

IESTI01 - TinyML

Image Classification
Edge Impulse Studio

Prof. Marcelo Rovai

July 28th, 2021



Egg Classification Application: Design, Train, Test and Deploy

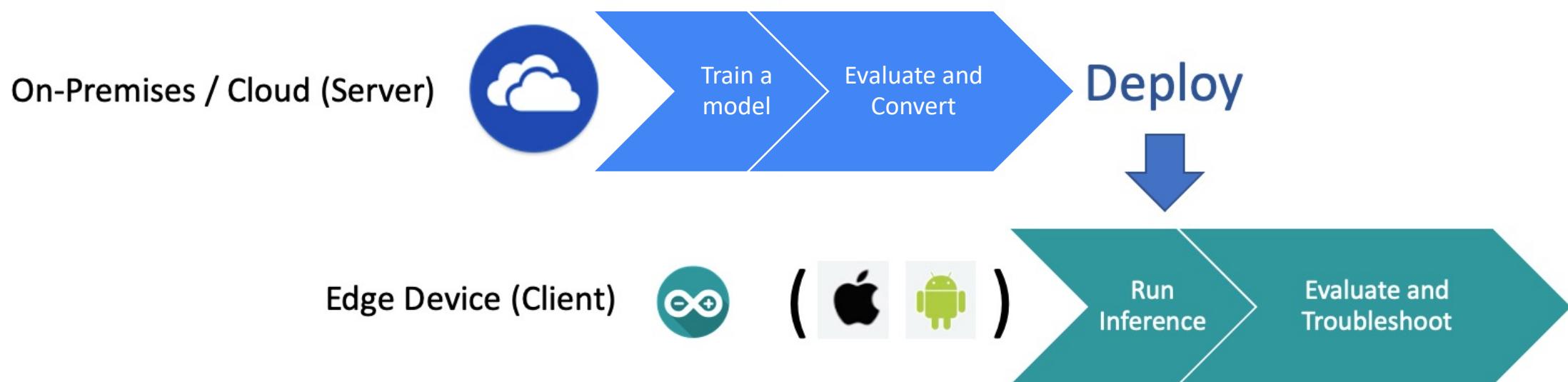
<https://studio.edgeimpulse.com/public/42476/latest>

Thanks to Daniel Situnayake (Edge Impulse)





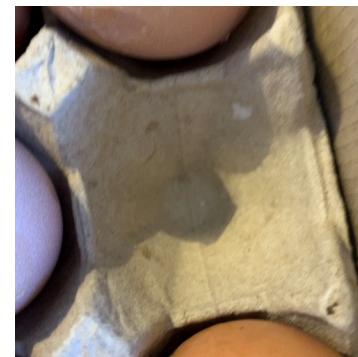
* Feature Extraction



Egg Classification Project

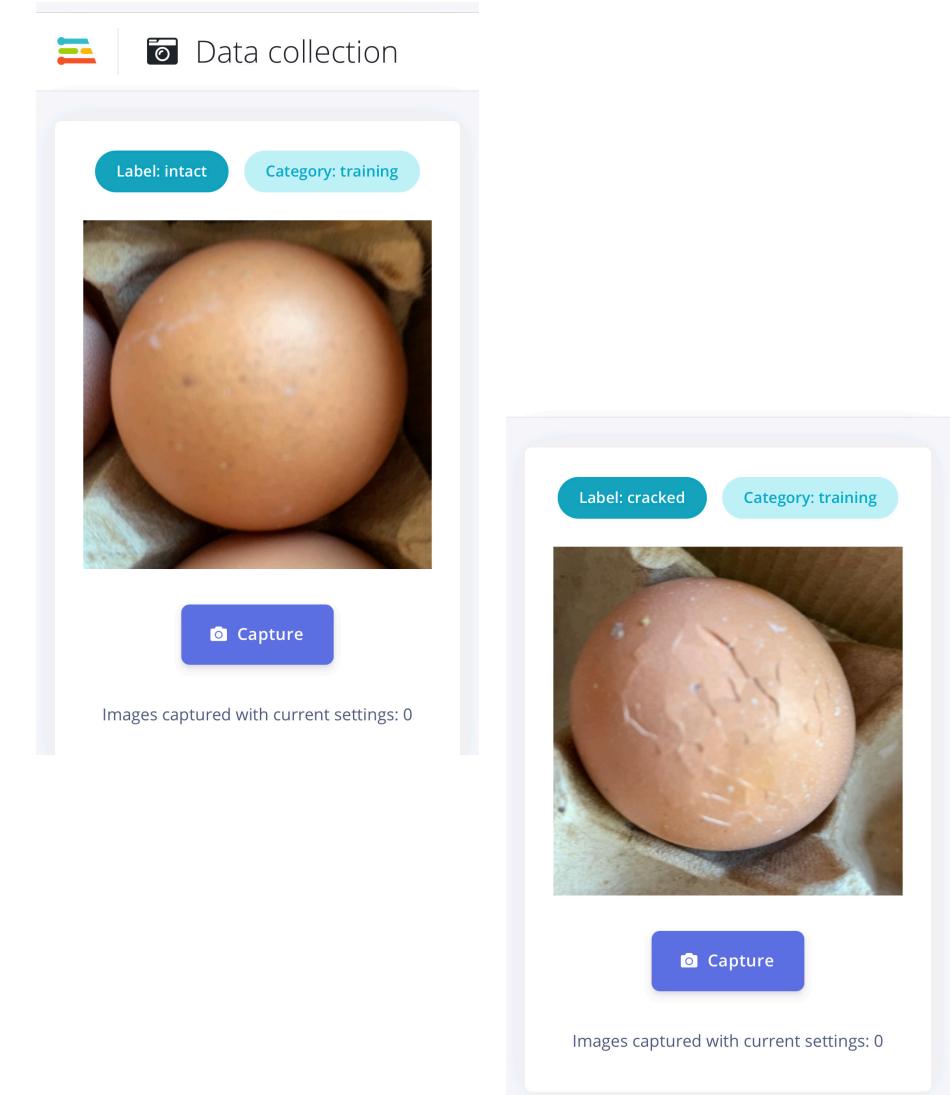
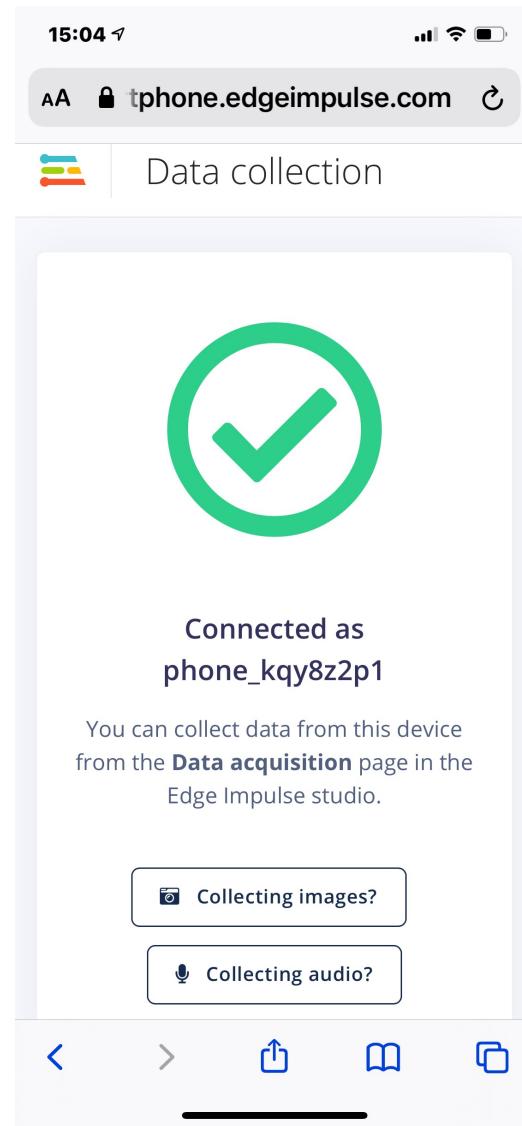
Decide a Goal

- Possible Egg Images:
 - Intact
 - Cracked
 - Empty (no egg)



Collect a dataset,
clean and pre-
process *

- Data Collection:
 - SmartPhone (easy)
 - Dataset (file)
 - Arduino (+ complex)

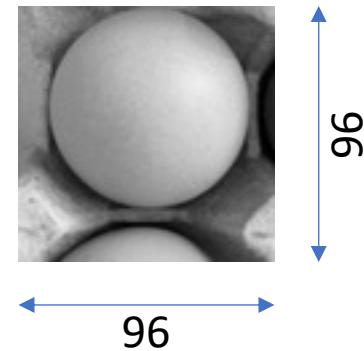
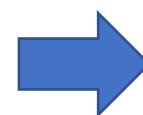


Collect a dataset,
clean and pre-
process *

- Image Pre-Process *:
 - Convert to Grayscale
 - Re-scale 96 x 96

* During Inference, the
OV7675 captures image as:

- QCIF (176 x 144)
- RGB565



IESTI01-T1 - Google Drive | 2011.14858.pdf | Project versioning - Eggs AI - E | Create impulse - Eggs AI Gray | Image Classification with Edge | +

studio.edgeimpulse.com/studio/42476/create-impulse

CREATE IMPULSE (EGGS AI GRayscale)

An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.

Image data

Axes
image

Image width: 96 Image height: 96

Resize mode: Fit shortest axis

For optimal accuracy with transfer learning blocks, use a 96x96 or 160x160 image size.

Image

Name: Image

Input axes: image

Transfer Learning (Images)

Name: Transfer learning

Input features: Image

Output features: 3 (cracked, empty, intact)

Output features

3 (cracked, empty, intact)

Save Impulse

Add a processing block

Add a learning block

© 2020 Edgimpulse Inc. All rights reserved

ei-eggs-ai-arduino....zip ei-eggs-ai-arduino....zip Show All

IESTI01-T1 - Google Drive | 2011.14858.pdf | Project versioning - Eggs AI - Ed | Image - Eggs AI Grayscale - Ed | Image Classification with Edge | +

studio.edgeimpulse.com/studio/42476/dsp/image/3

EDGE IMPULSE

IMAGE (EGGS AI GRayscale)

Parameters Generate features

Raw data

intact.2aq470fb (intact)



Raw features

0xf6fcf3, 0xf3f7f0, 0xeff2e7, 0xebeee2, 0xebefef, 0xe7ae4, 0xe6ece4, 0xe2e5dc, 0xdce0d8, 0xd8d8c...

Parameters

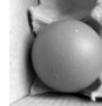
Image

Color depth: Grayscale

Save parameters

DSP result

Image



Processed features

0.9772, 0.9608, 0.9406, 0.9245, 0.9290, 0.9114, 0.9149, 0.8905, 0.8702, 0.8421, 0.8048, 0.8221, 0...

On-device performance

PROCESSING TIME: 11 ms.

PEAK RAM USAGE: 4 KB

© 2020 EdgeImpulse Inc. All rights reserved

ei-eggs-ai-arduino....zip ei-eggs-ai-arduino....zip Show All

IESTI01-T1 - Google Drive | 2011.14858.pdf | Project versioning - Eggs AI - E | Image - Eggs AI Grayscale - Ed | Image Classification with Edge | +

studio.edgeimpulse.com/studio/42476/dsp/image/3/generate-features

EDGE IMPULSE

IMAGE (EGGS AI GRayscale)

Parameters Generate features

Training set

Data in training set: 107 items

Classes: 3 (cracked, empty, intact)

Generate features

Feature explorer (107 samples)

X Axis: Visualization layer 1 | Y Axis: Visualization layer 2 | Z Axis: Visualization layer 3

Legend: cracked (blue), empty (orange), intact (green)

On-device performance

PROCESSING TIME: 11 ms.

PEAK RAM USAGE: 4 KB

© 2020 EdgeImpulse Inc. All rights reserved

ei-eggs-ai-arduino....zip ei-eggs-ai-arduino....zip Show All



Design a model architecture
(i.e.DNN, CNN)

Model

MobileNetV1 96x96 0.25

A pre-trained multi-layer convolutional network designed to efficiently classify images. Uses around 105.9K RAM and 301.6K ROM with default settings and optimizations. Works best with 96x96 input size. Supports both RGB and grayscale.

