cnn_v1_stage

November 25, 2022

[ ]:
```
%env LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CONDA_PREFIX/lib/
#%env TF_GPU_ALLOCATOR=cuda_malloc_async
```

```
env: LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CONDA_PREFIX/lib/
```

[ ]:
```
import os
print(os.environ["LD_LIBRARY_PATH"])
```

```
$LD_LIBRARY_PATH:$CONDA_PREFIX/lib/
```

[ ]:
```
import tensorflow as tf
import numpy as np
import pandas as pd
import os
import matplotlib.pyplot as plt

from tensorflow.keras import Sequential, models, Input
from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D,␣
 ↪Dropout, LeakyReLU, AveragePooling2D, GlobalAveragePooling2D,␣
 ↪BatchNormalization, TimeDistributed, LSTM, SpatialDropout2D, concatenate
from tensorflow.keras.optimizers import SGD, Adam
```

```
2022-11-25 11:53:21.459527: I tensorflow/core/platform/cpu_feature_guard.cc:193]
This TensorFlow binary is optimized with oneAPI Deep Neural Network Library
(oneDNN) to use the following CPU instructions in performance-critical
operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate
compiler flags.
2022-11-25 11:53:22.497099: E tensorflow/stream_executor/cuda/cuda_blas.cc:2981]
Unable to register cuBLAS factory: Attempting to register factory for plugin
cuBLAS when one has already been registered
2022-11-25 11:53:23.505095: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libnvinfer.so.7'; dlerror: libnvinfer.so.7: cannot open shared
object file: No such file or directory; LD_LIBRARY_PATH:
:/home/nkspartan/miniconda3/envs/tf-gpu/lib/
2022-11-25 11:53:23.505179: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
```

```
dynamic library 'libnvinfer_plugin.so.7'; dlerror: libnvinfer_plugin.so.7:
cannot open shared object file: No such file or directory; LD_LIBRARY_PATH:
:/home/nkspartan/miniconda3/envs/tf-gpu/lib/
2022-11-25 11:53:23.505185: W
tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Cannot
dlopen some TensorRT libraries. If you would like to use Nvidia GPU with
TensorRT, please make sure the missing libraries mentioned above are installed
properly.
```

[ ]:

```
gpus = tf.config.experimental.list_physical_devices('GPU')
for gpu in gpus:
  tf.config.experimental.set_memory_growth(gpu, True)
```

```
2022-11-25 11:53:24.976824: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:53:25.027810: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:53:25.028090: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
```

[ ]:

```
from tensorflow.python.client import device_lib

print('Default GPU Device: {}'.format(tf.test.gpu_device_name()))
```

```
2022-11-25 11:53:25.085015: I tensorflow/core/platform/cpu_feature_guard.cc:193]
This TensorFlow binary is optimized with oneAPI Deep Neural Network Library
(oneDNN) to use the following CPU instructions in performance-critical
operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate
compiler flags.
2022-11-25 11:53:25.086183: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:53:25.086397: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:53:25.086539: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
```

```
2022-11-25 11:53:25.929895: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:53:25.930508: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from S
```

```
Default GPU Device: /device:GPU:0
```

```
ysFS had negative value (-1), but there must be at least one NUMA node, so
returning NUMA node zero
2022-11-25 11:53:25.930819: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:53:25.931063: I
tensorflow/core/common_runtime/gpu/gpu_device.cc:1616] Created device
/device:GPU:0 with 4063 MB memory:  -> device: 0, name: NVIDIA GeForce RTX 2060,
pci bus id: 0000:08:00.0, compute capability: 7.5
```

0.1 Read the csv dataset to get the values for stage and discharge of the images

[ ]:
```
df = pd.read_csv("../../dataset/2012_2019_PlatteRiverWeir_features_merged_all.
 ↪csv")
df.head()
```

[ ]:
```
   Unnamed: 0           SensorTime          CaptureTime  \
0           0  2012-06-09 13:15:00  2012-06-09T13:09:07
1           1  2012-06-09 13:15:00  2012-06-09T13:10:29
2           2  2012-06-09 13:45:00  2012-06-09T13:44:01
3           3  2012-06-09 14:45:00  2012-06-09T14:44:30
4           4  2012-06-09 15:45:00  2012-06-09T15:44:59

                                  Filename Agency  SiteNumber TimeZone  Stage  \
0  StateLineWeir_20120609_Farrell_001.jpg   USGS     6674500      MDT   2.99
1  StateLineWeir_20120609_Farrell_002.jpg   USGS     6674500      MDT   2.99
2  StateLineWeir_20120609_Farrell_003.jpg   USGS     6674500      MDT   2.96
3  StateLineWeir_20120609_Farrell_004.jpg   USGS     6674500      MDT   2.94
4  StateLineWeir_20120609_Farrell_005.jpg   USGS     6674500      MDT   2.94

   Discharge        CalcTimestamp  …  WeirPt2X  WeirPt2Y  WwRawLineMin  \
0      916.0  2020-03-11T16:58:28  …        -1        -1           0.0
1      916.0  2020-03-11T16:58:33  …        -1        -1           0.0
2      873.0  2020-03-11T16:58:40  …        -1        -1           0.0
3      846.0  2020-03-11T16:58:47  …        -1        -1           0.0
4      846.0  2020-03-11T16:58:55  …        -1        -1           0.0
```

```
   WwRawLineMax  WwRawLineMean  WwRawLineSigma  WwCurveLineMin  \
0           0.0            0.0             0.0             0.0
1           0.0            0.0             0.0             0.0
2           0.0            0.0             0.0             0.0
3           0.0            0.0             0.0             0.0
4           0.0            0.0             0.0             0.0

   WwCurveLineMax  WwCurveLineMean  WwCurveLineSigma
0             0.0              0.0               0.0
1             0.0              0.0               0.0
2             0.0              0.0               0.0
3             0.0              0.0               0.0
4             0.0              0.0               0.0

[5 rows x 60 columns]
```

[ ]:

```
#df = df[["Filename", "Stage", "Discharge", 'SensorTime', "RiverArea",␣
 ↪"RiverWidth"]]
df = df[["Filename", "Stage", "Discharge", 'SensorTime']]
```

[ ]:

```
df['SensorTime'] = pd.to_datetime(df['SensorTime'])
df['Year'] = df['SensorTime'].dt.year
df.head()
```

[ ]:

```
                                 Filename  Stage  Discharge  \
0  StateLineWeir_20120609_Farrell_001.jpg   2.99      916.0
1  StateLineWeir_20120609_Farrell_002.jpg   2.99      916.0
2  StateLineWeir_20120609_Farrell_003.jpg   2.96      873.0
3  StateLineWeir_20120609_Farrell_004.jpg   2.94      846.0
4  StateLineWeir_20120609_Farrell_005.jpg   2.94      846.0

          SensorTime  Year
0 2012-06-09 13:15:00  2012
1 2012-06-09 13:15:00  2012
2 2012-06-09 13:45:00  2012
3 2012-06-09 14:45:00  2012
4 2012-06-09 15:45:00  2012
```

[ ]:

```
df = df.sort_values(by="SensorTime", ascending=True)
df.head()
```

[ ]:

```
                                 Filename  Stage  Discharge  \
0  StateLineWeir_20120609_Farrell_001.jpg   2.99      916.0
1  StateLineWeir_20120609_Farrell_002.jpg   2.99      916.0
2  StateLineWeir_20120609_Farrell_003.jpg   2.96      873.0
3  StateLineWeir_20120609_Farrell_004.jpg   2.94      846.0
4  StateLineWeir_20120609_Farrell_005.jpg   2.94      846.0
```

```
           SensorTime  Year
0 2012-06-09 13:15:00  2012
1 2012-06-09 13:15:00  2012
2 2012-06-09 13:45:00  2012
3 2012-06-09 14:45:00  2012
4 2012-06-09 15:45:00  2012
```

0.1.1 Remove outliers

[ ]:

```
df = df[df.Stage > 0]
df = df[df.Discharge > 0]
```

We consider values equal to 0 as outliers because from the photos it doesn't seem that it would be possible that at this time we would have a value of 0 for stage or discharge

[ ]:

```
df.shape
```

[ ]:

```
(40148, 5)
```

0.1.2 Scale the data

[ ]:

```
from sklearn.preprocessing import StandardScaler, MinMaxScaler, RobustScaler
from joblib import load

scaler = RobustScaler()
#scaler = load('std_scaler.joblib') # scaler with all the 42059 observations
```

Scale the data based only on the training dataset (in this case the training dataset is from 2012 to 2016)

[ ]:

```
#data_to_scale_fit = df[(df["Year"] >= 2012) & (df["Year"] <= 2016)][["Stage",␣
 ↪"Discharge"]]
data_to_scale_fit = df[(df["Year"] >= 2012) & (df["Year"] <=␣
 ↪2016)][["Discharge"]]
data_to_scale_fit
```

[ ]:

```
       Discharge
0          916.0
1          916.0
2          873.0
3          846.0
4          846.0
…          …
21416      279.0
21417      279.0
21418      279.0
```

```
21419      279.0
21420      279.0

[20304 rows x 1 columns]
```

[ ]:
```
scaler.fit(data_to_scale_fit)
```

[ ]:
```
RobustScaler()
```

[ ]:
```
#df[["Stage", "Discharge"]] = scaler.transform(df[["Stage", "Discharge"]])
#df[["Discharge"]] = scaler.transform(df[["Discharge"]])
df
```

