

**SISTEM DETEKSI PENYAKIT MATA PTERIGIUM
BERBASIS ANDROID MENGGUNAKAN METODE
CONVOLUTIONAL NEURAL NETWORK**

LAPORAN TUGAS AKHIR

Laporan ini Disusun untuk Memenuhi Salah Satu Syarat Memperoleh
Gelar Sarjana Strata 1 (S1) pada Program Studi Teknik Informatika
Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang



DISUSUN OLEH :

NUR MUHAMMAD SYAIFUDDIN

NIM 32601900026

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM SULTAN AGUNG
SEMARANG**

2022

FINAL PROJECT

***PTERIGIUM EYE DISEASE DETECTION SYSTEM BASED ON ANDROID
USING CONVOLUTIONAL NEURAL NETWORK METHOD***

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



Arranged By :

NUR MUHAMMAD SYAIFUDDIN

NIM 32601900026

***MAJORING OF INFORMATICS ENGINEERING
INDUSTRIAL TECHNOLOGY FACULTY
SULTAN AGUNG ISLAMIC UNIVERSITY
SEMARANG
FEBRUARY 2023***

LEMBAR PENGESAHAN PEMBIMBING

Laporan Tugas Akhir dengan judul **“Sistem Deteksi Penyakit Mata Pterigium Berbasis Android Menggunakan Metode Convolutional Neural Network”** ini disusun oleh :

Nama : Nur Muhammad Syaifuddin

NIM : 32601900026

Program Studi : Teknik Informatika

Telah disahkan oleh dosen pembimbing pada :

Hari :

Tanggal :

Mengesahkan,

Pembimbing I

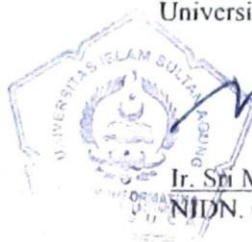
Pembimbing II

Dedy Kurniadi, S.T., M.Kom
NIDN. 0622058802

Bagus Satrio Waluyo Poetro, S.Kom., M.Cs
NIDN. 1027118801

Mengetahui,

Ketua Program Studi Teknik Informatika
Fakultas Teknologi Industri
Universitas Islam Sultan Agung



Ir. Sri Mulyono, M.Eng
NIDN. 0626066601

LEMBAR PENGESAHAN PENGUJI

Laporan tugas akhir dengan judul “Sistem Deteksi Penyakit Mata Pterigium Berbasis Android Menggunakan Metode Convolutional Neural Network” ini telah dipertahankan di depan dosen penguji Tugas Akhir pada :

Hari :

Tanggal :

TIM PENGUJI

Penguji I



Ir. Sri Mulyono, M.Eng
NIDN. 0626066601

Penguji II



Badie'ah, ST., M.Kom
NIDN. 0619018701



SURAT PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan dibawah ini :

Nama : Nur Muhammad Syaifuddin
NIM : 32601900026
Judul Tugas Akhir : Sistem Deteksi Penyakit Mata Pterigium Berbasis
Android Menggunakan Metode Convolutional
Neural Network

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

Semarang, 8 Februari 2023

Yang Menyatakan,



Nur Muhammad Syaifuddin

PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH

Saya yang bertanda tangan dibawah ini :

Nama : Nur Muhammad Syaifuddin
NIM : 32601900026
Program Studi : Teknik Informatika
Fakultas : Teknologi Industri
Alamat Asal : DK Mbatur Kidul, RT/RW 004/002, Desa Sembaturagung, Kec. Jakenan, Kab. Pati

Dengan ini menyatakan Karya Ilmiah berupa Tugas akhir dengan Judul : **Sistem Deteksi Penyakit Mata Pterigium Berbasis Android Menggunakan Metode Convolutional Neural Network**

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

Semarang, 8 Februari 2023

Yang menyatakan



Nur Muhammad syaifuddin

KATA PENGANTAR

Dengan mengucapkan syukur alhamdulillah atas kehadiran Allah SWT yang telah memberikan rahmat dan karunianya kepada penulis, sehingga dapat menyelesaikan Tugas Akhir dengan judul “Sistem Deteksi Penyakit Mata Pterigium Berbasis Android Menggunakan Metode *Convolutional Neural Network*” ini untuk memenuhi salah satu syarat menyelesaikan studi serta dalam rangka memperoleh gelar sarjana (S-1) pada Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang.

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

1. Rektor UNISSULA Bapak Prof. Dr. H. Gunarto, S.H., M.H yang mengizinkan penulis menimba ilmu di kampus ini.
2. Dekan Fakultas Teknologi Industri Ibu Dr. Novi Marlyana, S.T., M.T.
3. Dosen pembimbing I penulis Dedy Kurniadi, S.T., M.Kom yang telah meluangkan waktu dan memberi ilmu.
4. Dosen pembimbing II penulis Bagus Satrio Waluyo Poetro, S.Kom., M.Cs yang memberikan banyak nasehat dan saran.
5. Orang tua penulis yang telah mengizinkan untuk menyelesaikan laporan ini,
6. Dan kepada semua pihak yang tidak dapat saya sebutkan satu persatu.

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

Semarang, 8 Februari 2023

Nur Muhammad Syaifuddin

DAFTAR ISI

COVER	i
LEMBAR PENGESAHAN PEMBIMBING	iii
LEMBAR PENGESAHAN PENGUJI.....	iv
SURAT PERNYATAAN KEASLIAN TUGAS AKHIR	v
PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH	vi
KATA PENGANTAR.....	iii
DAFTAR ISI.....	viii
DAFTAR TABEL.....	xi
DAFTAR GAMBAR.....	xii
ABSTRAK	xiii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Perumusan Masalah.....	2
1.3 Pembatasan Masalah	2
1.4 Tujuan.....	3
1.5 Manfaat.....	3
1.6 Sistematika Penulisan.....	3
BAB II TINJAUAN PUSTAKA DAN DASAR TEORI	5
2.1 Tinjauan Pustaka	5
2.2 Dasar Teori.....	7
2.2.1 Pterigium.....	7
2.2.2 Penyebab Pterigium	8
2.2.3 <i>Machine Learning</i>	8
2.2.4 <i>Deep Learning</i>	9
2.2.5 <i>Convolutional Neural Network</i>	10
2.2.6 TensorFlow	16
2.2.7 Keras	16
2.2.8 TensorFlow Lite.....	16
BAB III METODOLOGI PENELITIAN	17
3.1 Metode Penelitian.....	17

3.1.1	<i>Data Collecting</i>	18
3.1.2	<i>Data Cleaning</i>	19
3.1.3	<i>Data Preprocessing</i>	20
3.1.4	<i>Data Processing</i>	22
3.1.5	<i>Training</i>	24
3.1.6	<i>Deployment Model</i>	25
3.1.7	Pembuatan Aplikasi Android	25
3.1.8	<i>Testing</i>	26
3.2	Analisis Kebutuhan	28
3.2.1	Tips Penggunaan Aplikasi.....	28
3.2.2	<i>Upload Citra</i>	29
3.2.3	<i>Cropping Citra</i>	29
3.2.4	Menampilkan Hasil <i>Cropping Citra</i>	29
3.2.5	Melakukan Prediksi.....	29
3.2.6	Menampilkan Hasil Prediksi.....	29
3.3	Analisis Sistem	29
3.3.1	TensorFlow	30
3.3.2	Keras	30
3.3.3	TensorFlow Lite	30
3.3.4	<i>Android Using Kotlin</i>	31
3.4	Perancangan Antarmuka.....	31
3.4.1	Halaman Beranda	32
3.4.2	Halaman Tips Penggunaan Aplikasi	32
3.4.3	Halaman <i>Upload Citra</i>	33
3.4.4	Halaman <i>Result</i>	34
BAB IV HASIL DAN ANALISIS PENELITIAN		35
4.1	Implementasi User Interface.....	35
4.1.1	Halaman Beranda	35
4.1.2	Tampilan Tips Penggunaan Aplikasi	36
4.1.3	Tampilan Unggah Foto	37
4.1.4	Tampilan Hasil Deteksi Citra.....	39
4.2	Proses Penggunaan dan Cara Kerja Sistem	40

4.2.1	Unggah Citra	40
4.2.2	<i>Crop</i> Citra.....	41
4.2.3	Konversi Citra sebagai <i>ByteBuffer</i>	41
4.2.4	Konversi <i>ByteBuffer</i> sebagai <i>TensorBuffer</i>	41
4.2.5	Klasifikasi Berdasarkan Aktivasi <i>Softmax</i>	41
4.3	Pengujian Sistem	41
4.4	Hasil dan Analisis.....	43
4.4.1	<i>Training</i> dan <i>Validation</i>	43
4.4.2	<i>Testing</i>	44
BAB V KESIMPULAN DAN SARAN		47
5.1	Kesimpulan.....	47
5.2	Saran	47
DAFTAR PUSTAKA		
LAMPIRAN		



DAFTAR TABEL

Tabel 2. 1 Perbandingan Akurasi Metode CNN Pada Deteksi Pterigium.....	6
Tabel 2. 2 Penjelasan Simbol Pada Formula Softmax	15
Tabel 3. 1 Pembagian Data Training, Validation, dan Testing.....	24
Tabel 3. 2 Konfigurasi Arsitektur CNN	24
Tabel 3. 3 Tabel Confusion Matrix	27
Tabel 3. 4 Tools yang Digunakan Dalam Pengembangan Sistem	29
Tabel 4. 1 Hasil Blackbox Testing	42
Tabel 4. 2 Plot Accuracy dan Loss.....	43
Tabel 4. 3 Model Terbaik di Setiap Konfigurasi Pada Tahap Training	44
Tabel 4. 4 Hasil Perhitungan Confusion Matrix Konfigurasi 1	45
Tabel 4. 5 Hasil Perhitungan Confusion Matrix Konfigurasi 2	45
Tabel 4. 6 Hasil Perhitungan Confusion Matrix Konfigurasi 3	45
Tabel 4. 7 Hasil Perhitungan Confusion Matrix Konfigurasi 4	45
Tabel 4. 8 Hasil Perhitungan Confusion Matrix Konfigurasi 5	45
Tabel 4. 9 Performa Testing Pada Platform Mobile	46
Tabel 4. 10 Perbandingan Akurasi Training, Validation, dan Testing.....	46

DAFTAR GAMBAR

Gambar 2. 1 Citra Mata Pterigium	7
Gambar 3. 1 Flowchart Langkah Penelitian.....	17
Gambar 3. 2 Dataset Class Pterigium	18
Gambar 3. 3 Dataset Class Normal	19
Gambar 3. 4 Dataset Sebelum Dibersihkan	19
Gambar 3. 5 Dataset Setelah Dibersihkan.....	20
Gambar 3. 6 Contoh Citra Mata Sebelum Proses Preprocessing	20
Gambar 3. 7 Contoh Citra Mata Setelah Proses Preprocessing	21
Gambar 3. 8 Dataset Class Pterigium Setelah di-crop	21
Gambar 3. 9 Dataset Class Normal Setelah di-crop.....	22
Gambar 3. 10 Resize Citra	22
Gambar 3. 11 Proses Deployment Model	25
Gambar 3. 12 Klasifikasi Citra Baru.....	25
Gambar 3. 13 Flowchart Alur Penggunaan Sistem.....	26
Gambar 3. 14 Halaman Beranda	32
Gambar 3. 15 Halaman Tips Penggunaan Aplikasi	33
Gambar 3. 16 Halaman Upload Foto	34
Gambar 3. 17 Halaman Result	34
Gambar 4. 1 Halaman Beranda	35
Gambar 4. 2 Tampilan Langkah 1 Tips Penggunaan Aplikasi	36
Gambar 4. 3 Tampilan Langkah 2 Mengambil Foto Mata.....	37
Gambar 4. 4 Opsi Pengambilan Foto Mata	38
Gambar 4. 5 Foto Mata yang Diambil dari Galeri	38
Gambar 4. 6 Hasil Crop Foto Mata yang Diambil dari Galeri.....	39
Gambar 4. 7 Hasil Klasifikasi	40

ABSTRAK

Bagi manusia, mata adalah salah satu indera yang sangat penting. Berbagai kegiatan dilakukan manusia melalui mata yang berfungsi menyerap informasi visual. Namun, mata adalah organ yang cukup rentan terhadap gangguan penglihatan, baik itu gangguan ringan maupun berat yang dapat menyebabkan kebutaan. Dengan terganggunya fungsi penglihatan, maka berbagai aktivitas yang dilakukan seseorang akan terganggu, dan dari berbagai macam gangguan kesehatan mata, salah satunya adalah pterigium. Pterigium adalah suatu penyakit yang mengganggu fungsi penglihatan dan apabila kondisinya sudah sangat parah bisa menyebabkan kebutaan, oleh karena itu, deteksi dini pterigium dapat dijadikan sebagai solusi dalam upaya pencegahan pterigium, dan deteksi pterigium dapat dilakukan menggunakan algoritma *Convolutional Neural Network*. Penelitian ini menggunakan algoritma *Convolutional Neural Network* untuk mendeteksi pterigium yang direpresentasikan ke dalam sistem berbasis android. Hasil penelitian dievaluasi berdasarkan akurasi pada tahap *training* dan *validation*, serta *testing* yang dilakukan secara manual pada *smartphone* android dan hasilnya akan dimasukkan ke dalam tabel *confusion matrix*. Dari 5 konfigurasi, model dengan kinerja terbaik pada tahap *training* dan *validation* adalah konfigurasi 4, yaitu nilai *loss* sebesar 0,2294, akurasi 88,89%, validasi *loss* 0,1052, dan validasi akurasi 100%. Dan pada tahap pengujian konfigurasi 1 dan 4, masing-masing nilai akurasi, *precision*, *recall*, dan F1 adalah 1 atau 100%.

Kata kunci: Mata, Pterigium, *Convolutional Neural Network*, Android

ABSTRACT

For humans, the eye is one of the most important senses. Various activities are carried out by humans through the eyes which function to absorb visual information. However, the eye is an organ that is quite vulnerable to visual disturbances, both mild and severe disorders that can cause blindness. With the disruption of the visual function, various activities carried out by a person will be disrupted, and from various kinds of eye health problems, one of which is pterygium. Pterygium is a disease that interferes with the function of vision and if the condition is very severe it can cause blindness, therefore, early detection of pterygium can be used as a solution in efforts to prevent pterygium, and pterygium detection can be carried out using the Convolutional Neural Network algorithm. This study uses the Convolutional Neural Network algorithm to detect pterygium which is represented in an Android-based system. The research results are evaluated based on accuracy at the training and validation stages, as well as testing that is done manually on an Android smartphone and the results will be entered into the confusion matrix table. Of the 5 configurations, the model with the best performance at the training and validation stage is configuration 4, which has a loss value of 0.2294, 88.89% accuracy, 0.1052 loss validation, and 100% accuracy validation. And in the configuration testing stages 1 and 4, the accuracy, precision, recall, and F1 values respectively are 1 or 100%.

