JTIM : Jurnal Teknologi Informasi dan Multimedia

p-ISSN : 2715-2529
e-ISSN : 2684-9151
https://journal.sekawan-org.id/index.php/jtim

Implementasi *Machine Learning* untuk Mendeteksi Penyakit Katarak menggunakan Kombinasi Ekstraksi Fitur dan *Neural Network* Berdasarkan Citra

Maspaeni [1], Bahtiar Imran [2,*], Alfian Hidayat[3] dan Surni Erniwati[4]

[1] Program Studi Sistem Informasi, Universitas Teknologi Mataram, Indonesia
[2] Program Studi Rekayasa Sistem Komputer, Universitas Teknologi Mataram, Indonesia
[3] Program Studi Teknik Informatika, Universitas Teknologi Mataram, Indonesia
[4] Program Studi Manajemen Informatika, Universitas Teknologi Mataram, Indoensia

* Korespondensi: bahtiarimranlombok@gmail.com

Sitasi: Maspaeni, M.; Imran, B.; Hidayat, A.; Erniwati, S. (**2025**). Implementasi *Machine Learning* untuk Mendeteksi Penyakit Katarak menggunakan Kombinasi Ekstraksi Fitur dan *Neural Network* Berdasarkan Citra. JTIM: Jurnal Teknologi Informasi Dan Multimedia, 7(2), 232-251. https://doi.org/10.35746/jtim.v7i2.621

Diterima: 25-10-2024
Direvisi: 21-11-2024
Disetujui: 24-12-2024

Abstract: According to data from the World Health Organization (WHO), more than 1.3 billion people worldwide experience visual impairments, with *Cataract*s being one of the main causes. *Cataract*s are an eye condition characterized by clouding of the lens, which can lead to blindness if left untreated. This study aims to accurately detect *Cataract*s using a combination of feature extraction and neural networks, utilizing digital fundus images. The *Dataset* used consists of 600 fundus images divided into 80% for training and 20% for testing. The feature extraction process is performed to identify distinctive characteristics of the images relevant to *Cataract* diagnosis. These features are then analyzed by a neural network to recognize patterns indicative of *Cataract*s. To optimize performance, this study implements a *hypertuning* process. Before tuning, the initial model achieved an accuracy of 0.83, with precision, recall, F1-score of 0.83, and an AUC of 0.92. After four stages of *hypertuning*, the model's performance improved progressively. The first tuning achieved an accuracy of 0.85, with precision, recall, and F1-score of 0.85, and an AUC of 0.93. In the second tuning, accuracy increased to 0.88, with precision of 0.87, recall of 0.88, F1-score of 0.87, and an AUC of 0.93. The third tuning maintained an accuracy of 0.88, with precision improving to 0.90, recall at 0.87, F1-score of 0.88, and an AUC of 0.94. The fourth tuning delivered the best results, with an accuracy of 0.90, precision of 0.92, recall of 0.89, F1-score of 0.90, and an AUC of 0.94. These results demonstrate that the *hypertuning* process plays a significant role in improving model performance.

Keywords: *Cataract*, machine learning, feature extraction, neural network

Abstrak: Berdasarkan data *World Health Organization (WHO),* lebih dari 1,3 miliar orang di dunia mengalami gangguan penglihatan, dengan katarak sebagai salah satu penyebab utama. Katarak merupakan penyakit mata yang ditandai dengan kekeruhan pada lensa mata, yang dapat menyebabkan kebutaan jika tidak ditangani. Penelitian ini bertujuan untuk mendeteksi katarak secara akurat menggunakan kombinasi ekstraksi fitur dan neural network, dengan memanfaatkan citra fundus mata digital. *Dataset* yang digunakan terdiri dari 600 citra fundus mata yang dibagi menjadi 80% untuk pelatihan dan 20% untuk pengujian. Proses ekstraksi fitur dilakukan untuk mengidentifikasi karakteristik khas dari citra yang relevan dengan diagnosis katarak. Fitur ini kemudian dianalisis oleh neural network untuk mengenali pola yang menunjukkan keberadaan katarak. Untuk mengoptimalkan performa, penelitian ini menerapkan proses *hypertuning*. Sebelum tuning, model awal menunjukkan akurasi 0.83, presisi 0.83, recall 0.83, F1-score 0.83, dan AUC 0.92. Setelah empat tahap *hypertuning*, performa model meningkat secara bertahap. Tuning pertama menghasilkan akurasi 0.85, presisi 0.85, recall 0.85, F1-score 0.85, dan AUC 0.93. Pada tuning kedua, akurasi meningkat menjadi 0.88, dengan presisi 0.87, recall 0.88, F1-score 0.87, dan AUC 0.93. Tuning ketiga

mempertahankan akurasi 0.88, dengan presisi 0.90, recall 0.87, F1-score 0.88, dan AUC 0.94. Tuning keempat memberikan hasil terbaik dengan akurasi 0.90, presisi 0.92, recall 0.89, F1-score 0.90, dan AUC 0.94. Hasil ini menunjukkan bahwa proses *hypertuning* berperan signifikan dalam meningkatkan kinerja model.

Kata kunci: katarak, *machine learning*, ektraksi fitur, *neural network*

1. Pendahuluan

Mata merupakan salah satu indera utama yang berperan penting dalam memperoleh informasi visual untuk mendukung berbagai aktivitas sehari-hari. Namun, gangguan penglihatan sering terjadi dan dapat bervariasi mulai dari ringan hingga parah. Berdasarkan data *World Health Organization* (WHO), lebih dari 1,3 miliar orang di seluruh dunia mengalami gangguan penglihatan, dengan sekitar 36 juta di antaranya menderita kebutaan, dan katarak menjadi penyebab terbesar masalah ini secara global [1]. Katarak merupakan kondisi dimana lensa mata menjadi keruh dan menghalangi cahaya masuk dengan baik [2], sehingga menyebabkan gangguan penglihatan yang semakin memburuk dari waktu ke waktu bahkan menyebabkan kebutaan.

Saat ini teknologi memegang peranan penting dalam bidang medis, khususnya dalam identifikasi berbagai penyakit. Perkembangan *Artificial Intelligence* (AI) menjadi solusi diagnosis medis yang sangat efektif karena kemampuannya menganalisis data dengan cepat dan akurat. Salah satu cabang dari AI yakni *machine learning* memungkinkan sistem untuk belajar dari data yang ada serta dapat mengidentifikasi pola-pola kompleks dan menghasilkan prediksi [3].

Meskipun pendekatan berbasis ekstraksi fitur dan neural network menunjukkan potensi besar dalam mendeteksi katarak, beberapa tantangan tetap ada dalam penerapannya. Salah satu tantangan utama adalah kualitas dan variasi data yang digunakan. Data citra fundus mata sering kali memiliki variasi dalam kualitas gambar akibat pencahayaan yang tidak merata dan adanya noise. Untuk itu penelitian mendalam perlu dilakukan dalam pengembangan metode ekstraksi fitur, desain neural network yang optimal, dan pengelolaan variasi data sangat penting untuk mengatasi tantangan seperti kualitas gambar yang bervariasi, risiko *overfitting*, dan keterbatasan generalisasi model.

Beberapa penelitian mengenai klasifikasi penyakit katarak telah dilakukan dengan berbagai metode dan menghasilkan hasil yang bervariasi. Salah satu penelitian oleh Bu'ulölö et al., 2021 menerapkan *Convolutional Neural Network (CNN)* untuk klasifikasi katarak berdasarkan 260 citra fundus mata, dengan akurasi tertinggi mencapai 92,93% [4]. Penelitian lainnya oleh Firdaus et al., 2022 menerapkan CNN dengan 512 citra mata dan mencapai akurasi tertinggi sebesar 99,74% [1]. Sementara itu, Penelitian Ramadhani et al., 2023 juga menggunakan CNN, tetapi dengan *Dataset* sebanyak 612 citra mata, menghasilkan akurasi tertinggi sebesar 92,5% [5]. [6] ini menunjukkan bahwa teknologi dapat memainkan peran penting dalam mengatasi kekurangan dokter spesialis di daerah yang kurang terlayani. Selain itu, sistem deteksi katarak berbasis jaringan saraf konvolusional yang dikembangkan oleh Karamihan et al. menunjukkan akurasi yang signifikan, menandakan bahwa pendekatan berbasis Machine Leaning dapat memberikan hasil yang lebih baik dibandingkan metode pemeriksaan mata tradisional. [7] lebih lanjut juga menunjukkan bahwa algoritma pembelajaran mendalam dapat digunakan untuk mengotomatisasi klasifikasi video operasi katarak, yang dapat meningkatkan efisiensi dan akurasi dalam pengenalan fase bedah. [8] menunjukkan bahwa integrasi Machine Leaning dalam prosedur bedah dapat meningkatkan hasil klinis secara keseluruhan. Meskipun banyak kemajuan telah dicapai, tantangan tetap ada dalam pengembangan sistem deteksi katarak yang sepenuhnya otomatis dan dapat diandalkan. Penelitian ini

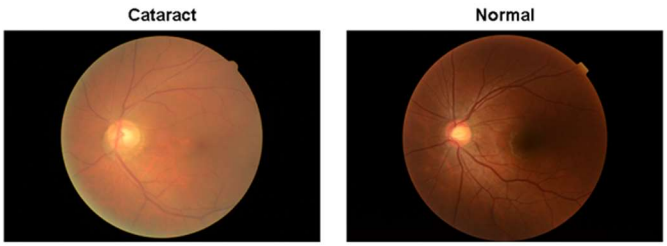
menunjukkan bahwa metode CNN dapat secara efektif mengekstrak fitur-fitur penting dari gambar, yang memungkinkan diagnosis yang lebih akurat dibandingkan dengan metode konvensional [9]. Dalam konteks aplikasi klinis, Wu et al. mencatat bahwa meskipun ada kemajuan dalam penggunaan AI untuk deteksi penyakit mata seperti diabetic retinopathy, katarak masih belum dikelola dengan platform yang dapat diterapkan secara klinis menggunakan algoritma deep learning [10], salah satu tantangan utama dalam penelitian ini adalah memastikan akurasi yang lebih tinggi saat mengklasifikasikan gambar ke dalam tahap katarak. Elloumi mengembangkan metode grading katarak yang menggabungkan deep convolutional networks dengan teknik stacking ensemble learning untuk meningkatkan performa klasifikasi [11].

