```
import tensorflow as tf
import tensorflow.keras as keras
import numpy as np
import pandas as pd
import gc
```

```
from pyarrow.parquet import ParquetFile
import pyarrow as pa

pf = ParquetFile('../input/quarksgluons/QCDToGGQQ_IMGjet_RH1all_jet0_run0_n36272.test.snappy.parque
first_rows = next(pf.iter_batches(batch_size = 15000, columns=['X_jets', 'y']))
df = pa.Table.from_batches([first_rows]).to_pandas()
del first_rows
```

```
X_dataset = np.array(np.array(np.array(df['X_jets'].tolist()).tolist()).tolist())
y_dataset = df['y'].to_numpy()
print(X_dataset.shape, y_dataset.shape)
del df
```

```
(15000, 3, 125, 125) (15000,)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_dataset, y_dataset, test_size = 0.2, random_s
X_train = np.moveaxis(X_train, 1, -1)
X_test = np.moveaxis(X_test, 1, -1)
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
gc.collect()
del X_dataset, y_dataset
```

```
(12000, 125, 125, 3) (12000,)
(3000, 125, 125, 3) (3000,)
```

```
import tensorflow as tf
import tensorflow.keras as keras
from keras.models import Sequential, Model, load_model
from keras import optimizers
from keras import layers
from keras.initializers import glorot_uniform, he_uniform
# kernel_initializer=he_uniform(seed=0)
```

```
data_augmentation = keras.Sequential(
    [
        layers.Normalization(),
    ],
    name="data_augmentation",
)
# data_augmentation.layers[0].adapt(X_train)
```

```
2022-03-26 12:24:20.057404: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
2022-03-26 12:24:20.144323: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
2022-03-26 12:24:20.145216: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
2022-03-26 12:24:20.146534: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlo
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2022-03-26 12:24:20.146918: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
2022-03-26 12:24:20.147664: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
2022-03-26 12:24:20.148369: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
```

```
2022-03-26 12:24:20.885356: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
2022-03-26 12:24:20.886242: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
2022-03-26 12:24:20.886901: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] success
2022-03-26 12:24:20.887492: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1510] Created de
```

```
from tensorflow import keras
from keras.models import Sequential, Model
from keras import layers
from keras import optimizers
from keras import regularizers

num_classes = 1
input_shape = (125, 125, 3)
model = Sequential(
    [
        keras.Input(shape=input_shape),
        layers.BatchNormalization(),
        layers.Conv2D(8, kernel_size=(3,3), activation="relu"),
        layers.BatchNormalization(),
        layers.MaxPooling2D(pool_size=(2,2)),
#         layers.Conv2D(16, kernel_size=(3,3), activation="relu"),
#         layers.BatchNormalization(),
#         layers.MaxPooling2D(pool_size=(2,2)),
#         layers.Conv2D(128, kernel_size=(3,3), activation="relu"),
#         layers.BatchNormalization(),
#         layers.MaxPooling2D(pool_size=(2,2)),
        layers.Flatten(),
        layers.Dense(512, activation="relu", kernel_regularizer=regularizers.l2(0.005)),
        layers.Dropout(0.5),
        layers.Dense(num_classes, activation="sigmoid", kernel_regularizer=regularizers.l2(0.001)),
    ]
)
model.summary()
# keras.utils.plot_model(model)
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
batch_normalization (BatchNo (None, 125, 125, 3)       12
_________________________________________________________________
conv2d (Conv2D)              (None, 123, 123, 8)       224
_________________________________________________________________
batch_normalization_1 (Batch (None, 123, 123, 8)       32
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 61, 61, 8)         0
_________________________________________________________________
flatten (Flatten)            (None, 29768)             0
_________________________________________________________________
dense (Dense)                (None, 512)               15241728
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 513
=================================================================
