Teste Terraria e Mine

June 26, 2022

[2]:
```python
import numpy as np
import os
from sklearn.metrics import confusion_matrix
import seaborn as sn; sn.set(font_scale=1.4)
from sklearn.utils import shuffle
import matplotlib.pyplot as plt
import cv2
import tensorflow as tf
from tqdm import tqdm

class_names = ['Terraria','Mine']
class_names_label = {class_name:i for i, class_name in enumerate(class_names)}
nb_classes = len(class_names)
IMAGE_SIZE = (150, 150)
```

[3]:
```python
def load_data():

    datasets = [r"C:\Users\neure\Downloads\I.A\Train",
                r"C:\Users\neure\Downloads\I.A\Validation"]
    output = []


    for dataset in datasets:
        images = []
        labels = []

        print("Loading {}".format(dataset))


        for folder in os.listdir(dataset):
            label = class_names_label[folder]


            for file in tqdm(os.listdir(os.path.join(dataset, folder))):


                img_path = os.path.join(os.path.join(dataset, folder), file)
```

```
                image = cv2.imread(img_path)
                image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
                image = cv2.resize(image, IMAGE_SIZE)


                images.append(image)
                labels.append(label)

        images = np.array(images, dtype = 'float32')
        labels = np.array(labels, dtype = 'int32')

        output.append((images, labels))

    return output
```

[4]:
```
(train_images, train_labels), (test_images, test_labels) = load_data()
```

```
Loading C:\Users\neure\Downloads\I.A\Train

100%|
   | 473/473 [00:07<00:00, 66.10it/s]
100%|
   | 500/500 [00:08<00:00, 61.87it/s]

Loading C:\Users\neure\Downloads\I.A\Validation

100%|
   | 456/456 [00:06<00:00, 65.58it/s]
100%|
   | 500/500 [00:08<00:00, 56.12it/s]
```

[5]:
```
train_images, train_labels = shuffle(train_images, train_labels,␣
 ↪random_state=25)
train_images.shape, train_labels.shape
```

[5]:
```
((973, 150, 150, 3), (973,))
```

[6]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(train_images,␣
 ↪train_labels,test_size=0.1)

x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

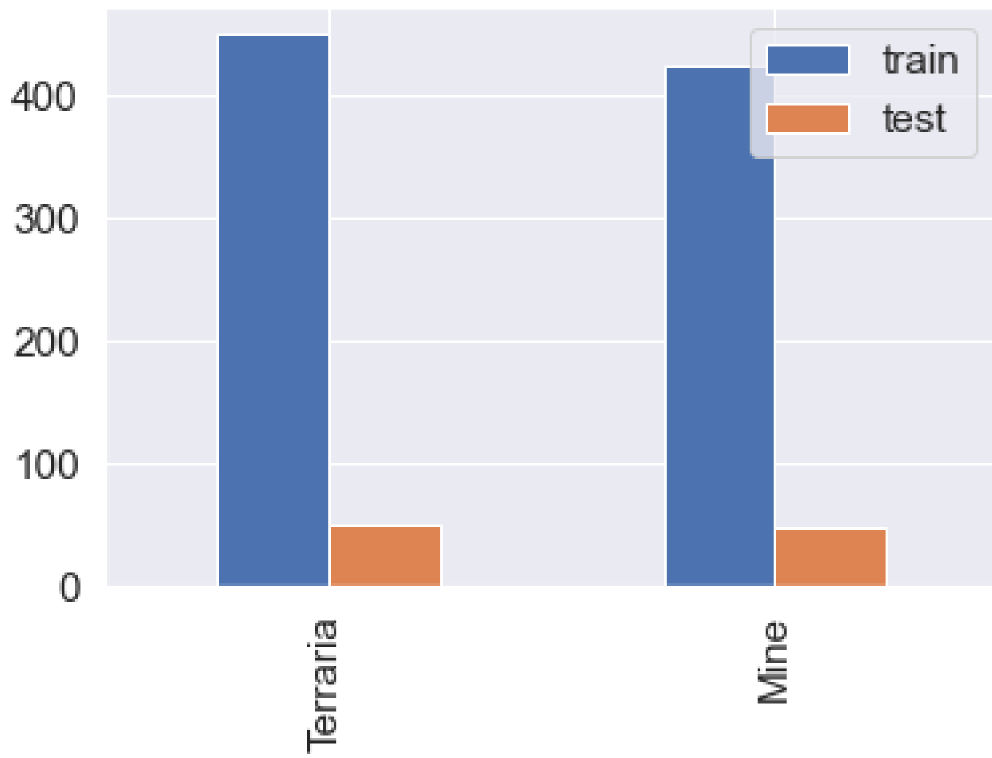
[6]:
```
((875, 150, 150, 3), (98, 150, 150, 3), (875,), (98,))
```

[11]:
```
n_train = x_train.shape[0]
n_test = x_test.shape[0]
```

```
print ("Number of training examples: {}".format(n_train))
print ("Number of testing examples: {}".format(n_test))
print ("Each image is of size: {}".format(IMAGE_SIZE))
```

```
Number of training examples: 875
Number of testing examples: 98
Each image is of size: (150, 150)
```

[12]:
```
import pandas as pd

_, train_counts = np.unique(y_train, return_counts=True)
_, test_counts = np.unique(y_test, return_counts=True)
pd.DataFrame({'train': train_counts,
                    'test': test_counts},
             index=class_names
            ).plot.bar()
plt.show()
```



[14]:
```
'''

x_train = train_images / 255.0
```

```
x_test_images = test_images / 255.0
x_train.shape,x_test_images.shape
'''
x_train = x_train / 255.0
x_test = x_test / 255.0
x_train.shape,x_test.shape
```

[14]: ((875, 150, 150, 3), (98, 150, 150, 3))

