VIDEO ANALYSIS

OPEN CV FOR VIDEO PROCESSING

Date	04 November 2022
Team ID	PNT2022TMID13480
Project Name	Emerging methods for the early detection of forest fires

Code:

```
{
  "cells": [
   {
     "cell_type": "code",
     "execution_count": 1,
     "metadata": {
       "id": "W8sBRTFDx9vh"
     },
     "outputs": [],
     "source": [
       "import keras\n",
       "from keras.preprocessing.image import ImageDataGenerator "
     ]
   },
   {
     "cell_type": "code",
     "execution_count": 2,
```

```json
    "metadata": {
      "id": "7gvCpfDH0MOD"
    },
    "outputs": [],
    "source": [
      "#Define the parameters/arguments for ImageDataGenerator class\n",
      "train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,rotation_range=180,zoom_range=0.2,horizontal_flip=True)\n",
      "\n",
      "test_datagen=ImageDataGenerator(rescale=1./255)"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 4,
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "id": "zTMlFh9K3XfS",
      "outputId": "63b623a6-ea8d-476f-e035-199a3991d566"
    },
    "outputs": [
      {
        "name": "stdout",
```

      "output_type": "stream",

      "text": [

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

      ]

     }

   ],

   "source": [

     "#Applying ImageDataGenerator functionality to trainset\n",

"x_train=train_datagen.flow_from_directory('/content/Dataset/Dataset/train_set',tar get_size=(128,128),batch_size=32,class_mode='binary')"

   ]

  },

  {

   "cell_type": "code",

   "execution_count": 5,

   "metadata": {

    "colab": {

     "base_uri": "https://localhost:8080/"

    },

    "id": "FyiVLj49OynB",

    "outputId": "cf37d6ba-5c44-4638-a712-c17fd073ae97"

   },

   "outputs": [

    {

     "name": "stdout",

      "output_type": "stream",
      "text": [
        "Found 121 images belonging to 2 classes.\n"
      ]
     }
    ],
    "source": [
      "#Applying ImageDataGenerator functionality to testset\n",

"x_test=test_datagen.flow_from_directory('/content/Dataset/Dataset/test_set',target_size=(128,128),batch_size=32,class_mode='binary')"

    ]
   },
   {
    "cell_type": "code",
    "execution_count": 6,
    "metadata": {
     "id": "mNBt4boEQ_EO"
    },
    "outputs": [],
    "source": [
     "#import model building libraries\n",
     "\n",
     "#To define Linear initialisation import Sequential\n",
     "from keras.models import Sequential\n",
     "#To add layers import Dense\n",

```
      "from keras.layers import Dense\n",
      "#To create Convolution kernel import Convolution2D\n",
      "from keras.layers import Convolution2D\n",
      "#import Maxpooling layer\n",
      "from keras.layers import MaxPooling2D\n",
      "#import flatten layer\n",
      "from keras.layers import Flatten\n",
      "import warnings\n",
      "warnings.filterwarnings('ignore')"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 7,
     "metadata": {
      "id": "WOxj9bU9-goY"
     },
     "outputs": [],
     "source": [
      "#initializing the model\n",
      "model=Sequential()"
     ]
    },
    {
     "cell_type": "code",
```

```
    "execution_count": 8,

    "metadata": {

     "id": "cADyODoXBAUl"

    },

    "outputs": [],

    "source": [

     "#add convolutional layer\n",

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

     "#add maxpooling layer\n",

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

     "#add flatten layer \n",

     "model.add(Flatten()) "

    ]

   },

   {

    "cell_type": "code",

    "execution_count": 9,

    "metadata": {

     "id": "C2mrC6T6Bk8u"

    },

    "outputs": [],

    "source": [

     "#add hidden layer\n",

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

```
      "#add output layer\n",
      "model.add(Dense(1,activation='sigmoid'))"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 10,
    "metadata": {
      "id": "pa8MgIjFGrVp"
    },
    "outputs": [],
    "source": [
      "#configure the learning process\n",

