



SEKOLAH TINGGI TEKNOLOGI TERPADU NURUL FIKRI

**Rancang Bangun Sistem Deteksi Luka Ringan Dengan Metode
*Convolutional Neural Network (CNN) Berbasis Mobile***

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk memperoleh gelar sarjana

Muhammad Amien Ramdhani

0110220168

PROGRAM STUDI TEKNIK INFORMATIKA

DEPOK

OKTOBER 2023

HALAMAN PERNYATAAN ORISINALITAS

**Skripsi/Tugas Akhir ini adalah hasil karya penulis,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.**

Nama : Muhammad Amien Ramdhani

NIM : 0110220168

Tanda Tangan : 

Tanggal : 21 Maret 2024

HALAMAN PENGESAHAN

Skripsi/Tugas Akhir ini diajukan oleh :

Nama : Muhammad Amien Ramdhani

NIM : 0110220168

Program Studi : Teknik Informatika

Judul Skripsi : Rancang Bangun Sistem Deteksi Luka Ringan Dengan
Metode *Convolutional Neural Network* (CNN) Berbasis
Mobile

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Komputer pada Program Studi Teknik Informatika, Sekolah Tinggi Teknologi Terpadu Nurul Fikri

DEWAN PENGUJI

Pembimbing I



(Ahmad Rio Adriansyah, S.Si. M.Si.)

Penguji I



(Dr. Sirojul Munir, S.Si, M.Kom.)

Ditetapkan di : Depok, Jawa Barat

Tanggal : 21 Maret 2024

KATA PENGANTAR

Dengan mengucapkan rasa syukur kepada Allah SWT, karena atas nikmat dan Rahmat-Nya, saya dapat mengerjakan dan menyelesaikan Tugas Akhir ini. Tugas akhir ini adalah salah satu syarat untuk memenuhi dan mendapatkan gelar sarjana komputer program studi teknik informatika pada Sekolah Tinggi Teknologi Terpadu Nurul Fikri Depok. Dengan bantuan dan dukungan dari semua pihak yang terlibat dalam membantu saya menuliskan dan mengerjakan tugas akhir maka saya mengucapkan terima kasih kepada:

1. Allah SWT.
2. Orang tua saya yang selalu mendukung dan mensupport saya dalam menyelesaikan tugas akhir ini.
3. Bapak Dr. Lukman Rosyidi, M.T., M.M. sebagai Ketua Sekolah Tinggi Teknologi Terpadu Nurul Fikri.
4. Ibu Tifanny Nabarian, S.Kom. M.T.I. sebagai Ketua Program Studi Teknik Informatika Sekolah Tinggi Teknologi Terpadu Nurul Fikri.
5. Bapak Ahmad Rio Adriansyah., S.Si., M.Si. selaku Dosen Pembimbing Tugas Akhir saya yang memberikan arahan dan mendukung saya dalam penulisan tugas akhir ini
6. Semua dosen di Sekolah Tinggi Teknologi Terpadu Nurul Fikri Depok yang telah memberikan ilmunya dan mengajarkan saya banyak hal yang membantu saya dalam menuliskan tugas akhir ini.
7. Bangkit Academy Manajer Adrianus Yoza Aprilio beserta karyawan yang telah memberikan banyak hal dan banyak ilmu yang sangat membantu saya dalam membangun dan menuliskan tugas akhir

Dalam penulisan ilmiah ini masih banyak yang perlu saya pelajari lagi dan perlu saya kembangkan lagi. Namun, dengan rasa syukur saya dapat menuliskan tugas akhir ini sampai selesai. Oleh karena, itu apabila terdapat kekurangan di dalam penulisan ilmiah ini, saya mohon maaf atas segala kekurangan saya dan saya menerima kritik dan saran dari para pembaca tulisan ilmiah saya.

Akhir kata, saya mengucapkan banyak terima kasih kepada semua pihak yang telah membantu saya dalam menuliskan tugas akhir ini. Semoga Allah SWT membalas semua kebaikan kepada semua pihak yang telah membantu saya . Semoga skripsi/tugas akhir ini membawa manfaat bagi seluruhnya.

Depok, 14 Oktober 2023

Muhammad Amien Ramdhani

**HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI
TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS**

Sebagai sivitas akademik Sekolah Tinggi Teknologi Terpadu Nurul Fikri, saya yang bertanda tangan di bawah ini:

Nama : Muhammad Amien Ramdhani
NIM : 0110220168
Program Studi : Teknik Informatika
Jenis karya : Skripsi / Tugas Akhir

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada STT-NF **Hak Bebas Royalti Non Eksklusif (*Non-exclusive Royalt - Free Right*)** atas karya ilmiah saya yang berjudul :

Rancang Bangun Sistem Deteksi Luka Ringan Dengan Metode *Convolutional Neural Network* (CNN) Berbasis *Mobile*

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini STT-NF berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : ...Depok.....

Pada tanggal : ...21 Maret 2024.....

Yang menyatakan



(M. Amien Ramdhani.....)

ABSTRAK

Nama : Muhammad Amien Ramdhani
NIM : 0110220168
Program Studi : Teknik Informatika
Judul : Rancang Bangun Sistem Deteksi Luka Ringan Dengan Metode *Convolutional Neural Network* (CNN) Berbasis *Mobile*

Luka adalah hilang atau rusaknya sebagian jaringan tubuh yang disebabkan oleh trauma tajam atau tumpul, perubahan suhu, paparan zat kimia, ledakan, sengatan listrik, maupun gigitan hewan. Dengan menggunakan dan memanfaatkan teknologi yang semakin pesat maka dapat dibuat sebuah sistem untuk mencegah infeksi dari luka khususnya luka ringan. Salah satu cara agar luka tersebut tidak menimbulkan infeksi maka luka tersebut harus diidentifikasi terlebih dahulu sehingga dapat memberikan pertolongan pertama dan obat sesuai luka yang dialami. Pada penelitian ini digunakan *machine learning* sebagai sarana dalam memudahkan masyarakat dalam mencegah infeksi luka ringan. Metode yang digunakan yaitu *Convolutional Neural Network* yang digunakan dalam pemrosesan gambar dan diimplementasikan ke sistem berbasis *mobile*. Hasil pembuatan sistem ini dapat membantu masyarakat untuk mencegah terjadinya infeksi luka ringan dan dapat mengobati luka dengan baik dan benar. Dari sistem yang telah dibuat dengan menggunakan *machine learning* dan metode *Convolutional Neural Network* untuk deteksi luka ringan mendapatkan Tingkat akurasi dari 50% - 95% dari 200 data gambar yang diproses. Sedangkan untuk klasifikasi luka ringan mendapatkan tingkat akurasi untuk data latih dan data validasi yaitu 94% untuk data latih dan 76% untuk validasi sehingga data mengalami overfitting dan data gambar yang dipakai adalah 325 gambar.

Kata kunci :

Machine Learning, Convolutonal Neural Network, Sistem, Luka Ringan, Mobile

ABSTRACT

Name : Muhammad Amien Ramdhani
NIM : 0110220168
Study Program : Informatics Engineering
Title : Design of a Mobile-Based Minor Wound Detection
System with Convolutional Neural Network (CNN)
Method

A wound is the loss or damage of part of the body tissue caused by sharp or blunt trauma, temperature changes, chemical exposure, explosions, electric shock, or animal bites. By using and utilizing increasingly rapid technology, a system can be made to prevent infection from wounds, especially minor ones. One way to prevent the wound from causing infection is to identify the wound first so that it can provide the first treatment and medicine according to the wound experienced. In this research, machine learning is used to facilitate the community in preventing minor wound infections. The method used is Convolutional Neural Network which is used in image processing and implemented into a mobile-based system. The results of this system can help people prevent minor wound infections and treat wounds properly and correctly. The system that has been made using machine learning and the Convolutional Neural Network method for minor wound detection, gets an accuracy rate of 50% - 95% of 200 processed image data. As for the classification of minor wounds, the accuracy rate for training data and validation data is 94% for training data and 76% for validation so the data is overfitting and the image data used is 325 images.

Key words :

Machine Learning, Artificial Neural Network, System, Minor Injuries, Mobile

DAFTAR ISI

Cover	1
HALAMAN PERNYATAAN ORISINALITAS	2
HALAMAN PENGESAHAN	3
KATA PENGANTAR	4
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI	6
ABSTRAK	7
ABSTRACT	8
DAFTAR ISI	9
DAFTAR GAMBAR	12
DAFTAR TABEL	14
BAB I PENDAHULUAN	15
1.1 Latar belakang	15
1.2 Rumusan Masalah	16
1.3 Tujuan Penelitian	16
1.4 Manfaat Penelitian	17
1.5 Batasan Masalah	17
1.6 Sistematika Penulisan	17
BAB II KAJIAN LITERATUR	19
2.1 Pengertian Sistem	19
2.2 Pengertian Artificial Intelligence	19
2.3 Pengertian Machine Learning	19
2.3.1 Supervised Learning	20
2.3.2 Unsupervised Learning	20
2.3.3 Semi-Supervised Learning	20
2.3.4 Reinforcement Learning	20
2.4 Pengertian Deep Learning	21
2.5 Pengertian Convolutional Neural Network	21
2.6 Pengertian Computer Vision	22

2.7	Pengertian Luka	23
2.8	Pengertian Tensorflow	24
2.9	Pengertian Mobile Application	24
2.10	Pengertian Deteksi	24
2.11	Pengertian Tensorflow Lite.....	24
2.12	Pengertian Transfer Learning.....	25
2.13	Google Colaboratory	25
2.14	Python	26
2.15	Penelitian Terkait	26
BAB III METODOLOGI PENELITIAN.....		29
3.1	Tahapan Penelitian	29
3.2	Rancangan Penelitian	32
3.2.1	Jenis Penelitian.....	32
3.2.2	Metode Analisis.....	32
3.2.3	Metode Pengumpulan Data	33
3.2.4	Metode Pengujian.....	33
3.2.5	Metode Implementasi dan Evaluasi	33
3.2.6	Lingkungan Pengembangan	33
3.2.7	Alat Penelitian	34
BAB IV IMPLEMENTASI DAN EVALUASI.....		36
4.1	Analisis dan Perancangan Melalui Metode CRISP-DM.....	36
4.1.1	Pemahaman Bisnis (<i>Bussiness Understanding</i>) <i>Object Detection</i>	36
4.1.2	Pemahaman Data (<i>Data Understanding</i>) <i>Object Detection</i>	39
4.1.3	Persiapan Data (<i>Data Preparation</i>) <i>Object Detection</i>	40
4.1.4	Pembuatan Model atau <i>Modelling Object Detection</i>	43
4.1.5	Evaluasi atau <i>Evaluation Object Detection</i>	52
4.1.6	Persiapan Data (<i>Data Preparation</i>) <i>Classification</i>	53
4.1.7	Pembuatan Model atau <i>Modelling Classification</i>	54
4.1.8	Evaluasi atau <i>Evaluation Classification</i>	58
4.1.9	Deployment model	61
4.2	Implementasi dan Evaluasi Sistem.....	71
4.2.1	Implementasi dan Evaluasi <i>Object Detection</i>	71

BAB V KESIMPULAN DAN SARAN.....	77
5.1 Kesimpulan	77
5.2 Saran.....	78
DAFTAR REFERENSI	79

DAFTAR GAMBAR

Gambar 1 Convolutional Neural Network	22
Gambar 2 Computer Vision (Image Recognition)	23
Gambar 3 Perbedaan Antara Traditional Learning dan Transfer Learning	25
Gambar 4 Alur Tahap Penelitian.....	29
Gambar 5 Pemahaman penelitian melalui Ialur	36
Gambar 6 Arsitektur Sistem Deteksi Luka Ringan.....	38
Gambar 7 Mencari data luka lecet dalam search engine google.....	39
Gambar 8 Mencari data luka bakar dalam search engine google.....	40
Gambar 9 Membuat folder untuk data yang akan diolah.....	41
Gambar 10 Melakukan proses labelling gambar dengan aplikasi labeling.....	41
Gambar 11 Memberikan segmentasi terhadap letak luka ringan	42
Gambar 12 Tensorflow Detection Model Zoo	44
Gambar 13 Install Tensorflow Object Detection Zoo.....	44
Gambar 14 Install library yaml dan tensorflow	44
Gambar 15 Install libraby yang dibutuhkan.....	44
Gambar 16 Menyiapkan Google Colab.....	45
Gambar 17 Data diupload dan digunakan menggunakan Google drive	45
Gambar 18 Split data menjadi data latih, data uji dan data validasi	46
Gambar 19 Membuat labelmap danTFRecord.....	46
Gambar 20 Konversi TFRecod menjadi CSV.....	47
Gambar 21 konfigurasi terhadap model transfer learning mobilenet-v2	48
Gambar 22 Konfigurasi Hyperparameter.....	48
Gambar 23 Konfigurasi Hyperparameter 2.....	49
Gambar 24 Arsitektur transfer learnng yang digunakan	50
Gambar 25 Proses training model.....	50
Gambar 26 Proses training model 2	51
Gambar 27 Konversi model kedalam TFLite.....	52
Gambar 28 Load Tensorboard	52
Gambar 29 Hasil evaluasi menggunakan Tensorboard.....	53

Gambar 30 Hasil uji coba model.....	53
Gambar 31 Membuat folder data sesuai kelas	54
Gambar 32 Import library yang dibutuhkan.....	55
Gambar 33 Ekstrak Data	55
Gambar 34 Segmentasi data menggunakan active contour.....	55
Gambar 35 Hasil segmentasi data	56
Gambar 36 Melakukan Augmentasi Gambar.....	57
Gambar 37 Membuat layer Convolutional Neural Network dengan transfer learning	57
Gambar 38 Metrik Akurasi model	58
Gambar 39 Metrik loss model.....	59
Gambar 40 Export model menjadi TFLite	60
Gambar 41 Hasil Confusion Matrix model.....	60
Gambar 42 Hasil Precision, recall dan F1-Score	61
Gambar 43 Import Tensorflow Lite model dalam Android Studio.....	62
Gambar 44 Mengatur ukuran bit	62
Gambar 45 Mengatur ukuran bit 2	62
Gambar 46 Preprocessing Image	63
Gambar 47 Preprocessing Image 2	63
Gambar 48 Klasifikasi data dengan label.....	64
Gambar 49 Klasifikasi data dengan label 2.....	64
Gambar 50 Klasifikasi data dengan label 3.....	64
Gambar 51 Halaman awal sistem.....	65
Gambar 52 Halaman utama sistem deteksi luka ringan	66
Gambar 53 Menu deteksi luka ringan	67
Gambar 54 Menampilkan data luka ringan	68
Gambar 55 Hasil Klasifikasi Luka	69
Gambar 56 Pertolongan pertama pada sistem	70

DAFTAR TABEL

Tabel 1 Penelitian Terkait	27
Tabel 2 Daftar data, format beserta jumlah data yang digunakan.....	39
Tabel 3 Pembagian Jenis Data	42
Tabel 4 Tabel data label dari setiap gambar.....	43
Tabel 5 Hasil akhir loss model yang telah dilatih	51
Tabel 6 Cross Data Validation Object Detection	72
Tabel 7 Cross Data Validation Classification	73
Tabel 8 Black Box Testing Fitur deteksi luka ringan	74
Tabel 9 User Acceptance Test.....	76

BAB I

PENDAHULUAN

Pada BAB I ini berisi pendahuluan penelitian yang meliputi latar belakang, rumusan masalah, tujuan dan manfaat penelitian, batasan masalah serta sistematika penulisan.

1.1 Latar belakang

Komputer merupakan salah satu perangkat keras yang digunakan oleh manusia untuk membantu meringankan pekerjaan manusia. Komputer ketika diciptakan harapannya komputer itu dapat memahami apa yang manusia inginkan. Hal tersebut dibuktikan adanya teknologi yang bernama *Deep learning*. *Deep learning* adalah salah satu teknologi yang terinspirasi dari otak manusia yang termasuk dari metode *Artificial Intelligence* (AI) yang mana pada deep learning ini komputer akan diajarkan untuk memproses sekumpulan data dan komputer akan mengenali data tersebut sehingga data dapat diproses dan digunakan manusia. Data yang dapat diproses oleh *deep learning* adalah berbagai macam mulai dari data gambar, teks, suara sehingga dari data tersebut dapat menghasilkan wawasan dan prediksi yang akurat.

Dengan perkembangan teknologi yang semakin pesat tentunya sangat membantu bagi manusia salah satunya yaitu munculnya teknologi *Artificial Intelligence* (AI), AI saat ini sudah banyak dimanfaatkan di berbagai bidang mulai dari bidang industri, bidang bisnis, bidang kesehatan, hingga ke bidang transportasi. Salah satu teknologi dari AI adalah *Machine Learning*. ML atau yang biasa dikenal dengan pembelajaran mesin adalah ilmu komputer yang bisa bekerja tanpa di program secara eksplisit. Salah satu contoh penerapan ML adalah pemrosesan gambar atau citra yang mana pada pemrosesan ini gambar akan dilatih oleh mesin sehingga mesin dapat mempelajari dan mengenali gambar untuk tujuan tertentu. Seperti, pemrosesan gambar untuk kebutuhan kesehatan dengan gambar yang diambil seperti citra *X-Ray*, citra *Magnetic Resonance Imaging* dan masih banyak lagi.

Luka adalah hilang atau rusaknya sebagian jaringan tubuh yang disebabkan oleh trauma tajam atau tumpul, perubahan suhu, paparan zat kimia, ledakan, sengatan listrik, maupun gigitan hewan [1]. Seseorang ketika terjatuh dari sepeda motor atau

tertusuk duri biasanya mendapatkan luka ringan. Luka ringan tersebut dapat mengakibatkan infeksi yang cukup berbahaya. Dari luka ringan tersebut terkadang manusia menyepelekan luka tersebut yang mana dari luka tersebut biasanya dapat menyebabkan kerusakan fungsi perlindungan kulit.

Berdasarkan uraian masalah diatas, maka diperlukan sebuah solusi untuk masyarakat Indonesia berupa penerapan teknologi ML yang akan dibahas pada penelitian ini. Teknologi ini dibangun dengan pemanfaatan ML dan akan dibangun berbasis *mobile apps* dan menggunakan fitur kamera untuk pengambilan gambarnya. Dengan mengimplementasikannya ke *mobile apps* diharapkan dapat mempermudah pengguna untuk mengaksesnya, tidak hanya mempermudah pengguna tapi juga mempermudah dalam proses pengembangan dan pemeliharaan. Teknologi ini berisikan tentang mengidentifikasi penyakit ringan dan berupa cara pencegahannya dan berisikan obat yang digunakan untuk mengobati luka ringan tersebut. Hal ini sekaligus diharapkan bahwa tim kami dapat memberikan solusi untuk menjawab tantangan permasalahan yang ada dengan membuat teknologi seperti yang telah dijelaskan sebelumnya.

1.2 Rumusan Masalah

Mengacu kepada permasalahan diatas, maka rumusan masalah yang diangka adalah sebagai berikut :

1. Bagaimana alur perancangan sistem deteksi luka ringan dengan metode *Convolutional Neural Network* dan bagaimana hasil dari sistem yang telah dibuat dengan metode tersebut?.
2. Bagaimana sistem deteksi luka ringan dapat memudahkan user untuk mendeteksi luka ringan?
3. Apakah sistem deteksi luka ringan dapat memberikan penanganan pertama dan obat untuk penanganan luka ringan tersebut?

1.3 Tujuan Penelitian

Tujuan dari dibuatnya penelitian ini adalah sebagai berikut :

1. Dapat merancang alur proses sistem deteksi luka ringan berbasis *mobile* dengan metode *Convolutional Neural Network* (CNN).
2. Dapat memudahkan user dalam melakukan pendeteksian luka ringan dengan sistem deteksi luka ringan berbasis *mobile*.
3. Dapat memberikan penanganan pertama untuk luka ringan yang telah terdeteksi dan memberikan obat untuk penanganan luka ringan tersebut.

1.4 Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut :

1. Sistem deteksi luka ringan dapat memberikan hasil deteksi luka ringan tersebut dan dapat mencegah terjadinya infeksi.
2. Sistem deteksi luka ringan memberikan penangan luka ringan untuk user sehingga luka tersebut dapat teratasi dan dapat terobati.
3. Sistem deteksi luka ringan dapat memberikan obat untuk luka ringan tersebut baik obat herbal maupun obat tradisional.

