Exam Project | Face Recognizer

By **Pernille Lørup & Stephan Djurhuus**
Institute **CPHBusiness**

Education **Software Development**
Elective **Artificial Intelligence**

Package Imports

```
import cv2
import numpy as np

import tensorflow as tf
from tensorflow.keras import models, backend
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, Dropout
```

Module Imports

```
import sys
sys.path.append('../')
from FaceDetector import FaceDetector
from FaceRecognizer import FaceRecognizer
from DataGenerator import DataGenerator
from Utilities import Displayer
```

Global Constants

```
face_detector = FaceDetector('FACE_DEFAULT_NB')

classes = ['dad', 'mom', 'son', 'daughter']

dad_path = '../data/PersonGroup/Family1-Dad'
mom_path = '../data/PersonGroup/Family1-Mom'
son_path = '../data/PersonGroup/Family1-Son'
daughter_path = '../data/PersonGroup/Family1-Daughter'
```

Data Generation

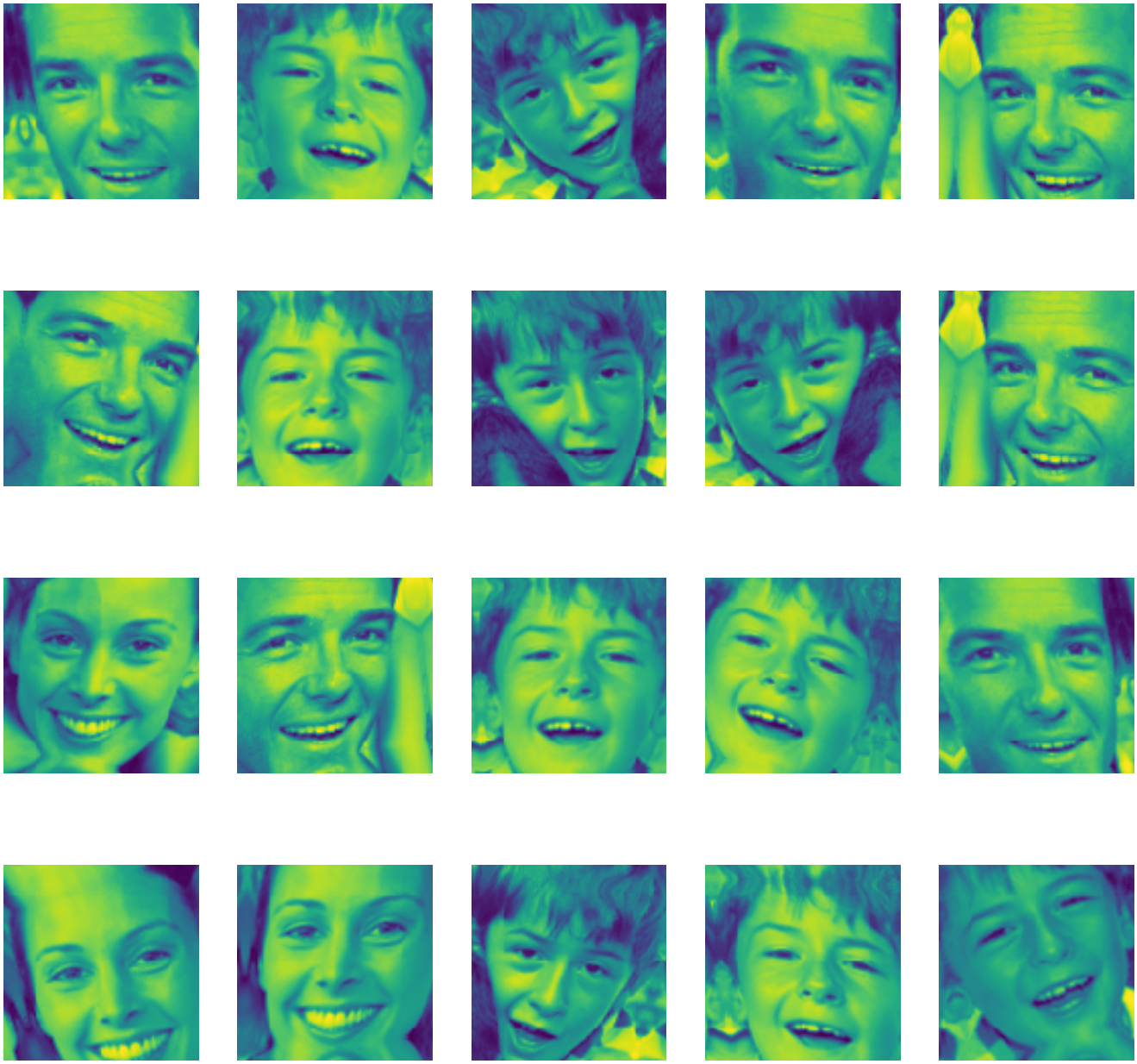
```
(train_dad_data, test_dad_data) = DataGenerator.generate(dad_path, 1000, label=0, test_siz
(train_mom_data, test_mom_data) = DataGenerator.generate(mom_path, 1000, label=1, test_siz
(train_son_data, test_son_data) = DataGenerator.generate(son_path, 1000, label=2, test_siz
(train_daughter_data, test_daughter_data) = DataGenerator.generate(daughter_path, 1000, la

(train_images, train_labels) = DataGenerator.merge_shuffle([train_dad_data, train_mom_data
(test_images, test_labels) = DataGenerator.merge_shuffle([test_dad_data, test_mom_data, te

# greyscalling
train_images_gray = np.array([cv2.cvtColor(image, cv2.COLOR_BGR2GRAY) for image in train_i
test_images_gray = np.array([cv2.cvtColor(image, cv2.COLOR_BGR2GRAY) for image in test_ima
```

