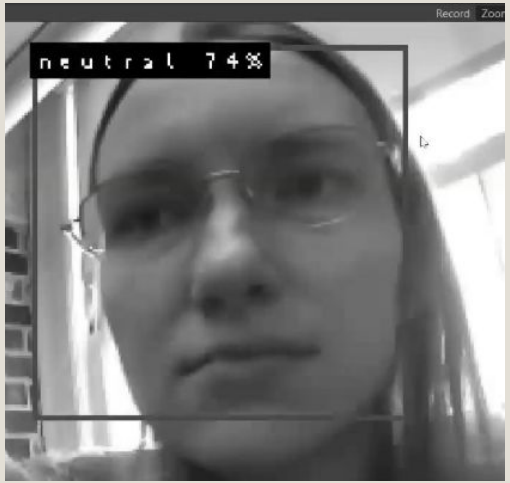
TinyVision

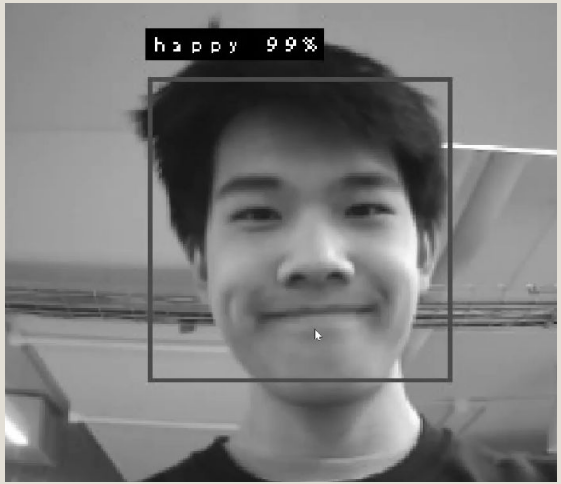
Masha, Rohina, Jordan

Team & Roles



Maria

- Hardware setup and integration
- Model training



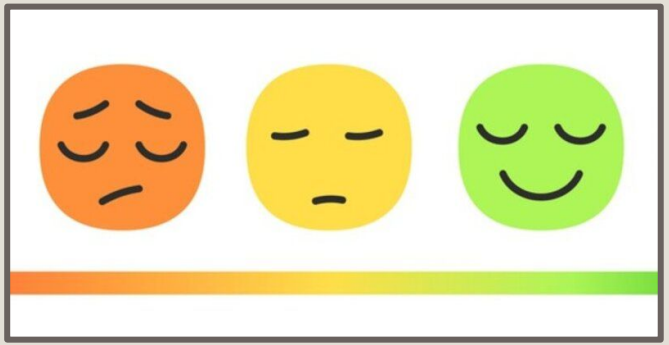
Jordan

- Model training
- Hardware & demo testing
- Model architecture research

Rohina

- Model training
- Dataset research and prep

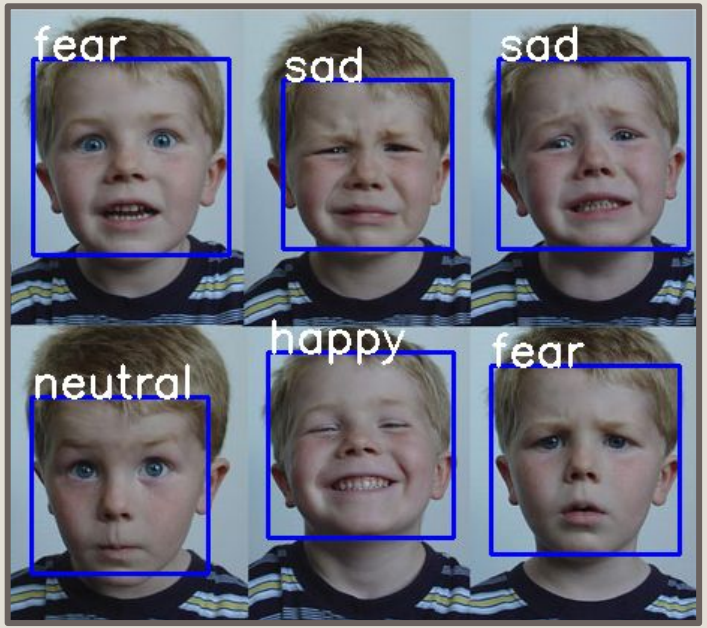
Agenda



1. Introduction to FER
2. FER13 & CNN Discussion
3. RAF-DB & Tradeoffs
4. FER-Plus & Optimizations
5. Final Results
6. Conclusions

Facial Expression Recognition (FER)

- Develop **real-time** system to detect **facial emotions**
- Positive vs. negative or emotion classes
  - Neutral
  - Happy
  - Angry
  - Sad
  - Surprise
  - ...
- Applications in **healthcare** to improve patient care
  - Therapy
  - Physical examinations
  - Research studies
- Need performance and energy **efficiency** + low **latency**



https://sefiks.com/2018/01/10/real-time-facial-expression-recognition-on-streaming-data/

Goals

- Original goal: **> 80% accuracy** in real-time detection of **3+ emotion classes** at **>20 FPS**
- Made FPS goal more realistic after research:
  - OpenMV Cam H7 Plus “running time ... using [MobileNetV2] is around 0.3 seconds on average” with custom database (Asmara et al., 2024)
  - New latency goal of **<300 ms** or **>3 FPS**

Research: datasets, existing models, & achievability

- Reviewed FER-2013 and CK Extended datasets
- Studied CNN & MobileNet-based FER models
- Prior work shows FER runs on OpenMV H7
- Confirms real-time edge deployment is feasible

