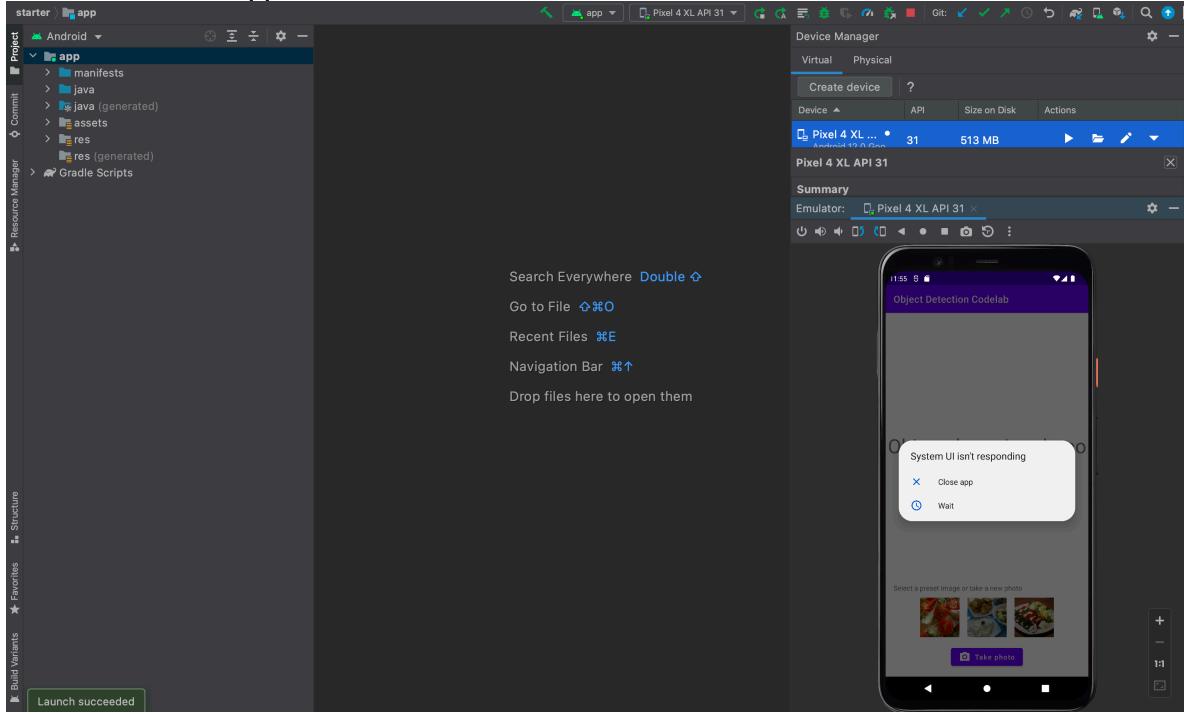
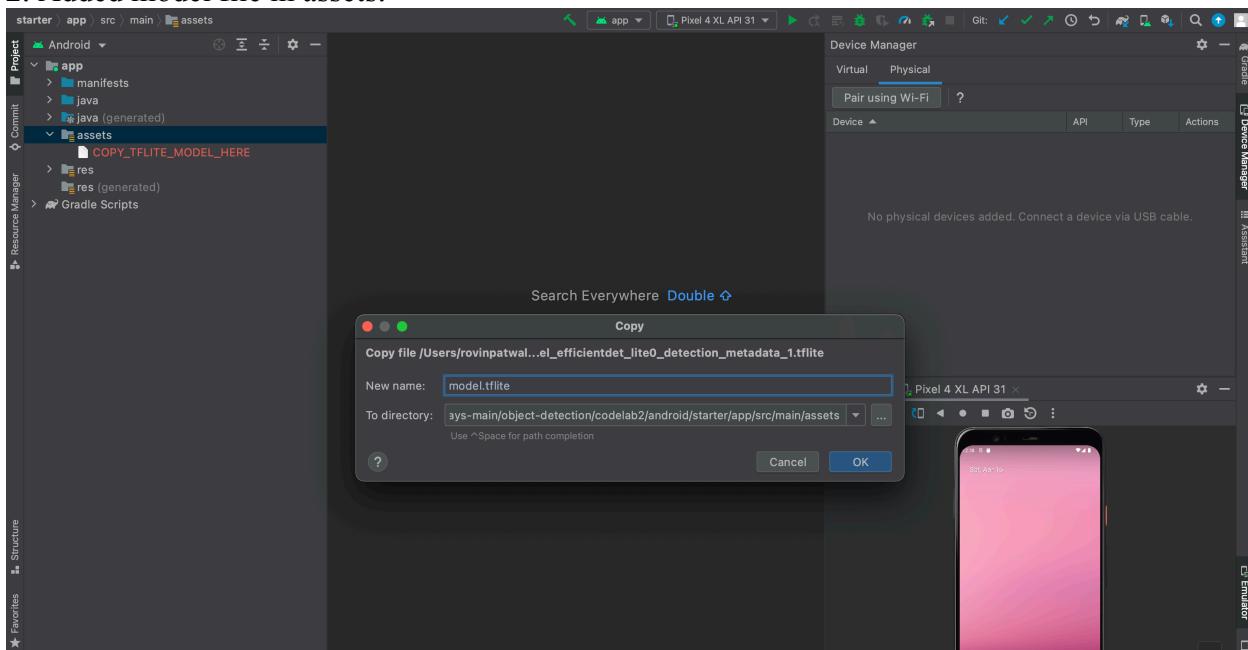


## a) Mobile App and Image training on device: Build and deploy a custom object detection model using tflite on android simulator

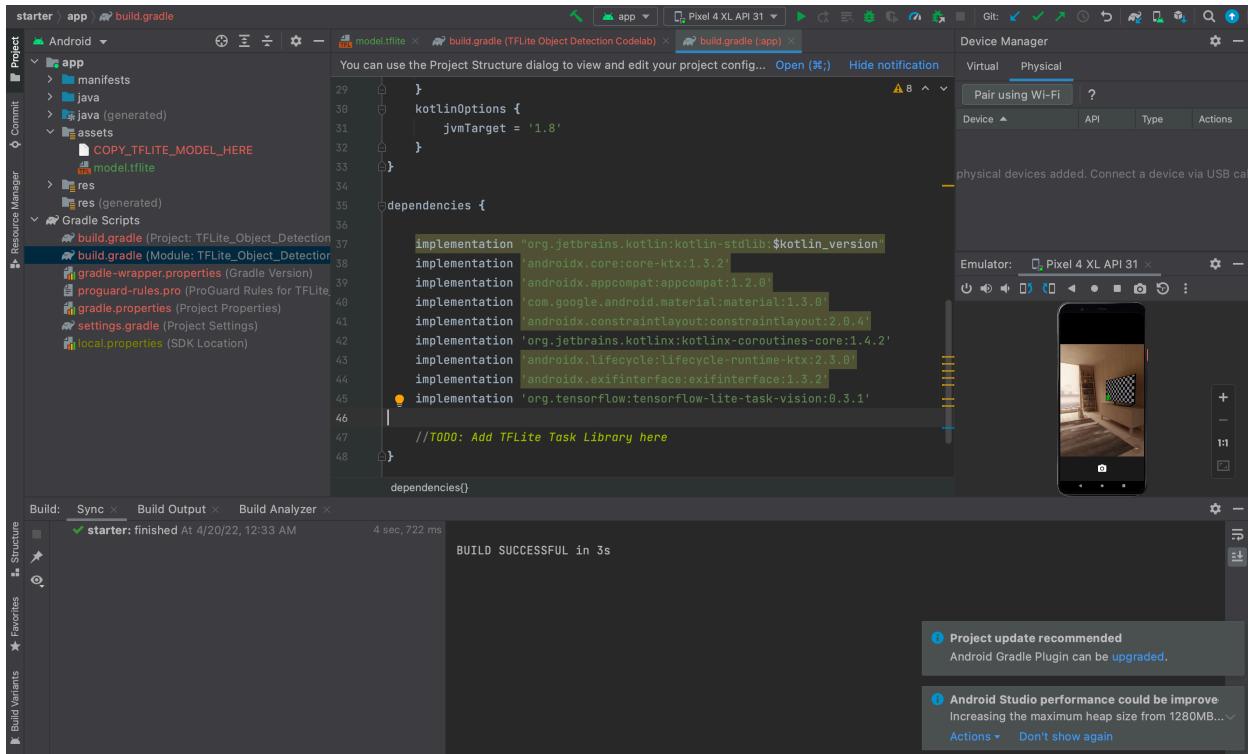
1: Ran the starter app:



