MODEL CNN BERBASIS EFFICIENTNET-LITE UNTUK DETEKSI RETINOPATI DIABETIK MENGGUNAKAN TENSORFLOW LITE

Muhammad, Indah Susilawati
Informatika, Universitas Mercu Buana Yogyakarta
Jl. Jembatan Merah No. 84C Gejayan - Yogyakarta
211110011@student.mercubuana-yogya.ac.id

ABSTRAK

Retinopati diabetik (RD) adalah komplikasi diabetes yang berpotensi menyebabkan kebutaan. Deteksi dini sangat penting, namun interpretasi citra fundus seringkali hanya dapat dilakukan oleh tenaga medis. Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi model *Lightweight Convolutional Neural Network* (CNN) berbasis EfficientNet-Lite0 untuk deteksi RD dari citra fundus, yang dapat diimplementasikan pada perangkat seluler. Dataset yang digunakan terdiri dari 9.850 citra fundus gabungan dari Eyepacs, APTOS, dan Messidor yang telah melalui tahap *preprocessing* seperti validasi, normalisasi, dan *resizing* menjadi 384x384 piksel. Data dibagi menjadi 80% untuk pelatihan, 10% validasi, dan 10% pengujian. Model EfficientNet-Lite0 digunakan sebagai *feature extractor* yang tidak dilatih ulang, diikuti oleh *dense layers* dan *output layer* sigmoid untuk klasifikasi biner (Normal/DR). Hasil evaluasi menunjukkan model mencapai akurasi 79,4%, presisi 83,5%, *recall* 73,3%, dan F1-skor 78,0%.

Kata kunci : *Retinopati Diabetik, EfficientNet-Lite0, TensorFlow Lite, Deteksi Citra Medis, Perangkat Seluler.*

1. PENDAHULUAN

Retinopati diabetik merupakan salah satu komplikasi serius akibat diabetes yang dapat menyebabkan kebutaan jika tidak terdeteksi dan ditangani dengan cepat [1]. Retinopati diabetik adalah salah satu penyebab kebutaan di dunia, terutama pada pasien dengan diabetes melitus yang tidak terkontrol dengan baik, dengan angka mencapai 4,8% dari 39 juta kasus kebutaan yang terjadi [2].

Saat ini, hasil pemeriksaan retinopati diabetik, seperti citra fundus mata yang diambil melalui fundus fotografi atau *Optical Coherence Tomography* (OCT), sering kali hanya dapat diinterpretasikan oleh tenaga medis. Pasien sendiri sering kali tidak memiliki akses untuk memahami atau melihat kembali hasil pemeriksaannya dengan mudah. Oleh karena itu, diperlukan suatu pendekatan yang memungkinkan pasien untuk dapat mengakses dan melihat kembali hasil pemeriksaan mereka dengan lebih mudah.

EfficientNet-Lite merupakan salah satu model Convolutional Neural Network (CNN) yang dirancang untuk perangkat dengan sumber daya terbatas, seperti ponsel atau perangkat *edge computing* [3]. Dengan memanfaatkan model ini, citra fundus mata yang telah diperiksa oleh dokter dapat diolah kembali sehingga pasien dapat melihat hasilnya dalam format yang lebih mudah diakses. Melalui implementasi berbasis *TensorFlow Lite*, model ini dapat dioptimalkan untuk berjalan pada perangkat dengan daya komputasi rendah, memungkinkan pasien untuk memantau kondisi mata mereka secara lebih mandiri.

Beberapa penelitian sebelumnya menunjukkan keunggulan CNN untuk mengekstraksi dan mengenali pola pada citra fundus. Salah satu arsitektur yang menonjol adalah MobileNetV3, yang juga didesain untuk perangkat kecil berhasil mengklasifikasi penyakit mata seperti katarak, retinopati diabetik, dan glaukoma melalui citra fundus dengan data sebanyak 3941 gambar dari 4000 gambar dengan nilai akurasi yang didapatkan mencapai 90% [4]. Pada kasus retinopati diabetik, metode Support Vector Machine (SVM) mendapatkan akurasi sebesar 95,93% dari 600 dataset, pada kasus serupa EfficientNet berhasil melakukan klasifikasi dengan akurasi sebesar 82.096% dari total 3362 citra fundus. Namun, belum ada penelitian yang secara khusus mengevaluasi performa model EfficientNet-Lite untuk klasifikasi *diabetic retinopathy* secara langsung di perangkat seluler terutama android, terutama dari segi akurasi, kecepatan inferensi, dan efisiensi penggunaan sumber daya.

2. TINJAUAN PUSTAKA

2.1. Retinopati Diabetik

Retinopati diabetik adalah salah satu komplikasi mikrovaskular akibat diabetes melitus yang menyerang retina mata dan dapat menyebabkan kebutaan jika tidak ditangani dengan baik. Penyakit ini terjadi akibat kerusakan pada pembuluh darah retina akibat kadar gula darah yang tinggi dalam waktu lama [5].

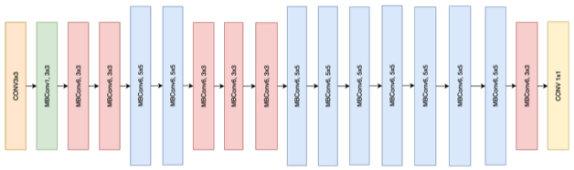
Secara klinis, retinopati diabetik terbagi menjadi dua tingkat keparahan. Retinopati Diabetik Non-Proliferatif (RDNP) merupakan tahap awal yang ditandai dengan mikroaneurisma, perdarahan kecil, dan eksudat pada retina. Gejalanya sering tidak disadari. Tahap lanjut disebut Retinopati Diabetik Proliferatif (RDP), ditandai dengan pertumbuhan pembuluh darah baru yang rapuh di retina, yang dapat menyebabkan perdarahan vitreous, ablasi retina, dan risiko kebutaan permanen [1].

