In [ ]:

```
import pandas as pd
from tensorflow import keras
```

In [ ]:

```
def preprocess_images(dir, subset, image_size):
    '''
    Uses the built-in keras image_dateset_from_directiory function to import, resiz
    shuffle, split, and label the database according to the parameters. The seed, s
    at 313, is arbitrary, but is necessary to ensure no overlap between the trainin
    and testing datasets
    '''
    imgs = keras.utils.image_dataset_from_directory(
        dir,
        subset=subset,
        labels='inferred',
        label_mode='binary',
        class_names=None,
        color_mode='grayscale',
        batch_size=16,
        seed=313,
        image_size=image_size,
        shuffle=True,
        validation_split=0.1,
        interpolation='bilinear',
)

    return imgs
```

In [ ]:

```
from matplotlib import pyplot as plt

# Set dataset directory and split into the training and testing datasets.
dir = 'datasets/brain_mri_scan_images'
image_size = (128, 128)

train_ds = preprocess_images(dir, 'training', image_size)
test_ds = preprocess_images(dir, 'validation', image_size)

# Optional visualization to ensure the dataset imported properly

# class_names = train_ds.class_names
# plt.figure(figsize=(10, 10))
# for images, labels in train_ds.take(1):
#   for i in range(9):
#     print(f'Labels = {int(labels[i])}')
#     ax = plt.subplot(3, 3, i + 1)
#     plt.imshow(images[i].numpy().astype("uint8"), cmap='gray')
#     plt.title(class_names[int(labels[i])] + str(int(labels[i])))
#     plt.axis("off")
```

```
Found 216 files belonging to 2 classes.
Using 195 files for training.
Found 216 files belonging to 2 classes.
Using 21 files for validation.
```

In [ ]:

```
from keras.callbacks import EarlyStopping, ModelCheckpoint, CSVLogger
from keras.models import Model
from keras.layers import Conv2D, Conv2DTranspose, Dense, Flatten, RandomFlip
from keras.layers.pooling import MaxPooling2D
from keras.layers import Input, concatenate
from keras.layers.core import Dropout, Lambda

# Takes in the input as a 128 x 128 image from the image size variable
inputs = Input((image_size[0], image_size[1], 1))

# Normalizes all input dimensions by dividing by 1
s = Lambda(lambda x: x / 255) (inputs)

# Randomly flips ~50% of the data to help improve the parameters
pp_1 = RandomFlip('horizontal') (s)

# U-Net Architecture, taken from the coding tutorial 6

c1 = Conv2D(16, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
c1 = Dropout(0.1) (c1)
c1 = Conv2D(16, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
p1 = MaxPooling2D((2, 2)) (c1)

c2 = Conv2D(32, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
c2 = Dropout(0.1) (c2)
c2 = Conv2D(32, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