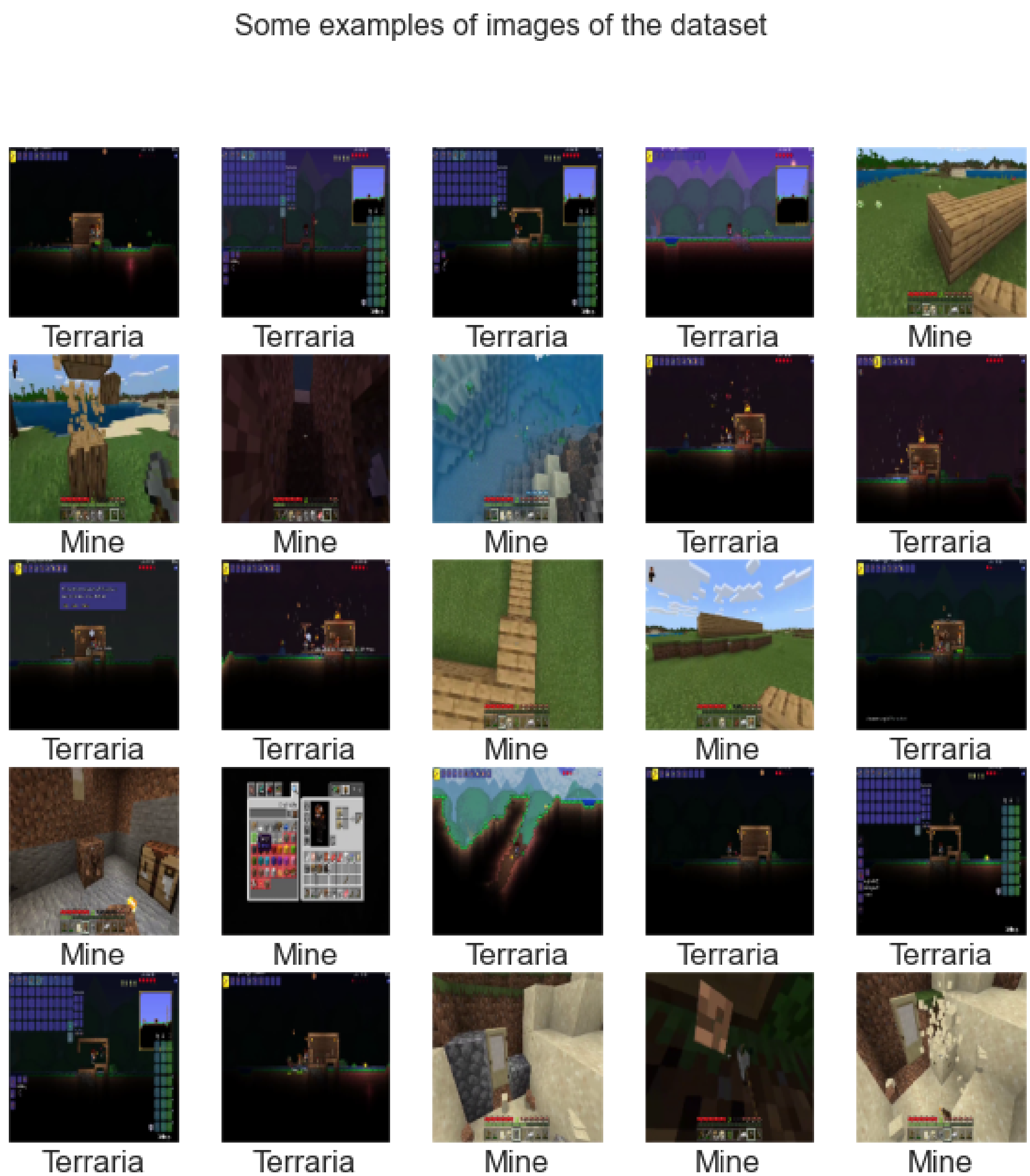
[15]:
```
def display_random_image(class_names, images, labels):
    """
        Display a random image from the images array and its correspond label␣
    ↪from the labels array.
    """

    index = np.random.randint(images.shape[0])
    plt.figure()
    plt.imshow(images[index])
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.title('Image #{} : '.format(index) + class_names[labels[index]])
    plt.show()
```

[16]:
```
display_random_image(class_names, x_train, y_train)
```

Image #600 : Terraria

[17]:
```
def display_examples(class_names, images, labels):

    fig = plt.figure(figsize=(10,10))
    fig.suptitle("Some examples of images of the dataset", fontsize=16)
    for i in range(25):
        plt.subplot(5,5,i+1)
        plt.xticks([])
        plt.yticks([])
        plt.grid(False)
        plt.imshow(images[i], cmap=plt.cm.binary)
        plt.xlabel(class_names[labels[i]])
    plt.show()
```

[18]:
```
display_examples(class_names, x_train, y_train)
```



[19]:
```
import tensorflow as tf
import keras

from keras.models import Sequential,load_model
from keras.layers import Dense, Conv2D ,LSTM, MaxPooling2D , Flatten , Dropout
from keras.layers import BatchNormalization,TimeDistributed

from keras.optimizers import Adam
```

```python
from keras.callbacks import ModelCheckpoint, EarlyStopping,ReduceLROnPlateau

import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = (15,5)
import seaborn as sns
import pandas as pd
import os
import random
import time
import os

import numpy as np
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

from scipy import signal
from scipy.fft import fftshift
```

