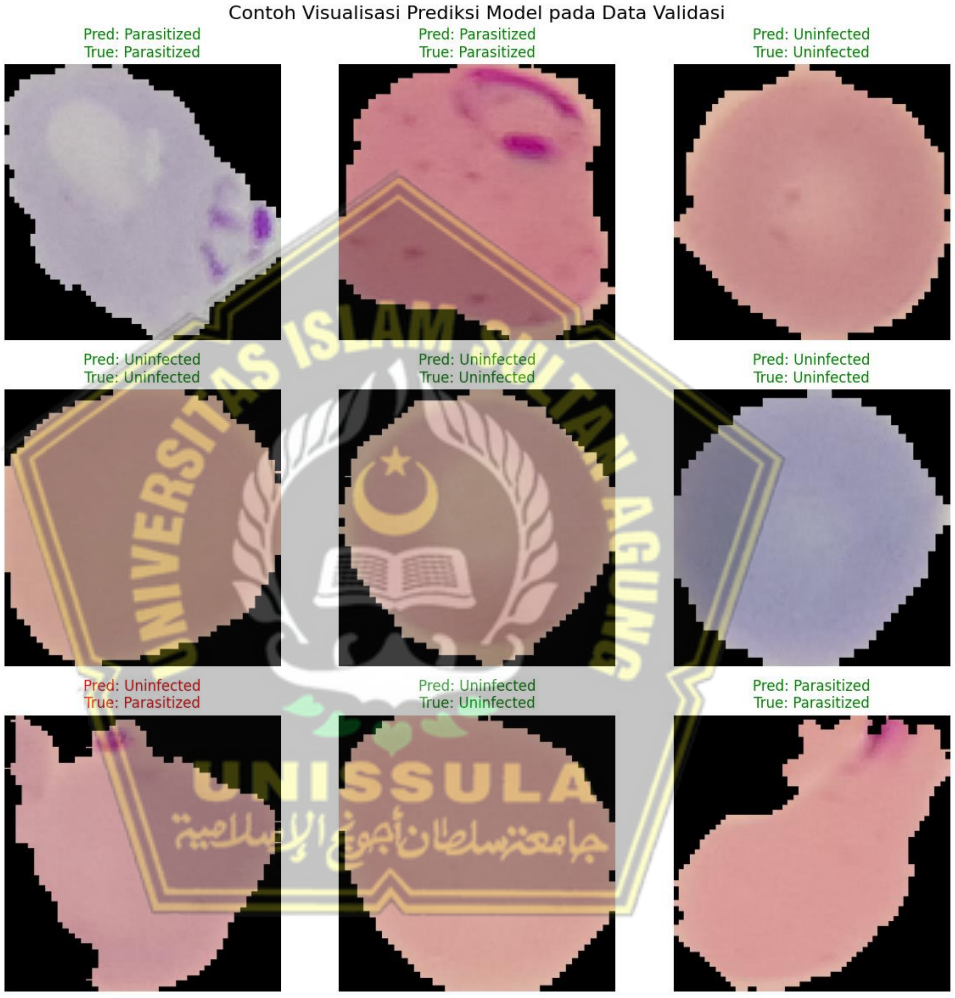
4.8 Visualisasi Hasil Prediksi Gambar

Visualisasi dan prediksi gambar dilakukan untuk melihat secara langsung kemampuan model dalam mengklasifikasikan citra uji, sekaligus memvalidasi hasil evaluasi kuantitatif yang telah diperoleh pada tahap sebelumnya. Melalui proses ini, beberapa citra dari dataset uji ditampilkan beserta label sebenarnya (*ground truth*) dan label prediksi model. Dengan demikian, dapat diamati bagaimana model mengidentifikasi kelas citra, baik pada kasus yang benar terklasifikasi maupun pada kasus yang keliru. Pendekatan ini memberikan gambaran yang lebih intuitif mengenai kinerja model, sehingga tidak hanya bergantung pada metrik numerik seperti akurasi atau AUC, tetapi juga pada bukti visual dari prediksi yang dihasilkan.

4.8.1 Visualisasi Model Data Validasi

diawali dengan menyiapkan ukuran *canvas* menggunakan plt.figure() dan mengambil daftar nama kelas dari val_generator. Sebanyak 9 citra diambil dari batch pertama data validasi. Untuk setiap citra, label sebenarnya (*true_label*) diambil dari data label, sedangkan label prediksi (*pred_label*) diperoleh dengan memberikan citra tersebut ke fungsi

model.predict() dan membandingkan hasil probabilitasnya dengan ambang batas 0,5. Jika prediksi sesuai dengan label sebenarnya, judul gambar diberi warna hijau, sedangkan jika salah diberi warna merah. Semua citra ditampilkan dalam bentuk grid 3×3, dilengkapi dengan informasi label prediksi dan label sebenarnya.