2. Added model file in assets:

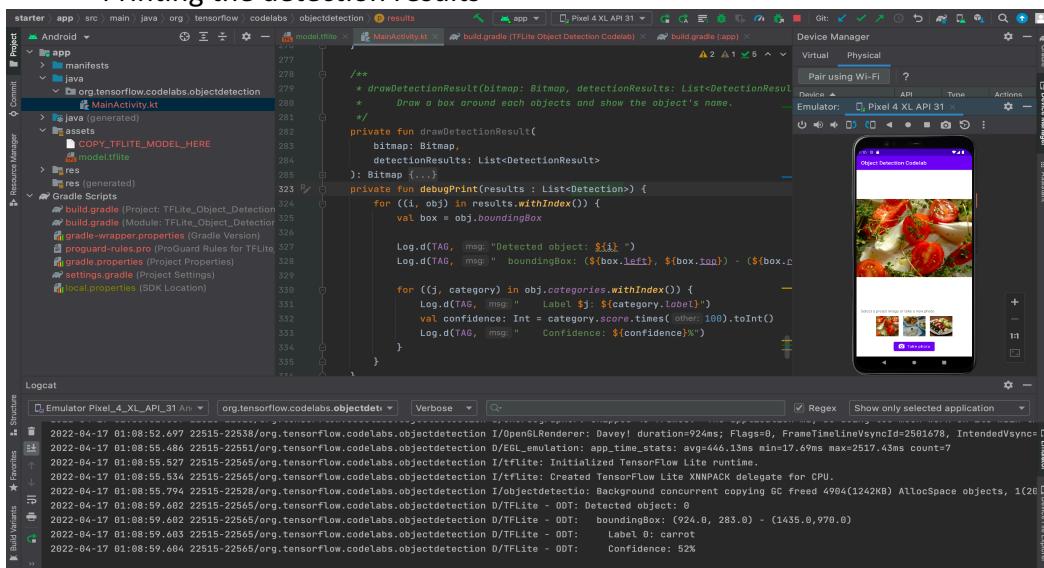


3. Added dependencies in build.grade file



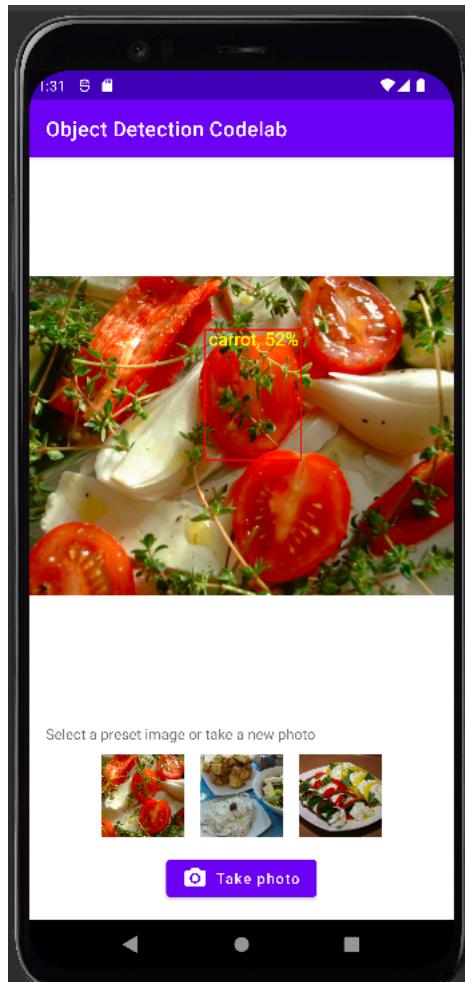
#### 4: Added code for:

- Creation of image object
- Creation of Detector instance
- Feeding Image(s) to the detector
- Printing the detection results



## 5. Logcat result after running the object detection:

```
Logcat
Emulator Pixel_4_XL_API_31 An org.tensorflow.codelabs.objectdetection Verbose Q Regex Show only selected application
2022-04-17 01:08:52.697 22515-22538/org.tensorflow.codelabs.objectdetection I/OpenGLRenderer: Davey! duration=924ms; Flags=0, FrameTimelineVsyncId=2501678, IntendedVsync=2501678, ActualVsync=2501678, SurfaceId=0, SurfaceType=1
2022-04-17 01:08:55.486 22515-22551/org.tensorflow.codelabs.objectdetection D/EGL_emulation: app_time_stats: avg=446.13ms min=17.69ms max=2517.43ms count=7
2022-04-17 01:08:55.527 22515-22565/org.tensorflow.codelabs.objectdetection I/tflite: Initialized TensorFlow Lite runtime.
2022-04-17 01:08:55.534 22515-22565/org.tensorflow.codelabs.objectdetection I/tflite: Created TensorFlow Lite XNNPACK delegate for CPU.
2022-04-17 01:08:55.794 22515-22528/org.tensorflow.codelabs.objectdetection I/objectdetection: Background concurrent copying GC freed 4984(1242KB) AllocSpace objects, 1(200K) free in 100ms
2022-04-17 01:08:59.602 22515-22565/org.tensorflow.codelabs.objectdetection D/TFLite - ODT: Detected object: 0
2022-04-17 01:08:59.602 22515-22565/org.tensorflow.codelabs.objectdetection D/TFLite - ODT: boundingBox: (924.0, 283.0) - (1435.0, 978.0)
2022-04-17 01:08:59.602 22515-22565/org.tensorflow.codelabs.objectdetection D/TFLite - ODT: Label 0: carrot
2022-04-17 01:08:59.604 22515-22565/org.tensorflow.codelabs.objectdetection D/TFLite - ODT: Confidence: 52%
```