[20]:

```python
def cnn_model():

    model = Sequential()

    model.add(Conv2D(filters = 64, kernel_size = (8,8),
                     padding = "same", activation = "relu",
                     input_shape=(150,150,3)))
    model.add(MaxPooling2D(pool_size = (4,4)))
    model.add(Dropout(0.5))
    model.add(BatchNormalization())

    model.add(Conv2D(filters = 64, kernel_size = (4,4),
                     padding = "same", activation = "relu",
                     input_shape=(150,150,3)))
    model.add(MaxPooling2D(pool_size = (2,2)))
    model.add(Dropout(0.5))
    model.add(BatchNormalization())

    model.add(Conv2D(filters = 64, kernel_size = (2,2),
                     padding = "same", activation = "relu",
                     input_shape=(150,150,3)))

    model.add(MaxPooling2D(pool_size = (1,1)))
    model.add(Dropout(0.5))
    model.add(BatchNormalization())


    model.add(Flatten())
```

```python
    model.add(Dense(64, activation = "relu"))
    model.add(Dense(8, activation = "softmax"))

    model.compile(optimizer ="adam", loss = "sparse_categorical_crossentropy",
                  metrics = ['accuracy'])

    return model

model = cnn_model()
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 150, 150, 64)      12352

 max_pooling2d (MaxPooling2D  (None, 37, 37, 64)       0
 )

 dropout (Dropout)           (None, 37, 37, 64)        0

 batch_normalization (BatchN  (None, 37, 37, 64)       256
 ormalization)

 conv2d_1 (Conv2D)           (None, 37, 37, 64)        65600

 max_pooling2d_1 (MaxPooling  (None, 18, 18, 64)       0
 2D)

 dropout_1 (Dropout)         (None, 18, 18, 64)        0

 batch_normalization_1 (Batc  (None, 18, 18, 64)       256
 hNormalization)

 conv2d_2 (Conv2D)           (None, 18, 18, 64)        16448

 max_pooling2d_2 (MaxPooling  (None, 18, 18, 64)       0
 2D)

 dropout_2 (Dropout)         (None, 18, 18, 64)        0

 batch_normalization_2 (Batc  (None, 18, 18, 64)       256
 hNormalization)

 flatten (Flatten)           (None, 20736)             0
```

```
 dense (Dense)               (None, 64)               1327168


 dense_1 (Dense)             (None, 8)                520

=================================================================
Total params: 1,422,856
Trainable params: 1,422,472
Non-trainable params: 384
_________________________________________________________________
```