1.5 Batasan Masalah

Pada penelitian ini dibatasi untuk beberapa hal sebagai berikut :

1. Pada penelitian ini lebih berfokus pada sistem menggunakan *machine learning* dan metode *convolutional neural network*.
2. Data luka ringan terbatas dengan 4 kelas luka ringan.
3. Sistem ini belum mendapatkan memberikan obat penangan secara lengkap hanya obat untuk pertolongan pertama untuk menghindari infeksi.

1.6 Sistematika Penulisan

Agar penulisan ini dapat tersusun dengan baik maka akan ditulis dengan sistematika runtut yang terdiri dari enam bab seperti berikut :

BAB I : PENDAHULUAN

Bab ini membahas mengenai latar belakang penulisan, rumusan masalah, tujuan, manfaat, batasan masalah, serta sistematika penulisan yang digunakan dalam Menyusun penelitian ini.

BAB II : KAJIAN LITERATUR

Bab ini berisikan berbagai landasan teori mengenai implementasi ML pada aplikasi berbasis *mobile* donasi serta beberapa penelitian lainnya yang menunjang penelitian.

BAB III : METODOLOGI PERANCANGAN

Bab ini membahas mengenai tahapan perancangan yang akan dijalankan selama proses penyelesaian penelitian ini.

BAB IV : IMPLEMENTASI DAN EVALUASI

Bab ini membahas mengenai implementasi dan evaluasi dari hasil yang telah dibuat dan terdapat bukti pengerjaan.

BAB V : KESIMPULAN DAN SARAN

Bab ini berisikan kesimpulan dan saran atas apa yang telah dikerjakan selama pembuatan dan penulisan.

BAB II

KAJIAN LITERATUR

Pada BAB II ini berisi definisi - definisi, teori- teori dengan analisis penelitian, penelitian terkait yang disajikan dalam bentuk tabel perbandingan penelitian dan penjelasannya.

2.1 Pengertian Sistem

Menurut Meriam-Webster sistem adalah interaksi secara teratur atau kelompok item yang saling bergantung membentuk satu kesatuan yang utuh. Sistem merupakan seperangkat ajaran, gagasan, atau asas yang terorganisasi biasanya dimaksudkan untuk menjelaskan pengaturan atau cara kerja dari keseluruhan yang sistematis[2]. Sistem merupakan suatu wadah untuk menjalankan seluruh perintah yang telah dibuat untuk mencapai suatu tujuan utama. Menurut Azhar Susanto sistem adalah Kumpulan atau grup dari sub sistem atau bagian atau komponen apapun baik fisik maupun non fisik yang saling berhubungan satu sama lain dan dapat bekerja sama untuk mencapai satu tujuan tertentu[2].

2.2 Pengertian Artificial Intelligence

Artificial Intelligence (AI) atau biasa disebut dengan kecerdasan buatan merupakan suatu teknologi yang berkaitan dengan pembuatan mesin yang dapat bertindak dan bereaksi secara tepat, mengadaptasi respon terhadap tuntutan situasi. Mesin tersebut harus menampilkan perilaku yang sebanding dengan perilaku yang dianggap membutuhkan kecerdasan pada manusia[3].

2.3 Pengertian Machine Learning

Definisi *machine learning* (ML) berdasarkan jurnal “*what is machine learning? A primer for the epidemiologist*” menjelaskan bahwa *machine learning* adalah cabang ilmu komputer yang secara luas bertujuan untuk memungkinkan komputer belajar tanpa diprogram secara langsung. Hal ini berawal dari gerakan kecerdasan buatan pada tahun 1950 dan menekankan pada tujuan dan aplikasi praktis, khususnya prediksi dan optimasi. Komputer belajar dalam pembelajaran mesin dengan meningkatkan kinerjanya dalam

melakukan tugas-tugas melalui pengalaman [4]. ML adalah bagian dari AI dan ML akan memproses data yang telah dimasukan dan akan dilatih dengan menggunakan konsep statistik dan matematika untuk memecahkan masalah. Adapun ML memiliki beberapa jenis seperti *supervised learning*, *unsupervised learning*, *semi-supervised learning*, *Reinforcement learning*.

2.3.1 Supervised Learning

Supervised learning atau yang biasa disebut pembelajaran terarah adalah salah satu metode pembelajaran mesin dimana hasil yang diharapkan pengguna, sudah diketahui atau dimiliki informasinya oleh sistem [5]. *Supervised learning* dalam arti singkatnya adalah sebuah metode pembelajaran mesin yang mana data-data dari mesin tersebut sudah diberi label sehingga memudahkan mesin untuk mengenalinya.

2.3.2 Unsupervised Learning

Unsupervised learning atau yang biasa disebut pembelajaran tidak terarah adalah metode lain dalam materi pembelajaran mesin pada metode ini hasil yang diharapkan tidak dapat diketahui oleh siapapun[5]. *Unsupervised learning* dapat disebut juga metode pembelajaran mesin yang datanya belum diberi label untuk mengenal data tersebut.

2.3.3 Semi-Supervised Learning

Semi-Supervised learning merupakan gabungan dua metode pembelajaran mesin yaitu *supervised learning* dan *unsupervised learning*. Pada metode ini data yang dimiliki untuk melakukan pelatihan model sebagian sudah diberi label dan sebagian lagi belum diberi label sehingga metode ini diberi nama *semi-supervised learning*[5]. Pada metode ini data yang dimasukan dan data yang dilatih merupakan gabungan dari data yang sudah diberi label dan data yang belum diberi label.

2.3.4 Reinforcement Learning

Reinforcement learning atau yang biasa dikenal dengan metode yang belajar menggunakan sistem *reward* dan *penalty*. Menurut winder, *reinforcement learning* adalah teknik yang mempelajari bagaimana membuat

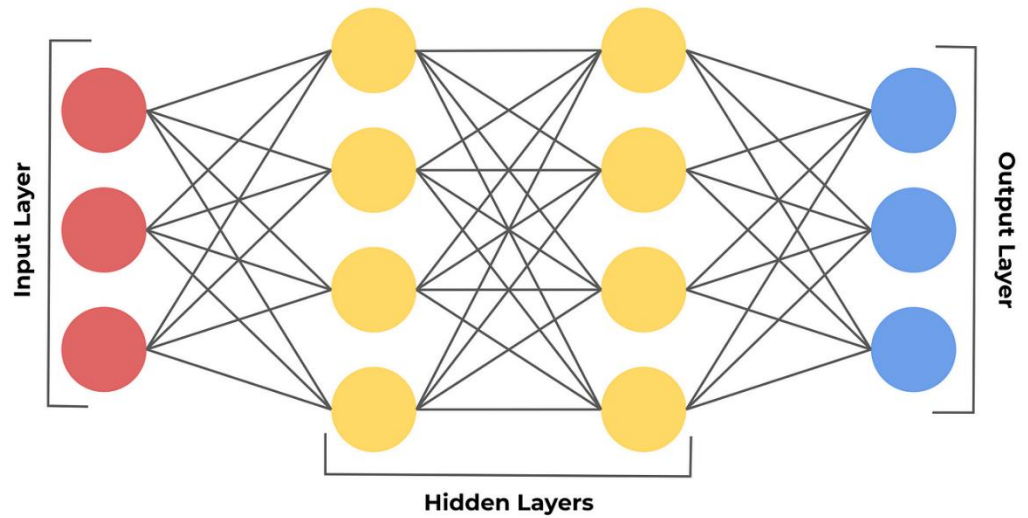
keputusan terbaik, secara berurutan untuk memaksimalkan ukuran sukses kehidupan nyata[5]. Contoh dari *reinforcement learning* adalah permainan catur dimana ketika berhasil memakan bidak catur maka komputer akan mendapat hadiah atau *reward* dan jika komputer melakukan salah langkah dan bidak caturnya dimakan maka komputer akan mendapatkan *penalty* atau sanksi karena melakukan kesalahan.

2.4 Pengertian Deep Learning

Deep learning adalah subbidang kecerdasan buatan yang berfokus pada pembuatan model jaringan syaraf tiruan besar yang mampu membuat keputusan berbasis data yang akurat. *Deep learning* sangat cocok untuk konteks dimana datanya kompleks dan dataset untuk memprosesnya tergolong dalam jumlah yang besar[6]. *Deep learning* merupakan bagian dari ML yang memproses data yang telah dimasukan dan biasanya datanya tergolong banyak dengan menggabungkan statistik dan matematika dengan arsitektur jaringan syaraf tiruan atau *convolutional neural network*.

2.5 Pengertian Convolutional Neural Network

Convolutional neural network (CNN) atau biasa disebut dengan jaringan syaraf tiruan adalah jaringan yang dirancang untuk *image recognition* atau pengenalan gambar. Pada awalnya CNN diterapkan pada tantangan pengenalan digit tulisan tangan (Fukushima 1980; LeCun 1989). Tujuan dari desain dasar CNN adalah membuat jaringan dimana *neuron-neuron* di lapisan awal jaringan akan mengekstrak fitur visual lokal, dan *neuron* di lapisan selanjutnya akan menggabungkan fitur-fitur ini untuk membentuk fitur tingkat tinggi[6].



Gambar 1 Convolutional Neural Network [7]

Gambar diatas adalah alur dari proses kerja CNN dimana pada gambar tersebut terdapat beberapa *neuron* yang berbentuk lingkaran dan dalam setiap *neuron* terdapat jaringan-jaringan yang menghubungkan setiap *neuronnya*.

2.6 Pengertian Computer Vision

Dalam istilah sederhana, *computer vision* adalah bagaimana komputer/mesin dapat melihat, teknik *computer vision* mampu memvisualisasikan data menganalisa berupa gambar atau dalam bentuk vidio. Tujuan utama dari *computer vision* adalah agar komputer atau mesin dapat meniru kemampuan perseptual manusia dan otak, atau bahkan dapat mengunggulinya untuk tujuan tertentu. *Computer vision* erat kaitannya dengan *image recognition*, *image recognition* merupakan salah satu bagian dari *computer vision* untuk mengenali suatu gambar yang sudah diberi label maupun belum yang diberi label.

robek sebesar 23,2%. Sebanyak 40,9% luka disebabkan oleh terjatuh dan 40,6% oleh kecelakaan motor. Penyebab lain yaitu benda tajam atau benda tumpul 7,3%, transportasi darat yang lain 7,1% dan kejatuhan sebesar 2,5% [1].

2.8 Pengertian Tensorflow

Tensorflow adalah *framework open-source* yang dikembangkan oleh perusahaan ternama yaitu google untuk membantu mempermudah pemrosesan model *machine learning*, *deep learning*, serta pemrosesan pekerjaan lainnya yang masih bersangkut paut dengan pekerjaan matematika dan statistik. *Tensorflow* biasa digunakan untuk mempermudah pembuatan model seperti model klasifikasi gambar menggunakan CNN, pemrosesan suara, dan pemrosesan bahasa manusia atau biasa disebut dengan *Natural Language Processing* (NLP). *Tensorflow* membantu untuk mengolah data-data yang dikumpulkan baik dalam jumlah sedikit maupun jumlah besar dan hal ini sangat membantu pemrosesan pembuatan model. Komponen inti *tensorflow* adalah tensor dan grafik komputasi yang melintasi node hingga edge[9].

2.9 Pengertian Mobile Application

Mobile application adalah aplikasi perangkat lunak yang dikembangkan secara khusus untuk digunakan pada perangkat kecil, komputasi nirkabel, seperti ponsel pintar dan tablet, bukan untuk perangkat komputer *desktop* atau laptop[10]. *Mobile application* merupakan aplikasi yang tertanam pada perangkat *smartphone* atau tablet yang mana aplikasi tersebut dapat digunakan untuk berbagai macam seperti untuk berbelanja, berinteraksi dengan teman dan lain sebagainya yang saling terhubung dengan adanya koneksi dari internet.

2.10 Pengertian Deteksi

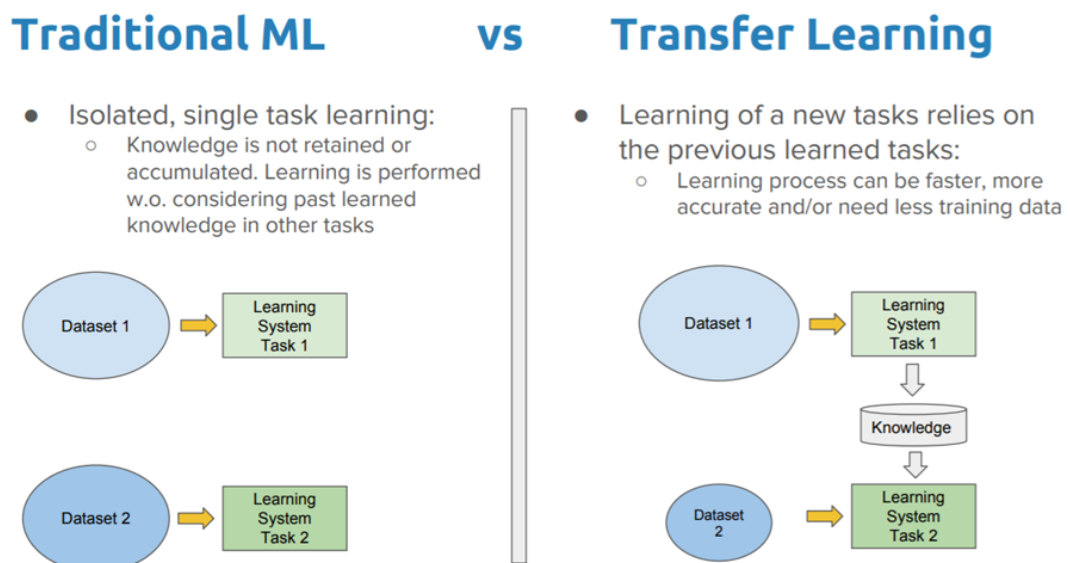
Deteksi adalah suatu proses untuk memeriksa atau melakukan pemeriksaan terhadap terhadap sesuatu dengan menggunakan cara dan teknik tertentu[11]. Deteksi digunakan untuk mengenali suatu objek atau sesuatu agar dapat dipahami.

2.11 Pengertian Tensorflow Lite

Tensorflow lite adalah evolusi dari *tensorflow model* (TFM) yang sudah mendukung penerapan perangkat seluler dan perangkat tertanam. Karena ada tren untuk menggabungkan *machine learning* dalam aplikasi seluler dan karena pengguna memiliki ekspektasi yang lebih tinggi pada aplikasi seluler dalam hal kamera dan suara, maka sangat insentif untuk mengoptimalkan TFM lebih lanjut untuk penggunaan seluler yang ringan[12].

2.12 Pengertian Transfer Learning

Transfer learning adalah metode pembelajaran mesin dimana model dikembangkan untuk suatu tugas yang dapat digunakan kembali sebagai titik awal untuk model pada tugas kedua[13]. *Transfer learning* merupakan suatu kumpulan data yang telah diproses sebelumnya dengan menggunakan *layer-layer* dari *convolutional neural network* dan hasil tersebut disimpan dan dapat digunakan untuk pelatihan lainnya.



Gambar 3 Perbedaan Antara *Traditional Learning* dan *Transfer Learning* [14]

Pada gambar 3 yaitu gambar mengenai perbedaan antara *traditional learning* dan *transfer learning*. Pada *traditional learning* sistem mempelajari data sesuai dengan data baru yang di input. Sedangkan, *transfer learning* adalah pelatihan data yang sudah pernah dilatih sebelumnya dan dapat digunakan pada sistem baru.

2.13 Google Colaboratory

Google Colaboratory merupakan tools yang disediakan oleh google yang dapat digunakan pada perangkat komputer dengan akses internet untuk menyelesaikan masalah seperti *machine learning*, *deep learning*, dan statistik. *Google colaboratory* mendukung beberapa bahasa pemrograman seperti python, dan R programming. *Google colaboratory* menyediakan akses gratis untuk user yang memiliki akun google namun, dibatasi untuk 1 gpu dan *google colaboratory* dapat menggunakan sistem CPU komputer lokal.

2.14 Python

Python merupakan salah satu bahasa pemrograman yang sering digunakan saat ini. Sama seperti bahasa manusia bahasa pemrograman memiliki banyak jenis seperti java, javascript, go, R dan masih banyak lagi. Bahasa pemrograman memiliki kelebihan dan kekurangan seperti halnya bahasa pemrograman PHP sangat bagus ketika digunakan dalam membangun dan merancang aplikasi berbasis website dan python juga sangat bagus untuk perkembangan website dan menghitung statistik data. Namun, dibalik itu semua bahasa pemrograman memiliki kemiripan sehingga ketika mempelajari bahasa pemrograman yang lain menjadi mudah dipahami. Pada intinya tingkat konsep sebagian besar memiliki data dalam variabel dan fungsi untuk melakukan sesuatu pada data tersebut[15].

2.15 LabelImg

LabelImg adalah alat anotasi grafis sumber terbuka yang memungkinkan kotak pembatas di sekitar objek dalam gambar dan memberi label. *LabelImg* dibuat dengan menggunakan bahasa pemrograman *python* dan menggunakan QT untuk antarmuka grafisnya[16].

