# Pengenalan Teachable Machine

*Teachable Machine* adalah *platform* Google AutoML berbasis web yang merupakan alat untuk membuat model *machine learning* (ML) dengan cepat, mudah, dan dapat diakses oleh semua orang. Kamu dapat membuat model untuk mengklasifikasikan gambar, audio, atau bahkan pose (gerakan). Setelah melakukan pembuatan model, kamu dapat mengunduh model yang dilatih untuk digunakan pada aplikasimu. Website *Teachable Machine* dapat diakses pada url <https://teachablemachine.withgoogle.com/>

Graphical user interface, application

Description automatically generated

Gambar 1. Halaman Utama Website *Teachable Machine*

Halaman Utama pada Gambar 1 berisi informasi mengenai *Teachable Machine* dan cara menggunakannya. Model *Teachable Machine* adalah model TensorFlow.js yang dapat bekerja di mana saja pada javascript yang berjalan. Mereka sangat cocok digabungkan dengan alat seperti Glitch, P5.js, Node.js & lainnya. Selain itu, model dapat diekspor ke beberapa format yang berbeda untuk digunakan ke tempat lain, seperti Coral, Arduino, Android & lainnya. Pada buku ini, kita akan melakukan klasifikasi gambar dengan menggunakan *Teachable Machine* dan implementasinya pada perangkat Android.

# Membuat Proyek Klasifikasi Gambar

Beberapa proyek yang dapat dilakukan pada *Teachable Machine* diakses melalui <https://teachablemachine.withgoogle.com/train> dan dapat dilihat pada Gambar 2.

Graphical user interface, application, website

Description automatically generated

Gambar 2. Halaman Proyek

Graphical user interface, text, application

Description automatically generatedKali ini kita akan melakukan klasifikasi gambar menggunakan *Teachable Machine*. Jika kamu sudah memiliki proyek yang disimpan pada Google Drive atau komputer, kamu dapat membukanya dengan memilih tombol ***Open an existing project from Drive*** pada Gambar 3 atau ***Open an existing project from a file*** pada Gambar 4.

Graphical user interface, text, application, chat or text message

Description automatically generatedGambar 3. Membuka Proyek yang Tersimpan pada Google Drive

Gambar 4. Membuka Proyek yang Tersimpan pada Komputer

Proyek Gambar (***Image Project***) merupakan bagian dari klasifikasi gambar yang mengajarkan mesin dengan gambar melalui berkas foto yang ada dikomputer atau gambar yang ditangkap oleh kamera kamu. Kita dapat membuat klasifikasi gambar dengan beberapa tahapan sebagai berikut.

1. Graphical user interface, text, application

   Description automatically generatedPilih menu *Image Project* seperti pada Gambar 5.

Gambar 5. Menu *Image Project*

1. Graphical user interface, text, application

   Description automatically generatedSetelah menu terpilih, kamu akan melihat sebuah *popup* untuk menentukan pilihan gambar yang akan diklasifikasikan seperti pada Gambar 6.

Gambar 6. *Popup* Pilihan Gambar

Apabila model kita nantinya digunakan untuk diterapkan pada *mobile apps* atau *desktop apps,* kamu dapat memilih ***Standard image model*** di mana ukuran piksel gambar berwarna adalah 224x224 dan ukuran model sekitar 5 megabyte (mb).

Dalam penerapan model pada *microcontrollers* seperti Raspberry, Arduino, dan lain-lain, kamu dapat memilih ***Embedded image model****.* Ukuran piksel gambar yang diproses pada *Embedded image model* berupa gambar *greyscale* dengan ukuran 96x96 dan perkiraan ukuran model sekitar 500kb.

Karena kita akan melakukan penerapan model Teachable Machine pada perangkat Android maka kita pilih ***Standard image model***.

1. Graphical user interface, text

   Description automatically generatedSetelah kita memilih *Standard image model*, kamu akan diarahkan ke url <https://teachablemachine.withgoogle.com/train/image> dengan tampilan yang dapat dilihat pada Gambar 7.