Dalam penelitian ini, kombinasi metode ekstraksi fitur dan *neural network* digunakan untuk mendeteksi katarak melalui citra fundus mata. Ekstraksi fitur bertujuan untuk mengambil karakteristik khas dari citra digital [12], yang kemudian dianlisis menggunakan *neural network*. *Neural network* yang terinspirasi dari cara kerja *neuron* pada otak manusia digunakan untuk menyelesaikan masalah klasifikasi atau prediksi [13], Selain itu, *Dataset* yang digunakan dalam penelitian ini berjumlah 300 citra fundus mata yang kemudian diperbanyak melalui teknik augmentasi, menghasilkan total 600 citra untuk pelatihan dan pengujian model. Proses augmentasi ini dilakukan untuk memperkaya variasi data, meningkatkan robustitas model, dan mengurangi risiko *overfitting*, sehingga model dapat belajar dari berbagai variasi citra yang lebih banyak. Penelitian ini juga menggunakan teknik *hypertuning* untuk mengoptimalkan performa model yang berbeda dari penelitian sebelumnya. Berbeda dengan studi-studi lain yang menggunakan pendekatan standar dalam pelatihan model, penelitian ini menerapkan proses *hypertuning* yang bertujuan untuk menyesuaikan hyperparameter jaringan neural secara sistematis dan efisien. *Hypertuning* ini berfokus pada pencarian kombinasi parameter terbaik, seperti jumlah lapisan, jumlah neuron, dan *learning rate*, yang dapat meningkatkan akurasi dan efisiensi model dalam mendeteksi katarak. Metode ini menjadi pembeda penting dan inovasi dalam penelitian ini, karena tidak banyak penelitian sebelumnya yang mengaplikasikan *hypertuning* dalam klasifikasi citra fundus mata untuk deteksi katarak.

2. Bahan dan Metode

2.1. Pengumpulan Data

Pada tahapan ini, dilakukan pengumpulan data untuk keperluan penelitian. Data yang dikumpulkan berupa citra fundus mata.



Gambar 1. Sampel citra fundus mata katarak dan normal pada *Dataset.*

Berdasarkan Gambar 1, *Dataset* citra fundus mata diperoleh melalui *platform* Kaggle https://www.kaggle.com/Datasets/jr2ngb/CataractDataset yang terdiri dari 400 citra fundus mata, dimana terdapat 300 citra berlabel "Normal" dan 100 citra berlabel "*Cataract*".

2.2. Preparation Data

Sebelum data dapat digunakan untuk analisis lebih lanjut, langkah yang harus dilakukan yakni prapemrosesan data. Prapemrosesan data merupakan tahapan awal dalam pengolahan data yang bertujuan untuk mengubah data mentah menjadi data yang

berkualitas sehingga data tersebut dapat diolah secara efektif [14]. Berikut merupakan tahapan dalam prapemrosesan data yang dilakukan pada penelitian:

2.2.1. Pembagian *Dataset*

Data berlabel "Normal" dan "*Cataract*" dibagi menjadi 80% untuk data *training* dan 20% untuk data *testing*. Berikut merupakan rincian pembagian dari *Dataset*:

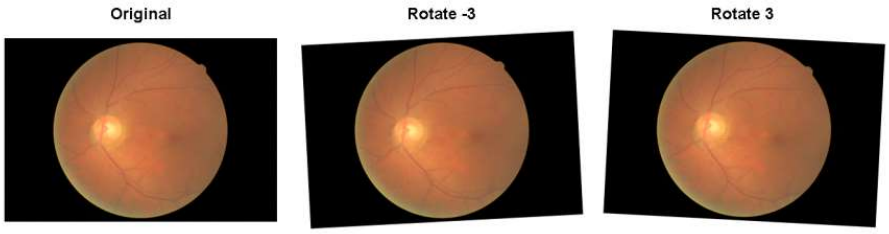
Tabel 1. Rincian pembagian *Dataset*.

Label	*Train* (80%)	*Test* (20%)	Jumlah (100%)
Normal	240	60	300
Cataract	80	20	100
	Total		**400**

Berdasarkan Tabel 1, jumlah *Dataset* yang akan digunakan untuk analisis menunjukkan ketidakseimbangan, sehingga akan dilakukan augmentasi data pada citra berlabel "*Cataract*" guna meningkatkan variasi citra.

2.2.2. Augmentasi Data

Augmentasi data merupakan metode yang dapat digunakan untuk mengatasi ketidakmampuan model secara akurat menggeneralisasi data baru (*overfitting*) dengan memperluas variasi dalam kumpulan data secara efektif. Umumnya augmentasi data dilakukan dengan cara mentransformasikan data atau bisa juga disebut salinan dari data asli tanpa mengubah label pada setiap bagian data [15].



Gambar 2. Augmentasi citra berlabel "*Cataract*".

Augmentasi dilakukan dengan teknik transformasi geometri berupa rotasi, dimana setiap citra diputar sebesar -3° dan 3°, hal ini membantu meningkatkan variasi citra tanpa mengubah bentuk objek aslinya.

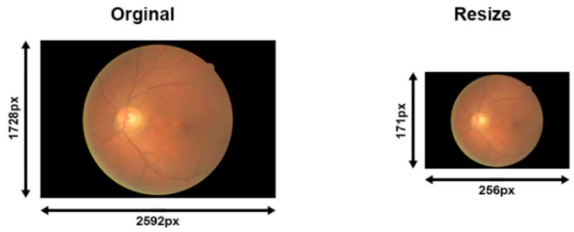
Tabel 2. Rincian *Dataset* setelah augmentasi data.

Label	*Train* (80%)	*Test* (20%)	Jumlah (100%)
Normal	240	60	300
Cataract	240	60	300
	Total		**600**

Berdasarkan Tabel 2, jumlah *Dataset* menunjukkan keseimbangan setelah dilakukan augmentasi data pada citra berlabel "*Cataract*". Dengan demikian, setiap label kini memiliki 240 citra untuk data *training* dan 60 citra untuk data *testing*.

2.2.3. Resize Citra

Resize dilakukan dengan mengatur dimensi lebar citra menjadi 256px tanpa mengubah rasio citra asli untuk mempermudah proses komputasi dan mengurangi kebutuhan akan sumber daya komputasi yang besar. Pada Gambar 3, dimensi pada citra asli yang awalnya 2592×1728px di*resize* menjadi 256×171px.

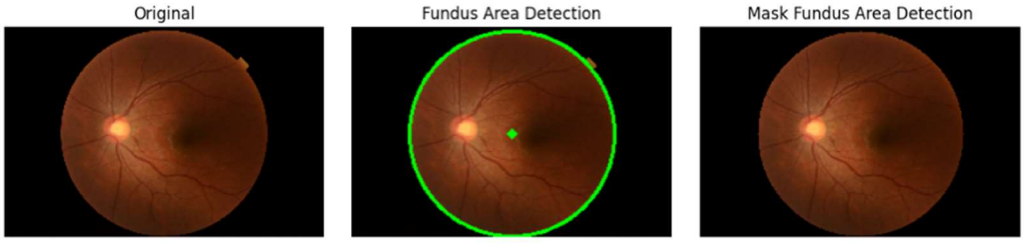


Gambar 3. *Resize* citra

2.3. Segmentasi Data

Segmentasi data merupakan proses yang bertujuan untuk memisahkan objek yang ingin diamati dari objek lain yang tidak relevan. Hal ini dilakukan untuk mendapatkan informasi yang tersimpan dalam citra tersebut. Pemisahan objek bermanfaat untuk mengidentifikasi dan mengekstrak objek-objek spesifik yang diperlukan selama proses segmentasi [16].

Salah satu teknik segmentasi yang umum digunakan yakni *hough transform*. *Hough transform* merupakan teknik mendeteksi berbagai bentuk geometri seperti lingkaran, garis, *elips*, dan sebagainya dalam citra. *Hough circle transform* merupakan salah satu varian dari *hough transform* yang khusus digunakan untuk mendeteksi bentuk lingkaran [17].



(a) (b) (c)

Gambar 4. Segmentasi citra fundus mata: (a) Citra asli; (b) Deteksi objek pada citra; (c) *Masking* hasil deteksi objek pada citra.

Berdasarkan Gambar 4, dilakukan deteksi objek dan memisahkan area penting seperti bagian retina, pembuluh darah retina, dan diskus optikus dari latar belakang pada citra asli.

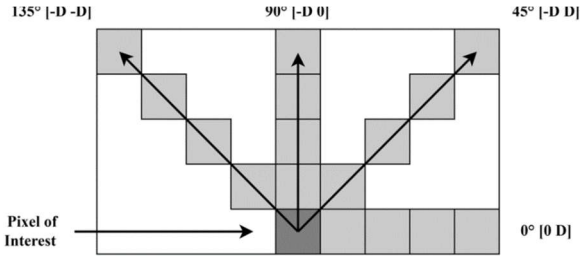
2.4. Ekstraksi Fitur GLCM

Salah satu metode yang dapat diterapkan dalam ekstraksi fitur pada citra yakni *Gray Level Co-occurrence Matrix* (GLCM). *Gray Level Co-occurrence Matrix* (GLCM) merupakan metode analisis statistik yang digunakan untuk ekstraksi fitur dari citra [18].