p2 = MaxPooling2D((2, 2)) (c2)

c3 = Conv2D(64, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
c3 = Dropout(0.2) (c3)
c3 = Conv2D(64, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
p3 = MaxPooling2D((2, 2)) (c3)

c4 = Conv2D(128, (3, 3), activation='elu', kernel_initializer='he_normal', padding=
c4 = Dropout(0.2) (c4)
c4 = Conv2D(128, (3, 3), activation='elu', kernel_initializer='he_normal', padding=
p4 = MaxPooling2D(pool_size=(2, 2)) (c4)

c5 = Conv2D(256, (3, 3), activation='elu', kernel_initializer='he_normal', padding=
c5 = Dropout(0.3) (c5)
c5 = Conv2D(256, (3, 3), activation='elu', kernel_initializer='he_normal', padding=

u6 = Conv2DTranspose(128, (2, 2), strides=(2, 2), padding='same') (c5)
u6 = concatenate([u6, c4])
c6 = Conv2D(128, (3, 3), activation='elu', kernel_initializer='he_normal', padding=
c6 = Dropout(0.2) (c6)
c6 = Conv2D(128, (3, 3), activation='elu', kernel_initializer='he_normal', padding=

u7 = Conv2DTranspose(64, (2, 2), strides=(2, 2), padding='same') (c6)
u7 = concatenate([u7, c3])
c7 = Conv2D(64, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
c7 = Dropout(0.2) (c7)
c7 = Conv2D(64, (3, 3), activation='elu', kernel_initializer='he_normal', padding='

u8 = Conv2DTranspose(32, (2, 2), strides=(2, 2), padding='same') (c7)
```

```
u8 = concatenate([u8, c2])
c8 = Conv2D(32, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
c8 = Dropout(0.1) (c8)
c8 = Conv2D(32, (3, 3), activation='elu', kernel_initializer='he_normal', padding='

u9 = Conv2DTranspose(16, (2, 2), strides=(2, 2), padding='same') (c8)
u9 = concatenate([u9, c1], axis=3)
c9 = Conv2D(16, (3, 3), activation='elu', kernel_initializer='he_normal', padding='
c9 = Dropout(0.1) (c9)
c9 = Conv2D(16, (3, 3), activation='elu', kernel_initializer='he_normal', padding='

f1 = Flatten() (c9)

outputs = Dense(1, activation="sigmoid") (f1)

model = Model(inputs=[inputs], outputs=[outputs])

model.compile(optimizer='adam', loss='binary_crossentropy')
```

In [ ]:

```
filepath = "model.h5"

earlystopper = EarlyStopping(patience=5, verbose=1)

checkpoint = ModelCheckpoint(filepath, monitor='val_loss', verbose=1,
                             save_best_only=True, mode='min')

csvlogger = CSVLogger('log.csv')


callbacks_list = [earlystopper, checkpoint, csvlogger]


model.compile(loss="binary_crossentropy", optimizer="adam", metrics=["accuracy"])
model.summary()
```

```
Model: "model_24"
____________________________________________________________________________________
______________
 Layer (type)                   Output Shape         Param #     Connected to
====================================================================================
==============
 input_25 (InputLayer)          [(None, 128, 128, 1  0           []
