PENERAPAN MODEL ARSITEKTUR *CONVOLUTIONAL NEURAL NETWORK (CNN)* DALAM DETEKSI DINI PENYAKIT MATA KATARAK

(Skripsi)

Oleh :

APRILIA NUR ILAHY

PROGRAM STUDI S1 TEKNIK INFORMATIKA

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS LAMPUNG

2025

PENERAPAN MODEL ARSITEKTUR *CONVOLUTIONAL NEURAL NETWORK (CNN)* DALAM DETEKSI DINI PENYAKIT MATA KATARAK

Oleh

APRILIA NUR ILAHY

Skripsi

Sebagai Salah Satu Syarat Untuk Mencapai Gelar
SARJANA TEKNIK

Pada

Program Studi S1 Teknik Informatika
Jurusan Teknik Elektro
Fakultas Teknik Universitas Lampung

FAKULTAS LAMPUNG
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

ABSTRAK

PENERAPAN MODEL ARSITEKTUR *CONVOLUTIONAL NEURAL NETWORK (CNN)* DALAM DETEKSI DINI PENYAKIT MATA KATARAK

Oleh

APRILIA NUR ILAHY

Katarak merupakan penyebab utama kebutaan di Indonesia, khususnya pada kelompok usia lanjut. Deteksi dini menjadi langkah krusial untuk mencegah kehilangan penglihatan permanen. Penelitian ini bertujuan untuk membangun dan mengevaluasi sistem klasifikasi citra anterior mata menggunakan arsitektur *Convolutional Neural Network (CNN) EfficientNetB0* dengan pendekatan *transfer learning* dan *fine-tuning*. Dataset yang digunakan terdiri dari 612 citra anterior mata berwarna, terdiri atas 306 citra katarak dan 306 citra normal, yang dibagi ke dalam subset pelatihan, validasi, dan pengujian. Augmentasi citra diterapkan hanya pada data pelatihan guna meningkatkan keragaman dan kemampuan generalisasi model. Penelitian dilakukan melalui dua tahap evaluasi utama, yaitu tuning hyperparameter terhadap empat jenis fungsi optimasi (Adagrad, SGD, RMSprop, dan Adam), serta validasi silang 5-fold. Hasil tuning menunjukkan bahwa optimizer Adam memberikan performa paling optimal dengan akurasi pelatihan sebesar 0,90, akurasi validasi 0,87, loss pelatihan 0,33, dan loss validasi 0,46. Berdasarkan kriteria evaluasi, hasil tersebut masuk dalam kategori akurasi optimal (≥ 0,80) dan loss optimal (≤ 0,50). Sebaliknya, Adagrad dan SGD menunjukkan akurasi rendah (< 0,60) dan loss tinggi (> 0,65), sedangkan RMSprop berada pada kategori cukup. Pada tahap evaluasi menggunakan 5-*fold cross-validation*, diperoleh rata-rata akurasi sebesar 96,56%, yang menunjukkan generalisasi model sangat baik. Hasil terbaik diperoleh pada Fold ke-1 dengan akurasi 100%. Pada evaluasi akhir, untuk kelas Katarak diperoleh *precision* 100%, *recall* 94,29%, dan *F1-Score* 97,09%. Untuk kelas Normal, *precision* 94,59%, *recall* 100%, dan *F1-Score* 97,23%. Seluruh nilai metrik berada pada rentang 0,90–1,00, yang dikategorikan sebagai "Sangat Baik".

Kata Kunci: Deteksi Katarak, *Convolutional Neural Network, EfficientNetB0,* Validasi Silang.

ABSTRACT

IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK (CNN) ARCHITECTURE FOR EARLY DETECTION OF CATARACT EYE DISEASE

By

APRILIA NUR ILAHY

Cataract is a leading cause of blindness in Indonesia, particularly among the elderly. Early detection is crucial to prevent permanent vision loss. This study aims to develop and evaluate an anterior eye image classification system using the Convolutional Neural Network (CNN) architecture EfficientNetB0 with a transfer learning and fine-tuning approach.The dataset used consists of 612 color anterior eye images, comprising 306 cataract and 306 normal images, which were divided into training, validation, and testing subsets. Data augmentation was applied only to the training data to enhance image diversity and improve model generalization. The research was conducted in two main evaluation stages: hyperparameter tuning on four optimization functions (Adagrad, SGD, RMSprop, and Adam), and 5-fold cross-validation. The tuning results showed that the Adam optimizer produced the best performance, achieving a training Accuracy of 0.90, validation Accuracy of 0.87, training loss of 0.33, and validation loss of 0.46. According to evaluation criteria, these results fall into the categories of optimal Accuracy (≥ 0.80) and optimal loss (≤ 0.50). In contrast, Adagrad and SGD produced low Accuracy (< 0.60) and high loss (> 0.65), while RMSprop was considered moderate.In the 5-fold cross-validation stage, the model achieved an average Accuracy of 96.56%, indicating excellent generalization. The best result was obtained in Fold 1 with an Accuracy of 100%. Final evaluation for the Cataract class showed a precision of 100%, recall of 94.29%, and F1-Score of 97.09%. For the Normal class, the precision was 94.59%, recall 100%, and F1-Score 97.23%. All evaluation scores fell within the 0.90–1.00 range, categorized as "Excellent".

***Keywords**: Cataract Detection, Convolutional Neural Network, EfficientNetB0, Cross-Validation.*

Judul	:	**PENERAPAN MODEL ARSITEKTUR *CONVOLUTIONAL NEURAL NETWORK (CNN)* DALAM DETEKSI DINI PENYAKIT MATA KATARAK**
Nama Mahasiswa	:	Aprilia Nur Hahy
Nomor Pokok Mahasiswa	:	**1815061031**
Jurusan	:	**Teknik Elektro**
Fakultas	:	**Teknik**

MENYETUJUI

1. Komisi Pembimbing

Pembimbing Utama	Pembimbing Pendamping
Yessi Mulyani, S.T.,M.T.	**Rio Ariestia Pradipta, S.Kom. M.T.I**
NIP. 197312262000122001	NIP. 19860323 201903 1013

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Informatika
Herlinawati, S.T.,M.T.	**Yessi Mulyani, S.T.,M.T.**
NIP. 197103141999032001	NIP. 197312262000122001

MENGESAHKAN

1. Tim Penguji

Ketua : **Yessi Mulyani, S.T.,M.T.** .....

Sekretaris : **Rio Ariestia Pradipta, S.Kom. M.T.I** .....

Penguji : **Ir. Muhammad Komarudin, S.T.,M.T** .....

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.
NIP. 19750928 200112 1 002

Tanggal Lulus Ujian Skripsi : **16 Juni 2025**

SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa skripsi yang saya buat dengan judul "Penerapan Model Arsitektur Convolutional Neural Network (CNN) dalam Deteksi Dini Penyakit Mata Katarak" dibuat tidak berdasarkan karya yang pernah dilakukan oleh orang lain. Bahwa karya ini tidak terdapat karya lain atau pendapat yang ditulis atau diterbitkan orang lain, kecuali tertulis diacu dalam naskah ini sebagaimana yang disebutkan di dalam Daftar Pustaka. Selain itu saya menyatakan pula bahwa skripsi ini dibuat oleh saya sendiri.

Apabila pernyataan saya tidak benar, maka saya bersedia dikenai sangsi sesuai dengan hukum yang berlaku.

Bandar Lampung, 18 Juni 2025

Yang membuat pernyataan,

Aprilia Nur Ilahy

NPM. 1815061031

RIWAYAT HIDUP

Aprilia Nur Ilahy dilahirkan di Bawang Putih, Lampung Timur pada tanggal 1 April 1999. Penulis merupakan anak kedua dari dua bersaudara pasangan Bapak Ngatijo dan Ibu Siti Masrikah. Penulis pertama kali mengenyam pendidikan sekolah dasar di SDN 2 Hargomulyo dan lulus pada tahun 2011 dilanjutkan menempuh pendidikan ke SMP Negeri 2 Sekampung dan lulus pada tahun 2014.

Kemudian penulis mengeyam pendidikan sekolah menengah atas di SMA Negeri 3 Metro Jurusan IPA dan lulus pada tahun 2017.

Pada tahun 2018, penulis memutuskan untuk melanjutkan pendidikan di Teknik Informatika Universitas Lampung dan diterima melalui jalur SBMPTN (Seleksi Bersama Masuk Perguruan Tinggi Negeri). Selama menjadi mahasiswa, penulis aktif mengikuti organisasi Himpunan Mahasiswa Teknik Elektro (HIMATRO) Fakultas Teknik Universitas Lampung. Penulis menjadi anggota Divisi Kaderisasi dan Pengembangan Organisasi di tahun 2018 – 2019. Selain itu, penulis juga sempat menjadi anggota lembaga riset Unila Robotika dan Otomasi (URO) divisi KRSBI yang berfokus pada perancangan dan pengembangan robot sepak bola beroda dan sempat mengikuti Kontes Robot Indonesia tingkat Nasional pada tahun 2019. Pada tahun. Pada tahun 2020 penulis pernah menjabat sebagai Bendahara Umum Forum Komunikasi Himpunan Mahasiswa Elektro Indonesia (FKHMEI) wilayah V. Kemudian pada tahun 2019 – 2020, penulis berkesempatan menjadi asisten di Laboratorium Teknik Digital serta menjadi asisten Praktikum Teknik Digital. Penulis sempat mengikuti kegiatan Studi Independen di Bangkit Academy pada tahun 2023 dan mengambil kelas Mobile Development, bersama rekan-rekan 1 tim berhasil menyelesaikan project akhir membuat *Aplikasi Deteksi Dini Katarak bernama ClearView*. Penulis melakukan kerja praktik di UPT Perpustakaan Universitas Lampung dengan laporan — *Pemetaan Basis Data dari Sistem Informasi E-Library Universitas Lampung ke Aplikasi Otomasi Perpustakaan INLislite Versi 3*.2 pada tahun 2023.

Allah tidak akan mengubah keadaan suatu kaum sampai mereka mengubah keadaan diri mereka sendiri."

(QS. Ar-Ra'd: 11)

"Jangan pernah takut untuk mencoba apalagi takut gagal, karena jika kamu percaya, bisa jadi kegagalan itu akan membuat suksesmu jadi lebih berkelas."

Aprilia Nur Ilahy

Alhamdulillahirabbil'alamin, Puji Syukur kepada Allah Subhanahu Wa Ta'ala atas segala rahmat dan hidayah-Nya serta tak lupa sholawat dan salam kepada Nabi Muhammad Shallallahu Alaihi Wa Sallam yang selalu menjadi suri tauladan di setiap kehidupan

DENGAN TULUS KUPERSEMBAHKAN KARYA INI
TERUNTUK :

"Kedua orang tuaku Ibu Siti Masrikah dan Bapak Ngatijo"

"Suamiku Tercinta Mas Achmad Sultoni"

"Calon Anakku Tersayang yang masih di perut ibu"

"Mamasku M. Abdul Rachman Wahid dan Mbakku Mala Sari"

"Sahabat-sahabat terbaikku, Titik Rahma Diyanti, Indah Wulan Sari, Putri Kurnia Sari, Rifki Romadhon, Sabria Gilang, Kiki Andrianto, Rahmad Andrianto"

"Seluruh dosen, keluarga angkatan 2018, dan civitas Jurusan Teknik Elektro Universitas Lampung"

Juga kepada semua orang yang bertanya, "Kapan Wisuda?", "Kok Ngga Lulus lulus?", dengan bangga kupersembahkan karya ini.

Dan Terimakasih untuk Diri Sendiri, Aprilia Nur Ilahy, dengan izin Allah serta doa suami dan orang orang terkasih, KAMU BISA.

Serta, almamater saya UNIVERSITAS LAMPUNG

KATA PENGANTAR

Puji syukur kepada Allah SWT yang atas rahmat serta hidayahnya dapat membantu penulis menyelesaikan skripsi yang berjudul PENERAPAN MODEL ARSITEKTUR *CONVOLUTIONAL NEURAL NETWORK (CNN)* DALAM DETEKSI DINI PENYAKIT MATA KATARAK

Selama mengerjakan skripsi ini, penulis mendapatkan banyak bantuan dan dukungan dari berbagai pihak. Oleh karena itu pada kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Bapak Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung.
2. Ibu Herlinawati, S.T., M.T., selaku Ketua Jurusan Teknik Elektro Fakultas Tekni Universitas Lampung.
3. Ibu Yessi Mulyani S.T., M.T., selaku Ketua Jurusan Teknik Elektro Universitas Lampung serta selaku penguji utama atas bimbingan serta masukan dalam pengerjaan skripsi.
4. Ibu Yessi Mulyani S.T., M.T., selaku pembimbing utama atas bimbingan, saran, motivasi, serta kesabaran kepada penulis selama penyelesaian skripsi.
5. Bapak Rio Ariestia P, S. Kom. M.T.I selaku pembimbing pendamping atas bimbingan, pengarahan, serta kesabaran kepada penulis selama penyelesaian skripsi yang telah membimbing penulis dari awal perkuliahan hingga dinyatakan sebagai Sarjana Teknik.
6. Bapak Ir. M. Komarudin, S.T.,M.T, selaku dosen penguji atas saran, masukkan serta arahannya kepada penulis.
7. Seluruh Dosen Teknik Elektro dan Informatika atas bimbingan dan kesabarannya dalam mendidik penulis.
8. Suamiku tercinta, mas Achmad Sultoni, trimakasih atas semua kasih sayang, cinta, do'a serta motivasi hingga penulis mampu menyelesaikan skripsi ini.

9. Calon anakku tersayang yang masih didalam perut ibu, lihat nak, ibumu bisa.
10. Seluruh keluarga tercinta, Ayah, Ibu, Mamas dan Mbak yang selalu memberikan motivasi serta dukungan doa sejak awal hingga penulis dapat menyelesaikan skripsi ini.
11. Sahabat-sahabatku dari kecil, Titik Rahma Diyanti, Indah Wulan Sari, Putri Kurnia Sari, Rifki Romadhon, yang selalu memberikan support yang sangat berharga
12. Sabria Gilang, Kiki Andrianto, Rahmad Andrianto atas semua support sistem yang pernah diberikan
13. Uji Khaidah, Sawitri Fina Kartika, Indria Agustina atas dukungan dan kebersamaan selama perkuliahan
14. Keluarga Besar Teknik Elektro dan Teknik Informatika angkatan 2018, terimakasih atas kehangatan dan semangat yang selama ini diberikan.
15. Mbak Rika, Mbak Nurul serta seluruh staff jurusan Teknik Elektro yang telah banyak membantu.
16. Semua pihak yang turut serta membantu dalam menyelesaikan penelitian ini yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna karena keterbatasan waktu dan pengetahuan yang dimiliki penulis. Oleh karena itu segala bentuk kritik maupun saran yang sifatnya membangun akan sangat diharapkan kedepannya. Akhir kata, semoga skripsi ini dapat bermanfaat dan berguna bagi semua pihak.