2.2. Citra Fundus Mata dan Pemeriksaan Retinopati Diabetik

Citra fundus mata diperoleh dari teknik pencitraan seperti fundus fotografi dan *Optical Coherence Tomography* (OCT). Menggunakan foto fundus portabel, model ini memiliki sensitivitas antara 64–94% dan spesifisitas 71–90% dalam mendeteksi retinopati diabetik yang berisiko mengancam penglihatan [6]. *Optical Coherence Tomography* (OCT) memberikan pencitraan lapisan retina dengan resolusi tinggi dan sering digunakan untuk mendiagnosis edema makula diabetic [7].

2.3. EfficientNet-Lite dalam Deteksi Citra Medis

EfficientNet-Lite adalah varian ringan dari arsitektur EfficientNet yang dirancang untuk perangkat dengan sumber daya terbatas seperti ponsel dan *edge computing* [3]. Model ini menggunakan teknik *compound scaling* untuk meningkatkan efisiensi tanpa mengorbankan akurasi. Berikut adalah arsitektur dari EfficientNet-Lite yang ditampilkan pada Gambar 1.



Gambar 1. Arsitektur EfficientNet-Lite0

Proses *classification* dalam arsitektur ini bertujuan untuk mengidentifikasi neuron-neuron hasil ekstraksi yang telah diperoleh melalui proses *feature learning*. Tahapan ini biasanya mencakup proses *flattening*, di mana data multidimensi diubah menjadi vektor satu dimensi, yang kemudian diteruskan ke *fully connected layer* [8]. Lapisan ini bertanggung jawab dalam menginterpretasikan fitur-fitur yang telah dipelajari untuk menghasilkan prediksi akhir atau klasifikasi dari citra medis yang dianalisis.

2.4. Penelitian Terdahulu

Beberapa penelitian terdahulu telah dilakukan terkait deteksi retinopati diabetik menggunakan citra fundus dan model *deep learning*. Penelitian berjudul *"Klasifikasi Diabetic Retinopathy Berbasis Pengolahan Citra Fundus dan Deep Learning"* [9] mengkaji penggunaan arsitektur EfficientNet untuk klasifikasi retinopati diabetik, dan menunjukkan hasil akurasi sebesar 82%, dengan nilai presisi 67%, recall 63%, serta F1-score sebesar 64%. Sementara itu, penelitian *"Deep Learning Untuk Klasifikasi Penyakit Retinopati Diabetik Menggunakan Arsitektur AlexNet dan Generative Adversarial Network"* [10] menemukan bahwa penggunaan AlexNet menghasilkan akurasi yang cukup rendah, yakni 25%, dengan nilai precision dan F1-score masing-masing sebesar 24%.

Penelitian lain yang berjudul "*Implementasi Seblock Pada Klasifikasi Citra Penyakit Mata Manusia Dengan Arsitektur Mobilenetv3-Small"[4]* menunjukkan performa yang cukup baik, di mana arsitektur MobileNet V3-Small mampu mencapai akurasi hingga 90% untuk klasifikasi berbagai penyakit mata seperti katarak, glaukoma, penyakit retina, dan lainnya. Selanjutnya, penelitian dengan judul *"Penerapan Convolutional Neural Network untuk Klasifikasi Tingkat Keparahan Retinopati Diabetik pada Penderita Diabetes Melitus" [11]* mencatat akurasi tertinggi mencapai 91% dalam klasifikasi tingkat keparahan retinopati diabetik. Terakhir, penelitian berjudul *"Klasifikasi Penyakit Diabetic Retinopathy Menggunakan Multilayer Perceptron"* [12] menunjukkan hasil akurasi sebesar 71%, dengan nilai precision 72% dan recall 71% berdasarkan data sampel sebanyak 1151 gambar.

Selain itu, penelitian berjudul *"Klasifikasi Buah dan Sayuran Segar atau Busuk Menggunakan Convolutional Neural Network"* [8] menunjukkan bahwa metode CNN juga dapat digunakan secara efektif untuk klasifikasi citra di domain lain, seperti pertanian. Dalam penelitian ini, CNN digunakan untuk mengklasifikasikan kondisi buah dan sayuran menjadi segar atau busuk, dengan hasil akurasi yang tinggi. Meskipun objek citra berbeda dengan citra fundus mata, pendekatan deep learning yang digunakan menunjukkan potensi generalisasi metode CNN dalam berbagai jenis klasifikasi citra. Hal ini menunjukkan bahwa CNN memiliki fleksibilitas dan kapabilitas yang tinggi dalam menyelesaikan permasalahan klasifikasi visual.

3. METODE PENELITIAN

Penelitian ini terdiri dari lima tahap yang sistematis, dimulai dari pengumpulan data, preprocessing data, pelatihan model, evaluasi model, implementasi model pada perangkat *mobile*.

3.1. Alat dan Bahan

Penelitian ini menggunakan perangkat keras berupa laptop Apple MacBook dengan chip M1 dan SSD eksternal sebagai penyimpanan dataset. Selain itu, sebuah smartphone Android digunakan untuk proses pengujian model hasil konversi dalam format *TensorFlow Lite*. Di sisi perangkat lunak, digunakan bahasa pemrograman *Python* dengan library *TensorFlow 2.x*, *TensorFlow Lite*, *TensorFlow Hub*, *Scikit-learn*, *Matplotlib*, dan *Seaborn*. Pelatihan dan eksperimen model dilakukan secara local.

