Sprint 4

A Gesture-based Tool for Sterile Browsing of Radiology Images

Model Deployment :

dataplatform.cloud.ibm.com/analytics/notebooks/v2/af4d524e-680a-4f3c-b2f9-f3964c3c2ce6/view?projectid=a811d06a-150f-4dc1-a347-42561...

IBM Watson Studio | Search in your workspaces | Buy | Rithik R's Account | Dallas | RR

Projects / A Gesture-Based Tool For Sterile ... / Model_building

In [14]:
```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='tAs6NHrXkeZtYKGMuuB4Kcz1zvK9jbBVa-3a-i1TXDHR',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'agesturebasedtoolforsterilebrowsi-donotdelete-pr-okekrvprcufqa2'
object_key = 'Dataset.zip'

streaming_body_1 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/
```

In [15]:
```
from io import BytesIO
import zipfile
```

```
unzip = zipfile.ZipFile(BytesIO(streaming_body_1.read()),'r')
file_path = unzip.namelist()
for path in file_path:
    unzip.extract(path)
```

In [16]:
```
ls
```

```
Dataset/
```

In [18]:
```
# This library helps add support for large, multi-dimensional arrays and matrices
import numpy as np
#open source used for both ML and DL for computation
import tensorflow as tf
#it is a plain stack of layers
from tensorflow.keras.models import Sequential
#Dense layer is the regular deeply connected neural network layer
from tensorflow.keras.layers import Dense,Flatten, Dropout
#Faltten-used fot flattening the input or change the dimension, MaxPooling2D-for downsampling the image for Convolutional layer
from tensorflow.keras.layers import Convolution2D,MaxPooling2D
#Its used for different augmentation of the image
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [19]:
```
#setting parameter for Image Data agumentation to the traing data
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
#Image Data agumentation to the testing data
test_datagen=ImageDataGenerator(rescale=1./255)
```

In [20]:
```
x_train = train_datagen.flow_from_directory(r"Dataset/train/",
                                            target_size=(64, 64),
                                            batch_size=3,
                                            color_mode='grayscale',
                                            class_mode='categorical')

#performing data agumentation to test data
x_test = test_datagen.flow_from_directory(r"Dataset/test/",
                                          target_size=(64, 64),
                                          batch_size=3,
                                          color_mode='grayscale',
                                          class_mode='categorical')
```

```
Found 594 images belonging to 6 classes.
Found 30 images belonging to 6 classes.
```

In [21]:
```
print(x_train.class_indices)
```

```
{'0': 0, '1': 1, '2': 2, '3': 3, '4': 4, '5': 5}
```

Model Creation

In [22]:
```
model = Sequential()
```

In [23]:
```
# First convolution layer and pooling
model.add(Convolution2D(32, (3, 3), input_shape=(64, 64, 1), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

In [24]:
```
# Second convolution layer and pooling
model.add(Convolution2D(32, (3, 3), activation='relu'))
# input_shape is going to be the pooled feature maps from the previous convolution layer
model.add(MaxPooling2D(pool_size=(2,2)))
```

In [25]:
```
# Flattening the layers i.e. input layer
model.add(Flatten())
```

In [26]:
```
# Adding a fully connected layer, i.e. Hidden Layer
model.add(Dense(units=512 , activation='relu'))
```

In [27]:
```
# softmax for categorical analysis, Output Layer
model.add(Dense(units=6, activation='softmax'))
```

In [28]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        320

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 29, 29, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 flatten (Flatten)           (None, 6272)              0

 dense (Dense)               (None, 512)               3211776

 dense_1 (Dense)             (None, 6)                 3078

=================================================================
Total params: 3,224,422
Trainable params: 3,224,422
Non-trainable params: 0
_________________________________________________________________
```

Model Compilation

In [29]:
```
# Compiling the CNN
# categorical_crossentropy for more than 2
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Model Fitting

