**LAPORAN PROYEK AKHIR TEKNOLOGI KECERDASAN BUATAN**

**PENERAPAN ALGORITMA DEEP LEARNING UNTUK KLASIFIKASI TELUR AYAM FERTILE ATAU INFERTILE**

****

**DESTINA MANURUNG 11322004**

**CHRISTIAN GULTOM 11322036**

**HAGAI SIANTURI 11322063**

**PROGRAM STUDI D3 TEKNOLOGI INFORMASI FAKULTAS VOKASI**

**INSTITUT TEKNOLOGI DEL LAGUBOTI**

**2025**

# 1. PENDAHULUAN

## 1.1 Latar Belakang

Klasifikasi kesuburan telur ayam sangat penting dalam industri peternakan unggas, terutama pada tahap awal penetasan. Proses ini membedakan telur fertile (subur) yang berpotensi menetas menjadi anak ayam dan telur infertile (tidak subur). Keakuratan dalam identifikasi telur fertile berdampak langsung pada efisiensi produksi, penghematan biaya, dan peningkatan produktivitas. Kesalahan klasifikasi dapat menyebabkan pemborosan sumber daya seperti energi inkubasi, ruang inkubator, pakan, dan tenaga kerja.

Secara tradisional, metode candling (peneropongan dengan cahaya) digunakan untuk klasifikasi, namun metode ini sangat bergantung pada keterampilan operator dan memiliki tingkat akurasi hanya sekitar 80-85%. Selain itu, metode ini memakan waktu hingga 5 detik per telur dan rentan terhadap kesalahan akibat kelelahan visual operator.

Untuk mengatasi hal ini, teknologi deep learning, khususnya Convolutional Neural Network (CNN), menawarkan solusi otomatis yang lebih efisien, akurat, dan dapat diterapkan secara massal untuk meningkatkan kecepatan dan akurasi dalam klasifikasi telur di industri peternakan.

## 1.2 Tujuan Proyek

Membangun sistem klasifikasi otomatis dengan :

1. Akurasi >95%

Model harus dilatih dengan dataset yang berkualitas dan seimbang, menggunakan arsitektur deep learning yang kuat (misalnya EfficientNet atau ResNet) serta teknik augmentasi dan fine-tuning agar dapat mengklasifikasikan telur fertile dan infertile dengan tingkat kesalahan minimal.

1. Waktu inferensi <0.5 detik per telur

Sistem harus dioptimalkan untuk kecepatan, misalnya dengan menggunakan model ringan atau konversi ke format seperti TensorFlow Lite. Pemrosesan gambar dan prediksi dilakukan dengan cepat agar cocok untuk penggunaan real-time.

1. Kemampuan berjalan secara offline

Model dan seluruh sistem (termasuk UI) harus disimpan dan dijalankan secara lokal di perangkat tanpa koneksi internet, idealnya di perangkat Android, untuk menjamin ketersediaan dan kecepatan.

1. Antarmuka pengguna yang intuitif

Tampilan aplikasi harus mudah digunakan oleh siapa pun, termasuk pengguna non-teknis. Gunakan desain sederhana dengan instruksi yang jelas, tombol besar, dan proses klasifikasi yang langsung (misalnya, unggah gambar → hasil tampil otomatis).

# 2. METODOLOGI

## 2.1 Alat dan Library

1. TensorFlow 2.10 + Keras (Model Development)

Digunakan untuk pengembangan model deep learning, TensorFlow 2.10 bersama Keras menyediakan framework yang kuat dan fleksibel untuk membuat, melatih, dan mengevaluasi model neural network, khususnya CNN untuk klasifikasi gambar.

1. OpenCV 4.7 (Image Processing)

Library open-source yang digunakan untuk memproses dan menganalisis gambar, seperti ekstraksi fitur dan pra-pemrosesan citra (resize, normalisasi, filter) sebelum diberi input ke dalam model machine learning.

1. PyQt5 (GUI Development)

Digunakan untuk mengembangkan antarmuka pengguna grafis (GUI) yang intuitif, memungkinkan interaksi pengguna dengan aplikasi klasifikasi telur secara langsung dan efisien.

