

**KLASIFIKASI PENYAKIT MATA PADA MANUSIA DENGAN
MENGUNAKAN MODEL ARSITEKTUR *VISION*
*TRANSFORMERS***

SKRIPSI

**SENTIA OVANIA PURBA
198160066**



**PROGRAM STUDI SARJANA TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MEDAN AREA
MEDAN
2024**

UNIVERSITAS MEDAN AREA

© Hak Cipta Di Lindungi Undang-Undang

1. Dilarang Mengutip sebagian atau seluruh dokumen ini tanpa mencantumkan sumber
2. Pengutipan hanya untuk keperluan pendidikan, penelitian dan penulisan karya ilmiah
3. Dilarang memperbanyak sebagian atau seluruh karya ini dalam bentuk apapun tanpa izin Universitas Medan Area

Document Accepted 11/7/24

Access From (repository.uma.ac.id)11/7/24

**KLASIFIKASI PENYAKIT MATA PADA MANUSIA
DENGAN MENGGUNAKAN MODEL ARSITEKTUR
VISION TRANSFORMERS**

SKRIPSI

Diajukan Untuk Memenuhi Sebagian Persyaratan Dalam Memperoleh Gelar
Sarjana Teknik Informatika Universitas Medan Area



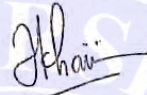
**Oleh:
SENTIA OVANIA PURBA
198160066**

**PROGRAM STUDI SARJANA TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MEDAN AREA
MEDAN
2024**

HALAMAN PENGESAHAN

Judul Skripsi : Klasifikasi Penyakit Mata Pada Manusia dengan menggunakan Model Arsitektur *Vision Transformers*
Nama : Sentia Ovania Purba
Npm : 198160066
Fakultas : Teknik
Prodi : Teknik Informatika

Disetujui Oleh
Komisi Pembimbing



Nurul Khairina, S.Kom, M.Kom
Pembimbing



Drs. Puji Astuti, S.T, M.T.
Dekan Fakultas Teknik



Dr. Rizki Nurul Hafidha, S.Kom, M.Kom
Ka. Prodi

Tanggal Lulus: 27 Maret 2024

HALAMAN PERNYATAAN

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

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

Medan, 27 Maret 2024



Sentia Ovania Purba
198160066



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

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

Nama : Sentia Ovania Purba
NPM : 198160066
Program Studi : Teknik Informatika
Fakultas : Teknik
Jenis Karya : Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Medan Area **Hak Bebas Royalti Noneklusif (Non-exclusive Royalty-Free Right)** atas karya ilmiah saya yang berjudul :

*Klasifikasi Penyakit Mata Pada Manusia Dengan Menggunakan Model
Arsitektur Vision Transformers*

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

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Medan

Pada Tanggal : 27 Maret 2024

Yang Menandatangani


Sentia Ovania Purba
1981600666

RIWAYAT HIDUP

Penulis dilahirkan di Desa Wonosari, pada tanggal 27 Oktober 1999 dari ayah Epraim Purba dan ibu Meryati Manurung. Penulis merupakan anak ketiga dan putri pertama dari enam bersaudara. Penulis menyelesaikan Pendidikan sekolah dasar di SD Negeri 101885 Kiri Hilir pada tahun 2012. Pada tahun yang sama penulis melanjutkan Pendidikan sekolah menengah pertama di SMPN 3 Tanjung Morawa selama 3(tiga) tahun dan selesai pada tahun 2015. Penulis melanjutkan Pendidikan selanjutnya yaitu sekolah menengah atas di SMKN 1 Lubuk Pakam selama 3 (tiga) tahun dan lulus pada tahun 2018. Setelah tamat sekolah SMK penulis tidak melanjutkan Pendidikan atau bekerja terlebih dahulu sebagai SPG Handbody Lovely. Kemudian tahun 2019 penulis melanjutkan Pendidikannya di Perguruan Tinggi Swasta dan mendaftar di salah satu Universitas yaitu Universitas Medan Area dengan jurusan Teknik informatika. Selama mengikuti perkuliahan, penulis mengikuti dan melaksanakan Kerja Praktek (KP) di PDAM Tirtanadi di Jalan Sisingamaraja No. 01 Medan 20122 Provinsi Sumatera Utara.

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Kuasa atas segala karuniaNya sehingga skripsi ini berhasil diselesaikan. Tema yang dipilih dalam penelitian ini dengan judul “Analisis Arsitektur *Deep Learning ViT (Vision Transformers)* Untuk Klasifikasi Penyakit Katarak Pada Manusia”.

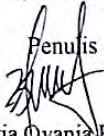
Skripsi ini merupakan salah satu syarat untuk menyelesaikan pendidikan untuk mencapai gelar sarjana di Program Studi Teknik Informatika Fakultas Teknik Universitas Medan Area. Pada kesempatan ini penulis mengucapkan terima kasih kepada :

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

Penulis menyadari bahwa penelitian ini masih memiliki kekurangan, oleh karena itu kritik dan saran yang bersifat membangun sangat penulis harapkan demi kesempurnaan penelitian ini. Penulis berharap tugas penelitian ini dapat

baik kalangan pendidikan maupun masyarakat. Akhir kata penulis ucapkan terima kasih.

Medan, 27 Maret 2024

Penulis

Sentia Ovania Purba



ABSTRAK

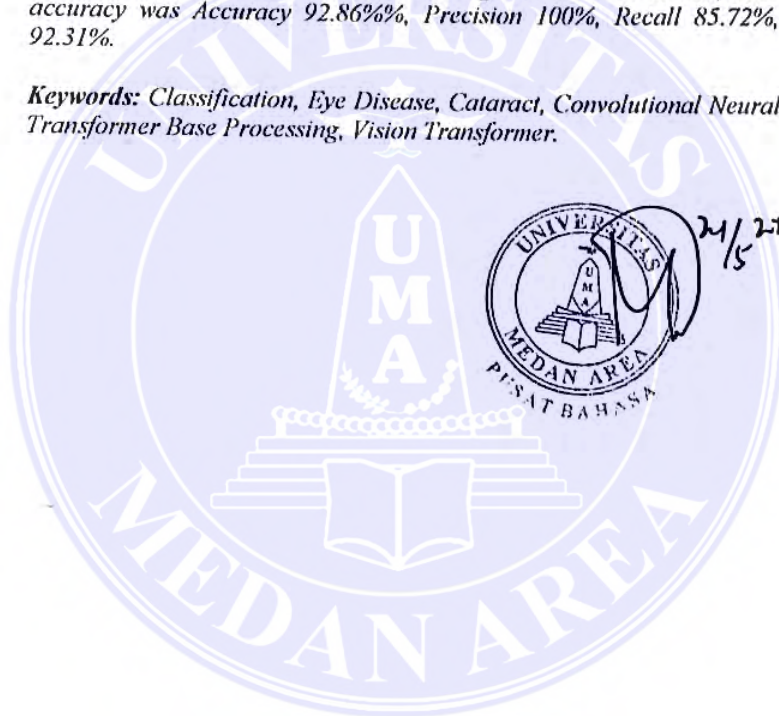
Mata merupakan alat indra makhluk hidup yang sangat penting khususnya pada manusia. Dimana jika retina mata mengalami kerusakan atau terkena penyakit, khususnya penyakit mata katarak maka kualitas retina mata tidak dapat digunakan dengan baik. Maka diperlukan pendekatan digital agar dapat mengenali mata tersebut positif katarak atau negatif secara cepat, tepat, efisien dan akurat. Sehingga penelitian ini membuat suatu investigasi penyakit mata manusia khususnya katarak melalui citra retina mata kedalam bentuk klasifikasi dengan cepat, efisien, dan akurat. Klasifikasi yang dilakukan dalam penelitian ini menggunakan model algoritma *Vision Transformer* dari CNN. Berdasarkan hasil *training* pada enam skenario yang diuji. dengan menggunakan dataset *training* memanfaatkan *Hyperparameter Epoch* 25 dan 50, *Input Shape* 224x224x3 (RGB channel), *batch size* 32 dan *Optimizer* (Adam, RMSprop, SGD) dan *Learning Rate* 0.001 dengan dataset 1120 citra rentia. Hasil dari penelitian ini dengan menggunakan *Vision Transformer* yang memanfaatkan *Hyperparameter Epoch* 25 dan 50, *Input Shape* 224x224x3 (RGB channel), *batch size* 32 dan *Optimizer* (Adam, RMSprop, SGD) dan *Learning Rate* 0.001 mendapatkan hasil secara berturut-turut, dimana akurasi terbaik yaitu *Akurasi* 92.86%%, *Precision* 100%, *Recall* 85.72%, *F1-score* 92,31%.

Kata Kunci: *Klasifikasi, Penyakit Mata, Katarak, Convolutional Neural Network, Pengolah Dasar Transformer, Vision Transformer.*

ABSTRACT

The eye is a very important sensory tool of living things, especially in humans. Where if the eye retina is damaged or affected by disease, especially cataract eye disease, the quality of the eye retina cannot be used properly. So a digital approach is needed in order to recognise the eye is positive or negative cataract quickly, precisely, efficiently and accurately. So this research made an investigation of human eye disease, especially cataracts through eye retina images into the form of classification quickly, efficiently, and accurately. The classification carried out in this study used the Vision Transformer algorithm model from CNN. Based on the results of training on six scenarios tested. Using the training dataset utilising Hyperparameter Epoch 25 and 50, Input Shape 224x224x3 (RGB channel), batch size 32 and Optimizer (Adam, RMSprop, SGD) and Learning Rate 0.001 with a dataset of 1120 retina images. The results of this study using Vision Transformer that utilized Hyperparameter Epoch 25 and 50, Input Shape 224x224x3 (RGB channel), batch size 32 and Optimizer (Adam, RMSprop, SGD) and Learning Rate 0.001 got consecutive results, where the best accuracy was Accuracy 92.86%, Precision 100%, Recall 85.72%, F1-score 92.31%.

Keywords: Classification, Eye Disease, Cataract, Convolutional Neural Network, Transformer Base Processing, Vision Transformer.



DAFTAR ISI

HALAMAN PENGESAHAN	ii
HALAMAN PERNYATAAN	iii
RIWAYAT HIDUP	v
KATA PENGANTAR	vi
ABSTRAK	viii
ABSTRACT	ix
DAFTAR ISI	x
DAFTAR GAMBAR	xii
DAFTAR TABEL	xiv
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	7
1.3 Batasan Masalah	7
1.4 Tujuan Penelitian	8
1.5 Manfaat Penelitian	8
1.6 Sistematika Penulisan	9
BAB II TINJAUAN PUSTAKA	12
2.1. Klasifikasi	12
2.2. Mata	12
2.3. Penyakit Mata	12
2.4 Katarak.....	13
2.5 Slit Lamp	13
2.6 <i>Deep Learning</i>	14
2.7 <i>Convolutional Neural Network (CNN)</i>	15
2.8 <i>Vision Transformer</i>	17
2.8.1 Komponen Utama Arsitektur <i>Vision Transformers (ViT)</i>	18
2.8.2 Model ViT Pre-Trained	18
2.9 Tahap Kinerja <i>Vision Transformer</i>	19
2.10 Confusion Matrix dan Classification Report	23
2.11 Penelitian Terdahulu	25
BAB III METODE PENELITIAN	28
3.1 Tahapan Penelitian.....	28
3.2 Teknik Penumpulan Data.....	29

3.3	Inisialisasi Hyperparameter.....	30
3.4	Alur Proses Klasifikasi	31
3.4	Simulasi Arsitektur <i>Vision Transformers</i>	33
BAB IV HASIL DAN PEMBAHASAN.....		50
4.1	Hasil dan Pembahasan	50
4.1.1	Pengumpulan Dataset	50
4.1.2	Pembagian Dataset	51
4.1.3	<i>Preprocessing</i> Dataset.....	52
4.1.4	Pemodelan Arsitektur Vision Transformer.....	58
4.1.5	Hasil Training Model	62
4.2	Hasil Evaluasi Model.....	73
BAB V KESIMPULAN DAN SARAN		83
5.1	Kesimpulan	83
5.2	Saran	83
DAFTAR PUSTAKA		85
LAMPIRAN.....		89

DAFTAR GAMBAR

Gambar 2. 1 Slit Lamp.....	14
Gambar 2. 2 Ilustrasi Machine Learning dan Deep Learning.....	15
Gambar 2. 3 Convolution Layer	16
Gambar 2. 4 Ilustrasi Proses Kinerja Vision Transformer	20
Gambar 2. 5 Confusion Matrix dan Rumus Classification Report.	24
Gambar 3. 1 Kerangka Kinerja Penelitian	28
Gambar 3. 2 Jumlah Dataset Yang Digunakan	30
Gambar 3. 3 Flowchart Kinerja Sistem Klasifikasi	32
Gambar 3. 4 Rancangan Simulasi Arsitektur Vision Transformer.....	34
Gambar 3. 5 Hasil Augmentasi Dataset	35
Gambar 3. 6 Patches Gambar Mata Katarak	38
Gambar 3. 7 Proses Linear Projection of flattened Patches.....	39
Gambar 3. 8 Simulasi classification Vision Transformer	49
Gambar 4. 1 Sampel Dataset Mata Katarak	50
Gambar 4. 2 Sampel Dataset Mata Normal.....	51
Gambar 4. 3 Sampel Dataset Kualitas Rendah	53
Gambar 4. 4 Proses Cropping Sampel Dataset	53
Gambar 4. 5 Proses Resize Sampel Dataset	54
Gambar 4. 6 Hasil Pelabelan Dataset Mata Katarak.....	55
Gambar 4. 7 Hasil Pelabelan Dataset Mata Normal.....	55
Gambar 4. 8 Split Dataset (Training, Validation, Testing)	56
Gambar 4. 9 Sampel Hasil Augmentasi Label Katarak	57
Gambar 4. 10 Sampel Hasil Augmentasi Label Normal.....	58
Gambar 4. 11 Plot Arsitektur Model.....	59
Gambar 4. 12 Hasil Pengujian Klasifikasi Pada Data Testing.....	62
Gambar 4. 13 Proses Fitting Model Skenario Ke-1	63
Gambar 4. 14 Plot Accuracy dan Loss Model Skenario Ke-1.....	64
Gambar 4. 15 Proses Fitting Model Skenario Ke-2	65
Gambar 4. 16 Plot Accuracy dan Loss Model Skenario Ke-2.....	66
Gambar 4. 17 Proses Fitting Model Skenario Ke-3	67
Gambar 4. 18 Plot Accuracy dan Loss Model Skenario Ke-3.....	67

Gambar 4. 19 Proses Fitting Model Skenario Ke-4	68
Gambar 4. 20 <i>Plot Accuracy</i> dan <i>Loss</i> Model Skenario Ke-4.....	69
Gambar 4. 21 Proses Fitting Model Skenario Ke-5	70
Gambar 4. 22 <i>Plot Accuracy</i> dan <i>Loss</i> Model Skenario Ke-5.....	71
Gambar 4. 23 Proses Fitting Model Skenario Ke-6	72
Gambar 4. 24 <i>Plot Accuracy</i> dan <i>Loss</i> Model Skenario Ke-6.....	73
Gambar 4. 25 Confusion Matrix dan Classification Report Model Skenario Ke-1.....	75
Gambar 4. 26 Confusion Matrix dan Classification Report Model Skenario Ke-2.....	77
Gambar 4. 27 Confusion Matrix dan Classification Report Model Skenario Ke-3.....	78
Gambar 4. 28 Confusion Matrix dan Classification Report Model Skenario Ke-4.....	78
Gambar 4. 29 Confusion Matrix dan Classification Report Model Skenario Ke-5.....	79
Gambar 4. 30 Confusion Matrix dan Classification Report Model Skenario Ke-6.....	80
Gambar 4. 31 Perbandingan Performa Matrix.....	81

DAFTAR TABEL

Tabel 2. 1 Penelitian Terdahulu	25
Tabel 3. 1 Insialisasi Hypreparameter	29
Tabel 4. 1 Pembagian Dataset	51
Tabel 4. 2 Skenario Training Model.....	61
Tabel 4. 3 Perbandingan Akurasi dari setiap Model Skenario	74
Tabel 4. 4 Hasil Confusion Matrix dan Classification Report	81
Tabel 4. 5 Perbandingan Akurasi Model dengan Penelitian Terdahulu.....	82



BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada indra penglihat yang sering disebut dengan mata adalah salah satu organ yang penting bagi manusia yang fungsinya sebagai sensor cahaya dan suatu sistem optik kompleks yang berfungsi untuk mengumpulkan cahaya dari lingkungan sekitar. Cahaya masuk melalui lensa yang berjalan ke retina dan diubah menjadi sinyal saraf untuk mendapatkan foto yang tajam dan harus transparan. Lensa mata tersusun dari protein lensa yang utamanya adalah protein larut air yang disebut dengan bentuk kapsul elastis. Kapsul elastis yang terdiri dari beberapa jaringan yang sejajar gradien indeksinya secara tidak seragam. Kapsul *elastis* dapat menyebabkan daya akomodasi pada organ lensa mata. Daya akomodasi ialah kemampuan mata untuk dapat memfokuskan penglihatan dengan jarak dekat dan jauh. Hal tersebut mengakibatkan terjadinya reaksi radikal bebas dengan *lipid membrane* sel lensa pada protein yang akan menyebabkan *cross-linking lipid* dan *protein* mengalami peningkatan tidak larut air (*water insoluble protein*) sehingga kejernihan pada lensa mata menurun dan mengakibatkan penyakit pada lensa mata atau sering disebut dengan Katarak. (Wiguna, Fardela, & Selly, 2019).