2.16 Penelitian Terkait

No	Nama dan Tahun	Judul	Topik	Subjek	Hasil
1	Rizqi Efrian M,Latifa U,2022	<i>Image Recognition</i> Berbasis <i>Convolutional Neural Network (CNN)</i> Untuk Mendeteksi Penyakit Kulit Pada Manusia	<i>Image Recognition</i> Berbasis CNN	Anak-Anak, Remaja dan Dewasa	Sistem pendeteksi penyakit kulit pada manusia
2	Abu M, Halim A, Rahman Abd, Ahmad Izanoordina, Hazirah Indra Nurul Sapiee Amalia, 2019	A Study on <i>Image Classification</i> based on <i>Deep Learning</i> and <i>Tensorflow</i>	<i>Image Recognition</i> <i>Deep Learning</i>	Tanaman Bunga	Menghasilkan sistem untuk mendeteksi bunga dengan <i>deep learning</i> dan framework <i>tensorflow</i>
3	Namruddin R, 2023	Klasifikasi Kesegaran Buah Apel Menggunakan Metode <i>Convolutional Neural Network (CNN)</i> Berbasis <i>Android</i>	<i>Convolutional Neural Network</i> Berbasis <i>Android</i>	Buah Apel	Aplikasi berbasis <i>Android</i> untuk mengklasifikasi kesegaran buah apel
4	Novaria Kunang Yesi, 2020	Pengembangan Aplikasi Pengenalan Aksara Komerling Menggunakan Metode <i>Deep Learning</i> Berbasis <i>Android</i>	<i>Deep Learning</i> Berbasis <i>Android</i>	Aksara Komerling	Aplikasi berbasis <i>Android</i> pengenalan aksara komering

Tabel 1 Penelitian Terkait

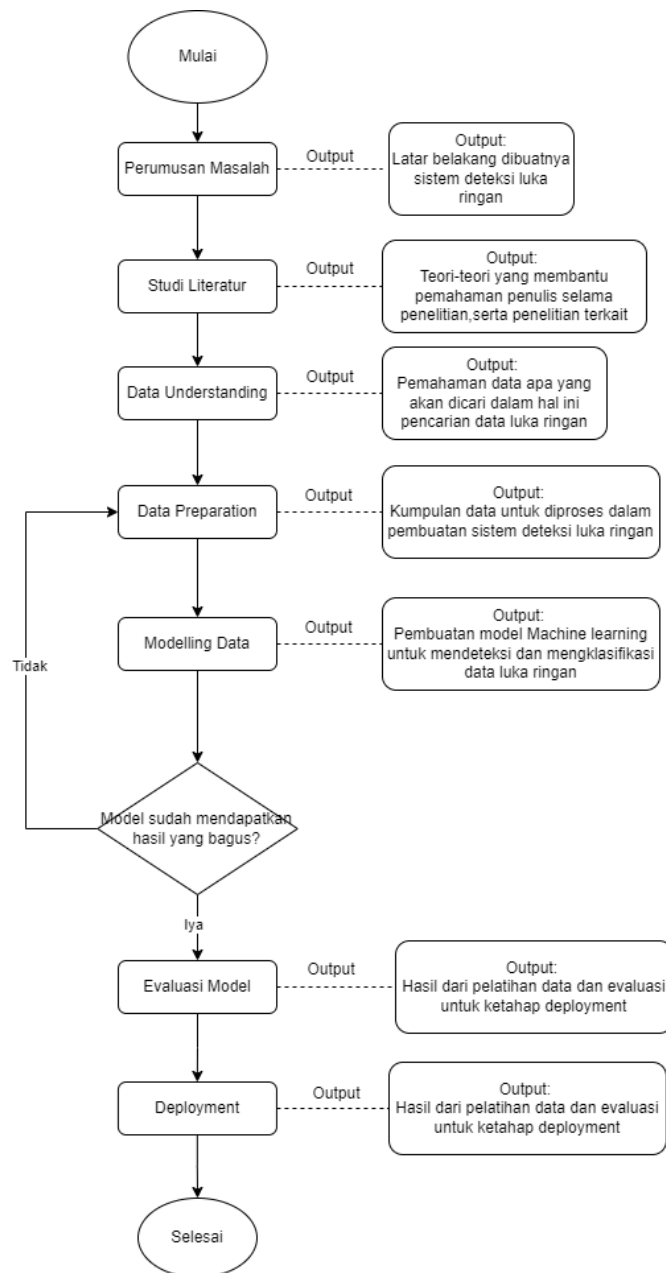
Pada penelitian ini bertujuan untuk mendeteksi luka ringan, terdapat penelitian-penelitian terdahulu yang membahas mengenai convolutional neural network (CNN) dengan berbagai macam data yang digunakan. Image Recognition Berbasis Convolutional Neural Network (CNN) Untuk Mendeteksi Penyakit Kulit Pada Manusia[17]. Klasifikasi Kesegaran Buah Apel Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Android[18]. Pada kedua penelitian ini berfokus menggunakan metode CNN yang mana metode tersebut berkaitan dengan penelitian yang sedang saya buat. Namun, pada kedua penelitian tersebut digunakan untuk proses klasifikasi buah apel dan mendeteksi penyakit kulit. Dari kedua penelitian tersebut terdapat satu penelitian yang hampir mirip dengan penelitian yang sedang dibuat dan yang membedakan yaitu *dataset* yang digunakan dan pengimplementasiannya pada penelitian yang saya buat sistem akan di implementasikan pada perangkat berbasis *mobile*. Kemudian pada penelitian yang lainnya yaitu penelitian mengenai A Study on *Image Classification* based on *Deep Learning* and *Tensorflow*[19] berfokus untuk mengklasifikasi gambar dengan menggunakan metode *deep learning* dan dibantu dengan *library tensorflow*. Pada acuan penelitian yang terkait yaitu mengenai Pengembangan Aplikasi Pengenalan Aksara Komerang Menggunakan Metode *Deep Learning* Berbasis *Android*[20]. Penelitian tersebut melakukan pengembangan terhadap aplikasi yang sebelumnya telah dibuat dan di implementasikan berbasis *android*.

BAB III

METODOLOGI PENELITIAN

Pada bab ini akan dibahas tahapan dan langkah-langkah dalam penulisan, rancangan, dan juga lingkup penelitian serta evaluasi yang dilakukan untuk penelitian yang lebih baik.

3.1 Tahapan Penelitian



Gambar 4 Alur Tahap Penelitian

Berikut adalah penjelasan dari gambar 3 yang merupakan tahapan-tahapan yang dilakukan dalam penelitian ini :

1. Perumusan Masalah

Pada tahap ini, penulis mencari tahu suatu permasalahan dari penelitian ini. Penulis menemukan banyak dari orang yang tidak memperdulikan mengenai luka yang dialaminya terlebih lagi luka ringan. Untuk itu, dilakukan pencarian informasi mengenai bagaimana agar orang dapat lebih peduli dengan luka tersebut. Hasil dari tahap ini merupakan latar belakang dari pembuatan sistem deteksi luka ringan ini sebagai solusi untuk mengatasi masalah ini.

2. Studi Literatur

Pada tahap ini, dilakukan proses studi literatur. Studi literatur dilakukan untuk memahami mengenai teori-teori dan konsep-konsep yang digunakan dalam melakukan penelitian. Proses studi literatur juga dilakukan untuk memilih metode yang terbaik untuk penelitian. Hasil dari studi literatur ini adalah pemahaman terhadap teori-teori untuk melakukan proses penelitian.

3. Data Understanding

Pada tahap ini, dilakukan proses pemahaman dan pengumpulan terhadap data yang dibutuhkan untuk membuat sistem deteksi luka ringan ini. Sebelum melakukan proses pengumpulan data maka harus memahami terlebih dahulu data yang akan digunakan dan dikumpulkan. Kemudian dilakukan proses pengumpulan data, pengumpulan data ini dilakukan dengan mencari sumber data berupa gambar dari kumpulan luka ringan. Data yang diperoleh berasal dari berbagai sumber. Hasil dari tahapan ini adalah kumpulan data yang telah diperoleh untuk dilakukan proses selanjutnya dalam pembuatan sistem deteksi luka ringan.

4. Data Preparation

Pada tahap ini adalah tahapan untuk melakukan persiapan terhadap data yang akan digunakan. Pada tahap ini maka dilakukan pengumpulan data yaitu mengumpulkan data berupa gambar-gambar luka ringan seperti gambar luka bakar, luka lecet, luka sayat dan luka tusuk. Setelah dirasa cukup dalam

mengumpulkan data-data gambar untuk diproses maka langkah selanjutnya akan diproses pada modelling data.

5. Modelling

Pada tahap ini, dilakukan proses modelling data dengan menggunakan metode yang sudah ditetapkan sebelumnya. Modelling data dengan menggunakan data-data yang telah dilakukan *preprocessing* sebelumnya dengan menggunakan metode *Convolutional Neural Network* sehingga menghasilkan sistem yang dapat digunakan. Dalam tahap perancangan ini, penulis mencoba beberapa metrik untuk pengujian terhadap model yang telah dibuat. Adapun metrik yang digunakan seperti metrik akurasi, *F1-score*, *Precision*, *Recall* dan *Mean Average Precision* (MAP). Setelah model telah berhasil didapat maka, langkah selanjutnya yaitu mengimplementasikan model tersebut ke dalam perangkat *mobile*. Untuk perangkat *mobile* yang digunakan oleh penulis yaitu, perangkat *mobile android*. Hasil dari tahap perancangan ini adalah suatu sistem deteksi luka ringan yang dapat digunakan untuk mendeteksi luka ringan.

6. Pengujian dan Evaluasi Model

Pada tahap ini, penulis melakukan pengujian terhadap model yang telah dibuat sebelumnya. Pengujian ini dilakukan untuk mengetahui seberapa mana model dapat berjalan dengan sesuai rencana. Jika, pada saat pengujian terdapat suatu kendala yang menyebabkan sistem tidak dapat berfungsi maka dilakukan proses evaluasi. Evaluasi ini dilakukan untuk memperbaiki sistem yang memiliki kendala sehingga sistem tersebut dapat digunakan tanpa kendala.

7. Kesimpulan

Tahapan ini merupakan tahapan dimana penulis memberi kesimpulan terhadap hasil penelitian yang dilakukan. Kemudian, dari hasil kesimpulan tersebut menghasilkan kritik dan saran yang diberikan untuk pengembangan penelitian yang lebih baik lagi. Hasil kesimpulan ini merupakan tahapan akhir dari penulis untuk melakukan penelitian.

3.2 Rancangan Penelitian

Rancangan penelitian ini disusun sebagai tahap awal yang akan menjelaskan lebih rinci mengenai langkah-langkah yang dilakukan pada penelitian meliputi jenis penelitian, metode analisis, metode pengumpulan data, metode pengujian serta lingkungan pengembangan.

3.2.1 Jenis Penelitian

Penelitian yang dilakukan merupakan rancang bangun sistem, yaitu sistem deteksi luka ringan. Rancang bangun yaitu merancang dan membangun sebuah sistem dari awal hingga menjadi sebuah sistem. Penelitian ini dilakukan untuk membuat sebuah sistem yang dapat mendeteksi luka ringan dan memberikan pertolongan pertama terhadap luka tersebut. Pada penelitian ini, penulis hanya berfokus untuk membuat sebuah sistem yang dapat diimplementasikan pada perangkat berbasis mobile kemudian perangkat tersebut dapat digunakan oleh user.

3.2.2 Metode Analisis

Metode analisis yang digunakan pada penelitian ini yaitu metode kuantitatif dan kualitatif. Metode kuantitatif dilakukan pada saat pengumpulan data dari berbagai sumber kemudian data tersebut dikumpulkan menjadi satu sehingga menjadi kumpulan data yang dapat digunakan. Pada metode kuantitatif dilakukan proses *cross validation* yaitu proses data akan dibagi menjadi tiga bagian yaitu data untuk proses *training*, *validation* dan *testing*. Metode *cross validation* ini dilakukan untuk mencegah data dari *overfitting*. Metode kualitatif dilakukan ketika pengujian sistem menggunakan *black box testing*. Dengan menggunakan metode kualitatif seperti ini ditujukan untuk mendapatkan sistem yang dapat digunakan dan dapat memudahkan dalam tahap evaluasi.

3.2.3 Metode Pengumpulan Data

Pada tahapan pengumpulan data dan informasi pada penelitian ini menggunakan metode Observasi yaitu metode yang dilakukan pada mengumpulkan data berupa gambar luka ringan. Observasi dilakukan dengan cara mencari data dari berbagai sumber yang relevan dengan penelitian.

3.2.4 Metode Pengujian

Pada sistem deteksi luka ringan ini metode pengujian yang digunakan yaitu *black box testing*. *Black box testing* merupakan metode *black box testing* adalah pengujian terhadap sistem yang telah dibuat dengan aspek dari *input* dan *output* dari sistem tersebut tanpa mengetahui struktur kode sistem tersebut. Pada pengujian ini penulis akan mencoba semua fitur yang telah dibuat pada sistem dan hasil dari pengujian ini akan menjadi landasan untuk pengembangan selanjutnya. Kemudian metode yang digunakan yaitu metode *cross validation* yaitu metode untuk membagi data-data kedalam tiga bagian yaitu data untuk *training*, *validation*, dan *testing*. Metode *cross validation* yang digunakan yaitu metode *Holdout Method*.

3.2.5 Metode Implementasi dan Evaluasi

Rancang bangun sistem mengimplementasikan perangkat berbasis *mobile*. Dengan mengimplementasikan berbasis *mobile* tujuannya agar user mendapatkan kemudahan dalam melakukan deteksi luka ringan. *Black box testing* dapat memudahkan proses evaluasi pada sistem karena dengan evaluasi tersebut dapat ditemukan bagian dari sistem yang tidak dapat berjalan dengan baik.

3.2.6 Lingkungan Pengembangan

Penelitian ini dilakukan pada 2 perangkat yaitu pada perangkat lokal komputer dan perangkat *mobile*. Pada perangkat komputer dilakukan pada dua tempat yaitu lokal komputer untuk melakukan proses *labelling* gambar dan penulisan kode program sistem, kemudian

dilakukan juga pada perangkat *cloud* yaitu pada *google colaboratory*. Pada *google colaboratory* dilakukan untuk membuat model *machine learning*. Kemudian perangkat *mobile* digunakan untuk melakukan *testing* terhadap data yang sudah dilakukan *deployment*

3.2.7 Alat Penelitian

Adapun alat dan bahan yang digunakan untuk melakukan penelitian adalah sebagai berikut :

1. Laptop HP 14s-CF3010
 - a. Processor : Intel I3 Gen 10
 - b. RAM : 12 Gb
 - c. System Type : 64-bit windows operating systemDigunakan dalam melakukan proses dokumentasi penelitian dalam bentuk tugas akhir.
2. Windows 10 home merupakan sistem operasi perangkat komputer yang penulis gunakan untuk rancang bangun sistem deteksi luka ringan
3. *Processor* Intel® Core™ i3-1005G1 CPU @ 1.20 GHz (4 CPUs), ~1.2 GHz
4. Microsoft Office word 2019 digunakan untuk melakukan proses penulisan penelitian ini dalam bentuk tertulis.
5. Google Chrome digunakan sebagai browser untuk melakukan perambanan dan penjelelahan terhadap website.
6. Github digunakan sebagai sarana penyimpanan dan pengumpulan progress rancang sistem.
7. Google Colaboratory digunakan sebagai menjalankan kode program dari model yang telah dibuat.
8. Kaggle digunakan untuk mencari kumpulan data yang digunakan.
9. Android studio *Giraffe* | 2022.3.1 digunakan untuk membangun sistem dari hasil tahap tahap sebelumnya.

10. *Python Programming Language* version 3.9 digunakan untuk menuliskan kode program dalam pembuatan model *machine learning*.
11. *Kotlin Programming Language* version 1.9.20 digunakan untuk menuliskan kode program dalam pembuatan sistem berbasis *android*.
12. *Tensorflow 2 Detection Model Zoo* digunakan sebagai model untuk melakukan *object detection*.
13. *LabelImg* digunakan untuk memberikan label kepada data gambar.

BAB IV

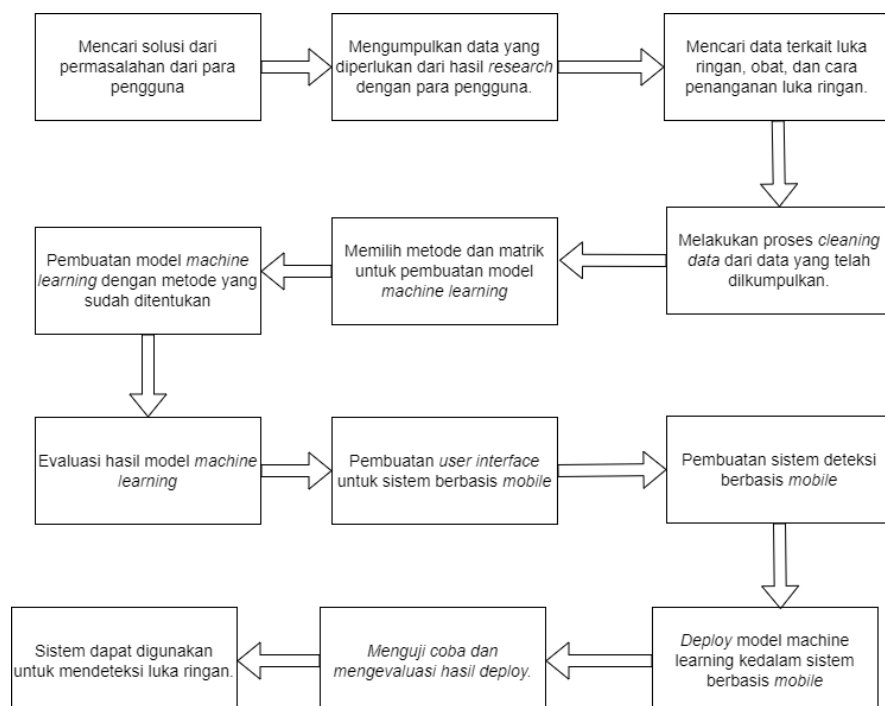
IMPLEMENTASI DAN EVALUASI

Pada bab ini akan dijelaskan implementasi dan evaluasi hasil analisis *user requirement* dan *use case diagram*. Pembahasan mengenai proses rancang bangun sistem deteksi luka ringan juga akan dibahas disini.

4.1 Analisis dan Perancangan Melalui Metode CRISP-DM

4.1.1 Pemahaman Bisnis (*Bussiness Understanding*) *Object Detection*

Pada tahapan pemahaman bisnis akan berfokus pada pemahaman tujuan kebutuhan berdasarkan penilaian bisnis. Kemudian pemahaman tersebut diubah menjadi sebuah rencana awal untuk melakukan penelitian yang dirancang untuk mencapai tujuan. Pada tahapan ini akan dibuat sebuah rancangan awal untuk dan alur apa saja yang akan dilakukan pada penelitian.

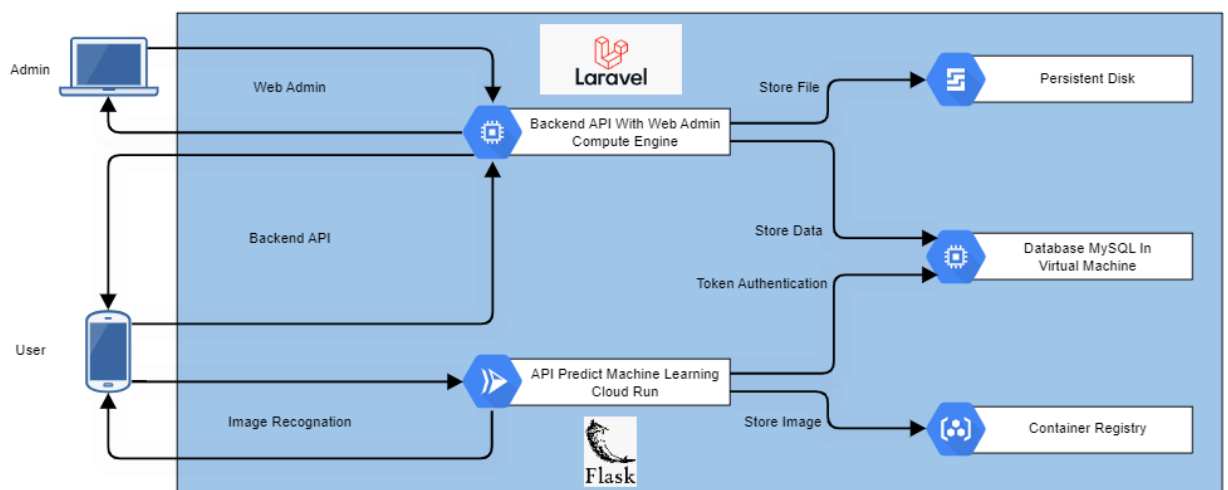


Gambar 5 Pemahaman penelitian melalui Ialur

Dari alur diatas dapat dilihat bahwa langkah-langkah yang dilakukan untuk melakukan penelitian adalah sebagai berikut :

1. Pada langkah pertama hal yang dilakukan yaitu mencari topik dan solusi dari permasalahan para pengguna.
2. Selanjutnya dari hasil topik dan permasalahan yang telah mendapatkan solusi kemudian akan dilakukan proses pengumpulan data dari hasil topik yang akan diselesaikan.
3. Mencari seluruh data yang terkait dengan penelitian dalam hal ini data yang digunakan yaitu data mengenai luka ringan antaranya luka lecet, luka bakar, luka tusuk, dan luka sayat.
4. Langkah selanjutnya yaitu proses *cleaning data* atau pemrosesan data sebelum digunakan untuk *modeling*. Pada pemrosesan ini dilakukan pelabelan terhadap data gambar yang akan digunakan. Untuk label gambarnya yaitu label data luka lecet, data luka bakar, data luka tusuk, dan data luka sayat. Untuk melakukan pelabelan digunakan alat bantu agar mempermudah prosesnya. Alat yang digunakan yaitu menggunakan aplikasi Bernama *labelimg*.
5. Data yang telah diberi label akan disimpan dalam format *.xml*.
6. Kemudian hasil label yang berformat *.xml* agar dapat terbaca dengan *framework tensorflow* maka file tersebut akan dikonversi menjadi format *.csv*. Setelah diformat *.csv* kemudian akan dirubah menjadi *TFRecord* atau *Tensorflow Record*. File *TFRecord* ini akan digunakan dalam proses *feeding data* atau membaca data input sehingga informasi *dataset* dapat diambil secara langsung.
7. Setelah data berhasil diberikan label dan *cleaning data* langkah selanjutnya yaitu membuat model dari data yang telah dikumpulkan dan menggunakan metode yang telah dipilih sebelumnya.
8. Membuat model *machine learning* dengan data yang sudah disiapkan.

