ANALISIS PERBANDINGAN ARSITEKTUR LENET DAN RESNET UNTUK KLASIFIKASI MALARIA

SKRIPSI

OLEH :

ALEX SIMANUNGKALIT

198160064

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MEDAN AREA

2024

ANALISIS PERBANDINGAN ARSITEKTUR LENET DAN RESNET UNTUK KLASIFIKASI MALARIA

SKRIPSI

Diajukan sebagai Salah Satu Syarat untuk Memperoleh

Gelar Sarjana di Fakultas Teknik

Universitas Medan Area

Oleh :

ALEX SIMANUNGKALIT

198160064

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MEDAN AREA

2024

LEMBAR PENGESAHAAN

Judul Skripsi : Analisis Perbandingan Arsitektur Lenet Dan Resnet Untuk Klasifikasi Malaria

Nama : Alex Simanungkalit

Npm : 198160064

Fakultas : Teknik

Disetujui Oleh
Pembimbing

Nurul Khairina,S.Kom,M.Kom
Pembimbing

Dr. Eng. Supriatno, S.T, M.T,
Dekan

Rizki Muliono S.Kom, M.Kom
Ka Prodi

Tanggal Lulus : 27 Maret 2024

HALAMAN PERNYATAAN

Saya menyatakan bahwa skripsi yang saya susun, sebagai syarat memperoleh gelar sarjana merupakan hasil karya tulis saya sendiri. Adapun bagian-bagian tertentu dalam penulisan skripsi ini yang saya kutip dari hasil karya orang lain setelah di tulis sumbernya secara jelas sesuai dengan norma,kaidah, dan etika penulis ilmiah.

Saya bersedia menerima sanksi pencabutan gelar akademik yang saya peroleh dan sanksi lainya dengan peraturan yang berlaku, apabila di kemudian hari ditemukan adanya plagiat dlam skripsi ini

Medan, 27 Maret 2024

Alex Simanungkalit
198160064

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR/SKRIPSI/TESIS KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Medan Area, Saya bertanda tangan dibawah ini:

Nama	: Alex Simanungkalit
Npm	:19816006
Program Studi	: Teknik Informatika
Fakultas	: Teknik
Jenis Karya	: Skripsi

Demi Pengembagan ilmu pengetahuan, menyetujui untuk memberikan kepada Univeristas Medan **Area Hak Bebas Royalti Nonekslutif** (***Non-exlusive Royalty-Free Right***) atas karya ilmiah saya yang berjudul :

Analisis perbandingan arsitektur LeNet dan ResNet untuk klasifikasi

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneklusif ini Universitas Medan Area berhak menyimpan, mengalih media/format-kan, mengelola dalam bentuk pengkalan data (*databases*), merawat, dan memmublikasikan skripsi saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagi pemilik Hak Cipta

Demikan peryataan ini saya buat dengan sebenarnya.

Dibuat di : Medan
Pada Tanggal : 27 Maret 2024
Yang Menyatakan

Alex Simanungkalit
198160064

Document Accepted 17/9/24

ABSTRAK

Penyakit yang akhir-akhir ini menjadi perbincangan hangat masyarakat di seluruh dunia ditemukan pertama kali pada negara Afrika dan negara lainnya mengenai malaria dan ada yang kemudian yang mengakibatkan malaria seperti Parasit yang berpotensi fatal penyebab malaria menginfeksi nyamuk Anopheles betina dan menyebar ke manusia melalui gigitan. Menurut perkiraan, terdapat 229 juta kasus malaria secara global pada tahun 2019 dan 409.000 orang meninggal akibat penyakit tersebut. Dengan adanya alat tersebut dapat membantu masyarakat agar lebih cepat dan mudah mengetahui penyakit yang diderita. Oleh karena itu, diperlukan sistem yang dapat menganalisa, mengenali, secara sensitive, akurat dan otomatis mendiagnosa manusia terkena penyakit malaria atau tidak. Metode yang disulkan untuk menyelesaikan permasalahan tersebut yaitu arsitektur LeNet dan ResNet untuk klasifikasi. Dalam penelitian ini, mengusulkan penggunaan arsitektur LeNet dan ResNet untuk membantu dalam mengklasifikasikan penyakit malaria. Dataset terdiri dari 6000 gambar sel darah yang terdiri daru 2 class. Pada scenario model menggunakan hyperparameter dengan epoch 20, batch size 64, optimizer Adam, learning rate 0.001. Model yang diusulkan mencapai kinerja dengan akurasi terbaik dengan akurasi LeNet yaitu 96% dan akurasi pada ResNet yaitu 100%. Pengujiannya berhasil serta berjalan dengan baik.

Kata Kunci: Klasifikasi, Malaria, ResNet, LeNet

ABSTRACT

The disease that has recently become a hot topic of discussion in communities around the world was first discovered in African countries and other countries regarding malaria and there are later resulting in malaria such as potentially fatal parasites that cause malaria infect female Anopheles mosquitoes and spread to humans through bites. According to estimates, there were 229 million cases of malaria globally in 2019 and 409,000 people died from the disease. With the existence of these tools, it can help people to find out more quickly and easily about the disease they are suffering from. Therefore, a system is needed that can analyze, recognize, sensitively, accurately and automatically diagnose whether humans have malaria or not. The proposed method to solve this problem was the LeNet and ResNet architecture for classification. In this study, proposed the use of LeNet and ResNet architecture to assist in classifying malaria. The dataset consists of 6000 blood cell images consisting of 2 classes. The model scenario used hyperparameters with epoch 20, batch size 64, optimizer Adam, learning rate 0.001. The proposed model achieves the best accuracy performance with LeNet accuracy of 96% and ResNet accuracy of 100%. The test is successful and runs well.

Keywords: *Classification, Malaria, ResNet, LeNet*

RIWAYAT HIDUP

Penulis dilahirkan di Medan Pada tanggal 23 Maret 2001 dari ayah Miduk Simanungkalit dan ibu Reliani Hutagaol Penulis merupakan putra pertama dari 3 bersaudara.penulis pertama kali mengenyam pendidikan di bangku SD Negeri 1 Patumbak pada tahun 2007- 2013, meneruskan pendidikan di SMP Swasta Panti Harapan 2013-2015, kemudian pindah ke SMP Negeri 4 Lawe Sigalgala pada tahun2015-2016 kemudian Tahun 2016-2019 Penulis lulus dari SMK Negeri 1 Patumbak dan pada tahun 2019 terdaftar sebagai mahasiswa Fakultas Teknik Informatika Universitas Medan Area.

Selama mengikuti perkuliahan, penulis menjadi asisten mata kuliah pada tahun ajaran kemudian mendapatkan beasiswa Bank Indonesia angkatan 9.0 Penulis melaksanakan praktek kerja lapangan (PKL) di Telkom STO Cinta Damai

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Kuasa atas segala karuniaNya sehingga skripsi ini berhasil diselesaikan. Tema yang dipilih dalam penelitian ini ialah *Deep Learning* dengan judul "Analisis perbandingan arsitektur LeNet dan ResNet Untuk malaria ".

Skripsi ini adalah salah satu syarat untuk menyelesaikan pendidikan untuk mencapai gelar sarjana di Program Studi Teknik Informatika Fakultas Teknik Universitas Medan Area. Pada kesempatan ini penulis mengucapkan terima kasih kepada :

1. Bapak Prof. Dr. Dadan Ramdan, M.Eng, M.Sc. selaku Rektor Universitas Medan Area.
2. Bapak Dr.Eng.,Suprianto,S.T,M selaku Dekan Fakultas Teknik Universitas Medan Area.
3. Bapak Rizki Muliono, S.Kom., M.Kom selaku Kepala Program Studi Teknik Informatika Universitas Medan Area.
4. Ibu Nurul Khairina,S.Kom,M.Kom selaku Dosen pembimbing yang telah membantu penulis dari segi materi dan moril sehingga penulis dapat menyelesaikanskripsi ini.
5. Orang tua penulis yaitu Bapak Miduk Simanungkalit dan Ibu Reliani Hutagaol yang telah mendoakan tiada henti dan memberikan semangat serta membantu penulis dalam segi materi dan moril sehingga penulis dapat menyelesaikan skripsi ini dengan sebaik baiknya.
6. Seluruh Dosen dan Staf Program Studi Teknik Informatika Universitas Medan Area.
7. Seluruh teman-teman yang sudah memberikan dukungannya selama penulisan skripsi ini, khususnya teman-teman Teknik Informatika angkatan 2019.
8. Seluruh pihak yang tidak dapat disebutkan satu persatu yang membantu dalam penyelesaikan skripsi ini.

Penulis menyadari bahwa tugas akhir/skripsi ini masih memiliki kekurangan , oleh karena itu kritik dan saran yang bersifat membangun sangat penulis harapkan untuk kesempurnaan tugas akhir/skripsi ini. Penulis juga berharap pada tugas akhir/ skripsi ini dapat bermanfaaat bagi kalangan pendidikan maupun masyarakat. Penulis ucapkan terima kasih.

Medan, 27 Maret 2024

Penulis,

Alex Simanungkalit

198160064

DAFTAR ISI

LEMBAR PENGESAHAN ..... **ii**
HALAMAN PERNYATAAN ..... **iii**
ABSTRAK ..... **iv**
DAFTAR RIWAYAT HIDUP ..... **vi**
KATA PENGANTAR ..... **viii**
DAFTAR ISI ..... **ix**
DAFTAR TABEL .....
DAFTAR GAMBAR .....

BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Rumusan Masalah 6
1.3 Batasan Masalah 6
1.4 Tujuan Penelitian 7
1.5 Manfaat Penelitian 7

BAB II TINJAUAN PUSTAKA **8**
2.1. Malaria 8
2.2. Deep Learning 10
2.3. Convolution Neural Network (CNN) 11
2.4. LeNet 14
2.5. ResNet 15
2.6. Hyperparameter Tuning 16
2.7. Darah 17
2.8. Penelitian Terkait 17

BAB III METODE PENELITIAN **22**
3.1 Spesifikasi Perangkat 22
3.2 Metode Penelitian 23
3.3 Teknik Pengumpulan Data 24
3.4 Analisis Data 25
3.5 Simulasi Arsitektur 25

3.6 Pemodelan Arsitektur *LeNet* dan *ResNet* 27
3.7 Augmentasi data 28
3.8 Hyperparameter 30
3.9 Metode Evaluasi 30
3.10 Penerapan manual Pada LeNet dan ResNet 32

BAB IV METODE PENELITIAN 22
4.1 Hasil 37
4.1.1 Collecting Dataset 37
4.1.2 Pre-processing Dataset 38
4.1.3 Modelling 43
4.1.4 Training Model 50
4.1.5 Evaluation Model 56
4.2 Pembahasan 63

BAB V KESIMPULAN DAN SARAN 65
5.1 Kesimpulan 65
5.2 Saran 65

DAFTAR PUSTAKA 67

DAFTAR GAMBAR

Gambar 2. 1 Citra Darah 10
Gambar 2. 2 Struktur Dasar CNN 11
Gambar 2. 3 Proses Convolution Layers 12
Gambar 2. 4 Proses Pooling Layer 13
Gambar 2. 5 Proses Fully Connected Layer n 13
Gambar 2. 6 Proses Pengerjaan Arsitektur *LeNet* 14
Gambar 2. 7 3Blok ResNet 15
Gambar 2. 8 Perhitungan Arsitektur Resnet 16
Gambar 3. 1 Kerangka Kerja Penelitian 23
Gambar 3. 2 Terinfeksi Malaria 24
Gambar 3. 3 Tidak Terinfeksi Malaria 24
Gambar 3. 4 Proses Zoom 29
Gambar 3. 5 Proses Resize 29
Gambar 3. 6 Residual Block Pada Resnet 33
Gambar 3. 7 Resnet Layer Struckture 33
Gambar 3. 8 Struktur Layer 34
Gambar 3. 9 Mencari Output Gambar 35
Gambar 4. 1 Tahapan Pre-processing Dataset 39
Gambar 4. 2 Proses Cropping dan Resize Dataset 40
Gambar 4. 3 Proses Split Dataset 41
Gambar 4. 4 Hasil Augmentasi Dataset 42
Gambar 4. 5 Rancangan Diagram Modelling 43
Gambar 4. 6 *Modelling* Arsitektur *LeNet5* 44
Gambar 4. 7 *Modelling* Arsitektur *ResNet* 48
Gambar 4. 8 Proses Training Model LeNet5 52
Gambar 4. 9 Plot Accuracy dan Loss Model LeNet5 54
Gambar 4. 10 Proses Training Model ResNet50 55
Gambar 4. 11 Plot Accuracy dan Loss Model ResNet50 56
Gambar 4. 12 Plot Perbandingan Accuracy dan Loss Model 61
Gambar 4. 13 Hasil Confusion Matrix Model LeNet5 62
Gambar 4. 14 Hasil Classification report Model LeNet5 63
Gambar 4. 15 Hasil Confusion Matrix Model ResNet50 65
Gambar 4. 16 Hasil Classification report Model ResNet50 65
Gambar 4. 17 Plot Performa Model LeNet5 dan ResNet50 68

DAFTAR TABEL

Tabel 2. 1 Ringkasan Penelitian Terkait 17
Tabel 3. 1 Perangkat Keras (Hardware) 22
Tabel 3. 2 Perangkat Lunak (Software) 22
Tabel 3. 3 Data Training Dan Data Testing 25
Tabel 3. 4 Hyperparameter 30
Tabel 3. 5 Kelas Positif dan Negatif 30
Tabel 4. 1 Hasil Collecting Dataset 38
Tabel 4. 2 Hyperparameter Training Model 50
Tabel 4. 3 Tabel Hasil Classification report Model LeNet5 57
Tabel 4. 4 Tabel Hasil Classification report Model ResNet 59
Tabel 4. 5 Hasil Perbandingan Performa Model 62
Tabel 4. 6 Perbandingan Akurasi Model dengan Penelitian Terdahulu 63