Katarak adalah penurunan *progresif* kejernihan pada lensa mata yang menjadi salah satu kondisi faktor tertentu yaitu faktor usia, aktivitas dan orang yang memiliki penyakit keturunan yang menjadi indikasi atau kondisi medis seperti diabetes, dihidrasi akut, gangguan atopik, hipertensi, dan asam urat.

Masyarakat yang mengalami gangguan penglihat pada mata sekitar lebih kurang 3,38% penduduk di dunia atau 253 juta orang menderita gangguan penglihat diantaranya 36 juta orang menderita kebutaan dan 217 juta orang mengalami gangguan penglihat sedang hingga berat. Penyebab gangguan penglihat tersebut adalah karena adanya kelainan refraksi yang tidak diketahui sebesar 48,99%, diikuti dengan katarak sebesar 25,81% dan dengan generasi yang terkait usia sebesar 4,1%. Penyebab kebutaan karena mata katarak sebesar 34,47%, penyebab yang tidak diketahui alasannya yaitu sebesar 20,26%, dan penyebab kebutaan dikarenakan glaucoma sebesar 8,30% (Simanjuntak, Fu'adah, Magdalena, Saidah, & Wiratama, 2022). Tingkat presentase sedikit 50% diseluruh dunia dengan hasil survey kebutaan di 15 provinsi pada tahun 2014-2016 menunjukkan 70-80% (Bu'ulolo, Jacobus, & Kambey, 2021). Banyak masyarakat umum sering tidak menyadari adanya katarak yang menyelimuti kornea mata pada manusia karena masih sulit membedakan mata normal dari mata katarak, *retina deses*, *glukoma* di lingkungan sekitar. Mengingat uraian sebelumnya, sangat penting untuk mengidentifikasi dan mengklasifikasi katarak sebelum kebutaan terjadi. Oleh karena itu, diperlukan sistem yang dapat membedakan antara mata normal dengan mata katarak (Firdaus, Imran, Bakti, & Suryadi, 2022).

Di area medis masih menggunakan system manual untuk memberikan hasil yang akurat kepada penderita penyakit katarak. Hal ini membuat hasil tidak akurat sehingga membuat seorang pasien penderita katarak tidak mengetahui hasilnya secara efisien dan akurat.

Dizaman perkembangan teknologi pengolahan citra digital merupakan solusi terpedeapan dalam melakukan pengklasifikasian penyakit katarak,

memberikan hasil yang akurat terhadap penderita secara efisien dan maksimal. Oleh karena itu sistem kecerdasan buatan dalam proses identifikasi yang dapat membantu menduplikasi kemampuan manusia dalam memahami suatu informasi seperti menentukan seseorang terkena penyakit katarak atau tidak terkena penyakit katarak. Dengan proses image processing citra digital kita dapat membantu dalam melakukan pengenalan karakteristik atau ciri tertentu yang menjadi suatu informasi dari objek yang membuat suatu citra dapat dibedakan, dikelompokkan dan dikenali melalui parameter dalam penyakit katarak. Proses klasifikasi suatu citra yang diaplikasikan menggunakan *deep learning* yang merupakan bidang dari *machine learning*. (Bu'ulolo, Jacobus, & Kambey, 2021)

Deep Learning adalah sub bidang dari *Machine Learning* yang menggunakan jaringan saraf tiruan. Model *Deep Learning* telah banyak dikembangkan dan digunakan dalam melakukan pembelajaran *Big Data* melalui *pre-processing*, ekstraksi fitur yang menjadi salah satu model dari *Convolutional Neural Network (CNN)*. Model ini adalah pengembangan dari *Multilayer Perceptron (MLP)* yang didesain untuk mengolah data dua dimensi dan memiliki jaringan yang dalam mengaplikasikan pada data citra. *CNN* memiliki beberapa lapisan yang mengubah input menjadi operasi konvolusi secara otomatis.

Lapisan yang dimiliki *CNN* dan operasi konvolusi yang dilakukan dalam proses pembelajaran *CNN* membutuhkan banyak parameter alam proses pelatihan yang dilakukan. Parameter yang digunakan *CNN* memiliki ratusan hingga jutaan parameter yang setiap masing-masingnya belajar mendekteksi berbagai gambar. Hal ini memungkinkan sebuah ruang desain untuk sistem memori akselator *CNN* yang besar untuk mencangkup semua kombinasi, hirarki memori aktivasi, memori

bobot, dan memori umum. Kelemahan CNN yaitu proses pelatihan model yang cukup lama dan banyak data yang diperlukan untuk melakukan pelatihan model dari CNN. Hal ini terjadi dikarenakan karena keterbatasan dari CNN yang salah satunya Teknik *patching*. Teknik *Patching* harus dilakukan secara tepat agar kebutuhan CNN terpenuhi.

Teknik *Patching* dalam mengklasifikasi penyakit katarak pada mata telah banyak diteliti dengan berbagai model metode arsitektur dari *CNN* yaitu *GoogLeNet*, *ResNet*, *VGG-19*, *SVM*, *Inception*, *Novel Angular Binary Pattern (NABP)*, *Discrete Wavelet Transform (DWT)*, *GCLM* dan sebagainya. Ada beberapa penelitian diantaranya yaitu klasifikasi katarak dengan metode *VGG-19* diperoleh hasil akurasi keseluruhan sebesar 97,47%, penelitian klasifikasi katarak dengan hasil akurasi sebesar 93,3% dengan menggunakan metode *GLCM*, klasifikasi katarak dengan hasil akurasi yang diperoleh sebesar 80%, dan waktu komputasi sebesar 6,89s dengan menggunakan metode *DWT* jenis Hear (Gifran , Magdalena, & Faudah, 2019).

Pada penelitian selanjutnya dilakukan penelitian klasifikasi katarak dengan menggunakan metode *KNN* yang memiliki tingkat akurasi sebesar 83,07%, untuk klasifikasi katarak dengan menggunakan metode *SVM* diperoleh 76,64% (Simanjuntak, Fu'adah, Magdalena, Saidah , & Wiratama, 2022).. Setelah itu pada tahun 2020 melakukan penelitian klasifikasi katarak menggunakan arsitektur *AlexNet* pada data mata normal dan mata katarak dengan nilai akurasi tertinggi 97,50%. Selanjutnya adanya penelitian pada tahun 2021 dengan menggunakan arsitektur *GoogLeNet* dan membagi data menjadi dua kelas yaitu mata normal dan

mata katarak dengan nilai akurasi sebesar 88%, Namun untuk saat ini tidak ada yang menggunakan evaluasi atau penelitian lain yang menggunakan metode kinerja yang lain (Simanjuntak, Fu'adah, Magdalena, Saidah , & Wiratama, 2022).

Berdasarkan pada beberapa penelitian yang digunakan dan telah dipaparkan, dapat dilihat bahwa CNN tidak memiliki Teknik *patching* secara khusus yang digunakan pada keseluruhan model CNN. *Patching* pada CNN digunakan untuk menggunakan penambahan pada ketersediaan data yang terbatas untuk memenuhi data CNN yang membutuhkan banyak suatu data. Penambahan dalam proses pembelajaran *deep learning* Teknik *patching* juga menambahkan waktu untuk melakukan mesin pada *pre-processing* sebelum diterapkannya CNN. Berbeda dengan *ViT (Vision Transformer)* yang merupakan bagian dari *Deep Learning* yang sudah memiliki Teknik *patching* pada saat membangun suatu model didalam arsitektur tersebut. (Dwi Putri, 2022)

Vision Transformer (ViT) merupakan model untuk melakukan klasifikasi gambar yang menggunakan *attention* serta menghilangkan perulangan serta operasi konvolusi yang dilakukan diatas *patching* pada sebuah gambar. *ViT* untuk pertama kalinya diperkenalkan pada tahun 2017 yang menggunakan arsitektur *ViT* dalam tugas penerjemah bahasa Inggris ke Jerman dan bahasa Inggris ke Prancis dan telah berhasil diterapkan dalam tugas klasifikasi gambar. Kelebihan dari *ViT* ialah merupakan arsitektur yang tidak menggunakan operasi konvolusi melainkan *attention* sehingga mampu mereduksi parameter yang banyak digunakan dalam CNN. Model *ViT* terbukti bahwa lebih baik dalam efisien dan akurasi komputasi

bahkan *ViT* menunjukkan bahwa kemampuan pengenalan bentuk dengan system visual manusia (Dwi Putri, 2022).

Teknik *patching* pada *ViT* memotong langsung gambar menjadi beberapa bagian *patch*. Kemudian setiap masing *patch* disusun secara linier, dilanjutkan untuk penambahan *embedding position*, dan tersusun urutan vector yang masing-masing menghasilkan vektorisasi ke *transformer encoder*. Model arsitektur *ViT* memiliki teknik khusus dalam melakukan *patching* sehingga, model ini tidak lagi membutuhkan *patching* yang terpisah. Kelebihan dari *ViT* adalah penggunaan memori yang tidak terlalu besar untuk melakukan proses pembelajaran karena sebuah parameter yang diperlukan dalam arsitektur *ViT* (Dwi Putri, 2022).

Beberapa penelitian yang menggunakan model arsitektur *Vision Transformer*. Pertama, penelitian yang menggunakan *Vision Transformer* dalam mendeteksi Covid-19 dari Citra *X-ray* dengan memanfaatkan citra paru-paru memperoleh nilai dari *F1-score* diatas 90%. Selanjutnya penelitian klasifikasi citra pengindra jauh dengan menggunakan model arsitektur ViT dengan menggunakan piksel 224 x 224 memperoleh hasil sebesar 0,93% dalam waktu yang digandakan yaitu 32 menit menjadi 62 menit (Bazi, Basmal, Rahhal, Dayil, & Al-Ajian, 2021). Kemudian penelitian yang menggunakan *Vision Transformer* pada *image captioning* dengan Bahasa Indonesia dengan memanfaatkan lapisan 4, 5, dan 6 memperoleh nilai sebesar 46,16% pada lapisan 4. (Al-Faruq & Fudholi).

Berdasarkan beberapa penelitian yang telah banyak dilakukan dapat dilihat bahwa klasifikasi menggunakan arsitektur *vision transformer* sudah mendapatkan kinerja yang sangat baik. Hal ini dikarenakan bahwa berdasarkan dari peneliti

yang menggunakan arsitektur CNN dalam melakukan klasifikasi gambar kurang efisien dan memiliki keterbatasan dalam melakukan teknik *patching*. Sehingga penerapan klasifikasi penyakit mata yang khusus meneliti penyakit mata katarak akan dilakukan dengan tahap pembagian dua kelas yaitu mata normal dan mata katarak serta dengan pola-pola yang baru dengan menggunakan model metode yang dipakai dan diteliti adalah **Metode Arsitektur *Vision Transformer***.

1.2 Rumusan Masalah

Berdasarkan dengan penjelasan dari latar belakang diatas, maka dapat diperoleh rumusan masalah dalam penelitian ini adalah sebagai berikut :

1. Bagaimana menerapkan model arsitektur *Vision Transformers (ViT)* dalam melakukan klasifikasi penyakit katarak pada manusi.
2. Berapa persen tingkat akurasi yang dihasilkan oleh model arsitektur *Vision Transformers* dalam melakukan klasifikasi penyakit katarak pada manusia.

1.3 Batasan Masalah

Untuk memahami penjelasan penelitian tersebut, maka diterapkan batasan-batasan masalah dari penelitian yang dilakukan. Adapun Batasan masalah dalam pembahasan penelitian ini ialah sebagai berikut :

1. Data diambil menggunakan kamera *Handphone Oppo A17* dengan jumlah 1120.
2. Data berupa data sekunder dan data primer
3. Data yang didapat memiliki 2 kelas yaitu mata normal dan mata katarak dengan masing-masing jumlah 560 katarak dan 560 mata normal.

4. Dataset terbagi menjadi 3 yaitu data training 70%, data testing 10%, dan data validasi 20%
5. Algoritma deep learning yang digunakan adalah *Convolutional Neural Network* arsitektur *ViT* yang memanfaatkan model *ViT-B16*.
6. Pengujian model *ViT-B16* menggunakan 6 skema dengan *input shape* (224 x 224), jumlah epoch (25 dan 50), batch size (32), *learning rate* (0,001) dan (0,009), serta optimizer (*Adam*, *RMSprop*, *SGD*) dijadikan statis pada setiap skenario model yang diuji.
7. *Tools/Library* dan bahasa program yang digunakan dalam penelitian ini adalah *Google Colab* dan *Python*.
8. Hasil klasifikasi menggunakan hasil dari Confusion Matrix dan Classification Report diantaranya yaitu *Akurasi*, *Precision*, *Recall*, *F1-Score*.

1.4 Tujuan Penelitian

Tujuan penelitian ini memiliki ruang lingkup yang luas, maka berdasarkan dari tujuan penelitian ini adalah sebagai berikut:

1. Membangun model *ViT-B16* dari arsitektur *Vision Transformers* dalam melakukan klasifikasi mata katarak dan mata normal agar dapat mengetahui berapa persen hasil akurasi katarak atau tidak secara efisien.
2. Menganalisa Tingkat pencapaian yang dihasilkan dari setiap beberapa skenario melalui *hyperparameter* dari arsitektur *Vision Transformers*.

1.5 Manfaat Penelitian

Adapun Manfaat dari penelitian ini dilakukan adalah sebagai berikut:

1. peneliti

- a. Hasil penelitian ini diharapkan dapat memberikan manfaat bagi peneliti untuk menjelaskan tentang suatu proses awal hingga proses akhir dalam melakukan klasifikasi dengan menggunakan metode arsitektur *Vision Transformers* dari model *ViT-B16*.
- b. Mengimplementasi teori serta ilmu yang didapatkan selama penelitian dilakukan khususnya klasifikasi penyakit katarak pada mata manusia.

2. Pihak Lain

a. Mahasiswa

Hasil dari penelitian ini diharapkan mahasiswa dapat memahami metode algoritma *depp learning* dari *Convolutional Neural Network* terkhususnya dari arsitektur *ViT* yang memanfaatkan model *ViT-B16*,

b. Universitas,

Hasil penelitian ini dapat memberikan bahan pedoman atau referensi untuk penelitian selanjutnya untuk dibidang *Deep Learning* dalam citra gambar dengan menggunakan arsitektur *Vision Transformers*, terkhususnya untuk Mahasiswa Program Studi Teknik Informatika Fakultas Teknik Medan Area.

c. Tempat penelitian

Hasil penelitian ini memberikan kemajuan kepada pihak rumah sakit dalam memeriksa keadaan mata pasien dengan hasil yang efisien.

1.6 Sistematika Penulisan

Dalam penyusunan penulisan dalam penelitian ini dibentuklah sistematika penulisan yang terdiri dari beberapa bagian yang terpenting sebagai berikut.

BAB I: PENDAHULUAN

Pada bagian dari bab ini dijelaskan tentang mengenai awal dari penelitian yang disusun secara terstruktur diantaranya Latar Belakang, Tujuan Penelitian, Rumusan Masalah, Batasan Masalah, dan Manfaat Penelitian.

BAB II: TINJAUAN PUSTAKA

Dibagian bab ini dijelaskan secara terstruktur tentang teori yang ditemukan untuk sebagai acuan dalam memperluas suatu informasi dalam melakukan klasifikasi katarak dengan citra retina pada manusia menggunakan metode *Vision Transformer*. Diliputi dengan dasar-dasar dari metode *Deep Learning* Arsitektur *Vision Transformers*.

BAB III : METODE PENELITIAN

Pada bagian bab ini dijelaskan alur dari metode penelitian ini yang disusun secara tersusun dan beskala. Agar mudah dipahami oleh pembaca.

BAB IV: HASIL DAN PEMBAHASAN

Bab ini berikan tentang kesimpulan dari keseluruhan penelitian yang dilakukan dalam waktu pengujian dari metode vision transformers menggunakan objek katarak dan bab ini juga berisi tentang saran tentang kekurangan dari penelitian ini gara untuk penelitian selanjutnya dapat melakukan suatu penelitian dengan pengembangan dari metode vision transformers.

BAB V: KESIMPULAN

Bab ini menjelaskan isi dari kesimpulan penelitian ini dilakukan dan saran untuk kekurangan dari penelitian in



BAB II

TINJAUAN PUSTAKA

2.1. Klasifikasi

Klasifikasi adalah salah satu teknik data mining untuk melakukan suatu pengelompokan informasi. Klasifikasi dilakukan untuk mengenali jenis kelas data dalam bentuk kelompok yang diteliti. Penelitian yang terkait dengan klasifikasi penyakit mata katarak sudah banyak dilakukan yakni, model metode arsitektur dari Algoritma *CNN* salah satu tekniknya ialah *Deep Learning* citra digital dalam metode klasifikasi data. (Nurhalizah, Minarno, & Wicaksono, 2022)

2.2. Mata

Mata merupakan alat indra yang terdapat pada manusia yang secara konstan menyesuaikan pada jumlah cahaya yang masuk, memusatkan perhatian pada objek yang dekat dan jauh serta menghasilkan gambaran segera di hantarkan pada otak manusia. Sebagai salah satu indra penglihatan, mata mempunyai bagian-bagian yang memiliki fungsi tersendiri, dimana salah satunya adalah bagian luar seperti kelopak mata dan alis, disertai bagian dalam yaitu seperti kornea, retina dan pupil. (Agustina, 2022).