9. Mengevaluasi hasil dari pembuatan model *machine learning* apakah model dapat dilanjutkan ke tahap *deploymen* atau model harus diperbaiki kembali.
10. Pembuatan *user interface* yang mana *user interface* yang dibuat yaitu berbasis *mobile*.
11. Melakukan *deployment* terhadap model *machine learning* yang telah dibuat dan di *deploy* pada perangkat *mobile*.
12. Melakukan uji coba terhadap sistem yang telah dibuat yaitu sistem yang menggunakan model *machine learning* dan di *deploy* pada perangkat *mobile* apakah sistem terdapat *bug* atau sistem sudah dapat digunakan.



Gambar 6 Arsitektur Sistem Deteksi Luka Ringan

Pada gambar diatas dapat dijelaskan bahwa untuk sistem yang dipakai yaitu menggunakan *web admin* dan *web Application Programming Interface (API)*. Untuk *web admin* digunakan untuk menyimpan data user seperti data *login*, data cara pertolongan pertama, dan data obat-obatan untuk menangani luka ringan. Sedangkan untuk *web API* digunakan untuk deteksi luka berat. Dan untuk menampilkannya akan ditampilkan dapat *platform* berbasis *mobile*

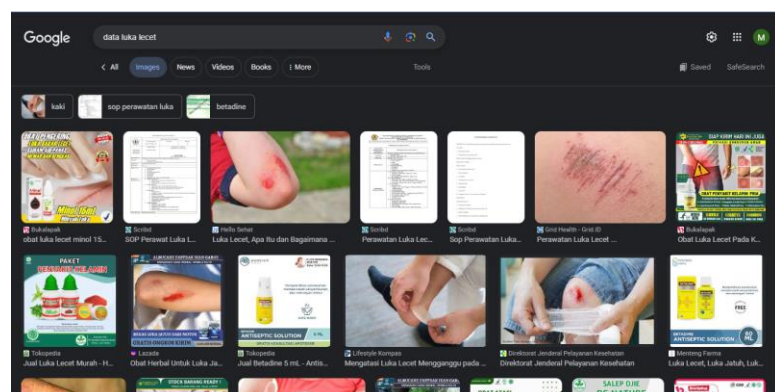
4.1.2 Pemahaman Data (*Data Understanding*) *Object Detection*

Data understanding dalam penelitian sistem deteksi adalah mencari data dan memahami data yang akan digunakan dalam tahap penelitian. Dalam tahap ini dibutuhkan data yang berkaitan dengan sistem deteksi luka ringan. Data yang dikumpulkan yaitu berupa kumpulan data gambar yang dapat diproses pada tahap modelling. Dalam pengumpulan gambar luka ringan data dapat dicari pada internet dan pada penelitian ini data yang didapat berasal dari berbagai sumber. Selanjutnya data yang sudah ditemukan kemudian dikumpulkan dan akan diproses kembali apakah data tersebut dipakai atau data tidak dapat dipakai.

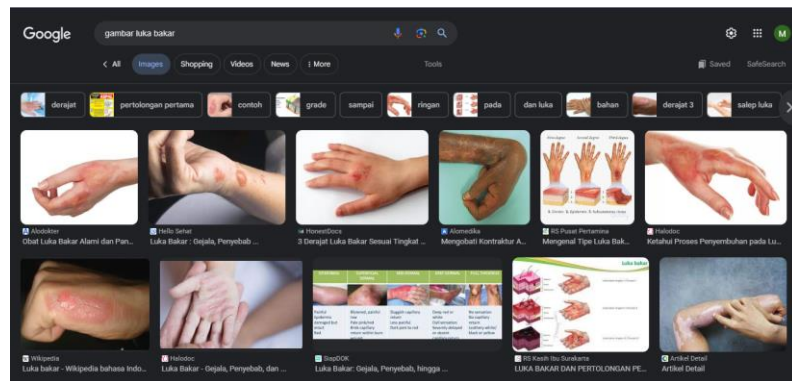
Data yang dikumpulkan	Format data yang dikumpulkan	Jumlah Data Yang dikumpulkan
Data luka lecet	Gambar (jpg,png,jpeg)	50
Data luka bakar	Gambar (jpg,png,jpeg)	50
Data luka tusuk	Gambar (jpg,png,jpeg)	50
Data luka sayat	Gambar (jpg,png,jpeg)	50

Tabel 2 Daftar data, format beserta jumlah data yang digunakan

Proses pengumpulan data dengan menggunakan *search engine* google.
Dengan menggunakan kata kunci nama luka yang akan dicari.



Gambar 7 Mencari data luka lecet dalam *search engine google*

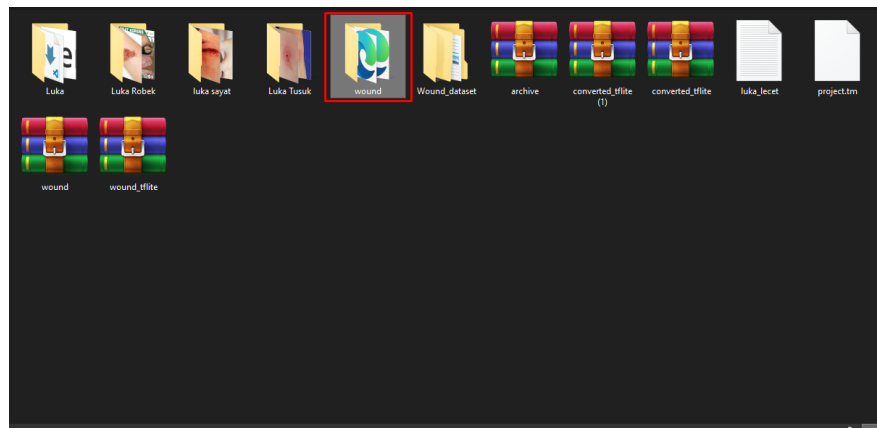


Gambar 8 Mencari data luka bakar dalam *search engine google*

Data yang sudah ditemukan dan sekiranya data dapat digunakan untuk proses *modelling* dapat diunduh kemudian data tersebut disimpan untuk digunakan pada proses *modelling*. Setelah pemahaman data dan pengumpulan data sudah dilakukan maka langkah selanjutnya yaitu persiapan data. Pada tahap pemahaman data tahap ini dapat dilakukan lagi jika model yang dibuat belum mencapai hasil yang diinginkan.

4.1.3 Persiapan Data (*Data Preparation*) *Object Detection*

Garis besar dari tahapan ini adalah evaluasi atau tahapan untuk memperbaiki masalah dalam data seperti data yang tidak dapat dibaca ketika proses *labelling*, data yang tidak sesuai dengan kriteria luka ringan, dan lain sebagainya. Pada tahapan ini merupakan tahapan yang sering dilakukan peninjauan ulang terhadap data yang akan digunakan pada saat *modelling*. Tujuan dari peninjauan ini adalah untuk menghindari terjadinya masalah sehingga ketika pada terjadi masalah terhadap data yang dikumpulkan maka hal tersebut dapat dihindari pada proses *modelling*. Kegiatan yang dilakukan pada tahapan ini antara lain yaitu memilih kasus dan parameter yang akan dianalisis (*select data*), melakukan transformasi terhadap parameter tertentu (*transformation*), dan melakukan proses pembersihan data (*cleaning data*).



Gambar 9 Membuat folder untuk data yang akan diolah

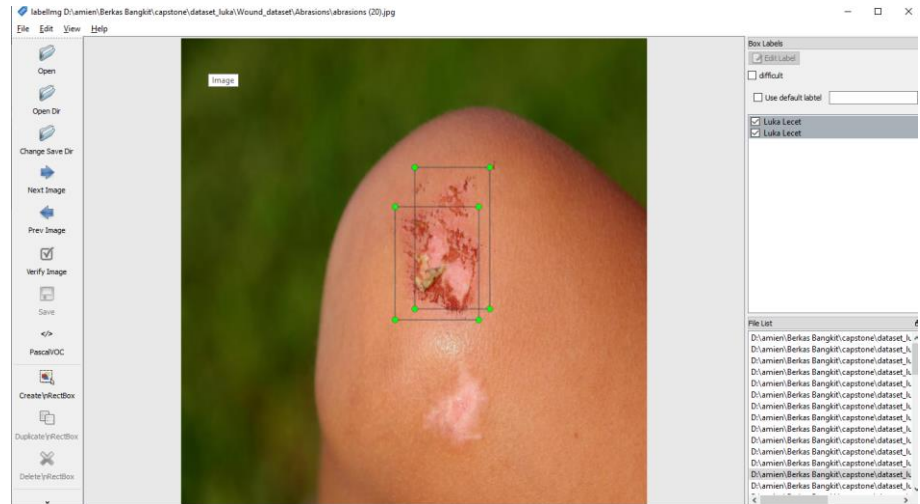
Pada gambar diatas hal yang dilakukan yaitu pembuatan folder baru untuk menyimpan data luka ringan yang telah didapatkan sebelumnya. Folder ini kedepannya akan digunakan dan di *upload* kedalam *google drive* agar ketika menggunakan data dapat dilakukan dengan mudah tidak perlu menginput secara berulang-ulang dikarenakan alat yang digunakan untuk pembuatan model yaitu menggunakan alat *google colaboratory*. Data gambar yang telah tersusun rapih dalam folder kemudian diberikan label dengan menggunakan tools labelling. Proses *labelling* dilakukan secara manual dengan memberikan label pada bagian luka.



Gambar 10 Melakukan proses labelling gambar dengan aplikasi labeling

Pada gambar diatas dilakukan proses *labelling* pada data gambar luka tusuk dengan memilih bagian yang menjadi luka kemudian

memberikannya label data tusuk. Hasil dari *labelling* gambar tersebut disimpan kedalam format *Extensible Markup Language* (XML).



Gambar 11 Memberikan segmentasi terhadap letak luka ringan

Setelah semua gambar telah diberikan label dengan menggunakan tools *labelling* selanjutnya data dalam folder tersebut di *upload* kedalam *google drive*. Tujuan dari *upload* data ke *google drive* adalah agar mempermudah untuk menjalankan model ketika sesi *google colab* telah berakhir. Tahap *data preparation* dapat dilakukan kembali jika model yang dibuat belum memenuhi hasil yang diinginkan. Tahap tersebut dapat diulang mulai dari pencarian data gambar kembali. Setelah data gambar sudah diberikan label sesuai dengan datanya selanjutnya data akan dibagi menjadi tiga bagian yaitu data latih atau data *training*, data uji atau data *testing* dan data validasi atau data *validation*. Adapun pembagian persentase untuk tiga bagian data tersebut adalah sebagai berikut.

NO	Jenis Data	Persentase
1	Data Latih atau Data <i>Training</i>	80%
2	Data Uji atau Data <i>Testing</i>	10%
3	Data Validasi atau Data <i>Validation</i>	10%

Tabel 3 Pembagian Jenis Data

Pada tabel diatas bahwa dalam pembagian data untuk data latih lebih banyak yaitu 80 persen. Hal ini dikarenakan model akan mencoba

melakukan pelatihan sebanyak mungkin sehingga ketika data akan diuji dapat memberikan hasil yang bagus. Sedangkan untuk data latih dan data uji diberikan persentase pembagian yang sama yaitu 10 persen.

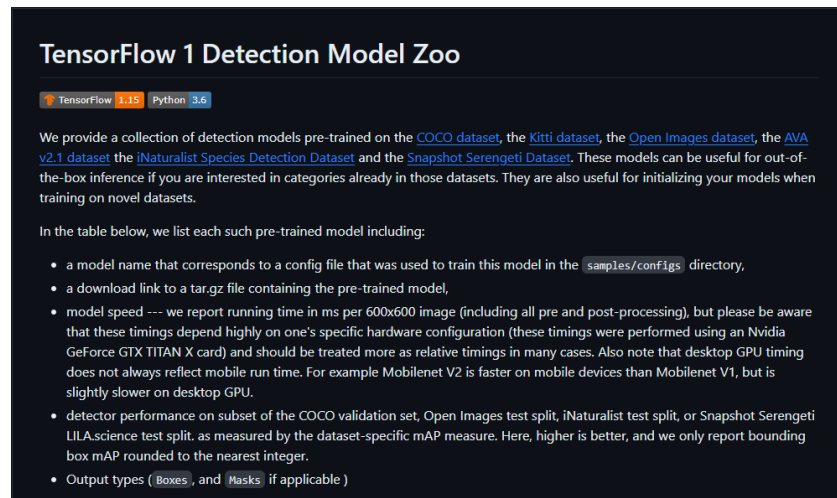
Setelah data diberi label maka jumlah label dalam setiap data menjadi memiliki jumlah label yang berbeda berikut ini jumlah data setiap gambar yang telah diberi label :

NO	Jenis Data	Jumlah label
1	Data luka lecet	268 label
2	Data luka bakar	149 label
3	Data luka tusuk	131 label
4	Data luka sayat	184 label

Tabel 4 Tabel data label dari setiap gambar

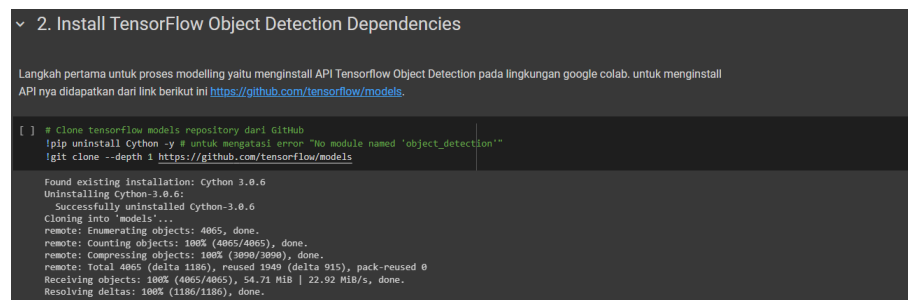
4.1.4 Pembuatan Model atau *Modelling Object Detection*

Pada tahapan ini dilakukan proses *modelling* terhadap data gambar yang telah melalui proses *select data*, *transformation*, dan *cleaning data*. Pada tahapan ini metode yang digunakan yaitu metode *deep learning* untuk melakukan *object detection* dan menggunakan *convolutional neural network* sebagai *layer* untuk memproses data gambar yang telah diolah. Sebelumnya pada *modelling* data ini digunakan alat bantu berupa *tensorflow object detection zoo*. *Tensorflow object detection zoo* merupakan kumpulan model *object detection* yang telah dilatih sebelumnya. Adapun model-model yang telah dilatih *tensorflow object detection* antara lain *COCO dataset*, *Kitti dataset*, *Open images dataset*, *AVA V2.1 dataset*, dan lain sebagainya.

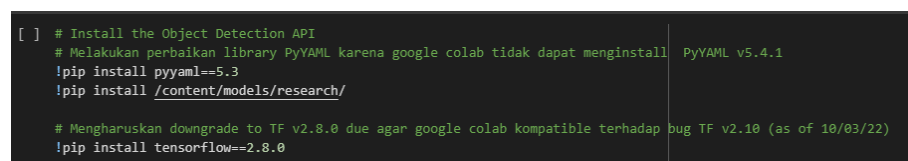


Gambar 12 *Tensorflow Detection Model Zoo*

Kemudian ke tahap selanjutnya yaitu mempersiapkan *library* yang dibutuhkan untuk proses *modelling*.

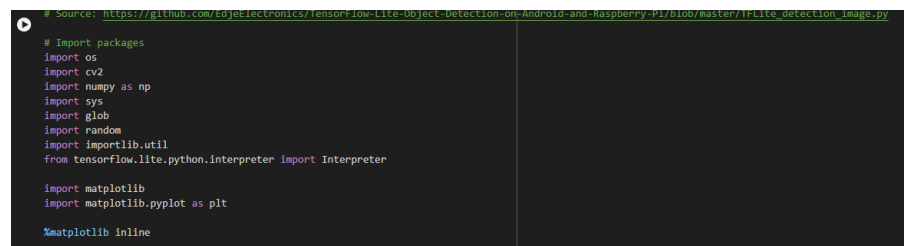


Gambar 13 *Install Tensorflow Object Detection Zoo*



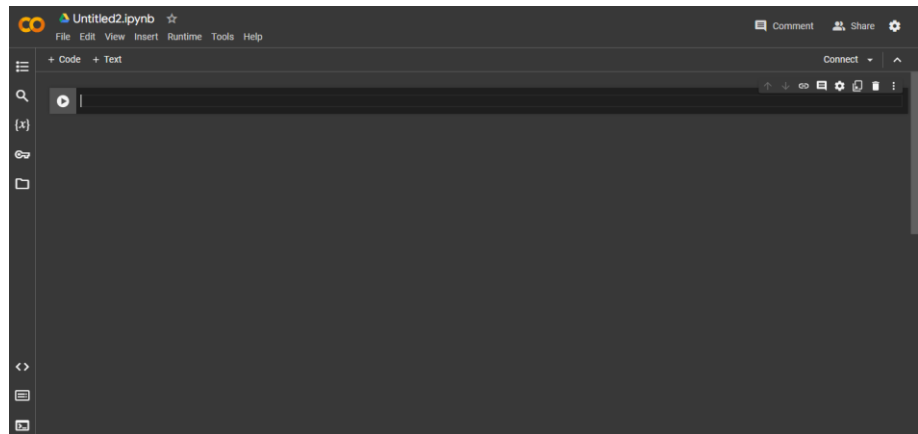
Gambar 14 *Install library yaml dan tensorflow*

Pada gambar diatas *library* yang dibutuhkan yaitu *tensorflow*, *pyYaml*, *drive*, *tarfile*, *re* (*Regular Expperession*), dan lain sebagainya.



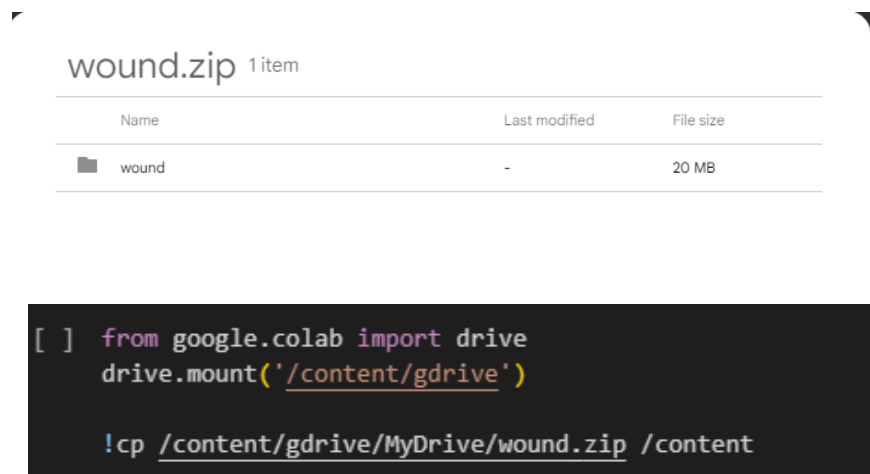
Gambar 15 *Install libraby yang dibutuhkan*

Sebelumnya model ini akan dijalankan pada *google collaboration* sehingga dibutuhkan *google collaboration* untuk menjalankan model ini.



Gambar 16 Menyiapkan *Google Colab*

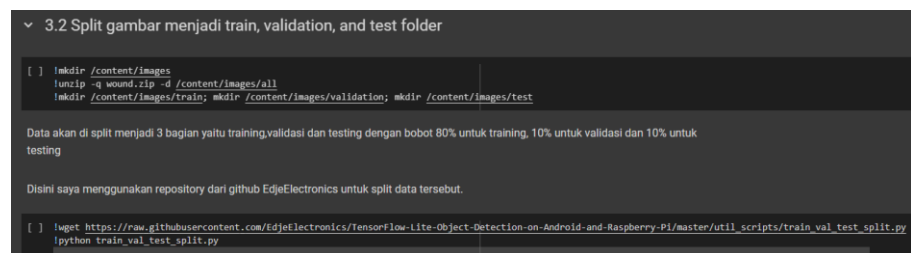
Kemudian jalankan *google colabratoy* dengan menggunakan runtime GPU yang telah disediakan oleh *google colab*. Selanjutnya *install* library *tensorflow object detection* pada *google colab* sehingga, *tensorflow object detection* dapat digunakan pada *google colab*. Selanjutnya masukkan data gambar luka ringan yang telah di *upload* pada *platform google drive*.