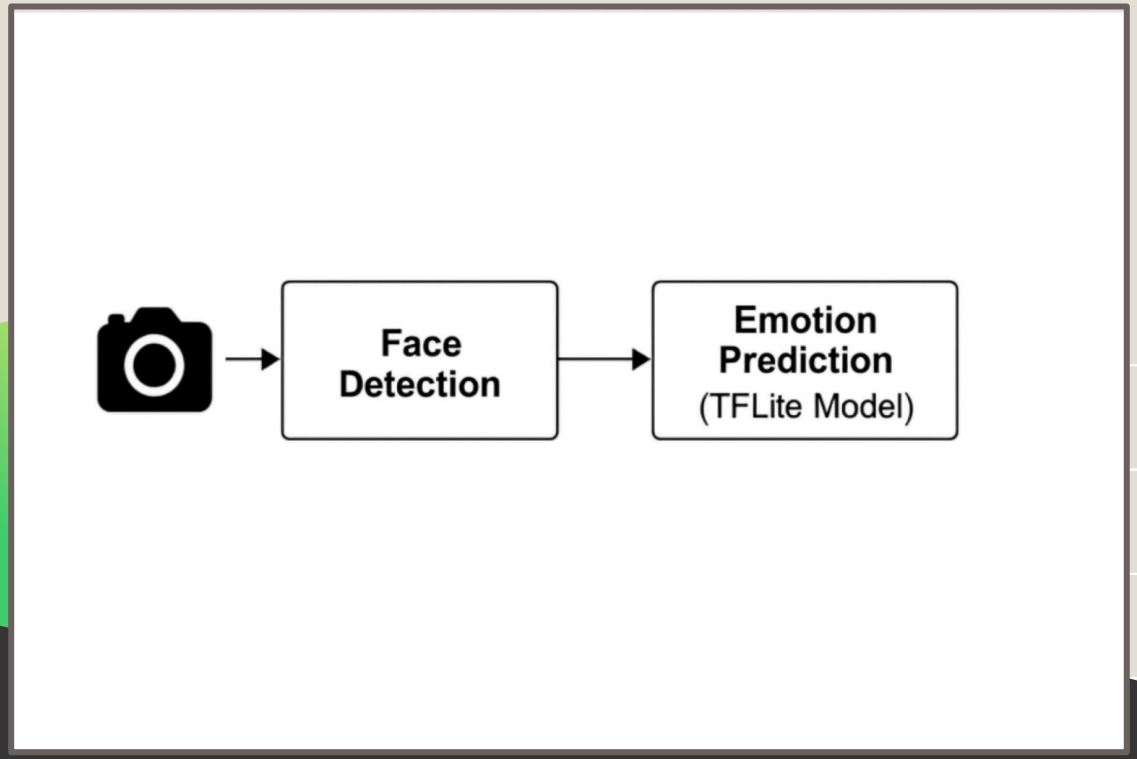
Hardware: OpenMV Cam H7 Plus

- ARM Cortex-M7 microcontroller
- On-board camera for real-time vision
- Supports TensorFlow Lite models
- Designed for low-power edge AI

Training Considerations

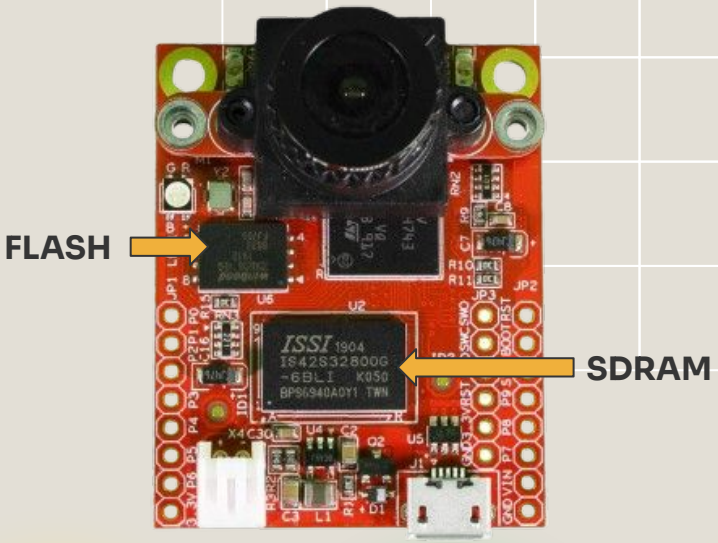
- Initial models achieved moderate accuracy
- Accuracy improved with fewer emotion classes
- Public datasets limit real-world generalization
- Lighting and image quality strongly affect accuracy

FER Model Pipeline

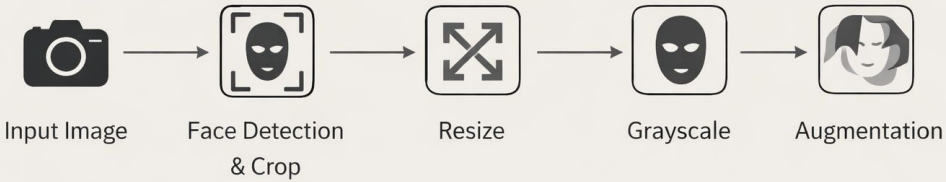


Constraints

- Image size
  - 48 x 48 or 96 x 96
  - Tradeoff acceptable for emotion detection task
- Grayscale
- Minimizing TFLite arena w/ shallower network layers
- 32 MB extern. Flash
- <16 MB from 32 MB peak RAM available
  - Frame buffers
  - TensorArena → params, ops
  - Overhead



Preprocessing Steps



MobileNetV2

- Transfer learning with the lowest possible alpha value of 0.35
- Pretrained on ImageNet
- First trained with Edge Impulse, which limits us to 1 hour of compute, or around 25-30 epochs...

Last training performance (validation set)

ACCURACY 70.8%

LOSS 0.72

Confusion matrix (validation set)

	HAPPINESS	NEUTRAL	SADNESS
HAPPINESS	83.6%	11.0%	5.3%
NEUTRAL	23.0%	65.9%	11.2%
SADNESS	27.2%	26.5%	46.3%
F1 SCORE	0.81	0.63	0.54

ACCURACY 56.21%

Metrics for Transfer learning

METRIC	VALUE
Area under ROC Curve	0.87
Weighted average Precision	0.71
Weighted average Recall	0.72
Weighted average F1 score	0.71