Bandar Lampung, 18 Juni 2025
Penulis,

Aprilia Nur Ilahy
NPM. 1815061031

DAFTAR ISI

halaman

DAFTAR GAMBAR

DAFTAR TABEL

BAB I. PENDAHULUAN

1.1 Latar Belakang

Katarak merupakan penyebab utama kebutaan di seluruh dunia, termasuk di Indonesia. Kondisi ini ditandai dengan kekeruhan pada lensa mata yang seharusnya transparan, sehingga menghambat masuknya cahaya ke retina secara optimal. Dampaknya, penderita mengalami gangguan penglihatan seperti pandangan kabur, sensitivitas terhadap cahaya, hingga kehilangan penglihatan secara bertahap. Menurut data dari Organisasi Kesehatan Dunia (WHO), katarak bertanggung jawab atas sekitar 51% kasus kebutaan secara global. Sebanyak 2,2 miliar orang mengalami gangguan penglihatan jarak dekat atau jauh. Pada setidaknya 1 miliar atau hampir setengah dari kasus ini, gangguan penglihatan dapat dicegah atau belum ditangani. Di antara 1 miliar orang tersebut, kondisi utama yang menyebabkan gangguan penglihatan jarak jauh atau kebutaan adalah katarak (94 juta), kelainan refraksi (88,4 juta), degenerasi makula terkait usia (8 juta), glaukoma (7,7 juta), retinopati diabetik (3,9 juta). Berdasarkan survei *Rapid Assessment of Avoidable Blindness (RAAB)* tahun 2014-2016, prevalensi kebutaan di Indonesia mencapai 3%, dengan katarak sebagai penyebab utama (81%). Faktor utama yang mempengaruhi hal ini adalah keterbatasan akses terhadap pelayanan kesehatan dan kurangnya upaya deteksi dini. Oleh karena itu, diperlukan sistem pendeteksian katarak secara dini yang bersifat cepat, akurat, dan mudah dijangkau oleh masyarakat.[1]

Perkembangan teknologi dalam bidang medis, khususnya dalam analisis citra medis, telah memberikan kontribusi besar dalam upaya deteksi dini berbagai penyakit, termasuk katarak. Salah satu teknologi yang semakin banyak digunakan adalah *Convolutional Neural Network (CNN)*. CNN adalah bagian dari *deep learning* yang sangat efektif untuk pengenalan pola dan klasifikasi gambar. CNN

telah terbukti memiliki kinerja yang sangat baik dalam berbagai aplikasi, mulai dari pengenalan wajah hingga diagnosis medis.

Penerapan CNN dalam deteksi dini penyakit katarak memiliki potensi besar untuk meningkatkan akurasi diagnosis dan efisiensi waktu. Beberapa penelitian sebelumnya telah menunjukkan bahwa model CNN dapat digunakan untuk mendeteksi katarak dengan tingkat akurasi yang tinggi. Sebagai contoh, penelitian oleh Ramadhani dkk [2] membuktikan bahwa algoritma CNN dapat dimanfaatkan untuk mendeteksi katarak pada gambar mata. Model CNN yang mereka kembangkan mampu membedakan antara mata normal dan mata yang mengalami katarak secara otomatis. Hasil penelitian ini menunjukkan bahwa CNN memiliki potensi besar dalam membantu proses diagnosis berbasis citra medis.

Penelitian yang dilakukan oleh Simanjuntak dkk [3] menggunakan pendekatan yang sedikit berbeda, yaitu dengan mengklasifikasikan gambar mata ke dalam empat kategori berdasarkan tingkat keparahan katarak. Mereka juga membandingkan beberapa arsitektur CNN populer, dan hasilnya menunjukkan bahwa pendekatan ini dapat mendukung klasifikasi yang lebih detail dan spesifik.

Sementara itu, Mullangi dkk [4] menggunakan model CNN pralatih seperti DenseNet dan ResNet untuk mendeteksi katarak melalui citra fundus. Penelitian ini juga memperkenalkan arsitektur baru bernama CatCNNNet yang dirancang agar efisien dan dapat digunakan langsung pada perangkat klinis maupun *mobile*. Pendekatan ini menunjukkan pentingnya kombinasi antara performa tinggi dan efisiensi komputasi dalam pengembangan sistem deteksi otomatis.Selain itu, metode CNN juga memungkinkan integrasi dengan sistem informasi rumah sakit untuk mempermudah proses diagnosis dan penanganan lebih lanjut.

Penelitian terdahulu menunjukkan bahwa metode CNN tidak hanya dapat digunakan untuk mendeteksi katarak tetapi juga untuk mengklasifikasikan tingkat keparahan penyakit. Walaupun beragam studi telah membuktikan efektivitas CNN dalam mendeteksi katarak, sebagian besar masih terbatas pada klasifikasi biner dan belum sepenuhnya mempertimbangkan berbagai tantangan praktis seperti pencahayaan yang bervariasi, sudut pengambilan gambar, serta kualitas citra yang

beragam. Di samping itu, masih sedikit penelitian yang secara khusus mengevaluasi performa arsitektur CNN ringan yang tetap efisien dan akurat untuk digunakan dalam perangkat dengan sumber daya terbatas.

Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi model *Convolutional Neural Network (CNN)* berbasis arsitektur *EfficientNetB0* untuk mendeteksi katarak secara otomatis dari citra mata. Pendekatan ini didesain agar tidak hanya menghasilkan akurasi klasifikasi yang tinggi, tetapi juga efisien secara komputasi dan siap diterapkan pada lingkungan nyata, seperti layanan kesehatan primer atau alat bantu diagnosis awal.

Model CNN dilatih menggunakan dataset citra mata beranotasi, yang telah melalui proses augmentasi dan pembagian sistematis ke dalam subset pelatihan, validasi, dan pengujian. Selain itu, dilakukan eksperimen hyperparameter tuning terhadap berbagai jenis optimizer (*Adam, SGD, RMSprop*, dan *Adagrad*) serta variasi nilai *learning rate* (1e-3 hingga 1e-5) untuk memperoleh konfigurasi pelatihan terbaik. Evaluasi performa model dilakukan secara menyeluruh dengan validasi silang 5-*fold* dan confusion matrix untu perhitungan metrik akurasi, presisi, *recall*, *F1-Score*, , ROC-AUC.

Dengan pendekatan eksperimental yang komprehensif dan penggunaan arsitektur modern, penelitian ini diharapkan dapat memberikan kontribusi lebih dibandingkan studi sebelumnya. Tidak hanya dari sisi akurasi klasifikasi, tetapi juga dari segi metodologi validasi dan potensi aplikasi di dunia nyata, menjadikan sistem yang dikembangkan layak untuk dikaji lebih lanjut dalam integrasi dengan platform berbasis web atau *mobile*.

1.2 Rumusan Masalah

Rumusan masalah yang akan dibahas dalam penelitian ini adalah sebagai berikut :

1. Bagaimana implementasi metode *Convolutional Neural Network* dalam mendeteksi penyakit mata katarak ?
2. Bagaimana hasil pendeteksian penyakit mata katarak pada suatu citra digital?

3. Bagaimana tingkat akurasi pendeteksian penyakit mata katarak pada suatu citra digital menggunakan *Convolutional Neural Network*?

1.3 Tujuan Penelitian

Tujuan dalam penelitian ini adalah sebagai berikut :

1. Mengimplementasikan model arsitektur *Convolutional Neural Network* (CNN) untuk mendeteksi penyakit mata katarak dan mengevaluasi kinerjanya berdasarkan dataset yang ada.
2. Mengetahui hasil pendeteksian penyakit mata katarak pada suatu citra digital.
3. Mengetahui seberapa tinggi tingkat akurasi penyakit mata katarak pada suatu citra digital

1.4 Manfaat Penelitian

Manfaat dalam penelitian ini adalah sebagai berikut :

1. Bagi peneliti bermanfaat untuk memberikan konntribusi terhadap penerapan CNN untuk klasifikasi citra medis khususnya deteksi katarak.
2. Bagi masyarakat, membantu deteksi dini katarak secara mandiri, terutama di wilayah yang sulit menjangkau layanan dokter spesialis mata.
3. Bagi dunia kesehatan, mendukung proses diagnosis katarak yang lebih cepat, akurat, dan efisien melalui sistem berbasis kecerdasan buatan.

1.5 Batasan Masalah

Batasan masalah dalam penelitian ini adalah sebagai berikut :

1. Penelitian ini hanya fokus pada deteksi dini penyakit mata katarak menggunakan metode *Convolutional Neural Network* (CNN)
2. Data yang digunakan adalah data citra mata yang sudah terlabel dengan baik untuk klasifikasi mata katarak dan mata normal
3. Dataset dimbil dari situs Kaggle yang digunakan sebanyak 612 citra, dengan label Normal 306 citra dan label Katarak 306 citra [5].

1.6 Sistematika Penulisan Skripsi

Sistematika penulisan yang digunakan pada skripsi ini adalah sebagai berikut:

BAB I. PENDAHULUAN

Pada bab ini menguraikan secara umum mengenai latar belakang penelitian, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan penelitian serta sistematika penulisan.

BAB II. TINJAUAN PUSTAKA

Pada bab ini membahas mengenai dasar teori yang digunakan sebagai sumber dalam memahami permasalah dalam melakukan penelitian mengenai Penerapan Metode *Convolutional Neural Network (CNN)* untuk Deteksi Dini Penyakit Mata Katarak

BAB III. METODOLOGI PENELITIAN

Pada bab ini membahas mengenai metodologi penelitian yang digunakan dalam Penerapan Metode *Convolutional Neural Network (CNN)* untuk Deteksi Dini Penyakit Mata Katarak

BAB IV. HASIL DAN PEMBAHASAN

Pada bab ini membahas mengenai hasil serta pembahasan yang diperoleh dalam penelitian Penerapan Metode *Convolutional Neural Network (CNN)* untuk Deteksi Dini Penyakit Mata Katarak.

BAB V. KESIMPULAN DAN SARAN

Pada bab ini berisi tentang kesimpulan dari hasil penelitian yang telah dilakukan serta saran-saran sebagai masukan untuk penelitian lebih lanjut di masa mendatang.

DAFTAR PUSTAKA

LAMPIRAN

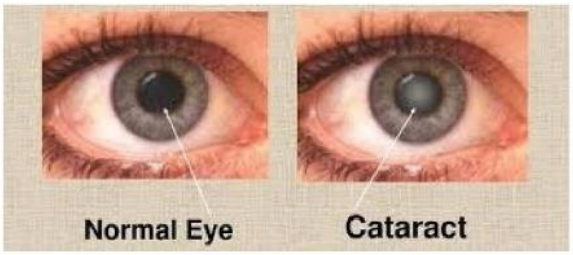
BAB II. TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Katarak dan Deteksinya melalui Citra Digital

Katarak adalah kondisi medis ketika lensa mata yang seharusnya jernih menjadi buram atau keruh. Perubahan ini menyebabkan cahaya tidak bisa masuk secara optimal ke retina, sehingga penglihatan menjadi kabur atau bahkan hilang sepenuhnya. Masalah ini sering terjadi pada lansia, tetapi juga bisa muncul akibat penyakit tertentu seperti diabetes, infeksi, atau sejak lahir (kongenital) [1].

Pendeteksian katarak berbasis citra umumnya dilakukan menggunakan citra anterior mata atau citra fundus retina. Penelitian ini menggunakan citra anterior mata sebagai datasetnya. Penggunaan citra anterior mata sebagai dataset dikarenakan citra anterior dapat diambil menggunakan kamera biasa atau slit-lamp, bahkan menggunakan kamera smartphone dengan *flashlight*, sehingga lebih mudah digunakan dalam konteks praktis, klinis, dan lapangan. Sementara itu, citra fundus memerlukan alat khusus seperti *fundus camera* atau retina *scanner,* yang mahal dan tidak selalu tersedia di fasilitas kesehatan primer [6]. Citra anterior mata dapat dilihat pada gambar 1.



Gambar 1 Citra Anterior Mata Normal dan Mata Katarak [1]

2.1.2 Pengolahan Citra Digital

Pengolahan citra digital merupakan proses transformasi dan manipulasi gambar dalam format digital dengan tujuan untuk meningkatkan kualitas visual, mengekstraksi informasi penting, serta memudahkan proses analisis terhadap objek yang terekam dalam citra [7].

Berdasarkan praktik pengolahan citra, terdapat beberapa tahapan utama yang umumnya dilakukan secara berurutan. Tahapan pertama adalah pra-pemrosesan (*preprocessing*), yang bertujuan untuk meningkatkan kualitas visual citra, misalnya dengan mengurangi noise melalui filter Gaussian atau median, serta meningkatkan kontras. Selanjutnya dilakukan segmentasi citra, yakni proses pemisahan antara objek yang diamati dengan latar belakangnya, sering kali dilakukan menggunakan metode thresholding atau *k-means clustering* [8].

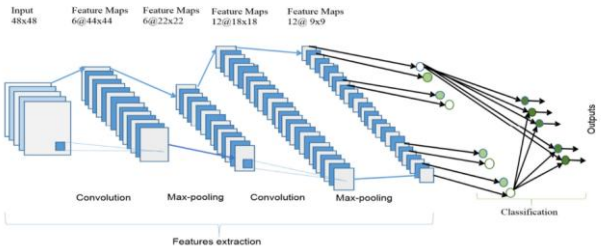
Setelah objek tersegmentasi dengan baik, tahap berikutnya adalah ekstraksi fitur, yaitu pengambilan informasi penting dari objek seperti bentuk, warna, tekstur, atau dimensi. Teknik ekstraksi fitur seperti GLCM (*Gray-Level Co-Occurrence Matrix*) digunakan untuk mendapatkan pola tekstur, sedangkan analisis statistik warna dapat digunakan untuk klasifikasi lebih lanjut. Informasi-informasi ini kemudian diproses menggunakan algoritma klasifikasi, baik berbasis machine learning maupun *deep learning*, seperti SVM, CNN, GoogLeNet, dan AlexNet, untuk mengidentifikasi atau mengenali objek atau kondisi tertentu pada citra [8].