## b) Mobile app and audio training on device : Build a custom pre-trained Audio Classification model

## 1. Data Set used for this mode is: Birdsong dataset.

Build a custom pre-trained Audio Classification model

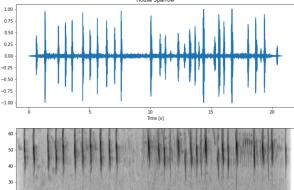
English Update

Before you begin

2. The Birds dataset

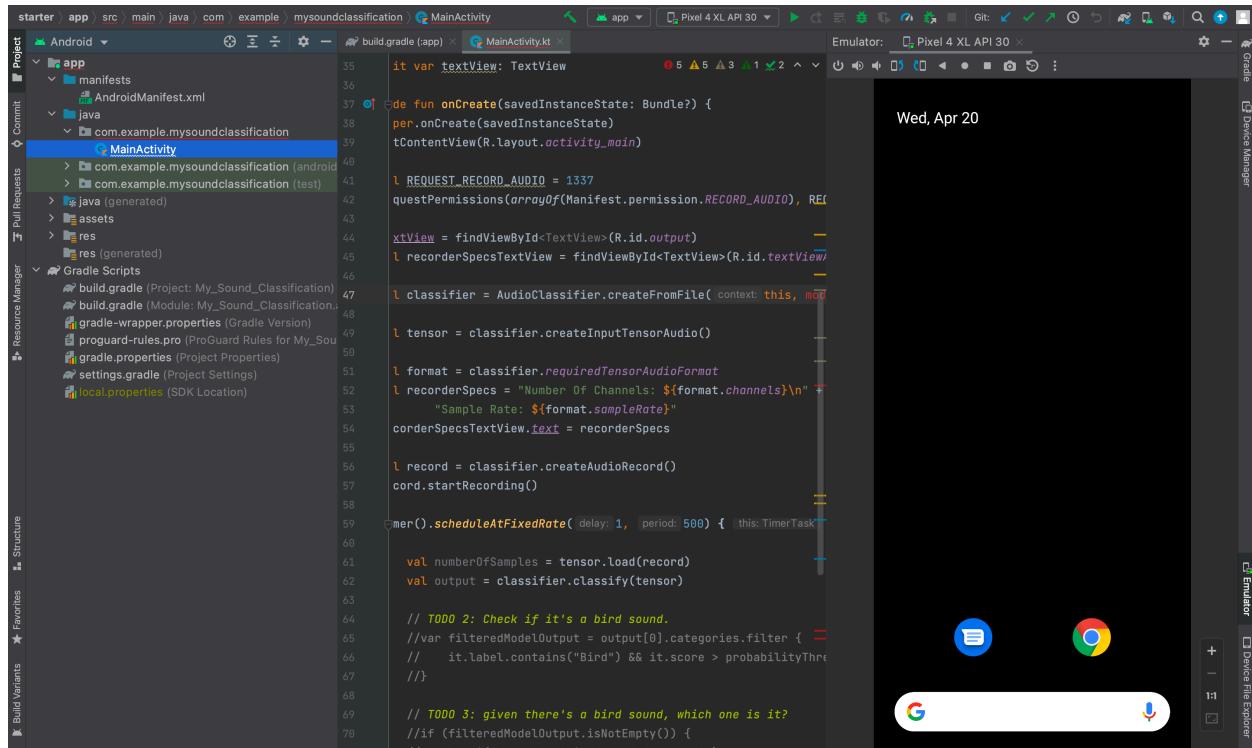
You will use a Birdsong dataset that is already prepared to make it easier to use. All the audio files come from the [Xeno-canto website](#).

This dataset contains songs from:

Name: House Sparrow	Code: houspa
	 [audio]
Name: Red Crossbill	Code: redcro

Back Next

## 2. Model after importing in Android Studio



The screenshot shows the Android Studio interface with the code editor open to `MainActivity.kt`. The code is written in Kotlin and handles audio classification. It includes imports for `androidx.core.util.Consumer`, `com.google.mlkit.vision.audio.AudioClassification`, and `com.google.mlkit.vision.common.InputImage`. The code initializes an `AudioClassifier`, sets up a recorder, and processes audio frames to classify them as bird sounds. A TODO comment indicates the next step is to check if the sound is a bird. The Android Studio interface also shows the project structure, build variants, and device manager.

```
it var textView: TextView
    de fun onCreate(savedInstanceState: Bundle?) {
        per.onCreate(savedInstanceState)
        tContentView(R.layout.activity_main)

        REQUEST_RECORD_AUDIO = 1337
        questPermissions(arrayOf(Manifest.permission.RECORD_AUDIO), RE

        textView = findViewById<TextView>(R.id.output)
        recorderSpecsTextView = findViewById<TextView>(R.id.textView

        classifier = AudioClassifier.createFromFile(context: this, mod

        tensor = classifier.createInputTensorAudio()

        format = classifier.requiredTensorAudioFormat
        recorderSpecs = "Number Of Channels: ${format.channels}\n"
        recorderSpecs += "Sample Rate: ${format.sampleRate}"
        corderSpecsTextView.text = recorderSpecs

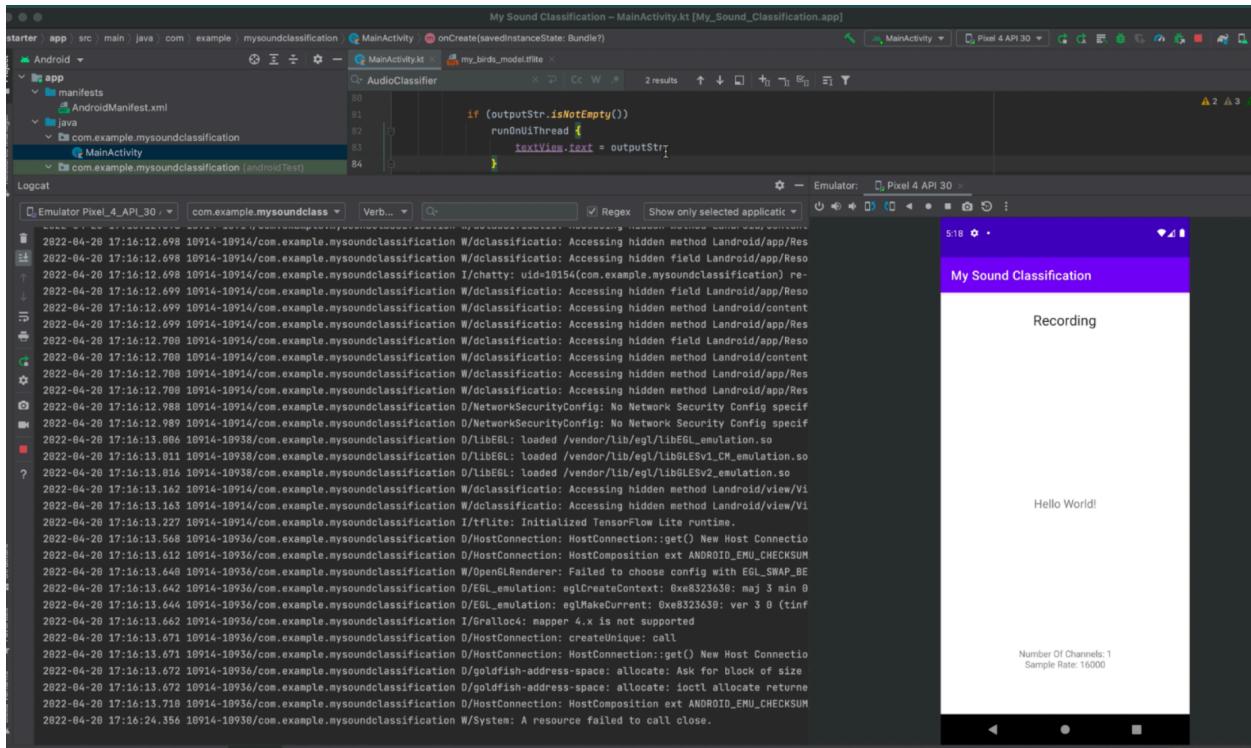
        record = classifier.createAudioRecord()
        cord.startRecording()

        timer().scheduleAtFixedRate( delay: 1, period: 500) { this:TimerTask

            val numberofSamples = tensor.load(record)
            val output = classifier.classify(tensor)

            // TODO 2: Check if it's a bird sound.
            //var filteredModelOutput = output[0].categories.filter { ==
            //    it.label.contains("Bird") && it.score > probabilityThre
            //}

            // TODO 3: given there's a bird sound, which one is it?
            //if (filteredModelOutput.isNotEmpty()) {
```



### c) Web app and image training on device: TensorFlow.js Transfer Learning Image Classifier

Code in glitch: <https://glitch.com/edit/#!/internal-dust-valley?path=README.md%3A1%3A0>  
 Live model link: <https://internal-dust-valley.glitch.me/>

The screenshot shows the Glitch interface with the project 'internal-dust-valley'. The main content area displays the README.md file, which contains the following text:

We've got something just for Glitch creators like you: a FREE sneak peek at our upcoming super-fast edge caching, powered by Fastly. Invites are going out first-come, first-served. Click to be one of the first. [Learn more](#)

**TensorFlow.js Boilerplate**

This very simple skeleton simply loads in TensorFlow.js and prints out the version once loaded to the DOM.

