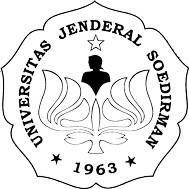
# 

ACC Komisi Syaiful 20210301

PROPOSAL TUGAS AKHIR

KLASIFIKASI CT SCAN DADA MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK IDENTIFIKASI POTENSI COVID-19

Disusun untuk memenuhi prasyarat memperoleh gelar Sarjana Teknik  
di Jurusan Teknik Elektro Universitas Jenderal Soedirman



Disusun oleh:

Rokhi Iman Sarofi

H1A017065

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN**

**UNIVERSITAS JENDERAL SOEDIRMAN**

**FAKULTAS TEKNIK**

**JURUSAN/PROGRAM STUDI TEKNIK ELEKTRO**

**PURBALINGGA**

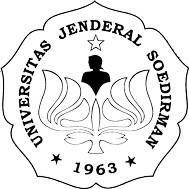
**2021**

# HALAMAN JUDUL

LAPORAN TUGAS AKHIR

KLASIFIKASI CT SCAN DADA MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK IDENTIFIKASI POTENSI COVID-19

Disusun untuk memenuhi prasyarat memperoleh gelar Sarjana Teknik  
di Jurusan Teknik Elektro Universitas Jenderal Soedirman



Disusun oleh:

Rokhi Iman Sarofi

H1A017065

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN**

**UNIVERSITAS JENDERAL SOEDIRMAN**

**FAKULTAS TEKNIK**

**JURUSAN/PROGRAM STUDI TEKNIK ELEKTRO**

**PURBALINGGA**

**2021**

# HALAMAN PENGESAHAN

Tugas Akhir dengan Judul:

KLASIFIKASI CT SCAN DADA MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK IDENTIFIKASI POTENSI COVID-19



Disusun oleh:

Rokhi Iman Sarofi  
H1A017065

Diajukan untuk memenuhi salah satu persyaratan  
memperoleh gelar Sarjana Teknik pada  
Jurusan/Program Studi Teknik Elektro  
Fakultas Teknik  
Universitas Jenderal Soedirman

Diterima dan disetujui  
Pada Tanggal : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| Pembimbing I  Imron Rosyadi, S.T., M.Sc.  NIP : 197909242003121003 |  | Pembimbing II/Lapangan  Muhammad Syaiful Aliim, S.T., M.T.  (NIP : 199009052019031021) |
| Mengetahui:  Dekan Fakultas Teknik  Dr. Eng. Suroso, S.T., M.Eng.  NIP. 197812242001121002 | | |

# HALAMAN PERNYATAAN

Dengan ini saya menyatakan bahwa dalam Laporan Tugas Akhir[[1]](#footnote-1) dengan judul ***“KLASIFIKASI CT SCAN DADA MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK IDENTIFIKASI POTENSI COVID-19”*** ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

|  |  |  |
| --- | --- | --- |
|  |  | Purbalingga, 5 Februari 2021  [materai sesuai ketentuan uu]  Ttd.  Rokhi Iman Sarofi  NIM. H1A017065 |

# HALAMAN MOTTO DAN PERSEMBAHAN

**MOTTO**

*“When there is will, there is way.”.*

**PERSEMBAHAN**

Laporan Tugas Akhir ini dapat diselesaikan atas dorongan, saran, serta  
bantuan pemikiran berbagai pihak. Pada kesempatan ini disampaikan ucapan  
terima kasih kepada :

1. Allah SWT, yang telah melimpahkan rahmat dan petunjuk selama pelaksanaan Tugas Akhir.
2. Kedua Orang Tua dan Saudara penulis atas dukungan baik moril maupun  
   materil selama pelaksanaan Tugas Akhir.
3. Ibu Farida Asriani, S.Si., M.T. selaku Ketua Jurusan Teknik Elektro
4. Bapak Imron Rosyadi, selaku pembimbing I.
5. Bapak Muhammad Syaiful Aliim selaku pembimbing II
6. Semua pihak yang telah membantu baik secara langsung maupun tidak langsung dalam pelaksanaan Tugas Akhir.

# RINGKASAN

KLASIFIKASI CT SCAN DADA MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK IDENTIFIKASI POTENSI COVID-19

Rokhi Iman Sarofi

*Corona virus Disease* 2019 (COVID-19) merupakan penyakit pernafasan akut yang disebabkan oleh virus corona jenis baru dan dapat menular dengan cepat melalui droplet. Penyakit ini memiliki gejala umum antara lain gejala gangguan pernapasan akut seperti demam, batuk, dan sesak napas. COVID-19 pertama ditemukan pada Desember 2019, di Wuhan, China.

Hingga saat ini proses untuk mendiagnosa dan konfirmasi *COVID-19* bergantung pada *real-time reverse-transcription–polymerase-chain-reaction* (RT-PCR) yang mendeteksi keberadaan *SARS-CoV-2*.[1] Selain konfirmasi hasil *RT-PCR*, elemen diagnostik utama lainnya yang dapat memfasilitasi identifikasi *COVID-19* adalah citra tomografi komputasi dada *(CT-Scan).*

Berdasarkan hal ini penulis ingin merancang sistem klasifikasi *CT Scan* untuk *COVID-19* berdasarkan citra dengan menggunakan metode convolutional neural network. Ada beberapa arsitektur CNN yang digunakan seperti *VGG16, MobileNet*, dan *ResNet*, kemudian penulis akan membandingkan hasil dari masing-masing arsitektur yang digunakan.

Kata kunci : *COVID-19*, *real-time reverse-transcription–polymerase-chain-reaction,CT Scan,* *CNN*

# *SUMMARY*

***CLASSIFICATION OF CHEST CT SCAN USES THE CONVOLUTIONAL NEURAL NETWORK (CNN) METHOD TO IDENTIFY POTENTIAL COVID-19***

Rokhi Iman Sarofi

*Coronavirus Disease 2019 (COVID-19) is an acute respiratory disease caused by a new type of coronavirus and can be transmitted quickly through droplets. This disease has common symptoms including acute respiratory symptoms such as fever, cough and shortness of breath. The first COVID-19 was discovered in December 2019, in Wuhan, China.*

*Until now the process to diagnose and confirm COVID-19 has relied on real-time reverse-transcription-polymerase-chain-reaction (RT-PCR) that detects the presence of SARS-CoV-2. [2] Apart from confirming RT-PCR results, another key diagnostic element that can facilitate identification of COVID-19 is a computed tomography image of the chest (CT-Scan).*

*Based on this, the writer wants to design a CT Scan classification system for COVID-19 based on images using the convolutional neural network method. There are several CNN architectures used such as VGG16, MobileNet, and ResNet, then the author will compare the results of each architecture used.*

*Keywords : COVID-19*, *real-time reverse-transcription–polymerase-chain-reaction,CT Scan,* *CNN*

# PRAKATA

Puji syukur kehadirat Allah SWT karena atas segala berkah, rahmat, dan  
hidayah-Nya penulis dapat menyelesaikan laporan Tugas Akhir dengan judul **“Klasifikasi CT Scan Dada Metode Convolutional Neural Network (CNN) untuk Identifikasi Potensi Covid-19”** dengan baik dan tepat waktu.

Laporan Tugas Akhir ini disusun sebagai salah satu syarat mata kuliah  
Tugas Akhir pada program studi Teknik Elektro - Universitas Jenderal Soedirman  
pada tahun 2020 yang sedang penulis jalani. Laporan Tugas Akhir ini dapat  
diselesaikan atas dorongan, saran, serta bantuan pemikiran berbagai pihak. Pada  
kesempatan ini disampaikan ucapan terima kasih kepada : Kedua orang tua, Ibu  
Farida Asriani, S.Si., M.T. selaku Ketua Jurusan Teknik Elektro Unsoed dan dosen  
pembimbing Tugas Akhir, dan segenap rekan kerja selama Tugas Akhir yang telah sabar membimbing dan banyak memberikan ilmu, sahabat-sahabat  
yang selalu memberikan dukungan, semangat, motivasi, serta doa dan semua  
pihak yang telah membantu dalam penelitian.

Akhir kata, Penulis menyadari bahwa masih banyak terdapat kekurangan  
dalam laporan Tugas Akhir ini, maka kritik dan saran yang membangun sangat  
diharapkan dari berbagai pihak. Akhir kata penulis berharap semoga Laporan ini  
dapat bermanfaat bagi semua yang membutuhkannya, terutama bagi yang akan  
menyusun Laporan Tugas Akhir terkait pada periode selanjutnya.

Purbalingga, 5 Februari 2021

Rokhi Iman Sarofi

# DAFTAR ISI

[i](#_Toc83471401)

[HALAMAN JUDUL i](#_Toc83471402)

[HALAMAN PENGESAHAN ii](#_Toc83471403)

[HALAMAN PERNYATAAN iii](#_Toc83471404)

[HALAMAN MOTTO DAN PERSEMBAHAN iv](#_Toc83471405)

[RINGKASAN v](#_Toc83471406)

[*SUMMARY* vi](#_Toc83471407)

[PRAKATA vii](#_Toc83471408)

[DAFTAR ISI viii](#_Toc83471409)

[DAFTAR TABEL liv](#_Toc83471410)

[DAFTAR GAMBAR lv](#_Toc83471411)

[DAFTAR LAMPIRAN lviii](#_Toc83471412)

[DAFTAR ISTILAH DAN SINGKATAN lix](#_Toc83471413)

[DAFTAR SIMBOL lx](#_Toc83471414)

[BAB 1 PENDAHULUAN 61](#_Toc83471415)

[1.1 Latar Belakang 61](#_Toc83471416)

[1.2 Rumusan Masalah 62](#_Toc83471417)

[1.3 Batasan Masalah 63](#_Toc83471418)

[1.4 Tujuan Penelitian 63](#_Toc83471419)

[1.5 Manfaat Penelitian 64](#_Toc83471420)

[1.6 Sistematika Penulisan 64](#_Toc83471421)

[BAB 2 TINJAUAN PUSTAKA 66](#_Toc83471422)

[2.1 Penelitian Terdahulu 66](#_Toc83471423)

[2.2 Pengolahan Citra 67](#_Toc83471424)

[*2.3 Deep Learning* 67](#_Toc83471425)

[*2.4 Convolutional Neural Network* 69](#_Toc83471426)

[*2.4.1 Input Layer* 69](#_Toc83471427)

[*2.4.2 Convolutional Layer* 70](#_Toc83471428)

[*2.4.3 Activation Layer* 71](#_Toc83471429)

[*2.4.4 Pooling Layer* 72](#_Toc83471430)

[*2.4.5 Fully Connected Layer* 73](#_Toc83471431)

[2.5 Arsitektur CNN 74](#_Toc83471432)

[2.5.1 Arsitektur *MobileNet* 74](#_Toc83471433)

[2.5.2 Arsitektur *InceptionV3* 77](#_Toc83471434)

[2.5.3 Arsitektur *VGG16* 78](#_Toc83471435)

[2.5.4 ResNet50 80](#_Toc83471436)

[*2.6 Google Colaboratory* 81](#_Toc83471437)

[*2.7 Framework Keras dan Tensorflow* 83](#_Toc83471438)

[*2.7.1 Tensorflow.js* 84](#_Toc83471439)

[*2.8 Website* 85](#_Toc83471440)

[2.8.1 HTML (*Hyper Text Markup Language*) 85](#_Toc83471441)

[2.8.2 CSS (*Cascading Style Sheets*) 86](#_Toc83471442)

[*2.8.3 Javascript* 86](#_Toc83471443)

[2.9 CT Scan 87](#_Toc83471444)

[*2.10 Coronavirus Disease 2019 (Covid-19)* 88](#_Toc83471445)

[BAB 3 METODE PENELITIAN 90](#_Toc83471446)

[3.1 Waktu dan Tempat 90](#_Toc83471447)

[3.2 Alat dan Bahan 90](#_Toc83471448)

[3.2.1 Perangkat Keras 90](#_Toc83471449)

[3.2.2 Perangkat Lunak 90](#_Toc83471450)

[3.2.3 DataSet 90](#_Toc83471451)

[3.3 Metode Penelitian 91](#_Toc83471452)

[3.3.1 Tahap Persiapan dan *Pre-Processing* Dataset 94](#_Toc83471453)

[3.3.2 Tahap Desain Arsitektur 94](#_Toc83471454)

[3.3.3 Tahap Pengujian 95](#_Toc83471455)

[3.4 Waktu dan Jadwal Penelitian 95](#_Toc83471456)

[BAB 4 HASIL DAN PEMBAHASAN 97](#_Toc83471457)

[*4.1 Dataset* 97](#_Toc83471458)

[4.2 Preprocessing 98](#_Toc83471459)

[*4.3* Perancangan Program pada *Google Colaboratory* 98](#_Toc83471460)

[4.4 Hasil Pelatihan pada *Google Colaboratory* 100](#_Toc83471461)

[4.4.1 *Lung Parenchyma Classification* *ResNet50* (Original Image) 100](#_Toc83471462)

[4.4.2 *Lung Parenchyma Classification ResNet50* (*Preprocessed Image)* 107](#_Toc83471463)

[4.4.3 Lung Parenchyma Classification VGG16 (Original Image) 114](#_Toc83471464)

[4.4.4 Lung Parenchyma Classification VGG16 (Preprocessed Image) 120](#_Toc83471465)

[4.4.5 Covid Positivity Classification ResNet50 (Original Image) 127](#_Toc83471466)

[4.4.6 Covid Positivity Classification ResNet50 (Preprocessed Image) 131](#_Toc83471467)

[4.4.7 Covid Positivity Classification VGG16 (Original Image) 135](#_Toc83471468)

[4.4.8 Covid Positivity Classification VGG16 (Preprocessed Image) 140](#_Toc83471469)

[4.4.9 Risk Classification ResNet50 (Original Image) 144](#_Toc83471470)

[4.4.10 Risk Classification ResNet50 (Preprocessed Image) 151](#_Toc83471471)

[4.4.11 Risk Classification VGG16 (Original Image) 157](#_Toc83471472)

[4.4.12 Risk Classification VGG16 (Preprocessed Image) 163](#_Toc83471473)

[4.4.13 Mortality Classification ResNet50 (Original Image) 169](#_Toc83471474)

[4.4.14 Mortality Classification ResNet50 (Preprocessed Image) 176](#_Toc83471475)

[4.4.15 Mortality Classification VGG16 (Original Image) 182](#_Toc83471476)

[4.4.16 Mortality Classification VGG16 (Preprocessed Image) 188](#_Toc83471477)

[4.5 Perbandingan Hasil Pelatihan 195](#_Toc83471478)

[4.5.1 Perbandingan Lung Parenchyma Classification Arsitektur VGG16 dan Arsitektur ResNet50 Original Image 195](#_Toc83471479)

[4.5.2 Perbandingan Covid Positivity Classification Arsitektur VGG16 dan Arsitektur ResNet50 197](#_Toc83471480)

[4.5.3 Perbandingan Risk Classification Arsitektur VGG16 dan Arsitektur ResNet50 199](#_Toc83471481)

[4.5.4 Perbandingan Mortality Classification Arsitektur VGG16 dan Arsitektur ResNet50 201](#_Toc83471482)

[4.6 Hasil Pengujian pada Google Colaboratory 203](#_Toc83471483)

[*4.6.1* Pengujian Klasifikasi *Lung Parenchyma* pada *Google Colaboratory* Menggunakan *Dataset Original Image* 204](#_Toc83471484)

[4.6.2 Pengujian pada *Google Colaboratory* Menggunakan *Dataset Preprocessed Image* 207](#_Toc83471611)

[4.6.3 Pengujian Klasifikasi *Covid Positivity* pada *Google Colaboratoty* Menggunakan *Dataset Original Image* 210](#_Toc83471738)

[4.6.4 Pengujian Klasifikasi *Covid Positivity* pada *Google Colaboratoty* Menggunakan *Dataset Preprocessed Image* 212](#_Toc83471825)

[4.6.5 Pengujian Klasifikasi *Risk* pada *Google Colaboratory* Menggunakan *Dataset Original Image* 214](#_Toc83471912)

[4.6.6 Pengujian Klasifikasi *Risk* pada *Google Colaboratory* Menggunakan *Dataset Preprocessed Image* 217](#_Toc83472039)

[4.6.7 Pengujian Klasifikasi *Mortality* pada *Google Colaboratory* Menggunakan *Dataset Original Image* 220](#_Toc83472166)

[4.6.8 Pengujian Klasifikasi *Mortality* pada *Google Colaboratory* Menggunakan *Dataset Preprocessed Image* 223](#_Toc83472293)

[4.7 Perbandingan Pengujian pada *Google Colaboratory* 226](#_Toc83472420)

[4.7.1 Perbandingan Pengujian Klasifikasi *Lung Parenchyma* pada *Google Colaboratory* 226](#_Toc83472421)

[*4.7.2* Perbandingan Pengujian *Covid Positivity* pada *Google Colaboratory* 227](#_Toc83472422)

[*4.7.3* Perbandingan Pengujian *Risk* pada *Google Colaboratory* 227](#_Toc83472423)

[*4.7.4* Perbandingan Pengujian *Mortality* pada *Google Colaboratory* 228](#_Toc83472424)

[4.8 Perancangan Aplikasi Web 229](#_Toc83472425)

[4.9 Hasil Pengujian pada Aplikasi Web 231](#_Toc83472426)

[4.9.1 Pengujian *Lung Parenchyma* pada Aplikasi Web menggunakan Citra Original 231](#_Toc83472427)

[4.9.2 Pengujian *Lung Parenchyma* pada Aplikasi Web Menggunakan *Dataset Preprocessed Image* 234](#_Toc83472554)

[4.9.3 Pengujian *Covid Positivity* pada Aplikasi Web Menggunakan *Dataset Original Image* 237](#_Toc83472681)

[4.9.4 Pengujian *Covid Positivity* pada Aplikasi Web Menggunakan *Dataset Preprocessed Image* 239](#_Toc83472768)

[4.9.5 Pengujian *Risk* pada Aplikasi Web Menggunakan *Dataset Original Image* 241](#_Toc83472855)

[4.9.6 Pengujian *Risk* pada Aplikasi Web Menggunakan *Dataset Preprocessed Image* 244](#_Toc83472982)

[4.9.7 Pengujian Klasifikasi *Mortality* pada *Google Colaboratory* Menggunakan *Dataset Original Image* 247](#_Toc83473109)

[4.9.8 Pengujian Klasifikasi *Mortality* pada *Google Colaboratory* Menggunakan *Dataset Preprocessed Image* 250](#_Toc83473236)

[BAB 5 SARAN DAN KESIMPULAN 253](#_Toc83473363)

[5.1 Kesimpulan 253](#_Toc83473364)

[5.2 Saran 255](#_Toc83473365)

[DAFTAR PUSTAKA 257](#_Toc83473366)

[LAMPIRAN 261](#_Toc83473367)

[*Lampiran 1.* Kode Sumber Sistem Klasifikasi pada *Google Colaboratory* 261](#_Toc83473368)

[*Lampiran 2.* Kode Sumber Pengujian Sistem Klasifikasi pada *Google Colaboratory* 279](#_Toc83473403)

[Lampiran 3. Kode Sumber Aplikasi Web Deteksi *Covid-19* menggunakan *CT Scan* Dada 281](#_Toc83473408)

[Lampiran 4. Gambar pengujian deteksi 289](#_Toc83473409)

[*Lampiran 5.* Hasil pengujian deteksi pada infrastruktur *google colaboratory* 289](#_Toc83473410)

[*Lampiran 6.* Hasil pengujian deteksi pada infrastruktur aplikasi web 290](#_Toc83473411)

[Lampiran 7. Lembar Permohonan TA 292](#_Toc83473412)

[Lampiran 8. Transkrip Nilai Sementara 293](#_Toc83473413)

[Lampiran 9. Kartu Studi Mahasiswa 295](#_Toc83473414)

[Lampiran 10. Kartu Kendali (Bukti Bimbingan) 296](#_Toc83473415)

[BIODATA PENULIS 298](#_Toc83473416)

# DAFTAR TABEL

Tabel 2.1Arsitektur MobileNet 32

Tabel 3.1Rincian Jadwal Penelitian 53

Tabel 4.1Hasil pengujian deteksi lung parenchyma menggunakan citra original 165

Tabel 4.2Hasil pengujian deteksi lung parenchyma menggunakan citra preprocessed 168

Tabel.4.3Hasil pengujian covid positivity pada google colaboratory dengan citra original 171

Tabel 4.4Tabel pengujian covid positivity menggunakan citra preprocessed 173

Tabel 4.5Hasil pengujian deteksi risk menggunakan citra preprocessed 175

Tabel 4.6Hasil pengujian deteksi risk menggunakan citra preprocessed 178

Tabel 4.7Hasil pengujian deteksi mortality menggunakan citra preprocessed 181

Tabel 4.8Hasil pengujian deteksi menggunakan citra preprocessed 184

Tabel 4.9 Hasil pengujian deteksi lung parenchyma pada aplikasi web menggunakan citra original 192

Tabel 4.10Hasil pengujian deteksi lung parenchyma pada aplikasi web menggunakan citra preprocessed 195

Tabel 4.11Hasil pengujian deteksi covid positivity pada aplikasi web menggunakan citra original 198

Tabel 4.12Hasil pengujian deteksi covid positivity pada aplikasi web menggunakan citra preprocessed 200

Tabel 4.13Hasil pengujian deteksi risk pada aplikasi web menggunakan citra original 202

Tabel 4.14Hasil pengujian deteksi risk pada aplikasi web menggunakan citra preprocessed 205

Tabel 4.15Hasil pengujian deteksi mortality menggunakan citra preprocessed 208

Tabel 4.16Hasil pengujian deteksi menggunakan citra preprocessed 211

# DAFTAR GAMBAR

Gambar 2.1Perbedaan antara lapisan layer pada Jaringan Saraf Tiruan dengan lapisan layer deep learning 25

Gambar 2.2Alur Proses CNN 27

Gambar 2.3Proses Input Layer 28

Gambar 2.4 Proses Convolutional Layer 29

Gambar 2.5Fungsi Activation Layer 30

Gambar 2.6Feature Map 31

Gambar 2.7Proses Pooling 31

Gambar 2.8 Proses Fully Conected Layer 32

Gambar 2.9Arsitektur Depthwise Separable Convolutional 34

Gambar 2.10Standard convolution (kiri), Depthwise separable convolution (kanan) 35

Gambar 2.11Arsitektur VGG16 37

Gambar 2.12Arsitektur ResNet50 39

Gambar 2.13Tampilan Antarmuka Google Colaboratory 40

Gambar 2.14 Susunan Perangkat Keras dan Perangkat Lunak Deep Learning 42

Gambar 2.15Citra CT Scan 46

Gambar 3.1Desain arsitektur sistem 50

Gambar 3.2Diagram alir sistem klasifikasi 51

Gambar 3.3Diagram alir pengujian 52

Gambar 3.4Diagram alir penelitian 53

Gambar 4.1Kurva akurasi pelatihan dan validasi 60

Gambar 4.2Kurva kesalahan (loss) pelatihan dan validasi LP ResNet50 original image 61

Gambar 4.3Confusion matrix LP ResNet50 original image 62

Gambar 4.4Kurva akurasi pelatihan dan validasi LP ResNet50 preprocessed image 67

Gambar 4.5Kurva kesalahan (loss) pelatihan dan validasi LP ResNet50 preprocessed image 68

Gambar 4.6Confusion matrix LP Resnet50 preprocessed image 69

Gambar 4.7Kurva akurasi pelatihan dan validasi LP VGG16 original image 73

Gambar 4.8Kurva kesalahan pelatihan dan validasi LP VGG16 original image 74

Gambar 4.9Confusion Matrix LP VGG16 original image 75

Gambar 4.10Kurva akurasi pelatihan dan validasi LP VGG16 preprocessed image 80

Gambar 4.11Kurva kesalahan pelatihan dan validasi LP VGG16 preprocessed image 81

Gambar 4.12 Confusion matrix LP VGG16 preprocessed image 82

Gambar 4.13Kurva akurasi pelatihan dan validasi CP ResNet50 original image 86

Gambar 4.14Kurva akurasi pelatihan dan validasi CP ResNet50 preprocessed image 87

Gambar 4.15Confusion matrix CP ResNet50 preprocessed image 88

Gambar 4.16 Kurva akurasi pelatihan dan validasi CP ResNet50 91

Gambar 4.17 Kurva kesalahan pelatihan dan validasi CP ResNet50 92

Gambar 4.18 Confusion matrix CP ResNet50 preprocessed image 93

Gambar 4.19 Kurva akurasi pelatihan dan validasi CP VGG16 original image 95

Gambar 4.20 Kurva kesalahan pelatihan dan validasi CP VGG16 96

Gambar 4.21 Confusion matrix CP VGG16 original image 97

Gambar 4.22 Kurva akurasi pelatihan dan validasi CP VGG16 100

Gambar 4.23 Kurva kesalahan pelatihan dan validasi CP VGG16 101

Gambar 4.24 Confusion matrix CP VGG16 preprocessed image 102

Gambar 4.25 Kurva akurasi pelatihan dan validasi risk ResNet50 original image 104

Gambar 4.26 Kurva akurasi pelatihan dan validasi risk ResNet50 original image 105

Gambar 4.27 Confusion matrix risk VGG16 original image 106

Gambar 4.28 Kurva akurasi pelatihan dan validasi risk ResNet50 111

Gambar 4.29 Kurva kesalahan pelatihan dan validasi risk ResNet50 112

Gambar 4.30 Confusion matrix risk ResNet50 preprocessed 113

Gambar 4.31 Kurva akurasi pelatihan dan validasi risk VGG16 original image 117

Gambar 4.32Kurva kesalahan pelatihan dan validassi risk VGG16 original image 118

Gambar 4.33Confusion matrix risk VGG16 original image 119

Gambar 4.34 Kurva akurasi pelatihan dan validasi risk VGG16 124

Gambar 4.35Kurva kesalahan pelatihan dan validasi risk VGG16 125

Gambar 4.36 Confusion matrix risk VGG16 preprocessed image 126

Gambar 4.37 Kurva akurasi pelatihan dan validasi mortality ResNet50 original image 130

Gambar 4.38 Kurva kesalahan pelatihan dan validasi mortality ResNet50 original image 131

Gambar 4.39 Confusion matrix mortality ResNet50 preprocessed image 132

Gambar 4.40 Kurva akurasi pelatihan dan validasi mortality ResNet50 137

Gambar 4.41 Kurva kesalahan pelatihan dan validasi mortality ResNet50 preprocessed image 138

Gambar 4.42 Confusion matrix mortality ResNet50 preprocessed image 139

Gambar 4.43 Kurva akurasi pelatihan dan validasi mortality VGG16 143

Gambar 4.44 Kurva kesalahan pelatihan dan validasi mortality VGG16 144

Gambar 4.45 Confusion matrix mortality VGG16 original image 145

Gambar 4.46 Kurva akurasi pelatihan dan validasi mortality VGG16 150

Gambar 4.47 Kurva kesalahan pelatihan dan validasi mortality VGG16 151

Gambar 4.48 Confusion matrix mortality VGG16 preprocessed image 152

Gambar 4.49 Grafik perbandingan akurasi pelatihan dan validasi 157

Gambar 4.50 Grafik perbandingan kesalahan pelatihan dan validasi 158

Gambar 4.51 Grafik perbandingan akurasi pelatihan dan validasi klasifikasi covid positivity 159

Gambar 4.52 Grafik perbandingan kesalahan pelatihan dan validasi 160

Gambar 4.53Grafik perbandingan akurasi pelatihan dan validasi risk classification 161

Gambar 4.54Grafik perbandingan kesalahan pelatihan dan validasi risk classification 162

Gambar 4.55 Grafik perbandingan akurasi pelatihan dan validasi 163

Gambar 4.56 Grafik perbandingan kesalahan pelatihan dan validasi 164

Gambar 4.57 Diagram alir pengujian pada google colaboratory 165

Gambar 4.58Grafik perbandingan hasil pengujian lung parenchyma pada google colaboratory 188

Gambar 4.59 Grafik perbandingan hasil pengujian covid positivity pada google colaboratory 189

Gambar 4.60 Grafik perbandingan hasil pengujian risk pada google colaboratory 190

Gambar 4.61Grafik perbandingan hasil pengujian mortality pada google colaboratory 191

Gambar 4.62Tampilan antarmuka aplikasi web identifikasi ct scan dada untuk covid-19 192

# DAFTAR LAMPIRAN

*Lampiran 1.* Kode Sumber Sistem Klasifikasi pada *Google Colaboratory* 224

*Lampiran 2.* Kode Sumber Pengujian Sistem Klasifikasi pada *Google Colaboratory* 242

Lampiran 3. Kode Sumber Aplikasi Web Deteksi *Covid-19* menggunakan *CT Scan* Dada 244

Lampiran 4. Gambar pengujian deteksi 252

*Lampiran 5.* Hasil pengujian deteksi pada infrastruktur *google colaboratory* 252

*Lampiran 6.* Hasil pengujian deteksi pada infrastruktur aplikasi web 253

Lampiran 7. Lembar Permohonan TA 255

Lampiran 8. Transkrip Nilai Sementara 256

Lampiran 9. Kartu Studi Mahasiswa 258

Lampiran 10. Kartu Kendali (Bukti Bimbingan) 259

# DAFTAR ISTILAH DAN SINGKATAN

|  |  |
| --- | --- |
| *CNN* | : *Convolutional Nerual Network*, merupakan salah satu metode dari deep learning |
| *DNN* | : *Deep Neural Network* |
| *Covid-19* | : *Coronavirus Disease* 2019, merupakan penyakit menular yang disebabkan oleh corona virus jenis baru |
| *Dataset* | : Objek yang merepresentasikan data dan relasinya di memori |
| *Python* | : Bahasa pemrograman tingkat tinggi yang bersifat *open source* |
| *CPU* | : *Central Processing Unit,* pusat kendali |
| *GPU* | : *Graphical Processing Unit,* unit pemrosesan grafis |
| *API* | : *Application Programming Interface,* merupakan penjermah komunikasi antara klien dan server |
| *Flask* | : *Web framework* dari *python* |

# DAFTAR SIMBOL

|  |  |
| --- | --- |
| TP : | *True Positive* |
| TN : | *True Negative* |
| FP : | *False Positive* |
| FN : | *False Negative* |

# PENDAHULUAN

## Latar Belakang

Pada bulan Desember 2019 ditemukan wabah pneumonia yang terkait dengan virus korona baru dan disebut dengan sindrom pernafasan akut parah atau *Corona virus 2* (*SARS-Cov-2*) di Wuhan, China. Virus ini dapat menular begitu cepat, ketika seseorang terkena infeksi virus tersbeut maka akan mengalami beberapa gejala umum antara lain gejala gangguan pernapasan akut seperti demam, batuk, dan sesak napas. Kemudian WHO menamai penyakit yang disebabkan oleh *Novel Coronavirus* sebagai *Coronavirus Disease 2019* (*COVID-19*).[2]

Hingga saat ini proses untuk mendiagnosa dan konfirmasi *COVID-19* bergantung pada *real-time reverse-transcription–polymerase-chain-reaction (RT-PCR)* yang mendeteksi keberadaan *SARS-CoV-2*.[1] Selain konfirmasi *hasil RT-PCR*, elemen diagnostik utama lainnya yang dapat memfasilitasi identifikasi *COVID-19* adalah citra tomografi komputasi dada *(CT-Scan).* Karakteristik pencitraan dada *(CT-Scan)* dari paru-paru yang terinfeksi dilaporkan termasuk ke dalam *ground glass opacity (GGO)* dan konsolidasi berkorelasi berat yang dapat mengidentifikasi COVID-19.[3]

*Deep learning* adalah sub bidang pembelajaran khusus *dari machine learning* dan merupakan pandangan baru tentang pembelajaran dari data yang menekankan pada lapisan *(layers)* pembelajaran berturut-turut dari representasi yang semakin bermakna.[4] Model *deep learning* dapat mempelajari komputasinya sendiri atau dapat bekerja dengan otak sendiri. Sebuah model *deep learning* dirancang untuk terus menganalisa data dengan struktur logika yang menyerupai cara bagaimana manusia mengambil sebuah keputusan. *Deep learning* banyak diimplementasikan untuk klasifikasi maupun identifikasi suatu objek melalui citra atau gambar dengan cara mempelajari fitur dari data tersebut secara otomatis.

*Convolutional Neural Network* (CNN) adalah pengembangan dari *Multilayer Perceptron (MLP)* yang didesain untuk mengolah data dua dimensi, misalnya pada citra. CNNtermasuk dalam jenis *Deep Neural Network* karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra.[5] Prinsip dari CNN adalah dapat belajar secara langsung dari data atau citra, sehingga dapat mengurangi beban pada pemrograman.

Oleh karena itu, untuk memberikan dasar bagi pemodelan sistemik yang dapat memfasilitasi diagnosis dini *COVID-19* dibutuhkan perancangan identifikasi *COVID-19* berdasarkan citra *CT-Scan* dengan menggunakan *deep learning*. Sehingga pada tugas akhir ini, penulis memilih bahan kajian tentang identifikasi *CT-Scan COVID-19* dengan judul **“KLASIFIKASI *CT SCAN DADA* MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK IDENTIFIKASI POTENSI COVID-19*”**

## Rumusan Masalah

Bagaimana perancangan arsitektur CNN untuk klasifikasi *lung parenchyma, risk, mortality* dan *Covid positivity* berdasarkan *CT Scan* dada?

Bagaimana pengujian klasifikasi lung parenchyma, risk, mortality dan Covid positivity berdasarkan CT Scan dada?

Bagaimana perancangan aplikasi *in-browser* sebagai antarmuka klasifikasi citra *CT Scan* dada?

## Batasan Masalah

Batasan masalah yang dibuat membatasi fokus penelitian pada tugas akhir ini adalah sebagai berikut.