Image Size

MobileNetV1 96x96 0.2

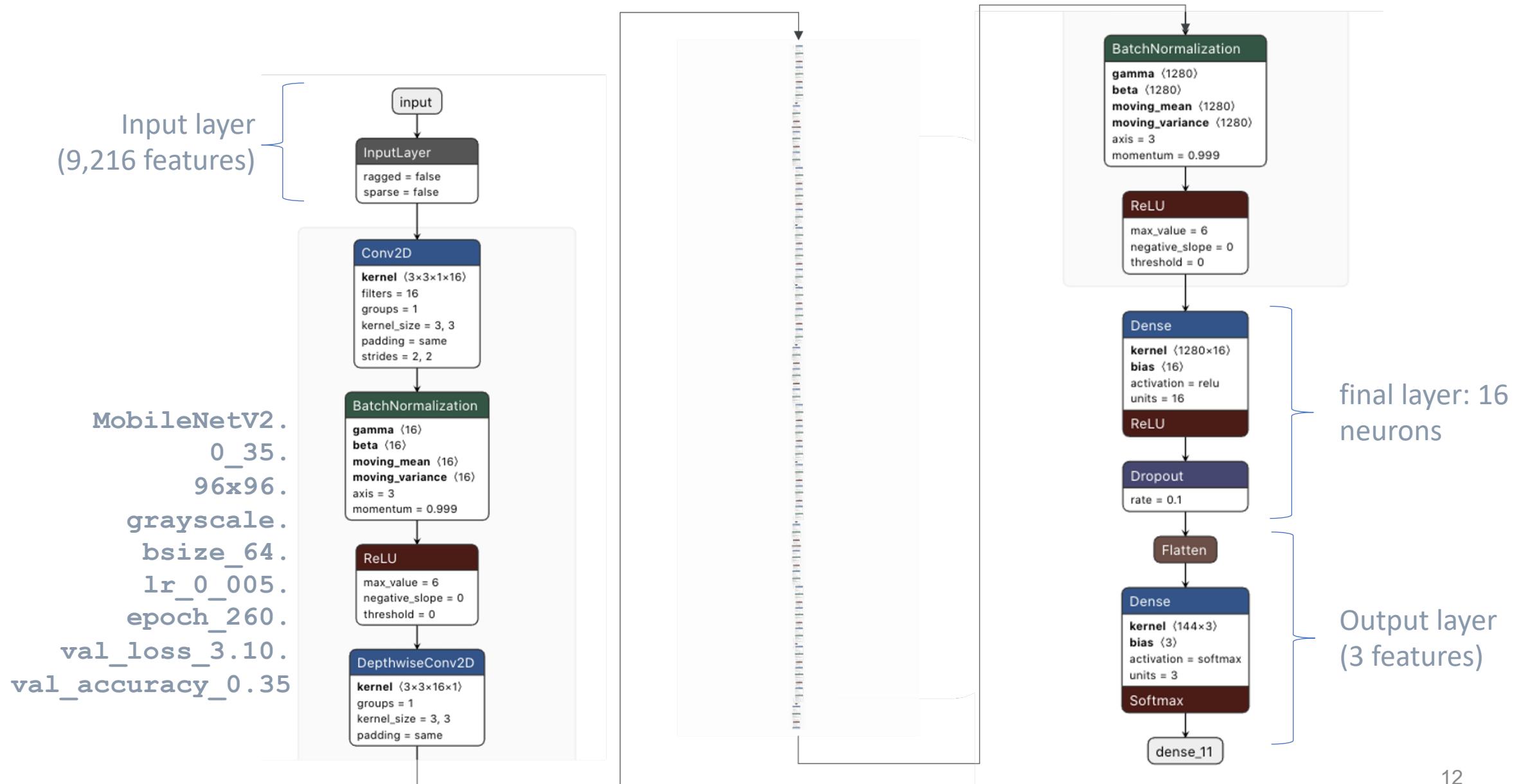
Uses around 83.1K RAM and 218.3K ROM with default settings and optimizations. Works best with 96x96 input size. Supports both RGB and grayscale.

Alpha

MobileNetV1 96x96 0.1

Uses around 53.2K RAM and 101K ROM with default settings and optimizations. Works best with 96x96 input size. Supports both RGB and grayscale.

MobileNetV2 96x96 0.35



IESTI01-T1 - Google Drive | 2011.14858.pdf | Project versioning - Eggs AI - E | Transfer learning - Eggs AI Gra | Image Classification with Edge | +

studio.edgeimpulse.com/studio/42476/learning/keras-transfer-image/9

EDGE IMPULSE

TRANSFER LEARNING (EGGS AI GRayscale)
#1 ▾ Big Model

Neural Network settings

Training settings

- Number of training cycles: 20
- Learning rate: 0.0005
- Data augmentation: checked
- Minimum confidence rating: 0.60

Neural network architecture

Input layer (9,216 features)

MobileNetV2 0.35 (final layer: 16 neurons, 0.1 dropout)

Choose a different model

Output layer (3 features)

Start training

Training output

Very Good Accuracy

Model

Last training performance (validation set)

Model version: Quantized (int8)

ACCURACY		LOSS	
100.0%	0.06		

Confusion matrix (validation set)

	CRACKED	EMPTY	INTACT
CRACKED	100%	0%	0%
EMPTY	0%	100%	0%
INTACT	0%	0%	100%
F1 SCORE	1.00	1.00	1.00

Feature explorer (full training set)

- cracked - correct
- empty - correct
- intact - correct
- intact - incorrect

Visualization layer 3

Visualization layer 2

Visualization layer 1

High Latency

On-device performance

INFERENCING TIME: 7,203 ms.

PEAK RAM USAGE: 300.5K

FLASH USAGE: 584.3K

Train a model

Evaluate and Convert

High Memory

ei-eggs-ai-arduino.zip

ei-eggs-ai-arduino.zip

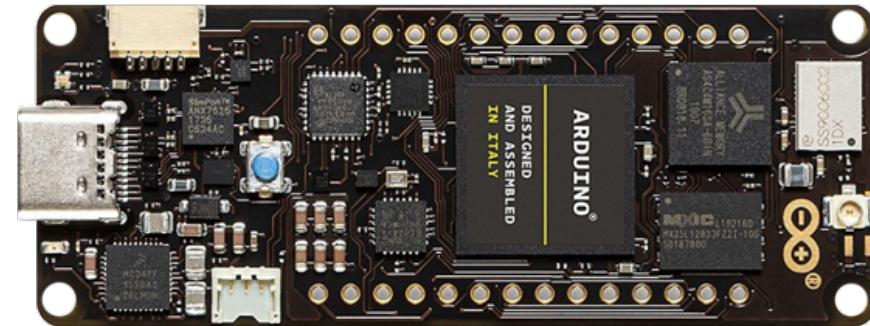
Show All

Benchmark

Arduino Nano 33 BLE Sense Cortex M4F 64 MHz



Problem → High Latency and Nano only has 256KB SRAM



Cortex-M7 216 MHz



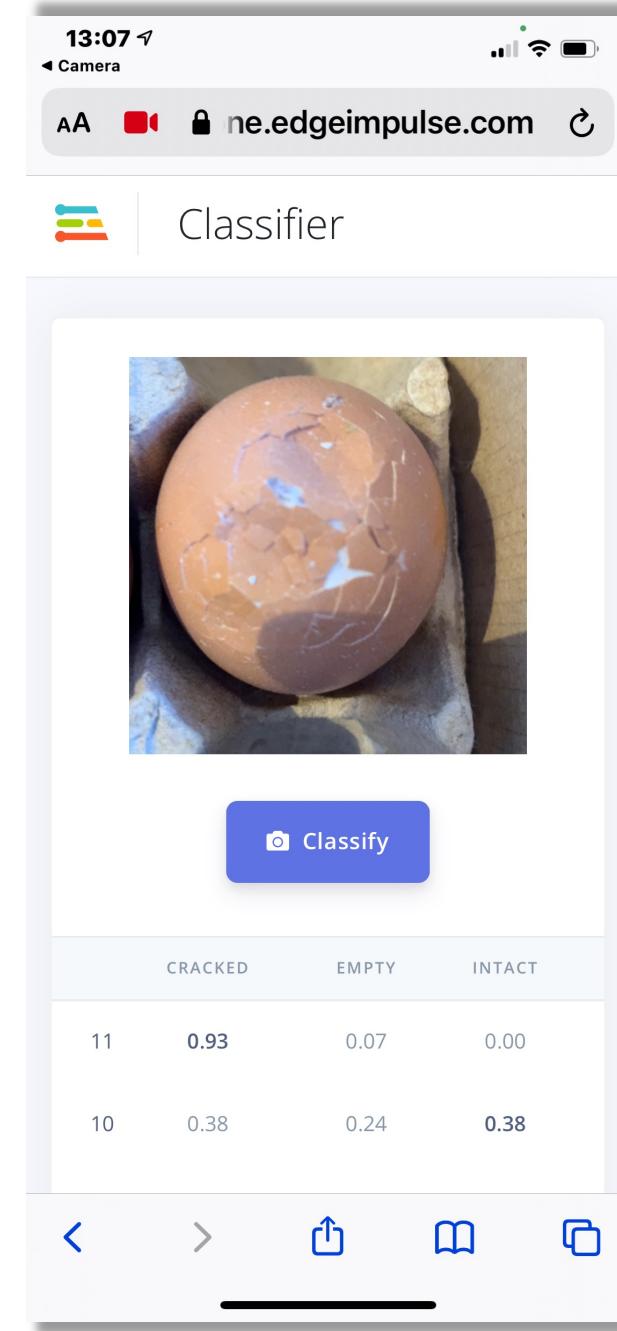
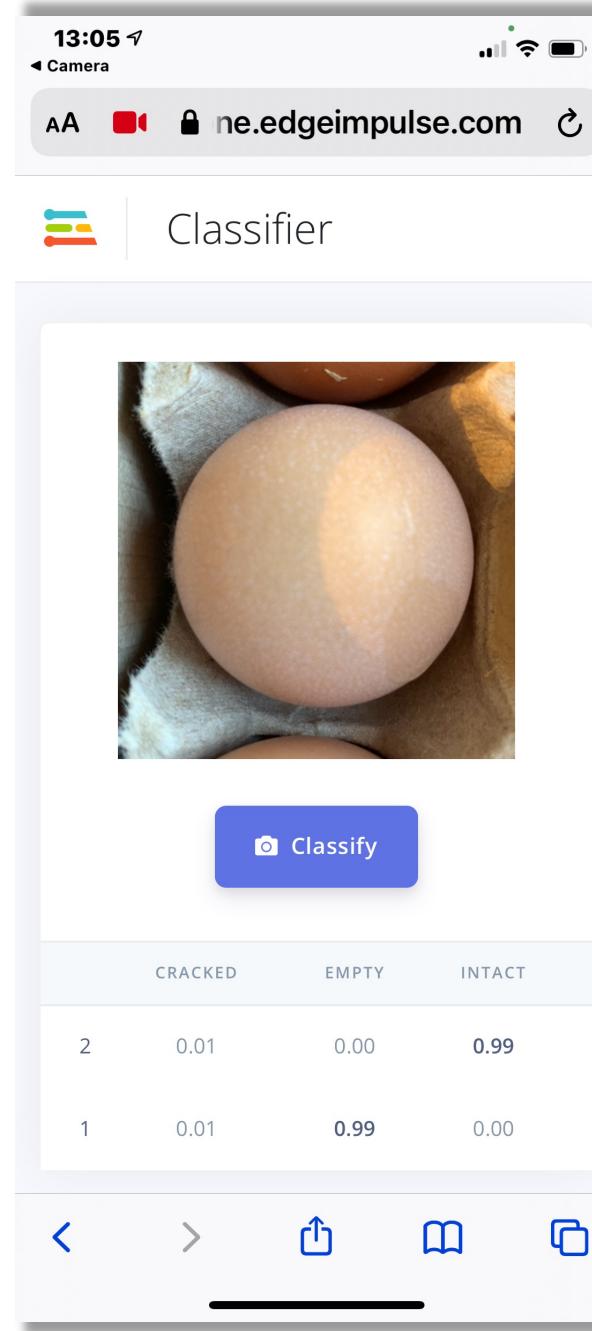
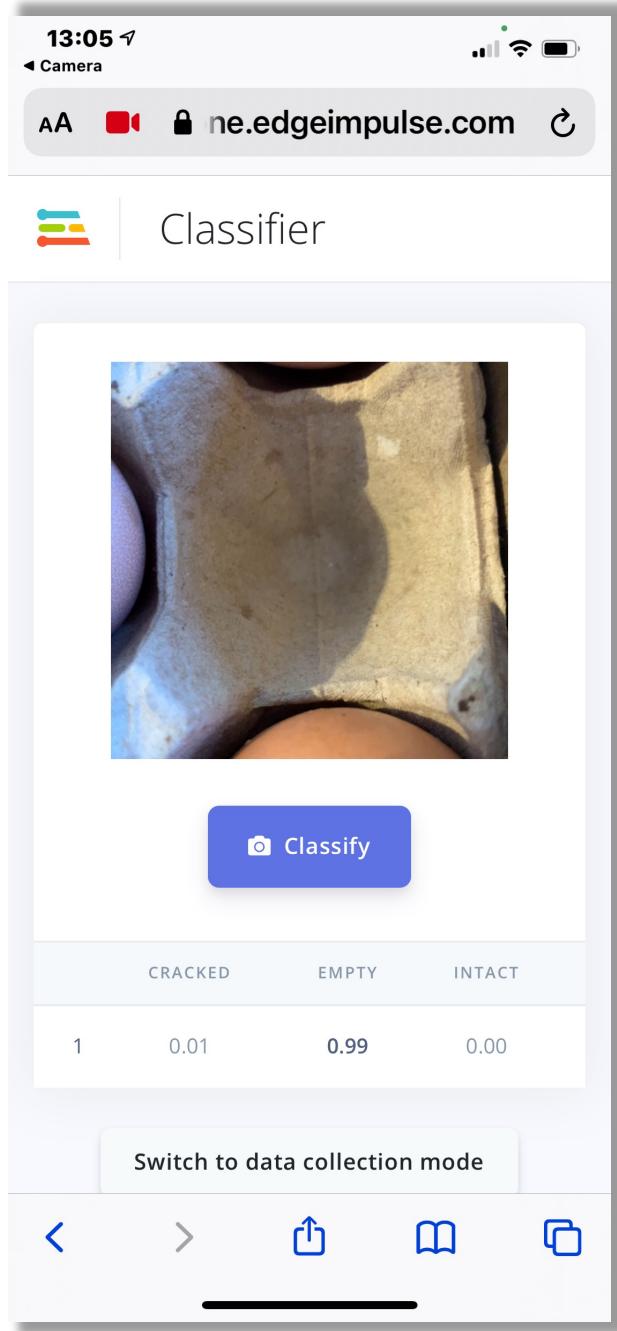
Intel Core i9 2.4GHz