                                )]

 lambda_24 (Lambda)             (None, 128, 128, 1)  0           ['input_25[0][0]']

 random_flip_24 (RandomFlip)    (None, 128, 128, 1)  0           ['lambda_24[0][0]']

 conv2d_432 (Conv2D)            (None, 128, 128, 16  160         ['random_flip_24[0]
[0]']
                                )

 dropout_216 (Dropout)          (None, 128, 128, 16  0           ['conv2d_432[0]
[0]']
                                )

 conv2d_433 (Conv2D)            (None, 128, 128, 16  2320        ['dropout_216[0]
[0]']
                                )

 max_pooling2d_96 (MaxPooling2D  (None, 64, 64, 16)  0           ['conv2d_433[0]
[0]']
 )

 conv2d_434 (Conv2D)            (None, 64, 64, 32)   4640        ['max_pooling2d_96
[0][0]']

 dropout_217 (Dropout)          (None, 64, 64, 32)   0           ['conv2d_434[0]
[0]']

 conv2d_435 (Conv2D)            (None, 64, 64, 32)   9248        ['dropout_217[0]
[0]']

 max_pooling2d_97 (MaxPooling2D  (None, 32, 32, 32)  0           ['conv2d_435[0]
[0]']
 )

 conv2d_436 (Conv2D)            (None, 32, 32, 64)   18496       ['max_pooling2d_97
[0][0]']

 dropout_218 (Dropout)          (None, 32, 32, 64)   0           ['conv2d_436[0]
[0]']

 conv2d_437 (Conv2D)            (None, 32, 32, 64)   36928       ['dropout_218[0]
[0]']

 max_pooling2d_98 (MaxPooling2D  (None, 16, 16, 64)  0           ['conv2d_437[0]
[0]']
 )

 conv2d_438 (Conv2D)            (None, 16, 16, 128)  73856       ['max_pooling2d_98
[0][0]']

 dropout_219 (Dropout)          (None, 16, 16, 128)  0           ['conv2d_438[0]
[0]']

 conv2d_439 (Conv2D)            (None, 16, 16, 128)  147584      ['dropout_219[0]
[0]']

 max_pooling2d_99 (MaxPooling2D  (None, 8, 8, 128)   0           ['conv2d_439[0]
[0]']
 )

 conv2d_440 (Conv2D)            (None, 8, 8, 256)    295168      ['max_pooling2d_99
[0][0]']

 dropout_220 (Dropout)          (None, 8, 8, 256)    0           ['conv2d_440[0]
[0]']

 conv2d_441 (Conv2D)            (None, 8, 8, 256)    590080      ['dropout_220[0]
[0]']

 conv2d_transpose_96 (Conv2DTra  (None, 16, 16, 128)  131200     ['conv2d_441[0]
[0]']
 nspose)

 concatenate_96 (Concatenate)   (None, 16, 16, 256)  0           ['conv2d_transpose_
96[0][0]',

                                                                  'conv2d_439[0]

[0]']

 conv2d_442 (Conv2D)            (None, 16, 16, 128)  295040      ['concatenate_96[0]
[0]']

 dropout_221 (Dropout)          (None, 16, 16, 128)  0           ['conv2d_442[0]
[0]']

 conv2d_443 (Conv2D)            (None, 16, 16, 128)  147584      ['dropout_221[0]
[0]']

 conv2d_transpose_97 (Conv2DTra  (None, 32, 32, 64)  32832       ['conv2d_443[0]
[0]']
 nspose)

 concatenate_97 (Concatenate)   (None, 32, 32, 128)  0           ['conv2d_transpose_