2.3. Penyakit Mata

Penyakit mata merupakan gangguan kesehatan pada mata yang menyebabkan gangguan pada penglihatan mata normal. Gejala umum yang biasa ditimbulkan dari penyakit mata yaitu merah, gatal, perih, gangguan penglihatan rabun dekat dan rabun jauh, dan kebutaan. Penyakit mata memiliki berbagai jenis penyakit yaitu katarak, glaucoma, dan retina desiasi. Namun untuk penelitian ini khusus untuk penyakit katarak (Mahardika, Widodo, & Rahman, 2019).

2.4 Katarak

Penyakit Katarak dapat terjadi pada manusia akibat adanya hidrasi lensa (Simanjuntak, Fu'adah, Magdalena, Saidah, & Wiratama, 2022). Pada umumnya Katarak ditandai dengan penglihatan yang buram akibat penuan atau bertambahnya usia. Penyakit katarak merupakan kondisi yang mengeruh pada bagian lensa mata yang tertutupi dengan suatu noda putih yang dapat mengganggu penglihatan (Gifran, Magdalena, & Faudah, 2019). Penglihatan mengalami penurunan dikarenakan lensa mata mengalami kekeruhan. Kekeruhan pada mata terjadi adanya protein pada mata membentuk suatu gumpalan. Namun, katarak bisa dialami pada kelompok usia muda dikarenakan penumpukkan glukosa yang dipicu oleh penyakit diabetes. Keadaan ini sangat mengganggu cahaya yang tidak dapat menembus lensa mata, sehingga menutup pandangan mata penderita katarak dalam melihat benda secara kabur seperti tertutup kabut. (Suwanda & Juniati, 2022).

2.5 Slit Lamp

Slit Lamp merupakan sebuah mikroskop binocular yang menghasilkan seberkas Cahaya lurus dari celah lampu pijar. Slit lamp adalah alat yang terdiri dari sumber Cahaya yang berintensitas tinggi dengan mendapatkan titik fokus untuk menyinari lembaran tipis Cahaya ke mata.

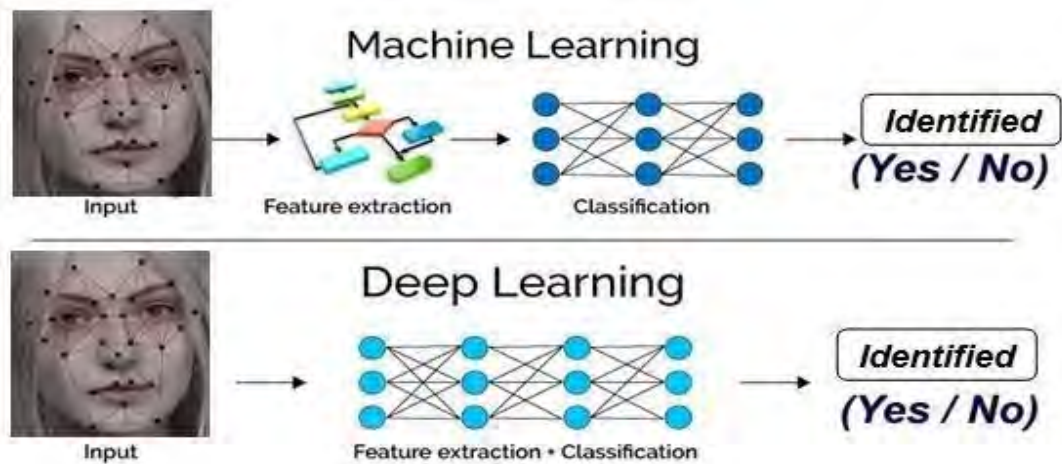


Gambar 2. 1 Slit Lamp

Seorang pemeriksa dapat mengamati setiap lapisan mata dengan mengatur intensitas cahaya yang diarahkan ke mata. Cahaya dari *slit lamp* akan menghasilkan pantulan atau refleksi.

2.6 *Deep Learning*

Deep Learning adalah merupakan sub-cabang dari *Machine Learning* yang memanfaatkan jaringan saraf tiruan (*Artificial Neural Network*) dan memiliki jumlah layer yang begitu banyak. Proses *Learning* dari metode ini dengan mengidentifikasi pola yang tersembunyi pada data dan mengklasifikasi untuk menghasilkan output yang akurat (Marpaung, 2023).



Gambar 2. 2 Ilustrasi Machine Learning dan Deep Learning

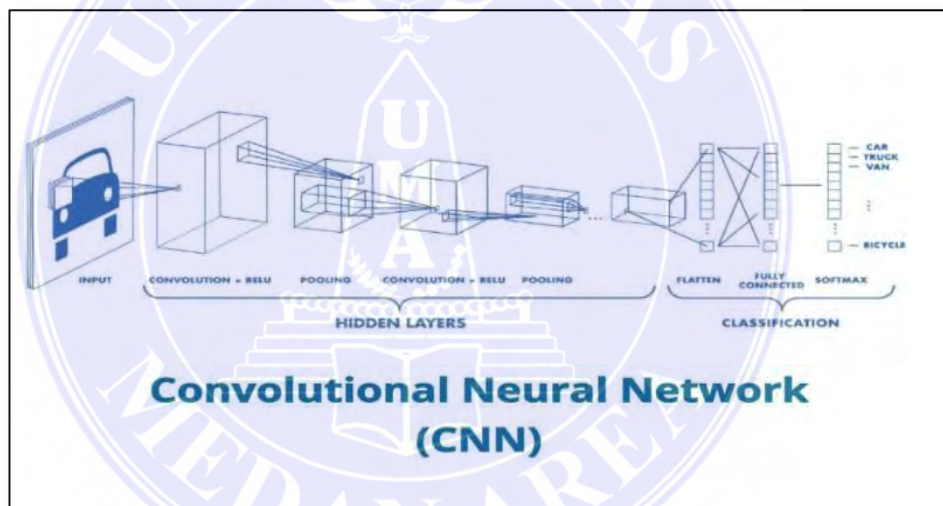
Sumber : (Halbouni , Gunawan , & Habaebi , 2022)

Perbedaan dari keduanya yaitu diantaranya *Machine Learning* dan *Deep Learning* terletak pada penggunaan *feature extractor*. Pada ML perlu dibuat *feature extractor* secara manual yang memakan waktu dan tenaga yang cukup banyak. Sementara DL untuk melakukan fitur-fitur *feature extractor* dan *Classification* dilakukan secara otomatis dan bersamaan. Namun hal ini memerlukan jumlah data yang besar untuk melatih algoritma dari *Deep Learning* (Halbouni , Gunawan , & Habaebi , 2022).

2.7 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) adalah salah satu jenis neural network yang biasa digunakan pada data gambar untuk mendeteksi dan mengenali objek. Dimana metode proses ini melakukan proses data ke dalam bentuk array yang memanfaatkan sifat nature signals yaitu koneksi lokal, *share weights*, *pooling*, dan penggunaan banyak lapisan.

Terdapat model dalam memproses pengolahan citra yaitu *Convolution Layer*, *Pooling Layer*, *Dropout Layer*, *Full Connected Layer*. Empat bagian citra yang akan diproses melalui filter, lalu citra akan direduksi dimana bagian ini akan mengambil nilai terbesar dari setiap grid dan dilanjutkan dengan mengurangi dimensi data dengan melakukan pencegahan *over-fiting* serta dilanjutkan dengan transformasi pada dimensi data agar diklasifikasikan. Setiap model harus memiliki data pelatihan yang baik agar proses dalam mengklasifikasi data dapat hasil yang akurat. Seperti pada gambar 2.5 dari bentuk *Convolution Neural Network*. Terdapat 4 jenis lapisan *Convolution Neural Network* yaitu:



Gambar 2. 3 Convolution Layer

Berikut ini penjelasan dari gambar 2.3 *convolution layer* yaitu sebagai berikut:

- 1) *Convolutional Layer* adalah lapisan konvolusi yang menggunakan filter untuk mendeteksi karakter dari suatu objek atau citra yang menghasilkan sebuah linear dari citra input yang sesuai. Proses parameter untuk memodifikasi setiap lapisan diantaranya ada filter, stride, dan padding. Dimana pada stride

mengontrol bagaimana proses filter pada sebuah input yang setiap pergerakannya memiliki ukuran yang panjang terhadap piksel yang telah ditentukan dengan nilai tertentu disekitar data. Setelah itu input data supaya menghasilkan hasil terlalu kecil dan informasi tidak dapat hilang.

- 2) *ReLU Layer Relu (Rectification Liner Unit)* merupakan suatu lapisan yang manfaatnya memperkenalkan *non linearitas* ke suatu jaringan hingga meningkatkan *representasi* dari suatu model dimana output menghasilkan berupa peta fitur yang telah diperbaiki.
- 3) *Pooling* Lapisan adalah merupakan suatu lapisan dalam mengambil suatu data yang kegunaannya untuk mengurangi dimensi pada fitur yang telah diperbaiki dan akan melewati lapisan penggabungan untuk menghasilkan fitur yang telah digabungkan.
- 4) *Fully Connected Layer* pada lapisan ini terkoneksi secara penuh pada setiap aktivitas yang telah dilakukan pada setiap lapisan yang terdahulu.

2.8 *Vision Transformer*

Vision Transformer merupakan salah satu jenis model dari metode arsitektur Transformer yang dikembangkan dari algoritma *depp leraning Convolutional Neural Network*. *Vision Transformers* dinyatakan dalam matematis nilai variable X menjadi inputan ukuran (H, W, C) dimana H adalah nilai tinggi, W adalah nilai lebar dan C adalah nilai jumlah saluran (Karimah, Anas, & Arsyad, 2023). Teknik ini membagi gambar masukan menjadi beberapa bagian yang lebih kecil. Pada hasil dari *patch* ini digabungkan sehingga membuat jaringan dapat memahami keseluruhan fitur dan struktur gambar secara keseluruhan tanpa bergantung pada

urutannya. Setiap gambar diproses sebagai sekumpulan *patch*, dan model mengekstrak fitur dari *patch* secara paralel.

2.8.1 Komponen Utama Arsitektur *Vision Transformers (ViT)*

Proses penyematan *patch* dan proses token kelas adalah dua elemen yang sangat penting dari arsitektur *ViT*. Proses ini sangat berfungsi untuk mengubah masing-masing *patch* yang rata menjadi mahir dalam menangkap fitur-fitur penting dari *patch* tertentu dan proses token kelas beroperasi dengan mengubah vektor token kelas yang dapat dilatih menjadi vektor fitur dengan dimensi yang diperkecil dan berfungsi untuk menggabungkan suatu informasi dari semua *patch* yang mewakili keseluruhan gambar.

Modul *multihead self-attention (MSA)* melakukan transformasi vector masukan menjadi vector kueri, kunci dan nilai. Proses yang sangat penting sebelum menghitung nilai akhir keluaran dan memberdayakan model untuk memahami saling ketergantungan dan interaksi yang rumit diantara beragam konsisten gambar. Dengan memperhatikan setiap titik skala untuk mempertimbangkan nilai bobot yang terkait dengan setiap nilai. Agar untuk memastikan stabilitas pelatihan jaringan saraf, seperti Teknik normalisasi lapisan dan koneksi sisa diterapkan pada keluaran yang dihasilkan.

2.8.2 Model ViT Pre-Trained

Adapun model yang dimiliki arsitektur *ViT* dan jumlah parameter yang berbeda dan arsitektur *ViT* memiliki model yang spesifik untuk digunakan dalam penelitian klasifikasi katarak, diantaranya yaitu :

1. ViT-B16

Model ViT-B16 adalah ViT dengan skala menengah yang memiliki 12 lapisan encoder transformator. Masing-masing dengan 16 *Attention Head*. Model ini telah banyak dilatih pada kumpulan ImageNet yang mencakup lebih dari satu juta gambar. Model ini dapat disesuaikan dengan tugas pengenalan gambar tertentu.

2. Model ViT-I16

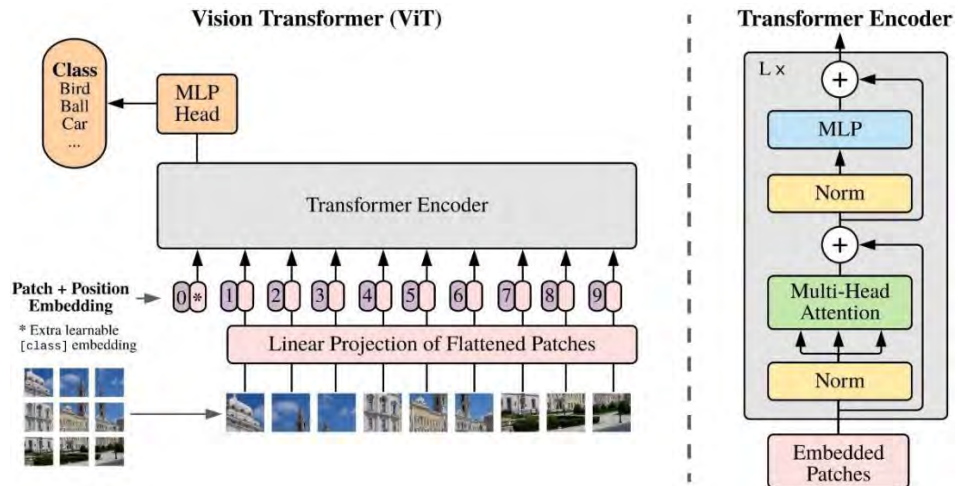
Model ViT-I16 merupakan bagian model ViT yang memiliki skala besar dan 16 lapisan *encoder transformator*. Masing-masing dengan 32 *attention head*. Model dilatih pada Kumpulan data sebelum JFT-300M yang berisi lebih dari 300 juta data gambar yang lebih kecil dalam tugas pengenalan gambar.

3. Model ViT-I32

Model ViT-I32 adalah model dari arsitektu ViT dengan skala besar yang memiliki 24 lapisan *encoder transformator* dengan masing-masing 32 *attention head* model ini dilatih sebelum pada saat Kumpulan data 21k-imagenet yang berisi lebih dari 14juta dari gambar.

2.9 Tahap Kinerja *Vision Transformer*

Pada tahap kinerja arsitektur *Vision Transformer* memiliki komponen utama dalam membangun klasifikasi gambar dengan menggunakan arsitektur *Vision Transformer* (Kurnia, Ningsi, Monalisa, & Fahmi, 2019). Adapun ilustrasi yang dapat dilihat seperti pada gambar dibawah ini.



Gambar 2. 4 Ilustrasi Proses Kinerja Vision Transformer

Berikut ini penjelasan dari komponen-komponen kinerja utama dalam arsitektur *vision transformer* yaitu:

1. *Patch Embedding*: adalah proses ini mengambil gambar sebagai input dan memisahkan menjadi bagian-bagian kecil. Setiap potongan kemudian diubah menjadi vektor fitur menggunakan lapisan linear operasi *embedding*. *Patch Embedding* menghasilkan representasi vektor untuk setiap patch dalam gambar. Kemudian meneruskan setiap patch melalui sebuah lapisan linear untuk menghasilkan vektor fitur. Untuk menjaga tata ruang potongan seperti pada gambar aslinya. Informasi posisi *epos* dikodekan dan ditambahkan ke representasi tambalan. Urutan tambalan tertanam yang dihasilkan dengan token z_0 diberikan dalam persamaan 1

Persamaan (1);

$$H_P = \frac{H}{P}, W_P = \frac{W}{P} \quad (2.1)$$

2. *Position Encoding* atau linear projection : tahap kinerja ini adalah tahap setelah melawati tahap *patch embedding*, dimana posisi spasial dari setiap patch dalam gambar ditambahkan menggunakan *positional encoding*. *Positional encoding* memberikan pengetahuan tentang letak spasial relatif setiap potongan dalam gambar. Hal ini membantu model untuk memahami hubungan spasial antara potongan-potongan tersebut.

Persamaan (2):

$$\begin{aligned} \text{Dense Layer } z^i \\ = w \cdot x^i + b \end{aligned} \quad (2.2)$$

3. *Transformer Encoder*: tahap selanjutnya yang setiap patch vektor telah memperoleh hasil dari melalui proses linear lapisan-*positional encoding* maka diteruskan melalui serangkaian dari *self-attention* dan lapisan *feed-forward neural network*. *Self attention* adalah model yang memperhatikan hubungan antara potongan-potongan yang berbeda sementara lapisan *feed-forward neural network* menghasilkan representasi fitur yang lebih banyak memiliki dua sub komponen utama :

- a) Blok *Self-Attention Multihead (MSA)* persamaan (3)
- b) Blok padat umpan-maju yang terhubung sepenuhnya (MLP) persamaan (4).
- c) Blok terakhir terdiri dari dua lapisan padat dengan aktivasi *GeLU* diantaranya. Masing-masing dari dua subkomponen *encoder* yang menggunakan koneksi lompatan *residual* dan didahului oleh lapisan normalisasi.