Gambar 4. 13 Prediksi Data Validasi

Gambar 4.13 di atas memperlihatkan enam contoh citra dari data validasi beserta label prediksi dan label sebenarnya. Tulisan berwarna hijau menunjukkan prediksi yang benar, sedangkan tulisan merah menandakan kesalahan prediksi. Terlihat bahwa mayoritas citra berhasil diklasifikasikan dengan tepat oleh model, baik untuk kelas *Parasitized* maupun *Uninfected*. Namun, terdapat beberapa kesalahan prediksi, misalnya pada citra baris

kedua kolom pertama, di mana model memprediksi *Uninfected* padahal label sebenarnya adalah *Parasitized*. Kesalahan seperti ini umumnya terjadi pada citra yang memiliki ciri visual samar atau mirip dengan kelas lain, misalnya kontras warna rendah atau bentuk parasit yang tidak jelas. Secara keseluruhan, visualisasi ini menguatkan hasil evaluasi kuantitatif bahwa model ResNet-50 dengan transfer learning memiliki performa tinggi dalam mendeteksi sel darah yang terinfeksi malaria, meskipun masih terdapat ruang untuk perbaikan terutama pada kasus-kasus dengan ciri visual ambigu.

4.8.2 Visualisasi Model Data Uji

Visualisasi hasil prediksi model pada data uji bertujuan untuk memberikan gambaran secara langsung mengenai kemampuan model dalam mengklasifikasikan citra sel darah yang terinfeksi malaria (*Parasitized*) dan yang tidak terinfeksi (*Uninfected*). Melalui visualisasi ini, setiap gambar dilengkapi dengan informasi label prediksi model dan label sebenarnya, sehingga dapat dengan mudah diidentifikasi apakah model berhasil atau gagal dalam melakukan klasifikasi. Pendekatan ini tidak hanya membantu dalam mengevaluasi kinerja model secara kuantitatif, tetapi juga memberikan wawasan kualitatif mengenai jenis kesalahan yang terjadi, misalnya kesalahan akibat kemiripan visual antara kedua kelas. Dengan demikian, visualisasi prediksi pada data uji menjadi bagian penting dalam analisis menyeluruh terhadap performa model yang dikembangkan.

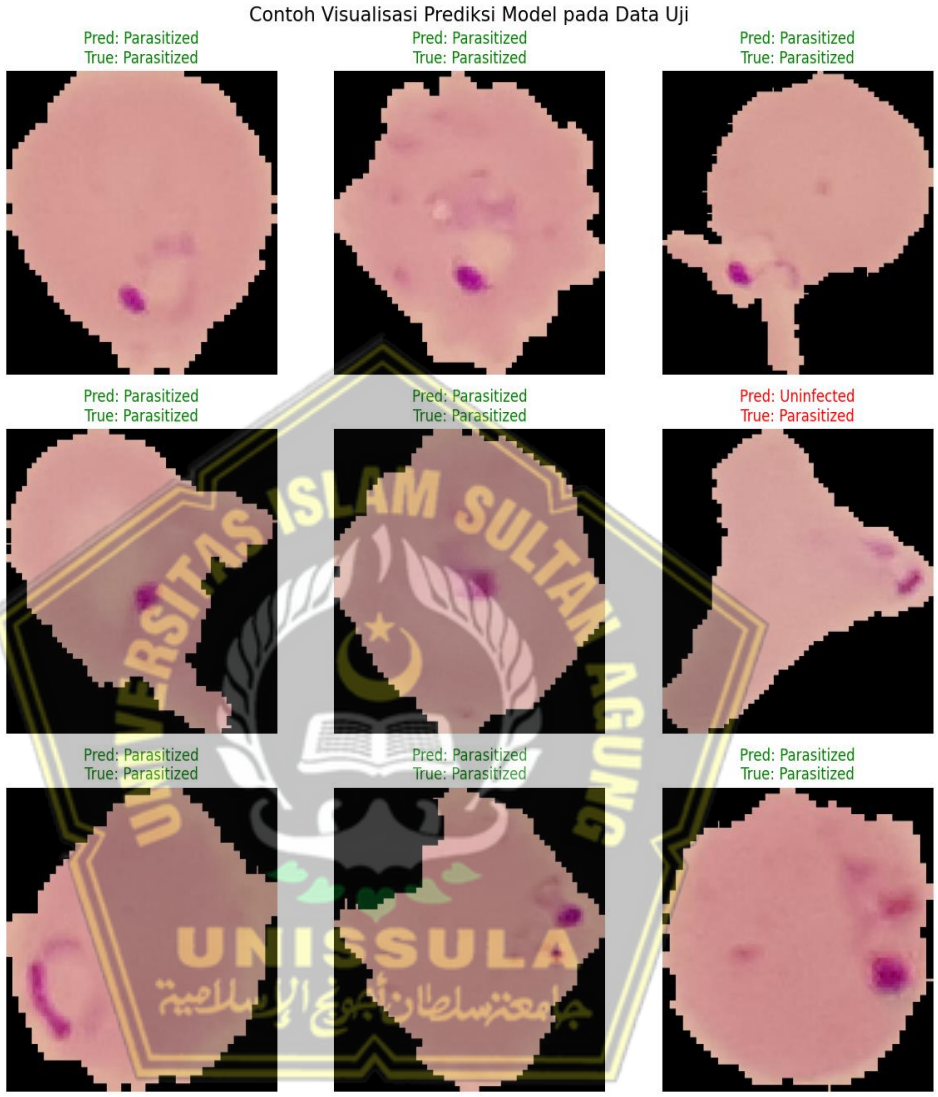
Dengan kode yang berfungsi untuk digunakan menampilkan contoh hasil prediksi model pada data uji (*test set*) sehingga dapat diamati secara langsung bagaimana kinerja model dalam membedakan citra sel darah yang terinfeksi (*Parasitized*) dan tidak terinfeksi (*Uninfected*). Pertama, ukuran kanvas gambar ditentukan dengan plt.figure(figsize=(12, 12)) agar setiap gambar terlihat jelas. Daftar nama kelas diambil dari test_generator.class_indices.keys() untuk digunakan sebagai label. Selanjutnya, satu batch pertama dari *test set* diambil menggunakan images,

labels = test_generator[0] tanpa proses pengacakan (*shuffle*), sehingga gambar yang ditampilkan adalah urutan asli dari batch tersebut.