Portenta H7

- Portenta form factor
- STM32H747
- Dual core
 - Arm Cortex M7@480MHz
 - Arm Cortex M4@240MHz
- up to 64 MByte SDRAM
- up to 128 MByte QSPI Flash
- Common Criteria Crypto
- WiFi b/g/n 65Mbps + BT 5.1
- 100Mbit Ethernet
- CAN
- USB-C with DisplayPort output





IEDT01-T1 - Google Drive | 2011.14858.pdf | Project versioning - Eggs AI - E | Transfer learning - Eggs AI Gra | Image Classification with Edge | +

studio.edgeimpulse.com/studio/42476/learning/keras-transfer-image/10

TRANSFER LEARNING (EGGS AI GRayscale)

#2 Small Model ★ Primary version

Neural Network settings

- Training settings
 - Number of training cycles: 20
 - Learning rate: 0.0005
 - Data augmentation: checked
 - Minimum confidence rating: 0.51
- Neural network architecture
 - Input layer (9,216 features)
 - MobileNetV1 0.2 (no final dense layer, 0.1 dropout) (highlighted with a red dashed box)
 - Choose a different model
 - Output layer (3 features)

Start training

Training output

Bad Accuracy

Model

Last training performance (validation set)

ACCURACY 68.2%	LOSS 0.75
--------------------------	---------------------

Confusion matrix (validation set)

	CRACKED	EMPTY	INTACT
CRACKED	88.9%	0%	11.1%
EMPTY	14.3%	85.7%	0%
INTACT	83.3%	0%	16.7%
F1 SCORE	0.70	0.92	0.25

Feature explorer (full training set)

Visualization layer 3

Visualization layer 2

Visualization layer 1

Legend:

- cracked - correct
- empty - correct
- intact - correct
- cracked - incorrect
- empty - incorrect
- intact - incorrect

Bad Latency

On-device performance

INFERENCING TIME: 1,686 ms.

PEAK RAM USAGE: 85.6K

FLASH USAGE: 223.9K

Low Memory

Train a model → Evaluate and Convert → Bad Latency → Bad Accuracy → Low Memory

ei-eggs-ai-arduino.zip ei-eggs-ai-arduino.zip Show All

IESTI01-T1 - Google Drive | 2011.14858.pdf | Project versioning - Eggs AI - E | Model testing - Eggs AI Grayscale | Image Classification with Edge | +

studio.edgeimpulse.com/studio/42476/validation

EDGE IMPULSE

MODEL TESTING (EGGS AI GRAYSCALE)

This lists all test data. You can manage this data through Data acquisition.

Test data

Set the 'expected outcome' for each sample to the desired outcome to automatically score the impulse.

SAMPLE NAME	EXPECTED OUTCOME	LENGTH	ACCURACY	RESULT	⋮
empty.2bbutrt8	empty	-	100%	1 empty	⋮
empty.2bbutqk0	empty	-	0%	1 uncertain	⋮
empty.2bbutpgk	empty	-	0%	1 uncertain	⋮
empty.2bbutojj	empty	-	0%	1 intact	⋮
empty.2bbutjn1	empty	-	0%	1 uncertain	⋮
empty.2bbutino	empty	-	0%	1 uncertain	⋮
empty.2bbutbgk	empty	-	0%	1 uncertain	⋮
empty.2bbut9q5	empty	-	0%	1 intact	⋮
empty.2bbut872	empty	-	0%	1 intact	⋮
empty.2bbut3de	empty	-	100%	1 empty	⋮
empty.2bbut1a4	empty	-	0%	1 uncertain	⋮
empty.2bbusv3u	empty	-	100%	1 empty	⋮
empty.2bbusv3s	empty	-	100%	1 empty	⋮
cracked.2aq4srj6	cracked	-	0%	1 uncertain	⋮
intact.2aq47bri	intact	-	0%	1 empty	⋮
empty.1u0ph6el	empty	-	0%	1 uncertain	⋮

Model testing output

Model testing results

ACCURACY **41.18%** %

	CRACKED	EMPTY	INTACT	UNCERTAIN
CRACKED	47.1%	0%	17.5%	35.3%
EMPTY	0%	25%	18.8%	56.3%
INTACT	11.1%	5.6%	50%	33.3%
F1 SCORE	0.59	0.38	0.55	

Feature explorer ⓘ

- cracked - correct
- empty - correct
- intact - correct
- cracked - incorrect
- empty - incorrect
- intact - incorrect

Test Model

IEDT01-T1 - Google Drive | 2011.14858.pdf | Project versioning - Eggs AI - E | Deployment - Eggs AI Grayscale | Image Classification with Edge | +

studio.edgeimpulse.com/studio/42476/deployment

EDGE IMPULSE

Build firmware
Or get a ready-to-go binary for your development board that includes your impulse.



Eta Compute ECM3532 AI Vision



Himax WE-I Plus



OpenMV library



Sony's Spresense



Linux boards

Select optimizations (optional)

Model optimizations can increase on-device performance but may reduce accuracy. Click below to analyze optimizations and see the recommended choices for your target. Or, just click Build to use the currently selected options.

Enable EON™ Compiler
Same accuracy, up to 50% less memory. Open source.

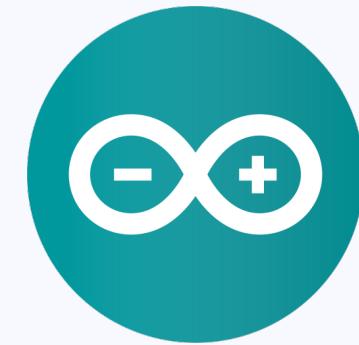
Available optimizations for Transfer learning

Optimization Type	RAM USAGE	LATENCY	CONFUSION MATRIX																
Quantized (int8) ★ <small>Currently selected</small>	85.6K <small>FLASH USAGE 223.9K</small>	1,686 ms <small>ACCURACY 37.25%</small>	<table border="1"> <tr><td>82.4</td><td>0</td><td>0</td><td>17.6</td></tr> <tr><td>43.8</td><td></td><td></td><td>31.3</td></tr> <tr><td>38.9</td><td>16.7</td><td>5.6</td><td>38.9</td></tr> <tr><td>11.1</td><td>5.6</td><td>50</td><td>33.3</td></tr> </table>	82.4	0	0	17.6	43.8			31.3	38.9	16.7	5.6	38.9	11.1	5.6	50	33.3
82.4	0	0	17.6																
43.8			31.3																
38.9	16.7	5.6	38.9																
11.1	5.6	50	33.3																
Unoptimized (float32) <small>Click to select</small>	249.5K <small>FLASH USAGE 580.5K</small>	6,254 ms <small>ACCURACY 41.18%</small>	<table border="1"> <tr><td>47.1</td><td>0</td><td>17.6</td><td>35.3</td></tr> <tr><td>0</td><td>25</td><td>18.8</td><td>56.3</td></tr> <tr><td>11.1</td><td>5.6</td><td>50</td><td>33.3</td></tr> </table>	47.1	0	17.6	35.3	0	25	18.8	56.3	11.1	5.6	50	33.3				
47.1	0	17.6	35.3																
0	25	18.8	56.3																
11.1	5.6	50	33.3																

Estimate for Arduino Nano 33 BLE Sense (Cortex-M4F 64MHz)

Build

Deploy



ei-eggs-ai-arduino....zip

ei-eggs-ai-arduino....zip

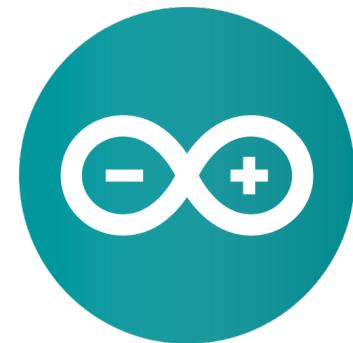
Show All

Egg Classification using Transfer Learning Model

Code Walkthrough!

static_buffer.ino

EI_Image_Classifier_Eggs.ino



Thanks to Jan Jongboom (Edge Impulse) and Jeremy Ellis (@rocksetta)

IESTI01-T1 - Google Drive | 2011.14858.pdf | Image - Eggs AI - Edge Impulse | Image Classification with Edge | + studio.edgeimpulse.com/studio/19868/dsp/image/3

EDGE IMPULSE

IMAGE (EGGS AI)

Parameters Generate features

Raw data

Raw features 

0x553735, 0x452a26, 0x341818, 0x24080b, 0xc0102, 0x1, 0xc0605, 0x281e1b, 0x382a25, 0x583e33, 0x73...

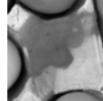
Parameters

Image

Color depth Grayscale

Save parameters

DSP result

Image 

Processed features 

0.2500, 0.1946, 0.1269, 0.0655, 0.0173, 0.0004, 0.0301, 0.1280, 0.1789, 0.2687, 0.3448, 0.4041, 0...