Gambar 5. Hasil konversi dari *masking* hasil deteksi objek pada citra menjadi citra *grayscale.*

Berdasarkan Gambar 4 (c), citra harus diubah terlebih dahulu menjadi *grayscale* sehingga terlihat seperti Gambar 5. Hal ini penting karena GLCM beroperasi pada citra skala abu-abu, dimana intensitas *pixel* menjadi fokus analisis. Penggunaan metode ini

bertujuan untuk menghitung frekuensi pasangan *pixel* dengan intensitas tertentu (*gray level*) yang muncul pada jarak dan sudut tertentu dalam citra. Jarak direpresentasikan sebagai *pixel* dan ditentukan sebesar 1px, sedangkan sudut direpresentasikan dalam derajat dengan interval 45°, yakni 0°, 45°, 90°, dan 135° [19].



Gambar 6. Arah rotasi GLCM.

Dari representasi pada Gambar 5, dapat diperoleh berbagai fitur yang menggambarkan tekstur citra. Fitur-fitur yang digunakan dalam penelitian yakni *contrast*, *correlation*, *homogeneity*, dan *energy*. Berikut merupakan rumus persamaan dari fitur yang digunakan:

$$contrast = \sum_{i,j=0}^{levels-1} P_{i,j}(i-j)^2 \quad (1)$$

$$correlation = \sum_{i,j=0}^{levels-1} P_{i,j}\left[\frac{(i-\mu_i)(j-\mu_j)}{\sqrt{(\sigma_i^2)(\sigma_j^2)}}\right] \quad (2)$$

$$homogeneity = \sum_{i,j=0}^{levels-1} \frac{P_{i,j}}{1+(i-j)^2} \quad (3)$$

$$energy = \sqrt{\sum_{i,j=0}^{levels-1} P_{i,j}^2} \quad (4)$$

Keterangan:

i, j	:	Indeks baris dan kolom dalam matriks GLCM.
$P_{i,j}$	:	Nilai probabilitas di posisi (i, j) dalam matriks GLCM.
μ_i, μ_j	:	Rata-rata dari indeks baris (i) dan kolom (i) yang dihitung dari matriks GLCM.
σ_i, σ_j	:	Standar deviasi dari indeks baris (i) dan kolom (j).
$levels$	:	Jumlah tingkat keabuan dalam citra. Pada citra *grayscale* 8-bit terdapat 256 tingkat keabuan.

2.5. Implementasi ANN

Salah satu jenis *neural network* dalam kecerdasan buatan yakni *Artificial Neural Network* (ANN) atau bisa juga disebut jaringan saraf tiruan. ANN dirancang untuk meniru cara manusia dalam berpikir dan memiliki kemampuan untuk memproses informasi, termasuk dari citra [18]. ANN dapat mempelajari pola dan fitur kompleks dalam data citra, sehingga mampu mengenali dan mengklasifikasikan objek dengan akurasi tinggi.

```python
# Fungsi untuk membangun model
def build_model(input_dim, output_dim):
    model = Sequential()
    model.add(Input(shape=(input_dim,)))
    model.add(Dense(128, activation='relu', kernel_regularizer=l2(0.01)))
    model.add(BatchNormalization())
    model.add(Dropout(0.1))
    model.add(Dense(64, activation='relu', kernel_regularizer=l2(0.01)))
    model.add(BatchNormalization())
    model.add(Dropout(0.1))
    model.add(Dense(32, activation='relu', kernel_regularizer=l2(0.01)))
    model.add(BatchNormalization())
    model.add(Dropout(0.1))
    model.add(Dense(output_dim, activation='sigmoid'))
    return model

# Membangun dan mengompilasi model
model = build_model(X_train.shape[1], y_train.shape[1])
optimizer = Adam(learning_rate=0.01)
model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=['accuracy'])

# Menampilkan ringkasan model
model.summary()

epoch=25

# Callback untuk early stopping
early_stopping = EarlyStopping(monitor='val_loss', patience=10)

# Melatih model
history = model.fit(X_train, y_train,
                    validation_data=(X_val, y_val),
                    epochs=epoch,
                    batch_size=32,
                    callbacks=[early_stopping],
                    verbose=1)

# Mengevaluasi model pada data uji
test_loss, test_accuracy = model.evaluate(X_test, y_test, verbose=0)
```

Gambar 7. Arsitektur model.

Berdasarkan Gambar 7, model ANN dibangun menggunakan Keras Sequential API. Arsitektur model didesain secara khusus untuk menangani kompleksitas data dengan mengintegrasikan berbagai lapisan dan teknik yang mendukung kinerja serta generalisasi model. Berikut merupakan detail dari arsitektur model yang dibuat:

2.5.1. *Input Layer*

Input layer menerima data dengan dimensi yang sama dengan jumlah fitur dalam *Dataset*.

2.5.2. Hiden Layer

Hidden layer terdiri dari tiga lapisan *Dense* (*fully connected*), pada lapisan pertama memiliki 128 *neuron*, lapisan kedua memiliki 64 *neuron*, dan lapisan ketiga memiliki 32 *neuron*. Setiap lapisan menggunakan fungsi aktivasi *Rectified Linear Unit* (ReLU) untuk menangani non-linearitas dalam data. Untuk mengurangi risiko *overfitting*, diterapkan beberapa teknik regularisasi, seperti Regularisasi L2 dengan nilai sebesar 0.01 yang bertujuan memberikan penalti terhadap bobot yang besar dan *Dropout* dengan rasio 0.1 yang secara acak menonaktifkan sejumlah neuron selama proses pelatihan. Menerapakan *Batch Normalization* pada setiap lapisan *Dense* untuk menormalkan *output* dari lapisan tersebut, mempercepat pelatihan dan stabilisasi model.

2.5.3. Output Layer

Output layer menggunakan fungsi aktivasi *sigmoid* untuk menghasilkan probabilitas dari kelas yang ada, sehingga cocok untuk masalah klasifikasi dua kelas.

Model yang telah dibangun, dikompilasi menggunakan *optimizer* Adam dengan *learning rate* sebesar 0.01 dan *loss function binary_crossentropy*, yang sesuai untuk masalah klasifikasi dua kelas. Model dilatih menggunakan data *training*, dimana sebanyak 80% untuk pelatihan dan 20% untuk validasi, serta jumlah *epoch* sebesar 25. *Callback early stopping* diterapkan untuk menghindari *overfitting* dengan menghentikan pelatihan jika tidak ada perbaikan pada *validation loss* dalam 10 *epoch* berturut-turut.

2.6. Evaluasi Kinerja

Evaluasi kinerja merupakan proses yang bertujuan untuk menilai seberapa efektif suatu sistem atau model dalam melaksanakan tugas yang telah ditentukan [20], [21]. Proses ini mencakup berbagai metrik untuk mengukur keakuratan, efisiensi, dan efektivitas model dalam mencapai hasil yang diharapkan. Metrik yang digunakan dalam evaluasi kinerja pada penelitian ini yakni *confusion matrix*, akurasi, presisi, *recall*, *f1- score*, dan Kurva ROC/AUC.

Tabel 3. *Confusion matrix*

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

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

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

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

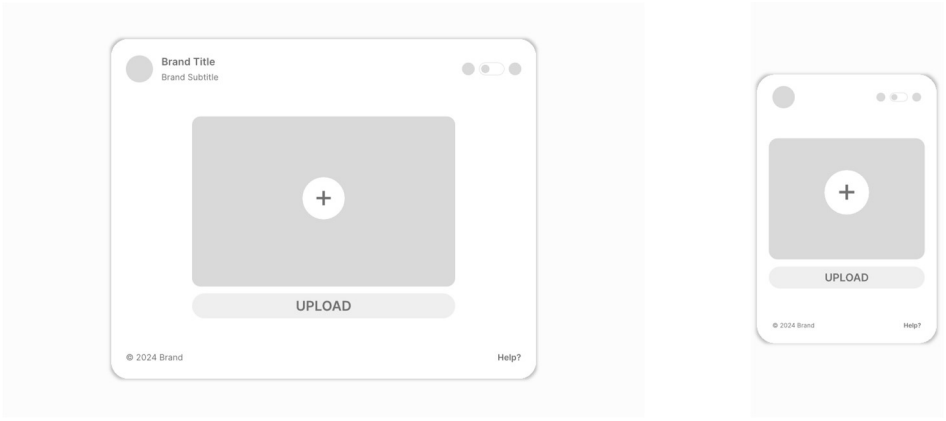
$$F1 - Score = 2 \times \frac{Presisi \times Recall}{Presisi + Recall} \quad (8)$$

Keterangan:

True Positive (TP) : Jumlah data yang sebenarnya positif dan diprediksi positif.
False Negative (FN) : Jumlah data yang sebenarnya positif tetapi diprediksi negatif.
False Positive (FP) : Jumlah data yang sebenarnya negatif tetapi diprediksi positif.
True Negative (TN) : Jumlah data yang sebenarnya negatif dan diprediksi negatif.

