TUGAS RISET INFORMATIKA
(METODOLOGI)

Dosen Pengampu :
Dr. Basuki Rahmat, S.Si. MT.

Disusun Oleh :

EGAR FIRMANSYAH
NPM. 22081010012

PROGRAM STUDI INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN” JAWA TIMUR
SURABAYA
2025

Metodologi

Fase 1 — Persiapan Dataset (Minggu 1–2)

- **Kelas (8 total, seimbang):**

  Healthy, Bacterial Leaf Blight, Brown Spot, Leaf Smut, Blast, Hispa, Stem Borer, BPH.

  (500 citra/kelas → total 4.000 gambar)

- **Labeling:**
  - YOLOv8n-Seg → polygon mask + bounding box (via Roboflow/CVAT)
  - EfficientNet → klasifikasi (CSV: filename, class_id)
  - U-Net → mask biner (0=sehat, 1=sakit)

- **Split data:** Train 70% | Val 15% | Test 15%

- **Augmentasi (Albumentations):**
  - Rotate ±25°, Flip H/V, Brightness (0.7–1.3), Contrast (0.8–1.2), Mosaic (YOLO)
  - Gunakan `seed=42` untuk reproduktibilitas

- **Tips Labeling:**

  Label 200–300 manual → train YOLO baseline → auto-label sisanya → verifikasi cepat.

Fase 2 — Preprocessing

Model	Ukuran Input	Normalisasi	Catatan
YOLOv8n-Seg	640×640	0–1 (bawaan Ultralytics)	segmentasi daun
EfficientNet-B3	224×224	mean/std ImageNet	klasifikasi
U-Net Lite	256×256	0–1	mask severity

Fase 3 — Training Model (Minggu 3–8)

◇ YOLOv8n-Seg (Deteksi + Mask)

- Konfigurasi: imgsz=640, epochs=100, batch=16, optimizer=AdamW, lr=0.001
- Target:
  - mAP@0.5 ≥ 90%, Precision ≥ 92%, Recall ≥ 88%
- Output: `yolov8n_leaf_seg.pt`

◇ EfficientNet-B3 (Klasifikasi)

- Transfer learning (ImageNet)
- Freeze 80% layer, fine-tune 20%
- Epochs=50, batch=32, EarlyStopping aktif
- Target: Accuracy ≥ 93%, F1 ≥ 90%

- Output: `efficientnet_b3_classification.h5`

◇ **U-Net Lite (Severity)**

- Arsitektur: Mobile U-Net (MobileNetV2 backbone)
- Loss = 0.5×Dice + 0.5×Binary Cross Entropy
- Target: IoU ≥ 85%, Dice ≥ 88%, error <5%
- Output: `unet_lite_severity.h5`

Fase 4 — Evaluasi Model

Model	Evaluasi	Target
YOLOv8n-Seg	mAP@0.5, mAP@0.5:0.95, Precision, Recall	≥ 90%
EfficientNet-B3	Accuracy, Precision, Recall, F1, Confusion Matrix	≥ 93%
U-Net Lite	IoU, Dice, Pixel Acc, Severity Error	IoU ≥ 85%, error ≤ 5%

Fase 5 — Integrasi Pipeline (Python Desktop Test)

- Gabungkan ketiga model jadi satu pipeline skrip.
- Ukur waktu per tahap:

  YOLO (~30ms), EffNet (~15ms), U-Net (~30ms), total ~70–100ms.
- Tambahkan **decision logic otomatis**:

    - Skip U-Net jika daun sehat/conf < 0.8.
    - Overlay transparan + teks hasil di citra.

Fase 6 — Optimisasi Model

- Quantization: FP32 → FP16 (server), FP16 → INT8 (mobile).
- Pruning 10–30% neuron tak aktif.
- Validasi:
    - Drop akurasi ≤ 3%
    - Ukuran model ↓ ≥ 50%
    - Latensi ↑ 2–3× lebih cepat.

Fase 7 — Konversi Model

- Export model ke format **TFLite (INT8)** / **ONNX**
- Tes hasil konversi di Python desktop → pastikan output ≈ original.
- Ukur waktu inferensi tiap model di CPU/GPU.

Fase 8 — Integrasi ke Flutter (Minggu 11–12)

📱 Arsitektur Flutter

```
camera → isolate inference → YOLO → crop → EfficientNet →
decision → (U-Net) → render overlay
```

🔧 Implementasi Teknis

- Plugin: `camera`, `tflite_flutter`, `tflite_flutter_helper`
- Model inference berjalan di **isolate** (thread terpisah)
- **Frame skipping** aktif (1 frame per 0.5–1 detik)

- **Render overlay** (CustomPainter) semi-transparan (α=0.3)
- **Input kamera** 320–416 px (trade-off akurasi & FPS)

Target performa:

Device	**FPS**	**Latency**
Desktop (GPU)	25–30	<50ms
Android Mid (CPU)	5–7	~200–250ms
Android NNAPI/GPU	10–15	~100–150ms

Fase 9 — Pengujian & Benchmark

- Uji akurasi, FPS, suhu, dan konsumsi baterai.
- Tes pada beberapa perangkat (low/mid/high-end).
- Dataset lapangan (50–100 sampel nyata) dibandingkan dengan diagnosis manual pakar.

Fase 10 — Dokumentasi & Laporan

- Visualisasi hasil:
  - Grafik mAP, Accuracy, F1, IoU, FPS
  - Contoh overlay hasil prediksi
  - Error gallery (kasus salah klasifikasi)
- Tabel perbandingan:

  - Sebelum vs sesudah optimisasi (size, latency)
  - Model base vs INT8

- Benchmark real device (FPS, latency, battery usage).