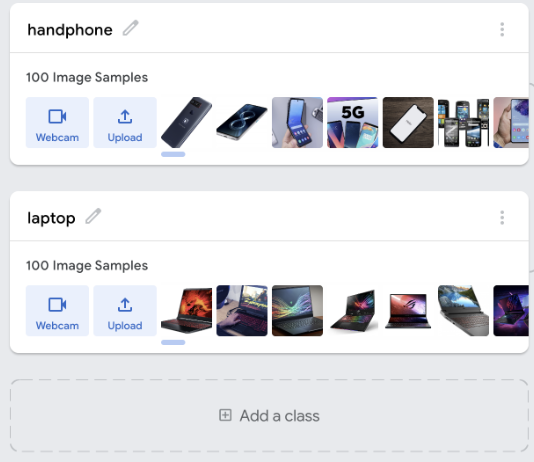
Gambar 7. Menu Utama *Image Project*

Secara *default,* Kelas data yang ditampilkan pada Gambar 7 hanya memiliki 2 kelas. Namun dengan perkembangannya saat ini *Teachable Machine* memberikan *statement* bahwa kita dapat menambahkan kelas data *as many classes as you like* (sebanyak apapun yang kamu mau).

Graphical user interface, application, Teams

Description automatically generatedBuat 2 kelas, yaitu kelas handphone dan laptop dengan masing-masing gambar yang sudah ada. Kemudian pilih *upload*, maka akan tampil pilihan seperti pada Gambar 8.

Gambar 8. Bagian Pemilihan Berkas Gambar

Pada Gambar 8 pilih ***Choose images from your files*** atau kamu bisa drag & drop berkas gambar langsung ke *Teachable Machine.* Jika sudah di-*upload,* pastikan tampilan telah sesuai seperti pada Gambar 9.

Gambar 9. Tampilan Gambar Ter-*upload*

Graphical user interface, application

Description automatically generatedGambar 10. Bagian *Training Image Project*

Setelah seluruh gambar ter-upload dan sesuai, langkah kita selanjutnya melakukan pengaturan parameter training dan test. Sesuaikan pengaturan seperti pada Gambar 10 dan klik tombol ***Train Model***.

Pada bagian *Training* (pelatihan) model, kita dapat memilih *Train model* dengan parameter yang telah diatur otomatis oleh *Teachable Machine* atau dapat melakukan *hyperparameter tunning* pada *dropdown Advanced* seperti pada Gambar 10*.* Kita dapat mengatur *Epochs*, *Batch size*, dan *Learning rate*-nya.

* *Epochs* adalah *hyperparameter* yang menentukan jumlah algoritma pembelajaran bekerja mengolah seluruh *dataset* *training*. Jumlah iterasi yang dilakukan dalam mengolah *dataset* diwakili oleh banyaknya nilai *Epochs.*
* *Batch size* adalah *hyperparameter* dari *gradient descent* yang mengontrol jumlah sampel *training* untuk diproses sebelum parameter internal model diperbarui. Perhitungan *Batch size* dapat dilakukan dengan Perhitungan 2.1 sebagai berikut.

(2.1)

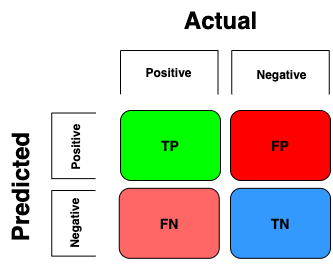
*Gradient descent* adalah algoritma optimasi yang digunakan untuk menemukan nilai-nilai parameter (koefisien) dari suatu fungsi (f) dengan tujuan meminimalkan fungsi *cost*. *Gradient descent* paling baik digunakan ketika parameter tidak dapat dihitung secara analitik (mis. Menggunakan aljabar linier) dan harus dicari dengan algoritma optimalisasi.