BAB I

PENDAHULUAN

1.1 Latar belakang

Penyakit yang akhir-akhir ini menjadi perbincangan hangat masyarakat di seluruh dunia ditemukan pertama kali pada negara Afrika dan negara lainnya mengenai malaria dan ada yang kemudian yang mengakibatkan malaria seperti Parasit yang berpotensi fatal penyebab malaria menginfeksi nyamuk Anopheles betina dan menyebar ke manusia melalui gigitan. Menurut perkiraan, terdapat 229 juta kasus malaria secara global pada tahun 2019 dan 409.000 orang meninggal akibat penyakit tersebut. Afrika adalah lokasi di mana malaria paling banyak ditemukan, menurut WHO. Dengan memanfaatkan informasi terkini dalam data pasien dan memanfaatkan teknik pembelajaran mesin, malaria dapat diidentifikasi sebelumnya. Dengan analisis riwayat data pasien menggunakan arsitektur LeNet maupun ResNet, penelitian ini berupaya mengklsifikasi dan menentukan arsitektur mana yang lebih tepat dalam memprediksi malaria. Sebelum sampai sekarang, teknik normalisasi min-max digunakan pada dataset, dan prosedur validasi silang digunakan dengan beberapa pengujian untuk menentukan pengaruh pada hasil. Afrika adalah lokasi di mana malaria paling banyak ditemukan, menurut WHO. Dengan memanfaatkan informasi terkini dalam data pasien dan memanfaatkan teknik pembelajaran mesin, malaria dapat diidentifikasi sebelumnya. Dalam beberapa penelitian, metode pembelajaran mesin termasuk Classification and Regression Trees, Naive Bayes, Artificial Neural Networks, Random Forests, dan bahkan Multi-Layer Perceptrons telah digunakan untuk

mendiagnosis malaria. (Ramadhan & Khoirunnis, 2021)

Temuan tahun 2018 menunjukkan bahwa diperkirakan ada 228 juta kasus malaria di seluruh dunia, dengan total 405 ribu kematian. Sebagian besar kasus ini 213 juta atau 93% terjadi di Afrika, diikuti oleh Asia Tenggara dengan 3,4% kasus khas dan Mediterania timur dengan 2,1 juta kasus.(WHO, 2020).

Malaria mempengaruhi 229 juta orang secara global pada tahun 2020, dengan Afrika menyumbang 94% dari semua kasus. Hingga 409.000 orang mungkin meninggal karena malaria pada tahun 2020, dengan anak balita merupakan sekitar 67% dari kematian. Di Afrika, terdapat sekitar 215 juta infeksi malaria dan 384.000 prediksi kematian akibat penyakit tersebut pada tahun 2020. (WHO, 2020).

Pada tahun 2021, 247 juta kasus malaria akan dilaporkan di seluruh dunia, membuat hampir separuh umat manusia berisiko tertular penyakit tersebut. Pada tahun 2021, diperkirakan akan ada 619.000 kematian terkait malaria di seluruh dunia. WHO Sekitar 80% dari semua kematian akibat malaria di benua Afrika disebabkan oleh anak-anak di bawah usia lima tahun, yang menyumbang hingga 95% dari semua kasus malaria dan 96% dari semua kematian akibat malaria di seluruh dunia. (WHO, 2023).

- Tes antigen :Tes ini Tergantung pada jenis tes yang digunakan, akurasi tes ini bervariasi. Sementara beberapa tes memiliki tingkat akurasi yang lebih rendah, tes lainnya memiliki tingkat akurasi sekitar 90-95%. Selain itu, kapasitas pengujian antigen cepat untuk mengidentifikasi jenis malaria yang tidak umum dibatasi..
- Tes PCR: tes ini memiliki tingkat akurasi yang sangat tinggi, yaitu

sekitar 95-99%. Tes PCR juga dapat mendeteksi infeksi pada tahap awal dan dapat membedakan jenis malaria yang berbeda

Annual Parasite Incidence (API) malaria di Indonesia meningkat dari 0,84 menjadi 0,93 per 1.000 penduduk pada tahun 2019 dibandingkan dengan tahun 2018. Pada tingkat kabupaten atau kota, terdapat sebanyak 300 kabupaten atau kota yang dieliminasi pada tahun 2019, namun di tingkat provinsi belum ada yang tereliminasi, meskipun ada tiga provinsi yang setiap kabupaten atau kotanya telah tereliminasi.(Lewinsca, Raharjo, & Nurjazuli, 2021).

Provinsi Papua (9,8% dan 28,6%), Provinsi Nusa Tenggara Timur (6,8% dan 23,3%), Provinsi Papua Barat (6,7% dan 19,4%), Provinsi Sulawesi Tengah (5,1% dan 12,5%), dan Provinsi Maluku (3,8% dan 10,7%) merupakan lima provinsi dengan angka kejadian dan prevalensi malaria tertinggi di Indonesia pada tahun 2013, menurut penelitian sebelumnya. 6 Daerah dengan angka kasus malaria tahunan tertinggi adalah Provinsi Papua yang memiliki API rate sebesar 41,31 per 1000 penduduk pada tahun 2018.7 Pada tahun 2014 dengan Annual Parasite Incident (API), angka penyakit malaria di Puskesmas Moru Provinsi Nusa Tenggara Timur sebesar 16,9%. berhubungan erat dengan prevalensi malaria spesies Anopheles, suhu, kelembaban, jarak dari tempat perkembangbiakan nyamuk, dan penelitian lain yang dilakukan di Kabupaten Seram, Provinsi Maluku, semuanya berdampak pada kejadian malaria.(Lewinsca, Raharjo, & Nurjazuli, 2021).

Anopheles sp. penyebab malaria. spesies nyamuk menyukai lingkungan tropis dan subtropis Indonesia, yang juga merupakan daerah berkembangbiakan nyamukn. Segala usia dapat terjangkit penyakit ini. Unsur perubahan iklim yang

berkaitan dengan fisik, kimia, biologi, lingkungan sosial, dan perilaku manusia berdampak pada meningkatnya prevalensi malaria. (Lewinsca, Raharjo, & Nurjazuli, 2021).

Adanya tempat berkembang biak nyamuk anopheles yang menyebar dan sulit dijangkau mengakibatkan susahnya melakukan penurunan angka kasus malaria, kondisi lingkungan rumah yang tidak memenuhi syarat kesehatan (ventilasi yang tidak memadai, plafon atap, dan dinding rumah), serta perilaku masyarakat yang beraktivitas di luar rumah pada malam hari dan sebelum subuh menjadi faktor penyebab masalah malaria yang masih berkembang di Indonesia .(Lewinsca, Raharjo, & Nurjazuli, 2021).

Sejumlah publikasi juga mengklaim bahwa variabel sosial ekonomi, kebersihan lingkungan, dan demografi dapat mempengaruhi prevalensi malaria di Indonesia. Oleh karena itu, peneliti tertarik untuk mempelajari lebih lanjut tentang variabel-variabel yang mempengaruhi prevalensi malaria di Indonesia dengan menganalisis tinjauan literatur selama lima tahun terakhir (2016–2020). (Lewinsca, Raharjo, & Nurjazuli, 2021)Pada penelitian penelitian sebelumnya tentang arsitektur resnet-50 yaitu klasifikasi sidik jari menggunakan resnet-50.

ResNet ditemukan pada tahun 2015 oleh Kaiming He, Xiangyu Zhang, Shaoqing Ren, dan Jian Sun. ResNet digunakan dalam melakukan konferensi *European Conference on Computer Vision* (ECCV). Resnet merupakan jaringan syaraf yang klasik(Jalil, 2022). Bisa melatih jaringan syaraf memiliki 150 lebih lapisan. Kelebihan dalam Melatih Jaringan yang begitu Dalam (Wu, 2019), pada metode resnet ini telah banyak dimanfaatkan peneliti dalam menyelesaikan macam macam permasalahan pada saat ini seperti (Mausavi, 2019) hasil yang

didapatkan gelombang CRED yang efesien dan handal,(Li, 2019) hasilnya sangat efesien dalam mendeteksi gambar secara adaptif pada skala yang beda.(Wu, 2019) hasilnya mendapatkan representasi yang halus dan dapat mempertahankan tingkat akurasi 71,4% .

Resnet adalah sebuah network buatan dari CNN yang melakukan berbagai tugas dalam mengatasi vanish gradient pada arsitektur Resnet dan teknik ini mendapatkan izin untuk lebih dalam dapat dilatih dengan efektif (Wang, 2022, Zeng, 2019). Resnet diharapkan bisa menghasilkan akurasi yang baik (Ridhovan, 2022). Resnet juga menambahkan *identity mapping* atau *skip connection* pada jaringan CNN (Abdalla, 2022). Resnet memiliki 2 cabang pada setiap blok atau unitnya. Cabang pertama yang melalukan operasi konvolusi dan normalisasi sedangkan yang cabang kedua cabang shorcut yang memiliki fungsi untuk menghubungkan dari blok ke blok. Resnet juga menjadi arsitektur yang populer dan sering dipakai dalam penglihatan komputer atau *vision computer* dalam berbagai pengenalan seperti gambar dan deteksi objek. Resnet juga memecahkan masalah dalam menambahkan suatu cara dalam melompati berbagai *residual learing.*

Lenet adalah model deep learning pertama yang berhasil diuji pada dataset konsep (MNIST) dan juga dataset image real-world (CIFAR-10). Lenet memiliki konsep layer-layer yang terdiri dari beberapa layer yang berinteraksi satu sama lain. Dalam penjelasan mendalam, LENET diusulkan oleh Yann LeCun pada tahun 1998. LENET awalnya diusulkan sebagai solusi yang lebih efisien untuk menangani gambar dibandingkan dengan metode lain seperti neural network konvulsiotif yang menggunakan jaringan berbagai banyak layer. LENET diuji pada dataset konsep (MNIST) dan juga dataset image real-world (CIFAR-10).

Performa LENET dalam dataset ini sangat baik dan memimpin hasil uji dalam beberapa tahun berikutnya. Dari sini, LENET telah menjadi dasar dari berbagai variasi dan peningkatan, seperti LeNet-5, GoogleNet, dan VGG. Model-model ini telah mendapatkan hasil yang sangat baik di banyak tugas dalam bidang visi komputer, seperti pengenalan objek, klasifikasi gambar, dan pelacakan objek dalam video.

Maka dari hasil literatur dan paparan diatas menjadikan referensi dalam melakukan penelitiaan. Sehingga peniliti melakukan penelitian dengan judul " **Analisis perbandingan Arsitektur *Le Net* dan *Res Net* untuk Klasifikasi Malaria".**

1.2 Rumusan Masalah

Berdasarkan penjelasan dari masalah yang telah di paparkan pada latar belakang, berikut ini merupakan rumusan masalah dalam penelitian yaitu: Bagaimana perbandingan Arsitektur *Le Net* dan *Res Net* dalam mengklasifikasi Malaria untuk mengetahui akurasi melalui citra.sempel darah.

1.3 Batasan Masalah.

Adapun batasan masalah dalam penelitian ini yaitu sebagai berikut :

1. Hasil penelitian yang dihasilkan berupa akurasi dari hasil klasifikasi malaria.
2. Data citra yang dipakai merupakan hasil citra darah dengan format .png
3. Adapun sumber data yang diperoleh pada penelitian ini diambil dari Laboratorium Amplas.
4. Arsitektur untuk mengklasifikasi malaria dengan arsitektur *ResNet* dan *LeNet*.

5. Otput menampilkan perbandingan malaria berdasarkan akurasi persentasi dalam klasifikasi kedua arsitektu *LeNet* dan *ResNet.*
6. Implementasi pada Penelitian ini menggunakan *Google Colab*
7. Citra yang diambil menngunakan bantuan Menggunakan kamera OPPO A53 2020.
8. Gambar diambil dengan bantuan *microscope*.
9. Jumlah data yang di gunakan yaitu sebanyak 6000 dataset
10. Citra malaria memiliki ukuran 224 x 224 piksel

1.4 Tujuan Penelitian.

Adapun menjadi tujuan dalam penelitian adalah sebagai berikut :

1. Mengklasifikasi malaria menggunakan citra darah.
2. Menerapkan metode *LeNet* dan *ResNet* untuk mengklasifikasi malaria.
3. Membandingkan hasil klasifikasi dari kedua arsitektur *LeNet* dan *ResNet*.

1.5 Manfaat Penelitian.

Adapun manfaat penelitian ini yaitu sebagai berikut :

1. Memperkaya keilmuan terkait *Deep Learning* dalam mengklasifikasi malaria.
2. Menerapkan metode *Lenet Dan ResNet* dalam mengklasifikasi malaria.
3. Menghasilkan *Deep Learning* dengan metode *LeNet* dan *ResNe* dalam pengenalan malaria.

BAB II

TINJAUAN PUSTAKA

2.1. Malaria

Penyakit Malaria adalah penyakit yang disebabkan Parasit (protozoa) dari genus Plasmodium oleh gigitan nyamuk yang menyebabkan malaria hidup dan bereplikasi di dalam sel darah merah manusia. Gigitan nyamuk Anopheles betina adalah metode utama penyebaran penyakit ini secara alami. Gejala awal malaria seringkali mirip flu dan termasuk demam tinggi, menggigil, dan sakit kepala. Semua kelompok usia dapat terkena penyakit ini. Gejala malaria, termasuk demam, sakit kepala, muntah, dan menggigil, mulai muncul 10 hari hingga 4 minggu setelah infeksi pertama. (Supranelfy & Oktarina, 2021).

