Android 애플리케이션 프로그래밍 – 이미지 분류 앱 개발

텐서플로 라이트 모델을 이용한 이미지 분류 APP 개발

부산대학교 정보의생명공학대학
정보컴퓨터공학부

이론

강의 목표와 구성

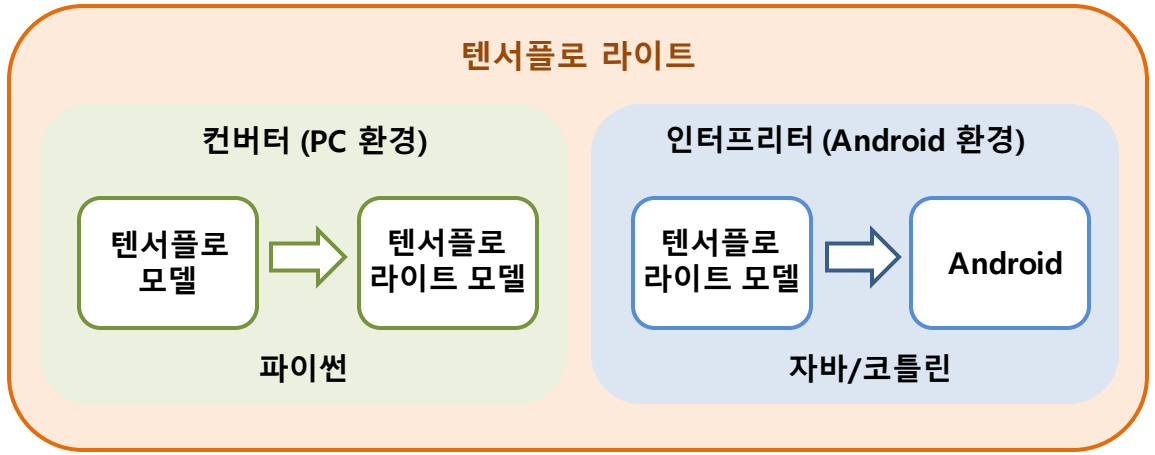
❖ 텐서플로 라이트 모델을 활용한 이미지 분류 앱 개발

- 텐서플로 라이트 개요
- TFLite 모델 로드
- Draw Activity 기반의 입력 데이터 생성 앱 개발
- 입력 이미지 전처리
- 추론 및 결과 해석

텐서플로 라이트 개요

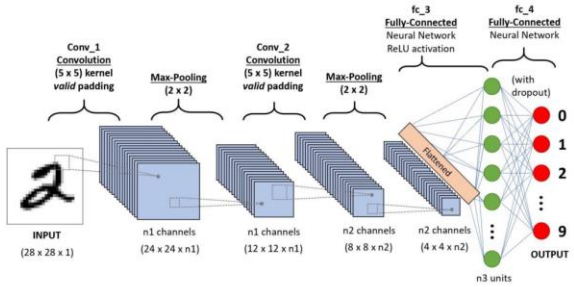
❖ 텐서플로 라이트(Tensorflow Lite)란?

- **모바일과 IoT 기기**에 딥러닝 모델을 배포하고 사용하기 위한 라이브러리
- **컨버터(Convertor)**와 **인터프리터(Interpreter)**로 구성

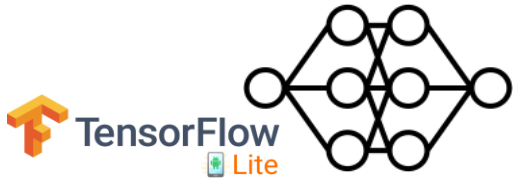


구분	텐서플로	텐서플로 라이트
개발언어	파이썬	파이썬/자바/코틀린
목적	딥러닝 모델 개발	딥러닝 모델 변환
연산자	텐서플로 모든 연산	텐서플로 연산 일부

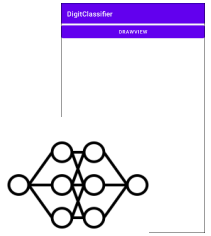
❖ 텐서플로 라이트 기반의 개발 Android 앱 프로세스



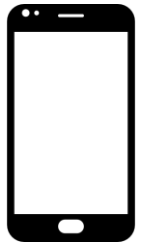
분류 모델 설계 및 학습



모델 변환 (텐서플로 모델 -> 텐서플로 라이트 모델)



텐서플로 라이트 연동 앱 개발

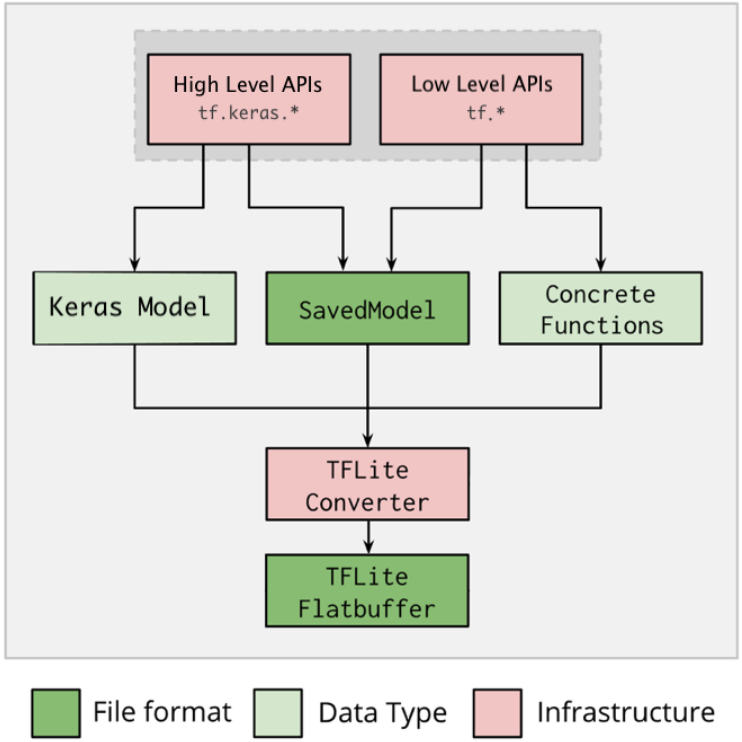


기기 배포

텐서플로 라이트 모델 변환

❖ 라이브러리를 통한 다양한 변환 지원

- 케라스 모델 (고수준 라이브러리)
- Saved 모델 (학습 결과 저장 모델)
- Concrete 함수 (저수준의 사용자 정의 함수)



❖ 케라스 모델 변환

```
mlp_model = tf.keras.models.Sequential([    # 케라스 모델 생성 함수
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dense(10, activation='softmax')
])
```

케라스 모델 구조

```
converter = tf.lite.TFLiteConverter.from_keras_model(mlp_model)
tflite_model = converter.convert() # 텐서플로 라이트 변환

with open('./keras_model.tflite', 'wb') as f:
  f.write(tflite_model) # 텐서플로 라이트 모델로 저장
```

❖ Saved 모델 변환

```
mlp_model.save("./mlp_model/") # 모델을 Saved 모델로 저장
```

```
converter = tf.lite.TFLiteConverter.from_saved_model("./mlp_model/")
tflite_model = converter.convert()

with open('./saved_model.tflite', 'wb') as f:
  f.write(tflite_model) # 텐서플로 라이트 모델로 저장
```

텐서플로 라이트 모델 변환

❖ Concrete 함수 변환

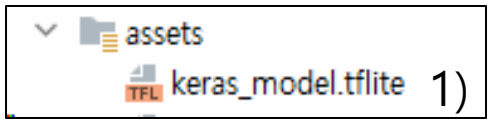
```
class Inc_Graph(tf.keras.layers.Layer):
    @tf.function # 함수 그래프 모드
    def call(self, inputs): # 사용자 정의 함수 (Input에 1을 더함)
        return inputs + 1
inc_g = Inc_Graph() # 인스턴스 생성

concrete_func = inc_g.call.get_concrete_function(tf.TensorSpec(shape=(1, 3), dtype=tf.float32))  # Concrete 함수 변환
```

```
print(concrete_fun(tf.constant([[1.0, 2.0, 3.0]]))) # tf.Tensor([[2. 3. 4.]], shape=(1, 3), dtype=float32) 출력
converter = tf.lite.TFLiteConverter.from_concrete_functions([concrete_func])
tflite_model = converter.convert() # 사용자 정의 모델을 텐서플로 라이트로 변환

with open('./concrete_func_model.tflite', 'wb') as f:
  f.write(tflite_model)  # 텐서플로 라이트로 저장
```

텐서플로 라이트 모델 로드

❖ 텐서플로 라이트 모델 로드

- **바이트 버퍼(ByteBuffer)와 인터프리터(Interpreter) 클래스 활용**
  1) Tflite 파일을 추가(프로젝트의 assets 폴더에 추가)
  2) Classifier 생성
  3) ByteBuffer 객체를 통한 tflite 파일로드
  4) Interpreter 객체 생성



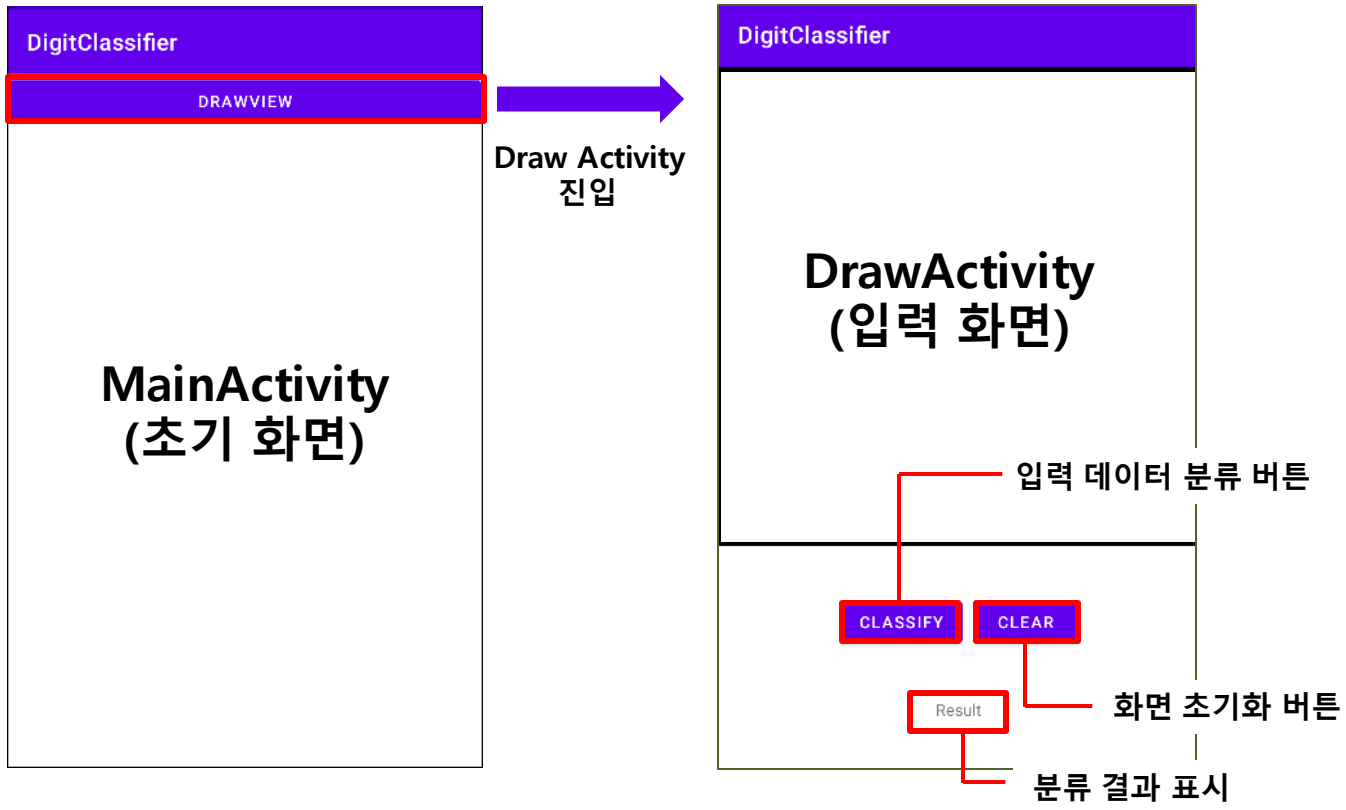

```
public class Classifier{
    private static final String MODEL_NAME = "keras_model.tflite";

    Context context;

    public Classifier(Context context) {
        this.context = context;
    }
}
```
2)

