classify

May 10, 2021

1 Classification Code

Code for classifying videos based on emotions present, and present the results. Note that most of this code is now in the server code (classify_video.py and classify_image.py), as my local machine did not have the power to process the videos quickly. At the end of this notebook is the code I used to display results from the server.

[2]:
```
from tensorflow.keras.models import load_model
from collections import deque
import numpy as np
from sklearn.preprocessing import LabelBinarizer
import argparse
import pickle
import cv2
import glob
import re
from sklearn.metrics import classification_report
```

Set out the parameters of the classification to perform

[2]:
```
MODEL = "fer-vgg"

# One of ravdess, ravdess-faces
DATASET = "ravdess-faces"
FOUR_EMOTIONS = False

if MODEL == "RN50":
    model = load_model('../models/best-models/resnet50.h5')

if MODEL == "fer-RN50":
    model = load_model('../models/best-models/fer-resnet50.h5')

if MODEL == "fer-vgg":
    model = load_model("../models/best-models/resnet50-vgg.h5")
```

[3]:
```
def mean_classify(preds_list):
    results = np.array(preds_list).mean(axis=0)
```

```
    i = np.argmax(results)
    return i

def max_classify(preds_list):
    results = np.array(preds_list).max(axis=0)
    i = np.argmax(results)
    return i
```

This section iterates through the video, calculating a list of predictions (preds_list).

[4]:
```
lb = pickle.loads(open("fer_label_bin", "rb").read())
if MODEL == "fer-ivadym":
    lb.classes_ = ["angry", "disgust", "fearful", "happy", "sad", "surprised",␣
 ↪"neutral"]
    print("hi")


face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +␣
 ↪'haarcascade_frontalface_default.xml')
grayscale = True

def get_preds_list(filename):
    mean = np.array([123.68, 116.779, 103.939][::1], dtype="float32")

    frames = []

    vs = cv2.VideoCapture(filename)

    # Loop over video frames
    while True:
        (grabbed, frame) = vs.read()
        if not grabbed:
            break

        if grayscale:
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            frame = cv2.cvtColor(frame, cv2.COLOR_GRAY2RGB)
        else:
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        frame = cv2.resize(frame, (197, 197)).astype("float32")
        frame -= 128.8006
        frame /= 64.6497
        frames.append(frame)


    preds_arr = model.predict(np.array(frames))
```

```
    vs.release()
    return preds_arr
```

hi

[ ]:
```
vidlist = glob.glob("../data/dataset/{}/*-30-720.mp4".format(DATASET))

emotion_dict = {
        1: "neutral",
        2: "calm",
        3: "happy",
        4: "sad",
        5: "angry",
        6: "fearful",
        7: "disgust",
        8: "surprised"
    }

actual = []
predictions = []

for i, video in enumerate(vidlist):
    print("Processing video {}/{}.".format(i, len(vidlist)))
    emotion = emotion_dict[int(re.search('(?<=\/0)\d', video).group())]
    if emotion in lb.classes_:
        actual.append(emotion)

        preds_list = get_preds_list(video)
        pred = mean_classify(preds_list)

        predictions.append(pred)

actual = lb.transform(actual).argmax(axis=1)

print(classification_report(actual,
        predictions, labels = range(8), target_names=lb.classes_))
```

1.1 Displaying server results

In this section we load classification results from the server, and display them in a neat way