Data Validations

```
Displayer.images(train_images[200:], 20)
Displayer.images(train_images_gray[200:], 20)
```

RGB images for CNN

Grayscaled images for ANN



Model Configurations

Convolutional Neural Network

```
cnn_model = models.Sequential([
    Conv2D(25, (2, 2), input_shape=(100, 100, 3)),
    Dropout(.4),
    MaxPooling2D((2, 2)),
    Conv2D(50, (2, 2), activation='relu'),
    MaxPooling2D((2, 2)),
    Conv2D(50, (2, 2), activation='relu'),
    Flatten(),
    Dense(50, activation='relu'),
    Dense(len(classes))
])
```

Artificial Neural Network

Softmax ≈ 50% accuracy

Sigmoid ≈ 50% accuracy

TanH ≈ 50% accuracy

```
def positive_softmax(input_data):
    relu_data = backend.relu(input_data)        # excluding negative values
    softmax_data = backend.softmax(relu_data)   #
    return softmax_data

ann_model = models.Sequential([
    Flatten(input_shape=(100, 100)),
    Dense(500, activation='relu'),
    Dense(250, activation='sigmoid'),
    Dense(len(classes)),
])
```

Face Recognizer Instantiations

```
cnn_recognizer = FaceRecognizer(classes, face_detector, model=cnn_model)
ann_recognizer = FaceRecognizer(classes, face_detector, model=ann_model)
```

Model Training

```
# cnn_recognizer.train_model((train_images, train_labels), (test_images, test_labels), epo
# ann_recognizer.train_model((train_images_gray, train_labels), (test_images_gray, test_la
```

Save & Load CNN Model with History

```
# cnn_recognizer.save_model('../models/cnn_model_2of4')
cnn_recognizer.load_model('../models/cnn_model_3of4')
cnn_recognizer.model_summary()
```

```
Model: "sequential_12"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_26 (Conv2D)           (None, 98, 98, 25)        700
_________________________________________________________________
dropout_9 (Dropout)          (None, 98, 98, 25)        0
_________________________________________________________________
max_pooling2d_17 (MaxPooling (None, 49, 49, 25)        0
_________________________________________________________________
conv2d_27 (Conv2D)           (None, 47, 47, 50)        11300
_________________________________________________________________
max_pooling2d_18 (MaxPooling (None, 23, 23, 50)        0
_________________________________________________________________
conv2d_28 (Conv2D)           (None, 21, 21, 50)        22550
_________________________________________________________________
flatten_12 (Flatten)         (None, 22050)             0
_________________________________________________________________
dense_27 (Dense)             (None, 50)                1102550
_________________________________________________________________
dense_28 (Dense)             (None, 4)                 204
=================================================================
Total params: 1,137,304
Trainable params: 1,137,304
Non-trainable params: 0
_________________________________________________________________
```