Rumus

$$z'_\ell = \text{MSA}(\text{LN}(z_{\ell-1})) + z_{\ell-1}, \ell = 1 \dots L \quad (2.3)$$

$$z_\ell = \text{MLP}(\text{LN}(z_\ell^t)) + z_\ell^t, \ell = 1 \dots L \quad (2.4)$$

pada lapisan terakhir pembuat encoder, diambil elemen pertama dalam urutan z ke pengklasifikasi kepala eksternal untuk memprediksi label kelas.

$$y = \text{LN}(z_\ell^t) \quad (2.5)$$

$$[Q, K, V] = zU_{QKV}, U_{QKV} \in \mathbb{R}^{d \times 3DK} \quad (2.6)$$

$$A = \text{softmax}\left(\frac{QK^T}{\sqrt{DK}}\right), A \in \mathbb{R}^{n \times n} \quad (2.7)$$

$$SA_{(z)} = A \cdot V \quad (2.8)$$

Bobot dari *attention* dengan menghitung produk titik penskalaan nilai kunci kueri. Tiga nilai dihasilkan yaitu Q (*queri*), K (kunci), dan V (nilai) dengan mengalikan elemen terhadap tiga matriks yang dipelajari U_{QKV} (Persamaan (5)). Untuk menentukan keterkaitan antara suatu unsur dengan unsur lainnya pada suatu deret, perkalian titik dihitung antara suatu unsur dengan unsur lainnya pada deret perkalian antara vektor Q unsur ini dengan vector K unsur lainnya. Hasilnya menentukan kepentingan relative dari potongan dalam urutan. Kemudian diskalakan dan dimasukkan kedalam *softmax* untuk menemukan *patch* dengan skor *attention* yang tinggi (persamaan (6)). Pada operasi penskalaan suatu objek dilakukan oleh blok SA serupa dengan objek standar, namun menggabungkan suatu dimensi kunci sebagai faktor penskalaan. suatu nilai setiap vector embedding patchnya dikalikan dengan output dari *softmax* untuk menemukan patch dengan skor *attention* yang tinggi (persamaan (6)). Operasi perhitungan

penuh diberikan persamaan dibawah ini. Hasil dari semua *multiple attention head* digabungkan menjadi satu dan kemudiadi proyeksikan melalui lapisan umpan-maju dengan bobot W yang dapat diketahui ke dimensi yang diinginkan. Operasi perhitungan ini dinyatakan dengan persamaan seperti dibawah ini.

Rumus:

$$MSA(z) = \text{Concat}(SA_1(z); SA_2(z); \dots SA_h(z))W, \quad W \in \mathbb{R}^{h \cdot DK \times D} \quad (2.8)$$

4. *Classification Head*: setelah informasi gambar diproses melalui beberapa blok *transformer encoder*, representasi fitur ini kemudian diinputkan ke lapisan output yang akan menghasilkan probabilitas untuk setiap kelas

2.10 Confusion Matrix dan Classification Report

Dalam *Machine Learning* dan *Deep Learning* adalah salah satu teknik untuk melakukan klasifikasi dengan metode *Supervised Learning*. Evaluasi performa penelitian ini merupakan tahap penting dalam *Life Cycle* model *Machine Learning* dan *Deep Learning*. Tahap eveluasi performa model klasifikasi terdapat dua Teknik yang umum digunakan yaitu *Confusion Matrix* dan *Classification Report* (Marpaung, 2023)

1. Confusion Matrix

Confusion matrix adalah tabel berukuran $N \times N$ (dengan N adalah jumlah kelas/label/kategori) yang memuat informasi tentang jumlah prediksi yang tepat atau salah dari suatu model klasifikasi, yang berguna untuk membandingkan nilai aktual dengan nilai prediksi. Setiap baris dalam matriks mewakili kelas yang sebenarnya, sedangkan setiap kolom mewakili kelas yang

diprediksi. *Confusion matrix* menghasilkan empat jenis nilai seperti pada gambar dibawah ini.

Label Sebenarnya		Label Prediksi	
		Positive (1)	Negative (0)
Positive (1)	TP (True Positive)	FN (False Negative) Error tipe 2	
	FP (False Positive) Error tipe 1	TN (True Negative)	
Negative (0)			

Classification Report Metrics	
$accuracy = \frac{TP + TN}{TP + FP + FN + TN}$	
$precision = \frac{TP}{TP + FP}$	
$recall = \frac{TP}{TP + FN}$	
$f1score = 2 \times \frac{recall \times precision}{recall + precision}$	

Gambar 2. 5 Confusion Matrix dan Rumus Classification Report.

Berikut ini penjelasan lebih detailnya dari sebuah gambar 2.5 yaitu :

- True Positive (TP)*: Prediksi nilai positif & nilai sebenarnya positif.
- True Negative (TN)*: Prediksi nilai negatif & nilai sebenarnya negative.
- False Positive (FP)*: Prediksi nilai positif & nilai sebenarnya negative
- False Negative (FN)*: Prediksi nilai negatif & nilai sebenarnya positif

2. *Classification Report*

Meskipun *confusion matrix* memberikan informasi yang detail tentang kinerja suatu model dalam melakukan klasifikasi, tetapi masih sulit bagi pengguna untuk memahami seberapa baik kinerja tersebut. Oleh karena itu, *confusion matrix* dapat digunakan untuk menghitung metrics yang dapat mengukur kinerja model, yang kemudian disebut sebagai *classification report* menggunakan beberapa *matrix*, yaitu:

- Accuracy* menggambarkan sebuah jumlah data dalam bentuk persentase yang dilakukan pada klasifikasi atau prediksi secara benar oleh algoritma tersebut.

$$\text{Rumus : } Accuracy = (TP + TN) / (TP + TN + FP + FN)$$

- b) *Precision* adalah nilai yang menggambarkan ketepatan akurasi dari metode yang digunakan dalam melakukan klasifikasi di kelas dengan hasil prediksi model yang benar.

$$\text{Rumus : Precision} = TP / (TP + FP)$$

- c) *Recall* menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi untuk mendapatkan nilai yang dapat diukur hasilnya dalam bentuk persentase data yang terklasifikasi dengan benar.

$$\text{Rumus : Recall} = TP / (TP + FN)$$

- d) *F1 score* menggambarkan perbandingan rata-rata precision dan recall yang dibobotkan.

$$\text{Rumus : F1score} = 2 \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$$

2.11 Penelitian Terdahulu

Dalam penelitian ini, peneliti terlebih dahulu mencari sebuah referensi dari tahun 2018-2022 (Firdaus, Imran, Bakti, & Suryadi, 2022), berdasarkan percobaan yang akan dilakukan sebagai bahan acuan untuk dilakukan penelitian ini (Labib, Hadi, Widayaka, & Paradisa, 2022) diantaranya yaitu:

Tabel 2. 1 Penelitian Terdahulu

No	Referensi	Hasil Penelitian
1.	(Gifran, Magdalena, & Faudah, 2019)	Pengujian klasifikasi katarak, glaucoma, retina deses menggunakan metode <i>DVY</i> dan <i>SVM</i> dengan diperoleh berdasarkan tabel pengujian bahwa didapatkan hasil akurasi yang terbaik dengan memakai jenis <i>DWT</i> yang diperoleh hasil

		akurasi 80% dan waktu komputasi sebesar 6,89% sehingga parameter memberikan akurasi yang terbaik dengan menggunakan ukuran 512x512.
2.	(Simanjuntak, Fu'adah, Magdalena, Saidah , & Wiratama, 2022)	Pengimplementasi yang menggunakan metode <i>CNN</i> berdasarkan gambar fundus yang memperoleh hasil akurasi yang terbaik yaitu dengan hasil 0,93 % dan 0,92 % dibandingkan dengan model arsitektur <i>CNN</i> yang lainnya.
3.	(Wiguna, Fardela, & Selly, 2019)	Histogram mampu membedakan citra katarak matur, imatur, normal sehingga dalam melakukan pengklasifikasian jenis katarak metode histogram mendapatkan hasil 0,90% dan 0,91% dari setiap mata dan mendapatkan ukuran lebar dari histogram
4.	(Dwi Putri, 2022)	Penelitian ini adalah untuk menentukan penyakit <i>glaucoma</i> menggunakan arsitektur <i>ViT</i> dengan tahapan penelitian adalah pengumpulan data yang dibagi menjadi 2 pembagian data yaitu data testing dan traning hingga mendapatkan hasil evaluasi kinerja akurasi, <i>sensitivity</i> , <i>spesifitas</i> , <i>F1-score</i> dan <i>cohen's kappa</i> berturut-turut adalah 96,74%, 95,10%, 97,57%, 95,10% dan 92,46% hasil yang diperoleh adalah hasil dari metode <i>Vision Transformer</i> yang sangat baik dan memiliki kinerja Teknik <i>patchingnya</i> sangat efisien.

5.	(Figo, Yudistra, & Widodo, 2023)	Penelitian dari pengujian deteksi Covid-19 dari citra X-ray menggunakan <i>ViT</i> dengan hasil pengujian yang dilakukan ialah pengaruh parameter, <i>augmentasi data</i> , <i>transfer learning</i> mendapatkan hasil evaluasi yang terbaik sebesar 0,96% dan pada waktu validasi 0,95% pada waktu testing.
----	----------------------------------	--

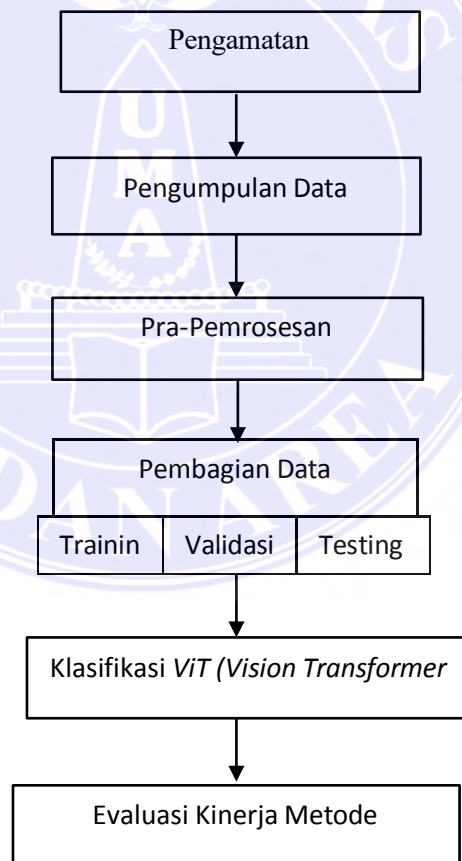


BAB III

METODE PENELITIAN

3.1 Tahapan Penelitian

Konteks untuk penelitian ini membutuhkan tahapan yang jelas dalam setiap proses yang dilewati. Tahap-tahap penelitian ini berguna sebagai panduan untuk melakukan penelitian. Tahapan prosesnya direncanakan secara terstruktur agar dan digunakan sebagai panduan untuk melakukan penelitian ini menjadi lebih terarah sampai mendapatkan hasil yang dicapai oleh peneliti. Berikut ini Langkah-langkah dalam melakukan penelitian ini dapat ditampilkan Gambar 3.1.

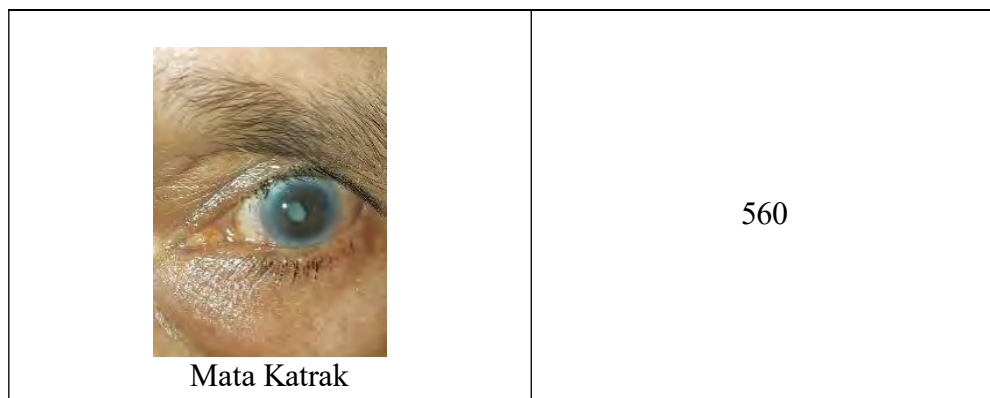


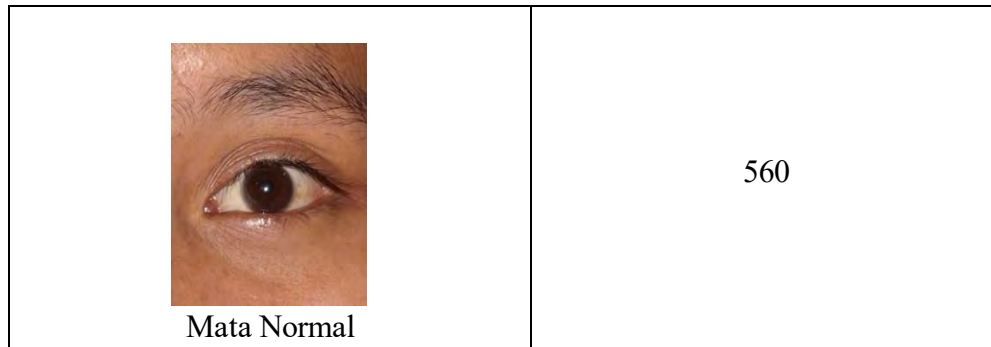
Gambar 3. 1 Kerangka Kinerja Penelitian

Pada Gambar 3.1 pertama, pengamatan dalam mempersiapkan data yang dikumpulkan menjadi *dataset* lalu *dataset* dikumpulkan dan dikelompokkan sesuai *class* nya masing-masing. Kemudian dilanjutkan dengan melakukan pra-pemrosesan data gambar dengan mengcroping data kemudian mengcompresskan ukuran gambar agar tidak terlalu besar dan menjadi sama rata. Selanjutnya meresize dan mengaugmentasikan data gambar. lalu dilakukan pembagian dataset dengan dibagi menjadi 3 yaitu data training, validasi, dan testing. Pada data training dilakukan dengan tujuan agar computer dapat mengenali objek. Selanjutnya menggunakan hyperparameter dengan hasil model yang terbaik sehingga mendapatkan klasifikasi pada data training. Lalu pada tahap tahap evaluasi dilakukan dengan mencoba menghitung dan mengukur Tingkat keberhasilan jika pola mirip atau mendekati pola training maka output dari klasifikasi yaitu hasil sedangkan data testing langsung pada tahap klasifikasi.

3.2 Teknik Penumpulan Data

Dataset dalam penelitian ini dilakukan dengan pengamatan secara langsung dan secara tidak langsung dengan mengumpulkan citra retina mata normal dan mata katarak pada UPT Rumah Sakit Khusus Mata Medan Helvetia. Diikuti dengan diskusi dengan dokter spesialis mata.





Gambar 3. 2 Jumlah Dataset Yang Digunakan

3.3 Insialisasi Hyperparameter

Insialisasi hyperparameter adalah suatu proses pengaturan nilai awal yang digunakan sebelum algoritma model pembelajaran dimulai. Berikut ini parameter dan value yang digunakan dalam penelitian tersebut

Tabel 3. 1 Insialisasi Hyperparameter

Parameter	Value
Epoch	{25,50}
Batch Size	{32}
Learning Rate	{0,001 ; 0,009}
Optimizer	{Adam, RMSprop, SGD}

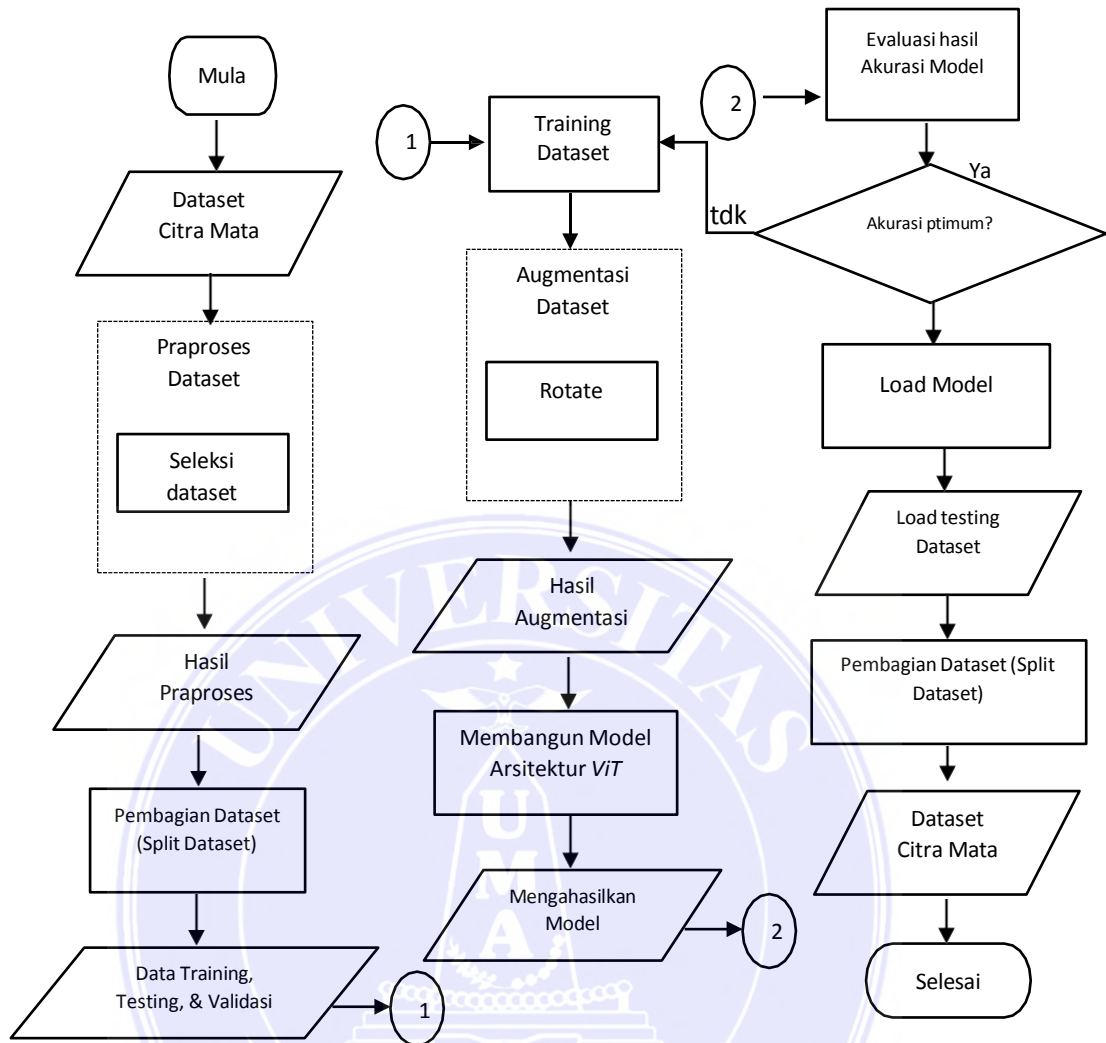
Pada Tabel 3.3 dapat dilihat bahwa *hyperparameter* yang dipakai yaitu *epoch* 15 dan 25 merupakan nilai yang digunakan untuk menentukan berapa kali proses pelatihan model untuk dibaca dan dipelajari oleh seluruh sampel di computer tersebut. Batch size yang dipakai adalah 32 yang merupakan ukuran jumlah gambar dalam beberpa kelompok yang berisi data sampel yang akan digunakan untuk melatih model dan sampel yang akan dibaca oleh model dan dipelajari melalui proses iterasi. *Learning Rate* yaitu 0,001 dan 0,009 yang merupakan salah satu parameter training yang berfungsi untuk menghitung nilai koreksi bobot pada waktu proses *training*. Semakin besar nilai *learning rate* maka proses

training akan berjalan semakin cepat dan ketelitian jaringan akan semakin berkurang dan optimizer yang digunakan adalah SGD, Adam, dan RMSprop. *Optimizer* bekerja untuk mengubah parameter contohnya seperti bobot dan learning rate dalam proses pelatihan model. SGD (*Stochastic Gradient Descent*) adalah metode yang tidak memperhitungkan kelengkungan fungsi kerugian. RMSprop (*Root Mean Square Propagation*) dan Adam (*Adaptive Moment Estimation*) adalah metode pengoptimalan yang lebih canggih dengan menyesuaikan kecepatan pembelajaran berdasarkan Riwayat gradien terkeni dengan memungkinkan mereka beradaptasi dengan bentuk fungsi kerugian dan menemukan Solusi optimal dengan lebih efisien.