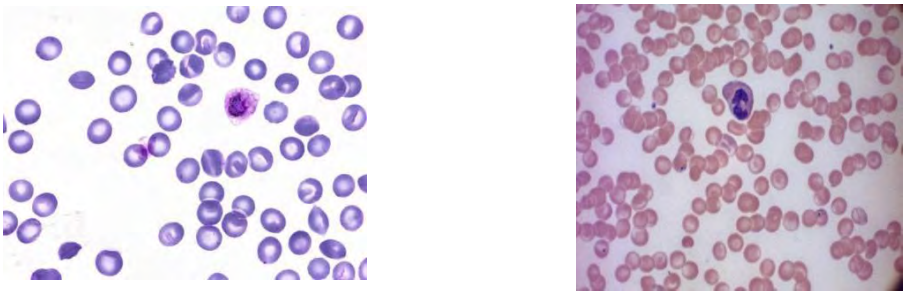
Sampai saat ini menurut (Center for Disease Control and Prevention, 2019) dikenal 4 jenis spesies plasmodium penyebab malaria pada manusia, yaitu:

1. *Plasmodium falciparum*, adalah parasit malaria yang ditemukan di daerah tropis dan subtropis di dunia. Diperkirakan setiap tahunnya ada 1 juta orang yang terbunuh akibat parasit ini, terutama di Afrika. Plasmodium falciparum adalah penyebab malaria tropika yang sering menyebabkan malaria yang berat, karena memiliki kemampuan melipat ganda secara cepat dalam darah sehingga dapat menyebabkan anemia. Selain itu *Plasmodium falciparum* dapat menyumbat pembuluh darah kecil. Ketika ini terjadi di otak akan menyebabkan malaria serebral dengan komplikasiyang dapat berakibat fatal (kematian).
2. *Plasmodium vivax*, adalah parasit malaria penyebab malaria tertiana yang

kebanyakan ditemukan di Asia, Amerika Latin, dan beberapa bagian di Afrika. Karena padatnya penduduk terutama di Asia menyebabkan *Plasmodium vivax* merupakan parasit malaria yang paling umum ditemukan pada manusia. Plasmodium vivax memiliki tahapan dormansi dalam hati (*hypnozoites*) yang dapat aktif dan menyerang darah(*relapse*) dalam beberapa bulan atau tahun setelah gigitan nyamuk yang terinfeksi.

3. *Plasmodium malariae*, adalah penyebab malaria *quartana* yang ditemukan di seluruh dunia. *Plasmodium malariae* adalah satu-satunya spesies parasit malaria pada manusia yang memiliki siklus quartan (siklus tiga hari), sedangkan tiga spesies lainnya memiliki siklus tertiana (siklus dua hari). Infeksi*Plasmodium malariae* mampu bertahan dalam waktu yang lama jika tidak diobati. Dalam beberapa kasus, infeksi kronis dapat berlangsung seumur hidup.Pada beberapa pasien kronis yang terinfeksi.
4. *Plasmodium ovale* dapat menyebabkan komplikasi yang serius seperti sindrom nefrotik. *Plasmodium ovale*, adalah parasit malaria yang menyebabkan malaria ovale tetapi jenis ini jarang dijumpai. *Plasmodium ovale* banyak ditemukandi Afrika (terutama Afrika Barat) dan pulau-pulau di Pasifik Barat. Plasmodium ovale secara biologis danmorfologis sangat mirip dengan *Plasmodium vivax*. Plasmodium ovale dapat menginfeksi individu yang negatif untuk golongan darah duffy (salah satu penggolongan darah selain ABO dan Rh) sedangkan *Plasmodium vivax* tidak. Golongan darah *duffy* banyak ditemukan pada penduduk Sub-Sahara Afrika. Hal ini menjelaskan prevalensi infeksi *Plasmodium ovale* banyak terjadi di sebagian besar Afrika.

Berikut merupakan sample citra darah malaria :



Gambar 2.1 gambar citra darah

2.2. Deep Learning

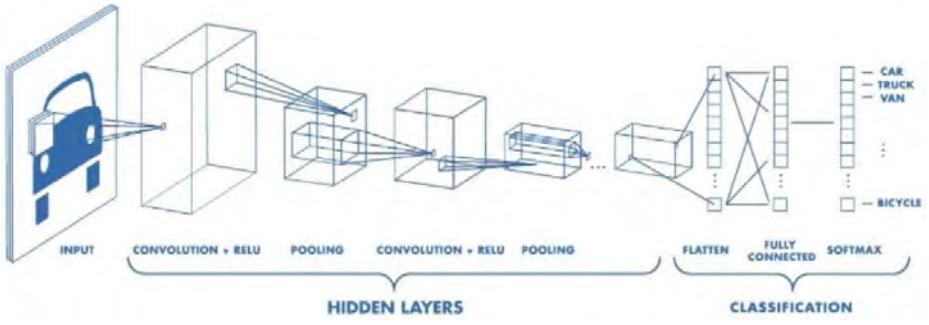
Pada tahun 2006 *deep learning* dikenal dan penggunaan sudah diterapkan diberbagai bidang seperti pengenalan suara, pengenalan citra, bahasa, dll. *Deep learning* adalah bagian dari *machine learning* ,*deep learning* ini terdiri dari banyak lapisan dalam bentuk tumpukan dan hadirnya *deep learning* ini membuat waktu menjadi efesien (Anshori, 2023) kemudian *deep learning* mempunyai kemampuan dalam memahami macam – macam data yang jumlah layernya yang banyak (Illahi, 2022). Berdasarkan kemampuannya dapat dibagi menjadi berbagai teori seperti *supervised learning* dan *unsupervised learning*. *Deep learning* ini dibentuk atau di rancang khusus agar bisa melakukan analisa seperti otak manusia dalam memutuskan dalam pengambilan keputusan(Peryanto, 2020, Rosalina, 2020) Dari fakta yang ada dilapangan *deeplearning* dapat membuktikan kemajuan yang begitu bagus dalam memahami sebuah data dan memanipulasi *images, language*, dan lainnya

Deep Learning adalah bagian dari machine learning yangdimana terinspirasi dari kortex manusia dengan menerapkan jaringan syaraf buatan yang memiliki banyak layer tersembunyi (*hidden layer*) yang dapat mempelajari komputasinya sendiri dengan menggunakan otaknya sendiri. Deep learning di rancang agar

dapat terus menganalisa data seperti pada otak manusia dalam mencapai sebuah keputusan. (Peryanto, 2019)**.** Deep learning telah terbukti semakin efisien dalam melakukan penilaian keamanan untuk perangkat IoT (Jacob & Darney, 2021)

2.3. Convolution Neural Network (CNN)

Convolution neural network merupakan sebuah *perceptron multilayer* yang didesain khusus dalam mengidentifikasi dari informasi gambar dua dimensi(Suhardin, 2021). Memiliki banyak lapisan yaitu lapisan output, lapisan konvolusi, lapisan sampel dan lapisan keluaran. Pada CNN ini memiliki keuntungan dalam menghindari eksplisit ekstraksi fitur(Dzaky, 2021). Secara implisit untuk belajar pada latihan data bobot dari permukaan pemetaan fitur sama dengan neuron, sehingga belajar jaringan secara paralel, kompeksitas jaringan berkurang, dalam mengadopsi struktur sub-sampel bisa didasari ruang dan waktu sehingga dapat pencapaian tingkat ketahanan,skala, perpihanan(Anggraeni, 2022). Informasi inputan dan topologi dapat menjadi pasangan yang sangat baik dan memiliki keunggulan unik dalam pengenalan gambar ataupun suara (Febriana, 2020).



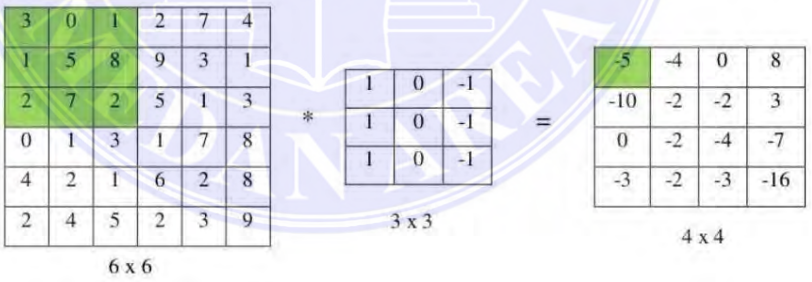
Gambar 2.2 Struktur Dasar CNN (Febriana, 2020)

Dalam pengembangan banyak arsitektur dari CNN yang banyak digunakan seperti *faster r-cnn,alexnet,vggnet,googlenet,resnet* dll. Secara umum ada 3 layer pada CNN, yaitu *convolution layers, pooling layers,dan fully connected layers.*

CNN (Convolutional Neural Network) adalah bagian dari Deep Learning untuk mengklasifikasikan suatu objek. Dalam perkembangannya, Convolutional Neural Network (CNN) adalah produk yang paling sukses karena merupakan salah satu kisah sukses pertama Deep Learning, jauh sebelum kemajuan terbaru dalam teknik pelatihan yang menghasilkan peningkatan kinerja pada jenis arsitektur lainnya. (Aggarwal, 2019).

2.3.1 Convolution Layers (CNN)

Convolution layers adalah bagian layer pada CNN merupakan inti dari sebauh sistem sebagai perhitungan terhadap pada lebar,tinggi, serta kedalaman pada gambar dan kernel. Convolution layers ini berfungsi untuk *filter* dan *feature map* dalam masukan gambar(Ristiawanto, 2021, Dzaky, 2021).

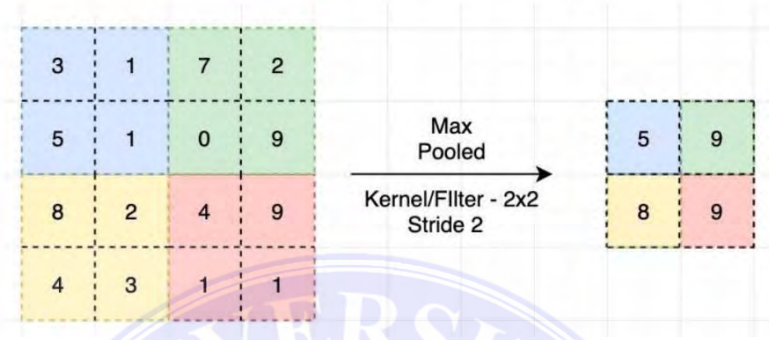


Gambar 2.3 Proses Convolution Layers(Dzaky,2021)

2.3.2 Pooling layer

Pooling layer berguna dalam mengurangi ukuran spasial yang bertujuan pengurangan parameter dan komputasi, dan juga dapat menghindari kondisi *overfitting* karena model mempunyai akurasi yang tinggi dalam memprediksi data latih tetapi gagal dalam mengenali data yangdi luar data latih. Ada banyak

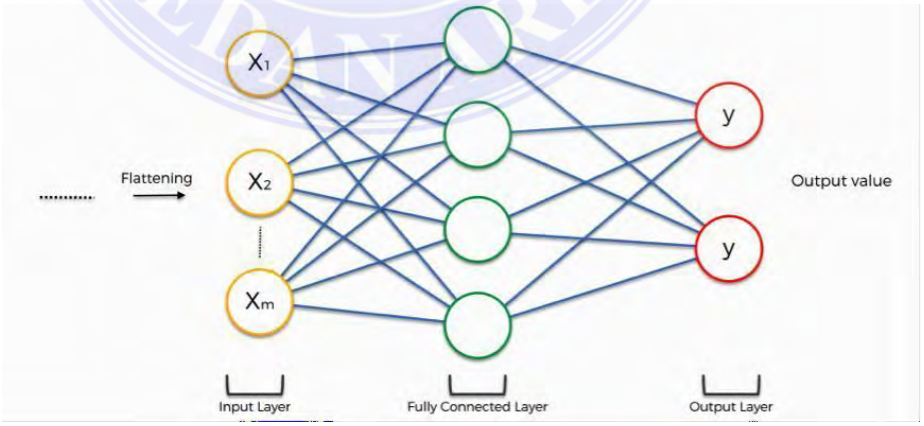
jenis pendekatan *pooling* sering digunakan *max pooling dan average pooling*, pada *max pooling* berfungsi untuk mengambil nilai maksimum pada bagian tertentu dan *average pooling*mengambil nilai rata – rata saja(Hariani, 2020).



Gambar 2.4 Proses Pooling Layer (Hariani, 2020)

2.3.3 Fully connected Layer

Fully connected layer adalah layer yang terdapat pada bagian terakhir. Setiap *neuron* terhubung dalam satu layer dengan layer yangsebelum atau sesudahnya. Layer ini sering digunakan sebagai output layer (*classfier*) dari arsitekstur CNN(illahi, 2022, Anshori, 2022).