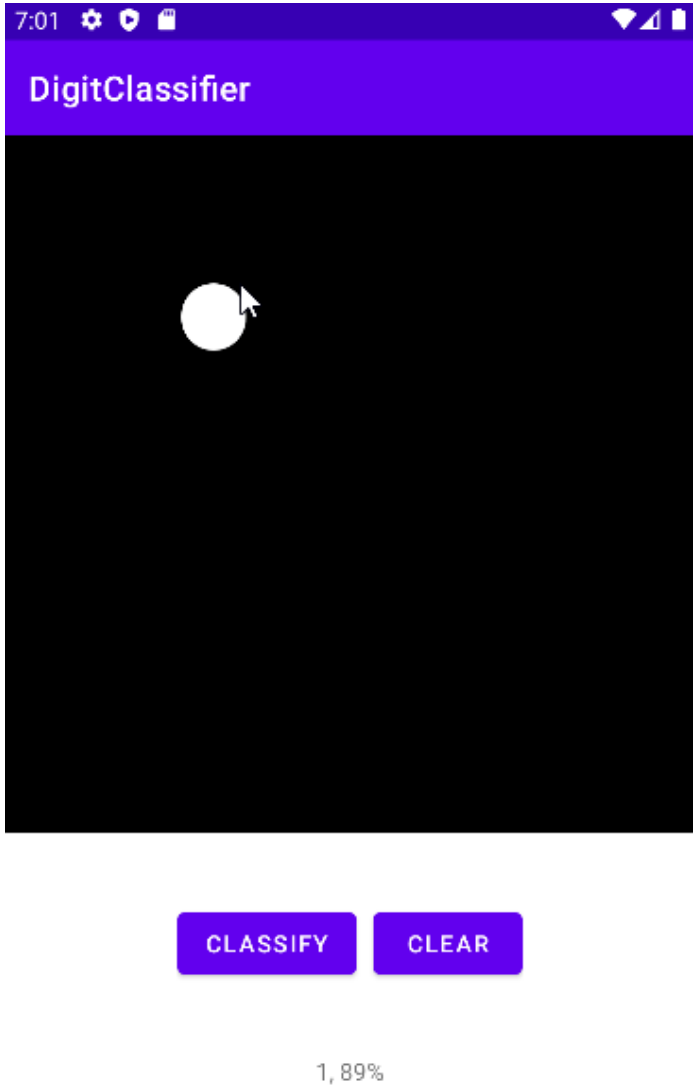
```
public void init() throws IOException{
    ByteBuffer model = loadModelFile(MODEL_NAME);
    model.order(ByteOrder.nativeOrder());
    interpreter = new Interpreter(model);

    initModelShape();
}
```
3, 4)

텐서플로 라이트 연동 앱 개발

❖ 텐서플로 라이트 연동 어플리케이션 구현

- 텐서플로 라이트 모델(MNIST 기반 숫자 분류)을 활용 가능한 앱 구현
- DrawActivity 를 활용한 숫자 이미지 데이터 생성 기능 구현
- 딥러닝 모델과 연동을 통한 입력 이미지 분류 기능 구현





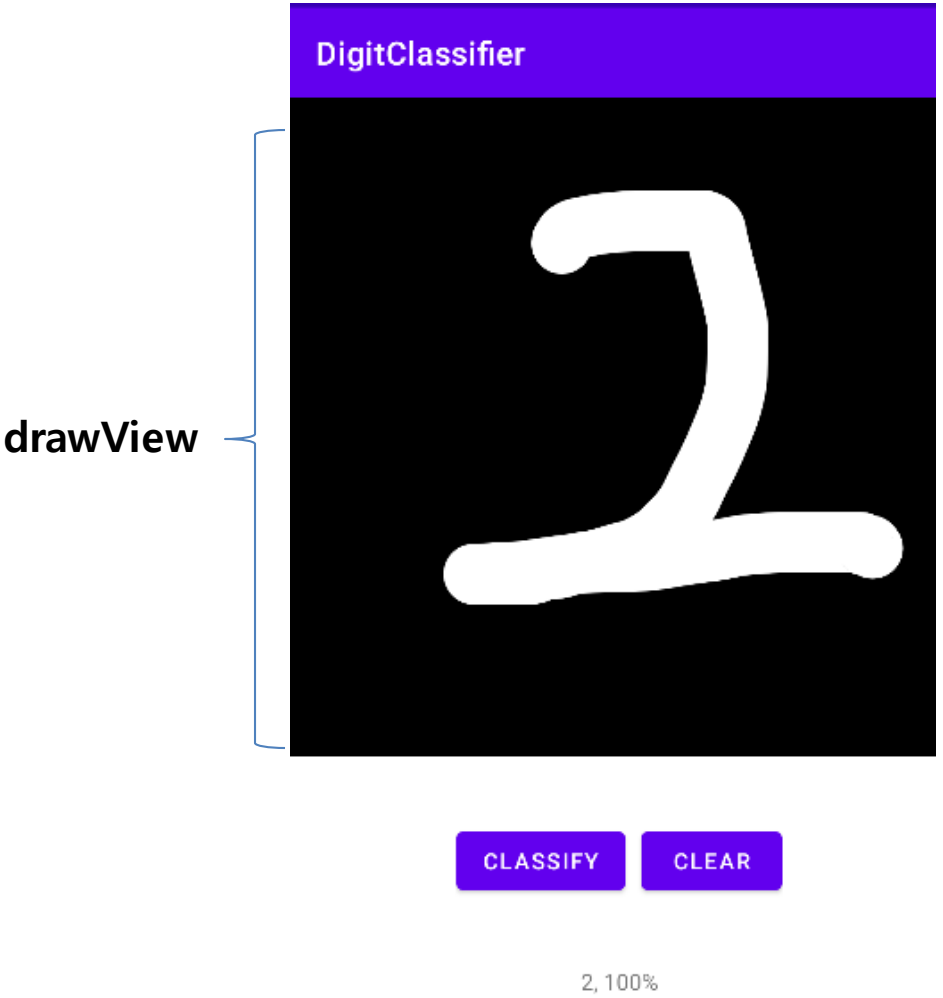
텐서플로 라이트 연동 앱 개발

❖ DrawActivity란?

- Android의 **터치스크린에서 손글씨를 작성할 수 있는 Activity**
- 손글씨를 이미지 형태로 저장 가능 => **딥러닝 모델 검증**에 활용 가능

❖ DrawActivity와 텐서플로 라이트를 통한 숫자 이미지 분류 과정

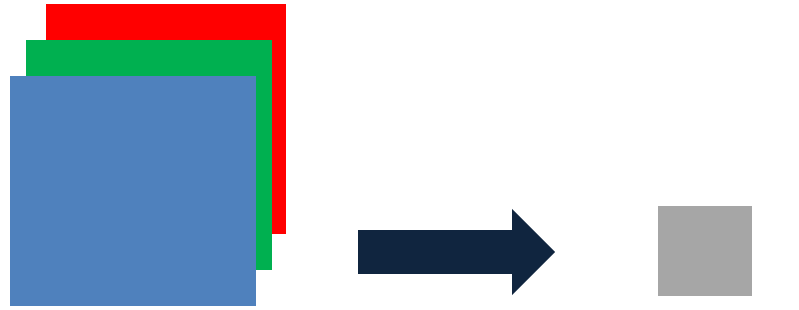
1) DrawActivity를 통해, drawView에 원하는 손글씨를 작성
2) CLASSIFY 버튼을 클릭 시, Draw Activity의 drawView에서 Bitmap 이미지를 가져옴
3) drawView의 이미지를 딥러닝 입력에 맞게 변환(전처리)
4) 인터프리터를 통한 텐서플로 라이트 모델을 실행 (입력 : 전처리 이미지)
5) 모델의 추론 결과를 해석
6) CLEAR 버튼 클릭 시, DrawActivity를 초기화하고 1)부터 반복



입력 이미지 전처리

❖ DrawActivity 이미지 형태

- DrawActivity에서 얻은 이미지는 Bitmap 형식의 RGB 이미지
- DrawActivity에서 얻은 이미지는 기기의 화면 크기에 따라 달라짐
- FHD+ 기기의 경우 DrawActivity에서 얻어지는 이미지 크기는 1080 x 1080 x 3

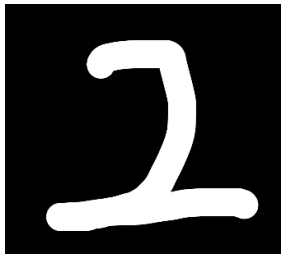


Draw Activity 이미지
(1080 x 1080 x 3)

입력 이미지
(28 x 28 x 1)

❖ 딥러닝 모델의 입력 이미지 형태

- 학습에 사용한 MNIST 데이터 셋은 28 x 28 x 1
- 딥러닝 모델의 입력 이미지의 크기는 학습에 사용한 이미지의 크기와 동일
- 즉, 1080 x 1080 x 3의 크기의 이미지는 28 x 28 x 1 의 크기로 변환이 요구됨



DrawActivity 이미지
(1080 x 1080 x 3)

입력 이미지 크기 변환

1080 x 1080 크기의 이미지를 28 x 28 크기의 이미지로 변환

입력 이미지 채널 변환

RGB 채널 이미지(28 x 28 x 3)를 Gray Scale 이미지(28 x 28 x 1)로 변환

입력 이미지 포맷 변환

Bitmap 포맷 이미지를 텐서플로 라이트의 입력 포맷인 ByteBuffer로 변환

추론 및 결과 해석

❖ 추론

- 변환된 이미지를 추론하기 위하여 Interpreter를 활용 가능
- Interpreter의 run(input, result)함수를 통해 입력 데이터의 결과를 확인 가능
- 손글씨 분류 모델(MNIST 데이터셋)의 경우, 0~9의 숫자를 10개의 클래스로 분류하기 때문에 출력 데이터에는 10개의 출력 결과(클래스)가 존재

```
public void classify(Bitmap image) {
    …

    float[][] result = new float[1][modelOutputClasses];

    interpreter.run(input, result);


    …
}
```