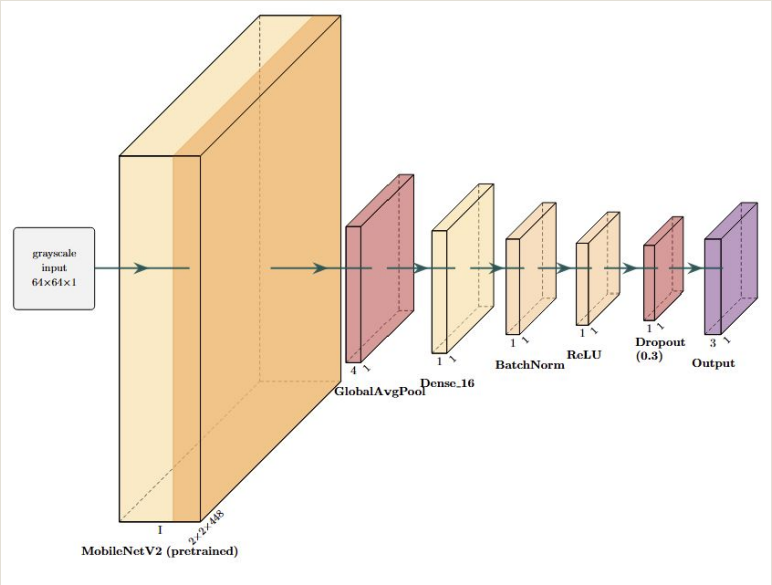
Confusion matrix

	HAPPINESS	NEUTRAL	SADNESS	UNCERTAIN
HAPPINESS	74.4%	5.8%	1.4%	18.4%
NEUTRAL	14.4%	43.4%	3.4%	38.8%
SADNESS	14.4%	10.5%	29.3%	45.8%
F1 SCORE	0.79	0.54	0.43	

Can We Do... Better?

- Only around **4 MB of 32 MB** available to load in the model
- Attempting to improve model via more epochs, additional layers, QAT, or larger images/enhancement
- Ran many trials, many did not fit....
- Trained model locally, then used Edge Impulse EON compiler to quantize

```
Memory before loading model:
  Free memory: 4330176 bytes (4228.69 KB)
  Allocated memory: 7104 bytes (6.94 KB)
```



Results

Model version: Quantized (int8)

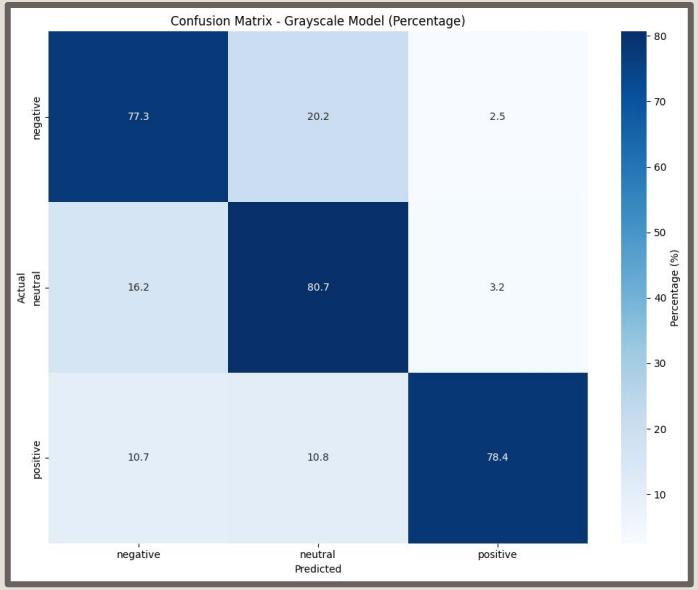
ACCURACY
67.95%

Metrics for Pretrained learn

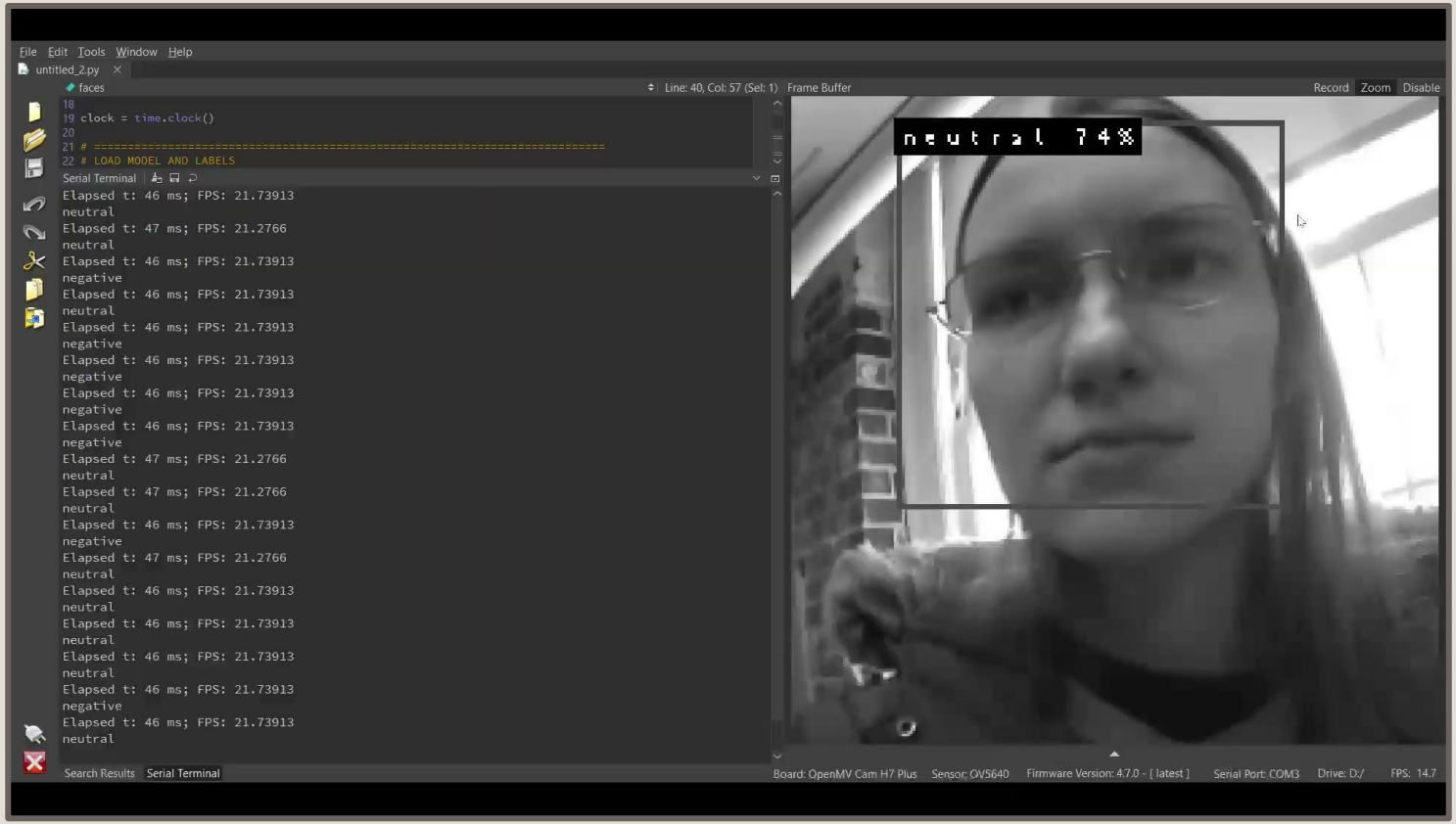
METRIC	VALUE
Area under ROC Curve	0.93
Weighted average Precision	0.80
Weighted average Recall	0.77
Weighted average F1 score	0.77