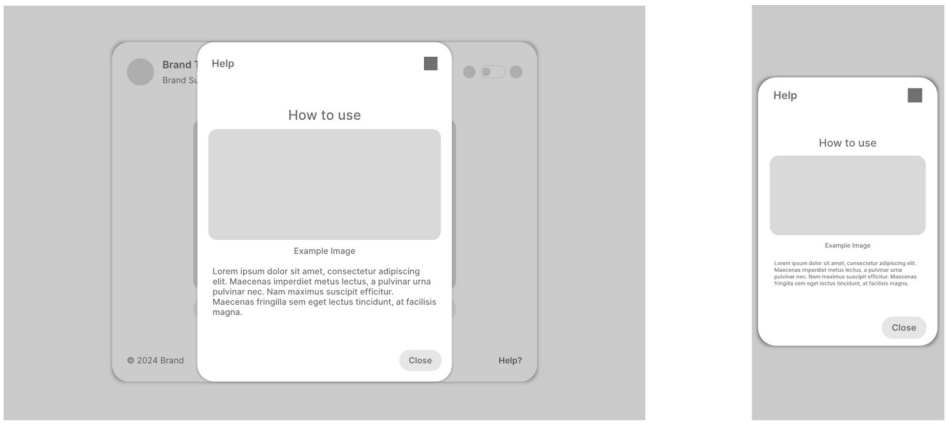
2.7. *Interface Design* dan *Deployment*

Interface design (desain antarmuka) dilakukan dengan tujuan untuk memudahkan proses perancangan *website* yang responsif pada tampilan *desktop* dan ponsel sesuai kebutuhan yang mencakup *home page*, *help modal page*, dan *result page*.



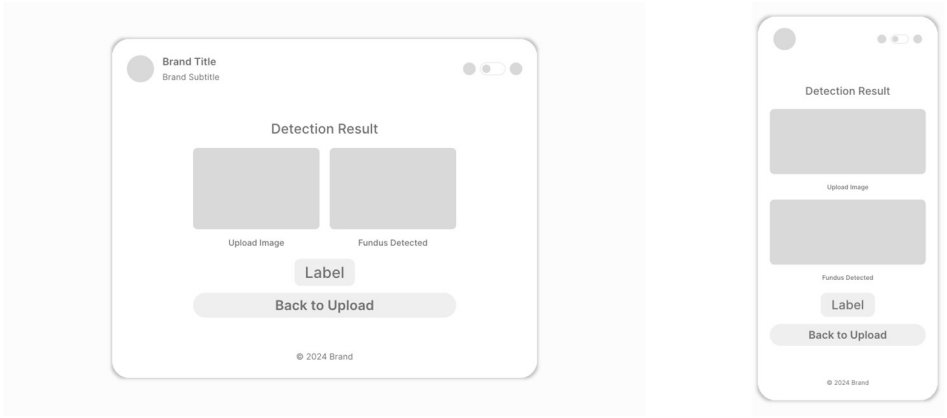
(a) (b)

Gambar 8. *Home page*: (a) *Interface design* untuk tampilan *desktop*; (b) *Interface design* untuk tampilan ponsel.



(a) (b)

Gambar 9. *Help modal page*: (a) *Interface design* untuk tampilan *desktop*; (b) *Interface design* untuk tampilan ponsel.



(a) (b)

Gambar 10. *Result page*: (a) *Interface design* untuk tampilan *desktop*; (b) *Interface design* untuk tampilan ponsel.

Sedangkan, *deployment* dilakukan untuk memfasilitasi penyediaan model *machine learning* secara *online*, sehingga dapat diakses serta digunakan dengan mudah, kapan pun dan dimana pun.

3. Hasil

Dari berbagai bahan dan metode yang telah dilalui, diperoleh hasil penelitian sebagai berikut:

3.1. Ekstraksi Fitur GLCM

Fitur-fitur yang telah diekstraksi disimpan dalam format ".csv", kemudian digunakan sebagai data *training* dan data *testing*. Berikut merupakan pratinjau dari data *training* dan data *testing*:

Tabel 4. Pratinjau ekstraksi fitur GLCM untuk *data training*.

	contrast	***correlation***	***homogeneity***	***energy***	***label***
0	135.762	0.983208	0.714768	0.531294	*Cataract*
1	112.439	0.985209	0.701917	0.520017	*Cataract*
2	72.69121	0.982226	0.743454	0.519726	*Cataract*
.....	.....	.....	.....	.....	.....
477	106.6323	0.973599	0.664447	0.492517	normal
478	114.3193	0.979689	0.657613	0.492828	normal
479	99.5152	0.98717	0.687279	0.50811	normal

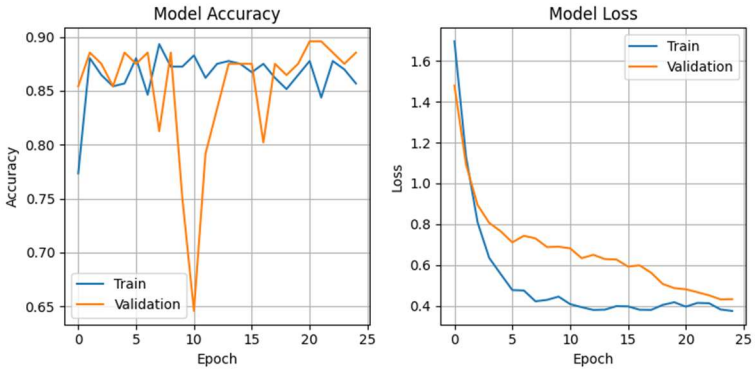
Tabel 5. Pratinjau ekstraksi fitur GLCM untuk *data testing*.

	contrast	***correlation***	***homogeneity***	***energy***	***label***
0	92.51251	0.983575	0.688109	0.519518	*Cataract*
1	109.9114	0.986313	0.711315	0.519581	*Cataract*
2	118.0651	0.98713	0.719718	0.508753	*Cataract*
.....	.....	.....	.....	.....	.....
117	142.6606	0.985105	0.717681	0.519892	normal
118	131.7889	0.985499	0.711401	0.519695	normal
119	136.7772	0.984312	0.67162	0.519151	normal

Berdasarkan Tabel 4 dan Tabel 5, pratinjau data menunjukkan bahwa data *training* terdiri dari 480 ekstraksi fitur GLCM dari citra dengan indeks mulai dari 0 hingga 479, sedangkan data *testing* terdiri dari 120 ekstraksi fitur GLCM dari citra dengan indeks mulai dari 0 hingga 119.

3.2. Implementasi ANN

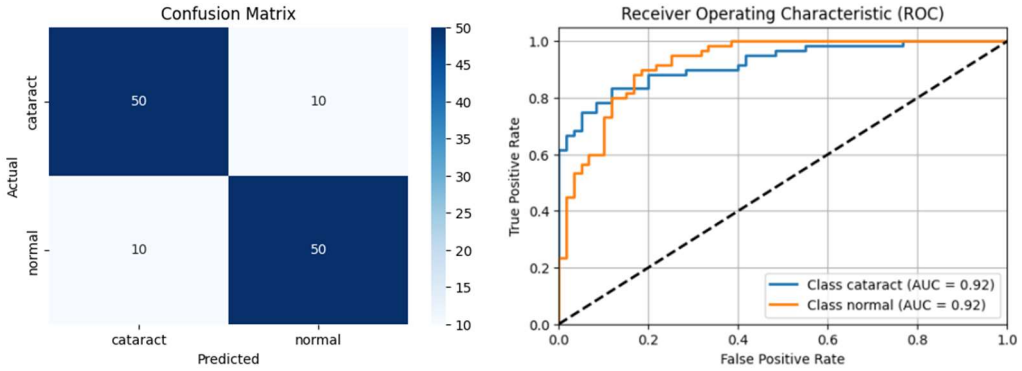
Dari model *Artificial Neural Network* (ANN) yang dibangun diperoleh grafik performa model *accuracy* dan model *loss* selama pelatihan menggunakan data *training*.



Gambar 11. Grafik Model *Accuracy* dan Model *Loss*.

Pada Gambar 11, menampilkan grafik performa model *accuracy* dan model *loss* selama pelatihan, dimana garis biru mewakili pelatihan dan garis oranye mewakili validasi. Grafik model *accuracy* menunjukkan peningkatan akurasi pada data pelatihan dan validasi seiring bertambahnya *epoch*. Akurasi pelatihan stabil sekitar 0.85 hingga 0.90, sementara akurasi validasi mengalami fluktuasi dengan penurunan tajam pada *epoch* ke-10 dari 25 *epoch*, yang mengindikasikan potensi *overfitting*. Grafik model *loss* menunjukkan penurunan *loss* pelatihan secara signifikan, yang menandakan adanya perbaikan dalam prediksi data pelatihan. Namun, penurunan *loss* pada data validasi terjadi lebih lambat dibandingkan dengan data pelatihan, dan kadang-kadang terjadi peningkatan *loss* pada data pelatihan, yang mengindikasikan potensi terjadinya *overfitting*.

Selanjutnya, model diuji menggunakan data *testing*. Pengujian dilakukan dengan tujuan untuk menilai seberapa baik model dapat melakukan prediksi pada data yang belum pernah dilihat sebelumnya menggunakan *confusion matrix*, akurasi, presisi, *recall*, *f1-score*, dan Kurva ROC/AUC.



(a) (b)

Gambar 12. Hasil pengujian: (a) *Confusion matrix*; (b) Kurva ROC.

Berdasarkan metrik pada Gambar 12, diperoleh hasil pengujian sebagai berikut:

Tabel 6. Hasil pengujian.

Akurasi	Presisi	*Recall*	*F1-Score*	AUC
0.83	0.83	0.83	0.83	0.92

Berdasarkan Tabel 6, model mencapai akurasi sebesar 0.83 dengan nilai presisi, *recall*, dan *f1-Score* yang sama yakni 0.83, serta AUC sebesar 0.92. Ini menunjukkan bahwa pada tahap awal pelatihan, model sudah menunjukkan kinerja yang cukup baik dalam hal keseimbangan antara presisi dan *recall*. Meskipun demikian, optimasi melalui *hyperparameter tuning* akan dilakukan untuk meningkatkan kinerja model.