❖ 추론 결과 해석

- 추론 결과는 분류 가능한 클래스의 개수만큼 전달됨 (0~9)
- 결과 값 배열에는 모델이 각 클래스(0~9)에 대해 추론한 확률이 실수 형태로 존재
- 가장 확률이 높은 클래스의 인덱스 및 값을 찾아내어 모델의 추론 결과 도출
  - 결과 값 배열 Argument의 최대 값 및 인덱스를 찾아서 반환

```
private Pair<Integer, Float> argmax(float[] array) {
    int argmax = 0;
    float max = array[0];
    for(int i = 1; i < array.length; i++) {
        float f = array[i];
        if(f > max) {
            argmax = i;
            max = f;
        }
    }
    return new Pair<>(argmax, max);
}
```

실습

실습 목표와 구성

1. **기초(따라하기) – 예제 1**
   - DrawActivity를 활용한 숫자 이미지 분류 앱 구현
2. **응용(로직구현) – 예제 2**
   - 디바이스 이미지 분류 앱 구현
3. **심화(과제물) – 예제 3**
   - FashionMnist 이미지 분류 앱 구현

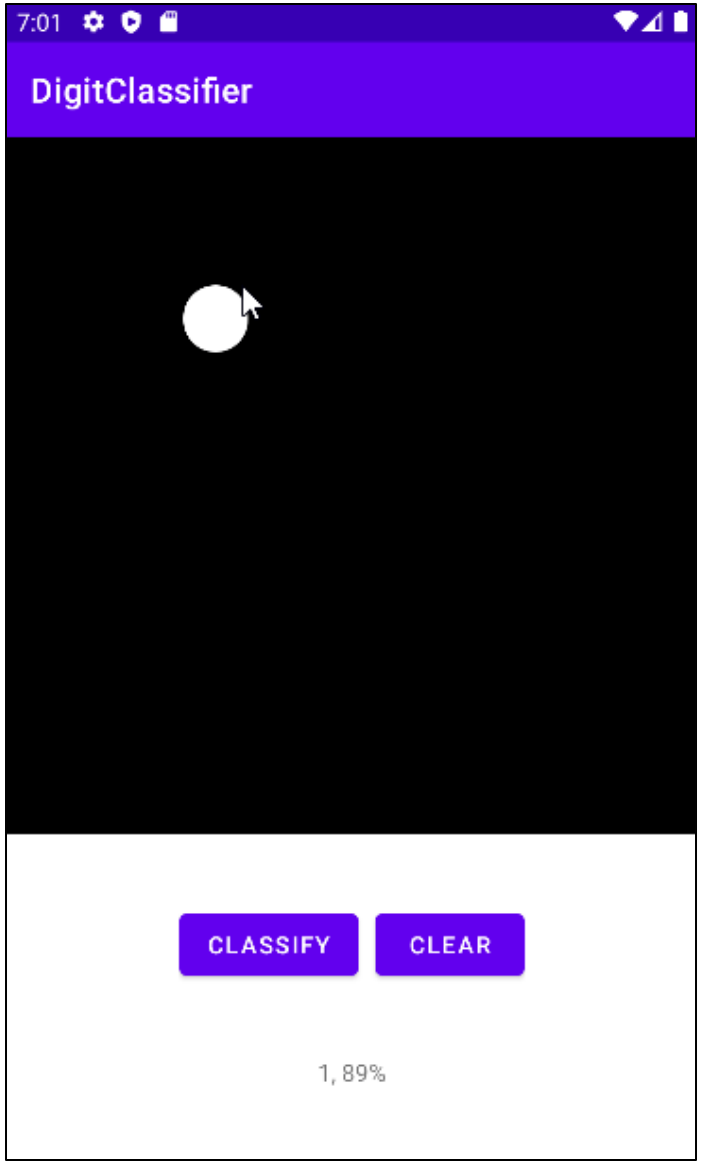
기초(따라하기) – 예제 1

❖ DrawActivity 기반 분류 앱 구현

1. 모델 학습 및 변환
2. 프로젝트 생성
3. 레포지터리와 의존성 추가
4. 텐서플로 라이트 모델 로드
5. 입력 이미지 전처리 코드 구현
6. 추론
7. DrawActivity 레이아웃 구성
8. MainActivity 레이아웃 구성
9. UI 로직 구현
10. DrawActivity 코드 구현
11. 텐서플로 라이트 모델 로드

AndroidManifest.xml

```
12        <activity android:name=".draw.DrawActivity"></activity>
13        <activity
14            android:name=".MainActivity"
```



기초(따라하기) – 예제 1

1. 모델 학습 및 변환

- **Step 1. 데이터 및 모델 구성하기**

In [1] :

```
import tensorflow as tf

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train, x_test = x_train / 255.0, x_test / 255.0

x_train_4d = x_train.reshape(-1, 28, 28, 1)
x_test_4d = x_test.reshape(-1, 28, 28, 1)

cnn_model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')])
```

기초(따라하기) – 예제 1

1. 모델 학습 및 변환

- **Step 2. 모델 학습하기**

```
In [2] :    cnn_model.compile(optimizer='adam',
                              loss='sparse_categorical_crossentropy',
                              metrics=['accuracy'])
            cnn_model.fit(x_train_4d, y_train, epochs=5)
```

```
Out [2] :   Epoch 1/5 1875/1875 [==============================] - 13s 3ms/step - loss: 0.1502 - accuracy: 0.9532
            Epoch 2/5 1875/1875 [==============================] - 6s 3ms/step - loss: 0.0486 - accuracy: 0.9849
            Epoch 3/5 1875/1875 [==============================] - 6s 3ms/step - loss: 0.0340 - accuracy: 0.9893
            Epoch 4/5 1875/1875 [==============================] - 6s 3ms/step - loss: 0.0265 - accuracy: 0.9917
            Epoch 5/5 1875/1875 [==============================] - 6s 3ms/step - loss: 0.0213 - accuracy: 0.9932
            Out[4]:
            <keras.callbacks.History at 0x1ef8452beb0>
```

- **Step 3. 모델 변환 및 저장하기**

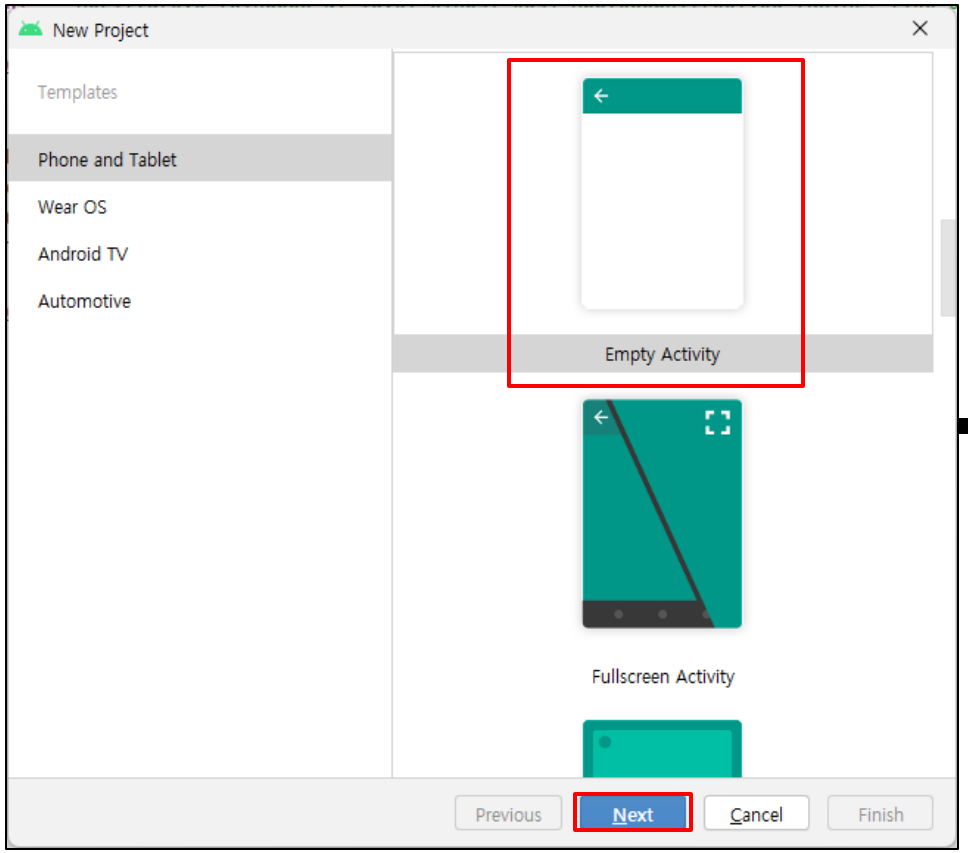
```
In [3] :    converter = tf.lite.TFLiteConverter.from_keras_model(cnn_model)
            tflite_model = converter.convert()

            with open('./keras_model_cnn.tflite', 'wb') as f:
                f.write(tflite_model)
```

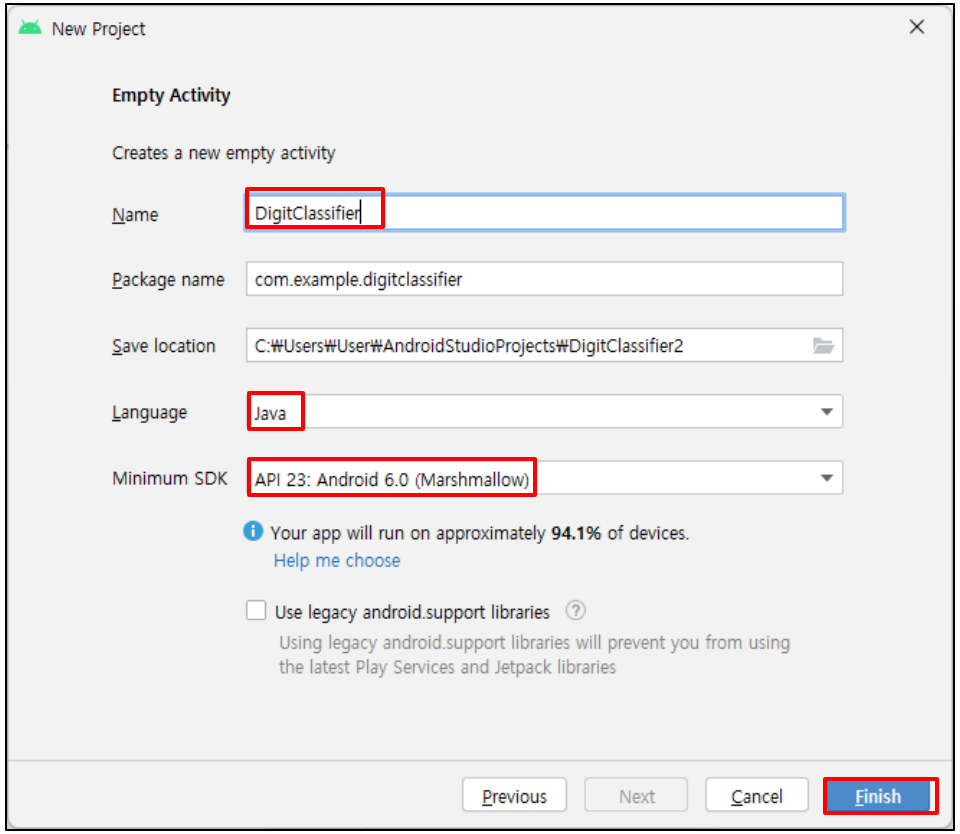
```
Out [3] :   INFO:tensorflow:Assets written to: C:₩Users₩User₩AppData₩Local₩Temp₩tmp9ufmh1r0₩assets
            WARNING:absl:Buffer deduplication procedure will be skipped when flatbuffer library is not properly loaded
```