Confusion matrix

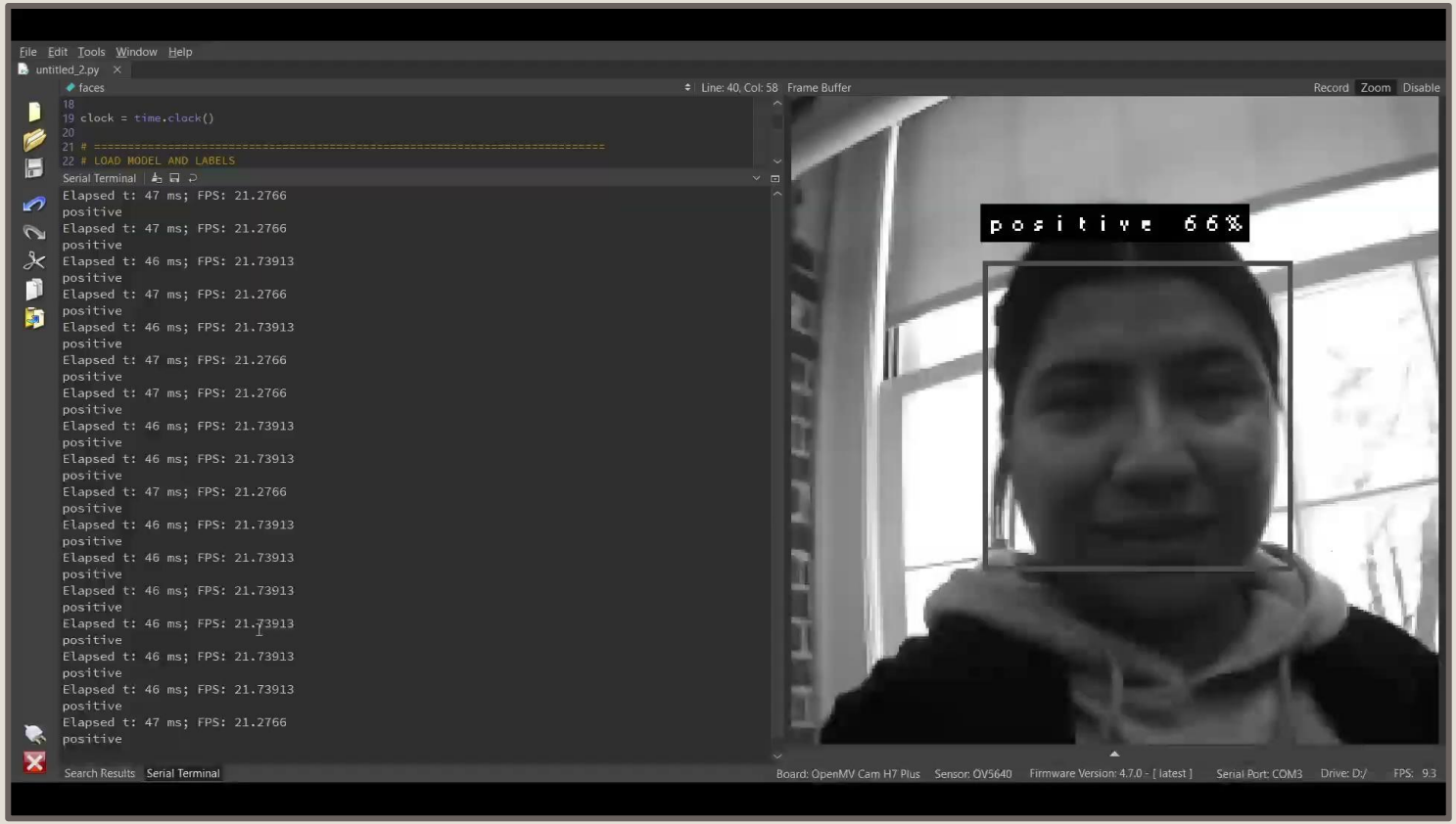
	POSITIVE	NEGATIVE	NEUTRAL	UNCERTAIN
POSITIVE	47.6%	23.4%	4.5%	24.5%
NEGATIVE	4.1%	74.1%	4.7%	17.1%
NEUTRAL	2.7%	3.1%	84.6%	9.6%
F1 SCORE	0.62	0.69	0.88	




```
Validation Accuracy: 0.7869
F1 Score: 0.7903
Precision: 0.8015
Recall: 0.7870
```



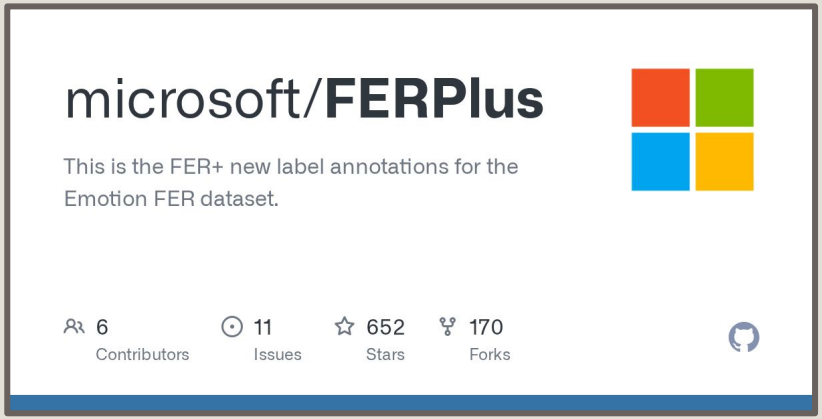
MobileNetV2 transfer learning demo



MobileNetV2 transfer learning demo

MobileNetV2 transfer learning demo

FERPlus & Optimizations



Higher accuracy achievable using new label annotations.
https://github.com/microsoft/FERPlus