3.2. Pengumpulan Data

Penelitian ini menggunakan subset dari dataset gabungan Eyepacs, APTOS, dan Messidor Diabetic Retinopathy yang tersedia di platform Kaggle, dengan total keseluruhan sekitar 93.000 citra retina. Perkiraan jumlah data pada masing-masing dataset adalah sekitar 88.702 citra dari Eyepacs, 3.662 dari APTOS, dan antara 1.200 hingga 1.748 dari Messidor. Dari total tersebut, sebanyak 9.850 citra dipilih secara acak dari ketiga sumber untuk digunakan sebagai data pelatihan dan pengujian, jumlah ini dipilih karena

keterbatasan sumber daya komputasi pelatihan. Pemilihan acak ini bertujuan untuk menjaga keragaman data dan meminimalkan bias, sehingga model yang dibangun dapat melakukan generalisasi dengan lebih baik terhadap berbagai tingkat keparahan retinopati diabetik.

Dataset ini berisi ribuan citra fundus mata manusia yang telah dianotasi oleh tenaga medis profesional berdasarkan tingkat keparahan retinopati diabetik. Total citra yang digunakan dalam penelitian ini adalah sebanyak 9.850 gambar yang mencakup berbagai kondisi, namun dalam penelitian ini diklasifikasikan secara biner, yaitu citra dengan kondisi normal dan citra dengan indikasi DR.

3.3. Prepocessing Data

Sebelum data digunakan dalam proses pelatihan, dilakukan tahap preprocessing untuk memastikan kualitas citra yang optimal dan konsistensi format data, preprocessing data meliputi:

a. Validasi dan Pembersihan Gambar
   Gambar yang tidak valid, corrupt, atau tidak sesuai dengan format dihapus menggunakan pengecekan dengan modul *PIL.Image.verify*. Proses ini penting untuk memastikan bahwa hanya data yang berkualitas yang digunakan dalam pelatihan model, seperti disarankan oleh Raji S [9] dalam studi mereka tentang teknik pembersihan dan augmentasi data untuk meningkatkan akurasi model pembelajaran mesin.
b. Normalisasi
   Nilai piksel dikonversi ke skala 0-1 untuk meningkatkan stabilitas pelatihan. Teknik normalisasi ini telah terbukti efektif dalam berbagai studi, termasuk oleh Widyaya [13] yang menunjukkan peningkatan akurasi klasifikasi setelah penerapan normalisasi pada citra fundus retina.
c. Resize
   Semua citra diubah ukurannya menjadi 384 × 384 piksel agar sesuai dengan input yang dibutuhkan oleh *EfficientNet-Lite0*. Penyesuaian ukuran ini penting untuk memastikan konsistensi input dan efisiensi komputasi, seperti yang diterapkan dalam penelitian oleh Minarno [14] yang menggunakan arsitektur *InceptionV3* untuk klasifikasi *diabetic retinopathy*.
d. Pembagian Data
   Data dibagi menjadi tiga bagian, yaitu 80% untuk pelatihan, 10% untuk validasi, dan 10% untuk pengujian. Pembagian ini bertujuan untuk mengevaluasi kinerja model secara menyeluruh dan mencegah *overfitting*. Pendekatan ini sejalan dengan praktik umum dalam pelatihan model pembelajaran mesin, sebagaimana dijelaskan oleh Widyaya [13] dalam studi mereka tentang pengaruh preprocessing terhadap klasifikasi *diabetic retinopathy*.

3.4. Arsitektur Model

Model yang digunakan adalah EfficientNet-Lite0 dari TensorFlow Hub, yang merupakan bagian dari keluarga Lightweight CNN dan dioptimalkan untuk perangkat mobile. Arsitektur model terdiri dari beberapa komponen utama, yaitu:

a. Input Layer (384x384 RGB)
   Untuk menyesuaikan ukuran citra fundus agar sesuai dengan format model.
b. EfficientNet-Lite0
   Sebagai feature extractor yang tidak dilatih ulang (non-trainable), sehingga efisien secara komputasi.
c. Dense Layer
   Dua layer berturut-turut (64 dan 32 neuron) dengan regularisasi seperti batch normalization, Leaky ReLU, dan dropout.
d. Output Layer
   Satu neuron dengan aktivasi sigmoid untuk klasifikasi biner (Normal/DR).

Model dikembangkan menggunakan *Keras Functional API* dan didistribusikan dengan *tf.distribute.Strategy* untuk efisiensi perangkat keras.

3.5. Implementasi dan Pelatihan Model

Pelatihan model dilakukan menggunakan framework *TensorFlow* dan *TensorFlow Lite* untuk memungkinkan optimasi model pada perangkat mobile. Parameter utama dalam pelatihan model meliputi:

Tabel 1. Tabel Parameter Model

Parameter	Nilai
Optimizer	Adam
Loss Function	Binar Focal Crossentropy
Batch Size	32
Jumlah Epochs	40
Learning Rate	0.0001
Callback	Early Stopping, ReduceLROnPlateau

3.6. Evaluasi Model

Setelah model selesai dilatih, evaluasi dilakukan untuk menilai kinerjanya dalam mengklasifikasikan tingkat keparahan *diabetic retinopathy* (DR). Evaluasi dilakukan menggunakan metrik seperti *confusion matrix, accuracy, precision, recall, dan f1-score* [15]. Metrik-metrik ini memberikan gambaran menyeluruh terhadap kemampuan model dalam mendeteksi DR, khususnya pada perangkat dengan sumber daya terbatas.

Dalam evaluasi model klasifikasi, terutama pada klasifikasi biner (dua kelas), digunakan beberapa istilah penting untuk mengukur kinerja model berdasarkan *confusion matrix*, yaitu:

a. *True Positive* (TP)
   Jumlah kasus *positif* yang diprediksi benar oleh model. Contoh model memprediksi bahwa pasien sakit (positif), dan pasien tersebut memang benar sakit.
b. *True Negative* (TN)
   Jumlah kasus *negatif* yang diprediksi benar oleh model. Contoh model memprediksi bahwa pasien

sehat (negatif), dan pasien tersebut memang benar sehat.

c. *False Positive* (FP)
Jumlah kasus negatif yang diprediksi salah sebagai positif oleh model. Contoh model memprediksi pasien sakit, padahal sebenarnya pasien sehat.

d. *False Negative* (FN)
Jumlah kasus positif yang diprediksi salah sebagai negatif oleh model. Contoh model memprediksi pasien sehat, padahal sebenarnya pasien sakit.