* *Learning rate* adalah salah satu parameter *training* untuk menghitung nilai koreksi bobot pada saat waktu proses *training.* Nilai learning rate ini berada pada range nol (0) sampai dengan satu (1). Semakin besar nilai learning rate, maka proses training akan berjalan semakin cepat namun ketelitian jaringan model akan semakin berkurang. Sebaliknya, semakin kecil nilai learning rate-nya, maka ketelitian jaringan model akan semakin tinggi atau bertambah dengan konsekuensi proses training akan memakan waktu yang semakin lama.

1. Setelah kamu mengatur parameter, maka model akan dilakukan evaluasi performa. *Dataset* yang kita masukkan akan dilakukan pembagian sebanyak 85% sebagai *data training* dan 15% sebagai *data test*.

* Data *training*/ *training samples* digunakan untuk melatih model secara benar dalam mengklasifikasikan data ke dalam kelas yang telah dibuat.
* Data test/ test samples tidak digunakan dalam melatih model, jadi setelah model dilatih dengan data training, data tersebut digunakan untuk mengevaluasi performa model pada data baru yang belum diketahui menggunakan *confusion matrix*.

Performa model diukur berdasarkan klasifikasi data *testing* yang dibandingkan dengan *ground truth*-nya. Perbandingan hasil klasifikasi dan *ground truth* data *testing* ditabulasikan ke dalam *confusion matrix* seperti pada Gambar 11.



Gambar 11. *Confusion Matrix*

Dari *confusion matrix* pada Gambar 11 terdapat empat istilah representasi hasil proses klasifikasi, yaitu *true positive, true negative, false positive*, dan *false negative* yang dapat dijelaskan sebagai berikut.

1. *True Positive* (TP) merupakan data positif yang diprediksi benar atau data kelas benar yang diklasifikasi kelas positif.
2. *True negative* (TN) merupakan data negatif yang diprediksi benar atau data kelas salah yang diklasifikasi kelas negatif.
3. *False positive* (FP) atau kesalahan tipe satu, merupakan data negatif yang salah diprediksikan klasifikasinya menjadi benar atau data kelas salah yang diklasifikasi sebagai data kelas positif.
4. *False negative* (FN) atau kesalahan tipe dua, merupakan data kelas benar yang salah diprediksikan klasifikasinya menjadi negatif atau data kelas benar yang diklasifikasi sebagai data kelas salah.

Perhitungan *confusion matrix* terbagi menjadi empat tahapan, yaitu *accuracy, precision, recall,* dan *f1-score* yang dapat dijelaskan sebagai berikut.

1. *Accuracy*

|  |  |
| --- | --- |
|  | (2.2) |

Perhitungan 3.2 digunakan untuk memperhitungkan akurasi klasifikasi dalam mengukur kinerja prediksi data.

1. *Precision*

|  |  |
| --- | --- |
|  | (2.3) |

Selain akurasi, hasil perlu dianalisa lebih lanjut untuk memperkuat ukuran kinerja suatu data menggunakan perhitungan 3.3. *Precision* adalah salah satu teknik untuk mengukur tingkat rasio observasi positif yang diprediksi dengan benar.

1. *Recall*

|  |  |
| --- | --- |
|  | (2.4) |

Tahapan selanjutnya adalah mengukur nilai *recall* menggunakan perhitungan 3.4, di mana rasio observasi positif diprediksi dengan benar terhadap semua observasi di kelas aktual.

1. *F1-score*

|  |  |
| --- | --- |
|  | (2.5) |

Dari semua teknik perhitungan pada *confusion matrix,* pengukuran menggunakan *f1-score* akan menghasilkan nilai yang seimbang antara perhitungan rata-rata dari *precision* dan *recall*.

Chart

Description automatically generatedDari 100 dataset pada tiap kelas (handphone dan laptop) dibagi menjadi data training 85% atau sekitar 85 data untuk setiap kelas dan data test 15% sekitar 15 data untuk setiap kelas. Hasil performa evaluasi *overall accuracy training* dan *test accuracy* dapat dilihat pada Gambar 12 dan Gambar 13.