[21]:

```
pat = 5
n_folds=5
epochs=50
batch_size=64

learning_rate_reduction = ReduceLROnPlateau(monitor='val_accuracy', patience =␣
↪5,
                                            verbose=1,factor=0.5, min_lr=0.0001)
early_stopping = EarlyStopping(monitor='loss', patience=pat, verbose=1)
model_checkpoint = ModelCheckpoint('subjek1CNN.h5',␣
↪verbose=1,save_best_only=True)

def fit_and_evaluate(t_x, val_x, t_y, val_y, EPOCHS=epochs,␣
↪BATCH_SIZE=batch_size):
    model = None
    model = cnn_model()
    results = model.fit(t_x, t_y, epochs=EPOCHS, batch_size=BATCH_SIZE,
                        callbacks=[
                                   learning_rate_reduction,
                                   early_stopping,
                                   model_checkpoint], verbose=1,␣
↪validation_split=0.1)

    print("Val Score: ", model.evaluate(val_x, val_y))
    '''
    predictions = model.predict_classes(val_x, batch_size=32, verbose=1)
    print(classification_report(val_y, predictions ))

    predictions = model.predict_classes(t_x, batch_size=32, verbose=1)
    cm = confusion_matrix(t_y,predictions)

    plt.figure(figsize = (5,5))
    sns.heatmap(cm,cmap= "Blues", linecolor = 'black' ,
                linewidth = 1 , annot = True, fmt='')
    '''
    return results
```

```python
model_history = []
for i in range(n_folds):
    print("Training on Fold: ",i+1)
    t_x, val_x, t_y, val_y = train_test_split(test_images, test_labels,
                                               test_size=0.1)
    model_history.append(fit_and_evaluate(t_x, val_x, t_y, val_y, epochs,␣
 ↪batch_size))

    print("======="*12, end="\n\n\n\n")
```

```
Training on Fold:  1
Epoch 1/50
13/13 [==============================] - ETA: 0s - loss: 0.3796 - accuracy:
0.8876
Epoch 1: val_loss improved from inf to 15.83479, saving model to subjek1CNN.h5
13/13 [==============================] - 40s 3s/step - loss: 0.3796 - accuracy:
0.8876 - val_loss: 15.8348 - val_accuracy: 0.5814 - lr: 0.0010
Epoch 2/50
13/13 [==============================] - ETA: 0s - loss: 0.0219 - accuracy:
0.9974
Epoch 2: val_loss did not improve from 15.83479
13/13 [==============================] - 35s 3s/step - loss: 0.0219 - accuracy:
0.9974 - val_loss: 19.7588 - val_accuracy: 0.5698 - lr: 0.0010
Epoch 3/50
13/13 [==============================] - ETA: 0s - loss: 0.0091 - accuracy:
0.9974
Epoch 3: val_loss improved from 15.83479 to 9.33517, saving model to
subjek1CNN.h5
13/13 [==============================] - 37s 3s/step - loss: 0.0091 - accuracy:
0.9974 - val_loss: 9.3352 - val_accuracy: 0.6512 - lr: 0.0010
Epoch 4/50
13/13 [==============================] - ETA: 0s - loss: 0.0014 - accuracy:
0.9987
Epoch 4: val_loss improved from 9.33517 to 2.84958, saving model to
subjek1CNN.h5
13/13 [==============================] - 36s 3s/step - loss: 0.0014 - accuracy:
0.9987 - val_loss: 2.8496 - val_accuracy: 0.8023 - lr: 0.0010
Epoch 5/50
13/13 [==============================] - ETA: 0s - loss: 0.0011 - accuracy:
1.0000
Epoch 5: val_loss improved from 2.84958 to 0.26965, saving model to
subjek1CNN.h5
13/13 [==============================] - 38s 3s/step - loss: 0.0011 - accuracy:
1.0000 - val_loss: 0.2696 - val_accuracy: 0.9767 - lr: 0.0010
Epoch 6/50
13/13 [==============================] - ETA: 0s - loss: 0.0026 - accuracy:
```

```
0.9987
Epoch 6: val_loss did not improve from 0.26965
13/13 [==============================] - 41s 3s/step - loss: 0.0026 - accuracy:
0.9987 - val_loss: 0.5505 - val_accuracy: 0.9419 - lr: 0.0010
Epoch 7/50
13/13 [==============================] - ETA: 0s - loss: 7.0396e-04 - accuracy:
1.0000
Epoch 7: val_loss improved from 0.26965 to 0.07680, saving model to
subjek1CNN.h5
13/13 [==============================] - 40s 3s/step - loss: 7.0396e-04 -
accuracy: 1.0000 - val_loss: 0.0768 - val_accuracy: 0.9767 - lr: 0.0010
Epoch 8/50
13/13 [==============================] - ETA: 0s - loss: 0.0152 - accuracy:
0.9987
Epoch 8: val_loss did not improve from 0.07680
13/13 [==============================] - 37s 3s/step - loss: 0.0152 - accuracy:
0.9987 - val_loss: 4.0489 - val_accuracy: 0.7442 - lr: 0.0010
Epoch 9/50
13/13 [==============================] - ETA: 0s - loss: 0.0076 - accuracy:
0.9974
Epoch 9: val_loss improved from 0.07680 to 0.02754, saving model to
subjek1CNN.h5
13/13 [==============================] - 38s 3s/step - loss: 0.0076 - accuracy:
0.9974 - val_loss: 0.0275 - val_accuracy: 0.9884 - lr: 0.0010
Epoch 10/50
13/13 [==============================] - ETA: 0s - loss: 0.0118 - accuracy:
0.9987
Epoch 10: val_loss improved from 0.02754 to 0.01672, saving model to
subjek1CNN.h5
13/13 [==============================] - 38s 3s/step - loss: 0.0118 - accuracy:
0.9987 - val_loss: 0.0167 - val_accuracy: 0.9767 - lr: 0.0010
Epoch 11/50
13/13 [==============================] - ETA: 0s - loss: 0.0126 - accuracy:
0.9948
Epoch 11: val_loss improved from 0.01672 to 0.00000, saving model to
subjek1CNN.h5
13/13 [==============================] - 37s 3s/step - loss: 0.0126 - accuracy:
0.9948 - val_loss: 2.3564e-07 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 12/50
13/13 [==============================] - ETA: 0s - loss: 0.0022 - accuracy:
0.9987
Epoch 12: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0022 - accuracy:
0.9987 - val_loss: 0.2285 - val_accuracy: 0.9651 - lr: 0.0010
Epoch 12: early stopping
3/3 [==============================] - 1s 419ms/step - loss: 0.2601 - accuracy:
0.9792
Val Score:  [0.2601049542427063, 0.9791666865348816]
```

```
================================================================================
====


Training on Fold:  2
Epoch 1/50
13/13 [==============================] - ETA: 0s - loss: 0.2840 - accuracy:
0.9302
Epoch 1: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 0.2840 - accuracy:
0.9302 - val_loss: 5.9392 - val_accuracy: 0.8256 - lr: 0.0010
Epoch 2/50
13/13 [==============================] - ETA: 0s - loss: 0.0228 - accuracy:
0.9922
Epoch 2: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0228 - accuracy:
0.9922 - val_loss: 6.3823 - val_accuracy: 0.8605 - lr: 0.0010
Epoch 3/50
13/13 [==============================] - ETA: 0s - loss: 0.0377 - accuracy:
0.9922
Epoch 3: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0377 - accuracy:
0.9922 - val_loss: 6.0103 - val_accuracy: 0.8256 - lr: 0.0010
Epoch 4/50
13/13 [==============================] - ETA: 0s - loss: 0.0058 - accuracy:
0.9961
Epoch 4: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0058 - accuracy:
0.9961 - val_loss: 2.0569 - val_accuracy: 0.9070 - lr: 0.0010
Epoch 5/50
13/13 [==============================] - ETA: 0s - loss: 0.0032 - accuracy:
0.9987
Epoch 5: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0032 - accuracy:
0.9987 - val_loss: 2.0878 - val_accuracy: 0.9070 - lr: 0.0010
Epoch 6/50
13/13 [==============================] - ETA: 0s - loss: 4.9537e-04 - accuracy:
1.0000
Epoch 6: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 4.9537e-04 -
accuracy: 1.0000 - val_loss: 0.0549 - val_accuracy: 0.9767 - lr: 0.0010
Epoch 7/50
13/13 [==============================] - ETA: 0s - loss: 6.9614e-04 - accuracy:
1.0000
Epoch 7: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 6.9614e-04 -
accuracy: 1.0000 - val_loss: 3.3573e-05 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 8/50
```

```
13/13 [==============================] - ETA: 0s - loss: 9.5092e-05 - accuracy:
1.0000
Epoch 8: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 9.5092e-05 -
accuracy: 1.0000 - val_loss: 6.1480e-06 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 9/50
13/13 [==============================] - ETA: 0s - loss: 0.0035 - accuracy:
0.9974
Epoch 9: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0035 - accuracy:
0.9974 - val_loss: 7.9194e-04 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 10/50
13/13 [==============================] - ETA: 0s - loss: 0.0150 - accuracy:
0.9961
Epoch 10: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0150 - accuracy:
0.9961 - val_loss: 0.0065 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 11/50
13/13 [==============================] - ETA: 0s - loss: 0.0083 - accuracy:
0.9987
Epoch 11: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0083 - accuracy:
0.9987 - val_loss: 0.2379 - val_accuracy: 0.9302 - lr: 0.0010
Epoch 12/50
13/13 [==============================] - ETA: 0s - loss: 0.0056 - accuracy:
0.9974
Epoch 12: ReduceLROnPlateau reducing learning rate to 0.0005000000237487257.

Epoch 12: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0056 - accuracy:
0.9974 - val_loss: 0.0012 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 13/50
13/13 [==============================] - ETA: 0s - loss: 4.1172e-04 - accuracy:
1.0000
Epoch 13: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 4.1172e-04 -
accuracy: 1.0000 - val_loss: 9.7011e-05 - val_accuracy: 1.0000 - lr: 5.0000e-04
Epoch 13: early stopping
3/3 [==============================] - 1s 433ms/step - loss: 0.0017 - accuracy:
1.0000
Val Score:  [0.0017018966609612107, 1.0]