3.4 Alur Proses Klasifikasi

Pada penelitian ini digunakan metode CNN dengan arsitektur *Vizion Transformer* akan diimplementasikan untuk mengklasifikasi mata katarak tersebut. Pada metode CNN dengan arsitektur *Vizion Transformer* dipilih karena merupakan salah satu algoritma *Deep Learning* yang memiliki tingkat akurasi yang cukup efisien. Oleh karena itu dengan menggunakan sistem yang memiliki tingkat kecerdasan untuk melakukan klasifikasi mata katarak atau tidak dengan akurasi tinggi dan efisien layaknya kemampuan manusia.

Langkah yang dilakukan dalam penelitian ini dapat ditampilkan dalam bentuk diagram alir (*flowchart*) contohnya pada Gambar 3.2.



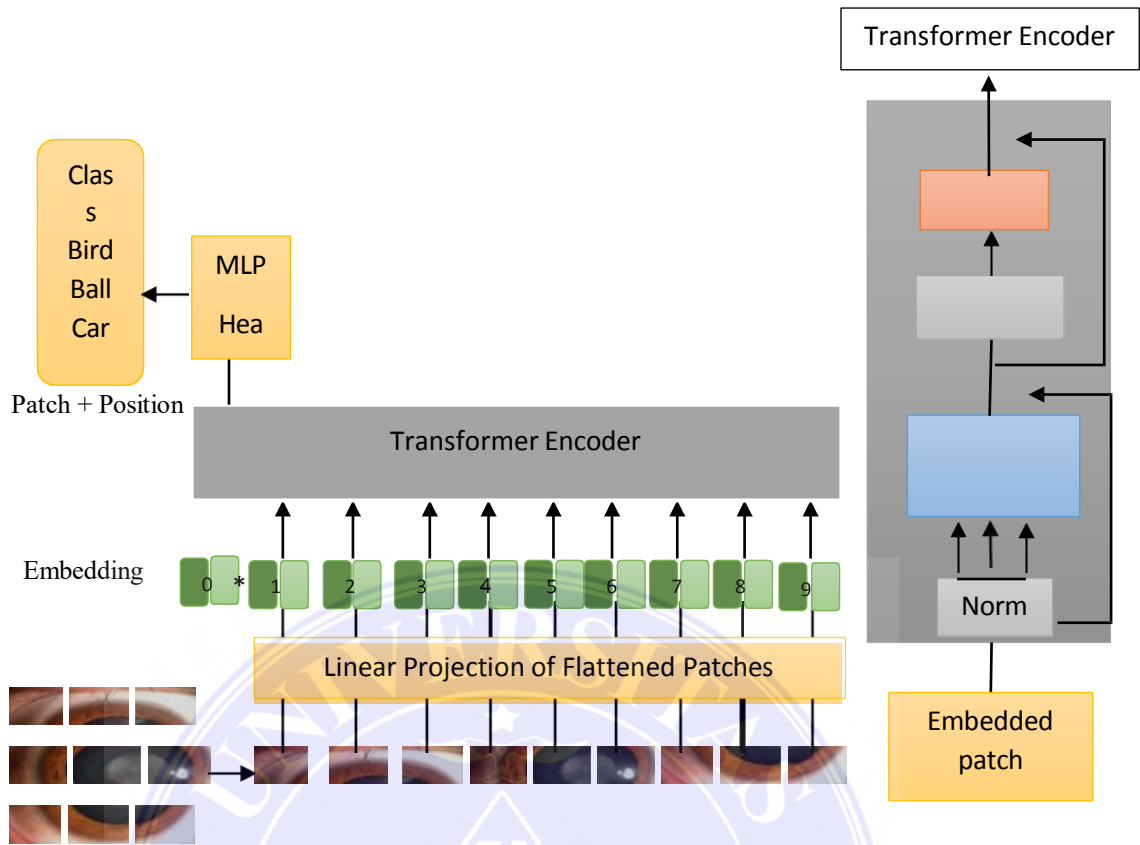
Gambar 3. 3 Flowchart Kinerja Sistem Klasifikasi

Keseluruhan proses klasifikasi pada Gambar 3.2 melalui beberapa langkah. Langkah pertama dilakukan pengumpulan *dataset* citra mata (mata katarak & mata normal). Langkah kedua akan dilakukan praproses (*preprocessing*) *dataset* agar data citra sesuai dengan kriteria standar saat memasuki arsitektur *ViT*. Langkah selanjutnya *dataset* akan dibagi menjadi data *training*, *testing*, dan *validasi*. Dimana data *training* akan digunakan untuk melatih *model* dan data *testing* digunakan untuk menguji *model* arsitektur *ViT*. Langkah keempat akan

dilakukan perancangan arsitektur model CNN dengan menggunakan data *training* yang didalam penelitian ini akan menggunakan arsitektur *ViT* untuk menghasilkan dengan akurasi yang akurat dan efisien. Dilanjutkan ke langkah selanjut yaitu langkah kelima, dimana model yang akan dihasilkan akan diuji menggunakan data *testing* untuk mengevaluasi hasilnya lalu memilih model terbaik dengan akurasi yang paling tinggi dan tahap berakhir.

3.4 Simulasi Arsitektur *Vision Transformers*

Penelitian ini akan dibuat *image classification model* untuk mengenali tingkat persentase mata katarak dengan menggunakan *pre-trained model Vision Transformers*. Arsitektur *Vision Transformers* dipilih karena arsitektur ini sangat efisien. Dimana arsitektur ini memberikan proses yang tidak begitu panjang. Proses arsitektur ini melalui sebuah *patch* gambar secakra terstruktur dari kiri atas dilanjutkan dengan kanan atas serta diikuti dengan gambar bawah kiri dan kanan hingga susunan gambar memiliki nilai $X_1, X_2, X_3 \dots \dots \dots X_n$ hingga semua gambar tersusun secara berurutan dan dilanjutkan bersamaan melewati proses lapisan-lapisan layer dari arsitektur *Visison Transformer* dan memberikan hasil akurasi dari hasil penelitian terdahulu dalam permasalahan klasifikasi citra. Berikut ini contoh simulasi dari metode *vision transformer*.



Gambar 3. 4 Rancangan Simulasi Arsitektur Vision Transformer

Gambar 3.7 menjelaskan tahap-tahap proses dari arsitektur *Vision Transformer* yang menggunakan pada klasifikasi pada citra mata katarak. Berikut ini penjelasan yang diuraikan sebagai berikut:

1. Pada tahap input layer, citra yang digunakan berukuran 224x224 angka 3 yang menandakan citra tersebut adalah channel image RGB pada citra warna (*true color*). Sebelum citra diinputkan, maka terlebih dahulu akan diaugmentasi seperti yang sudah dijelaskan dibagian *augmentasi* dataset diatas. Berikut ini proses pengambilan *augmentasi* dataset dilakukan seperti diatas. *Augmentasi* dilakukan pada citra yaitu augmentasi rotasi, dimana rotasi dilakukan 90°, Derajat searah jarum jam. Hitung nilai kordinat baru pada setiap piksel setelah rotasi, untuk rotasi 90°, dimana nilai titik kordinat baru adalah $(x', y'$ dengan rumus:

$$x' = \cos(\theta) * x - \sin(\theta) * y$$

$$y' = \sin(\theta) * x + \cos(\theta) * y$$

Keterangan:

x', y' : variable nilai piksel

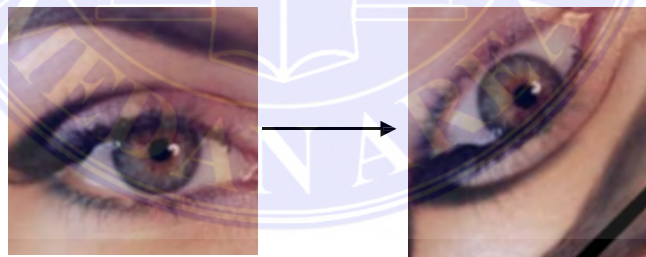
θ : sudut dari rotasi

Untuk piksel yang digunakan adalah :

$$(0,0) : x' = \cos(90) * 0 - \sin(90) * 0 = 0 ; y' = \sin(90) * 0 + \cos(90) * 0 = 0$$

$$(0,1) : x' = \cos(90) * 1 - \sin(90) * 0 = -1 ; y' = \sin(90) * 1 + \cos(90) * 0 = 1$$

Jadi hasil perhitungan pengambilan *augmentasi* dataset seperti pada gambar dibawah ini:



Gambar 3. 5 Hasil Augmentasi Dataset

- Selanjutnya citra tersebut diinputkan pada tahap *feature extractor* untuk mengekstraksi *fitur* dari dataset menggunakan *pre-trained model* dengan arsitektur *Vision Transformer* pada gambar 3.8 terdapat beberapa parameter

dari arsitektur *Vision Transformer* yang disesuaikan didalam penelitian tersebut yaitu sebagai berikut :

a) *Include_top=false*

Berfungsi untuk menghilangkan *layer* terakhir pada *model Vision Transformer*. *Layer output default* dari *Vision Transformer* adalah 12 atau lebih. Pada penelitian ini jumlah kelas dari output adalah 2 kelas yaitu mata katarak dan mata normal. Oleh karena itu penelitian ini menggunakan parameter *include_top=false* untuk menghilangkan layer terakhir pada *model Vision Transformer*.

b) *Weights = imagenet*

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

c) *Trainable = False*

berfungsi untuk semua layer tidak dapat ditraining ulang. Oleh karena, hal ini penelitian ini menggunakan metode CNN dari model arsitektur *vision transformer* sebagai *feature extractor* untuk mengekstraksi fitur dari citra dataset, dan tidak mengubah bobot-bobot yang sudah dilatih pada model *vision transformer*.

3. Pada *Tensorflow keras* atau *Pustaka keras*, arsitektur *Vision Transformer* terdiri dari 4 tahap dibagi dari beberapa blok layer yaitu *layer of multiheaded self-attention*, *MLP Block* dan *Layer norm (LN)*. Pada *Layer* pertama dilakukan operasi pengkodean posisi untuk melakukan *ekstraksi fitur* pada dataset citra input. Tahap ini adalah sebuah proses yang dikombinasi dengan *encoder* dan *decoding* dengan tahap awal melakukan sebuah *patch embedding* pada gambar

dilanjutkan pada *flattened patches* lalu gambar akan melewati sebuah lapisan dan setiap blok yang sudah disediakan oleh model arsitektur *Vision Transformer*. Dimana konvolusi yang digunakan pada input citra layer berukuran 224x224 dan channel 3 serta stride yang digunakan adalah 16x16. Pada penjelasan diatas adalah sebuah ilustrasi, maka dari itu untuk mempermudah ilustrasi tersebut maka dijelaskan perhitungan manual tersebut secara terurai agar model yang dibangun dapat kita ilustrasikan secara efisien melalui tahap perhitungan manual penelitian ini. berikut ini perhitungan manual untuk membangun model *Vision Transformer* :

a) *Patches + Position embedding*

Berfungsi untuk melakukan perpecahan pada gambar pada citra yang diinputkan dan telah diatur *hyperparameter* dan sebagainya yang berukuran 224*224. Namun, untuk perhitungan manualnya simulasi yang digunakan berukuran 48*48 maka hasil yang diperoleh adalah 9 *patches of size 16*16* dengan rumus sebagai berikut :

$$H_P = \frac{H}{P}, W_P = \frac{W}{P}$$

$$H_P = \frac{48}{16}, W_P = \frac{48}{16}$$

$$H_P = \frac{48}{16}, W_P = 48/16$$

Keterangan:

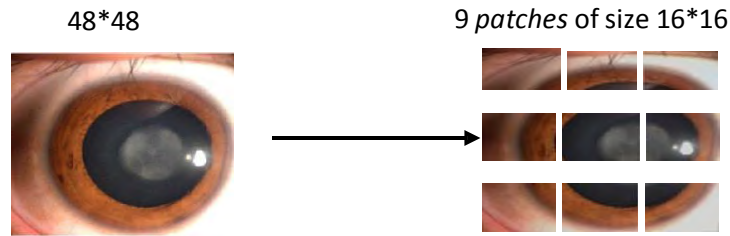
H_P : Tinggi

W_P : Panjang

Maka dibentuk dengan sederhana,

$$H_p = (48/16) * (48/16) = 9$$

Untuk jumlah *patch* telah diperoleh dan hasil yang dilakukan dapat dilihat pada gambar 3.9 dibawah ini.



Gambar 3. 6 Patches Gambar Mata Katarak

Dari hasil simulasi diatas maka *patches* pada gambar sama dengan 9. Setelah gambar telah dipotong menjadi 9 bagian maka gambar akan melewati proses Linear Projection of Flattened Patches.

b) *Flattened Patches + Proyeksi Linear*

Proses ini berfungsi untuk melakukan klasifikasi pada *flattened patches* dalam pengaplikasian *linear projection*. Dimana selesai langkah *patches* maka dilanjut ke proses *Flattened Patches + Proyeksi Linear* agar dataset tersebut dapat diklasifikasi dengan melalui proses ini Adapun langkah-langkah untu melewati proses *linear projrction of flattened patches* diantaranya yaitu :

$$\text{Citra X} = \begin{bmatrix} X_1 & X_2 & X_3 \\ X_4 & X_5 & X_6 \\ X_7 & X_8 & X_9 \end{bmatrix}$$

Untuk mempermudah perhitungan manualnya maka disederhanakan kedalam bentuk angka sebagai nilai dari piksel gambar yang nilai $X_1, X_2, X_3, \dots, X_9$ lalu diubah menjadi nilai 1, 2, 3, 9.

$$\text{Citra } X = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

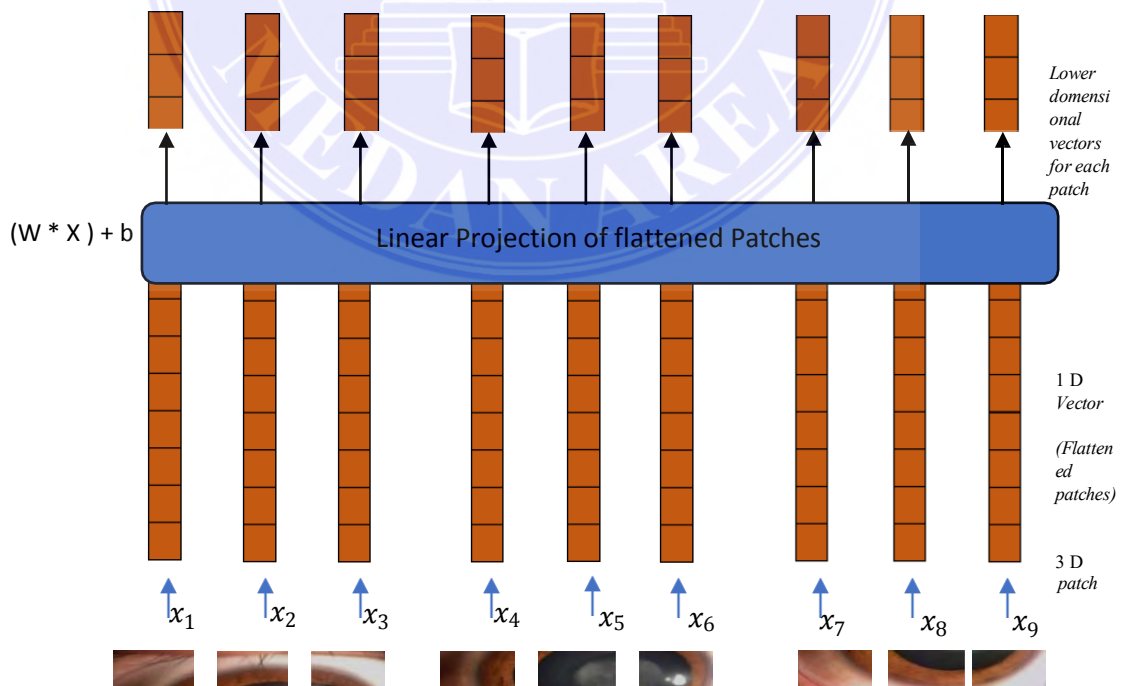
Kemudian selanjutnya nilai citra tersebut disederhanakan menjadi 1 vektor dimensi seperti dibawah ini:

$$\text{Citra } X = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{bmatrix}$$

$$\text{Weight}(W) = \begin{bmatrix} 0.5 & -0.2 & 0.8 & -0.3 & 0.11 & -0.4 & 0.14 & -0.5 & 0.17 \end{bmatrix}$$

$$\text{Bias}(b) = \begin{bmatrix} 0.1 \end{bmatrix}$$

Berikut ini contoh ilustrasi pengaplikasi dalam melakukan proses *linear projection of flattened patches* seperti pada gambar dibawah ini.



