{

"cells": [

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"colab": {

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

},

"id": "ynUOT9Yc6fQk",

"outputId": "df1a945a-f917-4156-8b99-18c07da26956"

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Mounted at /content/drive\n"

]

}

],

"source": [

"from google.colab import drive\n",

"drive.mount('/content/drive')"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"colab": {

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

},

"id": "NhP7V93wMVho",

"outputId": "35b72ad9-1b52-42f9-e2d5-7912c02d6bd0"

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"\u001b[0m\u001b[01;34mdrive\u001b[0m/ \u001b[01;34msample\_data\u001b[0m/\n"

]

}

],

"source": [

"ls"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"colab": {

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

},

"id": "\_VuPfwOuQYAm",

"outputId": "1c35e7ef-399f-4dcc-c090-946534fa802f"

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"/content/drive/.shortcut-targets-by-id/1LL5lvl6AsdVwW9LWVu\_GXEUCoV7jYm-c/Dataset\n"

]

}

],

"source": [

"cd//content/drive/MyDrive/Colab Notebooks/Dataset\n"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "kdGnCU-FQz8C",

"colab": {

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

},

"outputId": "0444b204-4ffd-4b90-e1a0-9508410b2e55"

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"IBM\_review.pptx photo-1589820296156-2454bb8a6ad1.jpg \u001b[0m\u001b[01;34mTRAIN\_SET\u001b[0m/\n",

"nutrition.h5 \u001b[01;34mTEST\_SET\u001b[0m/\n"

]

}

],

"source": [

"ls"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "whbZ5Uw35pHw"

},

"source": [

"### Importing Neccessary Libraries"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "60eg6zmo5pHx"

},

"outputs": [],

"source": [

"import numpy as np#used for numerical analysis\n",

"import tensorflow #open source used for both ML and DL for computation\n",

"from tensorflow.keras.models import Sequential #it is a plain stack of layers\n",

"from tensorflow.keras import layers #A layer consists of a tensor-in tensor-out computation function\n",

"#Dense layer is the regular deeply connected neural network layer\n",

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

"#Faltten-used fot flattening the input or change the dimension\n",

"from tensorflow.keras.layers import Conv2D,MaxPooling2D,Dropout #Convolutional layer\n",

"#MaxPooling2D-for downsampling the image\n",

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

"\n",

"\n"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "vnVt93M05pH0"

},

"source": [

"### Image Data Agumentation"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "-VLZKCTd5pH1"

},

"outputs": [],

"source": [

"#setting parameter for Image Data agumentation to the training data\n",

"train\_datagen = ImageDataGenerator(rescale=1./255,shear\_range=0.2,zoom\_range=0.2,horizontal\_flip=True)\n",

"#Image Data agumentation to the testing data\n",

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

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "kpsHveuq5pH4"

},

"source": [

"### Loading our data and performing data agumentation"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"colab": {

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

},

"id": "8Hkc9ffd5pH5",

"outputId": "e33f9587-33e3-44a0-8841-843d9f55a254"

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Found 4138 images belonging to 5 classes.\n",

"Found 929 images belonging to 3 classes.\n"

]

}

],

"source": [

"#performing data agumentation to train data\n",

"x\_train = train\_datagen.flow\_from\_directory(\n",

" r'/content/drive/MyDrive/Colab Notebooks/Dataset/TRAIN\_SET',\n",

" target\_size=(64, 64),batch\_size=5,color\_mode='rgb',class\_mode='sparse')\n",

"#performing data agumentation to test data\n",

"x\_test = test\_datagen.flow\_from\_directory(\n",

" r'/content/drive/MyDrive/Colab Notebooks/Dataset/TEST\_SET',\n",

" target\_size=(64, 64),batch\_size=5,color\_mode='rgb',class\_mode='sparse') "

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"colab": {

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

},

"id": "szwYFmls5pH8",

"outputId": "b2301196-ebac-4c81-f97f-a4d10d78dec2"

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}\n"

]

}

],

"source": [

"print(x\_train.class\_indices)#checking the number of classes"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"colab": {

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

},

"id": "8SliKn605pH-",

"outputId": "61c57cec-a36f-4aa8-f56f-fe426dfc2944"

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2}\n"

]

}

],

"source": [

"print(x\_test.class\_indices)#checking the number of classes"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"colab": {

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

},

"id": "yWWDoRDw5pIA",

"outputId": "298cbdd8-3d86-4682-9aa1-c8502f57c3da"

},

"outputs": [

{

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

"data": {

"text/plain": [

"Counter({0: 995, 1: 1374, 2: 1019, 3: 275, 4: 475})"

]

},

"metadata": {},

"execution\_count": 12

}

],

"source": [

"from collections import Counter as c\n",

"c(x\_train .labels)"

]

},

{

"cell\_type": "code",

"source": [

"# Initializing the CNN\n",

"classifier = Sequential()\n",

"\n",

"# First convolution layer and pooling\n",

"classifier.add(Conv2D(32, (3, 3), input\_shape=(64, 64, 3), activation='relu'))\n",

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

"\n",

"# Second convolution layer and pooling\n",

"classifier.add(Conv2D(32, (3, 3), activation='relu'))\n",

"\n",

"# input\_shape is going to be the pooled feature maps from the previous convolution layer\n",

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

"\n",

"# Flattening the layers\n",

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

"\n",

"# Adding a fully connected layer\n",

"classifier.add(Dense(units=128, activation='relu'))\n",

"classifier.add(Dense(units=5, activation='softmax')) # softmax for more than 2\n",