From these humble beginnings you can do some really great things.

Feel free to fork this and use it as a quick way to start coding with TensorFlow.js directly or following some other tutorial that needs TensorFlow.js to run.

**Your Project**

← index.html

We simply have a script tag in our HTML to grab the latest version of TensorFlow.js

In this case we simply reference the following:

```
<script src="https://cdn.jsdelivr.net/npm@tensorflow/tfjs/dist/tf.min.js">
```

However, if you want to pull in a particular version of TensorFlow.js you can do so like this:

```
<script src="https://cdn.jsdelivr.net/npm@tensorflow/tfjs@1.4.0/dist/
```

Optionally, if you want to include our TF.js visualization library you can do so using this import:

## 1. Edit HTML file:

The screenshot shows the Glitch interface with the project 'internal-dust-valley'. The main content area displays the index.html file, which contains the following code:

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title>Transfer Learning - TensorFlow.js</title>
5    <meta charset="utf-8">
6    <meta http-equiv="X-UA-Compatible" content="IE=edge">
7    <meta name="viewport" content="width=device-width, initial-scale=1">
8    <!-- Import the webpage's stylesheet -->
9    <link rel="stylesheet" href="/style.css">
10   </head>
11   <body>
12     <h1>Make your own "Teachable Machine" using Transfer Learning with MobileNet v3 in TensorFlow.js using saved graph model from TFHub.</h1>
13
14     <p id="status">Awaiting TF.js load</p>
15
16     <video id="webcam" autoplay muted></video>
17
18     <button id="enableCam">Enable Webcam</button>
19     <button class="dataCollector" data-ihot="0" data-name="Class 1">Gather Class 1 Data</button>
20     <button class="dataCollector" data-ihot="1" data-name="Class 2">Gather Class 2 Data</button>
21     <button id="train">Train & Predict!</button>
22     <button id="reset">Reset</button>
23
24     <!-- Import TensorFlow.js library -->
25     <script src="https://cdn.jsdelivr.net/npm@tensorflow/tfjs@3.11.0/dist/tf.min.js" type="text/javascript"></script>
26
27     <!-- Import the page's JavaScript to do some stuff -->
28     <script type="module" src="/script.js"></script>
29
30   </body>
31 </html>

```

## 2. Add CSS changes in style.css:

 Glitch

Thanks for joining us on the edge! We can't wait to show you what's possible. Watch your inbox for an invitation to try things out. [Learn more](#)



**internal-dust-valley**

- Settings
- Assets
- Files

LICENSE.md  
README.md  
index.html  
script.js  
style.css

**style.css** PRETTIER

```

28< body {
29   font-family: helvetica, arial, sans-serif;
30   margin: 2em;
31 }
32
33 h1 {
34   font-style: italic;
35   color: #FF6F00;
36 }
37
38 video {
39   clear: both;
40   display: block;
41   margin: 10px;
42   background: #000000;
43   width: 640px;
44   height: 480px;
45 }
46 .removed {
47   display: none;
48 }
49
50 #status {
51   font-size: 150%;
52 }
53

```

 Glitch

Thanks for joining us on the edge! We can't wait to show you what's possible. Watch your inbox for an invitation to try things out. [Learn more](#)



**internal-dust-valley**

- Settings
- Assets
- Files

LICENSE.md  
README.md  
index.html  
script.js  
style.css

**style.css** PRETTIER

```

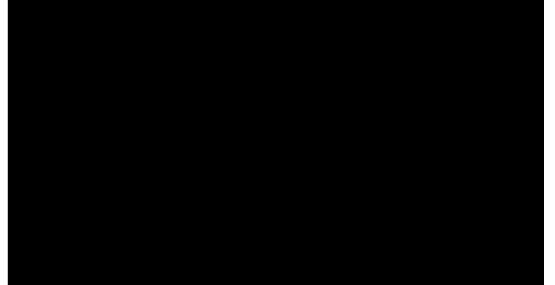
34 * Limitations under the License.
35 * -----
36 /**
37 */
38 /* CSS files add styling rules to your content */
39
40 body {
41   font-family: helvetica, arial, sans-serif;
42   margin: 2em;
43 }
44
45 h1 {
46   font-style: italic;
47   color: #FF6F00;
48 }
49
50 video {
51   clear: both;
52   display: block;
53   margin: 10px;
54   background: #000000;
55   width: 640px;
56   height: 480px;
57 }
58
59 button {
60   padding: 10px;
61   float: left;
62   margin: 5px 3px 5px 10px;
63 }
64
65 .removed {
66   display: none;
67 }
68
69 #status {
70

```

internal-dust-valley.glitch.me/

A "Teachable Machine" using Transfer Learning with MobileNet v3 in TensorFlow.js using saved graph model from TFHub.

Loaded TensorFlow.js - version: 3.11.0



### 3. Script.js changes:

glitch.com/edit/#!/internal-dust-valley?path=script.js%3A22%3A0

Glitch script.js Remix Share

Thanks for joining us on the edge! We can't wait to show you what's possible. Watch your inbox for an invitation to try things out. [Learn more](#)

**internal-dust-valley**

- Settings
- Assets
- Files

LICENSE.md  
README.md  
index.html  
script.js  
style.css

```

/*
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
const status = document.getElementById('status');
const STATUS = document.getElementById('status');
const VIDEO = document.getElementById('webcam');
const ENABLE_CAM_BUTTON = document.getElementById('enableCam');
const RESET_BUTTON = document.getElementById('reset');
const TRAIN_BUTTON = document.getElementById('train');
const MOBILE_NET_INPUT_WIDTH = 224;
const MOBILE_NET_INPUT_HEIGHT = 224;
const STOP_DATA_GATHER = -1;
const CLASS_NAMES = [];
if (status) {
  status.innerText = 'Loaded TensorFlow.js - version: ' + tf.version.tfjs;
}

```

glitch.com/edit/#!/internal-dust-valley?path=script.js

Glitch script.js Remix Share

Thanks for joining us on the edge! We can't wait to show you what's possible. Watch your inbox for an invitation to try things out. [Learn more](#)

**internal-dust-valley**

- Settings
- Assets
- Files

LICENSE.md  
README.md  
index.html  
script.js  
style.css

```

/*
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
const status = document.getElementById('status');
const STATUS = document.getElementById('status');
const VIDEO = document.getElementById('webcam');
const ENABLE_CAM_BUTTON = document.getElementById('enableCam');
const RESET_BUTTON = document.getElementById('reset');
const TRAIN_BUTTON = document.getElementById('train');
const MOBILE_NET_INPUT_WIDTH = 224;
const MOBILE_NET_INPUT_HEIGHT = 224;
const STOP_DATA_GATHER = -1;
const CLASS_NAMES = [];
let mobilenet = undefined;
let gatherDataState = STOP_DATA_GATHER;
let videoPlaying = false;
let trainingDataInputs = [];
let trainingDataOutputs = [];
let examplesCount = 0;
let predict = false;
ENABLE_CAM_BUTTON.addEventListener('click', enableCam);
TRAIN_BUTTON.addEventListener('click', trainAndPredict);
RESET_BUTTON.addEventListener('click', reset);
function enableCam() {
  // TODO: Fill this out later in the codelab!
}

```

A "Teachable Machine" using Transfer Learning with MobileNet v3 in TensorFlow.js using saved graph model from TFHub.

Awaiting TF.js load

#### 4. Load MobileNet model:

glitch.com/edit/#!/internal-dust-valley?path=script.js%3A87%3A28

script.js PRETTIER