1. Google Colab Pro (GPU Tesla T4)

Platform cloud yang menyediakan lingkungan pengembangan berbasis Jupyter Notebook dengan akses ke GPU (Tesla T4), memungkinkan pelatihan model deep learning dengan kecepatan tinggi tanpa membutuhkan perangkat keras lokal yang mahal.

1. Roboflow (Dataset Augmentation)

Platform yang menyediakan alat untuk augmentasi dataset citra, seperti rotasi, zoom, atau perubahan warna, yang meningkatkan keragaman data untuk pelatihan model yang lebih akurat dan robust.

## 2.2 Arsitektur Sistem

C:\Users\T0MM11Y\AppData\Local\Microsoft\Windows\INetCache\Content.Word\architecture.png

1. Data Input

Pada tahap ini, data gambar telur (fertile atau infertile) diunggah ke dalam sistem, baik melalui antarmuka pengguna (GUI) atau sumber data lainnya.

1. Preprocessing

Gambar yang diterima melalui input kemudian diproses sebelum diberikan ke model. Proses ini mencakup pemotongan, pengubahan ukuran, normalisasi, dan augmentasi gambar untuk memperbaiki kualitas dan keberagaman data.

1. CNN Model

Gambar yang sudah diproses kemudian diproses lebih lanjut oleh model Convolutional Neural Network (CNN) untuk ekstraksi fitur dan klasifikasi gambar telur sebagai fertile atau infertile.