FERPlus: Better Labeling

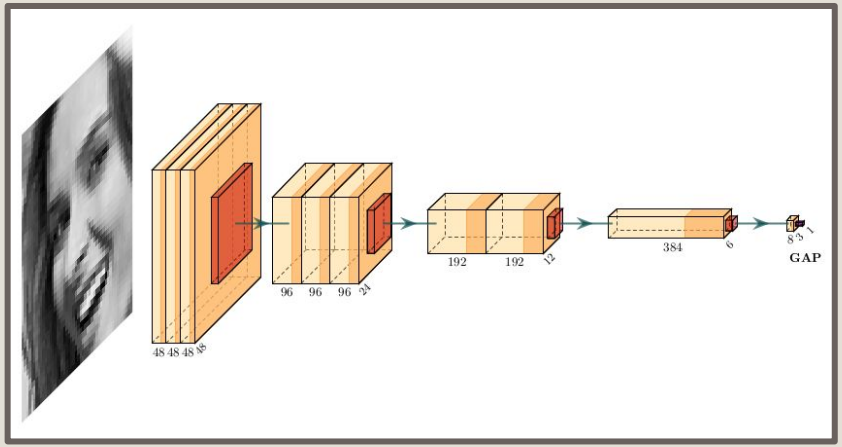
- New, less noisy, crowd-sourced labels (2016)
- Probability distribution among 8 classes (new contempt class)
- Removed bad samples

	Usage	Image name	neutral	happiness	surprise	sadness	anger	disgust	fear	contempt	unknown	NF
2	Training	fer0000000.png	4	0	0	1	3	2	0	0	0	0
3	Training	fer0000001.png	6	0	1	1	0	0	0	0	2	0
4	Training	fer0000002.png	5	0	0	3	1	0	0	0	1	0
5	Training	fer0000003.png	4	0	0	4	1	0	0	0	1	0
6	Training	fer0000004.png	9	0	0	1	0	0	0	0	0	0
7	Training	fer0000005.png	6	0	0	1	0	0	1	1	1	0
8	Training	fer0000006.png	2	0	0	8	0	0	0	0	0	0
9	Training	fer0000007.png	0	10	0	0	0	0	0	0	0	0
10	Training	fer0000008.png	0	10	0	0	0	0	0	0	0	0
11	Training	fer0000009.png	0	0	6	0	0	0	4	0	0	0
12	Training	fer0000010.png	2	0	0	0	8	0	0	0	0	0
13	Training	fer0000011.png	10	0	0	0	0	0	0	0	0	0
14	Training	fer0000012.png	5	0	0	3	0	0	0	0	2	0
15	Training	fer0000013.png	9	0	0	1	0	0	0	0	0	0
16	Training	fer0000014.png	0	10	0	0	0	0	0	0	0	0
17	Training	fer0000015.png	0	0	6	0	1	0	3	0	0	0
18	Training	fer0000016.png	4	6	0	0	0	0	0	0	0	0
19	Training		0	0	0	0	0	0	0	0	0	10
20	Training	fer0000018.png	1	0	2	4	2	0	0	0	1	0
21	Training	fer0000019.png	6	1	0	0	3	0	0	0	0	0

Higher accuracy achievable using new label annotations. https://arxiv.org/abs/1608.01041

Testing a Promising FER Model

- Adam Wiącek's trained **87%** model is **~174k params**, **244 KB size**
  - **334 ms** inference on Cam H7 Plus
- SeparableConv2D → BatchNorm and/or SpatialDropout→ LeakyReLU → MaxPooling2D



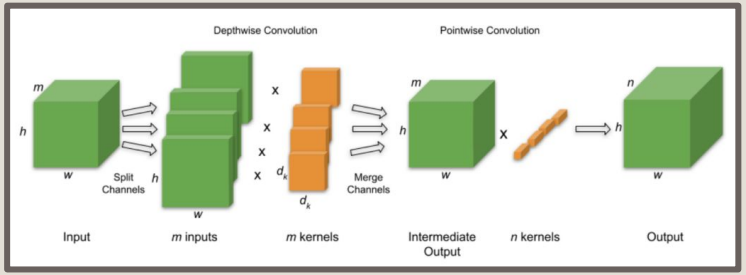
Continuation of Adam Wiącek's master's thesis on "Deep learning in facial emotion recognition".
https://github.com/vicksam/fer-model

Model: "sequential"