[15]:
```
for res in [144, 360, 720]:
    for fps in [5, 15, 30]:
```

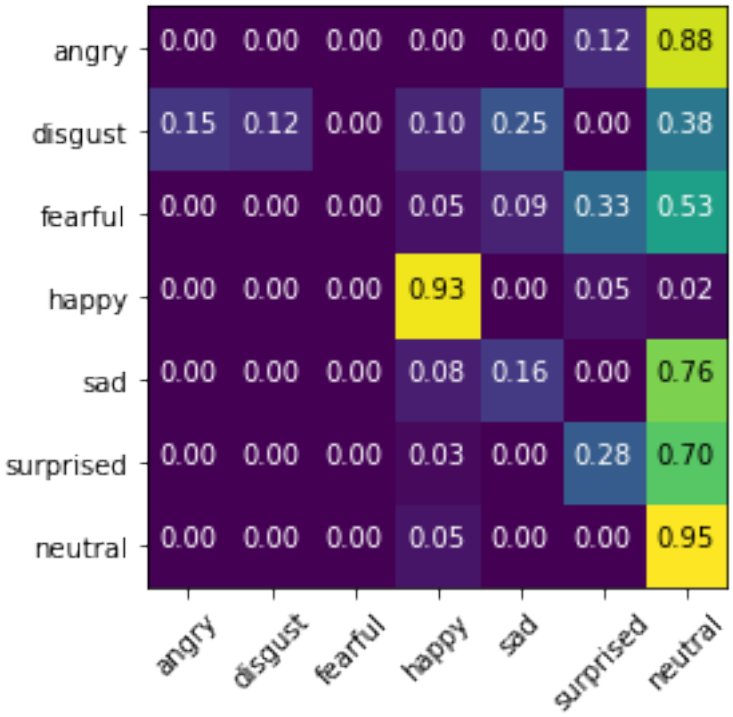
```
      ␣
↪print("------------------------------------------------------------------------{}␣
↪p / {} fps".format(res, fps))
       with open("results/ravdess-faces-RN50-vgg-{}-{}-mean.pickle".
↪format(fps, res), "rb") as handle:
           results_dict = pickle.load(handle)

       predictions = results_dict['predictions']
       lb = results_dict['lb']
       actual = results_dict['actual']

       print(classification_report(actual,
           predictions, labels = range(len(lb.classes_)), target_names=lb.
↪classes_, zero_division=0))

       matrix = confusion_matrix(actual, predictions, normalize='true')

       plt.figure()
       plt.imshow(matrix, interpolation="nearest")

       target_names = lb.classes_
       tick_marks = np.arange(len(target_names))
       plt.xticks(tick_marks, target_names, rotation=45)
       plt.yticks(tick_marks, target_names)


       thresh = matrix.max() / 1.5
       for i, j in itertools.product(range(matrix.shape[0]), range(matrix.
↪shape[1])):
           plt.text(j, i, "{:0.2f}".format(matrix[i, j]),
                    horizontalalignment="center",
                    color="white" if matrix[i, j] < thresh else "black")
```

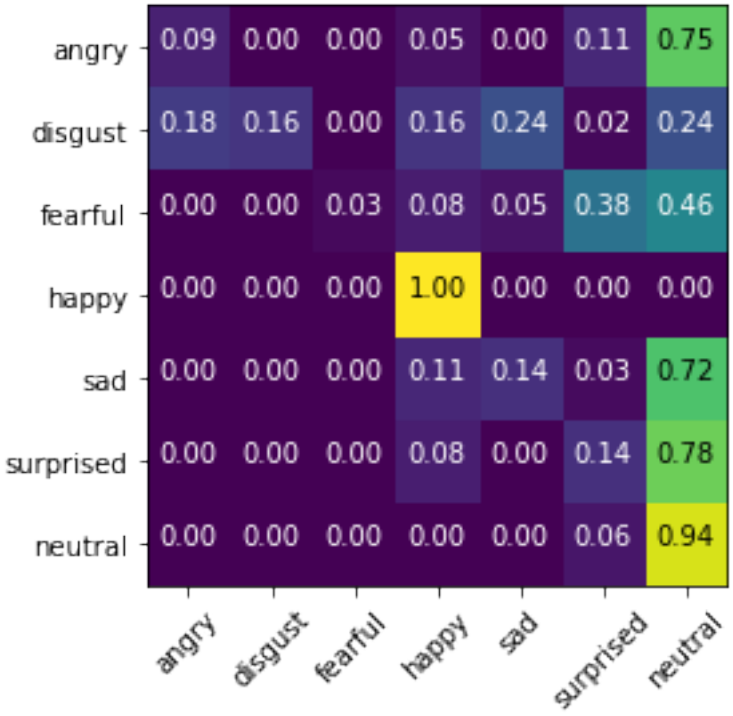
```
------------------------------------------------------------------------144 p
/ 5 fps
              precision    recall  f1-score   support

       angry       0.00      0.00      0.00        41
     disgust       1.00      0.12      0.22        40
     fearful       0.00      0.00      0.00        43
       happy       0.80      0.93      0.86        42
         sad       0.22      0.16      0.19        25
   surprised       0.34      0.28      0.31        40
     neutral       0.13      0.95      0.23        19

    accuracy                           0.31       250
   macro avg       0.36      0.35      0.26       250
```