기초(따라하기) – 예제 1

2. 프로젝트 생성

- **액티비티** : Empty Activity
- **프로젝트 명** : DigitClassifier
- **프로젝트 언어** : JAVA
- **최소 SDK** : API 23





기초(따라하기) – 예제 1

3. 레포지터리와 의존성 추가

- AndroidDraw 라이브러리 사용을 위해, Jitpack 레포지터리 추가

build.gradle (Project: DigitClassifier)

```
15  allprojects {
16      repositories {
17          google()
18          mavenCentral()
19          maven { url 'https://jitpack.io' }
20      }
21  }
22
23  task clean(type: Delete) {
24      delete rootProject.buildDir
25  }
```

- AndroidView 의존성 추가
- 텐서플로 라이트 라이브러리 의존성 추가

build.gradle (Module: DigitClassifier.app)

```
35  dependencies {
36      implementation 'com.github.divyanshub024:AndroidDraw:v0.1'
37      implementation 'org.tensorflow:tensorflow-lite:2.7.0'
38      implementation 'org.tensorflow:tensorflow-lite-support:0.3.0'
39      implementation 'androidx.appcompat:appcompat:1.2.0'
```

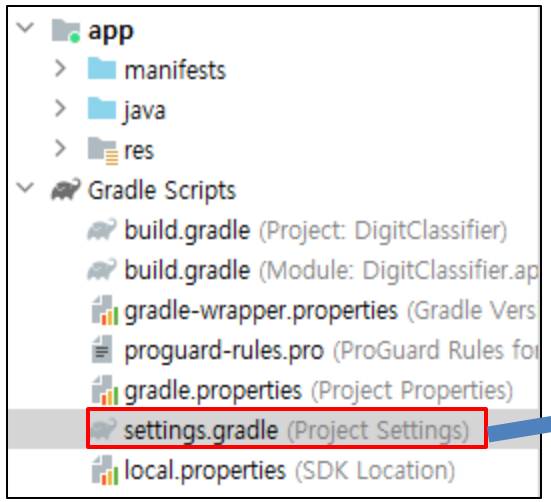
- Sync Project with Gradle Files 를 통한 Gradle 설정 동기화

Sync Project with Gradle Files

기초(따라하기) – 예제 1

3. 레포지터리와 의존성 추가

- 에러 발생시 setting.gradle 의 dependencyResolutionManagement 블록 제거
- Sync Project with Gradle Files 재 수행



Open

settings.gradle (Project Settings)

```
1  dependencyResolutionManagement { DependencyResolutionManagement it ->
2      repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS
3      repositories { RepositoryHandler it ->
4          google()
5          mavenCentral()
6          jcenter() // Warning: this repository is going to shut
7      }
8  }
9  rootProject.name = "DigitClassifier"
10 include ':app'
11
```

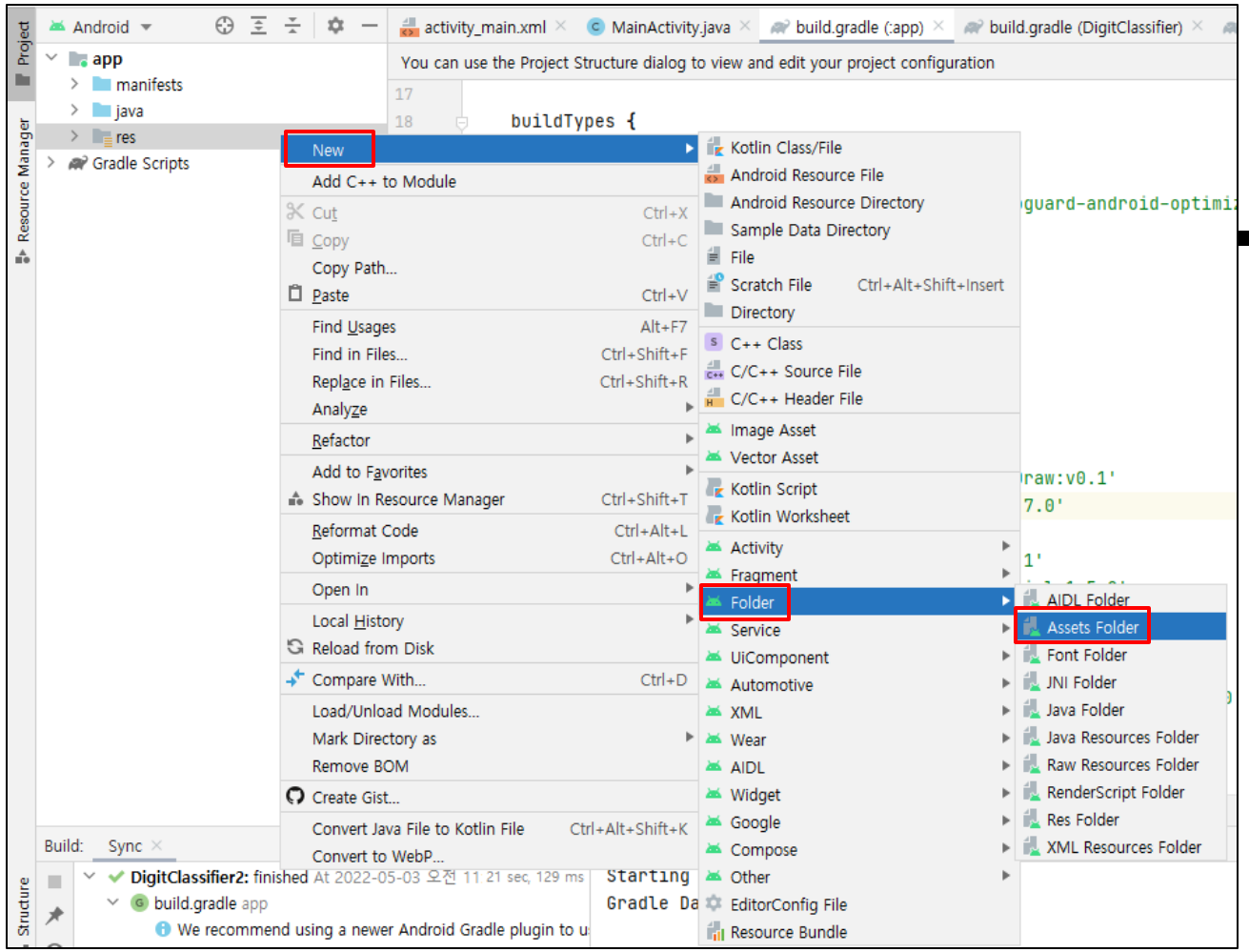
블록 제거

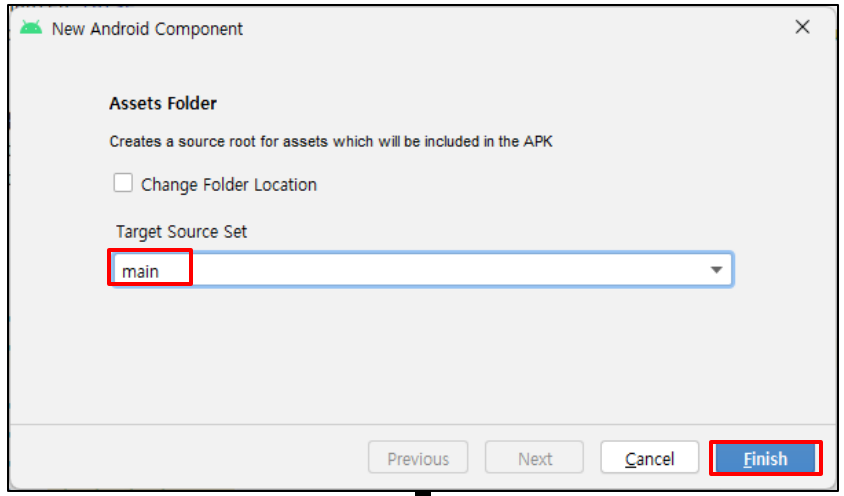
```
1  rootProject.name = "DigitClassifier"
2  include ':app'
```

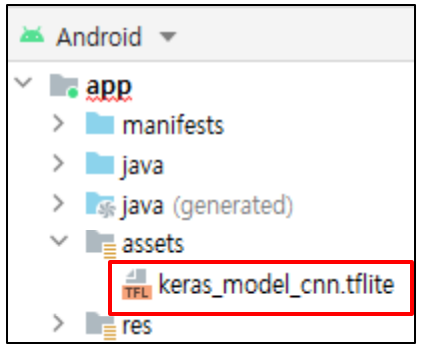
기초(따라하기) – 예제 1

4. 텐서플로 라이트 모델 로드

- Assets 폴더 생성 및 tflite 파일 추가 (DigitClassifier\app\src\main\assets)