Kemudian, dilakukan iterasi untuk mengambil 9 gambar pertama. Setiap gambar diproses dengan memperluas dimensinya menggunakan np.expand_dims agar sesuai dengan format input model, lalu diprediksi dengan model.predict. Hasil prediksi berbentuk probabilitas, sehingga digunakan batas nilai 0,5 untuk mengklasifikasikannya menjadi salah satu kelas. Label prediksi (pred_label) dibandingkan dengan label sebenarnya (true_label), dan warna judul ditentukan: hijau jika prediksi benar, merah jika salah. Setiap gambar ditampilkan dengan plt.imshow dan diberi judul yang memuat informasi "Pred" (hasil prediksi) dan "True" (label sebenarnya).

Terakhir, seluruh subplot diberi judul utama "Contoh Visualisasi Prediksi Model pada Data Uji" dengan plt.suptitle, tata letak diatur menggunakan plt.tight_layout() agar tidak saling bertumpuk, dan hasil visualisasi ditampilkan dengan plt.show(). Melalui langkah ini, pengguna dapat langsung mengidentifikasi tingkat akurasi visual model serta

mengamati contoh kasus di mana model berhasil atau gagal melakukan prediksi.



Gambar 4. 14 Visualisasi Data Uji

Berdasarkan gambar 4.14 hasil visualisasi pada model menunjukkan kinerja yang sangat baik pada data uji. Dari 9 sampel gambar yang ditampilkan, 8 gambar berhasil diklasifikasikan dengan benar (label prediksi sesuai label sebenarnya) dan hanya 1 gambar yang salah prediksi. Pada kasus benar, label "Pred" dan "True" sama-sama berwarna hijau, menandakan kecocokan antara prediksi dan label asli. Sementara pada kasus salah prediksi, tulisan berwarna merah terlihat pada label prediksi maupun label sebenarnya, menandakan ketidaksesuaian.

Mayoritas gambar yang ditampilkan berasal dari kelas Parasitized, dan model mampu mengenali ciri-ciri sel terinfeksi dengan akurat, seperti adanya bentuk ungu di dalam sel darah yang mengindikasikan keberadaan parasit malaria. Kesalahan prediksi yang terjadi kemungkinan disebabkan oleh citra yang memiliki ciri visual samar atau terlalu mirip dengan kelas lain, sehingga membingungkan model. Visualisasi ini membantu memastikan bahwa performa model tidak hanya dinilai dari angka akurasi, tetapi juga dari pengamatan langsung terhadap contoh prediksi pada citra nyata, sehingga memberikan gambaran yang lebih intuitif tentang kekuatan dan kelemahan model.

4.8.3 Visualisasi Gambar Manual

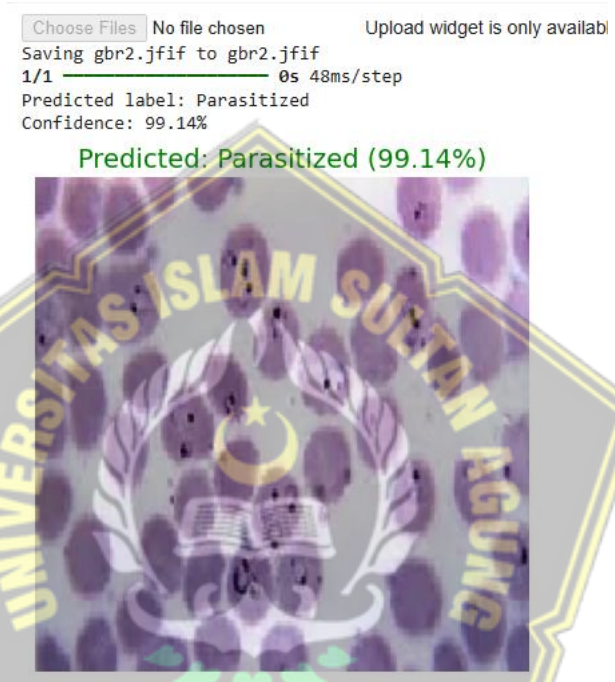
Pada tahap ini, dilakukan pengujian model menggunakan citra tunggal yang diunggah secara manual oleh pengguna. Tujuannya adalah untuk mengetahui bagaimana model memprediksi gambar baru yang tidak termasuk dalam dataset pelatihan, validasi, maupun pengujian. Proses ini berguna dalam simulasi penggunaan model di dunia nyata, di mana citra akan diambil dari sumber eksternal seperti hasil mikroskop, kemudian diproses untuk mendapatkan klasifikasi apakah sel darah termasuk kategori Parasitized atau Uninfected.