Gambar 12. Performa Evaluasi *Accuracy* per *epoch*

Pada Gambar 12 menunjukkan performa Evaluasi *overall accuracy training* mendapatkan nilai maksimal, yaitu acc:1 atau sebesar 100%. Sedangkan nilai *test accuracy* memiliki nilai yang tidak jauh berbeda, yaitu test acc:0.933333277702 atau sebesar 93%.

Chart

Description automatically generatedGambar 13. Performa Evaluasi *Loss*

Selain mendapatkan nilai performa evaluasi *overall accuracy training* dan *test accuracy*, kita juga diberikan informasi nilai loss dari masing-masing akurasi pada Gambar 13. *Loss overall accuracy training* didapatkan sebesar 0.0001957 atau sebesar 0.0002% dan *Loss test accuracy* didapatkan sebesar 0.07%.

Chart

Description automatically generatedPerforma pengujian setiap kelas dihitung menggunakan *confusion matrix* yang dapat dilihat pada Gambar 14.

Gambar 14. Performa Evaluasi Pengujian dengan *Confusion Matrix*

Dapat dilihat performa setiap kelas pada Gambar 14 untuk kelas handphone mendapatkan accuracy sebesar 0.87 atau 87% dan accuracy laptop sebesar 1 atau sebesar 100%. Hanya pada kelas handphone terdapat 2 dataset yang mengalami salah klasifikasi.

Graphical user interface, application

Description automatically generated

Gambar 15. *Preview* *Model*

Graphical user interface, application

Description automatically generatedSetelah kamu rasa performa model telah optimal dan sesuai ekspektasi, kamu bisa mencobanya secara langsung dengan mengupload berkas gambar yang ingin diuji seperti pada Gambar 15. Karena kita akan mencoba mengimplementasikan model tersebut pada aplikasi android, maka kita perlu mengunduh model pada menu *Export Model* pada Gambar 16.

Gambar 15. *Export* *Model*

# Implementasi Model pada Aplikasi Android

Table

Description automatically generatedSebelum mengimplementasi model yang sudah kita bangun pada aplikasi android, perlu dipastikan bahwa model telah dimiliki. Model yang akan digunakan adalah tensorflowlite pada folder converted\_tflite seperti pada Gambar 16. Pada model\_unquant.tflite buat duplikasi dengan nama model.tflite yang akan kita gunakan pada implementasi aplikasinya.

Gambar 16. *Export* *Model*

## Melakukan Instalasi Android Studio

Pastikan kamu men-*download* Android Studio versi terbaru pada link website <https://developer.android.com/studio?hl=id>.

Graphical user interface, text, application

Description automatically generated

Gambar 17. Tampilan Utama Website Android Studio

Kamu dapat langsung men-*download* Android Studio yang telah disesuaikan oleh Google berdasarkan perangkat yang kamu gunakan dengan klik tombol ***Download* Android Studio** pada Gambar 17.

Android Studio saat ini dapat berjalan pada 4 *Operating Systems* (OS), yaitu Windows, Mac, Linux, dan Chrome OS. Setiap OS memiliki tahapan instalasi yang berbeda, kamu dapat melihatnya pada link website berikut <https://developer.android.com/studio/install?hl=id>.

## Membuat Proyek Android

Graphical user interface, application, Teams

Description automatically generatedPertama, buka aplikasi Android Studio, maka tampilan utama aplikasi akan menampilkan seperti Gambar 18.

Gambar 18. Tampilan Utama Aplikasi Android Studio

Pada *Projects*, pilih *New Project* maka kamu akan diarahkan ke menu *Templates New Project* seperti Gambar 19.

Graphical user interface, application

Description automatically generatedGambar 19. Menu *Templates New Project*

Karena kita akan membuat aplikasi Android pada perangkat Smartphone, maka pilih *Templates Phone* and *Tablet* dengan *Empty Activity* seperti pada Gambar 19. Kemudian pilih *Next* untuk melanjutkan.

