PROJECT DEVELOPMENT

DELIVERY SPRINT 4

|  |  |
| --- | --- |
| DATE | 08-11-2022 |
| TEAM ID | PNT2022TMID40220 |
| PROJECT NAME | EMERGING METHODS FOR EARLY DEDECTION OF FOREST FIRES |

{

"nbformat": 4,

"nbformat\_minor": 0,

"metadata": {

"colab": {

"provenance": [],

"collapsed\_sections": []

},

"kernelspec": {

"name": "python3",

"display\_name": "Python 3"

},

"language\_info": {

"name": "python"

}

},

"cells": [

{

"cell\_type": "markdown",

"source": [

"# \*\*DATA COLLECTION\*\*"

],

"metadata": {

"id": "ANAn53JJCh0B"

}

},

{

"cell\_type": "code",

"execution\_count": 1,

"metadata": {

"id": "BXCT4OadBFNK"

},

"outputs": [],

"source": [

"import tensorflow as tf\n",

"import numpy as np\n",

"from tensorflow import keras\n",

"import os\n",

"import cv2\n",

"from tensorflow.keras.preprocessing.image import ImageDataGenerator\n",

"from tensorflow.keras.preprocessing import image\n",

"import matplotlib.pyplot as plt"

]

},

{

"cell\_type": "code",

"source": [

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

"test = ImageDataGenerator(rescale=1/255)\n",

"\n",

"train\_dataset = train.flow\_from\_directory(\"/content/drive/MyDrive/IBM/Dataset/train\_set\",\n",

" target\_size=(150,150),\n",

" batch\_size = 32,\n",

" class\_mode = 'binary')\n",

"\n",

"test\_dataset = test.flow\_from\_directory(\"/content/drive/MyDrive/IBM/Dataset/test\_set\",\n",

" target\_size=(150,150),\n",

" batch\_size =32,\n",

" class\_mode = 'binary')"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "T7KK5YZOBKXq",

"outputId": "4a4e56a9-8cbc-4b46-e64e-6eee6fff20b4"

},

"execution\_count": 5,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Found 20 images belonging to 2 classes.\n",

"Found 20 images belonging to 2 classes.\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"test\_dataset.class\_indices"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "uEP-FODSCUdj",

"outputId": "15658cf1-ebf3-4afd-e9f2-82647c91a0cb"

},

"execution\_count": 6,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"{'Fire': 0, 'NoFire': 1}"

]

},

"metadata": {},

"execution\_count": 6

}

]

},

{

"cell\_type": "markdown",

"source": [

"# \*\*MODEL BUILDING AND FITTING\*\*"

],

"metadata": {

"id": "tGRG7iuECvqH"

}

},

{

"cell\_type": "code",

"source": [

"model = keras.Sequential()\n",

"model.add(keras.layers.Conv2D(32,(3,3),activation='relu',input\_shape=(150,150,3)))\n",

"model.add(keras.layers.MaxPool2D(2,2))\n",

"model.add(keras.layers.Conv2D(64,(3,3),activation='relu'))\n",

"model.add(keras.layers.MaxPool2D(2,2))\n",

"model.add(keras.layers.Conv2D(128,(3,3),activation='relu'))\n",

"model.add(keras.layers.MaxPool2D(2,2))\n",

"model.add(keras.layers.Conv2D(128,(3,3),activation='relu'))\n",

"model.add(keras.layers.MaxPool2D(2,2))\n",

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

"model.add(keras.layers.Dense(512,activation='relu'))\n",

"model.add(keras.layers.Dense(1,activation='sigmoid'))"

],

"metadata": {

"id": "bYe4luWkCza1"

},

"execution\_count": 7,

"outputs": []

},

{

"cell\_type": "markdown",

"source": [

"\*\*COMPILE THE MODEL ✈\*\*"

],

"metadata": {

"id": "S7fEAbEnC7u0"

}

},

{

"cell\_type": "code",

"source": [

"model.compile(optimizer='adam',loss='binary\_crossentropy',metrics=['accuracy'])"

],

"metadata": {

"id": "xKNjrrgNC1T-"

},

"execution\_count": 8,

"outputs": []

},

{

"cell\_type": "markdown",

"source": [

"\*\*FIT THE MODEL ⚡\*\*"

],

"metadata": {

"id": "gARkf8tRDGfn"

}

},

{

"cell\_type": "code",

"source": [

"r = model.fit(train\_dataset,\n",

" epochs = 10,\n",

" validation\_data = test\_dataset)"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "qXOnG9aWDL0B",

"outputId": "b5f89445-2806-4d89-c16c-54f44f6c3134"

},

"execution\_count": 9,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Epoch 1/10\n",

"1/1 [==============================] - 12s 12s/step - loss: 0.6967 - accuracy: 0.5500 - val\_loss: 0.5912 - val\_accuracy: 0.6000\n",

