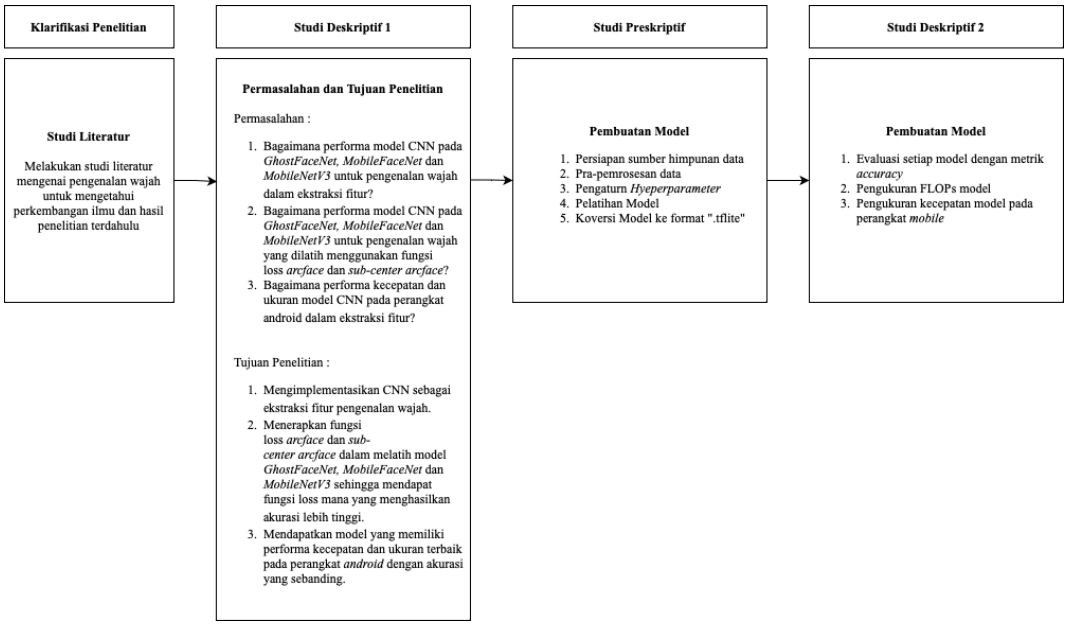
BAB III
METODE PENELITIAN

3.1 Desain Penelitian

Desain penelitian yang akan digunakan pada penelitian ini adalah menggunakan *Design Research Methodology* (DRM). DRM terdiri dari empat tahap yaitu Klarifikasi Penelitian atau *Research Clarification* (RC), Studi Deskriptif I atau *Descriptive Study I* (DS-I), Studi Preskriptif atau *Perscriptive Study* (PS), dan Studi Deskriptif II atau *Descriptive Study II* (DS-II). Dalam DRM jika penelitian yang dilakukan melalui tahap RC hingga DS-II tanpa iterasi, maka tipe penelitian tersebut termasuk ke dalam Pengembangan Pendukung (*Development of Support*) (Blessing & Chakrabarti, 2009). (Andriy Burkov, 2019). Berdasarkan kerangka kerja DRM, Gambar 3.1 menunjukkan skema penelitian yang akan dilakukan.



Gambar 3.1 Skema Penelitian

Berikut merupakan penjelasan dari Skema Penelitian pada Gambar 3.1 Skema Penelitian:

1. Klarifikasi Penelitian

   Pada tahap ini, studi literatur dilakukan untuk mengetahui perkembangan ilmu, mengetahui hasil penelitian terdahulu, memperjelas dan memfokuskan tujuan penelitian. Studi literatur menggunakan penelitian-penelitian yang relevan dan buku-buku untuk memperdalam pengetahuan tentang bidang yang diteliti.

2. Studi Deskriptif I

Pada tahap ini, permasalahan dan tujuan penelitian berdasarkan studi literatur di definisikan. Permasalahan merupakan pertanyaan-pertanyaan yang akan dipecahkan oleh penelitian yang dibuat berdasarkan latar belakang. Terdapat beberapa pertanyaan yang menunjukkan fokus permasalahan penelitian. Tujuan merupakan hasil yang diharapkan dan ingin dicapai pada penelitian yang akan dilakukan.