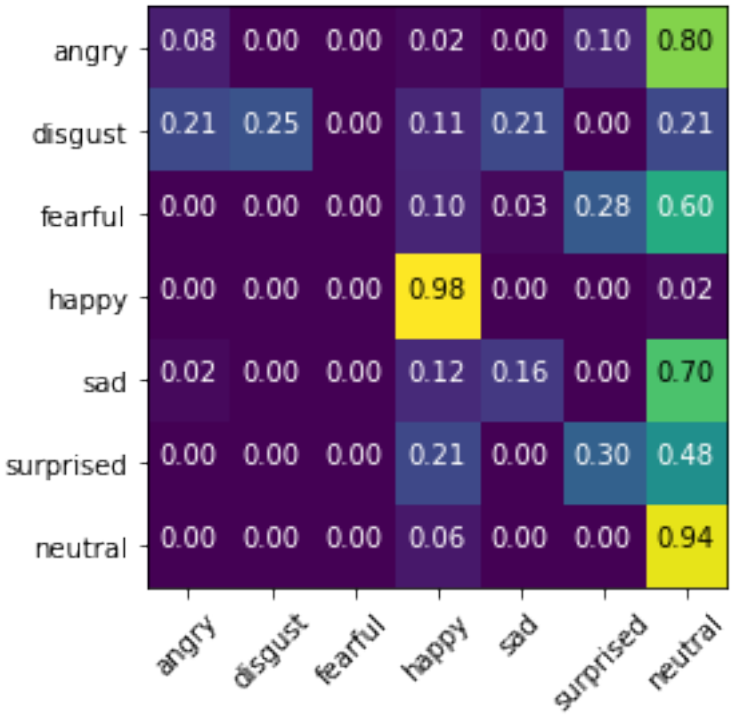
```
weighted avg       0.38      0.31      0.26       250

------------------------------------------------------------------------------144 p
/ 15 fps
              precision    recall  f1-score   support

       angry       0.33      0.09      0.14        44
     disgust       1.00      0.16      0.27        45
     fearful       1.00      0.03      0.05        37
       happy       0.65      1.00      0.79        35
         sad       0.28      0.14      0.19        36
   surprised       0.19      0.14      0.16        36
     neutral       0.12      0.94      0.22        17

    accuracy                           0.29       250
   macro avg       0.51      0.36      0.26       250
weighted avg       0.55      0.29      0.26       250

------------------------------------------------------------------------------144 p
/ 30 fps
              precision    recall  f1-score   support

       angry       0.36      0.08      0.13        49
     disgust       1.00      0.25      0.40        28
     fearful       0.00      0.00      0.00        40
       happy       0.66      0.98      0.78        41
         sad       0.50      0.16      0.25        43
   surprised       0.38      0.30      0.34        33
     neutral       0.11      0.94      0.20        16

    accuracy                           0.33       250
   macro avg       0.43      0.39      0.30       250
weighted avg       0.43      0.33      0.30       250

------------------------------------------------------------------------------360 p
/ 5 fps
              precision    recall  f1-score   support

       angry       0.25      0.05      0.09        38
     disgust       1.00      0.12      0.21        42
     fearful       1.00      0.05      0.09        44
       happy       0.54      0.84      0.66        31
         sad       0.33      0.17      0.22        36
   surprised       0.38      0.31      0.34        39
     neutral       0.12      0.85      0.22        20

    accuracy                           0.28       250
   macro avg       0.52      0.34      0.26       250
```