2.1.3 *Deep learning*

Deep learning merupakan pendekatan dalam pembelajaran mesin yang memungkinkan komputer untuk belajar langsung dari data mentah tanpa memerlukan desain fitur manual oleh manusia. Pendekatan ini menggunakan jaringan saraf dalam (*deep neural networks*) yang terdiri dari banyak lapisan, di mana tiap lapisan belajar membentuk representasi data secara bertahap dari yang sederhana hingga kompleks. Inti dari pendekatan ini adalah kemampuannya untuk secara otomatis menemukan pola penting dalam data besar dan kompleks[9].

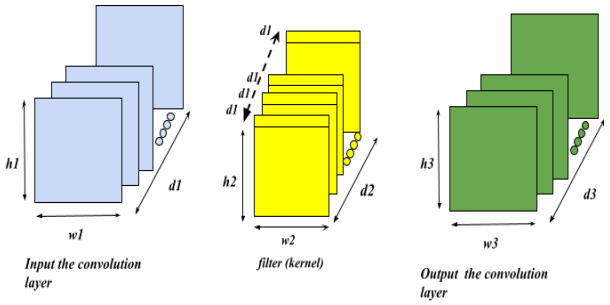
2.1.4 *Convolutional Neural Network* (CNN)

Convolutional Neural Network(CNN) adalah salah satu pendekatan terpenting dalam perkembangan teknologi kecerdasan buatan, khususnya dalam bidang *deep learning* yang berfokus pada pemrosesan data visual. Konsep dasar CNN terinspirasi dari sistem penglihatan biologis manusia, di mana objek diproses secara bertingkat melalui identifikasi ciri-ciri visual secara bertahap[10].



Gambar 2 Arsitektur CNN [10]

Gambar 2 menunjukan bahwa arsitektur CNN terbagi menjadi dua bagian utama yaitu bagian ekstraksi fitur dan bagian klasifikasi. Di bagian ekstraksi fitur, setiap



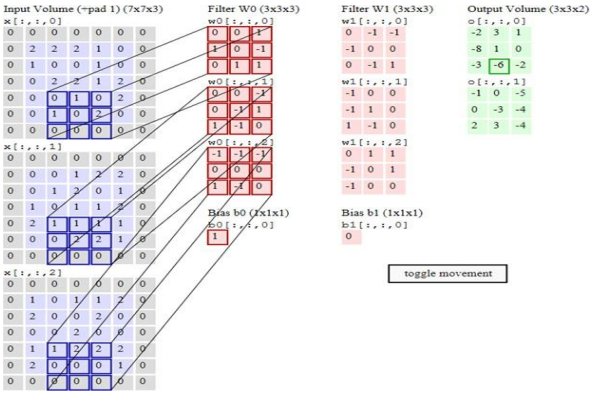
lapisan menerima input dari lapisan sebelumnya dan mengirimkan *output* ke lapisan berikutnya. CNN terdiri dari tiga jenis lapisan utama: konvolusi, *max-pooling*, dan klasifikasi. [10].

1. *Convolutional Layer* (Lapisan Konvolusi)

Gambar 3 Convolutional Layer [11]

Lapisan konvolusi yang diperlihat pada Gambar 3 adalah komponen utama CNN yang bertugas mengekstraksi fitur lokal dari input, seperti tepi, sudut, atau pola tekstur. Dalam lapisan ini, sejumlah filter (atau kernel) digunakan untuk menggeser (*slide*) di atas input dan melakukan operasi konvolusi—yakni, perkalian elemen per elemen (*element-wise multiplication*) antara filter dan bagian lokal dari *input*, kemudian dijumlahkan menjadi satu nilai [11].

Untuk setiap posisi kernel pada gambar, setiap angka dikalikan dengan angka yang sesuai pada matriks input (matriks biru) dan kemudian mereka semua dirangkum untuk nilai dalam posisi yang sesuai dalam matriks *output* (matriks hijau). Dengan *d1* > 1, hal yang sama terjadi untuk masing-masing saluran dan kemudian mereka ditambahkan bersama- sama kemudian disimpulkan dengan bias dari masing- masing filter dan ini membentuk nilai pada posisi yang sesuai dari matriks *output*[11]. Seperti terlihat pada Gambar 4.



Gambar 4 Pemetaan atau Perhitungan *input* ke *output* [11]

2. *Activation Layer* (Fungsi Aktivasi – ReLU)

Setelah konvolusi, hasilnya biasanya dilewatkan ke fungsi aktivasi non-linear. Fungsi aktivasi yang paling umum digunakan dalam CNN adalah ReLU (Rectified Linear Unit). Fungsi ini menyaring nilai negatif menjadi nol dan mempertahankan nilai positif apa adanya. ReLU mempercepat pelatihan jaringan karena menghindari masalah *vanishing gradient* yang sering terjadi pada fungsi aktivasi seperti sigmoid atau tanh. Fungsi ini juga membuat representasi menjadi lebih jarang (*sparse*), yang membantu dalam generalisasi model[11].

3. *Pooling Layer* (Lapisan Pooling)

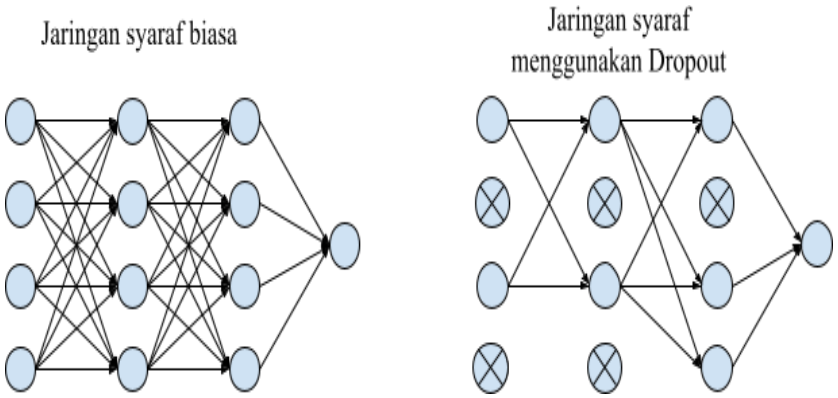
Pooling layer digunakan untuk mengurangi dimensi spasial dari feature map, mengurangi jumlah parameter dan komputasi, serta membuat representasi lebih tahan terhadap translasi kecil.. *Pooling* memungkinkan jaringan memiliki invariansi terhadap translasi kecil, sehingga fitur penting tetap dapat dikenali meskipun posisi objek dalam citra berubah sedikit[11].

4. *Fully Connected Layer* (Lapisan Terhubung Penuh)

Setelah serangkaian proses konvolusi dan *pooling*, hasil akhir diratakan (*flattened*) dan dikirim ke lapisan *fully connected*. Dalam lapisan ini, setiap neuron dihubungkan ke semua neuron di lapisan sebelumnya, sama seperti jaringan saraf konvensional. Lapisan ini berfungsi untuk menggabungkan fitur-fitur yang telah diekstraksi sebelumnya dan memproduksi *output* akhir seperti prediksi kelas menggunakan fungsi aktivasi[11].

2.1.4.1 *Dropout*

Dropout merupakan proses mencegah terjadinya *overfitting* dan juga mempercepat proses *learning*. *Dropout* mengacu kepada menghilangkan *neuron* yang berupa *hidden* maupun *layer* yang *visible* di dalam jaringan. Dengan menghilangkan suatu *neuron*, berarti menghilangkannya sementara dari jaringan yang ada. *Neuron* yang akan dihilangkan akan dipilih secara acak. Setiap *neuron* akan diberikan probabilitas yang bernilai antara 0 dan 1[12].



Gambar 5 Perbedaan ketika menggunakan *dropout* [11]

2.1.4.2 *Output Layer*

Output layer adalah lapisan terakhir yang memberikan hasil akhir dari jaringan. Untuk klasifikasi, lapisan ini biasanya menggunakan salah satu dari 2 fungsi aktivasi, yaitu sigmoid atau softmax, yang mengubah skor mentah (logit) menjadi probabilitas untuk tiap kelas [11].

Tabel 1 Fungsi Aktivasi pada *Output* Layer

Klasifikasi	Aktivasi *Output*	Bentuk *Output*	Penjelasan
Binary (2 kelas)	Sigmoid	Skalar (1 nilai)	Menghasilkan nilai antara 0 dan 1 yang mewakili probabilitas salah satu kelas. Biasanya dikombinasikan dengan binary crossentropy.
Multiclass (>2 kelas)	Softmax	Vektor (jumlah kelas)	Menghasilkan distribusi probabilitas untuk semua kelas. Biasanya dikombinasikan dengan categorical crossentropy.

Berdasarkan tabel 1, pada penelitian ini yakni deteksi katarak (antara "Normal" dan "Katarak"), *output layer* yang umumnya terdiri dari satu neuron dengan fungsi aktivasi sigmoid.

2.1.5 EfficientNetB0

EfficientNet diperkenalkan oleh Tan dan Le (2019) [13]sebagai solusi atas keterbatasan efisiensi pada arsitektur CNN konvensional. Model ini dirancang menggunakan prinsip compound scaling, yaitu pendekatan sistematis untuk memperbesar resolusi input, kedalaman layer, dan lebar jaringan secara proporsional agar efisiensi dan akurasi optimal. Dibandingkan dengan model seperti *ResNet-50* dan InceptionV3, *EfficientNetB0* mencapai akselerasi pelatihan hingga 8× lebih cepat dan jumlah parameter 5–10× lebih sedikit**,** sambil tetap mempertahankan atau bahkan meningkatkan akurasi klasifikasi [13].

Dalam penelitian ini, *EfficientNetB0* dimanfaatkan dengan *transfer learning*, di mana bobot awal yang dilatih pada dataset besar (seperti *ImageNet)* dapat digunakan kembali untuk tugas klasifikasi biner (Normal dan Katarak), yang sangat relevan pada dataset medis dengan jumlah data terbatas [13].

Karena penelitian ini melibatkan dataset relatif kecil (612 gambar), pemilihan *EfficientNetB0* menghindari *overfitting* yang lebih mungkin terjadi pada model besar seperti *ResNet152* atau *EfficientNet*B7 [14]. Dalam studi oleh Balaji et al. (2022) [15], *EfficientNetB0* dibandingkan dengan beberapa arsitektur CNN lain untuk deteksi tumor otak. Hasilnya, *EfficientNetB0* menghasilkan akurasi tertinggi 97,61%, melebihi model seperti VGG16 dan *Mobile*NetV2.

Penelitian oleh Padalia et al. (2022) menerapkan kombinasi CNN dan LSTM untuk deteksi katarak dari citra fundus. Meskipun model utamanya adalah CNN, mereka menyarankan penggunaan *EfficientNet* sebagai arsitektur alternatif yang lebih ringan dan akurat untuk implementasi dunia nyata [16]. Struktur *EfficientNetB0* dapat dirangkum dalam Tabel 2 berikut.

Tabel 2 Struktur Arsitektur *EfficientNetB0* [17]

Tahap	**Operator**	**Resolution**	**Channel**	**Repeats**	**SE**	**Exp**	**Activation**
Stem	Conv3x3	224×224	32	-	-	-	Swish
1	MBCon v1, 3×3	112×112	16	1	✓	1	Swish
2	MBCon v6, 3×3	112×112	24	2	✓	6	Swish
3	MBCon v6, 5×5	56×56	40	2	✓	6	Swish
4	MBCon v6, 3×3	28×28	80	3	✓	6	Swish
5	MBCon v6, 5×5	14×14	112	3	✓	6	Swish
6	MBCon v6, 5×5	14×14	192	4	✓	6	Swish
7	MBCon v6, 3×3	7×7	320	1	✓	6	Swish
Head	Conv1× 1 + Pool + FC	7×7	1280	-	-	-	-

Dari tabel 2 *EfficientNet*-B0 berperan sebagai model dasar dalam seri *EfficientNet* dan memiliki jumlah parameter paling sedikit. Meskipun tergolong sederhana, model ini tetap menunjukkan efektivitas tinggi dalam tugas klasifikasi gambar. Arsitekturnya terdiri dari beberapa blok berulang yang menggunakan *Depthwise Separable Convolution* serta fungsi aktivasi *Swish*[17].

2.1.6 Augmentasi

Augmentasi merupakan teknik penting dalam deep learning untuk memperbesar ukuran dataset secara artifisial dan meningkatkan generalisasi model. Augmentasi membantu mencegah *overfitting*, terutama ketika dataset berukuran kecil atau tidak seimbang, seperti yang sering terjadi pada domain medis. Melakukan augmentasi hanya pada data pelatihan merupakan praktik standar dalam pelatihan model

pembelajaran mesin. Tujuannya adalah agar model dilatih dengan variasi data yang lebih luas, namun tetap divalidasi dan diuji pada data yang mewakili kondisi nyata tanpa manipulasi tambahan [18]. Teknik Augmentasi yang akan digunakan dalam penelitian ini dapat dilihat pada tabel 3.

Tabel 3 Teknik Augmentasi yang Digunakan

No	Teknik Augmentasi	Alasan Penggunaan
1	Rotasi Acak (*rotation_range*=20)	Rotasi acak hingga ±20° meniru variasi alami posisi mata saat pengambilan gambar dan membantu model belajar invarian terhadap orientasi. Derajat rotasi dibatasi untuk menghindari distorsi struktural signifikan pada fitur mata [19].
2	Zoom Acak (*zoom_range*=0.2)	Zoom mengatasi variasi jarak antara kamera dan objek mata saat pengambilan gambar, membantu generalisasi terhadap skala. Nilai 20% dianggap aman untuk tidak menghilangkan informasi penting [20].
3	Pergeseran Horizontal (*width_shift_range*=0.1)	Simulasi variasi posisi horizontal mata dalam frame. Perubahan kecil (10%) menjaga agar ROI (region of interest) tetap berada dalam area pengamatan [20].
4	Pergeseran Vertikal (*height_shift_range*=0.1)	Sama dengan horizontal, tetapi untuk sumbu vertikal. Menambah ketahanan model terhadap sedikit kesalahan framing [21].
5	*Shear (shear_range*=0.1)	Teknik ini menambahkan distorsi perspektif ringan untuk memperkaya data terhadap kemungkinan kemiringan atau distorsi kamera kecil [27].
6	Flip Horizontal (*horizontal_flip=True*)	Karena struktur mata kiri dan kanan simetris, flipping horizontal meningkatkan variasi tanpa mengubah label kelas. Cocok untuk dataset mata [2].
7	*Fill Mode (fill_mode='nearest')*	Saat transformasi menyebabkan area kosong, mode ‘nearest’ mempertahankan kontinuitas piksel agar model tidak bingung terhadap noise tak alami [2].