Graphical user interface, text, application

Description automatically generatedGambar 20. Menu *Templates New Project*

Masukkan nama aplikasi yang akan dibuat yaitu MLwithTensorFlowLite dengan bahasa pemrograman yang kita gunakan adalah Java. Setelah itu, kita pilih *Minimum SDK*-nya adalah API 21: Android 5.0 (Lollipop) karena 98% aplikasi ini masih dapat berjalan disegala perangkat dengan OS terendah ini. Apabila sudah sesuai seperti Gambar 20, pilih *Finish* untuk melanjutkan.

## Meng-*import* Tensorflow Lite model

A screenshot of a computer

Description automatically generatedModel perlu di-import untuk memberikan fungsi pada Smartphone Android dalam mengklasifikasikan gambar menggunakan Tensorflow Lite.

Gambar 21. Implementasi TensorFlow Lite Model

Pada direktori proyek **app** klik kanan (Windows/Linux)/ klik dua jari(Mac) kemudian pilih *New* 🡪 *Other* 🡪 TensorFlow Lite Model seperti Gambar 21.

Graphical user interface, text, application

Description automatically generatedGambar 22. Implementasi TensorFlow Lite Model

Pada ***Model location*** Gambar 22 pilih lokasi model yang disimpan dengan nama model.tflite seperti pada Gambar 16. Kemudian, centang pada ***Automatically add build feature and dependencies to build.gradle***. Apabila sudah sesuai, kamu bisa lanjutkan dengan pilih tombol *Finish*.

Text

Description automatically generatedImplementasi model dapat dikatakan telah sesuai apabila proses telah selesai dan menunjukkan tampilan kode penggunaan model.tflite seperti pada Gambar 23.

Gambar 23. Kode Implementasi Tensorflow Lite Model

## Mendesain *Layout User Interface* (UI)

Graphical user interface

Description automatically generatedKamu akan membuat UI sederhana untuk menangkap gambar dan mengklasifikasikannya dengan 2 keluaran (*output*), yaitu *Classified as* yang merupakan hasil klasifikasi dan *Confidences* sebagai persentase akurasinya. Hasil akhir tampilan UI dapat dilihat pada Gambar 24.

Gambar 24. Kode Implementasi Tensorflow Lite Model

Dalam mendesain *layout* UI, kode program berada pada activity\_main.xml dalam direktori res 🡪 layout 🡪 activity\_main.xml.

* Relative Layout

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
  
   
</RelativeLayout>

* Button

<Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerInParent="true"  
 android:id="@+id/button"  
 android:text="TakePicture"  
 android:textAllCaps="false"  
 android:layout\_alignParentBottom="true"  
 android:textSize="21sp"  
 android:textStyle="bold"  
 />

* ImageView

<ImageView  
 android:layout\_width="370sp"  
 android:layout\_height="370sp"  
 android:layout\_centerHorizontal="true"  
 android:id="@+id/imageView"  
 android:layout\_marginTop="10sp"  
 />

* TextView

<TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerHorizontal="true"  
 android:text="Classifiedas:"  
 android:textStyle="bold"  
 android:textSize="20sp"  
 android:id="@+id/classified"  
 android:layout\_below="@+id/imageView"  
 android:layout\_marginTop="10sp"  
 />

<TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerHorizontal="true"  
 android:text=""  
 android:textColor="#C30000"  
 android:textStyle="bold"  
 android:textSize="27sp"  
 android:id="@+id/result"  
 android:layout\_below="@+id/classified"  
 />

<TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/confidencesText"  
 android:text="Confidences:"  
 android:textStyle="bold"  
 android:textSize="20sp"  
 android:layout\_centerHorizontal="true"  
 android:layout\_below="@+id/result"  
 android:layout\_marginTop="30sp"  
 />

<TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerHorizontal="true"  
 android:text=""  
 android:textColor="#000"  
 android:textSize="22sp"  
 android:id="@+id/confidence"  
 android:layout\_below="@+id/confidencesText"  
 />

Sehingga keseluruhan kode secara lengkap pada berkas activity\_main.xml adalah sebagai berikut:

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
  
 <Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerInParent="true"  
 android:id="@+id/button"  
 android:text="Take Picture"  
 android:textAllCaps="false"  
 android:layout\_alignParentBottom="true"  
 android:textSize="21sp"  
 android:textStyle="bold"  
 />

<ImageView  
 android:layout\_width="370sp"  
 android:layout\_height="370sp"  
 android:layout\_centerHorizontal="true"  
 android:id="@+id/imageView"  
 android:layout\_marginTop="10sp"  
 />

<TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerHorizontal="true"  
 android:text="Classified as:"  
 android:textStyle="bold"  
 android:textSize="20sp"  
 android:id="@+id/classified"  
 android:layout\_below="@+id/imageView"  
 android:layout\_marginTop="10sp"  
 />

<TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerHorizontal="true"  
 android:text=""  
 android:textColor="#C30000"  
 android:textStyle="bold"  
 android:textSize="27sp"  
 android:id="@+id/result"  
 android:layout\_below="@+id/classified"  
 />

<TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/confidencesText"  
 android:text="Confidences:"  
 android:textStyle="bold"  
 android:textSize="20sp"  
 android:layout\_centerHorizontal="true"  
 android:layout\_below="@+id/result"  
 android:layout\_marginTop="30sp"  
 />

<TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerHorizontal="true"  
 android:text=""  
 android:textColor="#000"  
 android:textSize="22sp"  
 android:id="@+id/confidence"  
 android:layout\_below="@+id/confidencesText"  
 />  
  
</RelativeLayout>

## Mengkonfigurasi Manifest Android

Fungsi yang akan digunakan pada aplikasi adalah mengaktifkan kamera. Kamera digunakan dalam menangkap gambar yang kemudian akan diklasifikasikan dengan model yang tertanam pada *smartphone.* Konfigurasi akses kamera dapat diterapkan pada file AndroidManifest.xml di folder manifests dengan kode sebagai berikut.

<uses-permission android:name="android.permission.CAMERA"/>

Sehingga keseluruhan kode secara lengkap pada berkas AndroidManifest.xml adalah sebagai berikut:

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.mlwithtensorflowlite">  
  
 <uses-permission android:name="android.permission.CAMERA"/>  
   
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.MLwithTensorFlowLite">  
 <activity  
 android:name=".MainActivity"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

## Memprogram Kelas Java Android

* *Import Library*

import androidx.annotation.Nullable;  
import androidx.annotation.RequiresApi;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.Manifest;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.graphics.Bitmap;  
import android.media.ThumbnailUtils;  
import android.os.Build;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.view.View;  
import android.widget.Button;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import com.example.mlwithtensorflowlite.ml.Model;  
  
import org.tensorflow.lite.DataType;  
import org.tensorflow.lite.support.tensorbuffer.TensorBuffer;  
  
import java.io.IOException;  
import java.nio.ByteBuffer;  
import java.nio.ByteOrder;

* *Class* MainActivity

public class MainActivity extends AppCompatActivity {  
  
 }

* Membuat Variabel

TextView result, confidence;  
ImageView imageView;  
Button picture;  
int imageSize = 224;

* Fungsi onCreate

@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 result = findViewById(R.id.*result*);  
 confidence = findViewById(R.id.*confidence*);  
 imageView = findViewById(R.id.*imageView*);  
 picture = findViewById(R.id.*button*);  
  
 picture.setOnClickListener(new View.OnClickListener() {  
 @RequiresApi(api = Build.VERSION\_CODES.*M*)  
 @Override  
 public void onClick(View view) {  
 // Launch camera if we have permission  
 if (checkSelfPermission(Manifest.permission.*CAMERA*) == PackageManager.*PERMISSION\_GRANTED*) {  
 Intent cameraIntent = new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  
 startActivityForResult(cameraIntent, 1);  
 } else {  
 //Request camera permission if we don't have it.  
 requestPermissions(new String[]{Manifest.permission.*CAMERA*}, 100);  
 }  
 }  
 });  
}