On-device performance 

PROCESSING TIME 11 ms. PEAK RAM USAGE 4 KB

© 2020 EdgeImpulse Inc. All rights reserved

ei-eggs-ai-arduino....zip Show All

MJRoBot (Marcelo Rovai)

A large orange arrow points from the top right towards the "DSP result" section.

```
* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM
* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THIS
* SOFTWARE.
```

```
 */
22
23 /* includes ----- */
24 #include <Eggs_AI_inferencing.h>
25
26 static const float features[] = {
27     // copy raw features here (for example from the 'Live classification' page)
28     // see https://docs.edgeimpulse.com/docs/running-your-edge-impulse-arduino
29     0xf6fcf3, 0xf3f7f0, 0xeff2e7, 0xebeee2, 0xebefe7, 0xe7ea4, 0xe6ece4, 0xe2e
30 };
31
32 /**
33 * @brief      Copy raw feature data in out_ptr
34 *             Function called by inference library
35 *
36 * @param[in]   offset    The offset
37 * @param[in]   length    The length
38 * @param       out_ptr   The out pointer
39 *
40 * @return     0
41 */
42 int raw_feature_get_data(size_t offset, size_t length, float *out_ptr) {
43     memcpy(out_ptr, features + offset, length * sizeof(float));
44     return 0;
45 }
46
```

Testing Model inference (static_buffer.ino)

```
/dev/cu.usbmodem144301
Send

Edge Impulse standalone inferencing (Arduino)
run_classifier returned: 0
Predictions (DSP: 8 ms., Classification: 694 ms., Anomaly: 0 ms.):
[0.27344, 0.54297, 0.18359]
    cracked: 0.27344
    empty: 0.54297
    intact: 0.18359
Edge Impulse standalone inferencing (Arduino)
run_classifier returned: 0
Predictions (DSP: 8 ms., Classification: 694 ms., Anomaly: 0 ms.):
[0.27344, 0.54297, 0.18359]
    cracked: 0.27344
    empty: 0.54297
    intact: 0.18359
Edge Impulse standalone inferencing (Arduino)
run_classifier returned: 0
Predictions (DSP: 8 ms., Classification: 694 ms., Anomaly: 0 ms.):
[0.27344, 0.54297, 0.18359]
    cracked: 0.27344
    empty: 0.54297
    intact: 0.18359

```

```
static_buffer.h
24 #include <Eggs_AI_inferencing.h>
25 #include <TinyMLShield.h>
26
27 static const float features[] = {
28     // copy raw features here (for example from the 'Live classification' page)
29     // see https://docs.edgeimpulse.com/docs/running-your-impulse-arduino
30     0x553735, 0x452a26, 0x341818, 0x24080b, 0xc0102, 0x1, 0xc0605, 0x281e1b, 0x382a25, 0x1
31 };
32
33 /**
34  * @brief      Arduino setup function
35 */
36 void setup() {
37     Serial.begin(115200);
38     while (!Serial);
39
39     Serial.println("IESTI01 - Edge Impulse - Image Inferencing");
40
41     // Initialize TinyML Kit
42     initializeShield();
43
44     // Initialize the OV7675 camera
45     if (!Camera.begin(QCIF, RGB565, 1, OV7675)) {
46         Serial.println("Failed to initialize camera");
47         while (1);
48     }
49 }
```

Done uploading.
Done in 14.803 seconds
reset()

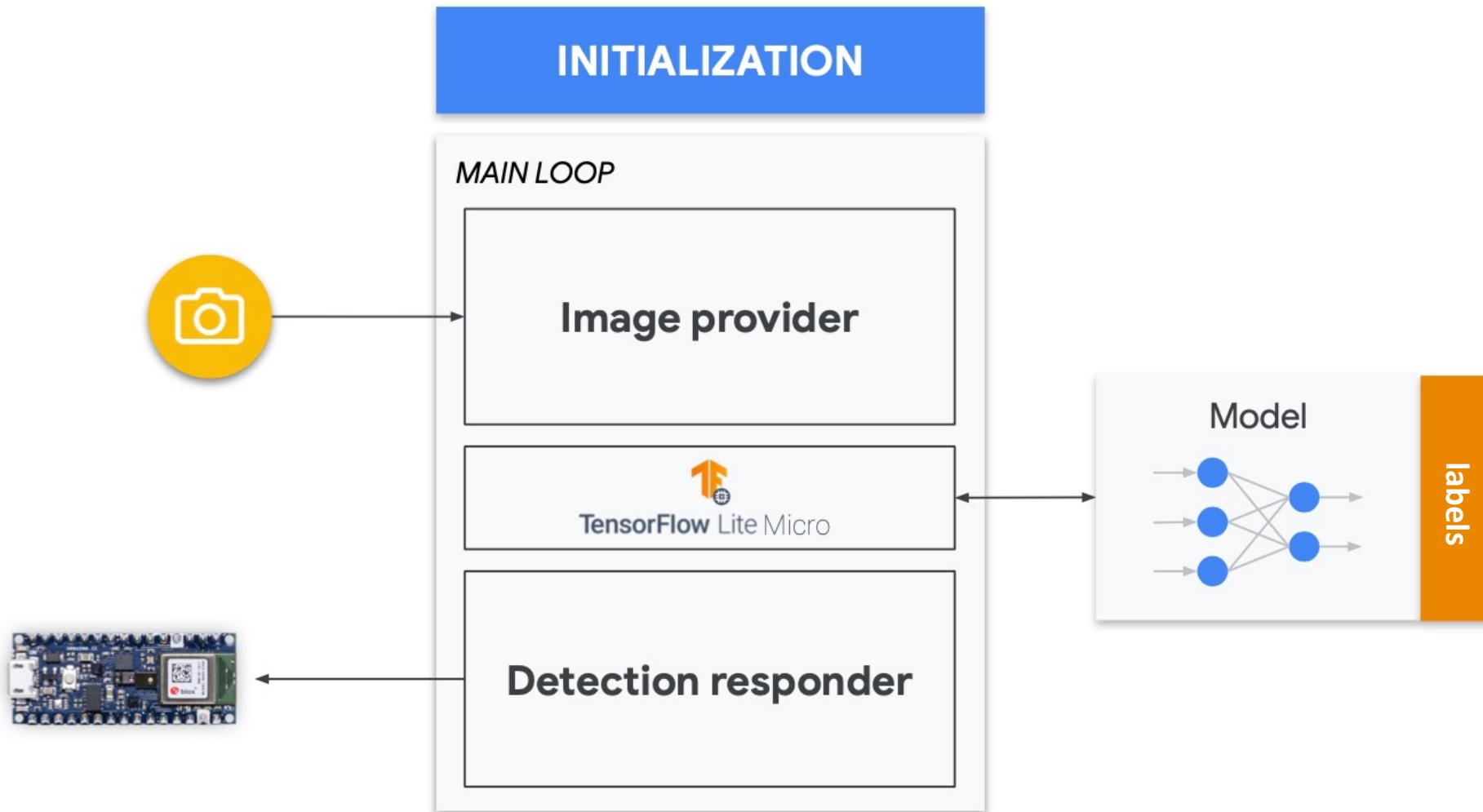
Testing Model inference and Camera initialization (static_buffer.ino)

Function to Copy raw feature data in out_ptr
This function is called by inference library

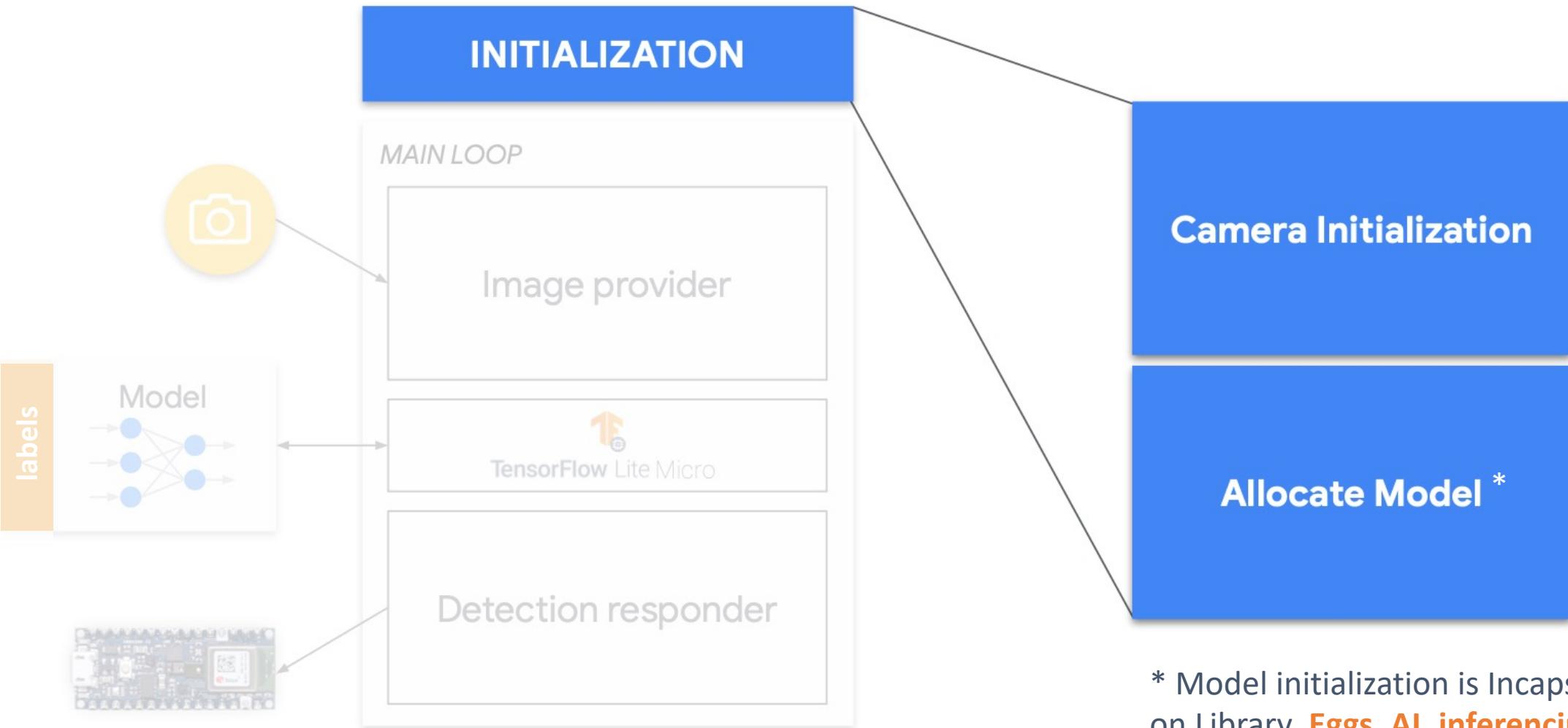
Initialize Kit and Camera

Egg Classification Components

(EI_Image_Classifier_Eggs.ino)



Initialization



* Model initialization is Incapsulated by EI
on Library **Eggs_AI_inferencing.h**

```
/* Includes */
#include <Eggs_AI_inferencing.h>
#include <TinyMLShield.h> // Includes the Arduino_OV767X.h library

/* Image Definitions */

// raw frame buffer from the camera (QCIF grayscale from camera = 176 * 144 * 1)
#define FRAME_BUFFER_COLS 176
#define FRAME_BUFFER_ROWS 144
uint16_t frame_buffer[FRAME_BUFFER_COLS * FRAME_BUFFER_ROWS] = { 0 };

// Resize image cutting out the edges (it is not a true resize)
#define CUTOUT_COLS EI_CLASSIFIER_INPUT_WIDTH
#define CUTOUT_ROWS EI_CLASSIFIER_INPUT_HEIGHT
const int cutout_row_start = (FRAME_BUFFER_ROWS - CUTOUT_ROWS) / 2;
const int cutout_col_start = (FRAME_BUFFER_COLS - CUTOUT_COLS) / 2;

/*
 * Functions to help convert color data
 */

// helper methods to convert from rgb -> 565 and vice versa this one not used
uint16_t rgb_to_565(uint8_t r, uint8_t g, uint8_t b) {
    return ((r >> 3) << 11) | ((g >> 2) << 5) | (b >> 3);
}

// function converts from RGB565b to RGB888 and is used.
void r565_to_rgb(uint16_t color, uint8_t *r, uint8_t *g, uint8_t *b) {
    *r = (color & 0x800) >> 8;
    *g = (color & 0x07E0) >> 3;
    *b = (color & 0x1F) << 3;
}

/*
```