1. Metode deep learning yang digunakan untuk klasifikasi *CT-Scan* dada untuk identifikasi potensi *Covid-19* menggunakan *Convolutional Neural Network* (CNN)*.*
2. Program klasifikasi *CT-Scan COVID-19* berdasarkan citra dibuat dengan Bahasa Pemrograman *Python* dengan antarmuka dan infrastruktur *Google Colaboratory*
3. Pembuatan model *deep learning* pada klasifikasi *CT-Scan COVID-19* dengan citra menggunakan *Framework Keras* dan *TensorFlow*
4. Penentuan jumlah dataset pelatihan, validasi dan pengujian tidak dibahas dalam penelitian ini.

## Tujuan Penelitian

* 1. Merancang arsitektur CNNuntuk klasifikasi *lung parenchyma, risk, mortality* dan *Covid positivity* berdasarkan *CT Scan* dada*.*
  2. Melatih arsitektur CNN untuk klasifikasi *lung parenchyma, risk, mortality* dan *Covid positivity* berdasarkan *CT Scan* dada*.*
  3. Menguji arsitektur CNN klasifikasi *lung parenchyma, risk, mortality* dan *Covid positivity* berdasarkan *CT Scan* dada.
  4. Menggabungkan dua metode klasifikasi terbaik dan melakukan pengujian terhadap metode tersebut.
  5. Menggunakan model klasifikasi *lung parenchyma* dan *risk classifier* sebagai *browser based classifier*.

## Manfaat Penelitian

Manfaat yang diharapkan setelah penelitian ini selesai dilaksanakan adalah sebagai berikut.

Mampu menerapakan ilmu yang telah didapatkan pada mata kuliah yang bersangkutan untuk menyelesaikan tugas akhir.

Menawarkan solusi yang lebih murah dan efisien untuk klasifikasi *CT Scan* dada untuk identifikasi potensi *Covid-19.*

## Sistematika Penulisan

Dalam penulisan laporan tugas akhir ini, dibuat sistematika penulisan sebagai berikut :

**BAB I Pendahuluan**

Bab Pendahuluan berisi tentang judul penelitian, latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, dan manfaat penelitian.

**BAB II Tinjauan Pustaka**

Bab Tinjauan Pustaka berisi tentang dasar kajian pustaka yang mendasari berbagai gagasan tentang penelitian terdahulu serta teori-teori pendukung untuk penelitian seperti penjelasan tentang pengolahan citra, *deep learning, convolutional neural network* (CNN), arsitektur CNN, *google colaboratory, framework keras* dan *tensorflow*, *website,* *CT Scan* dan *COVID-19*.

**BAB III Metode Penelitian**

Bab Metode Penelitian berisi tentang urutan langkah penelitian yang dilakukan, meliputi waktu dan tempat penelitian, alat dan bahan yang digunakan, metode penelitian, sumber data, alur penelitian dan jadwal penelitian.

**BAB IV Pembahasan**

Bab Pembahasan berisi tentang hasil dan analisa dari penelitian klasifikasi *CT Scan COVID-19* berdasarkan citra dengan menggunakan metode *convolutional neural network*. Pada bab ini juga akan membahas hasil kajian dan analisa yang disesuaikan dengan rumusan masalah yang dibuat.

**BAB V Penutup**

Bab Penutup berisi tentang kesimpulan dan saran atas hasil penelitian yang telah dilaksanakan.

# TINJAUAN PUSTAKA

## Penelitian Terdahulu

Dalam penyusunan penelitian tugas akhir “Klasifikasi CT Scan Dada Menggunakan Metode *Convolutional Neural Network* (CNN) untuk Identifikasi Potensi *Covid-19*” terdapat beberapa penelitian terdahulu yang berhubungan yaitu membahas tantang klasifikasi objek berdasarkan citra dengan *machine learning*. Berikut ini adalah beberapa contoh dari penelitian tersebut :

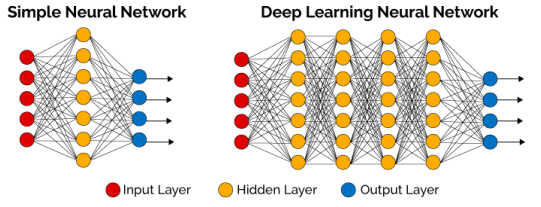
* 1. Penelitian Jepri dengan judul “Identifikasi Penyakit pada Daun Tomat dan Daun Singkong Berdasarkan Citra Daun Menggunakan Metode *Convolutional Neural Network* (CNN) Berbasis Android”[6]. Penelitian ini membahas tentang rancang bangun sistem berbasis *android* yang dapat mengklasifikasi penyakit pada daun tomat dan daun singkong berdasarkan citra dengan menggunakan arsitektur *VGG16*, *InceptionV3* dan *MobileNet*.
  2. Penelitian Andreas Galang Anugerah yang berjudul “Klasifikasi Tingkat Keganasan Kanker Paru-Paru pada Citra *Computed Tomography* (CT) *Scan* Menggunakan Metode *Convolutional Neural Network* (CNN)”[7]. Penelitian ini membahas tentang system klasifikasi tingkat keganasan kanker paru-paru berdasarkan citra *CT Scan* dengan menggunakan 4 arsitektur CNN yaitu *CanNet, LeNet, VGG16* dan *VGG19*.
  3. Penelitian Wanshan Ning, Shijun Lei, Jingjing Yang, Yukun Cao, Peiran Jiang, Qianqian ynag, Jiao Zhang, Xiaobei Wang, Fenghua Chen, Zhi Geng, Liang Xiong, Hongmei Zhou, Yaping Guo, Yulan Zeng, Heshui Shi, Lin Wang, Yu Xue dan Zheng Wang yang berjudul “*iCTCF: an Integrative Resource of Chest Computed Tomography Images and Clinical Features with COVID-19 Pneumonia”*[3]. Penelitian ini membahass tentang sistem yang dapat memprediksi dan mendiagnosa *COVID-19* dengan menggabungkan metode CNN untuk citra *CT Scan* dan metode DNN untuk *Clinical Features* (CF).

## Pengolahan Citra

Pengolahan citra adalah pemrosesan citra, khususnya dengan  
menggunakan komputer, menjadi citra yang kualitasnya lebih baik. Pengolahan  
citra bertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia  
atau mesin (dalam hal ini komputer). Teknik-teknik pengolahan citra  
mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan  
keluarannya juga citra yang berkualitas lebih baik daripada citra masukan.[8]

## *Deep Learning*

*Deep Learning* (Pembelajaran Dalam) atau sering dikenal dengan istilah  
Pembelajaran Struktural Mendalam (*Deep Structured Learning*) atau Pembelajaran Hierarki (*Hierarchical learning*) adalah salah satu cabang dari ilmu pembelajaran mesin (*Machine Learning*) yang terdiri algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam. Teknik dan algoritma dalam pembelajaran dalam dapat digunakan baik untuk kebutuhan pembelajaran terarah (*supervised learning*), pembelajaran tak terarah (*unsupervised learning*) dan semi-terarah (*semi-supervised learning*) dalam berbagai aplikasi seperti pengenalan citra, pengenalan suara, klasifikasi teks, dan sebagainya.[9]



Gambar 2.1Perbedaan antara lapisan layer pada Jaringan Saraf Tiruan dengan lapisan layer deep learning

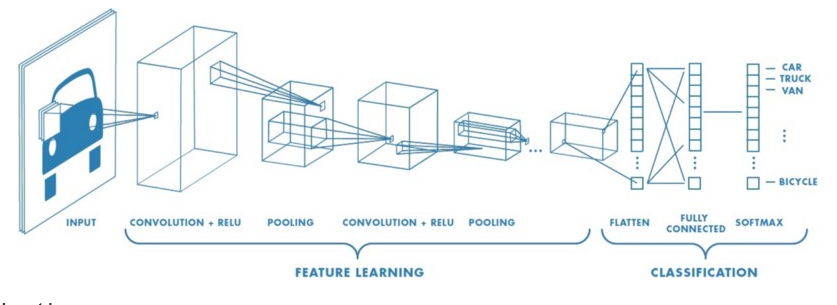
*Deep Learning* adalah salah satu jenis algoritma jaringan saraf tiruan yang menggunakan meta data sebagai *input* dan mengolahnya menggunakan sejumlah lapisan tersembunyi (*hidden layer*) transformasi non-linier dari data masukan untuk menghitung nilai *output*. Algoritma pada *Deep Learning* memiliki fitur yang unik yaitu sebuah fitur yang mampu mengekstraksi secara otomatis. Hal ini berarti algoritma yang dimilikinya secara otomatis dapat menangkap fitur yang relevan sebagai keperluan dalam pemecahan suatu masalah. Algoritma semacam ini sangat penting dalam sebuah kecerdasan buatan karena mampu mengurangi beban pemrograman dalam memilih fitur yang eksplisit.

Dalam jaringan saraf tiruan tipe *Deep Learning* setiap lapisan tersembunyi bertanggung jawab untuk melatih serangkaian fitur unik berdasarkan *output* dari jaringan sebelumnya. Algoritma ini akan menjadi semakin kompleks dan bersifat abstrak ketika jumlah lapisan tersembunyi (*hidden layer*) semakin bertambah banyak. Jaringan saraf yang dimiliki oleh *Deep Learning* terbentuk dari hierarki sederhana dengan beberapa lapisan hingga tingkat tinggi atau banyak lapisan(*multilayer*). Berdasarkan hal itulah *Deep Learning* dapat digunakan untuk memecahkan masalah kompleks yang lebih rumit dan terdiri dari sejumlah besar lapisan transformasi non linier.

## *Convolutional Neural Network*

*Convolutional Neural Network* (CNN) adalah pengembangan dari *Multilayer Perceptron* (MLP) yang didesain untuk mengolah data dua dimensi, misalnya pada citra. CNNtermasuk dalam jenis *Deep Neural Network* karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra.[5]

Alur dari proses CNN dalam mengolah citra masukan hingga mengklasifikasikan citra tersbut ke dalam kelas tertentu berdasarkan nilai keluarannya dapat dilihat pada Gambar 2.2



Gambar 2.2Alur Proses CNN

### *Input Layer*

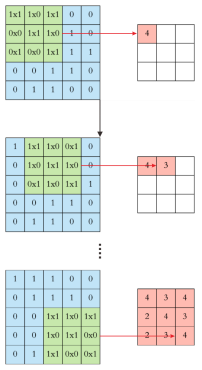
*Input Layer* padaCNN menampung nilai piksel dari citra masukan. Untuk citra dengan ukuran 64x64 dengan 3 channel warna, yaitu channel warna *RGB (Red, Green, Blue)*, maka yang menjadi masukan adalah piksel array yang memiliki ukuran 64x64x3.[10]

**

Gambar 2.3Proses Input Layer

### *Convolutional Layer*

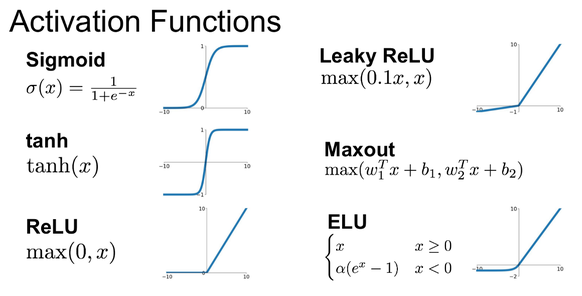
*Convolutional Layer* adalah inti dari CNN. *Convolutional Layer* menghasilkan citra baru yang menunjukkan fitur dari citra input. Dalam proses tersebut, *Convolutional Layer* menggunakan filter pada setiap citra yang menjadi masukan. Filter pada layer ini berupa array 2 dimensi yang memiliki ukuran 5x5, 3x3 atau 1x1. Proses *Convolutional Layer* dengan menggunakan filter pada layer ini akan menghasilkan *feature map* yang akan digunakan pada *Action Layer*. Gambar 2.4 menunjukkan alur proses dari *Convolutional Layer*



Gambar 2.4 Proses Convolutional Layer

### *Activation Layer*

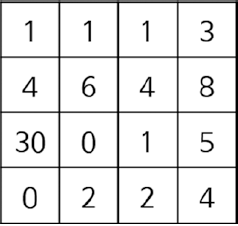
*Activation Layer* adalah layer dimana *feature map* dimasukkan ke dalam fungsi aktivasi. Fungsi aktivasi digunakan untuk mengubah nilai-nilai pada *feature map* pada *range* tertentu sesuai dengan fungsi aktivasi yang digunakan. Ini bertujuan untuk meneruskan nilai yang menampilkan fitur dominan dari citra yang masuk pada layer berikutnya.[10] Fungsi aktivasi yang umum digunakan dapat dilihat pada Gambar 2.5



Gambar 2.5Fungsi Activation Layer

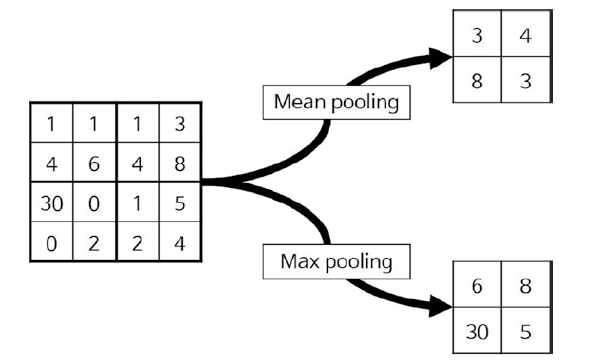
### *Pooling Layer*

*Pooling Layer* menerima masukan dari *activation layer* kemudian mengurangi jumlah parameternya. *Pooling* juga biasa disebut dengan *subsampling* atau *downsampling* yang mengurangi dimensi dari *feature map* tanpa menghilangkan informasi penting di dalamnya. Proses dalam *Pooling Layer* cukup sederhana, pertama proses akan menentukan *downsampling* yang akan digunakan pada *feature map*, misalnya 2x2. Setelah itu proses akan melakukan *pooling*  pada *feature map*, sebagai contoh akan digunakan *feature map* berukuran 4x4 berikut.



Gambar 2.6Feature Map

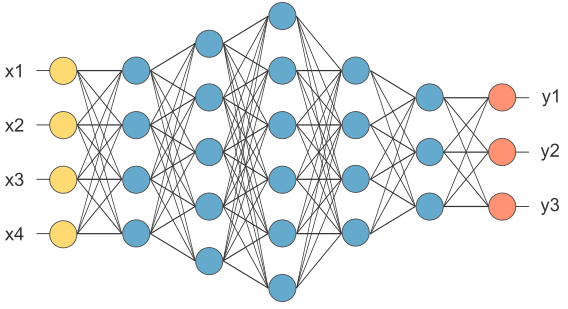
Kemudian selanjutnya proses akan menggunakan matriks 2x2 untuk melakukan *pooling*. Proses *pooling* sendiri terdiri dari beberapa macam seperti *Max Pooling*, *Mean Pooling* dan *Sum Pooling*.



Gambar 2.7Proses Pooling

### *Fully Connected Layer*

Setelah melalui beberapa proses di atas, kemudian hasil dari *pooling layer* akan digunakan sebagai masukan untuk *Fully Connected Layer*. Layer ini memiliki kesamaan struktur dengan *Artificial Neural Network* pada umumnya yaitu memiliki *input layer*, *hidden layer* dan *output layer* dimana masing-masing memiliki neuron-neuron yang saling terhubung dengan neuron-neuron pada layer tetangganya. Gambar 2.8 menunjukkan contoh dari proses *Fully Connected Layer*



Gambar 2.8 Proses Fully Conected Layer

Pada Gambar 2.8 dapat diketahui bahwa sebelum hasil *pooling* digunakan sebagai masukan, terlebih dahulu diubah menjadi vector (x1, x2, x3, dst) kemudian selanjutnya akan diproses ke dalam *Fully Connected Layer*. Pada layer terakhir di dalam *Fully Connected Layer* akan digunakan fungsi aktivasi *sigmoid* atau *softmax* untuk menentukan klasifikasi dari citra masukan dari *input layer CNN*.

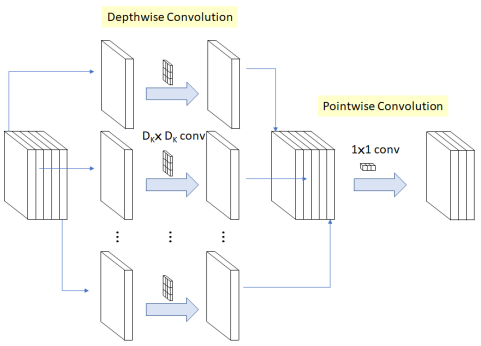
## Arsitektur CNN

### Arsitektur *MobileNet*

*MobileNet* merupakan salah satu arsitektur *Convolutional Neural Network* (CNN) yang dapat digunakan untuk mengatasi kebutuhan akan *computing resource* berlebih. Sama seperti namanya, *MobileNet* dibuat agar dapat digunakan untuk ponsel[11]. Perbedaan mendasar antara arsitektur *MobileNet* dan arsitektur CNN pada umumnya adalah penggunaan lapisan atau layer konvolusi dengan ketebalan filter yang sesuai dengan ketebalan citra masukan. *MobileNet* membagi konvolusi menjadi *depthwise convolution* dan *pointwise convolution*.

* + 1. *Depthwise Separable Convolution*

Model *MobileNet* didasarkan pada konvolusi mendalam yang dapat  
dipisahkan (*depthwise separable convolution*), yaitu bentuk konvolusi yang dibentuk dengan menguraikan konvolusi standar (*standard convolution*) menjadi konvolusi mendalam (*depthwise convolution*) dan konvolusi 1×1 yang disebut konvolusi searah (*pointwise convolution*) [12]. Untuk *MobileNets*, *depthwise convolution* menerapkan satu *filter* ke setiap saluran masukkannya. *Pointwise convolution* kemudian menerapkan konvolusi 1×1 untuk menggabungkan keluaran dari *depthwise convolution*. Konvolusi standar melakukan *filter* dan menggabungkan masukan ke dalam *set* keluaran baru dalam satu langkah. *Depthwise separable convolution* membaginya menjadi dua lapisan, lapisan terpisah untuk *filter* dan lapisan terpisah untuk menggabungkannya. Penguraian ini memiliki efek mengurangi komputasi dan ukuran model secara drastis.[12] Arsitektur dari *Depthwise separable convolution* dapat dilihat pada Gambar 2.9



Gambar 2.9Arsitektur Depthwise Separable Convolutional

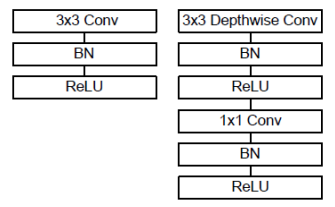
* + 1. Struktur Jaringan

Arsitektur *MobileNet* didefinisikan seperti pada **Error! Reference source not found.** . Semua lapisan diikuti oleh *batchnorm* (*backnormalization*) dan *ReLU non-linier* dengan pengecualian lapisan akhir yang terhubung penuh yang tidak memiliki nonlinier dan dimasukkan ke dalam lapisan *softmax* untuk klasifikasi.

Tabel 2.1Arsitektur MobileNet

|  |  |  |
| --- | --- | --- |
| *Type/Stride* | *Filter Shape* | *Input Size* |
| Conv / s2 | 3 × 3 × 3 × 32 | 224 × 224 × 3 |
| Conv dw / s1 | 3 × 3 × 32 dw | 112 × 112 × 32 |
| Conv / s1 | 1 × 1 × 32 × 64 | 112 × 112 × 32 |
| Conv dw / s2 | 3 × 3 × 64 dw | 112 × 112 × 64 |
| Conv / s1 | 1 × 1 × 64 × 128 | 56 × 56 × 64 |
| Conv dw / s1 | 3 × 3 × 128 dw | 56 × 56 × 128 |
| Conv / s1 | 1 × 1 × 128 × 128 | 56 × 56 × 128 |
| Conv dw / s2 | 3 × 3 × 128 dw | 56 × 56 × 128 |
| Conv / s1 | 1 × 1 × 128 × 256 | 28 × 28 × 128 |
| Conv dw / s1 | 3 x 3 x 256 dw | 28 x 28 x 256 |
| Conv / s1 | 1 x 1 x 256 x 256 | 28 x 28 x 256 |
| Conv dw / s2 | 3 x 3 x 256 dw | 28 x 28 x 256 |
| Conv / s1 | 1 × 1 × 256 × 512 | 14 × 14 × 256 |
| 5x Conv dw / s1 | 3 × 3 × 512 dw | 14 × 14 × 512 |
| 5x Conv / s1 | 1 × 1 × 512 × 512 | 14 × 14 × 512 |
| Conv dw / s2 | 3 × 3 × 512 dw | 14 × 14 × 512 |
| Conv / s1 | 1 × 1 × 512 × 1024 | 7 × 7 × 512 |
| Conv dw / s2 | 3 × 3 × 1024 dw | 7 × 7 × 1024 |
| Conv / s1 | 1 × 1 × 1024 × 1024 | 7 × 7 × 1024 |
| Avg Pool / s1 | Pool 7 × 7 | 7 × 7 × 1024 |
| FC / s1 | 1024 × 1000 | 1 × 1 × 1024 |
| Softmax / s1 | Classifier | 1 × 1 × 1000 |

Pada sebuah lapisan konvolusi standar hanya menggunakan lapisan  
konvolusi biasa sebesar 3x3. Sedangkan pada sebuah lapisan *depthwise separable  
convolution* dipisahkan menjadi dua konvolusi yaitu 3x3 *depthwise convolution* dan 1x1 *pointwise convolution* serta *batchnorm* dan *ReLU non-linier* di setiap lapisan konvolusinya seperti pada Gambar 2.10 berikut.



Gambar 2.10Standard convolution (kiri), Depthwise separable convolution (kanan)

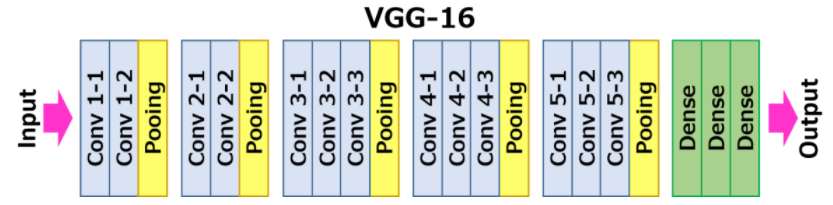
### Arsitektur *InceptionV3*

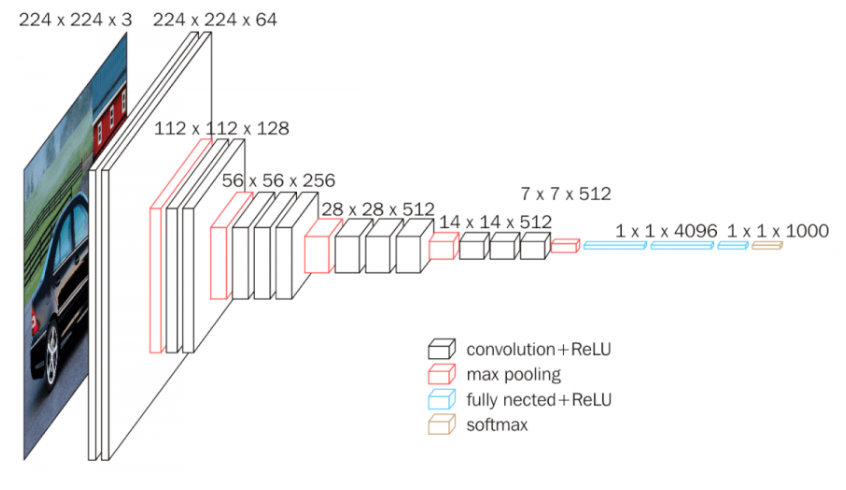
*InceptionV3* adalah sebuah model *deep learning convolutional neural  
network* yangdikembangkan oleh *Google* untukmemenuhi *ImageNet Large Visual* *recognition challenge* pada tahun 2012. Model *Inception* menggunakan filter pada layer *convolutional,* tidak seperti *convolutional layer* biasa. Hasil dari beberapa filter tersebut dijadikan satu lagi menggunakan *channel concat* sebelum masuk ke dalam iterasi berikutnya[13].

Tujuan dari modul *inception* adalah untuk bertindak sebagai *multilevel  
feature extractor* dengan menghitung filter-filter *convolutional* dalam modul yang  
sama. Hasil dari filter-filter tersebut kemudian ditumpukan ke dalam dimensi  
*channel* sebelum dimasukan ke dalam lapisan selanjutnya.

### Arsitektur *VGG16*

*VGG16* adalah model jaringan saraf konvolusional yang diusulkan oleh K. Simonyan dan A. Zisserman dari *Universitas Oxford* dalam makalahnya yang berjudul "*Very Deep Convolutional Networks for Large-Scale Image Recognition "*.[14] Model ini mencapai top-5 dengan akurasi pengujian 92,7% di *ImageNet*, yang memiliki kumpulan data lebih dari 14 juta gambar dan 1000 kelas. *VGG16* adalah salah satu model terkenal yang dikirim ke *ILSVRC-2014*. Gambar 2.11 berikut menunjukkan arsitektur *VGG16*.





Gambar 2.11Arsitektur VGG16

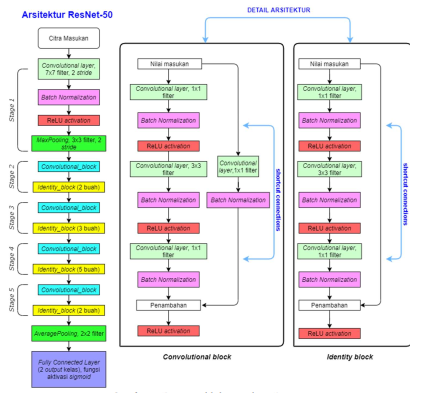
Input ke layer cov1 berupa citra berukuran tetap 224 x 224 RGB. Citra melewati tumpukan lapisan convolusional (conv.), Di mana filter digunakan dengan bidang penerima yang sangat kecil sebesar 3×3. Dalam salah satu konfigurasinya juga menggunakan filter konvolusi 1 × 1, yang dapat dilihat sebagai transformasi linier dari saluran input (diikuti oleh non-linieritas). Langkah (Stride) konvolusi ditetapkan ke 1 piksel; spasial padding konv. input lapisan sedemikian rupa sehingga resolusi spasial dipertahankan setelah konvolusi, yaitu padding adalah 1-piksel untuk konv. 3 × 3. lapisan. Penyatuan spasial dilakukan oleh lima lapisan max-pooling, yang mengikuti beberapa konv. lapisan (tidak semua lapisan konv. diikuti oleh max pooling). Max-pooling dilakukan melalui jendela 2x2 piksel, dan dengan 2 stride.

Tiga lapisan Fully-Connected (FC) mengikuti tumpukan lapisan convolusional (yang memiliki kedalaman berbeda dalam arsitektur berbeda) dua yang pertama memiliki masing-masing 4096 kanal, yang ketiga melakukan klasifikasi ILSVRC 1000 arah dan karenanya berisi 1000 kanal (satu untuk masing-masing kelas). Lapisan terakhir adalah lapisan soft-max. Konfigurasi dari lapisan Fully-Connected sama di semua jaringan.

Semua hidden layers dilengkapi dengan fungsi aktivasi ReLU. Juga dicatat bahwa tidak ada jaringan (kecuali satu) yang berisi Local Response Normalization (LRN), normalisasi tersebut tidak meningkatkan kinerja pada dataset ILSVRC, namun mengarah pada peningkatan konsumsi memori dan waktu komputasi.

### ResNet50

*Resnet50* merupakan salah satu arsitektur CNN yang memperkenalkan konsep *shortcut connections*. Konsep ini memiliki keterkaitan dengan *vanishing gradient problem* yang terjadi saat usaha memperdalam struktur suatu *network* dilakukan. Untuk memperdalam suatu *network* dengan tujuan meningkatkan performansinya tidak bisa dilakukan dengan hanya menumpuk *layer*. Semakin dalam suatu *network* dapat memunculkan *vanishing gradient problem* yang dapat membuat gradient menjadi sangat kecil dan mengakibatkan performansi menjadi turun.[15]

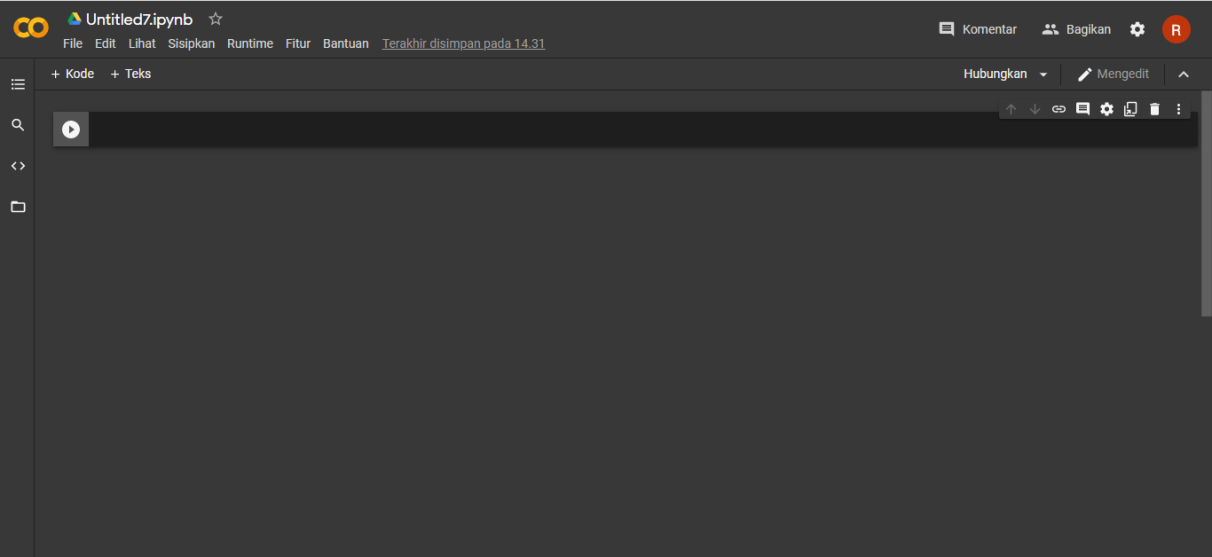


Gambar 2.12Arsitektur ResNet50

Oleh karena itu *ResNet* memperkenalkan konsep *shortcut connections* dimana dalam konsep ini fitur yang merupakan input dari *layer* sebelumnya dijadikan sebagai input terhadap output dari *layer* tersebut. Hal ini merupakan solusi untuk meminimalisir hilangnya fitur-fitur penting pada saat proses konvolusi. Secara keseluruhan *ResNet50* terdiri dari 5 stage proses konvolusi yang kemudian dilanjutkan *average pooling* dan diakhiri dengan *fully connected layer* sebagai layer prediksi.

## *Google Colaboratory*

*Google Colaboratory* adalah layanan berbasis *cloud Google* yang mereplikasi *Jupyter Notebook* pada *cloud*. Dengan *Google Colaboratory* tidak perlu menginstal apapun untuk menggunakannya. Pada sebagian besar hal, penggunaan *Google Colaboratory* sama seperti instalasi *pada Desktop Jupyter Notebook*.[16] *Google Colaboratory* ditujukan khusus untuk para pembaca yang menggunakan sesuatu selain pengaturan desktop standar untuk mengerjakan contoh - contoh. Gambar 2.13 menunjukkan tampilan antarmuka untuk Google Colaboratory.



Gambar 2.13Tampilan Antarmuka Google Colaboratory

Untuk menggunakan Google Colaboratory, diwajibkan harus memiliki akun Google untuk dapat mengakses agar sebagian besar fitur pada Google Colaboratory dapat berfungsi dengan baik.

Seperti halnya Jupyter Notebook, penggunaan Google Colaboratory untuk melakukan tugas tertentu dalam paradigma berorientasi sel. Terdapat kemiripan pada antarmuka Jupyter Notebook dan Google Colaboratory. Google Colaboratory dapat membuat berbagai tipe sel dan menggunakannya untuk membuat buku catatan.

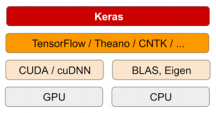
## *Framework Keras dan Tensorflow*

*Keras* adalah salah satu *framework deep learning* untuk *Python* yang mampu melatih dan mendefinisikan hampir semua jenis model *deep learning*.[4] *Keras* menawarkan *Application Programming Interface* (API) yang konsisten dan mampu berjalan di atas *Tensorflow, Theano* atau *Microsoft Cognitive Toolkit* (CNTK).

*Keras* memiliki beberapa fitur utama sebagai berikut :

* *Keras* memungkinkan kode yang sama untuk berjalan mulus di CPU atau GPU
* Memiliki API yang ramah pengguna dan memudahkan pembuatan prototipe model pembelajaran dalam dengan cepat
* Memiliki dukungan built-in untuk jaringan konvolusional (untuk *computer vision*), jaringan berulang (untuk pemrosesan urutan), dan kombinasi antara keduanya
* Mendukung arsitektur jaringan arbitrer: model multi-input atau multi-output, berbagi lapisan, berbagi model, dan sebagainya. *Keras* pada dasarnya cocok untuk membuat model pembelajaran dalam apa pun.

*Keras* didistribusikan di bawah lisensi MIT, yang berarti dapat digunakan secara bebas dalam proyek komersial. *Keras* kompatibel dengan semua versi *Python* dari 2.7 hingga 3.6.



Gambar 2.14 Susunan Perangkat Keras dan Perangkat Lunak Deep Learning

Gambar 2.14 merupakan susunan perangkat keras dan perangkat lunak deep learning. Melalui TensorFlow (atau Theano, atau CNTK), Keras dapat berjalan mulus di CPU dan GPU. Saat berjalan di CPU, TensorFlow sendiri menggabungkan library level rendah untuk operasi tensor yang disebut Eigen. Pada GPU, TensorFlow menggabungkan pustaka operasi deep learning yang dioptimalkan dengan baik yang disebut NVIDIA CUDA Deep Neural Network library (cuDNN).