Gambar 17 Data diupload dan digunakan menggunakan *Google drive*

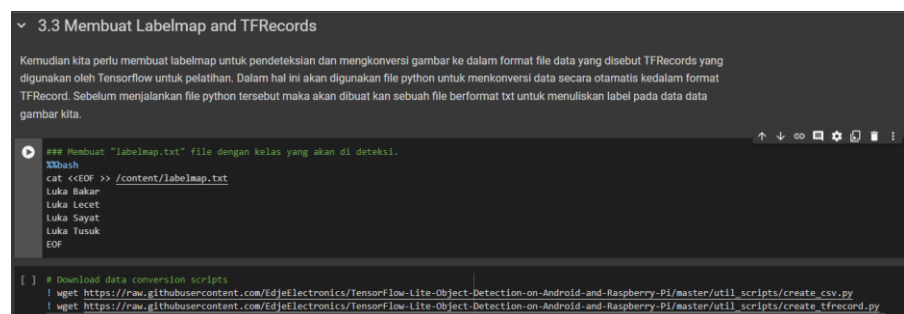
Setelah data berhasil di *upload*. Selanjutnya data akan dibagi menjadi tiga bagian yaitu bagian untuk *training*, *testing*, dan *validation*. Pada pembagian data ini komposisi atau bobot pembagian datanya

adalah 80 persen untuk data *testing*, 10 persen untuk data *validation*, 10 persenn untuk data *testing*. Pada pembagian tahap pembagian data digunakan file python pembantuan untuk melakukan *split* data. Adapun file python yang didapatkan yaitu berasal dari github *EdjeElectronics*.



Gambar 18 *Split* data menjadi data latih, data uji dan data validasi

Setelah data gambar dibagi menjadi tiga bagian atau data gambar di *split*. Selanjutnya dilakukan tahap untuk pembuatan labelmap dan *TFRecords*. *TFRecord* digunakan untuk pendeteksian dan menkonversi gambar ke dalam format file data yang disebut *TFRecords* yang digunakan oleh *tensorflow*.



Gambar 19 Membuat labelmap dan *TFRecord*

Pada tahapan ini diperlukan untuk menulis nama-nama label atau nama-nama kelas yang dipakai untuk *object detection*. Adapun kelas-kelasnya yaitu Luka Bakar, Luka Lecet, Luka Sayat, Luka Tusuk nama-nama kelas tersebut kemudian disimpan kedalam file berformat .txt dan diberi nama labelmap.txt. Setelah pembuatan nama label dan disimpan dengan format .txt dengan nama labelmap.txt selanjutnya label tersebut akan dijadikan label penamaan luka yang di deteksi. Selanjutnya yaitu mengubah data yang telah diberi label yang berformat .xml akan di konversi menjadi format .csv dengan

menggunakan *TFRecord* atau *tensorflow record*. Setelah itu data-data label gambar yang sudah dimasuka kedalam file dengan format .csv maka selanjutnya file dengan berformat .csv akan di konversi kembali kedalam format *tensorflow record* atau *TFRecord*.

```
[ ] # Membuat CSV data files and TFRecord files
!python3 create_csv.py
!python3 create_tfrecord.py --csv_input=images/train_labels.csv --labelmap=labelmap.txt --image_dir=images/train --output_path=train.tfrecord
!python3 create_tfrecord.py --csv_input=images/validation_labels.csv --labelmap=labelmap.txt --image_dir=images/validation --output_path=val.tfrecord

Successfully converted xml to csv.
Successfully converted xml to csv.
Successfully created the TFRecords: /content/train.tfrecord
Successfully created the TFRecords: /content/val.tfrecord

Kita akan menyimpan lokasi file TFRecord dan labelmap sebagai variabel sehingga kita dapat mereferensikannya nanti di google Colab ini.

[ ] train_record_fname = '/content/train.tfrecord'
    val_record_fname = '/content/val.tfrecord'
    label_map_pbtxt_fname = '/content/labelmap.pbtxt'
```

Gambar 20 Konversi *TFRecod* menjadi CSV

Selanjutnya yaitu mempersiapkan atau *setup configuration* untuk melakukan *training*. Pada *training model* ini digunakan suatu *transfer learning model* yaitu *ssd-mobile-v2-fpn-lite-320*. Transfer learning ini digunakan untuk melakukan proses *training* dengan menggunakan *ssd-mobile-v2-fpn-lite-320* model akan melakukan *training* sesuai dengan *ssd-mobile-v2-fpn-lite-320*. Dengan menggunakan model *transfer learning ssd-mobile-v2-fpn-lite-320* model ini memiliki akurasi yang bagus dan ketika model digunakan dalam jumlah data yang sedikit dapat memberikan hasil yang cukup baik. Hal ini dikarenakan model *ssd-mobile-v2-fpn-lite-320* memiliki kumpulan data pelatihan yang banyak dan memiliki kecepatan pelatihan yang baik.

4. Set Up Training Configuration

```
# Change the chosen_model variable to deploy different models available in the TF2 object detection zoo
chosen_model = 'ssd-mobilenet-v2-fpn-lite-320'

MODELS_CONFIG = {
    'ssd-mobilenet-v2': {
        'model_name': 'ssd_mobilenet_v2_320x320_coco17_tpu-8',
        'base_pipeline_file': 'ssd_mobilenet_v2_320x320_coco17_tpu-8.config',
        'pretrained_checkpoint': 'ssd_mobilenet_v2_320x320_coco17_tpu-8.tar.gz',
    },
    'efficientdet-d0': {
        'model_name': 'efficientdet_d0_coco17_tpu-32',
        'base_pipeline_file': 'ssd_efficientdet_d0_512x512_coco17_tpu-8.config',
        'pretrained_checkpoint': 'efficientdet_d0_coco17_tpu-32.tar.gz',
    },
    'ssd-mobilenet-v2-fpn-lite-320': {
        'model_name': 'ssd_mobilenet_v2_fpn_lite_320x320_coco17_tpu-8',
        'base_pipeline_file': 'ssd_mobilenet_v2_fpn_lite_320x320_coco17_tpu-8.config',
        'pretrained_checkpoint': 'ssd_mobilenet_v2_fpn_lite_320x320_coco17_tpu-8.tar.gz',
    },
    # The centernet model isn't working as of 9/10/22
    #'centernet-mobilenet-v2': {
    #     'model_name': 'centernet_mobilenetv2fpn_512x512_coco17_od',
    #     'base_pipeline_file': 'pipeline.config',
    #     'pretrained_checkpoint': 'centernet_mobilenetv2fpn_512x512_coco17_od.tar.gz',
    # }
}
```

Gambar 21 konfigurasi terhadap model *transfer learning mobilenet-v2*

Selanjutnya yang perlu di *setup* yaitu membuat parameter pada model seperti memberikan nilai *num_step* kemudian *batch_size* set lokasi penyimpanan, dan lain sebagainya.

```
# Set training parameters for the model
num_steps = 40000

if chosen_model == 'efficientdet-d0':
    batch_size = 4
else:
    batch_size = 16

[ ] # Set file locations and get number of classes for config file
pipeline_fname = '/content/models/mymodel/' + base_pipeline_file
fine_tune_checkpoint = '/content/models/mymodel/' + model_name + '/checkpoint/ckpt-0'

def get_num_classes(pbxtxt_fname):
    from object_detection.utils import label_map_util
    label_map = label_map_util.load_labelmap(pbxtxt_fname)
    categories = label_map_util.convert_label_map_to_categories(
        label_map, max_num_classes=90, use_display_name=True)
    category_index = label_map_util.create_category_index(categories)
    return len(category_index.keys())
num_classes = get_num_classes(label_map_pbxtxt_fname)
print('Total classes:', num_classes)

Total classes: 4
```

Gambar 22 Konfigurasi *Hyperparameter*


```

import re

%cd /content/models/mymodel
print('writing custom configuration file')

with open(pipeline_fname) as f:
    s = f.read()
with open('pipeline_file.config', 'w') as f:

    # Set fine_tune_checkpoint path
    s = re.sub('fine_tune_checkpoint: ".*?"',
                'fine_tune_checkpoint: "{}".format(fine_tune_checkpoint), s)

    # Set tfrecord files for train and test datasets
    s = re.sub(
        '(input_path: ".*?")(PATH_TO_BE_CONFIGURED/train)(.*?)', 'input_path: "{}".format(train_record_fname), s)
    s = re.sub(
        '(input_path: ".*?")(PATH_TO_BE_CONFIGURED/val)(.*?)', 'input_path: "{}".format(val_record_fname), s)

    # Set label_map_path
    s = re.sub(
        'label_map_path: ".*?"', 'label_map_path: "{}".format(label_map_pbtxt_fname), s)

    # Set batch_size
    s = re.sub('batch_size: [0-9]+',
                'batch_size: {}'.format(batch_size), s)

    # Set training steps, num_steps
    s = re.sub('num_steps: [0-9]+',

```

Gambar 23 Konfigurasi *Hyperparameter 2*

Setelah melakukan *setup* untuk persiapan *training model* dapat dilihat dalam *transfer learning* yang dipakai . Dalam transfer learning yang digunakan dapat dilihat arsitekturnya mulai dari *y_scale*, *x_scale*, *aspect_ratio*, *loss*, *optimizer*, *eval_input_reader* dan lain sebagainya.

```

# (Optional) Display the custom configuration file's contents
!cat /content/models/mymodel/pipeline_file.config

# SSD with Mobilenet v2 FPN-lite (go/fpn-lite) feature extractor, shared box
# predictor and focal loss (a mobile version of Retinanet).
# Retinanet: see Lin et al, https://arxiv.org/abs/1708.02002
# Trained on COCO, initialized from Imagenet classification checkpoint
# Train on TPU-8
#
# Achieves 22.2 mAP on COCO17 Val

model {
  ssd {
    inplace_batchnorm_update: true
    freeze_batchnorm: false
    num_classes: 4
    box_coder {

```

```

matcher {
  argmax_matcher {
    matched_threshold: 0.5
    unmatched_threshold: 0.5
    ignore_thresholds: false
    negatives_lower_than_unmatched: true
    force_match_for_each_row: true
    use_matmul_gather: true
  }
}
similarity_calculator {
  iou_similarity {
  }
}
}
encode_background_as_zeros: true
anchor_generator {
  multiscale_anchor_generator {
    min_level: 3
    max_level: 7
    anchor_scale: 4.0
    aspect_ratios: [1.0, 2.0, 0.5]
    scales_per_octave: 2
  }
}
image_resizer {
  fixed_shape_resizer {
    height: 320
    width: 320
  }
}

```

Gambar 24 Arsitektur *transfer learning* yang digunakan

Setelah semua persiapan untuk melakukan *training* sudah selesai. Maka, selanjutnya yaitu proses melakukan *training*. *Google colab* akan melakukan training pada model yang telah dipersiapkan sebelumnya dengan data yang telah dipersiapkan sebelumnya juga. Untuk proses *training* memakan waktu yang cukup lama. Hal ini dikarenakan model akan melakukan *training* setiap data gambar sesuai dengan *transfer learning* yang dipakai.

```

# Run training!
python /content/models/research/object_detection/model_main_tf2.py \
  --pipeline_config_path=(pipeline_file) \
  --model_dir=(model_dir) \
  --alsologtostderr \
  --num_train_steps=(num_steps) \
  --sample_1_of_n_eval_examples=1

/usr/local/lib/python3.10/dist-packages/tensorflow_io/python/ops/_init_.py:98: UserWarning: unable to load libtensorflow_io_plugins.so: unable to open file:
caused by: ['/usr/local/lib/python3.10/dist-packages/tensorflow_io/python/ops/libtensorflow_io_plugins.so: undefined symbol: _ZN3tsl5mutexCIev']
warnings.warn(f"unable to load libtensorflow_io_plugins.so: {e}")
/usr/local/lib/python3.10/dist-packages/tensorflow_io/python/ops/_init_.py:104: UserWarning: file system plugins are not loaded: unable to open file: libtens
caused by: ['/usr/local/lib/python3.10/dist-packages/tensorflow_io/python/ops/libtensorflow_io.so: undefined symbol: _ZNK10tensorflow4data11DatasetBase8Finali
warnings.warn(f"file system plugins are not loaded: {e}")
/usr/local/lib/python3.10/dist-packages/tensorflow_addons/utils/tfa_eol_msg.py:23: UserWarning:

TensorFlow Addons (TFA) has ended development and introduction of new features.
TFA has entered a minimal maintenance and release mode until a planned end of life in May 2024.
Please modify downstream libraries to take dependencies from other repositories in our TensorFlow community (e.g. Keras, Keras-CV, and Keras-NLP).

For more information see: https://github.com/tensorflow/addons/issues/2887

warnings.warn(
/usr/local/lib/python3.10/dist-packages/tensorflow_addons/utils/ensure_tf_install.py:53: UserWarning: Tensorflow Addons supports using Python ops for all Tens
The versions of TensorFlow you are currently using is 2.8.0 and is not supported.
Some things might work, some things might not.
If you were to encounter a bug, do not file an issue.
If you want to make sure you're using a tested and supported configuration, either change the TensorFlow version or the TensorFlow Addons's version.
You can find the compatibility matrix in TensorFlow Addon's readme:
https://github.com/tensorflow/addons/blob/master/README.md

```

Gambar 25 Proses *training* model

```
INFO:tensorflow:Step 39800 per-step time 0.307s
I1210 08:55:19.154525 133225358832256 model_lib_v2.py:705] Step 39800 per-step time 0.307s
INFO:tensorflow: {'loss/classification_loss': 0.024170881,
'loss/localization_loss': 0.026692796,
'loss/regularization_loss': 0.051858713,
'loss/total_loss': 0.10192231,
'learning_rate': 0.008252861}
I1210 08:55:19.154900 133225358832256 model_lib_v2.py:708] {'loss/classification_loss': 0.024170881,
'loss/localization_loss': 0.026692796,
'loss/regularization_loss': 0.051858713,
'loss/total_loss': 0.10192231,
'learning_rate': 0.008252861}
INFO:tensorflow:Step 39900 per-step time 0.305s
I1210 08:55:49.717071 133225358832256 model_lib_v2.py:705] Step 39900 per-step time 0.305s
INFO:tensorflow: {'loss/classification_loss': 0.025363076,
'loss/localization_loss': 0.029592525,
'loss/regularization_loss': 0.05102724,
'loss/total_loss': 0.10598284,
'learning_rate': 0.008097499}
I1210 08:55:49.717438 133225358832256 model_lib_v2.py:708] {'loss/classification_loss': 0.025363076,
'loss/localization_loss': 0.029592525,
'loss/regularization_loss': 0.05102724,
'loss/total_loss': 0.10598284,
'learning_rate': 0.008097499}
INFO:tensorflow:Step 40000 per-step time 0.307s
I1210 08:56:20.417769 133225358832256 model_lib_v2.py:705] Step 40000 per-step time 0.307s
INFO:tensorflow: {'loss/classification_loss': 0.03695945,
'loss/localization_loss': 0.03396865,
'loss/regularization_loss': 0.050996218,
'loss/total_loss': 0.12192431,
'learning_rate': 0.007943453}
I1210 08:56:20.418157 133225358832256 model_lib_v2.py:708] {'loss/classification_loss': 0.03695945,
```

Gambar 26 Proses *training* model 2

Setelah proses *training* selesai dan memakan waktu sekitar 2 jam maka akan terlihat hasilnya mulai dari *classification_loss*, *localization_loss*, *regularization_loss*, dan *total_loss*. Hasil akhirnya yaitu

Jenis Loss	Nilai Loss
<i>classification_loss</i>	0.03695945
<i>localization_loss</i>	0.03396865
<i>regularization_loss</i>	0.050996218
<i>total_loss</i>	0.12192431

Tabel 5 Hasil akhir *loss* model yang telah dilatih

Dari nilai loss diatas dapat diartikan bahwa data yang di *training* mendapatkan hasil yang baik dan dapat digunakan dan dapat dapat di *deploy* ke perangkat lain. Langkah selanjutnya yaitu melakukan *convert model* yang telah di *training* kedalam format *tensorflow lite*. *Loss* yang digunakan dalam deteksi objek yaitu *localization loss* yang berfungsi untuk mengetahui *error rate* dari 23 *bounding box* dan *ground truth box*. Kemudian, *classification loss* yang digunakan untuk mengetahui *error rate* dari kelas aktual yang ada dalam *ground truth* dan kelas hasil dari prediksi. Selanjutnya, *regularization loss* digunakan untuk mengetahui serangkain teknik yang berbeda yang tujuannya untuk menurunkan kompleksitas model *neural network* selama pelatihan sehingga mencegah dari *overfitting*.

```

# Make a directory to store the trained TFLite model
!mkdir /content/custom_model_lite
output_directory = '/content/custom_model_lite'

# Path to training directory (the conversion script automatically chooses the highest checkpoint file)
last_model_path = '/content/training'

!python /content/models/research/object_detection/export_tflite_graph_tf2.py \
--trained_checkpoint_dir {last_model_path} \
--output_directory {output_directory} \
--pipeline_config_path {pipeline_file}

```

Gambar 27 Konversi model kedalam *TFLite*

Pada gambar diatas adalah proses untuk melakukan *convert* model yang telah dibuat kedalam format *tensorflow lite*.

4.1.5 Evaluasi atau *Evaluation Object Detection*

Pada tahapan ini dilakukan proses evaluasi terhadap model yang telah dibuat sebelumnya. Pada proses evaluasi ini akan dilihat apakah model dapat dilanjutkan ke tahap berikutnya yaitu tahap *deployment* atau model perlu diperbaiki kembali sehingga mendapatkan hasil yang lebih bagus. Untuk mengevaluasi model yang telah dibuat digunakan *tensorboard* untuk melihat hasil dari *training* model yang telah dilakukan. Untuk menggunakan *tensorboard* terlebih dahulu *load* atau memanggil *tensorboard* kedalam *google colab*.

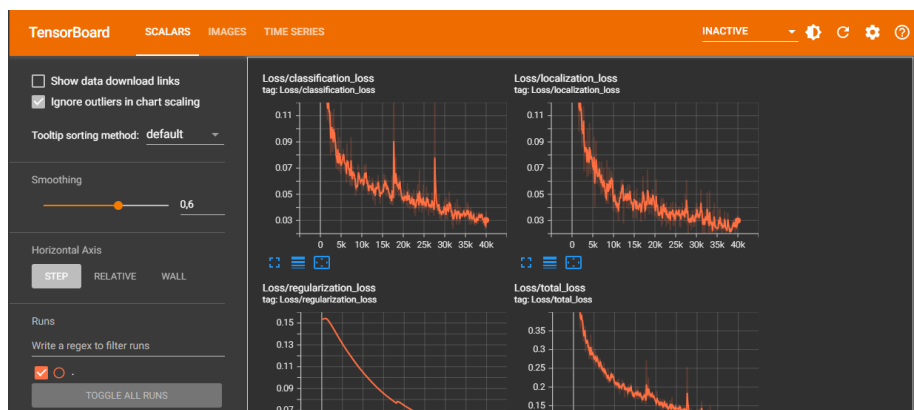
```

[ ] %load_ext tensorboard
    %tensorboard --logdir '/content/training/train'

```

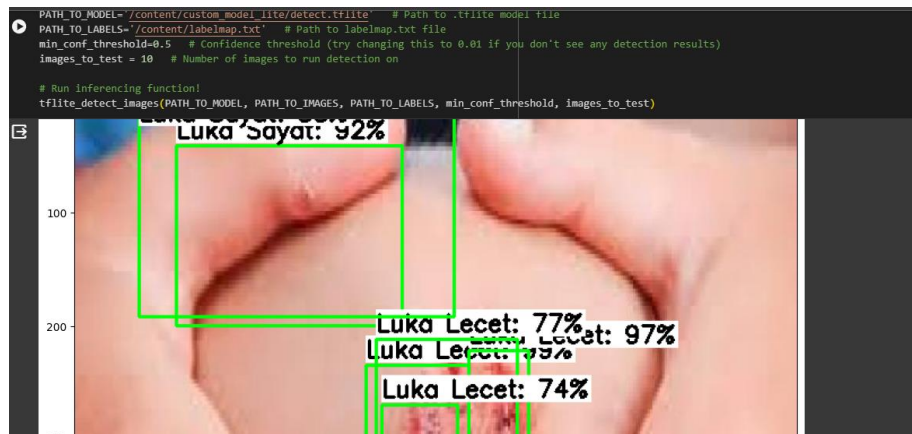
Gambar 28 Load Tensorboard

Setelah itu *tensorboard* akan muncul dalam *google colab* untuk memberikan hasil evaluasi hasil dari *training model* yang dilakukan sebelumnya.