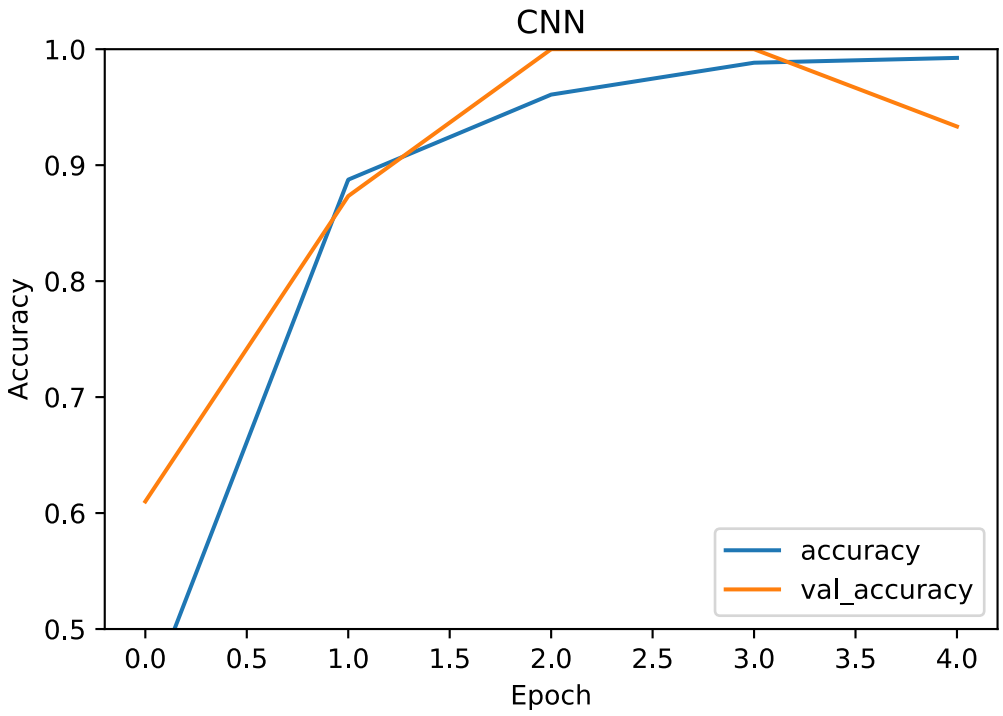
Save & Load ANN Model with History

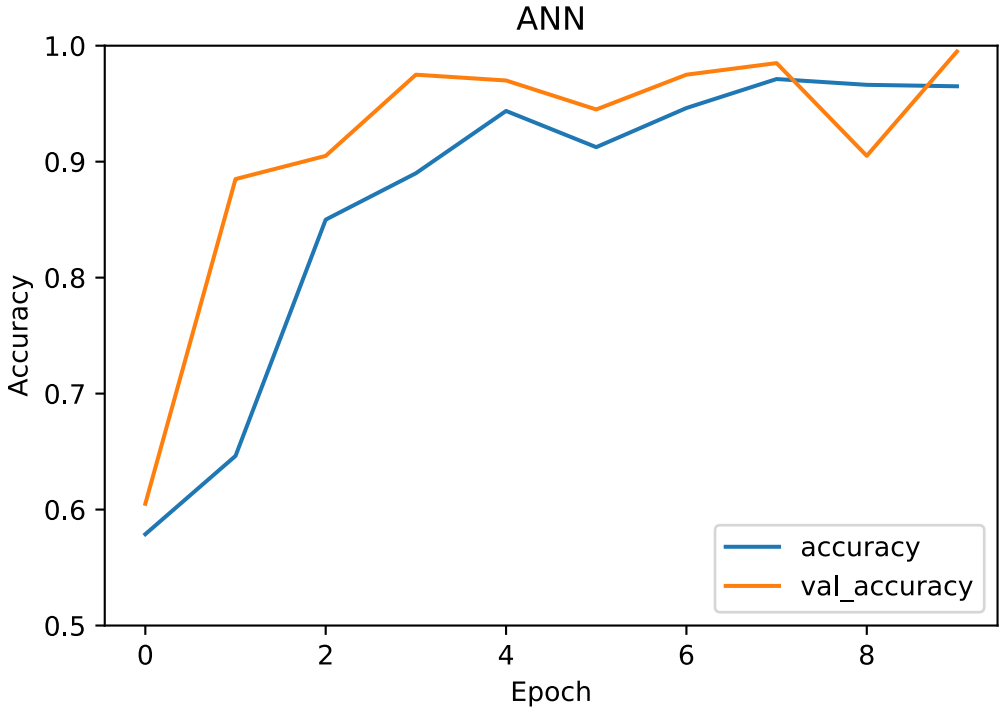
```
# ann_recognizer.save_model('../models/ann_model_2of4')
ann_recognizer.load_model('../models/ann_model_3of4')
ann_recognizer.model_summary()
```