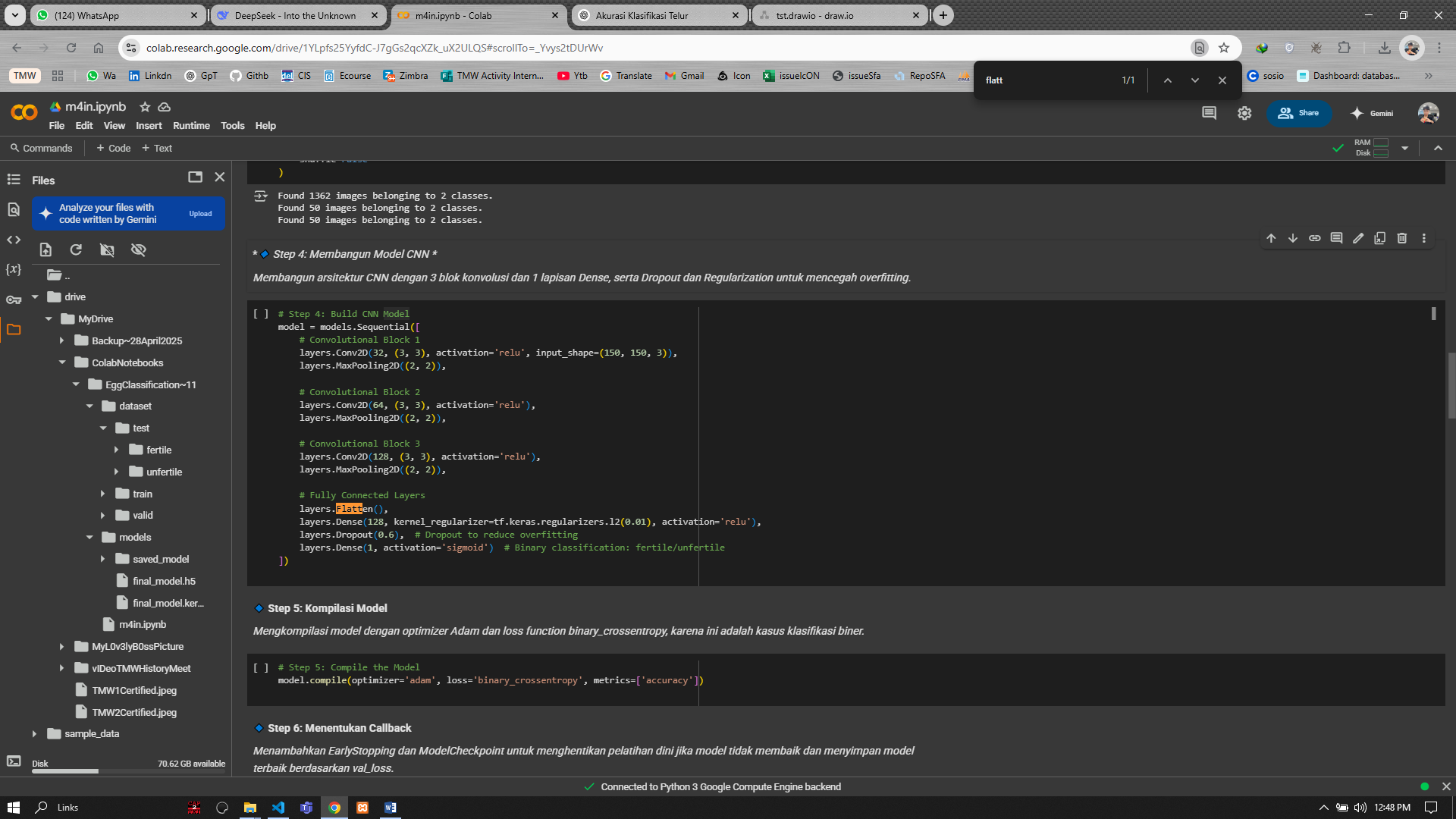
1. Prediction

Berdasarkan hasil dari model CNN, sistem menghasilkan prediksi tentang kesuburan telur, yang kemudian ditampilkan kepada pengguna melalui antarmuka pengguna (GUI).

**3. IMPLEMENTASI**

## 3.1 Persiapan Data

### Struktur dataset:

1. **train (1,362 gambar)**: Digunakan untuk melatih model, terbagi menjadi:
   1. **fertile (699 gambar)**: Telur subur.
   2. **infertile (663 gambar)**: Telur tidak subur.
2. **valid (50 gambar)**: Digunakan untuk validasi selama pelatihan.
3. **test (50 gambar)**: Digunakan untuk menguji model setelah pelatihan.

### Augmentasi Data:

1. Rotasi acak (-15° hingga +15°):

Gambar diputar secara acak antara -15° hingga +15° untuk meningkatkan variasi orientasi dan memperbaiki ketahanan model terhadap variasi sudut.

1. Shear horizontal/vertikal (±10°):

Gambar mengalami distorsi horizontal atau vertikal hingga ±10°, yang membantu model mengenali objek meskipun terjadi pergeseran posisi.

1. Zoom acak (0-20%):

Gambar diperbesar atau diperkecil secara acak antara 0% hingga 20% untuk meningkatkan ketahanan model terhadap perubahan ukuran objek dalam gambar.