```

1 /**
2 * @license
3 * Copyright 2018 Google LLC. All Rights Reserved.
4 * Licensed under the Apache License, Version 2.0 (the "License");
5 * you may not use this file except in compliance with the License.
6 * You may obtain a copy of the License at
7 *
8 * http://www.apache.org/licenses/LICENSE-2.0
9 *
10 * Unless required by applicable law or agreed to in writing, software
11 * distributed under the License is distributed on an "AS IS" BASIS,
12 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13 * See the License for the specific language governing permissions and
14 * limitations under the License.
15 * =====
16 */
17
18 const status = document.getElementById('status');
19 const STATUS = document.getElementById('status');
20 const VIDEO = document.getElementById('webcam');
21 const ENABLE_CAM_BUTTON = document.getElementById('enableCam');
22 const RESET_BUTTON = document.getElementById('reset');
23 const TRAIN_BUTTON = document.getElementById('train');
24 const MOBILE_NET_INPUT_WIDTH = 224;
25 const MOBILE_NET_INPUT_HEIGHT = 224;
26 const STOP_DATA_GATHER = -1;
27 const CLASS_NAMES = [];
28
29 let mobilenet = undefined;
30 let gatherDataState = STOP_DATA_GATHER;
31 let videoPlaying = false;
32 let trainingDataInputs = [];
33 let trainingDataOutputs = [];
34 let examplesCount = [];
35 let predict = false;
36
37
38 }
39
40 // Call the function immediately to start loading.
41 LoadMobileNetFeatureModel();
42
43 let model = tf.sequential();
44 model.add(tf.layers.dense({inputShape: [1024], units: 128, activation: 'relu'}));
45 model.add(tf.layers.dense({units: CLASS_NAMES.length, activation: 'softmax'}));
46 model.summary();
47
48 // Compile the model with the defined optimizer and specify a loss function to use
49 model.compile({
50   optimizer: 'adam',
51   // Use the correct loss function. If 2 classes of data, must use binaryCrossent
52   // Else categoricalCrossentropy is used if more than 2 classes.
53   loss: (CLASS_NAMES.length === 2) ? 'binaryCrossentropy': 'categoricalCrossentro
54   // As this is a classification problem you can record accuracy in the logs too!
55   metrics: ['accuracy']
56 });
57
58
59 
```

internal-dust-valley.glitch.me/

A "Teachable Machine" using Transfer Learning with MobileNet v3 in TensorFlow.js using saved graph model from TFHub.

MobileNet v3 loaded successfully!

## 5. Defined new model:

glitch.com/edit/#!/internal-dust-valley?path=script.js%3A87%3A22

script.js PRETTIER

```

84 }
85
86 // Call the function immediately to start loading.
87 LoadMobileNetFeatureModel();
88
89 let model = tf.sequential();
90 model.add(tf.layers.dense({inputShape: [1024], units: 128, activation: 'relu'}));
91 model.add(tf.layers.dense({units: CLASS_NAMES.length, activation: 'softmax'}));
92 model.summary();
93
94 // Compile the model with the defined optimizer and specify a loss function to use
95 model.compile({
96   optimizer: 'adam',
97   // Adam changes the learning rate over time which is useful.
98   // Use the correct loss function. If 2 classes of data, must use binaryCrossent
99   // Else categoricalCrossentropy is used if more than 2 classes.
100   loss: (CLASS_NAMES.length === 2) ? 'binaryCrossentropy': 'categoricalCrossentro
101   // As this is a classification problem you can record accuracy in the logs too!
102   metrics: ['accuracy']
103 });
104 
```

internal-dust-valley.glitch.me/

with MobileNet v3 in TensorFlow.js using saved graph model from TFHub.

MobileNet v3 loaded successfully!

## 6. Enabled webcam:

glitch.com/edit/#!/internal-dust-valley?path=script.js%3A44%3A1

Glitch script.js PRETTIER internal-dust-valley.glitch.me/ Update

Thanks for joining us on the edge! We can't wait to show you what's possible. Watch your inbox for an invitation to try things out. [Learn more](#)

**internal-dust-valley**

- Settings
- Assets
- Files

LICENSE.md  
README.md  
index.html  
script.js  
style.css

```

104     console.log(answer.shape);
105   });
106 
107 // Call the function immediately to start loading.
108 loadMobileNetFeatureModel();
109 
110 let model = tf.sequential();
111 model.add(tf.layers.dense({inputShape: [1024], unit
112 model.add(tf.layers.dense({units: CLASS_NAMES.length
113 model.summary());
114 
115 // Compile the model with the defined optimizer and
116 model.compile({
117   // Adam changes the learning rate over time which
118   // optimizer: "adam",
119   // Use the correct loss function. If 2 classes or
120   // Else categoricalCrossentropy is used if more than
121   // loss: (CLASS_NAMES.length === 2) ? 'binaryCrosser
122   // As this is a classification problem you can re
123   // metrics: ['accuracy']
124 });

```

internal-dust-valley.glitch.me/ MobileNet v3 loaded successfully!

Gather Class 1 Data Gather Class 2 Data Train & Predict! Reset

## Video:

glitch.com/edit/#!/internal-dust-valley?path=script.js%3A210%3A1

Glitch script.js PRETTIER internal-dust-valley.glitch.me/ Update

Thanks for joining us on the edge! We can't wait to show you what's possible. Watch your inbox for an invitation to try things out. [Learn more](#)

**internal-dust-valley**

- Settings
- Assets
- Files

LICENSE.md  
README.md  
index.html  
script.js  
style.css

