Face & Emotion Recognition using Deep Learning

W251 - Summer 2021

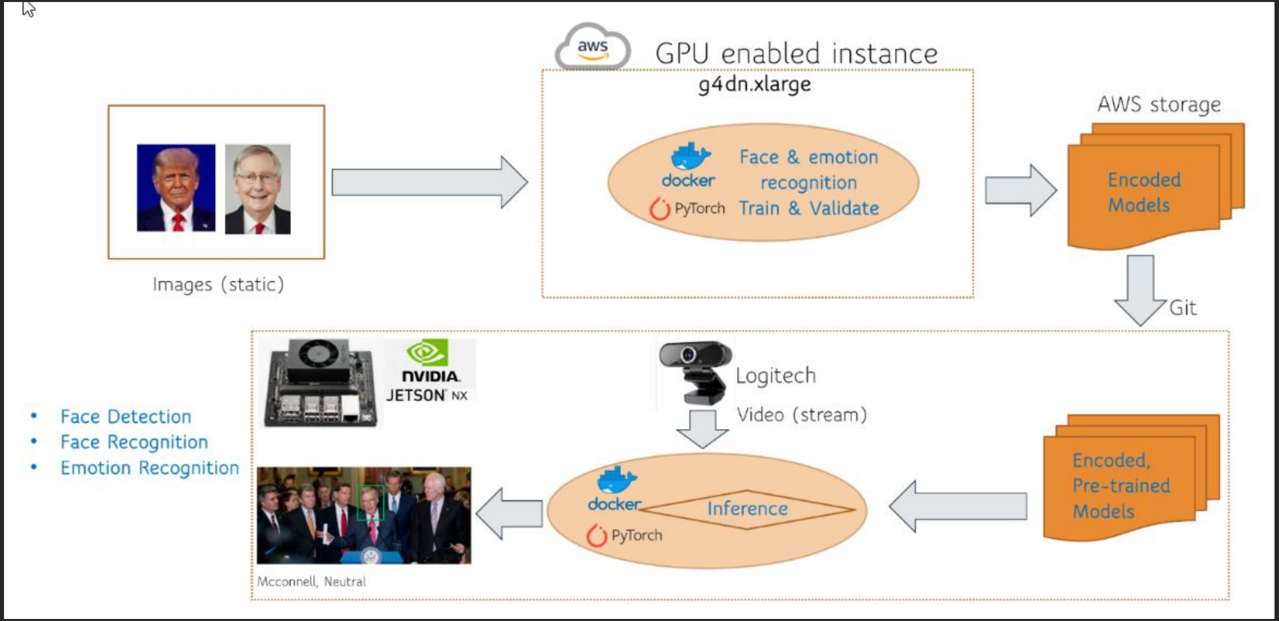
Diana Chacon, Josh Jonte, Jerico Johns, and Sudhrity Mondal

Professors:

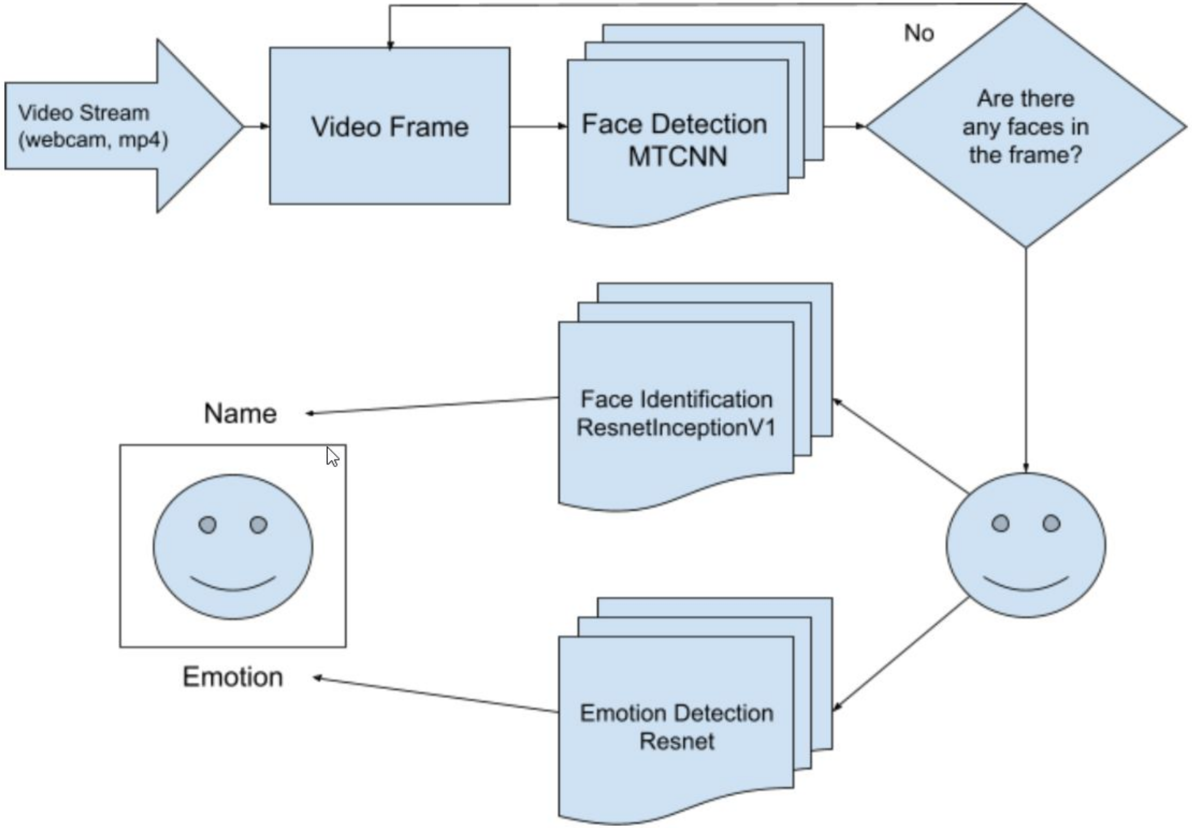
Brad DesAulniers & Darragh Hanley

Introduction

- Architecture
- Face Detection
- Face Recognition
- Emotion Recognition
- Integration
- Implementation
- Demo
- Results
- References

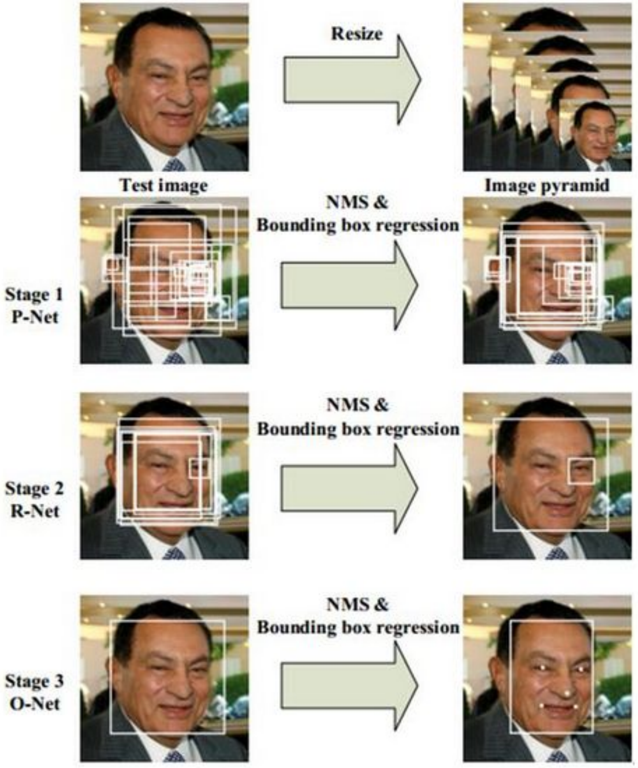


Architecture


Video Stream (webcam, mp4)
Video Frame
Face Detection MTCNN
No
Are there any faces in the frame?
Face Identification ResnetInceptionV1
Name
Emotion
Emotion Detection Resnet

Face Detection - MultiTask Cascaded Neural Network (MTCNN)

- State-of-the-art face detection (99+%)
- Combination of 3 neural networks
  1. Proposal
  2. Refinement
  3. Output
- Pretrained
- Not as vulnerable to lighting and head rotation as Haar (Project #3)
- Able to leverage the GPU