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten_6 (Flatten)          (None, 10000)             0
_________________________________________________________________
dense_17 (Dense)             (None, 500)               5000500
_________________________________________________________________
dense_18 (Dense)             (None, 250)               125250
_________________________________________________________________
dense_19 (Dense)             (None, 4)                 1004
=================================================================
Total params: 5,126,754
Trainable params: 5,126,754
Non-trainable params: 0
_________________________________________________________________
```

Model Train History

```
Displayer.acc_history(cnn_recognizer.history, 'CNN')
Displayer.acc_history(ann_recognizer.history, 'ANN')
```

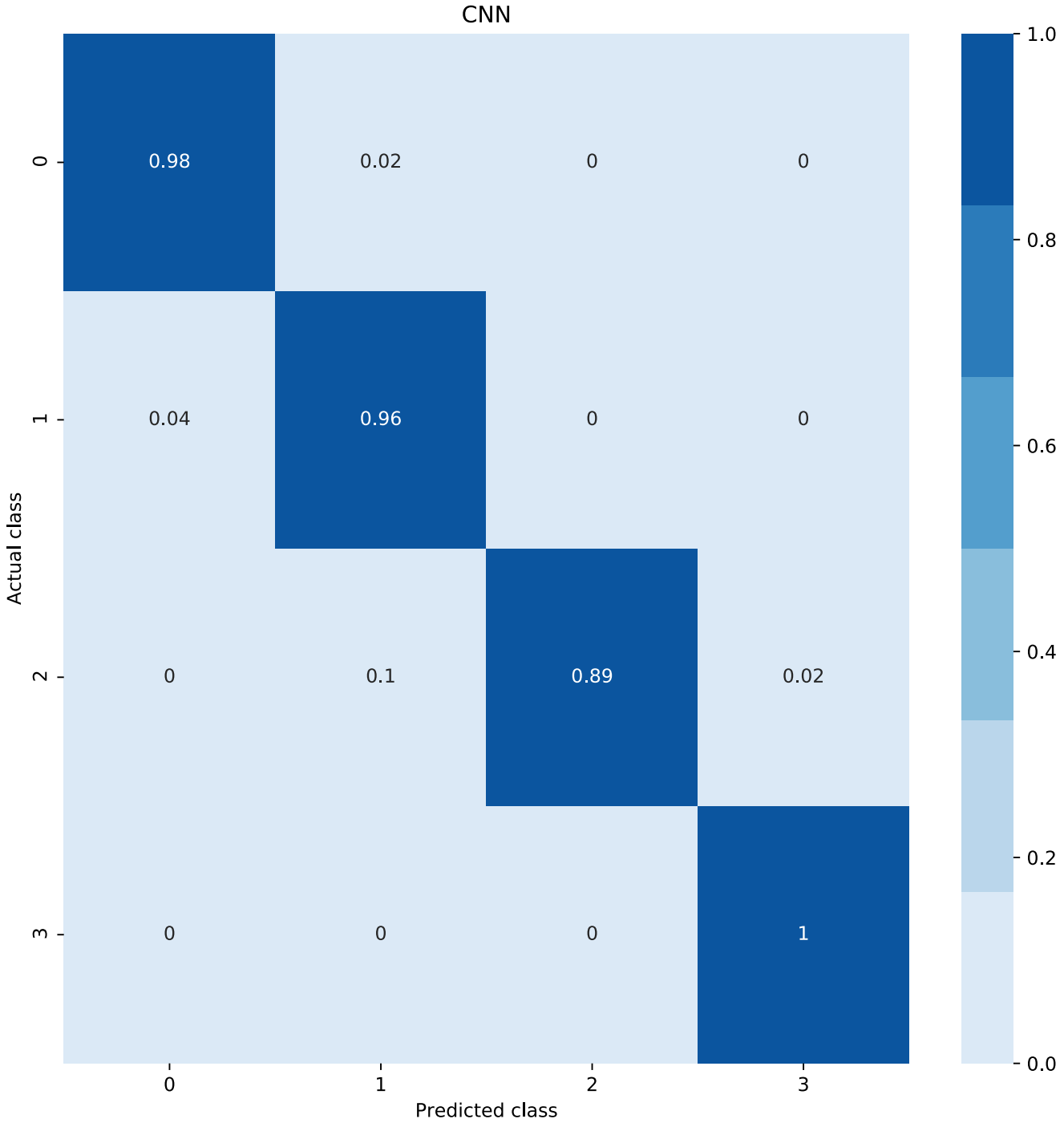




Model Confusion Matrix

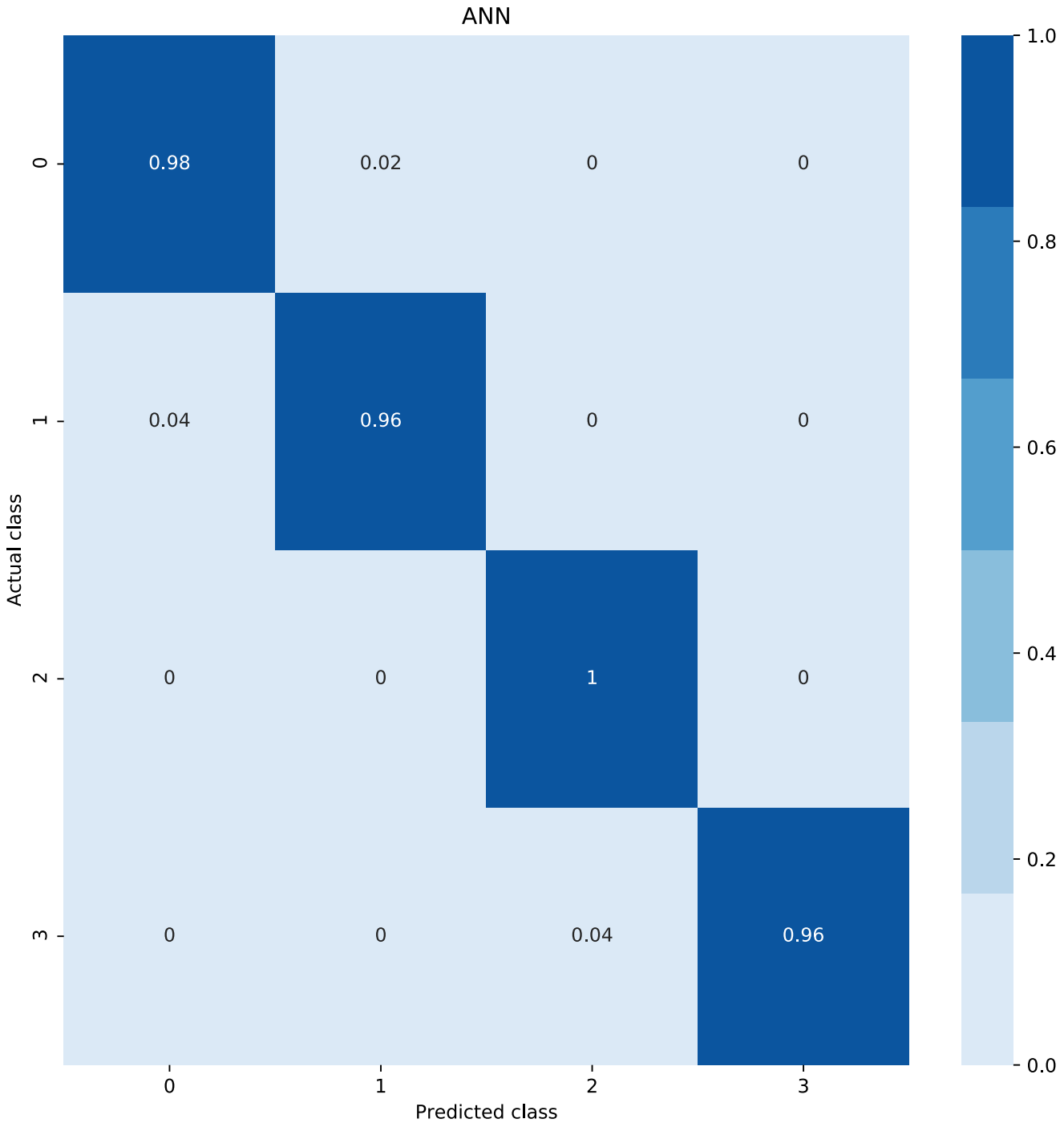
Convolutional Neural Network