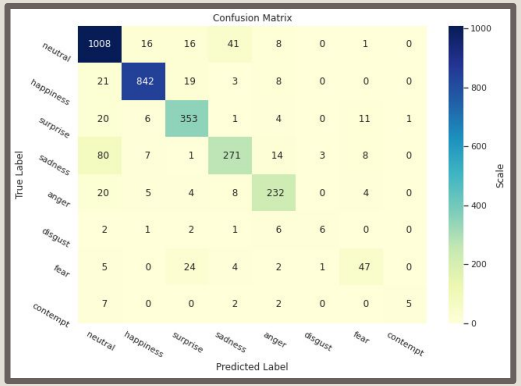
Layer (type)	Output Shape	Param #
separable_conv2d_20 (Separab	(None, 48, 48, 48)	105
batch_normalization_18 (Batc	(None, 48, 48, 48)	192
leaky_re_lu_18 (LeakyReLU)	(None, 48, 48, 48)	0
separable_conv2d_21 (Separab	(None, 48, 48, 48)	2784
batch_normalization_19 (Batc	(None, 48, 48, 48)	192
leaky_re_lu_19 (LeakyReLU)	(None, 48, 48, 48)	0
separable_conv2d_22 (Separab	(None, 48, 48, 48)	2784
batch_normalization_20 (Batc	(None, 48, 48, 48)	192
max_pooling2d_8 (MaxPooling2	(None, 24, 24, 48)	0
spatial_dropout2d_8 (Spatial	(None, 24, 24, 48)	0
leaky_re_lu_20 (LeakyReLU)	(None, 24, 24, 48)	0
separable_conv2d_23 (Separab	(None, 24, 24, 96)	5136
batch_normalization_21 (Batc	(None, 24, 24, 96)	384
leaky_re_lu_21 (LeakyReLU)	(None, 24, 24, 96)	0
separable_conv2d_24 (Separab	(None, 24, 24, 96)	10176
batch_normalization_22 (Batc	(None, 24, 24, 96)	384
leaky_re_lu_22 (LeakyReLU)	(None, 24, 24, 96)	0
separable_conv2d_25 (Separab	(None, 24, 24, 96)	10176
batch_normalization_23 (Batc	(None, 24, 24, 96)	384
max_pooling2d_9 (MaxPooling2	(None, 12, 12, 96)	0
spatial_dropout2d_9 (Spatial	(None, 12, 12, 96)	0
leaky_re_lu_23 (LeakyReLU)	(None, 12, 12, 96)	0
separable_conv2d_26 (Separab	(None, 12, 12, 192)	19488
batch_normalization_24 (Batc	(None, 12, 12, 192)	768
leaky_re_lu_24 (LeakyReLU)	(None, 12, 12, 192)	0
separable_conv2d_27 (Separab	(None, 12, 12, 192)	38784
batch_normalization_25 (Batc	(None, 12, 12, 192)	768
max_pooling2d_10 (MaxPooling	(None, 6, 6, 192)	0
spatial_dropout2d_10 (Spatia	(None, 6, 6, 192)	0
leaky_re_lu_25 (LeakyReLU)	(None, 6, 6, 192)	0
separable_conv2d_28 (Separab	(None, 6, 6, 384)	75840
batch_normalization_26 (Batc	(None, 6, 6, 384)	1536
max_pooling2d_11 (MaxPooling	(None, 3, 3, 384)	0
spatial_dropout2d_11 (Spatia	(None, 3, 3, 384)	0
leaky_re_lu_26 (LeakyReLU)	(None, 3, 3, 384)	0
separable_conv2d_29 (Separab	(None, 3, 3, 8)	3464
global_average_pooling2d_2 (	(None, 8)	0

Total params: 173,537
Trainable params: 171,137
Non-trainable params: 2,400

Optimizing for Cam H7 Plus



Depthwise Separable Convolution filters applied to input to produce output. Assume stride=1 and padding=d□-2. Source: Efficient Deep Learning Book



Heavy bias towards the first 5 emotions in Adam Wiącek's FER model. https://github.com/vicksam/fer-model

Reduce params & compute time:

SeparableConv2D replaces Conv2D

- Depthwise spatial conv on each channel + pointwise mixing of channels

SpatialDropout2D replaces regular Dropout

- Drop entire 2D feature maps (not point-elements) → promote fmap independence

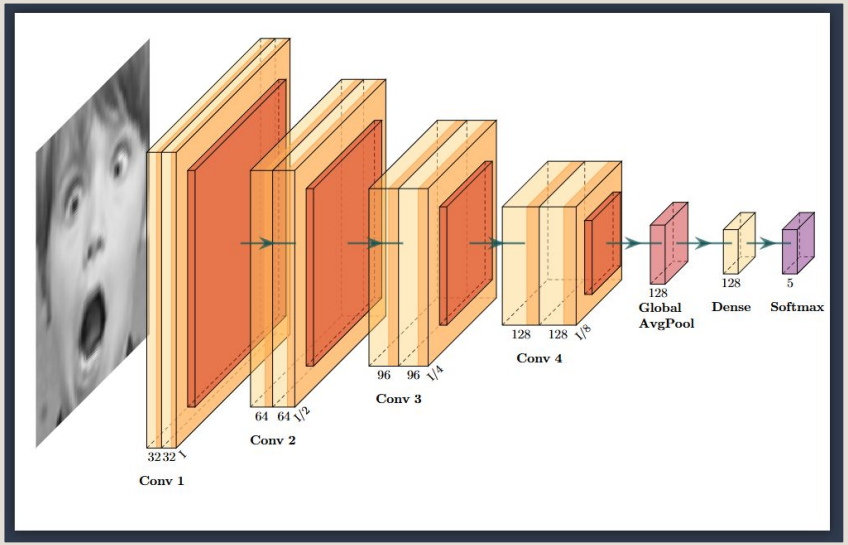
GlobalAveragePooling2D replaces Flatten

- Averages + pools + reshapes

- **Reduced to 5 classes**: neutral, happiness, surprise, sadness, angriness

Retraining Using Custom CNNs

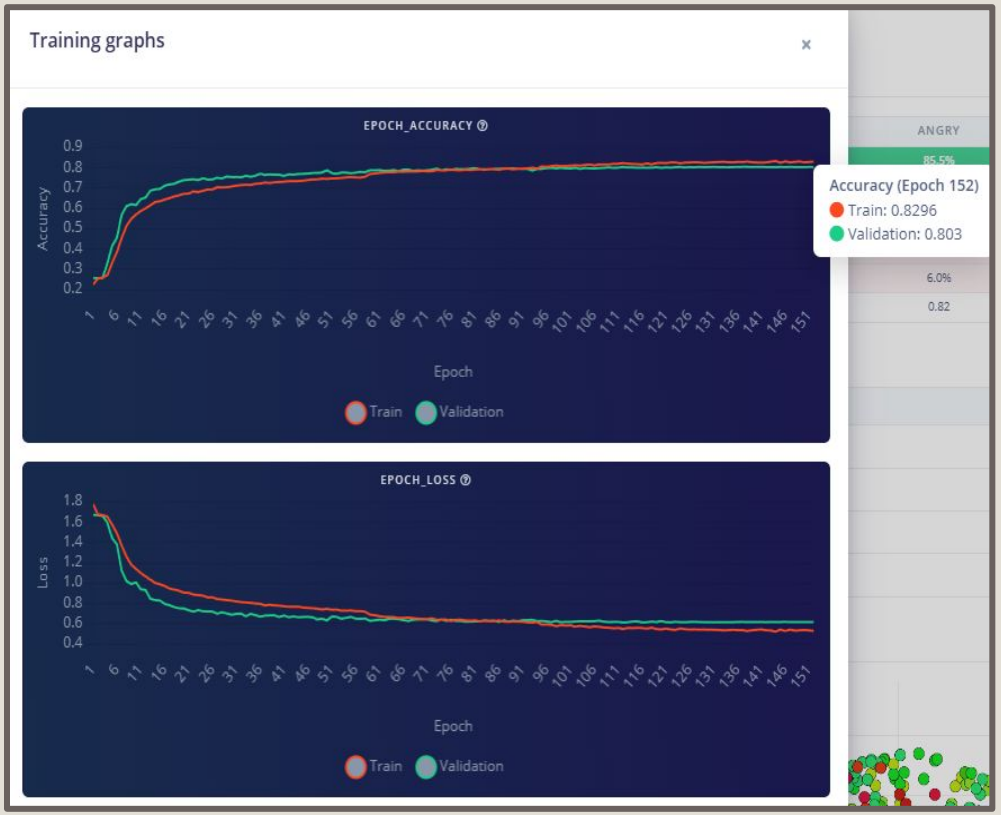
- 1st layer = regular Conv2D, all others **SeparableConv2D**
- **Regular ReLU** replaces LeakyReLU → faster convergence
- **Added Dense** layer after GAP → increased learning performance