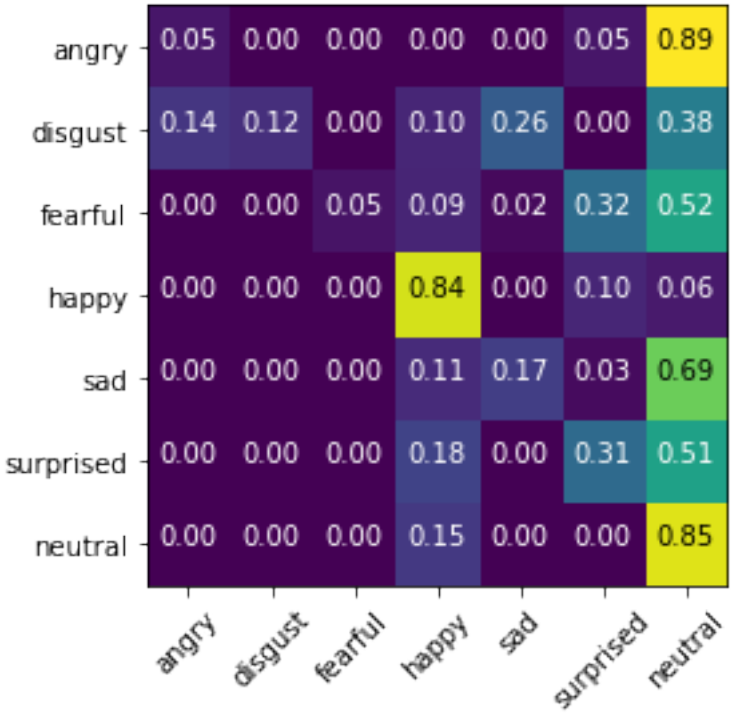
```
weighted avg       0.57      0.28      0.25       250

--------------------------------------------------------------------------360 p
/ 15 fps
              precision    recall  f1-score   support

       angry       0.00      0.00      0.00        30
     disgust       1.00      0.09      0.17        43
     fearful       0.00      0.00      0.00        44
       happy       0.61      0.92      0.73        39
         sad       0.40      0.22      0.28        37
   surprised       0.30      0.23      0.26        35
     neutral       0.14      0.82      0.24        22

    accuracy                           0.30       250
   macro avg       0.35      0.33      0.24       250
weighted avg       0.38      0.30      0.24       250

--------------------------------------------------------------------------360 p
/ 30 fps
              precision    recall  f1-score   support

       angry       0.18      0.06      0.09        34
     disgust       1.00      0.09      0.17        44
     fearful       0.00      0.00      0.00        33
       happy       0.64      0.93      0.76        42
         sad       0.21      0.09      0.12        35
   surprised       0.39      0.34      0.36        41
     neutral       0.16      0.95      0.28        21

    accuracy                           0.33       250
   macro avg       0.37      0.35      0.25       250
weighted avg       0.42      0.33      0.27       250

--------------------------------------------------------------------------720 p
/ 5 fps
              precision    recall  f1-score   support

       angry       0.10      0.02      0.04        42
     disgust       1.00      0.11      0.19        38
     fearful       1.00      0.02      0.05        43
       happy       0.58      0.93      0.72        30
         sad       0.27      0.11      0.16        35
   surprised       0.50      0.42      0.46        38
     neutral       0.16      0.92      0.27        24

    accuracy                           0.30       250
   macro avg       0.52      0.36      0.27       250
```

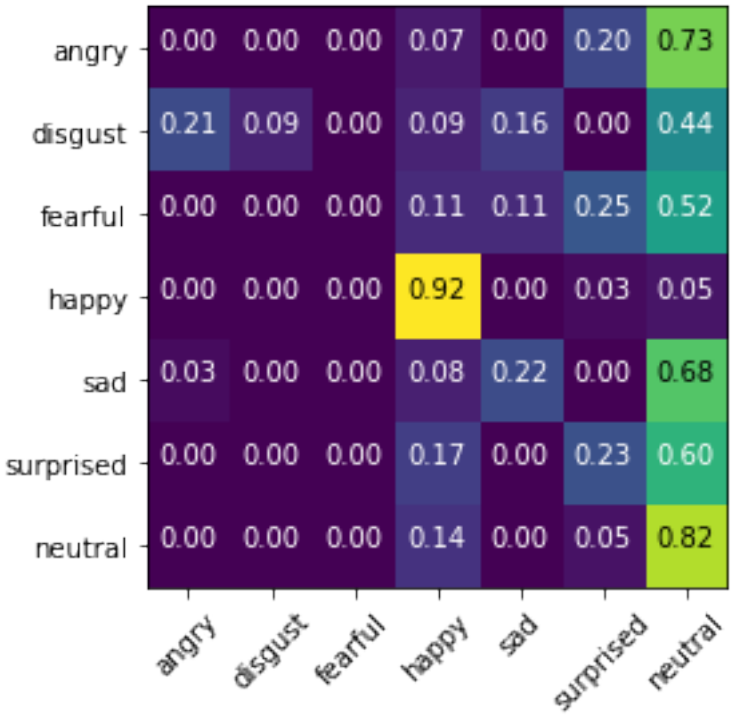
```
weighted avg       0.54      0.30      0.25       250

--------------------------------------------------------------------------------720 p
/ 15 fps
              precision    recall  f1-score   support

       angry       0.33      0.09      0.14        45
     disgust       1.00      0.11      0.20        37
     fearful       0.00      0.00      0.00        37
       happy       0.60      0.92      0.73        36
         sad       0.35      0.22      0.27        32
   surprised       0.34      0.28      0.31        43
     neutral       0.14      0.85      0.24        20

    accuracy                           0.31       250
   macro avg       0.39      0.35      0.27       250
weighted avg       0.41      0.31      0.26       250

--------------------------------------------------------------------------------720 p
/ 30 fps
              precision    recall  f1-score   support

       angry       0.08      0.03      0.04        38
     disgust       1.00      0.10      0.18        52
     fearful       0.00      0.00      0.00        40
       happy       0.74      0.94      0.83        33
         sad       0.13      0.08      0.10        39
   surprised       0.26      0.29      0.28        31
     neutral       0.11      0.88      0.20        17

    accuracy                           0.26       250
   macro avg       0.33      0.33      0.23       250
weighted avg       0.38      0.26      0.21       250
```