"model.compile(loss='binary_crossentropy',optimizer=\"adam\",metrics=[\"accuracy\"])"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 12,
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "id": "-TmR_z4-IGlo",
```

    "outputId": "776ee68c-1c77-4737-80c9-e86b57b51c37"

  },

  "outputs": [

   {

     "name": "stdout",

     "output_type": "stream",

     "text": [

       "Epoch 1/10\n",

       "14/14 [==============================] - 26s 2s/step - loss:
0.1884 - accuracy: 0.9128 - val_loss: 0.0690 - val_accuracy: 0.9669\n",

       "Epoch 2/10\n",

       "14/14 [==============================] - 25s 2s/step - loss:
0.2131 - accuracy: 0.8830 - val_loss: 0.0923 - val_accuracy: 0.9504\n",

       "Epoch 3/10\n",

       "14/14 [==============================] - 25s 2s/step - loss:
0.1947 - accuracy: 0.9151 - val_loss: 0.0740 - val_accuracy: 0.9587\n",

       "Epoch 4/10\n",

       "14/14 [==============================] - 25s 2s/step - loss:
0.1663 - accuracy: 0.9312 - val_loss: 0.0698 - val_accuracy: 0.9752\n",

       "Epoch 5/10\n",

       "14/14 [==============================] - 26s 2s/step - loss:
0.1668 - accuracy: 0.9404 - val_loss: 0.0611 - val_accuracy: 0.9835\n",

       "Epoch 6/10\n",

       "14/14 [==============================] - 25s 2s/step - loss:
0.1840 - accuracy: 0.9151 - val_loss: 0.0641 - val_accuracy: 0.9752\n",

       "Epoch 7/10\n",

       "14/14 [==============================] - 25s 2s/step - loss:
0.2018 - accuracy: 0.9128 - val_loss: 0.0846 - val_accuracy: 0.9752\n",

      "Epoch 8/10\n",

      "14/14 [==============================] - 25s 2s/step - loss: 0.1943 - accuracy: 0.9106 - val_loss: 0.0665 - val_accuracy: 0.9752\n",

      "Epoch 9/10\n",

      "14/14 [==============================] - 25s 2s/step - loss: 0.1984 - accuracy: 0.9151 - val_loss: 0.0715 - val_accuracy: 0.9669\n",

      "Epoch 10/10\n",

      "14/14 [==============================] - 26s 2s/step - loss: 0.1742 - accuracy: 0.9243 - val_loss: 0.0627 - val_accuracy: 0.9752\n"
     ]
    },
    {
     "data": {
      "text/plain": [
       "<keras.callbacks.History at 0x7f04f32388d0>"
      ]
     },
     "execution_count": 12,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "#Training the model\n",

"model.fit_generator(x_train,steps_per_epoch=14,epochs=10,validation_data=x_test,validation_steps=4)"

```
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 13,
  "metadata": {
    "id": "wy0ybBWnL5Jb"
  },
  "outputs": [],
  "source": [
    "model.save(\"forest1.h5\")"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 14,
  "metadata": {
    "id": "d8dYcGPqoEne"
  },
  "outputs": [],
  "source": [
    "#import load_model from keras.model\n",
    "from keras.models import load_model\n",
    "#import image class from keras\n",
    "from tensorflow.keras.preprocessing import image\n",
```

```
    "#import numpy\n",
    "import numpy as np\n",
    "#import cv2\n",
    "import cv2\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 15,
   "metadata": {
    "id": "Zkq9A29zpkm1"
   },
   "outputs": [],
   "source": [
    "#load the saved model\n",
    "model = load_model(\"forest1.h5\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 16,
   "metadata": {
    "id": "0wPII3sMps3A"
   },
   "outputs": [],
```