In [30]:
```
# It will generate packets of train and test data for training
model.fit_generator(x_train,
                    steps_per_epoch = 594/3 ,
                    epochs = 25,
                    validation_data = x_test,
                    validation_steps = 30/3 )
```

```
Epoch 1/25
198/198 [==============================] - 14s 68ms/step - loss: 1.2871 - accuracy: 0.5152 - val_loss: 0.7762 - val_accuracy: 0.6333
Epoch 2/25
198/198 [==============================] - 13s 68ms/step - loss: 0.5824 - accuracy: 0.7559 - val_loss: 0.4198 - val_accuracy: 0.8333
Epoch 3/25
198/198 [==============================] - 13s 67ms/step - loss: 0.4235 - accuracy: 0.8300 - val_loss: 0.1689 - val_accuracy: 0.9667
Epoch 4/25
198/198 [==============================] - 15s 75ms/step - loss: 0.2671 - accuracy: 0.8939 - val_loss: 0.2322 - val_accuracy: 0.9667
Epoch 5/25
198/198 [==============================] - 14s 69ms/step - loss: 0.2073 - accuracy: 0.9192 - val_loss: 0.2652 - val_accuracy: 0.9333
Epoch 6/25
198/198 [==============================] - 14s 68ms/step - loss: 0.1325 - accuracy: 0.9562 - val_loss: 0.2813 - val_accuracy: 0.9333
Epoch 7/25
198/198 [==============================] - 14s 69ms/step - loss: 0.1427 - accuracy: 0.9596 - val_loss: 0.2561 - val_accuracy: 0.9000
Epoch 8/25
198/198 [==============================] - 13s 68ms/step - loss: 0.1078 - accuracy: 0.9579 - val_loss: 0.2664 - val_accuracy: 0.9667
Epoch 9/25
198/198 [==============================] - 15s 76ms/step - loss: 0.1224 - accuracy: 0.9495 - val_loss: 0.1192 - val_accuracy: 0.9333
Epoch 10/25
198/198 [==============================] - 14s 68ms/step - loss: 0.0755 - accuracy: 0.9731 - val_loss: 0.4680 - val_accuracy: 0.9333
Epoch 11/25
198/198 [==============================] - 13s 66ms/step - loss: 0.0733 - accuracy: 0.9747 - val_loss: 0.3866 - val_accuracy: 0.9000
Epoch 12/25
198/198 [==============================] - 14s 68ms/step - loss: 0.0405 - accuracy: 0.9815 - val_loss: 0.5000 - val_accuracy: 0.9333
Epoch 13/25
198/198 [==============================] - 14s 73ms/step - loss: 0.0624 - accuracy: 0.9781 - val_loss: 0.3178 - val_accuracy: 0.9667
Epoch 14/25
198/198 [==============================] - 13s 67ms/step - loss: 0.0678 - accuracy: 0.9798 - val_loss: 0.5067 - val_accuracy: 0.9000
Epoch 15/25
198/198 [==============================] - 13s 66ms/step - loss: 0.0712 - accuracy: 0.9798 - val_loss: 0.3540 - val_accuracy: 0.9333
Epoch 16/25
198/198 [==============================] - 13s 67ms/step - loss: 0.0619 - accuracy: 0.9848 - val_loss: 0.3448 - val_accuracy: 0.9667
Epoch 17/25
198/198 [==============================] - 13s 68ms/step - loss: 0.0455 - accuracy: 0.9865 - val_loss: 0.2879 - val_accuracy: 0.9667
Epoch 18/25
198/198 [==============================] - 14s 70ms/step - loss: 0.0679 - accuracy: 0.9815 - val_loss: 0.3151 - val_accuracy: 0.9667
Epoch 19/25
198/198 [==============================] - 14s 70ms/step - loss: 0.0276 - accuracy: 0.9916 - val_loss: 0.2860 - val_accuracy: 0.9667
Epoch 20/25
198/198 [==============================] - 13s 67ms/step - loss: 0.0615 - accuracy: 0.9798 - val_loss: 0.4351 - val_accuracy: 0.9667
Epoch 21/25
198/198 [==============================] - 14s 70ms/step - loss: 0.0146 - accuracy: 0.9933 - val_loss: 0.3405 - val_accuracy: 0.9667
Epoch 22/25
198/198 [==============================] - 14s 69ms/step - loss: 0.0086 - accuracy: 0.9983 - val_loss: 0.4429 - val_accuracy: 0.9667
Epoch 23/25
198/198 [==============================] - 13s 66ms/step - loss: 0.0228 - accuracy: 0.9949 - val_loss: 0.5591 - val_accuracy: 0.9667
Epoch 24/25
198/198 [==============================] - 13s 67ms/step - loss: 0.0183 - accuracy: 0.9916 - val_loss: 0.6934 - val_accuracy: 0.9000
Epoch 25/25
198/198 [==============================] - 13s 67ms/step - loss: 0.0271 - accuracy: 0.9882 - val_loss: 0.4183 - val_accuracy: 0.9333
```