Kode diawali dengan proses unggah gambar melalui files.upload() sehingga pengguna dapat memilih file dari perangkat lokal. Setiap gambar yang diunggah kemudian diproses menggunakan image.load_img() untuk menyesuaikan ukuran menjadi 224×224 piksel, sesuai dimensi input yang digunakan oleh arsitektur ResNet-50. Selanjutnya, gambar diubah menjadi array numerik dengan image.img_to_array(), dinormalisasi ke rentang [0,1], dan ditambahkan dimensi batch menggunakan np.expand_dims() agar sesuai format input model.

Proses prediksi dilakukan dengan model.predict() yang menghasilkan probabilitas. Nilai probabilitas ini dibandingkan dengan ambang batas 0,5 untuk menentukan label prediksi. Jika nilai lebih kecil dari 0,5 maka gambar diklasifikasikan sebagai Parasitized, sedangkan jika lebih besar atau sama

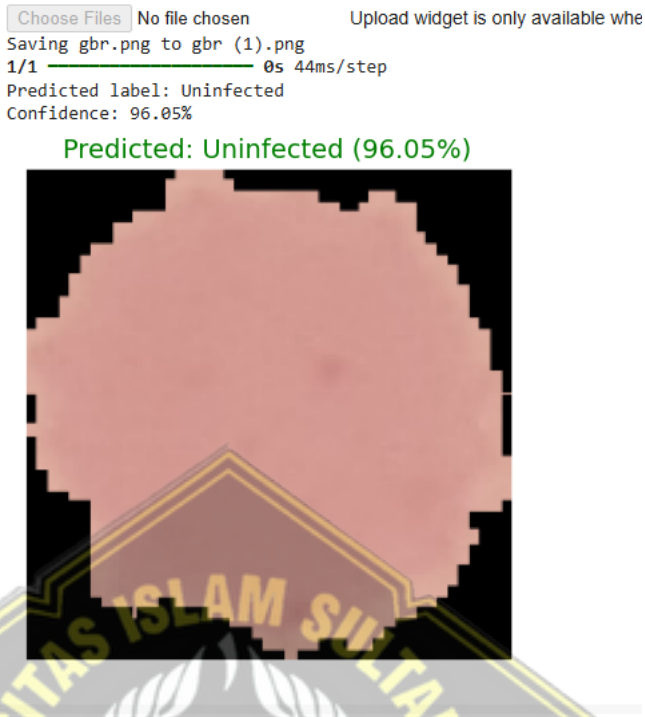
dengan 0,5 diklasifikasikan sebagai Uninfected. Tingkat keyakinan (*confidence*) dihitung dan ditampilkan dalam bentuk persentase.

Terakhir, gambar divisualisasikan menggunakan matplotlib.pyplot dengan judul yang memuat hasil prediksi beserta persentase keyakinannya. Dengan cara ini, pengguna dapat secara langsung melihat hasil analisis model terhadap gambar yang baru diunggah.



Gambar 4. 15 Hasil Visualisasi Prediksi Manual

Pada gambar 4.15, model berhasil prediksi gambar diluar pelatihan dengan baik, dengan hasil predicted : Parasitized (99.14).



Gambar 4. 16 Hasil Visualisasi Prediksi Manual 2

Pada gambar 4.16, model berhasil prediksi gambar diluar pelatihan dengan baik, dengan hasil predicted : Uninfected (96.05%).

Hasil pengujian pada citra tunggal menunjukkan bahwa model mampu memberikan prediksi secara cepat dan akurat terhadap gambar baru yang belum pernah dilihat sebelumnya. Dengan demikian, metode ini memiliki potensi besar untuk diterapkan dalam lingkungan nyata sebagai alat bantu deteksi dini malaria. Selain itu, tingkat keyakinan yang ditampilkan dapat membantu pengguna dalam menilai reliabilitas prediksi yang diberikan. Temuan ini kemudian akan dianalisis lebih lanjut pada bagian pembahasan untuk mengevaluasi konsistensi performa model terhadap berbagai jenis citra, termasuk kemungkinan kesalahan klasifikasi serta faktor-faktor yang mempengaruhinya.

4.8.4 Prediksi Gambar Tunggal dengan Penyimpanan Otomatis ke CSV dan PDF

Pada bagian ini, sistem deteksi malaria dikembangkan dengan tambahan fitur arsip prediksi otomatis yang mampu menyimpan setiap hasil klasifikasi ke dalam dua bentuk berkas, yaitu CSV dan PDF. Fitur ini dibuat