Keywords: Eye, Pterygium, Convolutional Neural Network, Android

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teknologi berkembang dengan begitu pesat, semua sektor kehidupan mulai beralih dari tenaga manusia menjadi pemanfaatan teknologi untuk memudahkan pekerjaan. Pemanfaatan teknologi ini akan menghemat tenaga, waktu, dan biaya jika dibandingkan menggunakan tenaga manusia. Sehingga, ini akan mengubah pola hidup manusia yang serba dimudahkan oleh teknologi.

Karena itulah penulis ingin membuat sebuah sistem yang bisa membantu dalam bidang kesehatan, khususnya adalah kesehatan mata. Sistem yang akan dibuat yaitu sistem yang mampu mengenali penyakit mata pterigium. Diharapkan nantinya masyarakat dari berbagai latar belakang dapat lebih mudah mengecek kondisi mata mereka lebih dini guna mencegah terjangkit penyakit mata tersebut. Sistem ini juga bisa dijadikan alternatif sebagai bahan pertimbangan sebelum mengecek kondisi mata mereka ke dokter mata, apabila mengalami gejala awal pterigium seperti muncul garis merah muda pada bagian putih mata.

Dan pada Laporan Tugas Akhir ini, penulis berfokus untuk mengambil studi kasus ini yaitu penyakit pterigium. Pterigium adalah penyakit mata yang ditandai dengan tumbuhnya daging atau selaput berbentuk segitiga berwarna merah muda yang menjalar dari ujung mata hingga ke lensa mata atau lebih tepatnya pada *epitel konjungtiva bulbi*. Mayoritas penyakit ini diderita oleh orang-orang yang tinggal di negara yang dekat dengan garis ekuator dan penderitanya terpapar sinar ultraviolet secara terus-menerus, dan akibat dari penyakit ini mata akan terasa kabur, gatal, berair, dan merah (Kusuma dkk., 2018).

Pterigium tersebar di seluruh dunia namun prevalensi penyakit ini meningkat pada daerah ekuator atau daerah tropis seperti Indonesia, yang mengakibatkan penduduknya berisiko tinggi terkena penyakit ini dan jumlahnya tidak sedikit, yaitu prevalensi untuk orang dewasa >40 tahun adalah 16,8%, laki-laki 16,1%, dan perempuan 17,6% (Anida & Wibowo, 2017).

Terlebih lagi Indonesia adalah negara maritim dan agraris, dimana mayoritas masyarakatnya bekerja sebagai petani dan nelayan, dimana mereka bekerja di bawah sinar matahari langsung, namun tidak terbatas pada dua pekerjaan itu saja, karena penderita penyakit ini biasanya orang yang bekerja di luar ruangan. Oleh karena itu, penulis bertujuan membuat sistem yang mampu mendeteksi penyakit pterigium sejak dini, sehingga penyakit ini bisa lebih cepat ditangani.

1.2 Perumusan Masalah

Bagaimana algoritma *Convolutional Neural Network* yang direpresentasikan dalam sistem berbasis android akan melakukan penentuan atau klasifikasi dari gambar mata yang memiliki probabilitas mengidap penyakit mata pterigium.

1.3 Pembatasan Masalah

Adapun batasan masalah dalam tugas akhir ini adalah:

1. Dataset yang digunakan berasal dari link berikut, <https://data.mendeley.com/datasets/t75wjsw6bw/2> dimana terdapat gambar mata positif (pterigium) dan negatif (normal) yang sudah divalidasi oleh dua dokter mata bersertifikat.
2. Jumlah data yang digunakan sebanyak 120 sampel, terdiri dari 60 mata positif pterigium dan 60 mata negatif pterigium.
3. Sistem yang dibuat hanya untuk *platform* android.
4. Algoritma yang digunakan dalam sistem ini adalah *Convolutional Neural Network*.
5. Luaran dari sistem ini adalah menampilkan hasil deteksi penyakit pterigium beserta probabilitasnya.
6. Pengambilan gambar dilakukan menggunakan kamera belakang, untuk menghindari gambar blur yang mengakibatkan kesalahan prediksi.

1.4 Tujuan

Tujuan tugas akhir ini adalah menciptakan sistem deteksi penyakit mata pterigium berbasis android yang mengimplementasikan metode *Convolutional Neural Network*.

1.5 Manfaat

Adapun manfaat tugas akhir ini adalah:

1. Mempermudah melakukan deteksi dini penyakit mata pterigium, terhadap seseorang yang berisiko terjangkit pterigium.
2. Memudahkan untuk segera melakukan tindakan pencegahan lanjutan terhadap penyakit mata pterigium.
3. Memudahkan seseorang dalam melakukan deteksi pterigium dimanapun dan kapanpun, selama memiliki gawai.

1.6 Sistematika Penulisan

Adapun sistematika penulisan yang akan dipakai penulis dalam pembuatan laporan tugas akhir ini adalah sebagai berikut:

BAB I : PENDAHULUAN

Pada Bab 1, penulis memaparkan urgensi dari penelitian yang diangkat, mulai dari penulisan latar belakang, membuat rumusan masalah, membatasi permasalahan yang dibahas, serta tujuan dan manfaat yang diperoleh, dan diakhiri oleh model sistematika penulisan.

BAB II : TINJAUAN PUSTAKA DAN DASAR TEORI

Pada bab 2, penulis akan memaparkan dasar teori yang digunakan, serta rujukan dari penelitian terdahulu yang akan digunakan dalam perancangan sistem, dan membantu penulis untuk memahami algoritma *convolutional neural network* selama proses penelitian.

BAB III : METODE PENELITIAN

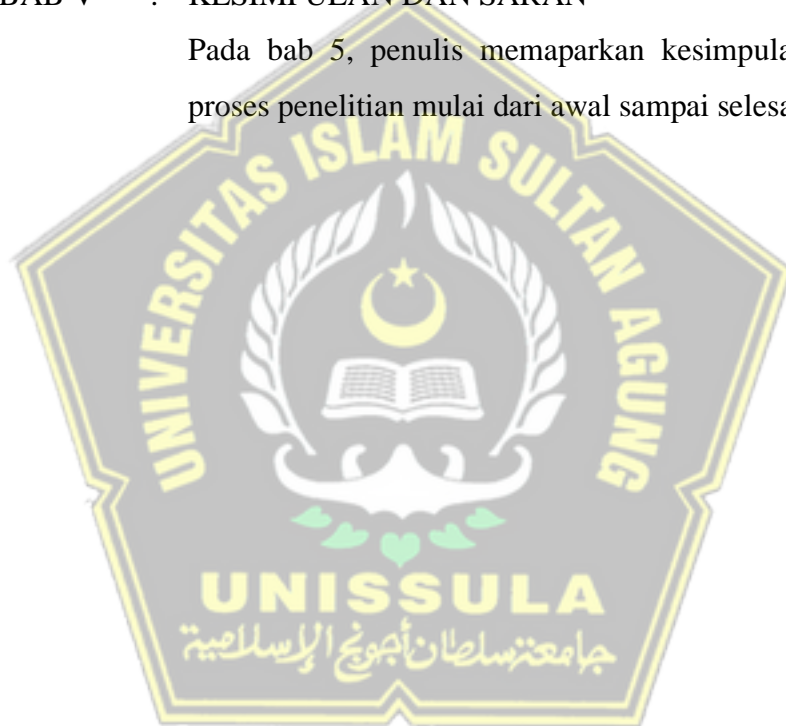
Pada bab 3, penulis mengungkapkan proses dan tahapan penelitian yang dimulai dari mendapatkan dataset hingga proses klasifikasi data uji yang ada.

BAB IV : HASIL DAN ANALISIS PENELITIAN

Pada bab 4, penulis memaparkan hasil penelitian, dimulai dengan hasil akhir sistem, klasifikasi data uji, dan akurasi dari sistem.

BAB V : KESIMPULAN DAN SARAN

Pada bab 5, penulis memaparkan kesimpulan dari hasil proses penelitian mulai dari awal sampai selesai.



BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Dalam sebuah penelitian yang telah dilakukan dengan judul “Distribusi dan Karakteristik Pterigium di Indonesia” menyatakan “Pterigium merupakan penyakit mata yang sering dijumpai di daerah beriklim tropis dan subtropis, dimana daerah ekuator akan mendapati prevalensi yang semakin tinggi, dan secara geografis Indonesia memiliki beberapa daerah yang terletak di ekuator” (Erry dkk., 2011). Penyakit pterigium merupakan penyakit mata yang ditandai dengan tumbuhnya selaput berbentuk segitiga yang tumbuh dicelah kelopak mata dan puncaknya mengarah ke bagian tengah kornea mata. Berdasarkan hasil Riset Kesehatan Dasar (Riskesdas) prevalensi pterigium di Indonesia pada kedua mata sebesar 3,2%, pterigium pada salah satu mata 1,9%. Prevalensi pterigium tertinggi pada kedua mata yaitu di Provinsi Sumatera Barat sebesar 9,4%, dan terendah di Provinsi DKI Jakarta 0,4%. Prevalensi pterigium tertinggi pada salah satu mata yaitu di Provinsi Nusa Tenggara barat sebesar 4,1%, dan terendah di Provinsi DKI Jakarta 0,2%.

Dalam sebuah penelitian yang dilakukan oleh Abdul Jalil Rozaqi dan kawan-kawan, mereka melakukan klasifikasi penyakit pada daun kentang melalui sebuah citra menggunakan metode *Convolutional Neural Network*. Akurasi yang didapat pada *epoch* ke-10 dari 922 data latih dan 230 data testing adalah 95% pada *training* dan 94% pada validasi (Rozaqi dkk., 2021).

Dalam sebuah penelitian yang dilakukan oleh Abdul Kholik, dimana peneliti melakukan klasifikasi tangkapan layar halaman instagram menggunakan metode *Convolutional Neural Network*. Hasil klasifikasi yang didapatkan cukup baik dengan akurasi yang didapat sebesar 91%, *precision* 93%, *recall* 90%, dan *F1 score* 91% (Kholik, 2021).

Dalam sebuah penelitian yang dilakukan oleh Siti Raihanah Abdani dan kawan-kawan, mereka melakukan penelitian untuk mendeteksi penyakit pterigium menggunakan arsitektur VGG-M melalui *Transfer Learning*. Hasil penelitian dari

konfigurasi 1 *Batch Normalization* dan *Dropout* menghasilkan akurasi 98,65%, konfigurasi 2 *Batch Normalization* 57,64%, konfigurasi 3 *Local Response Normalization* dan *Dropout* 98,33%, konfigurasi 4 *Local Response Configuration* 80%, dan konfigurasi 5 tanpa *Regularization* 59,02% (Abdani dkk., 2019).

Dalam penelitian yang dilakukan oleh Noviana Dewi dan Fiqih Ismawan, mereka melakukan penelitian untuk membuat sebuah sistem yang mampu mengenali wajah dengan metode *Convolutional Neural Network*. Hasil klasifikasi yang didapatkan sangat baik dengan data latih yang terbilang sangat sedikit yaitu sebanyak 35 sampel dan data uji 15 sampel, dengan akurasi didapat sebesar 99,84%, *precision* 98,4%, dan *recall* 98% (Dewi & Ismawan, 2021).

Sebelumnya sudah ada penelitian terdahulu yang melakukan deteksi pterigium menggunakan metode CNN, namun arsitektur atau konfigurasinya menggunakan *pre-trained CNN models (Transfer Learning)* yaitu model yang sudah terlatih dengan dataset yang besar dan melatihnya kembali dengan dataset yang dimiliki (Prakash & Kumari, 2019). Perbandingan akurasi dan model yang dipakai pada penelitian terdahulu dapat dilihat pada tabel 2.1.

Tabel 2. 1 Perbandingan Akurasi Metode CNN Pada Deteksi Pterigium

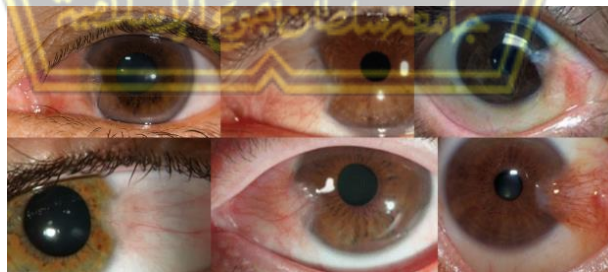
Penelitian	Dataset		Model	Akurasi
	<i>Train Set</i>	<i>Test Set</i>		
<i>Automated pterygium detection using deep neural network</i> (Zamani dkk., 2020)	193	193	<i>Pre-trained CNN</i> (VggNet16-wb)	99,22%
<i>Compact Convolutional Neural Networks for Pterygium Classification using Transfer Learning</i> (Abdani dkk., 2019)	60	60	<i>Pre-trained CNN</i> (VGG-M)	98,33%
<i>Implementation and Application of an Intelligent Pterygium Diagnosis System Based on Deep Learning</i> (Xu dkk., 2021)	750	470	<i>Pre-trained CNN</i> (EfficientNet-B6)	94,68%

Berdasarkan penjelasan dan perbandingan dari penelitian terdahulu, belum ada penelitian terkait deteksi pterigium menggunakan CNN model, yang mana arsitektur dan pengolahan data semuanya dilakukan secara mandiri, bukan menggunakan *pre-trained CNN Models (Transfer Learning)*. Sehingga, peneliti akan membuat sistem berbasis android yang dapat mendeteksi atau mengklasifikasikan pterigium menggunakan CNN model, dan menemukan arsitektur atau konfigurasi dengan jumlah layer yang lebih sedikit dari *pre-train CNN models*, sehingga proses pelatihan akan lebih banyak menghemat waktu, dengan tetap menghasilkan akurasi yang mendekati hasil akurasi *pre-trained CNN Models* yaitu $> 90\%$.

2.2 Dasar Teori

2.2.1 Pterigium

Bagi manusia, mata adalah salah satu indera yang sangat penting. Berbagai kegiatan dilakukan manusia melalui mata yang berfungsi menyerap informasi visual. Namun, mata adalah organ yang cukup rentan terhadap gangguan penglihatan, baik itu gangguan ringan maupun berat yang dapat menyebabkan kebutaan. Dengan terganggunya fungsi penglihatan, maka berbagai aktivitas yang dilakukan seseorang akan terganggu, dan dari berbagai macam gangguan kesehatan mata, salah satunya adalah pterigium.



Gambar 2. 1 Citra Mata Pterigium

Pterigium merupakan pertumbuhan jaringan yang mengandung pembuluh darah berbentuk segitiga seperti sayap, dimana jaringan tersebut berasal dari konjungtiva dan dapat menyebar ke limbus kornea dan seterusnya. Awal terbentuknya pterigium dikarenakan rusaknya limbal *stem cell* sebagai pembatas antara kornea dan konjungtiva. Paparan sinar ultraviolet mengakibatkan terjadinya

mutasi gen p53, sehingga memicu proliferasi sel epitel limbus membentuk jaringan fibrovaskular ke arah kornea. Pekerjaan di luar ruangan membuat seseorang terpapar sinar ultraviolet secara terus-menerus, inilah yang menjadi faktor utama kejadian pterigium. Faktor- faktor lain yang menyebabkan meningkatnya kejadian pterigium antara lain usia, faktor geografi, faktor genetik, jenis kelamin, iritasi kronik atau inflamasi, maupun riwayat terpapar lingkungan di luar rumah. Penderita pterigium akan mengalami perasaan mengganjal seperti adanya benda asing, mata merah, penurunan ketajaman penglihatan, terjadinya astigmatisme, dan menyebabkan gangguan lain seperti penglihatan ganda, inflamasi rekuren, gangguan pergerakan bola mata, bahkan jika kondisinya sudah parah bisa mengakibatkan kebutaan (Somba dkk., 2018).