### *Tensorflow.js*

Tensorflow.js merupakan tool Javascript library yang dikembangkan oleh Google dan diperkenalkan sebagai open-source pada tahun 2015, tool ini digunakan untuk training dan penggunaan machine learning model di browser. Tensorflow.js merupakan library tambahan untuk Tensorflow yang merupakan library dari machine learning untuk phyton. [17]

*TensorFlow.js* menyediakan banyak fitur dan model yang akan  
menyederhanakan tugas yang sulit dan menghabiskan waktu untuk dilatih model  
*machine learning*. Di sisi model, *TensorFlow.js* hadir dengan beberapa model  
pelatihan untuk tujuan yang berbeda, seperti *PoseNet* untuk memperkirakan pose  
yang dilakukan seseorang secara *real time*. *Toxicity classifier*, untuk mendeteksi  
apakah suatu teks mengandung konten *toxic*, dan yang terakhir model *Coco SSD*,  
model yang digunakan untuk mendeteksi objek yang mengidentifikasi dan  
melokalkan beberapa objek dalam sebuah gambar.

## *Website*

*Website* adalah halaman informasi yang disediakan melalui jalur internet sehingga bissa diakses selama terkoneksi dengan jaringan internet. Sebuah situs website umumnya merupakan bagian dari suatu nama domain atau sub domain di *World Wide Web* (WWW) di internet. WWW terdiri dari seluruh situs web yang tersedia kepada publik. Halaman – halaman sebuah situs web diakses dari sebuah URL yang menjadi akar (*root*) yang disebut dengan *homepage* (beranda).[18]

Sebuah halaman web adalah dokumen yang ditulis dalam format HTML (*Hyper Text Markup Language*) yang hampir selalu bias diakses melalui HTTP. HTTP yaitu protokol yang menyampaikan informasi dari *server website* untuk ditampilkan kepada *users* melalui web browser.

Beberapa website membutuhkan subskripsi (data masukan) agar user dpaat mengakses sebagian atau seluruh isi dari website tersebut. Misalnya pada beberapa situs bisnis yang membutuhkan subskripsi agar user dapat mengaksesnya.

### HTML (*Hyper Text Markup Language*)

Hypertext Markup Language (HTML) adalah bahasa markup yang umum digunakan untuk membuat halaman web. Sebenarnya HTML bukanlah sebuah bahasa pemrograman. Jika ditinjau dari namanya, HTML merupakan bahasa markup atau penandaan terhadap sebuah dokumen teks. Tanda tersebut di gunakan untuk menentukan format atau style dari teks yang ditandai.[19]

HTML dibuat oleh Tim Berners-Lee ketika masih bekerja untuk CERN dan dipopulerkan pertama kali oleh browser Mosaic. Selama awal tahun 1990 HTML mengalami perkembangan yang sangat pesat. Setiap pengembangan HTML pasti akan menambahkan kemampuan dan fasilitas yang lebih baik dari versi sebelumnya. Sebelum suatu HTML disahkan sebagai suatu dokumen HTML standar, ia harus disetujui dulu oleh W3C untuk dievaluasi secara ketat. Setiap terjadi perkembangan suatu versi HTML, maka mau tak mau browser pun harus memperbaiki diri agar bisa mendukung kode-kode HTML yang baru tersebut. Sebab jika tidak, browser tak akan bisa menampilkan HTML tersebut.

### CSS (*Cascading Style Sheets*)

CSS adalah bahasa penulisan yang digunakan untuk mendeskripsikan penampilan dari sebuah dokumen *markup*. Dalam penggunaannya CSS selalu digunakan bersama HTML. CSS biasanya disimpan di dalam sebuah file berekstensi .css dan disematkan di dalam dokumen HTML untuk memberikan *style* pada halaman tersebut. Pada dasarnya CSS dibuat untuk memisahkan *style* dari halaman web, sehingga antara konten pada HTML dan desain tampilan pada dokumen CSS dapat dikerjakan di dua tempat berbeda.[20]

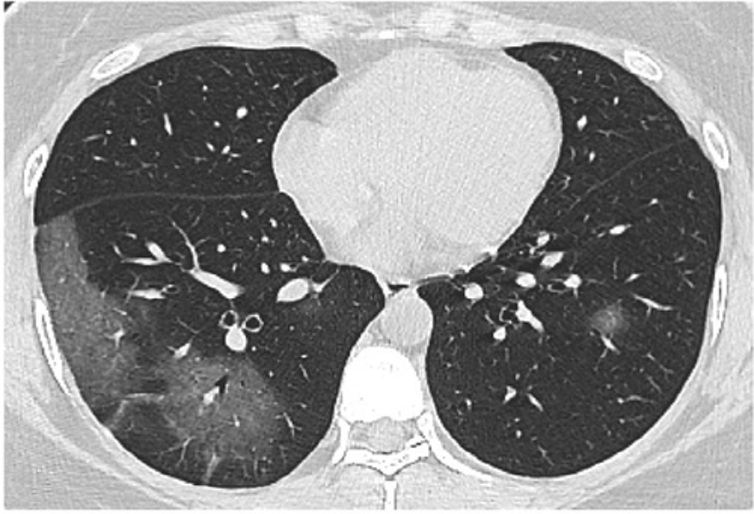
### *Javascript*

*JavaScript* adalah bahasa pemrograman web yang bersifat *Client Side Programming Language*. *Client Side Programming Language* adalah tipe bahasa pemrograman yang pemrosesannya dilakukan oleh *client*. Aplikasi *client* yang dimaksud merujuk kepada *web browser* seperti *Google Chrome, Mozilla Firefox, Opera Mini* dan sebagainya. [21]

*JavaScript* pertama kali dikembangkan pada pertengahan dekade 90-an. *JavaScript* berbeda dengan bahasa pemrograman *Java*. Untuk penulisannya, *JavaScript* dapat disisipkan di dalam dokumen HTML ataupun dijadikan dokumen tersendiri yang kemudian diasosiasikan dengan dokumen lain yang dituju. *JavaScript* mengimplementasikan fitur yang dirancang untuk mengendalikan bagaimana sebuah halaman web berinteraksi dengan *user*.

## CT Scan

*CT Scan* adalah Pemindaian tomografi yang menggabungkan serangkaian gambar sinar-X yang diambil dari berbagai sudut di sekitar tubuh dan menggunakan pemrosesan komputer untuk membuat gambar penampang (irisan) tulang, pembuluh darah, dan jaringan di dalam tubuh[22]. Gambar *CT scan* memberikan informasi yang lebih rinci daripada sinar-X biasa.



Gambar 2.15Citra CT Scan

*CT scan* memiliki beberapa kegunaan. *CT scan* dapat digunakan untuk memvisualisasikan hampir semua bagian tubuh dan digunakan untuk mendiagnosis penyakit atau cedera serta untuk merencanakan perawatan medis, bedah atau radiasi.

## *Coronavirus Disease 2019 (Covid-19)*

*Coronavirus Disease 2019 (COVID-19)* adalah penyakit saluran pernafasan akut yang disebabkan oleh *Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-COV2)* dan dapat menular[23]. *COVID-19* pertama ditemukan pada Desember 2019 di Wuhan, China. Selanjutnya tanggal 30 Januari 2020, *World Health Organization* (WHO) menetapkan kejadian tersebut sebagai Kedaruratan Kesehatan Masyarakat yang Meresahkan Dunia (KKMMD) dan pada tanggal 11 Maret 2020 *COVID-19* ditetapkan sebagai pandemi.

Bentuk gejala umum dari *COVID-19* berupa gejala pernafasan akut seperti demam , batuk kering dan sesak nafas. Dengan kasus yang berat akan menyebabkan pneumonia, sindrom pernafasan akut, gagal ginjal, hingga kematian.

Sampai sekarang untuk mendeteksi *COVID-19* menggunakan tes amplifikasi *asam nukleat* (NAAT) seperti *real time reversetranscription polymerase chain reaction (*rRT-PCR*)*[24]. Selain itu, ada beberapa elemen diagnostik utama lainnya yang dapat memfasilitasi identifikasi *COVID-19* termasuk fitur klinis (CF) dan pencitraan tomografi komputasi dada (*CT Scan* ) . Karakteristik pencitraan *CT* Scan dari paru-paru yang terinfeksi dilaporkan termasuk *ground glass opacity (*GGO*)* dan konsolidasi berkorelasi berat. Meskipun gambaran lengkap mengenai masing-masing elemen masih menunggu penggambaran penuh, penggabungan fitur elemen diagnostik tersebut di atas secara komprehensif dapat secara kolektif meningkatkan akurasi dan kemanjuran diagnosis.[3]

# METODE PENELITIAN

## Waktu dan Tempat

Penelitian ini dilakukan pada bulan Desember 2020 sampai bulan Maret 2021 di Kampus Fakultas Teknik Universitas Jenderal Soedirman yang terletak di Km5. Jl Mayor Jenderal Sungkono, Desa Blater, Kabupaten Purbalingga, Jawa Tengah.

## Alat dan Bahan

Alat dan bahan yang digunakan pada penelitian tugas akhir ini adalah sebagai berikut.

### Perangkat Keras

* + - 1. Laptop *ASUS M409DA* dengan spesifikasi *processor AMD Athlon Silver 3050U.*

### Perangkat Lunak

* + - * 1. Sistem Operasi *Windows 10 64 bit*
      1. *Web browser* *Google Chrome Version* 87.0.4280.88 (64-bit)

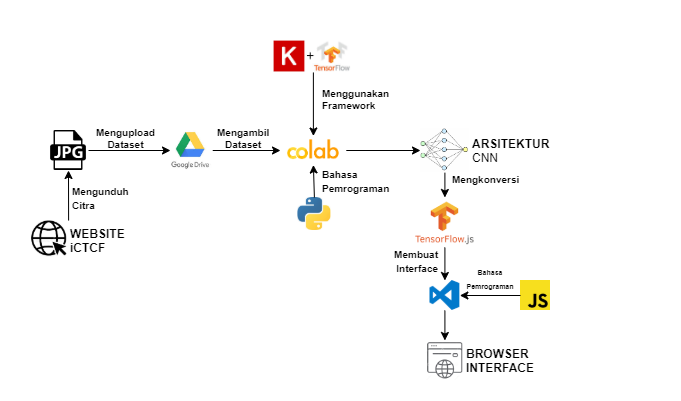
1. *Google Colaboratory*
2. *Google Colaboratory*
3. Layanan *Repository Web Development* pada platform *Github*

### DataSet

Pada penelitian ini digunakan dataset untuk melakukan *training* pada proses *Deep Learning* yang diperoleh dari kumpulan data set *CT Scan* dada di website iCTCFdalam bentuk gambar \*.jpg

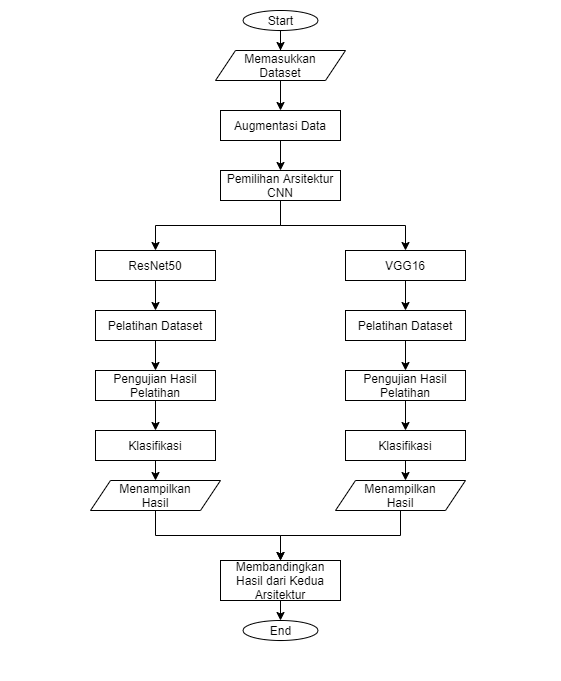
## Metode Penelitian

Penelitian ini dilaksanakan untuk melakukan empat klasifikasi, yaitu *lung parenchyma classifier, risk classifier, mortality classifier* dan *Covid positivity classifier* melalui *CT Scan* dada. Penelitian ini dibagi ke dalam beberapa tahapan yaitu tahap persiapan dan *pre-processing* dataset, tahap desain arsitektur, dan tahap pengujian. Desain arsitektur dari sistem yang akan dibuat disajikan pada **Error! Reference source not found.** berikut :



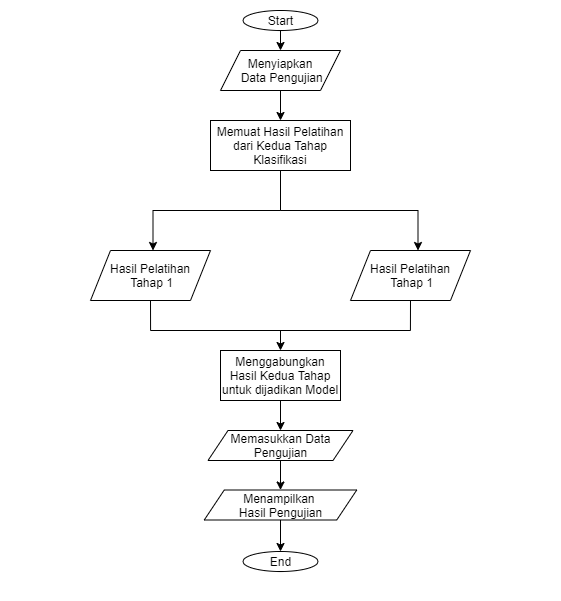
Gambar 3.1Desain arsitektur sistem

Kemudian untuk diagram alir masing – masing sistem klasifikasi pada penelitian ini dapat dilihat pada Gambar 3.2 berikut :



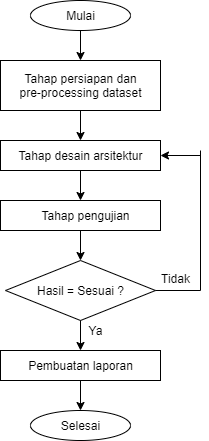
Gambar 3.2Diagram alir sistem klasifikasi

Untuk diagram alir pengujian sistem yang merupakan gabungan dari dua tahap dapat dilihat seperti pada Gambar 3.3 berikut :



Gambar 3.3Diagram alir pengujian

Adapun diagram alir dari tahapan-tahapan penelitian yang dilakukan adalah seperti yang disajikan pada Gambar 3.4berikut.



Gambar 3.4Diagram alir penelitian

### Tahap Persiapan dan *Pre-Processing* Dataset

Pada tahap ini penyiapan dan *pre-processing* *dataset* dilakukan dengan mengambil gambar dataset  *CT Scan* dada dengan format gambar “\*.jpg” dari web iCTCFyang selanjutnya akan dimasukkan ke *Google Colaboratory*.

### Tahap Desain Arsitektur

Proses awal desain arsitektur dimulai dari membuat kode sumber untuk program CNN dan mengimpor *Framework Keras* dan *TensorFlow* yang dibuat menggunakan infrastruktur *Google Colaboratory* dengan bahasa pemrograman *Python* dan disimpan dalam bentuk *file Jupyter Notebooks* “.ipynb” dan kemudian disimpan ke *Github*. Selanjutnya *dataset* pelatihan tersebut diambil dari *Google Drive* dan disimpan ke dalam tempat penyimpanan sementara pada *Google Colaboratory*. Sistem yang dirancang ini menggunakan dua macam arsitektur jaringan CNN, yaitu *ResNet50* dan *VGG16* untuk masing-masing tahap klasifikasi berdasarkan *CT Scan* sedangkan untuk metode yang digunakan adalah metode *transfer learning* dengan mengambil *file Keras* “\*.h5” yang sudah dilatih sebelumnya pada kedua arsitektur tersebut dan melakukan *feature extraction* terhadap *file* tersebut. Selain itu juga ditambah satu *filter*/*feature map* tambahan sebagai keluaran untuk klasifikasi tersebut.

### Tahap Pengujian

Pada tahap pengujian seluruh datasetpelatihan dilatih terhadap masing- masing arsitektur jaringan CNN yang digunakan sampai dengan panjangnya *epochs* yang ditentukan. Sesudah itu datasetpengujian diekstrak dan dibandingkan hasilnya dengan data pengujian untuk melakukan prediksi pada setiap tahap klasifikasi serta membandingkan hasilnya dengan kedua arsitektur yang digunakan.

Selain itu tahap pengujian juga dilakukan dengan menggabungkan dua tahap klasifikasi dengan mengambil unjuk kerja terbaik dari masing-masing tahap klasifikasi. Kemudian dari kedua tahap tersebut akan digabungkan dan dilakukan pengujian.

## Waktu dan Jadwal Penelitian

Penelitian dilaksanakan dalam waktu 4 bulan dimulai dari bulan Desember 2020 sampai dengan bulan Maret 2021 dengan rincian jadwal kegiatan sebagai berikut.

Tabel 3.1Rincian Jadwal Penelitian

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Kegiatan | Bulan 1 | | | | Bulan 2 | | | | Bulan 3 | | | | Bulan 4 | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1. | Tahap Persiapan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2. | Tahap *pre-processing data* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3. | Tahap Desain arsitektur |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4. | Tahap Pengujian dan Evaluasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5. | Tahap Akhir |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# HASIL DAN PEMBAHASAN

## *Dataset*

*Dataset* yang digunakan dalam proses pelatihan algoritma *Convolutional Neural Network* (CNN) dan proses pengujian model klasifikasi *CT scan* dada untuk deteksi dapat diakses melalui link <http://ictcf.biocuckoo.cn> . *Dataset* tersebut berisi citra *CT scan* dada dari 50 pasien untuk setiap kelas yang kemudian digunakan untuk melakukan empat proses klasifikasi, yaitu meliputi *Lung Parenchyma Classification* yang terdiri dari 3 kelas (pCT, nCT dan NiCT), *Covid Positivity Classification* yang terdiri dari 2 kelas (*Positive* dan *Negative*), *Risk Classification* yang terdiri dari 3 kelas (*Control, Type I*, dan *Type II*) dan *Mortality Classification* yang terdiri dari 3 kelas (*Cured, Deceased* dan *Unknown*).

Pada proses pelatihan klasifikasi *Lung Parenchyma*, jumlah dataset yang digunakan sebanyak 1540 citra, yang terdiri dari 512 citra NiCT, 512 citra pCT dan 516 citra nCT. Kemudian dari 1540 citra akan diambil sebanyak 20% untuk *dataset test* dan 20% untuk *dataset* validasi. Kemudian pada sistem klasifikasi *Covid Positivity* menggunakan 1203 citra yang terdiri dari 603 citra kelas *Negative* dan 600 citra kelas *Positive*. Selanjutnya dari 1203 citra akan diambil sebanyak 20% untuk *dataset test* dan 20% untuk *dataset* validasi. Sedangkan pada sistem klasifikasi *Risk* menggunakan 1552 citra yang terdiri dari 516 citra kelas *Control*, 516 citra kelas *Type I*, dan 520 citra kelas *Type II*. Dari 1552 citra, kemudian akan diambil sebanyak 20% *sebagai dataset test* dan 20% sebagai *dataset* validasi. Dan untuk sistem yang terakhir yaitu klasifikasi *Mortality* menggunakan 1583 citra yang terdiri dari 531 citra kelas *Cured*, 512 citra kelas *Deceased* dan 540 citra kelas *Unknown*. Dari 1583 citra akan diambil sebanyak 20% sebagai *dataset test* dan 20% untuk *dataset* validasi.

## Preprocessing

Dalam proses pelatihan *dataset*, menggunakan 2 jenis *dataset*, yaitu *dataset original* dan *dataset preprocessed*. *Preprocessing* adalah proses awal dilakukannya perbaikan suatu citra untuk menghilangkan *noise*.

Proses *preprocessing* untuk *dataset preprocessed* menerapkan beberapa tahap pemrosesan. Pertama, mengkonversi citra *CT scan* dada skala abu – abu menjadi *CT scan* dada biner. Kemudian menghilangkan *noise* pada citra dengan hanya menyisakan arsitektur citra yang terhubung dengan tubuh manusia, yaitu bagian parenkim paru – paru. Untuk mempertahankan bagian parenkim paru – paru yang berwarna hitam menggunakan algoritma *flood fill* yang bertujuan untuk mengisi latar belakang citra dengan warna putih. Proses yang terakhir adalah mengubah skala citra menjadi 200x200 piksel agar citra lebih cepat diproses dan menghindari distorsi dengan menggunakan *interpolasi bilinear*.

## Perancangan Program pada *Google Colaboratory*

Perancangan program pada Google Colaboratoty menggunakan bahasa pemrograman Python. Perancangan dilakukan dengan menggunakan fungsi library yang disediakan oleh Google Colaboratory dan menggunakan Framework Keras dengan backend Tensorflow untuk mempermudah dan mempercepat proses pelatihan dan pengujian pada program.

Untuk merancang system, hal pertama yang dilakukan adalah mengimpor fungsi library yang dibutuhkan untuk menjalankan program. Kemudian tahap selanjutnya adalah mengimpor dataset yang sebelumnya sudah disimpan pada Google Drive. Proses mengimpor dataset ke dalam directory Google Colab bersifat sementara. Kemudian setelah dataset selesai diimpor, proses selanjutnya adalah membuat dataframe untuk dataset pelatihan dan dataset pengujian dengan menggunakan rasio 80% : 20%.

Untuk meningkatkan akurasi pada pelatihan, dilakukan proses augmentasi. Proses augmentasi ini meliputi zoom, rotate, flip dan pembagian dataset pelatihan secara acak menjadi dataset pelatihan dan dataset validasi dengan rasio 80% : 20%. Setelah dilakukan augmentasi, dataset yang akan dilatih harus diatur ukurannya menjadi 224x224 piksel agar sesuai dengan masukan untuk arsitektur VGG16 dan ResNet50.

Tahap selanjutnya adalah pembuatan model atau arsitektur yang akan digunakan untuk pelatihan. Perancangan pada tahap ini dimulai dengan mengambil bagian dasar model untuk melakukan feature extraction pada model yang sudah dilatih sebelumnya yaitu pada arsitektur VGG16 dan arsitektur ResNet50. Selain menggunakan base model, system juga menambahkan lapisan Conv2d sengan aktivasi relu, lapisan dropout sebesar 0.2, lapisan global average pooling 2d, dan lapisan dense dengan aktivasi sigmoid dan softmax, model juga menggunakan optimizer adam. Selanjutnya model akan dilatih dengan nilai epoch sebesar 50. Hasil dari pelatihan tersebut berupa nilai akurasi, kesalahan dan validasi yang ditampilkan dalam bentuk kurva. Selain ditampilkan dalam bentuk kurva, nilai evaluasi juga akan ditampilkan dalam bentuk confusion matrix dan juga menampilkan nilai presisi, recall dan f1 score.

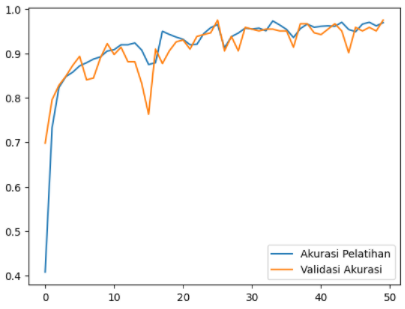
Tahap terakhir yaitu menyimpan model dan mengkonversi ke dalam bentuk .h5. Model yang telah dikonversi ke dalam .h5 kemudian akan digunakan pada aplikasi web app untuk proses deteksi.

## Hasil Pelatihan pada *Google Colaboratory*

Proses pelatihan untuk setiap sistem klasifikasi menggunakan CNN arsitektur VGG16 dan ResNet50, kemudian membandingkan hasil pelatihan dari masing – masing arsitektur. Selain arsitektur, faktor lain yang dapat mempengaruhi hasil pelatihan adalah dataset yang digunakan, yaitu dataset *original* dan *dataset preprocessed*.

### *Lung Parenchyma Classification* *ResNet50* (Original Image)

Proses pelatihan dan validasi pada klasifikasi *lung parenchyma* arsitektur *ResNet50* dengan *dataset original image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.1



Gambar 4.1Kurva akurasi pelatihan dan validasi

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis berwarna oranye yang menunjukkan nilai akurasi validasi mendekati nilai 1.0. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.9696, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.9755. Oleh karena nilai akurasi pelatihan dan nilai akurasi validasi mendekati 1.00 dan berada saling beriringan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *lung parenchyma* dengan arsitektur *ResNet50* dan dataset citra original memiliki kondisi yang baik atau disebut dengan kondisi goodfit.

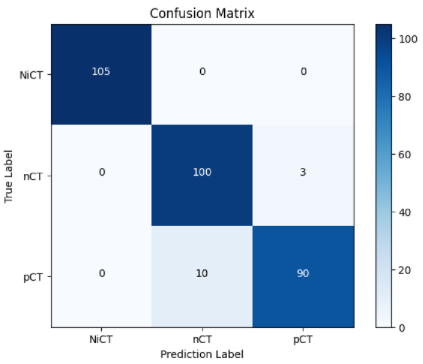
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.2



Gambar 4.2Kurva kesalahan (loss) pelatihan dan validasi LP ResNet50 original image

Kurva kesalahan (*loss*) diatas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi mendekati 0.0 seiring bertambahnya *epoch*. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.0802 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 0.0907. Oleh karena nilai kesalahan pelatihan dan nilai kesalahan validasi mendekati 0.0 secara stabil dan berada saling beriringan maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi lung parenchyma dengan arsitektur *ResNet50* dan dataset citra original memiliki kondisi yang baik atau disebut dengan kondisi *goodfit*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.3



Gambar 4.3Confusion matrix LP ResNet50 original image

Berdasarkan Gambar 4.3 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label NiCT terdapat 105 citra yang diprediksi benar untuk label NiCT, 0 citra diprediksi benar untuk label nCT, dan 0 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 105 dari 105 citra pengujian dan tingkat kesalahannya yaitu sebanyak 0 dari 105 citra pengujian. Kemudian untuk label nCT terdapat 0 citra yang diprediksi benar untuk label NiCT, 100 citra diprediksi benar untuk label nCT, dan 3 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 100 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 3 dari 105 citra pengujian. Sedangkan untuk label pCT terdapat 0 citra yang diprediksi benar untuk label NiCT, 10 citra diprediksi benar untuk label nCT, dan 90 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 90 dari 100 citra pengujian dan tingkat kesalahannya yaitu sebanyak 10 dari 100 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* 1. Akurasi
* Akurasi NiCT

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi nCT
* Akurasi pCT
* Makro akurasi
  1. Presisi
* Presisi NiCT

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi nCT
* Presisi pCT
* Makro presisi

Makro

* 1. Recall
* Recall NiCT

|  |  |
| --- | --- |
|  | (4.3) |

* Recall nCT
* Recall pCT
* Makro recall
  1. F1 score

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *lung parenchyma* dengan model ResNet50 dan citra *original* memiliki nilai mendekati 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### *Lung Parenchyma Classification ResNet50* (*Preprocessed Image)*

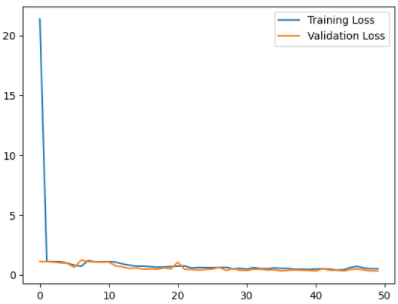
Proses pelatihan dan validasi pada klasifikasi *lung parenchyma* arsitektur *ResNet50* dengan dataset *preprocessed image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.4Kurva akurasi pelatihan dan validasi LP ResNet50 preprocessed image



Gambar 4.4Kurva akurasi pelatihan dan validasi LP ResNet50 preprocessed image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis berwarna oranye yang menunjukkan nilai akurasi validasi mendekati nilai 1.0. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.7611, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.8577. Oleh karena nilai akurasi pelatihan dan nilai akurasi validasi mendekati 1.00 meskipun tidak sebaik pada *original dataset*, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *lung parenchyma* dengan arsitektur *ResNet50* dan dataset citra *preprocessed* memiliki kondisi yang baik atau disebut dengan kondisi goodfit.

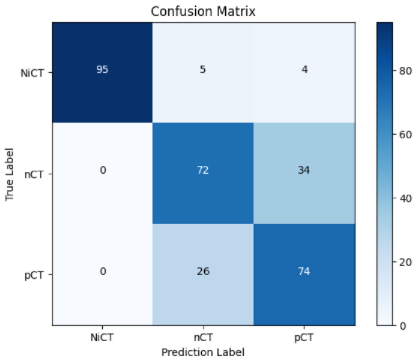
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.5



Gambar 4.5Kurva kesalahan (loss) pelatihan dan validasi LP ResNet50 preprocessed image

Kurva kesalahan (*loss*) diatas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi mendekati 0.0 seiring bertambahnya *epoch*. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.5170 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 0.3269. Oleh karena nilai kesalahan pelatihan dan nilai kesalahan validasi mendekati 0.0 secara stabil dan berada saling beriringan meskipun tidak sebaik pada pelatihan menggunakan citra original, namun dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi lung parenchyma dengan arsitektur *ResNet50* dan dataset citra *preprocessed* memiliki kondisi yang baik atau disebut dengan kondisi *goodfit*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.6



Gambar 4.6Confusion matrix LP Resnet50 preprocessed image

Berdasarkan Gambar 4.6 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label NiCT terdapat 95 citra yang diprediksi benar untuk label NiCT, 5 citra diprediksi benar untuk label nCT, dan 4 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 95 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 9 dari 104 citra pengujian. Kemudian untuk label nCT terdapat 0 citra yang diprediksi benar untuk label NiCT, 72 citra diprediksi benar untuk label nCT, dan 34 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 72 dari 106 citra pengujian dan tingkat kesalahannya yaitu sebanyak 34 dari 106 citra pengujian. Sedangkan untuk label pCT terdapat 0 citra yang diprediksi benar untuk label NiCT, 26 citra diprediksi benar untuk label nCT, dan 74 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 74 dari 100 citra pengujian dan tingkat kesalahannya yaitu sebanyak 26 dari 100 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* + 1. Akurasi
* Akurasi NiCT

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi nCT
* Akurasi pCT
* Makro akurasi
  + 1. Presisi
* Presisi NiCT

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi nCT
* Presisi pCT
* Makro presisi

Makro

* + 1. Recall
* Recall NiCT
* Recall nCT
* Recall pCT
* Makro recall

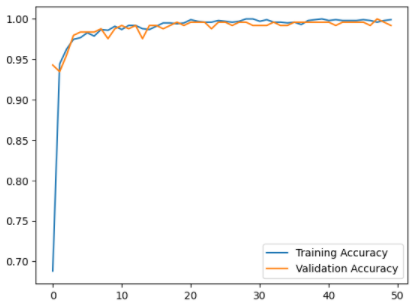
* 1. F1 score

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *lung parenchyma* dengan model ResNet50 dan citra *preprocessed* memiliki nilai mendekati 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Lung Parenchyma Classification VGG16 (Original Image)

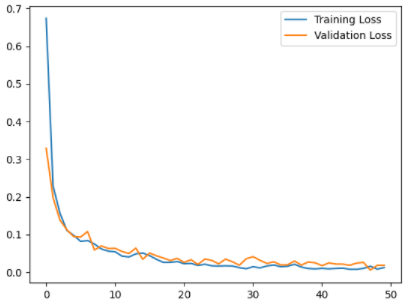
Proses pelatihan dan validasi pada klasifikasi *lung parenchyma* arsitektur VGG16 dengan dataset *original image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4..0.7



Gambar 4.7Kurva akurasi pelatihan dan validasi LP VGG16 original image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis berwarna oranye yang menunjukkan nilai akurasi validasi mendekati nilai 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.9990, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.9918. Oleh karena nilai akurasi pelatihan dan nilai akurasi validasi mendekati 1.00 dan berada saling beriringan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *lung parenchyma* dengan arsitektur *VGG16* dan dataset citra original memiliki kondisi yang baik atau disebut dengan kondisi *goodfit.*

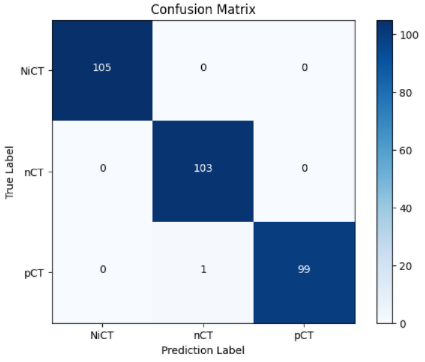
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.8



Gambar 4.8Kurva kesalahan pelatihan dan validasi LP VGG16 original image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi mendekati 0.0 seiring bertambahnya *epoch*. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.0121 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 0.0180. Oleh karena nilai kesalahan pelatihan dan nilai kesalahan validasi mendekati 0.0 secara stabil dan berada saling beriringan maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *lung parenchyma* dengan arsitektur *VGG16* dan dataset citra *original* memiliki kondisi yang baik atau disebut dengan kondisi *goodfit*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.9



Gambar 4.9Confusion Matrix LP VGG16 original image

Berdasarkan Gambar 4.9 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label NiCT terdapat 105 citra yang diprediksi benar untuk label NiCT, 0 citra diprediksi benar untuk label nCT, dan 0 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 105 dari 105 citra pengujian dan tingkat kesalahannya yaitu sebanyak 0 dari 105 citra pengujian. Kemudian untuk label nCT terdapat 0 citra yang diprediksi benar untuk label NiCT, 103 citra diprediksi benar untuk label nCT, dan 0 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 103 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 0 dari 103 citra pengujian. Sedangkan untuk label pCT terdapat 0 citra yang diprediksi benar untuk label NiCT, 1 citra diprediksi benar untuk label nCT, dan 99 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 99 dari 100 citra pengujian dan tingkat kesalahannya yaitu sebanyak 1 dari 100 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* + 1. Akurasi
* Akurasi NiCT

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi nCT
* Akurasi pCT
* Makro akurasi
  + 1. Presisi
* Presisi NiCT

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi nCT
* Presisi pCT
* Makro presisi

Makro

* + 1. Recall
* Recall NiCT

|  |  |
| --- | --- |
|  | (4.3) |

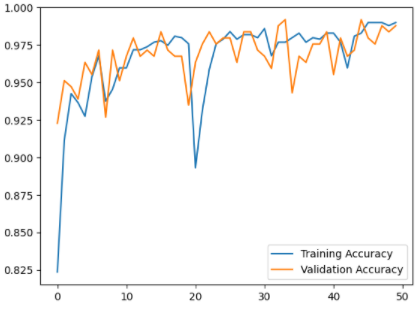
* Recall nCT
* Recall pCT
* Makro recall
  1. F1 score

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *lung parenchyma* dengan model VGG16 dan citra *original* memiliki nilai mendekati 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Lung Parenchyma Classification VGG16 (Preprocessed Image)

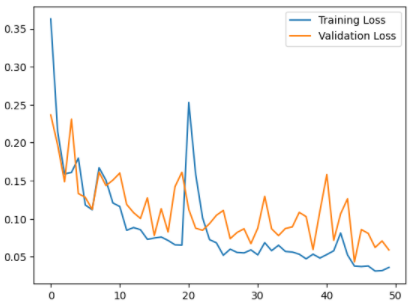
Proses pelatihan dan validasi pada klasifikasi *lung parenchyma* arsitektur VGG16 dengan *dataset* *preprocessed image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.10



Gambar 4.10Kurva akurasi pelatihan dan validasi LP VGG16 preprocessed image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis berwarna oranye yang menunjukkan nilai akurasi validasi mendekati nilai 1.0. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.9899, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.9878. Oleh karena nilai akurasi pelatihan dan nilai akurasi validasi mendekati 1.00 dan berada saling beriringan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *lung parenchyma* dengan arsitektur *VGG16* dan *dataset preprocessed image* memiliki kondisi yang baik atau disebut dengan kondisi *goodfit*.