```

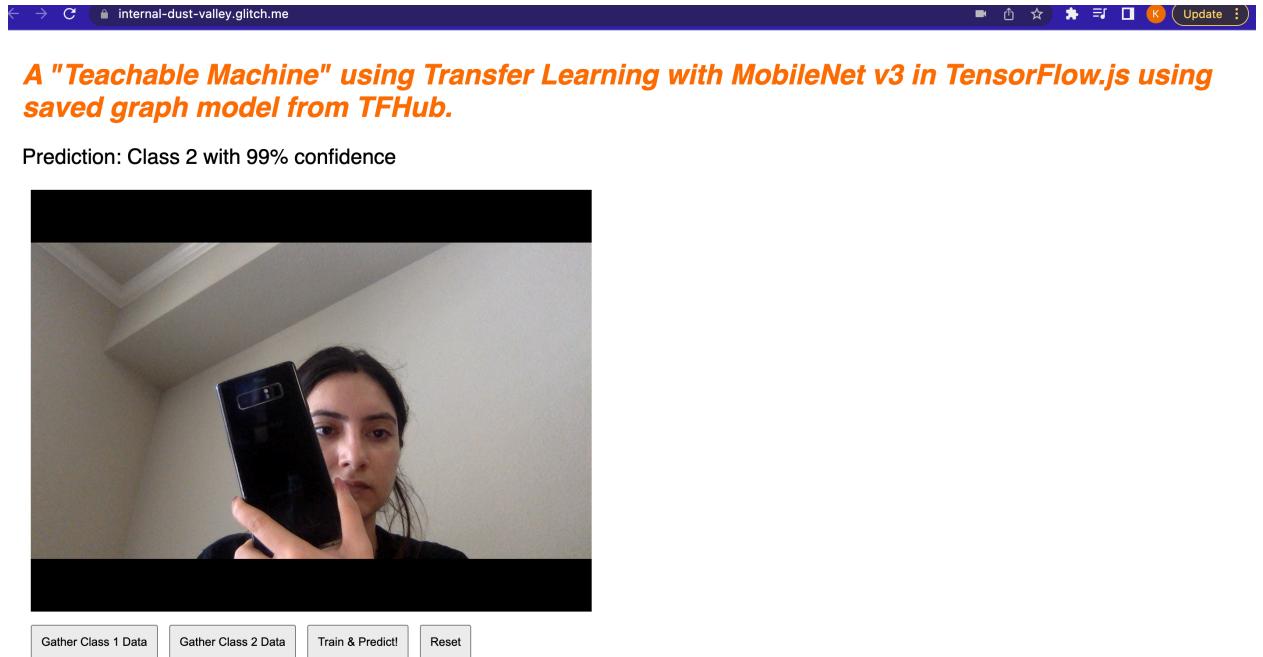
104 let trainingDataOutputs = [];
105 STATUS.innerText = 'No data collected';
106 
107 console.log('Tensors in memory: ' + tf.memory().numTensors);
108 
109 let dataCollectorButtons = document.querySelectorAll('button.dataCollector');
110 for (let i = 0; i < dataCollectorButtons.length; i++) {
111   dataCollectorButtons[i].addEventListener('mousedown', gatherDataForClass);
112   dataCollectorButtons[i].addEventListener('mouseup', gatherDataForClass);
113 }
114 
115 // Populate the human readable names for classes.
116 CLASS_NAMES.push(dataCollectorButtons[i].getAttribute('data-name'));
117 
118 
119 action gatherDataForClass() {
120   // TODO: Fill this out later in the codelab!
121   let classNumber = parseInt(this.getAttribute('data-1hot'));
122   gatherDataState = (gatherDataState === STOP_DATA_GATHER) ? classNumber : gatherDataLoop();
123 }
124 
125 let mobilenet = undefined;
126 let gatherDataState = STOP_DATA_GATHER;
127 let videoPlaying = false;
128 let trainingDataInputs = [];
129 let trainingDataOutputs = [];
130 let examplesCount = 0;
131 let predict = false;
132 
133 /*+
134 Loads the MobileNet model and warms it up so ready for use.
135 */
136 
137 function loadMobileNetFeatureModel() {
138   const URL =
139     'https://tfhub.dev/google/tfjs-model/imagenet/mobilenet_v3_small';
140 }

```

internal-dust-valley.glitch.me/ Prediction: Class 2 with 99% confidence

Gather Class 1 Data Gather Class 2 Data Train & Predict! Reset

Prediction tested:



#### d) Webapp and audio training on device : TensorFlow.js - Audio recognition using transfer learning

Live demo: <https://momentous-slime-trader.glitch.me/>  
 Code in glitch: <https://glitch.com/edit/#!/momentous-slime-trader?path=README.md%3A47%3A0>

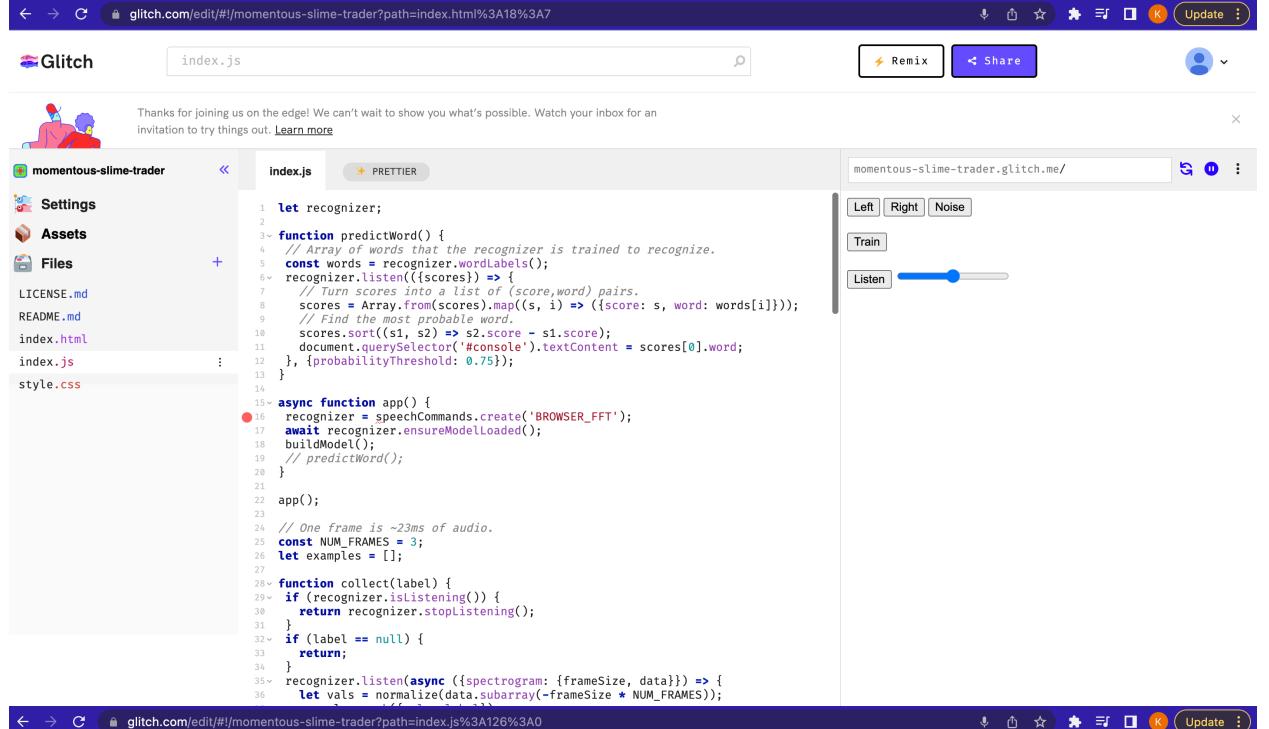
Index.html:

```

<html>
<head>
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>
<script src="https://cdn.jsdelivr.net/npm/@tensorflow-models/speech-commands"></script>
</head>
<body>
<div id="console"></div>
<button id="left" onmousedown="collect(0)" onmouseup="collect(null)">Left</button>
<button id="right" onmousedown="collect(1)" onmouseup="collect(null)">Right</button>
<button id="noise" onmousedown="collect(2)" onmouseup="collect(null)">Noise</button>
<br/><br/>
<button id="train" onclick="train()">Train</button>
<br/><br/>
<button id="listen" onclick="listen()">Listen</button>
<input type="range" id="output" min="0" max="10" step="0.1">
<script src="index.js"></script>
</body>
</html>

```

## Index.js:



The screenshot shows the Glitch editor interface with the file 'index.js' open. The code implements a speech recognition application using TensorFlow.js. It defines a 'recognizer' object, a 'predictWord()' function to find the most probable word from scores, and an 'app()' function to handle audio frames. A 'collect(label)' function uses the recognizer's 'listen()' method to collect examples. The 'buildModel()' function creates a neural network model with two layers: a depthwise convolutional layer and a max pooling layer. The 'moveSlider(labelTensor)' function moves a slider based on the predicted label.

