IBM CLOUD DEPLOYMENT

{

"cells": [

{

"cell\_type": "code",

"execution\_count": 1,

"metadata": {},

"outputs": [],

"source": [

"import keras\n",

"import tensorflow\n",

"\n",

"from tensorflow.keras.preprocessing.image import ImageDataGenerator"

]

},

{

"cell\_type": "code",

"execution\_count": 2,

"metadata": {},

"outputs": [],

"source": [

"import tensorflow\n",

"\n",

"from tensorflow.keras.preprocessing.image import ImageDataGenerator"

]

},

{

"cell\_type": "code",

"execution\_count": 5,

"metadata": {},

"outputs": [],

"source": [

"\n",

"import os, types\n",

"import pandas as pd\n",

"from botocore.client import Config\n",

"import ibm\_boto3\n",

"\n",

"def \_\_iter\_\_(self): return 0\n",

"\n",

"# @hidden\_cell\n",

"# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.\n",

"# You might want to remove those credentials before you share the notebook.\n",

"cos\_client = ibm\_boto3.client(service\_name='s3',\n",

" ibm\_api\_key\_id='JLXO-4-TMJB87CTQKc6dVclYtSXBMueJZxQKcaRUK0VP',\n",

" ibm\_auth\_endpoint=\"https://iam.cloud.ibm.com/oidc/token\",\n",

" config=Config(signature\_version='oauth'),\n",

" endpoint\_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')\n",

"\n",

"bucket = 'forestfiredetection-donotdelete-pr-bpytmsf9pwiglr'\n",

"object\_key = 'Dataset.zip'\n",

"\n",

"streaming\_body\_1 = cos\_client.get\_object(Bucket=bucket, Key=object\_key)['Body']\n",

"\n",

"# Your data file was loaded into a botocore.response.StreamingBody object.\n",

"# Please read the documentation of ibm\_boto3 and pandas to learn more about the possibilities to load the data.\n",

"# ibm\_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/\n",

"# pandas documentation: http://pandas.pydata.org/\n"

]

},

{

"cell\_type": "code",

"execution\_count": 6,

"metadata": {},

"outputs": [],

"source": [

"from io import BytesIO\n",

"import zipfile\n",

"unzip = zipfile.ZipFile(BytesIO(streaming\_body\_1.read()),'r')\n",

"file\_paths = unzip.namelist()\n",

"for path in file\_paths:\n",

" unzip.extract(path)"

]

},

{

"cell\_type": "code",

"execution\_count": 7,

"metadata": {},

"outputs": [],

"source": [

"train\_datagen = ImageDataGenerator(rescale=1./255,\n",

" shear\_range=0.2,\n",

" rotation\_range=180,\n",

" zoom\_range=0.2,\n",

" horizontal\_flip=True)\n",

"\n",

"test\_datagen = ImageDataGenerator(rescale=1./255)"

]

},

{

"cell\_type": "code",

"execution\_count": 8,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Found 436 images belonging to 2 classes.\n"

]

}

],

"source": [

"x\_train = train\_datagen.flow\_from\_directory(r'./Dataset/train\_set/',\n",

" target\_size=(128, 128),\n",

" batch\_size=32,\n",

" class\_mode='binary')"

]

},

{

"cell\_type": "code",

"execution\_count": 14,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Found 121 images belonging to 2 classes.\n"

]

}

],

"source": [

"x\_test = train\_datagen.flow\_from\_directory(r'./Dataset/test\_set/',\n",

" target\_size=(128, 128),\n",

" batch\_size=32,\n",

" class\_mode='binary')"

]

},

{

"cell\_type": "code",

"execution\_count": 15,

"metadata": {},

"outputs": [],

"source": [

"from tensorflow.keras.models import Sequential\n",

"from tensorflow.keras.layers import Dense, Convolution2D, MaxPooling2D, Flatten\n"

]

},

{

"cell\_type": "code",

"execution\_count": 16,

"metadata": {},

"outputs": [],

"source": [

"model = Sequential()\n",

"model.add(Convolution2D(32, (3,3), input\_shape=(128, 128, 3), activation=\"relu\"))\n",