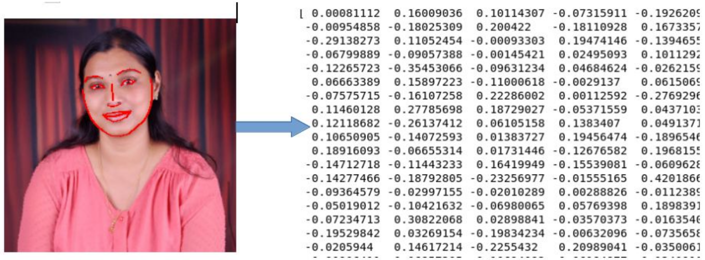
Generating Labelled Data

Applied MTCNN to a saved Zoom video

- Every 20 frames, save a snapshot
- Find the faces
- Save the faces
- Manually sort the faces into directories named after the people
- ~500 images

Face Recognition

Method 1: Facial Feature Distance

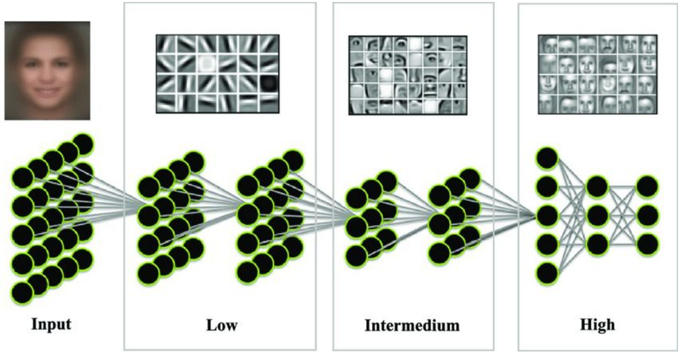


```
[ 0.00081112  0.16009036  0.10114307 -0.07315911 -0.1926209
 -0.00954858 -0.18025309  0.200422   -0.18110928  0.1673357
 -0.29138273  0.11052454 -0.00093303  0.19474146 -0.1394655
 -0.06799889 -0.09057388 -0.00145421  0.02495093  0.1011292
 -0.12265723 -0.35453066 -0.09631234  0.04684624 -0.0262159
  0.06663389  0.15897223 -0.11000618 -0.0029137   0.0615069
 -0.07575715 -0.16107258  0.22286002  0.00112592 -0.2769296
  0.11460128  0.27785698  0.18729027 -0.05371559  0.0437103
  0.12118682 -0.26137412  0.06105158  0.1383407   0.0491371
  0.10650905 -0.14072593  0.01383727  0.19456474 -0.1896546
  0.18916093 -0.06655314  0.01731446 -0.12676582  0.1968155
 -0.14712718 -0.11443233  0.16419949 -0.15539081 -0.0609628
 -0.14277466 -0.18792805 -0.23256977 -0.01555165  0.4201866
 -0.09364579 -0.02997155 -0.02010289  0.00288826 -0.0112389
 -0.05019012 -0.10421632 -0.06980065  0.05769398  0.1898391
 -0.07234713  0.30822068  0.02898841 -0.03570373 -0.0163546
 -0.19529842  0.03269154 -0.19834234 -0.00632096 -0.0735658
 -0.0205944   0.14617214 -0.2255432   0.20989041 -0.0350061
```

Source

Predicted Label = Minimum Euclidean Distance (Embeddings)

Method 2: Image-Based Identification



Source

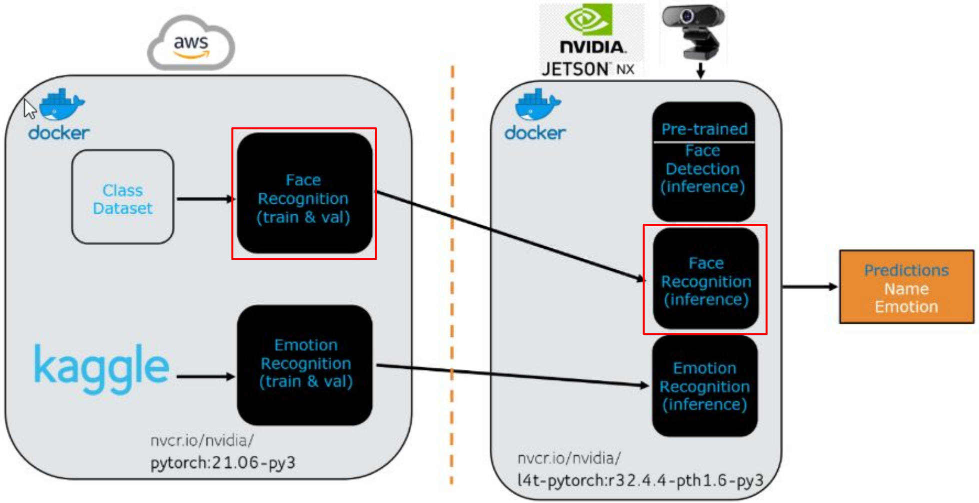
Predicted Label = Max Probability (Pixel Values)