3. Studi Preskriptif

Pada tahap ini, langkah-langkah pelatihan model pengenalan wajah diterapkan. Langkah- langkah tersebut mengacu pada penelitian terdahulu yang kemudian dimodifikasi untuk meningkatkan performa model yang dihasilkan.

4. Studi Deskriptif 2

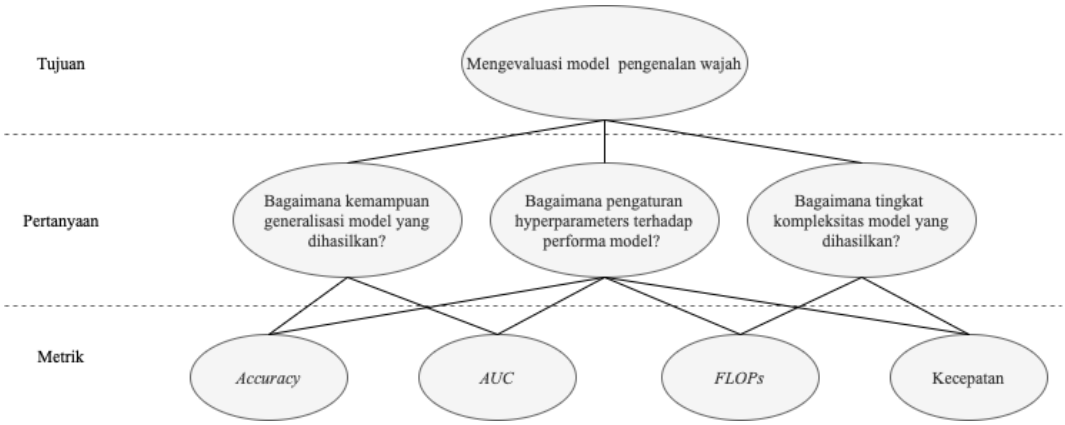
Pada tahap ini, performa dari model yang dihasilkan akan dibandingkan dengan model – model penelitian terdahulu. Beberapa metrik yang akan digunakan *accuracy*, AUC, FLOPs dan kecepatan. Kemudian, teknik validasi digunakan untuk mengevaluasi performa model yang telah dilatih dengan menggunakan *10-fold cross-validation*.

3.2 Sumber Himpunan Data

Sumber himpunan data yang digunakan pada penelitian ini adalah *dataset* publik yang pernah digunakan oleh penelitian terdahulu (Deng dkk., 2022) (Chen dkk., 2018) (Alansari dkk., 2023). *Dataset* ini tersedia di repositori github yang dibuat oleh InsightFace (Deng dkk., 2022) dan dapat diunduh pada laman https://github.com/deepinsight/insightface/tree/master/recognition/_datasets_. *Dataset* pelatihan yang akan digunakan dalam penelitian ini adalah CASIA-Webface yang sudah dibersihkan dengan jumlah citra wajah 490,623 dari 10,572 subjek. *Dataset* evaluasi menggunakan LFW dengan jumlah wajah citra 12,000, CFP-FP dengan jumlah citra wajah 14,000 dan AgeDB-30 dengan jumlah citra wajah 12,000. *Dataset* tersebut melalui pra-pemrosesan gambar dengan mengubah dimensi menjadi 112×112 piksel.

3.3 Instrumen Penelitian

Instrumen atau alat pengumpul data yang digunakan pada penelitian ini terdiri dari beberapa hal, di antaranya adalah komputasi yang digunakan, bahasa pemrograman yang digunakan, perangkat lunak dan *library* yang digunakan, serta metrik-metrik untuk mengevaluasi model pembelajaran mesin. Adapun untuk menetapkan fokus penelitian digunakan pendekatan *Goal Question Metric* (GQM) sebagai berikut:



Gambar 3.2 Goal Question Metrics (GQM)