2.2.2 Penyebab Pterigium

Pada sebuah penelitian yang dilakukan oleh Novita Rany dengan judul “Hubungan Lingkungan Kerja Dan Perilaku Nelayan Dengan Kejadian *Pterygium* Di Desa Kemang Kecamatan Pangkalan Kuras Kabupaten Pelalawan”, dimana dalam penelitian ini, peneliti mencoba mencari hubungan kejadian pterigium terhadap seseorang, berdasarkan lingkungan kerja dan perilaku mereka, dimana hasil penelitian ini, diambil kesimpulan bahwa faktor seseorang beraktivitas di luar ruangan lebih berisiko terkena pterigium dibandingkan seseorang yang bekerja di dalam ruangan, dan risiko semakin tinggi jika seseorang yang bekerja di luar ruangan tidak menggunakan alat pelindung diri, seperti kacamata dan topi, yang membuat seseorang secara langsung terpapar sinar ultraviolet, dan risikonya semakin bertambah seiring dengan lamanya bekerja di luar ruangan (Rany, 2017).

2.2.3 Machine Learning

Pembelajaran mesin (*Machine Learning*) adalah proses pembuatan mesin yang dapat belajar secara mandiri tanpa diprogram secara eksplisit. Proses manusia belajar dan men-generalisasi menjadi ide utama dalam *Machine Learning*. Klasifikasi dan prediksi adalah dua aplikasi utama dalam *Machine Learning*. Proses pelatihan atau *training* menjadi ciri khas dari *Machine Learning*, dimana data *training* dibutuhkan oleh model selama proses pembelajaran. Model yang dimaksud adalah program yang belajar secara otomatis menggunakan algoritma *Machine*

Learning untuk menyelesaikan masalah. Klasifikasi dalam studi kasus *Machine Learning* adalah bagaimana model mampu memilah objek berdasarkan ciri tertentu, selayaknya manusia dalam membedakan benda satu dengan benda lain, sedangkan prediksi atau regresi dalam studi kasus *Machine Learning* adalah bagaimana model mampu menerka keluaran dari data masukan berdasarkan data yang sudah dipelajari selama proses pelatihan (Hania, 2017).

Menurut (Pambudi dkk., 2020) *machine learning* adalah suatu metode ekstraksi data menjadi pola informasi tertentu, sedangkan (Fahrizal dkk., 2020) mendefinisikan *machine learning* adalah bagian dari *Artificial Intelligence* (AI) yang berfokus pada sistem yang mampu belajar secara otomatis tanpa diprogram oleh manusia.

Saat ini, penggunaan ataupun pemanfaatan *machine learning* sudah hampir merata diberbagai sektor yang manusia kerjakan, dari sektor kesehatan, transportasi, bahkan sampai dengan sektor politik, tujuan dari pemanfaatan *machine learning* sendiri pada umumnya adalah untuk melakukan tugas yang berulang-ulang, sehingga memudahkan pekerjaan manusia.

Pada penelitian yang dilakukan oleh Devica Natalia Br Ginting dan Sanjiwana Arjasakusuma yang berjudul “Pemetaan Lamun Menggunakan *Machine Learning* Dengan Citra Planetscope di Nusa Lembongan”, dimana hasil penelitian ini menunjukkan bahwa model yang dihasilkan dengan metode *machine learning* yaitu *Random Forest* dan *Decision Tree* memiliki akurasi yang baik, sehingga dapat digunakan untuk pemetaan lamun dengan lebih efektif dan efisien (Ginting & Arjasakusuma, 2021).

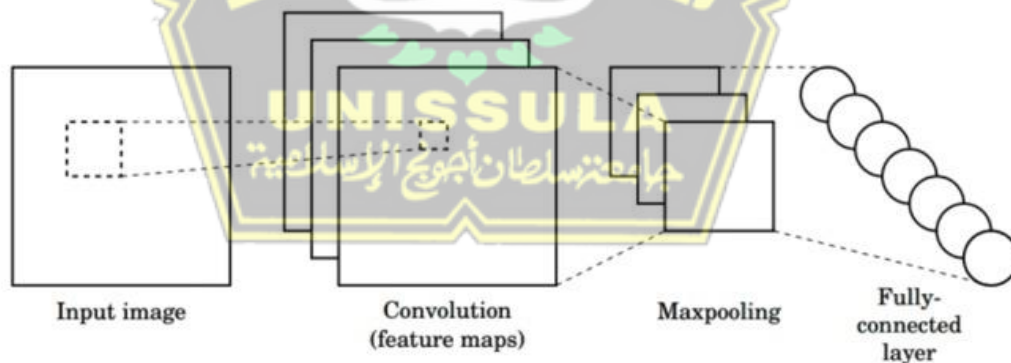
2.2.4 Deep Learning

Deep Learning (DL) adalah subbidang *machine learning* yang algoritmanya terinspirasi dari struktur otak manusia. Algoritma ini memungkinkan mesin untuk melihat pola atau data yang fiturnya tidak dapat ditentukan secara langsung, seperti data gambar, audio, dan video. Hilangnya gradien pada *backpropagation* membuat waktu yang dibutuhkan dalam proses pelatihan pada model *Deep Learning* semakin sedikit. Beberapa jenis *Deep Learning* antara lain *Deep Auto Encoder*, *Deep Belief Nets*, *Convolutional Neural Network*, dan lain-lain (Hania, 2017).

Pada penelitian yang dilakukan oleh Sarirotul Ilahiyah dan Agung Nilogiri yang berjudul “Implementasi *Deep Learning* Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan *Convolutional Neural Network*”, dimana peneliti menggunakan arsitektur Alexnet yang terdiri dari 20 lapisan, dan sistem mampu mengidentifikasi jenis genus tumbuhan dengan akurasi sebesar 90,8% (Ilahiyah & Nilogiri, 2018).

2.2.5 *Convolutional Neural Network*

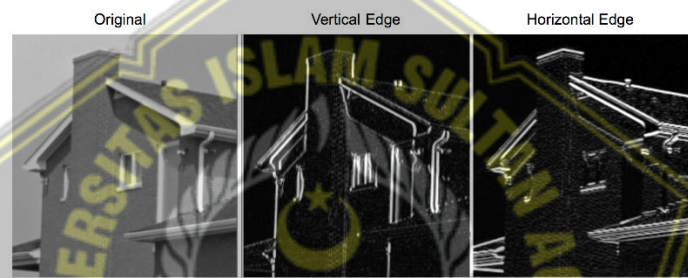
Convolutional Neural Network merupakan kumpulan lapisan pemrosesan yang menjalankan operasi konvolusi, dan beroperasi secara paralel yang terdiri dari beberapa elemen, dimana model ini terinspirasi oleh sistem saraf biologis (Romario dkk., 2020). *Image processing* adalah salah satu bidang *deep learning* yang dapat dipecahkan masalahnya dengan metode CNN, karena mampu menangani permasalahan kompleks, serta memiliki kinerja yang baik dan juga memiliki kemampuan klasifikasi dalam pengolahan data citra yang nantinya direpresentasikan dalam bentuk matriks. CNN juga adalah *supervised learning* yaitu model di-*training* terlebih dahulu untuk mengenali pola antara input data dan label *output* (Harani dkk., 2019).



Gambar 6. 1 Arsitektur *Convolutional Neural Network*

Gambar 6.1 pada arsitektur CNN tersebut, sebuah gambar masukan dideteksi atribut atau fiturnya dengan menggunakan konvolusi tiga filter. Hasil dari konvolusi disebut *feature map* yang kemudian akan dilakukan *max pooling* dan akan menghasilkan tiga buah gambar baru dengan resolusi yang lebih kecil. Terakhir, hasil *max pooling* dimasukkan ke dalam *fully connected layer*.

Dalam arsitektur CNN terbagi atas *feature extraction layer* dan *fully connected layer*. *Feature extraction* adalah proses mengekstrak fitur pada suatu objek (citra, teks, suara, dan sebagainya) untuk mengenali informasi prediktif. *Convolution layer* dan *pooling layer* adalah *feature extraction layer* yang ada pada arsitektur CNN. *Convolution layer* merupakan lapisan yang digunakan untuk melakukan operasi konvolusi pada gambar masukan. Layer ini termasuk blok utama pada model *Convolutional Neural Network* (CNN) yang didalamnya terdiri dari filter–filter berbentuk matriks yang secara mandiri dipelajari secara acak oleh model untuk melakukan operasi konvolusi yang bertujuan sebagai ekstraksi fitur untuk mempelajari representasi fitur dari gambar masukan (Putra dkk., 2017).

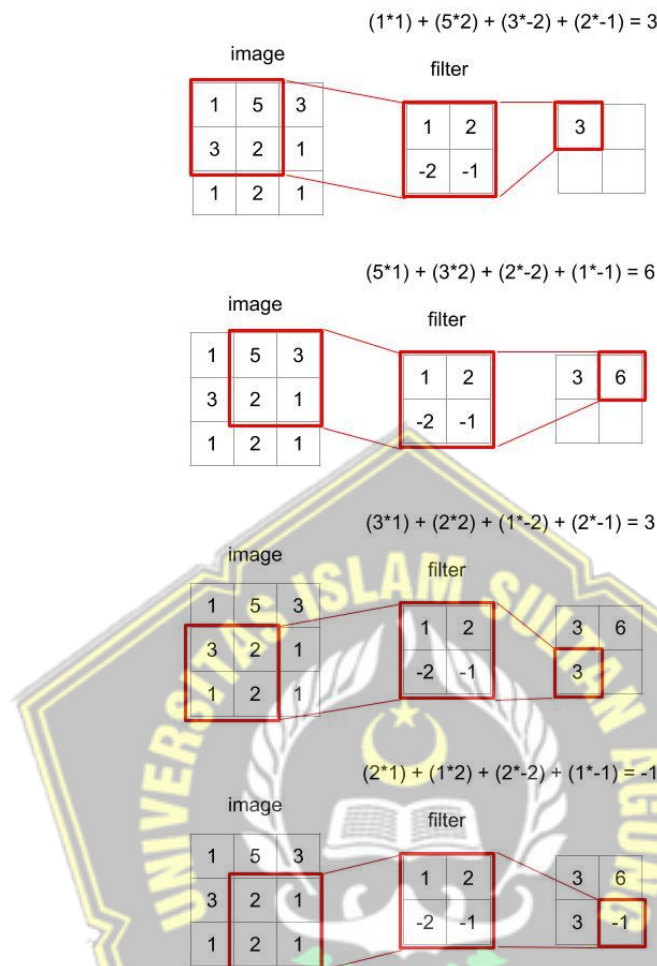


Gambar 6. 2 Contoh Filter yang Digunakan Saat Konvolusi

Filter tersebut nantinya akan digunakan saat konvolusi pada citra, yang mana tujuan dilakukannya untuk mengekstraksi fitur dari *input* citra. Konvolusi tersebut akan menghasilkan transformasi linear dari data yang dimasukkan sesuai informasi spasial yang tersedia pada data. Bobot pada layer tersebut akan menspesifikasikan filter yang digunakan sehingga filter dapat dilatih berdasarkan masukan pada *Convolutional Neural Network*.

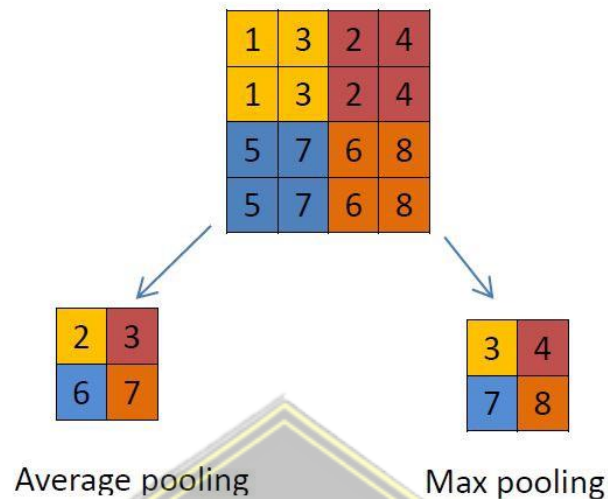
image			filter	
1	5	3	1	2
3	2	1	2	1
7	5	5		

Gambar 6. 3 Contoh Matriks *Image* dan *Filter*



Gambar 6. 4 Proses Konvolusi Citra

Selanjutnya masuk ke *pooling layer* yaitu lapisan yang berfungsi untuk mengurangi ukuran spasial dari fitur konvolusi sehingga sumber daya komputasi yang dibutuhkan dalam memproses data melalui pengurangan dimensi dari *feature map* (*downsampling*) dapat dikurangi, serta membuat komputasi yang semakin cepat karena parameter yang diperbarui semakin sedikit. Selain itu, berguna untuk mengekstraksi fitur dominan sehingga proses pelatihan model lebih efektif. Terdapat dua jenis *pooling layer*, yaitu *max pooling* dan *average pooling*. *Max pooling* mengembalikan nilai maksimum dari spasial konvolusi yang dicakup oleh filter/ kernel, sedangkan *average pooling* mengembalikan nilai rata-rata dari spasial konvolusi yang dicakup oleh filter/ kernel (Saha, 2018).

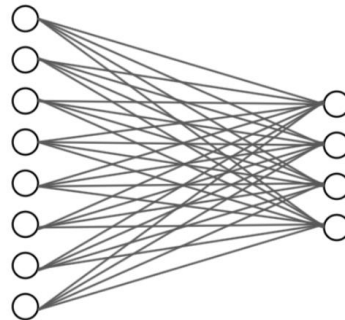


Gambar 6. 5 Konsep *Pooling*

Dan *fully connected layer* merupakan lapisan yang mentransformasi dimensi data agar dapat diklasifikasi secara linear. Gambar masukan pada *fully connected layer* akan masuk ke tahap *flatten* terlebih dahulu untuk membentuk ulang fitur (*reshape feature map*, merubah matriks 2D menjadi 1D) menjadi sebuah vektor (Reddy dkk., 2019).

Untuk mendapatkan hasil keluaran dari layer ini tidak dibutuhkan operasi konvolusi, tetapi menggunakan komputasi perkalian matriks yang diikuti dengan bias *offset*. Dengan penggunaan operasi tersebut, setiap neuron menerima masukan bilangan numerik dan memprosesnya untuk menghasilkan sebuah keluaran, dimana neuron memiliki koneksi penuh ke semua aktivasi dalam lapisan sebelumnya, sehingga layer ini disebut sebagai *fully connected layer* (Zhang dkk., 2019).