Ada begitu banyak teknik augmentasi, sebagai contoh dapat dilihat pada tabel berikut kenapa tidak digunakan teknik augmentasi yang lain.

Tabel 4 Teknik Augmentasi yang lain

Teknik Augmentasi	Alasan Tidak Digunakan
Vertical Flip	Tidak relevan untuk citra mata karena membalik secara vertikal akan menghasilkan struktur anatomi yang tidak realistis (mata terbalik), yang dapat membingungkan model dan menurunkan akurasi. [22]
Brightness/Contrast Adjustment	Perubahan pencahayaan ekstrem dapat menutupi fitur penting pada mata seperti opasitas lensa atau refleksi cahaya yang menjadi petunjuk katarak. Selain itu, pencahayaan yang tidak konsisten bisa memperkenalkan noise tak alami.[22]
Gaussian Noise / Blur	Menambahkan noise atau blur dapat merusak struktur penting seperti pupil, iris, atau kekeruhan lensa. Ini bisa memperburuk performa model karena mengaburkan ciri khas penyakit [23].
Cutout / Random Erasing	Menghapus bagian acak dari citra bisa menghilangkan fitur diagnostik penting. Dalam kasus citra medis, penghilangan area ini sangat berisiko karena informasi klinis utama bisa hilang [24].
Color Jitter / HSV Shift	Warna pada citra mata (terutama pada citra RGB klinis) tidak selalu mengandung informasi penting seperti pada dataset umum. Perubahan warna dapat menyesatkan model jika fitur warna tidak relevan [25].
Mixup / CutMix	Teknik ini menggabungkan dua gambar secara acak. Meskipun efektif di beberapa domain, teknik ini tidak cocok untuk citra medis karena bisa menciptakan citra yang tidak masuk akal secara anatomis dan menyebabkan label ambiguity [26].

2.1.7 *Transfer learning* dan *Fine-Tuning*

Transfer learning adalah metode dalam *deep learning* di mana pengetahuan dari model yang telah dilatih pada tugas atau dataset besar digunakan kembali untuk menyelesaikan tugas baru pada domain yang serupa namun dengan data terbatas. Pendekatan ini sangat bermanfaat pada bidang medis seperti deteksi katarak, karena pengumpulan dan anotasi data medis sering kali sulit, memerlukan keahlian klinis, dan berbiaya tinggi [25].

Setelah proses *transfer learning*, langkah *fine-tuning* dilakukan untuk membuka (*unfreeze*) sebagian besar atau seluruh layer dari model dasar. Tujuannya adalah agar bobot jaringan dapat dioptimalkan ulang terhadap pola spesifik dari dataset baru. Dalam penelitian ini, semua layer dari *EfficientNetB0* di-*unfreeze*, memungkinkan pelatihan penuh (*end-to-end)* terhadap data katarak yang tersedia [27].

Beberapa studi telah membuktikan bahwa kombinasi *transfer learning* dan fine-tuning pada arsitektur pralatih seperti ResNet, *Mobile*Net, dan *EfficientNet* secara signifikan meningkatkan performa deteksi penyakit dari citra retina atau citra anterior mata, dibandingkan pelatihan dari nol (*training from scratch*) [20].

2.1.8 Hyperparameter Tuning

Hyperparameter tuning adalah proses sistematis dalam memilih kombinasi parameter eksternal yang mengatur perilaku algoritma pembelajaran mesin tetapi tidak diperbarui selama pelatihan, seperti *learning rate*, *optimizer*, jumlah *epochs*, dan *batch size*. Berbeda dengan parameter model (seperti bobot dalam jaringan saraf), hyperparameter harus ditentukan sebelum pelatihan dimulai dan memiliki pengaruh signifikan terhadap performa akhir model [28].

Beberapa penelitian berikut menunjukkan bahwa tuning hyperparameter secara sistematis dapat menghasilkan peningkatan signifikan dalam akurasi dan stabilitas model. Gonzales Castro et al. (2024) membuktikan bahwa tuning optimizer dan parameter lainnya dapat meningkatkan nilai AUC pada prediksi kekambuhan kanker payudara dari 0,70 menjadi 0,84 menggunakan model XGBoost [29]. Sugiyanti dan Haris (2024) menggunakan metode Optuna untuk tuning hyperparameter pada model XGBoost dan berhasil menurunkan error (RMSE) dari 1052 menjadi 968 dalam prediksi tingkat penjualan [30]. Dalam bidang prediksi penyakit jantung, Ahamad et al. (2023) menunjukkan bahwa GridSearchCV pada model seperti SVM dan XGBoost meningkatkan akurasi secara signifikan dibanding konfigurasi default [31].

Apabila hyperparameter tuning tidak dilakukan dapat menyebabkan:

1. Kinerja Lebih Buruk dari Tebakan Acak
   Seperti yang ditunjukkan oleh Sipper (2022), beberapa model default justru memberikan performa lebih rendah dibandingkan random guessing, khususnya jika tidak dilakukan tuning [7].

2. Inefisiensi Pelatihan dan Konsumsi Sumber Daya
   Model yang tidak dituning memerlukan pelatihan ulang berulang-ulang atau berhenti lebih awal karena poor convergence, menghabiskan waktu komputasi dan sumber daya [32].

Dalam konteks CNN, khususnya arsitektur seperti *EfficientNetB0*, pemilihan optimizer dan *learning rate* sangat krusial karena berpengaruh langsung terhadap kecepatan konvergensi dan kemampuan model dalam menghindari local minima [33].

2.1.9 Optimizer

Optimizer adalah algoritma dalam proses pembelajaran mesin yang bertugas meminimalkan fungsi kerugian (loss function) dengan cara memperbarui bobot-bobot jaringan saraf secara iteratif berdasarkan nilai gradien. Pemilihan optimizer yang tepat sangat krusial karena memengaruhi kecepatan konvergensi, kestabilan pelatihan, dan kualitas generalisasi model [27]. Beberapa optimizer yang digunakan dalam pelatihan ini adalah:

1. *Stochastic Gradient Descent (SGD*): Optimizer klasik yang memperbarui bobot berdasarkan subset data (mini-batch). Meski sederhana, SGD sangat sensitif terhadap *learning rate* dan cenderung lambat dalam konvergensi [21].
2. *Adam (Adaptive Moment Estimation):* Mengombinasikan keunggulan dari Momentum dan *RMSprop*, Adam menyesuaikan *learning rate* setiap parameter secara adaptif [33].
3. *RMSprop*: Cocok untuk data non-stationary dan digunakan dalam RNN, *RMSprop* menyesuaikan *learning rate* berdasarkan pergerakan gradien rata-rata kuadrat [34].
4. *Adagrad*: Cocok untuk data sparce, tetapi menurunkan *learning rate* secara progresif, yang menyebabkan pembelajaran terhenti terlalu awal [34].

Adapun rentang nilai yang dapat dijadikan acuan untuk menilai baik atau buruknya hasil training dapat dilihat pada tabel 5 berikut.

Tabel 5 Acuan Penilaian Hasil Training & Evaluasi [18]

Kriteria	Nilai Rentang	Interpretasi
Akurasi Optimal	≥ 0.80	Generalisasi sangat baik
Akurasi Cukup	0.60 – 0.79	Masih bisa diterima, perlu tuning
Akurasi Rendah	< 0.60	Belum optimal, perlu perbaikan
Loss Optimal	≤ 0.50	Prediksi stabil dan presisi tinggi
Loss Cukup	0.51 – 0.65	Prediksi lumayan, belum ideal
Loss Tinggi	> 0.65	Prediksi lemah atau over/underfit

2.1.10 Validasi Silang

Validasi silang, seperti 5-*Fold Cross-Validation* yang digunakan dalam penelitian ini, adalah teknik evaluasi yang membagi data pelatihan ke dalam beberapa subset, dan melatih model pada sebagian subset sambil menguji pada sisanya secara bergantian. Metode ini memberikan estimasi performa yang lebih stabil dan menghindari bias subset tunggal [35].

Dalam *deep learning*, validasi silang sangat berguna terutama ketika ukuran dataset terbatas, seperti pada data citra medis. Ide utama dari *cross-validation* adalah membagi dataset menjadi beberapa subset (*fold*). Model kemudian dilatih dan divalidasi secara berulang dengan menggunakan kombinasi *fold* yang berbeda [12]. Hasil dari tiap iterasi akan dirata-ratakan untuk mendapatkan skor evaluasi keseluruhan. Langkah-langkah umum:

1. Dataset dibagi menjadi k bagian (*fold*s).
2. Pada setiap iterasi, satu *fold* digunakan sebagai data validasi, dan sisanya digunakan untuk pelatihan.
3. Proses diulang sebanyak k kali (sesuai jumlah *fold*).
4. Skor evaluasi dari setiap iterasi dirata-ratakan sebagai hasil akhir.

2.1.11 Evaluasi Model Klasifikasi

Evaluasi model merupakan tahap krusial dalam proses pembangunan sistem klasifikasi berbasis machine learning maupun *deep learning*. Tujuan utama dari

evaluasi adalah untuk mengukur kemampuan generalisasi model terhadap data yang belum pernah dilihat sebelumnya. Evaluasi yang tepat akan memastikan bahwa model tidak hanya menghafal data pelatihan (*overfitting*) tetapi mampu melakukan prediksi secara akurat pada data baru[12].

Dalam konteks klasifikasi biner seperti deteksi katarak dari citra mata, evaluasi tidak hanya bergantung pada akurasi semata, tetapi juga mempertimbangkan keseimbangan klasifikasi antara dua kelas (Normal dan Katarak). Oleh karena itu, berbagai metrik digunakan untuk memberikan gambaran yang lebih komprehensif terhadap performa model [36]

Berikut adalah metrik evaluasi utama yang digunakan dalam penelitian ini.

1. Confusion Matrix

   Confusion matrix (matriks kebingungan) adalah alat evaluasi yang menggambarkan hubungan antara prediksi model dan label aktual dalam bentuk tabel 2×2 (untuk klasifikasi biner).

Tabel 6 Confusion Matrix [37]

	Prediksi Positif	**Prediksi Negatif**
Aktual Positif	*True Positive (TP)*	*False Negative (FN)*
Aktual Negatif	*False Positive (FP)*	*True Negative (TN)*

Keterangan:

TP: Model benar mengklasifikasikan kelas positif

TN: Model benar mengklasifikasikan kelas negatif

FP: Model salah mengklasifikasikan negatif sebagai positif

FN: Model salah mengklasifikasikan positif sebagai negatif

Confusion matrix berguna untuk menurunkan metrik evaluasi lainnya seperti *precision*, *recall*, dan *F1-Score*.

2. *Classification Report*

Classification Report adalah ringkasan dari metrik-metrik evaluasi utama yang dihitung berdasarkan nilai dalam confusion matrix, antara lain:

Precision:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Mengukur seberapa tepat model dalam memprediksi kelas positif.

Recall (*Sensitivity*):

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

Mengukur kemampuan model mendeteksi semua data dari kelas positif.

F1-Score:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Merupakan rata-rata harmonis dari *precision* dan *recall*.

Accuracy:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Proporsi prediksi yang benar dari seluruh data.

Metrik-metrik ini penting dalam aplikasi medis, di mana kesalahan deteksi bisa berdampak besar.

Tabel berikut akan memperlihatkan parameter evaluasi baik buruknya sistem berdasarkan *classification report*

Tabel 7 Parameter Evaluasi Pengujian Sistem [18]

Nilai *Precision* / *Recall* / F1	**Interpretasi**
0.90 – 1.00	Sangat Baik
0.80 – 0.89	Baik
0.70 – 0.79	Cukup
0.60 – 0.69	Buruk
< 0.60	Sangat Buruk

Teknik evaluasi selanjutnya adalah *Receiver Operating Characteristic (ROC)* curve, yaitu grafik yang menampilkan *true positive rate (recall*) terhadap *false positive rate (FPR)* pada berbagai threshold [36]. Sedangkan AUC (*Area Under Curve)* adalah ukuran dari kemampuan model membedakan antara kelas:

Nilai AUC = 1.0 menunjukkan klasifikasi sempurna.

AUC = 0.5 setara dengan tebakan acak.

AUC < 0.5 menandakan performa lebih buruk dari tebakan acak [36].

Dalam penelitian ini, penggunaan ROC-AUC bertujuan untuk mengukur kemampuan diskriminatif model CNN terhadap dua kelas target (normal vs katarak) secara menyeluruh.

2.1.12 Phyton

Python adalah bahasa pemrograman tingkat tinggi yang dirancang dengan filosofi kemudahan dalam penulisan dan keterbacaan kode. *Python* bersifat *open source* dan mendukung berbagai paradigma pemrograman seperti pemrograman berorientasi objek, imperatif, dan fungsional [38]. *Python* sangat populer di kalangan ilmuwan data dan pengembang AI karena sintaksnya yang sederhana dan pustaka-pustaka komputasi yang kaya [39].

Beberapa pustaka (library) *Python* yang digunakan dalam penelitian ini dapat dilihat pada tabel 7 berikut.

Tabel 8 Pustaka Phyton yang Digunakan

No.	Pustaka *Python*	Fungsi
1	*TensorFlow*	Digunakan sebagai framework utama untuk membangun dan melatih model CNN [40] .
2	*Keras*	Antarmuka tingkat tinggi berbasis TensorFlow yang menyederhanakan pembangunan model *deep learning* [41].
3	*NumPy*	Operasi numerik, manipulasi array, dan komputasi multidimensi [42].
4	*Matplotlib*	Visualisasi data seperti grafik akurasi dan loss selama pelatihan [43].
5	*Seaborn*	Visualisasi statistik seperti heatmap untuk confusion matrix [44].
6	*Scikit-Learn*	Digunakan untuk cross-validation, evaluasi metrik (*precision*, *recall*, f1), dan splitting data [45].
7	*OpenCV*	Pemrosesan citra dan augmentasi gambar untuk kebutuhan pelatihan model [46].
8	*PIL (Pillow)*	Membaca, menyimpan, dan memodifikasi citra sebelum diproses oleh CNN [47].