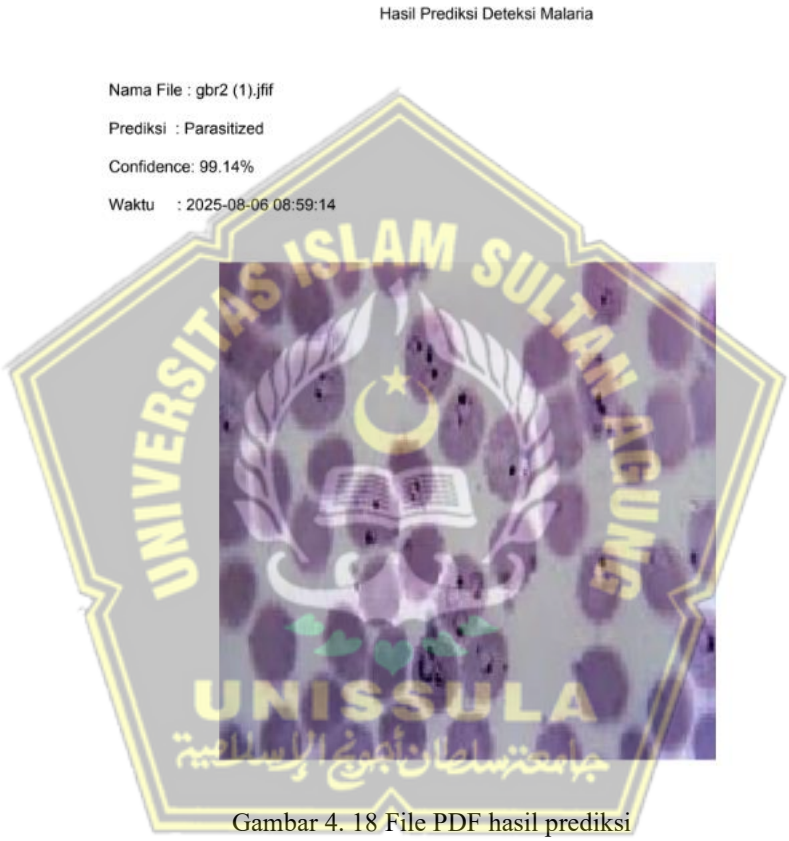
dengan tujuan agar setiap prediksi yang dilakukan tidak hanya ditampilkan secara visual di layar, tetapi juga terdokumentasi secara permanen untuk keperluan pelaporan, validasi, maupun analisis lebih lanjut. Dalam konteks penelitian, pencatatan hasil prediksi dalam format CSV akan mempermudah proses kompilasi data untuk evaluasi model, sedangkan laporan PDF yang dilengkapi gambar hasil klasifikasi dapat menjadi bukti visual yang memudahkan verifikasi manual oleh pihak terkait, seperti tenaga medis atau peneliti lain.

Dengan kode program yang ditampilkan menggunakan modul Google Colab files untuk mengunggah gambar, Keras Image Preprocessing untuk memuat dan menormalisasi citra, serta memanfaatkan model ResNet-50 hasil pelatihan sebelumnya untuk melakukan prediksi. Proses dimulai dengan mengunggah citra sel darah, kemudian citra diubah ukurannya menjadi 224×224 piksel dan dinormalisasi ke rentang [0,1] agar sesuai dengan input model. Setelah citra diproses, model memberikan probabilitas prediksi, yang kemudian dikonversi menjadi label “Parasitized” atau “Uninfected” berdasarkan ambang batas 0,5. Nilai probabilitas ini juga dikonversi menjadi tingkat keyakinan (confidence) dalam bentuk persentase.

Selanjutnya, hasil prediksi beserta nama file, confidence, dan timestamp disimpan ke dalam file CSV di Google Drive. CSV ini berfungsi sebagai arsip data yang dapat digunakan untuk analisis statistik, evaluasi kinerja model, atau pembuatan grafik. Selain itu, sistem juga menghasilkan laporan PDF yang memuat teks hasil prediksi dan gambar citra asli yang diprediksi. PDF ini diformat menggunakan library FPDF, sehingga memiliki tampilan terstruktur dan profesional. Untuk menjaga kebersihan direktori kerja, gambar sementara yang digunakan dalam proses pembuatan PDF akan dihapus setelah dokumen selesai dibuat.

	A	B	C	D
1	Filename	Prediction	Confidence (%)	Timestamp
2	gbr2 (1).jfif	Parasitized	99.14	2025-08-06 8:59:14
3	gbr.png	Uninfected	96.05	2025-08-06 9:01:32
4	gbr3 (1).jfif	Parasitized	75.15	2025-08-06 9:03:09
5	gbr3 (2).jfif	Parasitized	75.15	2025-08-06 9:10:13
6	gbr4.jfif	Uninfected	90.2	2025-08-06 9:11:36

Gambar 4. 17 File CSV hasil prediksi



Gambar 4. 18 File PDF hasil prediksi

Gambar 4. 19 File PDF hasil prediksi 2

Dari gambar 4.17 – 4.19 dapat disimpulkan bahwa Implementasi fitur penyimpanan hasil prediksi dalam format CSV dan PDF memberikan keuntungan signifikan pada sistem deteksi malaria. CSV memungkinkan pengumpulan data prediksi dalam jumlah besar untuk kebutuhan analisis dan penghitungan metrik evaluasi model, sedangkan PDF mempermudah proses komunikasi hasil kepada pihak yang tidak terlibat langsung dalam pengembangan sistem. Dengan adanya dokumentasi terstruktur ini, setiap prediksi yang dihasilkan oleh sistem dapat dilacak kembali (*traceable*) dan memiliki bukti visual yang sah. Hal ini tidak hanya meningkatkan profesionalisme sistem, tetapi juga mendukung integritas data dalam penelitian, memastikan bahwa hasil yang diperoleh dapat dipertanggungjawabkan secara ilmiah.