Gambar 2.5 Proses Fully Connected Layer (illahi, 2022)

2.4. LeNet

LeNet adalah suatu jaringan yang memiliki lapisan banyak berbasis Convolutional Neural Network (CNN) pertama kali yang dikenalkan oleh Yann LeCun. *LeNet* mempunyai jumlah lapisan yang lebih banyak daripada versi *LeNet* Pada *LeNet* Total tujuh lapisan , tidak terdiri dari input, masing-masing berisi parameter yang dapat dilatih, setiap lapisan memiliki pluralitas Peta Fitur , karakteristik dari masing-masing input berikut adalah contuh gambar proses pengerjaan arsitektur *LeNet*

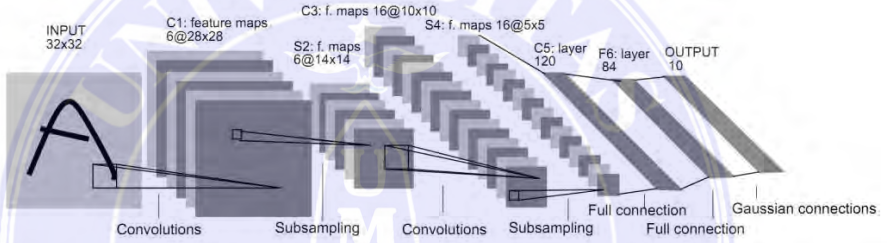


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

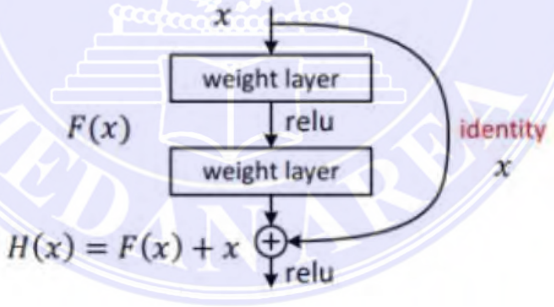
Gambar 2.6 Proses Pengerjaan Arsitektur *Lenet*

Pada dasarnya *LeNet* adalah suatu jaringan yang memiliki lapisan banyak berbasis Convolutional Neural Network (CNN) pertama kali yang dikenalkan oleh Yann LeCun. *LeNet* mempunyai jumlah lapisan yang lebih banyak daripada versi *LeNet* sebelumnya.

Pada *LeNet* Total tujuh lapisan , tidak terdiri dari input, masing-masing berisi parameter yang dapat dilatih; setiap lapisan memiliki pluralitas Peta Fitur, karakteristik dari masing-masing input berikut adalah contuh gambar proses pengerjaan arsitektur *LeNet* dan FeatureMap diekstraksi dengan menggunakan filter konvolusi, dan kemudian setiap FeatureMap Ada beberapa neuron

2.5. ResNet

Residual network adalah salah satu arsitektur dari *Convolution Neural Network (CNN)* diusulkan oleh He pada tahun 2015 tujuan arsitektur ini dibangun untuk mengatasi masalah dari *Deep Learning*, dalam pelatihan *deep learning* memakan banyak tenaga dan waktu juga lapisan yang jumlahnya terbatas. Maka dari itu, *deep learning* memberikan solusi dengan menggunakan *skip connection*. Kelebihan dari resnet ini adalah kinerja yang tidak menurun meskipun arsitekturnya semakin jauh kedalam dan semakin banyak digunakan(Zheng, 2019). Resnet ini juga memiliki perhitungan komputasi yang sangat ringan dalam melatih jaringan yang jauh lebih baik sehingga hasil dari pengujian tidak mengalami penurunan(Oyewola, 2021). Pada resnet ini memiliki bermacam arsitektur berdasarkan jumlah dari layer ,yaitu 18,34,50,101,bahkan 152 layer(Anshori, 2023).



Gambar 2.7 3Blok ResNet (Anshori, 2023)

Resnet memiliki *residual connection* sebagai mekanismenya yang merupakan bentuk dari sebuah koneksi pada jaringan syaraf tiruan bertujuan menambah jalan pintas diantara 2 titik yang berbeda(Thiodorus , 2021).

layer name	34-layer	50-layer	101-layer
conv1	$7 \times 7,64$, stride 2		
conv2_x	3×3 max pool, stride 2		
	$\begin{bmatrix} 3 \times 3,64 \\ 3 \times 3,64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1,64 \\ 3 \times 3,64 \\ 1 \times 1,256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1,64 \\ 3 \times 3,64 \\ 1 \times 1,256 \end{bmatrix} \times 3$
conv3_x	$\begin{bmatrix} 3 \times 3,128 \\ 3 \times 3,128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1,128 \\ 3 \times 3,128 \\ 1 \times 1,512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1,128 \\ 3 \times 3,128 \\ 1 \times 1,512 \end{bmatrix} \times 4$
conv4_x	$\begin{bmatrix} 3 \times 3,256 \\ 3 \times 3,256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1,256 \\ 3 \times 3,256 \\ 1 \times 1,1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1,256 \\ 3 \times 3,256 \\ 1 \times 1,1024 \end{bmatrix} \times 23$
conv5_x	$\begin{bmatrix} 3 \times 3,512 \\ 3 \times 3,512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1,512 \\ 3 \times 3,512 \\ 1 \times 1,2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1,512 \\ 3 \times 3,512 \\ 1 \times 1,2048 \end{bmatrix} \times 3$
	average pool,2048-d fc		

Gambar 2.8 Perhitungan Arsitektur Resnet (Ibrahim, 2022)

2.6. Hyperparameter Tuning

2.6.1 Grid Search

Grid search adalah metode yang berguna untuk pemodelan dalam mempelajari mesin atau (*mechine learning*) atau juga bisa untuk menentukan kombinasi dari parameter optimal dari suatu algoritma atau dari sebuah model (Nugraha, W, 2022). *Grid* yang artinya mengacu pada himpunan nilai yang untuk pada setiap parameternya akan dievaluasi (Anggoro, D. A., 2021). *Grid search* ini bekerja membangun serta mengevaluasi model pada setiap kombinasi parameter dalam *grid* tersebut setelah melakukan kombinasi patameter maka *grid search* akan memberikan hasil terbaik. Pada umumnya, tujuan utama adalah mencari kombinasi parameter yang memberikan kinerja terbaik untuk sebuah model. *Grid* membantu mengatasi tantangan optimal secara manual, dengan banyaknya jumlah parameter. Dapat secara otomatis mengevaluasi banyak kombinasi parameter dan model yang lebih baik (Turner, R., 2021).

2.6.2 Random Search

Random Search merupakan metode alternatif dalam mencari sebuah kombinasi parameter yang optimal dari pemodelan, *random search* ini sama dengan

grid search (Nurhopipah, A.,2021). Keduanya bertujuan dalam kombinasi parameter yang mempunyai kinerja yang terbaik dalam pemodelan (Navon, D.,2022). Meskipun *random search* memiliki tujuan yang sama dengan *grid search* tetapi *random search* ini memiliki perbedaan dalam menentukan kombinasi yang optimal dengan cara melakukan secara acak jumlah kombinasi yang akan dievaluasi. Namun, walaupun melakukan kombinasi secara acak *random search* ini memiliki keuntungan dalam efesien dibandingkan *grid search* yaitu pada kasus parameter yang ingin dioptimalkan cukup besar sehingga dengan cara mengacak maka hemat dalam waktu (Anggoro, D. A., 2021).

2.7. Darah

Darah merupakan cairan yang terdapat di dalam pembuluh darah yang memiliki fungsi mengatur keseimbangan asam dan basa,mentransportasikan O2, karbohidrat, dan metabolit, mengatur suhu tubuh dengan cara konduksi atau hantaran, membawa panas tubuh dari pusat produksi panas (hepar dan otot) untuk didistribusikan ke seluruh tubuh, dan pengaturan hormon dengan membawa dan mengantarkan dari kelenjar ke sasaran. Jumlah dalam tubuh bervariasi, tergantung dari berat badan seseorang. Pada orang dewasa, 1/13 berat badan atau kira-kira 4,5-5 liternya adalah darah. Faktor lain yang menentukan banyaknya darah adalah usia, pekerjaan, keadaan jantung, dan pembuluh darah (Syaifuddin, 2009)

2.8. Penelitian Terkait

Untuk melakukan penelitian harus adanya sebuah acuan dari penelitian sebelumnya agar bisa diketahui kontribusi apa saja yang dari penelitian tersebut. Berikut penelitian penelitian terdahulu yang terkait pada penelitian penulis.

Tabel 2.1 Ringkasan Penelitian Terkait

NO	Peneliti	Hasil
1	Muhammad Farhan Dwi Ryandra (2021)	Nilai akurasi arsitektur inception v-3 lebih tinggi yaitu 98% sedangkan resnet-50 bernilai 92,3%, Untuk nilai presisi juga arsitektur inception v-3 lebih tinggi dibanding resnet-50 yaitu 95,2% sedangkan resnet-50 bernilai 86,4%. Unuk sensitifitas dan nilai f-1 score arsitektur inception juga lebih tinggi yaitu secara berturut turut 97,3% dan 96,1% sedangkan resnet-50 secara berturut turut 88,7% dan 86,6%.
2	Alfiansyah NurAbadi (Suherman, 2021)	Berdasarkan hasil pengujian yang didapatkan pada penelitian ini, yaitu penggunaan arsitektur AlexNet dengan Average Pooling dan optimizer RMSprop menghasilkan akuras dan f1-score keseluruhan 99, 45% serta penggunaan arsitektur LeNet dengan Average Pooling dan optimizer RMSprop menghasilkan akurasi dan f1-score keseluruhan 99, 49%
3	Abdul Hafiez Suherman (2022)	Proses klasifikasi data citra daun akan diuji dengan kelas sebanyak 11 jenis daun klon dan jumlah dataset diaugmentasi sebesar 4400 data

		Arsitektur LeNet-5 akan digunakan pada pengujian model klasifikasi. Proses klasifikasi memperoleh hasil terbaik dengan nilai akurasi sebesar 94.55% dengan parameter optimizer Adam dan learning rate yang digunakan sebesar 0.001.
4	Hafidz Daffa Hekmatyar (2019)	Melakukan klasifikasi pneumonia. akurasi tertinggi diperoleh dengan model yang dikembangkan sebesar 97,12%. Ini diikuti oleh DenseNet201 dengan 96,83%, ResNet50 dengan 96,35%, InceptionV3 dengan 95,35%, GoogleNet dengan 94,05% dan AlexNet dengan 91,07%akurasi tertinggi diperoleh dengan model yang dikembangkan sebesar 97,12%. Ini diikuti olehDenseNet201 dengan 96,83%, ResNet-50 dengan 96,35%, Inception-V3 dengan 95,35%, GoogleNet dengan 94,05% dan Alexnet dengan 91,07%
5	Faiz Octa reynaldi (2021)	ResNet memiliki akurasi lebih tinggi dengan nilai rata-rata 100% pada Kucing Hitam dan 97.9% pada Kucing Putih jika dibandingkan dengan SSD MobileNetV1 dengan nilai rata-rata 99,66666667% pada Kucing Hitam dan nilai ratarata 78,733% pada Kucing Putih.

6	Achmad Fauzi Saksenata(2022)	Metode yang disulkan untuk menyelesaikan permasalahan tersebut Deep Learning (DL) yaitu Convolutional Neural Network (CNN)yang unggul untuk klasifikasi. Dalam penelitian ini, mengusulkan penggunaan CNN untuk membantu dalam mengklasifikasikan penyakit malaria. Dataset terdiri dari 27558 gambar sel darah. Model yang diusulkan mencapai kinerja dengan akurasi terbaik 96%.
7	Nurul Huda (2022)	Eksperimen yang dilakukan menggunakan metode yang berbeda yaitu dengan melakukan pre-prosessing citra sel darah merah dengan cara resize gambar, augmentasi gambar, dan reduksi fitur gambar menggunakan metode Pricipal Component Analysis (PCA), hasil dari fitur reduksi tersebut selanjutnya diklasifikasikan menggunakan metode CNN. Arsitektur DenseNet menunjukkan akurasi terbaik yaitu 98,30%.
8	Silvana Maretha Simbolon(2023)	Dua model CNN diimplementasikan, yakni model tanpa augmentasi data dan model dengan augmentasi data. Model tanpa

		augmentasi menunjukkan skor uji sebesar 94%, sementara model dengan augmentasi menunjukkan peningkatan kinerja yang signifikan dengan skor uji mencapai 96%. Akurasi keseluruhan model yang telah di-augmentasi pada set uji mencapai 94%, menandakan peningkatan yang berarti dalam ketepatan diagnostik.

BAB III

METODE PENELITIAN

3.1 Spesifikasi Perangkat.

Berikut ini merupakan perangkat keras dan perangkat lunak yang digunakan selama melakukan penelitian ini :

3.1.1 Perangkat Keras (*Hardware*)

Tabel 3.1 Perangkat Keras (*Hardware*)

No.	Hardware	Spesifikasi
1.	Perangkat	Laptop Acer Aspire 5
2.	Processor	AMD Ryzen 5 5500U with Radeon Graphics 2.10 GHz
3.	Monitor	35.6 cm
4.	Ram	8.0 B + 4.00 GB

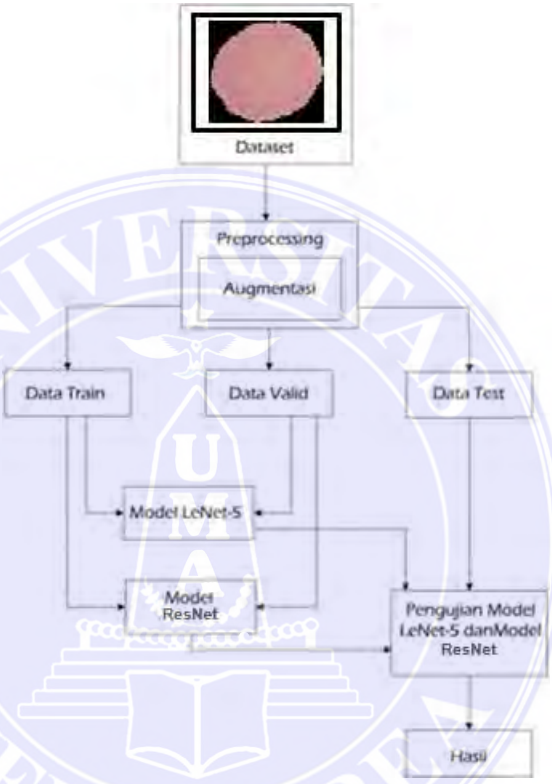
3.1.2 Perangkat Lunak (*Software*)

Tabel 3.2 Perangkat lunak (*Software*)

No.	Software	Spesifikasi
1.	OS	Windows 11 Home Single Language 64-bit
2.	Bahasa Pemrograman	Python
3.	Tools	Google Colab

3.2 Metode Penelitian

Metodologi penelitian yang dilakukan terdiri dari pengumpulan dataset, membangun model klasifikasi, training model klasifikasi, testing, dan perhitungan performa. Gambar 1 menunjukkan alur metodologi penelitian:



Gambar 3.1 Kerangka Kerja Penelitian

Pada Gambar 3.1 Pada Gambar 3.1 Data yang digunakan dalam penelitian ini diambil dari Laboratorium Sisingamangaraja dan tambahan data dari sumber terbuka yang terdiri dari data malaria melalui halaman https://www.kaggle.com/datasets/miracle9to9/ files1?select=Malaria+Cells dari sumber ini, kemudian menggabungkan dataset gabungan dengan tiga kelas yaitu terdampak malaria dan tidak terdampak malaria

3.3 Teknik Pengumpulan Data

Penelitian ini data masukan diperoleh dari citra yang diambil dari laboratorium untuk pengecekan darah di daerah amplas, Medan. Pada pengambilan data ini dilakukan secara langsung dengan cara mengambil gambar gambar dengan menggunakan microskop kemudian di foto menggunakan camera OS Android OPPO A53 2020. Gambar yang diambil dengan jaraksekitar 1 cm dari microskop dengan fokus dekat dengan layer komputer, kemudian hasil dari gambar disimpan dalam bentuk PNG. Kemudian dalam pengujian ini digunakan kelas diantara nya kelas terdampak dan tidak terdampak berikut adalah dataset yang akan di proses seperti pada gambar di bawah

Gambar 3.2 Terinfeksi Malaria

Adapun disertai dengan citra darah sebagai dataset dalam menentukan citra darah tersebut adalah termasuk terdampak malaria atau tidak maka dataset citra darah tidak terdampak diperlukan. Berikut ini citra tidak terdampak malaria dibawah ini

Gambar 3.3 Tidak Terinfeksi Malaria.