Gambar 29 Hasil evaluasi menggunakan *Tensorboard*

Pada *tensorboard* dapat dilihat secara grafik untuk *loss* terhadap model *training* yang telah dibuat. Terlihat dalam *tensorboard* bahwa *loss* yang didapat mengalami penurunan artinya bahwa model tersebut dapat mengenali data gambar dengan baik sehingga tidak banyak *loss* yang diberikan. Setelah itu untuk melakukan evaluasi yang selanjutnya terhadap model yang telah dibuat yaitu dengan menggunakan *metric Mean Average Precision (MAP)*. Pada evaluasi MAP ini akan diberikan hasil presentasi untuk deteksi dari masing-masing luka yang telah di *training*. Metrik *MAP* digunakan karena *MAP* adalah metrik yang bagus untuk membandingkan model secara kuantitatif. Namun, *MAP* ini tidak terlalu intuitif untuk membantu memahami seberapa baik kinerja model dalam mendeteksi objek dalam gambar.



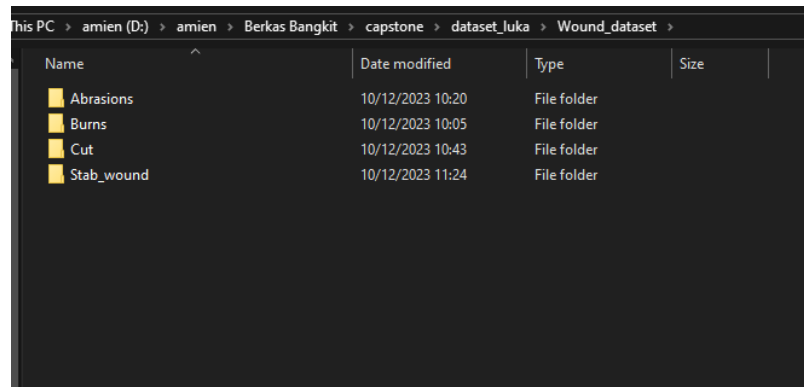
Gambar 30 Hasil uji coba model

Pada gambar diatas terdapat presentase hasil dari deteksi gambar luka ringan pada gambar tersebut terdeteksi terdapat luka lecet dengan presentase 77% dan terdapat luka sayat sebesar 92%. Hal ini menunjukkan bahwa hasil dari deteksi mendapatkan akurasi yang baik untuk melakukan pendeteksian terhadap luka ringan.

4.1.6 Persiapan Data (Data Preparation) *Classification*

Data yang dibutuhkan dalam proses klasifikasi ini adalah berupa data gambar luka ringan seperti luka lecet, luka bakar, luka sayat, dan luka tusuk. Data yang dipakai sama dengan data untuk

melakukan *object detection* bedanya data untuk klasifikasi data yang tidak diberikan label secara manual sehingga untuk menentukan kelas-kelas dari data tersebut kita harus memisahkannya masing-masing.



Gambar 31 Membuat folder data sesuai kelas

Dengan membuat folder utama kemudian berisikan folder-folder dari kelas-kelas data luka ringan. Sehingga, ketika melakukan klasifikasi menjadi lebih mudah karena data telah sesuai dengan foldernya masing-masing. Pada proses klasifikasi data yang digunakan bertambah dari data yang sebelumnya yaitu data untuk *object detection*. Untuk data klasifikasi ditambah beberapa data sehingga jumlah datanya yaitu 325 gambar terdiri dari 85 data gambar luka lecet (*Abrasions*), 83 data gambar luka bakar (*Burns*), 78 data gambar luka sayat (*Cut*), dan 79 data luka tusuk (*stab wound*).

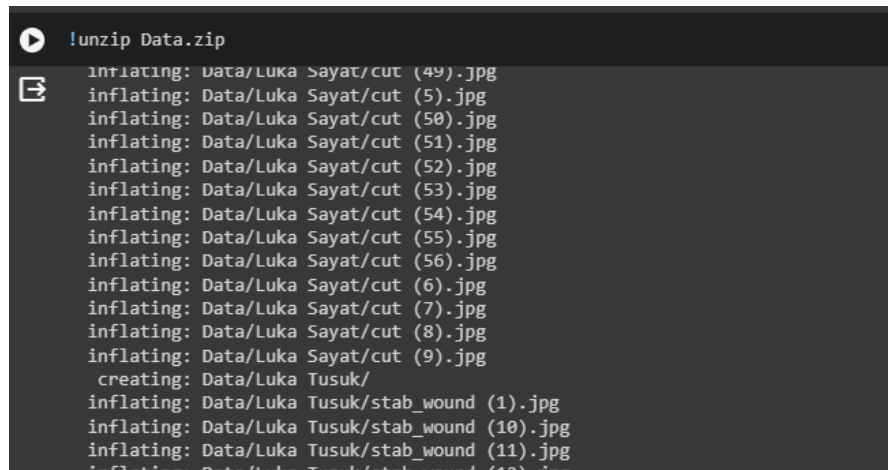
4.1.7 Pembuatan Model atau *Modelling Classification*

Langkah pertama yang dilakukan ketika membuat model *machine learning* adalah memasukan semua *library* yang dibutuhkan.

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import Input
from tensorflow.keras import models, layers
import numpy as np
import matplotlib.pyplot as plt
import pathlib
import cv2 as cv
import numpy as np
import pandas as pd
import os
import time
from skimage.filters import gaussian
from skimage.segmentation import active_contour
```

Gambar 32 *Import library* yang dibutuhkan

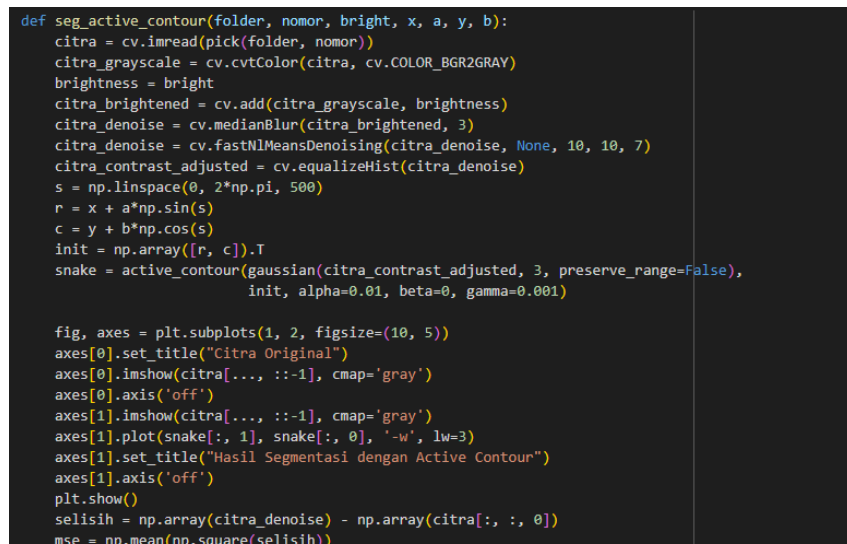
Selanjutnya memasukan data gambar yang telah diurutkan berdasarkan folder-foldernya setelah itu lakukan ekstrasi terhadap data tersebut dikarenakan data tersebut berbentuk format .zip.



```
!unzip Data.zip
inflating: Data/Luka Sayat/cut (49).jpg
inflating: Data/Luka Sayat/cut (5).jpg
inflating: Data/Luka Sayat/cut (50).jpg
inflating: Data/Luka Sayat/cut (51).jpg
inflating: Data/Luka Sayat/cut (52).jpg
inflating: Data/Luka Sayat/cut (53).jpg
inflating: Data/Luka Sayat/cut (54).jpg
inflating: Data/Luka Sayat/cut (55).jpg
inflating: Data/Luka Sayat/cut (56).jpg
inflating: Data/Luka Sayat/cut (6).jpg
inflating: Data/Luka Sayat/cut (7).jpg
inflating: Data/Luka Sayat/cut (8).jpg
inflating: Data/Luka Sayat/cut (9).jpg
creating: Data/Luka Tusuk/
inflating: Data/Luka Tusuk/stab_wound (1).jpg
inflating: Data/Luka Tusuk/stab_wound (10).jpg
inflating: Data/Luka Tusuk/stab_wound (11).jpg
inflating: Data/Luka Tusuk/stab_wound (12).jpg
```

Gambar 33 Ekstrak Data

Setelah itu lakukan segmentasi gambar dengan menggunakan metode *active contour*. Pada segmetanasi gambar menggunakan *active contour* gambar akan membuat area yang menjadi titik pusat dari luka tersebut dengan ditandai dengan garis dari *active contour* tersebut.



```
def seg_active_contour(folder, nomor, bright, x, a, y, b):
    citra = cv.imread(pick(folder, nomor))
    citra_grayscale = cv.cvtColor(citra, cv.COLOR_BGR2GRAY)
    brightness = bright
    citra_brightened = cv.add(citra_grayscale, brightness)
    citra_denoise = cv.medianBlur(citra_brightened, 3)
    citra_denoise = cv.fastNlMeansDenoising(citra_denoise, None, 10, 10, 7)
    citra_contrast_adjusted = cv.equalizeHist(citra_denoise)
    s = np.linspace(0, 2*np.pi, 500)
    r = x + a*np.sin(s)
    c = y + b*np.cos(s)
    init = np.array([r, c]).T
    snake = active_contour(gaussian(citra_contrast_adjusted, 3, preserve_range=False),
                           init, alpha=0.01, beta=0, gamma=0.001)

    fig, axes = plt.subplots(1, 2, figsize=(10, 5))
    axes[0].set_title("Citra Original")
    axes[0].imshow(citra[...,:-1], cmap='gray')
    axes[0].axis('off')
    axes[1].imshow(citra[...,:-1], cmap='gray')
    axes[1].plot(snake[:, 1], snake[:, 0], '-w', lw=3)
    axes[1].set_title("Hasil Segmentasi dengan Active Contour")
    axes[1].axis('off')
    plt.show()
    selisih = np.array(citra_denoise) - np.array(citra[:, :, 0])
    mse = np.mean(np.square(selisih))
```

Gambar 34 Segmentasi data menggunakan *active contour*



Gambar 35 Hasil segmentasi data

Setelah data gambar telah dilakukan segmentasi selanjutnya gambar akan dibuat model *machine learning* dengan menggunakan metode *convolutional neural network* (CNN). Adapun CNN yang dipakai menggunakan bantuan dari *transfer learning mobilenetv2*. Sebelum membuat model *machine learning* data gambar akan di *preprocessing* dengan menggunakan *augmented image* dari *library keras*. Setelah gambar dilakukan proses *augmentasi* selanjutnya data gambar akan di pisahkan menjadi dua bagian yaitu data untuk data latih dan data untuk data validasi. Untuk persentase pembagian data yaitu 80 persen dan 20 persen.. 80 persen untuk data latih dan 20 persen untuk data validasi. Kemudian setelah dibagi menjadi data latih dan data validasi selanjutnya data gambar akan di *resize* atau diubah ukuran gambarnya agar semua ukuran gambar sesuai dan merata. Untuk ukuran yang dipakai yaitu ukuran *224 pixels x 224 pixels*.


```

MAIN_DIR = "Data/"

train_datagen = ImageDataGenerator(rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    rotation_range=30,
    width_shift_range=0.2,
    height_shift_range=0.2,
    vertical_flip=True,
    validation_split=0.2) # set validation split

train_generator = train_datagen.flow_from_directory(
    MAIN_DIR,
    target_size=(224, 224),
    class_mode='categorical',
    subset='training') # set as training data

validation_generator = train_datagen.flow_from_directory(
    MAIN_DIR, # same directory as training data
    target_size=(224, 224),
    class_mode='categorical',
    subset='validation') # set as validation data

```

Gambar 36 Melakukan *Augmentasi Gambar*

Dalam proses *augmentasi gambar* terdapat beberapa langkah seperti gambar diperbesar, gambar di rotasi, gambar dibolak-balik dan lain sebagainya. Setelah proses *preprocessing* pada telah selesai langkah selanjutnya yaitu pembuatan *layer CNN* untuk melakukan proses *training* gambar.

```
model.summary()
```

Model: "model_8"

Layer (type)	Output Shape	Param #	Connected to
input_9 (InputLayer)	[(None, 224, 224, 3)]	0	[]
Conv1 (Conv2D)	(None, 112, 112, 32)	864	['input_9[0][0]']
bn_conv1 (BatchNormalization)	(None, 112, 112, 32)	128	['Conv1[0][0]']
Conv1_relu (ReLU)	(None, 112, 112, 32)	0	['bn_conv1[0][0]']
expanded_conv_depthwise (DepthwiseConv2D)	(None, 112, 112, 32)	288	['Conv1_relu[0][0]']
expanded_conv_depthwise_BN (BatchNormalization)	(None, 112, 112, 32)	128	['expanded_conv_depthwise[0][0]']
expanded_conv_depthwise_relu (ReLU)	(None, 112, 112, 32)	0	['expanded_conv_depthwise_BN[0][0]']
expanded_conv_project (Conv2D)	(None, 112, 112, 16)	512	['expanded_conv_depthwise_relu[0][0]']
expanded_conv_project_BN (BatchNormalization)	(None, 112, 112, 16)	64	['expanded_conv_project[0][0]']

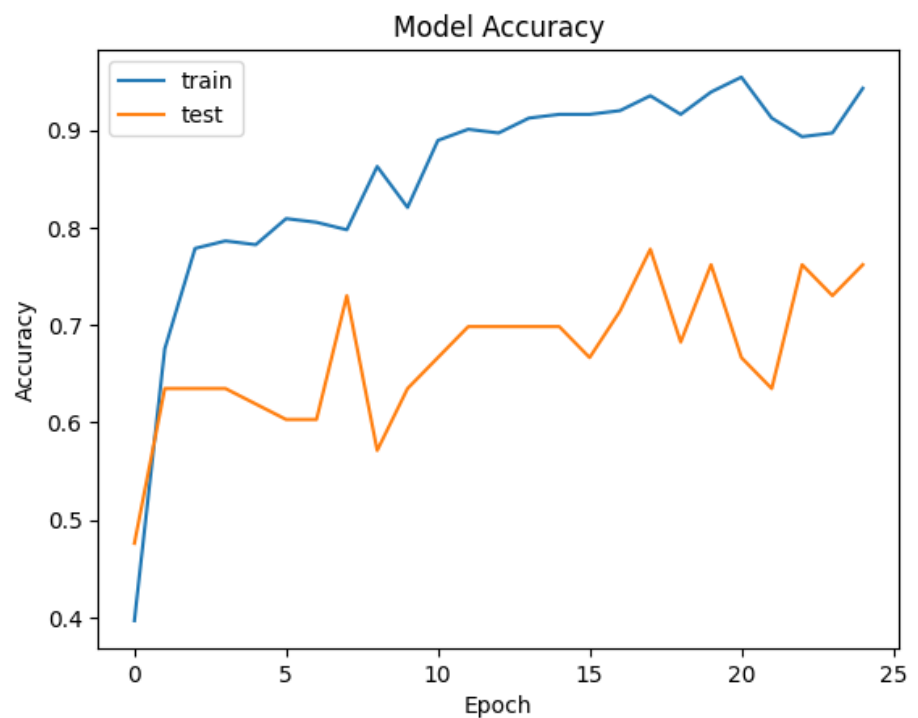
Gambar 37 Membuat *layer Convolutional Neural Network* dengan *transfer learning*

Untuk *layer CNN* yang dipakai yaitu *layer* dari *transfer learning mobilenetv2*. Kemudian dengan tambahan 3 *layer dense* dengan masing-masing *activation* yaitu *relu* dan *softmax*. Setelah itu lakukan konfigurasi untuk *hyperparameter* pada model seperti *learning rate*, *optimizer*, *loss*, *metrics*. Setelah semua konfigurasi telah dilakukan selanjutnya data yang sudah siap akan diproses *training* dengan metode *convolutional neural network* dengan percobaan sebanyak 20

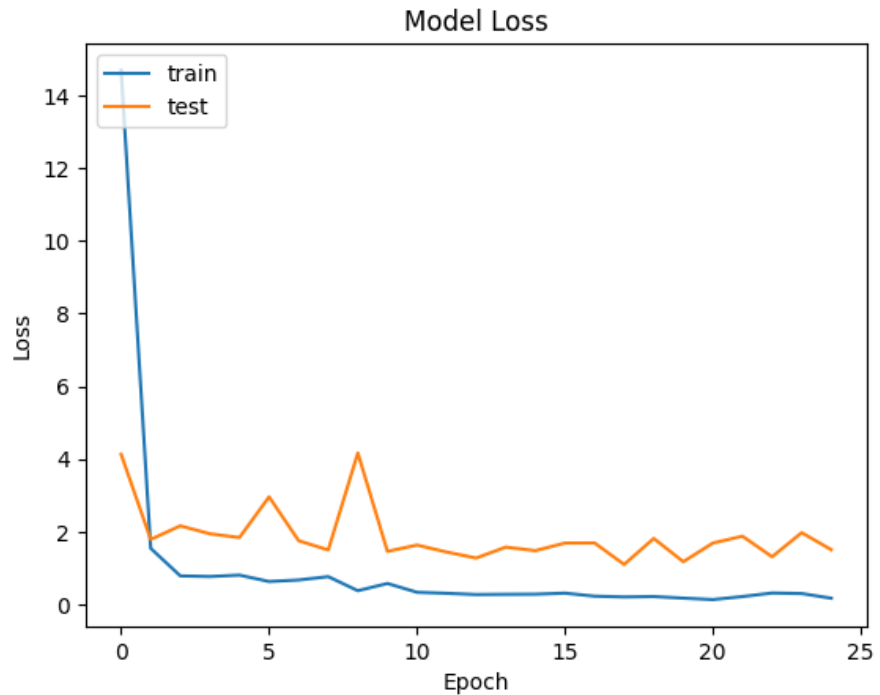
percobaan. Model akan melatih data dengan waktu yang tidak menentu.

4.1.8 Evaluasi atau Evaluation Classification

Pada tahapan evaluasi model klasifikasi ini adalah melihat hasil dari model yang sudah dilatih. Model dapat memberikan sebuah grafik dari hasil pelatihan data tersebut. Grafik yang ditampilkan dapat berupa grafik akurasi dan grafik *loss*.



Gambar 38 Metrik Akurasi model



Gambar 39 Metrik *loss* model

Dari kedua grafik diatas menunjukan bahwa data yang telah dilatih memiliki akurasi yang baik. Namun, untuk memvalidasi data dari hasil pelatihan akurasi untuk validasi kurang baik dibandingkan dengan hasil akurasi pelatihan. Hal ini dapat menyebabkan *overfitting*. salah satunya penyebab terjadinya *overfitting* adalah datanya itu sendiri. Karena data yang dipakai sangat sedikit dengan jumlah sekitar 230 data gambar hal ini yang menyebabkan data mengalami *overfitting*. Sehingga model dari data klasifikasi ini dapat memberikan hasil yang tidak sesuai ketika mendapatkan data yang tidak terdapat dalam data latih tersebut. Namun, untuk sementara model ini dapat kita pakai dan dapat kita lakukan *deployment* kedalam sistem berbasis *mobile*. Untuk melakukan *deployment* model harus dirubah atau di konversi menjadi format *tensorflow lite*.