2.1.13 Google Colab oratory (Google Colab)

Google Colab oratory atau yang biasa disebut *Google Colab* merupakan sebuah platform berbasis cloud yang sangat efektif untuk digunakan dalam pembelajaran pemrograman *Python*, khususnya bagi pemula. *Google Colab* memungkinkan pengguna untuk menulis dan mengeksekusi kode *Python* langsung dari browser tanpa perlu melakukan instalasi perangkat lunak tambahan, sehingga memberikan aksesibilitas yang tinggi dan kemudahan penggunaan. Platform ini tidak hanya mendukung aktivitas pengkodean individu, tetapi juga menyediakan fitur kolaborasi daring yang memfasilitasi kerja tim dalam waktu nyata. Dalam konteks pendidikan, penggunaan *Google Colab* dapat membantu siswa memahami konsep dasar pemrograman *python* dengan lebih mudah melalui pendekatan praktis, seperti penulisan sintaks dasar, penggunaan array, dan pemrosesan data. Selain itu, *Google Colab* juga mendukung integrasi dengan berbagai sumber data seperti file Excel untuk keperluan analisis data, menjadikannya sebagai alat yang serbaguna dan relevan di era digital saat ini. Dengan antarmuka yang ramah pengguna serta kemampuan komputasi yang mumpuni, *Google Colab* terbukti menjadi sarana yang ideal dalam pengenalan dan pembelajaran pemrograman bagi kalangan pelajar maupun masyarakat umum [48].

2.2 Penelitian Terkait

Terdapat beberapa penelitian terkait dengan penelitian ini yang dijadikan sebagai perbandingan serta rujukan mengenai metode serta hasil yang dicapai pada penelitian ini, yaitu sebagai berikut:

2.2.1 Implementasi Algoritma *Convolutional Neural Network* dalam Mengidentifikasi Dini Penyakit pada Mata Katarak oleh Ramadhani, F. A., Nurhasanah, N., & Arifin, M. T. (2023)

Penelitian ini menerapkan *Convolutional Neural Network*untuk klasifikasi citra mata menjadi dua kelas: katarak dan non-katarak. Model dibangun dari awal (*custom CNN*) dan mampu mencapai akurasi hingga 92,05%. Proses ini mencakup pemrosesan dataset lokal dan pengembangan aplikasi berbasis *mobile* sebagai bentuk implementasi nyata. Fokus utama adalah solusi deteksi dini yang dapat digunakan secara luas, terutama di daerah yang minim akses ke tenaga medis [2].

2.2.2 *Cataract Classification Based on Fundus Images Using Convolutional Neural Network* oleh Simanjuntak, A. P., Syahputra, R., & Andika, R. (2022)

Penelitian ini membandingkan arsitektur CNN populer seperti *GoogLeNet, MobileNet, ResNet,* dan model CNN yang diusulkan. Dataset awal (399 gambar) di-*augmentasi* menjadi 3200 gambar. Model terbaik menghasilkan akurasi 92%, dan klasifikasi dilakukan terhadap 4 kelas katarak (Normal*, Immature, Mature, Hypermature*), bukan hanya klasifikasi biner [3].

2.2.3 *Detection and Classification of Brain Tumors Using Deep Convolutional Neural Networks* oleh Ranit Sen, Gopinath Balaji & Harsh Kirty (2022)

Penelitian ini menerapkan berbagai arsitektur CNN pre-trained seperti *EfficientNetB0*, ResNet50, Xception, dan VGG16 untuk mendeteksi tiga jenis tumor otak: Glioma, Meningioma, dan Pituitary dari citra MRI. Dengan menggunakan teknik *transfer learning*, augmentasi data, dan praproses citra

(denoising, skull stripping, cropping), model *EfficientNetB0* mencapai akurasi 97,61%. Studi ini menunjukkan efektivitas CNN dalam klasifikasi multi-kelas pada citra medis [15].

2.2.4 *A CNN-LSTM Combination Network for Cataract Detection Using Eye Fundus Images* oleh Dishant Padalia & Abhisek Mazumdar (2022)

Penelitian ini mengusulkan model gabungan CNN-LSTM untuk mendeteksi katarak dari citra fundus. CNN digunakan untuk ekstraksi fitur visual, sedangkan LSTM berfungsi sebagai klasifikator, menangkap dependensi spasial. Model diuji pada dataset ODIR dan mencapai akurasi 97,53%. Kombinasi dua arsitektur ini menghasilkan performa deteksi katarak yang unggul dan efisien untuk diagnosa otomatis [16].

2.2.5 Klasifikasi Citra Menggunakan *Convolutional Neural Network* dan *K-Fold Cross Validation* oleh Ari Peryanto , Anton Yudhana , Rusydi Umar (2020)

Studi ini mengaplikasikan metode CNN untuk klasifikasi citra bunga dengan evaluasi menggunakan K-*Fold* Cross Validation. Hasil menunjukkan akurasi tertinggi 80,36% dan rata-rata 76,49%. Penelitian ini menekankan pentingnya cross-validation dalam menguji keandalan model CNN serta penggunaan preprocessing seperti cropping dan noise removal untuk meningkatkan akurasi [11].

2.2.6 *Influence of Optimal Hyperparameters on the Performance of Machine Learning Algorithms for Predicting Heart Disease* oleh Ghulab Nabi Ahamad , Shafiullah , Hira Fatima , Imdadullah , S. M. Zakariya , Mohamed Abbas , Mohammed S. Alqahtani & Mohammed Usman (2023)

Penelitian ini membahas pentingnya tuning hyperparameter dalam meningkatkan akurasi prediksi penyakit jantung dengan menggunakan enam algoritma ML termasuk SVM, Random Forest, dan XGBoost. Dengan penerapan GridSearchCV untuk optimasi, model XGBoost mencapai akurasi tertinggi 99,03%. Studi ini relevan sebagai referensi dalam pemilihan dan penyetelan parameter model pembelajaran mesin untuk aplikasi medis [31].

2.2.7 *Analysis of the Application of Hyperparameter Tuning in Machine Learning to Increase the Accuracy of Sales-Level Prediction* oleh Sugiyanti & Muhammad Haris (2024)

Penelitian ini menggunakan XGBoost dengan Optuna hyperparameter tuning dan deteksi outlier untuk meningkatkan akurasi prediksi penjualan. Hasilnya menunjukkan penurunan nilai error (RMSE 968, MAE 713), menandakan pentingnya tuning dan *preprocessing* dalam meningkatkan performa model [30].

2.2.8 *Impact of Hyperparameter Optimization to Enhance Machine Learning Performance: A Case Study on Breast Cancer Recurrence Prediction* oleh Lorena González-Castro, Marcela Chávez, Patrick Duflot, Martín Lopez-Nores, Valérie Bleret & Guilherme Del Fiol (2024)

Studi ini menyoroti pentingnya optimasi hyperparameter dalam meningkatkan performa model ML untuk prediksi kekambuhan kanker payudara. Lima algoritma (LR, DT, GB, XGB, dan DNN) diuji sebelum dan sesudah tuning. Hasil menunjukkan peningkatan AUC signifikan pasca tuning, dengan XGBoost mencapai AUC 0,84 dan DNN 0,75. Studi ini menegaskan bahwa pengabaian tahap tuning dapat mengarah pada pemilihan model suboptimal dalam konteks kesehatan[29].

2.2.9 Penerapan Arsitektur *EfficientNet*-B0 pada Klasifikasi Leukimia Tipe *Acute Lymphoblastik Leukimia* oleh Alfataniah Nur Fajrina , Zein Hanni Pradana , Sevia Indah Purnama & Shinta Romadhona (2024)

Penelitian ini menggunakan *EfficientNet*-B0 untuk mengklasifikasikan jenis leukemia ALL ke dalam empat kelas: *Benign, Early, Precursor, dan Progenitor*. Pengujian dilakukan terhadap kombinasi hyperparameter (epoch, *learning rate*, optimizer). Hasil akurasi pada data uji mencapai 98,48%, menunjukkan potensi arsitektur CNN ringan namun akurat dalam klasifikasi penyakit berbasis citra medis [17].

2.2.10 *Automated Cataract Detection Using Deep learning and Pre-Trained CNN Models* oleh Mullangi, P., Kumari, T. S., & Babu, N. V. (2024)

Menggunakan pretrained models seperti DenseNet121, ResNet50, *Mobile*Net, Inception-v3, dan lainnya. Dataset sebanyak 1130 citra di-*augmentasi* menjadi 4746 gambar. DenseNet-121 menghasilkan 92% akurasi, dan model akhir siap untuk *deployment mobile* dan *clinical screening* [4].

Penelitian ini memiliki keterkaitan dengan beberapa studi terdahulu diatas yang telah mengembangkan model deteksi katarak maupun klasifikasi citra medis menggunakan CNN. Penelitian Ramadhani et al. dan Simanjuntak et al. menjadi rujukan utama dalam klasifikasi katarak, meskipun metode yang digunakan berbeda, yaitu CNN custom dan arsitektur populer lainnya. Pendekatan yang digunakan dalam penelitian ini, yaitu transfer learning dengan *EfficientNetB0* dan augmentasi data, sejalan dengan studi Ranit Sen et al. dan Alfataniah et al. yang menunjukkan efektivitas model pretrained pada citra medis. Selain itu, implementasi validasi silang 5-*fold* mengacu pada pendekatan Peryanto et al., untuk meningkatkan keandalan model. Tuning hyperparameter, khususnya pemilihan optimizer, relevan dengan penelitian oleh Ahamad et al., Sugiyanti, dan González-Castro yang menekankan pentingnya optimasi dalam meningkatkan performa model. Secara keseluruhan, penelitian ini mengintegrasikan temuan penting dari berbagai studi sebelumnya dan berhasil mengembangkan sistem deteksi katarak yang akurat dan siap di-deploy dalam aplikasi nyata.

BAB III. METODE PENELITIAN

3.1 Jenis dan Pendekatan Penelitian

Jenis penelitian ini bersifat kuantitatif dengan pendekatan eksperimental berbasis pemrograman komputer menggunakan teknik *deep learning*. Model yang digunakan adalah *Convolutional Neural Network(CNN),* yang dikenal unggul dalam klasifikasi citra karena kemampuannya dalam mengekstraksi informasi spasial dari data visual.

3.2 Tempat dan Waktu Penelitian

Penelitian ini dilakukan secara daring dengan memanfaatkan platform *Google Colaboratory* karena menyediakan fasilitas GPU gratis untuk pelatihan model. Selain itu, proses pengujian lokal dilakukan menggunakan laptop pribadi dengan spesifikasi Lenovo Ideapad S340, prosesor AMD Ryzen 3, RAM 16GB, dan GPU NVIDIA GeForce. Penelitian ini berlangsung dari bulan Januari sampai bulan Mei 2025.

3.3 Sumber dan Jenis Data

Data yang digunakan adalah citra mata yang diperoleh dari situs Kaggle, yaitu "*Cataract Image Dataset*" [5]. Dataset ini terbagi menjadi dua kategori, yaitu mata normal dan mata dengan katarak. Terdapat total 612 gambar, terdiri dari 306 gambar untuk masing-masing kelas, yang kemudian digunakan dalam tahap pelatihan dan evaluasi model.

3.4 Teknik Pengumpulan Data

Data diperoleh dengan mengunduh dataset secara langsung dari repositori publik di Kaggle. Setelah proses pengunduhan, dilakukan validasi terhadap format file dan pelabelan ulang agar sesuai kebutuhan klasifikasi. Seluruh citra kemudian

diatur ulang ukurannya menjadi 224x224 piksel dan disesuaikan ke dalam format RGB agar kompatibel dengan arsitektur input model *EfficientNetB0*.

3.5 Prosedur Penelitian

Tahapan prosedur penelitian ini meliputi:

3.5.1 Pengolahan Data

1. Akuisisi Dataset

Proses awal dalam penelitian ini adalah memperoleh dataset citra mata dari platform Kaggle pada tautan https://www.kaggle.com/datasets/nandanp6/cataract-image-dataset. Dataset terdiri atas dua kelas, yaitu citra mata Normal dan Katarak, masing-masing berjumlah 306 citra, total 612 data dalam format .png. Dataset diunduh dalam format .zip dan diekstrak secara lokal sebelum diunggah ke Google Drive untuk memudahkan akses saat pelatihan model menggunakan *Google Colaboratory*.
Pemilihan dataset ini dilakukan secara selektif berdasarkan beberapa pertimbangan ilmiah sebagai berikut:

a. Dataset ini menyediakan dua kelas yang telah dilabeli secara eksplisit dalam dua direktori terpisah, yaitu Normal/ dan Katarak/. Hal ini sangat mendukung dalam pengembangan model klasifikasi biner, karena mempermudah dalam proses preprocessing, pelabelan otomatis, serta analisis performa model terhadap masing-masing kelas.
b. Citra yang digunakan dalam dataset ini merupakan gambar bagian anterior mata yang secara visual merepresentasikan kondisi nyata dari mata normal dan mata penderita katarak. Ini sesuai dengan tujuan penelitian, yaitu mendeteksi indikasi katarak berdasarkan fitur visual pada citra mata.
c. Dataset ini terdiri dari 612 citra (306 katarak dan 306 normal), yang merupakan jumlah cukup representatif untuk pelatihan awal model deep learning berskala ringan hingga menengah. Meskipun tidak terlalu besar, dataset ini cukup untuk dieksplorasi menggunakan metode augmentasi data dan validasi silang untuk meningkatkan generalisasi model.

d. Dataset ini bersifat open-access dan bebas digunakan untuk tujuan non-komersial serta penelitian, yang menjadikannya sumber data yang legal dan etis untuk digunakan dalam studi akademik.
e. Dataset ini telah diunduh dan digunakan oleh banyak pengguna di komunitas Kaggle, serta mendapatkan umpan balik positif, yang menunjukkan bahwa dataset ini layak dan cukup terpercaya sebagai dasar penelitian dalam bidang deteksi katarak berbasis citra. Salah satu penelitian terdahulu yang telah menggunakan dataset ini adalah Implementasi Algoritma *Convolutional Neural Network* dalam Mengidentifikasi Dini Penyakit pada Mata Katarak oleh Ramadhani, F. A., Nurhasanah, N., & Arifin, M. T. (2023) [2].