Pada penelitian ini, platform komunitas *Artificial Intelligence* (AI) & *Machine Learning* (ML) yaitu Kaggle Notebook digunakan sebagai komputasi untuk mengembangkan model CNN dengan batas per-sesi maksimal 12 jam. Spesifikasi komputasi tersebut ditujukkan pada Tabel 3.1. Adapun salah satu GPU yang dapat digunakan pada Kaggle Notebook ditujukkan pada

Tabel 3.2. Bahasa pemrograman yang digunakan untuk mengembangkan model pembelajaran mesin adalah bahasa pemrograman Python. Evaluasi kecepatan dilakukan pada perangkat android dengan spesifikasi yang ditunjukkan pada Tabel 3.3. Perangkat lunak atau *library* pendukung yang digunakan dalam penelitian ditunjukkan pada Tabel 3.4.

Tabel 3.1

Spesifikasi Komputasi Kaggle NoteBook

vCPUs	4
Memory	30 GB
Storage	20 GB

Tabel 3.2

GPU Kaggle NoteBook

Nama	**Jumlah**	**Core**	**VRAM**
P100 GPU	1	4	29 GB
T4 GPU	2	4	29 GB

Tabel 3.3

Spesifikasi Perangkat *Android*

Nama	Google Pixel 2
RAM	4 GB
Chipset	Qualcom Snapdragon 835
GPU	Adreno 540

Tabel 3.4

Perangkat Lunak dan *Library* yang digunakan

No.	Nama	Deskripsi
1	Visual Studio Code (VS Code)	VS Code digunakan untuk editor kode program pembelajaran mesin yang digunakan dalam modifikasi model CNN
2	Python	Bahasa pemrograman yang digunakan dalam pengembangan mesin pada penelitian adalah Python.
3	Python Package Index (PIP)	PIP merupakan sistem pengelolaan paket (*package manager*) untuk bahasa pemrograman Python yang digunakan untuk menginstal beberapa *library* pendukung dalam penelitian ini.
4	Jupyter Notebook	Jupyter Notebook yang digunakan adalah berasal dari ekstensi VS Code. Jupyter Notebook digunakan untuk menuliskan dan menjalankan kode pembelajaran mesin berbasis Python dengan ekstensi file ".ipynb".

5	Numpy	Numpy adalah *library* Python yang digunakan untuk mengelola dan memanipulasi data dalam bentuk array dan matriks.
6	Matplotlib	Matplotlib adalah *library* Python yang digunakan untuk membuat plot grafik dan visualisasi data.
7	Scikit-learn	Scikit-learn adalah *library* Python yang digunakan untuk membuat dan mengevaluasi model pembelajaran mesin. Beberapa algoritma dan fungsi pada scikit-learn digunakan dalam penelitian ini, seperti menormalisasi *embedding* dan evaluasi model.
8	Mxnet	Mxnet adalah *library* deep learning yang digunakan untuk membuat model CNN. Mxnet digunakan pada penelitian ini untuk pra-pemrosesan data dari format mxnet menjadi tensorflow *dataset*.
9	Tensorflow	Tensorflow adalah *library* Python yang digunakan untuk membuat model CNN. Tensorflow digunakan dalam penelitian ini untuk melatih algoritma Convolutional Neural Network (CNN) dan mengkonversi model menjadi format ".tflite".
10	Keras_insightface (Leondgarse, 2022b)	Keras_insightface merupakan implementasi keras dari Insightface yang merupakan hasil penelitian (Deng dkk., 2022). Keras_insightface digunakan dalam penelitian ini untuk pra-pemrosesan data, mengubah arsitektur model, melatih model dan evaluasi model.
11	Android Studio	Android Studio adalah IDE yang digunakan untuk mengukur kecepatan model pada perangkat *android*.
12	Keras_cv_attention_models (Leondgarse, 2022a)	Keras_cv_attention_models merupakan *library* Python yang digunakan untuk mengukur FLOPs pada model.