Berdasarkan nilai-nilai ini, dapat dihitung berbagai metrik evaluasi:

Accuracy merupakan metode evaluasi yang digunakan untuk menilai seberapa tepat model dalam memprediksi nilai yang benar dibandingkan dengan yang salah. Perhitungan *accuracy* secara matematis ditunjukkan pada Persamaan (1).

$$Accuracy = \frac{TP + TN}{TP+TN+FP+FN} \quad (1)$$

Precision merupakan metode evaluasi yang digunakan untuk mengukur sejauh mana model tepat dalam memprediksi data yang termasuk ke dalam suatu kelas tertentu. Perhitungan *precision* secara matematis ditunjukkan pada Persamaan (2).

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

Recall merupakan metode evaluasi yang digunakan untuk menilai sejauh mana model mampu mendeteksi data positif secara benar sesuai dengan kondisi sebenarnya. Perhitungan *recall* secara matematis ditunjukkan pada Persamaan (3).

$$Recal = \frac{TP}{TP+FN} \quad (3)$$

F1-Score merupakan metode evaluasi yang digunakan untuk mengukur kinerja model klasifikasi dengan menggabungkan rata-rata harmonis antara *precision* dan *recall*. Perhitungan *recall* secara matematis ditunjukkan pada Persamaan (4).

$$F1 = 2 \text{ x } \frac{Precision\ x\ Recall}{Precision\ x\ Recal} \quad (4)$$

Evaluasi ini penting untuk memastikan bahwa model tidak hanya akurat secara keseluruhan, tetapi juga sensitif terhadap kasus positif yang penting dalam konteks medis seperti DR.

3.7. Implementasi pada Perangkat Seluler

Model hasil pelatihan diekspor ke format *TensorFlow Lite (.tflite)* menggunakan *TFLiteConverter*. Proses optimasi dilakukan dengan menggunakan supported ops *TFLITE_BUILTINS*, optimization *DEFAULT*, dan tipe data *Float32*. Konfigurasi ini memastikan model dapat dikonversi ke format *TensorFlow Lite* dengan kompatibilitas optimal untuk pengembangan pada perangkat mobile atau *embedded systems*.

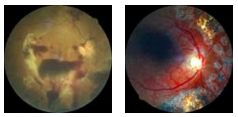
Hasil model TFLite kemudian diuji pada perangkat smartphone Android, untuk mengukur waktu *inferensi* (*latency*) dan kesesuaian ukuran model dengan penyimpanan perangkat. Dengan metodologi ini, diharapkan model dapat mendeteksi *diabetic retinopathy* secara akurat dan efisien pada berbagai perangkat, terutama di lingkungan dengan sumber daya terbatas.

4. HASIL DAN PEMBAHASAN

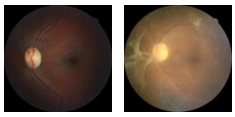
4.1. Alat dan Bahan

Dari 10.000 citra fundus yang dikumpulkan, 150 citra dinyatakan rusak atau tidak sesuai format. Setelah proses validasi, tersisa 9.850 citra yang kemudian dinormalisasi ke rentang [0,1] dan diubah ukurannya menjadi 384×384 piksel. Data dibagi menjadi 80% pelatihan, 10% validasi, 10% pengujian.

Pembagian ini mengikuti praktik umum untuk menjaga generalisasi model dan mencegah overfitting. Untuk memberikan gambaran visual mengenai data yang digunakan, Gambar 2 menunjukkan contoh citra fundus dari pasien yang terdeteksi mengalami retinopati diabetik (kelas positif), sedangkan Gambar 3 menampilkan citra fundus dari pasien yang tidak mengalami kelainan (kelas negatif).



Gambar 2. Gambar citra fundus positif retinopati diabetik



Gambar 3. Gambar citra fundus negatif retinopati diabetik

4.2. Pembagian Dataset

Setelah proses validasi dan preprocessing selesai, total 9.850 citra fundus yang telah bersih dan seragam dibagi menjadi tiga subset data.

Sebanyak 7.880 citra (80% dari 9.850) dialokasikan untuk pelatihan model. Data ini akan digunakan oleh model untuk mempelajari fitur-fitur dan pola-pola yang relevan untuk klasifikasi.

Sebanyak 985 citra (10% dari 9.850) digunakan sebagai data validasi. Set data ini berfungsi untuk memantau kinerja model selama proses pelatihan dan membantu dalam penyesuaian *hyperparameter* serta mendeteksi *overfitting* secara dini.

Sebanyak 985 citra (10% dari 9.850) disisihkan untuk pengujian akhir model. Data pengujian ini adalah data yang baru bagi model dan tidak pernah digunakan selama pelatihan atau validasi. Tujuannya adalah untuk memberikan evaluasi objektif dan tidak bias terhadap kemampuan generalisasi model pada data yang belum pernah dilihat sebelumnya.

Pembagian ini mengikuti praktik umum dalam pembelajaran mesin untuk memastikan bahwa model yang dihasilkan tidak hanya berkinerja baik pada data

pelatihan (mencegah *overfitting*), tetapi juga memiliki kemampuan generalisasi yang kuat pada data baru yang belum dikenal.