Fully Connected



Gambar 6. 6 Arsitektur *Fully Connected Layer*

Fungsi aktivasi atau disebut juga neuron merupakan fungsi non-linear yang mampu membuat sebuah model CNN untuk menyelesaikan permasalahan non-trivial. Setiap fungsi aktivasi mengambil sebuah nilai dan melakukan operasi matematika. Fungsi aktivasi pada arsitektur *Convolutional Neural Network* terletak pada perhitungan akhir keluaran *feature map* yaitu setelah proses konvolusi atau *pooling* untuk menghasilkan suatu pola fitur. Ada dua jenis fungsi aktivasi untuk *output layer* yaitu *softmax* dan *sigmoid* (Zufar & Setiyono, 2017).

Softmax yaitu menghitung probabilitas dari setiap kelas target untuk membantu menentukan kelas target berdasarkan masukan yang diberikan. Keuntungan penggunaan fungsi aktivasi *softmax* adalah rentang probabilitas *output* yang berkisar antara 0 dan 1, serta apabila keseluruhan probabilitas di semua target kelas dijumlahkan hasilnya adalah satu. Fungsi aktivasi *softmax* umumnya digunakan untuk kasus *multi class*, dimana fungsi tersebut akan mengembalikan peluang dari masing-masing kelas dan kelas target akan memiliki probabilitas lebih tinggi. *Softmax* menggunakan eksponensial (*e-power*) dari nilai masukan yang diberikan dan jumlah nilai eksponensial dari semua nilai dalam masukan. Maka, rasio eksponensial dari nilai masukan dan jumlah nilai eksponensial adalah keluaran dari fungsi *softmax* (Gerald & Lubis, 2020). Berikut adalah formula dari fungsi aktivasi *softmax*.

$$\sigma(\vec{Z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (1)$$

Keterangan:

Tabel 2. 2 Penjelasan Simbol Pada Formula *Softmax*

Simbol	Keterangan
\vec{Z}	Vektor <i>input</i> ke fungsi <i>softmax</i> , terdiri dari (z_0, \dots, z_k) .
z_i	Semua nilai z_i adalah elemen dari vektor <i>input</i> ke fungsi <i>softmax</i> , dan dapat mengambil nilai <i>real</i> apa pun, positif, nol, atau negatif. Misalnya, jaringan saraf dapat mengeluarkan vektor seperti $(0.62, -8.12, 2.53)$, yang bukan merupakan distribusi probabilitas yang valid, oleh karena itu mengapa <i>softmax</i> diperlukan.
e^{z_i}	Fungsi eksponensial standar diterapkan pada setiap elemen dari vektor <i>input</i> . Ini memberikan nilai positif di atas 0, yang akan sangat kecil jika masukannya negatif, dan sangat besar jika masukannya besar. Namun, itu masih belum ditetapkan dalam kisaran $(0, 1)$ yang diperlukan dari suatu probabilitas.
$\sum_{j=1}^K e^{z_j}$	Symbol tersebut adalah istilah normalisasi. Ini memastikan bahwa semua nilai keluaran fungsi akan berjumlah 1 dan masing-masing berada dalam kisaran $(0, 1)$, sehingga merupakan distribusi probabilitas yang valid.
K	Jumlah kelas dalam multi-class classifier.

Sedangkan fungsi aktivasi *sigmoid* mentransformasi rentang nilai dari input x menjadi antara 0 dan 1 dengan bentuk distribusi fungsi. Namun, fungsi *sigmoid* sudah tidak banyak lagi digunakan, karena proses *backpropagation* yang tidak ideal yang disebabkan keluaran fungsi aktivasi *sigmoid* tidak berpusat pada nilai nol (Putra dkk., 2017). Berikut adalah formula dari fungsi aktivasi *sigmoid*.

$$S(x) = \frac{1}{1 + e^{-x}} \quad (2)$$

Kemudian untuk *feature extraction* pada *convolution layer* dan *pooling layer*, aktivasi yang paling sering digunakan adalah *Rectified Linear Unit* (ReLU)

yang artinya unit linear yang diperbaiki. ReLU adalah fungsi aktivasi non-linear yang digunakan dalam *deep neural network*. Fungsi ini direpresentasikan sebagai:

$$f(x) = \max(0, x) \quad (3)$$

Menurut persamaan 3, keluaran ReLU adalah nilai maksimum antara nol dan nilai masukan. Suatu keluaran sama dengan nol bila nilai masukannya negatif dan keluarannya sama dengan nilai masukan apabila masukannya positif. Dengan demikian, dapat ditulis ulang persamaan 3 sebagai berikut:

$$f(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases} \quad (4)$$

2.2.6 TensorFlow

TensorFlow adalah *library open-source* yang dikembangkan oleh tim Google Brain (tim peneliti AI di Google) pada tahun 2015 yang ditulis dengan Bahasa C++ untuk aplikasi *Artificial Intelligence*. TensorFlow memudahkan pembuatan model *machine learning* yang dapat dipakai untuk *deep learning*, *computer vision*, *reinforcement learning*, serta *natural language processing*. TensorFlow bersifat *multiplatform* artinya dapat digunakan di Windows, MacOS, maupun Linux dan dapat dijalankan di CPU (*Central Processing Unit*), GPU (*Graphics Processing Unit*), dan TPU (*Tensor Processing Unit*) (Padilha & Lucena, 2020).

2.2.7 Keras

Keras adalah *library Deep Learning* yang dibangun di atas TensorFlow dengan struktur kode yang minimalis dan simpel dalam pengembangan model *Deep Learning*, dalam studi kasus ini yaitu CNN.

2.2.8 TensorFlow Lite

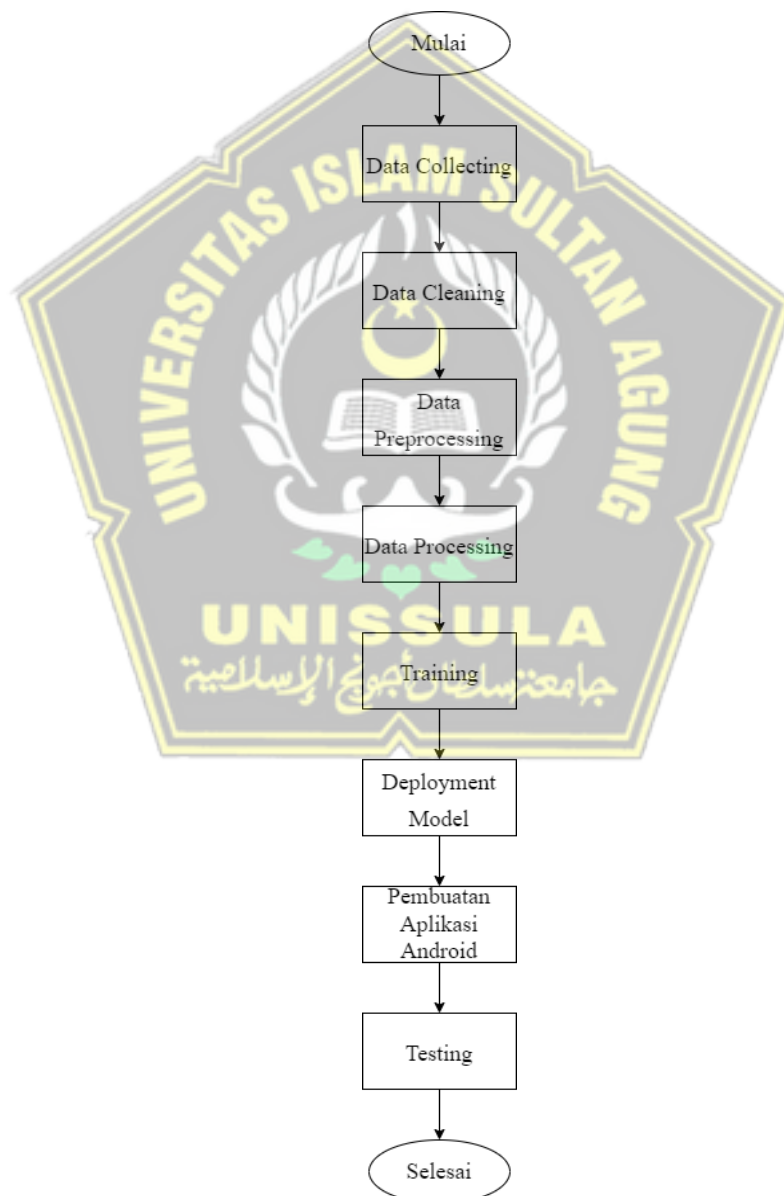
TensorFlow Lite adalah sebuah *library* untuk mengkonversi model TensorFlow menjadi sebuah file baru berekstensi *.tflite*. Proses konversi ini dinamakan *deployment model*. File *.tflite* tersebut akan digunakan untuk membuat program android yang mampu melakukan klasifikasi gambar dengan model yang sudah di-*training* sebelumnya.

BAB III

METODOLOGI PENELITIAN

3.1 Metode Penelitian

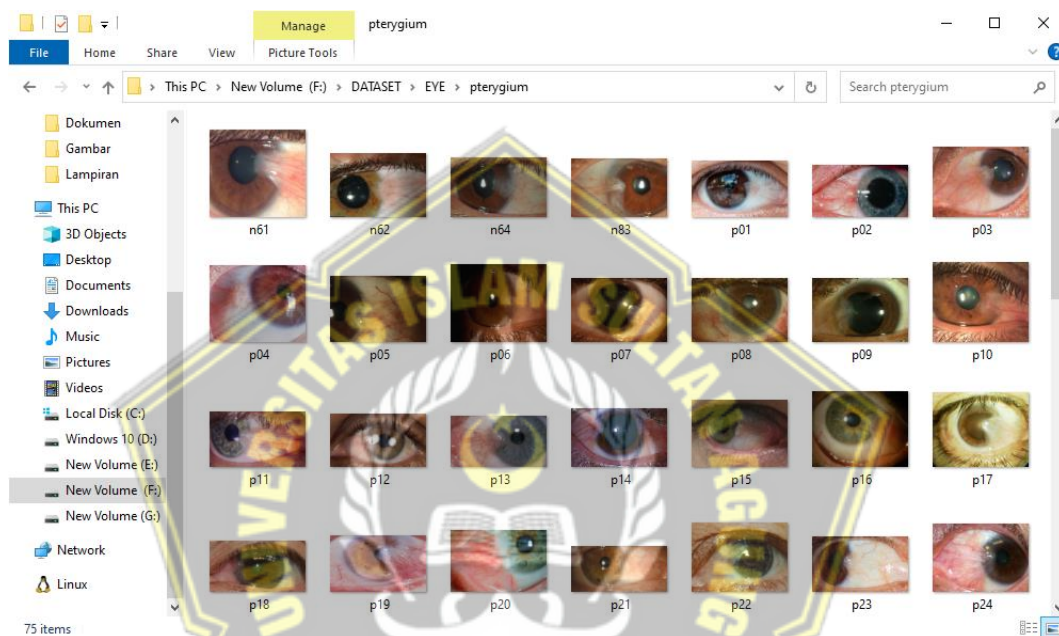
Dalam penelitian ini, metode atau algoritma yang digunakan adalah *Convolutional Neural Network*, dimana metode ini nantinya akan melakukan klasifikasi citra mata baru, dan ada beberapa tahapan yang perlu dilakukan dalam penelitian ini, seperti yang terlihat pada gambar 3.1:



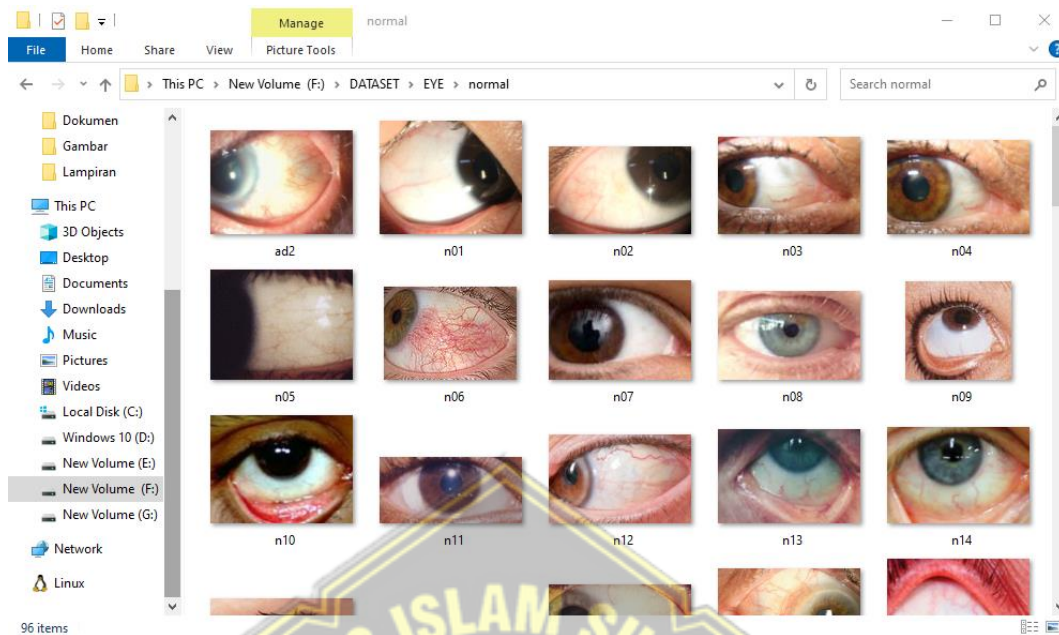
Gambar 3. 1 *Flowchart* Langkah Penelitian

3.1.1 Data Collecting

Tahap pertama yaitu mengumpulkan data berupa citra mata positif pterigium dan negatif pterigium. Data tersebut diambil dari link berikut, <https://data.mendeley.com/datasets/t75wjsw6bw/2>. Citra pada dataset tersebut telah diklasifikasikan secara manual oleh dua dokter mata bersertifikat, dimana gambar ambigu telah dikeluarkan untuk memperjelas *domain* gambar.