2. Pra-pemrosesan Data

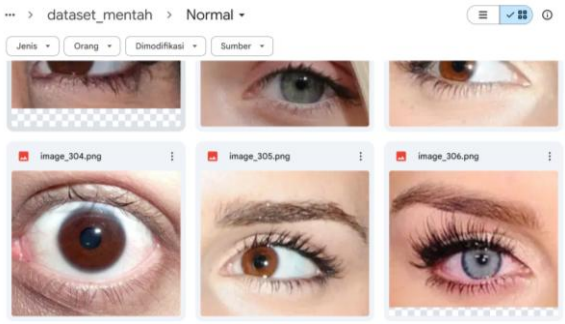
Pra-pemrosesan data merupakan tahap penting sebelum data digunakan untuk pelatihan model. Tujuannya adalah untuk memastikan bahwa data dalam format dan skala yang sesuai agar proses pelatihan model berjalan optimal. Pada penelitian ini, tahap pra-pemrosesan data meliputi beberapa langkah berikut:

a. Pembacaan dan Pengorganisasian Dataset

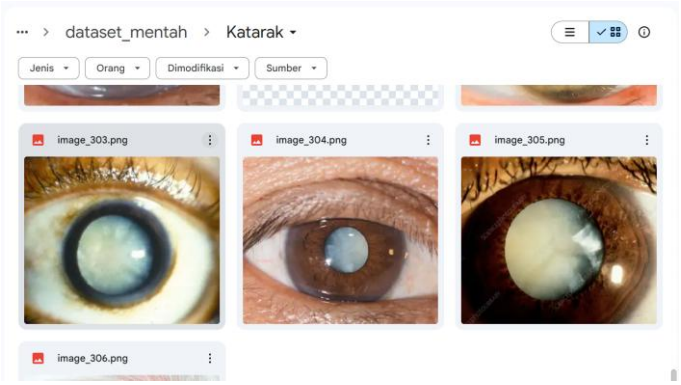
Dataset yang telah diunggah ke Google Drive diakses melalui perintah drive.mount() pada Colab agar dapat diproses dalam lingkungan notebook *Python*. Struktur folder disesuaikan agar mudah diolah dan dibaca oleh ImageDataGenerator. Adapun penjelasan mengenai pengorganisasian dataset citra mentah diperlihatkan pada Tabel 5. Adapun tampilan dataset citra mentah dengan kelas katarak dan normal dapat dilihat pada gambar 8 dan gambar 9.

Tabel 9 Dataset Citra Mentah [49]

No	**Nama *Folde*r Kelas Dataset Citra Mentah**	**Jumlah Data Citra**
1.	Normal	306
2.	Katarak	306
Total Dataset Citra		612



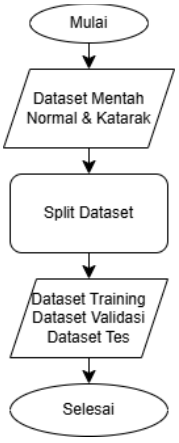
Gambar 6 Dataset Citra Mentah (Normal)



Gambar 7 Dataset Citra Mentah (Katarak)

b. Pembagian Dataset

Dataset dibagi menjadi tiga subset: training, validation, dan testing, dengan proporsi seimbang masing-masing 120 (train), 151 (val), dan 35 (test) per kelas. ini menggunakan modul os dan random untuk membagi serta mengacak data secara reproducible.



Gambar 8 Diagram Alir Pembagian Dataset

Gambar 8 menggambarkan alur pembagian dataset. Pada proses yang ditunjukkan dalam diagram alir, input yang digunakan adalah dataset mentah yang terdiri dari kumpulan citra mata dalam dua kategori, yaitu citra mata normal dan citra mata katarak. Dataset ini diperoleh dari sumber terbuka dan masih dalam kondisi belum diolah atau dibagi. Data mentah ini kemudian menjadi dasar untuk proses selanjutnya. Setelah dataset tersedia, dilakukan proses pembagian data atau split dataset, yang menghasilkan tiga output utama. Fungsi yang digunakan untuk membagi dataset adalah dengan menggunakan kode program manual berbasis *Python* yang memanfaatkan modul :

1. os.makedirs() untuk membuat direktori pembagian dataset,
2. os.listdir() untuk membaca file,
3. random.shuffle() untuk mengacak data,
4. serta logika slicing list (images[:n_train], images[n_train:n_train+n_test], dst) untuk menentukan subset data pelatihan, validasi, dan pengujian.

Meskipun fungsi seperti train_test_split() dari scikit-learn tersedia dan umum digunakan untuk membagi dataset secara otomatis, penelitian ini tidak menggunakannya karena beberapa pertimbangan berikut:

1. Pembagian dataset dilakukan secara manual dan terstruktur berdasarkan kelas (‘Normal’ dan ‘Katarak’) untuk memastikan jumlah data pada masing-masing subset (train, val, test) benar-benar seimbang untuk setiap kelas. Fungsi train_test_split() tidak secara langsung memberikan kontrol sebanyak itu, terutama untuk pembagian tiga subset sekaligus dengan distribusi per kelas yang eksplisit.
2. Pembagian dataset dilakukan sebelum augmentasi, dan hanya subset training yang mengalami augmentasi. Jika train_test_split() digunakan, maka pembagian data acak berpotensi mengganggu alur ini karena tidak menjamin pembagian berbasis folder atau struktur direktori yang mendukung ImageDataGenerator.
3. Proses pelatihan model dilakukan dengan menggunakan ImageDataGenerator.flow_from_directory() yang membutuhkan struktur folder tertentu, bukan hanya array data dan label. Oleh karena itu,

pembagian dataset dilakukan secara manual ke dalam folder train, val, dan test sesuai struktur yang dibutuhkan oleh fungsi ini.

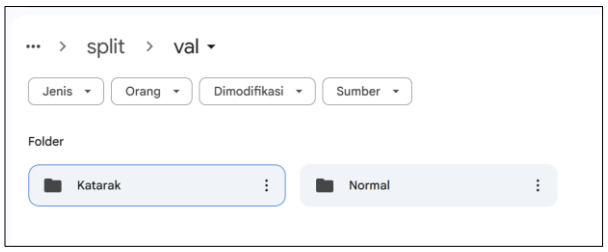
4. Dengan menyusun kode pembagian data secara manual dan mengatur seed acak (jika diimplementasikan), peneliti dapat memastikan hasil pembagian dataset dapat direproduksi dengan konsisten, sedangkan train_test_split() cenderung berubah-ubah jika tidak dikonfigurasi secara ketat.

Output dari proses ini adalah tiga subset dataset yang terdiri dari dataset training, dataset validasi, dan dataset tes. Dataset training digunakan untuk melatih model dalam mengenali pola-pola pada citra mata, dataset validasi digunakan untuk memantau dan mengevaluasi kinerja model selama pelatihan agar tidak mengalami *overfitting*, sedangkan dataset tes digunakan untuk mengukur kemampuan akhir model dalam mengklasifikasikan citra yang belum pernah dilihat sebelumnya. Ketiga subset data inilah yang menjadi keluaran dari proses pembagian dataset dan akan digunakan pada tahapan selanjutnya dalam pembangunan dan evaluasi model deteksi katarak.Hasil pembagian datasetnya dapat dilihat pada tabel 7. Tampilan *fold*er dataset yang sudah di bagi dapat dilihat pada gambar 9, gambar 10 dan gambar 11.

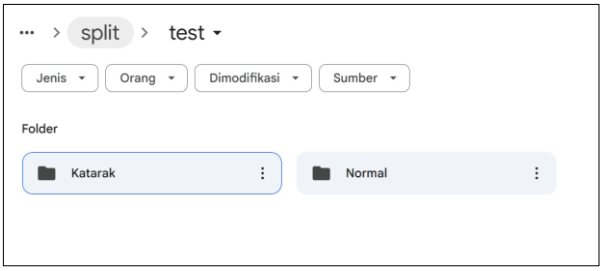
Tabel 10 Pembagian Dataset

Dataset Split	**Kelas**	**Jumlah Gambar**
Train	Normal	120
	Katarak	120
Validation	Normal	151
	Katarak	151
Test	Normal	35
	Katarak	35
Total Dataset		**612**

Gambar 9 Folder train [50]



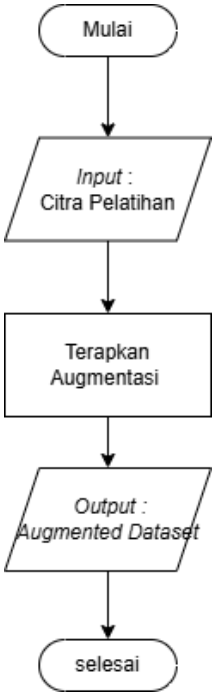
Gambar 10 Folder val [51]



Gambar 11 Folder test [52]

c. Augmentasi Data

Augmentasi data digunakan untuk menambah variasi citra latih agar model lebih robust terhadap perubahan bentuk, posisi, dan pencahayaan. Adapun alir dari proses augmentasi data dapat dilihat pada gambar 13.



Gambar 12 Diagram alir augmentasi

Berdasarkan gambar 12, input dari proses ini adalah dataset citra pelatihan (train) yang terdiri dari gambar mata normal dan katarak hasil dari pembagian dataset sebelumnya. Dataset ini memiliki jumlah yang terbatas, yaitu masing-masing 120 gambar per kelas. Untuk mengatasi keterbatasan tersebut, dilakukan proses augmentasi data, yaitu menambahkan variasi citra dengan cara memodifikasi gambar yang sudah ada menggunakan transformasi tertentu. Proses augmentasi ini dilakukan secara otomatis menggunakan fungsi ImageDataGenerator dari pustaka TensorFlow Keras. Fungsi ini dirancang untuk melakukan augmentasi gambar untuk menyimpan gambar hasil augmentasi ke dalam folder terpisah. Beberapa parameter yang digunakan dalam fungsi ini dapat dilihat pada tabel 8. Setelah konfigurasi ditetapkan, fungsi ini diterapkan pada folder berisi citra pelatihan, dan gambar hasil augmentasi disimpan ke folder baru (augmented_train) dalam bentuk gambar baru dengan nama file yang dimodifikasi. Proses augmentasi ini menghasilkan berbagai versi baru dari setiap gambar pelatihan, yang mencerminkan kondisi nyata yang lebih bervariasi. Data pada subset validasi dan test tidak dikenai augmentasi, hanya dilakukan normalisasi menggunakan rescale=1./255.

Tabel 11 Teknik Augmentasi

No	Teknik Augmentasi	Alasan Penggunaan
1	Rotasi Acak (rotation_range=20)	Rotasi acak hingga ±20° meniru variasi alami posisi mata saat pengambilan gambar dan membantu model belajar invarian terhadap orientasi. Derajat rotasi dibatasi untuk menghindari distorsi struktural signifikan pada fitur mata [19].
2	Zoom Acak (zoom_range=0.2)	Zoom mengatasi variasi jarak antara kamera dan objek mata saat pengambilan gambar, membantu generalisasi terhadap skala. Nilai 20% dianggap aman untuk tidak menghilangkan informasi penting [20].
3	Pergeseran Horizontal (width_shift_range=0.1)	Simulasi variasi posisi horizontal mata dalam frame. Perubahan kecil (10%) menjaga agar ROI (region of interest) tetap berada dalam area pengamatan [20].

4	Pergeseran Vertikal (height_shift_range=0.1)	Sama dengan horizontal, tetapi untuk sumbu vertikal. Menambah ketahanan model terhadap sedikit kesalahan framing [21].
5	Shear (shear_range=0.1)	Teknik ini menambahkan distorsi perspektif ringan untuk memperkaya data terhadap kemungkinan kemiringan atau distorsi kamera kecil [27].
6	Flip Horizontal (horizontal_flip=True)	Karena struktur mata kiri dan kanan simetris, flipping horizontal meningkatkan variasi tanpa mengubah label kelas. Cocok untuk dataset mata [2].
7	Fill Mode (fill_mode='nearest')	Saat transformasi menyebabkan area kosong, mode ‘nearest’ mempertahankan kontinuitas piksel agar model tidak bingung terhadap noise tak alami [2].

d. Skala dan Ukuran Citra

Pada tahap ini, dilakukan proses resizing atau penyesuaian ukuran citra agar seragam dan sesuai dengan kebutuhan arsitektur model *Convolutional Neural Network* (CNN) yang digunakan dalam penelitian ini, yaitu *EfficientNetB0*. Model tersebut membutuhkan input gambar berukuran 224x224 piksel dengan tiga kanal warna (RGB). Oleh karena itu, semua citra dalam dataset—baik data latih, validasi, maupun uji—perlu diubah ukurannya ke dalam resolusi tersebut sebelum digunakan dalam proses pelatihan dan evaluasi.

Proses resizing dilakukan secara otomatis menggunakan metode target_size=(224, 224) yang disediakan oleh fungsi flow_from_directory() dari kelas ImageDataGenerator pada library Keras. Fungsi ini memungkinkan pembacaan citra dari struktur folder dan sekaligus menerapkan resizing secara real-time saat data di-load ke dalam model

3.5.2 Membangun Sistem Model CNN

1. Perancangan Arsitektur CNN (Model *EfficientNetB0*)

Pada tahap ini, penelitian menggunakan *EfficientNetB0* sebagai arsitektur dasar (backbone) dalam pendekatan transfer learning untuk klasifikasi citra mata

menjadi dua kelas: *Normal* dan *Katarak*. Proses ini dilakukan dalam beberapa langkah, yaitu memuat arsitektur dasar, menambahkan layer klasifikasi kustom, dan melakukan fine-tuning.

a. Arsitektur Dasar (Base Model)

Model dasar *EfficientNetB0* dimuat menggunakan fungsi tf.keras.applications.*EfficientNetB0* dengan parameter include_top=False dan weights='imagenet'. Konfigurasi ini memungkinkan model memanfaatkan bobot pralatih dari dataset ImageNet, namun tanpa *fully connected layer* di akhir, sehingga dapat disesuaikan dengan kebutuhan klasifikasi biner dalam penelitian ini.