3.3. Evaluasi Kinerja

Evaluasi kerja yang dilakukan pada penelitian ini menggunakan *Hyperparameter Tuning* Pertama, *Hyperparameter* Tuning Kedua, *Hyperparameter Tuning* Ketiga dan *Hyperparameter Tuning* Keempat.

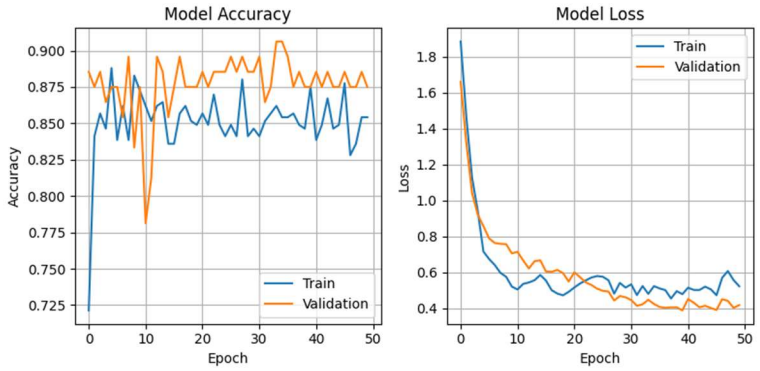
3.3.1. *Hyperparameter* Tuning Pertama

Pada *hyperparameter tuning* pertama, dilakukan penyesuaian terhadap rasio *Dropout* dari 0.1 menjadi 0.5, serta peningkatan jumlah *epoch* dari 25 menjadi 50, seperti pada gambar berikut:

```python
# Fungsi untuk membangun model
def build_model(input_dim, output_dim):
    model = Sequential()
    model.add(Input(shape=(input_dim,)))
    model.add(Dense(128, activation='relu', kernel_regularizer=l2(0.01)))
    model.add(BatchNormalization())
    model.add(Dropout(0.5))
    model.add(Dense(64, activation='relu', kernel_regularizer=l2(0.01)))
    model.add(BatchNormalization())
    model.add(Dropout(0.5))
    model.add(Dense(32, activation='relu', kernel_regularizer=l2(0.01)))
    model.add(BatchNormalization())
    model.add(Dropout(0.5))
    model.add(Dense(output_dim, activation='sigmoid'))
    return model

# Membangun dan mengompilasi model
model = build_model(X_train.shape[1], y_train.shape[1])
optimizer = Adam(learning_rate=0.01)
model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=['accuracy'])

# Menampilkan ringkasan model
model.summary()

epoch=50
```

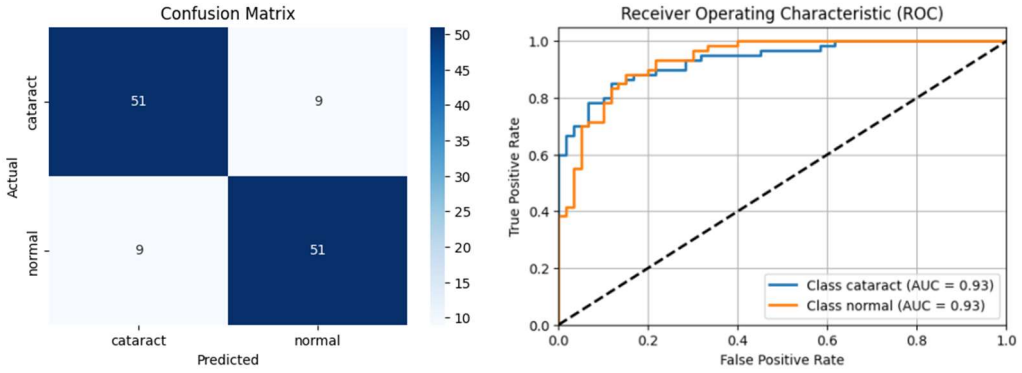
Gambar 13. *Hyperparameter tuning* pertama.

Berdasarkan *hyperparameter tuning* pada Gambar 13, diperoleh grafik performa model *accuracy* dan model *loss* sebagai berikut:



Gambar 14. Grafik model *accuracy* dan model *loss hyperparameter tuning* pertama.

Pada gambar 14, grafik model *accuracy* menunjukkan akurasi pelatihan cenderung stabil dikisaran 0.85 hingga 0.87 setelah beberapa *epoch* awal, meskipun terdapat fluktuasi kecil. Sementara itu, akurasi validasi cenderung lebih tinggi dibandingkan pelatihan, dengan stabil dikisaran 0.87 hingga 0.90 setelah fluktuasi diawal pelatihan. Ini menunjukkan bahwa model memiliki performa yang baik dalam memprediksi data validasi dibandingkan data pelatihan. Grafik model *loss* menunjukkan penurunan signifikan pada nilai *loss* baik untuk data pelatihan maupun validasi selama *epoch* awal. Setelah sekitar 20 *epoch*, *loss* pada data pelatihan menurun lebih lambat dan cenderung berfluktuasi dikisaran 0.4 hingga 0.6. Sementara itu, loss pada data validasi cenderung lebih stabil setelah sekitar 15 *epoch* dan juga berada dikisaran 0.4 hingga 0.6, menunjukkan bahwa model telah mencapai titik konvergensi.



(a) (b)

Gambar 15. Hasil pengujian *hyperparameter tuning* pertama: (a) *Confusion matrix*; (b) Kurva ROC.

Berdasarkan metrik pada Gambar 14, diperoleh hasil pengujian sebagai berikut:

Tabel 7. Hasil pengujian *hyperparameter tuning* pertama.

Akurasi	**Presisi**	***Recall***	***F1-Score***	**AUC**
0.85	0.85	0.85	0.85	0.93

Berdasarkan Tabel 7, model mencapai akurasi sebesar 0.85 dengan nilai presisi, *recall*, dan *f1-score* yang sama yakni 0.85, serta AUC sebesar 0.93. Ini menunjukkan bahwa model sudah menunjukkan kinerja yang cukup baik dibandingkan sebelum dilakukan *hyperparameter tuning*.

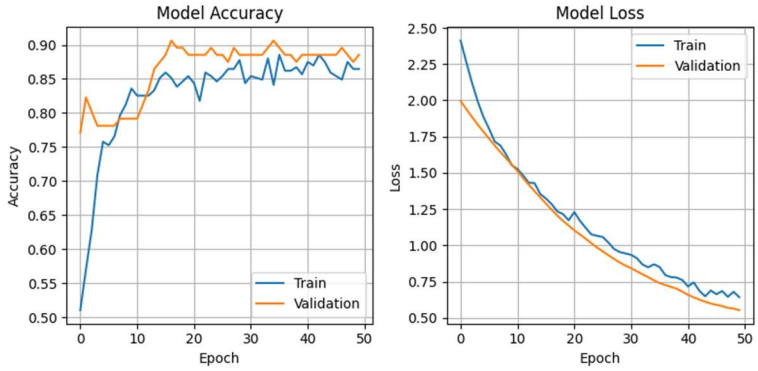
3.3.2. *Hyperparameter* Tuning Kedua

Pada *hyperparameter tuning* kedua, dilakukan penyesuaian terhadap *optimizer* Adam dengan *learning rate* sebesar 0.01 menjadi 0.001, seperti pada gambar berikut:

```
# Membangun dan mengompilasi model
model = build_model(X_train.shape[1], y_train.shape[1])
optimizer = Adam(learning_rate=0.001)
model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=['accuracy'])
```

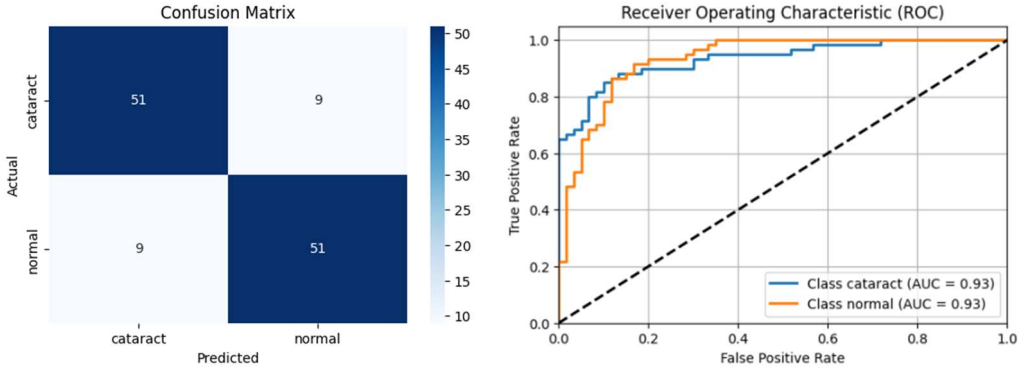
Gambar 16. *Hyperparameter tuning* kedua.

Berdasarkan *hyperparameter tuning* pada Gambar 16, diperoleh grafik performa model *accuracy* dan model *loss* sebagai berikut:



Gambar 17. Grafik model *accuracy* dan model *loss hyperparameter tuning* kedua.

Pada gambar 17, grafik model *accuracy* menunjukkan akurasi model pada data pelatihan mengalami peningkatan yang signifikan dari sekitar 0.55 pada *epoch* pertama hingga mencapai sekitar 0.88 pada akhir pelatihan. Akurasi validasi juga meningkat secara konsisten, mencapai puncak sekitar 0.90 pada *epoch* 15 dan relatif stabil di antara 0.87 hingga 0.90 untuk sisa *epoch*. Grafik model *loss* menunjukkan penurunan *loss* yang signifikan pada data pelatihan dan validasi. Pada awalnya, *loss* pelatihan berada di sekitar 2.5, kemudian menurun secara konsisten hingga mendekati 0.70 di akhir pelatihan. *Loss* validasi juga menurun dengan pola yang serupa, mulai dari sekitar 2.0 dan berakhir mendekati 0.75.