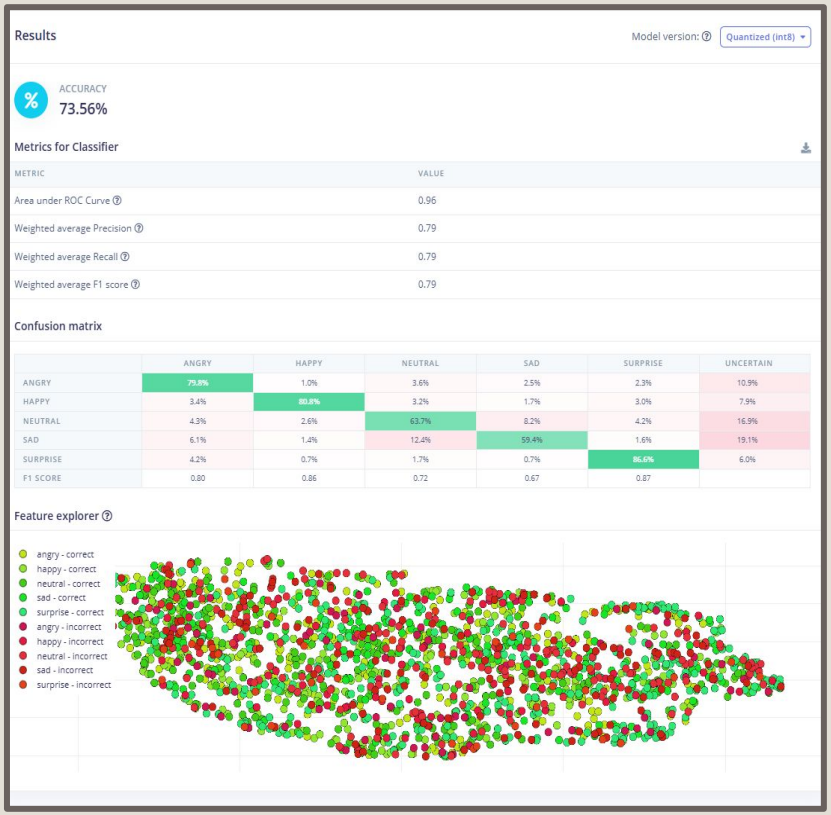


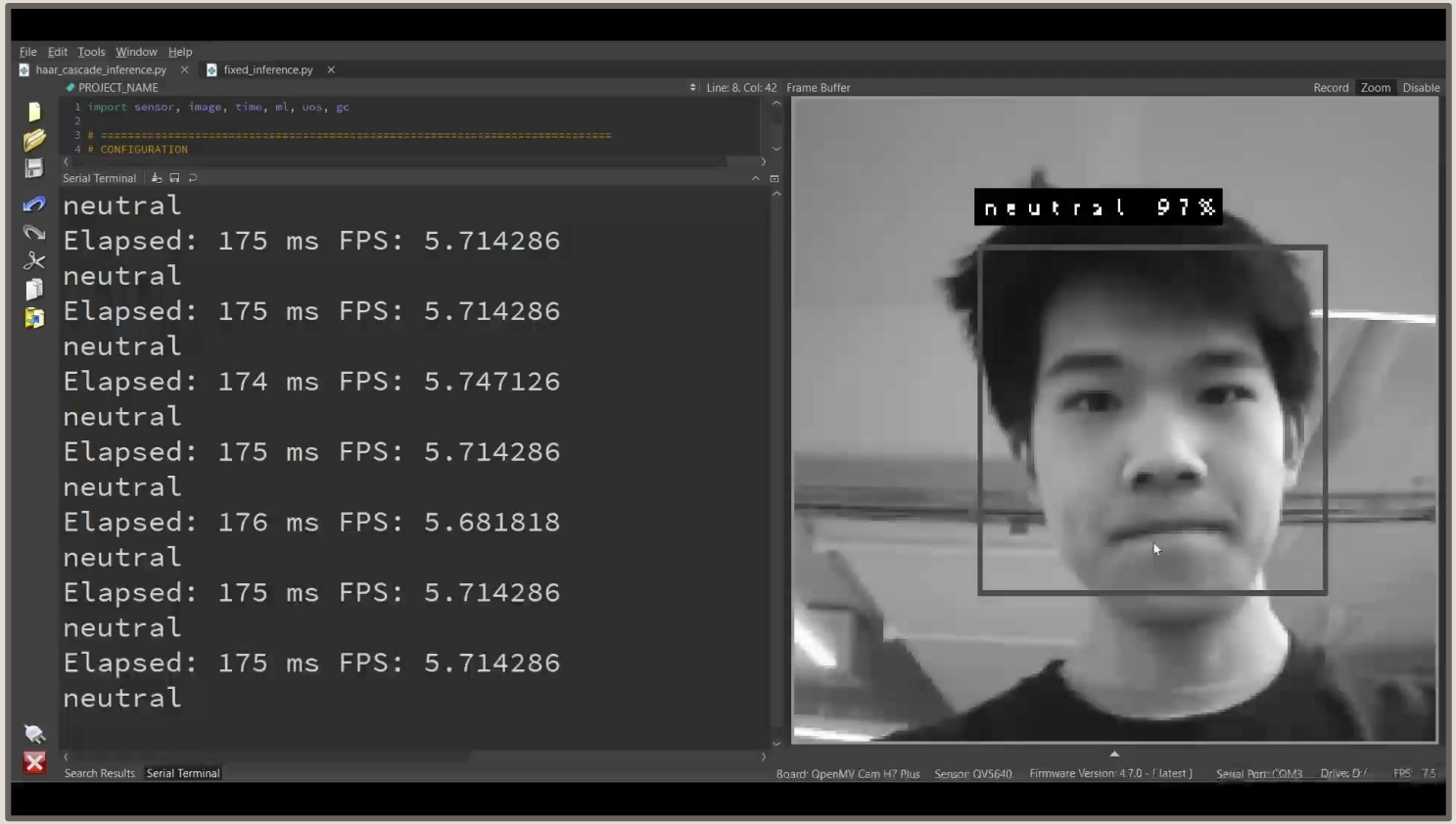
Our optimized CNN model based on Adam Wiącek's architecture
https://github.com/vicksam/fer-model