Out[30]: <keras.callbacks.History at 0x7f5524ebcbe0>

Saving the Model

In [31]:
```
model.save('gesture.h5')
```

In [32]:
```
!tar -zcvf gesture-classification-model.tgz gesture.h5
```

```
gesture.h5
```

IBM Deployment

In [33]:
```
!pip install ibm_watson_machine_learning
```

```
Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.255)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2022.9.24)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.8.9)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (4.8.2)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (21.3)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.3.4)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.26.7)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.26.0)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machi
ne_learning) (0.10.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_wats
on_machine_learning) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_mac
hine_learning) (2.11.0)
```

In [69]:
```
from ibm_watson_machine_learning import APIClient
```

In [70]:
```
wml_credentials = {
    "url":"https://us-south.ml.cloud.ibm.com",
        "apikey": "00QFbTkSTa9Pajo2rUtKC-URlxSvPn8A_9cH60GE6XQP"

}
```

In [71]:
```
client = APIClient(wml_credentials)
```

In [72]:
```
client
```

Out[72]: <ibm_watson_machine_learning.client.APIClient at 0x7f54accb9340>

In [73]:
```
client.spaces.list()
```

```
Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50
------------------------------------  ------------------  ------------------------
ID                                    NAME                CREATED
a76bf5bd-8553-45c8-bff9-ac13291f3e65  Gesture based tool  2022-11-01T09:34:21.631Z
------------------------------------  ------------------  ------------------------
```

In [74]:
```
space_id = "a76bf5bd-8553-45c8-bff9-ac13291f3e65"
```

```
In [75]: client.set.default_space(space_id)

Out[75]: 'SUCCESS'

In [76]: client.software_specifications.list()

------------------------------  ------------------------------------  ----
NAME                            ASSET_ID                              TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12       020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9       0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6               10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl       111e41b3-de2d-5422-a4d6-bf776828c4b7  base
runtime-22.1-py3.9              12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6         154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                    1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6          1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
kernel-spark3.3-r3.6            1c9e5454-f216-59dd-a20e-474a5cdf5988  base
pytorch-onnx_rt22.1-py3.9-edt   1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
tensorflow_2.1-py3.6            1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
spark-mllib_3.2                 20047f72-0a98-58c7-9ff5-a77b012eb8f5  base
tensorflow_2.4-py3.8-horovod    217c16f6-178f-56bf-824a-b19f20564c49  base
runtime-22.1-py3.9-cuda         26215f05-08c3-5a41-a1b0-da66306ce658  base

In [77]: software_space_uid=client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9')

In [83]: software_space_uid

Out[83]: 'acd9c798-6974-5d2f-a657-ce06e986df4d'

In [ ]: model_details=client.repository.store_model(model='gesture-classification-model.tgz',meta_props={
            client.repository.ModelMetaNames.NAME:"CNN Model Building",
            client.repository.ModelMetaNames.TYPE:'tensorflow_2.7',
            client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid
        })

In [81]: model_id = client.repository.get_model_id(model_details)

In [82]: model_id

Out[82]: '48c7db6a-7364-4576-8944-628211cc3bd4'

In [ ]:
```