Tabel 3.5

Confusion Matrix

Jenis Kelas	Aktual	
Prediksi	TP (*True Positive*)	FP (*False Positive*)
	TN (*True Negative*)	FN (*False Negative*)

Confusion Matrix pada Tabel 3.5 merupakan acuan dalam mengukur performa model yang dihasilkan setelah melalui tahapan klasifikasi kemiripan. Selanjutnya,

akan dikalkulasikan untuk mendapatkan metrik evaluasi tingkat akurasi yang terdapat pada Tabel 3.6. Menurut Gorunescu (2011), kategori tingkat performa model dalam klasifikasi menggunakan AUC ditunjukkan pada Tabel 3.7. Kemudian metrik – metrik evaluasi performa model ditunjukkan pada Tabel 3.8.

Tabel 3.6

Metrik - Metrik Evaluasi Tingkat Akurasi

No.	Nama Metrik	Persamaan	Deskripsi
1	TPR (*True Positive Rate*)	$TPR_i = \frac{TP}{TP + FP}$	TPR mengukur sejauh mana model mampu mengidentifikasi nilai positif yang benar.
2	FPR (*False Positive Rate*)	$FPR = \frac{FP}{FP + TN}$	FPR mengukur sejauh mana model membuat kesalahan.
3	ROC (*Receiver Operating Characteristics*)		ROC merupakan representasi grafis hubungan antara TPR dan FPR. Semakin dekat ROC dengan sumbu Y (0,1) maka kinerja model dikatakan semakin baik.
3	AUC (*Area Under Curve*)	$AUC = \int_0^1 TPR\ d\ FPR$	AUC mengukur area dibawah kurva *Receiver Operating Characteristics* (ROC).
4	*Accuracy*	$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$	*Accuracy* mengukur sejauh mana klasifikasi dapat memprediksi dengan benar semua kelas.

Tabel 3.7

Kategori nilai AUC

0.9 – 1.0	Sangat Baik
0.8 – 0.9	Baik
0.7 – 0.8	Cukup
0.6 – 0.7	Buruk
0.5 - 0.6	Sangat Buruk

Click or tap here to enter text.