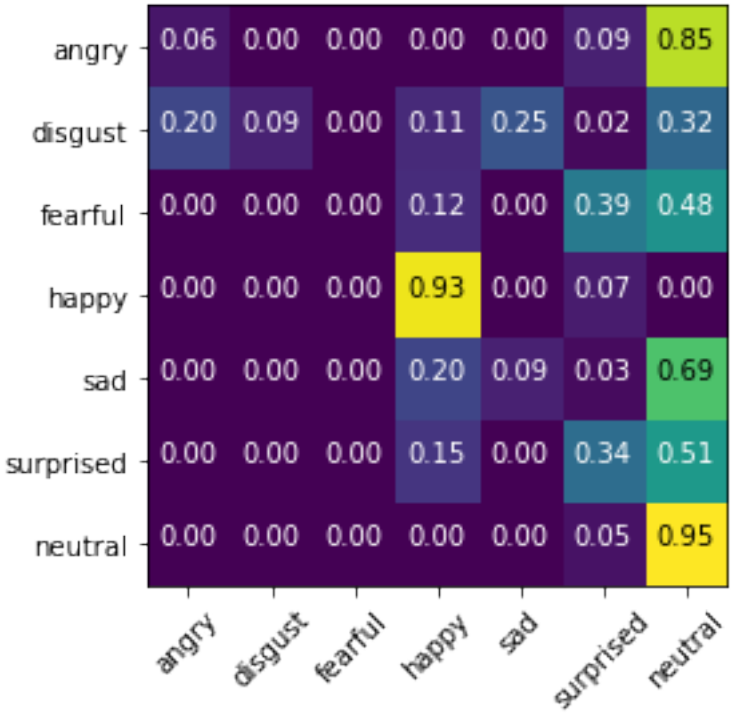

angry 0.00 0.00 0.00 0.00 0.00 0.12 0.88
disgust 0.15 0.12 0.00 0.10 0.25 0.00 0.38
fearful 0.00 0.00 0.00 0.05 0.09 0.33 0.53
happy 0.00 0.00 0.00 0.93 0.00 0.05 0.02
sad 0.00 0.00 0.00 0.08 0.16 0.00 0.76
surprised 0.00 0.00 0.00 0.03 0.00 0.28 0.70
neutral 0.00 0.00 0.00 0.05 0.00 0.00 0.95
angry disgust fearful happy sad surprised neutral