[ ]:
```
                                    Filename  Stage  Discharge  \
0      StateLineWeir_20120609_Farrell_001.jpg   2.99      916.0
1      StateLineWeir_20120609_Farrell_002.jpg   2.99      916.0
2      StateLineWeir_20120609_Farrell_003.jpg   2.96      873.0
3      StateLineWeir_20120609_Farrell_004.jpg   2.94      846.0
4      StateLineWeir_20120609_Farrell_005.jpg   2.94      846.0
…                                        …    …        …
42054  StateLineWeir_20191011_Farrell_409.jpg   2.54      434.0
42055  StateLineWeir_20191011_Farrell_410.jpg   2.54      434.0
42056  StateLineWeir_20191011_Farrell_411.jpg   2.54      434.0
42057  StateLineWeir_20191011_Farrell_412.jpg   2.54      434.0
42058  StateLineWeir_20191011_Farrell_413.jpg   2.54      434.0

               SensorTime  Year
0     2012-06-09 13:15:00  2012
1     2012-06-09 13:15:00  2012
2     2012-06-09 13:45:00  2012
3     2012-06-09 14:45:00  2012
4     2012-06-09 15:45:00  2012
…                     …    …
42054 2019-10-11 09:00:00  2019
42055 2019-10-11 10:00:00  2019
42056 2019-10-11 11:00:00  2019
42057 2019-10-11 12:00:00  2019
42058 2019-10-11 12:45:00  2019

[40148 rows x 5 columns]
```

[ ]:
```
df.describe()
```

[ ]:
```
              Stage     Discharge          Year
count  40148.000000  40148.000000  40148.000000
mean       2.903601   1017.063288   2016.168228
std        0.814612   1200.944046      1.997968
```

```
min        1.370000      6.730000   2012.000000
25%        2.280000    226.000000   2015.000000
50%        2.600000    451.500000   2016.000000
75%        3.320000   1390.000000   2018.000000
max        6.490000   7920.000000   2019.000000
```

[ ]:

```
from joblib import dump
#dump(scaler, 'std_scaler_train_value_0_outliers.joblib')
```

0.2 Create the dataset pipeline

[ ]:

```
#IMG_SIZE = 224
IMG_SIZE = 320
BATCH_SIZE = 8
FRAMES = 10
```

[ ]:

```
from dataset_transformer import make_dataset_with_time, make_dataset,␣
 ↪make_dataset_and_time

from dataset_transformer_2 import Dataset, Dataloader
```

[ ]:

```
path = "../../dataset/images_tmp_draw_sky"

#train_ds, train_size, val_ds, val_size, test_ds, test_size =␣
 ↪make_dataset_and_time(path, BATCH_SIZE, IMG_SIZE, FRAMES, df, 10, True,␣
 ↪"cnn")

train_dataset = Dataset(
    path,
    df[(df.Year >= 2012) & (df.Year <= 2016)],
    classes=[0, 1],
)

# Dataset for validation images
val_dataset = Dataset(
    path,
    df[(df.Year >= 2017) & (df.Year <= 2017)],
    classes=[0, 1],
)

test_dataset = Dataset(
    path,
    df[(df.Year >= 2018) & (df.Year <= 2019)],
    classes=[0, 1],
    shuffle=False
)
```

```
train_ds = Dataloader(train_dataset, batch_size=BATCH_SIZE, shuffle=True)
val_ds = Dataloader(val_dataset, batch_size=BATCH_SIZE, shuffle=False)
test_ds = Dataloader(test_dataset, batch_size=1, shuffle=False)
```

[ ]:

```
input_shape = 0
output_shape = 0

for i in range(0, 1):
    image_time, stage_discharge = test_ds[i]
    #print(np.array(image_time).shape)

    image = image_time["input_1"]
    stage_discharge = stage_discharge
    #print(stage_discharge.shape)

    print(stage_discharge.shape)

    input_shape = image.shape[1:]
    #input_shape = image_time.numpy().shape[1:]
    output_shape = stage_discharge.shape[1:]
```

```
(8, 1)
```

[ ]:

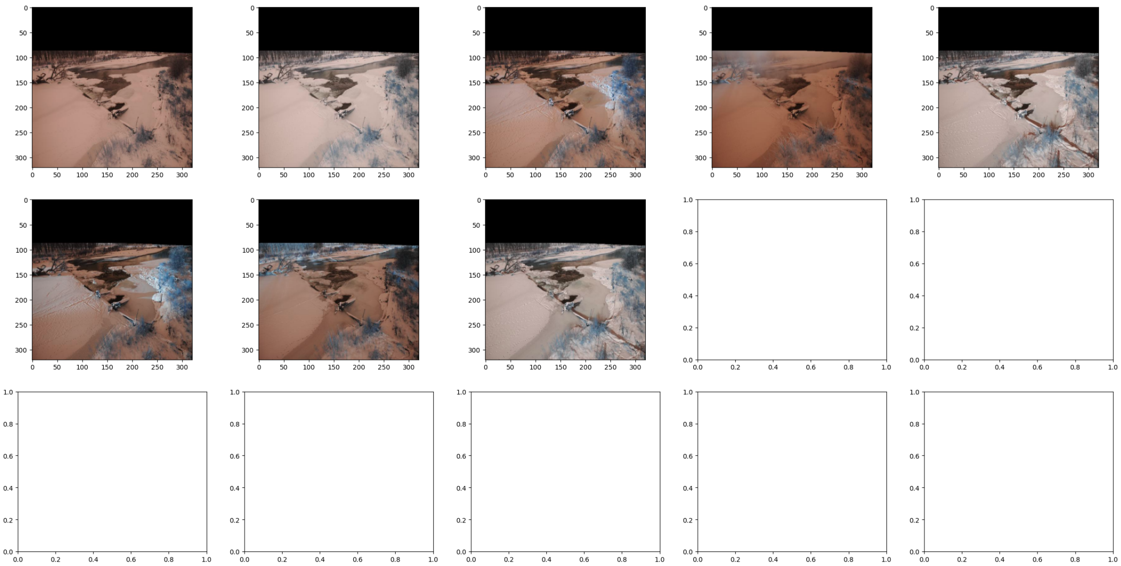
```
print(input_shape)
print(output_shape)
```

```
(320, 320, 3)
(1,)
```

0.3 Check images

[ ]:

```
fig, ax = plt.subplots(nrows=3, ncols=5, figsize=(30, 15))

for i in range(0, 1):
    image_time, stage_discharge = test_ds[i]
    images = image_time["input_1"]

    for img, ax in zip(images, ax.flatten()):
        #print(img.numpy()[:,:,3])
        #img = img.numpy()[:,:,:3]
        #img = img / 2 + 0.5     # unnormalize

        #print(img)
        ax.imshow(img)

plt.show()
```



0.4 Create model

[ ]:

```
from scipy import ndimage as ndi
from skimage.filters import gabor_kernel

def sobel_kernel(shape, dtype=None):
    #print(shape)
    sobel_x = tf.constant(
        [
            [-5, -4, 0, 4, 5],
            [-8, -10, 0, 10, 8],
            [-10, -20, 0, 20, 10],
            [-8, -10, 0, 10, 8],
            [-5, -4, 0, 4, 5]
        ], dtype=dtype )
    #create the missing dims.
    sobel_x = tf.reshape(sobel_x, (5, 5, 1, 1))

    #print(tf.shape(sobel_x))
    #tile the last 2 axis to get the expected dims.
    sobel_x = tf.tile(sobel_x, (1, 1, shape[-2],shape[-1]))

    #print(tf.shape(sobel_x))
    return sobel_x

def gfb_filter(shape, size=3, tlist=[1,2,3], slist=[2,5], flist=[0.01,0.25,0.
 ↪5], dtype=None):
    print(shape)
    fsize=np.ones([size,size])
    kernels = []
    for theta in tlist:
        theta = theta / 4. * np.pi
        for sigma in slist:
            for frequency in flist:
                kernel = np.real(gabor_kernel(frequency,␣
↪theta=theta,sigma_x=sigma, sigma_y=sigma))
                kernels.append(kernel)
    gfblist = []
    for k, kernel in enumerate(kernels):
        ck=ndi.convolve(fsize, kernel, mode='wrap')
        gfblist.append(ck)

    gfblist = np.asarray(gfblist).reshape(size,size,1,len(gfblist))
    gfblist = np.repeat(gfblist[:, :, :, :], gfblist.shape[1], axis=2)
    print(gfblist.shape)
    return tf.keras.backend.variable(gfblist, dtype='float32')
```

[ ]:
```
import segmentation_models as sm

seg_model = sm.Unet("resnet50", classes=1, activation="sigmoid")

seg_model.load_weights(
    f'model_weights/seg_model_resnet_50_1.hdf5')
```