(a) (b)

Gambar 18. Hasil pengujian *hyperparameter tuning* kedua: (a) *Confusion matrix*; (b) Kurva ROC.

Berdasarkan metrik pada Gambar 18, diperoleh hasil pengujian sebagai berikut:

Tabel 8. Hasil pengujian *hyperparameter tuning* kedua.

Akurasi	Presisi	*Recall*	*F1-Score*	AUC
0.88	0.87	0.88	0.87	0.93

Berdasarkan Tabel 8, model berhasil meningkatkan akurasi menjadi 0.88. Nilai presisi mencapai 0.87, sedangkan *recall* dan *f1-score* masing-masing sebesar 0.88 dan 0.87. AUC juga mengalami peningkatan menjadi 0.93, yang menunjukkan bahwa model mampu membedakan kelas dengan lebih baik dibandingkan dengan *hyperparameter tuning* pertama.

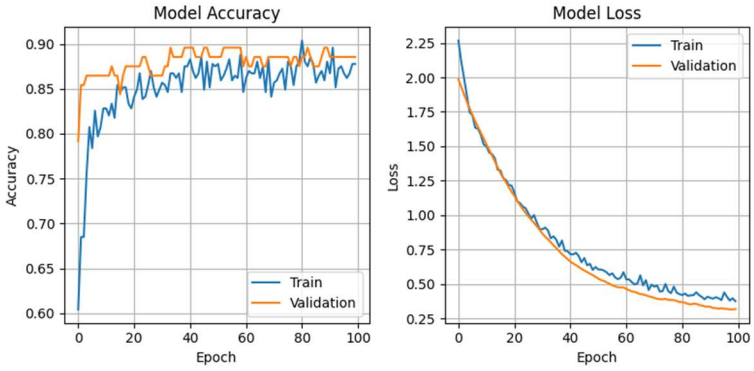
3.3.3. *Hyperparameter* Tuning Ketiga

Pada *hyperparameter tuning* ketiga, dilakukan penyesuaian terhadap jumlah *epoch* sebesar 50 menjadi 100, seperti pada gambar berikut:

```
# Menampilkan ringkasan model
model.summary()

epoch=100
```

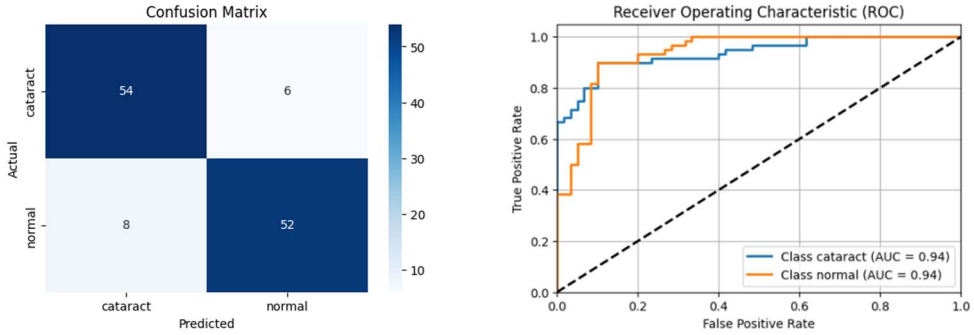
Gambar 19. *Hyperparameter tuning* ketiga.

Berdasarkan *hyperparameter tuning* pada Gambar 19, diperoleh grafik performa model *accuracy* dan model *loss* sebagai berikut:



Gambar 20. Grafik model *accuracy* dan model *loss hyperparameter tuning* ketiga.

Pada gambar 20, grafik model *accuracy* menunjukkan akurasi model pada data pelatihan mengalami peningkatan signifikan sejak awal pelatihan, mulai dari sekitar 0.60 pada *epoch* pertama hingga mencapai sekitar 0.88 di akhir pelatihan. Peningkatan ini cenderung stabil setelah sekitar 50 *epoch*. Sementara itu, akurasi pada data validasi lebih tinggi dari akurasi pelatihan di sebagian besar *epoch*. Akurasi validasi mencapai puncaknya di sekitar 0.90 pada *epoch* 20 dan stabil di antara 0.87 hingga 0.90 hingga akhir pelatihan. Perbedaan kecil antara akurasi pelatihan dan validasi menunjukkan bahwa model tidak mengalami *overfitting*. Grafik model *loss* menunjukkan *loss* pada data pelatihan menurun dengan konsisten dari sekitar 2.25 pada *epoch* pertama hingga mendekati 0.4 di *epoch* terakhir. Ini menunjukkan bahwa model semakin baik dalam memprediksi data pelatihan seiring bertambahnya *epoch*. *Loss* pada data validasi juga mengikuti pola yang serupa, dimulai dari sekitar 2.0 dan terus menurun hingga sekitar 0.3 pada akhir pelatihan. Tren *loss* yang serupa antara data pelatihan dan validasi menunjukkan model memiliki kinerja yang baik dan tidak ada tanda-tanda *overfitting*.



(a) (b)

Gambar 21. Hasil pengujian *hyperparameter tuning* ketiga: (a) *Confusion matrix*; (b) Kurva ROC.

Berdasarkan metrik pada Gambar 21, diperoleh hasil pengujian sebagai berikut:

Tabel 9. Hasil pengujian *hyperparameter tuning* ketiga.

Akurasi	**Presisi**	***Recall***	***F1-Score***	**AUC**
0.88	0.9	0.87	0.88	0.94

Berdasarkan Tabel 9, akurasi model tetap stabil pada 0.88, namun terdapat peningkatan pada presisi yang naik menjadi 0.90. Meskipun *recall* sedikit menurun menjadi 0.87, *f1-score* tetap pada 0.88. AUC juga mengalami peningkatan lebih lanjut menjadi 0.94, yang menunjukkan bahwa model terus berkembang dalam hal membedakan kelas positif dan negatif secara lebih akurat.

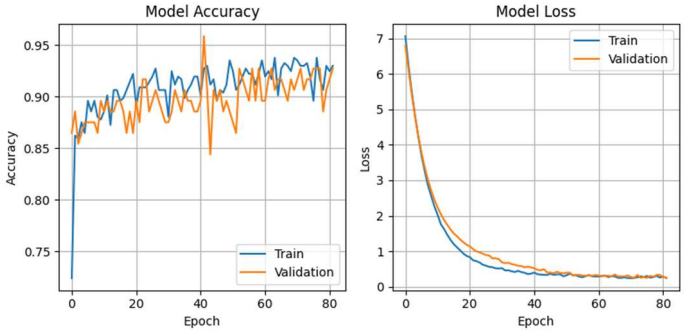
3.3.4. *Hyperparameter* Tuning Keempat

Pada *hyperparameter tuning* keempat, dilakukan penyesuaian terhadap Regularisasi L2 dengan nilai sebesar 0.01 menjadi 0.05 serta penghapusan *Dropout* dari model, seperti pada gambar berikut:

```python
# Fungsi untuk membangun model
def build_model(input_dim, output_dim):
    model = Sequential()
    model.add(Input(shape=(input_dim,)))
    model.add(Dense(128, activation='relu', kernel_regularizer=l2(0.05)))
    model.add(BatchNormalization())
    model.add(Dense(64, activation='relu', kernel_regularizer=l2(0.05)))
    model.add(BatchNormalization())
    model.add(Dense(32, activation='relu', kernel_regularizer=l2(0.05)))
    model.add(BatchNormalization())
    model.add(Dense(output_dim, activation='sigmoid'))
    return model
```

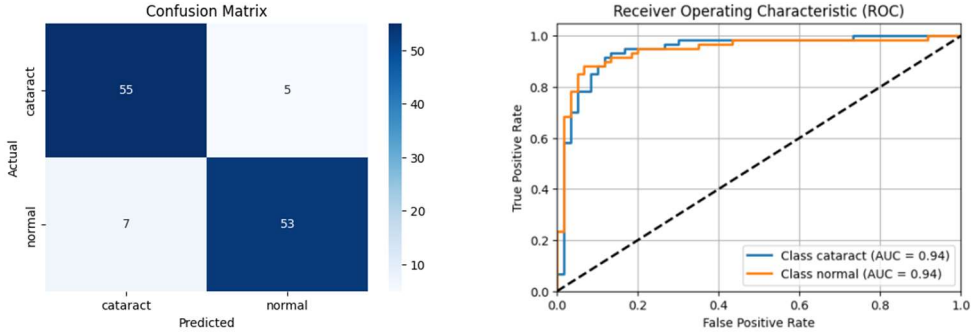
Gambar 22. *Hyperparameter tuning* keempat.

Berdasarkan *hyperparameter tuning* pada Gambar 22, diperoleh grafik performa model *accuracy* dan model *loss* sebagai berikut:



Gambar 23. Grafik model *accuracy* dan model *loss hyperparameter tuning* keempat.

Pada gambar 23, grafik model *accuracy* menunjukkan akurasi model pada data pelatihan dan validasi secara umum meningkat secara konsisten. Model menunjukkan akurasi yang stabil pada kedua *Dataset*, meskipun terdapat fluktuasi kecil dikisaran *epoch* ke-40, akurasi data pelatihan dan validasi pulih dengan cepat. Hal ini mengindikasikan bahwa model memiliki kemampuan generalisasi yang baik dan tidak mengalami *overfitting*. Grafik model *loss* menunjukkan penurunan *loss* yang signifikan pada data pelatihan dan validasi. Pada awal pelatihan, *loss* dimulai dari nilai sangat tinggi, sekitar 7.0, dan menurun dengan cepat selama 20 *epoch* pertama. Setelah itu, penurunan *loss* melambat dan stabil mendekati nilai sekitar 0.25 pada *epoch* ke-70. Penggunaan *Callback early stopping* berperan penting, menghentikan pelatihan yang awalnya direncanakan selama 100 *epoch* secara otomatis pada *epoch* ke-82, karena pada titik ini *loss* pada data validasi sudah tidak menunjukkan penurunan signifikan. Konsistensi antara kurva *loss* pada data pelatihan dan validasi menandakan bahwa model belajar dengan baik dan tidak mengalami masalah *overfitting*, meskipun terdapat beberapa fluktuasi dalam akurasi.