```

# Fetch the Keras session and save the model
# The signature definition is defined by the input and output tensors,
# and stored with the default serving key
import tempfile
import os

MODEL_DIR = tempfile.gettempdir()
version = 1
export_path = os.path.join(MODEL_DIR, str(version))
print('export_path = {}'.format(export_path))

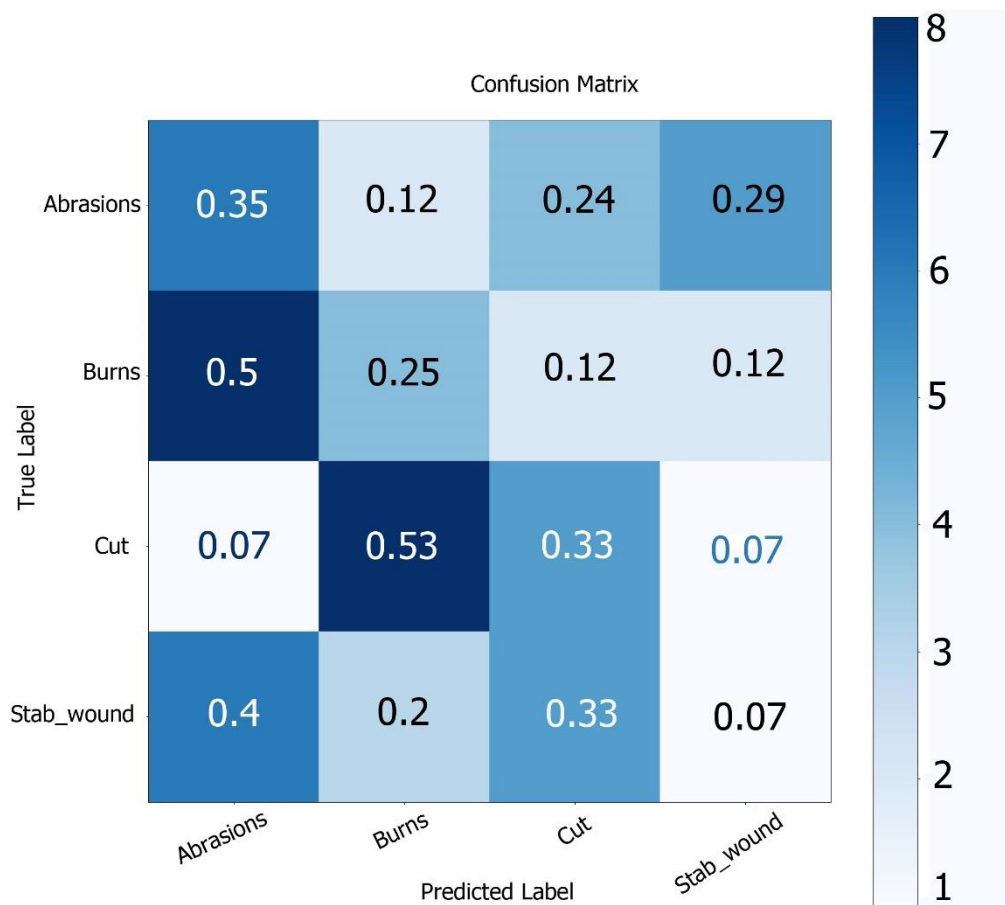
tf.keras.models.save_model(
    model,
    export_path,
    overwrite=True,
    include_optimizer=True,
    save_format=None,
    signatures=None,
    options=None
)

print('\nSaved model:')
!ls -l {export_path}

```

Gambar 40 Export model menjadi TFLite

Untuk melihat hasil keterikatan antara data luka itu sendiri dapat dilihat menggunakan *confusion matrix*.



Gambar 41 Hasil *Confusion Matrix* model

Pada *confusion matrix* diatas terlihat bahwa terdapat banyak kemiripan data antara data luka ringan dan data luka bakar hal tersebut juga yang mengakibatkan hasil dari validasi data menjadi kurang baik. Sehingga untuk memperbaiki hal tersebut adalah memperbaiki datanya lebih dalam lagi. Untuk metrik lainnya dapat dilihat dari gambar dibawah ini.

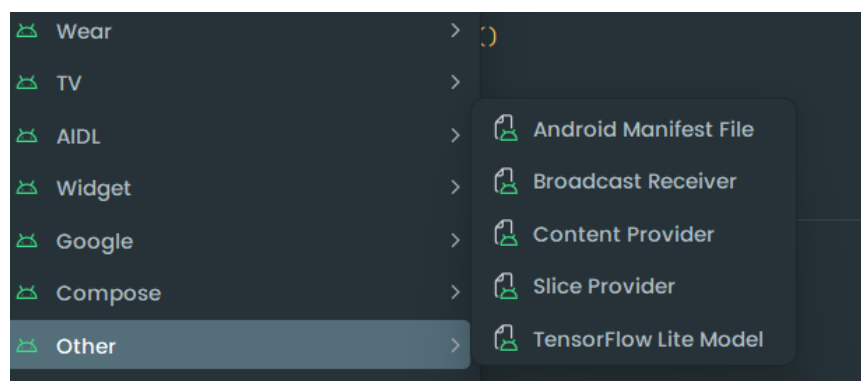
	precision	recall	f1-score	support
0	0.87	0.72	0.79	18
1	0.86	0.82	0.84	22
2	0.61	0.69	0.65	16
3	0.89	0.86	0.87	28
4	0.84	1.00	0.91	16
accuracy			0.82	100
macro avg	0.81	0.82	0.81	100
weighted avg	0.83	0.82	0.82	100

Gambar 42 Hasil *Precision*, *recall* dan *F1-Score*

Pengujian model terhadap metrik *precision*, *recall*, dan *f1-score*.

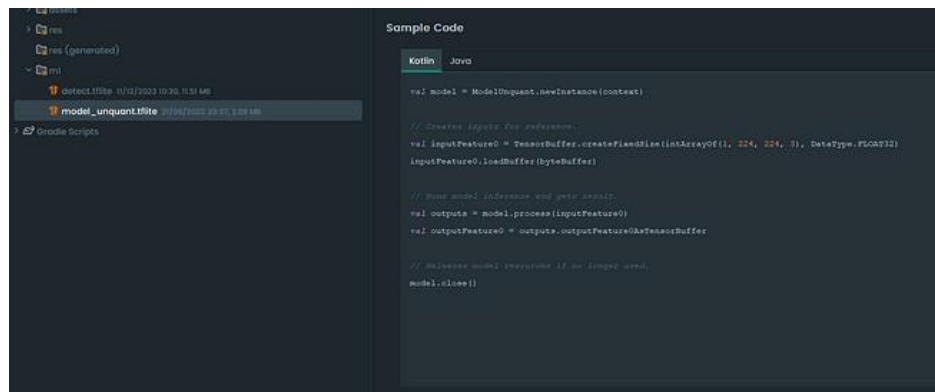
4.1.9 Deployment model

Pada tahapan *deployment* adalah tahapan untuk mengimplementasikan model yang telah dibuat. Dalam hal ini model yang akan di *deploy* adalah model klassifikasi luka ringan dengan menggunakan *teachable machine*. Model akan di *deploy* kedalam perangkat berbasis *mobile*. Dengan menggunakan bahasa Kotlin untuk proses *deployment* model kedalam perangkat berbasis *mobile*.



Gambar 43 Import *Tensorflow Lite* model dalam *Android Studio*

Pertama *import* terlebih dahulu *assets* untuk *tensorflow lite* didalam aplikasi *android studio*. Kemudian setelah menginsalasi *assets tensorflow lite* selanjutnya melihat spesifikasi model. Hal ini dilakukan karena akan berpengaruh pada ukuran bit gambar.



Gambar 44 Mengatur ukuran bit

Selanjutnya yaitu mengatur ukuran bit sesuai dengan model yang telah dibuat. Sebelumnya model yang dibuat dalam *teachable machine* adalah model yang memiliki bit gambar dengan ukuran 224 x 224. Sehingga dalam mengatur bit dalam *android studi* harus disamakan dengan bit yang sesuai dengan model.



Gambar 45 Mengatur ukuran bit 2

Setelah mengatur ukuran bit yang sesuai dengan model, langkah selanjutnya yaitu melakukan *preprocessing*.

```
private fun preprocessImage(bitmap: Bitmap): ByteBuffer {
    val inputSize = 224
    val imageBuffer = ByteBuffer.allocateDirect( capacity: 1 * inputSize * inputSize * 3 * 4)
    imageBuffer.order(ByteOrder.nativeOrder())

    // Preprocess the image into the input buffer
    val size = inputSize * inputSize
    val intValues = IntArray(size)
    val scaledBitmap = Bitmap.createScaledBitmap(bitmap, inputSize, inputSize, filter: true)
    scaledBitmap.getPixels(
        intValues,
        offset: 0,
        scaledBitmap.width,
        x: 0,
        y: 0,
        scaledBitmap.width,
        scaledBitmap.height
    )
}
```

Gambar 46 *Preprocessing Image*

```
var pixel = 0
for (i in 0 until < inputSize) {
    for (j in 0 until < inputSize) {
        val value = intValues[pixel++]
        imageBuffer.putFloat( value: (value shr 16 and 0xFF) / 255.0f)
        imageBuffer.putFloat( value: (value shr 8 and 0xFF) / 255.0f)
        imageBuffer.putFloat( value: (value and 0xFF) / 255.0f)
    }
}

return imageBuffer
}
```

Gambar 47 *Preprocessing Image 2*

Setelah melakukan proses *preprocessing* selanjutnya model akan dilakukan klassifikasi sesuai dengan kelas yang telah dibuat sebelumnya. Adapun kelas-kelas yang dibuat sebelumnya yaitu kelas luka lecet, luka bakar, luka sayat, dan luka tusuk.

```
private fun classifiImage(bitmap: Bitmap?) {
    val model = ModelUnquant.newInstance(applicationContext)
    val image = preprocessImage(bitmap!!)

    val inputFeature0 = TensorBuffer.createFixedSize(intArrayOf(1, 224, 224, 3), DataType.FLOAT32)

    if (image != null) {
        inputFeature0.loadBuffer(image)
    }

    val outputs = model.process(inputFeature0)
    val outputFeature0 = outputs.outputFeature0AsTensorBuffer

    val confidence0 = outputFeature0.floatArray
    var maxConfidence0: Float = 0f
    var max0 = 0
}
```

Gambar 48 Klasifikasi data dengan label

```
val messageResult = arrayOf(
    "Tidak Terdeteksi",
    "Luka Lecet",
    "Luka Bakar",
    "Luka Tusuk",
    "Luka Sayat",
)

for(j in confidence0.indices) {
    if(confidence0[j] > maxConfidence0) {
        maxConfidence0 = confidence0[j]
        max0 = j
    }
}
```

Gambar 49 Klasifikasi data dengan label 2

```
if (max0 == 0) {
    showMessage( check: false, message: "Undetected Wound")
} else {
    var data = messageResult[max0]

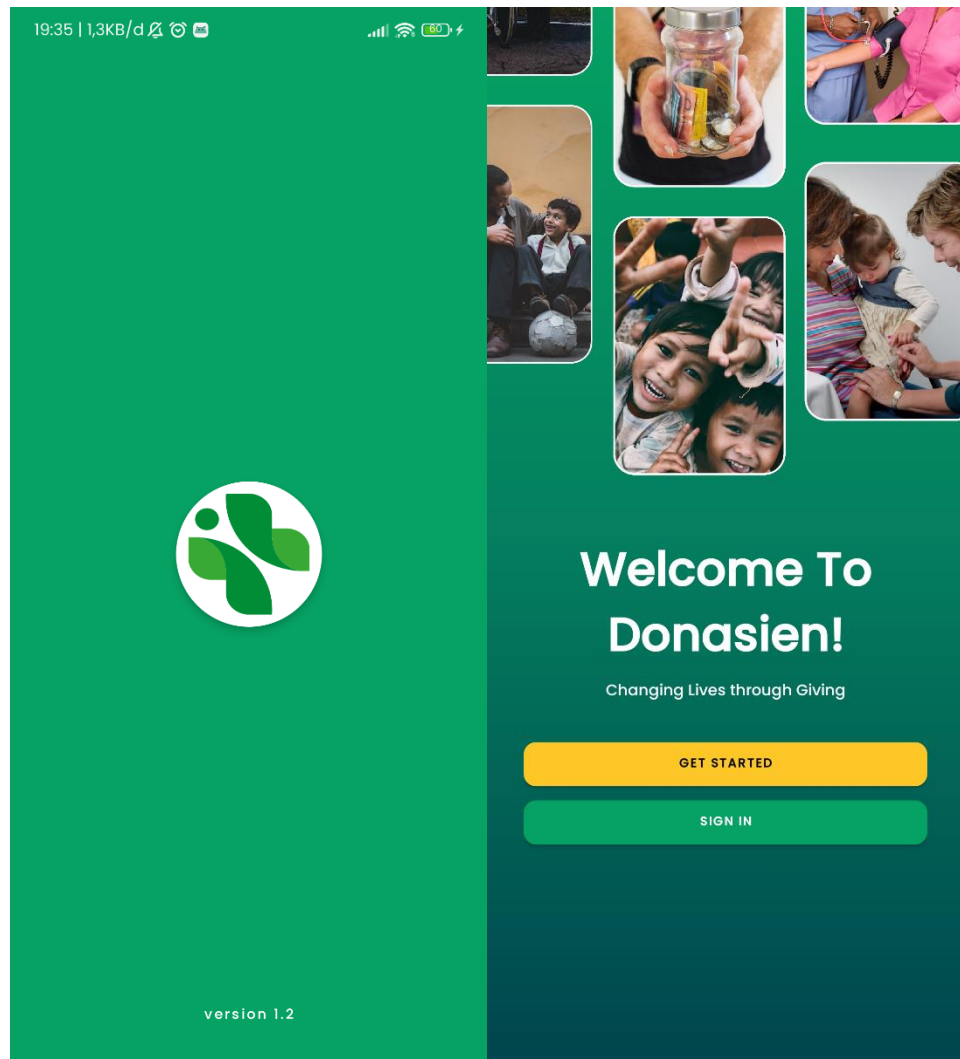
    Log.d( tag: "Hasil Output 0: ", data)