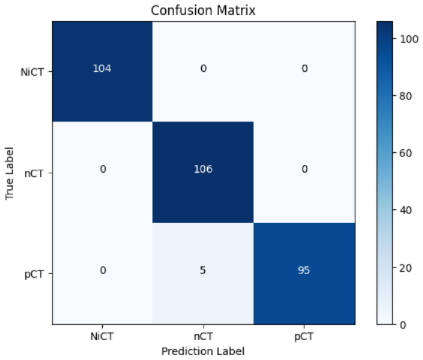
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.11



Gambar 4.11Kurva kesalahan pelatihan dan validasi LP VGG16 preprocessed image

Kurva kesalahan (*loss*) diatas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi mendekati 0.0 seiring bertambahnya *epoch*. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.0359 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 0.0588. Oleh karena nilai kesalahan pelatihan dan nilai kesalahan validasi mendekati 0.0 secara stabil dan berada saling beriringan maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi lung parenchyma dengan arsitektur *VGG16* dan dataset *preprocessed image* memiliki kondisi yang baik atau disebut dengan kondisi *goodfit*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.12



Gambar 4.12 Confusion matrix LP VGG16 preprocessed image

Berdasarkan Gambar 4.12 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label NiCT terdapat 104 citra yang diprediksi benar untuk label NiCT, 0 citra diprediksi benar untuk label nCT, dan 0 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 104 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 0 dari 104 citra pengujian. Kemudian untuk label nCT terdapat 0 citra yang diprediksi benar untuk label NiCT, 106 citra diprediksi benar untuk label nCT, dan 0 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 106 dari 106 citra pengujian dan tingkat kesalahannya yaitu sebanyak 0 dari 106 citra pengujian. Sedangkan untuk label pCT terdapat 0 citra yang diprediksi benar untuk label NiCT, 5 citra diprediksi benar untuk label nCT, dan 95 citra diprediksi benar untuk label pCT. Artinya tingkat kebenaran pada label NiCT sebesar 95 dari 100 citra pengujian dan tingkat kesalahannya yaitu sebanyak 5 dari 100 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* + 1. Akurasi
* Akurasi NiCT

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi nCT
* Akurasi pCT
* Makro akurasi
  + 1. Presisi
* Presisi NiCT

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi nCT
* Presisi pCT
* Makro presisi

Makro

* + 1. *Recall*
* *Recall* NiCT

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* nCT
* *Recall* pCT
* Makro *recall*

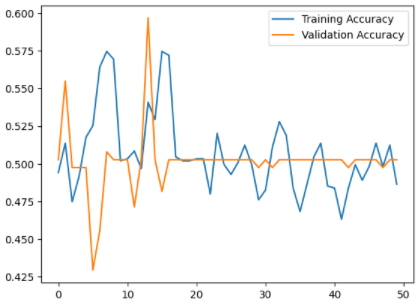
* 1. *F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *lung parenchyma* dengan model VGG16 dan citra *preprocessed* memiliki nilai mendekati 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Covid Positivity Classification ResNet50 (Original Image)

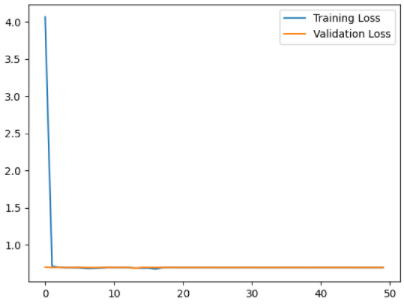
Proses pelatihan dan validasi pada klasifikasi *covid positivity* arsitektur ResNet50 dengan *dataset original image* menggunakan nilai epoch sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.13



Gambar 4.13Kurva akurasi pelatihan dan validasi CP ResNet50 original image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis berwarna oranye yang menunjukkan nilai akurasi validasi masih jauh dari nilai 1.0. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.4864, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.5026. Oleh karena nilai akurasi pelatihan dan nilai akurasi validasi masih jauh dari angka 1.00, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *covid positivity* dengan arsitektur *ResNet50* dan *dataset original image* memiliki kondisi yang kurang baik.

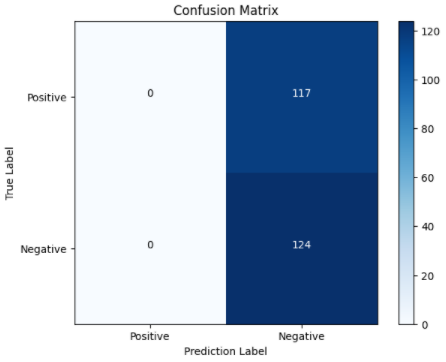
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.14



Gambar 4.14Kurva akurasi pelatihan dan validasi CP ResNet50 preprocessed image

Kurva kesalahan (*loss*) diatas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi masih jauh di atas nilai 0.0. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.6934 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 0.6931. Oleh karena nilai kesalahan pelatihan dan nilai kesalahan validasi masih jauh dari angka 0.0, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *risk* dengan arsitektur *ResNet50* yang menggunakan *dataset original image* memiliki kondisi yang kurang baik.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.15



Gambar 4.15Confusion matrix CP ResNet50 preprocessed image

Berdasarkan Gambar 4.15 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label *Positive* terdapat 0 citra yang diprediksi benar untuk label *Positive*, 117 citra diprediksi benar untuk label *Negative*. Artinya tingkat kebenaran pada label *Positive* sebesar 0 dari 117 citra pengujian dan tingkat kesalahannya yaitu sebanyak 117 dari 117 citra pengujian. Kemudian untuk label *Negative* terdapat 0 citra yang diprediksi benar untuk label *Positive*, 124 citra diprediksi benar untuk label *Negative*. Artinya tingkat kebenaran pada label *Negative* sebesar 124 dari 124 citra pengujian dan tingkat kesalahannya yaitu sebanyak 0 dari 124 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* + 1. Akurasi

|  |  |
| --- | --- |
|  | (4.1) |

* + 1. Presisi

|  |  |
| --- | --- |
|  | (4.2) |

* + 1. *Recall*

|  |  |
| --- | --- |
|  | (4.3) |

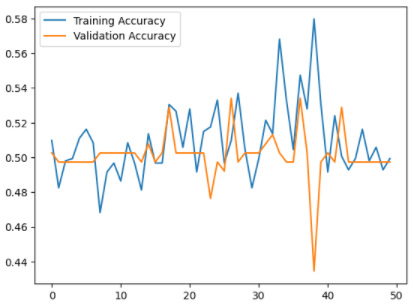
* 1. *F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *covid positivity* dengan model ResNet50 dan citra *original* memiliki nilai masih jauh dari nilai 1.00, sehingga model belum mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Covid Positivity Classification ResNet50 (Preprocessed Image)

Proses pelatihan dan validasi pada klasifikasi *covid positivity* arsitektur ResNet50 dengan *dataset preprocessed image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.16



Gambar 4.16 Kurva akurasi pelatihan dan validasi CP ResNet50

preprocessed image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis berwarna oranye yang menunjukkan nilai akurasi validasi masih jauh dari nilai 1.0. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.4994, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.4974. Oleh karena nilai akurasi pelatihan dan nilai akurasi validasi masih jauh dari angka 1.00, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *covid positivity* dengan arsitektur *ResNet50* dan *dataset preprocessed image* memiliki kondisi yang kurang baik.

Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.17

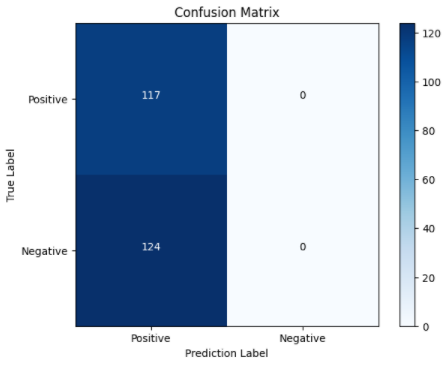


Gambar 4.17 Kurva kesalahan pelatihan dan validasi CP ResNet50

preprocessed image

Kurva kesalahan (*loss*) diatas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi masih jauh (garis oranye) di atas nilai 0.0. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.6926 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 0.6932. Oleh karena nilai kesalahan pelatihan dan nilai kesalahan validasi masih jauh dari angka 0.0, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *risk* dengan arsitektur *ResNet50* yang menggunakan *dataset preprocessed image* memiliki kondisi yang kurang baik.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.18



Gambar 4.18 Confusion matrix CP ResNet50 preprocessed image

Berdasarkan Gambar 4.18 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label *Positive* terdapat 117 citra yang diprediksi benar untuk label *Positive*, 0 citra diprediksi benar untuk label *Negative*. Artinya tingkat kebenaran pada label *Positive* sebesar 117 dari 117 citra pengujian dan tingkat kesalahannya yaitu sebanyak 0 dari 117 citra pengujian. Kemudian untuk label *Negative* terdapat 124 citra yang diprediksi benar untuk label *Positive*, 0 citra diprediksi benar untuk label *Negative*. Artinya tingkat kebenaran pada label *Negative* sebesar 0 dari 124 citra pengujian dan tingkat kesalahannya yaitu sebanyak 124 dari 124 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* + 1. Akurasi

|  |  |
| --- | --- |
|  | (4.1) |

* + 1. Presisi

|  |  |
| --- | --- |
|  | (4.2) |

* + 1. Recall

|  |  |
| --- | --- |
|  | (4.3) |

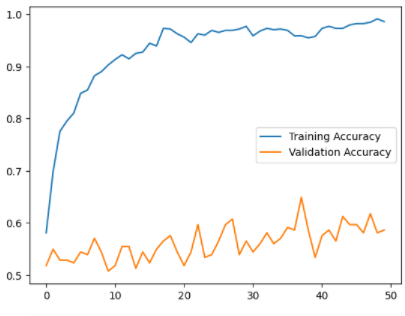
* + 1. F1 score

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *covid positivity* dengan model ResNet50 dan citra *preprocessed* memiliki nilai masih jauh dari nilai 1.00, sehingga model belum mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Covid Positivity Classification VGG16 (Original Image)

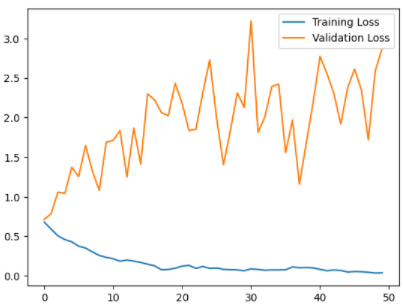
Proses pelatihan dan validasi pada klasifikasi *covid positivity* arsitektur VGG16 dengan dataset original image menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.19



Gambar 4.19 Kurva akurasi pelatihan dan validasi CP VGG16 original image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan mendekati angka 1.00. Namun, garis berwarna oranye yang menunjukkan nilai akurasi validasi sangat jauh dari nilai 1.0. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.9857, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.5864. Oleh karena nilai akurasi validasi masih jauh dari angka 1.00 dan berada jauh di bawah nilai validasi pelatihan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *covid positivity* dengan arsitektur *VGG16* dan *dataset original image* memiliki kondisi yang kurang baik atau disebut dalam kondisi *overfitting*.

Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.20

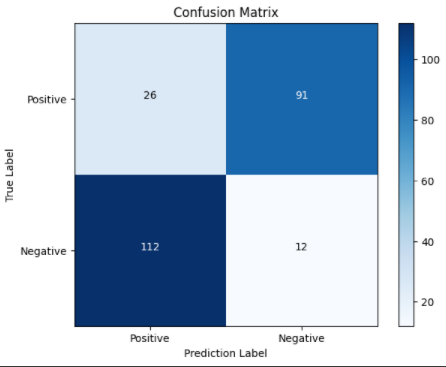


Gambar 4.20 Kurva kesalahan pelatihan dan validasi CP VGG16

original image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) mendekati angka 1.00. Namun, nilai kesalahan validasi masih jauh di atas nilai 0.0. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.0373 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 2.8761. Oleh karena nilai kesalahan validasi masih jauh dari angka 0.0 dan berada pada interval yang cukup jauh di atas nilai kesalahan pelatihan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *covid positivity* dengan arsitektur *VGG16* yang menggunakan *dataset original image* memiliki kondisi yang kurang baik atau dalam kondisi *overfitting*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.21



Gambar 4.21 Confusion matrix CP VGG16 original image

Berdasarkan Gambar 4.21 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label *Positive* terdapat 26 citra yang diprediksi benar untuk label *Positive*, 91 citra diprediksi benar untuk label *Negative*. Artinya tingkat kebenaran pada label *Positive* sebesar 26 dari 117 citra pengujian dan tingkat kesalahannya yaitu sebanyak 91 dari 117 citra pengujian. Kemudian untuk label *Negative* terdapat 112 citra yang diprediksi benar untuk label *Positive*, 12 citra diprediksi benar untuk label *Negative*. Artinya tingkat kebenaran pada label *Negative* sebesar 12 dari 124 citra pengujian dan tingkat kesalahannya yaitu sebanyak 112 dari 124 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* + - 1. Akurasi

|  |  |
| --- | --- |
|  | (4.1) |

* + - 1. Presisi

|  |  |
| --- | --- |
|  | (4.2) |

* + - 1. *Recall*

|  |  |
| --- | --- |
|  | (4.3) |

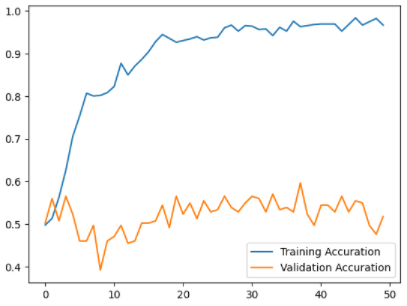
* + - 1. F1 score

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *covid positivity* dengan model VGG16 dan citra *original* memiliki nilai masih jauh dari nilai 1.00, sehingga model belum mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Covid Positivity Classification VGG16 (Preprocessed Image)

Proses pelatihan dan validasi pada klasifikasi *covid positivity* arsitektur VGG16 dengan *dataset preprocessed image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.22

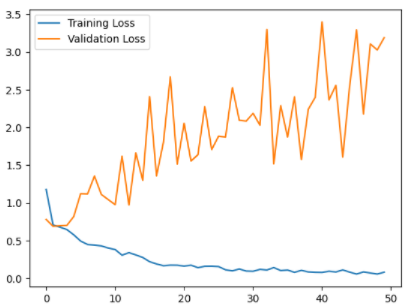


Gambar 4.22 Kurva akurasi pelatihan dan validasi CP VGG16

preprocessed image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan mendekati angka 1.00. Namun, garis berwarna oranye yang menunjukkan nilai akurasi validasi sangat jauh dari nilai 1.0. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.9663, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.5183. Oleh karena nilai akurasi validasi masih jauh dari angka 1.00 dan berada jauh di bawah nilai validasi pelatihan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *covid positivity* dengan arsitektur *VGG16* dan *dataset preprocessed image* memiliki kondisi yang kurang baik atau disebut dalam kondisi *overfitting*.

Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.23

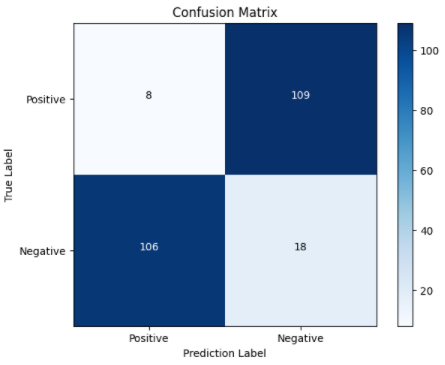


Gambar 4.23 Kurva kesalahan pelatihan dan validasi CP VGG16

preprocessed image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) mendekati angka 1.00. Namun, nilai kesalahan validasi masih jauh di atas nilai 0.0. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.0804 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 3.1888. Oleh karena nilai kesalahan validasi masih jauh dari angka 0.0 dan berada pada interval yang cukup jauh di atas nilai kesalahan pelatihan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *covid positivity* dengan arsitektur *VGG16* yang menggunakan *dataset preprocessed image* memiliki kondisi yang kurang baik atau dalam kondisi *overfitting*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.24



Gambar 4.24 Confusion matrix CP VGG16 preprocessed image

Berdasarkan Gambar 4.24 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label *Positive* terdapat 8 citra yang diprediksi benar untuk label *Positive*, 109 citra diprediksi benar untuk label *Negative*. Artinya tingkat kebenaran pada label *Positive* sebesar 8 dari 117 citra pengujian dan tingkat kesalahannya yaitu sebanyak 109 dari 117 citra pengujian. Kemudian untuk label *Negative* terdapat 106 citra yang diprediksi benar untuk label *Positive*, 18 citra diprediksi benar untuk label *Negative*. Artinya tingkat kebenaran pada label *Negative* sebesar 106 dari 124 citra pengujian dan tingkat kesalahannya yaitu sebanyak 18 dari 124 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* + - * 1. Akurasi

|  |  |
| --- | --- |
|  | (4.1) |

* + - * 1. Presisi

|  |  |
| --- | --- |
|  | (4.2) |

* + - * 1. *Recall*

|  |  |
| --- | --- |
|  | (4.3) |

* + - * 1. *F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *covid positivity* dengan model VGG16 dan citra *preprocessed* memiliki nilai masih jauh dari nilai 1.00, sehingga model belum mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Risk Classification ResNet50 (Original Image)

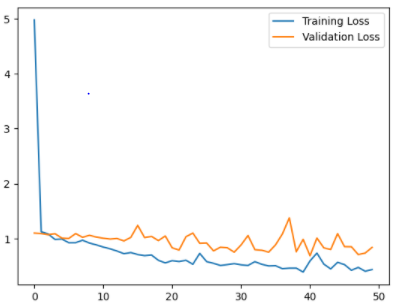
Proses pelatihan dan validasi pada klasifikasi *risk* arsitektur ResNet50 dengan dataset *original image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.25



Gambar 4.25 Kurva akurasi pelatihan dan validasi risk ResNet50 original image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan mendekati nilai 1.00. Namun untuk garis oranye yang menunjukkan akurasi validasi masih jauh dari angka 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.8199, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.6680. Oleh karena nilai akurasi validasi masih jauh dari angka 1.00 dan berada berada pada interval yang cukup jauh di bawah nilai akurasi pelatihan , maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *risk* dengan arsitektur *ResNet50* dan *dataset original image* memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

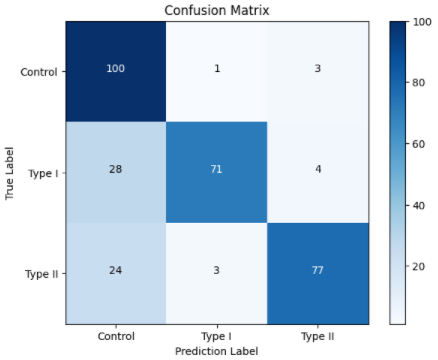
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.26



Gambar 4.26 Kurva akurasi pelatihan dan validasi risk ResNet50 original image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) mendekati angka 0.00. Namun, untuk nilai kesalahan validasi masih jauh dari nilai 0.00. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.4358, dan nilai kesalahan validasi yang diperoleh pada *epoch* ke-50 sebesar 0.8384. Oleh karena nilai kesalahan validasi berada cukup jauh di atas nilai kesalahan pelatihan maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *risk* dengan arsitektur *ResNet50* yang menggunakan dataset citra *original*  memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.27



Gambar 4.27 Confusion matrix risk VGG16 original image

Berdasarkan Gambar 4.27 dapat diketahui garis y menunjukkan label sebenarnya, sedangkan garis x menunjukkan label prediksi. Pada label *Control* terdapat 100 citra yang diprediksi benar untuk label *Control*, 1 citra diprediksi benar untuk label *Type I*, dan 3 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label *Control* sebesar 100 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 4 dari 104 citra pengujian. Kemudian untuk label *Type I* terdapat 28 citra yang diprediksi benar untuk label *Control*, 71 citra diprediksi benar untuk label *Type I*, dan 4 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label *Type I* sebesar 71 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 32 dari 103 citra pengujian. Sedangkan untuk label *Type II* terdapat 24 citra yang diprediksi benar untuk label *Control*, 3 citra diprediksi benar untuk label *Type I*, dan 77 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label Type I sebesar 77 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 27 dari 104 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

Akurasi

* Akurasi *Control*

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi Type I
* Akurasi *Type II*
* Makro akurasi

Presisi

* Presisi *Control*

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi *Type I*
* Presisi *Type II*
* Makro presisi

Makro

*Recall*

* *Recall* *Control*

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* *Type I*
* *Recall* *Type II*
* Makro *recall*

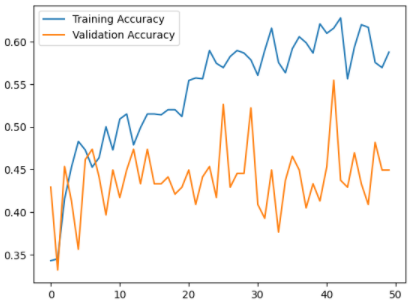
*F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *risk* dengan model ResNet50 dan citra *original* memiliki nilai m mendekati nilai 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Risk Classification ResNet50 (Preprocessed Image)

Proses pelatihan dan validasi pada klasifikasi *risk* arsitektur ResNet50 dengan dataset *preprocessed image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.28

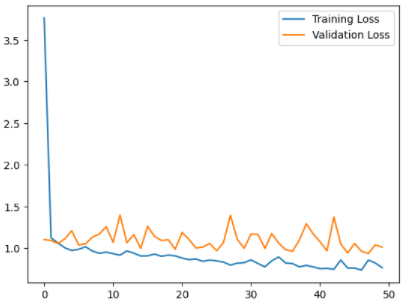


Gambar 4.28 Kurva akurasi pelatihan dan validasi risk ResNet50

preprocessed image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis oranye yang menunjukkan akurasi validasi masih jauh dari angka 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.5875, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.4494. Oleh karena nilai akurasi pelatihan dan validasi masih jauh dari angka 1.00, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *risk* dengan arsitektur *ResNet50* dan *dataset preprocessed image* memiliki kondisi yang kurang baik.

Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.29

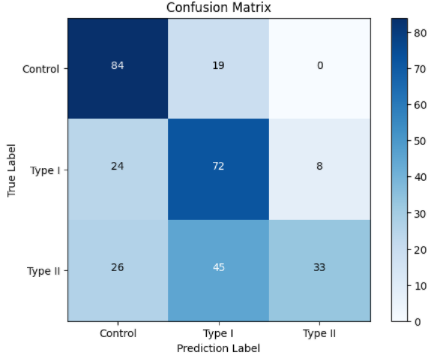


Gambar 4.29 Kurva kesalahan pelatihan dan validasi risk ResNet50

preprocessed image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi masih jauh dari nilai 0.00. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.7648, dan nilai kesalahan validasi yang diperoleh pada *epoch* ke-50 sebesar 1.0121. Oleh karena nilai kesalahan pelatihan dan kesalahan validasi berada cukup jauh di atas nilai 0.00, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *risk* dengan arsitektur *ResNet50* yang menggunakan dataset citra *preprocessed*  memiliki kondisi yang kurang baik.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.30

**

Gambar 4.30 Confusion matrix risk ResNet50 preprocessed

Berdasarkan Gambar 4.30 dapat diketahui garis y menunjukkan label sebenarnya, sedangkan garis x menunjukkan label prediksi. Pada label *Control* terdapat 84 citra yang diprediksi benar untuk label *Control*, 19 citra diprediksi benar untuk label *Type I*, dan 0 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label *Control* sebesar 84 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 19 dari 104 citra pengujian. Kemudian untuk label *Type I* terdapat 24 citra yang diprediksi benar untuk label *Control*, 72 citra diprediksi benar untuk label *Type I*, dan 8 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label *Type I* sebesar 72 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 32 dari 104 citra pengujian. Sedangkan untuk label *Type II* terdapat 26 citra yang diprediksi benar untuk label *Control*, 45 citra diprediksi benar untuk label *Type I*, dan 33 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label Type I sebesar 33 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 71 dari 104 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

Akurasi

* Akurasi *Control*

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi Type I
* Akurasi *Type II*
* Makro akurasi

Presisi

* Presisi *Control*

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi *Type I*
* Presisi *Type II*
* Makro presisi

Makro

*Recall*

* *Recall* *Control*

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* *Type I*
* *Recall* *Type II*
* Makro *recall*

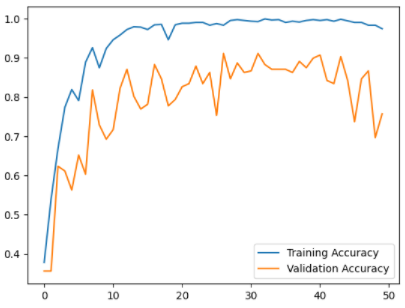
*F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *risk* dengan model ResNet50 dan citra *preprocessed* memiliki nilai mendekati nilai 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Risk Classification VGG16 (Original Image)

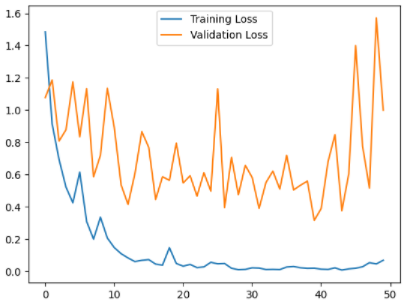
Proses pelatihan dan validasi pada klasifikasi *risk* arsitektur VGG16 dengan dataset *original image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.31



Gambar 4.31 Kurva akurasi pelatihan dan validasi risk VGG16 original image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis oranye yang menunjukkan akurasi validasi mendekati nilai 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.9960, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.8704. Oleh karena nilai akurasi pelatihan dan akurasi validasi mendekati angka 1.00 dan keduanya berada saling beriringan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *risk* dengan arsitektur *VGG16* menggunakan *dataset original image* memiliki kondisi yang baik atau disebut dengan kondisi *goodfit*.

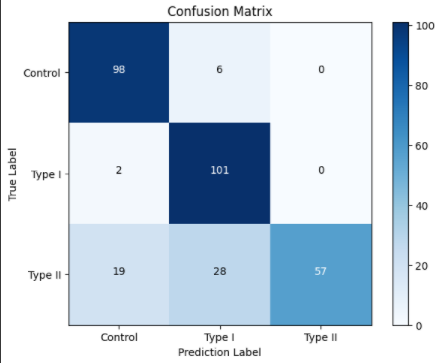
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.32



Gambar 4.32Kurva kesalahan pelatihan dan validassi risk VGG16 original image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) mendekati angka 0.00. Namun, untuk nilai kesalahan validasi masih jauh dari nilai 0.00. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.0122, dan nilai kesalahan validasi yang diperoleh pada *epoch* ke-50 sebesar 0.6217. Oleh karena nilai kesalahan validasi berada cukup jauh di atas nilai kesalahan pelatihan maka dapat disimpulkan bahwa kesalahan pelatihan dan kesalahan validasi pada klasifikasi *risk* dengan arsitektur *VGG16* yang menggunakan dataset citra *original*  memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.33



Gambar 4.33Confusion matrix risk VGG16 original image

Berdasarkan Gambar 4.33 dapat diketahui garis y menunjukkan label sebenarnya, sedangkan garis x menunjukkan label prediksi. Pada label *Control* terdapat 98 citra yang diprediksi benar untuk label *Control*, 6 citra diprediksi benar untuk label *Type I*, dan 0 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label *Control* sebesar 98 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 6 dari 104 citra pengujian. Kemudian untuk label *Type I* terdapat 2 citra yang diprediksi benar untuk label *Control*, 101 citra diprediksi benar untuk label *Type I*, dan 0 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label *Type I* sebesar 101 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 2 dari 103 citra pengujian. Sedangkan untuk label *Type II* terdapat 19 citra yang diprediksi benar untuk label *Control*, 28 citra diprediksi benar untuk label *Type I*, dan 57 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label Type I sebesar 57 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 47 dari 104 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

Akurasi

* Akurasi *Control*

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi Type I
* Akurasi *Type II*
* Makro akurasi

Presisi

* Presisi *Control*

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi *Type I*
* Presisi *Type II*
* Makro presisi

Makro

*Recall*

* *Recall* *Control*

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* *Type I*
* *Recall* *Type II*
* Makro *recall*

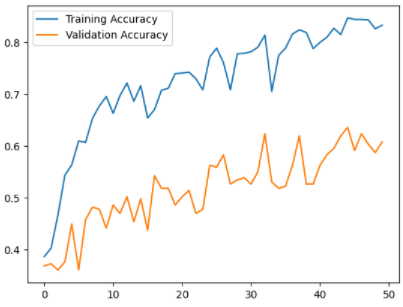
*F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *risk* dengan model VGG16 dan citra *original* memiliki nilai mendekati nilai 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Risk Classification VGG16 (Preprocessed Image)

Proses pelatihan dan validasi pada klasifikasi *risk* arsitektur VGG16 dengan dataset *preprocessed image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.34

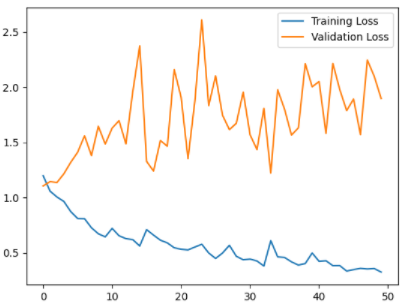


Gambar 4.34 Kurva akurasi pelatihan dan validasi risk VGG16

preprocessed image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan mendekati angka 1.00. Namun garis oranye yang menunjukkan akurasi validasi masih jauh di bawah nilai 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.8330, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.6073. Oleh karena nilai akurasi validasi masih jauh dari angka 1.00 dan bjauh di abwah akurasi pelatihan, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *risk* dengan arsitektur *VGG16* menggunakan *dataset preprocessed image* memiliki kondisi yang kurang baik atau disebut dengan kondisi over*fitting*.

Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat padaGambar 4.35

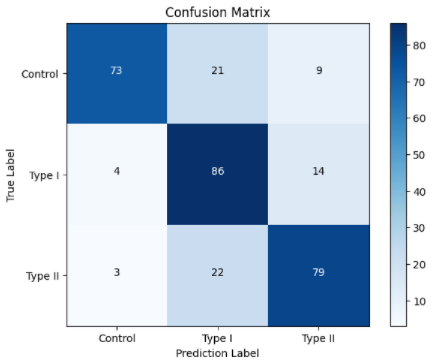


Gambar 4.35Kurva kesalahan pelatihan dan validasi risk VGG16

preprocessed image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) mendekati angka 0.00. Namun, untuk nilai kesalahan validasi sangat jauh dari nilai 0.00. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.3239, dan nilai kesalahan validasi yang diperoleh pada *epoch* ke-50 sebesar 1.8966. Oleh karena nilai kesalahan validasi berada sangat jauh di atas nilai kesalahan pelatihan maka dapat disimpulkan bahwa kesalahan pelatihan dan kesalahan validasi pada klasifikasi *risk* dengan arsitektur *VGG16* yang menggunakan dataset citra *preprocessed*  memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.36



Gambar 4.36 Confusion matrix risk VGG16 preprocessed image

Berdasarkan Gambar 4.36 dapat diketahui garis y menunjukkan label sebenarnya, sedangkan garis x menunjukkan label prediksi. Pada label *Control* terdapat 73 citra yang diprediksi benar untuk label *Control*, 21 citra diprediksi benar untuk label *Type I*, dan 9 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label *Control* sebesar 73 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 30 dari 103 citra pengujian. Kemudian untuk label *Type I* terdapat 4 citra yang diprediksi benar untuk label *Control*, 86 citra diprediksi benar untuk label *Type I*, dan 14 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label *Type I* sebesar 86 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 18 dari 104 citra pengujian. Sedangkan untuk label *Type II* terdapat 3 citra yang diprediksi benar untuk label *Control*, 22 citra diprediksi benar untuk label *Type I*, dan 79 citra diprediksi benar untuk label *Type II*. Artinya tingkat kebenaran pada label Type I sebesar 79 dari 104 citra pengujian dan tingkat kesalahannya yaitu sebanyak 25 dari 104 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

Akurasi

* Akurasi *Control*

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi Type I
* Akurasi *Type II*
* Makro akurasi

Presisi

* Presisi *Control*

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi *Type I*
* Presisi *Type II*
* Makro presisi

Recall

* *Recall* *Control*

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* *Type I*
* *Recall* *Type II*
* Makro *recall*

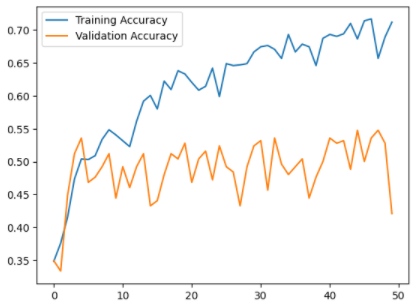
F1 score

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *risk* dengan model VGG16 dan citra *preprocessed* memiliki nilai mendekati nilai 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Mortality Classification ResNet50 (Original Image)

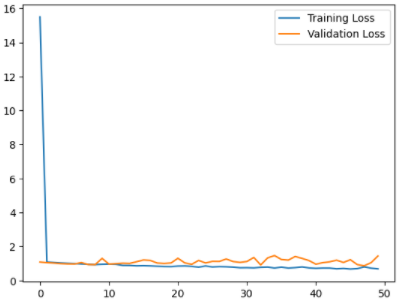
Proses pelatihan dan validasi pada klasifikasi *morbidity* arsitektur ResNet50 dengan dataset original image menggunakan nilai epoch sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.37



Gambar 4.37 Kurva akurasi pelatihan dan validasi mortality ResNet50 original image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan sedikit mendekati nilai 1.0. Namun untuk garis oranye yang menunjukkan akurasi validasi masih cukup jauh dari angka 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.7120, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.4206. Oleh karena nilai akurasi validasi masih jauh dari angka 1.00 dan berada berada pada interval yang cukup jauh di bawah nilai akurasi pelatihan , maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *mortality* dengan arsitektur *ResNet50* dan *dataset original image* memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

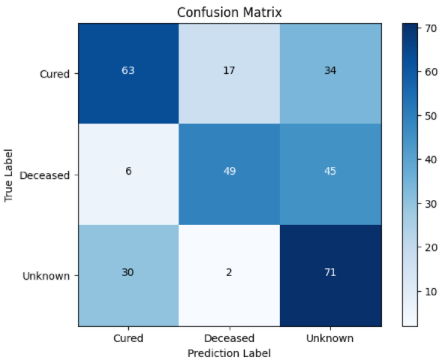
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.38



Gambar 4.38 Kurva kesalahan pelatihan dan validasi mortality ResNet50 original image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi masih jaug dari nilai 0.00. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.6858 dan nilai kesalahan akurasi yang diperoleh pada *epoch* ke-50 sebesar 1.4450. Oleh karena nilai kesalahan pelatihan dan nilai kesalahan validasi masih jauh dari angka 0.00 dan keduanya berada pada interval yang cukup jauh maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *mortality* dengan arsitektur *ResNet50* dan dataset citra *original* memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.0.39



Gambar 4.39 Confusion matrix mortality ResNet50 preprocessed image

Berdasarkan Gambar 4.39 dapat diketahui garis y menunjukkan label sebenarnya sedangkan garis x menunjukkan label prediksi. Pada label *Cured* terdapat 63 citra yang diprediksi benar untuk label *Cured*, 17 citra diprediksi benar untuk label *Deceased*, dan 34 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Cured* sebesar 63 dari 114 citra pengujian dan tingkat kesalahannya yaitu sebanyak 51 dari 114 citra pengujian. Kemudian untuk label *Deceased* terdapat 6 citra yang diprediksi benar untuk label *Cured*, 49 citra diprediksi benar untuk label *Deceased*, dan 45 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Deceased* sebesar 49 dari 100 citra pengujian dan tingkat kesalahannya yaitu sebanyak 51 dari 100 citra pengujian. Sedangkan untuk label Unknown terdapat 30 citra yang diprediksi benar untuk label *Cured*, 2 citra diprediksi benar untuk label *Deceased*, dan 71 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Unknown* sebesar 71 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 32 dari 103 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

* + 1. Akurasi
* Akurasi *Cured*

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi *Deceased*
* Akurasi *Unknown*
* Makro akurasi
  + 1. Presisi
* Presisi *Cured*

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi *Deceased*
* Presisi *Unknown*
* Makro presisi

Makro

* + 1. *Recall*
* *Recall* *Cured*

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* Deceased
* *Recall* *Unknown*
* Makro *recall*

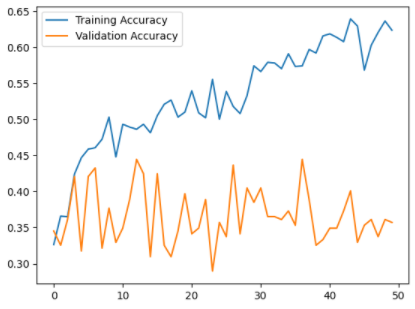
* + 1. *F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *mortality* dengan model ResNet50 dan citra *original* memiliki nilai mendekati nilai 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Mortality Classification ResNet50 (Preprocessed Image)

Proses pelatihan dan validasi pada klasifikasi *mortality* arsitektur ResNet50 dengan *dataset preprocessed image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.40



Gambar 4.40 Kurva akurasi pelatihan dan validasi mortality ResNet50

preprocessed image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan dan garis oranye yang menunjukkan akurasi validasi masih cukup jauh dari angka 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.6233, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.3571. Oleh karena nilai akurasi pelatihan dan nilai akurasi validasi masih jauh dari angka 1.00 dan keduanya berada pada interval yang cukup jauh, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *mortality* dengan arsitektur *ResNet50* dan *dataset preprocessed image* memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

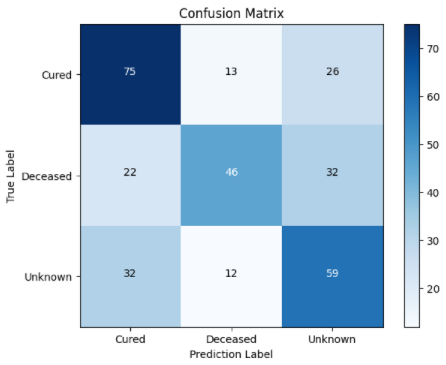
Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.41



Gambar 4.41 Kurva kesalahan pelatihan dan validasi mortality ResNet50 preprocessed image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) dan nilai kesalahan validasi masih jauh dari nilai 0.00. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.0852 dan nilai kesalahan validasi yang diperoleh pada *epoch* ke-50 sebesar 1.2999. Oleh karena nilai kesalahan validasi dan nilai kesalahan pelatihan masih jauh dari nilai 0.00, maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *mortality* dengan arsitektur *ResNet50* yang menggunakan dataset citra *preprocessed* memiliki kondisi yang kurang baik.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.42



Gambar 4.42 Confusion matrix mortality ResNet50 preprocessed image

Berdasarkan Gambar 4.42 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label *Cured* terdapat 75 citra yang diprediksi benar untuk label *Cured*, 13 citra diprediksi benar untuk label *Deceased*, dan 26 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Cured* sebesar 75 dari 114 citra pengujian dan tingkat kesalahannya yaitu sebanyak 39 dari 114 citra pengujian. Kemudian untuk label *Deceased* terdapat 22 citra yang diprediksi benar untuk label *Cured*, 46 citra diprediksi benar untuk label *Deceased*, dan 32 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Deceased* sebesar 46 dari 100 citra pengujian dan tingkat kesalahannya yaitu sebanyak 54 dari 100 citra pengujian. Sedangkan untuk label Unknown terdapat 32 citra yang diprediksi benar untuk label *Cured*, 12 citra diprediksi benar untuk label *Deceased*, dan 59 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Unknown* sebesar 59 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 44 dari 103 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

Akurasi

* Akurasi *Cured*

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi *Deceased*
* Akurasi *Unknown*
* Makro akurasi

Presisi

* Presisi *Cured*

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi *Deceased*
* Presisi *Unknown*
* Makro presisi

Makro

*Recall*

* *Recall* *Cured*

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* Deceased
* *Recall Unknown*
* Makro *recall*

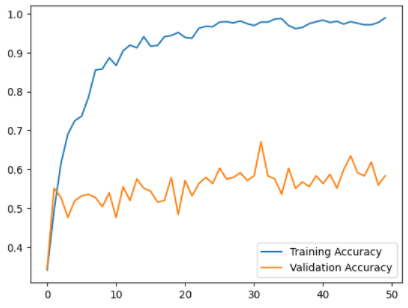
*F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *mortality* dengan model ResNet50 dan citra *preprocessed* memiliki nilai masih jauh dari nilai 1.00, sehingga model belum mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Mortality Classification VGG16 (Original Image)

Proses pelatihan dan validasi pada klasifikasi *mortality* arsitektur VGG16 dengan *dataset original image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.43



Gambar 4.43 Kurva akurasi pelatihan dan validasi mortality VGG16

original image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan mendekati nilai 1.00. Namun untuk garis oranye yang menunjukkan akurasi validasi masih cukup jauh dari angka 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.9892, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.5833. Oleh karena nilai akurasi validasi masih jauh dari angka 1.00 dan berada berada pada interval yang cukup jauh di bawah nilai akurasi pelatihan , maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *mortality* dengan arsitektur *VGG16* dan *dataset original image* memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfit*.

Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.44

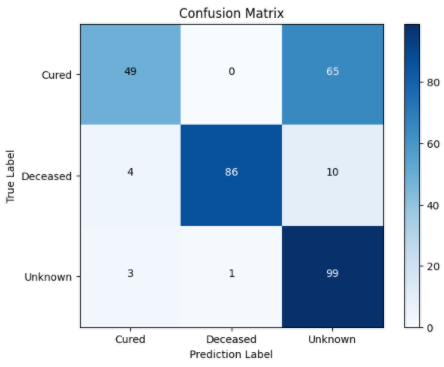


Gambar 4.44 Kurva kesalahan pelatihan dan validasi mortality VGG16

original image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) mendekati angka 0.00. Namun, untuk nilai kesalahan validasi masih jauh dari nilai 0.00. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.0312 dan nilai kesalahan validasi yang diperoleh pada *epoch* ke-50 sebesar 0.5833. Oleh karena nilai kesalahan validasi berada jauh di atas nilai kesalahan pelatihan maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *mortality* dengan arsitektur *VGG16* yang menggunakan dataset citra *original* memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.45



Gambar 4.45 Confusion matrix mortality VGG16 original image

Berdasarkan Gambar 4.45 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label *Cured* terdapat 43 citra yang diprediksi benar untuk label *Cured*, 0 citra diprediksi benar untuk label *Deceased*, dan 65 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Cured* sebesar 49 dari 114 citra pengujian dan tingkat kesalahannya yaitu sebanyak 63 dari 114 citra pengujian. Kemudian untuk label *Deceased* terdapat 4 citra yang diprediksi benar untuk label *Cured*, 86 citra diprediksi benar untuk label *Deceased*, dan 10 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Deceased* sebesar 86 dari 100 citra pengujian dan tingkat kesalahannya yaitu sebanyak 14 dari 100 citra pengujian. Sedangkan untuk label Unknown terdapat 3 citra yang diprediksi benar untuk label *Cured*, 1 citra diprediksi benar untuk label *Deceased*, dan 99 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Unknown* sebesar 99 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 4 dari 103 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

Akurasi

* Akurasi *Cured*

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi *Deceased*
* Akurasi *Unknown*
* Makro akurasi

Presisi

* Presisi *Cured*

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi *Deceased*
* Presisi *Unknown*
* Makro presisi

Makro

*Recall*

* *Recall Cured*

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* Deceased
* *Recall Unknown*
* Makro *recall*

*F1 score*

|  |  |
| --- | --- |
|  | (4.4) |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *mortality* dengan model VGG16 dan citra *original* memiliki nilai mendekati nilai 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

### Mortality Classification VGG16 (Preprocessed Image)

Proses pelatihan dan validasi pada klasifikasi *mortality* arsitektur VGG16 dengan *dataset preprocessed image* menggunakan nilai *epoch* sebesar 50. Hasil pelatihan berupa akurasi pelatihan dan akurasi validasi seperti ditampilkan pada Gambar 4.46



Gambar 4.46 Kurva akurasi pelatihan dan validasi mortality VGG16

preprocessed image

Berdasarkan kurva akurasi pelatihan di atas menunjukkan bahwa pada *epoch* ke-50, garis berwarna biru yang menunjukkan nilai akurasi pelatihan mendekati nilai 1.00. Namun untuk garis oranye yang menunjukkan akurasi validasi masih cukup jauh dari angka 1.00. Nilai akurasi pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.9527, sedangkan nilai akurasi validasi yang diperoleh pada *epoch* ke-50 sebesar 0.6071. Oleh karena nilai akurasi validasi masih jauh dari angka 1.00 dan berada berada pada interval yang cukup jauh di bawah nilai akurasi pelatihan , maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *mortality* dengan arsitektur *VGG16* dan *dataset preprocessed image* memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

Kemudian untuk kurva kesalahan (*loss*) pelatihan dan kesalahan (*loss*) validasi dapat dilihat pada Gambar 4.47



Gambar 4.47 Kurva kesalahan pelatihan dan validasi mortality VGG16

preprocessed image

Kurva kesalahan (*loss*) di atas menunjukkan bahwa nilai kesalahan pelatihan (garis biru) mendekati angka 0.00. Namun, untuk nilai kesalahan validasi masih jauh dari nilai 0.00. Nilai kesalahan pelatihan yang diperoleh pada *epoch* ke-50 sebesar 0.1520 dan nilai kesalahan validasi yang diperoleh pada *epoch* ke-50 sebesar 1.6519. Oleh karena nilai kesalahan validasi berada jauh di atas nilai kesalahan pelatihan maka dapat disimpulkan bahwa akurasi pelatihan dan akurasi validasi pada klasifikasi *mortality* dengan arsitektur *VGG16* yang menggunakan dataset citra *preprocessed* memiliki kondisi yang kurang baik atau disebut dengan kondisi *overfitting*.

Adapun untuk hasil prediksi dengan *confusion matrix* dapat dilihat pada Gambar 4.48



Gambar 4.48 Confusion matrix mortality VGG16 preprocessed image

Berdasarkan Gambar 4.48 dapat diketahui garis y menunjukkan label sesungguhnya sedangkan garis x menunjukkan label prediksi. Pada label *Cured* terdapat 85 citra yang diprediksi benar untuk label *Cured*, 5 citra diprediksi benar untuk label *Deceased*, dan 24 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Cured* sebesar 85 dari 114 citra pengujian dan tingkat kesalahannya yaitu sebanyak 29 dari 114 citra pengujian. Kemudian untuk label *Deceased* terdapat 15 citra yang diprediksi benar untuk label *Cured*, 80 citra diprediksi benar untuk label *Deceased*, dan 5 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Deceased* sebesar 80 dari 100 citra pengujian dan tingkat kesalahannya yaitu sebanyak 20 dari 100 citra pengujian. Sedangkan untuk label Unknown terdapat 8 citra yang diprediksi benar untuk label *Cured*, 3 citra diprediksi benar untuk label *Deceased*, dan 92 citra diprediksi benar untuk label *Unknown*. Artinya tingkat kebenaran pada label *Unknown* sebesar 92 dari 103 citra pengujian dan tingkat kesalahannya yaitu sebanyak 11 dari 103 citra pengujian.

Dari nilai kebenaran dan nilai prediksi tersebut, maka dapat digunakan untuk menghitung performa metrik berupa nilai akurasi, presisi, *recall* dan *F1 score* sebagai berikut.

Akurasi

* Akurasi *Cured*

|  |  |
| --- | --- |
|  | (4.1) |

* Akurasi *Deceased*
* Akurasi *Unknown*
* Makro akurasi

Presisi

* Presisi *Cured*

|  |  |
| --- | --- |
|  | (4.2) |

* Presisi *Deceased*
* Presisi *Unknown*
* Makro presisi

Makro

*Recall*

* *Recall Cured*

|  |  |
| --- | --- |
|  | (4.3) |

* *Recall* Deceased
* *Recall Unknown*
* Makro *recall*

*F1 score*

|  |  |
| --- | --- |
|  | (4.4) |
|  |  |

Berdasarkan perhitungan akurasi, presisi, recall dan F1 dapat disimpulkan bahwa klasifikasi *mortality* dengan model VGG16 dan citra *preprocessed* memiliki nilai mendekati nilai 1.00, sehingga model mampu melakukan klasifikasi *CT scan* dada secara akurat.

## Perbandingan Hasil Pelatihan

Untuk mengetahui perbedaan tingkat akurasi pelatihan dan validasi pada setiap sistem klasifikasi antara arsitektur VGG16 dan arsitektur ResNet50, maka dilakukan perbandingan grafik tingkat akurasi pelatihan dan dan kesalahan pada setiap pelatihannya.

### Perbandingan Lung Parenchyma Classification Arsitektur VGG16 dan Arsitektur ResNet50 Original Image

Grafik perbandingan akurasi pelatihan pada klasifikasi lung parenchyma menggunakan arsitektur VGG16 dan arsitektur ResNet50 ditampilkan pada Gambar 4.49

Gambar 4.49 Grafik perbandingan akurasi pelatihan dan validasi

klasifikasi lung parenchyma

Berdasarkan grafik perbandingan akurasi pelatihan dapat diketahui bahwa arsitektur VGG16 yang menggunakan *dataset* citra *original* memiliki nilai akurasi pelatihan dan akurasi validasi paling tinggi dibandingkan dengan arsitektur lainnya. Suatu model dapat dikatakan dalam kondisi *goodfit* jika nilai akurasinya tinggi dan perbedaan antara nilai akurasi pelatihan dan akurasi validasinya sangat kecil. Jika sebaliknya maka model tersebut dalam kondisi *overfitting* maupun *underfitting*.

Untuk grafik perbandingan akurasi pelatihan pada klasifikasi *lung parenchyma* menggunakan arsitektur VGG16 dan arsitektur ResNet 50 ditampilkan pada Gambar 4.50

Gambar 4.50 Grafik perbandingan kesalahan pelatihan dan validasi

Berdasarkan grafik perbandingan akurasi pelatihan di atas dapat diketahui bahwa model yang memiliki nilai kesalahan pelatihan dan kesalahan validasi paling rendah adalah arsitektur VGG16 dengan *dataset* citra *original*. Suatu model dapat dikatakan dalam kondisi baik jika memiliki nilai kesalahan rendah dan mendekati 0.00. Dari grafik perbandingan nilai akurasi dan kesalahan, maka dapat disimpulkan bahwa model dengan arsitektur VGG16 yang menggunakan citra *original* memiliki performa paling baik dibandingkan model lainnya.

### Perbandingan Covid Positivity Classification Arsitektur VGG16 dan Arsitektur ResNet50

Grafik perbandingan akurasi pelatihan pada klasifikasi *covid positivity* menggunakan arsitektur VGG16 dan arsitektur ResNet50 ditampilkan pada Gambar 4.51

Gambar 4.51 Grafik perbandingan akurasi pelatihan dan validasi klasifikasi covid positivity

Berdasarkan grafik perbandingan akurasi pelatihan dapat diketahui bahwa arsitektur VGG16 yang menggunakan *dataset* citra *original* memiliki nilai akurasi pelatihan dan akurasi validasi paling tinggi dibandingkan dengan arsitektur lainnya. Meskipun nilai akurasi validasinya masih jauh dari angka 1.00 dan berada pada interval jauh di bawah nilai akurasi pelatihan. Suatu model dapat dikatakan dalam kondisi *goodfit* jika nilai akurasinya tinggi dan perbedaan antara nilai akurasi pelatihan dan akurasi validasinya sangat kecil. Jika sebaliknya maka model tersebut dalam kondisi *overfitting* maupun *underfitting*.

Untuk grafik perbandingan akurasi pelatihan pada klasifikasi *covid positivity* menggunakan arsitektur VGG16 dan arsitektur ResNet50 ditampilkan pada Gambar 4.52

Gambar 4.52 Grafik perbandingan kesalahan pelatihan dan validasi

klasifikasi covid positivity

Berdasarkan grafik perbandingan akurasi pelatihan di atas dapat diketahui bahwa model yang memiliki nilai kesalahan pelatihan paling rendah adalah arsitektur VGG16 dengan *dataset* citra *original*. Namun, untuk nilai kesalahan validasinya masih jauh di atas nilai 0.00 dan nilai kesalahan pelatihan. Suatu model dapat dikatakan dalam kondisi baik jika memiliki nilai kesalahan rendah dan mendekati 0.00. Dari grafik perbandingan nilai akurasi dan kesalahan, maka dapat disimpulkan bahwa model dengan arsitektur VGG16 yang menggunakan citra *original* memiliki performa paling baik dibandingkan model lainnya, meskipun kondisinya *overfitting.*

### Perbandingan Risk Classification Arsitektur VGG16 dan Arsitektur ResNet50

Grafik perbandingan akurasi pelatihan pada klasifikasi lung parenchyma menggunakan arsitektur VGG16 dan arsitektur ResNet50 ditampilkan pada Gambar 4.53

Gambar 4.53Grafik perbandingan akurasi pelatihan dan validasi risk classification

Berdasarkan grafik perbandingan akurasi pelatihan dapat diketahui bahwa arsitektur VGG16 yang menggunakan *dataset* citra *original* memiliki nilai akurasi pelatihan dan akurasi validasi paling tinggi dibandingkan dengan arsitektur lainnya. Meskipun nilai akurasi validasinya masih jauh dari angka 1.00 dan berada padda interval jauh di bawah nilai akurasi pelatihan. Suatu model dapat dikatakan dalam kondisi *goodfit* jika nilai akurasinya tinggi dan perbedaan antara nilai akurasi pelatihan dan akurasi validasinya sangat kecil. Jika sebaliknya maka model tersebut dalam kondisi *overfitting* maupun *underfitting*.

Untuk grafik perbandingan akurasi pelatihan pada klasifikasi *covid positivity* menggunakan arsitektur VGG16 dan arsitektur ResNet50 ditampilkan pada

Gambar 4.54Grafik perbandingan kesalahan pelatihan dan validasi risk classification

Berdasarkan grafik perbandingan akurasi pelatihan di atas dapat diketahui bahwa model yang memiliki nilai kesalahan pelatihan paling rendah adalah arsitektur VGG16 dengan *dataset* citra *original*. Namun, untuk nilai kesalahan validasinya masih jauh di atas nilai 0.00 dan nilai kesalahan pelatihan. Suatu model dapat dikatakan dalam kondisi baik jika memiliki nilai kesalahan rendah dan mendekati 0.00. Dari grafik perbandingan nilai akurasi dan kesalahan, maka dapat disimpulkan bahwa model dengan arsitektur VGG16 yang menggunakan citra *original* memiliki performa paling baik dibandingkan model lainnya, meskipun kondisinya *overfitting.*

### Perbandingan Mortality Classification Arsitektur VGG16 dan Arsitektur ResNet50

Grafik perbandingan akurasi pelatihan pada klasifikasi *mortality* menggunakan arsitektur VGG16 dan arsitektur ResNet50 ditampilkan pada Gambar 4.55

Gambar 4.55 Grafik perbandingan akurasi pelatihan dan validasi

klasifikasi mortality

Berdasarkan grafik perbandingan akurasi pelatihan dapat diketahui bahwa arsitektur VGG16 yang menggunakan citra *original* memiliki nilai akurasi pelatihan paling tinggi dibandingkan dengan arsitektur lainnya. Namun untuk nilai akurasi validasi rata – rata untuk semua model masih jauh dari angka 1.00. Suatu model dapat dikatakan dalam kondisi goodfit jika nilai akurasinya tinggi dan perbedaan antara nilai akurasi pelatihan dan akurasi validasinya sangat kecil. Jika sebaliknya maka model tersebut dalam kondisi *overfitting* maupun *underfitting*.

Untuk grafik perbandingan kesalahan pelatihan pada klasifikasi *mortality* menggunakan arsitektur VGG16 dan arsitektur ResNet50 ditampilkan pada Gambar 4.56

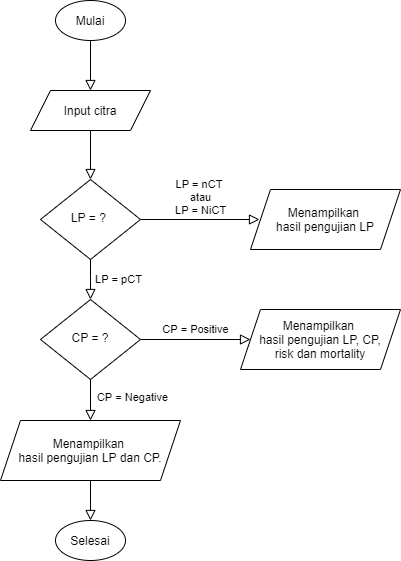
Gambar 4.56 Grafik perbandingan kesalahan pelatihan dan validasi

klasifikasi mortality

Berdasarkan grafik perbandingan akurasi pelatihan di atas dapat diketahui bahwa model yang memiliki nilai kesalahan pelatihan dan kesalahan validasi paling rendah adalah arsitektur VGG16 dengan citra *original*. Suatu model dapat dikatakan dalam kondisi baik jika memiliki nilai kesalahan rendah dan mendekati 0.00. Dari grafik perbandingan nilai akurasi dan kesalahan, dapat diketahui bahwa model dengan arsitektur VGG16 yang menggunakan citra *original* memiliki nilai kesalahan pelatihan mendekati 0.00. Namun, untuk nilai kesalahan validasinya masih jauh dari 0.00, maka dapat disimpulkan bahwa model tersebut memiliki performa kurang baik atau dalam kondisi *overfitting*.

## Hasil Pengujian pada Google Colaboratory

Proses pengujian pada *google colaboratory* menggunakan model dengan performa terbaik pada setiap sistem klasifikasi. Alur dari pengujian tersebut sesuai dengan diagram alir pada Gambar 4.57



Gambar 4.57 Diagram alir pengujian pada google colaboratory

Berdasarkan diagram alir tersebut dapat diketahui bahwa tahap pertama adalah menginputkan *CT scan* dada yang akan dilakukan deteksi. Kemudian, sistem akan mendeteksi kelas *lung parenchyma* dari *CT scan* dada tersebut, jika hasil deteksi *lung parenchyma* bernilai NiCT atau nCT, maka sistem akan langsung menampilkan hasil deteksi *lung parenchyma* beserta nilai probabilitasnya untuk setiap kelas. Sedangkan, ketika hasil deteksi citra bernilai pCT maka citra akan dilanjutkan ke proses deteksi *covid positivity*.

Pada tahap deteksi *covid positivity*, ketika hasil deteksi citra bernilai *Positive*, maka sistem akan menampilkan hasil deteksi dari *lung parenchyma*, *covid positivity, risk* dan i beserta nilai probabilitas setiap kelasnya. Namun ketika hasil deteksi *covid positivity* bernilai *Negative*, maka sistem hanya akan menampilkan hasil deteksi *covid positivity*nya saja beserta dengan nilai probabilitas setiap kelas.

### Pengujian Klasifikasi *Lung Parenchyma* pada *Google Colaboratory* Menggunakan *Dataset Original Image*

Pada pengujian *lung parenchyma* yang menggunakan *dataset original* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas NiCT, 10 citra kelas nCT dan 10 citra pCT. Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.1

Tabel 4.1Hasil pengujian deteksi lung parenchyma menggunakan citra original

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | NiCT001 | nCT (1.00) | Salah |
| 2 | NiCT002 | nCT (1.00) | Salah |
| 3 | NiCT003 | nCT (1.00) | Salah |
| 4 | NiCT004 | nCT (1.00) | Salah |
| 5 | NiCT005 | nCT (1.00) | Salah |
| 6 | NiCT006 | nCT (1.00) | Salah |
| 7 | NiCT007 | nCT (1.00) | Salah |
| 8 | NiCT008 | nCT (1.00) | Salah |
| 9 | NiCT009 | nCT (1.00) | Salah |
| 10 | NiCT010 | nCT (1.00) | Salah |
| 11 | nCT001 | nCT (1.00) | Benar |
| 12 | nCT002 | nCT (1.00) | Benar |
| 13 | nCT003 | nCT (1.00) | Benar |
| 14 | nCT004 | nCT (1.00) | Benar |
| 15 | nCT005 | nCT (1.00) | Benar |
| 16 | nCT006 | nCT (1.00) | Benar |
| 17 | nCT007 | nCT (1.00) | Benar |
| 18 | nCT008 | nCT (1.00) | Benar |
| 19 | nCT009 | nCT (1.00) | Benar |
| 20 | nCT010 | nCT (1.00) | Benar |
| 21 | pCT001 | nCT (1.00) | Salah |
| 22 | pCT002 | nCT (1.00) | Salah |
| 23 | pCT003 | nCT (1.00) | Salah |
| 24 | pCT004 | nCT (1.00) | Salah |
| 25 | pCT005 | nCT (1.00) | Salah |
| 26 | pCT006 | nCT (1.00) | Salah |
| 27 | pCT007 | pCT (1.00) | Salah |
| 28 | pCT008 | nCT (1.00) | Salah |
| 29 | pCT009 | nCT (1.00) | Salah |
| 30 | pCT010 | nCT (1.00) | Salah |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian menggunakan *dataset original* sebagian besar nilai pengujiannya bernilai salah dengan tingkat probabilitas 100%. Meskipun ada beberapa yang bernilai benar. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian dengan menggunakan citra *original* tidak akurat untuk melakukan deteksi citra *CT Scan* dada untuk *Covid-19*, yaitu dengan persentase akurasi hanya sebesar 33%.

### Pengujian pada *Google Colaboratory* Menggunakan *Dataset Preprocessed Image*

Pada pengujian *lung parenchyma* yang menggunakan dataset *preprocessed* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas NiCT, 10 citra kelas nCT dan 10 citra pCT. Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.2

Tabel 4.2Hasil pengujian deteksi lung parenchyma menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | NiCT01 | NiCT (0.95) | Benar |
| 2 | NiCT02 | NiCT (0.99) | Benar |
| 3 | NiCT03 | NiCT (0.99) | Benar |
| 4 | NiCT04 | NiCT (0.99) | Benar |
| 5 | NiCT05 | NiCT (0.99) | Benar |
| 6 | NiCT06 | NiCT (0.66) | Benar |
| 7 | NiCT07 | NiCT (0.99) | Benar |
| 8 | NiCT08 | NiCT (0.99) | Benar |
| 9 | NiCT09 | NiCT (0.99) | Benar |
| 10 | NiCT10 | NiCT (0.99) | Benar |
| 11 | nCT01 | nCT (1.00) | Benar |
| 12 | nCT02 | pCT (1.00) | Salah |
| 13 | nCT03 | nCT (1.00) | Benar |
| 14 | nCT04 | nCT (1.00) | Benar |
| 15 | nCT05 | pCT (1.00) | Salah |
| 16 | nCT06 | nCT (1.00) | Benar |
| 17 | nCT07 | nCT (1.00) | Benar |
| 18 | nCT08 | pCT (1.00) | Salah |
| 19 | nCT09 | nCT (1.00) | Benar |
| 20 | nCT10 | pCT (1.00) | Salah |
| 21 | pCT01 | pCT (1.00) | Benar |
| 22 | pCT02 | pCT (1.00) | Benar |
| 23 | pCT03 | pCT (1.00) | Benar |
| 24 | pCT04 | pCT (1.00) | Benar |
| 25 | pCT05 | pCT (1.00) | Benar |
| 26 | pCT06 | pCT (1.00) | Benar |
| 27 | pCT07 | pCT (1.00) | Benar |
| 28 | pCT08 | pCT (1.00) | Benar |
| 29 | pCT09 | pCT (1.00) | Benar |
| 30 | pCT10 | pCT (1.00) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian menggunakan dataset preprocessed sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 66% hingga 100%. Meskipun masih ada sedikit yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian dengan menggunakan citra preprocessed cukup akurat untuk melakukan deteksi citra CT Scan dada untuk Covid-19, yaitu dengan persentase akurasi sebesar 87%.

### Pengujian Klasifikasi *Covid Positivity* pada *Google Colaboratoty* Menggunakan *Dataset Original Image*

Pada pengujian klasifikasi *covid positivity* yang menggunakan *dataset original* terdapat 20 citra *CT scan* dada yang terdiri dari 10 citra kelas *Negative*, dan 10 citra *Positive*. Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.3

Tabel.4.3Hasil pengujian covid positivity pada google colaboratory dengan citra original

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Negative001 | Negative (1.00) | Benar |
| 2 | Negative002 | Negative (1.00) | Benar |
| 3 | Negative003 | Negative (1.00) | Benar |
| 4 | Negative004 | Negative (1.00) | Benar |
| 5 | Negative005 | Negative (1.00) | Benar |
| 6 | Negative006 | Negative (1.00) | Benar |
| 7 | Negative007 | Negative (1.00) | Benar |
| 8 | Negative008 | Negative (1.00) | Benar |
| 9 | Negative009 | Positive (0.83) | Salah |
| 10 | Negative010 | Negative (1.00) | Benar |
| 11 | Positive001 | Negative (1.00) | Salah |
| 12 | Positive002 | Positive (1.00) | Benar |
| 13 | Positive003 | Positive (1.00) | Benar |
| 14 | Positive004 | Negative (1.00) | Salah |
| 15 | Positive005 | Positive (0.97) | Benar |
| 16 | Positive006 | Positive (0.99) | Benar |
| 17 | Positive007 | Positive (1.00) | Benar |
| 18 | Positive008 | Positive (1.00) | Benar |
| 19 | Positive009 | Negative (1.00) | Salah |
| 20 | Positive010 | Negative (1.00) | Salah |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian menggunakan dataset *original* sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 83% hingga 100%. Meskipun masih ada yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian dengan menggunakan citra original cukup akurat untuk melakukan deteksi citra CT Scan dada untuk klasifikasi covid positivity, yaitu dengan persentase akurasi sebesar 75%.