Citra input dikonversi ke ukuran 224x224 piksel (RGB), yang sesuai dengan spesifikasi *EfficientNetB0*. Layer dasar ini berfungsi sebagai feature extractor, yang mengenali pola visual penting dari citra fundus mata.

b. Penambahan Layer Klasifikasi *(Custom Head)*

Setelah fitur diekstraksi oleh *EfficientNetB0*, dilakukan penambahan beberapa layer klasifikasi di atasnya. Layer tambahan ini dirancang untuk mengubah keluaran spasial dari base model menjadi prediksi akhir berupa probabilitas. Penambahan ini dilakukan secara bertahap menggunakan fungsi dari Keras Sequential API sebagai berikut:

Tabel 12 Penambahan Layer Klasifikasi

Layer	Fungsi	Kode Fungsi
GlobalAveragePooling2D()	Mengubah fitur spasial (2D) menjadi vektor 1D	*tf.keras.layers.GlobalAveragePooling2D()*
BatchNormalization()	Menstabilkan distribusi aktivasi	*tf.keras.layers.BatchNormalization()*
Dropout(0.5)	Mencegah *overfitting*	*tf.keras.layers.Dropout(0.5)*

Dense(128, activation='relu')	Lapisan hidden untuk pemrosesan non-linear	*tf.keras.layers.Dense(128, activation='relu')*
Dense(1, activation='sigmoid')	Lapisan akhir untuk prediksi biner	*tf.keras.layers.Dense(1, activation='sigmoid')*

Seluruh arsitektur ini kemudian disusun menggunakan API tf.keras. Model *EfficientNetB0* yang dimuat dengan include_top=False hanya menghasilkan representasi fitur dari citra input, tidak memiliki bagian klasifikasi yang sesuai untuk tugas spesifik (seperti deteksi katarak). Oleh karena itu, diperlukan penambahan layer klasifikasi (custom head) yang mampu:

a. Merangkum fitur spasial menjadi informasi global
b. Mengurangi kompleksitas dan *overfitting*
c. Menghasilkan output berupa probabilitas biner (0 atau 1)

Adapun tahapan dan proses penambahan layer adalah sebagai berikut :

1. *GlobalAveragePooling2D()*
   a. Tujuan: Mengubah output spasial dari CNN (berbentuk 3D: *tinggi* × *lebar* × *channel*) menjadi vektor 1D.
   b. Cara Kerja: Setiap saluran (channel) dirata-rata secara global (mengambil rata-rata seluruh piksel dalam channel tersebut).
   c. Keunggulan: Lebih sederhana dan tidak *overfitting* dibandingkan Flatten() + Dense.
2. *BatchNormalization()*
   a. Tujuan: Menormalkan data keluaran dari layer sebelumnya agar memiliki distribusi yang stabil.
   b. Cara Kerja: Menghitung rata-rata dan standar deviasi dari batch input, lalu menormalkan output.
   c. Manfaat: Mempercepat pelatihan, Mengurangi masalah eksploding atau vanishing gradient, Stabilisasi selama proses learning

3. *Dropout(rate=0.5)*
   a. Tujuan: Regularisasi – mencegah model menghafal data (*overfitting*).
   b. Cara Kerja: Saat pelatihan, Dropout akan menonaktifkan (drop) sejumlah neuron secara acak (di sini: 50%) pada setiap batch.
   c. Manfaat: Membantu model belajar pola yang lebih general.
4. *Dense(128, activation='relu')*
   a. Tujuan: Memberikan kemampuan non-linear pada model agar dapat menangkap pola kompleks.
   b. Cara Kerja: Layer fully connected dengan 128 neuron dan fungsi aktivasi ReLU (Rectified Linear Unit).
   c. Manfaat: Mengubah vektor fitur menjadi representasi baru yang lebih bermakna untuk klasifikasi.
5. *Dense(1, activation='sigmoid')*
   a. Tujuan: Layer output untuk klasifikasi biner.
   b. Cara Kerja: Mengubah nilai input menjadi angka antara 0 dan 1 dengan fungsi sigmoid, mewakili probabilitas kelas (misal: 0 = Normal, 1 = Katarak).
   c. Manfaat: Cocok untuk kasus *binary classification* karena hanya butuh satu neuron output.

c. Fine-Tuning

Tahapan fine-tuning dilakukan untuk menyesuaikan beberapa layer akhir *EfficientNetB0* dengan karakteristik dataset katarak. Strategi yang digunakan adalah:

1. Membekukan seluruh layer hingga indeks tertentu (misal fine_tune_at = 100)
2. Membuka layer di atasnya agar dapat dilatih (layer.trainable = True)
3. Menyusun ulang model dan mengkompilasinya kembali

3.5.3 Tuning Hyperparameter : Optimizer

Penelitian ini menguji empat jenis optimizer, yaitu Adam, Stochastic Gradient Descent (SGD), RMSprop, dan Adagrad, dengan konfigurasi yang seragam.

Masing-masing optimizer digunakan dengan nilai learning rate tetap sebesar 1e-4, jumlah epoch sebanyak 15 (untuk menjaga keseimbangan antara durasi pelatihan yang efisien dan kemungkinan model belajar secara optimal dari data yang tersedia), dan batch size sebesar 32 (nilai default yang umum digunakan dalam pelatihan model deep learning karena memberikan keseimbangan antara stabilitas gradien dan efisiensi komputasi). Tujuan dari pengujian ini adalah untuk membandingkan performa masing-masing algoritma optimasi dalam konteks klasifikasi biner (Normal vs Katarak) menggunakan arsitektur *EfficientNetB0*.

Untuk memudahkan proses pengujian, seluruh optimizer didefinisikan dalam sebuah struktur data dictionary bernama optimizer_map. Struktur ini memetakan nama optimizer dengan objek optimizer yang sesuai dari library tensorflow.keras.optimizers. Pemilihan optimizer dilakukan secara dinamis berdasarkan nama yang diberikan pengguna.

Fungsi yang Digunakan:

a. tf.keras.optimizers.Adam(), SGD(), RMSprop(), dan Adagrad()
b. Fungsi-fungsi ini merupakan implementasi dari berbagai algoritma optimasi dalam TensorFlow/Keras.

Alasan Penggunaan:

Pendekatan ini memberikan fleksibilitas serta kemudahan dalam pengujian berbagai optimizer hanya dengan mengubah satu parameter nama, tanpa perlu memodifikasi struktur kode pelatihan.

Setelah optimizer dipilih, model CNN yang telah dirancang dikompilasi menggunakan fungsi model.compile(). Fungsi ini menentukan optimizer, fungsi loss, dan metrik evaluasi yang akan digunakan selama pelatihan.

Fungsi yang Digunakan:

model.compile()

Alasan Penggunaan:

Fungsi ini merupakan langkah wajib sebelum pelatihan model. Fungsi loss binary_crossentropy dipilih karena tugas klasifikasi hanya terdiri atas dua kelas. Metrik *Accuracy* digunakan untuk mengevaluasi proporsi prediksi yang benar.

Selama pelatihan, dua buah callback utama digunakan untuk mengontrol dan mengoptimalkan proses, yaitu *EarlyStopping* dan *ModelCheckpoint*.

Fungsi yang Digunakan:

a. *EarlyStopping*: Menghentikan pelatihan apabila tidak terjadi perbaikan pada nilai loss validasi selama beberapa epoch.
b. *ModelCheckpoint*: Menyimpan model terbaik berdasarkan akurasi validasi tertinggi selama pelatihan.

Alasan Penggunaan:

a. *EarlyStopping* mencegah *overfitting* dan pemborosan sumber daya dengan menghentikan pelatihan lebih awal.
b. *ModelCheckpoint* menjamin bahwa hanya model dengan performa terbaik yang disimpan untuk evaluasi selanjutnya.

Model kemudian dilatih menggunakan metode model.fit(), dengan memasukkan data pelatihan dan validasi, serta callback yang telah disiapkan.

Fungsi yang Digunakan:

model.fit()

AlasanPenggunaan:

Fungsi ini adalah metode utama untuk melatih model neural network dalam Keras, serta mendukung integrasi callback untuk mengatur proses pelatihan secara otomatis dan efisien.

Apabila performa hasil training yang dilakukan sudah baik maka akan dilanjutkan ke tahap pengujian dan evaluasi sistem, apabila belum baik tuning hyperparameter akan di ulang. Adapun parameternya dapat dilihat pada tabel 5 bab 2.

3.5.4 Pengujian dan Evaluasi Sistem

Setelah proses pelatihan selesai, model dievaluasi untuk mengukur performanya dalam melakukan klasifikasi citra retina (Normal vs Katarak). Dilakukan pengujian model dengan validasi silang 5 *fold*. Untuk memastikan bahwa model yang dikembangkan memiliki kemampuan generalisasi yang baik dan tidak hanya optimal pada satu subset data tertentu, digunakan teknik 5-*Fold* Cross-Validation. Teknik ini membagi dataset menjadi 5 bagian (*fold*) yang seimbang. Pada setiap iterasi, satu *fold* digunakan sebagai data validasi, sementara empat *fold* lainnya

digunakan untuk pelatihan. Proses ini diulang sebanyak 5 kali, sehingga setiap data akan digunakan tepat satu kali sebagai data validasi. Selanjutnya dilakukan evaluasi sistem menggunakan confusion matrix berdasarkan data hasil pengujian. Dari confusion matrix yang didapatkan akan menjadi acuan untuk melakukan perhitungan metrik error yaitu *Precision, Recall, F1 Score dan Accuracy*. Digunakan juga kurva *Receiver Operating Characteristic (ROC)*.

3.5.5 Analisis Hasil Pengujian

Pada tahapan ini dilakukan analisis berdasarkan hasil evaluasi pengujian sistem untuk menentukan apakah model sudah baik atau belum berdasarkan parameter yang sudah di bahas paa bab 2 tabel 7.

3.5.6 Penyimpanan Model

Setelah model CNN dengan arsitektur *EfficientNetB0* selesai dilatih dan menghasilkan performa optimal, langkah selanjutnya adalah menyimpan model tersebut untuk digunakan kembali tanpa perlu melatih ulang. Penyimpanan model dilakukan menggunakan fungsi model.save() dari library TensorFlow/Keras. Fungsi ini menyimpan seluruh arsitektur model, bobot (weights), dan konfigurasi pelatihan ke dalam satu file dengan format .keras. Model yang telah disimpan dapat dimuat kembali dengan mudah menggunakan perintah tf.keras.models.load_model(), memungkinkan proses deployment dilakukan secara efisien. Dengan demikian, aplikasi hanya perlu memuat model yang sudah dilatih dan siap digunakan untuk melakukan prediksi secara langsung. Struktur akhir dari model *EfficientNetB0* yang digunakan terdiri dari dua bagian utama:

1. Ekstraksi Fitur: Menggunakan model *EfficientNetB0* pralatih dari *ImageNet* untuk mengenali fitur visual kompleks pada citra mata.

2. Klasifikasi: Layer tambahan seperti *GlobalAveragePooling2D, Dropout*, dan *Dense* dengan aktivasi sigmoid digunakan untuk mengklasifikasikan citra menjadi kategori Normal atau Katarak.

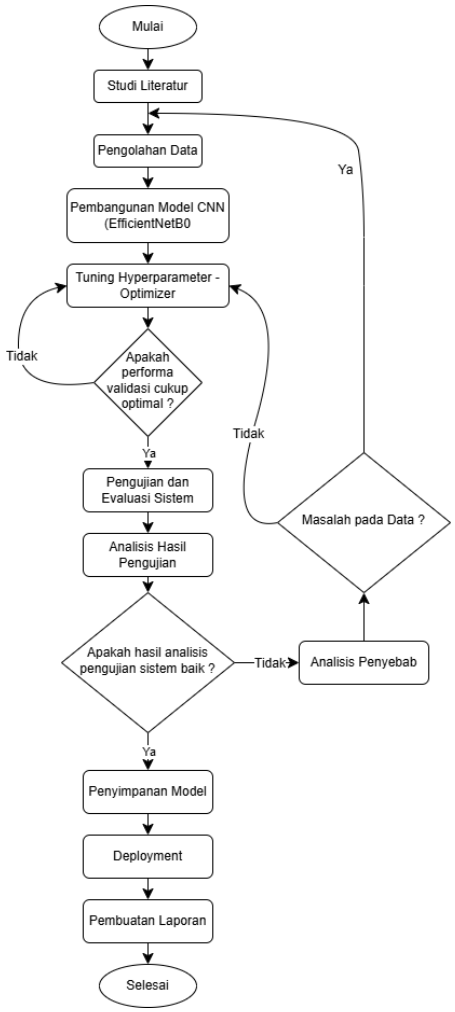
Total parameter model mencapai sekitar 5,3 juta, yang mencerminkan kompleksitas dan kapasitas model dalam mengenali pola citra secara akurat.

3.5.7 Deployment Model Deteksi Katarak

Deployment model dilakukan dengan mengintegrasikan model CNN yang telah dilatih ke dalam sebuah aplikasi berbasis web menggunakan framework Flask. Tujuan dari tahap ini adalah mengimplementasikan model dalam lingkungan nyata yang dapat diakses dan digunakan oleh pengguna akhir.

3.6 Diagram Alir Penelitian

Tahapan umum dalam penelitian ini dijelaskan sebagai berikut:



Gambar 13 Diagram Alir Penelitian

Diagram alir penelitian yang diperlihatkan pada gambar 13 ini menggambarkan tahapan sistematis yang dilakukan untuk mengembangkan dan mengevaluasi model CNN (*Convolutional Neural Network*) dalam mendeteksi katarak dari citra mata. Diagram alur proses penelitian ini disusun berdasarkan prinsip pengembangan model CNN [18], tuning hyperparameter [28], serta alur evaluasi sistem machine learning [13].

Proses diawali dengan studi literatur yang bertujuan untuk menggali berbagai teori, metode, dan temuan penelitian sebelumnya yang relevan dengan deteksi katarak dan pengolahan citra menggunakan *deep learning*. Setelah itu, dilakukan pengolahan data, yang mencakup pengunduhan dataset dari platform Kaggle, validasi format file, pelabelan ulang, serta resizing citra ke ukuran 224x224 piksel agar sesuai dengan masukan model *EfficientNetB0*. Dataset kemudian dibagi menjadi tiga subset, yaitu data latih (train), validasi (validation), dan uji (test). Pada tahap ini pula, dilakukan augmentasi data hanya terhadap citra pelatihan guna meningkatkan variasi data dan mencegah *overfitting*.

Setelah data siap, dilakukan pembangunan sistem dengan merancang model CNN berbasis arsitektur *EfficientNetB0*. Model ini dimodifikasi dengan menambahkan beberapa layer klasifikasi, seperti GlobalAveragePooling2D, *Dropout*, dan Dense dengan aktivasi sigmoid. Proses pelatihan model dilakukan menggunakan beberapa jenis optimizer (Adam, SGD, *RMSprop*, dan *Adagrad*), serta dilakukan penyetelan hyperparameter seperti *learning rate*. Untuk meningkatkan kinerja dan menghindari *overfitting*, digunakan callback seperti *EarlyStopping* dan *ModelCheckpoint*. Setelah model selesai dilatih, performanya dievaluasi pada data validasi. Jika hasil validasi belum memuaskan, maka dilakukan perbaikan melalui tuning ulang parameter, sehingga proses kembali ke tahap sebelumnya.