97[0][0]',

                                                                  'conv2d_437[0]

[0]']

 conv2d_444 (Conv2D)            (None, 32, 32, 64)   73792       ['concatenate_97[0]
[0]']

 dropout_222 (Dropout)          (None, 32, 32, 64)   0           ['conv2d_444[0]
[0]']

 conv2d_445 (Conv2D)            (None, 32, 32, 64)   36928       ['dropout_222[0]
[0]']
```

```
 conv2d_transpose_98 (Conv2DTra  (None, 64, 64, 32)  8224        ['conv2d_445[0]
[0]']
 nspose)

 concatenate_98 (Concatenate)   (None, 64, 64, 64)   0           ['conv2d_transpose_
98[0][0]',
                                                                  'conv2d_435[0]
[0]']

 conv2d_446 (Conv2D)            (None, 64, 64, 32)   18464       ['concatenate_98[0]
[0]']

 dropout_223 (Dropout)          (None, 64, 64, 32)   0           ['conv2d_446[0]
[0]']

 conv2d_447 (Conv2D)            (None, 64, 64, 32)   9248        ['dropout_223[0]
[0]']

 conv2d_transpose_99 (Conv2DTra  (None, 128, 128, 16  2064       ['conv2d_447[0]
[0]']
 nspose)                        )

 concatenate_99 (Concatenate)   (None, 128, 128, 32  0           ['conv2d_transpose_
99[0][0]',
                                )                                 'conv2d_433[0]
[0]']

 conv2d_448 (Conv2D)            (None, 128, 128, 16  4624        ['concatenate_99[0]
[0]']
                                )

 dropout_224 (Dropout)          (None, 128, 128, 16  0           ['conv2d_448[0]
[0]']
                                )

 conv2d_449 (Conv2D)            (None, 128, 128, 16  2320        ['dropout_224[0]
[0]']
                                )

 flatten_24 (Flatten)           (None, 262144)       0           ['conv2d_449[0]
[0]']

 dense_24 (Dense)               (None, 1)            262145      ['flatten_24[0]
[0]']

====================================================================================
==============
Total params: 2,202,945
Trainable params: 2,202,945
Non-trainable params: 0
____________________________________________________________________________________
______________
```

In [ ]:

```
batch_size = 16
epochs = 40
```

```
history = model.fit(train_ds, validation_data=test_ds, batch_size=batch_size, epoch
                    callbacks=callbacks_list)
```

```
Epoch 1/40
```

```
12/13 [==========================>...] - ETA: 0s - loss: 8.8496 - accuracy: 0.5833
Epoch 1: val_loss improved from inf to 15.77635, saving model to model.h5
13/13 [==============================] - 2s 80ms/step - loss: 8.7134 - accuracy: 0.5
897 - val_loss: 15.7763 - val_accuracy: 0.4286
Epoch 2/40
13/13 [==============================] - ETA: 0s - loss: 3.5044 - accuracy: 0.6821
Epoch 2: val_loss improved from 15.77635 to 2.96457, saving model to model.h5