3.4 Analisis Data

Sampel gambar yang digunakan dalam model uji coba sebanyak 3000 sampel model evaluasi yang digunakan sebagai penelitian ini membagi data menjadi dua bagian yaitu data training dan data testing. Kemudian lakukan crop pada sampel agar foto fokus ke darah sebagai objek utama, setelah itu lakukan resize pada sampel agar dalam proses training tidak begitu berat atau lama. Data yang digunakan sebagai data training sebanyak 70% dan sampel data testingnya sebanyak 30%. Berikut lampiran sampel Penyakit malaria :

Tabel 3.3 Data Training Dan Data Testing

No.	Kelas	Dataset		
		Training(70%)	Testing(20%)	Validasi(10%)
1.	Tidak Terinfeksi	2100	600	300
2.	Terinfeksi	2100	600	300
Total Data	**6000**	**4200**	**1200**	**600**

3.5 Simulasi Arsitektur

Penelitian ini akan dibuat image classification model untuk mengenali tingkat persentase malaria dengan menggunakan pre-trained model *LeNet* dan *ResNet* . Arsitektur *LeNet* dan *ResNet* dipilih karena arsitektur ini memiliki akurasi yang sama tingginya dan akan mencari perbandingan dari kedua arsitektur. Dimana arsitektur ini memberikan proses yang tidak begitu rumit.

Simulasi Pada lenet

1. Pada tahap input layer, citra yang digunakan berukuran 224x224 angka 3 yang menandakan citra tesebut adalah chanel image RGB pada citra warna (true color).

Sebelum citra diinputkan, maka terlebih dahulu akan diaugmentasi seperti yang sudah dijelaskan dibagian augmentasi dataset diatas.

2. Pada *LeNet* model yang di gunakan secara manual dengan aturan dasar pemodelan *LeNet.* Pada *LeNet* ada 2 convolusi yaitu

   convolusi yang pertama dengan filter =6, dan karnel size = 5, strides =1 dan activation yang di gunakan yaitu Tanh dengan input shape 224,224,3

   sama saja seperti convolusi yang pertama perbedaannya hanya di filter dengan filter =16, dan karnel size = 5, strides =1 dan activation yang di gunakan yaitu Tanh dengan input shape 224,224,3

3. Kemudian Flattened Patches + Proyeksi Linear

4. Kemudian di dense MODEL_LeNet5.add (Dense (units=120, activation='tanh')) MODEL_LeNet5.add (Dense (units=84, activation='tanh')) MODEL_LeNet5.add (Dense(2, activation='softmax'))

Simulasi Pada ResNet

1. Pada tahap input layer, citra yang digunakan berukuran 224x224 angka 3 yang menandakan citra tesebut adalah chanel image RGB pada citra warna (true color). Sebelum citra diinputkan, maka terlebih dahulu akan diaugmentasi seperti yang sudah dijelaskan dibagian augmentasi dataset diatas.
2. Selanjutnya citra tersebut diinputkan pada tahap feature extractor untuk mengekstraksi fitur dari dataset menggunakan pre-trained model dengan arsitektur *Resnet* pada gambar 3.8 terdapat beberapa parameter dari arsitektur *Resnet* yang disesuaikan didalam penelitian tersebut yaitu sebagai berikut :
   a) Include_top=false Berfungsi untuk menghilangkan layer terakhir pada model *Resnet*. Layer output default dari *Resnet* adalah 12 atau lebih. Pada penelitian ini jumlah kelas dari output adalah 2 kelas yaitu terdampak malaria dan tidak terdampak malaria. Oleh karena itu penelitian ini menggunakan parameter

include_top=false untuk menghilangkan layer terakhir pada model *Resnet*

b) Weights = imagenet

Berfungsi untuk dapat dapat menggunakan weights model dari *Resnet* yang sudah dilatih pada dataset imagenet.

c) Trainable = False berfungsi untuk semua layer tidak dapat ditraining ulang. Oleh karena, hal ini penelitian ini menggunakan metode CNN dari model arsitektur *Resnet* sebagai feature extractor untuk mengekstraksi fitur dari citra dataset, dan tidak mengubah bobotbobot yang sudah dilatih pada model *Resnet*.

3. Pada Tensorflow keras atau Pustaka keras, arsitektur *Resnet* terdiri dari 4 tahap dibagi dari beberpa blok layer yaitu layer of multiheaded self-attention, MLP Block dan Layernorm (LN). Pada Layer pertama dilakukan operasi pengkodean posisi untuk melakukan ekstraksi fitur pada dataset citra input. Tahap ini adalah sebuah proses yang dikombinasi dengan encoder dan decoding dengan tahap awal melalukan sebuah patch embbeding pada gamabar dilanjutkan pada flattened patches lalu gambar akan melewati sebuah lapisan dan setiap blok yang sudah disediakn oleh model arsitektur *Resnet*. Dimana konvolusi yang digunakan pada input citra layer berukuran 224x224 dan chanel 3 serta stride yang digunakan adalah 16x16. Pada penjelasan diatas adalah sebuah ilustrasi,

3.6 Pemodelan Arsitektur *LeNet* dan *ResNet*

Pada penelitian ini akan menggunakan metode arsitektur *ResNet* dan *LeNet* dalam melakukan klasifikasi Citra darah. Dataset final yang digunakan setelah dilakukan praproses adalah berjumlah 6000 citra dengan rincian 3000 buah citra data terinfeksi dan 3000 citra data yang tidak terinfeksi. Dataset terbagi menjadi tiga bagian, yaitu data training 70%, data validation 10% dan data testing 20%. Pembagian dataset ini akan mengurangi jumlah citra yang akan di training, sehingga perlu dilakukan augmentasi untuk memperbanyak jumlah dataset. Proses

augmentasi ini dilakukan sebelum melatih model menggunakan arsitektur *LeNet* maupun *Resnet*

3.7 Augmentasi data

Augmentasi dataset dilakukan untuk meningkatkan ukuran dataset agar mendapatkan banyak gambar yang berbeda. Proses ini dilakukan untuk mencegah situasi overfitting (memiliki kinerja baik selama pelatihan, tetapi buruk pada data baru) dalam proses pelatihan model dengan arsitektur *LeNet* dan *ResNet*. Augmentasi yang dilakukan yaitu pencerminan (flipping) secara horizontal adalah untuk membalik gambar secara horizontal, rotasi (rotating). Augmentasi dilakukan pada citra yaitu augmentasi rotasi dimana rotasi dilakukan 90 derajat Hitung nilai kordinat baru pada setiap piksel setelah rotasi, untuk rotasi 90^0, dimana nilai titik kordinat baru adalah (〖x^(′), y〗 ^′) dengan rumus :

$$x^{(\prime)} = \cos(\theta) * x - \sin(\theta) * y$$

$$y^{(\prime)} = \sin(\theta) * x - \cos(\theta) * y$$

Ket:

x, y : variable nilai piksel

θ : sudut dari rotasi Untuk piksel yang digunakan adalah

$$(0,0) : x^{\prime} = \cos(90) * 0 - \sin(90) * 0 = 0 \; ; \; y^{\prime}$$

$$= \sin(90) * 0 + \cos(90) * 0 = 0$$

$$(0,0) : x^{\prime} = \cos(90) * 1 - \sin(90) * 1 = -1 \; ; \; y^{\prime}$$

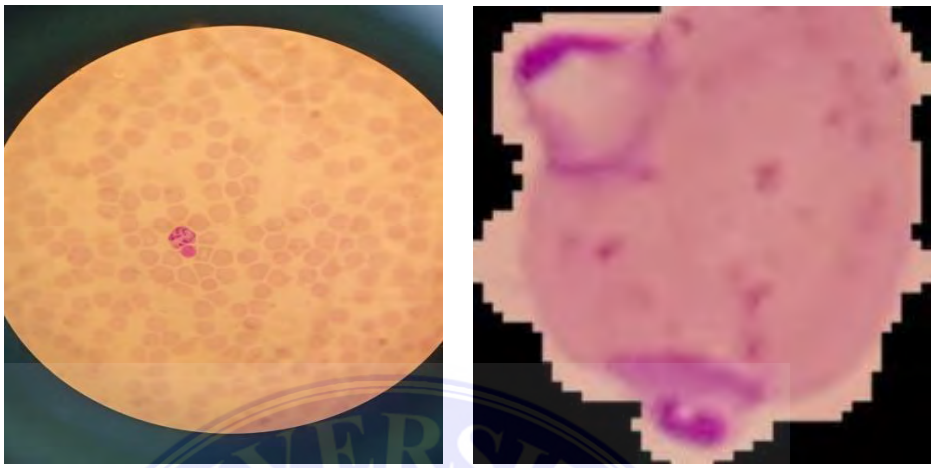
$$= \sin(90) * 0 + \cos(90) * 0 = 0$$

Adapun augmentasi data yang di lakukan adalah :

1. Memperbesar Citra (Zoom)

Proses zoom dilakukan untuk memperbesar citra dengan perbesaran 0.1 adalah untuk memperbesar citra dengan perbesaran sebanyak 1+0.1 dari luas

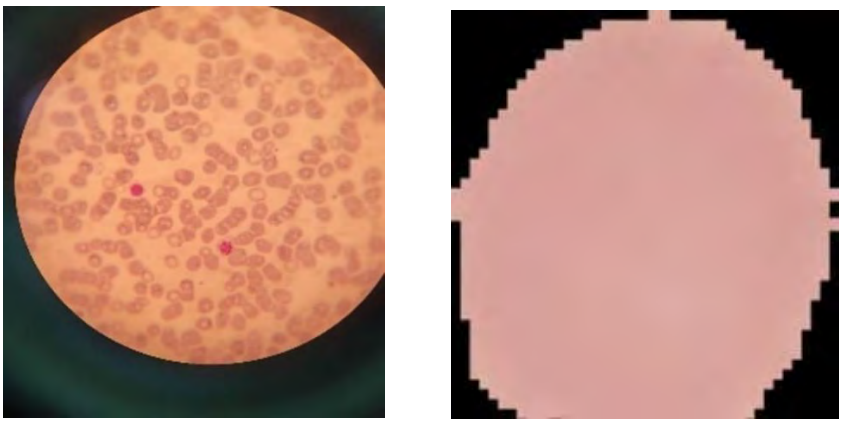
gambar yang dilakukan secara acak. Adapun ilustrasi pada proses zoom dapat disajikan pada Gambar 3.4



Gambar 3.4 proses zoom

2. Mengubah Resolusi (Resize)

Proses resize dimaksudkan untuk mengubah resolusi citra sesuai dengan yang di inginkan. Pada penelitian ini, citra dataset akan mengalami perubahan ukuran (resizing) menjadi 224x224 piksel. Proses ini bertujuan untuk mempercepat proses klasifikasi dan menyamaratakan ukuran citra setiap data dan juga untuk menyesuaikan dengan input shape didalam arsitektur *LeNet* maupun *ResNet* yang akan digunakan. Adapun ilustrasi pada proses resize dapat disajikan pada Gambar 3.5.