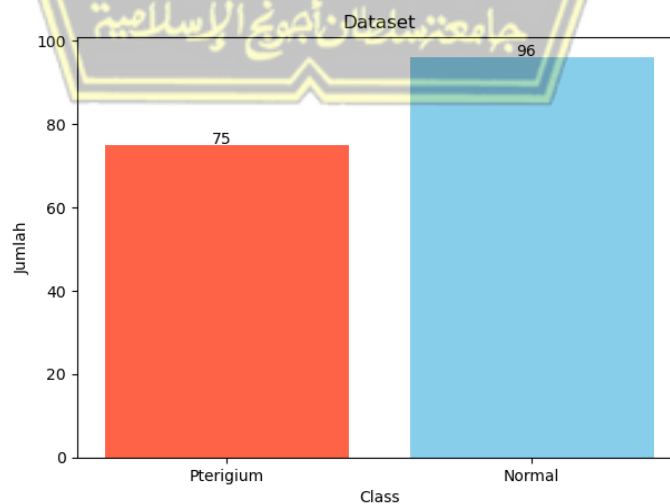


Gambar 3. 2 Dataset *Class Pterigium*

Gambar 3. 3 Dataset *Class Normal*

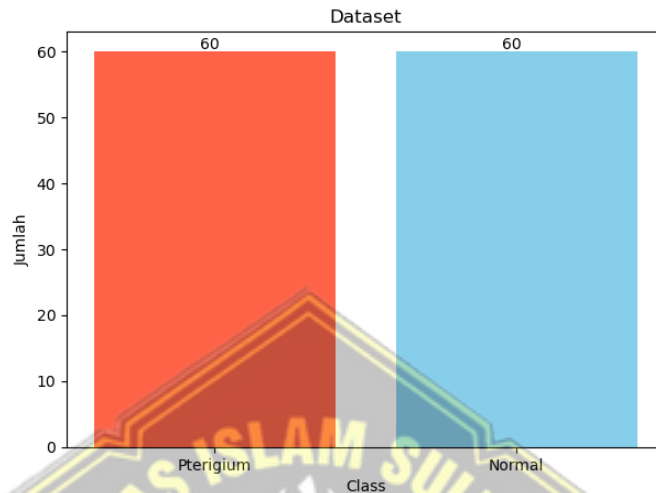
3.1.2 Data Cleaning

Saat ini dataset memiliki distribusi data yang tidak seimbang atau disebut *skewness distribution*, dimana pada *class* pterigium memiliki 75 data, sedangkan *class* normal memiliki 96 data. Ini akan menciptakan bias terhadap model, yaitu kecenderungan model untuk lebih sering memprediksi *class* normal, karena lebih sering mempelajari *class* tersebut.



Gambar 3. 4 Dataset Sebelum Dibersihkan

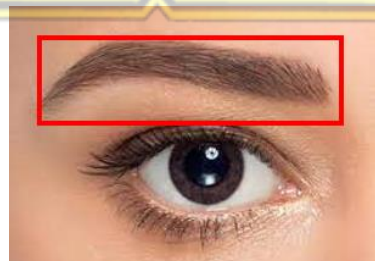
Dataset yang digunakan berjumlah 60 sampel mata normal dan 60 sampel mata pterigium, sehingga perlu menghapus 15 gambar pada *class* pterigium dan 36 gambar pada *class* normal secara acak.



Gambar 3. 5 Dataset Setelah Dibersihkan

3.1.3 Data Preprocessing

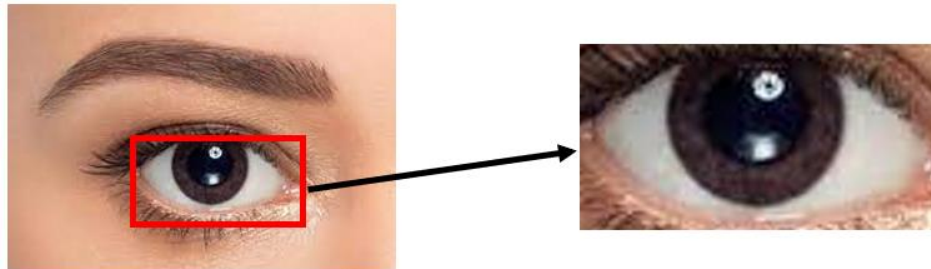
Sebelum masuk ke pengolahan data, perlu dilakukan *preprocessing* terhadap citra mata, dimana citra akan di-*crop* hingga mendekati pola bentuk mata. Ini dilakukan karena saat proses konvolusi jenis filter akan di-*training* secara mandiri oleh model untuk menemukan filter terbaik bagi model tersebut. Sehingga, jika tidak dilakukan *cropping*, bisa jadi model akan memilih jenis filter yang mendeteksi objek lain selain mata.



Gambar 3. 6 Contoh Citra Mata Sebelum Proses Preprocessing

Gambar 3.6 adalah contoh citra mata yang belum di-*crop* hingga pola bentuk mata. Jika data seperti ini dibiarkan, ketika model dilatih dan masuk tahap konvolusi, bisa jadi model akan memilih filter yang mampu mendeteksi alis,

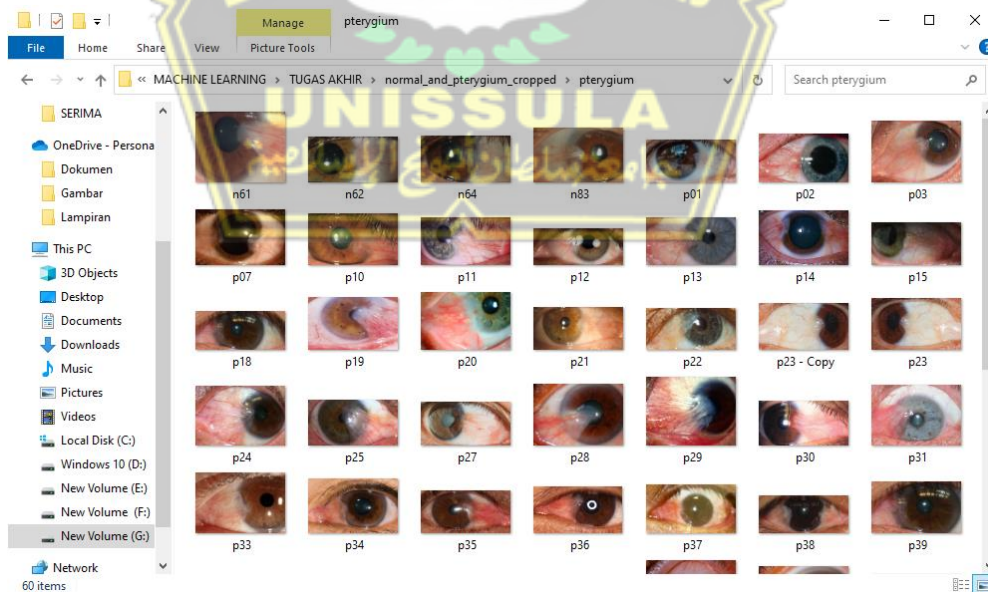
sehingga akan mempengaruhi hasil klasifikasi. Dalam studi kasus ini, alis bukanlah atribut yang digunakan dalam proses klasifikasi, karena penyakit pterigium terjadi dibagian mata saja.



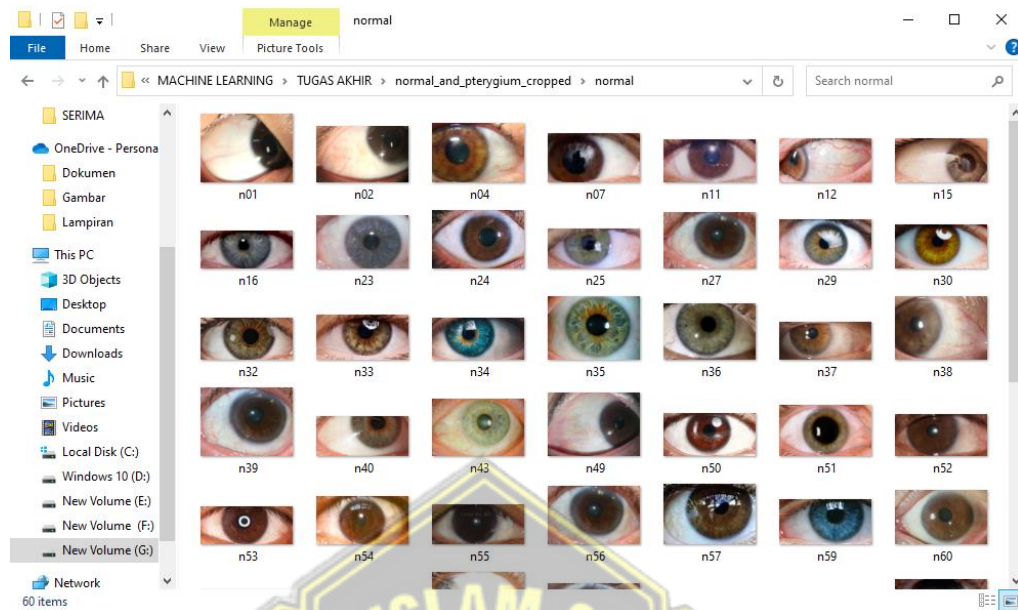
Gambar 3. 7 Contoh Citra Mata Setelah Proses *Preprocessing*

Gambar 3.7 adalah contoh citra mata yang sudah di-*crop* hingga pola bentuk mata. Memang masih ada objek lain yang terlihat yaitu bulu mata, namun citra lebih dominan menampilkan pola mata, sehingga besar kemungkinan model akan memilih filter yang sesuai untuk mendeteksi mata.

Gambar 3.8 dan 3.9 adalah dataset *class* pterigium dan *class* normal yang telah di-*crop* secara manual oleh peneliti dan siap masuk ke tahap selanjutnya, yaitu *Data Processing*.



Gambar 3. 8 Dataset Class Pterigium Setelah di-*crop*



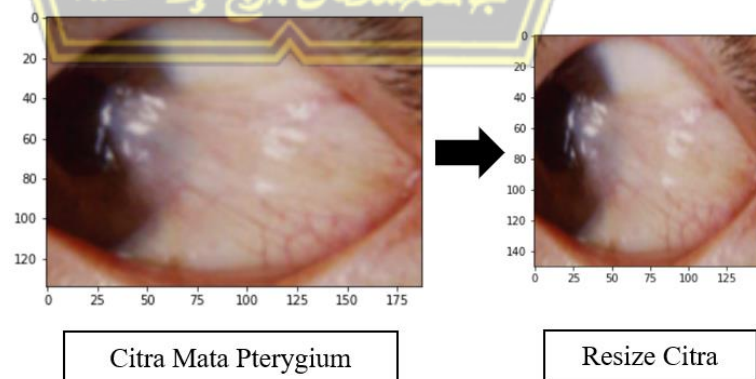
Gambar 3. 9 Dataset *Class Normal* Setelah di-crop

3.1.4 Data Processing

Sebelum masuk tahap *training*, data perlu diolah terlebih dahulu untuk memastikan kualitas dan representasi data yang sesuai dengan karakteristik model CNN, diantaranya *resize* citra, normalisasi, dan *image augmentation*.

A. *Resize* Citra

Citra di-*resize* dengan ukuran 150px*150px, dengan tujuan menyesuaikan ukuran citra pada *train set* dan *validation set*.



Gambar 3. 10 *Resize* Citra

B. Normalisasi

Kemudian citra akan dinormalisasi, gunanya untuk mengubah nilai dari sebuah fitur ke dalam skala yang sama. Sebuah warna memiliki intensitas nilai dari 0-255, setelah dilakukan normalisasi nilai tersebut akan bernilai dari 0-1.

C. Image Augmentation

Diketahui dataset yang yang dipakai berjumlah 120 sampel yang dibagi menjadi 90 data latih dan 30 data validasi serta uji. Dengan dataset yang sedikit akan membuat model yang dilatih berkemungkinan besar mengalami *overfitting*. *Overfitting* yaitu ketika model melakukan prediksi dengan baik pada data latih, dan prediksinya buruk pada data uji. Untuk mengatasi permasalahan ini salah satu cara yang dapat dilakukan adalah augmentasi gambar.

Augmentasi gambar adalah sebuah teknik untuk memperbanyak data latih. Prinsipnya yaitu melakukan duplikasi gambar yang sudah ada dengan beberapa variasi sehingga data seolah menjadi lebih banyak. Contoh variasinya yaitu *width shift range*, *height shift range*, *shear range*, *horizontal flip*, *brightness range*, dll. Saat melakukan proses *training*, model akan dilatih berkali-kali sebanyak *n-epochs*. Dengan menerapkan augmentasi gambar, setiap *epochs* akan melatih data yang berbeda, dan membuat model menjadi *good fit*, yaitu model memiliki prediksi yang baik pada data latih maupun data uji.

Pada penelitian ini variasi yang digunakan adalah *shear range*, *horizontal flip*, dan *fill mode*.

1. *Shear Range*, untuk intensitas geser (sudut geser berlawanan arah jarum jam dalam derajat).
2. *Horizontal Flip*, untuk membalikkan gambar secara horizontal.
3. *Fill Mode*, saat gambar bergeser maka nilai pixel yang kosong tersebut akan diisi sesuai dengan mode yang diberikan. Dalam kasus ini menggunakan *nearest* artinya mengisi area dengan pixel terdekat dan merenggangkannya.

3.1.5 Training

Dataset yang digunakan dibagi menjadi data *training*, *validation*, dan *testing* dengan pembagian seperti pada tabel 3.1.

Tabel 3. 1 Pembagian Data Training, Validation, dan Testing

Nama Data	Jumlah (%)	Jumlah Angka	Positif Pterigium	Negatif Pterigium
Data <i>Training</i>	75%	90	45	45
Data <i>Validation</i> dan <i>Testing</i>	25%	30	15	15

Total Data: 120

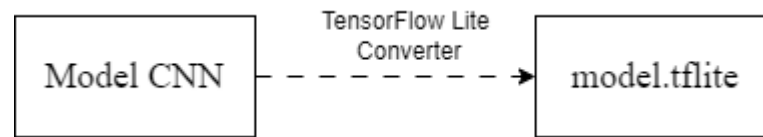
Data *validation* menggunakan sampel yang sama dengan *testing*, bedanya *validation* adalah *testing* di setiap proses *training*, sedangkan pada penelitian ini *testing* dilakukan secara manual langsung dari *smartphone*.

Pada tahap ini peneliti membuat 5 file (*notebook*) yang berisi masing-masing konfigurasi pada tabel 3.2 dan *training* dilakukan secara bersamaan, dimana tahapan *training* akan dilakukan sebanyak 30-epochs atau 30 kali pelatihan.

Tabel 3. 2 Konfigurasi Arsitektur CNN

	Convolution Layer	Pooling Layer	Hidden Layer	Dropout Layer
Konfigurasi 1	6	2	1	-
Konfigurasi 2	9	3	1	-
Konfigurasi 3	3	1	1	2
Konfigurasi 4	6	2	1	3
Konfigurasi 5	9	3	1	4

3.1.6 Deployment Model

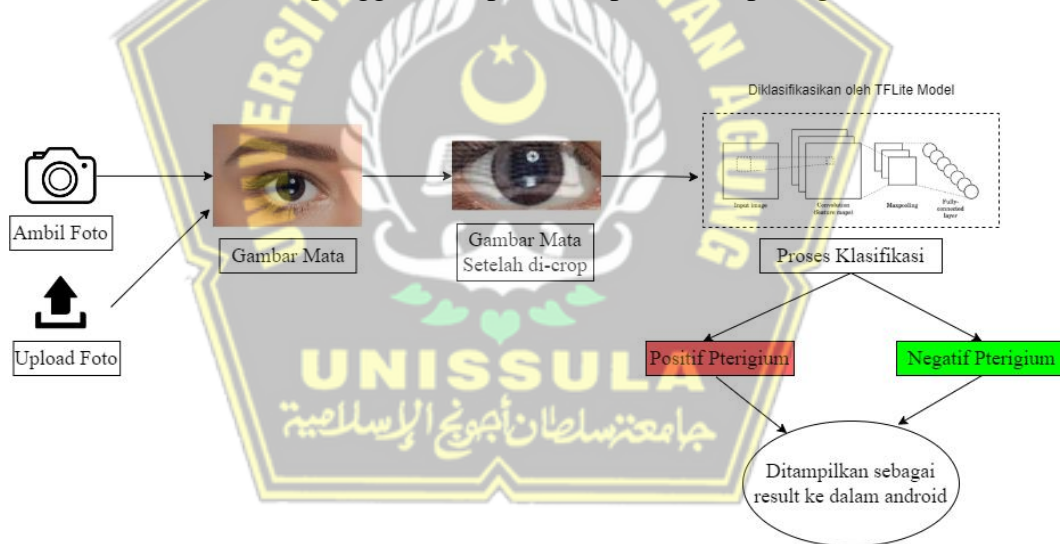


Gambar 3. 11 Proses *Deployment Model*

Setelah kelima konfigurasi pada tabel 3.2 telah di-*training*, maka model siap dikonversi ke dalam format .tflite, dengan bantuan TFLiteConverter yang sudah tersedia pada *library* TensorFlow. Kemudian, hasil konversi berupa file dengan ekstensi .tflite sudah dapat digunakan pada *environment* android untuk melakukan klasifikasi pterigium dan menampilkan probabilitasnya.

