```
import pyarrow.parquet as pq
import numpy as np
import pandas as pd
```

```
parquet_file = pq.ParquetFile('E2E_Regression.parquet.9')
```

```
X_train = []
for i in range(5139, 6423):
  X_train.append(parquet_file.read_row_group(i).column(9))
```

```
X_train = np.asarray(X_train)
X_train = np.squeeze(X_train, axis=1)
print(X_train.shape)

     (1284,)
```

```
print(X_train[2][2][124][124])

     0.0
```

```
x1 = []
for i in range(1284):
  for j in range(4):
    for k in range(125):
      x1.append(X_train[i][j][k][:])
```

```
x1 = np.asarray(x1)
print(x1.shape)
print(x1.dtype)
print(x1[0].dtype)
# print(X_train[0][0][0][0].dtype)

     (642000, 125)
     float64
     float64
```

```
np.ndarray.tofile(x1, 'X_val3.dat')
```

```
parquet_file.read_row_group(0).column(3).to_pandas()

     0    3.958357
     Name: am, dtype: float64
```

```
Y_train = []
for j in range(5139, 6423):
  Y_train.append(parquet_file.read_row_group(j).column(3))
```

```
Y_train=np.asarray(Y_train)
Y_train = np.squeeze(Y_train, axis=1)
print(Y_train.shape)
print(Y_train[0])
```

```
(1284,)
3.789943218231201
```

```
# y=np.asarray(y)
# y = np.squeeze(y, axis=1)
print(Y_train.shape)
print(Y_train[0])
print(Y_train.dtype)
```

```
(1284,)
3.789943218231201
float64
```

```
np.ndarray.tofile(Y_train, 'Y_train3.dat')
```

```
np.ndarray.tofile(Y_train, 'Y_val3.dat')
```

```
X_train=np.fromfile('X_train3.dat', dtype=np.float64)
X_train=np.reshape(X_train, (5139, 125, 125, 4))
print(X_train.shape)
Y_train=np.fromfile('Y_train3.dat', dtype=np.float64)
print(Y_train.shape)
X_val=np.fromfile('X_val3.dat', dtype=np.float64)
X_val=np.reshape(X_val, (1284, 125, 125, 4))
print(X_val.shape)
Y_val=np.fromfile('Y_val3.dat', dtype=np.float64)
print(Y_val.shape)
```

```
(5139, 125, 125, 4)
(5139,)
(1284, 125, 125, 4)
(1284,)
```

```
from __future__ import print_function, division

# from keras.datasets import mnist
from keras.applications.vgg19 import VGG19
from keras.layers import BatchNormalization, Activation
from keras.layers import Input
# from keras.layers.merge import _Merge
from keras.layers import Input, Dense, Reshape, Flatten, Dropout
from keras.layers import BatchNormalization, Activation, ZeroPadding2D
from keras.layers import BatchNormalization, Activation, ZeroPadding2D
from keras.layers.advanced_activations import LeakyReLU
from keras.layers.convolutional import UpSampling2D, Conv2D
from keras.models import Sequential, Model
from keras.optimizers import RMSprop,Adam,SGD
from functools import partial

import keras.backend as K
from scipy.ndimage import zoom

import matplotlib.pyplot as plt

import sys

import numpy as np
import glob
import cv2
from keras.layers import Lambda


from __future__ import print_function
from __future__ import absolute_import

import warnings
import numpy as np
import tensorflow as tf
from keras.models import Model
from keras import layers
from keras.layers import Activation
from keras.layers import Dense
from keras.layers import Input
from keras.layers import BatchNormalization
from keras.layers import Conv2D,UpSampling2D
from keras.layers import MaxPooling2D
from keras.layers import AveragePooling2D
from keras.layers import GlobalAveragePooling2D,Add
from keras.layers import GlobalMaxPooling2D
from keras.engine.topology import get_source_inputs
from keras.utils.layer_utils import convert_all_kernels_in_model
from keras.utils.data_utils import get_file
from keras import backend as K
from keras.applications.imagenet_utils import decode_predictions
# from keras_applications.imagenet_utils import _obtain_input_shape
from keras.preprocessing import image
from keras.applications.vgg19 import preprocess_input
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array


ip=Input(shape=(125, 125, 4))

x = layers.Conv2D(1024, (3, 3), activation='relu', padding='same', name='block5d_conv4')(ip)
x=BatchNormalization(momentum=0.8)(x)
```

```
x = layers.Conv2D(512, (3, 3), activation='relu', padding='same', name='block4d_conv4')(x)
x=BatchNormalization(momentum=0.8)(x)

x = layers.Conv2D(512, (3, 3), activation='relu', padding='same', name='block3d_conv4')(x)
x=BatchNormalization(momentum=0.8)(x)

x = layers.Conv2D(512, (3, 3), activation='relu', padding='same', name='block2d_conv2')(x)
x=BatchNormalization(momentum=0.8)(x)

x = layers.Conv2D(254, (3, 3), activation='relu', padding='same', name='block1d_conv2')(x)
x=BatchNormalization(momentum=0.8)(x)

x=layers.Conv2D(3, (3, 3), activation='relu', padding='same')(x)
x=layers.Flatten()(x)
out=layers.Dense(1, activation="linear")(x)
model=Model(ip,out)

print(model.summary())
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_1 (InputLayer)         [(None, 125, 125, 4)]     0         
_________________________________________________________________
block5d_conv4 (Conv2D)       (None, 125, 125, 1024)    37888     
_________________________________________________________________
batch_normalization (BatchNo (None, 125, 125, 1024)    4096      
_________________________________________________________________
block4d_conv4 (Conv2D)       (None, 125, 125, 512)     4719104   
_________________________________________________________________
batch_normalization_1 (Batch (None, 125, 125, 512)     2048      
_________________________________________________________________
block3d_conv4 (Conv2D)       (None, 125, 125, 512)     2359808   
_________________________________________________________________
batch_normalization_2 (Batch (None, 125, 125, 512)     2048      
_________________________________________________________________
block2d_conv2 (Conv2D)       (None, 125, 125, 512)     2359808   
_________________________________________________________________
batch_normalization_3 (Batch (None, 125, 125, 512)     2048      
_________________________________________________________________
block1d_conv2 (Conv2D)       (None, 125, 125, 254)     1170686   
_________________________________________________________________
batch_normalization_4 (Batch (None, 125, 125, 254)     1016      
_________________________________________________________________
conv2d (Conv2D)              (None, 125, 125, 3)       6861      
_________________________________________________________________
flatten (Flatten)            (None, 46875)             0         
_________________________________________________________________
dense (Dense)                (None, 1)                 46876     
=================================================================
Total params: 10,712,287
Trainable params: 10,706,659
Non-trainable params: 5,628
```

```
None
```

```
opt = Adam(learning_rate=0.0001)
# opt = Adam(lr=1e-3, decay=1e-3 / 200)
model.compile(optimizer=opt,loss='mse',metrics=['mape', 'mse', 'mae'])
```

```
model.load_weights('t3_30.h5')
model.fit(X_train, Y_train, validation_data=(X_val, Y_val), batch_size=32,epochs=30,verbose=1
```

```
Epoch 1/30
161/161 [==============================] - 1127s 7s/step - loss: 0.0678 - mape: 4.909
Epoch 2/30
161/161 [==============================] - 1033s 6s/step - loss: 0.0112 - mape: 2.237
Epoch 3/30
161/161 [==============================] - 1033s 6s/step - loss: 0.0101 - mape: 2.111
Epoch 4/30
161/161 [==============================] - 1034s 6s/step - loss: 0.0141 - mape: 2.486
Epoch 5/30
161/161 [==============================] - 1034s 6s/step - loss: 0.0095 - mape: 2.058
Epoch 6/30
161/161 [==============================] - 1035s 6s/step - loss: 0.0082 - mape: 1.879
Epoch 7/30
161/161 [==============================] - 1036s 6s/step - loss: 0.0092 - mape: 2.005
Epoch 8/30
161/161 [==============================] - 1035s 6s/step - loss: 0.0093 - mape: 2.045
Epoch 9/30
161/161 [==============================] - 1036s 6s/step - loss: 0.0086 - mape: 1.966
Epoch 10/30
161/161 [==============================] - 1036s 6s/step - loss: 0.0122 - mape: 2.246
Epoch 11/30
161/161 [==============================] - 1036s 6s/step - loss: 0.0136 - mape: 2.479
Epoch 12/30
161/161 [==============================] - 1037s 6s/step - loss: 0.0072 - mape: 1.761
Epoch 13/30
161/161 [==============================] - 1041s 6s/step - loss: 0.0067 - mape: 1.718
Epoch 14/30
161/161 [==============================] - 1042s 6s/step - loss: 0.0064 - mape: 1.662
Epoch 15/30
161/161 [==============================] - 1040s 6s/step - loss: 0.0096 - mape: 2.060
Epoch 16/30
161/161 [==============================] - 1041s 6s/step - loss: 0.0053 - mape: 1.522
Epoch 17/30
161/161 [==============================] - 1039s 6s/step - loss: 0.0050 - mape: 1.451
Epoch 18/30
161/161 [==============================] - 1037s 6s/step - loss: 0.0054 - mape: 1.530
Epoch 19/30
161/161 [==============================] - 1037s 6s/step - loss: 0.0046 - mape: 1.421
Epoch 20/30
161/161 [==============================] - 1038s 6s/step - loss: 0.0051 - mape: 1.484
Epoch 21/30
161/161 [==============================] - 1036s 6s/step - loss: 0.0063 - mape: 1.685
Epoch 22/30
161/161 [==============================] - 1038s 6s/step - loss: 0.0051 - mape: 1.488
```

```
    Epoch 23/30
    161/161 [==============================] - 1039s 6s/step - loss: 0.0053 - mape: 1.517
    Epoch 24/30
    161/161 [==============================] - 1039s 6s/step - loss: 0.0051 - mape: 1.476
    Epoch 25/30
    161/161 [==============================] - 1037s 6s/step - loss: 0.0039 - mape: 1.3064
    Epoch 26/30
    161/161 [==============================] - 1036s 6s/step - loss: 0.0043 - mape: 1.371
    Epoch 27/30
    161/161 [==============================] - 1037s 6s/step - loss: 0.0130 - mape: 2.308
    Epoch 28/30
    161/161 [==============================] - 1034s 6s/step - loss: 0.0117 - mape: 2.2589
    Epoch 29/30
    161/161 [==============================] - 1034s 6s/step - loss: 0.0098 - mape: 2.0359
```

```
model.load_weights('t3_60.h5')
model.fit(X_train, Y_train, validation_data=(X_val, Y_val), batch_size=32,epochs=30,verbose=1
```

```
    Epoch 1/30
    161/161 [==============================] - 1121s 7s/step - loss: 0.0090 - mape: 1.9860
    Epoch 2/30
    161/161 [==============================] - 1026s 6s/step - loss: 0.0052 - mape: 1.517
    Epoch 3/30
    161/161 [==============================] - 1027s 6s/step - loss: 0.0064 - mape: 1.660
    Epoch 4/30
    161/161 [==============================] - 1027s 6s/step - loss: 0.0046 - mape: 1.408
    Epoch 5/30
    161/161 [==============================] - 1029s 6s/step - loss: 0.0057 - mape: 1.5680
    Epoch 6/30
    161/161 [==============================] - 1028s 6s/step - loss: 0.0052 - mape: 1.508
    Epoch 7/30
    161/161 [==============================] - 1026s 6s/step - loss: 0.0028 - mape: 1.102
    Epoch 8/30
    161/161 [==============================] - 1027s 6s/step - loss: 0.0043 - mape: 1.374
    Epoch 9/30
    161/161 [==============================] - 1025s 6s/step - loss: 0.0051 - mape: 1.4719
    Epoch 10/30
    161/161 [==============================] - 1024s 6s/step - loss: 0.0058 - mape: 1.5500
    Epoch 11/30
    161/161 [==============================] - 1025s 6s/step - loss: 0.0060 - mape: 1.616
    Epoch 12/30
    161/161 [==============================] - 1025s 6s/step - loss: 0.0047 - mape: 1.426
    Epoch 13/30
    161/161 [==============================] - 1024s 6s/step - loss: 0.0030 - mape: 1.128
    Epoch 14/30
    161/161 [==============================] - 1025s 6s/step - loss: 0.0040 - mape: 1.330
    Epoch 15/30
    161/161 [==============================] - 1025s 6s/step - loss: 0.0052 - mape: 1.503
    Epoch 16/30
    161/161 [==============================] - 1025s 6s/step - loss: 0.0067 - mape: 1.7214
    Epoch 17/30
    161/161 [==============================] - 1024s 6s/step - loss: 0.0035 - mape: 1.247
    Epoch 18/30
    161/161 [==============================] - 1025s 6s/step - loss: 0.0027 - mape: 1.0819
    Epoch 19/30
```

```
161/161 [==============================] - 1027s 6s/step - loss: 0.0031 - mape: 1.163
Epoch 20/30
161/161 [==============================] - 1028s 6s/step - loss: 0.0026 - mape: 1.054
Epoch 21/30
161/161 [==============================] - 1025s 6s/step - loss: 0.0031 - mape: 1.160
Epoch 22/30
161/161 [==============================] - 1025s 6s/step - loss: 0.0035 - mape: 1.244
Epoch 23/30
161/161 [==============================] - 1026s 6s/step - loss: 0.0035 - mape: 1.225
Epoch 24/30
161/161 [==============================] - 1025s 6s/step - loss: 0.0030 - mape: 1.152
Epoch 25/30
161/161 [==============================] - 1026s 6s/step - loss: 0.0020 - mape: 0.925
Epoch 26/30
161/161 [==============================] - 1026s 6s/step - loss: 0.0020 - mape: 0.937
Epoch 27/30
161/161 [==============================] - 1026s 6s/step - loss: 0.0052 - mape: 1.520
Epoch 28/30
161/161 [==============================] - 1025s 6s/step - loss: 0.0032 - mape: 1.184
Epoch 29/30
161/161 [==============================] - 1025s 6s/step - loss: 0.0020 - mape: 0.939
```

```
import keras
print(keras.__version__)
```

```
2.4.3
```

```
model.save_weights('t3_90.h5')
```