```
predictions = cnn_recognizer.recognize_many(test_images)
Displayer.conf_matrix(predictions, test_labels, 'CNN')
```



Artificial Neural Network

```
predictions = ann_recognizer.recognize_many(test_images_gray)
Displayer.conf_matrix(predictions, test_labels, 'ANN')
```

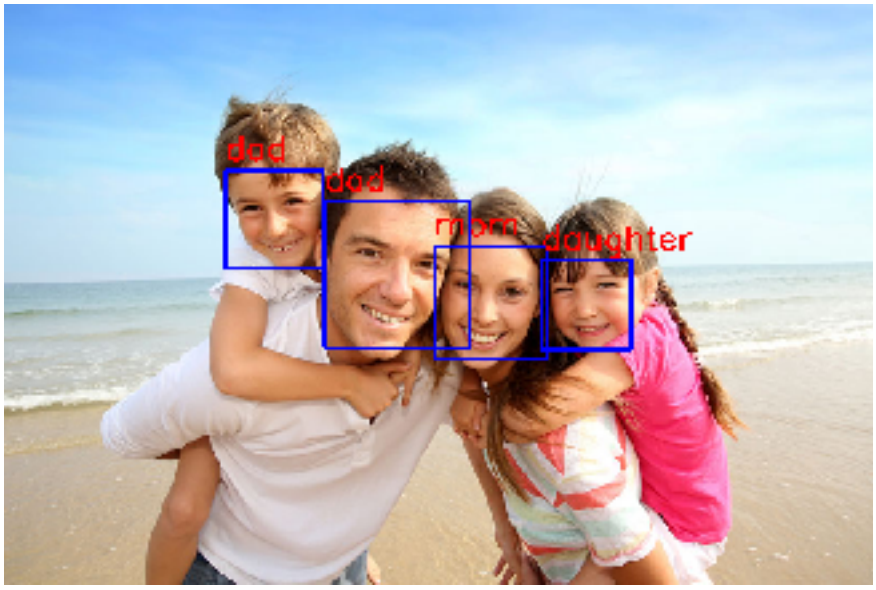


Model Validations

```
test_image_path = '../data/identification1.jpg'
test_image = cv2.imread(test_image_path)
test_image_gray = cv2.cvtColor(test_image, cv2.COLOR_BGR2GRAY)

faceDetector = FaceDetector('FACE_ALT2_NB')
faces_data = faceDetector.face_details(test_image)
```

Convolutional Neural Network

```
cnn_prediction_data = cnn_recognizer.face_predictions(faces_data)
Displayer.mark_predictions(test_image.copy(), cnn_prediction_data, classes)
```



Artificial Neural Network

```
ann_prediction_data = ann_recognizer.face_predictions(faces_data, gray=True)
Displayer.mark_predictions(test_image.copy(), ann_prediction_data, classes)
```