Our primary method due to familiarity and customizability

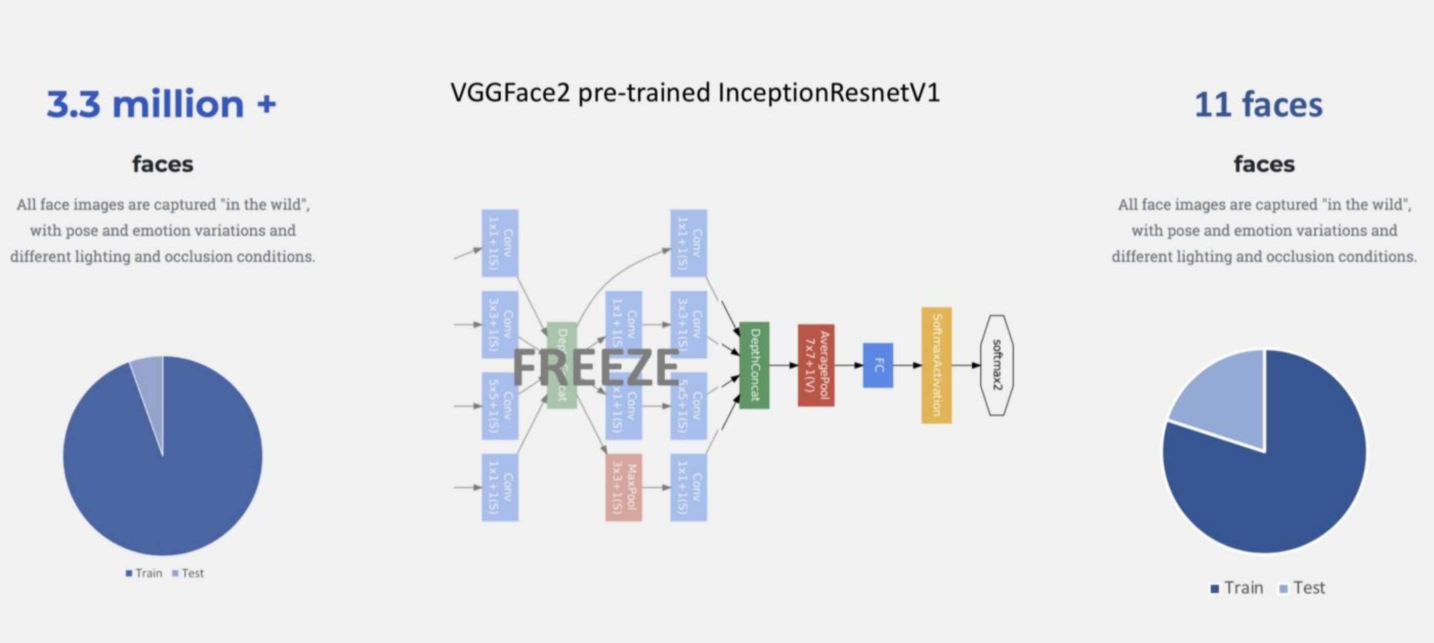
Face Recognition

InceptionResnetV1

- Pretrained on VGGFace2
- Updated on: 11 faces dataset (from class)
- Model
  - Update last convolutional block (5 layers)
  - Adjust dropout layer (p=0.6 → p = 0.2)
  - Parameters
    - epochs = 100
    - loss criterion = CrossEntropyLoss()
    - opt_func = torch.optim.Adam.
    - init_lr = 0.1
    - LR Scheduler = StepLR(step_size = 20, gamma = 0.5)
  - Accuracy: 100%


aws
docker
Class Dataset
Face Recognition (train & val)
kaggle
Emotion Recognition (train & val)
nvcr.io/nvidia/
pytorch:21.06-py3
NVIDIA
JETSON NX
docker
Pre-trained Face Detection (inference)
Face Recognition (inference)
Emotion Recognition (inference)
nvcr.io/nvidia/
l4t-pytorch:r32.4.4-pth1.6-py3
Predictions
Name
Emotion

Face Recognition



3.3 million +

faces

All face images are captured "in the wild", with pose and emotion variations and different lighting and occlusion conditions.