### Pengujian Klasifikasi *Covid Positivity* pada *Google Colaboratoty* Menggunakan *Dataset Preprocessed Image*

Pada pengujian klasifikasi *covid positivity* yang menggunakan *dataset preprocessed* terdapat 20 citra *CT scan* dada yang terdiri dari 10 citra kelas *Negative*, dan 10 citra *Positive.* Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.4

Tabel 4.4Tabel pengujian covid positivity menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Negative001 | Positive (0.99) | Salah |
| 2 | Negative002 | Positive (1.00) | Salah |
| 3 | Negative003 | Negative (1.00) | Benar |
| 4 | Negative004 | Positive (0.99) | Salah |
| 5 | Negative005 | Positive (1.00) | Salah |
| 6 | Negative006 | Positive (1.00) | Salah |
| 7 | Negative007 | Positive (1.00) | Salah |
| 8 | Negative008 | Negative (1.00) | Benar |
| 9 | Negative009 | Negative (1.00) | Benar |
| 10 | Negative010 | Positive (1.00) | Salah |
| 11 | Positive001 | Negative (1.00) | Salah |
| 12 | Positive002 | Positive (0.99) | Benar |
| 13 | Positive003 | Negative (0.99) | Salah |
| 14 | Positive004 | Negative (1.00) | Salah |
| 15 | Positive005 | Positive (0.99) | Benar |
| 16 | Positive006 | Negative (1.00) | Salah |
| 17 | Positive007 | Positive (0.99) | Benar |
| 18 | Positive008 | Positive (0.91) | Benar |
| 19 | Positive009 | Negative (1.00) | Salah |
| 20 | Positive010 | Negative (1.00) | Salah |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian menggunakan dataset preprocessed sebagian besar nilai pengujiannya bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian dengan menggunakan citra *preprocessed* belum akurat untuk melakukan deteksi citra CT Scan dada untuk klasifikasi *covid positivity*, yaitu dengan persentase akurasi hanya sebesar 35%.

### Pengujian Klasifikasi *Risk* pada *Google Colaboratory* Menggunakan *Dataset Original Image*

Pada pengujian klasifikasi *risk* yang menggunakan dataset *original* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas *Control,* 10 citra kelas *Type I* dan 10 citra *Type II*. Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.5

Tabel 4.5Hasil pengujian deteksi risk menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Control001 | Control (1.00) | Benar |
| 2 | Control002 | Control (1.00) | Benar |
| 3 | Control003 | Control (1.00) | Benar |
| 4 | Control004 | Control (1.00) | Benar |
| 5 | Control005 | Control (1.00) | Benar |
| 6 | Control006 | Control (1.00) | Benar |
| 7 | Control007 | Control (1.00) | Benar |
| 8 | Control008 | Control (1.00) | Benar |
| 9 | Control009 | Control (1.00) | Benar |
| 10 | Control010 | Control (1.00) | Benar |
| 11 | TypeI001 | Type I (1.00) | Benar |
| 12 | TypeI002 | Control (1.00) | Salah |
| 13 | TypeI003 | Control (1.00) | Salah |
| 14 | TypeI004 | Control (1.00) | Salah |
| 15 | TypeI005 | Control (1.00) | Salah |
| 16 | TypeI006 | Control (1.00) | Salah |
| 17 | TypeI007 | Control (1.00) | Salah |
| 18 | TypeI008 | Control (1.00) | Salah |
| 19 | TypeI009 | Control (1.00) | Salah |
| 20 | TypeI010 | Control (1.00) | Salah |
| 21 | TypeII001 | Type II (1.00) | Benar |
| 22 | TypeII002 | Type II (1.00) | Benar |
| 23 | TypeII003 | Control (0.95) | Salah |
| 24 | TypeII004 | Type II (1.00) | Benar |
| 25 | TypeII005 | Control (1.00) | Salah |
| 26 | TypeII006 | Type II (1.00) | Benar |
| 27 | TypeII007 | Control (1.00) | Salah |
| 28 | TypeII008 | Type II (1.00) | Benar |
| 29 | TypeII009 | Type II (1.00) | Benar |
| 30 | TypeII010 | Type II (1.00) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian klasifikasi risk menggunakan dataset original sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 95% hingga 100%. Meskipun masih ada yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian klasifikasi *risk* dengan menggunakan citra original belum cukup akurat untuk melakukan deteksi citra *CT Scan* dada untuk identifikasi *risk*, yaitu dengan persentase akurasi hanya sebesar 60%.

### Pengujian Klasifikasi *Risk* pada *Google Colaboratory* Menggunakan *Dataset Preprocessed Image*

Pada pengujian klasifikasi *risk* yang menggunakan dataset *preprocessed* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas *Control*, 10 citra kelas *Type I* dan 10 citra *Type II.* Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.6

Tabel 4.6Hasil pengujian deteksi risk menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Control001 | Control (1.00) | Benar |
| 2 | Control002 | Control (1.00) | Benar |
| 3 | Control003 | Control (1.00) | Benar |
| 4 | Control004 | Control (1.00) | Benar |
| 5 | Control005 | Control (1.00) | Benar |
| 6 | Control006 | Control (1.00) | Benar |
| 7 | Control007 | Control (1.00) | Benar |
| 8 | Control008 | Control (1.00) | Benar |
| 9 | Control009 | Control (1.00) | Benar |
| 10 | Control010 | Control (1.00) | Benar |
| 11 | TypeI001 | Control (1.00) | Salah |
| 12 | TypeI002 | Control (1.00) | Salah |
| 13 | TypeI003 | Control (1.00) | Salah |
| 14 | TypeI004 | Control (1.00) | Salah |
| 15 | TypeI005 | Control (1.00) | Salah |
| 16 | TypeI006 | Control (1.00) | Salah |
| 17 | TypeI007 | Control (1.00) | Salah |
| 18 | TypeI008 | Control (1.00) | Salah |
| 19 | TypeI009 | Control (1.00) | Salah |
| 20 | TypeI010 | Control (1.00) | Salah |
| 21 | TypeII001 | Control (1.00) | Salah |
| 22 | TypeII002 | Type II (1.00) | Benar |
| 23 | TypeII003 | Control (1.00) | Salah |
| 24 | TypeII004 | Type II (1.00) | Benar |
| 25 | TypeII005 | Type II (0.51) | Benar |
| 26 | TypeII006 | Control (1.00) | Salah |
| 27 | TypeII007 | Control (1.00) | Salah |
| 28 | TypeII008 | Control (1.00) | Salah |
| 29 | TypeII009 | Type II (1.00) | Benar |
| 30 | TypeII010 | Type II (0.99) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian klasifikasi risk menggunakan dataset original sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 95% hingga 100%. Meskipun masih ada yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian klasifikasi *risk* dengan menggunakan citra original belum cukup akurat untuk melakukan deteksi citra *CT Scan* dada, yaitu dengan persentase akurasi hanya sebesar 50%.

### Pengujian Klasifikasi *Mortality* pada *Google Colaboratory* Menggunakan *Dataset Original Image*

Pada pengujian klasifikasi *mortality* yang menggunakan dataset *original* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas *Cured*, 10 citra kelas *Deceased* dan 10 citra *Unknown.* Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.0.7

Tabel 4.7Hasil pengujian deteksi mortality menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Cured001 | Unknown (1.00) | Salah |
| 2 | Cured002 | Cured (1.00) | Benar |
| 3 | Cured003 | Cured (1.00) | Benar |
| 4 | Cured004 | Cured (1.00) | Benar |
| 5 | Cured005 | Cured (1.00) | Benar |
| 6 | Cured006 | Cured (1.00) | Benar |
| 7 | Cured007 | Cured (1.00) | Benar |
| 8 | Cured008 | Cured (1.00) | Benar |
| 9 | Cured009 | Cured (1.00) | Benar |
| 10 | Cured010 | Cured (1.00) | Benar |
| 11 | Deceased001 | Unknown (1.00) | Salah |
| 12 | Deceased002 | Cured (1.00) | Salah |
| 13 | Deceased003 | Deceased (1.00) | Benar |
| 14 | Deceased004 | Deceased (1.00) | Benar |
| 15 | Deceased005 | Deceased (1.00) | Benar |
| 16 | Deceased006 | Deceased (1.00) | Benar |
| 17 | Deceased007 | Deceased (1.00) | Benar |
| 18 | Deceased008 | Unknown (1.00) | Salah |
| 19 | Deceased009 | Cured (1.00) | Salah |
| 20 | Deceased010 | Deceased (0.88) | Benar |
| 21 | Unknown001 | Unknown (0.97) | Benar |
| 22 | Unknown002 | Unknown (1.00) | Benar |
| 23 | Unknown003 | Cured (1.00) | Salah |
| 24 | Unknown004 | Cured (1.00) | Salah |
| 25 | Unknown005 | Unknown (1.00) | Benar |
| 26 | Unknown006 | Unknown (1.00) | Benar |
| 27 | Unknown007 | Unknown (1.00) | Benar |
| 28 | Unknown008 | Deceased (1.00) | Salah |
| 29 | Unknown009 | Unknown (0.95) | Benar |
| 30 | Unknown010 | Unknown (1.00) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian klasifikasi *mortality* menggunakan dataset *original* sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 88% hingga 100%. Meskipun masih ada yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian klasifikasi *mortality* dengan menggunakan citra *original* cukup akurat untuk melakukan deteksi citra CT Scan dada untuk identifikas mortality, yaitu dengan persentase akurasi sebesar 73%.

### Pengujian Klasifikasi *Mortality* pada *Google Colaboratory* Menggunakan *Dataset Preprocessed Image*

Pada pengujian klasifikasi *mortality* yang menggunakan dataset *preprocessed* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas *Cured*, 10 citra kelas *Deceased* dan 10 citra *Unknown.* Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.0.8

Tabel 4.8Hasil pengujian deteksi menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Cured001 | Deceased (1.00) | Salah |
| 2 | Cured002 | Unknown (1.00) | Salah |
| 3 | Cured003 | Deceased (1.00) | Salah |
| 4 | Cured004 | Deceased (1.00) | Salah |
| 5 | Cured005 | Unknown (1.00) | Salah |
| 6 | Cured006 | Unknown (1.00) | Salah |
| 7 | Cured007 | Deceased (1.00) | Salah |
| 8 | Cured008 | Deceased (1.00) | Salah |
| 9 | Cured009 | Deceased (1.00) | Salah |
| 10 | Cured010 | Deceased (1.00) | Salah |
| 11 | Deceased001 | Deceased (1.00) | Benar |
| 12 | Deceased002 | Unknown (1.00) | Salah |
| 13 | Deceased003 | Unknown (1.00) | Salah |
| 14 | Deceased004 | Deceased (1.00) | Benar |
| 15 | Deceased005 | Deceased (1.00) | Benar |
| 16 | Deceased006 | Deceased (1.00) | Benar |
| 17 | Deceased007 | Unknown (1.00) | Salah |
| 18 | Deceased008 | Unknown (1.00) | Salah |
| 19 | Deceased009 | Unknown (1.00) | Salah |
| 20 | Deceased010 | Unknown (1.00) | Salah |
| 21 | Unknown001 | Unknown (1.00) | Benar |
| 22 | Unknown002 | Unknown (1.00) | Benar |
| 23 | Unknown003 | Unknown (1.00) | Benar |
| 24 | Unknown004 | Unknown (1.00) | Benar |
| 25 | Unknown005 | Unknown (1.00) | Benar |
| 26 | Unknown006 | Unknown (1.00) | Benar |
| 27 | Unknown007 | Unknown (1.00) | Benar |
| 28 | Unknown008 | Unknown (1.00) | Benar |
| 29 | Unknown009 | Unknown (1.00) | Benar |
| 30 | Unknown010 | Unknown (1.00) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian klasifikasi *mortality* menggunakan dataset *preprocessed* sebagian besar nilai pengujiannya masih bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian klasifikasi *mortality* dengan menggunakan citra *preprocessed* belum cukup akurat untuk melakukan deteksi citra *CT Scan* dada untuk identifikasi *mortality*, yaitu dengan persentase akurasi hanya sebesar 47%.

## Perbandingan Pengujian pada *Google Colaboratory*

### Perbandingan Pengujian Klasifikasi *Lung Parenchyma* pada *Google Colaboratory*

Berdasarkan hasil pengujian antara yang menggunakan citra *original* dengan citra *preprocessed,* maka dapat direpresentasikan ke dalam grafik perbandingan akurasi pengujian antara citra *original* dan citra *preprocessed* berikut.

Gambar 4.58Grafik perbandingan hasil pengujian lung parenchyma pada google colaboratory

Berdasarkan grafik hasil pengujian tersebut dapat diketahui bahwa pengujian menggunakan citra *preprocessed* lebih baik dibandingkan dengan pengujian yang menggunakan citra *original*. Oleh karena itu untuk melakukan klasifikasi dan deteksi *CT scan* dada untuk *Covid-19* lebih baik menggunakan citra *preprocessed*.

### Perbandingan Pengujian *Covid Positivity* pada *Google Colaboratory*

Berdasarkan hasil pengujian klasifikasi *covid positivity* antara yang menggunakan citra *original* dengan citra *preprocessed,* maka dapat direpresentasikan ke dalam grafik perbandingan akurasi pengujian antara citra *original* dan citra *preprocessed* berikut.

Gambar 4.59 Grafik perbandingan hasil pengujian covid positivity pada google colaboratory

Berdasarkan grafik hasil pengujian tersebut dapat diketahui bahwa pengujian menggunakan citra *preprocessed* lebih baik dibandingkan dengan pengujian yang menggunakan citra *original*. Oleh karena itu untuk melakukan klasifikasi dan deteksi *CT scan* dada untuk *covid positivity* lebih baik menggunakan citra *preprocessed.*

### Perbandingan Pengujian *Risk* pada *Google Colaboratory*

Berdasarkan hasil pengujian klasifikasi *risk* antara yang menggunakan citra *original* dengan citra *preprocessed,* maka dapat direpresentasikan ke dalam grafik perbandingan akurasi pengujian antara citra *original* dan citra *preprocessed* berikut.

Gambar 4.60 Grafik perbandingan hasil pengujian risk pada google colaboratory

Berdasarkan grafik hasil pengujian tersebut dapat diketahui bahwa pengujian menggunakan citra *original* lebih baik dibandingkan dengan pengujian yang menggunakan citra *preprocessed.* Oleh karena itu untuk melakukan klasifikasi dan deteksi *CT scan* dada untuk identifikasi *risk* lebih baik menggunakan citra *original*.

### Perbandingan Pengujian *Mortality* pada *Google Colaboratory*

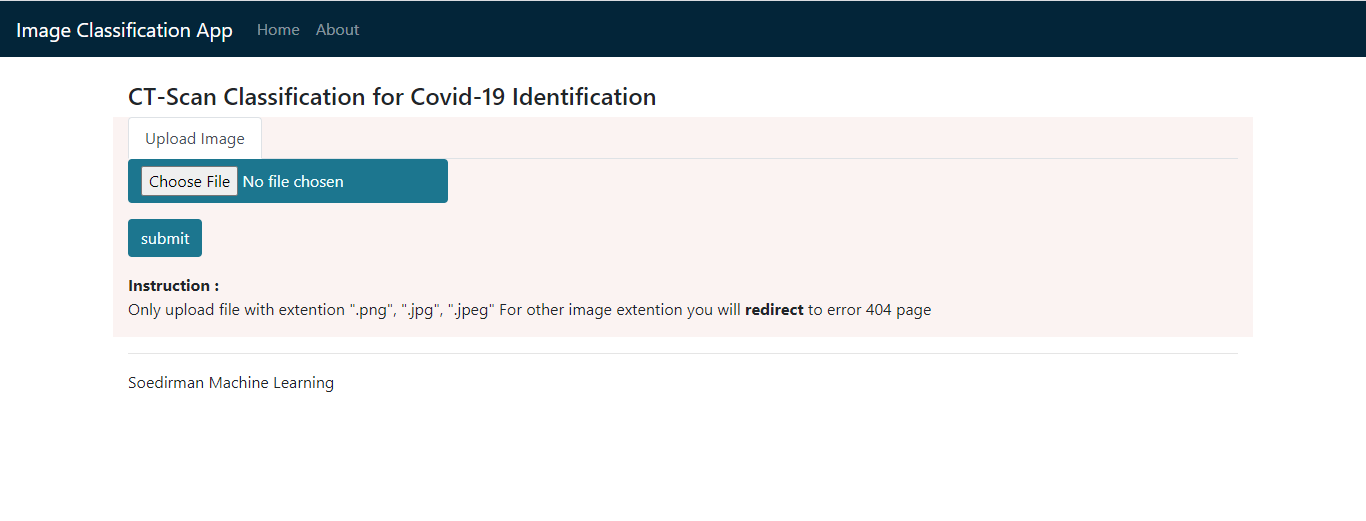
Berdasarkan hasil pengujian klasifikasi *mortality* antara yang menggunakan citra *original* dengan citra *preprocessed,* maka dapat direpresentasikan ke dalam grafik perbandingan akurasi pengujian antara citra original dan citra preprocessed berikut.

Gambar 4.61Grafik perbandingan hasil pengujian mortality pada google colaboratory

Berdasarkan grafik hasil pengujian tersebut dapat diketahui bahwa pengujian menggunakan citra *original* lebih baik dibandingkan dengan pengujian yang menggunakan citra *preprocessed.* Oleh karena itu untuk melakukan klasifikasi dan deteksi *CT scan* dada untuk identifikasi  lebih baik menggunakan citra *original*.

## Perancangan Aplikasi Web

Dalam merancang aplikasi web, hal pertama yang dilakukan adalah menentukan tampilan antarmuka dari aplikasi tersebut. Pada penelitian ini tampilan antarmuka menggunakan *template* pada *bootstrap* yang memiliki dua menu utama yaitu menu “Home” dan “About”. Gambar berikut merupakan tampilan antarmuka pada aplikasi web Identifikasi *CT Scan* Dada untuk *Covid-19.*

**

Gambar 4.62Tampilan antarmuka aplikasi web identifikasi ct scan dada untuk covid-19

Dapat dilihat pada Gambar 4.62 bahwa pada aplikasi web tersebut terdiri dari dua menu utama, yaitu menu “Home” dan “About”. Menu “Home” digunakan sebagai halaman untuk menginputkan citra *CT scan* dada dan melakukan deteksi. Sedangkan menu “About” berisi penjelasan dari aplikasi web tersebut.

Model yang digunakan dalam aplikasi web ini merupakan model dengan tingkat akurasi terbaik dari masing – masing proses klasifikasi yang kemudian telah dikonversi ke dalam format “.h5”. Pada *Visual Studio Code*, program dirancang dengan menggunakan bahasa pemrograman *python* dan *framework flask* untuk melakukan proses deteksi pada aplikasi web. Kemudian terdapat *folder* *templates* yang berisi program dengan format “.html” untuk mengelola tampilan pada web, selain itu aplikasi web ini juga memanfaatkan *template* yang diunduh dari *bootstrap*.

Kemudian *folder static* berisikan beberapa *sub-folder*, seperti *sub-folder* css dan js untuk menyimpan *template* dari *bootstrap*, *sub-folder models* sebagai tempat penyimpanan model yang akan digunakan pada aplikasi web. Dan yang terakhir adalag *sub-folder uploads* sebagaintempat penyimpanan citra *CT scan* dada yang telah diinputkan pada aplikasi web.

## Hasil Pengujian pada Aplikasi Web

Proses pengujian pada aplikasi web menggunakan model dengan tingkat akurasi terbaik dari masing – masing sistem klasifikasi. Secara teknis alur pengujian pada aplikasi web sama dengan pengujian pada *google colaboratory* yaitu seperti pada diagram alir pada Gambar 4.0.56. Pengujian pada aplikasi web juga menggunakan dua jenis citra, yaitu citra *original* dan citra *preprocessed* untuk membandingkan hasil diantara keduanya. Berikut adalah hasil dari pengujian pada aplikasi web untuk masing – masing sistem klasifikasi.

### Pengujian *Lung Parenchyma* pada Aplikasi Web menggunakan Citra Original

Pada pengujian aplikasi web untuk *lung parenchyma* yang menggunakan dataset *original* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas pCT, 10 citra kelas nCT, dan 10 citra NiCT*.* Hasil pengujian deteksi *CT scan* dada pada aplikasi web ditampilkan pada Tabel 4.9

Tabel 4.9 Hasil pengujian deteksi lung parenchyma pada aplikasi web menggunakan citra original

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | NiCT001 | nCT (1.00) | Salah |
| 2 | NiCT002 | nCT (1.00) | Salah |
| 3 | NiCT003 | nCT (1.00) | Salah |
| 4 | NiCT004 | nCT (1.00) | Salah |
| 5 | NiCT005 | nCT (1.00) | Salah |
| 6 | NiCT006 | nCT (1.00) | Salah |
| 7 | NiCT007 | nCT (1.00) | Salah |
| 8 | NiCT008 | nCT (1.00) | Salah |
| 9 | NiCT009 | nCT (1.00) | Salah |
| 10 | NiCT010 | nCT (1.00) | Salah |
| 11 | nCT001 | nCT (1.00) | Benar |
| 12 | nCT002 | nCT (1.00) | Benar |
| 13 | nCT003 | nCT (1.00) | Benar |
| 14 | nCT004 | nCT (1.00) | Benar |
| 15 | nCT005 | nCT (1.00) | Benar |
| 16 | nCT006 | nCT (1.00) | Benar |
| 17 | nCT007 | nCT (1.00) | Benar |
| 18 | nCT008 | nCT (1.00) | Benar |
| 19 | nCT009 | nCT (1.00) | Benar |
| 20 | nCT010 | nCT (1.00) | Benar |
| 21 | pCT001 | nCT (1.00) | Salah |
| 22 | pCT002 | nCT (1.00) | Salah |
| 23 | pCT003 | nCT (1.00) | Salah |
| 24 | pCT004 | nCT (1.00) | Salah |
| 25 | pCT005 | nCT (1.00) | Salah |
| 26 | pCT006 | nCT (1.00) | Salah |
| 27 | pCT007 | pCT (1.00) | Salah |
| 28 | pCT008 | nCT (1.00) | Salah |
| 29 | pCT009 | nCT (1.00) | Salah |
| 30 | pCT010 | nCT (1.00) | Salah |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian menggunakan *dataset original* sebagian besar nilai pengujiannya bernilai salah dengan tingkat probabilitas 100%. Meskipun ada beberapa yang bernilai benar. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian dengan menggunakan citra *original* tidak akurat untuk melakukan deteksi citra *CT Scan* dada untuk *Covid-19*, yaitu dengan persentase akurasi hanya sebesar 33%.

### Pengujian *Lung Parenchyma* pada Aplikasi Web Menggunakan *Dataset Preprocessed Image*

Pada pengujian aplikasi web untuk *lung parenchyma* yang menggunakan dataset *preprocessed* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas NiCT, 10 citra kelas nCT dan 10 citra pCT. Hasil pengujian deteksi *CT scan* dada pada aplikasi web ditampilkan pada Tabel 4.10

Tabel 4.10Hasil pengujian deteksi lung parenchyma pada aplikasi web menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | NiCT01 | NiCT (0.95) | Benar |
| 2 | NiCT02 | NiCT (0.99) | Benar |
| 3 | NiCT03 | NiCT (0.99) | Benar |
| 4 | NiCT04 | NiCT (0.99) | Benar |
| 5 | NiCT05 | NiCT (0.99) | Benar |
| 6 | NiCT06 | NiCT (0.66) | Benar |
| 7 | NiCT07 | NiCT (0.99) | Benar |
| 8 | NiCT08 | NiCT (0.99) | Benar |
| 9 | NiCT09 | NiCT (0.99) | Benar |
| 10 | NiCT10 | NiCT (0.99) | Benar |
| 11 | nCT01 | nCT (1.00) | Benar |
| 12 | nCT02 | pCT (1.00) | Salah |
| 13 | nCT03 | nCT (1.00) | Benar |
| 14 | nCT04 | nCT (1.00) | Benar |
| 15 | nCT05 | pCT (1.00) | Salah |
| 16 | nCT06 | nCT (1.00) | Benar |
| 17 | nCT07 | nCT (1.00) | Benar |
| 18 | nCT08 | pCT (1.00) | Salah |
| 19 | nCT09 | nCT (1.00) | Benar |
| 20 | nCT10 | pCT (1.00) | Salah |
| 21 | pCT01 | pCT (1.00) | Benar |
| 22 | pCT02 | pCT (1.00) | Benar |
| 23 | pCT03 | pCT (1.00) | Benar |
| 24 | pCT04 | pCT (1.00) | Benar |
| 25 | pCT05 | pCT (1.00) | Benar |
| 26 | pCT06 | pCT (1.00) | Benar |
| 27 | pCT07 | pCT (1.00) | Benar |
| 28 | pCT08 | pCT (1.00) | Benar |
| 29 | pCT09 | pCT (1.00) | Benar |
| 30 | pCT10 | pCT (1.00) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian pada aplikasi web untuk *lung parenchyma* menggunakan *dataset preprocessed* sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 66% hingga 100%. Meskipun masih ada sedikit yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian *lung parenchyma* pada aplikasi web dengan menggunakan citra *preprocessed* cukup akurat untuk melakukan deteksi citra CT Scan dada untuk Covid-19, yaitu dengan persentase akurasi sebesar 87%.

### Pengujian *Covid Positivity* pada Aplikasi Web Menggunakan *Dataset Original Image*

Pada pengujian aplikasi web untuk *covid positivity* yang menggunakan *dataset original* terdapat 20 citra *CT scan* dada yang terdiri dari 10 citra kelas *Negative*, dan 10 citra *Positive*. Hasil pengujian deteksi *CT scan* dada pada aplikasi web ditampilkan pada Tabel 4.11

Tabel 4.11Hasil pengujian deteksi covid positivity pada aplikasi web menggunakan citra original

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Negative001 | Negative (1.00) | Benar |
| 2 | Negative002 | Negative (1.00) | Benar |
| 3 | Negative003 | Negative (1.00) | Benar |
| 4 | Negative004 | Negative (1.00) | Benar |
| 5 | Negative005 | Negative (1.00) | Benar |
| 6 | Negative006 | Negative (1.00) | Benar |
| 7 | Negative007 | Negative (1.00) | Benar |
| 8 | Negative008 | Negative (1.00) | Benar |
| 9 | Negative009 | Positive (0.83) | Salah |
| 10 | Negative010 | Negative (1.00) | Benar |
| 11 | Positive001 | Negative (1.00) | Salah |
| 12 | Positive002 | Positive (1.00) | Benar |
| 13 | Positive003 | Positive (1.00) | Benar |
| 14 | Positive004 | Negative (1.00) | Salah |
| 15 | Positive005 | Positive (0.97) | Benar |
| 16 | Positive006 | Positive (0.99) | Benar |
| 17 | Positive007 | Positive (1.00) | Benar |
| 18 | Positive008 | Positive (1.00) | Benar |
| 19 | Positive009 | Negative (1.00) | Salah |
| 20 | Positive010 | Negative (1.00) | Salah |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pengujian *covid positivity* pada aplikasi web menggunakan dataset *original* sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 83% hingga 100%. Meskipun masih ada yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian aplikasi web untuk *covid positivity* dengan menggunakan citra *original* cukup akurat untuk melakukan deteksi citra *CT Scan* dada untuk *covid positivity*, yaitu dengan persentase akurasi sebesar 75%.

### Pengujian *Covid Positivity* pada Aplikasi Web Menggunakan *Dataset Preprocessed Image*

Pada pengujian aplikasi web untuk *covid positivity* yang menggunakan *dataset preprocessed* terdapat 20 citra *CT scan* dada yang terdiri dari 10 citra kelas *Negative*, dan 10 citra *Positive.* Hasil pengujian deteksi *CT scan* dada pada aplikasi web ditampilkan pada Tabel 4.12

Tabel 4.12Hasil pengujian deteksi covid positivity pada aplikasi web menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Negative001 | Positive (0.99) | Salah |
| 2 | Negative002 | Positive (1.00) | Salah |
| 3 | Negative003 | Negative (1.00) | Benar |
| 4 | Negative004 | Positive (0.99) | Salah |
| 5 | Negative005 | Positive (1.00) | Salah |
| 6 | Negative006 | Positive (1.00) | Salah |
| 7 | Negative007 | Positive (1.00) | Salah |
| 8 | Negative008 | Negative (1.00) | Benar |
| 9 | Negative009 | Negative (1.00) | Benar |
| 10 | Negative010 | Positive (1.00) | Salah |
| 11 | Positive001 | Negative (1.00) | Salah |
| 12 | Positive002 | Positive (0.99) | Benar |
| 13 | Positive003 | Negative (0.99) | Salah |
| 14 | Positive004 | Negative (1.00) | Salah |
| 15 | Positive005 | Positive (0.99) | Benar |
| 16 | Positive006 | Negative (1.00) | Salah |
| 17 | Positive007 | Positive (0.99) | Benar |
| 18 | Positive008 | Positive (0.91) | Benar |
| 19 | Positive009 | Negative (1.00) | Salah |
| 20 | Positive010 | Negative (1.00) | Salah |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian aplikasi web untuk *covid positivity* menggunakan *dataset preprocessed* sebagian besar nilai pengujiannya bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian aplikasi web dengan menggunakan citra *preprocessed* belum akurat untuk melakukan deteksi *covid positivity*, yaitu dengan persentase akurasi hanya sebesar 35%.

### Pengujian *Risk* pada Aplikasi Web Menggunakan *Dataset Original Image*

Pada pengujian aplikasi web untuk deteksi *risk* yang menggunakan dataset *original* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas *Control,* 10 citra kelas *Type I* dan 10 citra *Type II*. Hasil pengujian deteksi *CT scan* dada pada aplikasi web ditampilkan pada Tabel 4.13

Tabel 4.13Hasil pengujian deteksi risk pada aplikasi web menggunakan citra original

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Control001 | Control (1.00) | Benar |
| 2 | Control002 | Control (1.00) | Benar |
| 3 | Control003 | Control (1.00) | Benar |
| 4 | Control004 | Control (1.00) | Benar |
| 5 | Control005 | Control (1.00) | Benar |
| 6 | Control006 | Control (1.00) | Benar |
| 7 | Control007 | Control (1.00) | Benar |
| 8 | Control008 | Control (1.00) | Benar |
| 9 | Control009 | Control (1.00) | Benar |
| 10 | Control010 | Control (1.00) | Benar |
| 11 | TypeI001 | Type I (1.00) | Benar |
| 12 | TypeI002 | Control (1.00) | Salah |
| 13 | TypeI003 | Control (1.00) | Salah |
| 14 | TypeI004 | Control (1.00) | Salah |
| 15 | TypeI005 | Control (1.00) | Salah |
| 16 | TypeI006 | Control (1.00) | Salah |
| 17 | TypeI007 | Control (1.00) | Salah |
| 18 | TypeI008 | Control (1.00) | Salah |
| 19 | TypeI009 | Control (1.00) | Salah |
| 20 | TypeI010 | Control (1.00) | Salah |
| 21 | TypeII001 | Type II (1.00) | Benar |
| 22 | TypeII002 | Type II (1.00) | Benar |
| 23 | TypeII003 | Control (0.95) | Salah |
| 24 | TypeII004 | Type II (1.00) | Benar |
| 25 | TypeII005 | Control (1.00) | Salah |
| 26 | TypeII006 | Type II (1.00) | Benar |
| 27 | TypeII007 | Control (1.00) | Salah |
| 28 | TypeII008 | Type II (1.00) | Benar |
| 29 | TypeII009 | Type II (1.00) | Benar |
| 30 | TypeII010 | Type II (1.00) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian aplikasi web untuk deteksi *risk* menggunakan dataset *original* sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 95% hingga 100%. Meskipun masih ada yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian aplikasi web dengan menggunakan citra *original* belum cukup akurat untuk melakukan deteksi *risk*, yaitu dengan persentase akurasi hanya sebesar 60%.

### Pengujian *Risk* pada Aplikasi Web Menggunakan *Dataset Preprocessed Image*

Pada pengujian aplikasi web untuk deteksi *risk* yang menggunakan dataset *preprocessed* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas *Control*, 10 citra kelas *Type I* dan 10 citra *Type II.* Hasil pengujian deteksi *CT scan* dada pada aplikasi web ditampilkan pada Tabel 4.14

Tabel 4.14Hasil pengujian deteksi risk pada aplikasi web menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Control001 | Control (1.00) | Benar |
| 2 | Control002 | Control (1.00) | Benar |
| 3 | Control003 | Control (1.00) | Benar |
| 4 | Control004 | Control (1.00) | Benar |
| 5 | Control005 | Control (1.00) | Benar |
| 6 | Control006 | Control (1.00) | Benar |
| 7 | Control007 | Control (1.00) | Benar |
| 8 | Control008 | Control (1.00) | Benar |
| 9 | Control009 | Control (1.00) | Benar |
| 10 | Control010 | Control (1.00) | Benar |
| 11 | TypeI001 | Control (1.00) | Salah |
| 12 | TypeI002 | Control (1.00) | Salah |
| 13 | TypeI003 | Control (1.00) | Salah |
| 14 | TypeI004 | Control (1.00) | Salah |
| 15 | TypeI005 | Control (1.00) | Salah |
| 16 | TypeI006 | Control (1.00) | Salah |
| 17 | TypeI007 | Control (1.00) | Salah |
| 18 | TypeI008 | Control (1.00) | Salah |
| 19 | TypeI009 | Control (1.00) | Salah |
| 20 | TypeI010 | Control (1.00) | Salah |
| 21 | TypeII001 | Control (1.00) | Salah |
| 22 | TypeII002 | Type II (1.00) | Benar |
| 23 | TypeII003 | Control (1.00) | Salah |
| 24 | TypeII004 | Type II (1.00) | Benar |
| 25 | TypeII005 | Type II (0.51) | Benar |
| 26 | TypeII006 | Control (1.00) | Salah |
| 27 | TypeII007 | Control (1.00) | Salah |
| 28 | TypeII008 | Control (1.00) | Salah |
| 29 | TypeII009 | Type II (1.00) | Benar |
| 30 | TypeII010 | Type II (0.99) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian aplikasi web untuk deteksi *risk* menggunakan *dataset original* sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 95% hingga 100%. Meskipun masih ada yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian aplikasi web dengan menggunakan citra original belum cukup akurat untuk melakukan deteksi *risk*, yaitu dengan persentase akurasi hanya sebesar 50%.