Gambar 3. 7 Proses Linear Projection of flattened Patches

Setelah melihat ilustrasi diatas maka dapat dihitung *linear projection* dengan rumus:

$$z = (w * x) + b$$

$$z = (0.5 * 1) + (-0.2 * 2) + (0.8 * 3) + (-0.3 * 4) + (-0.6 * 6) + (0.1 * 7) + (0.4 * 8) + (-0.5 * 9) + 0.1$$

$$z = (0.5) + (-0.4) + (2.4) + (-1.2) + (-3.6) + (0.7) + (3.2) + (-4.5) + 0.1$$

$$z = 0.7$$

c) *Embedded patch*

Proses ini berfungsi untuk membantuk melakukan suatu ekstraksi fitur-fitur dari citra dan memberikan suatu informasi terhadap posisi citra secara *relative* pada setiap *patch* citra dataset. Maka dari itu proses perhitungan *embedded patch* dapat dihitung. Proses perhitungan *embedded patch* menggunakan suatu *weight matrix* dan *bias* beserta nilai input *vector* dari *patch* maka dari itu nilai inputnya yaitu:

$$W = \begin{bmatrix} 0.5 & -0.2 & 0.8 \\ -0.3 & 0.7 & -0.6 \\ 0.1 & 0.4 & -0.5 \end{bmatrix}$$

$$b = \begin{bmatrix} 0.1 \\ -0.2 \\ 0.3 \end{bmatrix}$$

$$\text{vektor}(x) = \begin{bmatrix} 10 & 20 & 15 \end{bmatrix}$$

Rumus:

$$E_{patch} = X.W + b$$

$$E_{patch} = \begin{bmatrix} 10 & 20 & 15 \end{bmatrix} \cdot \begin{bmatrix} 0.5 & -0.2 & 0.8 \\ 0.3 & 0.7 & -0.6 \\ 0.1 & 0.4 & -0.5 \end{bmatrix} + \begin{bmatrix} 0.1 \\ -0.2 \\ 0.3 \end{bmatrix}$$

Selanjutnya perhitungan ini dapat dilakukan dengan sederhana sebagai berikut:

$$\begin{aligned} e_1 &= (10 \times 0.5) + (20 \times (-0.3)) + (15 \times 0.1) + 0.1 \\ &= 5 - 6 + 1.5 + 0.1 = 0.6 \end{aligned}$$

$$\begin{aligned} e_2 &= (10 \times 0.2) + (20 \times 0.7) + (15 \times 0.4) + 0.2 \\ &= -2 + 14 + 6 - 0. = 17.8 \end{aligned}$$

$$\begin{aligned} e_3 &= (10 \times 0.8) + (20 \times 0.6) + (15 \times (-0.5)) + 0.3 \\ &= 8 - 12 - 7.5 + 0.3 = -11.2 \end{aligned}$$

maka hasil yang diperoleh dari proses *embedded patch* (E_{patch}) adalah 0.6, 17.8, -11.2 pada *patch embedded*.

d) *Norm (normalisasi)*

Proses *norm* pada *Vision Transformer* adalah proses yang dilakukan setelah melewati proses perhitungan dari nilai *embedding patch*. Fungsi dari *norm* adalah untuk meningkatkan stabilitas dan kecepatan konvergensi model selama pelatihan.

Rumus :

$$x_{normalized} = \frac{x}{||x||_2}$$

Keterangan:

$||x||_2$: *vektor* dari x

Maka nilai input untuk dapat dihitung normalisasi tersebut yaitu sebagai berikut :

$$\text{nilai } embedded \text{ patch} = [0.6, 17.8, -11.2]$$

Setelah nilai input telah diinput maka, dilanjutkan untuk menghitung nilai normalisasi pada vektor E_{patch} :

$$\|E_{patch}\|_2 = \sqrt{0.6^2 + 17.8^2 + (-11.2)^2}$$

$$\|E_{patch}\|_2 = \sqrt{0.36 + 316.84 + 125.44}$$

$$\|E_{patch}\|_2 = \sqrt{442.64}$$

$$= 21.03$$

Selanjutnya hasil data tersebut dinormalisasi dari E_{patch} dengan Rumus:

$$E_{patch-normalized} = \frac{E_{patch}}{\|E_{patch}\|_2}$$

$$E_{patch-normalized} = \frac{[0.6, 17.8, -11.2]}{\|21.03\|_2}$$

$$E_{patch-normalized} = [0.028, 0.846, -0.532]$$

Jadi setelah proses normalisasi, vektor embedding dari patch yang telah dinormalisasikan dan dapat digunakan sebagai input untuk tahap selanjutnya dalam model *Vision Transformer* dimana perhitungan ini dapat membantu meningkatkan stabilitas dan *konvergensi* model selama pelatihan.

e) *MSA + normalisasi*

Proses ini berfungsi untuk memahami interaksi antara setiap elemen dalam suatu urutan lalu setelah proses *MSA* dilakukan maka hasil dari proses tersebut maka dilakukan *normalisasi* lagi agar hasil perhitungan tersebut dapat meningkatkan stabilitas dan konvergensi selama pelatihan model dilakukan. Berikut ini perhitungan dari *MSA* dibawah ini:

$$X = 0.6, 17.8, -11.2$$

Lalu untuk mempermudah prosesnya maka dapat disederhanakan dengan satu head yang memiliki parameter *MS*.

Parameter *WQ* (*query*), *WK* (*Key*), dan *WV* (*value*) :

$$WQ = \begin{bmatrix} 0.1 & -0.2 & 0.3 \\ 0.2 & -0.3 & 0.1 \\ -0.1 & 0.1 & 0.2 \end{bmatrix}$$

$$WK = \begin{bmatrix} 0.2 & 0.1 & -0.1 \\ -0.3 & -0.2 & 0.3 \\ 0.1 & 0.4 & -0.2 \end{bmatrix}$$

$$WV = \begin{bmatrix} 0.3 & 0.2 & -0.1 \\ 0.1 & -0.1 & 0.2 \\ -0.2 & 0.3 & 0.4 \end{bmatrix}$$

setelah nilai input dari setiap parameter di inputkan maka dapat dihitung dengan Rumus:

$$Q = X \cdot WQ \quad V = X \cdot WV \quad K = XWK$$

$$Attention_Score = softmax \left(\frac{Q \cdot K^T}{\sqrt{d_k}} \right)$$

$$Attention_Output = Attention_Score \cdot V$$

Keterangan:

Q : Query x : nilai *embedding patch*

K : Kunci V : Value

$$\bullet Q = [0.6, 17.8, -17.2] \cdot \begin{bmatrix} 0.1 & -0.2 & 0.3 \\ 0.2 & -0.3 & 0.1 \\ -0.1 & 0.1 & 0.2 \end{bmatrix}$$

$$= [1.9, -1.2, 2.5]$$

$$\bullet K = [0.6, 17.8, -17.2] \cdot \begin{bmatrix} 0.2 & 0.1 & -0.1 \\ -0.3 & -0.2 & 0.3 \\ 0.1 & 0.4 & -0.2 \end{bmatrix}$$

$$= [-0.1, 4.7, -4.5]$$

$$\bullet V = [0.6, 17.8, -17.2] \cdot \begin{bmatrix} 0.3 & 0.2 & -0.1 \\ 0.1 & -0.1 & 0.2 \\ -0.2 & 0.3 & 0.4 \end{bmatrix}$$

$$= [1.1, -0.7, 2.8]$$

Setelah nilai Q , K , dan V telah diperoleh maka dapat dihitung nilai

Attention_Score:

$$\bullet \textit{Attention_Score} = \textit{softmax} \left(\frac{Q \cdot K}{\sqrt{d_k}} \right)$$

$$= \textit{softmax} \left(\frac{[1.9, -1.2, 2.5] \cdot [-0.1, 4.7, -4.5]}{\sqrt{3}} \right)$$

$$= \textit{softmax} \left(\frac{[1.9 \times 0.1 - 1.2 \times 4.7 + 2.5 \times 4.5]}{\sqrt{3}} \right)$$

$$= \textit{softmax} \left(\frac{-21.18}{\sqrt{3}} \right)$$

$$\textit{Attention_Score} = -12.26$$

Nilai Q , K , V dan *Attention_Score* telah diperoleh maka hitung nilai

Attention_Output:

$$\begin{aligned}
 \bullet \textit{Attention_Output} &= \textit{Attention_Score} \textit{V} \\
 &= -12.26 \times [1.1, -0.7, 2.8] \\
 &= [13.486, -8.582, 34.328]
 \end{aligned}$$

Dari hasil perhitungan diatas dapat disimpulkan bahwa proses perhitungan ini memberi satu komponen utama dalam blok pada setiap *elemen patch* gambar. Agar dalam setiap patch gambar tetap dalam suatu urutan yang teratur. Setelah proses perhitungan manual MSA(*Multhi-Head Self Attention*) telah diperoleh hasilnya, maka dilanjutkan untuk melakukan normalisasi pada MSA(*Multhi-Head Self Attention*) dan dari MSA-*normalisasi*. Dilanjutkan untuk menghitung nilai rata-ratanya dengan inputan *Transformer_Output*.

$$\textit{Transformer_Output} = [13.486, -8.582, 34.328].$$

1. Pertama hitung nilai rata-rata (μ):

$$(\mu) = \frac{1}{n} \sum_{i=1}^n \textit{transformer_output}_i$$

$$(\mu) = \frac{1}{3} \times (13.486 - 8.582 + 34.328)$$

$$(\mu) = \frac{0.5}{3} = 13.0773$$

Jadi hasil nilai rata-ratanya = 13,0773

Lalu hitung *deviasi standar* (σ):

$$(\sigma) = \sqrt{\frac{1}{n} \sum_{i=1}^n (\textit{transformer_output}_i - \mu)^2}$$

$$(\sigma) = \sqrt{\frac{1}{3} \times ((13.486 - 13.0773)^2 + (-8.582 - 13.0773)^2 + (34.328 - 13.0773)^2)}$$

$$(\sigma) = \sqrt{\frac{1}{3} \times ((0.4087)^2 + (-21.6593)^2 + (21.2507)^2)}$$

$$(\sigma) = \sqrt{\frac{1}{3} \times (0,0556) + (156,3750) + (150,5307)}$$

$$(\sigma) = \sqrt{\frac{1}{3} \times 306,9613}$$

$$(\sigma) = \sqrt{\frac{306,9613}{3}}$$

$$(\sigma) = \sqrt{102,3204} \approx 10,1153$$

Jadi hasil nilai *deviasi standar* adalah $(\sigma) = \sqrt{102,3204}$

2. Setelah itu hitung normalisasi *layer*

$$\text{Layer_Normalized_Output}_i = \frac{\text{transformer_output}_i - (\mu)}{\sqrt{\sigma^2 + \epsilon}} \quad \text{Nilai } \epsilon = 1e - 5 \rightarrow (\text{nilai yang umum digunakan})$$

Maka dapat dihitung

$$\begin{aligned} \bullet \text{Layer_Normalized_Output}_1 &= \frac{13.486 - 13.0773}{\sqrt{10,1153^2 + (1e-5)}} = \frac{-8.582 - 13.0773}{\sqrt{102,3204 + (1e-5)}} \\ &= \frac{13.486 - 13.0773}{10,1153 - 4} = 0,0668 \end{aligned}$$

$$\begin{aligned} \blacksquare \text{Layer_Normalized_Output}_2 &= \frac{-8.582 - 13.0773}{\sqrt{10,1153^2 + (1e-5)}} = \frac{-8.582 - 13.0773}{\sqrt{102,3204 + (1e-5)}} \\ &= \frac{-8.582 - 13.0773}{10,1153 - 4} = -3,5418 \end{aligned}$$

$$\begin{aligned} \blacksquare \text{Layer_Normalized_Output}_3 &= \frac{34.328 - 13.0773}{\sqrt{10,1153^2 + (1e-5)}} = \frac{34.328 - 13.0773}{\sqrt{102,3204 + (1e-5)}} \\ &= \frac{34.328 - 13.0773}{10,1153 - 4} = 3,4750 \end{aligned}$$

Jadi hasil layer_normalized_output adalah [0.0668, -3.5418, 3.4750].

f) *MLP (Multilayer Perceptron)*

Proses *MLP (Multilayer Perceptron)* dalam model *Vision Transformer (ViT)* adalah proses yang melakukan peningkatan kemampuan pada model untuk menangkap kompleksitas informasi secara visual melalui setiap blok transformer yang dihasilkan setelah melewati *MSA (Multi-Head Self Attention)*. Berikut ini proses perhitungan yang dilakukan *MLP (Multilayer Perceptron)* dalam model *Vision Transformer (ViT)* dengan inputannya yaitu:

$$\text{MLP_Input} = [0.5, -0.3, 0.8].$$

Lapisan memiliki sembilan lapisan yaitu FC1, FC2, FC3...FC9. Namun disini perhitungan manual tersebut hanya menghitung dua lapisan saja dengan parameter sebagai berikut:

1. Lapisan Linear Pertama (FC1):

$$\text{WFC1} = \begin{bmatrix} 0.5 & -0.2 & 0.8 \\ -0.3 & 0.7 & -0.6 \\ 0.1 & 0.4 & -0.5 \end{bmatrix}$$

$$\text{Bfc1} = \begin{bmatrix} 0.1 \\ -0.2 \\ 0.3 \end{bmatrix}$$

2. Lapisan Linear Kedua (FC2):

$$\text{WFC2} = \begin{bmatrix} 0.3 & -0.2 & 0.4 \\ -0.3 & 0.6 & -0.5 \\ 0.1 & 0.2 & -0.4 \end{bmatrix}$$

$$\text{Bfc2} = \begin{bmatrix} 0.2 \\ -0.1 \\ 0.3 \end{bmatrix}$$

hitung output MLP:

$$\bullet FC1_ouput = MLP_Input \cdot W_{FC1} + b_{fc1}$$

$$FC1_ouput = [0.5, -0.3, 0.8] \cdot \begin{bmatrix} 0.5 & -0.2 & 0.8 \\ -0.3 & 0.7 & -0.6 \\ 0.1 & 0.4 & -0.5 \end{bmatrix} + \begin{bmatrix} 0.1 \\ -0.2 \\ 0.3 \end{bmatrix}$$

$$= [0.5, -0.3, 0.8] \cdot \begin{bmatrix} 0.6 & 0.1 & 0.9 \\ -0.5 & 0.5 & 0.8 \\ 0.4 & 0.7 & -0.2 \end{bmatrix}$$

$$FC1_ouput = \begin{bmatrix} 0.3 & -0.03 & 0.72 \\ -0.25 & 0.15 & -0.64 \\ 0.2 & 0.21 & -0.16 \end{bmatrix}$$

$$\bullet \text{Fungsi ReLu} :$$

$$\text{ReLu_ouput} = \max(FC1_ouput, 0)$$

$$\text{ReLu_ouput} = \max. \begin{bmatrix} 0.3 & -0.03 & 0.72 \\ 0.25 & 0.15 & -0.64 \\ 0.2 & 0.21 & -0.16 \end{bmatrix}, 0$$