```
    "source": [
      "img=image.load_img('/content/Dataset/Dataset/test_set/with
fire/180802_CarrFire_010_large_700x467.jpg')\n",
      "x=image.img_to_array(img)\n",
      "res = cv2.resize(x, dsize=(128, 128), interpolation=cv2.INTER_CUBIC)\n",
      "#expand the image shape\n",
      "x=np.expand_dims(res,axis=0)"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 17,
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "id": "_X7xSZkHr6Ef",
      "outputId": "265c3e8d-9c96-4e1f-980a-dd4ce2c32933"
    },
    "outputs": [
      {
        "name": "stdout",
        "output_type": "stream",
        "text": [
          "1/1 [==============================] - 0s 142ms/step\n"
        ]
```

```json
      }
    ],
    "source": [
      "pred=model.predict(x)"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 18,
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "id": "FqvlyDfrtIMk",
      "outputId": "c61d1d16-36a0-4324-d713-eaad9b9f8540"
    },
    "outputs": [
      {
        "data": {
          "text/plain": [
            "array([[1.]], dtype=float32)"
          ]
        },
        "execution_count": 18,
        "metadata": {},
```

```
      "output_type": "execute_result"
     }
    ],
    "source": [
     "pred"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 23,
    "metadata": {
     "colab": {
      "base_uri": "https://localhost:8080/"
     },
     "id": "sP_TqPewf9yW",
     "outputId": "4211ea41-5651-4427-9b2d-b580e8e40541"
    },
    "outputs": [
     {
      "name": "stdout",
      "output_type": "stream",
      "text": [
       "Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/\n",
       "Collecting twilio\n",
       "  Downloading twilio-7.15.0-py2.py3-none-any.whl (1.4 MB)\n",
```

          "\u001b[K |████████████████████████████████| 1.4
MB 5.3 MB/s \n",

          "\u001b[?25hRequirement 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: pytz in /usr/local/lib/python3.7/dist-
packages (from twilio) (2022.5)\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",

          "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",

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

          "Successfully installed PyJWT-2.6.0 twilio-7.15.0\n"

       ]

     }

   ],

   "source": [

     "pip install twilio"

   ]

 },

 {

   "cell_type": "code",

"execution_count": 28,

"metadata": {

"colab": {

"base_uri": "https://localhost:8080/"

},

"id": "FhN5NyRbgKMd",

"outputId": "2d331b6b-e2b7-41b4-997c-b8142ad124f4"

},

"outputs": [

{

"name": "stdout",

"output_type": "stream",

"text": [

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

"Requirement already satisfied: pygobject in /usr/lib/python3/dist-packages (3.26.1)\n"

]

}

],

"source": [

"pip install pygobject"

]

},

{

"cell_type": "code",

```
      "execution_count": 35,
      "metadata": {
        "id": "rn0FqT0YVV98"
      },
      "outputs": [],
      "source": [
        "#import opencv library\n",
        "import cv2\n",
        "#import numpy\n",
        "import numpy as np\n",
        "#import image function from keras\n",
        "from tensorflow.keras.preprocessing import image\n",
        "#import load_model from keras\n",
        "from keras.models import load_model\n",
        "#import Client from twilio API\n",
        "from twilio.rest import Client\n",
        "#import playsound package\n",
        "from playsound import playsound"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": null,
      "metadata": {
        "colab": {
```

```json
      "base_uri": "https://localhost:8080/",
       "height": 235
     },
    "id": "ZdDJ4NOISyUy",
    "outputId": "de88c80e-3dee-4598-9537-ac93d09032a3"
   },
   "outputs": [],
   "source": [
     "#load the saved model\n",
     "model=load_model('forest1.h5')\n",
     "#define video\n",
     "video=cv2.VideoCapture(0)\n",
     "#define the features\n",
     "name=['forest','with fire']\n",
     "\n"
    ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
     "id": "8MFj3cXkbP2z"
   },
   "outputs": [],
   "source": []
```

```json
    }
  ],
  "metadata": {
    "colab": {
      "collapsed_sections": [],
      "provenance": []
    },
    "kernelspec": {
      "display_name": "Python 3",
      "name": "python3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "nbformat": 4,
  "nbformat_minor": 0
}
```