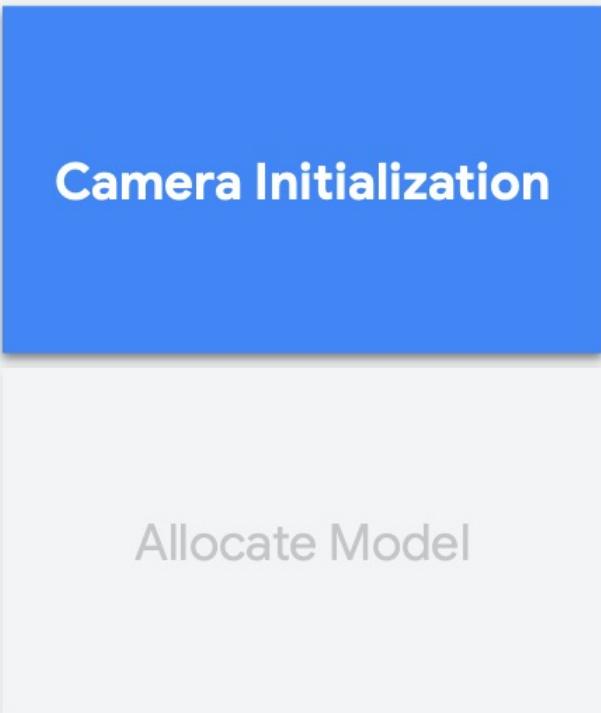
Done in 13.373 seconds
reset()

Including main libraries for inference and Camera handling

Main Image definitions

Auxiliary functions for image format conversion

Initialization



```
Camera.begin(QCIF, RGB565, 1, OV7675)
```

Color

Camera

Resolution

FPS

```
EL_Image_Classifier_Eggs | Arduino 1.8.15

EI_Image_Classifier_Eggs §
107
108 void setup()
109 {
110     Serial.begin(115200);
111     while (!Serial);
112
113     ei_printf("IESTI01 - Edge Impulse - Image Inferencing");
114     // summary of inferencing settings (from model_metadata.h)
115     ei_printf("Inferencing settings:\n");
116     ei_printf("\tNN_INPUT_FRAME_SIZE: %d\n", EI_CLASSIFIER_NN_INPUT_FRAME_SIZE);
117     ei_printf("\tINPUT_WIDTH_COLS: %d\n", EI_CLASSIFIER_INPUT_WIDTH);
118     ei_printf("\tINPUT_HEIGHT_ROWS: %d\n", EI_CLASSIFIER_INPUT_HEIGHT);
119     ei_printf("\tDSP_INPUT_FRAME_SIZE: %d\n", EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE);
120     ei_printf("\tTFLITE_ARENA_SIZE: %d\n", EI_CLASSIFIER_TFLITE_ARENA_SIZE);
121     ei_printf("\tInterval: %.2f ms.\n", (float)EI_CLASSIFIER_INTERVAL_MS);
122     ei_printf("\tFrame size: %d\n", EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE);
123     ei_printf("\tSample length: %d ms.\n", EI_CLASSIFIER_RAW_SAMPLE_COUNT / 16);
124     ei_printf("\tNo. of classes: %d\n", sizeof(ei_classifier_inferencing_categories) /
125               sizeof(ei_classifier_inferencing_categories[0]));
126
127     // Initialize TinyML Kit
128     initializeShield();
129 /*
130  * VGA - 640 x 480
131  * CIF - 352 x 240
132  * QVGA - 320 x 240
133  * QCIF - 176 x 144
134 */
135     // Initialize the OV7675 camera
136     if (!Camera.begin(QCIF, RGB565, 1, OV7675)) {
137         Serial.println("Failed to initialize camera");
138         while (1);
139     }
140 }

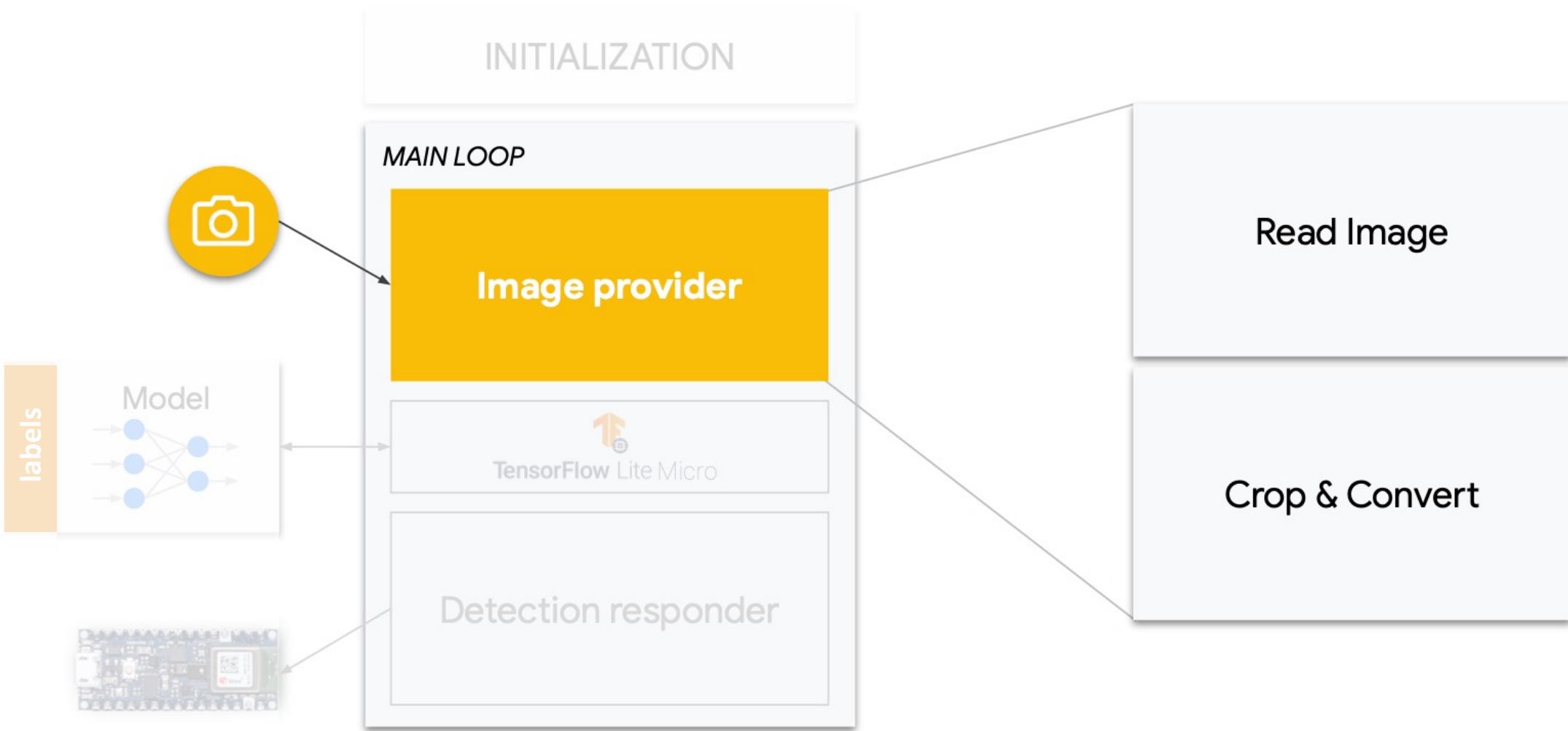
[=====] 98% (83/84 pages)write(addr=0x34,size=0x1000)
writeBuffer(scr_addr=0x34, dst_addr=0x53000, size=0x1000)
[=====] 100% (84/84 pages)
Done in 13.373 seconds
reset()

131
```

Initialize Serial and display main inference settings as Image height/width, TFLM Arena size, etc.

Initialize TinyML Kit and Camera

Pre-processing



Pre-processing

Read Image

Crop & Convert



144



176

```
// Get image from Camera  
Camera.readFrame((uint8_t*)frame_buffer);  
  
// Set up pointer to look after data, crop it and convert it to RGB888  
signal_t signal;  
signal.total_length = CUTOUT_COLS * CUTOUT_ROWS;  
signal.get_data = &cutout_get_data;
```

EL_Image_Classifier_Eggs | Arduino 1.8.15

```

65
66 int cutout_get_data(size_t offset, size_t length, float *out_ptr) {
67     // offset and length naturally operate on the *cutout*,
68     // so we need to cut it out from the real framebuffer
69     size_t bytes_left = length;
70     size_t out_ptr_ix = 0;
71
72     // read byte for byte
73     while (bytes_left != 0) {
74         // find location of the byte in the cutout
75         size_t cutout_row = floor(offset / CUTOUT_COLS);
76         size_t cutout_col = offset - (cutout_row * CUTOUT_COLS);
77
78         // then read the value from the real frame buffer
79         size_t frame_buffer_row = cutout_row + cutout_row_start;
80         size_t frame_buffer_col = cutout_col + cutout_col_start;
81
82         // grab the value and convert to r/g/b
83         uint16_t pixelTemp = frame_buffer[(frame_buffer_row * FRAME_BUFFER_COLS) + frame_buffer_col];
84
85         // This line needed to switch big and little endians
86         uint16_t pixel = (pixelTemp>>8) | (pixelTemp<<8);
87
88         uint8_t r, g, b;
89         r565_to_rgb(pixel, &r, &g, &b);
90
91         // then convert to out_ptr format
92         float pixel_f = (r << 16) + (g << 8) + b;
93         //float pixel_f = (r << 16) | (g << 8) | b;
94         out_ptr[out_ptr_ix] = pixel_f;
95
96         // and go to the next pixel
97         out_ptr_ix++;
98         offset++;
99         bytes_left--;
100    }
101
102    return 0;
103 }

```

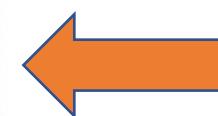
```

[----] 100% (84/84 pages) written to 0x34 (0x34 pages) in 1.16s (0x34, size=0x1000)
writeBuffer(scr_addr=0x34, dst_addr=0x53000, size=0x1000)
[-----] 100% (84/84 pages)
Done in 13.373 seconds
reset()

