FCN-FFA-30E-14L-basis-test-02

March 24, 2021

1 Are Relations Relevant in CNNs? *A Study Based on a Facial Dataset*

1.1 Testing FCN with Features Further Apart *(30 Epochs - 13 Layers)*

1.1.1 Imports, Seed, GPU integration

[1]:
```
import numpy as np
import random
import tensorflow as tf
```

[2]:
```
# Seeds for better reproducibility
seed = 42
np.random.seed(seed)
random.seed(seed)
tf.random.set_seed(seed)
```

[3]:
```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.metrics import confusion_matrix
import itertools
import matplotlib.pyplot as plt
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
%matplotlib inline
```

[4]:
```
physical_devices = tf.config.experimental.list_physical_devices('GPU')
print("Num GPUs Available: ", len(physical_devices))
tf.config.experimental.set_memory_growth(physical_devices[0], True)
```

```
Num GPUs Available:  1
```

1.1.2 Data preparation

[5]:
```
test_path = '../../../picasso_dataset/basis-data/middle/test'
```

[6]:
```
test_batches = ImageDataGenerator(preprocessing_function=tf.keras.applications.
 ↪vgg16.preprocess_input) \
```

```
    .flow_from_directory(directory=test_path, target_size=(224,224),␣
↪classes=['no_face', 'face'], batch_size=10, shuffle=False)
```

```
Found 3000 images belonging to 2 classes.
```

[7]:
```
assert test_batches.n == 3000
assert test_batches.num_classes == 2
```

1.1.3 Loading the trained FCN

[8]:
```
filename='../models/FCN-FFA-30E-14L-02.h5'
loaded_model = load_model(filename)
```

1.1.4 Accuracy and loss of the trained model

[9]:
```
scores = loaded_model.evaluate(test_batches, verbose=2)
print("Accuracy: %.2f%%" % (scores[1]*100))
print("Loss: %.2f%%" % (scores[0]*100))
```

```
300/300 - 7s - loss: 0.5801 - accuracy: 0.8757
Accuracy: 87.57%
Loss: 58.01%
```

1.1.5 Testing the FCN

[10]:
```
predictions = loaded_model.predict(x=test_batches, steps=len(test_batches),␣
↪verbose=0)
```

1.1.6 Index of wrongly predicted pictures

[11]:
```
y_true=test_batches.classes
y_pred=np.argmax(predictions, axis=-1)
cm = confusion_matrix(y_true = y_true, y_pred = y_pred)
```

[12]:
```
face_but_predicted_no_face=[]
no_face_but_predicted_face=[]

for i in range(len(predictions)):
        if y_true[i] != y_pred[i]:
            if y_true[i] == 1:
                face_but_predicted_no_face.append(i+8001-1500) #Index of file␣
↪on disk
            else:
                no_face_but_predicted_face.append(i+8001) #Index of file on disk
```

```
print("Data from class 'face', that was wrongly predicted as 'no-face' [",␣
 ↪len(face_but_predicted_no_face), "] :")
print(face_but_predicted_no_face)
print("-------------------------------------------------------------------------------
print("Data from class 'no-face', that was wrongly predicted as 'face' [",␣
 ↪len(no_face_but_predicted_face), "] :")
print(no_face_but_predicted_face)
```

```
Data from class 'face', that was wrongly predicted as 'no-face' [ 359 ] :
[8018, 8019, 8020, 8026, 8027, 8035, 8036, 8039, 8041, 8046, 8057, 8062, 8064,
8067, 8068, 8075, 8077, 8078, 8085, 8088, 8094, 8095, 8096, 8099, 8101, 8104,
8116, 8124, 8126, 8127, 8129, 8130, 8136, 8141, 8144, 8146, 8147, 8148, 8149,
8160, 8165, 8170, 8174, 8181, 8182, 8186, 8188, 8191, 8194, 8200, 8201, 8214,
8217, 8218, 8219, 8224, 8232, 8235, 8240, 8244, 8250, 8252, 8256, 8259, 8264,
8270, 8274, 8279, 8287, 8296, 8297, 8303, 8306, 8307, 8311, 8324, 8326, 8329,
8331, 8341, 8342, 8343, 8355, 8357, 8358, 8363, 8365, 8380, 8381, 8388, 8390,
8394, 8398, 8400, 8401, 8407, 8408, 8410, 8413, 8414, 8415, 8431, 8434, 8436,
8441, 8452, 8458, 8462, 8471, 8474, 8478, 8481, 8491, 8493, 8494, 8496, 8498,
8499, 8500, 8508, 8514, 8517, 8525, 8526, 8530, 8531, 8536, 8537, 8542, 8546,
8547, 8550, 8554, 8555, 8557, 8561, 8564, 8565, 8575, 8581, 8582, 8583, 8594,
8604, 8608, 8612, 8613, 8614, 8625, 8627, 8628, 8634, 8639, 8643, 8645, 8646,
8650, 8657, 8658, 8661, 8662, 8663, 8664, 8668, 8669, 8671, 8672, 8686, 8689,
8690, 8694, 8696, 8699, 8700, 8706, 8709, 8712, 8720, 8721, 8725, 8727, 8732,
8737, 8739, 8743, 8744, 8745, 8746, 8750, 8758, 8759, 8760, 8762, 8766, 8769,
8774, 8776, 8784, 8786, 8788, 8793, 8798, 8801, 8802, 8809, 8819, 8822, 8823,
8830, 8835, 8836, 8840, 8843, 8849, 8855, 8859, 8867, 8870, 8874, 8888, 8890,
8894, 8897, 8900, 8903, 8908, 8911, 8918, 8920, 8929, 8931, 8932, 8934, 8943,
8945, 8948, 8957, 8958, 8963, 8975, 8990, 8992, 8997, 9001, 9013, 9027, 9028,
9032, 9040, 9042, 9045, 9047, 9050, 9056, 9057, 9064, 9071, 9073, 9076, 9081,
9086, 9087, 9099, 9103, 9107, 9118, 9128, 9131, 9133, 9138, 9145, 9148, 9151,
9161, 9171, 9172, 9173, 9175, 9179, 9180, 9185, 9186, 9189, 9192, 9194, 9195,
9203, 9216, 9218, 9225, 9228, 9234, 9237, 9239, 9240, 9247, 9259, 9262, 9265,
9272, 9277, 9284, 9285, 9286, 9289, 9293, 9298, 9299, 9311, 9312, 9317, 9321,
9325, 9326, 9330, 9334, 9337, 9345, 9346, 9368, 9370, 9371, 9372, 9374, 9375,
9376, 9377, 9378, 9381, 9383, 9389, 9391, 9394, 9400, 9405, 9411, 9412, 9413,
9415, 9419, 9426, 9431, 9435, 9439, 9443, 9446, 9449, 9459, 9462, 9463, 9469,
9471, 9477, 9479, 9484, 9485, 9493, 9495, 9497]
--------------------------------------------------------------------------------
--------------
Data from class 'no-face', that was wrongly predicted as 'face' [ 14 ] :
[8019, 8059, 8226, 8266, 8474, 8483, 8549, 8576, 8840, 8843, 8991, 9344, 9438,
9456]
```