4.3. Penerapan Arsitektur Model

Model yang digunakan dalam penelitian ini adalah EfficientNet-Lite0, sebuah arsitektur *Convolutional Neural Network (CNN)* yang dioptimalkan untuk efisiensi komputasi, menjadikannya pilihan yang ideal untuk aplikasi pada perangkat *mobile* atau lingkungan dengan sumber daya terbatas. Pemilihan EfficientNet-Lite0 didasarkan pada kemampuannya menyeimbangkan akurasi klasifikasi dengan kecepatan inferensi yang tinggi.

Model ini dirancang untuk klasifikasi biner (Normal/*Diabetic Retinopathy*) dan terdiri dari beberapa komponen utama:

a. Input Layer
   Model menerima input berupa citra fundus retina dengan dimensi 384x384 piksel dan 3 kanal warna (RGB). Ukuran ini telah distandarisasi melalui proses *preprocessing* untuk memenuhi spesifikasi input *EfficientNet-Lite0*.
b. Ekstraksi Fitur
   Sebagai tulang punggung arsitektur, digunakan EfficientNet-Lite0 yang diimpor dari TensorFlow Hub. Bagian ini menggunakan EfficientNet-Lite0 dari TensorFlow Hub sebagai pondasi utama arsitektur. Fungsinya adalah untuk mengambil ciri-ciri penting dari gambar fundus retina yang dimasukkan.
c. Head Klasifikasi Kustom (Lapisan Dense)
   Setelah fitur diekstraksi oleh EfficientNet-Lite0, outputnya disalurkan ke serangkaian lapisan *dense* (juga dikenal sebagai *fully connected layers*) yang dirancang khusus untuk tugas klasifikasi biner ini. Bagian ini terdiri dari dua lapisan *dense* berturut-turut.
   Lapisan *dense* pertama memiliki 64 neuron. Setelah lapisan *dense*, diterapkan batch normalization untuk menstabilkan dan mempercepat pelatihan. Ini diikuti oleh fungsi aktivasi *leaky relu* (α=0.2) yang membantu mengatasi masalah *dying relu* dengan memungkinkan gradien kecil untuk input negatif. Terakhir, dropout dengan laju 0.4 diterapkan untuk mengurangi *overfitting* dengan secara acak menonaktifkan sebagian neuron selama pelatihan. Regularisasi L2 dengan nilai 1e−4 juga diterapkan pada *kernel* lapisan ini untuk mencegah *overfitting* lebih lanjut.
   Lapisan *dense* kedua: memiliki 32 neuron. Sama seperti lapisan sebelumnya, diikuti oleh batch normalization, fungsi aktivasi leaky relu (α=0.2), dan dropout dengan laju 0.3**.** Regularisasi L2 dengan nilai 1e−4 juga diterapkan. Lapisan-lapisan ini bertugas untuk memetakan fitur-fitur abstrak yang diekstraksi oleh *feature extractor* ke dalam representasi yang lebih tinggi dan spesifik untuk tugas klasifikasi.
d. Output Layer
   Lapisan terakhir terdiri dari satu neuron dengan fungsi aktivasi sigmoid. Fungsi aktivasi Sigmoid menghasilkan nilai output antara 0 dan 1, yang ideal untuk tugas klasifikasi biner, di mana nilai tersebut dapat diinterpretasikan sebagai probabilitas kelas positif (misalnya, *Diabetic Retinopathy*). Penggunaan dtype='float32' pada lapisan aktivasi memastikan output probabilitas dalam presisi tunggal.

Model ini dikembangkan menggunakan *Keras Functional API* dari TensorFlow, yang memungkinkan fleksibilitas dalam mendefinisikan aliran data antar lapisan. Implementasi juga memanfaatkan tf.distribute.Strategy untuk mengoptimalkan penggunaan perangkat keras yang tersedia selama pelatihan.

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 384, 384, 3)	0
lambda (Lambda)	(None, 1280)	0
dense (Dense)	(None, 64)	81,984
batch_normalization (BatchNormalization)	(None, 64)	256
leaky_re_lu (LeakyReLU)	(None, 64)	0
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 32)	2,080
batch_normalization_1 (BatchNormalization)	(None, 32)	128
leaky_re_lu_1 (LeakyReLU)	(None, 32)	0
dropout_1 (Dropout)	(None, 32)	0
dense_2 (Dense)	(None, 1)	33
activation (Activation)	(None, 1)	0

Gambar 4. Ringkasan arsitektur model

Dari gambar 4, dapat diamati beberapa poin penting. Input layer menerima citra dengan ukuran(384, 384, 3), menunjukkan citra RGB beresolusi 384x384 piksel. Tidak ada parameter yang perlu dilatih pada lapisan input.

Pada *lambda* merupakan lapisan yang berfungsi sebagai *wrapper* untuk EfficientNet-Lite0 dari TensorFlow Hub. Outputnya adalah vektor fitur dengan dimensi 1280, yang merupakan hasil ekstraksi fitur oleh EfficientNet-Lite0. Penting untuk dicatat bahwa lapisan ini tidak memiliki parameter yang dapat dilatih (param # = 0) karena EfficientNet-Lite0 diatur sebagai trainable=False.

Lapisan *dense* memiliki 64 neuron dan menyumbang 81.984 parameter. Ini merupakan kontribusi parameter terbesar karena menerima input 1280 fitur dari lapisan lambda. Lapisan dense 1 dengan 32 neuran, memiliki 2.080 parameter. lapisan dense 2 yakni lapisan output dengan 1 neuron, memiliki 33 parameter.

Lapisan *Batch Normalization* memiliki sejumlah kecil parameter (256 dan 128) yang digunakan untuk menyesuaikan rata-rata dan varians aktivasi. Lapisan aktivasi dan *dropout* tidak memiliki parameter yang dapat dilatih (param # = 0) karena fungsi utamanya adalah transformasi non-linear atau regulasi.