[ ]:
```
from classification_models.keras import Classifiers

def create_model(input_shape, output_shape, option="normal"):
    model = Sequential()

    if option == "transfer":
        # Inputs
        input_base = Input(shape=input_shape, name="input_1")
        #time_area_input = Input(shape=(2), name="input_2")
        time_input = Input(shape=(1), name="input_2")


        base_model = tf.keras.applications.ResNet50V2(include_top=False,
                                                       weights='imagenet',
                                                       input_shape=input_shape)

        #base_model = ResNet34(include_top=False, weights='imagenet',␣
↪input_shape=input_shape)

        for layer in base_model.layers:
            layer.trainable = False
        base_model._name = 'base_model_ResNet50V2'


        cnn_model = base_model(input_base)


        cnn_model = Dropout(0.5)(cnn_model)
        cnn_model = GlobalAveragePooling2D()(cnn_model)
        cnn_model = Dense(1024, activation="relu")(cnn_model)
        cnn_model = Dense(512, activation="relu")(cnn_model)

        #edge_detection = Conv2D(4, kernel_size=(5, 5),␣
↪kernel_initializer=sobel_kernel, strides=(2, 2), activation='relu',␣
↪trainable=False)

        #edge_detection = edge_detection(input_base)
        #edge_detection.trainable = False
        #edge_detection = GlobalAveragePooling2D()(edge_detection)
        #edge_detection = Dense(512, activation="elu")(edge_detection)
        #edge_detection = Dense(512, activation="elu")(edge_detection)

        """
        gfb = Conv2D(filters=18, kernel_size=3, kernel_initializer=gfb_filter,␣
↪strides=1, padding='valid', trainable=False, name="Gabor_filter")
        gfb = gfb(input_base)
        gfb.trainable = False
        gfb = GlobalAveragePooling2D()(gfb)
        gfb = Dense(1024, activation="relu")(gfb)
        gfb = Dense(512, activation="relu")(gfb)
        """

        combined = concatenate([cnn_model, time_input], name="combined_model")
        #combined = BatchNormalization()(combined)

        cnn_time = Dense(513, activation="elu")(combined)
        #cnn_time = Dropout(0.3)(cnn_time)
        cnn_time = Dense(256, activation="elu")(cnn_time)
        #cnn_time = Dropout(0.3)(cnn_time)
        cnn_time = Dense(128, activation="elu")(cnn_time)
        cnn_time = Dense(64, activation="elu")(cnn_time)
        output = Dense(output_shape, activation='linear')(cnn_time)

        model = tf.keras.Model([input_base, time_input], output,␣
↪name="cnn_segmentation")
    elif option == "normal":
        model.add(Input(shape=input_shape))
```

```
       """model.add(Conv2D(16, kernel_size=(3, 3), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
       model.add(BatchNormalization())

       model.add(Conv2D(32, kernel_size=(3, 3), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
       model.add(BatchNormalization())

       model.add(Conv2D(32, kernel_size=(3, 3), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(Conv2D(32, kernel_size=(3, 3), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
       model.add(BatchNormalization())

       model.add(Conv2D(64, kernel_size=(4, 4), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(Conv2D(64, kernel_size=(4, 4), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
       model.add(BatchNormalization())

       model.add(Conv2D(64, kernel_size=(4, 4), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(Conv2D(64, kernel_size=(4, 4), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
       model.add(BatchNormalization())

       model.add(Conv2D(64, kernel_size=(3, 3), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(Conv2D(64, kernel_size=(3, 3), activation="elu",␣
↪padding='same', kernel_initializer='he_uniform'))
       model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
       model.add(BatchNormalization())

       model.add(GlobalAveragePooling2D())

       model.add(Dense(512, activation='elu'))
       model.add(Dropout(0.3))
       model.add(Dense(512, activation='elu'))
       model.add(Dropout(0.3))
       model.add(Dense(256, activation='elu'))
```

```
        model.add(Dense(64, activation='elu'))"""

        model.add(Conv2D(32, kernel_size=(4, 4), strides=(2, 2),␣
↪padding='same', activation="elu"))
        model.add(MaxPooling2D(pool_size=(2, 2)))

        model.add(Conv2D(32, kernel_size=(3, 3), strides=(2, 2),␣
↪activation="elu", padding='same'))
        model.add(MaxPooling2D(pool_size=(2, 2)))

        model.add(Conv2D(32, kernel_size=(3, 3), activation="elu",␣
↪padding='same'))
        #model.add(MaxPooling2D(pool_size=(2, 2)))

        model.add(Conv2D(32, kernel_size=(3, 3), activation='elu'))

        model.add(Conv2D(32, kernel_size=(3, 3), activation='elu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))

        model.add(Conv2D(64, kernel_size=(2, 2), activation='elu'))

        model.add(Conv2D(64, kernel_size=(2, 2), activation='elu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))

        model.add(Flatten())
        model.add(Dense(2048, activation='relu'))
        model.add(Dense(2048, activation='relu'))
        model.add(Dense(1024, activation='relu'))
        model.add(Dense(1024, activation='relu'))

    #model.add(Dense(output_shape, activation='linear')) # linear regression␣
↪output layer

    return model
```

[ ]:
```
model = create_model(input_shape, output_shape[0], "transfer")
```

[ ]:
```
model.summary()
```

```
Model: "cnn_segmentation"
________________________________________________________________________________
__________________
 Layer (type)                   Output Shape         Param #     Connected to
================================================================================
==================
 input_1 (InputLayer)           [(None, 320, 320, 3  0           []
                                )]
 base_model_ResNet50V2 (Functio  (None, 10, 10, 2048  23564800
['input_1[0][0]']
 nal)                           )

 dropout (Dropout)              (None, 10, 10, 2048  0
['base_model_ResNet50V2[0][0]']
                                )

 global_average_pooling2d (Glob  (None, 2048)        0
['dropout[0][0]']
 alAveragePooling2D)

 dense_5 (Dense)                (None, 1024)         2098176
['global_average_pooling2d[0][0]'
                                                                  ]

 dense_6 (Dense)                (None, 512)          524800
['dense_5[0][0]']

 input_2 (InputLayer)           [(None, 1)]          0            []

 combined_model (Concatenate)   (None, 513)          0
['dense_6[0][0]',
'input_2[0][0]']

 dense_7 (Dense)                (None, 513)          263682
['combined_model[0][0]']

 dense_8 (Dense)                (None, 256)          131584
['dense_7[0][0]']

 dense_9 (Dense)                (None, 128)          32896
['dense_8[0][0]']

 dense_10 (Dense)               (None, 64)           8256
['dense_9[0][0]']

 dense_11 (Dense)               (None, 1)            65
['dense_10[0][0]']

================================================================================
==================
Total params: 26,624,259
Trainable params: 3,059,459
Non-trainable params: 23,564,800
________________________________________________________________________________
__________________
```

[ ]:

```
def compile_model(loss_func, optimizer, metrics=["accuracy"]):
    model.compile(loss=loss_func, optimizer=optimizer, metrics=metrics)
```

[ ]:

```
import tensorflow_addons as tfa
```

[ ]:

```
sgd = SGD(learning_rate=0.01, decay=1e-3, momentum=0.9, nesterov=True)
adam = Adam(learning_rate=1e-3, decay=1e-3 / 200)

"""compile_model(tfa.losses.PinballLoss(tau=.6), adam,  [
              'mse', tf.keras.metrics.RootMeanSquaredError(name='rmse'), 'mae',␣
 ↪'mape'])"""

"""compile_model(tf.keras.losses.Huber(), adam,  [
              'mse', tf.keras.metrics.RootMeanSquaredError(name='rmse'), 'mae',␣
 ↪'mape'])"""

compile_model("mse", adam,  [
              'mse', tf.keras.metrics.RootMeanSquaredError(name='rmse'), 'mae',␣
 ↪'mape'])
```

[ ]:

```
def fit_model(training_values, validation_values=None, epochs=10, steps=32,␣
 ↪val_steps=32, callbacks=[]):
    return model.fit(training_values, validation_data=validation_values,␣
 ↪epochs=epochs, steps_per_epoch=steps, validation_steps=val_steps,␣
 ↪callbacks=callbacks)
```

[ ]:

```
import datetime

date_actual = datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
log_dir = "logs/fit/" + date_actual
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,␣
 ↪histogram_freq=1)

es_callback = tf.keras.callbacks.EarlyStopping(monitor='val_loss', mode='min',␣
 ↪verbose=1, patience=15)

checkpoint_callback = tf.keras.callbacks.
 ↪ModelCheckpoint(filepath=f"model_weights/{date_actual}_cnn_best_weights.
 ↪hdf5",
                               monitor='val_loss',
                               verbose=1,
                               save_best_only=True)
```

[ ]:

```
# batch_size = 0 because we already have batch size in tf dataset
```

```
model_h = fit_model(train_ds, val_ds, epochs=100, steps=len(train_ds),␣
 ↪val_steps=len(val_ds), callbacks=[tensorboard_callback, checkpoint_callback,␣
 ↪es_callback, tf.keras.callbacks.ReduceLROnPlateau(patience=5)])

#model.fit(train_ds, validation_data=val_ds, epochs=60,␣
 ↪steps_per_epoch=len(train_ds), validation_steps=len(val_ds),␣
 ↪callbacks=[tensorboard_callback, checkpoint_callback, es_callback, tf.keras.
 ↪callbacks.ReduceLROnPlateau(patience=5)])
```

```
Epoch 1/100
```

```
2022-11-24 21:21:53.905353: I tensorflow/stream_executor/cuda/cuda_dnn.cc:384]
Loaded cuDNN version 8100
2022-11-24 21:21:55.199009: I
tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child
process: No such file or directory
2022-11-24 21:21:55.200399: I
tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child
process: No such file or directory
2022-11-24 21:21:55.200480: W tensorflow/stream_executor/gpu/asm_compiler.cc:80]
Couldn't get ptxas version string: INTERNAL: Couldn't invoke ptxas --version
2022-11-24 21:21:55.202030: I
tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child
process: No such file or directory
2022-11-24 21:21:55.202173: W
tensorflow/stream_executor/gpu/redzone_allocator.cc:314] INTERNAL: Failed to
launch ptxas
Relying on driver to perform ptx compilation.
Modify $PATH to customize ptxas location.
This message will be only logged once.
2022-11-24 21:21:56.387001: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.21GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
2022-11-24 21:21:56.387040: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.21GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
2022-11-24 21:21:56.596477: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.30GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
2022-11-24 21:21:56.596527: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.30GiB with freed_by_count=0. The caller
```

```
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
2022-11-24 21:21:56.841150: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.54GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
2022-11-24 21:21:56.841186: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.54GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.

2538/2538 [==============================] - ETA: 0s - loss: 3.1358 - mse:
3.1358 - rmse: 1.7708 - mae: 0.6604 - mape: 23.3414

2022-11-24 21:26:27.013064: W
tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 566231040
exceeds 10% of free system memory.
2022-11-24 21:26:27.218105: W
tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 503316480
exceeds 10% of free system memory.
2022-11-24 21:26:27.565439: W
tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 566231040
exceeds 10% of free system memory.
2022-11-24 21:26:27.940428: W
tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 566231040
exceeds 10% of free system memory.
2022-11-24 21:26:28.237012: W
tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 503316480
exceeds 10% of free system memory.


Epoch 1: val_loss improved from inf to 0.14867, saving model to
model_weights/20221124-212150_cnn_best_weights.hdf5
2538/2538 [==============================] - 278s 107ms/step - loss: 3.1358 -
mse: 3.1358 - rmse: 1.7708 - mae: 0.6604 - mape: 23.3414 - val_loss: 0.1487 -
val_mse: 0.1487 - val_rmse: 0.3856 - val_mae: 0.2858 - val_mape: 8.6893 - lr:
0.0010
Epoch 2/100
2538/2538 [==============================] - ETA: 0s - loss: 0.2386 - mse:
0.2386 - rmse: 0.4884 - mae: 0.3452 - mape: 11.8577
Epoch 2: val_loss did not improve from 0.14867
2538/2538 [==============================] - 261s 103ms/step - loss: 0.2386 -
mse: 0.2386 - rmse: 0.4884 - mae: 0.3452 - mape: 11.8577 - val_loss: 0.7105 -
val_mse: 0.7105 - val_rmse: 0.8429 - val_mae: 0.7684 - val_mape: 24.2654 - lr:
0.0010
Epoch 3/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1895 - mse:
```

```
0.1895 - rmse: 0.4353 - mae: 0.2996 - mape: 10.2114
Epoch 3: val_loss did not improve from 0.14867
2538/2538 [==============================] - 260s 102ms/step - loss: 0.1895 -
mse: 0.1895 - rmse: 0.4353 - mae: 0.2996 - mape: 10.2114 - val_loss: 0.3360 -
val_mse: 0.3360 - val_rmse: 0.5797 - val_mae: 0.5015 - val_mape: 15.4744 - lr:
0.0010
Epoch 4/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1931 - mse:
0.1931 - rmse: 0.4395 - mae: 0.2998 - mape: 10.2581
Epoch 4: val_loss improved from 0.14867 to 0.08927, saving model to
model_weights/20221124-212150_cnn_best_weights.hdf5
2538/2538 [==============================] - 262s 103ms/step - loss: 0.1931 -
mse: 0.1931 - rmse: 0.4395 - mae: 0.2998 - mape: 10.2581 - val_loss: 0.0893 -
val_mse: 0.0893 - val_rmse: 0.2988 - val_mae: 0.2171 - val_mape: 6.7587 - lr:
0.0010
Epoch 5/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1488 - mse:
0.1488 - rmse: 0.3857 - mae: 0.2629 - mape: 8.9841
Epoch 5: val_loss did not improve from 0.08927
2538/2538 [==============================] - 263s 104ms/step - loss: 0.1488 -
mse: 0.1488 - rmse: 0.3857 - mae: 0.2629 - mape: 8.9841 - val_loss: 0.1522 -
val_mse: 0.1522 - val_rmse: 0.3901 - val_mae: 0.3079 - val_mape: 9.8317 - lr:
0.0010
Epoch 6/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1329 - mse:
0.1329 - rmse: 0.3645 - mae: 0.2444 - mape: 8.3617
Epoch 6: val_loss did not improve from 0.08927
2538/2538 [==============================] - 262s 103ms/step - loss: 0.1329 -
mse: 0.1329 - rmse: 0.3645 - mae: 0.2444 - mape: 8.3617 - val_loss: 0.2256 -
val_mse: 0.2256 - val_rmse: 0.4750 - val_mae: 0.3723 - val_mape: 11.6466 - lr:
0.0010
Epoch 7/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1189 - mse:
0.1189 - rmse: 0.3449 - mae: 0.2273 - mape: 7.7810
Epoch 7: val_loss did not improve from 0.08927
2538/2538 [==============================] - 263s 104ms/step - loss: 0.1189 -
mse: 0.1189 - rmse: 0.3449 - mae: 0.2273 - mape: 7.7810 - val_loss: 0.1959 -
val_mse: 0.1959 - val_rmse: 0.4426 - val_mae: 0.3009 - val_mape: 9.2265 - lr:
0.0010
Epoch 8/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1197 - mse:
0.1197 - rmse: 0.3460 - mae: 0.2302 - mape: 7.9117
Epoch 8: val_loss did not improve from 0.08927
2538/2538 [==============================] - 265s 104ms/step - loss: 0.1197 -
mse: 0.1197 - rmse: 0.3460 - mae: 0.2302 - mape: 7.9117 - val_loss: 0.1973 -
val_mse: 0.1973 - val_rmse: 0.4441 - val_mae: 0.3051 - val_mape: 9.2742 - lr:
0.0010
Epoch 9/100
```

```
2538/2538 [==============================] - ETA: 0s - loss: 0.7158 - mse:
0.7158 - rmse: 0.8461 - mae: 0.3721 - mape: 12.1265
Epoch 9: val_loss did not improve from 0.08927
2538/2538 [==============================] - 264s 104ms/step - loss: 0.7158 -
mse: 0.7158 - rmse: 0.8461 - mae: 0.3721 - mape: 12.1265 - val_loss: 0.2933 -
val_mse: 0.2933 - val_rmse: 0.5416 - val_mae: 0.4273 - val_mape: 12.3974 - lr:
0.0010
Epoch 10/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1877 - mse:
0.1877 - rmse: 0.4333 - mae: 0.2713 - mape: 8.7439
Epoch 10: val_loss did not improve from 0.08927
2538/2538 [==============================] - 267s 105ms/step - loss: 0.1877 -
mse: 0.1877 - rmse: 0.4333 - mae: 0.2713 - mape: 8.7439 - val_loss: 0.1482 -
val_mse: 0.1482 - val_rmse: 0.3850 - val_mae: 0.2738 - val_mape: 8.4133 - lr:
1.0000e-04
Epoch 11/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1694 - mse:
0.1694 - rmse: 0.4116 - mae: 0.2556 - mape: 8.2853
Epoch 11: val_loss did not improve from 0.08927
2538/2538 [==============================] - 266s 105ms/step - loss: 0.1694 -
mse: 0.1694 - rmse: 0.4116 - mae: 0.2556 - mape: 8.2853 - val_loss: 0.1605 -
val_mse: 0.1605 - val_rmse: 0.4006 - val_mae: 0.2836 - val_mape: 8.5856 - lr:
1.0000e-04
Epoch 12/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1631 - mse:
0.1631 - rmse: 0.4039 - mae: 0.2495 - mape: 8.0556
Epoch 12: val_loss did not improve from 0.08927
2538/2538 [==============================] - 267s 105ms/step - loss: 0.1631 -
mse: 0.1631 - rmse: 0.4039 - mae: 0.2495 - mape: 8.0556 - val_loss: 0.1635 -
val_mse: 0.1635 - val_rmse: 0.4043 - val_mae: 0.2950 - val_mape: 9.1206 - lr:
1.0000e-04
Epoch 13/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1572 - mse:
0.1572 - rmse: 0.3965 - mae: 0.2448 - mape: 7.9186
Epoch 13: val_loss did not improve from 0.08927
2538/2538 [==============================] - 266s 105ms/step - loss: 0.1572 -
mse: 0.1572 - rmse: 0.3965 - mae: 0.2448 - mape: 7.9186 - val_loss: 0.2036 -
val_mse: 0.2036 - val_rmse: 0.4512 - val_mae: 0.3240 - val_mape: 9.9381 - lr:
1.0000e-04
Epoch 14/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1391 - mse:
0.1391 - rmse: 0.3729 - mae: 0.2317 - mape: 7.5976
Epoch 14: val_loss did not improve from 0.08927
2538/2538 [==============================] - 266s 105ms/step - loss: 0.1391 -
mse: 0.1391 - rmse: 0.3729 - mae: 0.2317 - mape: 7.5976 - val_loss: 0.1761 -
val_mse: 0.1761 - val_rmse: 0.4197 - val_mae: 0.3157 - val_mape: 9.8832 - lr:
1.0000e-04
Epoch 15/100
```

```
2538/2538 [==============================] - ETA: 0s - loss: 0.1208 - mse:
0.1208 - rmse: 0.3476 - mae: 0.2166 - mape: 7.1439
Epoch 15: val_loss did not improve from 0.08927
2538/2538 [==============================] - 271s 107ms/step - loss: 0.1208 -
mse: 0.1208 - rmse: 0.3476 - mae: 0.2166 - mape: 7.1439 - val_loss: 0.1676 -
val_mse: 0.1676 - val_rmse: 0.4093 - val_mae: 0.3035 - val_mape: 9.4201 - lr:
1.0000e-05
Epoch 16/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1204 - mse:
0.1204 - rmse: 0.3470 - mae: 0.2162 - mape: 7.1619
Epoch 16: val_loss did not improve from 0.08927
2538/2538 [==============================] - 269s 106ms/step - loss: 0.1204 -
mse: 0.1204 - rmse: 0.3470 - mae: 0.2162 - mape: 7.1619 - val_loss: 0.1801 -
val_mse: 0.1801 - val_rmse: 0.4244 - val_mae: 0.3100 - val_mape: 9.6213 - lr:
1.0000e-05
Epoch 17/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1172 - mse:
0.1172 - rmse: 0.3423 - mae: 0.2144 - mape: 7.1048
Epoch 17: val_loss did not improve from 0.08927
2538/2538 [==============================] - 265s 104ms/step - loss: 0.1172 -
mse: 0.1172 - rmse: 0.3423 - mae: 0.2144 - mape: 7.1048 - val_loss: 0.1724 -
val_mse: 0.1724 - val_rmse: 0.4152 - val_mae: 0.3039 - val_mape: 9.4628 - lr:
1.0000e-05
Epoch 18/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1164 - mse:
0.1164 - rmse: 0.3411 - mae: 0.2135 - mape: 7.0689
Epoch 18: val_loss did not improve from 0.08927
2538/2538 [==============================] - 265s 104ms/step - loss: 0.1164 -
mse: 0.1164 - rmse: 0.3411 - mae: 0.2135 - mape: 7.0689 - val_loss: 0.1649 -
val_mse: 0.1649 - val_rmse: 0.4061 - val_mae: 0.2997 - val_mape: 9.3374 - lr:
1.0000e-05
Epoch 19/100
2538/2538 [==============================] - ETA: 0s - loss: 0.1156 - mse:
0.1156 - rmse: 0.3400 - mae: 0.2132 - mape: 7.0579
Epoch 19: val_loss did not improve from 0.08927
2538/2538 [==============================] - 266s 105ms/step - loss: 0.1156 -
mse: 0.1156 - rmse: 0.3400 - mae: 0.2132 - mape: 7.0579 - val_loss: 0.1653 -
val_mse: 0.1653 - val_rmse: 0.4066 - val_mae: 0.3037 - val_mape: 9.4544 - lr:
1.0000e-05
Epoch 19: early stopping
```

0.5 Evaluate model

[ ]:

```
print(date_actual)
```

```
20221124-212150
```

```
[ ]: #best_model = models.load_model(f'model_weights/{date_actual}_cnn_best_weights.
      ↪hdf5', custom_objects={"gfb_filter": gfb_filter, "sobel_kernel":␣
      ↪sobel_kernel})

     best_model = models.load_model(f'model_weights/20221124-212150_cnn_best_weights.
      ↪hdf5', custom_objects={"gfb_filter": gfb_filter, "sobel_kernel":␣
      ↪sobel_kernel})

     #best_model = models.load_model(f'model_weights/{date_actual}_cnn_best_weights.
      ↪hdf5', custom_objects={"gfb_filter": gfb_filter, "sobel_kernel":␣
      ↪sobel_kernel})
     #best_model = models.load_model(f'best_models_weights/cnn_best_weights_v9.hdf5')
```

```
2022-11-25 11:54:36.549396: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:54:36.549613: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:54:36.549808: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:54:36.549994: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:54:36.550140: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:980] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2022-11-25 11:54:36.550258: I
tensorflow/core/common_runtime/gpu/gpu_device.cc:1616] Created device
/job:localhost/replica:0/task:0/device:GPU:0 with 4063 MB memory:  -> device: 0,
name: NVIDIA GeForce RTX 2060, pci bus id: 0000:08:00.0, compute capability: 7.5
```

```
[ ]: def evaluate_model(model, test_values, steps):
       score = model.evaluate(test_values, steps=steps)
       return score
```

```
[ ]: test_loss, test_mse, test_rmse, test_mae, test_mape =␣
      ↪evaluate_model(best_model, test_ds, steps=len(test_ds))
```

```
2022-11-25 11:55:43.191337: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
```

```
out of memory trying to allocate 4.15GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
2022-11-25 11:55:43.191387: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.15GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
2022-11-25 11:55:43.388164: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.27GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
2022-11-25 11:55:43.388201: W
tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran
out of memory trying to allocate 4.27GiB with freed_by_count=0. The caller
indicates that this is not a failure, but this may mean that there could be
performance gains if more memory were available.
```

```
12727/12727 [==============================] - 163s 13ms/step - loss: 0.0764 -
mse: 0.0764 - rmse: 0.2765 - mae: 0.1920 - mape: 7.2217
```

[ ]:

```
#predictions = best_model.predict(test_ds, steps=np.ceil(test_size /␣
 ↪BATCH_SIZE))
y_pred = best_model.predict(test_ds, steps=len(test_ds))
```

```
12727/12727 [==============================] - 151s 12ms/step
```

[ ]:

```
y_real_test = []
```

[ ]:

```
for i in range(len(test_ds)):
    image_time, stage_discharge = test_ds[i]
    images = image_time["input_1"]
    stage = stage_discharge

    #print(stage[0])

    y_real_test.append(stage[0])
```

[ ]:

```
y_real_test = np.array(y_real_test)
```

[ ]:

```
from sklearn.metrics import r2_score, mean_absolute_percentage_error,␣
 ↪mean_absolute_error, mean_squared_error
from statsmodels.tools.eval_measures import stde
```

[ ]:

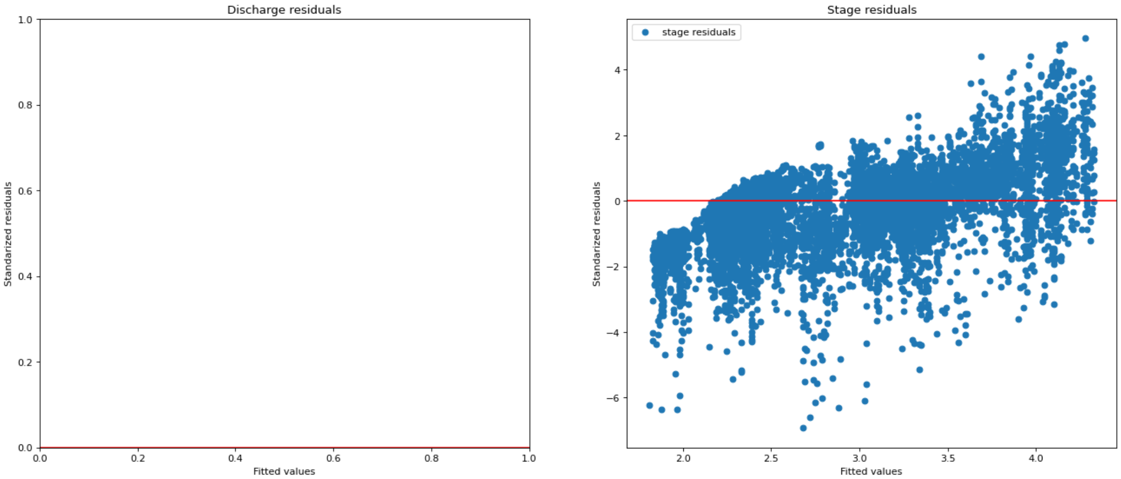
```
print("R^2: ", r2_score(y_real_test, y_pred))
print("mse: ", mean_squared_error(y_real_test, y_pred))
print("rmse: ", mean_squared_error(y_real_test, y_pred, squared=False))
print("mae: ", mean_absolute_error(y_real_test, y_pred))
print("mape: ", mean_absolute_percentage_error(y_real_test, y_pred))
print("Error estandar: ", stde(y_real_test.squeeze(),
      y_pred.squeeze(), ddof=2))
```

```
R^2:  0.8043055230971803
mse:  0.07642631697574952
rmse:  0.2764531008611579
mae:  0.19199198128694336
mape:  0.07221717524291128
Error estandar:  0.2659437455875203
```

0.5.1 Residual analysis

[ ]:

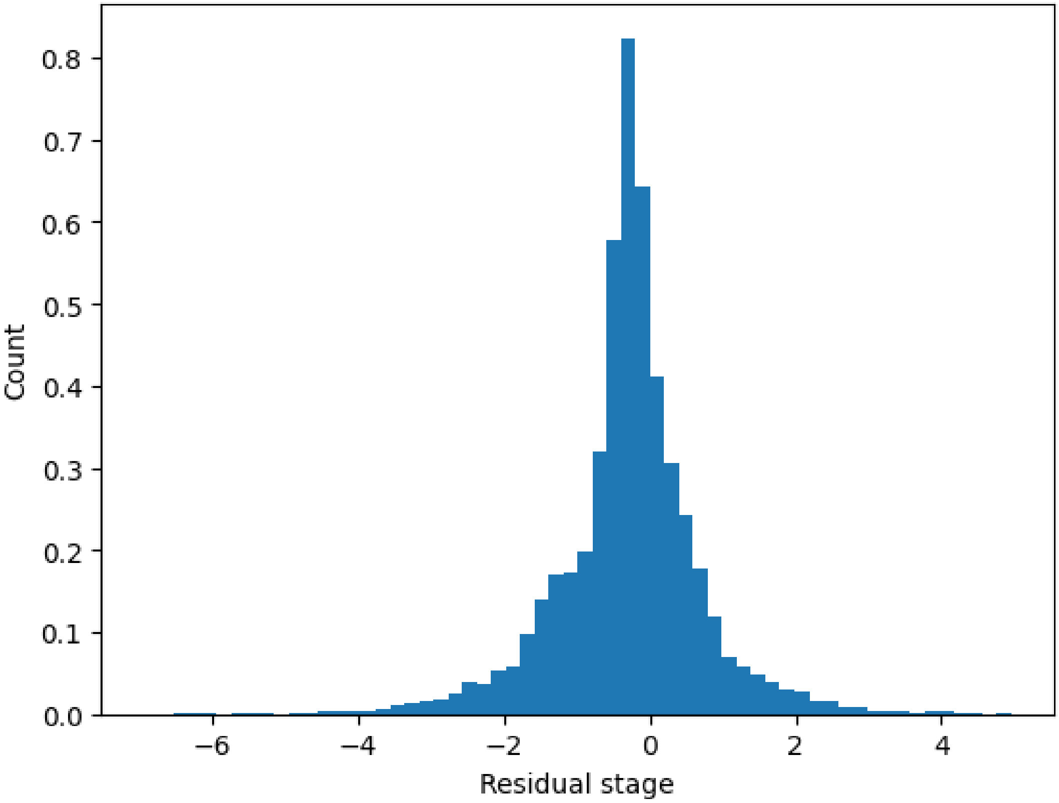
```
residuals = y_real_test - y_pred
residuals_std = residuals/residuals.std()

y_real_stage = y_real_test
residual_stage = residuals

#y_real_discharge = np.array([i[-1] for i in y_test])
#residual_discharge = np.array([i[-1] for i in residuals])


figure, ax = plt.subplots(ncols=2, figsize=(20, 8), dpi=80)

ax[1].scatter(y_real_stage, residual_stage / residual_stage.std(), label="stage␣
 ↪residuals")
#ax[0].scatter(y_real_discharge, residual_discharge / residual_discharge.std(),␣
 ↪label="discharge residuals")
ax[1].axhline(y=0.0, color='r', linestyle='-')
ax[0].axhline(y=0.0, color='r', linestyle='-')

ax[1].set_title("Stage residuals")
ax[0].set_title("Discharge residuals")

ax[1].set_xlabel("Fitted values")
ax[0].set_xlabel("Fitted values")
ax[1].set_ylabel("Standarized residuals")
ax[0].set_ylabel("Standarized residuals")

plt.legend()
plt.show()
```



Check residuals

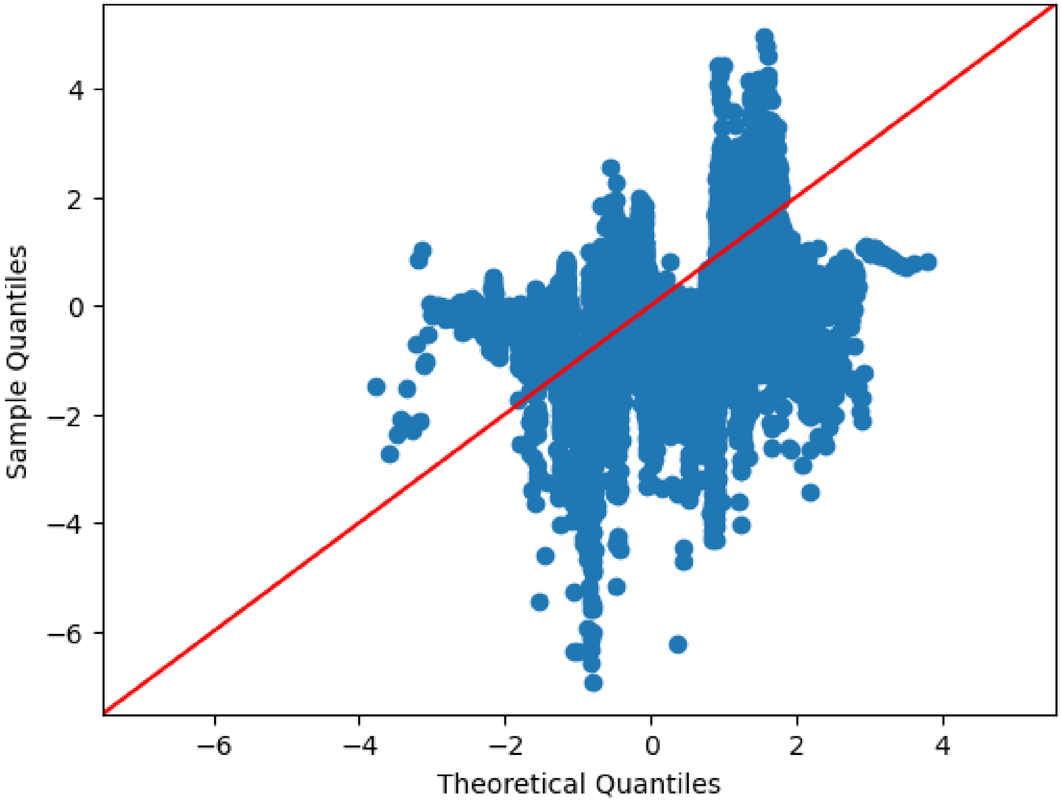
[ ]:

```
plt.hist(residual_stage / residual_stage.std(), density=True, bins = 60)
plt.ylabel('Count')
plt.xlabel('Residual stage');
plt.show()
```



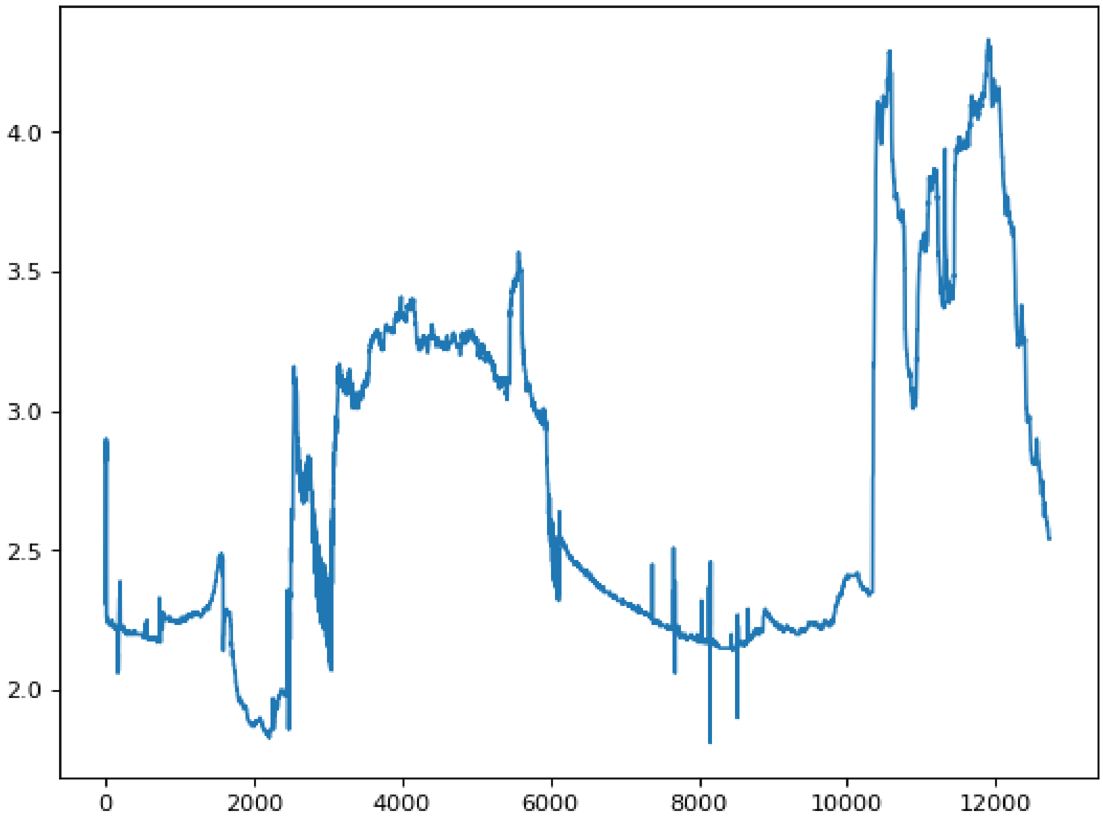
```
[ ]: import statsmodels.api as sm
     from statsmodels.stats.diagnostic import normal_ad
```

```
[ ]: figure = sm.qqplot(residual_stage / residual_stage.std(), line='45',␣
      ↪label='discharge')
     plt.show()
```



```
[ ]: plt.figure(figsize=(8, 6), dpi=80)
     plt.plot(np.arange(len(y_real_test)), y_real_test, label="Stage real")
```

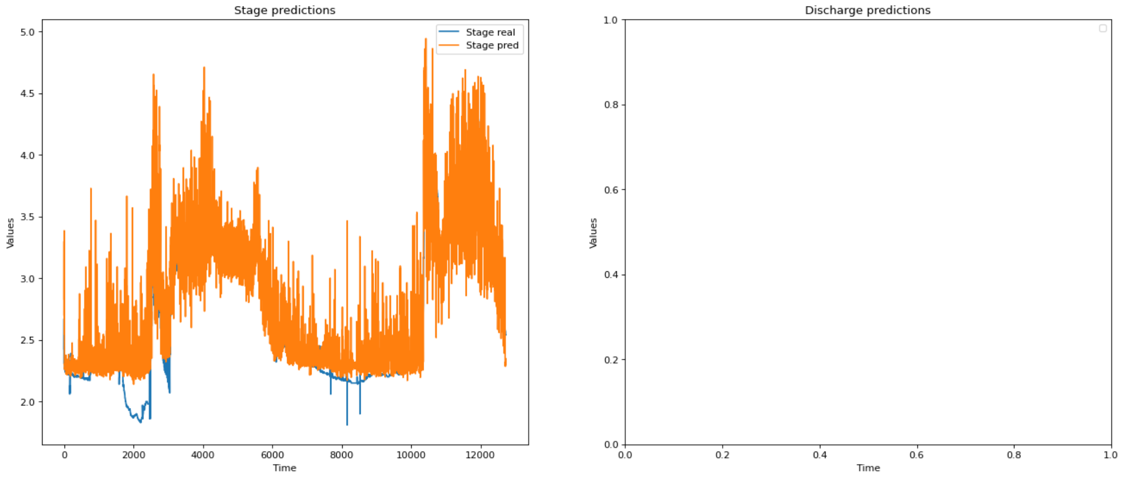
```
[ ]: [<matplotlib.lines.Line2D at 0x7f828c3ad210>]
```



[ ]:

```
figure, ax = plt.subplots(ncols=2, figsize=(20, 8), dpi=80)

ax[0].plot(np.arange(len(y_real_test)), y_real_test, label="Stage real")
ax[0].plot(np.arange(len(y_real_test)), y_pred, label="Stage pred")

ax[0].set_title("Stage predictions")
ax[1].set_title("Discharge predictions")

ax[1].set_ylabel("Values")
ax[0].set_ylabel("Values")
ax[1].set_xlabel("Time")
ax[0].set_xlabel("Time")

ax[0].legend()
ax[1].legend()
plt.show()
```

```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



0.6 Visualize layers

[ ]:

```
layer_outputs = [layer.output for layer in best_model.layers[:12]]
# Extracts the outputs of the top 12 layers
activation_model = models.Model(inputs=best_model.input, outputs=layer_outputs)␣
 ↪# Creates a model that will return these outputs, given the model input
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In [115], line 3
      1 layer_outputs = [layer.output for layer in best_model.layers[:12]]
      2 # Extracts the outputs of the top 12 layers
----> 3 activation_model = models.Model(inputs=best_model.input,␣
 ↪outputs=layer_outputs)

File ~/miniconda3/envs/tf-gpu/lib/python3.10/site-packages/tensorflow/python/
 ↪trackable/base.py:205, in no_automatic_dependency_tracking.<locals>.
 ↪_method_wrapper(self, *args, **kwargs)
    203 self._self_setattr_tracking = False  # pylint: disable=protected-access
    204 try:
--> 205   result = method(self, *args, **kwargs)
    206 finally:
    207   self._self_setattr_tracking = previous_value  # pylint:␣
 ↪disable=protected-access

File ~/miniconda3/envs/tf-gpu/lib/python3.10/site-packages/keras/engine/
 ↪functional.py:165, in Functional.__init__(self, inputs, outputs, name,␣
 ↪trainable, **kwargs)
    156     if not all(
    157         [
    158             functional_utils.is_input_keras_tensor(t)
    159             for t in tf.nest.flatten(inputs)
    160         ]
    161     ):
    162         inputs, outputs = functional_utils.clone_graph_nodes(
    163             inputs, outputs
    164         )
--> 165 self._init_graph_network(inputs, outputs)

File ~/miniconda3/envs/tf-gpu/lib/python3.10/site-packages/tensorflow/python/
 ↪trackable/base.py:205, in no_automatic_dependency_tracking.<locals>.
 ↪_method_wrapper(self, *args, **kwargs)
    203 self._self_setattr_tracking = False  # pylint: disable=protected-access
    204 try:
--> 205   result = method(self, *args, **kwargs)
    206 finally:
    207   self._self_setattr_tracking = previous_value  # pylint:␣
 ↪disable=protected-access

File ~/miniconda3/envs/tf-gpu/lib/python3.10/site-packages/keras/engine/
 ↪functional.py:264, in Functional._init_graph_network(self, inputs, outputs)
    261     self._input_coordinates.append((layer, node_index, tensor_index))
    263 # Keep track of the network's nodes and layers.
--> 264 nodes, nodes_by_depth, layers, _ = _map_graph_network(
    265     self.inputs, self.outputs
    266 )
    267 self._network_nodes = nodes
    268 self._nodes_by_depth = nodes_by_depth

File ~/miniconda3/envs/tf-gpu/lib/python3.10/site-packages/keras/engine/
 ↪functional.py:1128, in _map_graph_network(inputs, outputs)
   1126 for x in tf.nest.flatten(node.keras_inputs):
   1127     if id(x) not in computable_tensors:
-> 1128         raise ValueError(
   1129             f"Graph disconnected: cannot obtain value for "
   1130             f'tensor {x} at layer "{layer.name}". '
   1131             "The following previous layers were accessed "
   1132             f"without issue: {layers_with_complete_input}"
   1133         )
   1134 for x in tf.nest.flatten(node.outputs):
   1135     computable_tensors.add(id(x))

ValueError: Graph disconnected: cannot obtain value for tensor␣
 ↪KerasTensor(type_spec=TensorSpec(shape=(None, 320, 320, 3), dtype=tf.float32,␣
 ↪name='input_1'), name='input_1', description="created by layer 'input_1'") at␣
 ↪layer "conv1_pad". The following previous layers were accessed without issue:␣
 ↪[]
```

[ ]:
```
activations = activation_model.predict(test_ds.take(1))
```

```
1/1 [==============================] - 0s 369ms/step
```

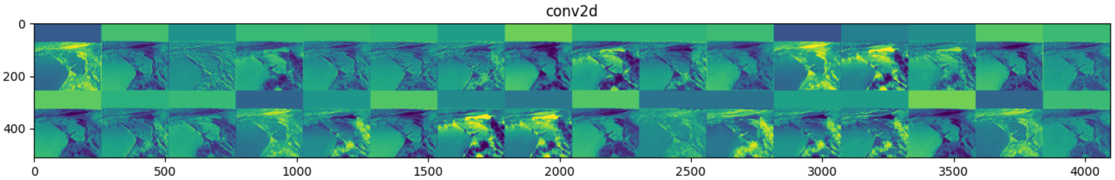
[ ]:

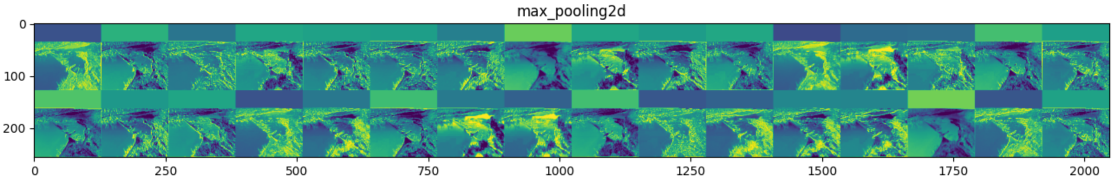
```
import matplotlib.pyplot as plt

layer_names = []
for layer in best_model.layers[:12]:
    layer_names.append(layer.name) # Names of the layers, so you can have them␣
 ↪as part of your plot

images_per_row = 16

for layer_name, layer_activation in zip(layer_names, activations): # Displays␣
 ↪the feature maps
    n_features = layer_activation.shape[-1] # Number of features in the feature␣
 ↪map
    size = layer_activation.shape[1] #The feature map has shape (1, size, size,␣
 ↪n_features).
    n_cols = n_features // images_per_row # Tiles the activation channels in␣
 ↪this matrix
    display_grid = np.zeros((size * n_cols, images_per_row * size))

    print(layer_name)
    if "flatten" in layer_name or "dense" in layer_name: break

    for col in range(n_cols): # Tiles each filter into a big horizontal grid
        for row in range(images_per_row):
            channel_image = layer_activation[0,
                                             :, :,
                                             col * images_per_row + row]
            channel_image -= channel_image.mean() # Post-processes the feature␣
 ↪to make it visually palatable
            channel_image /= channel_image.std()
            channel_image *= 64
            channel_image += 128
            channel_image = np.clip(channel_image, 0, 255).astype('uint8')
            display_grid[col * size : (col + 1) * size, # Displays the grid
                         row * size : (row + 1) * size] = channel_image
    scale = 1. / size
    plt.figure(figsize=(scale * display_grid.shape[1],
                        scale * display_grid.shape[0]))
    plt.title(layer_name)
    plt.grid(False)
    plt.imshow(display_grid, aspect='auto', cmap='viridis')
```

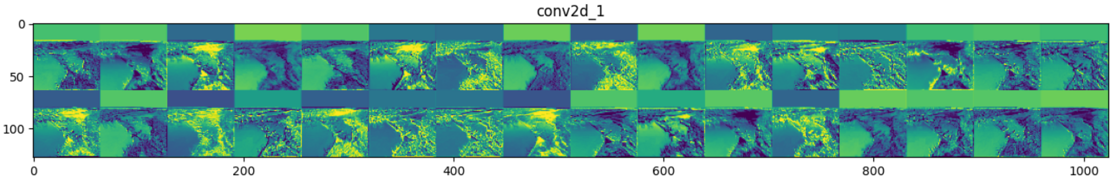
```
conv2d
max_pooling2d
conv2d_1
```

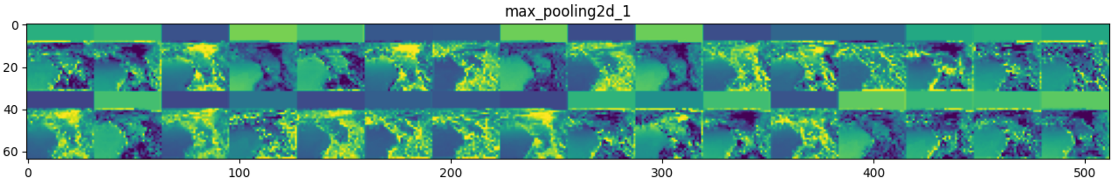
```
max_pooling2d_1
conv2d_2
conv2d_3
conv2d_4
max_pooling2d_2
conv2d_5
conv2d_6
max_pooling2d_3
```

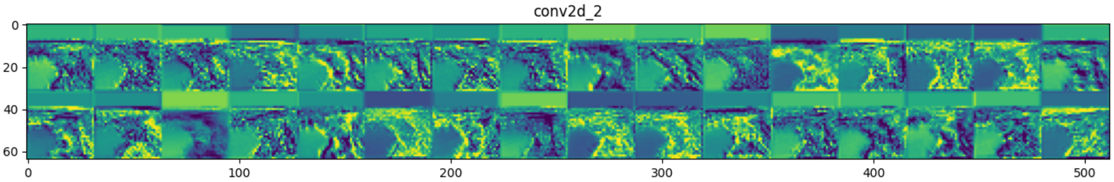
```
---------------------------------------------------------------------------
MemoryError                               Traceback (most recent call last)
Cell In [50], line 13
     11 size = layer_activation.shape[1] #The feature map has shape (1, size,␣
 ↪size, n_features).
     12 n_cols = n_features // images_per_row # Tiles the activation channels i
 ↪this matrix
---> 13 display_grid = np.zeros((size * n_cols, images_per_row * size))
     15 print(layer_name)
     16 if "flatten" in layer_name or "dense" in layer_name: break

MemoryError: Unable to allocate 91.1 GiB for an array with shape (331776, 36864
 ↪and data type float64
```

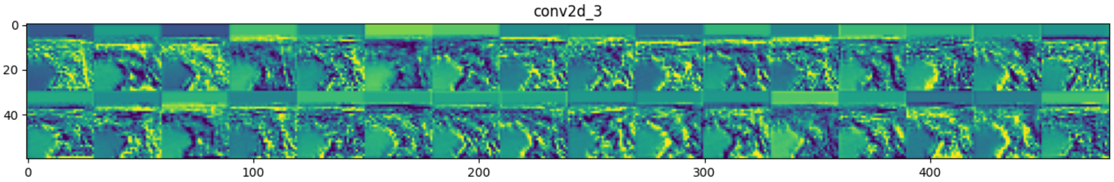


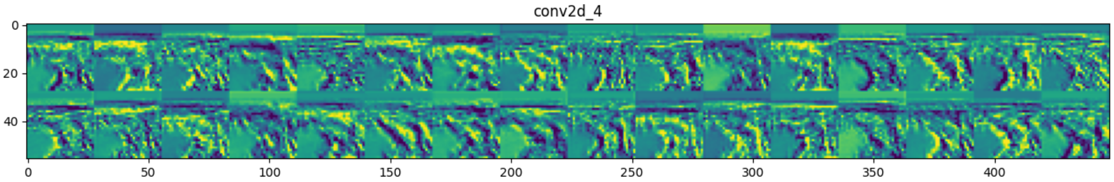


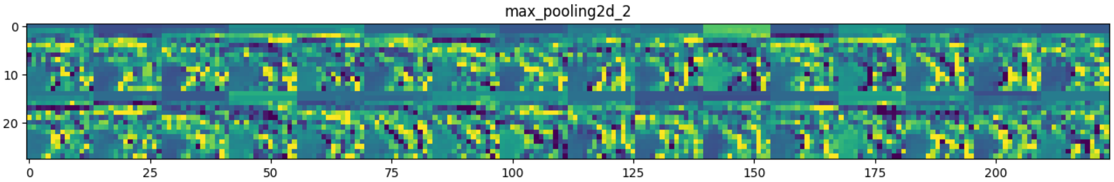


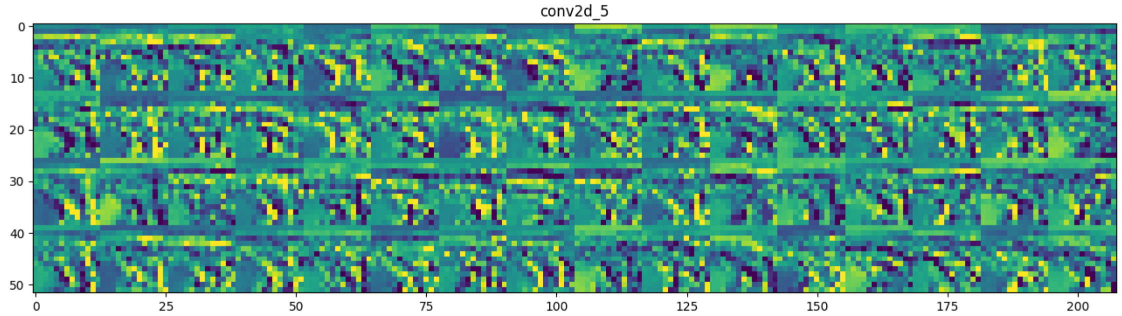

max_pooling2d_1
0
20
40
60
0
100
200
300
400
500

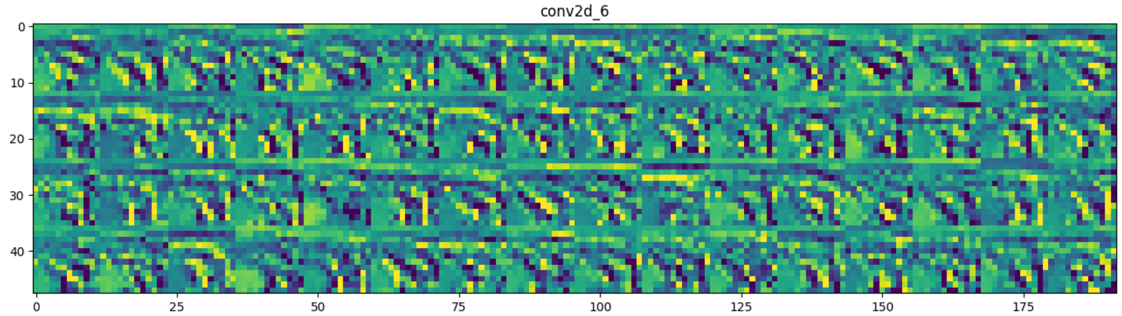

conv2d_2
0
20
40
60
0
100
200
300
400
500

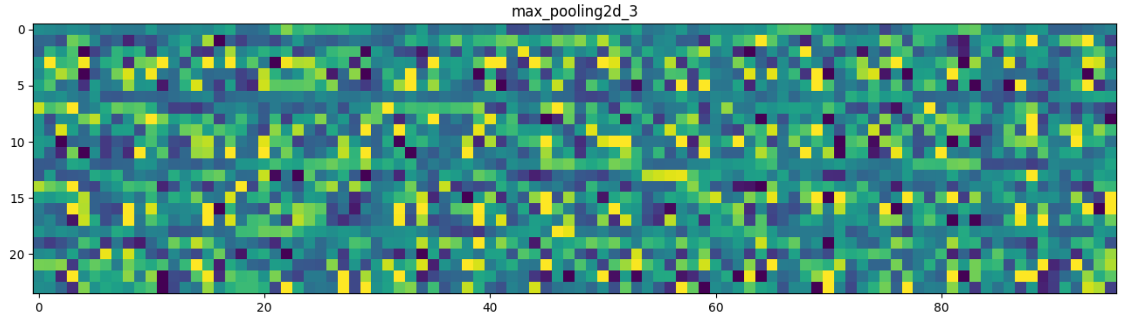

conv2d_3
0
20
40
0
100
200
300
400


conv2d_4
0
20
40
0
50
100
150
200
250
300
350
400


max_pooling2d_2
0
10
20
0
25
50
75
100
125
150
175
200


conv2d_5
0
10
20
30
40
50
0
25
50
75
100
125
150
175
200


conv2d_6
0
10
20
30
40
0
25
50
75
100
125
150
175


max_pooling2d_3
0
5
10
15
20
0
20
40
60
80