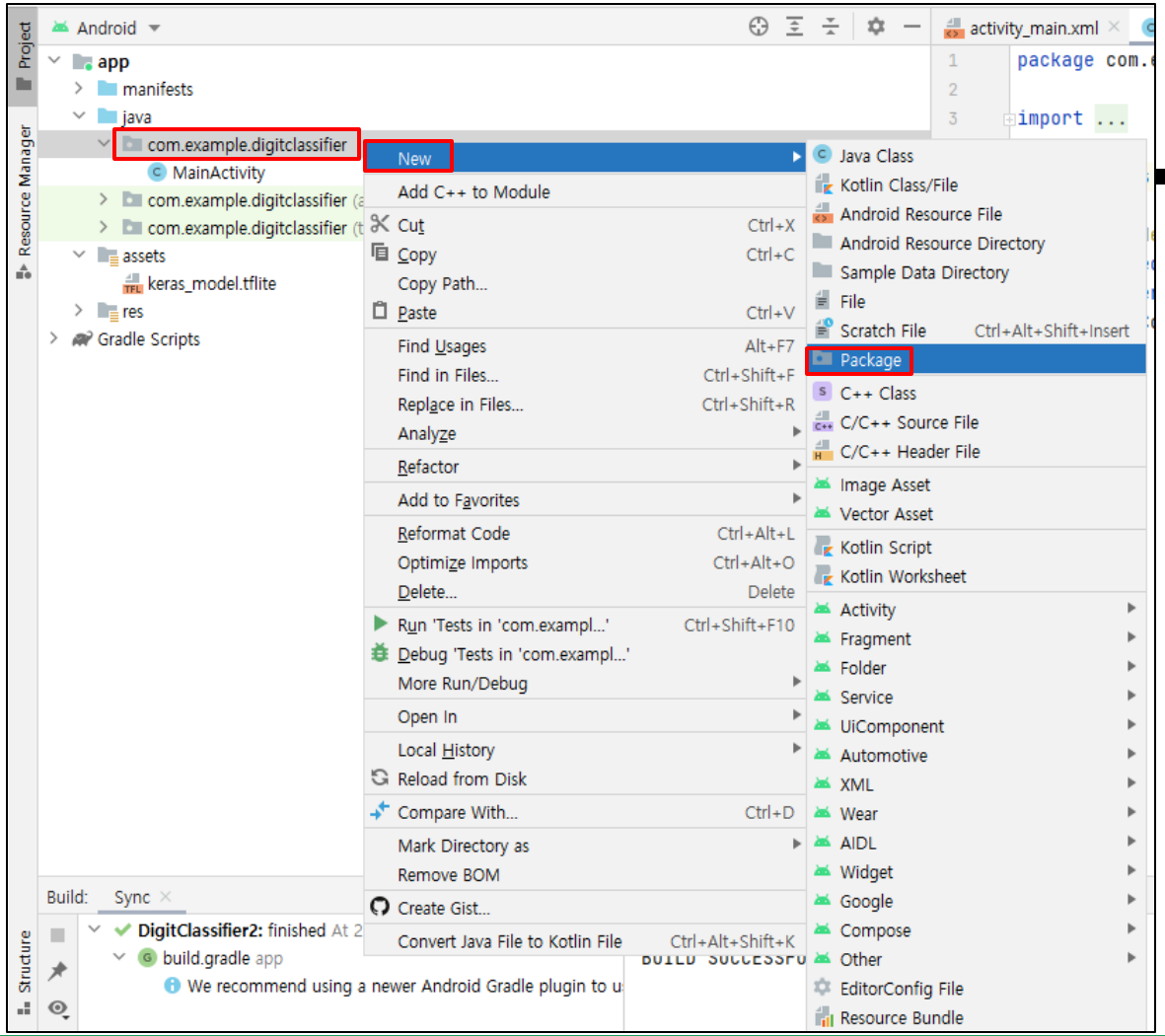


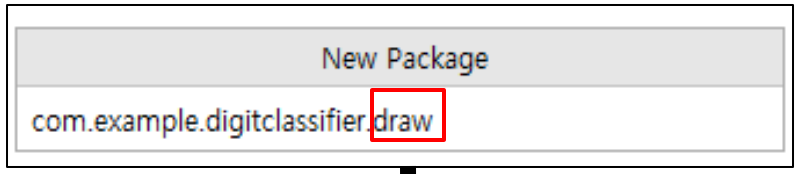
추가하기

기초(따라하기) – 예제 1

4. 텐서플로 라이트 모델 로드

- 패키지 만들기 – tflite, draw
  - MainActivity가 존재하는 폴더에 생성

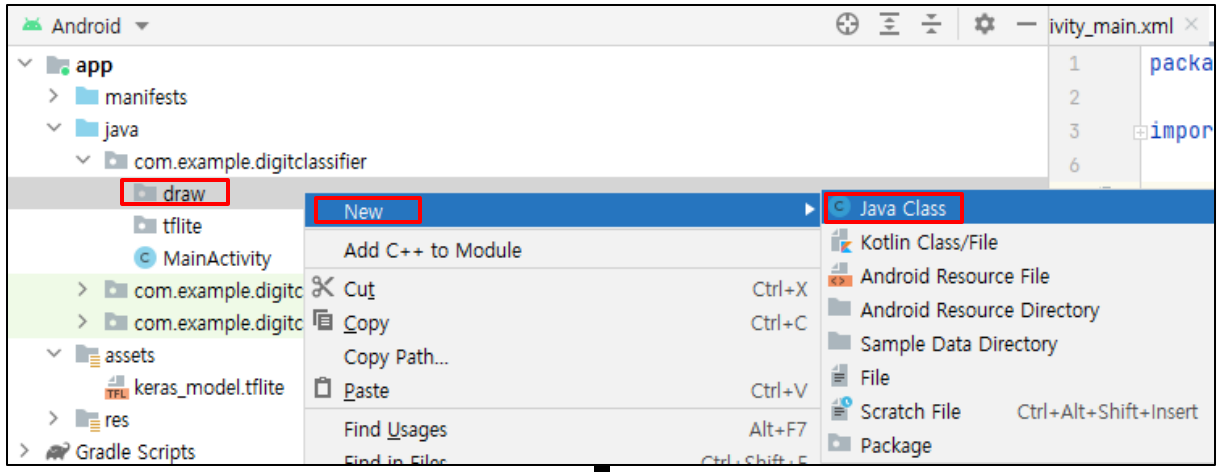


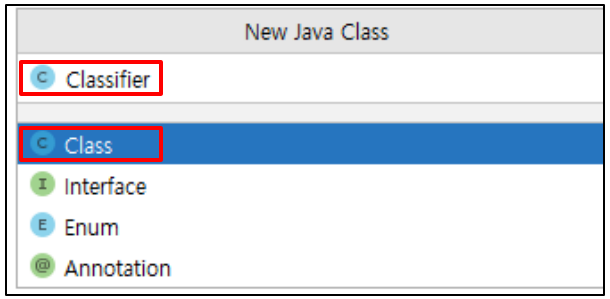


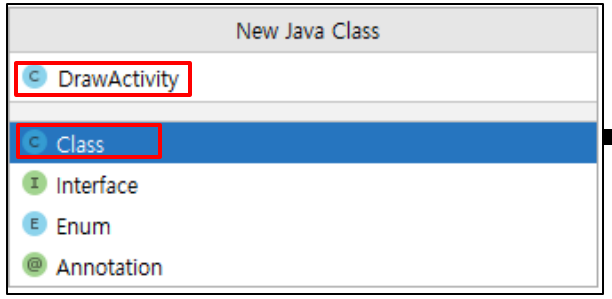
기초(따라하기) – 예제 1

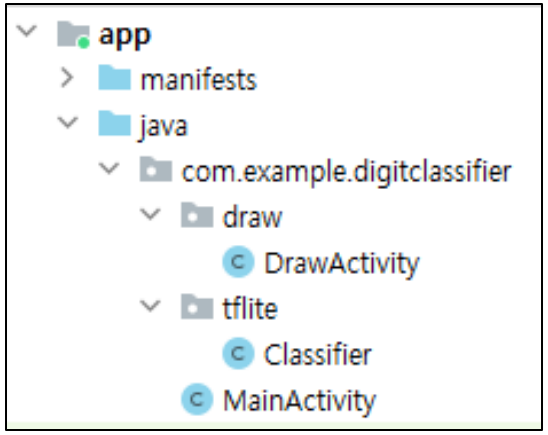
4. 텐서플로 라이트 모델 로드

- 클래스 만들기 – DrawActivity, Classifier
  - 각 draw, tflite 패키지 아래에 생성









클래스 생성완료

기초(따라하기) – 예제 1

4. 텐서플로 라이트 모델 로드

- (22 line) : 인터프리터 선언
- (30-35 line) : ByteBuffer 선언 및 Interpreter 선언을 통한 텐서플로 라이트 모델 로드

Classifier.java

```
18  public class Classifier{
19      private static final String MODEL_NAME = "keras_model_cnn.tflite";
20
21      Context context;
22      Interpreter interpreter = null;
23      int modelInputWidth, modelInputHeight, modelInputChannel;
24      int modelOutputClasses;
25
26      public Classifier(Context context) {
27          this.context = context;
28      }
29
30      public void init() throws IOException {
31          ByteBuffer model = loadModelFile(MODEL_NAME);
32          model.order(ByteOrder.nativeOrder());
33          interpreter = new Interpreter(model);
34
35          initModelShape();
36      }
```

기초(따라하기) – 예제 1

4. 텐서플로 라이트 모델 로드

- (39 line) : Assets 리소스 접근이 가능한 객체 생성
- (40 line) : Assets 리소스의 경로 획득
- (41-46 line) : MappedByteBuffer 자료형으로 파일 읽기

Classifier.java

```
38      private ByteBuffer loadModelFile(String modelName) throws IOException {
39          AssetManager am = context.getAssets();
40          AssetFileDescriptor afd = am.openFd(modelName);
41          FileInputStream fis = new FileInputStream(afd.getFileDescriptor());
42          FileChannel fc = fis.getChannel();
43          long startOffset = afd.getStartOffset();
44          long declaredLength = afd.getDeclaredLength();
45
46          return fc.map(FileChannel.MapMode.READ_ONLY, startOffset, declaredLength);
47      }
```

기초(따라하기) – 예제 1

5. 입력 이미지 전처리 코드 구현

- (50-54 line) : interpreter로부터 로드한 모델의 input shape를 변수로 읽어옴 (Channel, Width, Height)
- (56-58 line) : interpreter로부터 로드한 모델의 output shape를 변수로 읽어옴 (Class 수)

Draw Activity 이미지
(1080 x 1080 x 3)

입력 이미지
(28 x 28 x 1)

Classifier.java

```
49      private void initModelShape() {
50          Tensor inputTensor = interpreter.getInputTensor( inputIndex: 0);
51          int[] inputShape = inputTensor.shape();
52          modelInputChannel = inputShape[0];
53          modelInputWidth = inputShape[1];
54          modelInputHeight = inputShape[2];
55
56          Tensor outputTensor = interpreter.getOutputTensor( outputIndex: 0);
57          int[] outputShape = outputTensor.shape();
58          modelOutputClasses = outputShape[1];
59      }
```

기초(따라하기) – 예제 1

5. 입력 이미지 전처리 코드 구현

- (61-63 line) : 입력 이미지 크기 변환
- (65-84 line) : 입력 이미지 채널 및 포맷 변환

Classifier.java

```
61      private Bitmap resizeBitmap(Bitmap bitmap) {
62          return Bitmap.createScaledBitmap(bitmap, modelInputWidth, modelInputHeight,  filter: false);
63      }
64
65  @   private ByteBuffer convertBitmapToGrayByteBuffer(Bitmap bitmap) {
66          ByteBuffer byteByffer = ByteBuffer.allocateDirect(bitmap.getByteCount());
67          byteByffer.order(ByteOrder.nativeOrder());
68
69          int[] pixels = new int[bitmap.getWidth() * bitmap.getHeight()];
70          bitmap.getPixels(pixels,  offset: 0, bitmap.getWidth(),  x: 0,  y: 0, bitmap.getWidth(), bitmap.getHeight());
71
72          for (int pixel : pixels) {
73              int r = pixel >> 16 & 0xFF;
74              int g = pixel >> 8 & 0xFF;
75              int b = pixel & 0xFF;
76
77              float avgPixelValue = (r + g + b) / 3.0f;
78              float normalizedPixelValue = avgPixelValue / 255.0f;
79
80              byteByffer.putFloat(normalizedPixelValue);
81          }
82
83          return byteByffer;
84      }
```