Model ini memiliki total 84.481 parameter. Dari jumlah tersebut, sebanyak 84.289 parameter bersifat trainable, yang berasal dari bagian *Custom Classification Head*, termasuk lapisan dense dan batch normalization. Jumlah parameter yang dapat dilatih ini tergolong kecil karena feature extractor EfficientNet-Lite0 dibekukan selama pelatihan. Sementara itu, hanya terdapat 192 parameter yang non-trainable, semuanya berasal dari EfficientNet-Lite0 yang dibekukan. Hal ini menunjukkan bahwa model sangat efisien dalam hal jumlah parameter yang perlu dioptimalkan selama proses pelatihan.

4.4. Penerapan dan Pelatihan Model

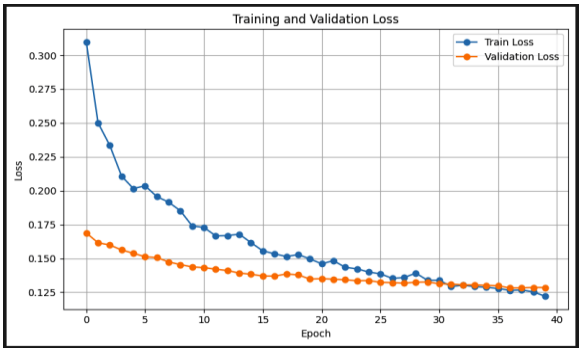
Pelatihan model dilakukan selama maksimal 40 epoch dengan pengaturan early stopping, yang akan menghentikan pelatihan apabila tidak terdapat peningkatan signifikan pada nilai *validation loss*. Pelatihan menggunakan strategi *tf.distribute.get_strategy* agar kompatibel dengan arsitektur Mac M1 dan mendukung komputasi efisien.

Model dibangun menggunakan pendekatan Keras Functional API dengan memanfaatkan *feature extractor* dari arsitektur EfficientNet-Lite0 melalui TensorFlow Hub, yang berperan sebagai ekstraktor fitur tanpa pelatihan ulang (trainable=False). Ukuran input citra ditetapkan sebesar 384×384 piksel.

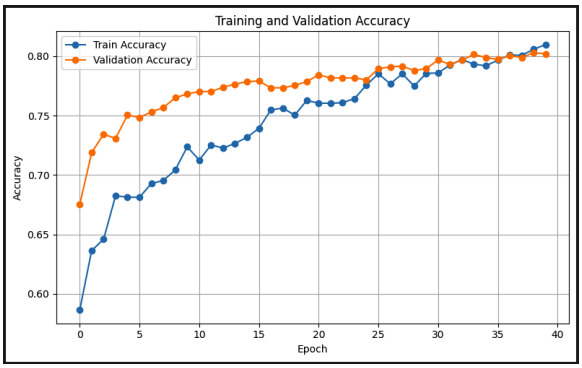
Setelah ekstraksi fitur, model menambahkan beberapa lapisan dense sebagai *custom classification head*. Terdapat dua lapisan dense berturut-turut (64 dan 32 neuron), masing-masing dilengkapi dengan regularisasi L2 (l2(1e-4)), batch normalization, fungsi aktivasi Leaky ReLU (alpha=0.2), dan dropout untuk mencegah overfitting (sebesar 0.4 dan 0.3). Lapisan output terdiri dari satu neuron dengan fungsi aktivasi sigmoid untuk tugas klasifikasi biner (retinopati diabetik atau normal).

Model dikompilasi menggunakan optimizer Adam dengan *learning rate* 1e-4, fungsi loss *Binary Focal Crossentropy* untuk menangani ketidakseimbangan kelas, serta metrik akurasi sebagai evaluasi utama. Berdasarkan proses pelatihan, model menunjukkan tren konvergensi yang stabil, ditandai dengan penurunan nilai *loss* secara bertahap dan peningkatan akurasi yang konsisten pada data pelatihan maupun validasi.

Berdasarkan Gambar 6 Model mencapai akurasi pelatihan sebesar 81.2% dan akurasi validasi sebesar 80.0%. Kemudian dari Gambar 5 Nilai *loss* pada pelatihan menurun dari 0.31 menjadi 0.125, dan *validation loss* relatif stabil pada kisaran 0.13–0.14, menunjukkan proses pelatihan yang efisien dan minim *overfitting*. Grafik akurasi memperlihatkan bahwa model tidak mengalami fluktuasi ekstrem selama pelatihan, yang menandakan kestabilan proses dan generalisasi yang baik terhadap data validasi.



Gambar 5. Grafik *training dan validation loss*



Gambar 6. Grafik *training dan validation accuracy*

Selain itu, penggunaan mekanisme *early stopping* membantu menghentikan pelatihan secara otomatis saat model mulai stagnan, menjaga model tetap optimal tanpa mengalami *overfitting*.

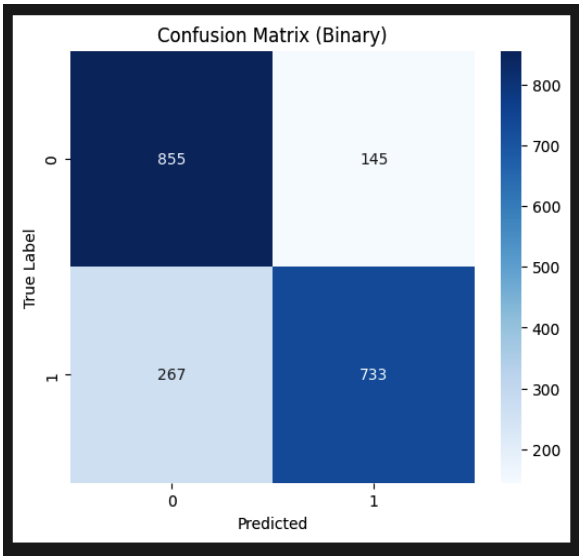
4.5. Evaluasi Model pada Data Uji

Setelah proses pelatihan selesai, model dievaluasi menggunakan dataset pengujian untuk mengukur kemampuan generalisasi terhadap data yang belum pernah dilihat sebelumnya. Dataset uji terdiri dari 980 citra fundus yang telah melalui preprocessing dan pelabelan biner (0 untuk non DR, 1 untuk DR).