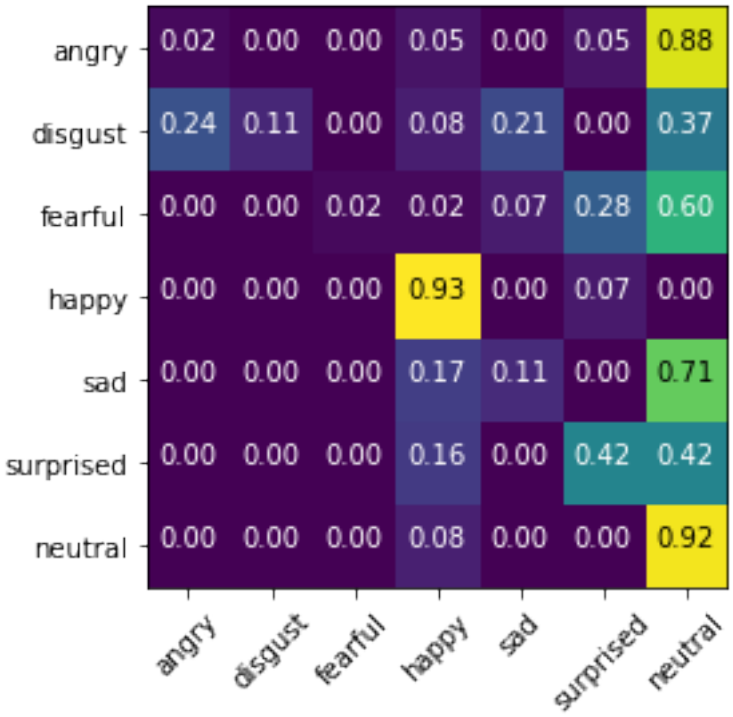

angry 0.09 0.00 0.00 0.05 0.00 0.11 0.75
disgust 0.18 0.16 0.00 0.16 0.24 0.02 0.24
fearful 0.00 0.00 0.03 0.08 0.05 0.38 0.46
happy 0.00 0.00 0.00 1.00 0.00 0.00 0.00
sad 0.00 0.00 0.00 0.11 0.14 0.03 0.72
surprised 0.00 0.00 0.00 0.08 0.00 0.14 0.78
neutral 0.00 0.00 0.00 0.00 0.00 0.06 0.94
angry disgust fearful happy sad surprised neutral

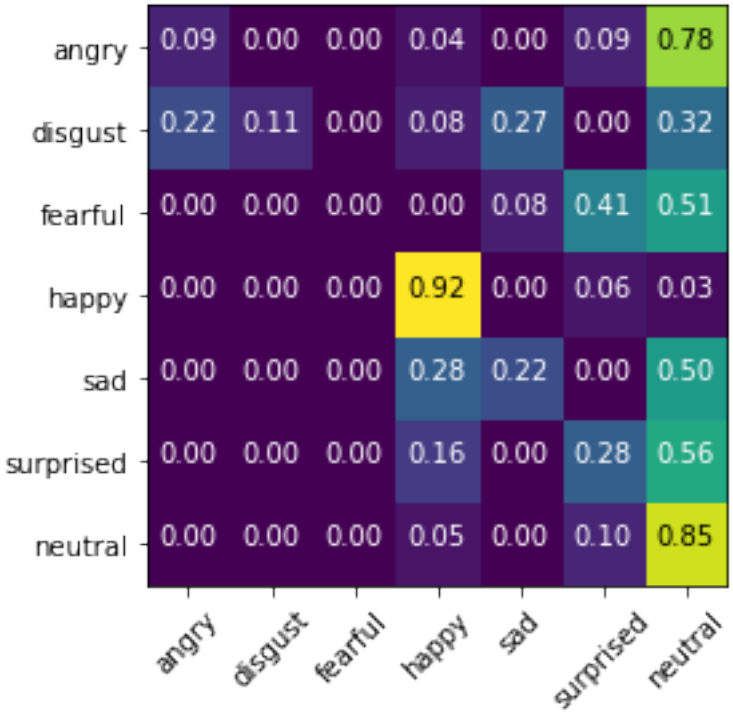

angry 0.08 0.00 0.00 0.02 0.00 0.10 0.80
disgust 0.21 0.25 0.00 0.11 0.21 0.00 0.21
fearful 0.00 0.00 0.00 0.10 0.03 0.28 0.60
happy 0.00 0.00 0.00 0.98 0.00 0.00 0.02
sad 0.02 0.00 0.00 0.12 0.16 0.00 0.70
surprised 0.00 0.00 0.00 0.21 0.00 0.30 0.48
neutral 0.00 0.00 0.00 0.06 0.00 0.00 0.94
angry disgust fearful happy sad surprised neutral