Apabila model menunjukkan hasil yang baik, langkah berikutnya adalah melakukan pengujian sistem dengan validasi silang menggunakan data uji yang tidak pernah dilihat oleh model sebelumnya. Evaluasi dilakukan menggunakan confusion matrix, *Classification Report* (*precision*, *recall*, *F1-Score*), dan ROC-AUC. Hasil pengujian kemudian dianalisis secara mendalam untuk mengetahui kekuatan dan kelemahan model, termasuk dari sisi stabilitas model yang diuji menggunakan teknik 5-*fold* cross-validation. Jika hasil pengujian dianggap kurang optimal, maka

model dapat dikembangkan ulang untuk diperbaiki. Untuk parameter yang digunakan sebagai acuan baik atau tidaknya hasil training maupun evaluasi dapat dilihat di tabel 3 pada bab II. Namun, jika kinerja pengujian telah memenuhi kriteria yang diharapkan, maka penelitian dilanjutkan ke tahap akhir berupa penyimpanan model terbaik, deployment model dan penyusunan laporan lengkap mengenai seluruh proses dan hasil yang diperoleh. Dengan demikian, penelitian ditutup setelah dokumentasi tersusun dan siap dipublikasikan.

BAB V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Adapun kesimpulan dari hasil penelitian ini adalah:

1. Penelitian ini berhasil mengimplementasikan metode *Convolutional Neural Network (CNN)* dengan arsitektur *EfficientNetB0* untuk mendeteksi penyakit mata katarak dari citra digital melalui tahapan akuisisi dan pra-pemrosesan data, augmentasi citra, pembangunan model menggunakan *transfer learning* dan fine-tuning, hingga pelatihan dan validasi model menggunakan 5-*fold* cross-validation..

2. Hasil pendeteksian penyakit mata katarak pada citra digital menunjukkan performa yang tinggi. Model mampu membedakan antara citra mata normal dan katarak dengan sangat baik. Dari pengujian terhadap 70 citra uji (35 normal dan 35 katarak), diperoleh distribusi hasil klasifikasi yang sangat akurat, ditandai dengan hanya dua kasus false negative dan tanpa false positive.

3. Tingkat akurasi deteksi katarak menggunakan CNN mencapai rata-rata 96,56% pada validasi silang dan 97,14% pada pengujian akhir, dengan *precision*, *recall*, dan *F1-Score* yang juga tinggi untuk kedua kelas. Hal ini menunjukkan bahwa model memiliki kinerja yang cukup memuaskan dalam mengidentifikasi kondisi mata.

5.2 Saran

Berikut beberapa hal yang disarankan dalam penelitian ini:

1. Penelitian berikutnya dapat mencoba arsitektur lain seperti ResNet, DenseNet, atau *EfficientNet* versi yang lebih tinggi (misalnya B1 atau B3) untuk melihat apakah hasil deteksi bisa lebih akurat.

2. Augmentasi yang digunakan saat ini sudah cukup baik, namun dapat diuji kembali kombinasi transformasi yang lebih bervariasi seperti brightness adjustment, contrast enhancement, atau CLAHE (Contrast Limited Adaptive Histogram Equalization) untuk meningkatkan keragaman data pelatihan, khususnya untuk kondisi pencahayaan yang ekstrem.
3. Meskipun akurasi cukup tinggi, terdapat perbedaan performa antar *fold*, khususnya pada *Fold* 4 dan 5. Disarankan untuk menambahkan teknik regularisasi tambahan seperti L2 regularization pada layer Dense atau melakukan *data cleaning* pada dataset untuk menghindari noise atau outlier.
4. Perlu dilakukan pengujian model terhadap data dari sumber yang berbeda untuk melihat apakah model tetap bekerja dengan baik pada data baru yang belum pernah digunakan sebelumnya.
5. Perlu ditambahkan visualisasi seperti Grad-CAM untuk menunjukkan bagian citra mata yang menjadi fokus model saat mendeteksi katarak, sehingga hasilnya lebih mudah dipahami oleh dokter atau pengguna awam.

DAFTAR PUSTAKA

[1] W. Nugraha and S. Cahyana, Buku Referensi Katarak dan Penanganannya (Edisi Revisi). Jember: Unej Press, 2021.

[2] F. Ramadhani and A. Satria, "Implementasi Algoritma Convolutional Neural Network dalam Mengidentifikasi Dini Penyakit pada Mata Katarak," Jurnal Teknologi Informasi, 2023.

[3] R. B. J. Simanjuntak, A. Fariza, and Y. R. Prayogi, "Eye Disease Classification Based on Fundus Images Using Convolutional Neural Network," IES 2023 - Int. Electron. Symp., vol. 6, pp. 563–568, 2023, doi: 10.1109/IES59143.2023.10242558.

[4] P. Mullangi, S. M. Manasa, S. S. N. Kowsalya, and G. R. Naidu, "Automated Cataract Detection Using Deep Learning and Pre-Trained CNN Models," International Journal of Advanced Computer Science and Applications, vol. 13, no. 3, pp. 1895–1904, 2024.

[5] "Cataract dataset," Kaggle. Accessed: May 31, 2025. [Online]. Available: https://www.kaggle.com/datasets/nandanp6/cataract-image-dataset

[6] G. Pakuliene et al., "Anterior segment optical coherence tomography imaging and ocular biometry in cataract patients with open angle glaucoma comorbidity," BMC Ophthalmology, vol. 21, no. 1, p. 127, Dec. 2021, doi: 10.1186/s12886-021-01874-x.

[7] "Modul Pengolahan Citra Digital 2017 | PDF | Seni," Scribd. Accessed: Jun. 03, 2025. [Online]. Available: https://id.scribd.com/document/429367669/Modul-Pengolahan-Citra-Digital-2017

[8] R. S. Segall and P. Rajbhandari, "Image Processing of Big Data for Plant Diseases of Four Different Plant Categories," Int. J. Comput. Vis. Image Process., vol. 14, no. 1, pp. 1–32, 2024, doi: 10.4018/ijcvip.353913.

[9] R. M. M.Kom., CEH., CCNA, Pengantar Dasar Deep Learning. Jakarta: Serasi Media Teknologi, 2024.

[10] M. Z. Alom et al., "A state-of-the-art survey on deep learning theory and architectures," Electronics, vol. 8, no. 3, pp. 1–67, 2019, doi: 10.3390/electronics8030292.

[11] A. Peryanto, A. Yudhana, and R. Umar, "Klasifikasi Citra Menggunakan Convolutional Neural Network dan K Fold Cross Validation," J. Appl. Inform. Comput., vol. 4, no. 1, pp. 45–51, 2020, doi: 10.30871/jaic.v4i1.2017.

[12] H. Kinsley and D. Kukieła, Neural Networks from Scratch in *Python*, 2020.

[13] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," arXiv preprint, arXiv:1905.11946, 2019.

[14] M. T.R et al., "An XAI-enhanced *EfficientNetB0* framework for *precision* brain tumor detection in MRI imaging," Elsevier, vol. 410, p. 110227, 2024.

[15] G. Balaji, R. Sen, and H. Kirty, "Detection and Classification of Brain Tumors Using Deep Convolutional Neural Networks," arXiv preprint, arXiv:2208.13264, 2022, doi: 10.48550/arXiv.2208.13264.

[16] D. Padalia, A. Mazumdar, and B. Singh, "A CNN-LSTM Combination Network for Cataract Detection using Eye Fundus Images," arXiv preprint, arXiv:2210.16093, 2022.

[17] A. N. Fajrina et al., "Penerapan Arsitektur EfficientNet-B0 Pada Klasifikasi Leukimia Tipe Acute Lymphoblastik Leukimia," J. Ris. Rekayasa Elektro, vol. 6, no. 1, p. 59, Jun. 2024, doi: 10.30595/jrre.v6i1.22090.

[18] J. Heaton, "Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep Learning," Genet. Program. Evolvable Mach., vol. 19, pp. 305–307, 2018, doi: 10.1007/s10710-017-9314-z.

[19] F. Chollet, Deep Learning with *Python*. New York: Manning Publications, 2018.

[20] A. A. Rakib et al., "EfficientNet-Based Model for Automated Classification of Retinal Diseases Using Fundus Images," Eur. J. Comput. Sci. Inf. Technol., vol. 12, no. 8, pp. 48–61, Aug. 2024.

[21] M. G. Summa et al., "Large-Scale Machine Learning with Stochastic Gradient Descent," in Statistical Learning and Data Science, Chapman and Hall/CRC, 2011, pp. 33–42.

[22] A. Buslaev et al., "Albumentations: Fast and Flexible Image Augmentations," Information, vol. 11, no. 2, p. 125, Feb. 2020, doi: 10.3390/info11020125.

[23] C. Shorten and T. M. Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," J. Big Data, vol. 6, no. 1, p. 60, Dec. 2019, doi: 10.1186/s40537-019-0197-0.

[24] Z. Zhong et al., "Random Erasing Data Augmentation," Proc. AAAI Conf. Artif. Intell., vol. 34, no. 07, pp. 13001–13008, 2020.

[25] N. Tajbakhsh et al., “Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?,” IEEE Trans. Med. Imaging, vol. 35, no. 5, pp. 1299–1312, May 2016, doi: 10.1109/TMI.2016.2535302.

[26] S. Yun et al., “CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features,” arXiv preprint, arXiv:1905.04899, 2019.

[27] S. Ruder, “An Overview of Gradient Descent Optimization Algorithms,” arXiv preprint, arXiv:1609.04747, 2017.

[28] J. Bergstra and Y. Bengio, “Random Search for Hyper-Parameter Optimization,” J. Mach. Learn. Res., vol. 13, no. Feb, pp. 281–305, 2012.

[29] L. González-Castro et al., “Impact of Hyperparameter Optimization to Enhance Machine Learning Performance: A Case Study on Breast Cancer Recurrence Prediction,” Appl. Sci., vol. 14, no. 13, p. 5909, Jul. 2024.

[30] M. Haris, “Analysis of the Application of Hyperparameter Tuning in Machine Learning to Increase the *Accuracy* of Sales-Level Prediction,” J. Inform., vol. 7, no. 1, 2024.

[31] G. N. Ahamad et al., “Influence of Optimal Hyperparameters on the Performance of Machine Learning Algorithms for Predicting Heart Disease,” Processes, vol. 11, no. 3, p. 734, Mar. 2023.

[32] S. Raschka, J. Patterson, and C. Nolet, “Machine Learning in *Python*: Main Developments and Technology Trends,” Information, vol. 11, no. 4, p. 193, 2020.

[33] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” arXiv preprint, arXiv:1412.6980, 2017.

[34] J. Duchi, E. Hazan, and Y. Singer, “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization,” J. Mach. Learn. Res., vol. 12, pp. 2121–2159, 2011.

[35] R. Kohavi, “A Study of Cross-Validation and Bootstrap for *Accuracy* Estimation and Model Selection,” Proc. IJCAI, vol. 14, pp. 1137–1145, 1995.

[36] T. Fawcett, “An Introduction to ROC Analysis,” Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, Jun. 2006.

[37] S. Raschka and V. Mirjalili, *Python* Machine Learning. Packt Publishing, 2019.

[38] G. van Rossum, “The *Python* Language Reference.” *Python* Software Foundation.

[39] M. Lutz, Learning *Python*, 5th ed. Sebastopol: O’Reilly Media, 2013.

[40] "TensorFlow," TensorFlow.org. Accessed: May 27, 2025. [Online]. Available: https://www.tensorflow.org/?hl=id

[41] "Keras: Deep Learning for Humans," Keras.io. Accessed: May 27, 2025. [Online]. Available: https://keras.io/

[42] C. R. Harris et al., "Array Programming with NumPy," Nature, vol. 585, no. 7825, pp. 357–362, Sep. 2020.

[43] "Matplotlib — Visualization with *Python*," Matplotlib.org. Accessed: May 28, 2025. [Online]. Available: https://matplotlib.org/

[44] M. Waskom, "Seaborn: Statistical Data Visualization," J. Open Source Softw., vol. 6, no. 60, p. 3021, 2021.

[45] "scikit-learn: Machine Learning in *Python*," scikit-learn.org. Accessed: May 28, 2025. [Online]. Available: https://scikit-learn.org/stable/

[46] "OpenCV Documentation," OpenCV.org. Accessed: May 28, 2025. [Online]. Available: https://docs.opencv.org/

[47] "Pillow (PIL Fork)," pillow.readthedocs.io. Accessed: May 28, 2025. [Online]. Available: https://pillow.readthedocs.io/en/stable/

[48] F. Wilyani, Q. N. Arif, and F. Aslimar, "Pengenalan Dasar Pemrograman *Python* dengan *Google Colab* oratory," J. Pelayanan dan Pengabdian Masy. Indones., vol. 3, no. 1, pp. 08–14, 2024, doi: 10.55606/jppmi.v3i1.1087.

[49] A. N. Ilahy, "Dataset Deteksi Katarak (Versi Mentah)," 2025. [Online]. Available: https://drive.google.com/drive/folders/1kmtI5yEmdCkagQe--uJ8OAsnW77gpcSs

[50] A. N. Ilahy, "Hasil Split Data, Folder Train (Dataset Pelatihan)," 2025. [Online]. Available: https://drive.google.com/drive/folders/1rDsxslK7_dF4G36vxCVFwaDLUAgLvkcf

[51] A. N. Ilahy, "Hasil Split Data, Folder Val (Data Validasi)," 2025. [Online]. Available: https://drive.google.com/drive/folders/1rE2KioeVY7M-OD5XaQ-PTEOlTq0Jt_i6

[52] A. N. Ilahy, "Hasil Split Data, Folder Test (Data Pengujian)," 2025. [Online]. Available: https://drive.google.com/drive/folders/1raj-CcAa2-5gDeEorv3fqHJsEB-TczO8

[53] A. N. Ilahy, "Hasil Augmentasi, Folder train_augmented," 2025. [Online]. Available: https://drive.google.com/drive/folders/1rzquEQjQyR-iBpzNPovyqzoxA_ZMIBCy

[54] "*Google Colab* , dokumentasi kode program hyperparameter tuning optimizer" Accessed: Jun. 18, 2025. [Online]. Available: https://colab.research.google.com/drive/1ybFDGMXt8VBv26e15nIDGsh7HTCAMOrT

[55] "*Google Colab* , dokumentasi kode program 5 fold cross validation" Accessed: Jun. 18, 2025. [Online]. Available: https://colab.research.google.com/drive/1dQCc_3yQDou1zEwNh7LSiCInGQplJIdZ?usp=sharing

[56] "*Google Colab* , dokumentasi kode program Demo aplikasi web" Accessed: Jun. 18, 2025. [Online]. Available: https://colab.research.google.com/drive/1ybFDGMXt8VBv26e15nIDGsh7HTCAMOrT?usp=sharing