Hasil evaluasi menunjukkan bahwa model mencapai akurasi sebesar 80.06% dan loss sebesar 0.4117 pada data uji. Angka ini mencerminkan konsistensi performa model, yang sejalan dengan hasil validasi selama proses pelatihan.

Evaluasi lebih lanjut dilakukan dengan menggunakan *confusion matrix*, yang menggambarkan distribusi prediksi model terhadap label yang sebenarnya.

Berdasarkan Gambar 7, Model berhasil mengidentifikasi 855 citra fundus normal dengan tepat (*true negative*), menunjukkan kemampuannya dalam mengenali mata yang sehat. Namun, terdapat 145 citra normal yang salah diklasifikasikan sebagai mengalami retinopati diabetik (*false positive*), berpotensi menimbulkan alarm palsu dan rujukan yang tidak perlu.



Gambar 7. Confusion matrix hasil evaluasi pada data uji

Sebaliknya, 267 citra yang sebenarnya mengalami retinopati diabetik tidak terdeteksi oleh model dan diklasifikasikan sebagai normal (*false negative*). Ini merupakan kesalahan yang cukup serius karena dapat menyebabkan keterlambatan penanganan.

Sementara itu, model mampu mengklasifikasikan 733 citra penderita retinopati diabetik dengan benar (*true positive*), menunjukkan efektivitas dalam mendeteksi kasus yang membutuhkan perhatian medis.

Secara keseluruhan, hasil ini menunjukkan bahwa model memiliki performa yang cukup baik dalam mengklasifikasikan citra fundus, namun masih terdapat ruang untuk perbaikan terutama dalam mengurangi tingkat false negative yang dapat berdampak pada keselamatan pasien.

Dari nilai-nilai tersebut, dihitung beberapa metrik penting:

a. *Accuracy*

$$\frac{TP+TN}{TP+TN+FP+FN} = \frac{733+855}{733+855+145+267} \approx 0.794$$

b. *Precision*

$$\frac{TP}{TP+FP} = \frac{733}{733+145} \approx 0.835$$

c. *Recall*

$$\frac{TP}{TP+FN} = \frac{733}{733+267} \approx 0.733$$

d. *F1 Score*

$$\frac{2x\ (Precision\ x\ Recall)}{Precision + Recall} \approx 0.780$$

Tabel 2. Tabel metrik evaluasi model

Metrik	Nilai
Accuracy	79.4%
Precision	83.5%
Recall	73.3%
F1-Score	78.0%

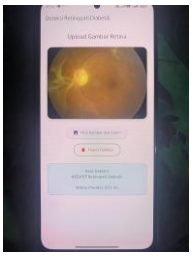
Berdasarkan hasil pada Tabel 2 model memperoleh akurasi sebesar 79.4%, yang menunjukkan bahwa hampir 8 dari 10 prediksi model sudah tepat. Namun demikian, nilai *recall* sebesar 73.3% mengindikasikan bahwa masih ada sejumlah kasus retinopati diabetik yang tidak berhasil dideteksi oleh model (false negative). Sebaliknya, *precision* tinggi (83.5%) menandakan bahwa sebagian besar prediksi positif model benar-benar mengandung retinopati.

Perbedaan antara precision dan recall menghasilkan *F1-score* sebesar 78.0%, yang merupakan gabungan dari keduanya. Nilai ini menunjukkan bahwa model bekerja cukup baik, namun masih perlu ditingkatkan pada bagian recall agar lebih peka dalam mendeteksi kasus positif. Ini penting karena dalam konteks medis, kesalahan tidak mendeteksi pasien yang sakit bisa berdampak serius.

4.6. Konversi Model ke Tensorflow Lite dan Pengujian pada Perangkat Seluler

Setelah proses pelatihan selesai, model dikonversi ke format *TensorFlow Lite* untuk memungkinkan integrasi ke dalam aplikasi mobile berbasis Flutter. Proses konversi dilakukan menggunakan *TFLiteConverter* dengan langkah-langkah sebagai berikut.

Dalam proses konversi model ke format *TensorFlow Lite*, beberapa konfigurasi penting dilakukan untuk memastikan kompatibilitas dan efisiensi. Pertama, digunakan metode *from_saved_model* untuk mengonversi model yang telah disimpan dalam format *SavedModel*. Selanjutnya, dilakukan pengaturan *supported_ops* untuk menentukan operator-operator yang didukung, guna memastikan model dapat dijalankan dengan baik pada perangkat mobile. Optimasi juga diterapkan dengan menggunakan *Optimize.DEFAULT*, yang mengaktifkan optimasi default untuk mengurangi ukuran model sekaligus meningkatkan efisiensi. Terakhir, ditentukan spesifikasi tipe data tf.float32 untuk memastikan konsistensi dan kompatibilitas tipe data yang digunakan dalam model *TFLite*.



Gambar 8. Tampilan hasil klasifikasi negatif DR pada aplikasi mobile di perangkat redmi note 12 4G

Gambar 9. Tampilan hasil klasifikasi positif DR pada aplikasi mobile di perangkat redmi note 12 4G

Pada Gambar 8 dan 9 terlihat model *TFLite* yang telah dikonversi berhasil diintegrasikan ke dalam aplikasi Flutter dan diuji pada perangkat Redmi Note 12 4G. Hasil pengujian menunjukkan bahwa model mampu melakukan inferensi dengan waktu respon rata-rata di bawah satu detik per citra.