================================================================================
====


Training on Fold:  3
Epoch 1/50
13/13 [==============================] - ETA: 0s - loss: 0.3800 - accuracy:
```

```
0.8953
Epoch 1: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 0.3800 - accuracy:
0.8953 - val_loss: 4.2356 - val_accuracy: 0.8140 - lr: 0.0010
Epoch 2/50
13/13 [==============================] - ETA: 0s - loss: 0.0420 - accuracy:
0.9922
Epoch 2: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0420 - accuracy:
0.9922 - val_loss: 7.2755 - val_accuracy: 0.8023 - lr: 0.0010
Epoch 3/50
13/13 [==============================] - ETA: 0s - loss: 0.0261 - accuracy:
0.9922
Epoch 3: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0261 - accuracy:
0.9922 - val_loss: 3.7072 - val_accuracy: 0.8837 - lr: 0.0010
Epoch 4/50
13/13 [==============================] - ETA: 0s - loss: 0.0039 - accuracy:
0.9974
Epoch 4: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0039 - accuracy:
0.9974 - val_loss: 1.6103 - val_accuracy: 0.8837 - lr: 0.0010
Epoch 5/50
13/13 [==============================] - ETA: 0s - loss: 0.0045 - accuracy:
0.9987
Epoch 5: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0045 - accuracy:
0.9987 - val_loss: 0.1066 - val_accuracy: 0.9419 - lr: 0.0010
Epoch 6/50
13/13 [==============================] - ETA: 0s - loss: 0.0031 - accuracy:
0.9987
Epoch 6: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 0.0031 - accuracy:
0.9987 - val_loss: 0.1191 - val_accuracy: 0.9767 - lr: 0.0010
Epoch 7/50
13/13 [==============================] - ETA: 0s - loss: 6.5973e-05 - accuracy:
1.0000
Epoch 7: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 6.5973e-05 -
accuracy: 1.0000 - val_loss: 0.0961 - val_accuracy: 0.9884 - lr: 0.0010
Epoch 8/50
13/13 [==============================] - ETA: 0s - loss: 0.0022 - accuracy:
0.9987
Epoch 8: val_loss did not improve from 0.00000
13/13 [==============================] - 41s 3s/step - loss: 0.0022 - accuracy:
0.9987 - val_loss: 0.0080 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 9/50
13/13 [==============================] - ETA: 0s - loss: 2.9391e-04 - accuracy:
```

```
1.0000
Epoch 9: val_loss did not improve from 0.00000
13/13 [==============================] - 41s 3s/step - loss: 2.9391e-04 -
accuracy: 1.0000 - val_loss: 0.0028 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 10/50
13/13 [==============================] - ETA: 0s - loss: 0.0012 - accuracy:
0.9987
Epoch 10: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0012 - accuracy:
0.9987 - val_loss: 0.0150 - val_accuracy: 0.9884 - lr: 0.0010
Epoch 11/50
13/13 [==============================] - ETA: 0s - loss: 0.0023 - accuracy:
0.9987
Epoch 11: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0023 - accuracy:
0.9987 - val_loss: 0.0523 - val_accuracy: 0.9884 - lr: 0.0010
Epoch 12/50
13/13 [==============================] - ETA: 0s - loss: 1.8465e-04 - accuracy:
1.0000
Epoch 12: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 1.8465e-04 -
accuracy: 1.0000 - val_loss: 0.0059 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 12: early stopping
3/3 [==============================] - 1s 434ms/step - loss: 0.0069 - accuracy:
1.0000
Val Score:  [0.00688589783385396, 1.0]
================================================================================
====


Training on Fold:  4
Epoch 1/50
13/13 [==============================] - ETA: 0s - loss: 0.3737 - accuracy:
0.8915
Epoch 1: val_loss did not improve from 0.00000
13/13 [==============================] - 40s 3s/step - loss: 0.3737 - accuracy:
0.8915 - val_loss: 2.5598 - val_accuracy: 0.8605 - lr: 0.0010
Epoch 2/50
13/13 [==============================] - ETA: 0s - loss: 0.0197 - accuracy:
0.9961
Epoch 2: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 0.0197 - accuracy:
0.9961 - val_loss: 1.5610 - val_accuracy: 0.8837 - lr: 0.0010
Epoch 3/50
13/13 [==============================] - ETA: 0s - loss: 0.0021 - accuracy:
1.0000
Epoch 3: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 0.0021 - accuracy:
```

```
1.0000 - val_loss: 0.3511 - val_accuracy: 0.9419 - lr: 0.0010
Epoch 4/50
13/13 [==============================] - ETA: 0s - loss: 0.0064 - accuracy:
0.9987
Epoch 4: val_loss did not improve from 0.00000
13/13 [==============================] - 40s 3s/step - loss: 0.0064 - accuracy:
0.9987 - val_loss: 0.5522 - val_accuracy: 0.9535 - lr: 0.0010
Epoch 5/50
13/13 [==============================] - ETA: 0s - loss: 0.0036 - accuracy:
0.9974
Epoch 5: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0036 - accuracy:
0.9974 - val_loss: 2.5776e-04 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 6/50
13/13 [==============================] - ETA: 0s - loss: 0.0037 - accuracy:
0.9987
Epoch 6: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0037 - accuracy:
0.9987 - val_loss: 0.6012 - val_accuracy: 0.9186 - lr: 0.0010
Epoch 7/50
13/13 [==============================] - ETA: 0s - loss: 0.0078 - accuracy:
0.9987
Epoch 7: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0078 - accuracy:
0.9987 - val_loss: 0.7188 - val_accuracy: 0.9186 - lr: 0.0010
Epoch 8/50
13/13 [==============================] - ETA: 0s - loss: 0.0024 - accuracy:
0.9987
Epoch 8: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 0.0024 - accuracy:
0.9987 - val_loss: 0.7746 - val_accuracy: 0.9186 - lr: 0.0010
Epoch 8: early stopping
3/3 [==============================] - 1s 466ms/step - loss: 0.4935 - accuracy:
0.8958
Val Score:  [0.4934908151626587, 0.8958333134651184]
================================================================================
====


Training on Fold:  5
Epoch 1/50
13/13 [==============================] - ETA: 0s - loss: 0.4021 - accuracy:
0.8863
Epoch 1: val_loss did not improve from 0.00000
13/13 [==============================] - 41s 3s/step - loss: 0.4021 - accuracy:
0.8863 - val_loss: 5.7818 - val_accuracy: 0.8488 - lr: 0.0010
Epoch 2/50
13/13 [==============================] - ETA: 0s - loss: 0.0275 - accuracy:
```

```
0.9922
Epoch 2: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 0.0275 - accuracy:
0.9922 - val_loss: 1.7459 - val_accuracy: 0.9070 - lr: 0.0010
Epoch 3/50
13/13 [==============================] - ETA: 0s - loss: 0.0439 - accuracy:
0.9948
Epoch 3: val_loss did not improve from 0.00000
13/13 [==============================] - 39s 3s/step - loss: 0.0439 - accuracy:
0.9948 - val_loss: 1.9707 - val_accuracy: 0.8953 - lr: 0.0010
Epoch 4/50
13/13 [==============================] - ETA: 0s - loss: 0.0121 - accuracy:
0.9948
Epoch 4: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0121 - accuracy:
0.9948 - val_loss: 0.1060 - val_accuracy: 0.9884 - lr: 0.0010
Epoch 5/50
13/13 [==============================] - ETA: 0s - loss: 0.0122 - accuracy:
0.9961
Epoch 5: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0122 - accuracy:
0.9961 - val_loss: 0.2983 - val_accuracy: 0.9767 - lr: 0.0010
Epoch 6/50
13/13 [==============================] - ETA: 0s - loss: 0.0072 - accuracy:
0.9987
Epoch 6: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0072 - accuracy:
0.9987 - val_loss: 0.2546 - val_accuracy: 0.9767 - lr: 0.0010
Epoch 7/50
13/13 [==============================] - ETA: 0s - loss: 1.2061e-04 - accuracy:
1.0000
Epoch 7: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 1.2061e-04 -
accuracy: 1.0000 - val_loss: 0.0041 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 8/50
13/13 [==============================] - ETA: 0s - loss: 0.0087 - accuracy:
0.9974
Epoch 8: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0087 - accuracy:
0.9974 - val_loss: 0.0119 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 9/50
13/13 [==============================] - ETA: 0s - loss: 0.0773 - accuracy:
0.9897
Epoch 9: val_loss did not improve from 0.00000
13/13 [==============================] - 37s 3s/step - loss: 0.0773 - accuracy:
0.9897 - val_loss: 0.0419 - val_accuracy: 0.9767 - lr: 0.0010
Epoch 10/50
13/13 [==============================] - ETA: 0s - loss: 0.0011 - accuracy:
```

```
1.0000
Epoch 10: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0011 - accuracy:
1.0000 - val_loss: 0.0020 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 11/50
13/13 [==============================] - ETA: 0s - loss: 0.0159 - accuracy:
0.9974
Epoch 11: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0159 - accuracy:
0.9974 - val_loss: 7.6820e-05 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 12/50
13/13 [==============================] - ETA: 0s - loss: 0.0019 - accuracy:
0.9987
Epoch 12: ReduceLROnPlateau reducing learning rate to 0.0005000000237487257.

Epoch 12: val_loss did not improve from 0.00000
13/13 [==============================] - 38s 3s/step - loss: 0.0019 - accuracy:
0.9987 - val_loss: 3.7633e-04 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 12: early stopping
3/3 [==============================] - 1s 415ms/step - loss: 7.2923e-04 -
accuracy: 1.0000
Val Score:  [0.0007292316877283156, 1.0]
================================================================================
====
```