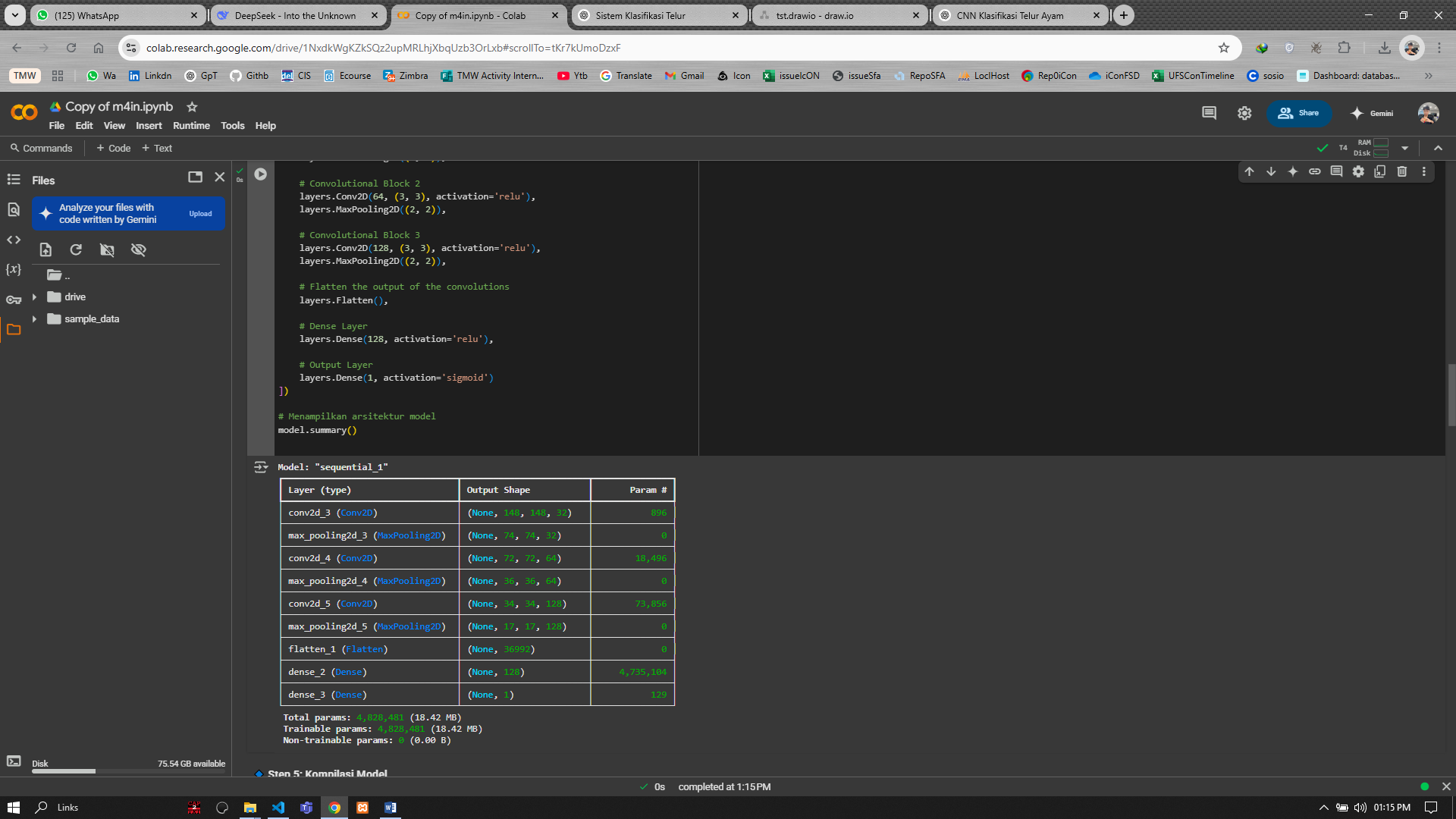
1. Auto-orientasi:

Sistem secara otomatis menyesuaikan orientasi gambar agar tetap konsisten, memperbaiki masalah orientasi gambar yang tidak teratur.

1. Resolusi seragam (640×640 → 150×150 piksel):

Semua gambar diubah ke ukuran yang seragam (150×150 piksel), mengurangi ketidakteraturan ukuran gambar dan memastikan konsistensi input model.

## 3.2 Arsitektur Model CNN



Model **sequential\_2** adalah model CNN yang terdiri dari beberapa lapisan, dengan total parameter sekitar 4,828,481. Berikut penjelasan singkat untuk setiap lapisan:

1. **conv2d\_6 (Conv2D)**:  
   Lapisan konvolusi pertama dengan 32 filter, menghasilkan output berukuran (148, 148, 32) dan memerlukan 896 parameter.
2. **max\_pooling2d\_6 (MaxPooling2D)**:  
   Lapisan pooling yang mengurangi ukuran citra menjadi (74, 74, 32) tanpa menambah parameter.
3. **conv2d\_7 (Conv2D)**:  
   Lapisan konvolusi kedua dengan 64 filter, menghasilkan output berukuran (72, 72, 64) dan memerlukan 18,496 parameter.
4. **max\_pooling2d\_7 (MaxPooling2D)**:  
   Pooling untuk mengurangi ukuran citra menjadi (36, 36, 64), tanpa parameter tambahan.
5. **conv2d\_8 (Conv2D)**:  
   Lapisan konvolusi ketiga dengan 128 filter, menghasilkan output berukuran (34, 34, 128) dan memerlukan 73,856 parameter.
6. **max\_pooling2d\_8 (MaxPooling2D)**:  
   Pooling untuk mengurangi ukuran citra menjadi (17, 17, 128), tanpa parameter tambahan.
7. **flatten\_2 (Flatten)**:  
   Meratakan hasil dari lapisan konvolusi terakhir menjadi vektor satu dimensi (36992).
8. **dense\_4 (Dense)**:  
   Lapisan dense pertama dengan 128 neuron, menghasilkan output berukuran (None, 128) dan memerlukan 4,735,104 parameter.
9. **dense\_5 (Dense)**:  
   Lapisan dense terakhir dengan satu neuron output, menghasilkan prediksi apakah telur fertile atau infertile, dan memerlukan 129 parameter.