```

65

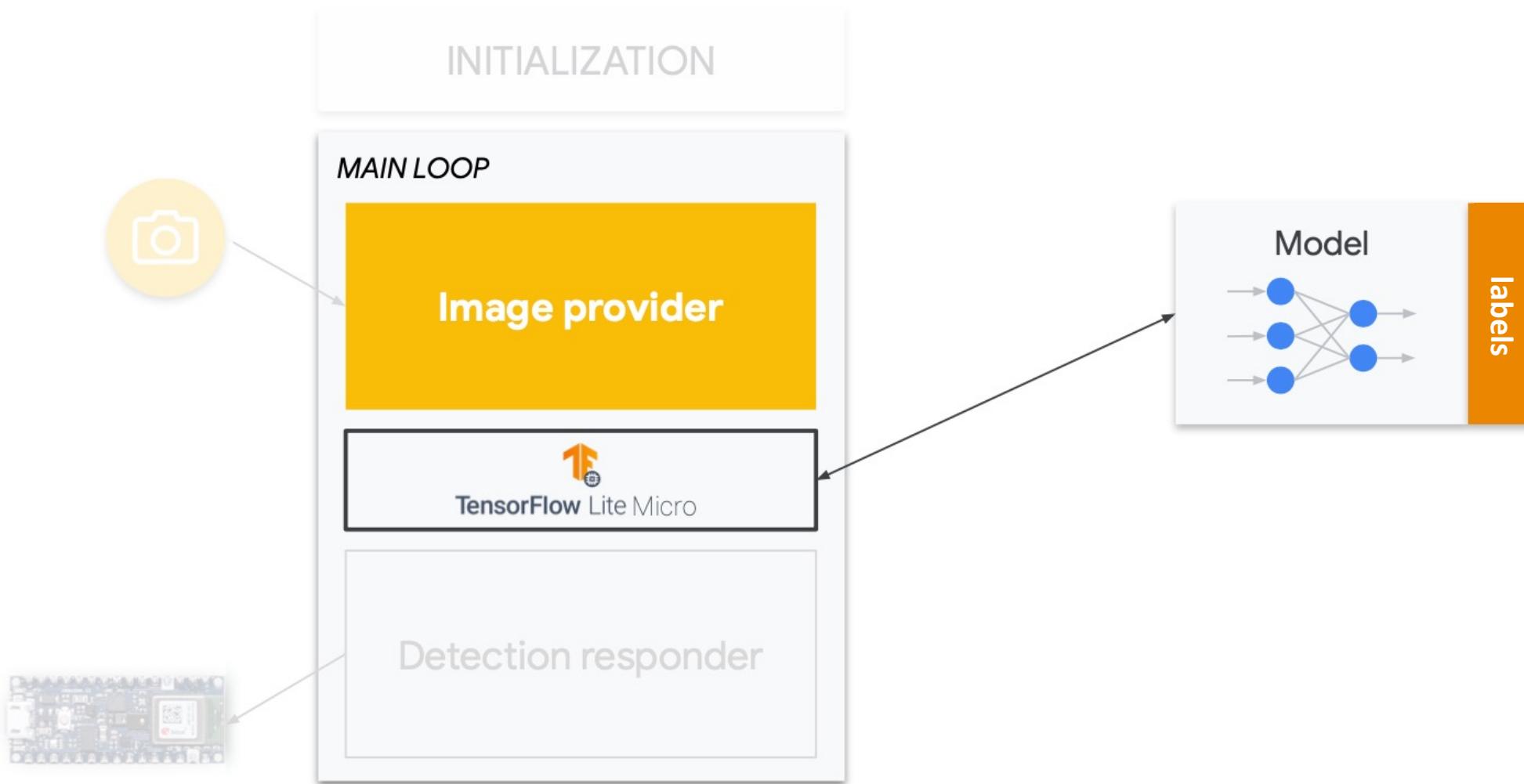
Arduino Nano 33 BLE on /dev/cu.usbmodem144301



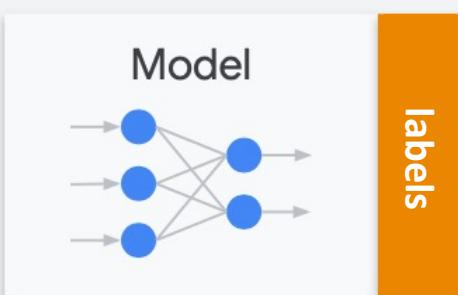
This function is called by the classifier (during loop), to get image data (frame buffer)

NOTE: There is not a separate copy of the cutout here, so we'll read from the frame buffer dynamically

Interpreter + Model



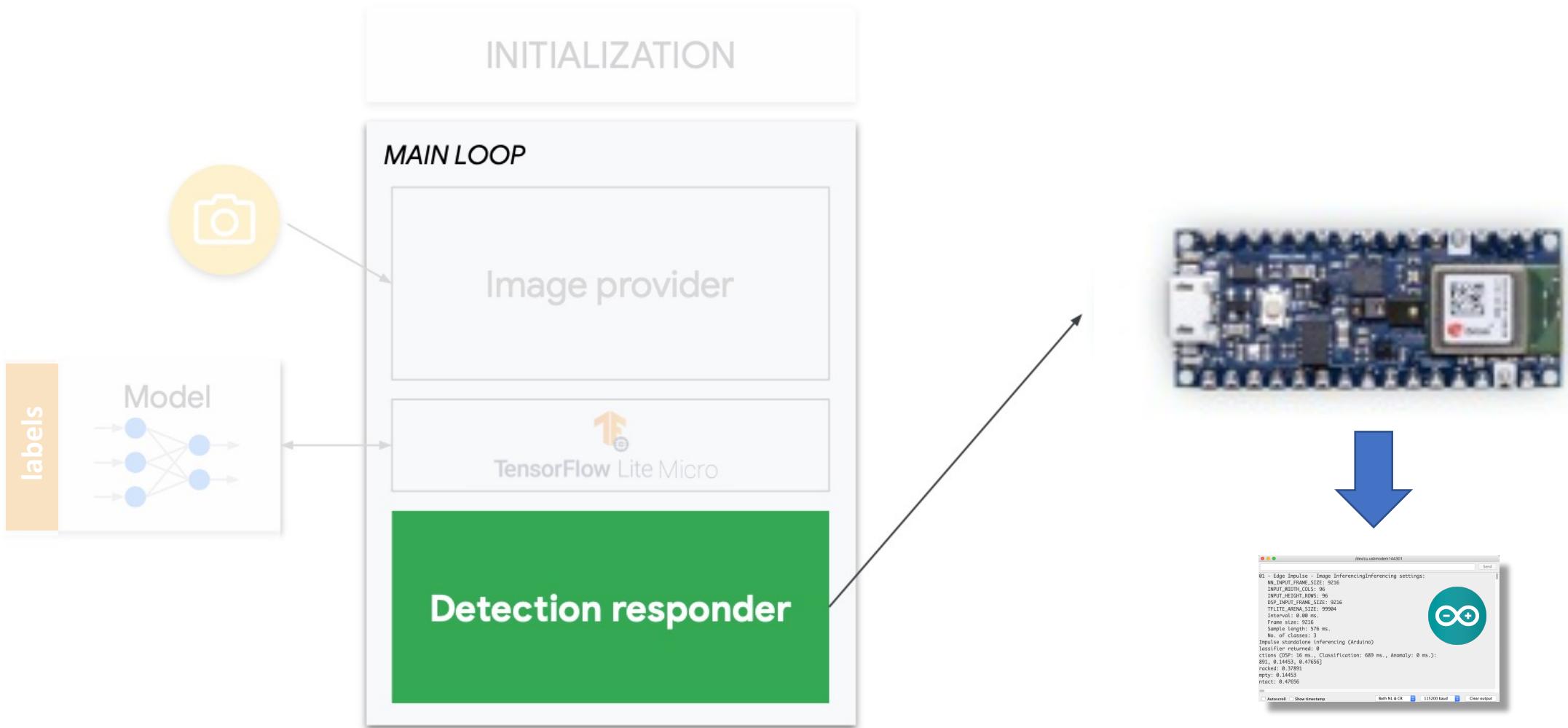
Interpreter + Model



```
// invoke the impulse
EI_IMPULSE_ERROR res = run_classifier(&signal, &result, false /* debug */);
ei_printf("run_classifier returned: %d\n", res);

if (res != 0) return;
```

Post-processing



Detection responder

```
// print the predictions
ei_printf("Predictions ");
ei_printf("(DSP: %d ms., Classification: %d ms., Anomaly: %d ms.)",
         result.timing.dsp, result.timing.classification, result.timing.anomaly);
ei_printf(": \n");
ei_printf("[");
for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
    ei_printf("%.5f", result.classification[ix].value);

/*
* The code portion below is used to show what image the camera is seem
* (for test only)
*/
    Serial.println();
    for (size_t ix = 0; ix < signal.total_length; ix++) {
        float value[1];
        signal.get_data(ix, 1, value);
        ei_printf("0x%06x", (int)value[0]);
        if (ix != signal.total_length - 1) {
            ei_printf(", ");
        }
    }
}
```

Edge Impulse standalone inferencing (Arduino)

run_classifier returned: 0

Predictions (DSP: 16 ms., Classification: 690 ms., Anomaly: 0 ms.):

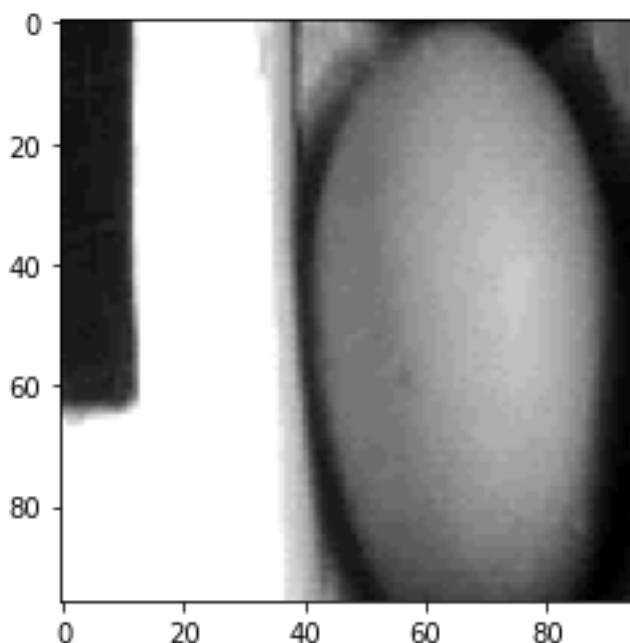
[0.42969, 0.35156, 0.21875]

cracked: 0.42969

empty: 0.35156

intact: 0.21875

0x203838, 0x203838, 0x203c38, 0x203c30, 0x203c38, 0x203c38, 0x203430, 0x203830, 0x203830,



Edge Impulse standalone inferencing (Arduino)

run_classifier returned: 0

Predictions (DSP: 16 ms., Classification: 690 ms., Anomaly: 0 ms.):

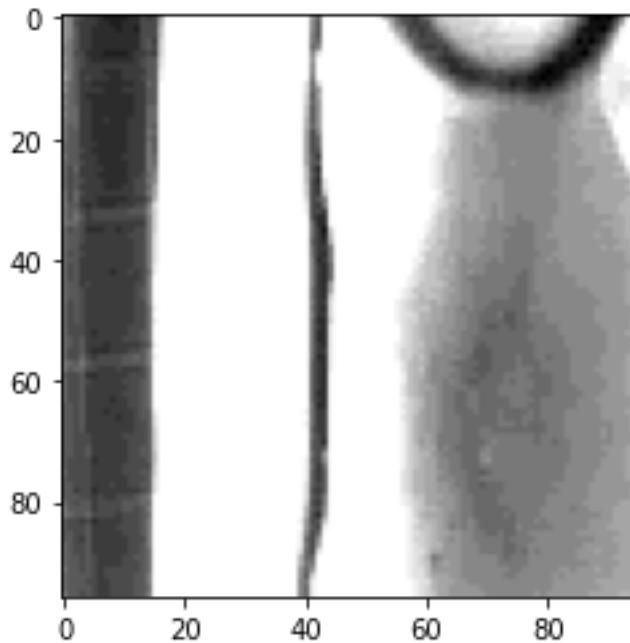
[0.22266, 0.39062, 0.38281]

cracked: 0.22266

empty: 0.39062

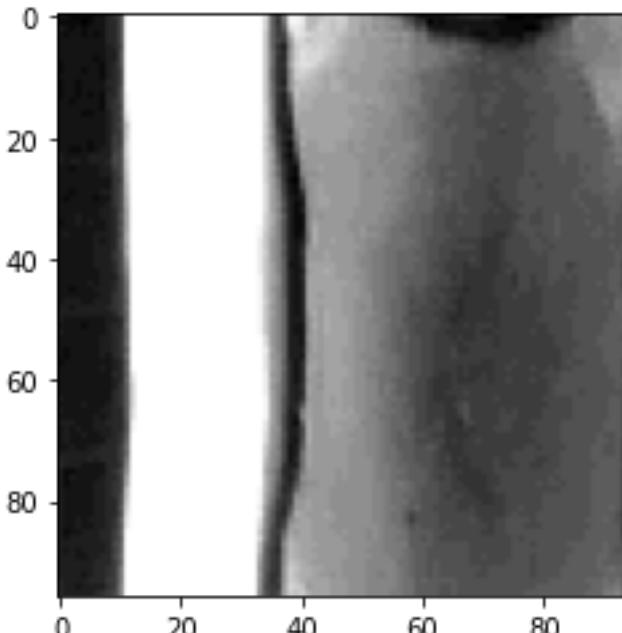
intact: 0.38281

0xc0f0f0, 0xb0f0c8, 0x98d0c0, 0x90c8a8, 0x88bca0, 0x88b898, 0x80b490, 0x78b488,



```
Edge Impulse standalone inferencing (Arduino)
run_classifier returned: 0
Predictions (DSP: 16 ms., Classification: 690 ms., Anomaly: 0 ms.):
[0.30469, 0.45703, 0.23438]
    cracked: 0.30469
    empty: 0.45703
    intact: 0.23438

0x387048, 0x387048, 0x307048, 0x307048, 0x306c40, 0x306440, 0x386840, 0x386c40, 0x386c40
```