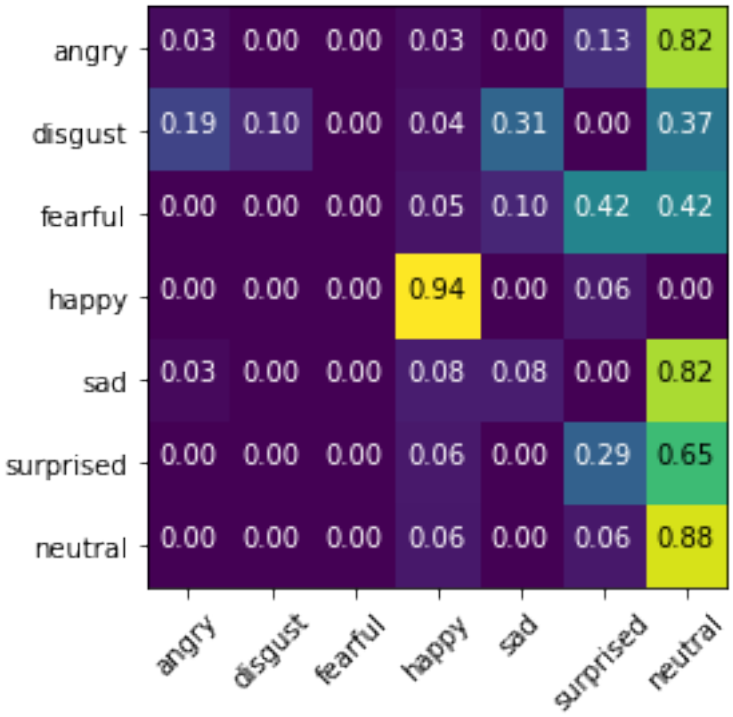

angry 0.05 0.00 0.00 0.00 0.00 0.05 0.89
disgust 0.14 0.12 0.00 0.10 0.26 0.00 0.38
fearful 0.00 0.00 0.05 0.09 0.02 0.32 0.52
happy 0.00 0.00 0.00 0.84 0.00 0.10 0.06
sad 0.00 0.00 0.00 0.11 0.17 0.03 0.69
surprised 0.00 0.00 0.00 0.18 0.00 0.31 0.51
neutral 0.00 0.00 0.00 0.15 0.00 0.00 0.85
angry disgust fearful happy sad surprised neutral


angry 0.00 0.00 0.00 0.07 0.00 0.20 0.73
disgust 0.21 0.09 0.00 0.09 0.16 0.00 0.44
fearful 0.00 0.00 0.00 0.11 0.11 0.25 0.52
happy 0.00 0.00 0.00 0.92 0.00 0.03 0.05
sad 0.03 0.00 0.00 0.08 0.22 0.00 0.68
surprised 0.00 0.00 0.00 0.17 0.00 0.23 0.60
neutral 0.00 0.00 0.00 0.14 0.00 0.05 0.82
angry disgust fearful happy sad surprised neutral


angry 0.06 0.00 0.00 0.00 0.00 0.09 0.85
disgust 0.20 0.09 0.00 0.11 0.25 0.02 0.32
fearful 0.00 0.00 0.00 0.12 0.00 0.39 0.48
happy 0.00 0.00 0.00 0.93 0.00 0.07 0.00
sad 0.00 0.00 0.00 0.20 0.09 0.03 0.69
surprised 0.00 0.00 0.00 0.15 0.00 0.34 0.51
neutral 0.00 0.00 0.00 0.00 0.00 0.05 0.95
angry disgust fearful happy sad surprised neutral


angry 0.02 0.00 0.00 0.05 0.00 0.05 0.88
disgust 0.24 0.11 0.00 0.08 0.21 0.00 0.37
fearful 0.00 0.00 0.02 0.02 0.07 0.28 0.60
happy 0.00 0.00 0.00 0.93 0.00 0.07 0.00
sad 0.00 0.00 0.00 0.17 0.11 0.00 0.71
surprised 0.00 0.00 0.00 0.16 0.00 0.42 0.42
neutral 0.00 0.00 0.00 0.08 0.00 0.00 0.92
angry disgust fearful happy sad surprised neutral


angry 0.09 0.00 0.00 0.04 0.00 0.09 0.78
disgust 0.22 0.11 0.00 0.08 0.27 0.00 0.32
fearful 0.00 0.00 0.00 0.00 0.08 0.41 0.51
happy 0.00 0.00 0.00 0.92 0.00 0.06 0.03
sad 0.00 0.00 0.00 0.28 0.22 0.00 0.50
surprised 0.00 0.00 0.00 0.16 0.00 0.28 0.56
neutral 0.00 0.00 0.00 0.05 0.00 0.10 0.85
angry disgust fearful happy sad surprised neutral



1.2 Classification test

This is a basic test of the classifier - it simply overalys the classifiers predictions over a video in order to check whether the classification matches my perception