"Epoch 2/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.6388 - accuracy: 0.5500 - val\_loss: 0.9704 - val\_accuracy: 0.5000\n",

"Epoch 3/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.9761 - accuracy: 0.5000 - val\_loss: 0.6340 - val\_accuracy: 0.5000\n",

"Epoch 4/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.6478 - accuracy: 0.5000 - val\_loss: 0.6171 - val\_accuracy: 0.9500\n",

"Epoch 5/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.6174 - accuracy: 0.8500 - val\_loss: 0.6139 - val\_accuracy: 0.7500\n",

"Epoch 6/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.6048 - accuracy: 0.8000 - val\_loss: 0.5628 - val\_accuracy: 0.8500\n",

"Epoch 7/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.5632 - accuracy: 0.8500 - val\_loss: 0.4429 - val\_accuracy: 1.0000\n",

"Epoch 8/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.4820 - accuracy: 0.8500 - val\_loss: 0.2908 - val\_accuracy: 1.0000\n",

"Epoch 9/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.3741 - accuracy: 0.9000 - val\_loss: 0.1425 - val\_accuracy: 1.0000\n",

"Epoch 10/10\n",

"1/1 [==============================] - 2s 2s/step - loss: 0.2571 - accuracy: 0.9000 - val\_loss: 0.0666 - val\_accuracy: 1.0000\n"

]

}

]

},

{

"cell\_type": "markdown",

"source": [

"\*\*LOSS VS ITERATION GRAPH ⤴\*\*"

],

"metadata": {

"id": "kD2KcKiLEo\_k"

}

},

{

"cell\_type": "code",

"source": [

"import seaborn as plt\n",

"\n",

"plt.relplot(data=r.history, kind=\"line\")"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 387

},

"id": "YHzoKL\_REupV",

"outputId": "01a01820-4e6d-4631-dfdb-4a45be634231"

},

"execution\_count": 23,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"<seaborn.axisgrid.FacetGrid at 0x7fbea1851bd0>"

]

},

"metadata": {},

"execution\_count": 23

},

{

"output\_type": "display\_data",

"data": {

"text/plain": [

"<Figure size 461x360 with 1 Axes>"

],

"image/png": "\n"

},

"metadata": {

"needs\_background": "light"

}

}

]

},

{

"cell\_type": "markdown",

"source": [

"\*\*ACCURACY/ITERATION ⤴\*\*"

],

"metadata": {

"id": "OBVYXZMFGoN8"

}

},

{

"cell\_type": "code",

"source": [

"import seaborn as plt\n",

"\n",

"plt.relplot(data=[r.history[\"accuracy\"],r.history[\"val\_accuracy\"]], kind=\"line\")"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 387

},

"outputId": "169663a4-561c-4c4c-c053-386c6a36e0c2",

"id": "lKZQaOTrF0zQ"

},

"execution\_count": 30,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"<seaborn.axisgrid.FacetGrid at 0x7fbea1454c90>"

]

},

"metadata": {},

"execution\_count": 30

},

{

"output\_type": "display\_data",

"data": {

"text/plain": [

"<Figure size 402.375x360 with 1 Axes>"

],

"image/png": "\n"

},

"metadata": {

"needs\_background": "light"

}

}

]

},

{

"cell\_type": "markdown",

"source": [

"\*\*FUNCTION - RETURNS IMAGE WITH LABEL 🌴\*\*"

],

"metadata": {

"id": "k8RPTPTmG6V-"

}

},

{

"cell\_type": "code",

"source": [

"import matplotlib.pyplot as plt\n",

"import random\n",

"\n",

"def predictImage(filename, message = False):\n",

" img1 = image.load\_img(filename,target\_size=(150,150))\n",

" plt.imshow(img1)\n",

" Y = image.img\_to\_array(img1)\n",

" X = np.expand\_dims(Y,axis=0)\n",

" val = model.predict(X)\n",

" print(val)\n",

" if val == 0:\n",

" if message: sendMessage(\"Fire Detected\")\n",

" return plt.xlabel(\"Fire\",fontsize=30)\n",

" elif val == 1:\n",

" return plt.xlabel(\"No Fire\",fontsize=30)"

],

"metadata": {

"id": "90lqzBB1HFea"

},

"execution\_count": 110,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"def loadAndPredictImage(className):\n",

" local\_path = \"/content/drive/MyDrive/IBM/Dataset/test\_set/\" + className + \"/\"\n",

" files = []\n",

" for filename in os.listdir(local\_path):\n",

" if filename.endswith(\"jpg\"): \n",

" files.append(local\_path+filename)\n",

"\n",

" ran = random.choice(files)\n",

" predictImage(ran)"

],

"metadata": {

"id": "JfxnNChDHIjz"

},

"execution\_count": 88,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"loadAndPredictImage(\"Fire\")"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 338

},

"id": "eZqZgir9K04Y",

"outputId": "f2d8c8ce-33e7-4500-9804-c5a900f0c729"