Edge Impulse standalone inferencing (Arduino)

run_classifier returned: 0

Predictions (DSP: 16 ms., Classification: 690 ms., Anomaly: 0 ms.):

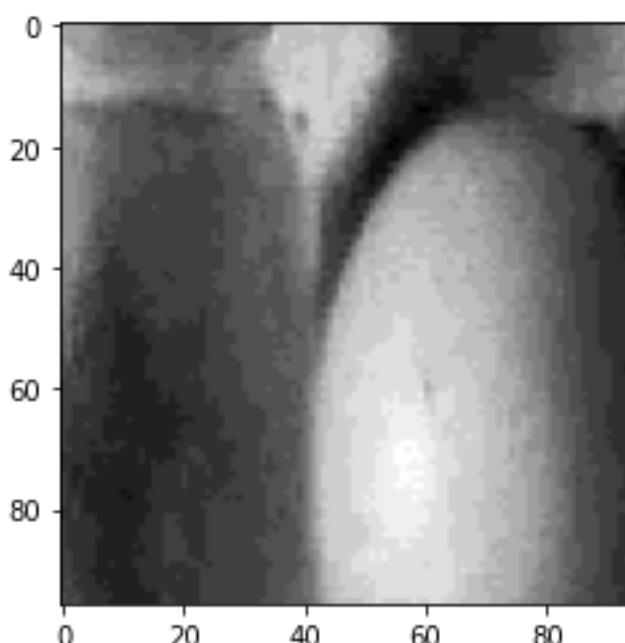
[0.41797, 0.30469, 0.27734]

cracked: 0.41797

empty: 0.30469

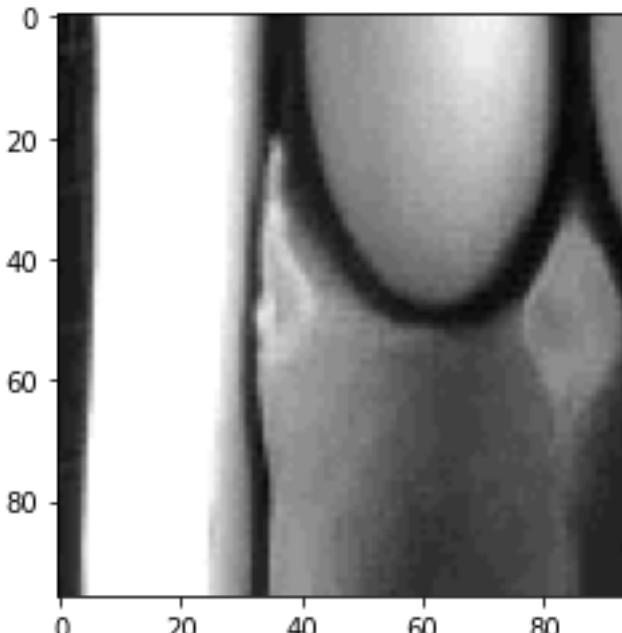
intact: 0.27734

0x709080, 0x709080, 0x709080, 0x709080, 0x709078, 0x688878, 0x688470, 0x608068,



```
Edge Impulse standalone inferencing (Arduino)
run_classifier returned: 0
Predictions (DSP: 16 ms., Classification: 690 ms., Anomaly: 0 ms.):
[0.17969, 0.60156, 0.21875]
    cracked: 0.17969
    empty: 0.60156
    intact: 0.21875

0x305858, 0x285458, 0x407060, 0x386468, 0x385858, 0x485c60, 0xb8bc78, 0xf0f0f0,
```



Eggs Classification using Transfer Learning

Coding under the hood!

ei-eggs-ai-grayscale-transfer-learning.ipynb



The screenshot shows a Google Colab notebook titled "ei-eggs-ai-grayscale-transfer.ipynb". The interface includes a top bar with tabs for Google Drive, the notebook, and several other documents. The main area has a "Files" sidebar on the left containing a tree view of files and folders, including "content" which contains "sample_data" with "x_train.npy" and "y_train.npy". The main workspace contains two code blocks.

Code Block 1:

```
[1] 1 import numpy as np
2 import requests
3
4 API_KEY = 'ei_90...1c9fb4'
5
6 def download_data(url):
7     response = requests.get(url, headers={'x-api-key': API_KEY})
8     if response.status_code == 200:
9         return response.content
10    else:
11        print(response.content)
12        raise ConnectionError('Could not download data file')
13
14 X = download_data('https://studio.edgeimpulse.com/v1/api/42476/training/9/x')
15 Y = download_data('https://studio.edgeimpulse.com/v1/api/42476/training/9/y')
16
```

Code Block 2:

```
1 with open('x_train.npy', 'wb') as file:
2     file.write(X)
3 with open('y_train.npy', 'wb') as file:
4     file.write(Y)
5 X = np.load('x_train.npy')
6 Y = np.load('y_train.npy')[::,0]
```

The status bar at the bottom indicates "completed at 3:36 PM".

Meu Drive - Google Drive ei-eggs-ai-grayscale-transfer- 2011.14858.pdf Project versioning - Eggs AI - E Transfer learning - Eggs AI Gra Image Classification with Edge

colab.research.google.com/drive/1ERVHDZAejqL7dH_apMXrydyFEOxuhjzl#scrollTo=JfFvaQGbWBsc

ei-eggs-ai-grayscale-transfer-learning.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings User profile

RAM Disk Editing

Files

- bin
- boot
- content
 - sample_data
 - x_train.npy
 - y_train.npy
- datab
- dev
- etc
- home
- lib
- lib32
- lib64
- media
- mnt
- opt
- proc
- root
- run
- sbin
- srv
- sys
- tensorflow-1.15.2
- tmp

Disk 69.08 GB available

+ Code + Text

```
[3]: X.shape, Y.shape
      ((107, 9216), (107,))

[4]: set(Y)
      {1, 2, 3}

[20]: import matplotlib.pyplot as plt
       import seaborn as sns

[21]: image = X[90]
       image = np.reshape(image, (96, 96))
       plt.figure(figsize=(6,5));
       sns.heatmap(image, cmap='gray');
```

0 10
4 8
8 12
12 16
16 20
20 24
24 28
28 32
32 36
36 40
40 44
44 48
48 52
52 56
56 60
60 64
64 68
68 72
72 76
76 80
80 84
84 88
88 92

0 10
4 8
8 12
12 16
16 20
20 24
24 28
28 32
32 36
36 40
40 44
44 48
48 52
52 56
56 60
60 64
64 68
68 72
72 76
76 80
80 84
84 88
88 92

0s completed at 3:48 PM

Meu Drive - Google Drive ei-eggs-ai-grayscale-transfer- 2011.14858.pdf Project versioning - Eggs AI - E Transfer learning - Eggs AI Gra Image Classification with Edge

colab.research.google.com/drive/1ERVHDZAejqL7dH_apMXrydyFEOxuhjzl#scrollTo=XWOJrBU4UTMt

ei-eggs-ai-grayscale-transfer-learning.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings User icon

RAM Disk Editing

Files

+ Code + Text

Define our labels and split the data up in a test and training set:

```
1 import sys, os, random
2 import tensorflow as tf
3 from sklearn.model_selection import train_test_split
4
5 # Set random seeds for repeatable results
6 RANDOM_SEED = 3
7 random.seed(RANDOM_SEED)
8 np.random.seed(RANDOM_SEED)
9 tf.random.set_seed(RANDOM_SEED)
10
11 classes_values = [ "cracked", "empty", "intact" ]
12 classes = len(classes_values)
13
14 Y = tf.keras.utils.to_categorical(Y - 1, classes)
15
16 X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=1)
17
18 input_length = X_train[0].shape[0]
19
20 train_dataset = tf.data.Dataset.from_tensor_slices((X_train, Y_train))
21 validation_dataset = tf.data.Dataset.from_tensor_slices((X_test, Y_test))
22
23 def get_reshape_function(reshape_to):
24     def reshape(image, label):
25         return tf.reshape(image, reshape_to), label
26     return reshape
27
28 callbacks = []
```

Disk 69.08 GB available

0s completed at 3:48 PM

This screenshot shows a Google Colab notebook titled "ei-eggs-ai-grayscale-transfer-learning.ipynb". The notebook interface includes a sidebar with file navigation, a main code editor, and various status indicators like RAM usage. The code in the editor is Python, utilizing TensorFlow and scikit-learn for data processing. It imports necessary libraries, sets random seeds for reproducibility, defines class labels, splits the dataset into training and testing sets, and defines a function to reshape images. The status bar at the bottom indicates the code was run in 0 seconds at 3:48 PM.

Meu Drive - Google Drive x ei-eggs-ai-grayscale-transfer- x 2011.14858.pdf x Project versioning - Eggs AI - E x Transfer learning - Eggs AI Gra x Image Classification with Edge x + colab.research.google.com/drive/1ERVHDZAejql7dH_apMXrydyFEOxuhjzl#scrollTo=woiUzzTkYI24

ei-eggs-ai-grayscale-transfer-learning.ipynb

All changes saved

Files

- sample_data
- transfer-learning-...
- saved_base_mod...
- x_train.npy
- y_train.npy
- datalab
- dev
- etc
- home
- lib
- lib32
- lib64
- media
- mnt
- opt
- proc
- root
- run
- sbin
- srv
- sys
- tensorflow-1.15.2
- tmp
- tools
- usr
- var

Disk 69.04 GB available

+ Code + Text

0s

```
1 base_model.trainable = False
2
3 model = Sequential()
4 model.add(InputLayer(input_shape=INPUT_SHAPE, name='x_input'))
5 # Don't include the base model's top layers
6 last_layer_index = -3
7 model.add(Model(inputs=base_model.inputs, outputs=base_model.layers[last_layer_index].output))
8 model.add(Dense(16, activation='relu'))
9 model.add(Dropout(0.1))
10 model.add(Flatten())
11 model.add(Dense(classes, activation='softmax'))
12
13 model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.0005),
14                 loss='categorical_crossentropy',
15                 metrics=['accuracy'])
16 model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
model (Functional)	(None, 3, 3, 1280)	409920
dense (Dense)	(None, 3, 3, 16)	20496
dropout (Dropout)	(None, 3, 3, 16)	0
flatten (Flatten)	(None, 144)	0
dense_1 (Dense)	(None, 3)	435

Total params: 430,851
Trainable params: 20,931
Non-trainable params: 409,920

1 0s completed at 3:59 PM

saved_base_model.h5 Show All

The screenshot shows a Google Colab interface with the following details:

- Tab Bar:** Meu Drive - Google Drive, ei-eggs-ai-grayscale-transfer.ipynb, 2011.14858.pdf, Project versioning - Eggs AI - E, Transfer learning - Eggs AI Gra, Image Classification with Edge.
- Title Bar:** colab.research.google.com/drive/1ERVHDZAejqL7dH_apMXrydyFEOxuhjzl#scrollTo=obSS9Sb7aNSx
- Header:** Comment, Share, Settings, User Profile.
- File Explorer (Files):** Shows a directory structure:
 - sample_data
 - transfer-learning-...
 - saved_base_mod...
 - x_train.npy
 - y_train.npy
 - datab
 - dev
 - etc
 - home
 - lib
 - lib32
 - lib64
 - media
 - mnt
 - opt
 - proc
 - root
 - run
 - sbin
 - srv
 - sys
 - tensorflow-1.15.2Disk usage: 69.04 GB available.
- Code Editor:** The notebook cell contains Python code for data augmentation and dataset preparation:

```
1 # Set the data to the expected input shape
2 train_dataset = train_dataset.map(get_reshape_function(INPUT_SHAPE), tf.data.experimental.AUTOTUNE)
3 validation_dataset = validation_dataset.map(get_reshape_function(INPUT_SHAPE), tf.data.experimental.AUTOTUNE)
4
5 # Implements the data augmentation policy
6 def augment_image(image, label):
7     # Flips the image randomly
8     image = tf.image.random_flip_left_right(image)
9
10    # Increase the image size, then randomly crop it down to
11    # the original dimensions
12    resize_factor = random.uniform(1, 1.2)
13    new_height = math.floor(resize_factor * INPUT_SHAPE[0])
14    new_width = math.floor(resize_factor * INPUT_SHAPE[1])
15    image = tf.image.resize_with_crop_or_pad(image, new_height, new_width)
16    image = tf.image.random_crop(image, size=INPUT_SHAPE)
17
18    # Vary the brightness of the image
19    image = tf.image.random_brightness(image, max_delta=0.2)
20
21    return image, label
22
23 train_dataset = train_dataset.map(augment_image, tf.data.experimental.AUTOTUNE)
24
25 BATCH_SIZE = 32
26 train_dataset = train_dataset.batch(BATCH_SIZE, drop_remainder=False)
27 validation_dataset = validation_dataset.batch(BATCH_SIZE, drop_remainder=False)
```
- Output:** A single cell output showing the status: "0s completed at 4:01 PM".
- Bottom Navigation:** saved_base_model.h5, Show All.

Meu Drive - Google Drive x ei-eggs-ai-grayscale-transfer.ipynb x 2011.14858.pdf x Project versioning - Eggs AI x Transfer learning - Eggs AI x Image Classification with Edge x +

colab.research.google.com/drive/1ERVHDZAejqL7dH_apMXrydyFEOxuhjzl#scrollTo=AUAv9rE3aGNd&uniqifier=2

File Edit View Insert Runtime Tools Help All changes saved

Files

+ Code + Text

```
1 history = model.fit(train_dataset,
2                         validation_data=validation_dataset,
3                         epochs=20,
4                         verbose=2,
5                         callbacks=callbacks)
6
7 print('')
8 print('Initial training done.', flush=True)
```

Epoch 1/20
3/3 - 4s - loss: 1.2242 - accuracy: 0.4118 - val_loss: 0.6396 - val_accuracy: 0.7727
Epoch 2/20
3/3 - 0s - loss: 0.7888 - accuracy: 0.6000 - val_loss: 0.4710 - val_accuracy: 0.9545
Epoch 3/20
3/3 - 0s - loss: 0.6237 - accuracy: 0.8471 - val_loss: 0.3624 - val_accuracy: 0.9545
Epoch 4/20
3/3 - 0s - loss: 0.4586 - accuracy: 0.8824 - val_loss: 0.2761 - val_accuracy: 0.9545
Epoch 5/20
3/3 - 0s - loss: 0.3083 - accuracy: 0.9412 - val_loss: 0.2109 - val_accuracy: 0.9545
Epoch 6/20
3/3 - 0s - loss: 0.2795 - accuracy: 0.9176 - val_loss: 0.1567 - val_accuracy: 0.9545
Epoch 7/20
3/3 - 0s - loss: 0.2026 - accuracy: 0.9529 - val_loss: 0.1003 - val_accuracy: 1.0000
Epoch 8/20
3/3 - 0s - loss: 0.1696 - accuracy: 0.9529 - val_loss: 0.0662 - val_accuracy: 1.0000
Epoch 9/20
3/3 - 0s - loss: 0.0956 - accuracy: 0.9765 - val_loss: 0.0570 - val_accuracy: 1.0000
Epoch 10/20
3/3 - 0s - loss: 0.0609 - accuracy: 0.9882 - val_loss: 0.0501 - val_accuracy: 1.0000
Epoch 11/20
3/3 - 0s - loss: 0.0669 - accuracy: 0.9765 - val_loss: 0.0337 - val_accuracy: 1.0000
Epoch 12/20
3/3 - 0s - loss: 0.0374 - accuracy: 1.0000 - val_loss: 0.0253 - val_accuracy: 1.0000
Epoch 13/20
3/3 - 0s - loss: 0.0363 - accuracy: 1.0000 - val_loss: 0.0243 - val_accuracy: 1.0000
Epoch 14/20
3/3 - 0s - loss: 0.0712 - accuracy: 0.9765 - val_loss: 0.0340 - val_accuracy: 1.0000
Epoch 15/20
3/3 - 0s - loss: 0.0508 - accuracy: 0.9882 - val_loss: 0.0533 - val_accuracy: 1.0000
Epoch 16/20
3/3 - 0s - loss: 0.0186 - accuracy: 1.0000 - val_loss: 0.0630 - val_accuracy: 1.0000
Epoch 17/20
3/3 - 0s - loss: 0.0403 - accuracy: 0.9882 - val_loss: 0.0454 - val_accuracy: 1.0000
Epoch 18/20

Disk 69.04 GB available

13s completed at 4:24 PM

saved_base_model.h5

Show All

Meu Drive - Google Drive ei-eggs-ai-grayscale-transfer- 2011.14858.pdf Project versioning - Eggs AI - E Transfer learning - Eggs AI Gra Image Classification with Edge

colab.research.google.com/drive/1ERVHDZAejqL7dH_apMXrydyFEOxuhjzl#scrollTo=AUAv9rE3aGNd&uniqifier=2

ei-eggs-ai-grayscale-transfer-learning.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Files

+ Code + Text

RAM Disk Editing

File Edit View Insert Runtime Tools Help All changes saved

Files

.. sample_data transfer-learning-weights saved_base_model.h5 x_train.npy y_train.npy

```
[60] 1 def plot_result(history, res= 'loss'):
2     plt.plot(history.history[res])
3     plt.plot(history.history['val_'+res])
4     plt.title(res+ ' vs. epochs')
5     plt.ylabel(res)
6     plt.xlabel('Epoch')
7     if res == 'loss':
8         plt.legend(['Training', res], loc='upper right')
9     else:
10        plt.legend(['Training', res], loc='lower right')
11    plt.show()
```

0s 1 plot_result(history, res= 'loss')
2 plot_result(history, res= 'accuracy')

loss vs. epochs

accuracy vs. epochs

Disk 69.04 GB available

13s completed at 4:24 PM

saved_base_model.h5 Show All

Meu Drive - Google Drive x ei-eggs-ai-grayscale-transfer- x 2011.14858.pdf x Project versioning - Eggs AI - E x Transfer learning - Eggs AI Gra x Image Classification with Edge x + colab.research.google.com/drive/1ERVHDZAejqL7dH_apMXrydyFEOxuhjzl#scrollTo=96cvmqcymLpZ&uniquifier=2

Comment Share Settings User icon

ei-eggs-ai-grayscale-transfer-learning.ipynb ★

File Edit View Insert Runtime Tools Help All changes saved

Files

+ Code + Text

RAM Disk Editing

TensorFlow Lite - Micro

```
[84] 1 TF_MODEL = 'saved_model'
2 TFLITE_MODEL = 'saved_model.tflite'
3 TFLITE_MICRO_MODEL = 'saved_model.cc'

[40s] 1 # Convert TF model to a tflite model
2 from tensorflow.keras.models import load_model
3
4 model_egg_detection = load_model(TF_MODEL)
5 converter = tf.lite.TFLiteConverter.from_keras_model(model_egg_detection)
6 converter.optimizations = [tf.lite.Optimize.DEFAULT]
7 tflite_model = converter.convert()
8
9 tflite_model_size = open(TFLITE_MODEL, "wb").write(tflite_model)
10 print("Quantized model (DEFAULT) is {:.0f} bytes".format(tflite_model_size))
```

Generate a TensorFlow Lite for Microcontrollers Model

To convert the TensorFlow Lite quantized model into a C source file that can be loaded by TensorFlow Lite for Microcontrollers on Arduino we simply need to use the `xxd` tool to convert the `.tflite` file into a `.cc` file.

```
[86] 1 !apt-get update && apt-get -qq install xxd

[87] 1 !xxd -i {TFLITE_MODEL} > {TFLITE_MICRO_MODEL}
2 REPLACE_TEXT = TFLITE_MODEL.replace('/', '_').replace('.', '_')
3 !sed -i 's/'{REPLACE_TEXT}'/_g_model/g' {TFLITE_MICRO_MODEL}
```

If you'd like to download your model for safekeeping:

1. On the left of the UI click on the folder icon
2. Click on the three dots to the right of the `model_mask_detection.cc` file and select download

Note that the model is too big to be used on an Arduino NANO, as we expected.

```
[88] 1 !cat {TFLITE_MICRO_MODEL}
0x24, 0x15, 0x00, 0x00, 0xdc, 0xf, 0x00, 0x00, 0x94, 0xa, 0x00, 0x00,
2s completed at 5:01 PM
```

Disk 69.01 GB available

saved_base_model.h5 Show All

The screenshot shows a Google Colab interface with the following details:

- Tab Bar:** Meu Drive - Google Drive, ei-eggs-ai-grayscale-transfer.ipynb, 2011.14858.pdf, Project versioning - Eggs AI - E, Transfer learning - Eggs AI Gra, Image Classification with Edge.
- Title Bar:** colab.research.google.com/drive/1ERVHDZAejql7dH_apMXrydyFEOxuhjzl#scrollTo=AUAv9rE3aGND
- Header:** Comment, Share, Settings, User Profile.
- File Explorer (Files):** Shows a directory structure with files like sample_data, transfer-learning..., saved_base_mod..., x_train.npy, y_train.npy, and various system folders (datalab, dev, etc). A progress bar indicates 69.04 GB available.
- Code Editor:** The main area displays Python code for training a machine learning model. The code includes:

```
1 model.fit(train_dataset,
2             validation_data=validation_dataset,
3             epochs=20, verbose=2,
4             callbacks=callbacks)
5
6 print('')
7 print('Initial training done.', flush=True)
8
9 # How many epochs we will fine tune the model
10 FINE_TUNE_EPOCHS = 10
11 # What percentage of the base model's layers we will fine tune
12 FINE_TUNE_PERCENTAGE = 65
13
14 print('Fine-tuning model for {} epochs...'.format(FINE_TUNE_EPOCHS), flush=True)
15
16 # Determine which layer to begin fine tuning at
17 model_layer_count = len(model.layers)
18 fine_tune_from = math.ceil(model_layer_count * ((100 - FINE_TUNE_PERCENTAGE) / 100))
19
20 # Allow the entire base model to be trained
21 model.trainable = True
22 # Freeze all the layers before the 'fine_tune_from' layer
23 for layer in model.layers[:fine_tune_from]:
24     layer.trainable = False
25
26 model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.000045),
27                 loss='categorical_crossentropy',
28                 metrics=['accuracy'])
29
30 history = model.fit(train_dataset,
31                      epochs=FINE_TUNE_EPOCHS,
32                      verbose=2,
33                      validation_data=validation_dataset,
34                      callbacks=callbacks)
35
```
- Runtime Status:** RAM 100% (green checkmark), Disk 100% (grey bar), Editing mode.
- Bottom Status Bar:** ✓ 0s completed at 4:01 PM.
- Bottom Navigation:** saved_base_model.h5, Show All.

Reading Material

Main references

- [Harvard School of Engineering and Applied Sciences - CS249r: Tiny Machine Learning](#)
- [Professional Certificate in Tiny Machine Learning \(TinyML\) – edX/Harvard](#)
- [Introduction to Embedded Machine Learning \(Coursera\)](#)
- [Text Book: "TinyML" by Pete Warden, Daniel Situnayake](#)

I want to thank [Shawn Hymel](#) and [Edge Impulse](#), [Pete Warden](#) and [Laurence Moroney](#) from Google and specially Harvard professor [Vijay Janapa Reddi, Ph.D.](#) student [Brian Plancher](#) and their staff for preparing the excellent material on TinyML that is the basis of this course at UNIFEI.

The IESTI01 course is part of the [TinyML4D](#), an initiative to make TinyML education available to everyone globally.

Thanks
And stay safe!