```
let recognizer;
// Array of words that the recognizer is trained to recognize.
const words = recognizer.wordLabels();
recognizer.listen(({scores}) => {
  // Turn scores into a list of (score,word) pairs.
  scores = Array.from(scores).map((s, i) => ({score: s, word: words[i]}));
  // Find the most probable word.
  scores.sort((s1, s2) => s2.score - s1.score);
  document.querySelector('#console').textContent = scores[0].word;
}, {probabilityThreshold: 0.75});

async function app() {
  recognizer = speechCommands.create('BROWSER_FFT');
  await recognizer.ensureModelLoaded();
  buildModel();
  // predictWord();
}

app();

// One frame is ~23ms of audio.
const NUM_FRAMES = 3;
let examples = [];

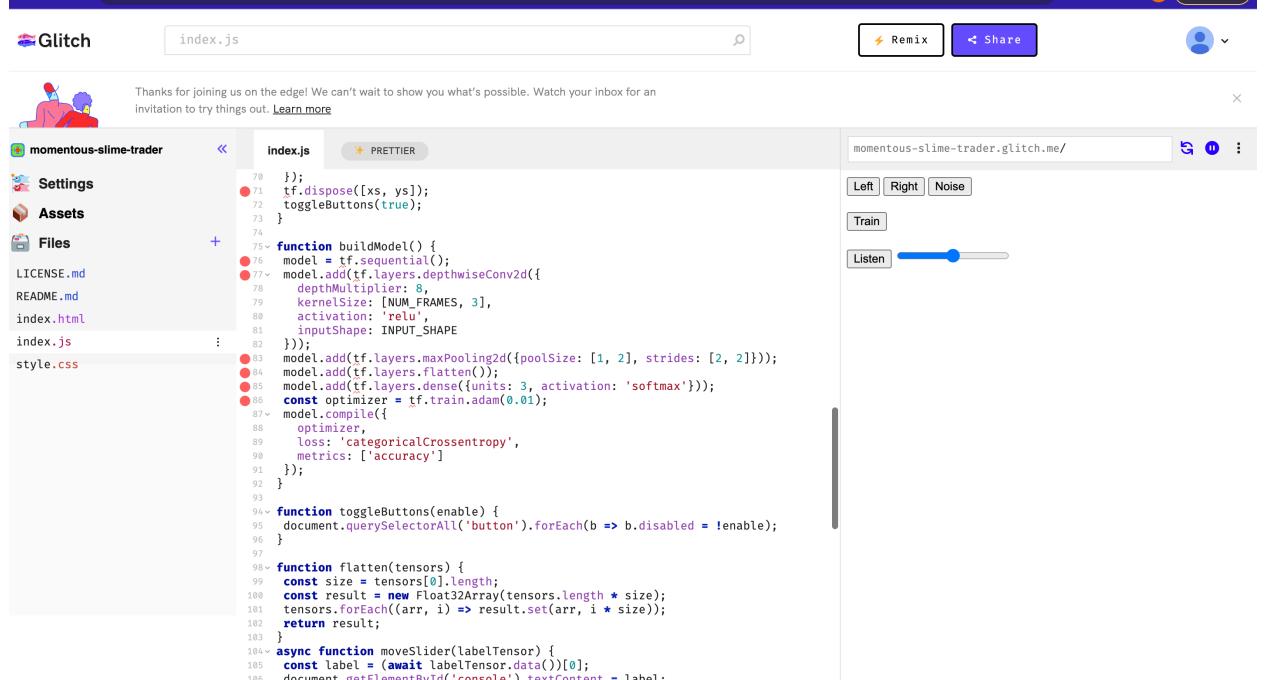
function collect(label) {
  if (recognizer.isListening()) {
    return recognizer.stopListening();
  }
  if (label == null) {
    return;
  }
  recognizer.listen(async ({spectrogram: {frameSize, data}}) => {
    let vals = normalize(data.subarray(-frameSize * NUM_FRAMES));
  });
}

function buildModel() {
  model = tf.sequential();
  model.add(tf.layers.depthwiseConv2d({
    depthMultiplier: 8,
    kernelSize: [NUM_FRAMES, 3],
    activation: 'relu',
    inputShape: INPUT_SHAPE
  }));
  model.add(tf.layers.maxPooling2d({poolSize: [1, 2], strides: [2, 2]}));
  model.add(tf.layers.flatten());
  model.add(tf.layers.dense({units: 3, activation: 'softmax'}));
  const optimizer = tf.train.adam(0.01);
  model.compile({
    optimizer,
    loss: 'categoricalCrossentropy',
    metrics: ['accuracy']
  });
}

function toggleButtons(enable) {
  document.querySelectorAll('button').forEach(b => b.disabled = !enable);
}

function flatten(tensors) {
  const size = tensors[0].length;
  const result = new Float32Array(tensors.length * size);
  tensors.forEach((arr, i) => result.set(arr, i * size));
  return result;
}