VGGFace2 pre-trained InceptionResnetV1

11 faces

faces

All face images are captured "in the wild", with pose and emotion variations and different lighting and occlusion conditions.

Source

Face Recognition: The Challenge

From Static Images to Live Webcam Footage:

- The Illumination problem
- The Pose problem

Solutions:

Training Transforms:

- transforms.RandomResizedCrop(size = (160,160), scale = (0.7,1.0))
- transforms.RandomHorizontalFlip()
- transforms.RandomPerspective()

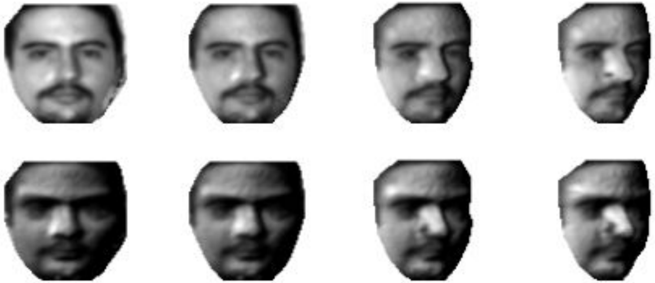
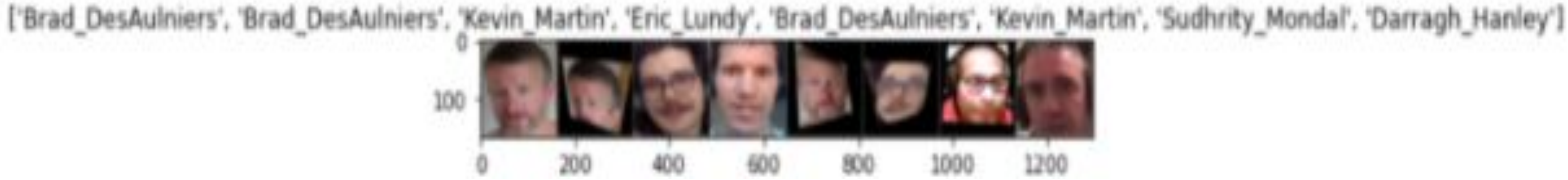


Figure 3: The pose (and illumination) problem.

Source

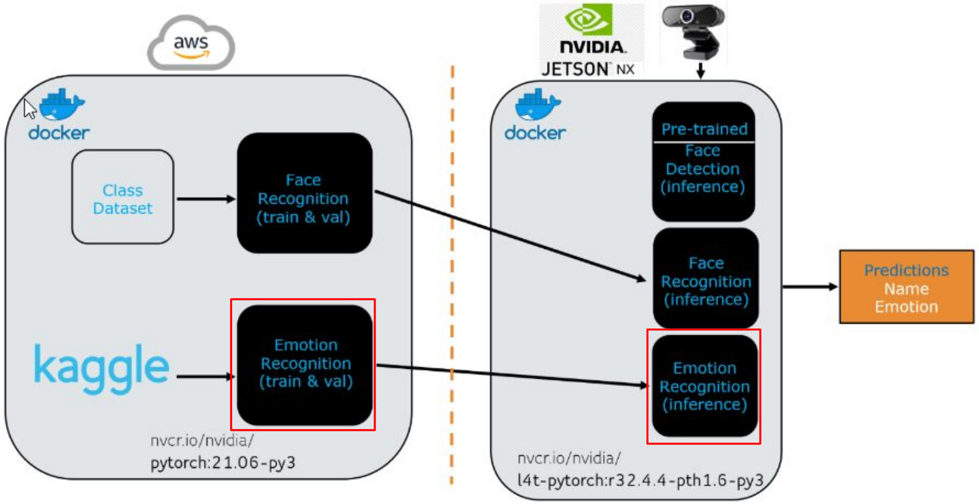


```
['Brad_DesAulniers', 'Brad_DesAulniers', 'Kevin_Martin', 'Eric_Lundy', 'Brad_DesAulniers', 'Kevin_Martin', 'Sudhrity_Mondal', 'Darragh_Hanley']
```