기초(따라하기) – 예제 1

6. 추론

- (87 line) : 함수를 통한 데이터 전처리
- (89 line) : 결과 저장을 위한 배열 선언
- (91 line) : Interpreter의 run()함수활용한 텐서플로 라이트 모델 분류 실행
- (93 line) : 최대 값 추출
- (96-107 line) : 출력 데이터는 10개의 출력 클래스 (0 ~ 9) 중 가장 높은 확률 값을 찾는 argmax 함수 구현
- (109-112 line) : 인터프리터 닫기

Classifier.java

```
86      public Pair<Integer, Float> classify(Bitmap image) {
87          ByteBuffer buffer = convertBitmapToGrayByteBuffer(resizeBitmap(image));
88
89          float[][] result = new float[1][modelOutputClasses];
90
91          interpreter.run(buffer, result);
92
93          return argmax(result[0]);
94      }
95
96  @   private Pair<Integer, Float> argmax(float[] array) {
97          int argmax = 0;
98          float max = array[0];
99          for(int i = 1; i < array.length; i++) {
100             float f = array[i];
101             if(f > max) {
102                 argmax = i;
103                 max = f;
104             }
105         }
106         return new Pair<>(argmax, max);
107     }
108
109     public void finish() {
110         if(interpreter != null)
111             interpreter.close();
112     }
113 }
```

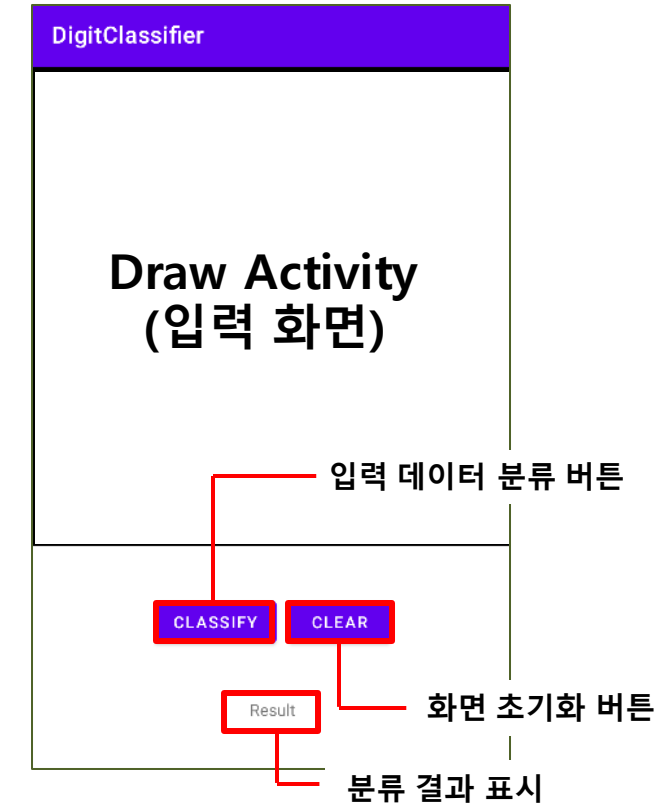
기초(따라하기) – 예제 1

7. DrawActivity 레이아웃 구성

- 터치를 통해 직접 숫자 이미지를 그려 넣는 Activity
- Main Activity에서 이미지 입력 창으로 넘어가는 기능 구성

- Activity 생성
  - **액티비티** : Empty Activity
  - **액티비티 명** : DrawActivity
  - **레이아웃 파일 명** : activity_draw
  - **프로젝트 언어** : JAVA

- 레이아웃 구성 요소
  - 터치 입력을 받아 숫자를 그리는 DrawView
  - 추론을 시작하거나 초기화 하기 위한 CLASSIFY, CLEAR Button
  - 추론 결과를 보여주는 TextView



기초(따라하기) – 예제 1

7. DrawActivity 레이아웃 구성

- Activity_draw.xml 생성

- (2-8 line) : Constraintlayout 설정
- (10-16 line) : Drawview 위젯 설정

activity_draw.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.constraintlayout.widget.ConstraintLayout
3       xmlns:android="http://schemas.android.com/apk/res/android"
4       xmlns:app="http://schemas.android.com/apk/res-auto"
5       xmlns:tools="http://schemas.android.com/tools"
6       android:layout_width="match_parent"
7       android:layout_height="match_parent"
8       tools:context=".draw.DrawActivity">
9
10      <com.divyanshu.draw.widget.DrawView
11          android:id="@+id/drawView"
12          android:layout_width="match_parent"
13          android:layout_height="0dp"
14          app:layout_constraintDimensionRatio="1:1"
15          app:layout_constraintTop_toTopOf="parent">
16      </com.divyanshu.draw.widget.DrawView>
```

기초(따라하기) – 예제 1

7. DrawActivity 레이아웃 구성

- (18-26 line) : LinearLayout 설정
- (28-32 line) : 분류 버튼 설정
- (34-39 line) : 초기화 버튼 설정

activity_draw.xml

```
18      <LinearLayout
19          android:id="@+id/buttonLayout"
20          android:layout_width="wrap_content"
21          android:layout_height="wrap_content"
22          android:orientation="horizontal"
23          app:layout_constraintTop_toBottomOf="@id/drawView"
24          app:layout_constraintBottom_toTopOf="@id/resultView"
25          app:layout_constraintStart_toStartOf="parent"
26          app:layout_constraintEnd_toEndOf="parent">
27
28          <Button
29              android:id="@+id/classifyBtn"
30              android:layout_width="wrap_content"
31              android:layout_height="wrap_content"
32              android:text="Classify" />
33
34          <Button
35              android:id="@+id/clearBtn"
36              android:layout_width="wrap_content"
37              android:layout_height="wrap_content"
38              android:text="Clear"
39              android:layout_marginStart="10dp" />
40
41      </LinearLayout>
```

기초(따라하기) – 예제 1

7. DrawActivity 레이아웃 구성

- (43-51 line) : 결과창 표시 TextView 설정

activity_draw.xml

```
43      <TextView
44          android:id="@+id/resultView"
45          android:layout_width="wrap_content"
46          android:layout_height="wrap_content"
47          android:text="Result"
48          app:layout_constraintTop_toBottomOf="@id/buttonLayout"
49          app:layout_constraintBottom_toBottomOf="parent"
50          app:layout_constraintStart_toStartOf="parent"
51          app:layout_constraintEnd_toEndOf="parent" />
52
53  </androidx.constraintlayout.widget.ConstraintLayout>
```

기초(따라하기) – 예제 1

8. MainActivity 레이아웃 구성

- (2-8 line) : MainAcitivity 레이아웃 설정
- (10-15 line) : DrawView로 넘어가는 버튼 설정

activity_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      tools:context=".MainActivity">
9
10     <Button
11         android:id="@+id/drawBtn"
12         android:layout_width="wrap_content"
13         android:layout_height="wrap_content"
14         android:layout_weight="1"
15         android:text="DrawView" />
16
17 </LinearLayout>
```

기초(따라하기) – 예제 1

9. UI 로직 구현

- (18 line) : findViewById() 함수를 통한 DrawView버튼 연결
- (19 line) : 버튼 리스너 설정
- (20 line) : DrawActivity를 실행하는 인텐트 생성
- (21 line) : StartActivity() 함수에 인텐트를 전달

MainActivity.java

```
11  public class MainActivity extends AppCompatActivity {
12
13      @Override
14      protected void onCreate(Bundle savedInstanceState) {
15          super.onCreate(savedInstanceState);
16          setContentView(R.layout.activity_main);
17
18          Button drawBtn = findViewById(R.id.drawBtn);
19          drawBtn.setOnClickListener(view -> {
20              Intent i = new Intent( packageContext: MainActivity.this, DrawActivity.class);
21              startActivity(i);
22          });
23      }
24  }
```

기초(따라하기) – 예제 1

10. DrawActivity 코드 구현

- (29-31 line) : 선 두께 및 실선 크기 변경
- (36-37 line) : Drawview로부터 이미지 추출
- (40-45 line) : TextView를 통한 결과 표시

DrawActivity.java

```
20  public class DrawActivity extends AppCompatActivity {
21      Classifier cls;
22
23      @Override
24      protected void onCreate(Bundle savedInstanceState) {
25          super.onCreate(savedInstanceState);
26          setContentView(R.layout.activity_draw);
27
28          DrawView drawView = findViewById(R.id.drawView);
29          drawView.setStrokeWidth(100.0f);
30          drawView.setBackgroundColor(Color.BLACK);
31          drawView.setColor(Color.WHITE);
32
33          TextView resultView = findViewById(R.id.resultView);
34
35          Button classifyBtn = findViewById(R.id.classifyBtn);
36          classifyBtn.setOnClickListener(v -> {
37              Bitmap image = drawView.getBitmap();
38
39              Pair<Integer, Float> res = cls.classify(image);
40              String outStr = String.format(
41                      Locale.ENGLISH,
42                      format: "%d, %.0f%%",
43                      ...args: res.first,
44                      res.second * 100.0f);
45              resultView.setText(outStr);
46          });
```