* Fungsi classifyImage

public void classifyImage(Bitmap image){  
 try {  
 Model model = Model.*newInstance*(getApplicationContext());  
  
 // Creates inputs for reference.  
 TensorBuffer inputFeature0 = TensorBuffer.*createFixedSize*(new int[]{1, 224, 224, 3}, DataType.*FLOAT32*);  
 ByteBuffer byteBuffer = ByteBuffer.*allocateDirect*(4 \* imageSize \* imageSize \* 3);  
 byteBuffer.order(ByteOrder.*nativeOrder*());  
  
 int [] intValues = new int[imageSize \* imageSize];  
 image.getPixels(intValues, 0, image.getWidth(), 0, 0, image.getWidth(), image.getHeight());  
 int pixel = 0;  
 for(int i = 0; i < imageSize; i++){  
 for(int j = 0; j < imageSize; j++){  
 int val = intValues[pixel++]; // RGB  
 byteBuffer.putFloat(((val >> 16) & 0xFF) \* (1.f / 255.f));  
 byteBuffer.putFloat(((val >> 8) & 0xFF) \* (1.f / 255.f));  
 byteBuffer.putFloat((val & 0xFF) \* (1.f / 255.f));  
 }  
 }  
  
 inputFeature0.loadBuffer(byteBuffer);  
  
 // Runs model inference and gets result.  
 Model.Outputs outputs = model.process(inputFeature0);  
 TensorBuffer outputFeature0 = outputs.getOutputFeature0AsTensorBuffer();  
  
 float[] confidences = outputFeature0.getFloatArray();  
 int maxPos = 0;  
 float maxConfidence = 0;  
 for(int i = 0; i < confidences.length; i++){  
 if(confidences[i] > maxConfidence){  
 maxConfidence = confidences[i];  
 maxPos = i;  
 }  
 }  
 String[] classes = {"handphone", "laptop"};  
  
 result.setText(classes[maxPos]);  
  
 String s = "";  
 for(int i = 0; i < classes.length; i++){  
 s += String.*format*("%s: %.1f%%\n", classes[i], confidences[i] \* 100);  
 }  
 confidence.setText(s);  
  
 // Releases model resources if no longer used.  
 model.close();  
 } catch (IOException e) {  
 // *TODO Handle the exception* }  
  
}

* Fungsi onActivityResult

@Override  
public void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  
 if (requestCode == 1 && resultCode == *RESULT\_OK*) {  
 Bitmap image = (Bitmap) data.getExtras().get("data");  
 int dimension = Math.*min*(image.getWidth(), image.getHeight());  
 image = ThumbnailUtils.*extractThumbnail*(image, dimension, dimension);  
 imageView.setImageBitmap(image);  
  
 image = Bitmap.*createScaledBitmap*(image, imageSize, imageSize, false);  
 classifyImage(image);  
 }  
 super.onActivityResult(requestCode, resultCode, data);  
}

Sehingga keseluruhan kode secara lengkap pada berkas MainActivity.java adalah sebagai berikut:

package com.example.mlwithtensorflowlite;  
  
import androidx.annotation.Nullable;  
import androidx.annotation.RequiresApi;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.Manifest;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.graphics.Bitmap;  
import android.media.ThumbnailUtils;  
import android.os.Build;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.view.View;  
import android.widget.Button;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import com.example.mlwithtensorflowlite.ml.Model;  
  
import org.tensorflow.lite.DataType;  
import org.tensorflow.lite.support.tensorbuffer.TensorBuffer;  
  