$$\text{ReLu_ouput} = [0.2, 0.3, 0.15]$$

$$\bullet \text{Lapisan linear kedua} :$$

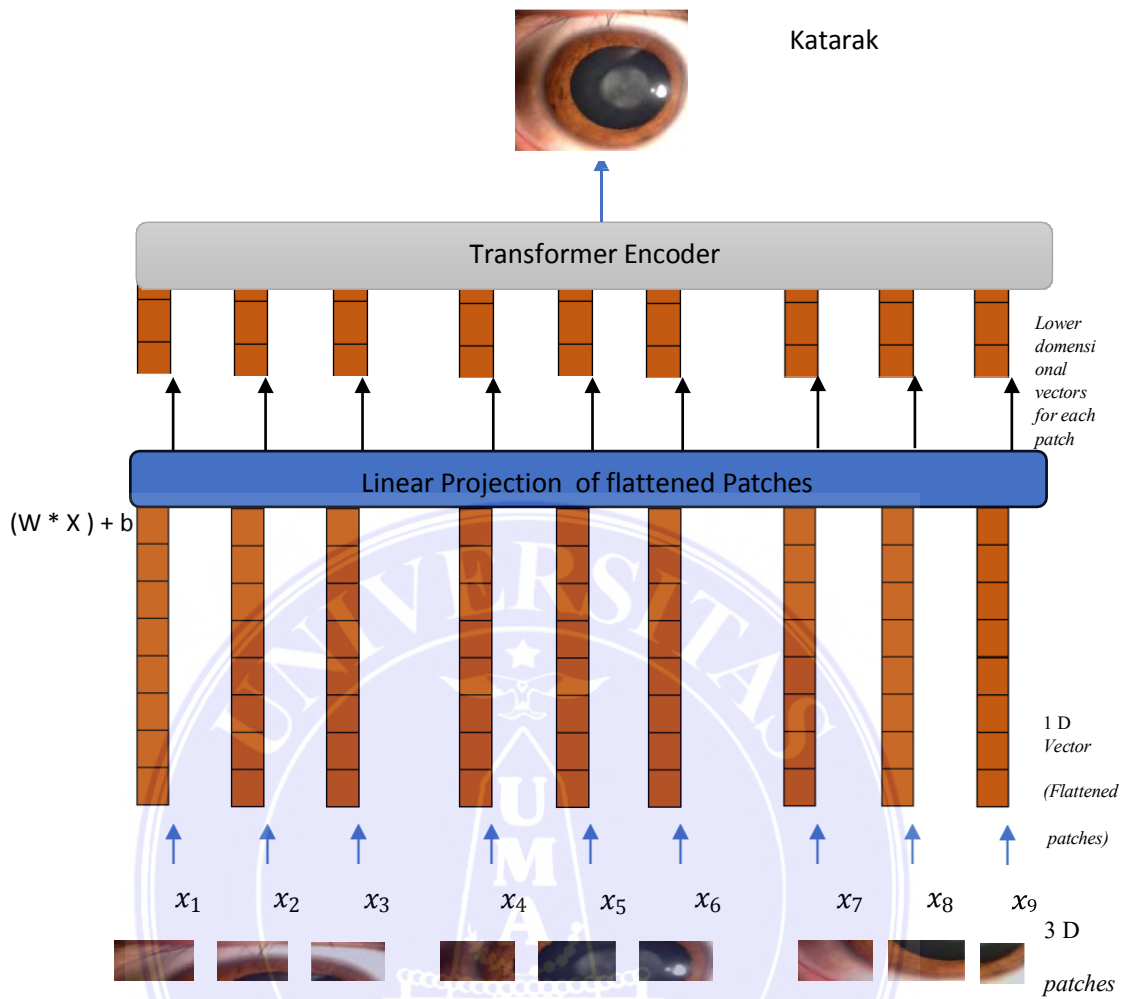
$$FC2_ouput = \text{ReLu_ouput} \cdot W_{FC2} + b_{fc2}$$

$$FC2_ouput = [0.2, 0.3, 0.15] \cdot \begin{bmatrix} 0.3 & -0.1 & 0.4 \\ -0.3 & 0.6 & -0.2 \\ 0.1 & 0.4 & -0.5 \end{bmatrix} + \begin{bmatrix} 0.2 \\ -0.1 \\ 0.3 \end{bmatrix}$$

$$FC2_ouput = [0.1, 0.9, 0.03]$$

Jadi output MLP dari dua layer adalah $[0.1, 0.9, 0.03]$ dari 9 patch gambar

proses ini akan dilakukan berulang-ulang sampai ke proses sembilan *patch* citra serta memperoleh hasil output MLP (*Multilayer Perceptron*). Kemudian setelah proses MLP telah selesai maka hasil classification akan memperoleh hasil dari setiap label kelas yang diberikan.



Gambar 3. 8 Simulasi classification Vision Transformer

proses ini akan dilakukan berulang-ulang sampai ke proses patch 9 citra serta memperoleh hasil output MLP (*Multilayer Perceptron*). Kemudian setelah proses MLP telah selesai maka hasil classification akan memperoleh hasil dari setiap label kelas yang diberikan.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang dapat diambil dari hasil analisis dan pengujian dalam melakukan klasifikasi penyakit mata katarak pada manusia dengan menggunakan model arsitektur *Vision Transformer (ViT-B16)*, yaitu sebagai berikut

1. Berdasarkan hasil *training* pada enam model skenario yang diujikan pada arsitektur *ViT-B16*, diperoleh *validation accuracy* pada model skenario ke-1 sebesar 84.82% dengan nilai *loss* 0.3480, model skenario ke-2 sebesar 89.29% dengan nilai *loss* 0.2279, model skenario ke-3 sebesar 89.29% dengan nilai *loss* 0.2552, model skenario ke-4 sebesar 90.18% dengan nilai *loss* 0.2055, model skenario ke-5 sebesar 92.86% dengan nilai *loss* 0.2516, dan model skenario ke-6 sebesar 92.86% dengan nilai *loss* 0.2504.
2. Berdasarkan hasil pengujian menggunakan *data testing* pada enam skenario *model* yang diujikan, diperoleh hasil terbaik pada *model* skenario ke-6 dengan tingkat akurasi 92.86% dengan nilai *loss* 0.2504 menggunakan *hyperparameter batch size* 32, *optimizer RMSprop*, dan *learning rate* 0.001.

5.2 Saran

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

1. Penelitian selanjutnya dapat menambahkan dataset lebih banyak dengan metode ini.

2. Penelitian selanjutnya dapat menggunakan *hyperparameter* yang lebih banyak lagi contohnya dari *learning rate*, *optimizer*, *batch size*.
3. Penelitian selanjutnya dapat mengembangkan (*deploy*) model kedalam aplikasi siap pakai dalam berbasis *website* atau *mobile*.



DAFTAR PUSTAKA

- Agustina, N. (2022). *view_artikel/1697/mata-adalah-jendela-dunia*. Retrieved from https://yankes.kemkes.go.id/view_artikel/1697/mata-adalah-jendela-dunia#
- Al-Faruq, U. A., & Fudholi, D. H. (n.d.). *Implementasi Arsitektur Transformer pada Image Captioning dengan Bahasa Indonesia*. 2020.
- Ali, M. S., & Islam, M. K. (2023). A hyper-tuned Vision Transformer model with Explainable AI for Eye disease detection and classification from medical images.
- Athar, M. W. (2022). *Convolutional Vision Transformer (CvT) untuk Peningkatan Kualitas Citra Hasil Inverse Halftoning*. UPT Perpustakaan Universitas Sebelas Maret.
- Bazi, Y., Basmal, L., Rahhal, M. M., Dayil, R. A., & Al-Ajian, N. (2021). *Vision Transformers untuk Klasifikasi Citra Penginderaan Jauh* (Vol. 13). MDPI: Penginderaan Jauh.
- Bu'ulolo, G. J., Jacobus, A., & Kambey, F. D. (2021). *Identification of Cataract Eye Disease Using Convolutional Neural Network* (Vol. 16). Jurnal Teknik Informatika.
- Chen, J.-N., Sun, S., He, J., Torr, P., Yuille, A., & Bai, S. (2021). *TransMix: Menghadiri Mix untuk Vision Transformers*. Universitas Johns Hopkins.

Dwi Putri, D. L. (2022). Klasifikasi Citra Retina Dalam Menentukan Penyakit Glaukoma Menggunakan Arsitektur Vision Transformer. *Sains Bidang Studi Matematika*, 25.

Figo, J. A., Yudistra, N., & Widodo, A. W. (2023). Deteksi Covid-19 dari Citra X-ray menggunakan Vision Transformer. *Pengembangan Teknologi Informasi dan Ilmu Komputer*, 1116-1125.

Firdaus, D. H., Imran, B., Bakti, L. D., & Suryadi, E. (2022). *Klasifikasi Penyakit Katarak Pada Mata Menggunakan Metode Convolutional Neural Network (Cnn) Berbasis Web* (Vol. 1). Jurnal Kecerdasan Buatan dan Teknologi Informasi (JKBTI).

Gifran, N. A., Magdalena, R., & Faudah, Y. N. (2019). *Classification Of Cataract Using Discrete Wavelet Transform And Support Vector Machine* (Vol. 6). E-Proceeding of Engineering.

Halbouni, A., Gunawan, T. S., & Habaebi, M. H. (2022). Machine Learning and Deep Learning Approaches for CyberSecurity: A Review. *IEEE Access*.

Karimah, K., Anas, K., & Arsyad, M. (2023). *Hubungan Katarak Dengan Diabetes Melitus Di Poliklinik Mata Rs Yarsi Periode Tahun 2021-2022 Dan Tinjauannya Menurut Pandangan Islam* (Vol. 3). Universitas YARSI: Cerdika: Jurnal Ilmiah Indonesia.

Kurnia, F., Ningsi, S. W., Monalisa, S., & Fahmi, I. (2019). *Prediksi Penyakit Mata Katarak Dan Non Katarak Dengan Menggunakan Metode K-Nearest Neighbor*. Politeknik Negeri Banjarmasin: Prosiding SNRT (Seminar Nasional Riset Terapan).

- Labib, M. D., Hadi, S., Widayaka, P. D., & Paradisa, I. S. (2022). *Convolutional Neural Network for Cataract Maturity Classification Based on LeNet*. e-ISSN Jurnal Varian.
- Mahardika, A. N., Widodo, A. W., & Rahman, M. A. (2019). *Diagnosis Penyakit Mata menggunakan Metode Improved K-Nearest Neighbor* (Vol. 3). Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer.
- Marpaung, F. (2023). *Klasifikasi Daun Teh Siap Panen Menggunakan Cnn*. Skripsi, 14-99.
- Nayoan, R. A. (2022). *Pemahaman Visual di Dalam Ruangan dengan Image Captioning berbasis Transformer*.
- Nurhalizah, R., Minarno, A. M., & Wicaksono, G. W. (2022). *Klasifikasi Penyakit Katarak Pada Mata Manusia Menggunakan Metode Convolutional Neural Network* (Vol. 4).
- Simanjuntak, R. B., Fu'adah, Y., Magdalena, R., Saidah, S., & Wiratama, A. B. (2022). *Cataract Classification Based on Fundus Images Using Convolutional Neural Network*. Jurnal Internasional Pada Visualisasi Informatika.
- Sirajudeen, A., Balasubramaniam, A., & Karthikeyan, S. (2022). *Novel angular binary pattern (NABP) and kernel based convolutional neural networks classifier for cataract detection* (Vol. 81). ACM: Multimedia Tools and Applications.

Suwanda, A. E., & Juniati, D. (2022). Klasifikasi Penyakit Mata Berdasarkan Citra FundusRetina Menggunakan Dimensi Fraktal Box CountingDan Fuzzy K-Means. *Jurnal Penelitian Matematika dan Pendidikan Matematika*, 10.

Wiguna, G. A., Fardela, R., & Selly, J. B. (2019). *Klasifikasi Tingkat Maturitas Katarak Berbasis Citra Digital Berdasarkan Jangkauan (Range) Nilai Histogram*



LAMPIRAN

1. Lampiran Source Code

Pengujian Data Pada *Google Colaboraty*

```

# Mount drive
from google.colab import drive
drive.mount('/content/drive')
# Define directory dataset
DATASET_DIR = "/content/drive/MyDrive/My Dataset/DATASET-
KATARAK/"
TRAIN_DIR = '/content/drive/MyDrive/My Dataset/DATASET-
KATARAK/TRAIN/'
VAL_DIR = '/content/drive/MyDrive/My Dataset/DATASET-
KATARAK/VAL/'
TEST_DIR = '/content/drive/MyDrive/My Dataset/DATASET-
KATARAK/TEST/'
# Get image directory
import glob

TRAIN_IMAGES = glob.glob(f"{TRAIN_DIR}**/*.png")
VAL_IMAGES = glob.glob(f"{VAL_DIR}**/*.png")
TEST_IMAGES = glob.glob(f"{TEST_DIR}**/*.png")
# Get dataset (training, validation & testing) sizes
TRAIN_SIZE = len(TRAIN_IMAGES)
VAL_SIZE = len(VAL_IMAGES)
TEST_SIZE = len(TEST_IMAGES)

# Get dataset size
TOTAL_DATASET = TRAIN_SIZE + VAL_SIZE + TEST_SIZE

# View samples counts
print(f'Sample data training:\t\t{TRAIN_SIZE}')
print(f'Sample data validation:\t\t{VAL_SIZE}')
print(f'Sample data testing:\t\t{TEST_SIZE}')
print('=====')
print(f'Total dataset:\t\t\t{TOTAL_DATASET}')
# Configuration parameter
class CONFIG:
    EPOCHS_25 = 25
    EPOCHS_50 = 50
    BATCH_SIZE = 32
    SEED = 42
    TF_SEED = 768
    HEIGHT = 224
    WIDTH = 224

```

```
CHANNELS = 3
IMAGE_SIZE = (224, 224, 3)
# Create pandas dataframes for directory and labels
import pandas as pd

# Generate labels
def generate_labels(image_paths):
    return [_ .split('/')[ -2: ] [0] for _ in image_paths]

# Create dataframe
def build_df(image_paths, labels):
    df = pd.DataFrame({
        'image_path': image_paths,
        'label': generate_labels(labels)
    })

    # Shuffle and return dataframes
    return df.sample(frac=1,
random_state=CONFIG.SEED).reset_index(drop=True)
# Build the dataframes
TRAIN_DF = build_df(TRAIN_IMAGES,
generate_labels(TRAIN_IMAGES))
VAL_DF = build_df(VAL_IMAGES, generate_labels(VAL_IMAGES))
TEST_DF = build_df(TEST_IMAGES, generate_labels(TEST_IMAGES))
# View 3 sample data training
TRAIN_DF.head(3)
# View 3 sample data validation
VAL_DF.head(3)
# View 3 sample data testing
TEST_DF.head(3)
# Label encode image labels
from sklearn.preprocessing import LabelEncoder

# Generate label encoder
LABEL_ENCODER = LabelEncoder()

# Label encode image labels data training
TRAIN_DF['label_encoded'] =
LABEL_ENCODER.fit_transform(TRAIN_DF.label)

# View first 3 samples
TRAIN_DF.head(3)
# Label encode image labels data validation
VAL_DF['label_encoded'] =
LABEL_ENCODER.fit_transform(VAL_DF.label)

# View first 3 samples
```

```

VAL_DF.head(3)
# Label encode image labels data testing
TEST_DF['label_encoded'] =
LABEL_ENCODER.fit_transform(TEST_DF.label)

# View first 3 samples
TEST_DF.head(3)
# Get class names and number of classes from label_encoder
NUM_CLASSES = len(LABEL_ENCODER.classes_)
CLASS_NAMES = LABEL_ENCODER.classes_

print(f'Number of classes: {NUM_CLASSES}')
print(f'Classes: {CLASS_NAMES}')
# Load & view random sample image
import tensorflow as tf

def _load(image_path):
    # Read and decode an image file to a uint8 tensor
    IMAGE = tf.io.read_file(image_path)
    IMAGE = tf.io.decode_jpeg(IMAGE, channels=3)

    # Resize image
    IMAGE = tf.image.resize(IMAGE, [CONFIG.HEIGHT,
CONFIG.WIDTH],
method=tf.image.ResizeMethod.LANCZ
OS3)

    # Convert image dtype to float32 & normalize
    IMAGE = tf.cast(IMAGE, tf.float32)/255.0
    return IMAGE

def view_sample(image, label, color_map='rgb', fig_size=(8,
10)):
    plt.figure(figsize=fig_size)

    if color_map=='rgb':
        plt.imshow(image)
    else:
        plt.imshow(tf.image.rgb_to_grayscale(image),
cmap=color_map)

    plt.title(f'Label: {label}', fontsize=16)
    return

# View multiple random samples of training dataset
import random
import matplotlib.pyplot as plt

```



```

def view_multiple_samples(df, sample_loader, count=10,
color_map='rgb', fig_size=(14, 10)):
    ROWS = count//2
    if count%5 > 0:
        ROWS +=1

    IDX = random.sample(df.index.to_list(), count)
    FIG = plt.figure(figsize=fig_size)

    for column, _ in enumerate(IDX):
        plt.subplot(ROWS, 5, column+1)
        plt.title(f'Label : {df.label[_]}', color='darkgreen',
fontsize=12, pad=10)

        if color_map=='rgb':
            plt.imshow(sample_loader(df.image_path[_]))
        else:
            plt.imshow(tf.image.rgb_to_grayscale(sample_loader
(df.image_path[_])), cmap=color_map)

    return

view_multiple_samples(TRAIN_DF, _load,
count=10, color_map='gray',
fig_size=(20, 20))
# Image data augmentation layer
import tensorflow as tf
from tensorflow.keras import Sequential, layers

augmentation_layer = Sequential([
    layers.RandomRotation(factor=(-0.25, 0.2501)),
    layers.RandomFlip(mode='horizontal_and_vertical',
seed=CONFIG.TF_SEED),
    # layers.RandomZoom(height_factor=(-0.1, 0.1),
width_factor=(-0.1, 0.1), seed=CONFIG.TF_SEED),
], name='augmentation_layer')
# Select random sample from train_df
import random

idx = random.sample(TRAIN_DF.index.to_list(), 3)[0]

# Load the random sample and label
sample_image, sample_label = _load(TRAIN_DF.image_path[idx]),
TRAIN_DF.label[idx]
# View sample of result augmentation
import matplotlib.pyplot as plt