},

"execution\_count": 91,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"1/1 [==============================] - 0s 43ms/step\n",

"[[0.]]\n"

]

},

{

"output\_type": "display\_data",

"data": {

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

],

"image/png": "\n"

},

"metadata": {

"needs\_background": "light"

}

}

]

},

{

"cell\_type": "markdown",

"source": [

"# \*\*VIDEO PROCESSING\*\*"

],

"metadata": {

"id": "X52nBh88GZkA"

}

},

{

"cell\_type": "code",

"source": [

"from google.colab.patches import cv2\_imshow\n",

"\n",

"#define video from file\n",

"video=cv2.VideoCapture(\"/content/Pexels Videos 2330708.mp4\") \n",

"#define the features \n",

"name=['forest','with fire']\n",

"\n",

"while(1):\n",

" success, frame= video.read() \n",

" cv2.imwrite(\"image.jpg\",frame)\n",

" img=image.load\_img(\"image.jpg\",target\_size=(150,150)) \n",

" x=image.img\_to\_array(img) \n",

" x=np.expand\_dims(x,axis=0)\n",

" pred=model.predict(x) \n",

" print(pred[0])\n",

" cv2.putText(frame,\"predicted class=\"+str(name[int(pred[0])]),(100,100),cv2.FONT\_HERSHEY\_SIMPLEX,1, (0,0,0), 1) \n",

" \n",

" if pred[0]==1:\n",

" # sendMessage(\"Forest Fire is detected, stay alert\")\n",

" print(message.sid)\n",

" print('Fire Detected') \n",

" print('SMS sent!')\n",

" else:\n",

" print('No Danger') \n",

"video.release() \n",

"cv2.destryoAllWindows()"

],

"metadata": {

"id": "LrFZbvydGcO2"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "markdown",

"source": [

"# \*\*TWILIO 📲\*\*"

],

"metadata": {

"id": "2b5beVCwL5Ou"

}

},

{

"cell\_type": "code",

"source": [

"!pip install twilio"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "VyocPfZzL5Cb",

"outputId": "0cf8ba6c-060e-4d05-9a3a-e5cf3ba5c134"

},

"execution\_count": 92,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/\n",

"Collecting twilio\n",

" Downloading twilio-7.15.2-py2.py3-none-any.whl (1.4 MB)\n",

"\u001b[K |████████████████████████████████| 1.4 MB 7.6 MB/s \n",

"\u001b[?25hRequirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from twilio) (2022.6)\n",

"Requirement already satisfied: requests>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from twilio) (2.23.0)\n",

"Collecting PyJWT<3.0.0,>=2.0.0\n",

" Downloading PyJWT-2.6.0-py3-none-any.whl (20 kB)\n",

"Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests>=2.0.0->twilio) (2022.9.24)\n",

"Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests>=2.0.0->twilio) (1.24.3)\n",

"Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests>=2.0.0->twilio) (2.10)\n",

"Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests>=2.0.0->twilio) (3.0.4)\n",

"Installing collected packages: PyJWT, twilio\n",

"Successfully installed PyJWT-2.6.0 twilio-7.15.2\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"numbers = [\"+919790194400\",\"+919788372297\"] # Array of numbers"

],

"metadata": {

"id": "qNMqRBKpQDDJ"

},

"execution\_count": 113,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"from twilio.rest import Client \n",

" \n",

"account\_sid = os.getenv(\"twilio\_sid\")\n",

"auth\_token = os.getenv(\"twilio\_auth\_token\") \n",

"client = Client(account\_sid, auth\_token) \n",

"\n",

"def sendMessage(message):\n",

" for i in numbers:\n",

" message = client.messages.create(messaging\_service\_sid=os.getenv(\"twilio\_messagin\_service\_sid\"), \n",

" body=message, \n",

" to=i, \n",

" ) \n",

" print(\"SENT TO :\",message.sid)"

],

"metadata": {

"id": "O0VoYPFyMLDC"

},

"execution\_count": 108,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"img = \"with fire (140).jpg\" # PASS THE IMAGE FILE HERE\n",

"predictImage(img, True)"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 391

},

"id": "eCUNr2OYRDKe",

"outputId": "f5eb9b6b-7fd9-4432-d2a0-2d1bc81a0911"

},

"execution\_count": 115,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"1/1 [==============================] - 0s 34ms/step\n",

"[[0.]]\n",

"SENT TO : SM2442749770969cfb343d45388efdf389\n",

"SENT TO : SM0b3b6e1f47130618d27fd10c635cfdb5\n"

]

},

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"Text(0.5, 0, 'Fire')"

]

},

"metadata": {},

"execution\_count": 115

},

{

"output\_type": "display\_data",

"data": {

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

],

"image/png": "\n"

},

"metadata": {

"needs\_background": "light"

}

}

]

}

]

}