기초(따라하기) – 예제 1

10. DrawActivity 코드 구현

- (48-51 line) : DrawView 초기화

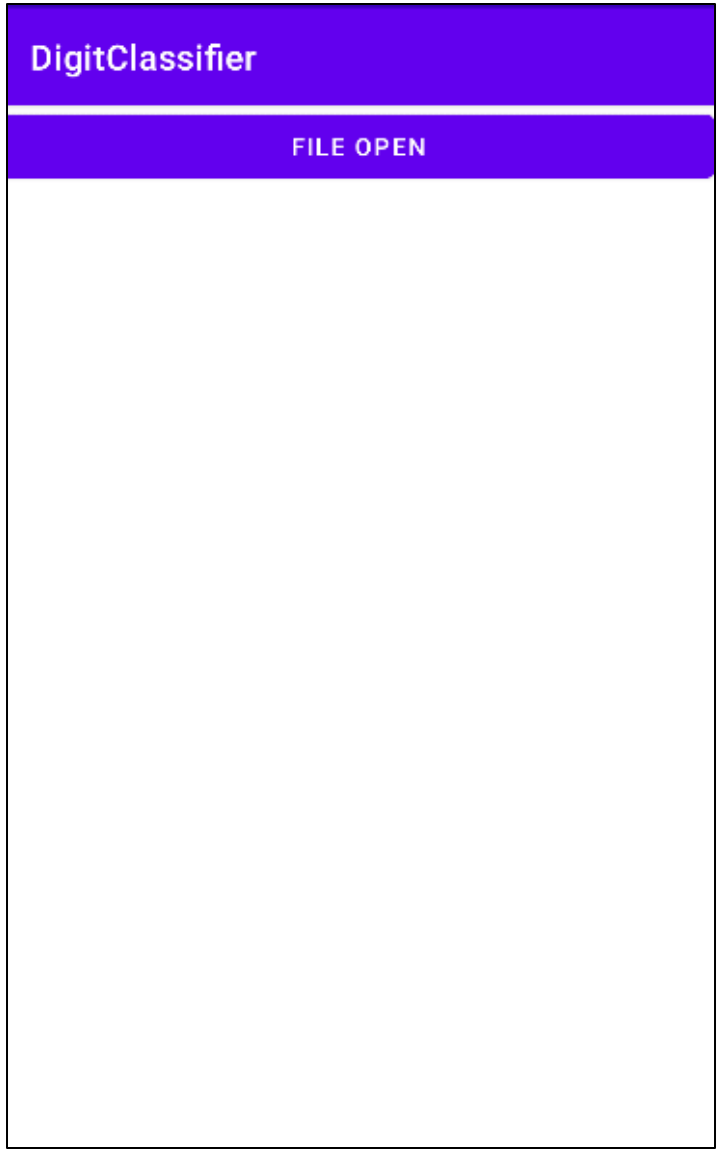
DrawActivity.java

```
48          Button clearBtn = findViewById(R.id.clearBtn);
49          clearBtn.setOnClickListener(v -> {
50              drawView.clearCanvas();
51          });
52
53          cls = new Classifier( context: this);
54          try {
55              cls.init();
56          } catch(IOException ioe) {
57              Log.d( tag: "DigitClassifier",  msg: "failed to init Classifier", ioe);
58          }
59
60      }
61
62      @Override
63      protected void onDestroy() {
64          cls.finish();
65          super.onDestroy();
66      }
67  }
```

응용(로직구현) – 예제 2

❖ 디바이스에 저장된 이미지 분류

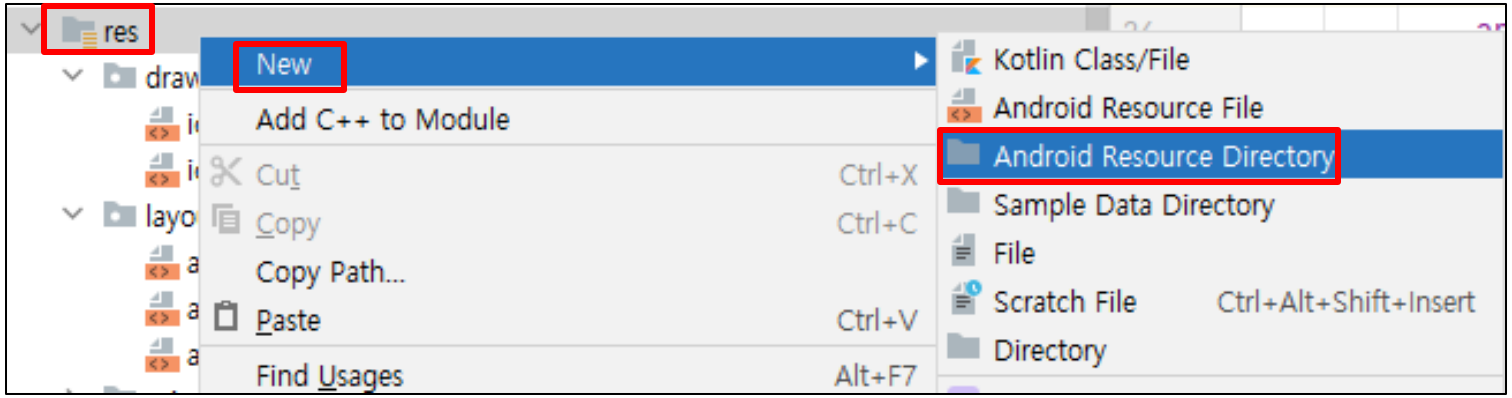
- 예제 1번을 기반으로 수정 할 것!
  - 예제 1을 참조하되, 이어서 만들지 말 것!
  - 예시 GIF 처럼 동작할 수 있도록 조치할 것!
  - 0~9까지의 이미지를 AVM에 넣을 것!



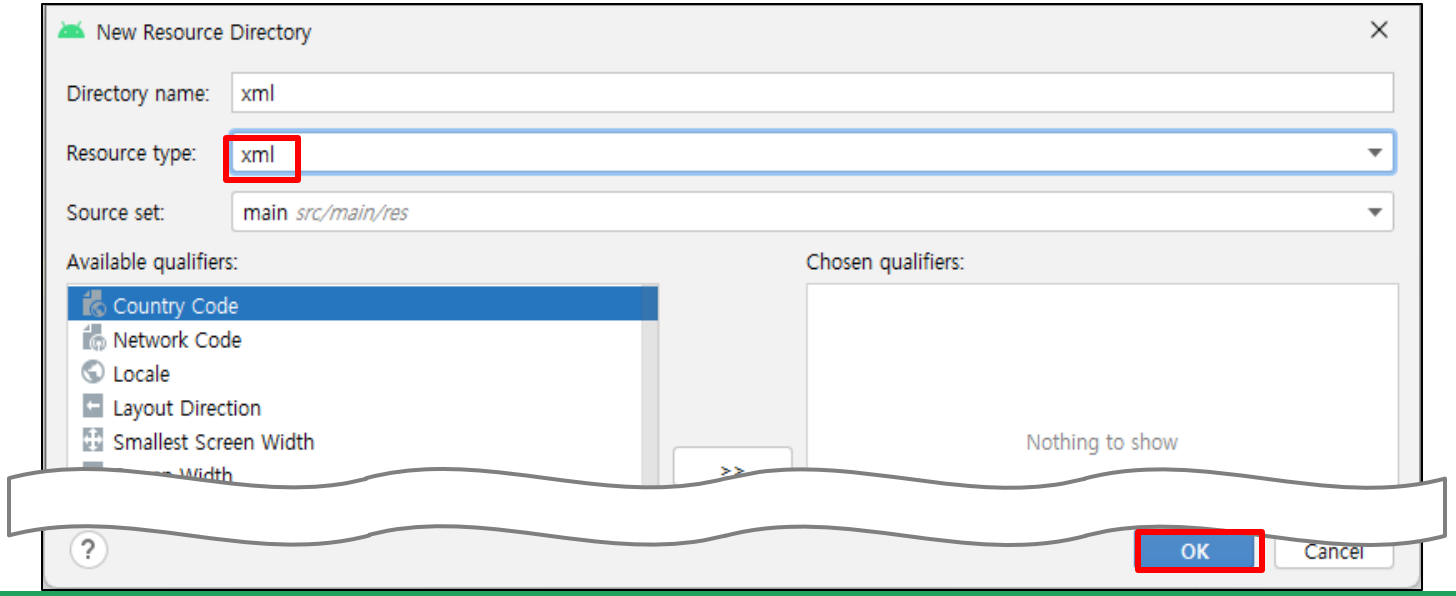
응용(로직구현) – 예제 2

❖ 이미지를 불러오기 위한 설정

- res 폴더 우클릭 -> New -> Android Resource Directory



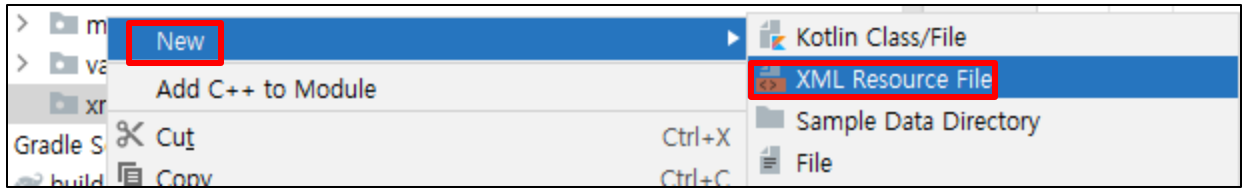
- Directory name : xml, Resource type : xml 설정 후 확인



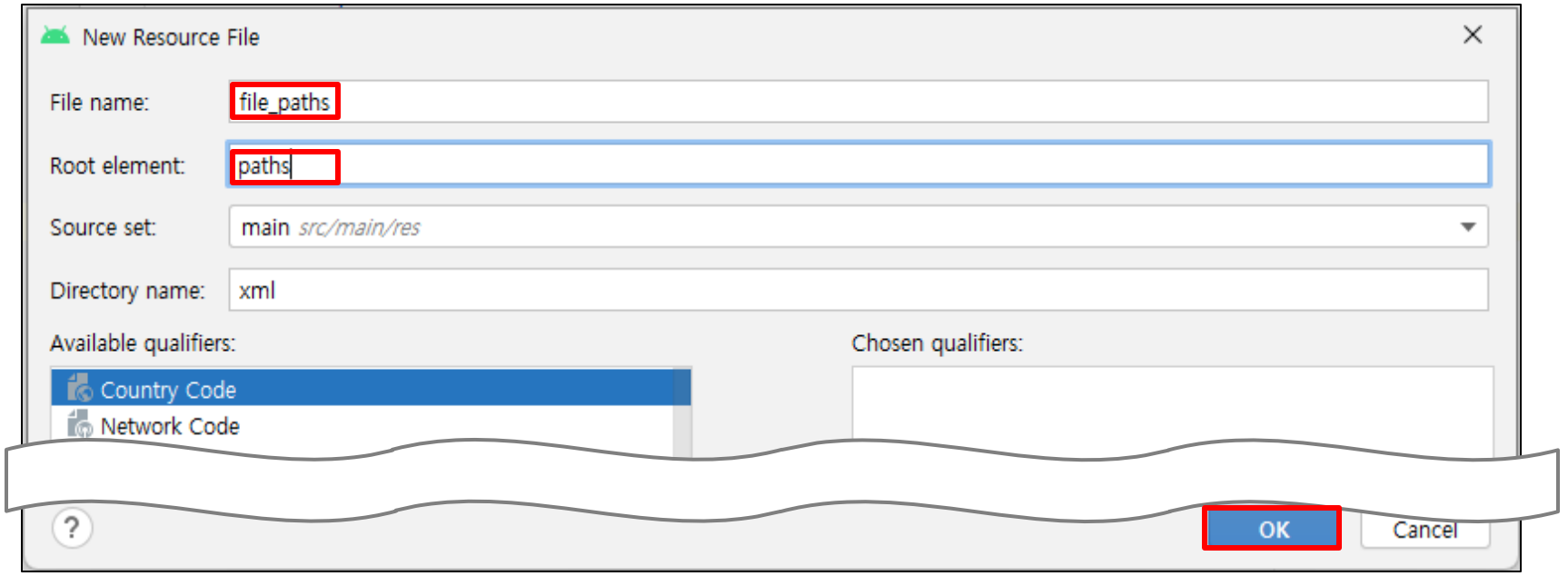
응용(로직구현) – 예제 2

❖ 이미지를 불러오기 위한 설정

- xml 폴더 우클릭 -> New -> XML Resource File 클릭



- **File name** : file_paths, Root element : paths 설정 후, 확인



응용(로직구현) – 예제 2

❖ 디바이스에서 이미지를 불러오기 위한 설정

- 아래의 코드를 방금 생성한 xml 파일에 추가

```xml
<?xml version="1.0" encoding="utf-8"?>
<paths xmlns:android="http://schemas.android.com/apk/res/android">
    <external-path
        name="images"
        path="Android/data/com.example.Digitclassifier/files/Pictures" />
</paths>
```