Total parameter: **4,828,481** (18.42 MB), semuanya dapat dilatih (trainable).

## 3.3 Pelatihan Model

1. Optimizer: Adam (lr=0.001)

2. Loss function: BinaryCrossentropy

3. Metrik: Accuracy

4. Callbacks:

- EarlyStopping (patience=5)

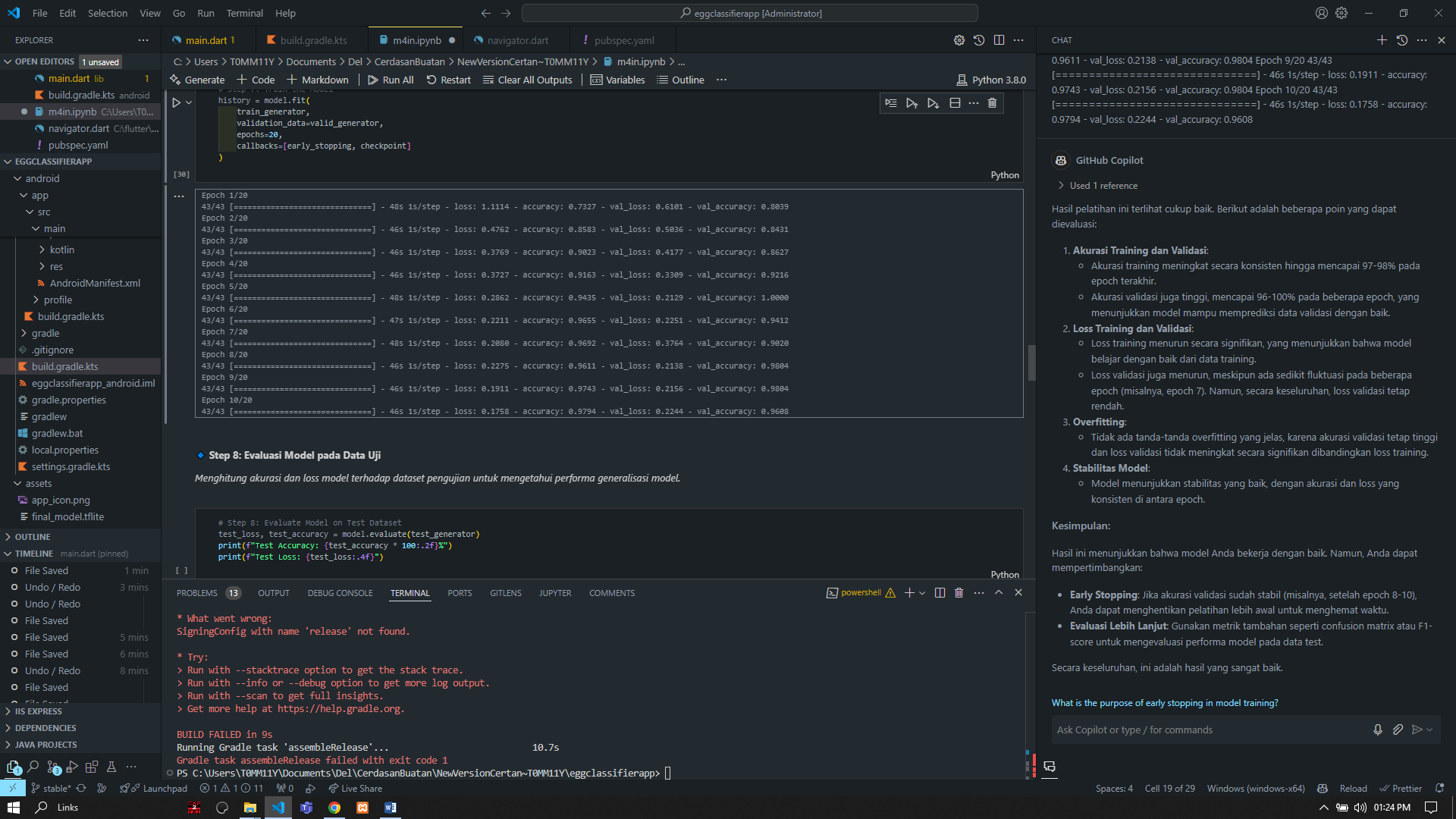
- ModelCheckpoint

5. Epochs: 20

6. Batch size: 32

**4. Hasil dan Evaluasi**

## 4.1Hasil Training



ringkasan hasil pelatihan model:

* **Epoch 1-5**: Akurasi meningkat dari 73% menjadi 94.35%, dengan validasi mencapai 100% pada epoch 5.
* **Epoch 6-10**: Akurasi terus meningkat, mencapai 97.94% di epoch 10, meskipun validasi mengalami fluktuasi.

Model menunjukkan peningkatan akurasi yang signifikan dan penurunan loss sepanjang pelatihan.

Berikut adalah interpretasi dari hasilnya:

* **True Negatives (TN)**: 22 — Model berhasil mengklasifikasikan dengan benar telur yang tidak fertile.
* **False Positives (FP)**: 1 — Model salah mengklasifikasikan telur yang tidak fertile sebagai fertile.
* **False Negatives (FN)**: 1 — Model salah mengklasifikasikan telur yang fertile sebagai tidak fertile.
* **True Positives (TP)**: 26 — Model berhasil mengklasifikasikan dengan benar telur yang fertile.

### 4.2 Evaluasi Kinerja Model:

**1. Akurasi:**

TP + TN  
——————— = (26 + 22) / 50 = 0.96  
TP + TN + FP + FN

Akurasi 96%, menunjukkan kinerja model yang sangat baik.

**2. Precision (Fertile):**

TP  