    showMessage( check: true, data)
}

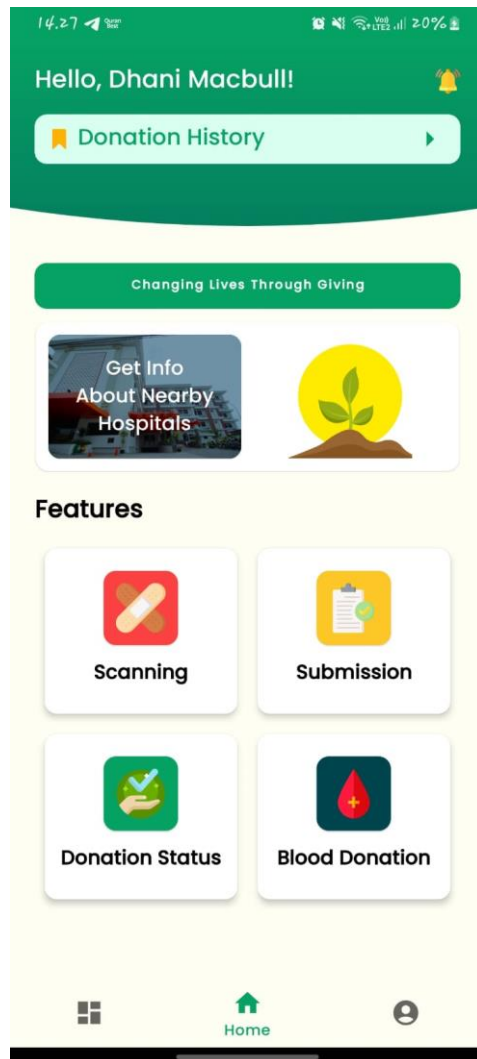
model.close()
}
```

Gambar 50 Klasifikasi data dengan label 3

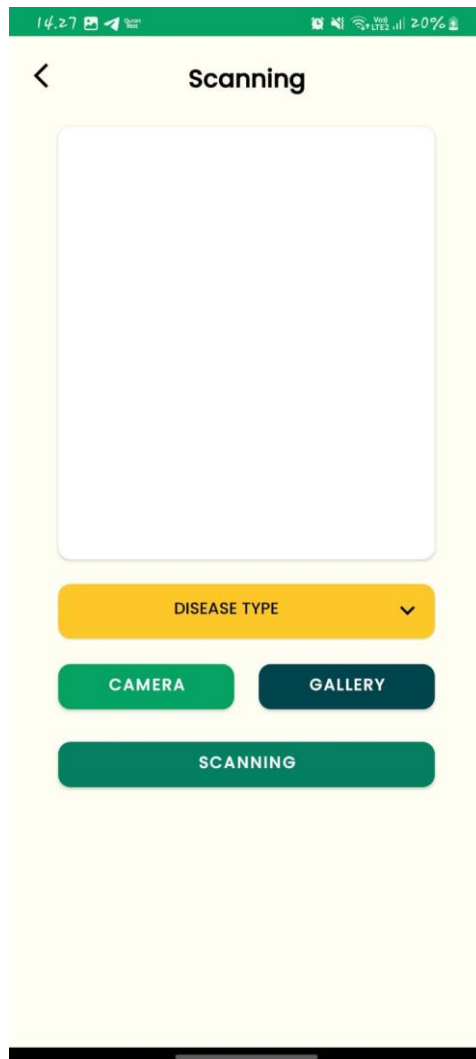
Setelah semua langkah sudah dilakukan maka, selanjutnya pengujian. Ketika pengujian maka akan tampil sebuah sistem berbasis *mobile* dari hasil penggabungan dan implemetantasi dari *interface mobile* dan model yang telah dibuat sebelumnya. Ini hasil dari pengujian di sistem berbasis *mobile*.



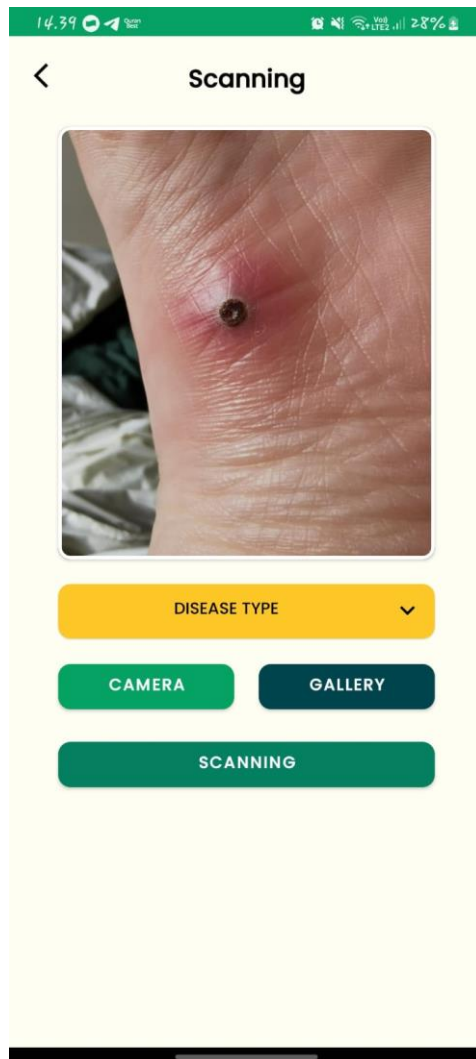
Gambar 51 Halaman awal sistem



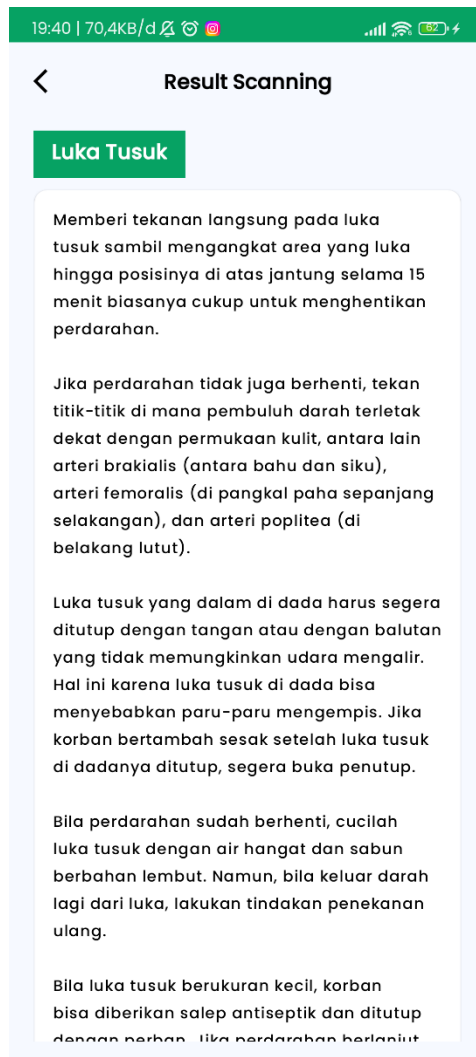
Gambar 52 Halaman utama sistem deteksi luka ringan



Gambar 53 Menu deteksi luka ringan



Gambar 54 Menampilkan data luka ringan



Gambar 55 Hasil Klasifikasi Luka



Gambar 56 Pertolongan pertama pada sistem

Dari Gambar diatas dapat dilihat bahwa model yang telah dibuat sebelumnya dapat diimplementasikan kedalam sistem berbasis *mobile*. Ketika pengujian dengan memasukan data gambar sistem memberikan hasil klasifikasi dari gambar dan memberikan keterangan luka sesuai dengan data dalam contoh diatas data yang dimasukan yaitu berupa data luka tusuk dan sistem memberikan hasil klasifikasi dengan keterangan luka tusuk sesuai dengan data gambar. Selain itu, sistem memberikan keterangan tambahan berupa beberapa cara penanganan pertama atau pertolongan pertama ketika mendapatkan luka tersebut.

Sistem memberikan resep atau obat untuk mengantisipasi terjadinya infeksi baik itu berupa obat alami atau obat kimia.

4.2 Implementasi dan Evaluasi Sistem

4.2.1 Implementasi dan Evaluasi *Object Detection*

Model yang telah dibuat sebelumnya maka akan dilakukan implementasi, pengujian dan evaluasi. Untuk metode pengujian dan evaluasi dilakukan dengan dua cara yaitu dengan menggunakan metode *cross data validation* dan *black box testing*.

- *Cross Data Validation Object Detection*

Pengujian dan evaluasi yang dilakukan pertama yaitu dengan menggunakan metode *cross data validation*. Pada proses pelatihan model terdapat metrik *accuracy*, *loss accuracy*, *validation accuracy* dan *loss validation*. Namun, ketika melakukan pembuatan model *object detection* hanya terdapat metrik *classification loss*, *localization loss*, *regularization loss* dan *total loss*. Untuk hasil pengujian dan evaluasinya dilakukan dengan 100 *steps* setiap *loss* nya.

No	Steps	<i>Classification loss</i>	<i>Localization loss</i>	<i>Regularization loss</i>	<i>Total loss</i>	<i>Learning rate</i>
1	0-100	0.41611898	0.45700082	0.15348144	1.0266013	0.0319994
2	100-200	0.2993229	0.3887098	0.15348673	0.8415195	0.0373328
3	200-300	0.2165613	0.28043512	0.15364765	0.65064406	0.0426662
4	300-400	0.29076695	0.2911204	0.15386361	0.735751	0.047999598
5	400-500	0.21583965	0.25836632	0.15396398	0.62816995	0.053333
6	500-600	0.16465117	0.24099588	0.15416464	0.5598117	0.0586664
7	600-700	0.17839341	0.19083102	0.15420072	0.52342516	0.0639998
8	700-800	0.13803355	0.20545453	0.15417352	0.49766162	0.069333196
9	800-900	0.14013825	0.16366297	0.15424931	0.45805055	0.074666604

10	900-1000	0.1469528	0.14519961	0.15408763	0.44624004	0.08
11	1000-1100	0.12351806	0.16977733	0.15393291	0.4472283	0.07999918
12	1100-1200	0.13408458	0.14691614	0.15354127	0.434542	0.079996705
13	1200-1300	0.13581981	0.12943803	0.15312132	0.41837916	0.0799926
14	1300-1400	0.1278437	0.14161147	0.1526877	0.42214286	0.07998685
15	1400-1500	0.121901065	0.13306145	0.15220599	0.4071685	0.07997945
16	1500-1600	0.11583414	0.10845146	0.15165794	0.37594354	0.079970405
17	1600-1700	0.106362015	0.09755136	0.15110904	0.35502243	0.07995972
18	1700-1800	0.112659805	0.11566903	0.1504976	0.37882644	0.0799474
19	1800-1900	0.13475288	0.110053666	0.14990568	0.39471224	0.07993342
20	1900-2000	0.11372569	0.12969072	0.1493115	0.3927279	0.07991781

Tabel 6 *Cross Data Validation Object Detection*

Pada tabel diatas setiap step mendapati *loss* yang selalu berkurang yang artinya model tersebut dapat berjalan dengan baik dan dapat digunakan untuk di implementasikan ke dalam aplikasi baik berbasis *mobile* ataupun yang lainnya. Selain itu, pada setiap langkahnya terdapat *learning rate* yang berbeda-beda yang artinya model akan mengevelauasi setiap langkahnya sehinga model dapat memberikan hasil yang terbaik.

- *Cross Validation Classification*

Pada metode pengujian *cross validation* klasifikasi dilakukan dua pengujian yaitu pengujian pada *accuracy* dan *validation accuracy*. Adapun dalam *accuracy* dan *validation accuracy* terdapat masing-masing *loss* yaitu *loss accuracy* dan *validation loss*. Pengujian akurasi ini dilakukan untuk melihat hasil dari pelatihan terhadap data yang telah dilatih apakah data tersebut mendapatkan hasil yang bagus atau hasil tersebut perlu diperbaiki lagi. Adapun hasil evaluasi dapat dilihat dalam tabel dibawah ini.

No	Epochs	Accuracy	Loss Accuracy	Validation Accuracy	Loss Validation
1	Epochs 1	0.3969	14.6983	0.4762	4.1325
2	Epochs 2	0.6756	1.5446	0.6349	1.7900
3	Epochs 3	0.7786	0.7864	0.6349	2.1628
4	Epochs 4	0.7863	0.7692	0.6349	1.9448
5	Epochs 5	0.7824	0.8077	0.6190	1.8417
6	Epochs 6	0.8092	0.6322	0.6032	2.9604
7	Epochs 7	0.8053	0.6737	0.6032	1.7535
8	Epochs 8	0.7977	0.7659	0.7302	1.4988
9	Epochs 9	0.8626	0.3789	0.5714	4.1675
10	Epochs 10	0.8206	0.5779	0.6349	1.4608
11	Epochs 11	0.8893	0.3360	0.6667	1.6343
12	Epochs 12	0.9008	0.3079	0.6984	1.4426
13	Epochs 13	0.8969	0.2738	0.6984	1.2774
14	Epochs 14	0.9122	0.2797	0.6984	1.5777
15	Epochs 15	0.9160	0.2834	0.6984	1.4810
16	Epochs 16	0.9160	0.3120	0.6667	1.6892
17	Epochs 17	0.9198	0.2284	0.7143	1.6925
18	Epochs 18	0.9351	0.2078	0.7778	1.1005
19	Epochs 19	0.9160	0.2186	0.6825	1.8170
20	Epochs 20	0.9389	0.1745	0.7619	1.1761

Tabel 7 Cross Data Validation Classification

Pada tabel diatas terlihat bahwa setiap stepsnya akurasi bertambah baik dan *loss* dalam akurasinya selalu berkurang artinya model melatih data dengan baik. Namun, ketika memvalidasi data model mendapatkan akurasinya yang kurang baik sehingga hal ini juga yang menyebabkan *loss* pada validasi akurasi menjadi besar. Hal ini pula yang dapat menyebabkan model mengalami *overfitting*. Hal ini harus dihindarkan karena akan menyebabkan model tidak dapat mengenal data baru yang tidak sesuai dengan data dari hasil model yang telah dilatih.

- Hasil Pengujian *Black Box testing* fitur *scan* luka ringan

No	Skenario Pengujian	Test Case	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
1	Klik menu <i>scanning</i> pada aplikasi donasien	Tombol menu <i>scanning</i>	Membuka menu <i>scanning</i> dan menampilkan	Sesuai harapan tombol	Valid

			komponen-komponen pada menu	berfungsi dengan baik	
2	Klik tombol kamera pada menu <i>scanning</i>	Tombol kamera	Dapat membuka kamera pada perangkat dan dapat mengambil gambar sebagai data yang akan di <i>scanning</i>	Sesuai harapan tombol berfungsi dengan baik dan gambar dapat diambil	Valid
3	Klik tombol <i>gallery</i> pada menu <i>scanning</i>	Tombol <i>Galery</i>	Dapat membuka galeri dalam perangkat dan dapat mengambil gambar dari perangkat	Sesuai harapan tombol berfungsi dengan baik dan data gambar dapat diambil	Valid
4	Klik tombol <i>scanning</i>	Tombol <i>Scanning</i>	Dapat memberikan hasil dari data gambar yang di input	Sesuai harapan dan memberikan hasil <i>scanning</i> sesuai data	Valid
5	Menampilkan gambar dari data yang telah dimasukan	Menu tampil gambar	Dapat menampilkan gambar sesuai data yang dimasukkan	Sesuai harapan dan menampilkan gambar pada menu gambar	Valid

Tabel 8 *Black Box Testing* Fitur deteksi luka ringan

Dari kedua hasil pengujian diatas yaitu pengujian *cross data validation* dan *black box testing* sistem dapat digunakan dengan semestinya. Namun, terdapat beberapa kesalahan dalam memprediksi hasil gambar. Hal ini dikarenakan kurangnya data yang digunakan ketika membuat model yang akan di *deploy* dalam perangkat berbasis *mobile*.

- Pengujian *User Acceptance Test* sistem deteksi luka ringan

No	Fitur	Deskripsi	Hasil	Keterangan
1	Halaman Beranda	Tampilan awal sistem deteksi luka ringan	Sesuai/OK	Menu-menu sudah tampil dan dapat berjalan dengan baik
2	Deteksi Luka Ringan	Mendeteksi gambar yang diambil dari kamera atau galeri kemudian memberikan hasil dari deteksi	Sesuai/OK	Fitur sudah dapat digunakan namun saat ini dikarenakan server dalam kondisi tidak aktif sehingga fitur ini tidak dapat memberikan hasil deteksi
3	Ambil Gambar dari galeri atau kamera	Mengambil gambar luka ringan	Sesuai/OK	Dapat menampilkan gambar melalui kamera atau dari galeri
4	<i>Login</i>	<i>Login</i> untuk user akses sistem deteksi luka ringa	Sesuai/OK	Memuat data user yang telah mendaftar pada sistem
5	<i>Register</i>	<i>Register</i> untuk membuat akun dalam sistem deteksi luka ringan	Sesuai/OK	Dapat membuat akun baru dengan menggunakan email dan memberikan passwordnya
6	Model klasifikasi luka ringan	Model <i>machine learning</i>	Kurang sesuai	Model yang telah dibuat mendapatkan akurasi yang baik namun untuk melakukan validasi akurasinya belum cukup

				baik sehingga hasil dari model tersebut dapat mengakibatkan hasil deteksi yang kurang sesuai hal ini dikarenakan minimnya data yang didapatkan
--	--	--	--	--

Tabel 9 User Acceptance Test

Dari 6 fitur yang dibuat maka semuanya dapat berjalan namun ada beberapa fitur yang mengalami kendala hal ini terjadi pada fitur deteksi luka ringan. Faktor yang menyebabkan fitur mengalami kendala yaitu server yang digunakan dalam kondisi nonaktif dikarenakan masa berlaku server yang belum diperpanjang. Maka dari 6 fitur itu berjalan semuanya dengan baik dan cukup sesuai (84%).

Dari kedua model yang telah dibuat maka dari model untuk deteksi objek luka ringan dengan jumlah data gambar sebanyak 200 gambar didapatkan hasil yang baik yaitu mendapatkan akurasi di rentang 50% - 95% sehingga model untuk deteksi objek luka ringan cukup baik untuk digunakan. Namun, untuk model deteksi objek belum dapat di implementasikan kedalam sistem berbasis *mobile*. Sedangkan, untuk model klasifikasi mendapatkan akurasi yang kurang baik karena model mengalami *overfitting* hal ini disebabkan karena minimnya data gambar luka ringan yang diperoleh sehingga metrik akurasi yang digunakan dalam model ini untuk data latih dan data validasi memiliki perbedaan akurasi yang cukup jauh. Untuk data latih mendapatkan akurasi sekitar 94% dan untuk data validasi mendapatkan akurasi 76% dari 325 data gambar luka ringan yang digunakan. Model klasifikasi yang telah dibuat dan di implementasikan kedalam sistem berbasis *mobile* dapat berjalan dan memberikan hasil yang cukup baik. Namun, terkadang hasil prediksi tidak selalu tepat dengan gambar yang di *input*.

BAB V

KESIMPULAN DAN SARAN

Pada bab ini sebagai penutup akan dijelaskan kesimpulan dan saran atas hasil yang telah dilakukan pada tahap sebelumnya yaitu implementasi dan perancangan. Kesimpulan adalah jawaban dari hasil rumusan masalah yang dijelaskan diawal. Sedangkan, saran adalah masukan yang diberikan untuk tahap pengembangan selanjutnya untuk sistem deteksi ini.

5.1 Kesimpulan

Berdasarkan hasil perancangan dan evaluasi pada penelitian kali ini, tujuan penelitian dapat tercapai serta dapat menjawab rumusan masalah yang telah dijelaskan pada bab I. Berikut ini adalah kesimpulannya :

- a) Alur perancangan sistem deteksi luka ringan ini menggunakan *Convolutional Neural Network* (CNN) sebagai metode dalam membuat model *machine learning*. *Convolutinal Neural Network* digunakan untuk membuat model sebagai *object detection* dan *classification*. Dalam penggunaan CNN ini digunakan dua cara yaitu CNN yang menggunakan *transfer learning* yaitu CNN yang sudah dibuat sebelumnya dan dapat langsung dipakai ketika membuat model dan CNN yang mennggunakan sesuai dengan kebutuhan dalam membuat model atau CNN murni tanpa *transfer learning*. CNN juga digunakan sebagai layer untuk citra, CNN akan memanfaatkan proses konvolusi dengan menggerakan sebuah *kernel* atau filter berukuran tertentu ke sebuah citra atau gambar dan komputer akan mendapatkan informasi representative dari hasil perkalian gambar tersebut dengan menggunakan kernel atau filter tersebut. Hasil dari pembuatan model tersebut untuk deteksi objek mendapatkan akurasi sekitar 50% - 95% dari hasil prediksi dan untuk hasil dari model klasifikasi mendapatkan tingkat akurasi untuk data latih yaitu 96% dan untuk data validasi mendapatkan akurasi 74% sehingga ketika dilakukan testing data maka data yang diprediksi terkadang tepat dan terkadang tidak sesuai dengan gambar yang dimasukkan.

- b) Dari model yang telah dibuat kemudian model di *deploy* kedalam perangkat berbasis *mobile*. Hal ini dilakukan agar model yang telah dibuat dapat digunakan untuk para pengguna. Pengguna dapat menggunakan sistem deteksi luka ringan sebagai alat untuk mengidentifikasi luka ringan dengan menggunakan perangkat *mobile* dari masing-masing pengguna.
- c) Dari hasil deteksi luka ringan yang dilakukan oleh pengguna, sistem akan memberikan beberapa pertolongan pertama untuk mencegah pengguna dari infeksi luka ringan. Tidak hanya itu sistem akan memberikan daftar obat baik obat dari tanaman atau obat medis yang dapat digunakan untuk mengobati dan mencegah luka tersebut dari infeksi.

5.2 Saran

Saran merupakan masukan untuk pengembangan selanjutnya agar sistem yang telah dibuat dapat berkembang lebih baik lagi. Berikut beberapa saran yang diberikan agar sistem dapat berkembang lebih baik lagi:

- a) Dalam sistem ini masih terdapat prediksi luka ringan yang tidak sesuai dengan data yang di input hal ini terjadi karena kurangnya data yang diolah sehingga kedepannya sistem ini dapat memberikan hasil yang lebih akurat dengan menambah banyak data luka ringan dan data yang lebih baik untuk diolah.
- b) Sistem deteksi luka ringan ini berjalan bersamaan dengan sistem donasi pasien. Kedepannya kedua sistem dapat dipisahkan antara sistem deteksi luka ringan dan sistem donasi. Sehingga, pengguna dapat menggunakannya lebih nyaman dan lebih mudah.
- c) Sistem deteksi luka ringan saat ini hanya dapat memprediksi luka ringan yang dialami oleh pengguna. Namun, pengguna masih kebingungan jika luka yang dialami sudah mengalami infeksi sehingga pengguna menjadi takut terhadap luka tersebut. Kedepannya sistem deteksi luka ringan ini dapat memberikan fitur chat kepada dokter spesialis luka jika pengguna mengalami infeksi dari luka tersebut.

DAFTAR REFERENSI

- [1] R. Wintoko and A. Dwi Nur Yadika, “2893-3593-1-PB”.
- [2] Anugerah Ayu Sendari, “Pengertian Sistem Menurut Para Ahli, Karakteristik dan Macamnya,” ,14 Februari 2021 (Online),(Diakses 04 November 2023).
- [3] J. Finlay and A. Dix, “An Introduction to Artificial Intelligence.” doi: 10.4324/9781003072485.
- [4] Q. Bi, K. E. Goodman, J. Kaminsky, and J. Lessler, “What is machine learning? A primer for the epidemiologist,” *Am J Epidemiol*, vol. 188, no. 12, pp. 2222–2239, Dec. 2019, doi: 10.1093/aje/kwz189.
- [5] J. S. Suroso, “JENIS JENIS MACHINE LEARNING (Jarot Suroso),” ,27 September 2022 (Online), (Diakses 04 November 2023).
- [6] J. D. Kelleher, *Deep Learning*, 2019th ed., vol. The MIT Press,[2019]. Cambridge, Massachusetts: 2019, 1974.
- [7] R. Jagtap, “How an Algorithm Was Made to Think Like the Brain,” ,01 April 2021 (Online), (Diakses 04 November 2023).
- [8] A. Innovation, “Deep Learning applied to Computer Vision,” , 24 Februari 2022 (Online), (Diakses 05 November 2023).
- [9] J. Gifari, “Apa yang dimaksud dengan Tensorflow dan Bagaimana Penggunaannya?,” , 17 November 2020 (Online), (Diakses 05 November 2023).
- [10] P. Weichbroth, “Usability of mobile applications: A systematic literature study,” *IEEE Access*, vol. 8, pp. 55563–55577, 2020, doi: 10.1109/ACCESS.2020.2981892.
- [11] A. Rilo Pambudi, “JIP (Jurnal Informatika Polinema) DETEKSI KEASLIAN UANG KERTAS BERDASARKAN WATERMARK DENGAN PENGOLAHAN CITRA DIGITAL”.
- [12] O. Alsing, “Mobile Object Detection using TensorFlow Lite and Transfer Learning,” 2018.

- [13] J. Brownlee, "A Gentle Introduction to Transfer Learning for Deep Learning," , September 16, 2019 (online), (Diakses 12 November 2023).
- [14] P. Baheti, "A Newbie-Friendly Guide to Transfer Learning," , 12 Oktober 2021 (online), Diakses (12 November 2023).
- [15] S. Yuill and H. Halpin, "Python," 2006.
- [16] C. Xie *et al.*, "Improving Transferability of Adversarial Examples with Input Diversity," Mar. 2018.
- [17] M. Rizqi Efrian *et al.*, "IMAGE RECOGNITION BERBASIS CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK MENDETEKSI PENYAKIT KULIT PADA MANUSIA," *Jurnal POLEKTRO: Jurnal Power Elektronik*, vol. 11, no. 1, p. 2022.
- [18] R. Namruddin, "Klasifikasi Kesegaran Buah Apel Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Android."
- [19] M. A. Abu, A. Halim, A. Rahman, I. Ahmad, N. Hazirah Indra, and A. Sapiee, "A study on Image Classification based on Deep Learning and Tensorflow," 2019. [Online]. Available: <http://www.irphouse.com>563
- [20] Y. Novaria Kunang, "Pengembangan Aplikasi Pengenalan Aksara Komerling Menggunakan Metode Deep Learning Berbasis Android," 2020. [Online]. Available: <https://journal-computing.org/index.php/journal-cisa/index>