3.1.7 Pembuatan Aplikasi Android

Setelah model siap digunakan, selanjutnya proses pembuatan aplikasi android. Hasil dan alur penggunaan aplikasi dapat dilihat pada gambar 3.12.

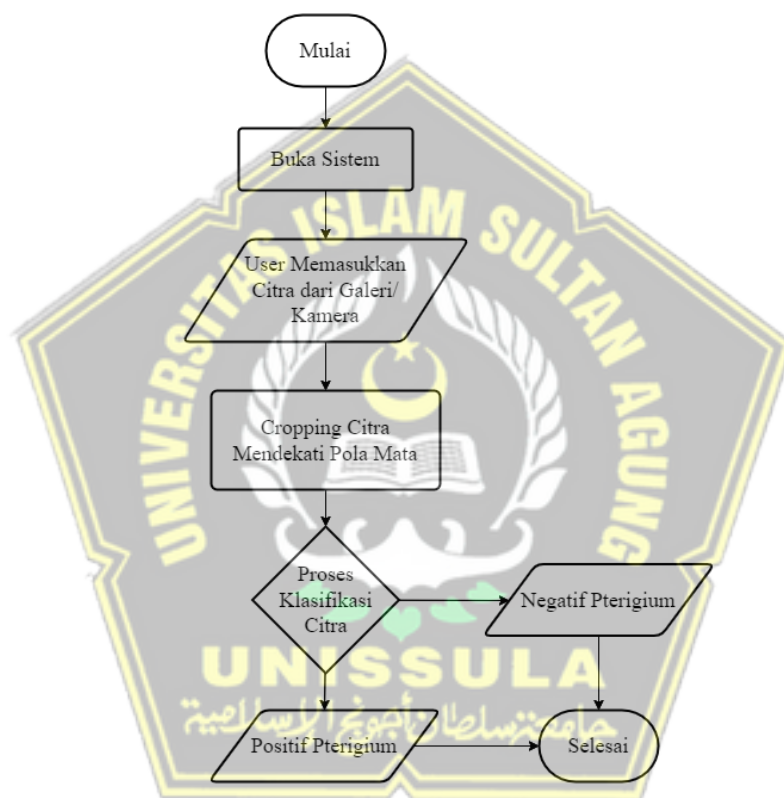


Gambar 3. 12 Klasifikasi Citra Baru

Gambar 3.12 menunjukkan alur proses klasifikasi citra, dimana pengguna akan memasukan citra mata yang dapat diambil langsung dari kamera atau memilih dari galeri, kemudian citra mata akan di-*crop* dan masuk ke proses klasifikasi oleh model TFLite, setelah itu sistem akan menampilkan hasil deteksi atau klasifikasi citra yaitu positif atau negatif pterigium.

3.1.8 Testing

Pada tahap pengujian dilakukan langsung secara manual pada *smartphone* dengan tipe Samsung Galaxy A50, Android 11 OS. Kinerja sistem dievaluasi berdasarkan hasil klasifikasi *confusion matrix*, yang menampilkan informasi terkait *accuracy*, *precision*, *recall*, dan F1 guna memastikan model memiliki kinerja yang baik. Gambar 3.13 adalah alur penggunaan sistem untuk melakukan pengujian pada data *testing*, yaitu 15 citra positif dan 15 citra negatif pterigium.



Gambar 3. 13 *Flowchart* Alur Penggunaan Sistem

Gambar 3.13 adalah *flowchart* alur penggunaan sistem yang dimulai dari pengguna membuka aplikasi, dengan begitu pengguna bisa memulai proses deteksi citra. Pertama, pengguna harus meng-*upload* citra terlebih dahulu, disini sistem memberikan dua pilihan yaitu mengambil citra langsung dari kamera atau memilih dari galeri. Setelah citra dipilih, citra wajib di-*crop* hingga mendekati pola mata, kemudian citra akan diklasifikasikan oleh model .tflite dan menampilkan hasilnya ke pengguna. Tahapan pengujian dilakukan persis seperti pengguna menggunakan sistem.

Tabel 3. 3 Tabel *Confusion Matrix*

	<i>Actually Positive</i>	<i>Actually Negative</i>
<i>Predicted Positive</i>	N (TP)	N (FP)
<i>Predicted Negative</i>	N (FN)	N (TN)

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (5)$$

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \quad (8)$$

Penjelasan terkait *confusion matrix*, *accuracy*, *precision*, *recall*, dan F1 adalah sebagai berikut:

A. *Confusion Matrix*

Confusion matrix adalah tabel yang merangkum seberapa sukses model klasifikasi dalam memprediksi data uji. *Confusion matrix* adalah tabel berisi 4 kombinasi berbeda dari nilai prediksi dan nilai aktual. Ada beberapa istilah yang merepresentasikan hasil klasifikasi dalam *confusion matrix*:

1. *True Positive*, ketika hasil prediksi positif sesuai dengan nilai sebenarnya yaitu positif.
2. *False Positive*, ketika hasil prediksi positif tidak sesuai dengan dengan nilai sebenarnya yaitu negatif.
3. *True Negative*, ketika hasil prediksi negatif sesuai dengan dengan nilai sebenarnya yaitu negatif.
4. *False Negative*, ketika hasil prediksi negatif tidak sesuai dengan nilai sebenarnya yaitu positif.

B. *Accuracy*

Accuracy adalah rasio prediksi benar (positif dan negatif) dengan keseluruhan data yang menggambarkan seberapa akurat model dalam melakukan prediksi. Dengan kata lain, *accuracy* merupakan tingkat kedekatan nilai prediksi dengan nilai aktual.

C. *Precision*

Precision adalah rasio prediksi benar (positif atau negatif) dengan jumlah hasil prediksi positif atau negatif.

D. *Recall*

Recall adalah rasio prediksi benar (positif atau negatif) dengan jumlah nilai aktual (positif atau negatif).

E. *F1-Score*

F1-Score atau bisa disebut skor F1 merupakan perbandingan rata-rata *precision* dan *recall* yang dibobotkan.

3.2 Analisis Kebutuhan

Pada tahap analisis kebutuhan ini, sistem dianalisis terkait apa saja yang sistem bisa lakukan, mulai dari proses *input* hingga mengeluarkan hasil dari klasifikasi yang dilakukan sistem ini. Ada beberapa proses atau fungsi yang harus ada pada sistem ini, diantaranya sebagai berikut:

3.2.1 Tips Penggunaan Aplikasi

Tips penggunaan aplikasi adalah hal yang penting agar pengguna tidak salah dalam memasukkan citra. Citra mata yang dimasukkan harus di-*crop* hingga mendekati bentuk pola mata untuk menghindari kesalahan prediksi. Ada beberapa aturan untuk memaksimalkan hasil prediksi, yaitu sebagai berikut:

1. Citra mata disarankan untuk menatap ke depan kamera/ melirik ke samping kanan atau kiri.
2. Memastikan pencahayaannya baik agar citra tidak blur.

3.2.2 Upload Citra

Fitur *upload* citra adalah fitur utama pada sistem ini, dimana pengguna nantinya diberikan pilihan untuk mengambil citra langsung dari kamera atau memilih citra yang sudah ada di galeri.

3.2.3 Cropping Citra

Fitur *upload* citra adalah fitur yang fundamental untuk menghasilkan citra yang siap untuk diprediksi. Citra akan di-*crop* sesuai dengan tips penggunaan aplikasi yang telah disampaikan sebelumnya.

3.2.4 Menampilkan Hasil Cropping Citra

Fungsi selanjutnya adalah menampilkan hasil *cropping* untuk memastikan hasil citra sudah tepat dan sesuai dengan tips penggunaan aplikasi, serta memastikan citra sudah di-*crop* hingga mendekati bentuk pola mata.

3.2.5 Melakukan Prediksi

Fungsi ini bertugas untuk melakukan prediksi pada citra yang telah di-*crop* sesuai dengan tips penggunaan aplikasi. Sistem ini akan melakukan prediksi dari model TensorFlow Lite yang sudah dikonversi dari model CNN yang telah di-*training* sebelumnya.

3.2.6 Menampilkan Hasil Prediksi

Fungsi terakhir sistem ini yaitu menampilkan hasil prediksi kepada pengguna berupa keterangan positif atau negatif pterigium, serta menampilkan probabilitasnya.

3.3 Analisis Sistem

Pada tahap analisis sistem, akan dianalisis *tools* apa saja yang akan digunakan dalam mengembangkan sistem ini, yang dapat dilihat pada tabel 3.4:

Tabel 3. 4 Tools yang Digunakan Dalam Pengembangan Sistem

<i>Tools</i>	<i>Version</i>
Laptop Asus A456UR	Window 10
Samsung Galaxy A50	Android 11
Jupyter Notebook	6.4.5

TensorFlow, Keras, TensorFlow Lite Converter	2.11.10
Python	3.9.7
Android Studio	Dolphin (2021.3.1 Patch 1)
Gradle	6.5
Kotlin	1.3.72

3.3.1 TensorFlow

TensorFlow adalah *library open-source* yang dikembangkan oleh tim Google Brain (tim peneliti AI di Google) pada tahun 2015 yang ditulis dengan Bahasa C++ untuk aplikasi *Artificial Intelligence*. TensorFlow memudahkan pembuatan model *machine learning* yang dapat dipakai untuk *deep learning*, *computer vision*, *reinforcement learning*, serta *natural language processing*. TensorFlow bersifat *multiplatform* artinya dapat digunakan di Windows, MacOS, maupun Linux dan dapat dijalankan di CPU (*Central Processing Unit*), GPU (*Graphics Processing Unit*), dan TPU (*Tensor Processing Unit*) (Padilha & Lucena, 2020). Pada penelitian ini versi TensorFlow yang digunakan adalah 2.11.0.

3.3.2 Keras

Untuk membangun model CNN diperlukan *library* Keras. Keras adalah *library Deep Learning* yang dibangun di atas TensorFlow yang sangat minimalis dan simpel dalam pengembangan model *Deep Learning*, dalam studi kasus ini yaitu CNN.

3.3.3 TensorFlow Lite

Agar model dapat dijalankan di *platform* android, maka perlu melakukan *deployment* model CNN yang sudah dibuat. Proses *deployment* ini akan mengkonversi model CNN menjadi sebuah file berekstensi *.tflite*, sehingga model dapat diintegrasikan dengan *platform* android. File inilah yang akan digunakan untuk membuat program android yang mampu melakukan klasifikasi gambar dengan model yang sudah di-*training* sebelumnya.

3.3.4 Android Using Kotlin

Sistem berbasis android ini dibangun menggunakan bahasa pemrograman kotlin versi JDK 11. Kotlin dipilih karena keunggulannya seperti:

1. *Compatibility*

Kotlin sepenuhnya kompatibel dengan JDK 6 (*Java Development Kit*). Ini memastikan aplikasi yang dibangun dengan kotlin dapat berjalan dengan baik pada perangkat android lama. Kotlin pun didukung penuh pengembangannya oleh Android Studio.

2. *Performance*

Kotlin memiliki struktur *bytecode* yang sama dengan java, ini membuat aplikasi yang dibangun dengan kotlin dapat berjalan setara atau bahkan lebih cepat dengan aplikasi yang dibangun dengan java.

3. *Interoperability*

Kotlin dapat digunakan bersamaan dengan bahasa pemrograman java, sehingga kode lama yang dibangun dengan java tidak perlu dibangun ulang dengan bahasa pemrograman kotlin.

4. *Compilation Time*

Kotlin mendukung *incremental compilation*, sehingga proses *build* biasanya sama atau lebih cepat dengan java. *Incremental compilation* sendiri adalah sejenis komputasi inkremental yang diterapkan pada bidang kompilasi, sedangkan kompiler biasa membuat apa yang disebut *clean build* membangun semua modul program, sedangkan kompiler tambahan hanya mengkompilasi ulang bagian program yang dimodifikasi.

3.4 Perancangan Antarmuka

Pada tahap perancangan antarmuka adalah desain *mockup* yang akan diaplikasikan ke dalam sistem. Berikut adalah rancangan antarmuka dari Sistem Deteksi Penyakit Mata Pterigium Berbasis Android Menggunakan Metode *Convolutional Neural Network*.

3.4.1 Halaman Beranda

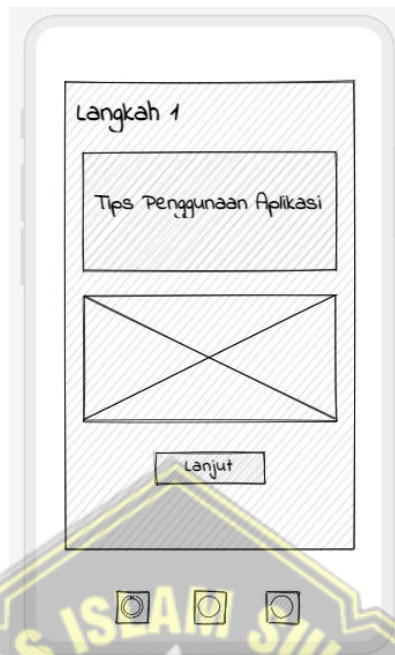
Halaman beranda adalah halaman utama aplikasi, dimana terdapat sedikit penjelasan terkait penyakit pterigium dan penyebabnya. Untuk memulai proses deteksi citra pengguna bisa mengklik tombol “Cek Mata”.