Gambar 3.5 proses resize

3.8 Hyperparameter

Tabel 3.4 Hyperparameter

No.	Parameter	Nilai
1.	Optimizer	Adam
2.	Learning Rate	0.0001
3.	Batch Size	64
4.	Epoch	20

3.9 Metode Evaluasi

3.9.1 Parameter Performansi

Parameter performansi memiliki kegunaan untuk menjadi tolak ukur dari penilaian kinerja dan kualitas sistem yang dibuat dan juga memiliki fungsi untuk mengetahui sistem sudah mampu mempelajari data setiap kelasnya. Untuk mengetahui kualitas dari sistem bisa dilihat dari nilai *confusion matrix* ini bertujuan untuk mengalisis tingkat akurasi, *recall*, presisi, *F1-score*, dan *F2-score.*

1. Confusion Matrix

Confusion Matrix adalah parameter yang digunakan untuk mengevaluasi kualitas dan kerja dari model yang ada. Nilai dari *confusion* ini menghitung akurasi,presisi,*recall* dan *F1-score, F1-score.*

Tabel 3.5 Kelas Positive dan Negative

Confusion Matrix		Kelas aktual	
		positif	Negatif
Kelas Prediksi	Positif	TP	FP
	Negatif	FN	TN

Keterangan :

True Positive (TP) : Data hasil aktual positif, mengidentifikasi dengan benar positif

True Negative(TN) : data hasil aktual negatif, mengidentifikasi dengan benar negatif

False Positive (FP) : data hasil aktual negatif,salah mengindentifikasi dengan positif

False Negative (FN) : data hasil aktual positif, salah mengidentifikasi dengan negatif

2. Akurasi.

Akurasi adalah parameter untuk perbandingan kelas yang diprediksi benar dengan total seluruh data yang digunakan. Akurasi ini berfungsi untuk mengetahui efektifitas dari kelas dalam klasifikasi. Menghitung nilai akurasi rumusnya sebagai berikut :

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{..... 3.1}$$

3. Presisi

Presisi adalah parameter dalam menentukan sebuah nilai ketepatan informasi yang di minta oleh user dan hasil yang diberikanoleh sistem yang telah dibuat dan presisi ini memberikan hasil berdasarkan persen. Bagaimana klasifikasi kelas yang benar terhadapdata prediksi benar. Rumus menghitung persen sebagai berikut :

$$Presisi = \frac{TP}{TP+FP} \quad \text{..... 3.2}$$

4. Recall

Recall adalah parameter untuk mengidentifikasi nilai positif dan di lakukan perbandingan total data benar terprediksi positif. Kalausemakin tinggi nilai positif makanya tinggi pula nilai Recall. Rumus mencari Recall sebagai berikut :

$$Recall = \frac{\text{TP}}{\text{FN+TP}} \quad \text{3.3}$$

5. F1-Score

F1-Score adalah nilai rata -rata presisi dan Recall. Rumusmenghitungnya sebagai berikut :

$$F1 = 2 . \frac{\text{Presisi . Recall}}{\text{Presisi+Recall}} = \frac{\text{TP}}{\text{TP+1/2(FP+FN)}} \quad \text{3.4}$$

6. F2-Score

F2-score merupakan ukuran dari evaluasi klasifikasi biner yang berguna untuk kinerja model dengan timbangan presisi atau recall dengan cara memperkuat bobotnya. F2-score ini biasanya digunakandalam kasus kasus FN dan FP, F2-score ini bisa dihitung dengan confusion matrix. Rumusnya sebagai berikut : [2]

$$F2 - Score = (1 + \beta 2)\, x \frac{\text{Presisi x Recall}}{(\beta 2 \text{ x Presisi}) + Recall}$$

3.10 Penerapan manual Pada LeNet dan ResNet

3.10.1 ResNet

- ResNet pada ResNet ada yang di namakan residual block

Gambar 3.6 Residual Block Pada Resnet

Dengan rumus yaitu

$$Y = F(X) + X$$

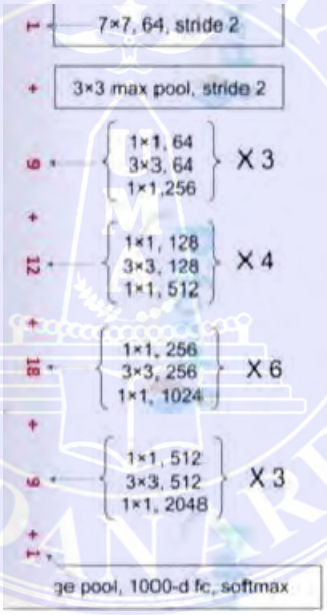
Membuat F(X) = 0 sehingga kitab bisa membuat Y=X dengan cara

Y = F(X) + X

Y = 0 + X

Y = X

- Resnet Layer Struckture

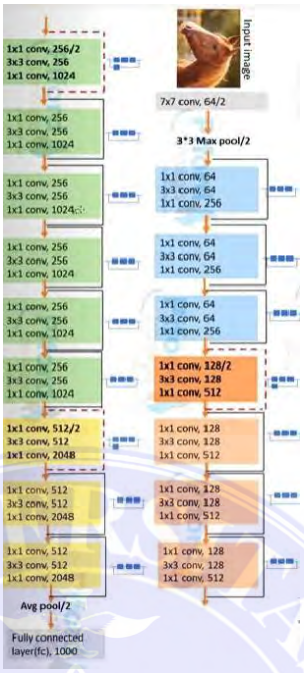


Gambar 3.7 Resnet Layer Struckture

Pada struktur layer di atas dapat kita gitung jumlah dari stuktur tersebut yaitu :

1 + 9 + 12 + 18 + 9 + 1 = 50

Dapat kita lihat pada truktur layer di bawah ini :



Gambar 3.8 Struktur Layer

Untuk menghitung gambar pertama pada layer ResNet :

Dik:

Filter size = 7*7 dan 64 demikian fiture map

Sride =2

Padding = 3

$$\frac{n+2p-f}{s}+1 * \frac{n+2p-f}{s}+1$$

$$\frac{224+2*3-7}{2}+1 * \frac{224+2*3-7}{2}+1$$

$$\frac{224+6-7}{2}+1 * \frac{224+6-7}{2}+1$$

$$112*112*64$$

Untuk menghitung gambar maxpooling layer ke 2 pada layer ResNet :

Imput image =112*112*64

$$\frac{n+2p-f}{s}+1 * \frac{n+2p-f}{s}+1$$

$$\frac{112+2*3-7}{2}+1 * \frac{112+2*3-7}{2}+1$$

$$\frac{112+6-7}{2}+1 * \frac{112+6-7}{2}+1$$

56*56*64

Kemudian kita akan mencari output yaitu dengangan cara :

Dengan input image = 56*56*64

$$\frac{n+2p-f}{s}+1 * \frac{n+2p-f}{s}+1$$

$$\frac{56+2*3-7}{2}+1 * \frac{56+2*3-7}{2}+1$$

$$\frac{56+6-7}{2}+1 * \frac{56+6-7}{2}+1$$

28*28*64

Gambar 3.9 mencari output gambar

Maka dapat lah input size dari gambar dan output = (56*56*64 = 28*28*128)

Tampilan zero padding image

3.10.2 LeNet

Untuk menghitung gambar pertama pada layer LeNet :

Input image 32*32*1,

Filter 6 dari ukuran (5x5)

Stride = 1

$$\frac{n+2p-f}{s}+1 * \frac{n+2p-f}{s}+1$$

$$\frac{32-5}{2}+1 * \frac{32-5}{2}+1$$

28*28*6

Untuk menghitung gambar layer ke 2 pada layer LeNet :

Input image 28*28*6,

Filter 2 dari ukuran (2x2)

Stride = 2

P=0

$$\frac{n+2p-f}{s}+1 * \frac{n+2p-f}{s}+1$$

$$\frac{28-2}{2}+1 * \frac{28-2}{2}+1$$

14*14*6

Untuk menghitung gambar terahir pada layer LeNet :

Input image 32*32*1,

Filter 6 dari ukuran (5x5)

Stride = 1

$$\frac{n+2p-f}{s}+1 * \frac{n+2p-f}{s}+1$$

$$\frac{32-5}{2}+1 * \frac{32-5}{2}+1$$

28*28*6

Maka dapat lah input size dari gambar dan output = (32*32*1 = 5*5*16)

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang dapat diambil dari hasil analisis dan pengujian pada klasifikasi penyakit malaria dengan membandingkan model arsitektur *LeNet5* dan *ResNet50*, yaitu sebagai berikut:

1. Penerapan model arsitektur *LeNet5* dan *ResNet50* menghasilkan performa yang baik dalam melakukan klasifikasi penyakit malaria. Hasil pengujian dengan data *testing* menghasilkan *score accuracy* 96.1667% untuk *LeNet5* dan 100% untuk *ResNet50*. Model arsitektur *LeNet5* dan *ResNet50* di *training* menggunakan *hyperparameter* berupa *optimizer Adam*, *learning rate* 0.0001, *batch size 64*, dan jumlah *epoch* 20.
2. Hasil evaluasi performa model menggunakan *confusion matrix* menghasilkan *score accuracy* 96.1667%, *precision score* 96.3211%, *recall score* 96%, dan *f1-score* 96.1603% untuk *LeNet5*. Sedangkan pada model *ResNet50* diperoleh *score accuracy* 100%, *precision score* 100%, *recall score* 100%, dan *f1-score* 100%. Model terbaik (*best model*) dalam melakukan klasifikasi penyakit malaria adalah *ResNet50*.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, beberapa saran yang dapat dipertimbangkan pada penelitian selanjutnya adalah sebagai berikut:

1. Penelitian ini menerapkan perbandingan model arsitektur *LeNet5* dan *ResNet50* dalam mengklasifikasi penyakit malaria. Pada penelitian selanjutnya dapat menerapkan model arsitektur lain sehingga bisa

digunakan sebagai acuan panilaian performa model dari sudut pandang yang lain.

2. Penelitian selanjutnya dapat mengembangkan (*deploy*) model kedalam aplikasi siap pakai sehingga penggunaannya dalam melakukan klasifikasi pneyakit malaria dapt lebih mudah.

DAFTAR PUSTAKA

Abdalla, H. B., Ahmed, A. M., Zeebaree, S. R., Alkhayyat, A., & Ihnaini, B. (2022). Rider weed deep residual network-based incremental model for text classification using multidimensional features and MapReduce. *PeerJ Computer Science*, *8*, e937.

Anggoro, D. A., & Mukti, S. S. (2021). Performance Comparison of Grid Search and Random Search Methods for Hyperparameter Tuning in Extreme Gradient Boosting Algorithm to Predict Chronic Kidney Failure. *International Journal of Intelligent Engineering & Systems*, *14*(6).

Anggraeni, D. S., Widayana, A., Rahayu, P. D., & Rozikin, C. (2022). Metode Algoritma Convolutional Neural Network pada Klasifikasi Penyakit Tanaman Cabai. *STRING (Satuan Tulisan Riset dan Inovasi Teknologi)*, *7*(1), 73-78.

Anshori, R. B., Fauzi, H., & Siadari, T. S. (2023). Klasifikasi Citra Kanker Serviks Menggunakan Deep Residual Network. *eProceedings of Engineering*, *9*(6).

Dzaky, A. T. R., & Al Maki, W. F. (2021). Deteksi Penyakit Tanaman Cabai Menggunakan Metode Convolutional Neural Network. *eProceedings of Engineering*, *8*(2).

Erwandi, R., & Suyanto, S. (2020). Klasifikasi Kanker Payudara Menggunakan Residual Neural Network. *Indonesia Journal on Computing (Indo-JC)*, *5*(1), 45-52.

FEBRIANA, B. (2020). *TA: Identifikasi Penyakit Daun Apel Menggunakan Resnet 50 Dilated Convolution Neural Network* (Doctoral dissertation, Institut Teknologi Nasional Bandung).

Hariyani, Y. S., Hadiyoso, S., & Siadari, T. S. (2020). Deteksi Penyakit Covid-19 Berdasarkan Citra X-Ray Menggunakan Deep Residual Network. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, *8*(2), 443.

Hodiyah, I., Hartini, E., & Amilin, A. (2019). Efikasi pestisida nabati dalam pengendalian penyakit antraknosa pada tanaman cabai (Capsicum annum L.). *Jurnal Agroekoteknologi*, *11*(2), 189-199.

Ibrahim, N. U. R., LESTARY, G. A., HANAFI, F. S., SALEH, K., PRATIWI, N.

K. C., HAQ, M. S., & MASTUR, A. I. (2022). Klasifikasi Tingkat Kematangan Pucuk Daun Teh menggunakan Metode Convolutional Neural Network. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, *10*(1), 162.

Illahi, P. P., Fauzi, H., & Siadari, T. S. (2022). Klasifikasi Penyakit Pneumonia Dan Covid-19 Berbasis Citra X-ray Menggunakan Arsitektur Deep Residual Network. *eProceedings of Engineering*, *9*(4).

Indarwati, S. A., & Susilawati, I. (2022). Sistem Pakar Diagnosa Penyakit Pada Tanaman Cabai Merah Menggunakan Metode Certainty Factor Dan Weighted Berbasis Web. *Journal Of Information System And Artificial Intelligence*, *2*(2), 56-63.

Jalil, A. J., & Reda, N. M. (2022). Infrared Thermal Image Gender Classifier Based on the Deep ResNet Model. *Advances in Human-Computer Interaction*, *2022*.

Li, J., Fang, F., Mei, K., & Zhang, G. (2019). Multi-scale residual network for image super-resolution. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 517-532).

Mousavi, S. M., Zhu, W., Sheng, Y., & Beroza, G. C. (2019). CRED: A deep residual network of convolutional and recurrent units for earthquake signal detection. *Scientific reports*, *9*(1), 1-14.

Mutmaini, L. F. (2019). *Pengaruh Interval Pemberian Dengan Bermacam Pestisida Nabati Terhadap Hama Tanaman Cabai Rawit (Capsicum Frustescens L.)* (Doctoral dissertation, Universitas Islam Riau).

Navon, D., & Bronstein, A. M. (2022). Random Search Hyper-Parameter Tuning: Expected Improvement Estimation and the Corresponding Lower Bound. *arXiv preprint arXiv:2208.08170*.

Nugraha, W., & Sasongko, A. (2022). Hyperparameter Tuning on Classification Algorithm with Grid Search. *Sistemasi: Jurnal Sistem Informasi*, *11*(2), 391-401.

Nurhopipah, A., & Larasati, N. A. (2021). CNN hyperparameter optimization using random grid coarse-to-fine search for face classification. *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, andControl*, 19-26.

Oyewola, D. O., Dada, E. G., Misra, S., & Damaševičius, R. (2021). Detecting cassava mosaic disease using a deep residual convolutional neural network with distinct block processing. *PeerJ Computer Science*, *7*, e352.