(a) (b)

Gambar 24. Hasil pengujian *hyperparameter tuning* keempat: (a) *Confusion matrix*; (b) Kurva ROC.

Berdasarkan metrik pada Gambar 24, diperoleh hasil pengujian sebagai berikut:

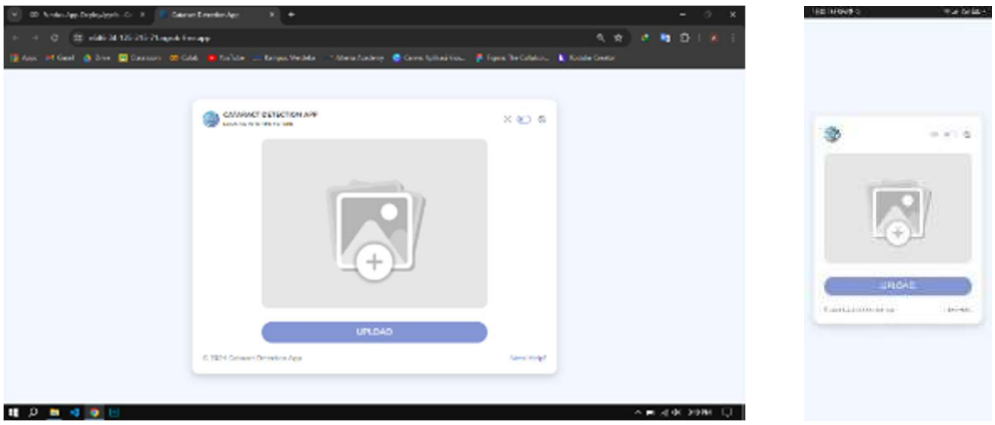
Tabel 10. Hasil pengujian *hyperparameter tuning* keempat.

Akurasi	**Presisi**	***Recall***	***F1-Score***	**AUC**
0.9	0.92	0.89	0.9	0.94

Berdasarkan Tabel 10, model mencapai performa terbaiknya dengan akurasi sebesar 0.90. Nilai presisi mencapai 0.92, sementara *recall* berada pada 0.89 dan *f1-score* pada 0.90, serta AUC tetap pada 0.94. Peningkatan ini menunjukkan bahwa model semakin akurat dalam memprediksi dan mampu memberikan hasil yang lebih andal.

3.4. *Interface Desgin* dan *Deployment*

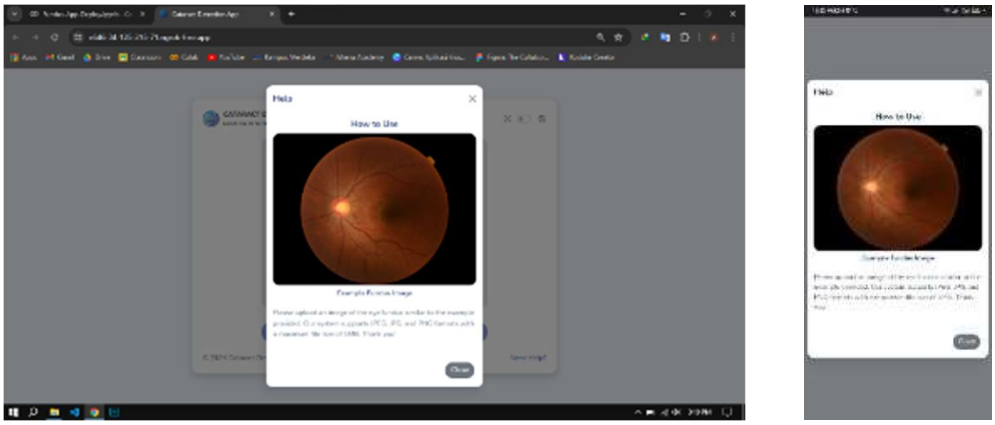
Model *machine learning* yang dibuat selanjutnya diintegrasikan dengan *website* berdasarkan *interface design* pada Gambar 8, Gambar 9, dan Gambar 10 serta di-*deploy* menggunakan Flask dan ngrok untuk memfasilitasi penyediaan model *machine learning* secara *online*, sehingga dapat diakses serta digunakan dengan mudah, kapan pun dan dimana pun.



(a) (b)

Gambar 25. *Home page*: (a) Tampilan *website* pada *desktop*; (b) Tampilan *website* pada ponsel.

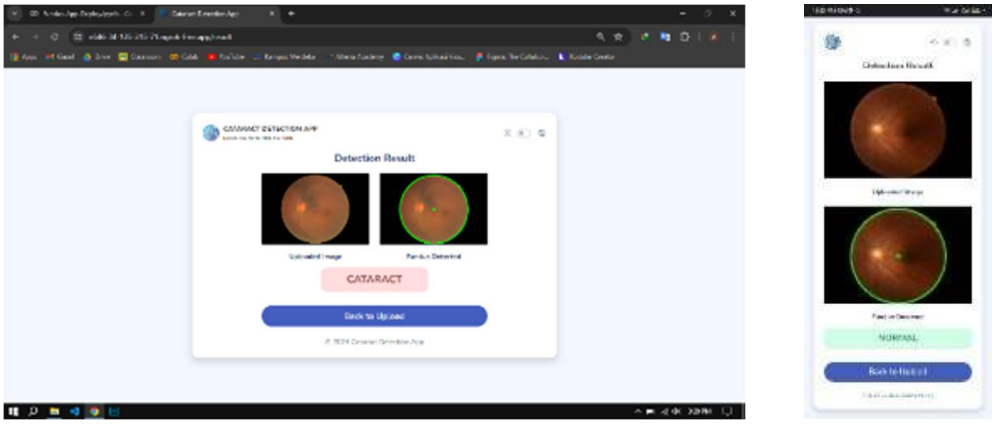
Pada Gambar 25, terdapat fitur pengunggahan citra fundus mata yang dapat diakses dengan mengklik area unggah atau tombol tambah (+). Proses ini memungkinkan pemilihan citra yang ingin diunggah. Setelah memilih, akan ditampilkan pratinjau dari citra yang akan diunggah.



(a) (b)

Gambar 26. *Help modal page*: (a) Tampilan *website* pada *desktop*; (b) Tampilan *website* pada ponsel.

Pada Gambar 26, menampilkan sebuah *modal* atau *popup* yang memberikan panduan tentang cara pengunggahan citra serta ketentuan yang harus dipatuhi dalam proses pengunggahan.



(a) (b)

Gambar 27. *Result page*: (a) Tampilan *website* pada *desktop*; (b) Tampilan *website* pada ponsel.

Pada Gambar 27, menampilkan hasil dari unggahan citra yang telah dilakukan sebelumnya, yang berupa visualisasi citra unggah dan citra yang telah melalui proses deteksi objek, serta menunjukkan hasil deteksi yang mengindikasikan apakah kondisi yang terdeteksi "*Cataract*" atau "Normal".

Pada proses Hyperparameter Tuning Keempat, dilakukan penyesuaian nilai Regularisasi L2, yang semula sebesar 0.01 menjadi 0.05, serta penghapusan lapisan Dropout dari model. Perubahan ini menghasilkan performa terbaik dibandingkan tuning sebelumnya. Grafik performa menunjukkan bahwa akurasi model pada data pelatihan dan validasi meningkat secara konsisten, dengan akurasi yang stabil meskipun terdapat fluktuasi kecil pada epoch tertentu. Selain itu, kurva loss menunjukkan penurunan signifikan yang konsisten pada data pelatihan dan validasi hingga mencapai stabilitas pada epoch ke-82, di mana proses pelatihan dihentikan secara otomatis menggunakan callback early stopping. Berdasarkan evaluasi hasil pengujian, model mencapai akurasi sebesar 90%, dengan presisi sebesar 92%, recall sebesar 89%, dan F1-Score sebesar 90%. Selain itu, nilai AUC sebesar 0.94 menunjukkan bahwa model memiliki kemampuan yang sangat baik dalam membedakan kelas positif dan negatif. Hasil ini membuktikan bahwa hyperparameter tuning keempat memberikan dampak signifikan terhadap peningkatan kinerja model, terutama dalam menghasilkan prediksi yang lebih akurat dan generalisasi yang lebih baik.

4. Diskusi

Secara keseluruhan, model menunjukkan peningkatan yang signifikan dalam kinerja. Peningkatan bertahap pada akurasi, presisi, *recall*, *f1-score*, dan AUC disetiap *hyperparameter tuning* menunjukkan bahwa model semakin mampu mengenali pola data dan menghasilkan prediksi yang lebih akurat. Kinerja yang terus membaik ini juga mengindikasikan bahwa model belajar dengan lebih efektif, sehingga prediksi yang dihasilkan semakin tepat. Berikut merupakan perbandingan hasil dari penelitian sejenis yang telah dilakukan sebelumnya:

Tabel 11. Perbandingan hasil penelitian sejenis yang telah dilakukan sebelumnya.