[ ]:

```python

```

[25]:

```python
from keras.utils.np_utils import to_categorical
from sklearn.preprocessing import LabelEncoder
```

[26]:

```python
t_x, val_x, t_y, val_y = train_test_split(test_images, test_labels, test_size=0.
↪2, )
t_x.shape, val_x.shape, t_y.shape, val_y.shape
```

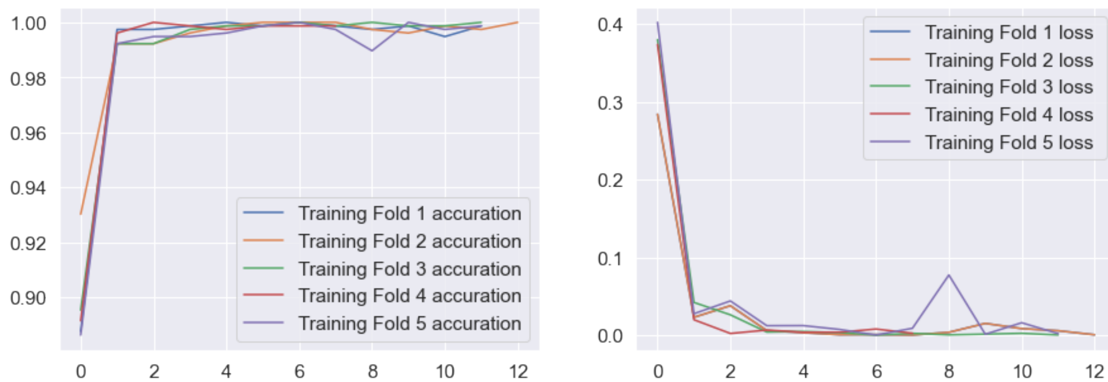
[26]:

```
((764, 150, 150, 3), (192, 150, 150, 3), (764,), (192,))
```