——— = 26 / 27 ≈ 96.3%  
TP + FP

Model jarang salah dalam memprediksi telur fertile.

**3. Recall (Fertile):**

TP  
——— = 26 / 27 ≈ 96.3%  
TP + FN

Model mampu mendeteksi sebagian besar telur fertile dengan baik.

**4. F1-Score (Fertile):**

≈ 96.3%

Menunjukkan keseimbangan antara precision dan recall.

**Kesimpulan:**  
Model bekerja sangat baik dengan hanya 2 kesalahan dari 50 sampel. Sangat cocok untuk klasifikasi telur fertile dan infertile.

**5. DEPLOYMENT APLIKASI MOBILE**

**Fitur Utama**

* Deteksi dari kamera langsung (30 FPS)
* Analisis gambar dari galeri
* Tampilan confidence score (0-100%)

**Persyaratan Sistem**

| **Platform** | **Minimum Requirement** |
| --- | --- |
| **Android** | API Level 24+ (Android 7.0) |
| **iOS** | iOS 12.0+ |
| **Hardware** | RAM 2GB, GPU dengan dukungan NNAPI |

**Demo Aplikasi**

**Catatan Penting**

* Model harus dikonversi ke format .tflite terlebih dahulu
* Untuk performa terbaik, gunakan perangkat dengan dukungan GPU
* Aplikasi telah diuji pada:
  + Xiaomi Redmi Note 10 (Android 12)
  + iPhone SE (iOS 15)

Lampiran:

1. [Link APK Demo](https://example.com/download)
2. [Source Dataset](https://universe.roboflow.com/eggs-i8jui/egggs3/dataset/1)
3. [Source Code GitHub](https://gist.github.com/T0MM11Y/1365a92d30054d7e88942335d53a5235)
4. [Dokumentasi TFLite Flutter](https://pub.dev/packages/tflite_flutter)