Judul	*Dataset*	Kelas	Metode	Akurasi
Identification of Cataract Eye Disease using Convolutional Neural Network [4]	260 citra fundus mata	130 kelas "*Cataract*" 130 kelas "Normal"	CNN	92.93%
Klasifikasi Penyakit Katarak pada Mata menggunakan Metode *Convolutional Neural Network (CNN)* Berbasis Web [1]	512 citra mata	256 kelas "*Cataract*" 256 kelas "Normal"	CNN	99.74%
Implementasi Algoritma *Convolutional Neural Network* dalam Mengidentifikasi Dini Penyakit pada Mata Katarak [5]	612 citra mata	306 kelas "*Cataract*" 306 kelas "Normal"	CNN	92.5%
Implementasi *Machine Learning* untuk Mendeteksi Penyakit Katarak menggunakan Kombinasi Ekstraksi Fitur dan *Neural Network* Berdasarkan Citra	600 citra fundus mata	300 kelas "*Cataract*" 300 kelas "Normal"	GLCM & ANN	90%

Berdasarkan Tabel 11, akurasi penelitian ini lebih rendah dibandingkan beberapa penelitian sebelumnya. Meskipun demikian, hasilnya tetap kompetitif dalam konteks penggunaan metode dan *Dataset* yang sebanding.

5. Kesimpulan

Berdasarkan penelitian yang telah dilakukan mulai dari tahap pengumpulan data, prapemrosesan data, segmentasi data, ekstraksi fitur GLCM, implementasi ANN, evaluasi kinerja, hingga *interface design* dan *deployment*, diperoleh model *machine learning* dengan kapabilitas deteksi penyakit katarak yang menunjukkan peningkatan performa seiring dengan optimasi melalui *hyperparameter tuning*. Sebelum dilakukan *hyperparameter tuning*, model mampu mencapai akurasi 83%, kemudian model mampu mencapai akurasi 85% pada *hyperparameter tuning* pertama, meningkat menjadi 88% pada *hyperparameter tuning* kedua, stabil 88% pada *hyperparameter tuning* ketiga, dan mencapai akurasi 90% pada *hyperparameter tuning* keempat. Selain itu, model *machine learning* telah di-*deploy* menggunakan Flask dan ngrok, yang memungkinkan aksesibilitas yang mudah dan fleksibel, sehingga dapat diakses serta digunakan kapan pun dan dimana pun. Diharapkan performa model *machine learning* dapat lebih dioptimalkan untuk mencapai tingkat akurasi dan reliabilitas yang lebih tinggi di masa mendatang,

Referensi

[1] D. Hananta Firdaus, B. Imran, L. Darmawan Bakti, and E. Suryadi, "Klasifikasi Penyakit Katarak Pada Mata Menggunakan Metode *Convolutional Neural Network (CNN)* Berbasis Web," *J. Kecerdasan Buatan dan Teknol. Inf.*, vol. 1, no. 3, pp. 18–26, 2022, doi: https://doi.org/10.69916/jkbti.v1i3.6.

[2] R. Nurlizah, A. eko Minarno, and G. W. Wicaksono, "Klasifikasi Penyakit Katarak Pada Mata Manusia Menggunakan Metode Convolutional Neural Network," *J. Repos.*, vol. 4, no. 4, pp. 491–496, 2024, doi: https://doi.org/10.22219/repositor.v4i4.32285.

[3] A. Roihan, P. A. Sunarya, and A. S. Rafika, "Pemanfaatan Machine Learning dalam Berbagai Bidang: Review paper," *IJCIT (Indonesian J. Comput. Inf. Technol.*, vol. 5, no. 1, pp. 75–82, 2020, doi: https://doi.org/10.31294/ijcit.v5i1.7951.

[4] S. Bhat, S. Mosalagi, T. Bhalerao, P. Katkar, and R. Pitale, "*Cataract* Eye Prediction using Machine Learning," *Int. J. Comput. Appl.*, vol. 176, no. 35, pp. 46–48, 2020, doi: https://doi.org/10.5120/ijca2020920441.

[5] F. Ramadhani, A. Satria, and S. Salamah, "Implementasi Algoritma Convolutional Neural Network dalam Mengidentifikasi Dini Penyakit pada Mata Katarak," *sudo J. Tek. Inform.*, vol. 2, no. 4, pp. 167–175, 2023, doi: https://doi.org/10.56211/sudo.v2i4.408.

[6] K. C. Karamihan, I. D. F. Agustino, R. B. B. Bionesta, F. C. Tuason, S. V. E. Arellano, and P. A. M. Esguerra, "SBC-Based *Cataract* Detection System Using Deep Convolutional Neural Network With Transfer Learning Algorithm," *Int. J. Recent Technol. Eng.*, 2019, doi: https://doi.org/10.35940/ijrte.b3368.078219.

[7] O. Zisimopoulos *et al.*, "DeepPhase: Surgical Phase Recognition in *CATARACT*S Videos," 2018, https://link.springer.com/chapter/10.1007/978-3-030-00937-3_31.

[8] W. M. Diyana Zaki, H. A. Mutalib, L. A. Ramlan, A. Hussain, and A. Mustapha, "Towards a Connected Mobile *Cataract* Screening System: A Future Approach," *J. Imaging*, 2022, doi: https://doi.org/10.3390/jimaging8020041.

[9] X. Liu *et al.*, "Localization and Diagnosis Framework for Pediatric *Cataract*s Based on Slit-Lamp Images Using Deep Features of a Convolutional Neural Network," *PLoS One*, 2017, doi: https://doi.org/10.1371/journal.pone.0168606.

[10] X. Wu *et al.*, "Universal Artificial Intelligence Platform for Collaborative Management of *Cataract*s," *Br. J. Ophthalmol.*, 2019, doi: https://doi.org/10.1136/bjophthalmol-2019-314729.

[11] Y. Elloumi, "*Cataract* Grading Method Based on Deep Convolutional Neural Networks and Stacking Ensemble Learning," *Int. J. Imaging Syst. Technol.*, 2022, doi: https://doi.org/10.1002/ima.22722.

[12] R. Hidayat, S. Agustiani, S. K. Wildah, A. Mustopa, and R. A. Safitri, "Penerapan Metode Pembelajaran Menggunakan Ekstraksi Fitur dan Algoritma Klasifikasi untuk Identifikasi Pengenalan Iris," *J. Tek. Komput. AMIK BSI*, vol. 8, no. 2, pp. 174–180, 2022, doi: https://doi.org/10.31294/jtk.v4i2.

[13] I. I. Ridho, G. Mahalisa, D. R. Sari, and I. Fikri, "METODE NEURAL NETWORK UNTUK PENENTUAN AKURASI PREDIKSI HARGA RUMAH," *Technologia*, vol. 1, no. 1, pp. 2020–2022, 2022, doi: http://dx.doi.org/10.31602/tji.v13i1.6252.

[14] F. Alghifari and D. Juardi, "Penerapan Data Mining Pada Penjualan Makanan Dan Minuman Menggunakan Metode Algoritma Naïve Bayes," *J. Ilm. Inform.*, vol. 9, no. 02, pp. 75–81, 2021, doi: https://doi.org/10.33884/jif.v9i02.3755.

[15] R. Z. Fadillah, A. Irawan, M. Susanty, and I. Artikel, "Data Augmentasi Untuk Mengatasi Keterbatasan Data Pada Model Penerjemah Bahasa Isyarat Indonesia (BISINDO)," *J. Inform.*, vol. 8, no. 2, pp. 208–214, 2021, [Online]. Available: https://ejournal.bsi.ac.id/ejurnal/index.php/ji/article/view/10768

[16] D. Juniati and A. E. Suwanda, "Klasifikasi Penyakit Mata Berdasarkan Citra Fundus Retina Menggunakan Dimensi Fraktal Box Counting Dan Fuzzy K-Means," *Prox. J. Penelit. Mat. dan Pendidik. Mat.*, vol. 5, no. 1, pp. 10–18, 2022, doi: https://doi.org/10.30605/proximal.v5i1.1623.

[17] C. U. P. Negara and K. Firdausy, "Pengolahan Citra menggunakan Metode Otsu dan Hough Circle Transform untuk Prototipe Alat Sortir Buah Apel," *Semin. Nas. Apl. Teknol. Inf.*, pp. 1–7, 2019, https://journal.uii.ac.id/Snati/article/view/13393.

[18] B. Imran, L. D. Samsumar, and A. Subki, "Combination of gray level co-occurrence matrix and artificial neural networks for classification of COVID-19 based on chest X-ray images," *Int. J. Artif. Intell.*, vol. 13, no. 2, pp. 1625–1631, 2024, doi: https://doi.org/10.11591/ijai.v13.i2.pp1625-1631.

[19] R. Kurniawan, R. H. Nasution, S. Marwah, and B. Nasution, "Penerapan Metode Gray Level Co-Occurrence Matrix Dan Knn Untuk Mendeteksi Tingkat Kematangan Buah Mengkudu," *J. Sci. Soc. Res.*, vol. 4307, no. 2, pp. 765–772, 2024, [Online]. Available: https://jurnal.goretanpena.com/index.php/JSSR/article/view/1836

[20] A. F. Suahati, A. A. Nurrahman, and O. Rukmana, "Penggunaan Jaringan Syaraf Tiruan – Backpropagation dalam Memprediksi Jumlah Mahasiswa Baru," *J. Media Tek. dan Sist. Ind.*, vol. 6, no. 1, p. 21, 2022, doi: https://doi.org/10.35194/jmtsi.v6i1.1589.

[21] R. W. Hutabri, R. Magdalena, and Y. N. Fu'adah, "PERANCANGAN SISTEM DETEKSI KATARAK MENGGUNAKAN METODE PRINCIPAL COMPONENT ANALYSIS (PCA) DAN KNEAREST NEIGHBOR (K-NN) Analisis Performansi Sistem Deteksi Katarak," *Semin. Nas. Inov. dan Apl. Teknol. di Ind. 2018*, pp. 321–327, 2018, [Online]. Available: https://ejournal.itn.ac.id/index.php/seniati/article/view/1390