13/13 [==============================] - 1s 56ms/step - loss: 3.5044 - accuracy: 0.6
821 - val_loss: 2.9646 - val_accuracy: 0.7143
Epoch 3/40
13/13 [==============================] - ETA: 0s - loss: 1.5063 - accuracy: 0.7590
Epoch 3: val_loss improved from 2.96457 to 1.32371, saving model to model.h5
13/13 [==============================] - 1s 56ms/step - loss: 1.5063 - accuracy: 0.7
590 - val_loss: 1.3237 - val_accuracy: 0.8095
Epoch 4/40
13/13 [==============================] - ETA: 0s - loss: 0.3561 - accuracy: 0.8974
Epoch 4: val_loss improved from 1.32371 to 1.11910, saving model to model.h5
13/13 [==============================] - 1s 55ms/step - loss: 0.3561 - accuracy: 0.8
974 - val_loss: 1.1191 - val_accuracy: 0.7143
Epoch 5/40
13/13 [==============================] - ETA: 0s - loss: 0.1476 - accuracy: 0.9282
Epoch 5: val_loss did not improve from 1.11910
13/13 [==============================] - 1s 46ms/step - loss: 0.1476 - accuracy: 0.9
282 - val_loss: 1.7994 - val_accuracy: 0.7619
Epoch 6/40
13/13 [==============================] - ETA: 0s - loss: 0.0624 - accuracy: 0.9744
Epoch 6: val_loss improved from 1.11910 to 0.96843, saving model to model.h5
13/13 [==============================] - 1s 56ms/step - loss: 0.0624 - accuracy: 0.9
744 - val_loss: 0.9684 - val_accuracy: 0.8095
Epoch 7/40
13/13 [==============================] - ETA: 0s - loss: 0.0281 - accuracy: 0.9897
Epoch 7: val_loss improved from 0.96843 to 0.84683, saving model to model.h5
13/13 [==============================] - 1s 56ms/step - loss: 0.0281 - accuracy: 0.9
897 - val_loss: 0.8468 - val_accuracy: 0.8095
Epoch 8/40
13/13 [==============================] - ETA: 0s - loss: 0.0118 - accuracy: 1.0000
Epoch 8: val_loss improved from 0.84683 to 0.50385, saving model to model.h5
13/13 [==============================] - 1s 56ms/step - loss: 0.0118 - accuracy: 1.0
000 - val_loss: 0.5039 - val_accuracy: 0.9048
Epoch 9/40
13/13 [==============================] - ETA: 0s - loss: 0.0094 - accuracy: 1.0000
Epoch 9: val_loss did not improve from 0.50385
13/13 [==============================] - 1s 47ms/step - loss: 0.0094 - accuracy: 1.0
000 - val_loss: 0.7098 - val_accuracy: 0.8571
Epoch 10/40
13/13 [==============================] - ETA: 0s - loss: 0.0062 - accuracy: 1.0000
Epoch 10: val_loss did not improve from 0.50385
13/13 [==============================] - 1s 46ms/step - loss: 0.0062 - accuracy: 1.0
000 - val_loss: 0.8317 - val_accuracy: 0.8095
Epoch 11/40
13/13 [==============================] - ETA: 0s - loss: 0.0048 - accuracy: 1.0000
Epoch 11: val_loss did not improve from 0.50385
13/13 [==============================] - 1s 47ms/step - loss: 0.0048 - accuracy: 1.0
000 - val_loss: 0.6316 - val_accuracy: 0.8571
Epoch 12/40