Gambar 3. 14 Halaman Beranda

3.4.2 Halaman Tips Penggunaan Aplikasi

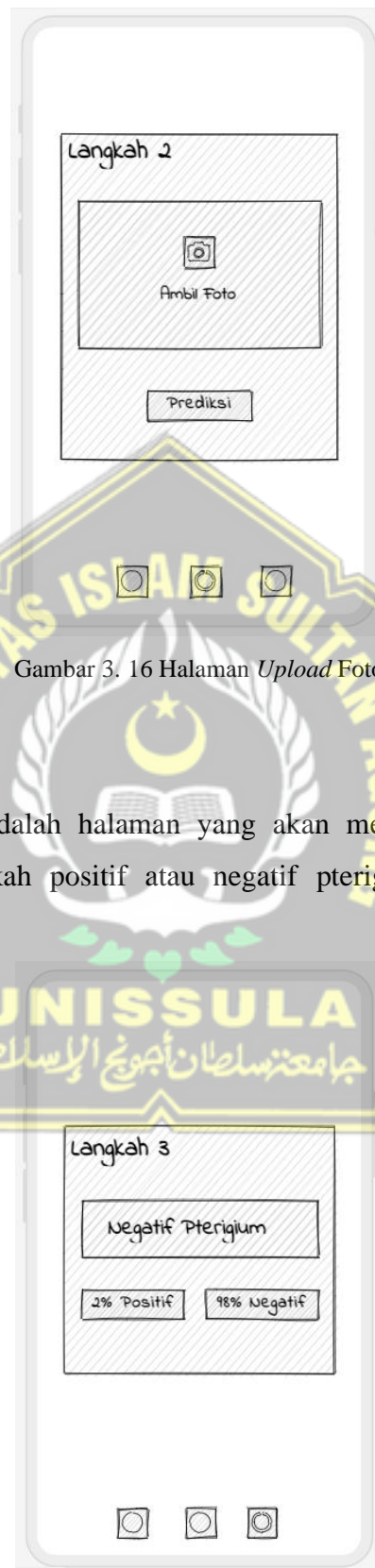
Halaman tips penggunaan aplikasi berisi tips saat mengambil dan *cropping* citra. Citra mata disarankan menghadap lurus depan kamera/ melirik ke kanan atau kiri, dan di-*crop* hingga bentuk pola mata, agar model dapat memprediksi dengan tepat.



Gambar 3. 15 Halaman Tips Penggunaan Aplikasi

3.4.3 Halaman *Upload* Citra

Halaman *upload* citra adalah halaman dimana pengguna akan mengambil citra mata untuk dilakukan prediksi. Pengguna diberi dua pilihan saat pengambilan citra, yaitu apakah mengambil langsung dari kamera atau memilih dari galeri.

Gambar 3. 16 Halaman *Upload Foto*

3.4.4 Halaman *Result*

Halaman *result* adalah halaman yang akan menampilkan hasil prediksi kepada pengguna, apakah positif atau negatif pterigium, serta menampilkan probabilitasnya.

Gambar 3. 17 Halaman *Result*

BAB IV

HASIL DAN ANALISIS PENELITIAN

4.1 Implementasi User Interface

User interface atau tampilan pengguna didesain sesimpel mungkin namun tetap modern dengan tampilan yang luwes dan interaktif demi kenyamanan pengguna. Subbab 4.1.1 – 4.1.4 adalah *user interface* dari sistem berbasis android yang telah dibuat oleh penulis.

4.1.1 Halaman Beranda

Pada halaman beranda ini, pengguna akan melihat halaman beranda aplikasi, dimana terdapat sedikit penjelasan terkait penyakit pterigium dan penyebabnya, serta tombol “Cek Mata” yang mana ketika diklik akan masuk ke halaman proses deteksi mata.



Gambar 4. 1 Halaman Beranda

Gambar 4.1 adalah tampilan halaman beranda yang akan terbuka pertama kali saat aplikasi dijalankan. Untuk memulai deteksi mata, caranya dengan mengklik tombol “Cek Mata”.

4.1.2 Tampilan Tips Penggunaan Aplikasi

Tampilan ini adalah langkah pertama dalam proses deteksi mata. Pengguna akan membaca tips bagaimana tata cara mengambil foto mata yang tepat agar menghasilkan prediksi yang baik.

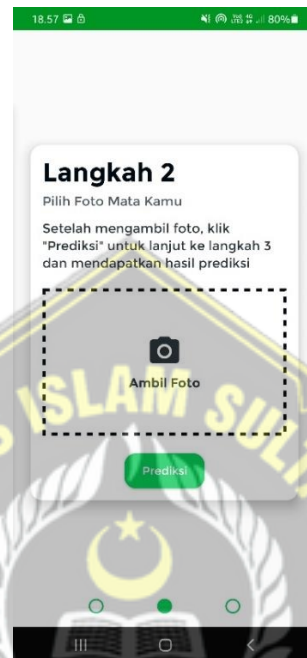


Gambar 4. 2 Tampilan Langkah 1 Tips Penggunaan Aplikasi

Gambar 4.2 adalah tampilan langkah pertama dalam proses deteksi mata. Dijelaskan bahwa untuk mendapatkan hasil prediksi yang baik, maka foto yang diambil sebaiknya melirik ke depan kamera atau melirik ke kanan/ kiri, serta pencahayaan yang baik. Untuk masuk ke langkah 2 pengguna mengklik tombol “Lanjut”.

4.1.3 Tampilan Unggah Foto

Tampilan ini adalah langkah kedua dalam proses deteksi mata. Pengguna akan memasukkan gambar mata yang bisa diambil langsung dari kamera atau galeri.



Gambar 4. 3 Tampilan Langkah 2 Mengambil Foto Mata

Gambar 4.3 adalah tampilan langkah kedua dalam proses deteksi mata. Untuk mengambil foto mata pengguna mengklik ikon kamera, kemudian akan muncul *bottom dialog* yang menampilkan pilihan kamera dan galeri seperti gambar 4.4.

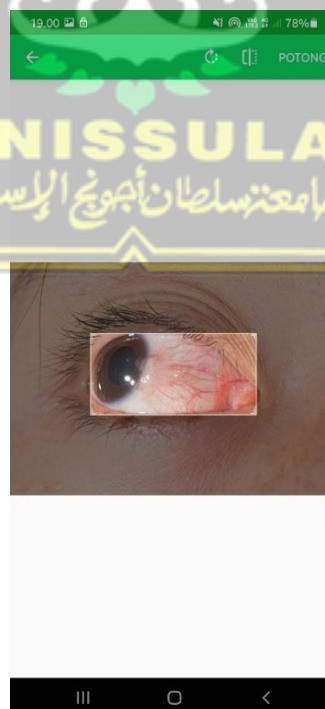


Ambil Foto Mata Kamu



Gambar 4. 4 Opsi Pengambilan Foto Mata

Setelah foto mata diambil, kemudian foto tersebut akan di-*crop* hingga mendekati bentuk pola mata. Selanjutnya foto mata tersebut dapat langsung dilakukan prediksi dengan menekan tombol “Prediksi”.



Gambar 4. 5 Foto Mata yang Diambil dari Galeri



Gambar 4. 6 Hasil *Crop* Foto Mata yang Diambil dari Galeri

4.1.4 Tampilan Hasil Deteksi Citra

Tampilan ini adalah langkah ketiga atau terakhir dalam proses deteksi mata. Pengguna akan melihat hasil deteksi mata dari citra mata yang telah dimasukkan sebelumnya, dengan menampilkan hasil klasifikasi mata, apakah negatif atau positif pterigium.



Gambar 4. 7 Hasil Klasifikasi

Gambar 4.7 adalah hasil deteksi citra mata dari gambar 4.6 yang diambil dari galeri. Hasil deteksi citra tersebut adalah “Positif Pterigium”, dengan presentase positif 99,97% dan negatif 0,03%.

4.2 Proses Penggunaan dan Cara Kerja Sistem

Bagian ini akan menjelaskan proses klasifikasi citra baru yang dimasukkan pengguna dengan model TensorFlow Lite yang telah dibuat.

4.2.1 Unggah Citra

Pada tahap pertama ini, pengguna mengunggah sebuah citra mata baru, dimana terdapat dua pilihan untuk melakukannya, yaitu pilihan pertama mengambil langsung dari kamera atau pilihan kedua memilih dari galeri.

4.2.2 Crop Citra

Setelah pengguna mengambil atau memilih citra yang akan digunakan, selanjutnya citra akan masuk ke area *crop*, dimana pengguna diminta untuk meng-*crop* hingga mendekati pola bentuk mata. Ini perlu dilakukan untuk mendapat hasil klasifikasi yang tepat.

4.2.3 Konversi Citra sebagai *ByteBuffer*

Selanjutnya yaitu sistem akan mengkonversi citra yang sudah di-*crop* menjadi *ByteBuffer*, yang nantinya akan dikonversi menjadi *TensorBuffer*. Ini perlu dilakukan karena *TensorBuffer* hanya dapat dikonversi dari *ByteBuffer*.

4.2.4 Konversi *ByteBuffer* sebagai *TensorBuffer*

Selanjutnya yaitu mengkonversi *ByteBuffer* menjadi *TensorBuffer*. Ini perlu dilakukan karena model TensorFlow Lite hanya dapat memproses *TensorBuffer*, yang nantinya akan didapatkan hasil dari klasifikasi citra yang dimasukkan pengguna.

4.2.5 Klasifikasi Berdasarkan Aktivasi *Softmax*

Lalu selanjutnya citra yang sudah diubah menjadi *TensorBuffer* akan diproses oleh model TensorFlow Lite untuk dilakukan klasifikasi. Hasil klasifikasi dihitung berdasarkan aktivasi *softmax* yaitu menghitung probabilitas dari setiap kelas target atas semua kelas target yang memungkinkan dan akan membantu untuk menentukan kelas target untuk masukkan yang diberikan. Probabilitas keseluruhan target akan berjumlah satu, dengan rentang nilai 0 - 1.

Sebetulnya untuk klasifikasi *binary* aktivasi yang digunakan adalah *sigmoid*, namun setelah dilakukan pengujian sistem mengalami *force close* saat melakukan klasifikasi, sehingga penulis mengganti aktivasinya menjadi *softmax*.

4.3 Pengujian Sistem

Pada tahap ini akan dilakukan pengujian sistem, dimana metode yang dipakai adalah *blackbox testing*. *Blackbox testing* merupakan pengujian perangkat lunak yang bertujuan untuk melakukan pengetesan fungsionalitas sistem tanpa perlu mengetahui struktur kode sistem. Ada beberapa jenis *blackbox testing*, diantaranya *functional testing*, *non-functional testing*, dan *regression testing*. Dalam studi kasus

ini, penulis menggunakan jenis *functional testing*, dimana pengujian ini dilakukan untuk melakukan pengetesan setiap fungsi yang ada pada sistem. Tabel 4.1 adalah hasil *function testing* pada sistem ini.

Tabel 4. 1 Hasil *Blackbox Testing*

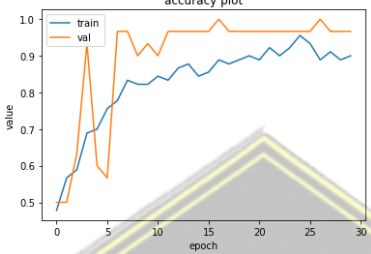
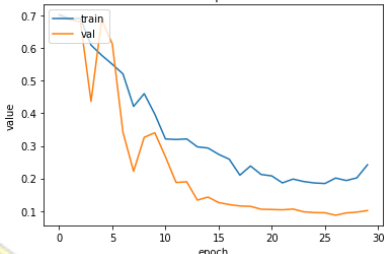
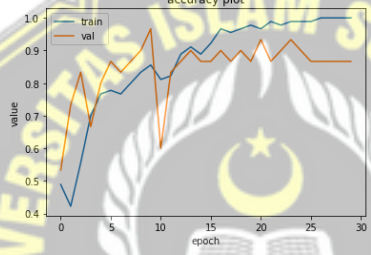
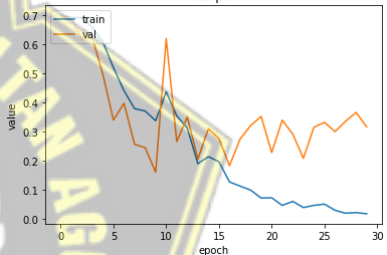
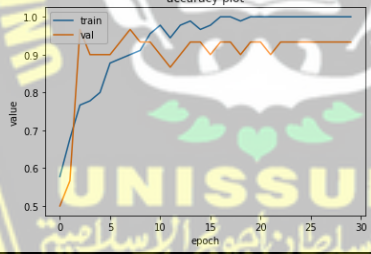
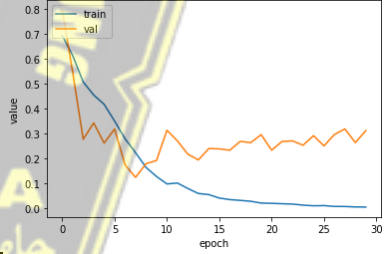
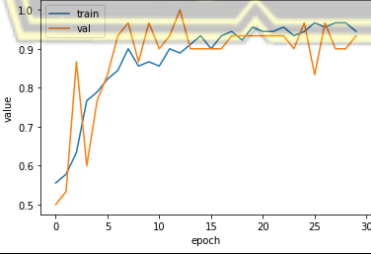
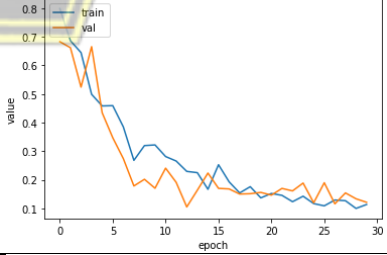
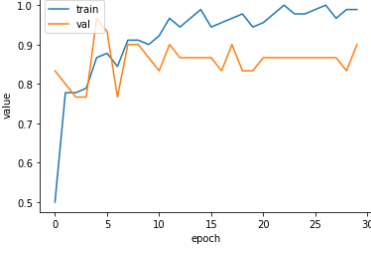
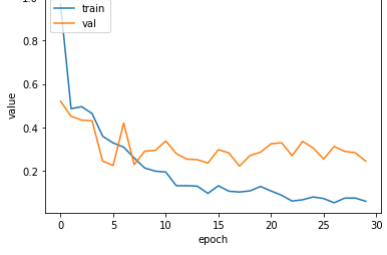
Skenario Pengujian	Kasus Pengujian	Hasil Pengujian	Kesimpulan
Mengambil foto dari kamera	Mengambil foto mata dari kamera	Sesuai	Normal
Memilih foto dari galeri	Memilih foto mata dari galeri	Sesuai	Normal
<i>Upload image</i>	Melakukan <i>upload image</i>	Sesuai	Normal
Melakukan <i>cropping</i> citra	Melakukan <i>cropping</i> citra yang di- <i>upload</i>	Sesuai	Normal
<i>Testing</i> citra negatif pterigium	Melakukan testing citra negatif pterigium	Sesuai	Normal
<i>Testing</i> citra positif pterigium	Melakukan testing citra positif pterigium	Sesuai	Normal
<i>Testing result deteksi</i>	Mengecek hasil deteksi citra	Sesuai	Normal

Dari hasil pengujian yang dilakukan dapat dilihat pada tabel 4.1 bahwa sistem sudah dapat menjalankan semua fungsinya dengan baik. Dimana terdapat tujuh fungsi utama yang menjalani tahap pengujian, untuk memastikan fungsi berjalan dengan semestinya.