[33]:

```python
import matplotlib.pyplot as plt
fig, (ax1, ax2) =  plt.subplots( ncols=2, sharex=True)
ax1.plot(model_history[0].history['accuracy'], label='Training Fold 1␣
↪accuration')
ax1.plot(model_history[1].history['accuracy'], label='Training Fold 2␣
↪accuration')
ax1.plot(model_history[2].history['accuracy'], label='Training Fold 3␣
↪accuration')
```

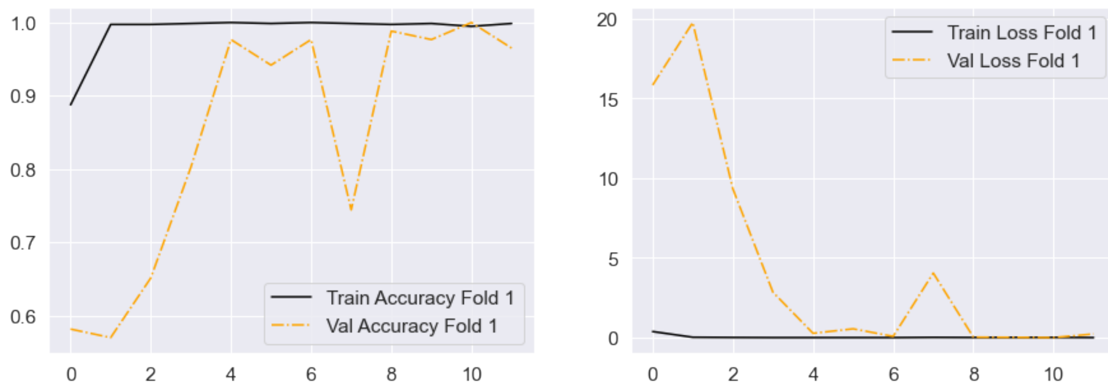
```
ax1.plot(model_history[3].history['accuracy'], label='Training Fold 4␣
 ↪accuration')
ax1.plot(model_history[4].history['accuracy'], label='Training Fold 5␣
 ↪accuration')
ax1.legend()
ax2.plot(model_history[1].history['loss'], label='Training Fold 1 loss')
ax2.plot(model_history[1].history['loss'], label='Training Fold 2 loss')
ax2.plot(model_history[2].history['loss'], label='Training Fold 3 loss')
ax2.plot(model_history[3].history['loss'], label='Training Fold 4 loss ')
ax2.plot(model_history[4].history['loss'], label='Training Fold 5 loss')
ax2.legend()
plt.show()
```



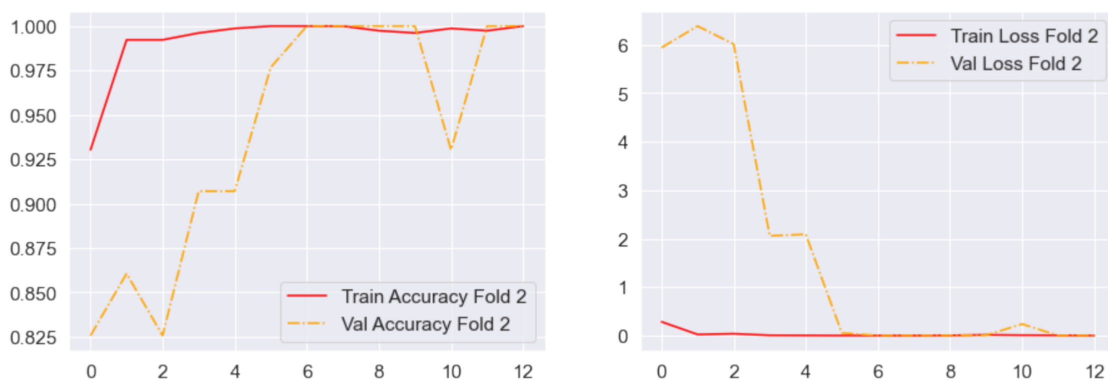
[34]:
```
import matplotlib.pyplot as plt
fig, (plt1, plt2)  =  plt.subplots( ncols=2, sharex=True)
fig = plt.figure(figsize=(5,5))
plt1.plot(model_history[0].history['accuracy'], label='Train Accuracy Fold 1',␣
 ↪color='black')
plt1.plot(model_history[0].history['val_accuracy'], label='Val Accuracy Fold␣
 ↪1', color='orange', linestyle = "dashdot")
plt1.legend()
plt2.plot(model_history[0].history['loss'], label='Train Loss Fold 1',␣
 ↪color='black')
plt2.plot(model_history[0].history['val_loss'], label='Val Loss Fold 1',␣
 ↪color='orange', linestyle = "dashdot")
plt2.legend()
plt.show()

fig, (plt1, plt2)  =  plt.subplots( ncols=2, sharex=True)
fig = plt.figure(figsize=(5,5))
plt1.plot(model_history[1].history['accuracy'], label='Train Accuracy Fold 2',␣
 ↪color='red')
```

```
plt1.plot(model_history[1].history['val_accuracy'], label='Val Accuracy Fold 2', color='orange', linestyle = "dashdot")
plt1.legend()
plt2.plot(model_history[1].history['loss'], label='Train Loss Fold 2', color='red')
plt2.plot(model_history[1].history['val_loss'], label='Val Loss Fold 2', color='orange', linestyle = "dashdot")
plt2.legend()
plt.show()


fig, (plt1, plt2)  =  plt.subplots( ncols=2, sharex=True)
fig = plt.figure(figsize=(5,5))
plt1.plot(model_history[2].history['accuracy'], label='Train Accuracy Fold 3', color='green')
plt1.plot(model_history[2].history['val_accuracy'], label='Val Accuracy Fold 3', color='orange', linestyle = "dashdot")
plt1.legend()
plt2.plot(model_history[2].history['loss'], label='Train Loss Fold 3', color='green')
plt2.plot(model_history[2].history['val_loss'], label='Val Loss Fold 3', color='orange', linestyle = "dashdot")
plt2.legend()
plt.show()


fig, (plt1, plt2)  =  plt.subplots( ncols=2, sharex=True)
fig = plt.figure(figsize=(5,5))
plt1.plot(model_history[3].history['accuracy'], label='Train Accuracy Fold 4', color='blue')
plt1.plot(model_history[3].history['val_accuracy'], label='Val Accuracy Fold 4', color='orange', linestyle = "dashdot")
plt1.legend()
plt2.plot(model_history[3].history['loss'], label='Train Loss Fold 4', color='blue')
plt2.plot(model_history[3].history['val_loss'], label='Val Loss Fold 4', color='orange', linestyle = "dashdot")
plt2.legend()
plt.show()


fig, (plt1, plt2)  =  plt.subplots( ncols=2, sharex=True)
fig = plt.figure(figsize=(5,5))
plt1.plot(model_history[4].history['accuracy'], label='Train Accuracy Fold 5', color='purple')
```