Emotion Recognition

Model 1

- ResNet9 - Based Model
- Dataset: FER2013 Kaggle Image dataset
- Model
  - Three convolution blocks
  - Classes: Neutral, Happy, Surprise, Sad, and Angry
  - Parameters
    - epochs = 64
    - max_lr = 0.008
    - grad_clip = 0.1
    - weight_decay = 1e-4
    - opt_func = torch.optim.Adam
  - Accuracy: 63.7%


aws
docker
Class Dataset
Face Recognition (train & val)
kaggle
Emotion Recognition (train & val)
nvcr.io/nvidia/
pytorch:21.06-py3
NVIDIA
JETSON NX
docker
Pre-trained Face Detection (inference)
Face Recognition (inference)
Emotion Recognition (inference)
nvcr.io/nvidia/
l4t-pytorch:r32.4.4-pth1.6-py3
Predictions
Name
Emotion

Emotion Recognition

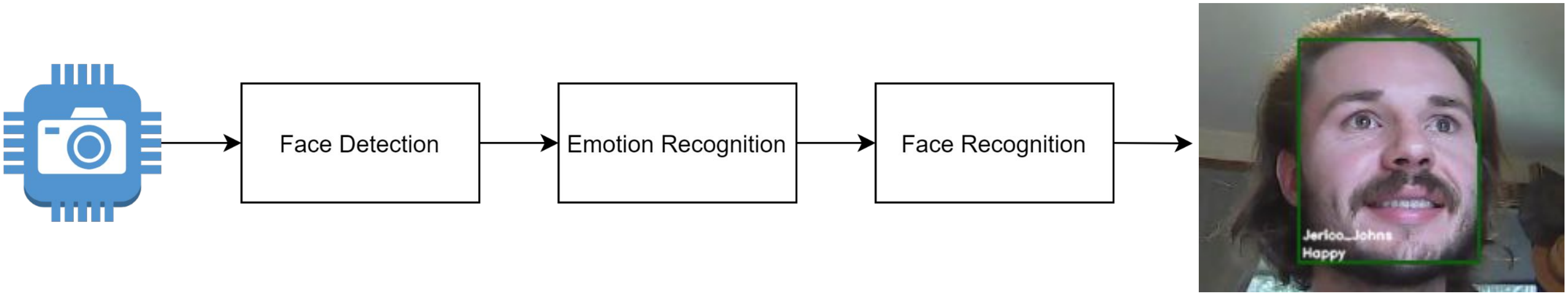
Model 2

- Custom CNN Model
- Dataset: FER2013 Kaggle image dataset
- Model
  - 7 layers
  - Classes: Neutral, Happy, Surprise, Sad, and Angry
  - Parameters
    - epochs = 90
    - max_lr = 0.008
    - grad_clip = 0.1
    - weight_decay = 1e-4
    - opt_func = torch.optim.Adam.
  - Accuracy: 38.8%

Model 3

- Pre-trained model
- Dataset: FER2013 kaggle csv dataset
- Model
  - 7 layers
  - Classes: Neutral, Happy, Surprised, Sad, Angry, Disgust, and Fear.
  - Not able to locate the fer2013.csv file, for this reason we used the pre-trained model
  - Accuracy: 78 - 85%