4.4 Hasil dan Analisis

4.4.1 Training dan Validation

Tabel 4. 2 Plot Accuracy dan Loss

	Plot	
	Accuracy	Loss
Konfigurasi 1		
Konfigurasi 2		
Konfigurasi 3		
Konfigurasi 4		
Konfigurasi 5		

Tabel 4.2 menunjukkan plot *accuracy* dan *loss* pada masing-masing konfigurasi. Semakin naik grafik pada plot *accuracy*, dan semakin turun grafik pada plot *loss*, serta grafik *training* dan *validation* yang rapat dan saling berdekatan, maka model semakin baik.

Tabel 4. 3 Model Terbaik di Setiap Konfigurasi Pada Tahap Training

	epochs	loss	acc	val-loss	val-acc	learning rate
Konfigurasi 1	17/30	0,2591	0,8889	0,1199	1	1,2500e-05
Konfigurasi 2	10/30	0,3378	0,8556	0,1609	0,9667	1,0000e-04
Konfigurasi 3	3/30	0,5071	0,7667	0,2771	0,9667	1,0000e-04
Konfigurasi 4	13/30	0,2294	0,8889	0,1052	1	2.5000e-05
Konfigurasi 5	5/30	0,3591	0,8667	0,2438	0,9667	5,0000e-05

Tabel 4.3 menunjukkan model terbaik dari setiap konfigurasi pada tahap *training*, dimana dari kelima konfigurasi tersebut terdapat satu konfigurasi terbaik yaitu konfigurasi 4 yang memiliki loss sebesar 0,2294, acc 0,8889, val-loss 0,1052, dan val-acc sebesar 1 atau 100% dan memiliki plot *accuracy* yang naik dan plot *loss* yang menurun, serta grafik *training* dan *validation* yang cukup rapat dan saling berdekatan.

4.4.2 Testing

Selanjutnya adalah tahap pengujian, tujuannya untuk mengetahui tingkat akurasi sistem terhadap data *testing* sejumlah 30 sampel, dimana pengujian dilakukan secara manual pada *smartphone* tipe Samsung A50, Android 11 OS.

Dalam perhitungan akurasi menghasilkan *confusion matrix*, yang digunakan untuk mencari nilai *accuracy*, *precision*, *recall*, dan F1. Perhitungan ini dilakukan terhadap kelima konfigurasi untuk menemukan konfigurasi terbaik dalam melakukan deteksi atau klasifikasi pterigium. Rincian konfigurasi arsitektur CNN telah dijelaskan pada subbab 3.1.5 tabel 3.2. Berikut adalah hasil perhitungan *confusion matrix* dari kelima konfigurasi pada tabel 4.4 – 4.8.

Tabel 4. 4 Hasil Perhitungan *Confusion Matrix* Konfigurasi 1

	<i>Actually Positive</i>	<i>Actually Negative</i>
<i>Predicted Positive</i>	15 (TP)	0 (FP)
<i>Predicted Negative</i>	0 (FN)	15 (TN)

Tabel 4. 5 Hasil Perhitungan *Confusion Matrix* Konfigurasi 2

	<i>Actually Positive</i>	<i>Actually Negative</i>
<i>Predicted Positive</i>	15 (TP)	10 (FP)
<i>Predicted Negative</i>	0 (FN)	5 (TN)

Tabel 4. 6 Hasil Perhitungan *Confusion Matrix* Konfigurasi 3

	<i>Actually Positive</i>	<i>Actually Negative</i>
<i>Predicted Positive</i>	15 (TP)	14 (FP)
<i>Predicted Negative</i>	0 (FN)	1 (TN)

Tabel 4. 7 Hasil Perhitungan *Confusion Matrix* Konfigurasi 4

	<i>Actually Positive</i>	<i>Actually Negative</i>
<i>Predicted Positive</i>	15 (TP)	0 (FP)
<i>Predicted Negative</i>	0 (FN)	15 (TN)

Tabel 4. 8 Hasil Perhitungan *Confusion Matrix* Konfigurasi 5

	<i>Actually Positive</i>	<i>Actually Negative</i>
<i>Predicted Positive</i>	15 (TP)	1 (FP)
<i>Predicted Negative</i>	0 (FN)	14 (TN)

Berdasarkan data yang di dapat dari *confusion matrix* pada tabel 4.4 - 4.8, maka dapat diketahui nilai *accuracy*, *precision*, *recall*, dan F1 dari setiap konfigurasi, berdasarkan formula atau rumus yang ada pada subbab 3.1.8, sehingga didapatkan hasil seperti pada tabel 4.9.

Tabel 4. 9 Performa Testing Pada *Platform Mobile*

	<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>	<i>F1</i>
Konfigurasi 1	1	1	1	1
Konfigurasi 2	0,67	0,6	1	0,75
Konfigurasi 3	0,53	0,52	1	0,68
Konfigurasi 4	1	1	1	1
Konfigurasi 5	0,97	0,94	1	0,97

Berdasarkan hasil pengujian yang telah dilakukan, terdapat dua konfigurasi yang memiliki nilai sama yaitu konfigurasi 1 dan konfigurasi 4, dengan nilai akurasi, *precision*, *recall*, dan F1 masing-masing sebesar 1 atau 100%. Pada konfigurasi 2 akurasi sebesar 0,67 atau 67% dan konfigurasi 3 sebesar 0,53 atau 53%, sedangkan pada konfigurasi 5 memiliki akurasi yang cukup tinggi yaitu sebesar 0,97 atau 97%.

Tabel 4. 10 Perbandingan Akurasi *Training*, *Validation*, dan *Testing*

	<i>Training</i>	<i>Validation</i>	<i>Testing</i>
Konfigurasi 1	0,8889	1	1
Konfigurasi 4	0,8889	1	1

Tabel 4.10 adalah perbandingan akurasi *training*, *validation*, dan *testing* pada konfigurasi 1 dan konfigurasi 4, yaitu memiliki akurasi yang sama pada *training* sebesar 88,89%, *validation* 100%, dan *testing* 100%.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil penelitian yang telah dilakukan penulis, dapat ditarik kesimpulan bahwa sistem deteksi atau klasifikasi pterigium dapat diimplementasikan dengan cukup baik terhadap sistem berbasis android, dan berikut beberapa poin kesimpulan yang penulis dapat selama penelitian ini:

1. *Convolutional Neural Network* dapat diimplementasikan untuk melakukan deteksi citra pterigium dengan hasil yang cukup baik pada sistem berbasis android.
2. Pada tahap *training*, konfigurasi terbaik terdapat pada konfigurasi 4, yaitu model terbaik pada epochs ke-13, dengan loss sebesar 0,2294, acc 0,8889, val-loss 0,1052, dan val-acc 1.
3. Dari hasil pengujian melalui *smartphone*, sistem deteksi atau klasifikasi pterigium memiliki akurasi yang tinggi yaitu 100% pada konfigurasi 1 (6 *convolution layer*, 2 *pooling layer*, dan 1 *hidden layer*) dan konfigurasi 4 (6 *convolution layer*, 2 *pooling layer*, 1 *hidden layer*, dan 3 *dropout layer*).
4. Konfigurasi 1 dan konfigurasi 4 memiliki akurasi *training*, validasi dan pengujian yang sama yaitu akurasi *training* sebesar 88,89%, serta akurasi validasi dan pengujian sebesar 1 atau 100%, yang artinya tidak ada perubahan akurasi validasi pada saat tahap *training* dan akurasi *testing* setelah *deployment* model.

5.2 Saran

Berdasarkan penelitian yang sudah ada, untuk penelitian yang akan datang penulis menyarankan:

1. Dari hasil penelitian ini, sistem memiliki akurasi pengujian sebesar 100%, namun sistem tetap akan melakukan prediksi atau klasifikasi pada citra bukan mata. Diharapkan untuk penelitian selanjutnya sistem dapat

dikembangkan, sehingga hanya melakukan prediksi atau klasifikasi pada citra mata saja.

2. Saat ini sistem hanya tersedia pada perangkat android, sehingga diharapkan dapat dikembangkan pada perangkat IOS dan atau sistem berbasis *website*.



DAFTAR PUSTAKA

- Abdani, S. R., Zulkifley, M. A., & Hussain, A. (2019). Compact Convolutional Neural Networks for Pterygium Classification using Transfer Learning. *Proceedings of the 2019 IEEE International Conference on Signal and Image Processing Applications, ICSIPA 2019, September*, 140–143. <https://doi.org/10.1109/ICSIPA45851.2019.8977757>
- Anida, M., & Wibowo, A. (2017). Wanita Usia 48 Tahun dengan Pterigium Stadium 2. *J Medula Unila*, 7, 48–51.
- Dewi, N., & Ismawan, F. (2021). Implementasi Deep Learning Menggunakan Cnn Untuk Sistem Pengenalan Wajah. *Faktor Exacta*, 14(1), 34. <https://doi.org/10.30998/faktorexacta.v14i1.8989>
- Erry, E., Mulyani, U. A., & Susilowati, D. (2011). Distribusi dan Karakteristik Pterigium di Indonesia. *Buletin Penelitian Sistem Kesehatan*, 14(1). <https://doi.org/10.22435/bpsk.v14i1.Jan.2311>
- Fahrizal, Reynaldi, F. O., & Hikmah, N. (2020). Implementasi Machine Learning pada Sistem PETS Identification Menggunakan Python Berbasis Ubuntu. *Journal of Information System, Informatics and Computing*, 4(1), 86–91.
- Gerald, C., & Lubis, C. (2020). Pendeteksian Dan Pengenalan Jenis Mobil Menggunakan Algoritma You Only Look Once Dan Convolutional Neural Network. 197–199. <https://journal.untar.ac.id/index.php/jiksi/article/view/11495>
- Ginting, D. N. B., & Arjasakusuma, S. (2021). Pemetaan Lamun Menggunakan Machine Learning Dengan Citra Planetscope Di Nusa Lembongan. *Jurnal Kelautan Tropis*, 24(3), 323–332. <https://doi.org/10.14710/jkt.v24i3.11180>
- Hania, A. A. (2017). Mengenal Artificial Intelligence, Machine Learning, & Deep Learning. *Jurnal Teknologi Indonesia*, 1(June), 1–6. <https://amt-it.com/mengenal-perbedaan-artificial-intelligence-machine-learning-deep-learning/>
- Harani, N. H., Prianto, C., & Hasanah, M. (2019). Deteksi Objek Dan Pengenalan Karakter Plat Nomor Kendaraan Indonesia Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Python. *Jurnal Teknik Informatika*, 11(3), 47–53.
- Ilahiyah, S., & Nilogiri, A. (2018). Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. *JUSTINDO (Jurnal Sistem dan Teknologi Informasi Indonesia)*, 3(2), 49–56.
- Kholik, A. (2021). Klasifikasi Menggunakan Convolutional Neural Network (Cnn) Pada Tangkapan Layar Halaman Instagram. *Jdmsi*, 2(2), 10–20.
- Kusuma, P. D., Setianingsih, C., Elektro, F. T., Telkom, U., Pakar, S., & Transform, H. (2018). Deteksi Penyakit Pterigium Menggunakan Hough Transform Dan Forward Chaining Detection of Pterigium Disease Using Hough Transform and. *eProceedings* ..., 5(3), 6126–6133. <https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/7982>

- Padilha, T. P. P., & Lucena, L. E. A. de. (2020). A Systematic Review About Use of TensorFlow for Image Classification and Word Embedding in the Brazilian Context. *Academic Journal on Computing, Engineering and Applied Mathematics*, 1(2), 24–27. <https://doi.org/10.20873/uft.2675-3588.2020.v1n2.p24-27>
- Pambudi, H. K., Kusuma, P. G. A., Yulianti, F., & Julian, K. A. (2020). Prediksi Status Pengiriman Barang Menggunakan Metode Machine Learning. *Jurnal Ilmiah Teknologi Infomasi Terapan*, 6(2), 100–109. <https://doi.org/10.33197/jitter.vol6.iss2.2020.396>
- Prakash, R. M., & Kumari, R. S. S. (2019). Classification of MR brain images for detection of tumor with transfer learning from pre-trained CNN models. *2019 International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET 2019, May 2022*, 508–511. <https://doi.org/10.1109/WiSPNET45539.2019.9032811>
- Putra, I. W. S. E., Wijaya, A. Y., & Soelaiman, R. (2017). Klasifikasi Citra Menggunakan Convolutional Neural Network (Cnn) Pada Caltech 101. *Jurnal Teknik ITS*, 5(1), 76. <http://repository.its.ac.id/48842/>
- Rany, N. (2017). Hubungan Lingkungan Kerja Dan Perilaku Nelayan Terhadap Kejadian Pterigium Di Desa Kemang Kecamatan Pangkalan Kuras Kabupaten Pelalawan. *Jurnal Kesehatan Komunitas*, 3(4), 153–158. <https://doi.org/10.25311/keskom.vol3.iss4.203>
- Reddy, R. V. K., Rao, B. S., & Raju, K. P. (2019). Handwritten Hindi Digits Recognition Using Convolutional Neural Network with RMSprop Optimization. *Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, ICICCS 2018, June*, 45–51. <https://doi.org/10.1109/ICCONS.2018.8662969>
- Romario, M. H., Ihsanto, E., & Kadarina, T. M. (2020). Sistem Hitung dan Klasifikasi Objek dengan Metode Convolutional Neural Network. *Jurnal Teknologi Elektro*, 11(2), 108. <https://doi.org/10.22441/jte.2020.v11i2.007>
- Rozaqi, A. J., Sunyoto, A., & Arief, M. rudyanto. (2021). Deteksi Penyakit Pada Daun Kentang Menggunakan Pengolahan Citra dengan Metode Convolutional Neural Network. *Creative Information Technology Journal*, 8(1), 22. <https://doi.org/10.24076/citec.2021v8i1.263>
- Saha, S. (2018). *A Comprehensive Guide to Convolutional Neural Networks*. <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>
- Somba, S. M., Saerang, J. S. M., & Tongku, Y. (2018). Gambaran Pengetahuan Masyarakat yang Bekerja sebagai Nelayan tentang Pterigium di Desa Kapitu Kabupaten Minahasa Selatan. *e-CliniC*, 6(2). <https://doi.org/10.35790/ec1.6.2.2018.21992>
- Xu, W., Jin, L., Zhu, P. Z., He, K., Yang, W. H., & Wu, M. N. (2021). Implementation and Application of an Intelligent Pterygium Diagnosis System Based on Deep Learning. *Frontiers in Psychology*, 12(October), 1–8. <https://doi.org/10.3389/fpsyg.2021.759229>
- Zamani, N. S. M., Zaki, W. M. D. W., Huddin, A. B., Hussain, A., Mutalib, H. A., & Ali, A. (2020). Automated pterygium detection using deep neural network.

IEEE Access, 8, 191659–191672.

<https://doi.org/10.1109/ACCESS.2020.3030787>

Zhang, Q., Zhang, M., Chen, T., Sun, Z., Ma, Y., & Yu, B. (2019). Recent advances in convolutional neural network acceleration. *Neurocomputing*, 323, 37–51. <https://doi.org/10.1016/j.neucom.2018.09.038>

Zufar, M., & Setiyono, B. (2017). Convolutional Neural Networks Untuk Pengenalan Wajah Secara Real-Time. *Jurnal Sains dan Seni ITS*, 5(2), 128862.