### Pengujian Klasifikasi *Mortality* pada *Google Colaboratory* Menggunakan *Dataset Original Image*

Pada pengujian klasifikasi *mortality* yang menggunakan dataset *original* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas *Cured*, 10 citra kelas *Deceased* dan 10 citra *Unknown.* Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.15

Tabel 4.15Hasil pengujian deteksi mortality menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Cured001 | Unknown (1.00) | Salah |
| 2 | Cured002 | Cured (1.00) | Benar |
| 3 | Cured003 | Cured (1.00) | Benar |
| 4 | Cured004 | Cured (1.00) | Benar |
| 5 | Cured005 | Cured (1.00) | Benar |
| 6 | Cured006 | Cured (1.00) | Benar |
| 7 | Cured007 | Cured (1.00) | Benar |
| 8 | Cured008 | Cured (1.00) | Benar |
| 9 | Cured009 | Cured (1.00) | Benar |
| 10 | Cured010 | Cured (1.00) | Benar |
| 11 | Deceased001 | Unknown (1.00) | Salah |
| 12 | Deceased002 | Cured (1.00) | Salah |
| 13 | Deceased003 | Deceased (1.00) | Benar |
| 14 | Deceased004 | Deceased (1.00) | Benar |
| 15 | Deceased005 | Deceased (1.00) | Benar |
| 16 | Deceased006 | Deceased (1.00) | Benar |
| 17 | Deceased007 | Deceased (1.00) | Benar |
| 18 | Deceased008 | Unknown (1.00) | Salah |
| 19 | Deceased009 | Cured (1.00) | Salah |
| 20 | Deceased010 | Deceased (0.88) | Benar |
| 21 | Unknown001 | Unknown (0.97) | Benar |
| 22 | Unknown002 | Unknown (1.00) | Benar |
| 23 | Unknown003 | Cured (1.00) | Salah |
| 24 | Unknown004 | Cured (1.00) | Salah |
| 25 | Unknown005 | Unknown (1.00) | Benar |
| 26 | Unknown006 | Unknown (1.00) | Benar |
| 27 | Unknown007 | Unknown (1.00) | Benar |
| 28 | Unknown008 | Deceased (1.00) | Salah |
| 29 | Unknown009 | Unknown (0.95) | Benar |
| 30 | Unknown010 | Unknown (1.00) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian klasifikasi *mortality* menggunakan dataset *original* sebagian besar nilai pengujiannya bernilai benar dengan tingkat probabilitas mulai dari 88% hingga 100%. Meskipun masih ada yang bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian klasifikasi *mortality* dengan menggunakan citra *original* cukup akurat untuk melakukan deteksi citra CT Scan dada untuk identifikas mortality, yaitu dengan persentase akurasi sebesar 73%.

### Pengujian Klasifikasi *Mortality* pada *Google Colaboratory* Menggunakan *Dataset Preprocessed Image*

Pada pengujian klasifikasi *mortality* yang menggunakan dataset *preprocessed* terdapat 30 citra *CT scan* dada yang terdiri dari 10 citra kelas *Cured*, 10 citra kelas *Deceased* dan 10 citra *Unknown.* Hasil pengujian deteksi *CT scan* dada pada *google colaboratory* ditampilkan pada Tabel 4.16

Tabel 4.16Hasil pengujian deteksi menggunakan citra preprocessed

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujianke - | NamaCitra | Hasil Prediksi **(Tingkat Probabilitas)** | Keterangan |
| 1 | Cured001 | Deceased (1.00) | Salah |
| 2 | Cured002 | Unknown (1.00) | Salah |
| 3 | Cured003 | Deceased (1.00) | Salah |
| 4 | Cured004 | Deceased (1.00) | Salah |
| 5 | Cured005 | Unknown (1.00) | Salah |
| 6 | Cured006 | Unknown (1.00) | Salah |
| 7 | Cured007 | Deceased (1.00) | Salah |
| 8 | Cured008 | Deceased (1.00) | Salah |
| 9 | Cured009 | Deceased (1.00) | Salah |
| 10 | Cured010 | Deceased (1.00) | Salah |
| 11 | Deceased001 | Deceased (1.00) | Benar |
| 12 | Deceased002 | Unknown (1.00) | Salah |
| 13 | Deceased003 | Unknown (1.00) | Salah |
| 14 | Deceased004 | Deceased (1.00) | Benar |
| 15 | Deceased005 | Deceased (1.00) | Benar |
| 16 | Deceased006 | Deceased (1.00) | Benar |
| 17 | Deceased007 | Unknown (1.00) | Salah |
| 18 | Deceased008 | Unknown (1.00) | Salah |
| 19 | Deceased009 | Unknown (1.00) | Salah |
| 20 | Deceased010 | Unknown (1.00) | Salah |
| 21 | Unknown001 | Unknown (1.00) | Benar |
| 22 | Unknown002 | Unknown (1.00) | Benar |
| 23 | Unknown003 | Unknown (1.00) | Benar |
| 24 | Unknown004 | Unknown (1.00) | Benar |
| 25 | Unknown005 | Unknown (1.00) | Benar |
| 26 | Unknown006 | Unknown (1.00) | Benar |
| 27 | Unknown007 | Unknown (1.00) | Benar |
| 28 | Unknown008 | Unknown (1.00) | Benar |
| 29 | Unknown009 | Unknown (1.00) | Benar |
| 30 | Unknown010 | Unknown (1.00) | Benar |

Berdasarkan tabel hasil pengujian di atas dapat diketahui bahwa pada pengujian klasifikasi *mortality* menggunakan dataset *preprocessed* sebagian besar nilai pengujiannya masih bernilai salah. Untuk mengetahui tingkat keberhasilan pengujian, dilakukan perhitungan persentase keberhasilan pengujian sebagai berikut

|  |  |
| --- | --- |
|  | (4.5) |

Dari hasil perhitungan tersebut dapat disimpulkan bahwa hasil pengujian klasifikasi *mortality* dengan menggunakan citra *preprocessed* belum cukup akurat untuk melakukan deteksi citra *CT Scan* dada untuk identifikasi *mortality*, yaitu dengan persentase akurasi hanya sebesar 47%.

# SARAN DAN KESIMPULAN

## Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, maka diperoleh kesimpulan sebagai berikut.

* + - 1. Perancangan arsitektur CNN dari *VGG16* dan *ResNet50* untuk klasifikasi *CT Scan* dada dilakukan dengan metode ekstraksi fitur agar dapat memanfaatkan model yang telah dilatih sehingga menghasilkan model yang baik.
      2. Proses pelatihan dan validasi model klasifikasi *CT Scan* dada untuk deteksi *Covid-19* pada infrastruktur *Google Colaboratory* menggunakan *framework Keras* dan *Tensorflow* untuk menghemat waktu dan mendapatkan hasil yang baik.
      3. Proses klasifikasi *CT Scan* dada untuk *Covid-19* dibagi menjadi 4 sistem klasifikasi, yaitu *Lung Parenchyma Classifier, Covid Positivity Classifier*, *Risk Classifier* dan *Mortality Classifier.*
      4. Dataset pada penelitian ini dibagi menjadi 2 untuk masing – masing sistem klasifikasi, yaitu *original dataset* dan *preprocessed dataset*.
      5. Hasil pelatihan dan validasi terbaik pada klasifikasi *lung parenchyma classifier* menggunakan model VGG16 dan *original dataset* , dengan nilai akurasi pelatihan sebesar 0.9990 dan nilai validasi pelatihan sebesar 0.9918, sedangkan untuk nilai kesalahan akurasi sebesar 0.0121 dan kesalahan validasi sebesar 0.0180.
      6. Hasil pelatihan dan validasi terbaik pada *covid positivity classifier* menggunakan model *VGG16* dan *original dataset*, dengan nilai akurasi pelatihan sebesar 0.9857 dan nilai validasi pelatihan sebesar 0.5864, sedangkan untuk nilai kesalahan akurasi sebesar 0.0373 dan kesalahan validasi sebesar 2.8761
      7. Hasil pelatihan dan validasi terbaik pada *risk classifier* menggunakan model *VGG16* dan *original dataset*, dengan nilai akurasi pelatihan sebesar 0.9960 dan nilai validasi pelatihan sebesar 0.8764, sedangkan untuk nilai kesalahan akurasi sebesar 0.0122 dan kesalahan validasi sebesar 0.6217.
      8. Hasil pelatihan dan validasi terbaik pada *mortality classifier* menggunakan model *VGG16* dan *original dataset*, dengan nilai akurasi pelatihan sebesar 0.7120 dan nilai validasi pelatihan sebesar 0.4206, sedangkan untuk nilai kesalahan akurasi sebesar 0.6858 dan kesalahan validasi sebesar 1.4456.
      9. Model yang digunakan untuk deteksi *Covid-19* menggunakan citra *CT Scan* dada merupakan model terbaik untuk setiap sistem klasifikasi, yaitu *VGG16* dengan menggunakan *original dataset.*
      10. Hasil pengujian terbaik pada *google colaboratory* dan aplikasi web untuk *lung parenchyma classifier* menggunakan *preprocessed dataset*, dengan nilai akurasi sebesar 87%.
      11. Hasil pengujian terbaik pada *google colaboratory* dan aplikasi web untuk *covid positivity classifier* menggunakan *original dataset*, dengan nilai akurasi sebesar 75%.
      12. Hasil pengujian terbaik pada *google colaboratory* dan aplikasi web untuk *risk classifier* menggunakan *original dataset*, dengan nilai akurasi sebesar 60%.
      13. Hasil pengujian terbaik pada *google colaboratory* dan aplikasi web untuk *mortality classifier* menggunakan *original dataset*, dengan nilai akurasi sebesar 73%.
      14. Pengujian pada *google colaboratory* dan aplikasi web untuk deteksi *Covid-19* menggunakan citra *CT scan* dada dengan menggunakan *original dataset* memiliki nilai akurasi lebih baik jika dibandingkan dengan menggunakan *preprocessed dataset.*

## Saran

Saran yang perlu diperhatikan dan dikembangkan pada tugas akhir ini agar lebih baik lagi sebagai berikut

Sistem klasifikasi pada penelitian ini dapat ditambahkan dengan menggunakan dataset *Clinical Features* untuk metode DNN.

Nilai akurasi validasi dan kesalahan validasi pada setiap model agar dapat disempurnakan lagi untuk meningkatkan keakuratan dalam proses pengujian.

Rancangan aplikasi web yang telah dibangun dapat dikembangkan dengan menambahkan fitur – fitur seperti fitur penyimpanan data dan transfer data ke perangkat lain.

# DAFTAR PUSTAKA

[1] Q. Li *et al.*, “Early Transmission Dynamics in Wuhan, China, of Novel Coronavirus–Infected Pneumonia,” *N. Engl. J. Med.*, vol. 382, no. 13, pp. 1199–1207, 2020, doi: 10.1056/nejmoa2001316.

[2] Z. Y. Zu *et al.*, “H13. Coronavirus Disease 2019 (COVID-19): A Perspective from ChinaZu, Z. Y., Jiang, M. D., Xu, P. P., Chen, W., Ni, Q. Q., Lu, G. M., & Zhang, L. J. (2020). H13. Coronavirus Disease 2019 (COVID-19): A Perspective from China. Radiology, 200490. https://doi,” *Radiology*, vol. 2019, p. 200490, 2020.

[3] W. Ning *et al.*, *iCTCF: an integrative resource of chest computed tomography images and clinical features of patients with COVID-19 pneumonia*. 2020.

[4] F. Chollet, *Deep Learning with Python*. Shelter Island: Manning Publications Co, 2017.

[5] W. S. Eka Putra, “Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101,” *J. Tek. ITS*, vol. 5, no. 1, 2016, doi: 10.12962/j23373539.v5i1.15696.

[6] Jepri, “IDENTIFIKASI PENYAKIT PADA DAUN TOMAT DAN DAUN SINGKONG BERDASARKAN CITRA DAUN MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) BERBASIS ANDROID,” 2019.

[7] A. G. Anugerah and D. Informatika, “KLASIFIKASI TINGKAT KEGANASAN KANKER PARU-PARU PADA COMPUTED TOMOGRAPHY ( CT ) SCAN MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK,” 2018.

[8] R. Munir, *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Bandung: Informatika, 2004.

[9] Xenonstack, “Log Analytics With Deep Learning And Machine Learning,” 2017. https://medium.com/@xenonstack/loganalytics-with-deep-learning-and-machine-learning-20a1891ff70e (accessed Nov. 11, 2020).

[10] Sofyan, “Pengenalan Convolutional Neural Network – Part 1,” 2019. http://sofyantandungan.com/pengenalan-convolutional-neural-network-part-1/ (accessed Nov. 11, 2020).

[11] R. Okta, “MobileNet: Deteksi Objek pada Platform Mobile,” 2018. https://medium.com/nodeflux/mobilenet-deteksi-objek-pada-platform-mobile-bbbf3806e4b3 (accessed Nov. 15, 2020).

[12] A. G. Howard *et al.*, “MobileNets: Efficient convolutional neural networks for mobile vision applications,” *arXiv*, no. April 2017, 2017.

[13] S. Materials, K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition A. Object Detection Baselines,” pp. 9–12, 2015, [Online]. Available: http://host.robots.ox.ac.uk:8080/anonymous/3OJ4OJ.html,.

[14] Neurohive, “VGG16 – Convolutional Network for Classification and Detection,” 2018. https://neurohive.io/en/popular-networks/vgg16/ (accessed Nov. 15, 2020).

[15] F. Nashrullah, S. Adhi, and G. Budiman, “Investigasi Parameter Epoch Pada Arsitektur ResNet- 50 Untuk Klasifikasi Pornografi,” *J. Comput. Electron. Telecommun.*, 2020.

[16] J. Paul Mueller and L. Massaron, “What is Google Colaboratory?” https://www.dummies.com/programming/python/working-with-google-colaboratory-notebooks/ (accessed Nov. 15, 2020).

[17] A. Rahman, “Belajar Tensorflow.js,” 2019. https://alfian-pesan.medium.com/belajar-tensorflow-js-a9753f1237a1#:~:text=TensorFlowJs ini merupakan tool JavaScript,merupakan ML library untuk phyton. (accessed Feb. 21, 2021).

[18] R. Hidayatullah, “Pembuatan Desain Website Sebagai Penunjang Company Profile CV. Hensindo.,” pp. 11–25, 2016, [Online]. Available: http://sir.stikom.edu/id/eprint/2329/5/BAB\_III.pdf.

[19] D. Lavarino and W. Yustanti, “RANCANG BANGUN E – VOTING BERBASIS WEBSITE DI UNIVERSITAS NEGERI SURABAYA,” *J. Manaj. Inform.*, vol. 6, pp. 72–81, 2016.

[20] T. Haryanto, “Pengenalan dan Sintaks Dasar CSS,” 2016. https://www.codepolitan.com/pengenalan-dan-sintaks-dasar-css (accessed Feb. 16, 2021).

[21] O. Pahlevi, A. Mulyani, and M. Khoir, “Sistem Informasi Inventori Barang Menggunakan Metode Object Oriented Di Pt. Livaza Teknologi Indonesia Jakarta,” *J. PROSISKO*, vol. 5, no. 1, 2018, [Online]. Available: https://livaza.com/.

[22] M. Clinic, “CT Scan.” https://www.mayoclinic.org/tests-procedures/ct-scan/about/pac-20393675 (accessed Nov. 15, 2020).

[23] Kemenkes RI, “QnA : Pertanyaan dan Jawaban Terkait COVID-19.” https://covid19.kemkes.go.id/qna-pertanyaan-dan-jawaban-terkait-covid-19/#Apakah\_Coronavirus\_dan\_COVID-19\_itu (accessed Nov. 15, 2020).

[24] World Health Organization, “Deteksi antigen dalam diagnosis infeksi SARS-CoV-2 menggunakan imunoasai cepat,” no. September, 2020, [Online]. Available: https://www.who.int/docs/default-source/searo/indonesia/covid19/deteksi-antigen-dalam-diagnosis-infeksi-sars-cov-2-menggunakan-imunoasai-cepat.pdf?sfvrsn=222f2be3\_2.

# LAMPIRAN

1. Kode Sumber Sistem Klasifikasi pada *Google Colaboratory*
   1. Kode Sumber *Lung Parenchyma Classifier* dengan arsitektur *ResNet50* dan *original dataset* pada Google Colaboratory.

|  |
| --- |
| Mengimpor Dataset dari Google Drive **from** **google.colab** **import** drive  drive.mount('/content/gdrive') Mengimpor fungsi library **from** **\_\_future\_\_** **import** absolute\_import, division, print\_function, unicode\_literals  **try**:  *# The %tensorflow\_version magic only works in colab.*  %tensorflow\_version 2.x  **except** **Exception**:  **pass**  **import** **numpy** **as** **np**  **import** **math**, **os**, **sys**  **import** **itertools**  **import** **os**  **import** **matplotlib.pyplot** **as** **plt**  plt.style.use('default')  **from** **scipy** **import** ndimage  **from** **skimage** **import** measure, morphology  **from** **skimage.io** **import** imsave, imread  **from** **skimage.filters** **import** threshold\_otsu  **from** **skimage.transform** **import** resize  **from** **skimage** **import** io  **from** **skimage.transform** **import** rotate, AffineTransform, warp  **from** **skimage** **import** img\_as\_ubyte  **from** **skimage.util** **import** random\_noise  **import** **tensorflow** **as** **tf**  **from** **sklearn** **import** svm, datasets  **from** **sklearn.metrics** **import** confusion\_matrix  **import** **pandas** **as** **pd**  **import** **random** Membuat Dataframe untuk Dataset mypath= 'gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/CT/Original/'  file\_name = []  tag = []  full\_path = []  **for** path, subdirs, files **in** os.walk(mypath):  **for** name **in** files:  full\_path.append(os.path.join(path, name))  tag.append(path.split('/')[-1])  file\_name.append(name)  *# memasukan variabel yang sudah dikumpulkan pada looping di atas menjadi sebuah dataframe agar rapih*  df = pd.DataFrame({"path":full\_path,'file\_name':file\_name,"tag":tag})  df.groupby(['tag']).size() Membagi Dataset ke dalam Bentuk Train Data dan Test Data *#load library untuk train test split*  **from** **sklearn.model\_selection** **import** train\_test\_split  *#variabel yang digunakan pada pemisahan data ini*  X= df['path']  y= df['tag']  *# split dataset awal menjadi data train dan test*  X\_train, X\_test, y\_train, y\_test = train\_test\_split(  X, y, test\_size=0.2, random\_state=300)  *# menyatukan kedalam masing-masing dataframe*  df\_tr = pd.DataFrame({'path':X\_train  ,'tag':y\_train  ,'set':'train'})  df\_te = pd.DataFrame({'path':X\_test  ,'tag':y\_test  ,'set':'test'})  print('train size', len(df\_tr))  print('test size', len(df\_te))  *# melihat proporsi pada masing masing set apakah sudah ok atau masih ada yang ingin diubah*  df\_all = df\_tr.append([df\_te]).reset\_index(drop=1)\  print('===================================================== **\n**')  print(df\_all.groupby(['set','tag']).size(),'**\n**')  print('===================================================== **\n**')  *#cek sample datanya*  df\_all.sample(3) Membuat Folder Baru untuk Dataset *## create folders*  os.makedirs('Dataset/') Menyalin Dataset ke dalam Folder Dataset Baru datasource\_path = "/content/gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/CT/Original/"  dataset\_path = "Dataset/"  **for** index, row **in** tq(df\_all.iterrows()):    *#detect filepath*  file\_path = row['path']  **if** os.path.exists(file\_path) == **False**:  file\_path = os.path.join(datasource\_path,row['tag'],row['image'].split('.')[0])    *#make folder destination dirs*  **if** os.path.exists(os.path.join(dataset\_path,row['set'],row['tag'])) == **False**:  os.makedirs(os.path.join(dataset\_path,row['set'],row['tag']))    *#define file dest*  destination\_file\_name = file\_path.split('/')[-1]  file\_dest = os.path.join(dataset\_path,row['set'],row['tag'],destination\_file\_name)    *#copy file from source to dest*  **if** os.path.exists(file\_dest) == **False**:  shutil.copy2(file\_path,file\_dest) Pre-processing dan Augmentasi Data datagen = tf.keras.preprocessing.image.ImageDataGenerator(  rescale=1./255,  validation\_split = 0.2,  rotation\_range = 30,  horizontal\_flip = **True**,  shear\_range = 0.2,  zoom\_range = 0.1,  vertical\_flip = **True**,  fill\_mode = "nearest") Mengimpor Dataset *#Memuat semua gambar ke memori untuk pertama kali*  *#Memuat dataset pelatihan*  IMAGE\_SIZE = 224  BATCH\_SIZE = 66  base\_dir = os.path.join('Dataset/train/')  train\_generator = datagen.flow\_from\_directory(  base\_dir,  target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),  batch\_size=BATCH\_SIZE,  subset='training',  class\_mode= 'categorical')  val\_generator = datagen.flow\_from\_directory(  base\_dir,  target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),  batch\_size=BATCH\_SIZE,  subset='validation',  class\_mode= 'categorical')  *#Memuat dataset pengujian*  X\_test = []  y\_test = []  labels = ['NiCT',  'nCT',  'pCT',]  **for** i,label **in** enumerate(labels):  folder = os.path.join("Dataset/test",label)  files = sorted(os.listdir(folder))  files = [x **for** x **in** files **if** x.endswith(".jpg")]  **for** k,file **in** enumerate(files):  image\_path = os.path.join(folder, file)    image = imread(image\_path)/255.  image = resize(image,(224,224))  X\_test.append(image)  category = os.path.split(folder)[-1]  y\_test.append(i)  X\_test = np.array(X\_test)  y\_test = np.array(y\_test)  *#Menampilkan bentuk dari masing-masing dataset*  **for** image\_batch, label\_batch **in** train\_generator:  **break**  print("Bentuk array dari dataset train (pelatihan) adalah:", image\_batch.shape,label\_batch.shape)  **for** image\_batch, label\_batch **in** val\_generator:  **break**  print("Bentuk array dari dataset validation (validasi) adalah:", image\_batch.shape,label\_batch.shape)  print("Bentuk array dari dataset test (pengujian) adalah:", X\_test.shape,y\_test.shape) Menyimpan label print (train\_generator.class\_indices)  labels\_txt = '**\n**'.join(sorted(train\_generator.class\_indices.keys()))  **with** open('labels.txt', 'w') **as** f:  f.write(labels\_txt) Membuat model CNN IMG\_SHAPE = (224, 224, 3)  *# Membuat model dasar (base model) dari pre-trained model MobileNet*  base\_model = tf.keras.applications.ResNet50(input\_shape=IMG\_SHAPE,  include\_top=**False**,  weights='imagenet')  base\_model.trainable = **False**  base\_model.summary()  **import** **keras**  **from** **keras** **import** backend **as** K  **from** **keras.models** **import** Sequential  **from** **keras** **import** layers  **from** **keras.utils.np\_utils** **import** to\_categorical  **from** **sklearn.model\_selection** **import** train\_test\_split  model = tf.keras.Sequential([  base\_model,  tf.keras.layers.Conv2D(1024, 3, activation='relu'),  tf.keras.layers.Dropout(0.2),  tf.keras.layers.Conv2D(2048, 3, activation='relu'),  tf.keras.layers.Dropout(0.2),  tf.keras.layers.GlobalAveragePooling2D(),  tf.keras.layers.Dense(3, activation='softmax')  ]  )  model.compile("adam",loss="categorical\_crossentropy",metrics=["acc"])  model.summary() Melatih Model history = model.fit\_generator(train\_generator,  epochs=50,  validation\_data=val\_generator) Menampilkan Hasil Pelatihan plt.plot(history.history["acc"],label="Akurasi Pelatihan")  plt.plot(history.history["val\_acc"],label="Validasi Akurasi")  plt.legend()  plt.show()  plt.plot(history.history["loss"],label="Kesalahan Pelatihan")  plt.plot(history.history["val\_loss"],label="Validasi Kesalahan")  plt.legend()  plt.show() Memuat Dataset Pengujian *#Menampilkan matriks yang benar dan matriks hasil prediksi*  *#Label yang benar*  y\_true = np.argmax(y\_test2,axis=1)  *#Label prediksi*  Y\_pred = model.predict(X\_test)  y\_pred = np.argmax(Y\_pred, axis=1)  print(y\_true)  print(y\_pred) Menggunakan Model print('Number of trainable variables = **{}**'.format(len(model.trainable\_variables))) Memprediksi Citra Secara Individu n = 44 *#Jangan melampaui (nilai dari gambar test - 1)*  plt.imshow(X\_test[n])  plt.show()  true\_label = np.argmax(y\_test2,axis=1)[n]  print("Label yang benar adalah:",true\_label,":",labels[true\_label])  prediction = model.predict(X\_test[n][np.newaxis,...])[0]  print("Nilai yang diprediksi adalah:",prediction)  predicted\_label = np.argmax(prediction)  print("Label yang diprediksi adalah:",predicted\_label,":",labels[predicted\_label])  **if** true\_label == predicted\_label:  print("Prediksi benar")  **else**:  print("Prediksi salah") Confusion Matrix **from** **sklearn.metrics** **import** confusion\_matrix  **from** **sklearn.utils.multiclass** **import** unique\_labels  **def** plot\_confusion\_matrix(y\_true, y\_pred, classes,  normalize=**False**,  title=**None**,  cmap=plt.cm.Blues):  *"""*  *This function prints and plots the confusion matrix.*  *Normalization can be applied by setting `normalize=True`.*  *"""*  **if** **not** title:  **if** normalize:  title = 'Normalized confusion matrix'  **else**:  title = 'Confusion matrix, without normalization'  *# Compute confusion matrix*  cm = confusion\_matrix(y\_true, y\_pred)  *# Only use the labels that appear in the data*  *#classes = classes[unique\_labels(y\_true, y\_pred)]*  **if** normalize:  cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]  print("Normalized confusion matrix")  **else**:  print('Confusion matrix, without normalization')  print(cm)  fig, ax = plt.subplots(figsize=(5,5))  im = ax.imshow(cm, interpolation='nearest', cmap=cmap)  *#ax.figure.colorbar(im, ax=ax)*  *# We want to show all ticks...*  ax.set(xticks=np.arange(cm.shape[1]),  yticks=np.arange(cm.shape[0]),  *# ... and label them with the respective list entries*  xticklabels=classes, yticklabels=classes,  title=title,  ylabel='Label Benar',  xlabel='Label Prediksi')  *# Rotate the tick labels and set their alignment.*  plt.setp(ax.get\_xticklabels(), rotation=45, ha="right",  rotation\_mode="anchor")  *# Loop over data dimensions and create text annotations.*  fmt = '.2f' **if** normalize **else** 'd'  thresh = cm.max() / 2.  **for** i **in** range(cm.shape[0]):  **for** j **in** range(cm.shape[1]):  ax.text(j, i, format(cm[i, j], fmt),  ha="center", va="center",  color="white" **if** cm[i, j] > thresh **else** "black")  fig.tight\_layout()  **return** ax  np.set\_printoptions(precision=2)  plot\_confusion\_matrix(y\_true, y\_pred, classes=labels, normalize=**True**,  title='Normalized confusion matrix')  **from** **sklearn.metrics** **import** classification\_report  print (classification\_report(y\_true, y\_pred)) Menyimpan dan mengkonversi Model ke “.h5” model.save('LP.h5') |

* 1. Kode sumber *Lung Parenchyma Classifier* dengan arsitektur *VGG16* dan *preprocessed dataset* pada *Google Colaboratory*

|  |
| --- |
| Mengimpor Dataset dari Google Drive **from** **google.colab** **import** drive  drive.mount('/content/gdrive') Mengimpor fungsi library **from** **\_\_future\_\_** **import** absolute\_import, division, print\_function, unicode\_literals  **try**:  *# The %tensorflow\_version magic only works in colab.*  %tensorflow\_version 2.x  **except** **Exception**:  **pass**  **import** **numpy** **as** **np**  **import** **math**, **os**, **sys**  **import** **itertools**  **import** **os**  **import** **matplotlib.pyplot** **as** **plt**  plt.style.use('default')  **from** **scipy** **import** ndimage  **from** **skimage** **import** measure, morphology  **from** **skimage.io** **import** imsave, imread  **from** **skimage.filters** **import** threshold\_otsu  **from** **skimage.transform** **import** resize  **from** **skimage** **import** io  **from** **skimage.transform** **import** rotate, AffineTransform, warp  **from** **skimage** **import** img\_as\_ubyte  **from** **skimage.util** **import** random\_noise  **import** **tensorflow** **as** **tf**  **from** **sklearn** **import** svm, datasets  **from** **sklearn.metrics** **import** confusion\_matrix  **import** **pandas** **as** **pd**  **import** **random** Membuat Dataframe untuk Dataset mypath= '/content/gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/CT/Pre-processed/'    file\_name = []  tag = []  full\_path = []  **for** path, subdirs, files **in** os.walk(mypath):  **for** name **in** files:  full\_path.append(os.path.join(path, name))  tag.append(path.split('/')[-1])  file\_name.append(name)  *# memasukan variabel yang sudah dikumpulkan pada looping di atas menjadi sebuah dataframe agar rapih*  df = pd.DataFrame({"path":full\_path,'file\_name':file\_name,"tag":tag})  df.groupby(['tag']).size() Membagi Dataset ke dalam Bentuk Train Data dan Test Data *#load library untuk train test split*  **from** **sklearn.model\_selection** **import** train\_test\_split  *#variabel yang digunakan pada pemisahan data ini*  X= df['path']  y= df['tag']  *# split dataset awal menjadi data train dan test*  X\_train, X\_test, y\_train, y\_test = train\_test\_split(  X, y, test\_size=0.2, random\_state=300)  *# menyatukan kedalam masing-masing dataframe*  df\_tr = pd.DataFrame({'path':X\_train  ,'tag':y\_train  ,'set':'train'})  df\_te = pd.DataFrame({'path':X\_test  ,'tag':y\_test  ,'set':'test'})  print('train size', len(df\_tr))  print('test size', len(df\_te))  *# melihat proporsi pada masing masing set apakah sudah ok atau masih ada yang ingin diubah*  df\_all = df\_tr.append([df\_te]).reset\_index(drop=1)\  print('===================================================== **\n**')  print(df\_all.groupby(['set','tag']).size(),'**\n**')  print('===================================================== **\n**')  *#cek sample datanya*  df\_all.sample(3) Membuat Folder Baru untuk Dataset *## create folders*  os.makedirs('Dataset/') Menyalin Dataset ke dalam Folder Dataset Baru datasource\_path = "/content/gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/CT/Pre-processed/"  dataset\_path = "Dataset/"  **for** index, row **in** tq(df\_all.iterrows()):    *#detect filepath*  file\_path = row['path']  **if** os.path.exists(file\_path) == **False**:  file\_path = os.path.join(datasource\_path,row['tag'],row['image'].split('.')[0])    *#make folder destination dirs*  **if** os.path.exists(os.path.join(dataset\_path,row['set'],row['tag'])) == **False**:  os.makedirs(os.path.join(dataset\_path,row['set'],row['tag']))    *#define file dest*  destination\_file\_name = file\_path.split('/')[-1]  file\_dest = os.path.join(dataset\_path,row['set'],row['tag'],destination\_file\_name)    *#copy file from source to dest*  **if** os.path.exists(file\_dest) == **False**:  shutil.copy2(file\_path,file\_dest) Pre-processing dan Augmentasi Data datagen = tf.keras.preprocessing.image.ImageDataGenerator(  rescale=1./255,  validation\_split = 0.2,  rotation\_range = 30,  horizontal\_flip = **True**,  shear\_range = 0.2,  zoom\_range = 0.1,  vertical\_flip = **True**,  fill\_mode = "nearest") Mengimpor Dataset *#Memuat semua gambar ke memori untuk pertama kali*  *#Memuat dataset pelatihan*  IMAGE\_SIZE = 224  BATCH\_SIZE = 66  base\_dir = os.path.join('Dataset/train/')  train\_generator = datagen.flow\_from\_directory(  base\_dir,  target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),  batch\_size=BATCH\_SIZE,  subset='training',  class\_mode= 'categorical')  val\_generator = datagen.flow\_from\_directory(  base\_dir,  target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),  batch\_size=BATCH\_SIZE,  subset='validation',  class\_mode= 'categorical')  *#Memuat dataset pengujian*  X\_test = []  y\_test = []  labels = ['NiCT',  'nCT',  'pCT',]  **for** i,label **in** enumerate(labels):  folder = os.path.join("Dataset/test",label)  files = sorted(os.listdir(folder))  files = [x **for** x **in** files **if** x.endswith(".jpg")]  **for** k,file **in** enumerate(files):  image\_path = os.path.join(folder, file)    image = imread(image\_path)/255.  image = resize(image,(224,224))  X\_test.append(image)  category = os.path.split(folder)[-1]  y\_test.append(i)  X\_test = np.array(X\_test)  y\_test = np.array(y\_test)  *#Menampilkan bentuk dari masing-masing dataset*  **for** image\_batch, label\_batch **in** train\_generator:  **break**  print("Bentuk array dari dataset train (pelatihan) adalah:", image\_batch.shape,label\_batch.shape)  **for** image\_batch, label\_batch **in** val\_generator:  **break**  print("Bentuk array dari dataset validation (validasi) adalah:", image\_batch.shape,label\_batch.shape)  print("Bentuk array dari dataset test (pengujian) adalah:", X\_test.shape,y\_test.shape) Menyimpan label print (train\_generator.class\_indices)  labels\_txt = '**\n**'.join(sorted(train\_generator.class\_indices.keys()))  **with** open('labels.txt', 'w') **as** f:  f.write(labels\_txt) Membuat model CNN IMG\_SHAPE = (224, 224, 3)  *# Membuat model dasar (base model) dari pre-trained model VGG16*  base\_model = tf.keras.applications.VGG16(input\_shape=IMG\_SHAPE,  include\_top=**False**,  weights='imagenet')  base\_model.trainable = **False**  base\_model.summary()  **import** **keras**  **from** **keras** **import** backend **as** K  **from** **keras.models** **import** Sequential  **from** **keras** **import** layers  **from** **keras.utils.np\_utils** **import** to\_categorical  **from** **sklearn.model\_selection** **import** train\_test\_split  model = tf.keras.Sequential([  base\_model,  tf.keras.layers.Conv2D(1024, 3, activation='relu'),  tf.keras.layers.Dropout(0.2),  tf.keras.layers.Conv2D(2048, 3, activation='relu'),  tf.keras.layers.Dropout(0.2),  tf.keras.layers.GlobalAveragePooling2D(),  tf.keras.layers.Dense(3, activation='softmax')  ]  )  model.compile("adam",loss="categorical\_crossentropy",metrics=["acc"])  model.summary() Melatih Model history = model.fit\_generator(train\_generator,  epochs=50,  validation\_data=val\_generator) Menampilkan Hasil Pelatihan plt.plot(history.history["acc"],label="Akurasi Pelatihan")  plt.plot(history.history["val\_acc"],label="Validasi Akurasi")  plt.legend()  plt.show()  plt.plot(history.history["loss"],label="Kesalahan Pelatihan")  plt.plot(history.history["val\_loss"],label="Validasi Kesalahan")  plt.legend()  plt.show() Memuat Dataset Pengujian *#Menampilkan matriks yang benar dan matriks hasil prediksi*  *#Label yang benar*  y\_true = np.argmax(y\_test2,axis=1)  *#Label prediksi*  Y\_pred = model.predict(X\_test)  y\_pred = np.argmax(Y\_pred, axis=1)  print(y\_true)  print(y\_pred) Menggunakan Model print('Number of trainable variables = **{}**'.format(len(model.trainable\_variables))) Memprediksi Citra Secara Individu n = 44 *#Jangan melampaui (nilai dari gambar test - 1)*  plt.imshow(X\_test[n])  plt.show()  true\_label = np.argmax(y\_test2,axis=1)[n]  print("Label yang benar adalah:",true\_label,":",labels[true\_label])  prediction = model.predict(X\_test[n][np.newaxis,...])[0]  print("Nilai yang diprediksi adalah:",prediction)  predicted\_label = np.argmax(prediction)  print("Label yang diprediksi adalah:",predicted\_label,":",labels[predicted\_label])  **if** true\_label == predicted\_label:  print("Prediksi benar")  **else**:  print("Prediksi salah") Confusion Matrix **from** **sklearn.metrics** **import** confusion\_matrix  **from** **sklearn.utils.multiclass** **import** unique\_labels  **def** plot\_confusion\_matrix(y\_true, y\_pred, classes,  normalize=**False**,  title=**None**,  cmap=plt.cm.Blues):  *"""*  *This function prints and plots the confusion matrix.*  *Normalization can be applied by setting `normalize=True`.*  *"""*  **if** **not** title:  **if** normalize:  title = 'Normalized confusion matrix'  **else**:  title = 'Confusion matrix, without normalization'  *# Compute confusion matrix*  cm = confusion\_matrix(y\_true, y\_pred)  *# Only use the labels that appear in the data*  *#classes = classes[unique\_labels(y\_true, y\_pred)]*  **if** normalize:  cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]  print("Normalized confusion matrix")  **else**:  print('Confusion matrix, without normalization')  print(cm)  fig, ax = plt.subplots(figsize=(5,5))  im = ax.imshow(cm, interpolation='nearest', cmap=cmap)  *#ax.figure.colorbar(im, ax=ax)*  *# We want to show all ticks...*  ax.set(xticks=np.arange(cm.shape[1]),  yticks=np.arange(cm.shape[0]),  *# ... and label them with the respective list entries*  xticklabels=classes, yticklabels=classes,  title=title,  ylabel='Label Benar',  xlabel='Label Prediksi')  *# Rotate the tick labels and set their alignment.*  plt.setp(ax.get\_xticklabels(), rotation=45, ha="right",  rotation\_mode="anchor")  *# Loop over data dimensions and create text annotations.*  fmt = '.2f' **if** normalize **else** 'd'  thresh = cm.max() / 2.  **for** i **in** range(cm.shape[0]):  **for** j **in** range(cm.shape[1]):  ax.text(j, i, format(cm[i, j], fmt),  ha="center", va="center",  color="white" **if** cm[i, j] > thresh **else** "black")  fig.tight\_layout()  **return** ax  np.set\_printoptions(precision=2)  plot\_confusion\_matrix(y\_true, y\_pred, classes=labels, normalize=**True**,  title='Normalized confusion matrix')  **from** **sklearn.metrics** **import** classification\_report  print (classification\_report(y\_true, y\_pred)) Menyimpan dan mengkonversi Model ke “.h5” model.save('LP.h5') |