Peryanto, A., Yudhana, A., & Umar, R. (2020). Rancang Bangun Klasifikasi Citra Dengan Teknologi Deep Learning Berbasis Metode Convolutional Neural Network. *Format J. Ilm. Tek. Inform*, *8*(2), 138.

Permadi, J., & Harjoko, A. (2019). Identifikasi Penyakit Cabai Berdasarkan Gejala Bercak Daun dan Penampakan Conidia Menggunakan Probabilistic Neural

Network. *SEMNASKIT 2015*.

Puspitasari, A. M., Ratnawati, D. E., & Widodo, A. W. (2019). Klasifikasi penyakit gigi dan mulut menggunakan metode Support Vector Machine. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN*, *2548*, 964X.

Pratiwi, A. E. N. (2019). Sistem Diagnosa Penyakit Pada Tanaman Cabai Merah Dengan Metode Backward Chaining (Studi Kasus: Petani Cabai Merah Desa Grobongan Kabupaten Madiun).

Ridhovan, A., & Suharso, A. (2022). Penerapan metode residual network (RESNET) dalam klasifikasi penyakit pada daun gandum. *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, *7*(1), 58-65.

Ristiawanto, S. P., Irawan, B., & Setianingsih, C. (2021). Pengenalan ekspresi wajah berbasis convolutional neural network menggunakan arsitektur residual network-50. *eProceedings of Engineering*, *8*(5).

Rosalina, R., & Wijaya, A. (2020). Pendeteksian Penyakit pada Daun Cabai dengan Menggunakan Metode Deep Learning. *Jurnal Teknik Informatika dan Sistem Informasi*, *6*(3).

Saputra, F. B., Kallista, M., & Setianingsih, C. (2023). Deteksi Social Distancing Dan Penggunaan Masker Di Restoran Menggunakan Algoritma Residual Network (RESNET). *eProceedings of Engineering*, *10*(1).

Suhardin, I., Patombongi, A., & Islah, A. M. (2021). Mengidentifikasi Jenis Tanaman Berdasarkan Citra Daun Menggunakan Algoritma Convolutional Neural Network. *Simtek: jurnal sistem informasi dan teknik komputer*, *6*(2), 100-108.

Syarif, A. K. (2021). *Sistem Klasifikasi Penyakit Tanaman Cabai Menggunakan Metode Deep Learning dengan Library TensorFlow Lite* (Doctoral dissertation, Universitas Hasanuddin).

Tanjung, M. Y., Kristalisasi, E. N., & Yuniasih, B. (2019). Keanekaragaman Hama dan Penyakit Pada Tanaman Cabai Merah (Capsicum annum L) PadaDaerah Pesisir dan Dataran Rendah. *Jurnal Agromast*, *3*(1).

Thiodorus, G., Prasetia, A., Ardhani, L. A., & Yudistira, N. (2021). Klasifikasi citra makanan/non makanan menggunakan metode Transfer Learning dengan model Residual Network. *Teknologi: Jurnal Ilmiah Sistem Informasi*, *11*(2), 74-83.

Tuhumury, G. N. C., & Amanupunyo, H. R. (2019). Kerusakan tanaman cabai akibat penyakit virus di Desa Waimital Kecamatan Kairatu. *Agrologia*, *2*(1).

Turner, R., Eriksson, D., McCourt, M., Kiili, J., Laaksonen, E., Xu, Z., & Guyon,

I. (2021, August). Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. In *NeurIPS 2020 Competition and Demonstration Track* (pp. 3-26). PMLR.

Wang, F., & Ying, Y. (2022). Evaluation of students' innovation and entrepreneurship ability based on resnet network. *Mobile Information Systems*, *2022*.

Wu, Z., Nagarajan, T., Kumar, A., Rennie, S., Davis, L. S., Grauman, K., & Feris,

R. (2019). Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 8817-8826).

Zheng, S., Rahmat, R. W. O., Khalid, F., & Nasharuddin, N. A. (2019). 3D texture-based face recognition system using fine-tuned deep residual networks. *PeerJ Computer Science*, *5*, e236.

Zikra, F., Usman, K., & Patmasari, R. (2021, September). Deteksi Penyakit Cabai Berdasarkan Citra Daun Menggunakan Metode Gray Level Co-Occurence Matrix Dan Support Vector Machine. In *Prosiding Seminar Nasional Darmajaya* (Vol. 1, pp. 105-113).