Integration

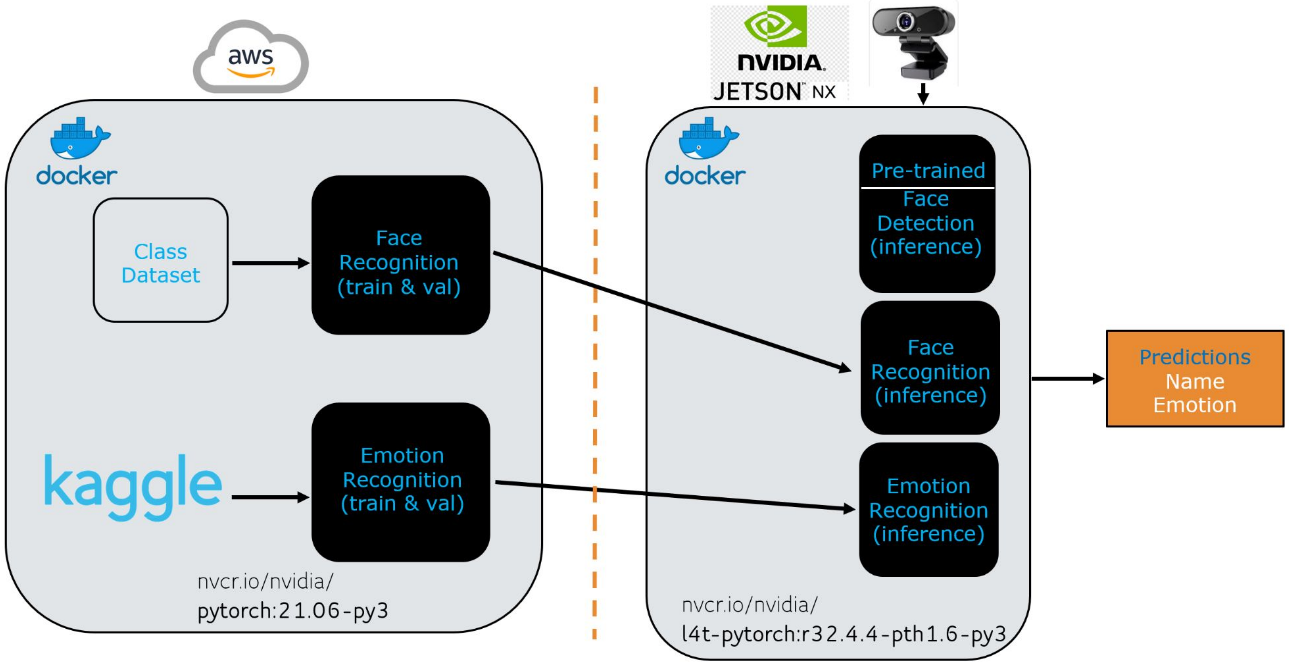


Face Detection

Emotion Recognition

Face Recognition

Implementation


aws
docker
Class Dataset
Face Recognition (train & val)
kaggle
Emotion Recognition (train & val)
nvcr.io/nvidia/
pytorch:21.06-py3
NVIDIA
JETSON NX
docker
Pre-trained
Face Detection (inference)
Face Recognition (inference)
Emotion Recognition (inference)
nvcr.io/nvidia/
l4t-pytorch:r32.4.4-pth1.6-py3
Predictions
Name
Emotion

Demo

Git Repository &
Jupyter Notebook

Results

We integrated Face detection pre-trained, Face recognition transfer and Emotion recognition models to detect faces in a video stream, recognize faces and recognize facial image emotion with ***fair accuracy****. This will be demonstrated during the project presentation.*

Conclusions

1. Using facial features and embeddings may offer more efficient and accurate inference on live facial recognition.
2. We would like to have more time to experiment with various transformations on the training data to overcome accuracy limitations.
3. We believe that if our emotion recognition dataset was more customized for the specific people in our dataset, (e.g. pictures of classmates happy or sad) the results would be more accurate.

Use Cases: Zoom meeting sentiment, Netflix movie rating inference, Improving Dining Experiences.

References

Face Detection

timesler/facenet-pytorch: Pretrained Pytorch face detection (MTCNN) and facial recognition (InceptionResnet) models
Face Detection - https://github.com/timesler/facenet-pytorch

[1604.02878] Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks - https://arxiv.org/abs/1604.02878

Face Recognition

jchaykow/AGN-pytorch - https://github.com/jchaykow/AGN-pytorch & Tutorial on Finetuning Facial Recognition Classifier (Article)

[1710.08092] VGGFace2: A dataset for recognising faces across pose and age - https://arxiv.org/abs/1710.08092

Emotion Recognition

https://www.kaggle.com/dataset/de270025c781ba47a3a6d774a0d670452bfb4dc9d2d6b13740cdb0c17aa7bf2b

https://medium.com/swlh/emotion-detection-using-pytorch-4f6fbfd14b2e

https://github.com/shangeth/Facial-Emotion-Recognition-PyTorch-ONNX

https://www.kaggle.com/balmukund/fer-2013-pytorch-implementation