Selain itu, penggunaan *CPU* dan memori selama proses inferensi tetap berada dalam batas yang wajar, sehingga tidak menyebabkan lag maupun overheating pada perangkat. Dari segi akurasi, hasil klasifikasi pada perangkat mobile tetap konsisten dengan hasil yang diperoleh di lingkungan pengembangan, menandakan bahwa proses konversi ke *TFLite* tidak menurunkan performa model. Integrasi dengan aplikasi *Flutter* juga berjalan dengan lancar melalui penggunaan plugin *tflite_flutter*, yang memungkinkan inferensi dilakukan langsung di perangkat tanpa membutuhkan koneksi internet.

Pengujian ini membuktikan bahwa model yang dikembangkan tidak hanya akurat, tetapi juga efisien dan praktis untuk diterapkan dalam aplikasi mobile, khususnya di lingkungan dengan sumber daya terbatas.

5. KESIMPULAN DAN SARAN

Penelitian ini berhasil mengembangkan model Lightweight CNN berbasis EfficientNet-Lite0 untuk mendeteksi retinopati diabetik dari citra fundus, dengan *accuracy* 79,4%, *precision* 83,5%, *recall* 73,3%, dan *f1-score* 78,0%. Model telah dikonversi ke *TensorFlow Lite* dan diintegrasikan ke dalam aplikasi Flutter, menunjukkan performa inferensi yang cepat, stabil, dan efisien di perangkat Android tanpa penurunan akurasi. Hasil ini menunjukkan bahwa model tidak hanya efektif secara algoritmik, tetapi juga layak digunakan dalam aplikasi seluler nyata untuk deteksi dini retinopati diabetik. Untuk penelitian selanjutnya, disarankan agar peningkatan performa *recall* menjadi prioritas, mengingat pentingnya mendeteksi kasus positif secara akurat guna mencegah keterlambatan diagnosis. Upaya ini dapat dilakukan melalui penyeimbangan kelas, augmentasi data yang lebih beragam, dan penggunaan metode pembelajaran yang sensitif terhadap biaya kesalahan klasifikasi.

DAFTAR PUSTAKA

[1] A. Riset *dkk.*, "FAKUMI MEDICAL JOURNAL Hubungan Jenis Retinopati Diabetik dengan Lama Menderita Diabetes Melitus dan Kadar HbA1C," 2022.

[2] R. F. N. Purnama, "Retinopati Diabetik : Manifestasi Klinis, Diagnosis, Tatalaksana dan Pencegahan," *Lombok Medical Journal*, vol. 2, no. 1, hlm. 39–42, Mei 2023, doi: 10.29303/lmj.v2i1.2410.

[3] C.-C. Wang, C.-T. Chiu, dan J.-Y. Chang, "EfficientNet-eLite: Extremely Lightweight and Efficient CNN Models for Edge Devices by Network Candidate Search," Sep 2020, [Daring]. Tersedia pada: http://arxiv.org/abs/2009.07409

[4] R. Shalehuddin Albawani *dkk.*, "IMPLEMENTASI SEBLOCK PADA KLASIFIKASI CITRA PENYAKIT MATA MANUSIA DENGAN ARSITEKTUR MOBILENETV3-SMALL," 2024.

[5] O. Trisera *dkk.*, "Retinopati Diabetik yang Mengancam Penglihatan," 2024.

[6] P. Muslima, E. iskandar, dan A. Prahasta, "VALIDITAS PEMERIKSAAN FOTO FUNDUS PORTABEL 3 LAPANGPANDANG MIDRIATIK SEBAGAI ALAT SKRINING RETINOPATIDIABETIK RINGAN," vol. 2, 2020.

[7] R. Rodiah, S. Madenda, dan D. Tri, "Pengolahan Citra FUndus Diabetik Retinopati edisi 2," 2022. [Daring]. Tersedia pada: https://www.researchgate.net/publication/330618377

[8] E. A. N. Munfati dan A. Witanti, "Klasifikasi Buah dan Sayuran Segar atau Busuk Menggunakan Convolutional Neural Network," vol. 9, hlm. 27–38, 2024.

[9] S. H. Abdullah, R. Magdalena, dan Fu'adah Yunendah Nur, "KLASIFIKASI DIABETIC RETINOPATHY BERBASIS PENGOLAHAN CITRA FUNDUS DAN DEEP LEARNING," vol. 2, 2022.

[10] J. I. Massie dan A. M. Widodo, "DEEP LEARNING UNTUK KLASIFIKASI PENYAKIT RETINOPATI DIABETIK MENGGUNAKAN ARSITEKTUR ALEXNET DAN GENERATIVE ADVERSARIAL NETWORK," *Jurnal SIMETRIS*, vol. 14, no. 2, 2023.

[11] F. H. Syahrul dan P. S. Sasongko, "Penerapan Convolutional Neural Network Untuk Klasifikasi Tingkat Keparahan Retinopati Diabetik Pada Penderita Diabetes Melitus," 2022.

[12] U. Erdiansyah, A. I. Lubis, dan G. Syahputra, "Klasifikasi Penyakit Diabetic Retinopathy Menggunakan Multilayer Perceptron," 2022.

[13] J. E. Widyaya dan S. Budi, "Pengaruh Preprocessing Terhadap Klasifikasi Diabetic Retinopathy dengan Pendekatan Transfer Learning Convolutional Neural Network," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 7, no. 1, Apr 2021, doi: 10.28932/jutisi.v7i1.3327.

[14] A. E. Minarno, A. Dwija Bagaskara, F. Bimantoro, dan W. Suharso, "Classification of Diabetic Retinopathy Based on Fundus Image Using InceptionV3," 2025. [Daring]. Tersedia pada: www.joiv.org/index.php/joiv

[15] N. Raji, S. Manohar, dan V. Rajasekar, "From Pixels to Precision_ Techniques for Image Dataset Refinement," hlm. 276–282, 2025.