"model.add(MaxPooling2D(pool\_size=(2,2)))\n",

"model.add(Flatten())\n",

"model.add(Dense(150,activation=\"relu\"))\n",

"model.add(Dense(1, activation=\"sigmoid\"))\n"

]

},

{

"cell\_type": "code",

"execution\_count": 17,

"metadata": {},

"outputs": [],

"source": [

"model.compile(loss=\"binary\_crossentropy\",\n",

" optimizer=\"adam\",\n",

" metrics=[\"accuracy\"])"

]

},

{

"cell\_type": "code",

"execution\_count": 18,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Epoch 1/10\n",

"14/14 [==============================] - 24s 2s/step - loss: 2.0141 - accuracy: 0.7133 - val\_loss: 0.1630 - val\_accuracy: 0.9421\n",

"Epoch 2/10\n",

"14/14 [==============================] - 22s 2s/step - loss: 0.3240 - accuracy: 0.8922 - val\_loss: 0.1051 - val\_accuracy: 0.9835\n",

"Epoch 3/10\n",

"14/14 [==============================] - 22s 2s/step - loss: 0.2306 - accuracy: 0.9014 - val\_loss: 0.1186 - val\_accuracy: 0.9421\n",

"Epoch 4/10\n",

"14/14 [==============================] - 21s 2s/step - loss: 0.1938 - accuracy: 0.9174 - val\_loss: 0.0852 - val\_accuracy: 0.9752\n",

"Epoch 5/10\n",

"14/14 [==============================] - 22s 1s/step - loss: 0.1953 - accuracy: 0.9243 - val\_loss: 0.1242 - val\_accuracy: 0.9339\n",

"Epoch 6/10\n",

"14/14 [==============================] - 21s 2s/step - loss: 0.1797 - accuracy: 0.9128 - val\_loss: 0.0790 - val\_accuracy: 0.9835\n",

"Epoch 7/10\n",

"14/14 [==============================] - 21s 1s/step - loss: 0.1688 - accuracy: 0.9335 - val\_loss: 0.0905 - val\_accuracy: 0.9421\n",

"Epoch 8/10\n",

"14/14 [==============================] - 21s 1s/step - loss: 0.1727 - accuracy: 0.9220 - val\_loss: 0.1370 - val\_accuracy: 0.9256\n",

"Epoch 9/10\n",

"14/14 [==============================] - 22s 2s/step - loss: 0.2078 - accuracy: 0.9128 - val\_loss: 0.0687 - val\_accuracy: 0.9917\n",

"Epoch 10/10\n",

"14/14 [==============================] - 22s 2s/step - loss: 0.1622 - accuracy: 0.9266 - val\_loss: 0.0673 - val\_accuracy: 0.9835\n"

]

},

{

"data": {

"text/plain": [

"<keras.callbacks.History at 0x7f63f52b16d0>"

]

},

"execution\_count": 18,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"model.fit(x\_train, steps\_per\_epoch=14, epochs=10, validation\_data=x\_test, validation\_steps=4)"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Save the model"

]

},

{

"cell\_type": "code",

"execution\_count": 19,

"metadata": {},

"outputs": [],

"source": [

"model.save(\"model.h5\")"

]

},

{

"cell\_type": "code",

"execution\_count": 20,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"model.h5\r\n"

]

}

],

"source": [

"!tar -zcvf model.tgz model.h5"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Prediction"

]

},

{

"cell\_type": "code",

"execution\_count": 21,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Collecting watson-machine-learning-client\n",

" Downloading watson\_machine\_learning\_client-1.0.391-py3-none-any.whl (538 kB)\n",

"\u001b[K |████████████████████████████████| 538 kB 18.1 MB/s eta 0:00:01\n",

"\u001b[?25hRequirement already satisfied: tqdm in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (4.62.3)\n",

"Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (0.8.9)\n",

"Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.26.7)\n",

"Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.3.4)\n",

"Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2022.9.24)\n",

"Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.11.0)\n",

"Requirement already satisfied: boto3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.18.21)\n",

"Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.26.0)\n",

"Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (0.3.3)\n",

"Requirement already satisfied: botocore<1.22.0,>=1.21.21 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (1.21.41)\n",

"Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.10.0)\n",

"Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.5.0)\n",

"Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client) (2.8.2)\n",

"Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client) (1.15.0)\n",

"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->watson-machine-learning-client) (2.11.0)\n",

"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->watson-machine-learning-client) (2.11.0)\n",

"Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (2.0.4)\n",

"Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (3.3)\n",

"Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (2021.3)\n",

"Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (1.20.3)\n",

"Installing collected packages: watson-machine-learning-client\n",

"Successfully installed watson-machine-learning-client-1.0.391\n"

]

}

],

"source": [

"!pip install watson-machine-learning-client"

]

},

{

"cell\_type": "code",

"execution\_count": 22,

"metadata": {},

"outputs": [],

"source": [

"from ibm\_watson\_machine\_learning import APIClient\n",

"\n",

"API\_KEY = \"5W65wtnV1kus0WCtJ4HEMzw-lVetPUlY\_B2Nje3fDq4p\"\n",

"\n",

"credentials = {\n",

" \"url\": \"https://us-south.ml.cloud.ibm.com\",\n",

" \"apikey\": API\_KEY\n",

"}\n",

"\n",

"client = APIClient(credentials)"

]

},

{

"cell\_type": "code",

"execution\_count": 24,

"metadata": {},

"outputs": [],

"source": [

"def guid\_from\_space\_name(client, space\_name):\n",

" space = client.spaces.get\_details()\n",

" return(next(item for item in space['resources'] if item['entity']['name'] == space\_name)['metadata']['id'])"

]

},

{

"cell\_type": "code",

"execution\_count": 32,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"{'resources': []}"

]

},

"execution\_count": 32,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

" space = client.spaces.get\_details()\n",

" space"

]

},

{

"cell\_type": "code",

"execution\_count": 33,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Space UID: d9308ab8-179a-48da-974b-d986f1649bd5\n"

]

}

],

"source": [

"space\_uid = guid\_from\_space\_name(client, 'Forest fire detection')\n",

"print(\"Space UID: \", space\_uid)"

]

},

{

"cell\_type": "code",

"execution\_count": 34,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"'SUCCESS'"

]

},

"execution\_count": 34,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"client.set.default\_space(space\_uid)"

]

},

{

"cell\_type": "code",

"execution\_count": 35,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"'12b83a17-24d8-5082-900f-0ab31fbfd3cb'"

]

},

"execution\_count": 35,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"software\_spec\_uid = client.software\_specifications.get\_uid\_by\_name(\"runtime-22.1-py3.9\")\n",

"software\_spec\_uid"

]

},

{

"cell\_type": "code",

"execution\_count": 36,

"metadata": {},

"outputs": [],

"source": [

"model\_details = client.repository.store\_model(model=\"model.tgz\", meta\_props={\n",

" client.repository.ModelMetaNames.NAME: \"CNN\",\n",

" client.repository.ModelMetaNames.TYPE: \"tensorflow\_2.7\",\n",

" client.repository.ModelMetaNames.SOFTWARE\_SPEC\_UID: software\_spec\_uid\n",

"})\n",

"\n",

"model\_id = client.repository.get\_model\_id(model\_details)"

]

},

{

"cell\_type": "code",

"execution\_count": 37,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Successfully saved model content to file: 'model.tar.gz'\n"

]

},

{

"data": {

"text/plain": [

"'/home/wsuser/work/model.tar.gz'"

]

},

"execution\_count": 37,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"client.repository.download(model\_id, \"model.tar.gz\")"

]

}

],

"metadata": {

"kernelspec": {

"display\_name": "Python 3.9",

"language": "python",

"name": "python3"

},

"language\_info": {

"codemirror\_mode": {

"name": "ipython",

"version": 3

},

"file\_extension": ".py",

"mimetype": "text/x-python",

"name": "python",

"nbconvert\_exporter": "python",

"pygments\_lexer": "ipython3",

"version": "3.9.13"

},

"vscode": {

"interpreter": {

"hash": "a9cff5a362bc38ef45d817ae74b1af54d6a076e3d773891282bce078b815ba34"

}

}

},

"nbformat": 4,

"nbformat\_minor": 2

}