Tabel 3.8

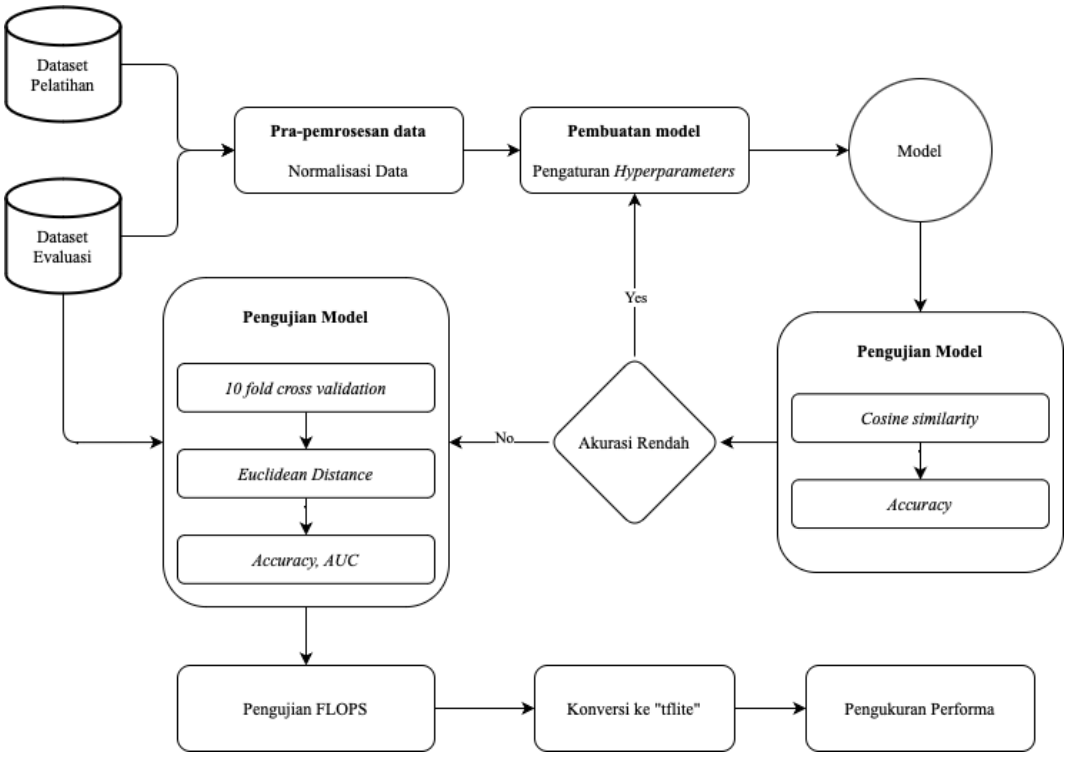
Metrik – Metrik Evaluasi Performa model

No.	Nama Metrik	Deskripsi
1	FLOPs (*Floating Operations*)	FLOPs mengukur seberapa kompleks komputasi suatu model.
2	Kecepatan	Kecepatan mengukur waktu sebelum dan sesudah model melakukan prediksi atau ekstraksi fitur.

Metrik FLOPs yang ditunjukkan pada Tabel 3.8 merupakan salah satu metrik untuk mengukur penggunaan sumber daya komputasi dan latensi pada sebuah model (Tang dkk., 2018).

3.4 Prosedur Penelitian

Penelitian ini akan menggunakan kerangka kerja yang mengacu pada beberapa penelitian sebelumnya dan telah dimodifikasi untuk proses pembuatan model yang kemudian akan di evaluasi menggunakan beberapa rumus-rumus terkait untuk mengukur performa model. Kerangka kerja yang diusulkan dalam penelitian ini ditunjukkan oleh Gambar 3.3.



Gambar 3.3 Kerangka kerja yang diusulkan

Berikut adalah langkah-langkah yang diimplementasikan pada prosedur tersebut:

1. Persiapan Data

   Pada tahap ini, data yang telah diunduh merupakan *dataset* pelatihan dan *dataset* evaluasi. *Dataset* tersebut memiliki format ".bin". Kemudian, data tersebut diekstrak menjadi folder dengan nama sebagai subjek atau kelas dan berisi citra wajah berdasarkan subjek atau kelas.

2. Pra-pemrosesan Gambar

   Pra pemrosesan gambar dilakukan dengan membaca semua kelas dan dikonversi ke dalam format *tensorflow dataset*. Kemudian, citra wajah akan di normalisasi.

3. Pembuatan Model

   Sebelum pelatihan model, pengaturan *hyperparameters* dilakukan dengan menerapkan fungsi loss, merubah arsitektur model dan menentukan parameter lain seperti *optimizer, batch size, learning rate* dan jumlah *epochs*. Kemudian, *callback* setiap akhir *epoch* digunakan seperti evaluasi, pengurangan *learning rate* dan penyimpanan model secara otomatis.

4. Klasifikasi dan Evaluasi

   Saat pelatihan model berlangsung, tahapan klasifikasi dan evaluasi digunakan dengan menghasilkan metrik *accuracy* pada *dataset* evaluasi tanpa menggunakan teknik validasi *10-fold cross validation*. Setelah pelatihan selesai, klasifikasi dan evaluasi dilakukan ulang dengan teknik validasi *10-fold cross validation*. Model yang terbaik akan dibandingkan dengan model lainnya berdasarkan metrik – metrik evaluasi yang telah dipaparkan.

5. Analisis Hasil

   Hasil dari tahap sebelumnya akan dianalisis lebih lanjut pada tahap ini. Model dengan tingkat akurasi rendah atau tidak sebanding dengan penelitian terdahulu akan dilatih ulang dengan mengubah *hyperparameters.*

6. Pengujian Performa

   Model – model dengan hasil terbaik akan dilakukan pengujian performa dengan menghitung FLOPs. Kemudian, model akan dikonversi ke format ".tflite" untuk dapat dilakukan pengujian performa kecepatan pada perangkat *android*.

3.5 Analisis Data

Seluruh hasil yang diperoleh dari setiap model akan dianalisis lebih lanjut dengan analisis komparatif untuk mendapatkan kesimpulan penelitian. Hasil dari setiap metrik diperoleh secara langsung menggunakan *scikit-learn* dan Android Studio. Visualisasi terhadap proses pelatihan model dan kurva ROC dilakukan menggunakan library *matplotlib* dan didukung juga dengan library *scikit-learn*.