import java.io.IOException;  
import java.nio.ByteBuffer;  
import java.nio.ByteOrder;  
  
public class MainActivity extends AppCompatActivity {  
  
 TextView result, confidence;  
 ImageView imageView;  
 Button picture;  
 int imageSize = 224;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 result = findViewById(R.id.*result*);  
 confidence = findViewById(R.id.*confidence*);  
 imageView = findViewById(R.id.*imageView*);  
 picture = findViewById(R.id.*button*);  
  
 picture.setOnClickListener(new View.OnClickListener() {  
 @RequiresApi(api = Build.VERSION\_CODES.*M*)  
 @Override  
 public void onClick(View view) {  
 // Launch camera if we have permission  
 if (checkSelfPermission(Manifest.permission.*CAMERA*) == PackageManager.*PERMISSION\_GRANTED*) {  
 Intent cameraIntent = new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  
 startActivityForResult(cameraIntent, 1);  
 } else {  
 //Request camera permission if we don't have it.  
 requestPermissions(new String[]{Manifest.permission.*CAMERA*}, 100);  
 }  
 }  
 });  
 }  
  
 public void classifyImage(Bitmap image){  
 try {  
 Model model = Model.*newInstance*(getApplicationContext());  
  
 // Creates inputs for reference.  
 TensorBuffer inputFeature0 = TensorBuffer.*createFixedSize*(new int[]{1, 224, 224, 3}, DataType.*FLOAT32*);  
 ByteBuffer byteBuffer = ByteBuffer.*allocateDirect*(4 \* imageSize \* imageSize \* 3);  
 byteBuffer.order(ByteOrder.*nativeOrder*());  
  
 int [] intValues = new int[imageSize \* imageSize];  
 image.getPixels(intValues, 0, image.getWidth(), 0, 0, image.getWidth(), image.getHeight());  
 int pixel = 0;  
 for(int i = 0; i < imageSize; i++){  
 for(int j = 0; j < imageSize; j++){  
 int val = intValues[pixel++]; // RGB  
 byteBuffer.putFloat(((val >> 16) & 0xFF) \* (1.f / 255.f));  
 byteBuffer.putFloat(((val >> 8) & 0xFF) \* (1.f / 255.f));  
 byteBuffer.putFloat((val & 0xFF) \* (1.f / 255.f));  
 }  
 }  
  
 inputFeature0.loadBuffer(byteBuffer);  
  
 // Runs model inference and gets result.  
 Model.Outputs outputs = model.process(inputFeature0);  
 TensorBuffer outputFeature0 = outputs.getOutputFeature0AsTensorBuffer();  
  
 float[] confidences = outputFeature0.getFloatArray();  
 int maxPos = 0;  
 float maxConfidence = 0;  
 for(int i = 0; i < confidences.length; i++){  
 if(confidences[i] > maxConfidence){  
 maxConfidence = confidences[i];  
 maxPos = i;  
 }  
 }  
 String[] classes = {"handphone", "laptop"};  
  
 result.setText(classes[maxPos]);  
  
 String s = "";  
 for(int i = 0; i < classes.length; i++){  
 s += String.*format*("%s: %.1f%%\n", classes[i], confidences[i] \* 100);  
 }  
 confidence.setText(s);  
  
 // Releases model resources if no longer used.  
 model.close();  
 } catch (IOException e) {  
 // *TODO Handle the exception* }  
  
 }  
  
 @Override  
 public void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  
 if (requestCode == 1 && resultCode == *RESULT\_OK*) {  
 Bitmap image = (Bitmap) data.getExtras().get("data");  
 int dimension = Math.*min*(image.getWidth(), image.getHeight());  
 image = ThumbnailUtils.*extractThumbnail*(image, dimension, dimension);  
 imageView.setImageBitmap(image);  
  
 image = Bitmap.*createScaledBitmap*(image, imageSize, imageSize, false);  
 classifyImage(image);  
 }  
 super.onActivityResult(requestCode, resultCode, data);  
 }  
}