async function moveSlider(labelTensor) {
  const label = (await labelTensor.data())[0];
  document.getElementById('console').textContent = label;
}
```



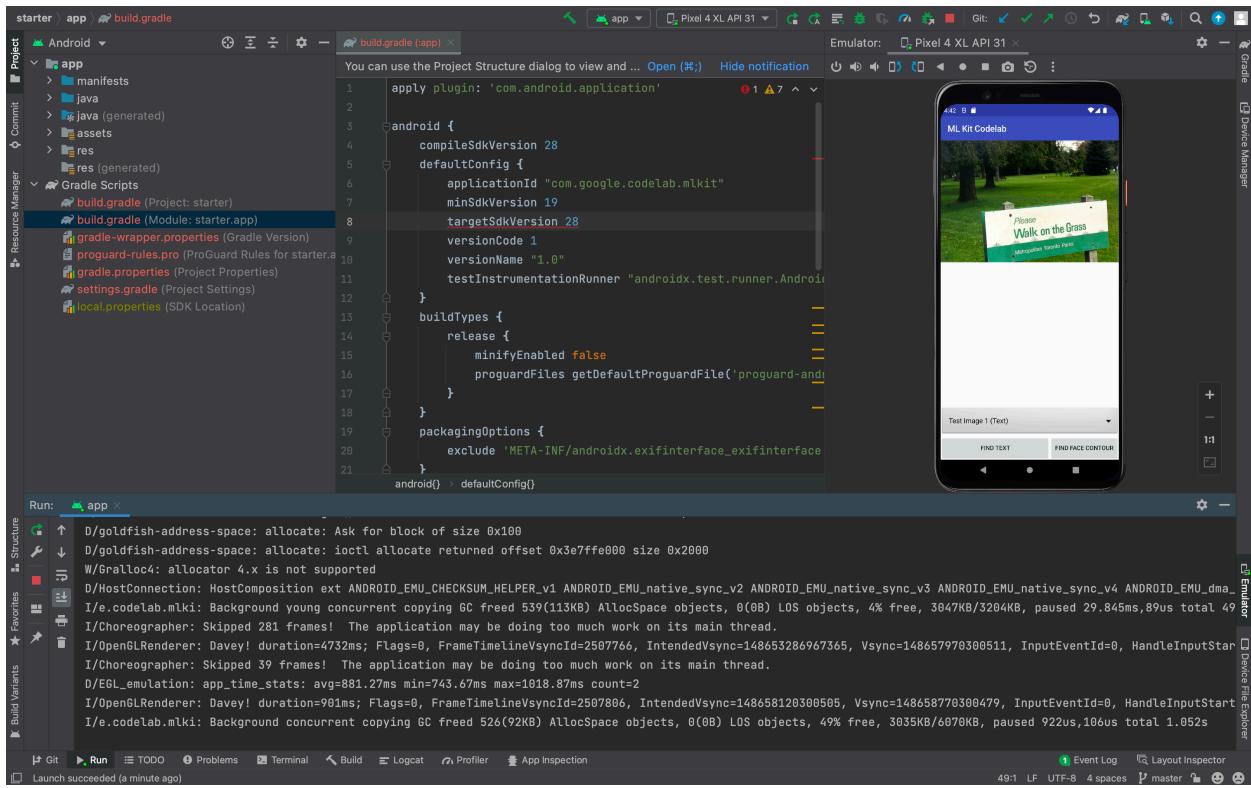
The second screenshot shows the same Glitch editor interface, but with many lines of code highlighted in red. These red highlights are likely indicating syntax errors or warnings in the code, such as undeclared variables or incorrect function calls.

Prediction result:

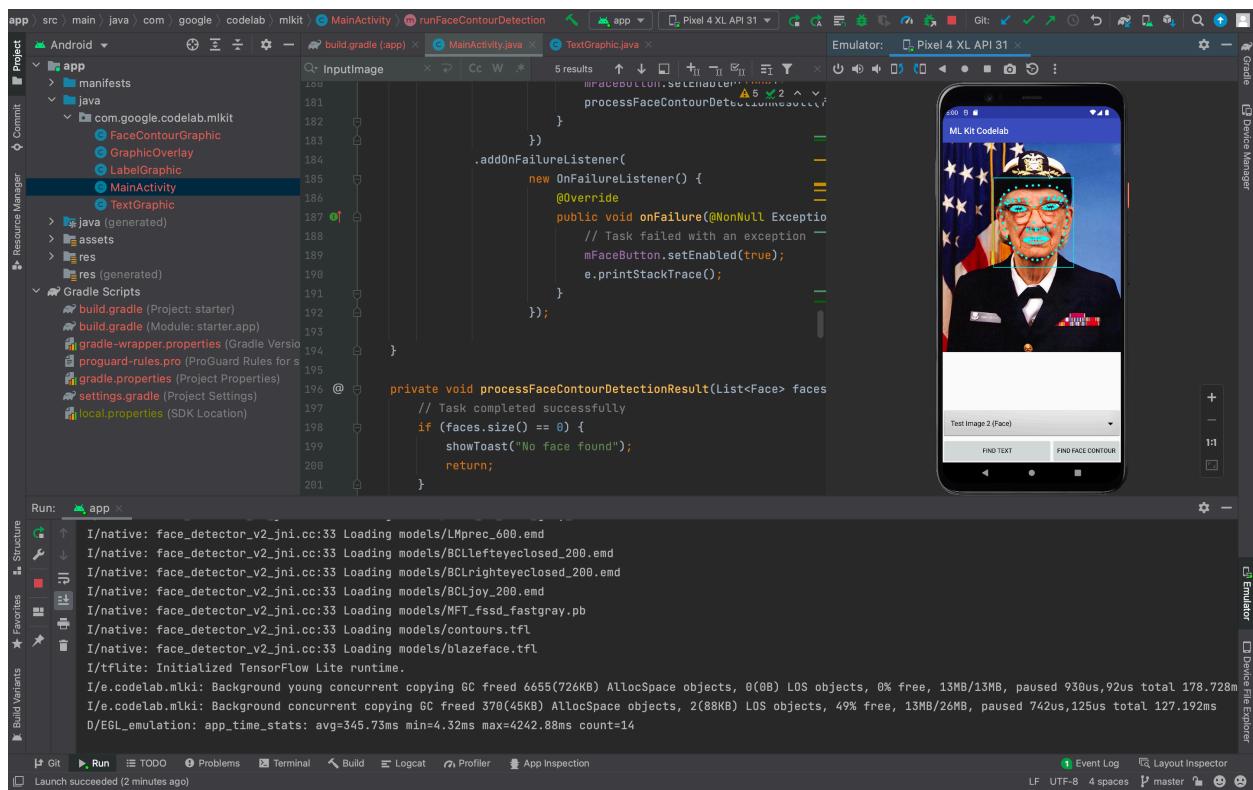


e) Using out of box sdk to do ML ondevice : Use ML Kit to perform [Recognize text and facial features with ML Kit: Android](#) (Links to an external site.), and [Recognize, Identify Language and Translate text with ML Kit and CameraX: Android](#)

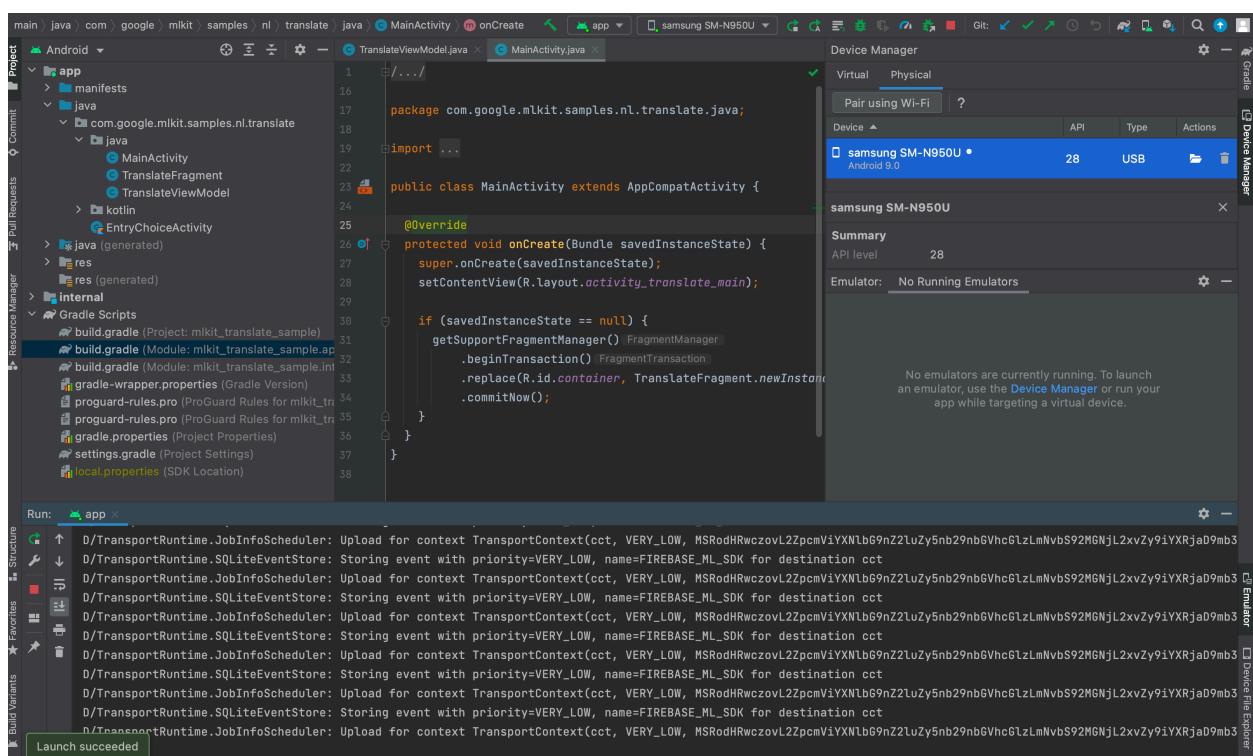
## Recognize text and facial features with ML Kit: Android



## **On-Device face contour detection:**



## Recognize, Identify Language and Translate text with ML Kit and CameraX: Android



Connected Android phone: Samsung Galaxy note as a physical device in Android Studio  
Translated text from English -> Telugu, Hindi, Japanese, French and Portuguese.



The image shows a screenshot of the MLKit-Translate app interface on a mobile device. The screen is split into two main sections, each representing a translation card.

**Left Card (7:19):**

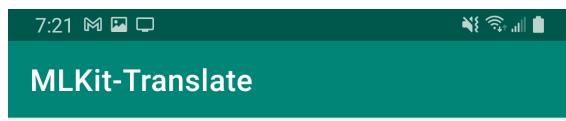
- Source text:** This is ML
- Target language dropdown:** en - Engl.. ▾
- Model selection button:**
- Target language dropdown:** hi - Hindi ▾
- DELETE MODEL**

Below the card:  
Downloaded models: [en, es, fr, hi]  
યહ એમએલ હૈ

**Right Card (7:20):**

- Source text:** Hello world
- Target language dropdown:** en - Engl.. ▾
- Model selection button:**
- Target language dropdown:** te - Telu.. ▾
- DELETE MODEL**

Below the card:  
Downloaded models: [en, es, fr, hi, ja, kn, te]  
హల్లు వరద్



Downloaded models: [en, es, fr, hi, ja, kn, pt, te]

AI é inteligência artificial