Lampiran Codingan Program

```
# Import library
import os
import  glob
import random
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import tensorflow  as  tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras import Sequential
from tensorflow.keras.models import Sequential
from tensorflow.keras.utils import image_dataset_from_directory
from tensorflow.keras.models import load_model
from tensorflow.keras.layers import MaxPooling2D,
AveragePooling2D, GlobalAveragePooling2D
from tensorflow.keras.layers import Conv2D, Flatten, Dense,
Dropout
from tensorflow.keras.optimizers import Adam, Adamax, AdamW,
Adadelta, Adagrad, Adafactor, SGD, RMSprop
from tensorflow.keras.losses import BinaryCrossentropy,
CategoricalCrossentropy
from tensorflow.keras.callbacks import EarlyStopping,
ReduceLROnPlateau, ModelCheckpoint

from sklearn.metrics import confusion_matrix,
classification_report
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score
from sklearn.metrics import precision_recall_fscore_support

from IPython.display import clear_output
import warnings
warnings.filterwarnings('ignore')
print('Library loaded...\n')
# print('Tensorflow Version\t:', tf.__version__)
# print('Keras Version\t\t:', keras.__version__)
!pip show tensorflow
# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')
# Define Path of Dataset (Training, Validation, Testing)
TRAIN_PATH = '/content/drive/MyDrive/My Dataset/Malaria
Dataset/Training/'
```

```
VAL_PATH = '/content/drive/MyDrive/My Dataset/Malaria
Dataset/Validation/'
TEST_PATH = '/content/drive/MyDrive/My Dataset/Malaria
Dataset/Testing/'

# Collect number of class & classes name
CLASS_NAMES = sorted(os.listdir(TRAIN_PATH))
N_CLASSES = len(CLASS_NAMES)
print(f"Number of classes\t: {N_CLASSES}")
print(f"Classes name\t\t: {CLASS_NAMES}")
# Get image directory
TRAIN_SIZE = len(glob.glob(f"{TRAIN_PATH}**/*.png"))
VAL_SIZE = len(glob.glob(f"{VAL_PATH}**/*.png"))
TEST_SIZE = len(glob.glob(f"{TEST_PATH}**/*.png"))

# Get dataset (training, validation & testing) size
N_DATASET = TRAIN_SIZE + VAL_SIZE + TEST_SIZE

# View samples counts
print(f'Sample data training\t:{TRAIN_SIZE}')
print(f'Sample data validation\t:{VAL_SIZE}')
print(f'Sample data testing\t:{TEST_SIZE}')
print('=====================================')
print(f'Total dataset\t\t:{N_DATASET}')
# Plot Dataset
COUNT_TRAIN={}
for x in CLASS_NAMES:
    NUM_TRAIN=len(os.listdir(os.path.join(TRAIN_PATH, x)))
    COUNT_TRAIN[x]=NUM_TRAIN

COUNT_VAL={}
for y in CLASS_NAMES:
    NUM_TRAIN=len(os.listdir(os.path.join(VAL_PATH, y)))
    COUNT_VAL[y]=NUM_TRAIN

COUNT_TEST={}
for z in CLASS_NAMES:
    NUM_TEST=len(os.listdir(os.path.join(TEST_PATH, z)))
    COUNT_TEST[z]=NUM_TEST

plt.figure(figsize=(20, 15))
plt.subplot(5, 3, 1)
plt.bar(list(COUNT_TRAIN.keys()), list(COUNT_TRAIN.values()),
width=0.5, align="center")
plt.xticks(rotation=0)
plt.title("Data Training", fontsize=14, pad=10)
```

```
plt.xlabel("Type of Diagnosis", fontsize=12, color="darkblue",
labelpad=5)
plt.ylabel("Number of Diagnosed", fontsize=12,
color="darkblue", labelpad=5)

data_train = image_dataset_from_directory(TRAIN_PATH)
plt.figure(figsize=(20, 15))
for img, label in data_train.take(2):
    for i in range(10):
        ax = plt.subplot(4, 5, i+1)
        plt.imshow(img[i].numpy().astype('uint8'))
        plt.title(CLASS_NAMES[int(label[i])])
        plt.axis('off')

clear_output()
# Configuration Parameters
SEED = 23
tf.random.set_seed(SEED)
np.random.seed(SEED)

IMAGE_SIZE = 224, 224
BATCH_SIZE = 64
EPOCHS = 20
OPTIMIZER = tf.keras.optimizers.Adam
LEARNING_RATE = 0.0001
  first_image = image[0]
  for i in range(6):
    ax = plt.subplot(1, 6, i + 1)
    augmented_image =
DATA_AUGMENTATION(tf.expand_dims(first_image, 0))
    plt.imshow(augmented_image[0])
    plt.axis('off')
# Callback for Modelling
REDUCE_LR = ReduceLROnPlateau(monitor='val_loss', factor=0.2,
patience=5, min_lr=1e-5)
EARLY_STOP = EarlyStopping(monitor='val_loss', patience=10,
verbose=1)

MODEL_LeNet5 = Sequential(name='LeNet5')
DATA_AUGMENTATION

MODEL_LeNet5.add(Conv2D(filters=6, kernel_size=(5, 5),
strides=(1, 1), activation='tanh', input_shape=(224, 224, 3)))
MODEL_LeNet5.add(AveragePooling2D(pool_size=(2, 2), strides=(2,
2), padding='valid'))
```

```
MODEL_LeNet5.add(Conv2D(filters=16, kernel_size=(5, 5),
strides=(1, 1), activation='tanh'))
MODEL_LeNet5.add(AveragePooling2D(pool_size=(2, 2), strides=(2,
2)))

MODEL_LeNet5.add(Flatten())

MODEL_LeNet5.add(Dense(units=120, activation='tanh'))
MODEL_LeNet5.add(Dense(units=84, activation='tanh'))
MODEL_LeNet5.add(Dense(2, activation='softmax'))
MODEL_LeNet5.summary()

# Compile model
MODEL_LeNet5.compile(optimizer=OPTIMIZER(learning_rate=LEARNING
_RATE),
                     loss=CategoricalCrossentropy(),
                     metrics=['accuracy'])
# Training model
HIST_TRAIN_LeNet5 = MODEL_LeNet5.fit(TRAIN_DATA,
                                     validation_data=VAL_DATA,
                                     epochs=EPOCHS,
                                     callbacks=[REDUCE_LR])  #
callbacks=[REDUCE_LR, EARLY_STOP])
# Save model
MODEL_LeNet5.save('/content/drive/MyDrive/My
Model/final_best_model_LeNet5.h5')
# Create model (using Transfer Learning/Pretrained Model)
MODEL_ResNet50 = Sequential(name='ResNet50')
BASE_MODEL = tf.keras.applications.ResNet50(include_top=False,
weights="imagenet", input_shape=(224, 224, 3))

for layer in BASE_MODEL.layers:
  layer.trainable=True

IDX_LOSS_LeNet5 = np.argmin(VAL_LOSS_LeNet5)
VAL_LOWEST_LeNet5 = VAL_LOSS_LeNet5[IDX_LOSS_LeNet5]
IDX_ACC_LeNet5 = np.argmax(VAL_ACC_LeNet5)
ACC_HIGHEST_LeNet5 = VAL_ACC_LeNet5[IDX_ACC_LeNet5]
EPOCHS_RANGE_LeNet5 = [i+1 for i in
range(len(TRAIN_ACC_LeNet5))]
LOSS_LABEL_LeNet5 = f'best epoch={str(IDX_LOSS_LeNet5 + 1)}'
ACC_LABEL_LeNet5 = f'best epoch={str(IDX_ACC_LeNet5 + 1)}'

plt.figure(figsize=(15, 20))
plt.subplot(5, 2, 1)
```

```
# Accuracy
plt.plot(HIST_TRAIN_LeNet5.epoch,
HIST_TRAIN_LeNet5.history['accuracy'], label="train_accuracy")
plt.plot(HIST_TRAIN_LeNet5.epoch,
HIST_TRAIN_LeNet5.history['val_accuracy'],
label="val_accuracy")
plt.scatter(IDX_ACC_LeNet5 + 1 , ACC_HIGHEST_LeNet5, s= 30, c=
'blue', label=ACC_LABEL_LeNet5)
plt.title("Accuracy Model LeNet5", fontsize=14, pad=10)
plt.xlabel("Epochs", fontsize=12)
plt.ylabel("Accuracy",  fontsize=12)
plt.grid(alpha=0.2)
plt.legend()

# Adding Subplot 1 (For Loss)
plt.subplot(5, 2, 2)
plt.plot(HIST_TRAIN_LeNet5.epoch,
HIST_TRAIN_LeNet5.history['loss'], label="train_loss")
plt.plot(HIST_TRAIN_LeNet5.epoch,
HIST_TRAIN_LeNet5.history['val_loss'], label="val_loss")
plt.scatter(IDX_LOSS_LeNet5 + 1, VAL_LOWEST_LeNet5, s= 30, c=
'blue', label=LOSS_LABEL_LeNet5)
plt.title("Loss Model LeNet5", fontsize=14, pad=10)
plt.xlabel("Epochs", fontsize=12)
plt.ylabel("Loss",  fontsize=12)
plt.grid(alpha=0.2)
plt.legend()

plt.tight_layout
plt.show()

plt.figure(figsize=(14.43, 3))
plt.plot(LR_LeNet5, "ro-", label='Learning Rate')
plt.title('Learning Rate Model LeNet5')
plt.xlabel("Epochs", fontsize=12)
plt.ylabel("Loss", fontsize=12)
plt.grid(alpha=0.2)
plt.legend()
plt.tight_layout
plt.show()
# Define hist traing model
TRAIN_ACC_ResNet50 = HIST_TRAIN_ResNet50.history['accuracy']
TRAIN_LOSS_ResNet50 = HIST_TRAIN_ResNet50.history['loss']
VAL_ACC_ResNet50 = HIST_TRAIN_ResNet50.history['val_accuracy']
VAL_LOSS_ResNet50 = HIST_TRAIN_ResNet50.history['val_loss']
LR_ResNet50 = HIST_TRAIN_ResNet50.history['lr']
```

```
IDX_LOSS_ResNet50 = np.argmin(VAL_LOSS_ResNet50)
VAL_LOWEST_ResNet50 = VAL_LOSS_ResNet50[IDX_LOSS_ResNet50]
IDX_ACC_ResNet50 = np.argmax(VAL_ACC_ResNet50)
ACC_HIGHEST_ResNet50 = VAL_ACC_ResNet50[IDX_ACC_ResNet50]
EPOCHS_RANGE_ResNet50 = [i+1 for i in
range(len(TRAIN_ACC_ResNet50))]
LOSS_LABEL_ResNet50 = f'best epoch={str(IDX_LOSS_ResNet50 +
1)}'
ACC_LABEL_ResNet50 = f'best epoch={str(IDX_ACC_ResNet50 + 1)}'

plt.figure(figsize=(15, 20))
plt.subplot(5, 2, 1)

# Accuracy
plt.plot(HIST_TRAIN_ResNet50.epoch,
HIST_TRAIN_ResNet50.history['accuracy'],
label="train_accuracy")
plt.plot(HIST_TRAIN_ResNet50.epoch,
HIST_TRAIN_ResNet50.history['val_accuracy'],
label="val_accuracy")
plt.scatter(IDX_ACC_ResNet50 + 1 , ACC_HIGHEST_ResNet50, s= 30,
c= 'blue', label=ACC_LABEL_ResNet50)
plt.title("Accuracy Model ResNet50", fontsize=14, pad=10)
plt.xlabel("Epochs", fontsize=12)
plt.ylabel("Accuracy", fontsize=12)
plt.grid(alpha=0.2)
plt.legend()

# Adding Subplot 1 (For Loss)
plt.subplot(5, 2, 2)
plt.plot(HIST_TRAIN_ResNet50.epoch,
HIST_TRAIN_ResNet50.history['loss'], label="train_loss")
plt.plot(HIST_TRAIN_ResNet50.epoch,
HIST_TRAIN_ResNet50.history['val_loss'], label="val_loss")
plt.scatter(IDX_LOSS_ResNet50 + 1, VAL_LOWEST_ResNet50, s=30,
c='blue', label=ACC_LABEL_ResNet50)
plt.title("Loss Model ResNet50", fontsize=14, pad=10)
plt.xlabel("Epochs", fontsize=12)
plt.ylabel("Loss", fontsize=12)
plt.grid(alpha=0.2)
plt.legend()

plt.tight_layout
plt.show()

plt.figure(figsize=(14.61, 3))
plt.plot(LR_ResNet50, "go-", label='Learning Rate')
```

```
plt.title('Learning Rate Model ResNet50')
plt.xlabel("Epochs", fontsize=12)
plt.ylabel("Loss", fontsize=12)
plt.grid(alpha=0.2)
plt.legend()
plt.tight_layout
plt.show()
plt.figure(figsize=(15, 20))
plt.subplot(5, 2, 1)

# Accuracy
plt.plot(VAL_ACC_LeNet5)
plt.scatter(IDX_ACC_LeNet5 + 1 , ACC_HIGHEST_LeNet5, s=30,
c='blue', label=ACC_LABEL_LeNet5)
plt.plot(VAL_ACC_ResNet50)
plt.scatter(IDX_ACC_ResNet50 + 1 , ACC_HIGHEST_ResNet50, s=30,
c='blue', label=ACC_LABEL_ResNet50)
plt.title('Model Accuracy LeNet5 & ResNet50')
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.grid(alpha=0.2)
plt.legend(['Model LeNet5','Model ResNet50'], loc='lower
right')

# Loss
plt.subplot(5, 2, 2)
plt.plot(VAL_LOSS_LeNet5)
plt.scatter(IDX_LOSS_LeNet5 + 1, VAL_LOWEST_LeNet5, s=30,
c='blue', label=ACC_LABEL_LeNet5)
plt.plot(VAL_LOSS_ResNet50)
plt.scatter(IDX_LOSS_ResNet50 + 1, VAL_LOWEST_ResNet50, s=30,
c='blue', label=ACC_LABEL_ResNet50)
plt.title('Model Loss LeNet5 & ResNet50')
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.grid(alpha=0.2)
plt.legend(['Model LeNet5','Model ResNet50'], loc='upper
right')

plt.tight_layout
plt.show()

plt.figure(figsize=(14.61, 3))
plt.plot(LR_LeNet5, "ro-", label='Learning Rate LeNet5')
plt.plot(LR_ResNet50, "go-", label='Learning Rate ResNet50')
plt.title('Learning Rate Model LeNet5 & ResNet50')
plt.xlabel("Epochs", fontsize=12)
```

```
plt.ylabel("Loss", fontsize=12)
plt.grid(alpha=0.2)
plt.legend()
plt.tight_layout
plt.show()
TRUE_LABELS_LeNet5 = []
PREDICTED_LABELS_LeNet5 = []

# Iterate through test dataset to get true labels
for images, labels in TEST_DATA:
    TRUE_LABELS_LeNet5.extend(np.argmax(labels.numpy(),
axis=1))

    # Predict labels using the model
    MODEL_PREDICT_LeNet5 =
load_model('/content/drive/MyDrive/My
Model/final_best_model_LeNet5.h5')
    PREDICTION_LeNet5 = MODEL_PREDICT_LeNet5.predict(images,
verbose=0)
    PREDICTED_LABELS_LeNet5.extend(np.argmax(PREDICTION_LeNet5,
axis=1))

# Convert lists to NumPy arrays for easier processing
TRUE_LABELS_LeNet5 = np.array(TRUE_LABELS_LeNet5)
PREDICTED_LABELS_LeNet5 = np.array(PREDICTED_LABELS_LeNet5)

# Plot Confusion Matrix of model
CM_LeNet5 = confusion_matrix(TRUE_LABELS_LeNet5,
PREDICTED_LABELS_LeNet5)
plt.figure(figsize=(5, 3))
sns.heatmap(CM_LeNet5, annot=True, fmt='.4g', linewidths=2,
cmap='YlGnBu', cbar=True, xticklabels=CLASS_NAMES,
yticklabels=CLASS_NAMES)
plt.xlabel('Predicted Class', fontsize=12, color="darkblue",
labelpad=20)
plt.ylabel('True Class', fontsize=12, color="darkblue",
labelpad=20)
plt.title('Confusion Matrix Model LeNet5', fontsize=12, pad=20)
plt.show()
# Plot Confusion Matrix of model
TRUE_LABELS_ResNet50 = []
PREDICTED_LABELS_ResNet50 = []

# Iterate through test dataset to get true labels
for images, labels in TEST_DATA:
    TRUE_LABELS_ResNet50.extend(np.argmax(labels.numpy(),
axis=1))  # Assuming one-hot encoded labels
```

```
    # Predict labels using the model
    MODEL_PREDICT_ResNet5 =
load_model('/content/drive/MyDrive/My
Model/final_best_model_ResNet50.h5')
    PREDICTION_ResNet50 = MODEL_PREDICT_ResNet5.predict(images,
verbose=0)
    PREDICTED_LABELS_ResNet50.extend(np.argmax(PREDICTION_ResNe
t50, axis=1))

# Convert lists to NumPy arrays for easier processing
TRUE_LABELS_ResNet50 = np.array(TRUE_LABELS_ResNet50)
PREDICTED_LABELS_ResNet50 = np.array(PREDICTED_LABELS_ResNet50)

CM_ResNet5 = confusion_matrix(TRUE_LABELS_ResNet50,
PREDICTED_LABELS_ResNet50)
plt.figure(figsize=(5, 3))
sns.heatmap(CM_ResNet5, annot=True, fmt='.4g', linewidths=2,
cmap='YlGnBu', cbar=True, xticklabels=CLASS_NAMES,
yticklabels=CLASS_NAMES)
plt.xlabel('Predicted Class', fontsize=12, color="darkblue",
labelpad=20)
plt.ylabel('True Class', fontsize=12, color="darkblue",
labelpad=20)
plt.title('Confusion Matrix Model ResNet50', fontsize=12,
pad=20)
plt.show()
# Classification Report model LeNet5
print(classification_report(TRUE_LABELS_LeNet5,
                            PREDICTED_LABELS_LeNet5,
                            target_names=CLASS_NAMES))
# Classification Report model ResNet50
print(classification_report(TRUE_LABELS_ResNet50,
                            PREDICTED_LABELS_ResNet50,
                            target_names=CLASS_NAMES))
# Generate model LeNet5 probabilities and associated
predictions
MODEL_PREDICTION_LeNet5 = load_model('/content/drive/MyDrive/My
Model/final_best_model_LeNet5.h5')

TEST_PROB_LeNet5 = MODEL_PREDICTION_LeNet5.predict(TEST_DATA,
verbose=1)
TEST_PREDICT_LeNet5 = tf.argmax(TEST_PROB_LeNet5, axis=1)
# Record Classification Metrics
def generate_preformance_scores(y_true, y_pred,
y_probabilities):
```

```
    model_accuracy = accuracy_score(TRUE_LABELS_LeNet5,
PREDICTED_LABELS_LeNet5)
    model_precision, model_recall, model_f1, _ = (
        precision_recall_fscore_support(
            TRUE_LABELS_LeNet5,
            PREDICTED_LABELS_LeNet5,
            average="weighted"
        )
    )

    print(f'\nPerformance Metrics Model LeNet5:\n')
    print('============================================')
    print(f'accuracy_score:\t\t{model_accuracy:.4f}')
    print('____________________________________________')
    print(f'precision_score:\t{model_precision:.4f}')
    print('____________________________________________')
    print(f'recall_score:\t\t{model_recall:.4f}')
    print('____________________________________________')
    print(f'f1_score:\t\t{model_f1:.4f}')
    print('============================================')

    preformance_scores = {
        'accuracy_score': model_accuracy,
        'precision_score': model_precision,
        'recall_score': model_recall,
        'f1_score': model_f1
    }
    return preformance_scores
# Generate model LeNet5 performance scores
PERFORMANCE_LeNet5 = generate_preformance_scores(TEST_DATA,
                                                 TEST_PREDICT_L
eNet5,
                                                 TEST_PROB_LeNe
t5)
# Generate model ResNet50 probabilities and associated
predictions
MODEL_PREDICTION_ResNet50 =
load_model('/content/drive/MyDrive/My
Model/final_best_model_ResNet50.h5')

TEST_PROB_ResNet50 =
MODEL_PREDICTION_ResNet50.predict(TEST_DATA, verbose=1)
TEST_PREDICT_ResNet50= tf.argmax(TEST_PROB_ResNet50, axis=1)
# Record Classification Metrics ResNet50
def generate_preformance_scores(y_true, y_pred,
y_probabilities):
```

```
    model_accuracy = accuracy_score(TRUE_LABELS_ResNet50,
PREDICTED_LABELS_ResNet50)
    model_precision, model_recall, model_f1, _ = (
        precision_recall_fscore_support(
            TRUE_LABELS_ResNet50,
            PREDICTED_LABELS_ResNet50,
            average="weighted"
        )
    )

    print(f'\nPerformance Metrics Model Resnet50:\n')
    print('============================================')
    print(f'accuracy_score:\t\t{model_accuracy:.4f}')
    print('____________________________________________')
    print(f'precision_score:\t{model_precision:.4f}')
    print('____________________________________________')
    print(f'recall_score:\t\t{model_recall:.4f}')
    print('____________________________________________')
    print(f'f1_score:\t\t{model_f1:.4f}')
    print('============================================')

    preformance_scores = {
        'accuracy_score': model_accuracy,
        'precision_score': model_precision,
        'recall_score': model_recall,
        'f1_score': model_f1
    }
    return preformance_scores
# Generate model ResNet50 performance scores
PERFORMANCE_ResNet50 = generate_preformance_scores(TEST_DATA,
                                                   TEST_PREDICT
_ResNet50,
                                                   TEST_PROB_Re
sNet50)
# Record metrics with DataFrame
PERFORMANCE_DF = pd.DataFrame({
    'Model LeNet5': PERFORMANCE_LeNet5,
    'Model ResNet50': PERFORMANCE_ResNet50
}).T

# View Performance DataFrame
PERFORMANCE_DF
# Plot performance model
PERFORMANCE_DF.plot(kind="bar", figsize=(5,
3)).legend(bbox_to_anchor=(1.0, 1.0))
plt.xticks(rotation=0)
plt.title('Performance Metrics', fontsize=12, pad=10);
```

```
import requests
from io import BytesIO
from PIL import Image

# Define input shape
input_size = (224,224)
channel = (3,)
input_shape = input_size + channel
# Define labels
labels = ['Parasitized', 'Uninfected']

# Define preprocess function
def preprocess(img,input_size):
    nimg = img.convert('RGB').resize(input_size, resample=0)
    img_arr = (np.array(nimg))#/255
    return img_arr

def reshape(imgs_arr):
    return np.stack(imgs_arr, axis=0)

# Load model
MODEL_PREDICTED_LeNet5 = load_model("/content/drive/MyDrive/My
Model/final_best_model_LeNet5.h5")
MODEL_PREDICTED_ResNet50 =
load_model("/content/drive/MyDrive/My
Model/final_best_model_ResNet50.h5")
```