Layer (type)	Output Shape	Param #
conv1_1 (Conv2D)	(None, 48, 48, 32)	320
bn1_1 (BatchNormalization)	(None, 48, 48, 32)	128
sepconv1_2 (SeparableConv2D)	(None, 48, 48, 32)	1,344
bn1_2 (BatchNormalization)	(None, 48, 48, 32)	128
pool1 (MaxPooling2D)	(None, 24, 24, 32)	0
spatial_dropout1 (SpatialDropout2D)	(None, 24, 24, 32)	0
sepconv2_1 (SeparableConv2D)	(None, 24, 24, 64)	2,400
bn2_1 (BatchNormalization)	(None, 24, 24, 64)	256
sepconv2_2 (SeparableConv2D)	(None, 24, 24, 64)	4,736
bn2_2 (BatchNormalization)	(None, 24, 24, 64)	256
pool2 (MaxPooling2D)	(None, 12, 12, 64)	0
spatial_dropout2 (SpatialDropout2D)	(None, 12, 12, 64)	0
sepconv3_1 (SeparableConv2D)	(None, 12, 12, 96)	6,816
bn3_1 (BatchNormalization)	(None, 12, 12, 96)	384
sepconv3_2 (SeparableConv2D)	(None, 12, 12, 96)	10,176
bn3_2 (BatchNormalization)	(None, 12, 12, 96)	384
pool3 (MaxPooling2D)	(None, 6, 6, 96)	0
spatial_dropout3 (SpatialDropout2D)	(None, 6, 6, 96)	0
sepconv4_1 (SeparableConv2D)	(None, 6, 6, 128)	13,280
bn4_1 (BatchNormalization)	(None, 6, 6, 128)	512
sepconv4_2 (SeparableConv2D)	(None, 6, 6, 128)	17,664
bn4_2 (BatchNormalization)	(None, 6, 6, 128)	512
pool4 (MaxPooling2D)	(None, 3, 3, 128)	0
spatial_dropout4 (SpatialDropout2D)	(None, 3, 3, 128)	0
global_avg_pool (GlobalAveragePooling2D)	(None, 128)	0
dropout_gap (Dropout)	(None, 128)	0
fc1 (Dense)	(None, 128)	16,512
dropout_fc (Dropout)	(None, 128)	0
predictions (Dense)	(None, 7)	903

Total params: 76,711 (299.65 KB)
Trainable params: 75,431 (294.65 KB)
Non-trainable params: 1,280 (5.00 KB)


Training graphs
EPOCH_ACCURACY
Accuracy
Epoch
Train
Validation
Accuracy (Epoch 152)
Train: 0.8296
Validation: 0.803
EPOCH_LOSS
Loss
Epoch
Train
Validation
ANGRY
85.5%
6.0%
0.82


Results
Model version:
Quantized (int8)
ACCURACY
73.56%
Metrics for Classifier
METRIC
VALUE
Area under ROC Curve 0.96
Weighted average Precision 0.79
Weighted average Recall 0.79
Weighted average F1 score 0.79
Confusion matrix
ANGRY HAPPY NEUTRAL SAD SURPRISE UNCERTAIN
ANGRY 75.8% 1.0% 3.6% 2.5% 2.3% 10.9%
HAPPY 3.4% 80.8% 3.2% 1.7% 3.0% 7.9%
NEUTRAL 4.3% 2.6% 63.7% 8.2% 4.2% 16.9%
SAD 6.1% 1.4% 12.4% 59.4% 1.6% 19.1%
SURPRISE 4.2% 0.7% 1.7% 0.7% 86.6% 6.0%
F1 SCORE 0.80 0.86 0.72 0.67 0.87
Feature explorer
angry - correct
happy - correct
neutral - correct
sad - correct
surprise - correct
angry - incorrect
happy - incorrect
neutral - incorrect
sad - incorrect
surprise - incorrect



Custom CNN demo; Peak RAM usage: 158.5 KB; Flash usage 160.6KB

Custom CNN demo; Peak RAM usage: 158.5 KB; Flash usage 160.6KB

Future Work

- Achieved goals of **80%+ training acc** and **>3 FPS inference time** in real-world testing
- **Standardize** real-world testing to calculate meaningful “accuracy”
  - Control lighting, contrast, background
  - Run tests across different people
- Explore **hyperparameter optimization**; mainly tweaked by observations in this project
  - Learning rate
  - Regularization strength
  - Maximum arch. depth
  - Kernel type / size
  - Batch size

References, Thank You!

OpenMV Cam H7 Plus Documentation:

https://openmv.io/products/openmv-cam-h7-plus

https://docs.openmv.io/library/omv.image.html#class-haarcascade-feature-descriptor

Facial Expression Recognition Literature:

https://github.com/vicksam/fer-model

https://github.com/microsoft/FERPlus

Asmara et al., 2024; https://joiv.org/index.php/joiv/article/view/2299/0

Other: https://www.tensorflow.org/