|  |  |
| --- | --- |
|  | import os, types |
|  | from ibm\_watson\_machine\_learning import APIClient |
|  | import pandas as pd |
|  | from botocore.client import Config |
|  | import ibm\_boto3 |
|  | from io import BytesIO |
|  | import zipfile |
|  |  |
|  | #Due to privacy concerns, I've not mentioned the API Keys and Endpoints Here |
|  | def \_\_iter\_\_(self): return 0 |
|  |  |
|  | cos\_client = ibm\_boto3.client(service\_name='s3', |
|  | ibm\_api\_key\_id=<api\_key>, |
|  | ibm\_auth\_endpoint=<end\_point>, |
|  | config=Config(signature\_version='oauth'), |
|  | endpoint\_url=<end\_point\_url>') |
|  |  |
|  | bucket = <bucket\_name> |
|  | object\_key = <object\_key> |
|  |  |
|  | streaming\_body\_1 = cos\_client.get\_object(Bucket=bucket, Key=object\_key)['Body'] |
|  |  |
|  | unzip=zipfile.ZipFile(BytesIO(streaming\_body\_1.read()),'r') |
|  | filepaths=unzip.namelist() |
|  | for path in filepaths: |
|  | unzip.extract(path) |
|  |  |
|  | train\_datagen = ImageDataGenerator( rescale=1./255, |
|  | rotation\_range=10., |
|  | width\_shift\_range=0.1, |
|  | height\_shift\_range=0.1, |
|  | zoom\_range=0.2, |
|  | horizontal\_flip=True |
|  | ) |
|  | train\_gen = train\_datagen.flow\_from\_directory( |
|  | r'/home/wsuser/work/Finger Dataset/train', |
|  | target\_size=(128,128), |
|  | color\_mode='grayscale', |
|  | batch\_size=32, |
|  | classes=['0','1','2','3','4','5'], |
|  | class\_mode='categorical' |
|  | ) |
|  | test\_datagen = ImageDataGenerator( rescale=1./255 ) |
|  | test\_gen = test\_datagen.flow\_from\_directory( |
|  | r'/home/wsuser/work/Finger Dataset/test', |
|  | target\_size=(128,128), |
|  | color\_mode='grayscale', |
|  | batch\_size=32, |
|  | classes=['0','1','2','3','4','5'], |
|  | class\_mode='categorical' |
|  | ) |
|  | model=Sequential() |
|  | model.add(BatchNormalization(input\_shape = (128,128,1))) |
|  | model.add(Convolution2D(32, (3,3), activation ='relu', input\_shape = (128, 128, 1))) |
|  | model.add(MaxPooling2D(pool\_size=2)) |
|  | model.add(Convolution2D(filters=6,kernel\_size=4,padding='same',activation='relu')) |
|  | model.add(MaxPooling2D(pool\_size=2)) |
|  | model.add(Convolution2D(filters=128,kernel\_size=3,padding='same',activation='relu')) |
|  | model.add(MaxPooling2D(pool\_size=2)) |
|  | model.add(Convolution2D(filters=128,kernel\_size=2,padding='same',activation='relu')) |
|  | model.add(MaxPooling2D(pool\_size=2)) |
|  | model.add(Flatten()) |
|  | model.add(Dense(units=128,activation = 'relu')) |
|  | model.add(Dense(units = 64, activation = 'relu')) |
|  | model.add(Dense(units = 32, activation = 'relu')) |
|  | model.add(Dense(units = 6, activation = 'softmax')) |
|  | model.summary() |
|  | model.compile(optimizer='adam', loss = 'categorical\_crossentropy',metrics = ['accuracy']) |
|  | model.fit\_generator(train\_gen, |
|  | epochs=20, |
|  | steps\_per\_epoch=18000//32, |
|  | validation\_data=test\_gen, |
|  | verbose = 1,validation\_steps=3600//32) |
|  | model.save('gesture.h5') |
|  | wml\_credentials={ |
|  | "url":'https://us-south.ml.cloud.ibm.com', |
|  | "apikey":'on6wVLLy-ERS74JlvyDrFdJ35GRaHzaCtKxejqR7euwG' |
|  | } |
|  | client=APIClient(wml\_credentials) |
|  |  |
|  |  |
|  | def guid\_from\_space\_name(client,space\_name): |
|  | space=client.spaces.get\_details() |
|  | return(next(item for item in space['resources'] if item['entity']['name']==space\_name)['metadata']['id']) |
|  |  |
|  |  |
|  | space\_uid=guid\_from\_space\_name(client,'Gesture\_Deploy') |
|  | client.set.default\_space(space\_uid) |
|  |  |
|  | software\_spec\_uid=client.software\_specifications.get\_uid\_by\_name('tensorflow\_rt22.1-py3.9') |
|  |  |
|  | !tar -zcvf gesture\_based\_tool.tgz gesture.h5 |
|  |  |
|  | model\_details=client.repository.store\_model(model='gesture\_based\_tool.tgz',meta\_props={ |
|  | client.repository.ModelMetaNames.NAME:"Gesture Based Tool", |
|  | client.repository.ModelMetaNames.TYPE:"tensorflow\_2.7", |
|  | client.repository.ModelMetaNames.SOFTWARE\_SPEC\_UID:software\_spec\_uid |
|  | } |
|  | ) |
|  | model\_id=client.repository.get\_model\_id(model\_details) |