Selengkapnya dapat dilihat pada halaman berikut :

<https://github.com/Soedirman-Machine-Learning/CT-Scan-Classification-for-COVID19/tree/main/.ipynb%20checkpoints>

1. Kode Sumber Pengujian Sistem Klasifikasi pada *Google Colaboratory*

|  |
| --- |
| Import library **import** **os**  **import** **numpy** **as** **np**  **import** **matplotlib.pyplot** **as** **plt**  **import** **tensorflow** **as** **tf**  **from** **tensorflow.keras.preprocessing.image** **import** ImageDataGenerator, array\_to\_img, img\_to\_array, load\_img Import dataset from Google Drive **from** **google.colab** **import** drive  drive.mount('/content/gdrive') Load model *# load saved model and try predictions*  *# load CT model*  CT\_model = tf.keras.models.load\_model('/content/gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/Model/model\_CT.h5')  CT\_model.layers[0].input\_shape  *# load risk model*  risk\_model = tf.keras.models.load\_model('/content/gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/Model/risk.h5')  risk\_model.layers[0].input\_shape  *# load mortality model*  mor\_model = tf.keras.models.load\_model('/content/gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/Model/mor.h5')  mor\_model.layers[0].input\_shape  *# load covid positivity model*  CP\_model = tf.keras.models.load\_model('/content/gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/Model/CP.h5')  CP\_model.layers[0].input\_shape Use model to predict image *# image prediction*  img1 = os.path.join('/content/gdrive/Shareddrives/Soedirman-Machine-Learning/CT SCAN COVID-19/CT/Pre-processed/pCT/pCT106.jpg' )  img\_CT = load\_img(img1, target\_size=(224, 224))  x = img\_to\_array(img\_CT)  x = np.expand\_dims(x, axis=0)  images = np.vstack([x])  *# Perform prediction*  *# Predict image CT*  classes\_LP = ["NiCT", "nCT", "pCT"]  predictions\_LP = CT\_model.predict\_classes(images, batch\_size=10)  prob\_LP = CT\_model.predict\_proba(images)  classes\_CP = ["Negative", "Positive"]  predictions\_CP = CP\_model.predict\_classes(images, batch\_size=10)  prob\_CP = CP\_model.predict\_proba(images)  classes\_mor = ["Cured", "Deceased", "Unknown"]  predictions\_mor = mor\_model.predict\_classes(images, batch\_size=10)  prob\_mor = mor\_model.predict\_proba(images)  classes\_risk = ["Control", "Type I", "Type II"]  predictions\_risk = risk\_model.predict\_classes(images, batch\_size=10)  prob\_risk = risk\_model.predict\_proba(images)  *# display image and prediction*  plt.figure()  plt.subplot(121)  plt.title("Image CT Prediction")  plt.imshow(img\_CT)  **if** predictions\_LP == 2 **and** predictions\_CP == 1:  print("Lung Parenchyma Predicted class: " + str(classes\_LP[predictions\_LP[0]]))  print("Nilai probabilitasnya adalah:",prob\_LP)  print("Covid Positivity Predicted class: " + str(classes\_CP[predictions\_CP[0]]))  print("Nilai probabilitasnya adalah:",prob\_CP)  print("Risk Predicted class: " + str(classes\_risk[predictions\_risk[0]]))  print("Nilai probabilitasnya adalah:",prob\_risk)  print("Mortality Predicted class: " + str(classes\_mor[predictions\_mor[0]]))  print("Nilai probabilitasnya adalah:",prob\_mor)  **elif** predictions\_LP == 2 **and** predictions\_CP == 0:  print("Lung Parenchyma Predicted class: " + str(classes\_LP[predictions\_LP[0]]))  print("Nilai probabilitasnya adalah:",prob\_LP)  print("Covid Positivity Predicted class: " + str(classes\_CP[predictions\_CP[0]]))  print("Nilai probabilitasnya adalah:",prob\_CP)  **elif** predictions\_LP == 0 **or** predictions\_LP == 1:  print("Lung Parenchyma Predicted class: " + str(classes\_LP[predictions\_LP[0]]))  print("Nilai probabilitasnya adalah:",prob\_LP)  **return** images |

1. Kode Sumber Aplikasi Web Deteksi *Covid-19* menggunakan *CT Scan* Dada
   1. Flask\_app.py

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | from \_\_future\_\_ import division, print\_function | |  |  | |  | # coding=utf-8 | |  | import sys | |  | import glob | |  | import re | |  | import os | |  | from flask import Flask, render\_template, Response, url\_for, redirect | |  | from flask import request | |  |  | |  | # Keras | |  | from keras.preprocessing import image | |  | from keras.models import load\_model | |  | import h5py | |  |  | |  | # TF | |  | import numpy as np | |  | import tensorflow as tf | |  | from tensorflow import keras | |  |  | |  | print(tf.version.VERSION) | |  |  | |  | app = Flask(\_\_name\_\_) | |  |  | |  | BASE\_PATH = os.getcwd() | |  | UPLOAD\_PATH = os.path.join(BASE\_PATH,'static/upload/') | |  | MODEL\_PATH\_LP = os.path.join(BASE\_PATH, "static/models", "LP.h5") | |  | MODEL\_PATH\_CP = os.path.join(BASE\_PATH, "static/models", "CP.h5") | |  | MODEL\_PATH\_RISK = os.path.join(BASE\_PATH, "static/models", "risk.h5") | |  | MODEL\_PATH\_MOR = os.path.join(BASE\_PATH, "static/models", "mor.h5") | |  |  | |  | model\_LP = load\_model(MODEL\_PATH\_LP) | |  | model\_CP = load\_model(MODEL\_PATH\_CP) | |  | model\_risk = load\_model(MODEL\_PATH\_RISK) | |  | model\_mor = load\_model(MODEL\_PATH\_MOR) | |  | print("Model loaded") | |  |  | |  |  | |  | def model\_predict\_LP(img\_path, model): | |  | classes\_LP = ["NiCT", "nCT", "pCT"] | |  | img = image.load\_img(img\_path, target\_size=(224, 224)) | |  | img = image.img\_to\_array(img) | |  | img = np.expand\_dims(img, axis=0) | |  | result = model.predict(img) | |  | confidence = np.amax(result) | |  | text = str(classes\_LP[np.argmax(result)]) | |  | return text, confidence | |  | # for i in result: | |  | # confidence = np.amax(i) | |  | # text = classes\_LP[np.argmax(i)] | |  | # return text, confidence | |  |  | |  |  | |  | def model\_predict\_CP(img\_path, model): | |  | classes\_CP = ["Negative", "Positive"] | |  | img = image.load\_img(img\_path, target\_size=(224, 224)) | |  | img = image.img\_to\_array(img) | |  | img = np.expand\_dims(img, axis=0) | |  | result = model.predict(img) | |  | confidence = np.amax(result) | |  | text = str(classes\_CP[np.argmax(result)]) | |  | return text, confidence | |  |  | |  |  | |  | def model\_predict\_risk(img\_path, model): | |  | classes\_risk = ["Control", "Type I", "Type II"] | |  | img = image.load\_img(img\_path, target\_size=(224, 224)) | |  | img = image.img\_to\_array(img) | |  | img = np.expand\_dims(img, axis=0) | |  | result = model.predict(img) | |  | confidence = np.amax(result) | |  | text = str(classes\_risk[np.argmax(result)]) | |  | return text, confidence | |  |  | |  |  | |  | def model\_predict\_mor(img\_path, model): | |  | classes\_mor = ["Cured", "Deceased", "Unknown"] | |  | img = image.load\_img(img\_path, target\_size=(224, 224)) | |  | img = image.img\_to\_array(img) | |  | img = np.expand\_dims(img, axis=0) | |  | result = model.predict(img) | |  | confidence = np.amax(result) | |  | text = str(classes\_mor[np.argmax(result)]) | |  | return text, confidence | |  |  | |  | @app.errorhandler(404) | |  | def error404(error): | |  | message = "ERROR 404 OCCURED. Page not found. Please go the home page and try again" | |  | return render\_template("error.html", message=message) | |  |  | |  | @app.errorhandler(405) | |  | def error405(error): | |  | message = "Error 405, Method not found" | |  | return render\_template("error.html",message=message) | |  |  | |  | @app.errorhandler(500) | |  | def error500(error): | |  | message = "INTERNAL ERROR 500, Error occurs in the program" | |  | return render\_template("error.html",message=message) | |  |  | |  |  | |  | @app.route('/',methods=['GET','POST']) | |  | def index(): | |  | if request.method == "POST": | |  | upload\_file = request.files['image\_name'] | |  | filename = upload\_file.filename | |  | print('The filename that has been uploaded =',filename) | |  | # know the extension of filename | |  | # all only .jpg, .png, .jpeg | |  | ext = filename.split('.')[-1] | |  | print('The extension of the filename =',ext) | |  | if ext.lower() in ['png','jpg','jpeg']: | |  | pred\_type = 0 | |  | # saving the image | |  | path\_save = os.path.join(UPLOAD\_PATH,filename) | |  | upload\_file.save(path\_save) | |  | print('File saved succesfully') | |  | # Make prediction | |  | preds\_LP, conf\_LP = model\_predict\_LP(path\_save, model\_LP) | |  | conf\_LP = int(round(conf\_LP, 4) \* 100) | |  |  | |  | preds\_CP, conf\_CP = model\_predict\_CP(path\_save, model\_CP) | |  | conf\_CP = int(round(conf\_CP, 4) \* 100) | |  |  | |  | preds\_risk, conf\_risk = model\_predict\_risk(path\_save, model\_risk) | |  | conf\_risk = int(round(conf\_risk, 4) \* 100) | |  |  | |  | preds\_mor, conf\_mor = model\_predict\_mor(path\_save, model\_mor) | |  | conf\_mor = int(round(conf\_mor, 4) \* 100) | |  |  | |  | if preds\_LP == "pCT" and preds\_CP == "Positive": | |  | pred\_type = 3 | |  | elif preds\_LP == "pCT" and preds\_CP == "Negative": | |  | pred\_type = 2 | |  | elif preds\_LP != "pCT": | |  | pred\_type = 1 | |  | else: | |  | pred\_type = 0 | |  |  | |  | print("Prediksi: ", preds\_mor) | |  | print("Probabilitas: ", conf\_mor, "%") | |  | return render\_template("upload.html", | |  | prediction\_LP=preds\_LP, | |  | confidence\_LP=conf\_LP, | |  | prediction\_CP=preds\_CP, | |  | confidence\_CP=conf\_CP, | |  | prediction\_risk=preds\_risk, | |  | confidence\_risk=conf\_risk, | |  | prediction\_mor=preds\_mor, | |  | confidence\_mor=conf\_mor, | |  | extension=False, | |  | fileupload=True, | |  | prediction\_type=pred\_type, | |  | image\_filename=filename, | |  | ) | |  |  | |  | else: | |  | print('Use only the extension with .jpg, .png, .jpeg ') | |  | return render\_template('upload.html', extension=True,fileupload=False) | |  |  | |  | else: | |  | return render\_template('upload.html',fileupload=False,extension=False) | |  |  | |  | @app.route('/about/') | |  | def about(): | |  | return render\_template('about.html') | |  |  | |  | if \_\_name\_\_ == "\_\_main\_\_": | |  | app.run(debug=True) | |

* 1. Index.html

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | <!DOCTYPE html> | |  | <html> | |  |  | |  | <head> | |  | <title>Image Classification Flask App running with Machine Learning</title> | |  | <!-- Bootstrap 4 --> | |  | <link rel="stylesheet" href="/static/css/bootstrap.css"> | |  | <script src="/static/js/bootstrap.js"></script> | |  | <script src="/static/js/bootstrap.bundle.js"></script> | |  | <script src="/static/js/jquery-3.6.0.js"></script> | |  | </head> | |  |  | |  | <body> | |  | <nav class="navbar navbar-expand-lg navbar-dark" style="background-color: #032539;"> | |  | <a class="navbar-brand" href="/">Image Classification App</a> | |  | <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation"> | |  | <span class="navbar-toggler-icon"></span> | |  | </button> | |  | <div class="collapse navbar-collapse" id="navbarNavAltMarkup"> | |  | <div class="navbar-nav"> | |  | <a class="nav-item nav-link" href="/">Home <span class="sr-only">(current)</span></a> | |  | <a class="nav-item nav-link" href="/about/">About</a> | |  | </div> | |  | </div> | |  | </nav> | |  | <div class="container"> | |  | <br> | |  | <h4>CT-Scan Classification for Covid-19 Identification</h4> | |  | </div> | |  |  | |  |  | |  | {% block body %} {% endblock %} | |  |  | |  |  | |  |  | |  | <!-- Footer --> | |  | <div class="container"> | |  | <hr> | |  | <footer> | |  | Soedirman Machine Learning | |  | </footer> | |  | </div> | |  | </body> | |  |  | |  | </html> | |

* 1. upload.html

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | {% extends 'index.html' %} {% block body %} | |  |  | |  | <div class="container" id="myrow"> | |  | <ul class="nav nav-tabs"> | |  | <li class="nav-item"> | |  | <a class="nav-link active" id="uploadimg" href="#upload-image">Upload Image</a> | |  | </li> | |  | </ul> | |  |  | |  | <!--Tab panes--> | |  | <div> | |  |  | |  |  | |  | </div class="tab-content"> | |  | <div class="tab-pane active" id="upload-image" role="tabpanel" aria-labelledby="uploadimg"> | |  | <div class="row"> | |  | <div class="col"> | |  | <form action="#" method="POST" enctype="multipart/form-data"> | |  | <!--File input--> | |  | <p> | |  | <input type="file" name="image\_name" class="btn" style="background-color: #1c768f; color:azure;" required> | |  | </p> | |  | <!--Submit button--> | |  | <p> | |  | <input type="submit" value="submit" class="btn" style="background-color: #1c768f; color:azure"> | |  | </p> | |  |  | |  | <p><strong>Instruction :</strong> <br> Only upload file with extention ".png", ".jpg", ".jpeg" For other image extention you will <strong>redirect</strong> to error 404 page | |  |  | |  | </p> | |  |  | |  | </form> | |  |  | |  | </div> | |  | </div> | |  | </div> | |  | </div> | |  |  | |  | {% if fileupload==True and prediction\_type==1 %} | |  | <div class="container"> | |  | <div class="row"> | |  | <div class="col col-8"> | |  | <img src="/static/upload/{{ image\_filename }}" alt="uploaded image" width="300" height="300"> | |  | <h2> Lung Parenchyma Prediction : <i> {{prediction\_LP}} </i></h2> | |  | <h2> Lung Parenchyma Confidence : <i> {{confidence\_LP}} </i> %</h2> | |  | </div> | |  | </div> | |  | </div> | |  | {% endif %} {% if fileupload==True and prediction\_type==2 %} | |  | <div class="container"> | |  | <div class="row"> | |  | <div class="col col-8"> | |  | <img src="/static/upload/{{ image\_filename }}" alt="uploaded image" width="300" height="300"> | |  | <h2> Lung Parenchyma Prediction : <i> {{prediction\_LP}} </i></h2> | |  | <h2> Lung Parenchyma Confidence : <i> {{confidence\_LP}} </i> %</h2> | |  | <h2> Covid Positivity Prediction : <i> {{prediction\_CP}} </i></h2> | |  | <h2> Covid Positivity Confidence : <i> {{confidence\_CP}} </i> %</h2> | |  | </div> | |  | </div> | |  | </div> | |  | {% endif %} {% if fileupload==True and prediction\_type==3 %} | |  | <div class="container"> | |  | <div class="row"> | |  | <div class="col col-8"> | |  | <img src="/static/upload/{{ image\_filename }}" alt="uploaded image" width="300" height="300"> | |  | <h5> Lung Parenchyma Prediction : <i> {{prediction\_LP}} </i></h5> | |  | <h5> Lung Parenchyma Confidence : <i> {{confidence\_LP}} </i> %</h5> | |  | <h5> Covid Positivity Prediction : <i> {{prediction\_CP}} </i></h5> | |  | <h5> Covid Positivity Confidence : <i> {{confidence\_CP}} </i> %</h5> | |  | <h5> Risk Prediction : <i> {{prediction\_risk}} </i></h5> | |  | <h5> Risk Confidence : <i> {{confidence\_risk}} </i> %</h5> | |  | <h5> Mortality Prediction : <i> {{prediction\_mor}} </i></h5> | |  | <h5> Mortality Confidence : <i> {{confidence\_mor}} </i> %</h5> | |  | </div> | |  | </div> | |  | </div> | |  | {% endif %} {% if extension %} | |  | <div class="container" id=myrow\_result> | |  | <div class="row"> | |  | <p>Invalid extension of the image</p> | |  | <p>Make sure the image extension is of following:".png", ".jpg", ".jpeg"</p> | |  | </div> | |  | </div> | |  |  | |  | {% endif %} | |  |  | |  | <style> | |  | #myrow { | |  | background-color: #fbf3f2; | |  | } | |  |  | |  | #myrow\_result { | |  | background-color: #fbf3f2; | |  | padding: 5%; | |  | align-content: center; | |  | } | |  | </style> | |  |  | |  | {% endblock %} | |

Selengkapnya dapat dilihat pada halaman berikut:

<https://github.com/Soedirman-Machine-Learning/CT-Scan-Classification-for-COVID19/tree/main/Web%20Application>

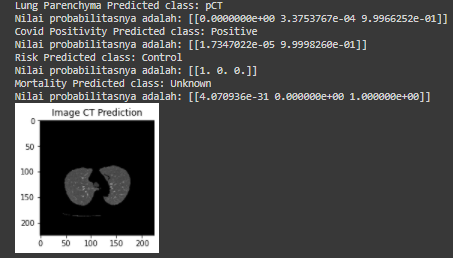
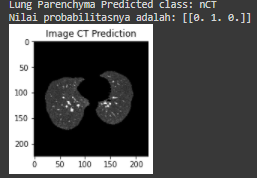
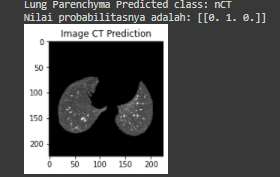
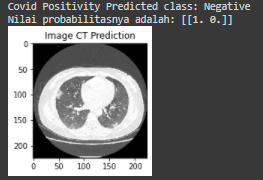
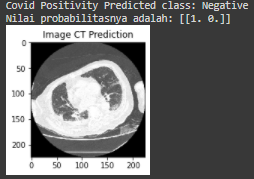
1. Gambar pengujian deteksi

Selengkapnya dapat dilihat pada halaman berikut:

1. Hasil pengujian deteksi pada infrastruktur *google colaboratory*

Selengkapnya dapat dilihat pada halaman berikut :

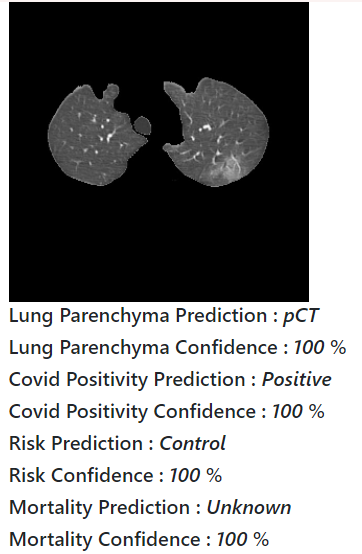
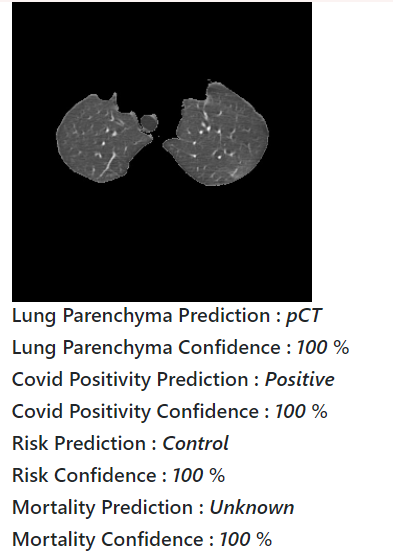
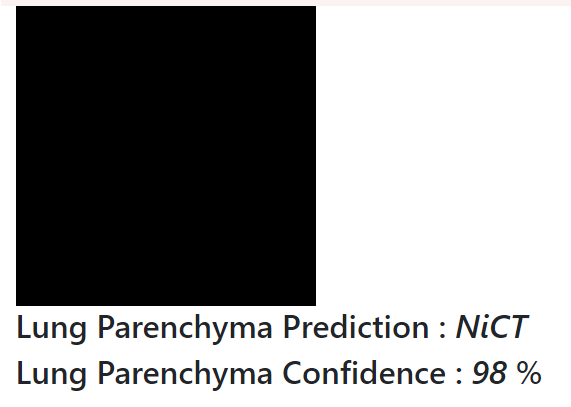
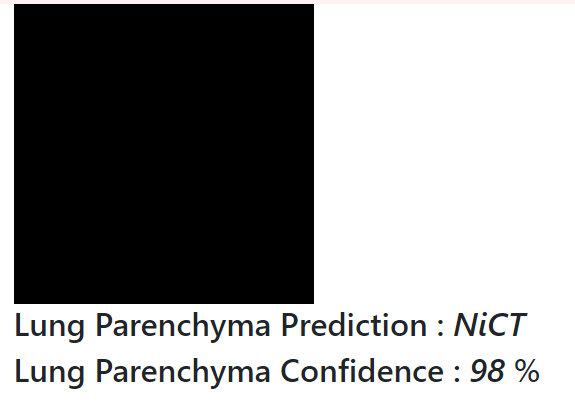
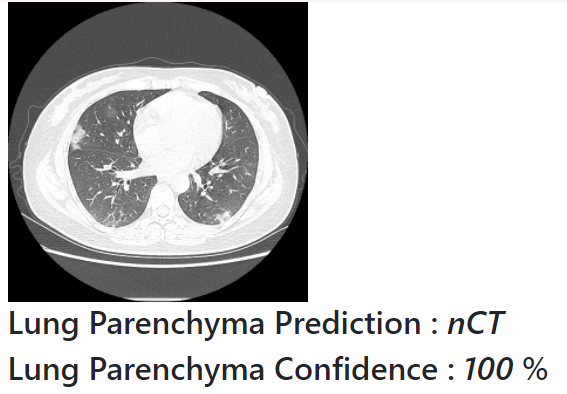
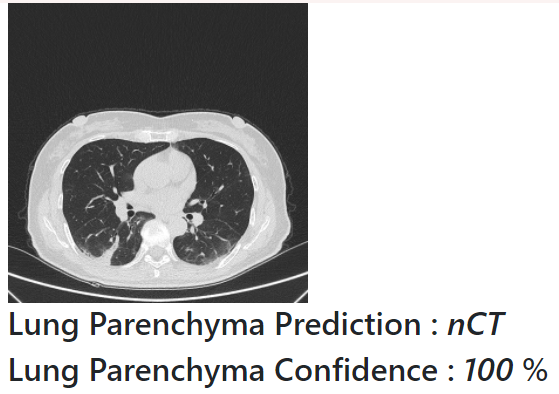
<https://github.com/Soedirman-Machine-Learning/CT-Scan-Classification-for-COVID19/tree/main/Pengujian(Colaboratory)/Hasil%20Pengujian>



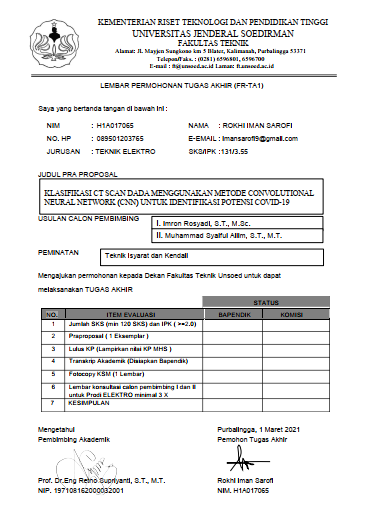
1. Hasil pengujian deteksi pada infrastruktur aplikasi web

Selengkapnya dapat dilihat pada halaman berikut :

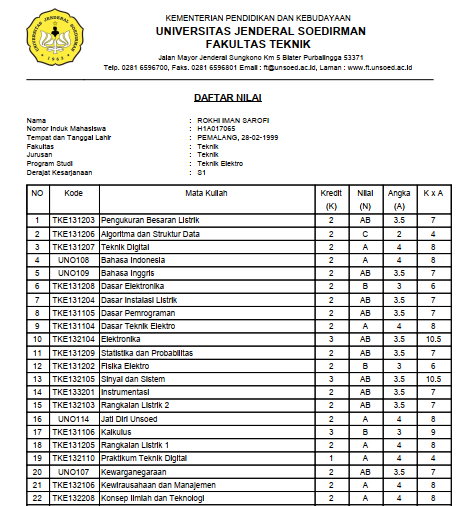
<https://github.com/Soedirman-Machine-Learning/CT-Scan-Classification-for-COVID19/tree/main/Hasil%20Pengujian%20(Web%20App)>



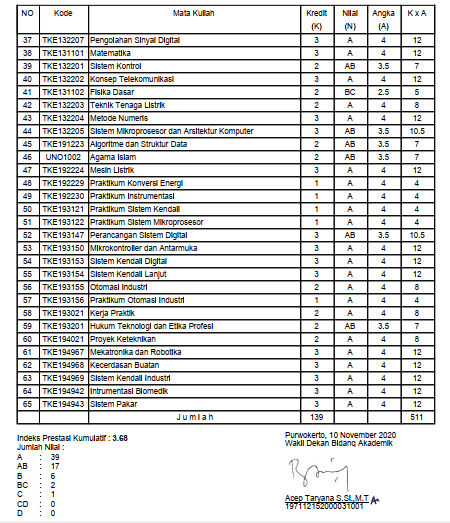
1. Lembar Permohonan TA



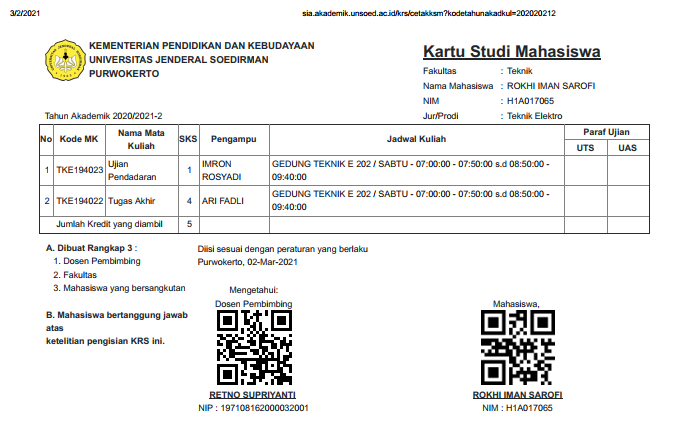
1. Transkrip Nilai Sementara



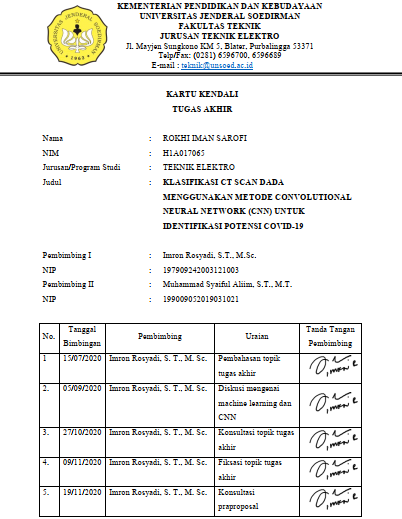


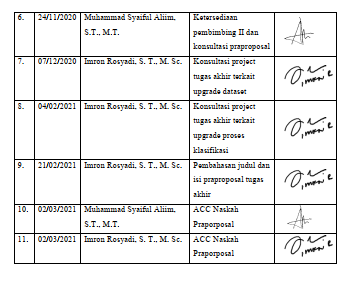


1. Kartu Studi Mahasiswa



1. Kartu Kendali (Bukti Bimbingan)





# BIODATA PENULIS

Biodata penulis berisi terkait dengan identitas penulis (nama, kontak email), riwayat akademis (pendidikan) penulis ditulis dari yang paling , skill, serta prestasi penulis.

A. Identitas

|  |  |  |
| --- | --- | --- |
| Nama | : | Rokhi Iman Sarofi |
| NIM | : | H1A017065 |
| Tempat, tanggal lahir | : | Pemalang, 28 Februari 1999 |
| Alamat | : | Desa Pecangakan RT 01/RW 02, Comal, Pemalang, Jawa Tengah 52363 |
| No. Telp. | : | 089501203765 |
| Alamat e-mail | : | [rokhi.sarofi@mhs.unsoed.ac.id](mailto:rokhi.sarofi@mhs.unsoed.ac.id) |

B. Riwayat Pendidikan Akademik

| **Periode** | **Jenjang** | **Institusi** |
| --- | --- | --- |
| 2017 – 2021 | S1 | Teknik Elektro Universitas Jenderal Soedirman |
| 2014 – 2017 | SMA | SMAN 1 Comal |
| 2011 – 2014 | SMP | SMPN 1 Comal |

C. Riwayat Pendidikan Non Formal (jika ada)

| **Tahun** | **Keahlian** | **Penyelenggara** | **Kota** |
| --- | --- | --- | --- |
| 2014 | Bahasa Inggris Tingkat Mahir | Lembaga Kursus xxxxx | Purwokerto |
| 2013 | Kemanan Jaringan Mikrotik Tingkat Mahir | Lembaga xxxxxxx | Jakarta |

D. Prestasi

| **Tahun** | **Tingkat** | **Prestasi** |
| --- | --- | --- |
| 2014 | Nasional | Juara 1 lomba penulisan karya ilmiah, Yogyakarta |
| 2013 | Internasional | Medali emas olimpiade sains internasional, Dakka, India |

E. Keahlian (tuliskan secara diskriptif)

Memiliki minat di bidang pengembangan perangkat tertanam. Mampu merancang sistem embedded berbasiskan mikro kontroler atmega, arduino dan ESP8266. Terlibat secara aktif dalam kegiatan asistem Laboratorium Sistem Telekomunikasi dan Informasi sebagai asisten praktikum Algoritma dan Struktur Data, Jaringan Komputer, dan Dasar Pemrograman.

1. [↑](#footnote-ref-1)