1.1.7 Confusion matrix

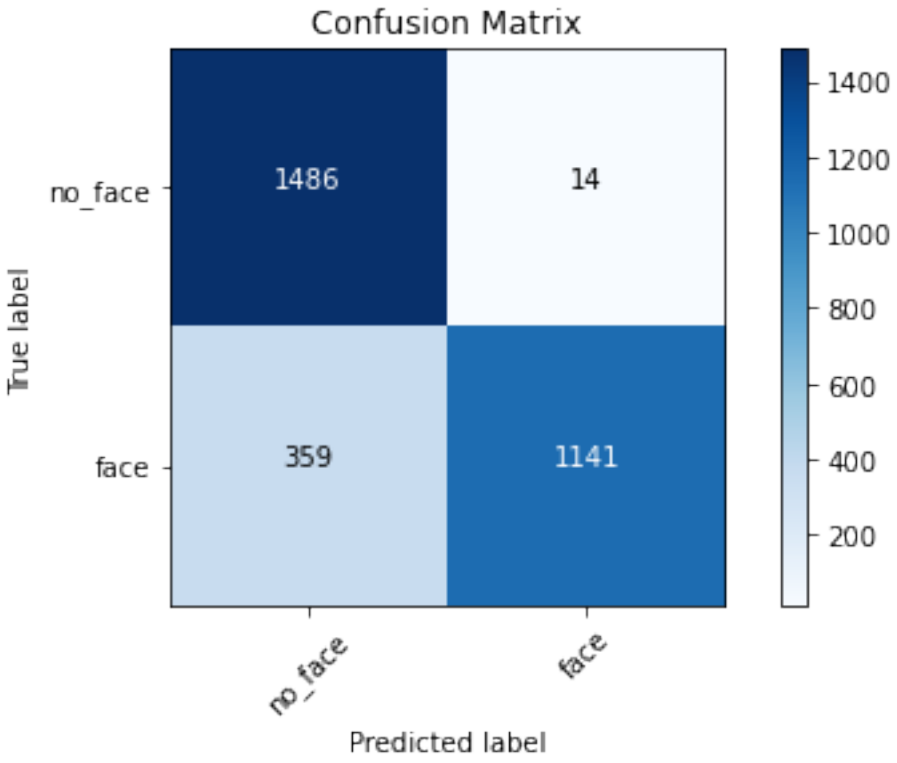
[13]:
```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
```

[14]:
```
test_batches.class_indices
```

[14]:
```
{'no_face': 0, 'face': 1}
```

[15]:
```
cm_plot_labels = ['no_face','face']
plot_confusion_matrix(cm=cm, classes=cm_plot_labels, title='Confusion Matrix')
```

```
Confusion matrix, without normalization
[[1486   14]
 [ 359 1141]]
```


Confusion Matrix
1486
14
359
1141
no_face
face
True label
no_face
face
Predicted label
1400
1200
1000
800
600
400
200