- AndroidManifest.xml 파일에 저장소 관련 권한 추가

```xml
        </activity>
        <provider
            android:name="androidx.core.content.FileProvider"
            android:authorities="com.example.digitclassifier"
            android:grantUriPermissions="true">
            <meta-data
                android:name="android.support.FILE_PROVIDER_PATHS"
                android:resource="@xml/file_paths" />
        </provider>
    </application>
```

응용(로직구현) – 예제 2

```
10      <ImageView
11          android:id="@+id/imageView"
12          android:layout_width="match_parent"
13          android:layout_height="0dp"
14          app:layout_constraintDimensionRatio="1:1"
15          app:layout_constraintTop_toTopOf="parent" />
16
17      <LinearLayout
18          android:id="@+id/buttonLayout"
19          android:layout_width="wrap_content"
20          android:layout_height="wrap_content"
21          android:orientation="horizontal"
22          app:layout_constraintTop_toBottomOf="@id/imageView"
23          app:layout_constraintBottom_toTopOf="@id/textView"
24          app:layout_constraintStart_toStartOf="parent"
25          app:layout_constraintEnd_toEndOf="parent">
26
27          <Button
28              android:id="@+id/classifyBtn"
29              android:layout_width="wrap_content"
30              android:layout_height="wrap_content"
31              android:text="Classify"
32              app:layout_constraintEnd_toEndOf="parent"
33              app:layout_constraintStart_toStartOf="parent"
34              app:layout_constraintTop_toBottomOf="@id/imageView" />
35
36          <Button
37              android:id="@+id/selectBtn"
38              android:layout_width="161dp"
39              android:layout_height="wrap_content"
40              android:text="Select File"
41              android:layout_marginStart="10dp"
42              app:layout_constraintEnd_toEndOf="parent"
43              app:layout_constraintStart_toStartOf="parent"
44              app:layout_constraintTop_toBottomOf="@id/imageView" />
45
46      </LinearLayout>
```

❖ GalleryActivity 레이아웃

- activity_gallery.xml 생성
  - res/layout/
  - 최상위 태그 - constraintLayout

```
48      <TextView
49          android:id="@+id/textView"
50          android:layout_width="wrap_content"
51          android:layout_height="wrap_content"
52          android:text="Result"
53          app:layout_constraintBottom_toBottomOf="parent"
54          app:layout_constraintStart_toStartOf="parent"
55          app:layout_constraintEnd_toEndOf="parent" />
```

응용(로직구현) – 예제 2

❖ GalleryActivity 코드

- (25-31 line) : GalleryActivity 클래스의 멤버 변수 선언

```
24  public class GalleryActivity extends AppCompatActivity {
25      public static final String TAG = "[IC]GalleryActivity";
26      public static final int GALLERY_IMAGE_REQUEST_CODE = 1;
27
28      Classifier cls;
29      Bitmap bitmap = null;
30      private ImageView imageView;
31      private TextView textView;
```

응용(로직구현) – 예제 2

❖ **GalleryActivity 코드**

- (38-43 line) : 레이아웃 관련 요소 설정
- (39 line) : Select 버튼 동작 설정
- (44 – 55 line) : 분류 버튼 클릭 시 동작 설정
- (57 line) : Classifier 생성

GalleryActivity.java

```
33      @Override
34      protected void onCreate(Bundle savedInstanceState) {
35          super.onCreate(savedInstanceState);
36          setContentView(R.layout.activity_gallery);
37
38
39          selectBtn.setOnClickListener(v -> getImageFromGallery());
40
41
42
43
44
45          classifyBtn.setOnClickListener(v -> {
46              if(bitmap != null) {
47                  Pair<Integer, Float> res = cls.classify(bitmap);
48                  String outStr = String.format(
49                          Locale.ENGLISH,
50                          format: "%d, %.0f%%",
51                          ...args: res.first,
52                          res.second * 100.0f);
53
54              }
55          });
56
57          cls = new Classifier( context: this);
58          try {
59              cls.init();
60          } catch (IOException ioe) {
61              ioe.printStackTrace();
62          }
63
64      }
```

응용(로직구현) – 예제 2

❖ GalleryActivity 코드

- (66-69 line) : 저장소로부터 이미지를 불러오기 위한 intent 설정 추가

GalleryActivity.java

```
66      private void getImageFromGallery(){
67          Intent intent = new Intent(Intent.ACTION_GET_CONTENT).setType("image/*");
68          startActivityForResult(intent, GALLERY_IMAGE_REQUEST_CODE);
69      }
```

응용(로직구현) – 예제 2

❖ **GalleryActivity 코드**

- (74 - 75 line) : 파일 불러오기 결과 유효성 체크
- (76-78 line) : 파일 데이터 유효성 체크
- (82-91 line) : Android 버전에 따른 이미지 처리

```
71      @Override
72      public void onActivityResult(int requestCode, int resultCode, Intent data) {
73          super.onActivityResult(requestCode, resultCode, data);
74          if (resultCode == Activity.RESULT_OK &&
75                  requestCode == GALLERY_IMAGE_REQUEST_CODE) {
76              if (data == null) {
77                  return;
78              }
79
80              Uri selectedImage = data.getData();
81
82              try {
83                  if(Build.VERSION.SDK_INT >= 29) {
84                      ImageDecoder.Source src =
85                              ImageDecoder.createSource(getContentResolver(), selectedImage);
86                      bitmap = ImageDecoder.decodeBitmap(src).copy(Bitmap.Config.ARGB_8888,  isMutable: true);
87                  } else {
88                      bitmap = MediaStore.Images.Media.getBitmap(getContentResolver(), selectedImage);
89                  }
90              } catch (IOException ioe) {
91                  Log.e(TAG,  msg: "Failed to read Image", ioe);
92              }
93
94              if(bitmap != null) {
95                  imageView.setImageBitmap(bitmap);
96              }
97          }
98      }
```

응용(로직구현) – 예제 2

❖ **GalleryActivity 코드**

- (101-104 line) : 종료를 위한 처리

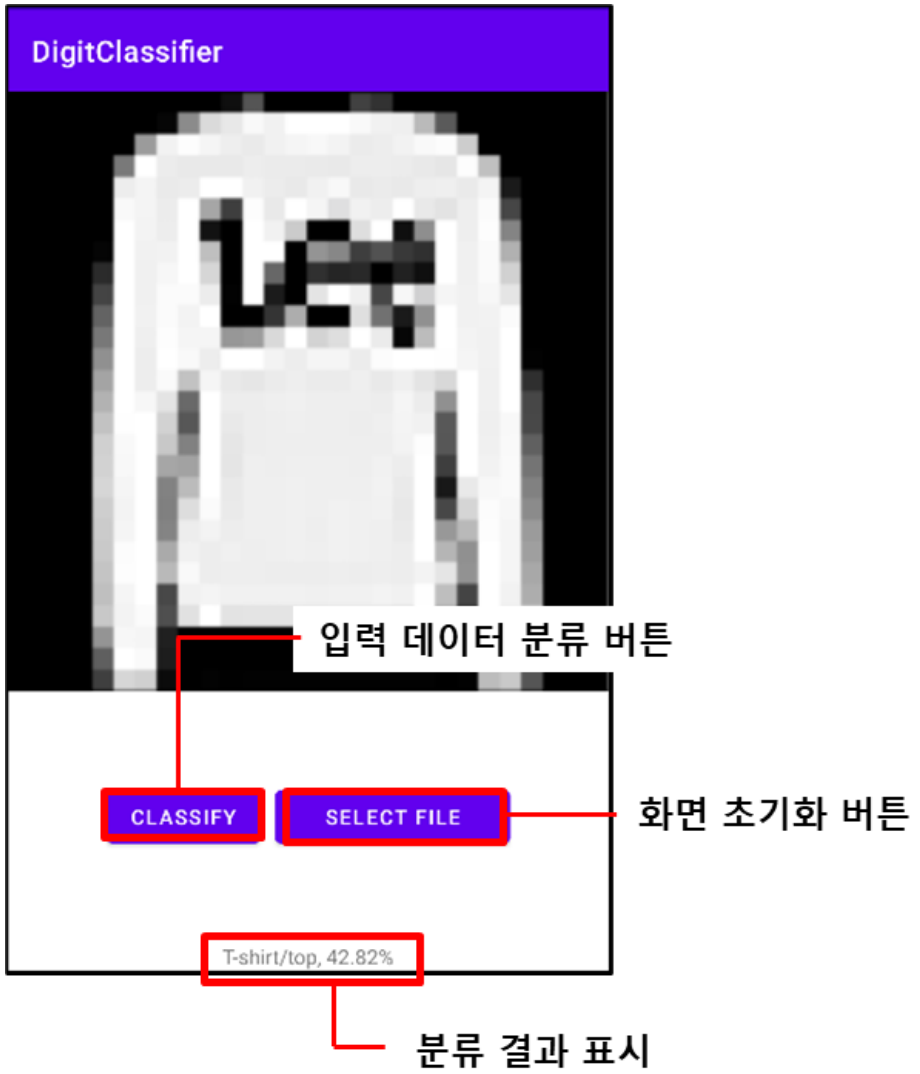
GalleryActivity.java

```
101     protected void onDestroy()
102         cls.finish();
103         super.onDestroy();
104     }
105 }
```

심화(과제물) – 예제 3

❖ FashionMnist 분류 앱 개발

- 조건
  - FashionMnist 모델 직접 학습
  - FashionMnist 모델 직접 변환
  - 파일선택 및 분류 버튼을 통한 분류 제공
  - 분류 결과는 class 명으로 출력
  - 제출
    - 안드로이드 프로젝트
    - ipython
      - » 모델 생성, 모델 변환 모두 포함
  - 참조
    - https://www.kaggle.com/zalando-research/fashionmnist



심화(과제물) – 예제 3

❖ FashionMnist 분류 앱 개발

- 모델 생성 코드
  - 해당 코드를 참조하여서 모델 변환 부분을 구현할 것!

```
In [1] :    import tensorflow as tf

            fashion_mnist  = tf.keras.datasets.fashion_mnist
            (x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()

            x_train, x_test = x_train / 255.0, x_test / 255.0

            x_train_4d = x_train.reshape(-1, 28, 28, 1)
            x_test_4d = x_test.reshape(-1, 28, 28, 1)

            cnn_model = tf.keras.models.Sequential([
                tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),
                tf.keras.layers.MaxPooling2D((2, 2)),
                tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
                tf.keras.layers.MaxPooling2D((2, 2)),
                tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
                tf.keras.layers.Flatten(),
                tf.keras.layers.Dense(64, activation='relu'),
                tf.keras.layers.Dense(10, activation='softmax')])

            cnn_model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
            cnn_model.fit(x_train_4d, y_train, epochs=5)
```