13/13 [==============================] - ETA: 0s - loss: 0.0023 - accuracy: 1.0000
```

```
Epoch 12: val_loss did not improve from 0.50385
13/13 [==============================] - 1s 46ms/step - loss: 0.0023 - accuracy: 1.0
000 - val_loss: 0.7343 - val_accuracy: 0.8095
Epoch 13/40
13/13 [==============================] - ETA: 0s - loss: 0.0022 - accuracy: 1.0000
Epoch 13: val_loss did not improve from 0.50385
13/13 [==============================] - 1s 47ms/step - loss: 0.0022 - accuracy: 1.0
000 - val_loss: 0.6668 - val_accuracy: 0.9048
Epoch 13: early stopping
```

In [ ]:

```python
import os
import re

# All this code is to check if the new, best model beats the existing best in both
# best-model model if it does

mvl_index = history.history['val_loss'].index(min(history.history['val_loss']))
min_val_loss = round(history.history['val_loss'][mvl_index], 4)
val_accuracy = round(history.history['val_accuracy'][mvl_index], 4)

print(history.history['val_loss'])
print(f'Index = {mvl_index}, Val_Loss = {min_val_loss}, Val_Acc = {val_accuracy}')

for file in os.listdir('.'):
    if file.endswith('.h5') and file.startswith('model-best'):
        best_val_loss, best_val_accuracy = re.findall('\d+\.\d+', file)
        print(best_val_loss, best_val_accuracy)


if min_val_loss < float(best_val_loss) and val_accuracy > float(best_val_accuracy):
    best_model = f'model-best ({best_val_loss} - {best_val_accuracy}).h5'
    new_model = f'model-best ({min_val_loss} - {val_accuracy}).h5'

    print(best_model, new_model)

    os.rename('model.h5', new_model)
    os.remove(best_model)

    os.remove('log-best.csv')
    os.rename('log.csv', 'log-best.csv')
```

```
[15.776347160339355, 2.964573621749878, 1.323712706565857, 1.1190996170043945, 1.799
4273900985718, 0.9684255719184875, 0.8468292951583862, 0.5038547515869141, 0.7097520
82824707, 0.8316628336906433, 0.6316357851028442, 0.7343470454216003, 0.666828274726
8677]
Index = 7, Val_Loss = 0.5039, Val_Acc = 0.9048
0.4147 0.8571
```

In [ ]:

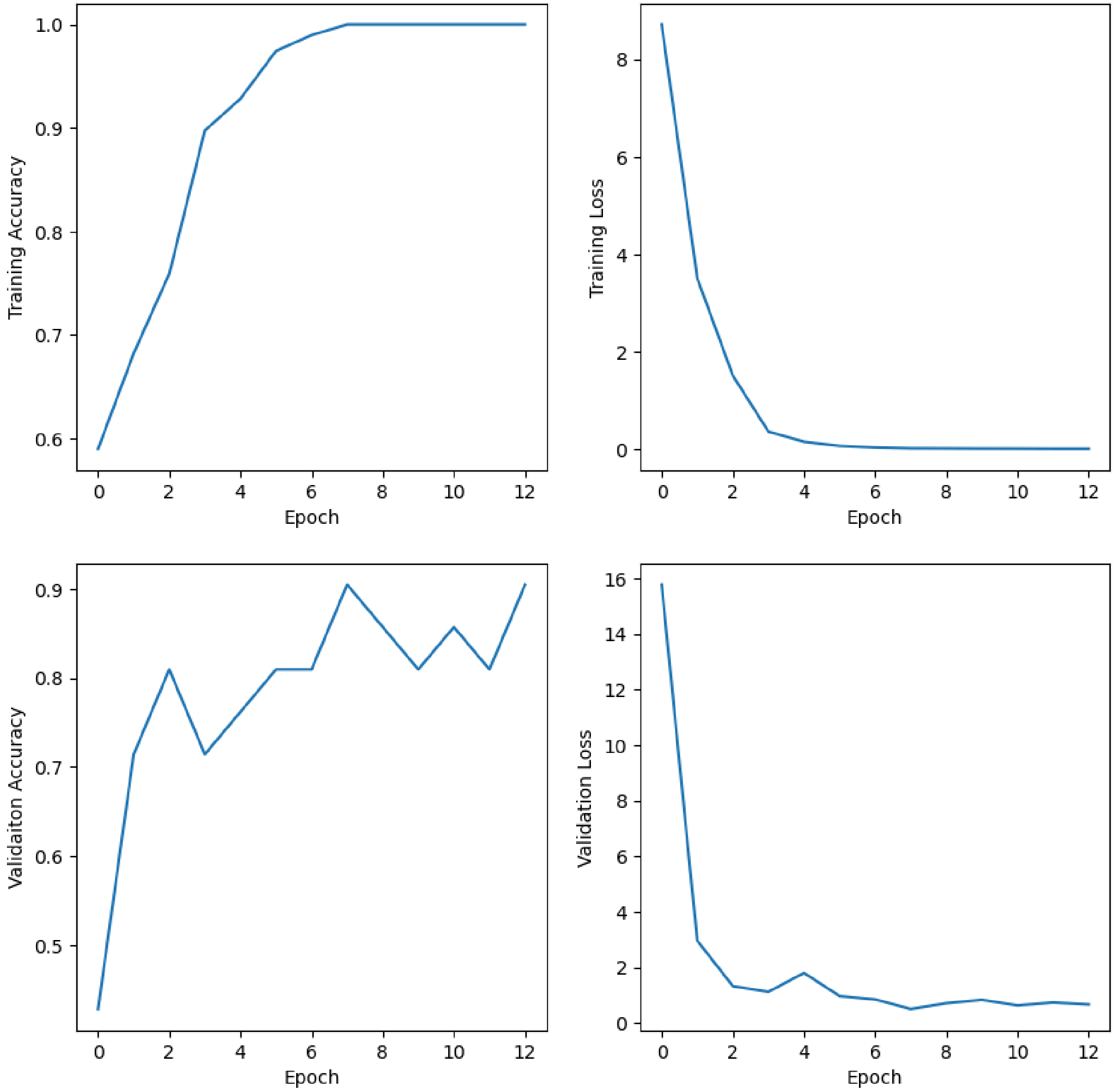
```python
output = pd.read_csv('log-best.csv')
# print(output.head())

plt.figure(1, figsize=(10, 10))
plt.subplot(2, 2, 1)
plt.plot(output['epoch'], output['accuracy'])
plt.xlabel('Epoch')
plt.ylabel('Training Accuracy')
plt.subplot(2, 2, 2)
plt.plot(output['epoch'], output['loss'])
plt.xlabel('Epoch')
plt.ylabel('Training Loss')

plt.subplot(2, 2, 3)
plt.plot(output['epoch'], output['val_accuracy'])
plt.xlabel('Epoch')
plt.ylabel('Validaiton Accuracy')

plt.subplot(2, 2, 4)
plt.plot(output['epoch'], output['val_loss'])
plt.xlabel('Epoch')
plt.ylabel('Validation Loss')
```

```
Out[ ]: Text(0, 0.5, 'Validation Loss')
```


Training Accuracy
Epoch
Training Loss
Epoch
Validaiton Accuracy
Epoch
Validation Loss
Epoch