FCN-Baseline-30E-14L-FFA-test-01

March 24, 2021

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

1.1 Testing Baseline FCN *(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/FFA-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-B-30E-14L-01.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: 0.3381 - accuracy: 0.8903
Accuracy: 89.03%
Loss: 33.81%
```

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' [ 124 ] :
[8015, 8018, 8024, 8025, 8028, 8032, 8054, 8064, 8067, 8084, 8086, 8091, 8102,
8123, 8132, 8138, 8143, 8152, 8154, 8160, 8188, 8258, 8278, 8282, 8290, 8298,
8300, 8320, 8322, 8327, 8339, 8341, 8356, 8357, 8365, 8419, 8424, 8435, 8444,
8451, 8466, 8475, 8484, 8486, 8496, 8503, 8520, 8523, 8527, 8533, 8545, 8554,
8555, 8556, 8561, 8571, 8594, 8596, 8598, 8600, 8607, 8619, 8632, 8642, 8657,
8685, 8694, 8719, 8726, 8779, 8816, 8854, 8857, 8871, 8893, 8914, 8919, 8928,
8942, 8948, 8952, 8958, 8962, 8978, 8991, 9000, 9001, 9005, 9026, 9048, 9052,
9061, 9126, 9135, 9152, 9164, 9166, 9183, 9184, 9188, 9208, 9210, 9221, 9222,
9224, 9248, 9258, 9263, 9280, 9288, 9305, 9343, 9356, 9359, 9377, 9378, 9385,
9390, 9426, 9446, 9450, 9475, 9476, 9497]
-------------------------------------------------------------------------------
--------------
Data from class 'no-face', that was wrongly predicted as 'face' [ 205 ] :
[8001, 8006, 8007, 8013, 8015, 8029, 8043, 8045, 8049, 8053, 8060, 8068, 8080,
8083, 8084, 8091, 8094, 8095, 8104, 8114, 8135, 8148, 8151, 8158, 8162, 8164,
8172, 8173, 8179, 8180, 8185, 8192, 8193, 8199, 8200, 8222, 8227, 8244, 8251,
8252, 8256, 8270, 8271, 8281, 8282, 8287, 8288, 8293, 8306, 8317, 8319, 8322,
8323, 8329, 8331, 8334, 8342, 8354, 8355, 8364, 8369, 8380, 8394, 8402, 8415,
8420, 8428, 8429, 8431, 8432, 8441, 8443, 8448, 8451, 8452, 8458, 8466, 8472,
8473, 8487, 8488, 8497, 8509, 8511, 8512, 8524, 8535, 8539, 8541, 8546, 8549,
8550, 8556, 8558, 8559, 8561, 8564, 8580, 8583, 8587, 8602, 8604, 8607, 8611,
8618, 8623, 8643, 8645, 8652, 8657, 8665, 8681, 8683, 8688, 8694, 8695, 8697,
8700, 8709, 8725, 8732, 8743, 8757, 8764, 8774, 8778, 8787, 8795, 8812, 8814,
8820, 8824, 8840, 8842, 8849, 8850, 8851, 8883, 8885, 8894, 8896, 8898, 8910,
8915, 8924, 8934, 8958, 8961, 8979, 8994, 8998, 9003, 9012, 9020, 9057, 9060,
9061, 9064, 9073, 9082, 9092, 9093, 9094, 9097, 9122, 9137, 9146, 9200, 9224,
9234, 9236, 9237, 9241, 9249, 9267, 9272, 9285, 9305, 9320, 9333, 9338, 9342,
9344, 9351, 9360, 9376, 9386, 9388, 9400, 9404, 9408, 9414, 9416, 9419, 9426,
9439, 9445, 9448, 9450, 9458, 9477, 9479, 9480, 9483, 9494]
```

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)
```

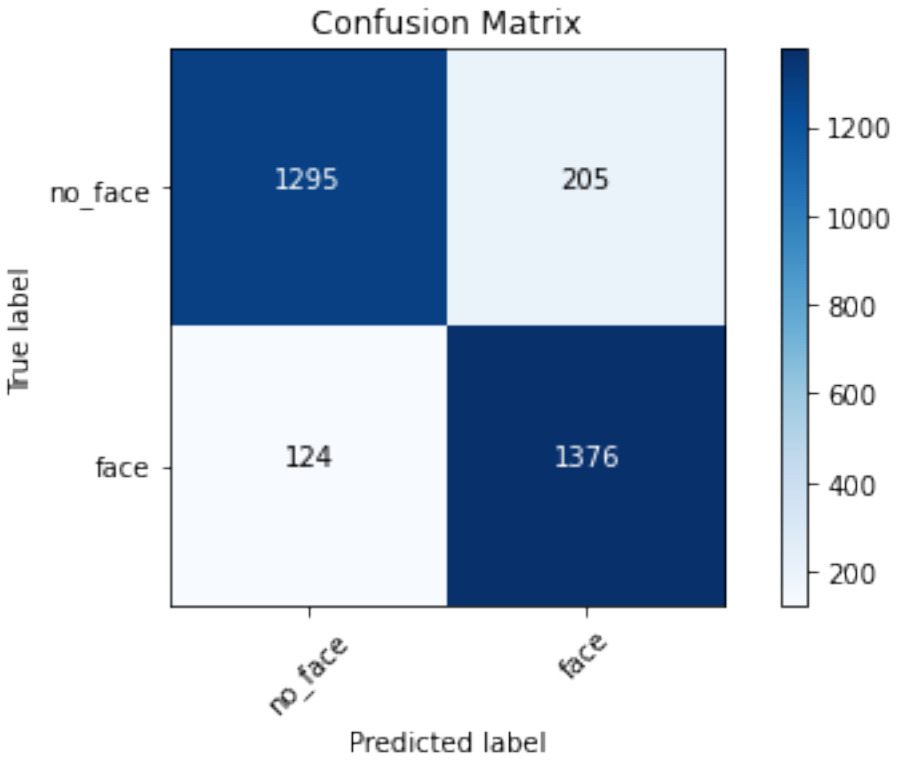
```
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    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
[[1295  205]
 [ 124 1376]]
```


Confusion Matrix
1295
205
124
1376
no_face
face
True label
no_face
face
Predicted label
1200
1000
800
600
400
200