4.9 Hasil Deployment

Sebagai tahap akhir implementasi, model klasifikasi malaria yang telah dilatih dan dievaluasi di-deploy menjadi sebuah aplikasi web interaktif. Langkah ini bertujuan untuk menjadikan model tidak hanya

sebatas kode eksperimen, tetapi juga sebuah sistem fungsional yang dapat diakses dengan mudah oleh pengguna maupun tenaga medis. Deployment dilakukan menggunakan framework Streamlit, sebuah pustaka Python yang mendukung pengembangan aplikasi web sederhana dan interaktif secara cepat untuk kebutuhan data sains maupun machine learning.

Proses deployment dimulai dengan menyimpan model terbaik (best_model_80split.h5) hasil pelatihan arsitektur ResNet-50. Model ini kemudian dimuat kembali ke dalam aplikasi utama melalui skrip app.py yang dirancang untuk mengatur seluruh alur antarmuka pengguna. Logika sistem mencakup fungsi untuk memproses citra mikroskopis sel darah, melakukan prediksi klasifikasi ke dalam dua kelas (*Parasitized* dan *Uninfected*), serta menampilkan hasil berupa label prediksi, probabilitas, dan citra input di layar antarmuka. Selain itu, aplikasi juga dilengkapi dengan fitur penyimpanan log hasil prediksi dalam format CSV, serta ekspor hasil prediksi ke file PDF sebagai laporan digital.

Aplikasi dijalankan secara lokal melalui perintah terminal streamlit run app.py, sehingga penelitian ini berhasil diwujudkan menjadi sebuah sistem praktis yang dapat diakses melalui peramban web pada alamat localhost. Hasil akhir dari proses ini adalah sebuah sistem deteksi malaria berbasis deep learning yang tidak hanya berfungsi sebagai alat bantu diagnosis awal, tetapi juga mendukung dokumentasi serta pelaporan digital di laboratorium.

Sistem ini ditujukan sebagai *decision support system* bagi tenaga laboratorium atau peneliti medis, sehingga hasil prediksi dapat menjadi acuan awal sebelum dilakukan pemeriksaan lanjutan secara manual. Selain itu, sistem ini juga dapat membantu meningkatkan efisiensi waktu analisis, menyediakan laporan digital dalam bentuk PDF untuk keperluan dokumentasi, serta berpotensi digunakan sebagai alat skrining awal di daerah dengan keterbatasan tenaga analis.

1. Halaman Awal

Gambar 4. 20 Halaman Awal

Pada Gambar 4.20 ditampilkan halaman utama dari sistem deteksi malaria yang telah berhasil di-deploy menjadi sebuah aplikasi web interaktif menggunakan framework Streamlit. Sistem ini dirancang dengan antarmuka yang sederhana namun fungsional agar mudah digunakan. Pada bagian atas, terdapat judul aplikasi "Deteksi Malaria dengan Model ResNet-50" yang menunjukkan identitas system.

Di bawah judul, pengguna dapat mengunggah citra sel darah merah dalam format JPG, JPEG, atau PNG melalui fitur Drag and Drop File atau tombol Browse files. Selain itu, tersedia pula kolom input nama pasien untuk mencatat identitas pasien yang akan diprediksi.

Bagian bawah aplikasi menampilkan menu Riwayat Prediksi, yang memungkinkan pengguna untuk memfilter hasil prediksi berdasarkan nama pasien. Fitur ini berfungsi sebagai arsip digital yang menyimpan riwayat deteksi malaria, sehingga hasil prediksi dapat ditinjau kembali maupun diekspor ke dalam format CSV atau PDF untuk kebutuhan pelaporan

laboratorium. Dengan desain antarmuka ini, sistem mampu memberikan pengalaman interaktif yang mendukung kebutuhan diagnostik awal malaria secara lebih cepat, praktis, dan terdokumentasi.

2. Halaman Hasil Prediksi

	Waktu	Nama Pasien	Nama File	Prediksi	Probabilitas	Raw Output
14	2025-08-30 14:20:49	riko	cobagbr.png	Parasitized	99.93%	0.0007
15	2025-08-30 14:20:57	jio	cobagbr.png	Parasitized	99.93%	0.0007
16	2025-08-30 14:31:26	jio	cobaaab.jpeg	Parasitized	92.94%	0.0706
17	2025-08-31 10:04:55	febro	cobaaab.jpeg	Parasitized	92.94%	0.0706
18	2025-08-31 10:05:42	febro	cobaaab.jpeg	Parasitized	92.94%	0.0706
19	2025-08-31 10:05:51	febro	cobaaab.jpeg	Parasitized	92.94%	0.0706
20	2025-08-31 10:08:12	febro	cobaaab.jpeg	Parasitized	92.94%	0.0706
21	2025-08-31 10:10:19	febro	cobaaab.jpeg	Parasitized	92.94%	0.0706
22	2025-08-31 10:10:42	febro	cobaaab.jpeg	Parasitized	92.94%	0.0706
23	2025-08-31 11:22:16	zevin	cobaaab.jpeg	Parasitized	92.94%	0.0706

Gambar 4. 21 Halaman Prediksi

Gambar 4.21 menampilkan hasil prediksi dari sistem deteksi malaria berbasis ResNet-50 yang telah di-deploy menggunakan Streamlit. Pada tampilan ini, pengguna dapat memasukkan nama pasien serta mengunggah citra sel darah merah untuk dianalisis. Setelah gambar diproses oleh model, sistem memberikan output berupa hasil prediksi dengan dua kemungkinan kelas, yaitu *Parasitized* dan *Uninfected*. Selain label prediksi, ditampilkan pula nilai probabilitas prediksi dalam bentuk persentase, serta nilai *raw sigmoid output* yang menunjukkan keluaran asli dari fungsi aktivasi model. Sebagai contoh, pada gambar ditunjukkan hasil prediksi dengan label *Parasitized* dengan tingkat probabilitas 92,94%. Selain informasi numerik, citra input juga ditampilkan kembali pada antarmuka sebagai bentuk visualisasi data yang dianalisis, Sebagai tambahan, sistem ini juga dilengkapi dengan fitur unduh hasil dalam bentuk PDF, sehingga laporan prediksi dapat dicetak atau diberikan langsung kepada pasien sebagai dokumen pendukung hasil analisis.

Selanjutnya juga bisa menampilkan fitur *Riwayat Prediksi* yang berfungsi sebagai log seluruh aktivitas prediksi yang dilakukan oleh sistem. Riwayat ini tersimpan dalam bentuk tabel yang memuat informasi waktu prediksi, nama pasien, nama file citra, hasil klasifikasi, probabilitas prediksi, serta nilai *raw output* dari sigmoid. Dengan adanya fitur ini, pengguna maupun tenaga medis dapat melakukan pelacakan kembali terhadap data pasien yang telah diuji, sekaligus menjadi dokumentasi digital yang akurat. Sistem juga menyediakan opsi untuk mengunduh riwayat prediksi dalam format CSV, sehingga data dapat dikelola lebih lanjut untuk kepentingan analisis maupun arsip.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan mengenai Implementasi Metode Transfer Learning untuk Deteksi Malaria Menggunakan ResNet-50, dapat disimpulkan bahwa metode ini mampu memberikan kinerja yang baik dalam mengklasifikasikan citra sel darah ke dalam dua kelas, yaitu *Parasitized* dan *Uninfected*. Hasil evaluasi menunjukkan bahwa model ResNet-50 mencapai akurasi sebesar 93%, presisi 96%, recall 97%, F1-score 93%, serta nilai AUC 98%. Nilai-nilai ini menunjukkan bahwa model memiliki kemampuan generalisasi yang baik, sensitif dalam mendeteksi sel terinfeksi, serta cukup seimbang antara presisi dan recall. Evaluasi visual melalui grafik akurasi dan loss juga memperlihatkan tren stabil, tanpa indikasi overfitting signifikan. Dengan demikian, tujuan penelitian untuk membangun dan mengevaluasi model klasifikasi malaria berbasis ResNet-50 dapat dikatakan tercapai, meskipun pengujian masih terbatas pada dataset tertentu dan dapat dikembangkan lebih lanjut dengan data yang lebih beragam dan aplikasi langsung di laboratorium.

5.2 Saran

Adapun saran yang dapat diberikan untuk penelitian selanjutnya adalah perlunya penambahan variasi dataset, baik dari segi jumlah maupun keragaman sumber citra, agar model dapat beradaptasi pada berbagai kondisi gambar dan tingkat kualitas yang berbeda. Penelitian berikutnya juga dapat mencoba arsitektur *deep learning* lainnya, seperti EfficientNet, DenseNet, atau Vision Transformer, untuk membandingkan kinerja dan efektivitasnya dengan ResNet-50. Selain itu, optimasi *hyperparameter* melalui teknik seperti *learning rate scheduling*, penyesuaian *batch size*, atau pemilihan *optimizer* yang lebih tepat diharapkan dapat meningkatkan akurasi dan mempercepat proses pelatihan. Penggunaan teknik augmentasi lanjutan, seperti *Generative Adversarial Networks (GAN)* atau *style*

transfer, juga dapat menjadi alternatif untuk memperkaya data latih dengan citra sintetis yang realistis.

DAFTAR PUSTAKA

Adoma, A. F., Henry, N. M. dan Chen, W. (2020) "Comparative Analyses of Bert, Roberta, Distilbert, and Xlnet for Text-Based Emotion Recognition," *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing, ICCWAMTIP 2020*, hal. 117–121. doi: 10.1109/ICCWAMTIP51612.2020.9317379.

Aisyah, D. N. *et al.* (2024) "The Changing Incidence of Malaria in Indonesia: A 9-Year Analysis of Surveillance Data," *Advances in Public Health*, 2024(1). doi: 10.1155/adph/2703477.

Alarcón-Soto, Y. *et al.* (2020) "Data Science in Biomedicine," hal. 1–20. Tersedia pada: http://arxiv.org/abs/1909.04486.

Alzubaidi, L. *et al.* (2021) *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions*, *Journal of Big Data*. Springer International Publishing. doi: 10.1186/s40537-021-00444-8.

Fransisca, L. *et al.* (2025) "Enhanced data quality to improve malaria surveillance in Papua, Indonesia," *Malaria Journal* , 24(1). doi: 10.1186/s12936-025-05358-x.

Gao, Y. *et al.* (2023) "Distinguishing infectivity in patients with pulmonary tuberculosis using deep learning," *Frontiers in Public Health*, 11(1). doi: 10.3389/fpubh.2023.1247141.

Ho, Y. dan Wookey, S. (2020) "The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling," *IEEE Access*, 8, hal. 4806–4813. doi: 10.1109/ACCESS.2019.2962617.

Hussain, M., Bird, J. J. dan Faria, D. R. (2020) "A study on CNN transfer learning for image classification," *Advances in Intelligent Systems and Computing*, 840, hal. 191–202. doi: 10.1007/978-3-319-97982-3_16.

Maeda-Gutiérrez, V. *et al.* (2020) "Comparison of convolutional neural network architectures for classification of tomato plant diseases," *Applied Sciences (Switzerland)*, 10(4). doi: 10.3390/app10041245.

Mukti, I. Z. dan Biswas, D. (2020) "Transfer Learning Based Plant Diseases Detection Using ResNet50," *2019 4th International Conference on Electrical Information and Communication Technology, EICT 2019*, (December), hal. 1–6. doi: 10.1109/EICT48899.2019.9068805.

Nagendra, S. *et al.* (2024) "Techniques Versus Standard Microscopy in a Reference Laboratory," (December).

Opoku Afriyie, S. *et al.* (2023) "Accuracy of diagnosis among clinical malaria patients: comparing microscopy, RDT and a highly sensitive quantitative PCR looking at the implications for submicroscopic infections," *Malaria Journal*, 22(1), hal. 1–11. doi: 10.1186/s12936-023-04506-5.

Rahman, T. *et al.* (2020) "Transfer learning with deep Convolutional Neural Network (CNN) for pneumonia detection using chest X-ray," *Applied Sciences (Switzerland)*, 10(9). doi: 10.3390/app10093233.

Ramadhan, F. M. dan Riwurohi, J. E. (2024) "Perbandingan Optimizer, Batch Size Dan Epoch Pada Metode Convolution Neural Network," *JSiI (Jurnal Sistem Informasi)*, 11(2), hal. 134–138. doi: 10.30656/jsii.v11i2.9249.

Ruicun Liu, Tuoyu Liu, Tingting Dan, ..., Xianchao Zhang, H. C. dan Teng, Y. (2023) "AIDMAN: An AI-based object detection system for malaria diagnosis from smartphone thin-bloodsmear images."

Shanqing, G., Pednekar, M. dan Slater, R. (2020) "Improve Image Classification Using Data Augmentation and Neural Networks," *SMU Data Science Review*, 2(2), hal. 1–43. Tersedia pada: https://scholar.smu.edu/datasciencereviewhttp://digitalrepository.smu.edu.Availableat:https://scholar.smu.edu/datasciencereview/vol2/iss2/1.

Sinha, S. dan Gupta, N. (2023) "Computer-aided Diagnosis of Malaria through Transfer Learning using the ResNet50 Backbone," 36, hal. 1–9.

Subramanian, S., Ganapathiraman, V. dan Barrett, C. (2024) "Hop, skip,

jump to Convergence: Dynamics of Learning Rate Transitions for Improved Training of Large Language Models," *EMNLP 2024 - 2024 Conference on Empirical Methods in Natural Language Processing, Findings of EMNLP 2024*, hal. 16349–16362. doi: 10.18653/v1/2024.findings-emnlp.954.

Tegegne, B. *et al.* (2025) "Performance of BIOCREDIT Pf/Pv lactate dehydrogenase-based malaria rapid diagnostic test among pregnant women with suspected malaria infection in Bahir Dar City Administration, northwest Ethiopia," *PLoS ONE*, 20(5 May), hal. 1–13. doi: 10.1371/journal.pone.0322362.

Thabtah, F. *et al.* (2020) "Data imbalance in classification: Experimental evaluation," *Information Sciences*, 513, hal. 429–441. doi: 10.1016/j.ins.2019.11.004.

Tilasefana, R. A. dan Putra, R. E. (2023) "Penerapan Metode Deep Learning Menggunakan Algoritma CNN Dengan Arsitektur VGG NET Untuk Pengenalan Cuaca," *Journal of Informatics and Computer Science (JINACS)*, 05(1), hal. 48–57.

WHO (2020) "The 2023 WHO World malaria report," *The Lancet Microbe*, 5(3), hal. e214. doi: 10.1016/s2666-5247(24)00016-8.

WHO (2021) *Word Malaria Report 2021*, *Word Malaria report Geneva: World Health Organization. (2021). Licence: CC BY-NC-SA 3.0 IGO*. Tersedia pada: https://www.who.int/teams/global-malaria-programme/reports/world-malaria-report-2021.

Wu, Y. dan Liu, L. (2023) "Selecting and Composing Learning Rate Policies for Deep Neural Networks," *ACM Transactions on Intelligent Systems and Technology*, 14(2). doi: 10.1145/3570508.

Zeng, G. (2020) "On the confusion matrix in credit scoring and its analytical properties," *Communications in Statistics - Theory and Methods*, 49(9), hal. 2080–2093. doi: 10.1080/03610926.2019.1568485.

Zhuang, Q. *et al.* (2021) "Zero-shot Event Extraction via Transfer Learning: Challenges and Insights," *ACL-IJCNLP 2021 - 59th Annual Meeting of*

the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference, 2, hal. 322–332. doi: 10.18653/v1/2021.acl-short.42.

Zufar, M. (2020) "Introductory Computer Vision and Image Processing," *Sensor Review*, 18(3), hal. 2–4. doi: 10.1108/sr.1998.08718cae.001.