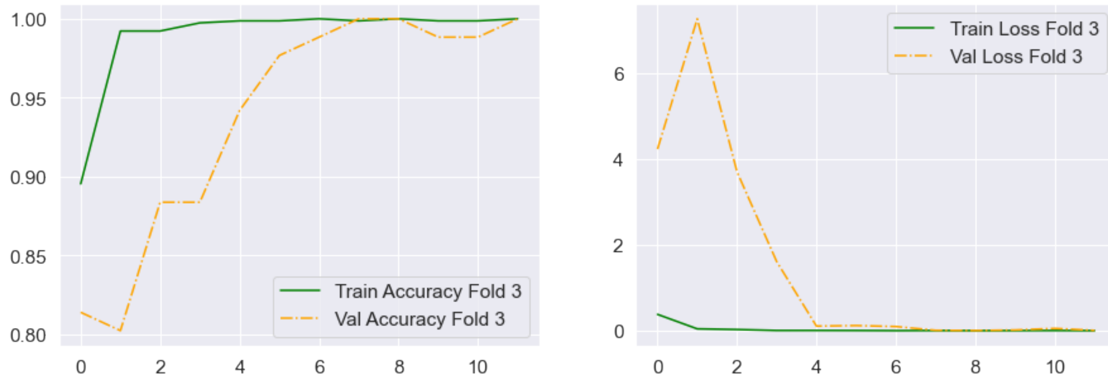
```
plt1.plot(model_history[4].history['val_accuracy'], label='Val Accuracy Fold 5', color='orange', linestyle = "dashdot")
plt1.legend()
plt2.plot(model_history[4].history['loss'], label='Train Loss Fold 4', color='purple')
plt2.plot(model_history[4].history['val_loss'], label='Val Loss Fold 4', color='orange', linestyle = "dashdot")
plt2.legend()
plt.show()
```



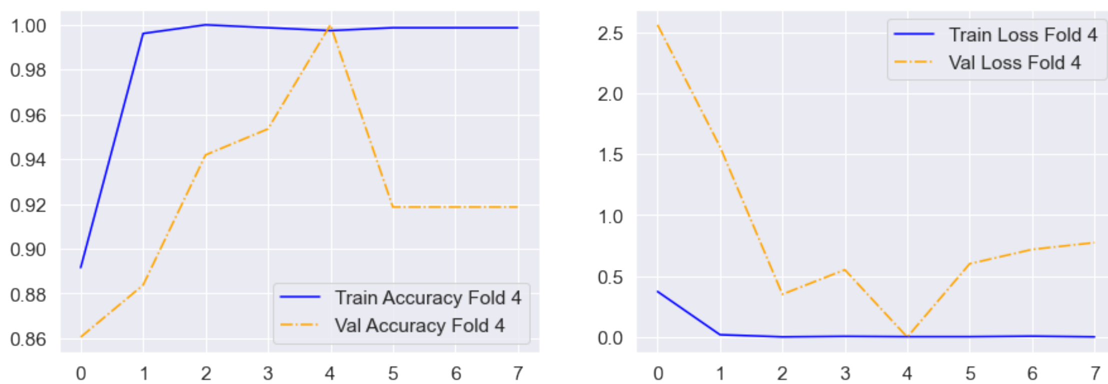
<Figure size 360x360 with 0 Axes>



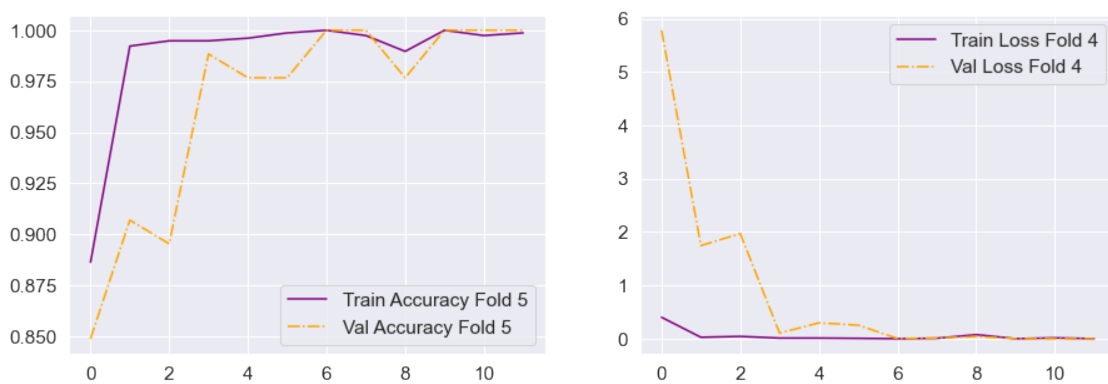
<Figure size 360x360 with 0 Axes>



<Figure size 360x360 with 0 Axes>



<Figure size 360x360 with 0 Axes>



<Figure size 360x360 with 0 Axes>