Total params: 15,242,509
Trainable params: 15,242,487
Non-trainable params: 22
_________________________________________________________________
```

```
lr_schedule = keras.optimizers.schedules.ExponentialDecay(
    initial_learning_rate=1e-4,
    decay_steps=4000,
    decay_rate=0.9)
opt_func = keras.optimizers.Adam(learning_rate=lr_schedule)

checkpoint_filepath = 'saved_model'
checkpoint_callback = keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath,
    monitor='val_binary_accuracy',
    save_weights_only=True,
    save_best_only=True)

model.compile(loss='binary_crossentropy',
              optimizer=opt_func,
              metrics=[
                keras.metrics.BinaryAccuracy(name="binary_accuracy", dtype=float, threshold=0.5),
                keras.metrics.AUC(name="auc", from_logits=True),
              ],
             )

history = model.fit(X_train,
          y_train,
          epochs=20,
          validation_split=0.2,
          batch_size=32,
          shuffle=True,
          callbacks=[checkpoint_callback])
```

```
2022-03-26 12:24:22.449042: W tensorflow/core/framework/cpu_allocator_impl.cc:80] Allocation of
2022-03-26 12:24:24.261840: W tensorflow/core/framework/cpu_allocator_impl.cc:80] Allocation of
2022-03-26 12:24:25.581100: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] Non
Epoch 1/20
2022-03-26 12:24:27.334661: I tensorflow/stream_executor/cuda/cuda_dnn.cc:369] Loaded cuDNN ver
300/300 [==============================] - 12s 17ms/step - loss: 4.4862 - binary_accuracy: 0.62
Epoch 2/20
300/300 [==============================] - 4s 13ms/step - loss: 2.6028 - binary_accuracy: 0.768
Epoch 3/20
300/300 [==============================] - 4s 13ms/step - loss: 1.7622 - binary_accuracy: 0.815
Epoch 4/20
300/300 [==============================] - 4s 13ms/step - loss: 1.2983 - binary_accuracy: 0.848
Epoch 5/20
300/300 [==============================] - 4s 13ms/step - loss: 1.0462 - binary_accuracy: 0.868
Epoch 6/20
300/300 [==============================] - 4s 13ms/step - loss: 0.8908 - binary_accuracy: 0.887
Epoch 7/20
300/300 [==============================] - 4s 13ms/step - loss: 0.7915 - binary_accuracy: 0.898
Epoch 8/20
300/300 [==============================] - 4s 13ms/step - loss: 0.7188 - binary_accuracy: 0.912
Epoch 9/20
300/300 [==============================] - 4s 12ms/step - loss: 0.6761 - binary_accuracy: 0.918
Epoch 10/20
300/300 [==============================] - 4s 13ms/step - loss: 0.6309 - binary_accuracy: 0.931
Epoch 11/20
300/300 [==============================] - 4s 13ms/step - loss: 0.6004 - binary_accuracy: 0.931
Epoch 12/20
300/300 [==============================] - 4s 13ms/step - loss: 0.5775 - binary_accuracy: 0.935
Epoch 13/20
300/300 [==============================] - 4s 14ms/step - loss: 0.5503 - binary_accuracy: 0.944
Epoch 14/20
300/300 [==============================] - 4s 13ms/step - loss: 0.5301 - binary_accuracy: 0.943
Epoch 15/20
300/300 [==============================] - 4s 13ms/step - loss: 0.5231 - binary_accuracy: 0.946
Epoch 16/20
300/300 [==============================] - 4s 13ms/step - loss: 0.5234 - binary_accuracy: 0.944
```

```
Epoch 17/20
300/300 [==============================] - 4s 13ms/step - loss: 0.5018 - binary_accuracy: 0.953
Epoch 18/20
300/300 [==============================] - 4s 13ms/step - loss: 0.4634 - binary_accuracy: 0.959
Epoch 19/20
300/300 [==============================] - 4s 13ms/step - loss: 0.4541 - binary_accuracy: 0.959
Epoch 20/20
300/300 [==============================] - 4s 13ms/step - loss: 0.4538 - binary_accuracy: 0.959
```

```
model.load_weights(checkpoint_filepath)
_, accuracy, auc = model.evaluate(X_test, y_test)
print(f"Test accuracy: {accuracy}")
print(f"Test AUC: {auc}")
```

```
94/94 [==============================] - 1s 6ms/step - loss: 1.2723 - binary_accuracy: 0.6800 -
Test accuracy: 0.6800000071525574
Test AUC: 0.7313769459724426
```

```
import matplotlib.pyplot as plt
print(history.history.keys())
# summarize history for accuracy
plt.plot(history.history['binary_accuracy'])
plt.plot(history.history['val_binary_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'valid'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'valid'], loc='upper left')
plt.show()
```