```

```

image = sample_image
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(8, 15))

# Set the spacing between subplots
fig.tight_layout(pad=6.0)

# View original image
ax1.set_title('Original Image', fontsize=12, color='red',
pad=10)
ax1.imshow(image);

# View augmented image
ax2.set_title('Augmented Image', fontsize=12,
color='darkgreen', pad=10)
ax2.imshow(augmentation_layer(image));
# Create input data pipeline
def encode_labels(labels, encode_depth=2):
    return tf.one_hot(labels, depth=encode_depth).numpy()

def create_pipeline(df, load_function, augment=False,
batch_size=32, shuffle=False, cache=None, prefetch=False):
    # Get image paths and labels from dataframe
    IMAGE_PATHS = df.image_path
    IMAGE_LABELS = encode_labels(df.label_encoded)
    AUTOTUNE = tf.data.AUTOTUNE

    # Create dataset with raw data from dataframe
    DS = tf.data.Dataset.from_tensor_slices((IMAGE_PATHS,
IMAGE_LABELS))

    # Map augmentation layer and load function to dataset
inputs if augment is True
    # Else map only the load function
if augment:
        DS = DS.map(lambda x, y:
(augmentation_layer(load_function(x)), y),
num_parallel_calls=AUTOTUNE)
    else:
        DS = DS.map(lambda x, y: (load_function(x), y),
num_parallel_calls=AUTOTUNE)

    # Apply shuffling based on condition
if shuffle:
        DS = DS.shuffle(buffer_size=1000)

    # Apply batching
DS = DS.batch(batch_size)

```

```

# Apply caching based on condition
# Note: Use cache in memory (cache='') if the data is
small enough to fit in memory!!!
if cache != None:
    DS = DS.cache(cache)

# Apply prefetching based on condition
# Note: This will result in memory trade-offs
if prefetch:
    DS = DS.prefetch(buffer_size=AUTOTUNE)

# Return the dataset
return DS

# Generate Train input pipeline
TRAIN_DS = create_pipeline(TRAIN_DF, _load, augment=True,
                           batch_size=CONFIG.BATCH_SIZE,
                           shuffle=False, prefetch=True)

# Generate Validation input pipeline
VAL_DS = create_pipeline(VAL_DF, _load,
                          batch_size=CONFIG.BATCH_SIZE,
                          shuffle=False, prefetch=False)

# Generate Test input pipeline
TEST_DS = create_pipeline(TEST_DF, _load,
                           batch_size=CONFIG.BATCH_SIZE,
                           shuffle=False, prefetch=False)

# View string representation of datasets
print('=====')
print('Train Input Data Pipeline:\n\n', TRAIN_DS)
print('=====')
print('Validation Input Data Pipeline:\n\n', VAL_DS)
print('=====')
print('Test Input Data Pipeline:\n\n', TEST_DS)
print('=====')

# View sample of image patches
!pip install -q patchify
from patchify import patchify # Library to extract patches
import cv2
import numpy as np
from matplotlib import pyplot as plt

image_size = 224 # Default image size of ViT-B16
PATCH_SIZE = 16 # Default patch size of ViT-B16
NUM_PATCHES= (image_size // PATCH_SIZE) ** 2

```

```

image_to_patches = cv2.imread('/content/drive/MyDrive/My
Dataset/DATASET-KATARAK/TRAIN/normal/normal002.png', 0)

# Split the image into small images of shape [16, 16]
patches = patchify(image_to_patches, (PATCH_SIZE, PATCH_SIZE),
step=PATCH_SIZE) # Step=16 for 224 patches means no overlap

print(f"Image size\t\t: {image_size} X {image_size}")
print(f"Patch size\t\t: {PATCH_SIZE} X {PATCH_SIZE}")
print(f"Number of patches\t: {NUM_PATCHES}")
print(f"=====\n")

# Show image ori
# plt.figure(figsize=(5, 5))
# plt.axis('off')
# plt.imshow(image_to_patches)

# Show image patches
plt.figure(figsize=(5, 5))
square = 14 # Default size of ViT-B16
ix = 1
for i in range(square):
    for j in range(square):
        # specify subplot and turn of axis
        ax = plt.subplot(square, square, ix)
        ax.set_xticks([])
        ax.set_yticks([])
        # plot
        plt.imshow(patches[i, j, :], cmap="gray")
        ix += 1
plt.show()
# install library vit-keras
!pip install -q vit-keras
# install library tensorflow_addons
!pip install tensorflow_addons
# Get Vision Transformer model
from vit_keras import vit

# Download the model
base_vit_b16_model = vit.vit_b16(
    image_size=224,
    activation='sigmoid',
    pretrained=True,
    include_top=False, # tidak mengikutsertakan fully-
connected layer (karena akan didefinisikan secara terpisah dan
menyesuaikan

```

```

                                # dengan dataset yang dipakai),
dengan kata lain yang diambil hanya feature extractor layernya
    pretrained_top=False,
    classes=2
)

# Freeze model layers for inference-mode only
for layer in base_vit_b16_model.layers:
    layer.trainable = False
# Summary & plot base model ViTs_B16
from keras.utils import plot_model

# Summary model
base_vit_b16_model.summary()

# Plot model
plot_model(base_vit_b16_model, to_file='base-model-
vitb16.png', show_shapes=True, show_layer_names=False,
rankdir='TB', expand_nested=False, dpi=80)
# Define Vision Transformer (ViTs-B16) Model
def vit_b16_model():

    INITIALIZER = tf.keras.initializers.GlorotNormal()

    vit_b16_sequential = Sequential([
        # Feature Extractor Layer
        layers.Input(shape=CONFIG.IMAGE_SIZE, dtype=tf.
float32, name='input_image'),
        base_vit_b16_model, # Feature Extractor pakai pre-
trained model (ViTs-B16)

        # Fully Connected Layer
        layers.Dropout(0.5), # 0.2
        layers.Dense(128, activation='relu',
kernel_initializer=INITIALIZER),
        layers.Dense(2, dtype=tf.float32,
activation='sigmoid', kernel_initializer=INITIALIZER)
    ], name='ViTs_B16_Sequential_Model')

    return vit_b16_sequential
# Summary & plot base model ViTs_B16
from tensorflow.keras.utils import plot_model

# Generate model
model_vit_b16 = vit_b16_model()

# Summary model

```



```

model_vit_b16.summary()

# Plot model
plot_model(model_vit_b16, to_file='my-model-vitb16.png',
show_shapes=True, show_layer_names=False, rankdir='TB',
expand_nested=False, dpi=80)
# Define Early Stopping Callback
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras import metrics

early_stopping_callback = EarlyStopping(
    monitor='val_auc',
    mode='max',
    patience=10,
    verbose=1,
    min_delta=0.0001,
    restore_best_weights=True)

# Define Callbacks and Metrics lists
CALLBACKS = [early_stopping_callback]
METRICS = [metrics.AUC(name='auc'), 'accuracy']
# Define train model
def train_model(model, num_epochs, callbacks_list,
tf_train_data,
                tf_valid_data=None, shuffling=False):

    model_history = {}

    if tf_valid_data != None:
        model_history = model.fit(tf_train_data,
                                epochs=num_epochs,
                                validation_data=tf_valid_data,
                                validation_steps=int(len(tf_
valid_data)),
                                callbacks=callbacks_list,
                                shuffle=shuffling)

    if tf_valid_data == None:
        model_history = model.fit(tf_train_data,
                                epochs=num_epochs,
                                callbacks=callbacks_list,
                                shuffle=shuffling)

    return model_history

# Define model skenario 1
model_skenario1 = model_vit_b16
# Train ViTs Model Skenario 1 (opt=Adam; lr=0.009; epochs=25)

```

```
from tensorflow.keras.optimizers import Adam

tf.random.set_seed(CONFIG.SEED)
OPT = Adam(learning_rate=0.009)

# Compile the model
model_skenario1.compile(
    loss=tf.keras.losses.BinaryCrossentropy(),
    optimizer=OPT,
    metrics=METRICS
)

# Train the model
print(f'Training {model_skenario1.name}.')
print(f'Train on {len(TRAIN_DF)} samples, validate on
{len(VAL_DF)} samples.')
print('-----\n')

hist_model_skenario1 = train_model(
    model_skenario1,
    CONFIG.EPOCHS_25,
    CALLBACKS,
    TRAIN_DS, VAL_DS,
    shuffling=False
)
# Evaluate model skenario 1
eval_model_skenario1 = model_skenario1.evaluate(TEST_DS)
# Save model skenario 1
model_skenario1.save('/content/drive/MyDrive/My
Model/vit_b16_epoch25-optAdam-lr0.009.h5')
# Define model skenario 2
model_skenario2 = model_vit_b16
# Train ViTs Model Skenario 2 (opt=RMSprop; lr=0.009;
epochs=25)
from tensorflow.keras.optimizers import RMSprop

tf.random.set_seed(CONFIG.SEED)
OPT = RMSprop(learning_rate=0.009)

# Compile the model
model_skenario2.compile(
    loss=tf.keras.losses.BinaryCrossentropy(),
    optimizer=OPT,
    metrics=METRICS
)

# Train the model
```

```
print(f'Training {model_skenario2.name}.')
print(f'Train on {len(TRAIN_DF)} samples, validate on
{len(VAL_DF)} samples.')
print('-----\n')

hist_model_skenario2 = train_model(
    model_skenario2,
    CONFIG.EPOCHS_25,
    CALLBACKS,
    TRAIN_DS, VAL_DS,
    shuffling=False
)
# Evaluate model skenario 2
eval_model_skenario2 = model_skenario2.evaluate(TEST_DS)
# Save model skenario 2
model_skenario2.save('/content/drive/MyDrive/My
Model/vit_b16_epoch25-optRmsProp-lr0.009.h5')
# Define model skenario 3
model_skenario3 = model_vit_b16
# Train ViTs Model Skenario 3 (opt=SGD; lr=0.009; epochs=25)
from tensorflow.keras.optimizers import SGD

tf.random.set_seed(CONFIG.SEED)
OPT = SGD(learning_rate=0.009)

# Compile the model
model_skenario3.compile(
    loss=tf.keras.losses.BinaryCrossentropy(),
    optimizer=OPT,
    metrics=METRICS
)

# Train the model
print(f'Training {model_skenario3.name}.')
print(f'Train on {len(TRAIN_DF)} samples, validate on
{len(VAL_DF)} samples.')
print('-----\n')

hist_model_skenario3 = train_model(
    model_skenario3,
    CONFIG.EPOCHS_25,
    CALLBACKS,
    TRAIN_DS, VAL_DS,
    shuffling=False
)
# Evaluate model skenario 3
eval_model_skenario3 = model_skenario3.evaluate(TEST_DS)
```

```
# Save model skenario 3
model_skenario3.save('/content/drive/MyDrive/My
Model/vit_b16_epoch25-optSGD-lr0.009.h5')
# Define model skenario 4
model_skenario4 = model_vit_b16
# Train ViTs Model Skenario 4 (opt=Adam; lr=0.001; epochs=30)
from tensorflow.keras.optimizers import Adam

tf.random.set_seed(CONFIG.SEED)
OPT = Adam(learning_rate=0.001)

# Compile the model
model_skenario4.compile(
    loss=tf.keras.losses.BinaryCrossentropy(),
    optimizer=OPT,
    metrics=METRICS
)


# Train the model
print(f'Training {model_skenario4.name}.')
print(f'Train on {len(TRAIN_DF)} samples, validate on
{len(VAL_DF)} samples.')
print('-----\n')

hist_model_skenario4 = train_model(
    model_skenario4,
    CONFIG.EPOCHS_50,
    CALLBACKS,
    TRAIN_DS, VAL_DS,
    shuffling=False
)

# Evaluate model skenario 4
eval_model_skenario4 = model_skenario4.evaluate(TEST_DS)
# Save model skenario 4
model_skenario4.save('/content/drive/MyDrive/My
Model/vit_b16_epoch50-optAdam-lr0.001.h5')
# Define model skenario 5
model_skenario5 = model_vit_b16
# Train ViTs Model Skenario 5 (opt=RMSprop; lr=0.001;
epochs=50)
from tensorflow.keras.optimizers import RMSprop

tf.random.set_seed(CONFIG.SEED)
OPT = RMSprop(learning_rate=0.001)
```


2. Surat SK Pembimbing



 **UNIVERSITAS MEDAN AREA**
FAKULTAS TEKNIK

Kampus I : Jalan Kolonel M. Y. S. Soediro/Jalan P. O. H. Simanungkalang Nomor 1 Medan (061) 7361870, 7361180, 7384340, 7366781, Fax (061) 7366908 Medan 20223
Kampus II : Jalan Setia Budi Nomor 79 / Jalan Sei Selayu Nomor 711 A. (061) 8225602, Fax (061) 8226331 Medan 20122
Website: www.teknik.uma.ac.id E-mail: umv.medan@uma.ac.id

Nomor : 667/FT.6/01.10/IX/2023
Lamp : -
Hal : **Perubahan Judul Tugas Akhir** 1 September 2023

Yth. Pembimbing Tugas Akhir
Nurul Khairina, S. Kom, M. Kom
di
Tempat

Dengan hormat, Sehubungan dengan adanya perubahan judul tugas akhir maka perlu diterbitkan kembali SK Pembimbing Skripsi baru atas nama mahasiswa tersebut :

Nama : Sentia Ovania Purba
N P M : 198160066
Jurusan : Teknik Informatika

Maka dengan hormat kami mengharapkan kesediaan saudara :

Nurul Khairina, S. Kom, M. Kom (Sebagai Pembimbing)

Adapun Tugas Akhir Skripsi berjudul :


"Klasifikasi Jenis Penyakit Mata pada Manusia dengan menggunakan Model Arsitektur Vision Transformer".

SK Pembimbing ini berlaku selama enam bulan terhitung sejak SK ini diterbitkan. Jika proses pembimbing melebihi batas waktu yang telah ditetapkan, SK ini dapat ditinjau ulang.

Demikian kami sampaikan, atas kesediaan saudara diucapkan terima kasih.


Dr. Rahmad Syah, S. Kom, M. Kom

3. Surat Pengantar Riset

 **UNIVERSITAS MEDAN AREA**
FAKULTAS TEKNIK

Kampus I : Jalan Kolam Nomor 1 Medan Estate/Jalan PBSI Nomor 1 ☎ (061) 7366878, 7360168, 7364348, 7366781, Fax: (061) 7368998 Medan 20223
Kampus II : Jalan Setiabudi Nomor 79 / Jalan Sei Sanyu Nomor 70 A, ☎ (061) 8229602, Fax: (061) 8226331 Medan 20122
Website: www.teknik.uma.ac.id E-mail: univ_medanarea@uma.ac.id

Nomor : 669 /FT.6/01.10/IX/2023
Lamp : -
Hal : **Penelitian Dan Pengambilan Data Tugas Akhir**

1 September 2023

Yth. Pimpinan UPT Rumah Sakit Khusus Mata Provinsi Sumatera utara
Helvetia Timur, Kec. Medan Helvetia
Di
Medan

Dengan hormat,
Kami mohon kesediaan Bapak/Ibu berkenan untuk memberikan izin dan kesempatan kepada mahasiswa kami tersebut dibawah ini :


NO	NAMA	NPM	PRODI
I	Sentia Ovania Purba	198160066	Teknik Informatika

Untuk melaksanakan Penelitian dan Pengambilan Data Tugas Akhir pada perusahaan/Instansi yang Bapak/Ibu Pimpin.

Perlu kami jelaskan bahwa Pengambilan Data tersebut adalah semata-mata untuk tujuan ilmiah dan Skripsi yang merupakan salah satu syarat bagi mahasiswa tersebut untuk mengikuti ujian sarjana pada Fakultas Teknik Universitas Medan Area dan tidak untuk dipublikasikan, dengan judul penelitian :

Klasifikasi Jenis Penyakit Mata pada Manusia dengan menggunakan Model Arsitektur Vision Transformer

Atas perhatian dan kerja sama yang baik diucapkan terima kasih.


Dr. Rahmad Syah, S. Kom, M. Kom

Tembusan :
1. Ka. BAMAI
2. Mahasiswa
3. File

4. Surat Selesai Riset

**PEMERINTAH PROVINSI SUMATERA UTARA**
DINAS KESEHATAN
UPTD RUMAH SAKIT KHUSUS MATA
Jalan Kapten Sumartono No. 1 Telp: (061) 80031788 – 80031788
Fas: (061) 80031788 Medan - 20124. E-mail: uptdskm@uma.ac.id

Medan, 12 November 2023

Nomor : 800.1.11/ 443 /RSKM/XI/2023
Lampiran :
Perihal : Telah Selesai Melaksanakan Penelitian

Yth. Dekan Fakultas Teknik
Universitas Medan Area
di Tempat

Sehubungan dengan surat dari Dekan Fakultas Teknik Universitas Medan Area Nomor : 669/FT.6/01.10/IX/2023 Tanggal 01 September 2023 perihal Penelitian dan Pengambilan Data Tugas Akhir, atas nama :

NO.	NAMA	N P M	JUDUL SKRIPSI
1	Sentia Ovania Purba	198160066	Klasifikasi Penyakit Mata pada Manusia dengan Menggunakan Model Arsitektur <i>Vision Transformers</i>

Sesuai dengan hal tersebut diatas, maka dengan ini kami sampaikan bahwa mahasiswa tersebut telah selesai melakukan penelitian di UPTD Rumah Sakit Khusus Mata Provinsi Sumatera Utara.

Demikian kami sampaikan untuk dipergunakan seperlunya.

**DIREKTUR UPTD RS KHUSUS MATA
PROVINSI SUMATERA UTARA**

Dr. DEWI CHAILATY, M.Kes
PEMBINA TK. I
NIP. 19700129 200212 2 003