[ ]:
```
grayscale = True


DATASET = "ravdess-faces"
mean = np.array([123.68, 116.779, 103.939][::1], dtype="float32")

vidlist = glob.glob("../data/dataset/{}/*-30-144.mp4".format(DATASET))

for i, video in enumerate(vidlist):

    vs = cv2.VideoCapture(video)

    # Loop over video frames
    frame_list = []
    while True:
        (grabbed, frame) = vs.read()
        if not grabbed:
            break
```

```
        output = frame.copy()

        if grayscale:
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            frame = cv2.cvtColor(frame, cv2.COLOR_GRAY2RGB)
        else:
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        output = frame

        frame = cv2.resize(frame, (197, 197)).astype("float32")
        frame -= 128.8006
        frame /= 64.6497
        frame_list.append(frame)

        if len(frame_list) >= 32:
            frames = np.array(frame_list)
            preds = model.predict(np.expand_dims(frames, axis=0))[0]
            frame_list.pop(0)


            i = np.argmax(preds)
            preds[i] = 0
            i =  np.argmax(preds)
            label = lb.classes_[i]

            text = "{}".format(label)
            cv2.putText(output, text, (20, 20), cv2.FONT_HERSHEY_SIMPLEX, 1.25,␣
↪(0, 255, 0), 5)

        cv2.imshow("Output", output)
        key = cv2.waitKey(1) & 0xFF


        # if the `q` key was pressed, break from the loop
        if key == ord("q"):
            break

    vs.release()
cv2.destroyWindow('Output')
```

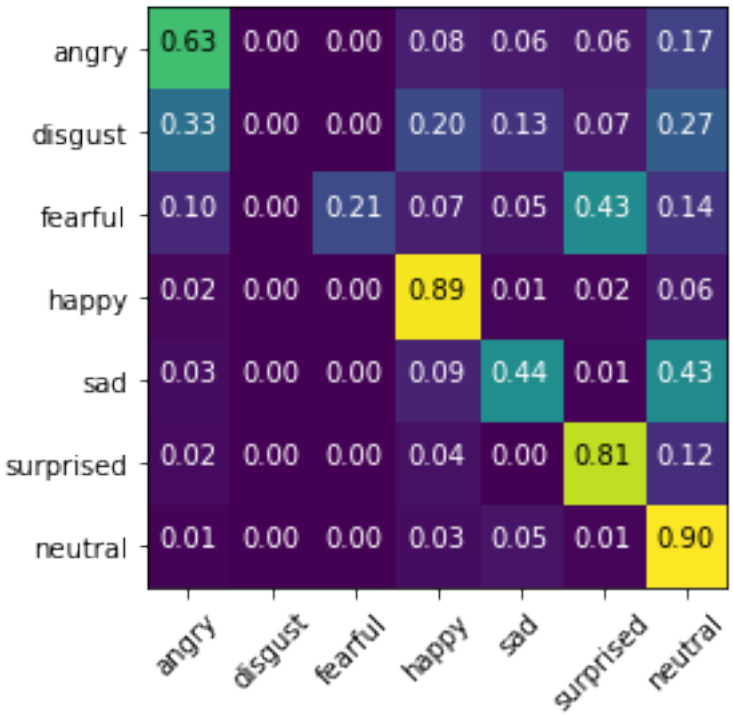
1.3 Image classification results

This section loads the results of our image classification from the server, and displays them as a confusion matrix and classification report.

[5]:
```
import pickle
from sklearn.metrics import confusion_matrix
import itertools
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import classification_report, accuracy_score

with open("../../../notebooks/results/my-imagenet-rn50.pickle", "rb") as handle:
    results_dict = pickle.load(handle)

predictions = results_dict['predictions']
lb = results_dict['lb']
actual = results_dict['actual']

print(classification_report(actual,
    predictions, labels = range(len(lb.classes_)), target_names=lb.classes_,␣
 ↪zero_division=0))

print("accuracy : {}".format(accuracy_score(actual, predictions)))

matrix = confusion_matrix(actual, predictions, normalize='true')

plt.figure()
plt.imshow(matrix, interpolation="nearest")

target_names = lb.classes_
tick_marks = np.arange(len(target_names))
plt.xticks(tick_marks, target_names, rotation=45)
plt.yticks(tick_marks, target_names)


thresh = matrix.max() / 1.5
for i, j in itertools.product(range(matrix.shape[0]), range(matrix.shape[1])):
    plt.text(j, i, "{:0.2f}".format(matrix[i, j]),
             horizontalalignment="center",
             color="white" if matrix[i, j] < thresh else "black")
```

```
              precision    recall  f1-score   support

       angry       0.72      0.63      0.67       266
     disgust       0.00      0.00      0.00        15
     fearful       0.89      0.21      0.34        81
       happy       0.88      0.89      0.88       895
         sad       0.66      0.44      0.53       385
   surprised       0.79      0.81      0.80       405
     neutral       0.75      0.90      0.82      1101
```

```
    accuracy                           0.78      3148
   macro avg       0.67      0.55      0.58      3148
weighted avg       0.78      0.78      0.77      3148

accuracy : 0.7836721728081322
```



[ ]: