cnn_v7

October 14, 2022

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

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

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

```
:/home/nkspartan/miniconda3/envs/tf-gpu/lib/:/home/nkspartan/miniconda3/envs/tf-gpu/lib/
```

[ ]:
```
import tensorflow as tf
import numpy as np
import pandas as pd
import os
import keras

from keras import Sequential, models, Input
from keras.layers import Dense, Flatten, Conv2D, MaxPooling2D, Dropout,␣
↪LeakyReLU, AveragePooling2D
from keras.optimizers import SGD, Adam
```

[ ]:
```
from tensorflow.python.client import device_lib

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

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

```
2022-10-14 11:39:11.186822: 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.
2022-10-14 11:39:11.209616: 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
2022-10-14 11:39:11.261664: 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
2022-10-14 11:39:11.261850: 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
2022-10-14 11:39:12.100458: 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
2022-10-14 11:39:12.101067: 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
2022-10-14 11:39:12.101224: 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
2022-10-14 11:39:12.101367: I
tensorflow/core/common_runtime/gpu/gpu_device.cc:1616] Created device
/device:GPU:0 with 4023 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"]]
```

0.1.1 Scale the data

[ ]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
```

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

[ ]:
```
                                    Filename     Stage  Discharge
0      StateLineWeir_20120609_Farrell_001.jpg  0.138117  -0.046094
1      StateLineWeir_20120609_Farrell_002.jpg  0.138117  -0.046094
2      StateLineWeir_20120609_Farrell_003.jpg  0.100875  -0.082160
3      StateLineWeir_20120609_Farrell_004.jpg  0.076046  -0.104807
4      StateLineWeir_20120609_Farrell_005.jpg  0.076046  -0.104807
…                                        …         …         …
42054  StateLineWeir_20191011_Farrell_409.jpg -0.420526  -0.450369
42055  StateLineWeir_20191011_Farrell_410.jpg -0.420526  -0.450369
42056  StateLineWeir_20191011_Farrell_411.jpg -0.420526  -0.450369
42057  StateLineWeir_20191011_Farrell_412.jpg -0.420526  -0.450369
42058  StateLineWeir_20191011_Farrell_413.jpg -0.420526  -0.450369

[42059 rows x 3 columns]
```

[ ]:
```
from joblib import dump, load
dump(scaler, 'std_scaler.joblib', compress=True)
```

[ ]:
```
['std_scaler.joblib']
```

0.2 Create the dataset pipeline

[ ]:
```
IMG_SIZE = 512
BATCH_SIZE = 32
```

[ ]:
```
from glob import glob

def make_dataset(path, batch_size, df, seed=None):
  np.random.seed(seed)

  def parse_image(filename):
    image = tf.io.read_file(filename)
    image = tf.image.decode_jpeg(image, channels=3)
    #image = tf.image.resize(image, [IMG_SIZE, IMG_SIZE])
    image = tf.cast(image, tf.float32)
    image /= 255
    return image

  def configure_for_performance(ds):
    ds = ds.shuffle(buffer_size=100)
    ds = ds.batch(batch_size)
    ds = ds.repeat()
    ds = ds.prefetch(buffer_size=tf.data.experimental.AUTOTUNE)
    return ds

  filenames = glob(path + '/*')

  # make train, val and test splits of the dataset (70%, 10%, 20% split)
  split1 = int(0.7 * len(filenames))
  split2 = int(0.8 * len(filenames))

  np.random.shuffle(filenames)
  train_files = filenames[:split1] # up to split 1 (ex 70%)
  val_files = filenames[split1:split2] # from ex. 70% to 80%
  test_files = filenames[split2:] # from ex. 80% until the end

  # create stage values
  stage_train_values = [df[df.Filename == file.split('/')[-1]].Stage.values for␣
↪file in train_files]
  stage_val_values = [df[df.Filename == file.split('/')[-1]].Stage.values for␣
↪file in val_files]
```

```
 stage_test_values = [df[df.Filename == file.split('/')[-1]].Stage.values for␣
↪file in test_files]

 # create discharge values
 discharge_train_values = [df[df.Filename == file.split(
     '/')[-1]].Discharge.values for file in train_files]
 discharge_val_values = [df[df.Filename == file.split(
     '/')[-1]].Discharge.values for file in val_files]
 discharge_test_values = [df[df.Filename == file.split(
     '/')[-1]].Discharge.values for file in test_files]

 # join stage and discharge values
 stage_discharge_train_values = [[np.squeeze(s), np.squeeze(d)] for s, d in␣
↪zip(stage_train_values, discharge_train_values)]
 stage_discharge_val_values = [[np.squeeze(s), np.squeeze(d)] for s, d in␣
↪zip(stage_val_values, discharge_val_values)]
 stage_discharge_test_values = [[np.squeeze(s), np.squeeze(
     d)] for s, d in zip(stage_test_values, discharge_test_values)]

 # create images dataset (train, val, test)
 filenames_train_ds = tf.data.Dataset.from_tensor_slices(train_files)
 filenames_val_ds = tf.data.Dataset.from_tensor_slices(val_files)
 filenames_test_ds = tf.data.Dataset.from_tensor_slices(test_files)

 images_train_ds = filenames_train_ds.map(parse_image, num_parallel_calls=5)
 images_val_ds = filenames_val_ds.map(parse_image, num_parallel_calls=5)
 images_test_ds = filenames_test_ds.map(parse_image, num_parallel_calls=5)

 # create stage and discharge dataset (train, val, test)
 stage_discharge_train_ds = tf.data.Dataset.
↪from_tensor_slices(stage_discharge_train_values)
 stage_discharge_val_ds = tf.data.Dataset.
↪from_tensor_slices(stage_discharge_val_values)
 stage_discharge_test_ds = tf.data.Dataset.from_tensor_slices(
     stage_discharge_test_values)

 # create tensorflow dataset of images and values (train, val, test)
 train_ds = tf.data.Dataset.zip((images_train_ds, stage_discharge_train_ds))
 train_ds = configure_for_performance(train_ds)
 val_ds = tf.data.Dataset.zip((images_val_ds, stage_discharge_val_ds))
 val_ds = configure_for_performance(val_ds)
 test_ds = tf.data.Dataset.zip((images_test_ds, stage_discharge_test_ds))
 test_ds = configure_for_performance(test_ds)

 return train_ds, len(train_files), val_ds, len(val_files), test_ds,␣
↪len(test_files)
```

[ ]:
```
path = "../../dataset/images"

train_ds, train_size, val_ds, val_size, test_ds, test_size = make_dataset(path,␣
 ↪BATCH_SIZE, df, 0)
```

```
2022-10-14 11:47:25.673029: 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
2022-10-14 11:47:25.673317: 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
2022-10-14 11:47:25.673566: 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
2022-10-14 11:47:25.674117: 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
2022-10-14 11:47:25.674268: 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
2022-10-14 11:47:25.674407: 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
2022-10-14 11:47:25.674587: 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
2022-10-14 11:47:25.674729: 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
2022-10-14 11:47:25.674844: I
tensorflow/core/common_runtime/gpu/gpu_device.cc:1616] Created device
/job:localhost/replica:0/task:0/device:GPU:0 with 4023 MB memory:  -> device: 0,
name: NVIDIA GeForce RTX 2060, pci bus id: 0000:08:00.0, compute capability: 7.5
```

[ ]:
```
input_shape = 0
output_shape = 0

for image, stage_discharge in train_ds.take(1):
    print(image.numpy().shape)
```

```
    print(stage_discharge.numpy().shape)

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

```
(32, 512, 512, 3)
(32, 2)
```

[ ]:
```
print(input_shape)
print(output_shape)
```

```
(512, 512, 3)
(2,)
```

0.3 Create model

[ ]:
```
def create_model(input_shape, output_shape):
    model = Sequential()

    model.add(Input(shape=input_shape))

    model.add(Conv2D(64, kernel_size=(4, 4), strides=(2, 2), padding='same',␣
↪activation=LeakyReLU()))
    model.add(MaxPooling2D(pool_size=(4, 4)))

    model.add(Conv2D(64, kernel_size=(4, 4), activation=LeakyReLU(),␣
↪padding='same'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', padding='same'))
    model.add(AveragePooling2D(pool_size=(2, 2)))

    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
    model.add(AveragePooling2D(pool_size=(2, 2)))



    model.add(Flatten())
    model.add(Dense(64, activation='relu'))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(output_shape, activation='linear')) # linear regression␣
↪output layer

    return model
```

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

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

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 256, 256, 64)      3136

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

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

 max_pooling2d_1 (MaxPooling  (None, 32, 32, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 32, 32, 32)        18464

 average_pooling2d (AverageP  (None, 16, 16, 32)       0
 ooling2D)

 conv2d_3 (Conv2D)           (None, 14, 14, 32)        9248

 average_pooling2d_1 (Averag  (None, 7, 7, 32)         0
 ePooling2D)

 flatten (Flatten)           (None, 1568)              0

 dense (Dense)               (None, 64)                100416

 dense_1 (Dense)             (None, 64)                4160

 dense_2 (Dense)             (None, 32)                2080

 dense_3 (Dense)             (None, 32)                1056

 dense_4 (Dense)             (None, 2)                 66

=================================================================
Total params: 204,226
Trainable params: 204,226
Non-trainable params: 0
_________________________________________________________________
```

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

[ ]:
```
sgd = SGD(learning_rate=0.01, decay=1e-4, momentum=0.9, nesterov=True)
adam = Adam(learning_rate=1e-3, decay=1e-3 / 100)

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)

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

[ ]:
```
# batch_size = 0 because we already have batch size in tf dataset
history = fit_model(train_ds, val_ds, epochs=25, steps=np.ceil(train_size /␣
 ↪BATCH_SIZE), val_steps=np.ceil(val_size / BATCH_SIZE),␣
 ↪callbacks=[tensorboard_callback, checkpoint_callback])
```

```
Epoch 1/25
```

```
2022-10-14 11:47:51.620111: I tensorflow/stream_executor/cuda/cuda_dnn.cc:384]
Loaded cuDNN version 8100
2022-10-14 11:47:52.860907: I
tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child
process: No such file or directory
2022-10-14 11:47:52.861741: I
tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child
process: No such file or directory
2022-10-14 11:47:52.861762: W tensorflow/stream_executor/gpu/asm_compiler.cc:80]
Couldn't get ptxas version string: INTERNAL: Couldn't invoke ptxas --version
2022-10-14 11:47:52.862848: I
tensorflow/core/platform/default/subprocess.cc:304] Start cannot spawn child
process: No such file or directory
2022-10-14 11:47:52.862909: 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.
921/921 [==============================] - ETA: 0s - loss: 0.1460 - mse: 0.1460
- rmse: 0.3822 - mae: 0.2167 - mape: 86.9484
Epoch 1: val_mse improved from inf to 0.03423, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 105s 109ms/step - loss: 0.1460 - mse:
0.1460 - rmse: 0.3822 - mae: 0.2167 - mape: 86.9484 - val_loss: 0.0342 -
val_mse: 0.0342 - val_rmse: 0.1850 - val_mae: 0.1270 - val_mape: 40.8599
Epoch 2/25
920/921 [============================>.] - ETA: 0s - loss: 0.0285 - mse: 0.0285
- rmse: 0.1687 - mae: 0.1139 - mape: 52.6991
Epoch 2: val_mse improved from 0.03423 to 0.03168, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 110s 119ms/step - loss: 0.0285 - mse:
0.0285 - rmse: 0.1687 - mae: 0.1139 - mape: 52.6979 - val_loss: 0.0317 -
val_mse: 0.0317 - val_rmse: 0.1780 - val_mae: 0.1230 - val_mape: 33.6909
Epoch 3/25
920/921 [============================>.] - ETA: 0s - loss: 0.0159 - mse: 0.0159
- rmse: 0.1260 - mae: 0.0856 - mape: 34.2474
Epoch 3: val_mse improved from 0.03168 to 0.02314, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 111s 120ms/step - loss: 0.0159 - mse:
0.0159 - rmse: 0.1260 - mae: 0.0856 - mape: 34.2464 - val_loss: 0.0231 -
val_mse: 0.0231 - val_rmse: 0.1521 - val_mae: 0.0996 - val_mape: 26.9171
Epoch 4/25
920/921 [============================>.] - ETA: 0s - loss: 0.0125 - mse: 0.0125
- rmse: 0.1119 - mae: 0.0751 - mape: 38.1384
Epoch 4: val_mse improved from 0.02314 to 0.01050, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 110s 120ms/step - loss: 0.0125 - mse:
0.0125 - rmse: 0.1119 - mae: 0.0751 - mape: 38.1372 - val_loss: 0.0105 -
val_mse: 0.0105 - val_rmse: 0.1025 - val_mae: 0.0684 - val_mape: 22.3800
Epoch 5/25
920/921 [============================>.] - ETA: 0s - loss: 0.0109 - mse: 0.0109
- rmse: 0.1043 - mae: 0.0695 - mape: 29.6158
Epoch 5: val_mse did not improve from 0.01050
921/921 [==============================] - 109s 118ms/step - loss: 0.0109 - mse:
0.0109 - rmse: 0.1043 - mae: 0.0695 - mape: 29.6150 - val_loss: 0.0119 -
val_mse: 0.0119 - val_rmse: 0.1089 - val_mae: 0.0718 - val_mape: 21.1961
Epoch 6/25
920/921 [============================>.] - ETA: 0s - loss: 0.0087 - mse: 0.0087
- rmse: 0.0932 - mae: 0.0615 - mape: 27.1488
Epoch 6: val_mse did not improve from 0.01050
```

```
921/921 [==============================] - 109s 118ms/step - loss: 0.0087 - mse:
0.0087 - rmse: 0.0932 - mae: 0.0615 - mape: 27.1481 - val_loss: 0.0141 -
val_mse: 0.0141 - val_rmse: 0.1189 - val_mae: 0.0727 - val_mape: 17.5537
Epoch 7/25
920/921 [============================>.] - ETA: 0s - loss: 0.0076 - mse: 0.0076
- rmse: 0.0874 - mae: 0.0567 - mape: 27.3970
Epoch 7: val_mse improved from 0.01050 to 0.00580, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 108s 118ms/step - loss: 0.0076 - mse:
0.0076 - rmse: 0.0874 - mae: 0.0567 - mape: 27.3963 - val_loss: 0.0058 -
val_mse: 0.0058 - val_rmse: 0.0762 - val_mae: 0.0499 - val_mape: 14.4002
Epoch 8/25
920/921 [============================>.] - ETA: 0s - loss: 0.0071 - mse: 0.0071
- rmse: 0.0841 - mae: 0.0540 - mape: 28.0689
Epoch 8: val_mse did not improve from 0.00580
921/921 [==============================] - 109s 119ms/step - loss: 0.0071 - mse:
0.0071 - rmse: 0.0841 - mae: 0.0540 - mape: 28.0681 - val_loss: 0.0149 -
val_mse: 0.0149 - val_rmse: 0.1221 - val_mae: 0.0708 - val_mape: 18.1489
Epoch 9/25
920/921 [============================>.] - ETA: 0s - loss: 0.0068 - mse: 0.0068
- rmse: 0.0824 - mae: 0.0523 - mape: 24.4000
Epoch 9: val_mse improved from 0.00580 to 0.00412, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 108s 117ms/step - loss: 0.0068 - mse:
0.0068 - rmse: 0.0824 - mae: 0.0523 - mape: 24.3992 - val_loss: 0.0041 -
val_mse: 0.0041 - val_rmse: 0.0642 - val_mae: 0.0425 - val_mape: 15.7825
Epoch 10/25
920/921 [============================>.] - ETA: 0s - loss: 0.0058 - mse: 0.0058
- rmse: 0.0763 - mae: 0.0491 - mape: 22.7052
Epoch 10: val_mse did not improve from 0.00412
921/921 [==============================] - 108s 117ms/step - loss: 0.0058 - mse:
0.0058 - rmse: 0.0763 - mae: 0.0491 - mape: 22.7046 - val_loss: 0.0059 -
val_mse: 0.0059 - val_rmse: 0.0771 - val_mae: 0.0525 - val_mape: 18.6875
Epoch 11/25
920/921 [============================>.] - ETA: 0s - loss: 0.0052 - mse: 0.0052
- rmse: 0.0721 - mae: 0.0467 - mape: 22.0623
Epoch 11: val_mse did not improve from 0.00412
921/921 [==============================] - 108s 117ms/step - loss: 0.0052 - mse:
0.0052 - rmse: 0.0721 - mae: 0.0467 - mape: 22.0616 - val_loss: 0.0053 -
val_mse: 0.0053 - val_rmse: 0.0726 - val_mae: 0.0452 - val_mape: 13.8431
Epoch 12/25
920/921 [============================>.] - ETA: 0s - loss: 0.0046 - mse: 0.0046
- rmse: 0.0680 - mae: 0.0432 - mape: 21.0112
Epoch 12: val_mse did not improve from 0.00412
921/921 [==============================] - 106s 115ms/step - loss: 0.0046 - mse:
0.0046 - rmse: 0.0680 - mae: 0.0432 - mape: 21.0106 - val_loss: 0.0049 -
val_mse: 0.0049 - val_rmse: 0.0698 - val_mae: 0.0453 - val_mape: 14.0386
Epoch 13/25
```

```
920/921 [============================>.] - ETA: 0s - loss: 0.0043 - mse: 0.0043
- rmse: 0.0653 - mae: 0.0419 - mape: 20.2350
Epoch 13: val_mse did not improve from 0.00412
921/921 [==============================] - 100s 109ms/step - loss: 0.0043 - mse:
0.0043 - rmse: 0.0653 - mae: 0.0419 - mape: 20.2343 - val_loss: 0.0042 -
val_mse: 0.0042 - val_rmse: 0.0646 - val_mae: 0.0393 - val_mape: 12.5412
Epoch 14/25
920/921 [============================>.] - ETA: 0s - loss: 0.0051 - mse: 0.0051
- rmse: 0.0712 - mae: 0.0437 - mape: 18.7385
Epoch 14: val_mse did not improve from 0.00412
921/921 [==============================] - 107s 116ms/step - loss: 0.0051 - mse:
0.0051 - rmse: 0.0712 - mae: 0.0437 - mape: 18.7380 - val_loss: 0.0062 -
val_mse: 0.0062 - val_rmse: 0.0789 - val_mae: 0.0558 - val_mape: 18.8260
Epoch 15/25
920/921 [============================>.] - ETA: 0s - loss: 0.0038 - mse: 0.0038
- rmse: 0.0613 - mae: 0.0393 - mape: 22.1371
Epoch 15: val_mse improved from 0.00412 to 0.00397, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 108s 117ms/step - loss: 0.0038 - mse:
0.0038 - rmse: 0.0613 - mae: 0.0393 - mape: 22.1364 - val_loss: 0.0040 -
val_mse: 0.0040 - val_rmse: 0.0630 - val_mae: 0.0400 - val_mape: 15.2230
Epoch 16/25
920/921 [============================>.] - ETA: 0s - loss: 0.0031 - mse: 0.0031
- rmse: 0.0556 - mae: 0.0357 - mape: 18.3191
Epoch 16: val_mse improved from 0.00397 to 0.00314, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 107s 116ms/step - loss: 0.0031 - mse:
0.0031 - rmse: 0.0556 - mae: 0.0357 - mape: 18.3187 - val_loss: 0.0031 -
val_mse: 0.0031 - val_rmse: 0.0560 - val_mae: 0.0375 - val_mape: 11.7202
Epoch 17/25
920/921 [============================>.] - ETA: 0s - loss: 0.0036 - mse: 0.0036
- rmse: 0.0600 - mae: 0.0374 - mape: 16.9154
Epoch 17: val_mse improved from 0.00314 to 0.00313, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 107s 117ms/step - loss: 0.0036 - mse:
0.0036 - rmse: 0.0600 - mae: 0.0374 - mape: 16.9149 - val_loss: 0.0031 -
val_mse: 0.0031 - val_rmse: 0.0559 - val_mae: 0.0357 - val_mape: 13.3308
Epoch 18/25
920/921 [============================>.] - ETA: 0s - loss: 0.0028 - mse: 0.0028
- rmse: 0.0526 - mae: 0.0338 - mape: 16.8378
Epoch 18: val_mse improved from 0.00313 to 0.00271, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 107s 116ms/step - loss: 0.0028 - mse:
0.0028 - rmse: 0.0526 - mae: 0.0338 - mape: 16.8374 - val_loss: 0.0027 -
val_mse: 0.0027 - val_rmse: 0.0520 - val_mae: 0.0339 - val_mape: 12.4455
Epoch 19/25
920/921 [============================>.] - ETA: 0s - loss: 0.0028 - mse: 0.0028
- rmse: 0.0531 - mae: 0.0337 - mape: 17.3883
```

```
Epoch 19: val_mse did not improve from 0.00271
921/921 [==============================] - 105s 115ms/step - loss: 0.0028 - mse:
0.0028 - rmse: 0.0531 - mae: 0.0337 - mape: 17.3879 - val_loss: 0.0044 -
val_mse: 0.0044 - val_rmse: 0.0665 - val_mae: 0.0454 - val_mape: 16.2450
Epoch 20/25
920/921 [============================>.] - ETA: 0s - loss: 0.0026 - mse: 0.0026
- rmse: 0.0506 - mae: 0.0323 - mape: 19.3537
Epoch 20: val_mse did not improve from 0.00271
921/921 [==============================] - 110s 119ms/step - loss: 0.0026 - mse:
0.0026 - rmse: 0.0506 - mae: 0.0323 - mape: 19.3531 - val_loss: 0.0036 -
val_mse: 0.0036 - val_rmse: 0.0599 - val_mae: 0.0378 - val_mape: 13.7792
Epoch 21/25
920/921 [============================>.] - ETA: 0s - loss: 0.0028 - mse: 0.0028
- rmse: 0.0532 - mae: 0.0337 - mape: 15.7936
Epoch 21: val_mse did not improve from 0.00271
921/921 [==============================] - 109s 118ms/step - loss: 0.0028 - mse:
0.0028 - rmse: 0.0532 - mae: 0.0337 - mape: 15.7930 - val_loss: 0.0028 -
val_mse: 0.0028 - val_rmse: 0.0532 - val_mae: 0.0334 - val_mape: 10.8679
Epoch 22/25
920/921 [============================>.] - ETA: 0s - loss: 0.0026 - mse: 0.0026
- rmse: 0.0510 - mae: 0.0317 - mape: 16.7444
Epoch 22: val_mse improved from 0.00271 to 0.00269, saving model to
model_weights/20221014-114748_cnn_best_weights.hdf5
921/921 [==============================] - 110s 119ms/step - loss: 0.0026 - mse:
0.0026 - rmse: 0.0510 - mae: 0.0317 - mape: 16.7439 - val_loss: 0.0027 -
val_mse: 0.0027 - val_rmse: 0.0519 - val_mae: 0.0333 - val_mape: 11.0753
Epoch 23/25
920/921 [============================>.] - ETA: 0s - loss: 0.0024 - mse: 0.0024
- rmse: 0.0490 - mae: 0.0309 - mape: 15.0125
Epoch 23: val_mse did not improve from 0.00269
921/921 [==============================] - 104s 113ms/step - loss: 0.0024 - mse:
0.0024 - rmse: 0.0490 - mae: 0.0309 - mape: 15.0121 - val_loss: 0.0074 -
val_mse: 0.0074 - val_rmse: 0.0862 - val_mae: 0.0431 - val_mape: 10.6888
Epoch 24/25
920/921 [============================>.] - ETA: 0s - loss: 0.0094 - mse: 0.0094
- rmse: 0.0970 - mae: 0.0517 - mape: 24.8800
Epoch 24: val_mse did not improve from 0.00269
921/921 [==============================] - 103s 112ms/step - loss: 0.0094 - mse:
0.0094 - rmse: 0.0970 - mae: 0.0517 - mape: 24.8794 - val_loss: 0.0031 -
val_mse: 0.0031 - val_rmse: 0.0555 - val_mae: 0.0361 - val_mape: 12.2362
Epoch 25/25
920/921 [============================>.] - ETA: 0s - loss: 0.0025 - mse: 0.0025
- rmse: 0.0496 - mae: 0.0316 - mape: 16.3576
Epoch 25: val_mse did not improve from 0.00269
921/921 [==============================] - 109s 119ms/step - loss: 0.0025 - mse:
0.0025 - rmse: 0.0496 - mae: 0.0316 - mape: 16.3573 - val_loss: 0.0027 -
val_mse: 0.0027 - val_rmse: 0.0519 - val_mae: 0.0347 - val_mape: 10.8955
```

0.4 Evaluate model

[ ]:
```
print(date_actual)
```

```
20221014-114748
```

[ ]:
```
best_model = models.load_model(f'model_weights/{date_actual}_cnn_best_weights.
↪hdf5')
```

[ ]:
```
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=np.ceil(test_size / BATCH_SIZE))
```

```
263/263 [==============================] - 17s 64ms/step - loss: 0.0056 - mse:
0.0056 - rmse: 0.0746 - mae: 0.0339 - mape: 17.5864
```

[ ]:
```
predictions = best_model.predict(test_ds, steps=np.ceil(test_size / BATCH_SIZE))
```

```
263/263 [==============================] - 17s 64ms/step
```

[ ]:
```
for image, stage_discharge in test_ds.take(1):
        predictions = best_model.predict(x=image)

        stage_discharge_test_values = stage_discharge[:2].numpy()
        predictions_values = predictions[:2]

        diff = predictions_values.flatten() - stage_discharge_test_values.
↪flatten()
        percentDiff = (diff / stage_discharge_test_values.flatten()) * 100
        absPercentDiff = np.abs(percentDiff)
        # compute the mean and standard deviation of the absolute percentage
        # difference
        mean = np.mean(absPercentDiff)
        std = np.std(absPercentDiff)
        # finally, show some statistics on our model
        print(mean)
        print(std)

        stage_discharge_test_values = stage_discharge[:10]
        predictions_values = predictions[:10]

        for i in range(len(stage_discharge_test_values.numpy())):
                print(f"pred stage: {scaler.
↪inverse_transform(predictions_values)[i][0]}, actual stage: {scaler.
↪inverse_transform(stage_discharge_test_values)[i][0]}")
```

```
            print(f"pred discharge: {scaler.
↪inverse_transform(predictions_values)[i][1]}, actual discharge: {scaler.
↪inverse_transform(stage_discharge_test_values)[i][1]}")
```

```
1/1 [==============================] - 0s 124ms/step
3.2816958696629652
1.3114376019170655
pred stage: 2.3442065715789795, actual stage: 2.32
pred discharge: 300.8064880371094, actual discharge: 268.0
pred stage: 3.580303192138672, actual stage: 3.6
pred discharge: 1679.95751953125, actual discharge: 1690.0
pred stage: 2.792109966278076, actual stage: 2.83
pred discharge: 689.0064086914062, actual discharge: 717.0
pred stage: 2.2538816928863525, actual stage: 2.24
pred discharge: 218.16615295410156, actual discharge: 197.0
pred stage: 3.1894640922546387, actual stage: 3.21
pred discharge: 1358.2999267578125, actual discharge: 1310.0
pred stage: 3.4943759441375732, actual stage: 3.51
pred discharge: 1605.3167724609375, actual discharge: 1650.0
pred stage: 2.461885929107666, actual stage: 2.47
pred discharge: 382.6040344238281, actual discharge: 375.0
pred stage: 2.8839025497436523, actual stage: 2.77
pred discharge: 788.4247436523438, actual discharge: 634.0
pred stage: 3.52038836479187, actual stage: 3.47
pred discharge: 1667.7935791015625, actual discharge: 1620.0
pred stage: 2.184338092803955, actual stage: 2.16
pred discharge: 150.7027130126953, actual discharge: 159.0
```

[ ]:

0.5 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
```

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

```
1/1 [==============================] - 0s 222ms/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
```

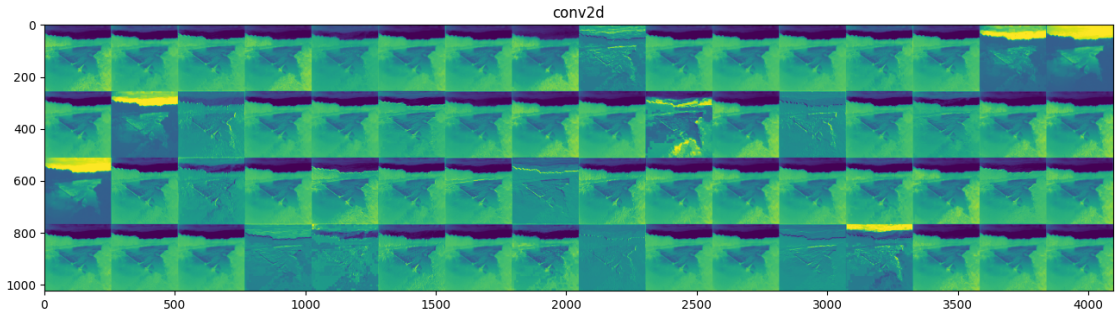
```
images_per_row = 16

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): 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')
```

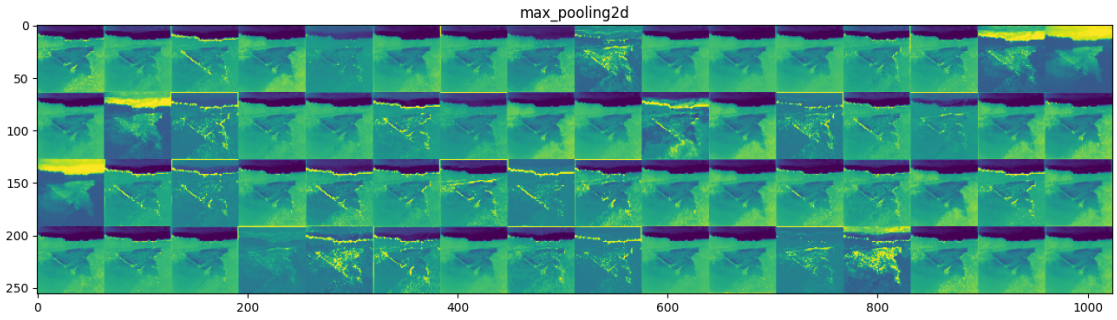
```
conv2d
max_pooling2d
conv2d_1
max_pooling2d_1
conv2d_2
average_pooling2d
conv2d_3
average_pooling2d_1
```

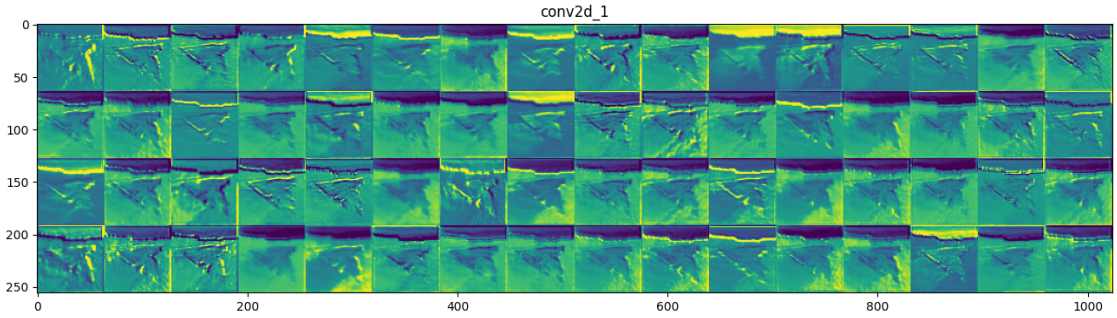
```
/tmp/ipykernel_11977/2269795348.py:24: RuntimeWarning: invalid value encountered
in divide
```

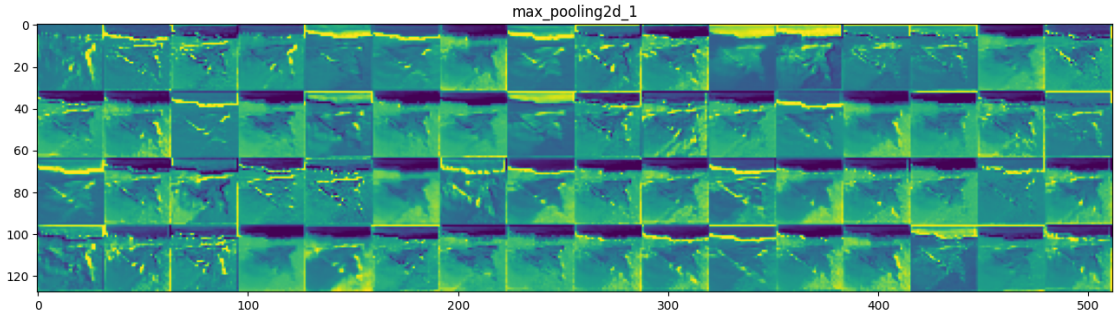
```
channel_image /= channel_image.std()
```

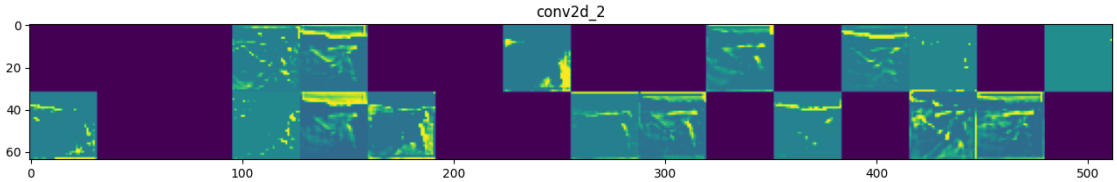
```
---------------------------------------------------------------------------
MemoryError                               Traceback (most recent call last)
Cell In [35], line 13
     11 size = layer_activation.shape[1] #The feature map has shape (1, size,␣
 ↪size, n_features).
     12 n_cols = n_features // 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): break

MemoryError: Unable to allocate 28.7 GiB for an array with shape (153664, 25088␣
 ↪and data type float64
```

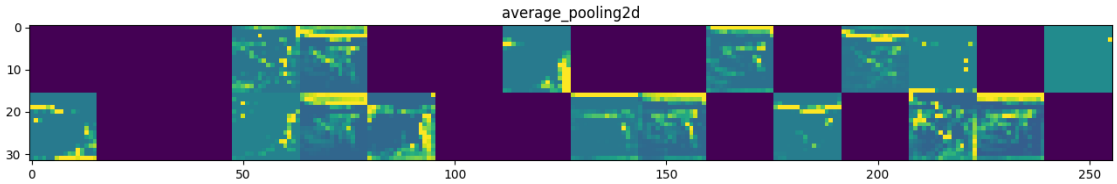


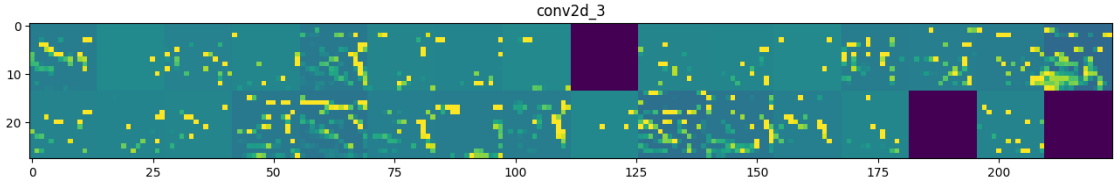


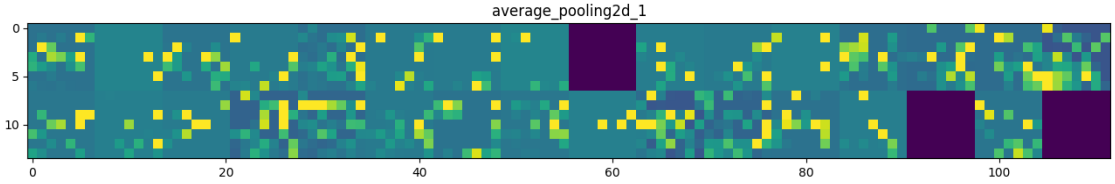

conv2d_1
0
50
100
150
200
250
0
200
400
600
800
1000


max_pooling2d_1
0
20
40
60
80
100
120
0
100
200
300
400
500


conv2d_2
0
20
40
60
0
100
200
300
400
500


average_pooling2d
0
10
20
30
0
50
100
150
200
250


conv2d_3
0
10
20
0
25
50
75
100
125
150
175
200


average_pooling2d_1
0
5
10
0
20
40
60
80
100