"\n"

],

"metadata": {

"id": "7U7d0yvp1HEO"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"classifier.summary()#summary of our model"

],

"metadata": {

"colab": {

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

},

"id": "WvrA--WP1MFZ",

"outputId": "0a3474f4-6517-493d-c5d0-ea31c45b4d8e"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Model: \"sequential\"\n",

"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n",

" Layer (type) Output Shape Param # \n",

"=================================================================\n",

" conv2d (Conv2D) (None, 62, 62, 32) 896 \n",

" \n",

" max\_pooling2d (MaxPooling2D (None, 31, 31, 32) 0 \n",

" ) \n",

" \n",

" conv2d\_1 (Conv2D) (None, 29, 29, 32) 9248 \n",

" \n",

" max\_pooling2d\_1 (MaxPooling (None, 14, 14, 32) 0 \n",

" 2D) \n",

" \n",

" flatten (Flatten) (None, 6272) 0 \n",

" \n",

" dense (Dense) (None, 128) 802944 \n",

" \n",

" dense\_1 (Dense) (None, 5) 645 \n",

" \n",

"=================================================================\n",

"Total params: 813,733\n",

"Trainable params: 813,733\n",

"Non-trainable params: 0\n",

"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"# Compiling the CNN\n",

"# categorical\_crossentropy for more than 2\n",

"classifier.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy']) "

],

"metadata": {

"id": "j6ekSlkp1Nr-"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"classifier.fit\_generator(\n",

" generator=x\_train,steps\_per\_epoch = len(x\_train),\n",

" epochs=10, validation\_data=x\_test,validation\_steps = len(x\_test))# No of images in test set"

],

"metadata": {

"colab": {

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

},

"id": "8YkGfW7e1V4q",

"outputId": "ca0f5698-b658-4e53-ff8b-6214a6abef2d"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "stream",

"name": "stderr",

"text": [

"/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:3: UserWarning: `Model.fit\_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.\n",

" This is separate from the ipykernel package so we can avoid doing imports until\n"

]

},

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Epoch 1/10\n",

"828/828 [==============================] - 1189s 1s/step - loss: 0.5894 - accuracy: 0.7748 - val\_loss: 0.5930 - val\_accuracy: 0.7427\n",

"Epoch 2/10\n",

"828/828 [==============================] - 29s 35ms/step - loss: 0.4094 - accuracy: 0.8371 - val\_loss: 0.5117 - val\_accuracy: 0.8159\n",

"Epoch 3/10\n",

"828/828 [==============================] - 27s 33ms/step - loss: 0.3728 - accuracy: 0.8586 - val\_loss: 0.3814 - val\_accuracy: 0.8558\n",

"Epoch 4/10\n",

"828/828 [==============================] - 29s 35ms/step - loss: 0.3465 - accuracy: 0.8664 - val\_loss: 0.4036 - val\_accuracy: 0.8525\n",

"Epoch 5/10\n",

"828/828 [==============================] - 28s 33ms/step - loss: 0.3175 - accuracy: 0.8797 - val\_loss: 0.4061 - val\_accuracy: 0.8428\n",

"Epoch 6/10\n",

"828/828 [==============================] - 29s 35ms/step - loss: 0.3020 - accuracy: 0.8896 - val\_loss: 0.3806 - val\_accuracy: 0.8558\n",

"Epoch 7/10\n",

"828/828 [==============================] - 30s 36ms/step - loss: 0.2848 - accuracy: 0.8888 - val\_loss: 0.4778 - val\_accuracy: 0.8041\n",

"Epoch 8/10\n",

"828/828 [==============================] - 29s 35ms/step - loss: 0.2673 - accuracy: 0.8980 - val\_loss: 0.4117 - val\_accuracy: 0.8385\n",

"Epoch 9/10\n",

"828/828 [==============================] - 31s 37ms/step - loss: 0.2485 - accuracy: 0.9065 - val\_loss: 0.3935 - val\_accuracy: 0.8611\n",

"Epoch 10/10\n",

"828/828 [==============================] - 28s 34ms/step - loss: 0.2399 - accuracy: 0.9120 - val\_loss: 0.4292 - val\_accuracy: 0.8525\n"

]

},

{

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

"data": {

"text/plain": [

"<keras.callbacks.History at 0x7f40e2afcf10>"

]

},

"metadata": {},

"execution\_count": 16

}

]

},

{

"cell\_type": "code",

"source": [

"# Save the model\n",

"classifier.save('nutrition.h5')"

],

"metadata": {

"id": "Qe2POBdU7akr"

},

"execution\_count": 40,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"from tensorflow.keras.models import load\_model\n",

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

"import numpy as np"

],

"metadata": {

"id": "XnTA5ZZ68Lv3"

},

"execution\_count": 27,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"img = image.load\_img(\"/content/drive/MyDrive/Colab Notebooks/Dataset/TRAIN\_SET/APPLES/n07740461\_10067.jpg\",target\_size= (64,64))#loading of the image\n",

"img"

],

"metadata": {

"colab": {

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

"height": 81

},

"id": "rgFMwRuH8Pnp",

"outputId": "d018abe7-7002-44a3-aa15-7101bd60721c"

},

"execution\_count": 28,

"outputs": [

{

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

"data": {

"text/plain": [

"<PIL.Image.Image image mode=RGB size=64x64 at 0x7F40DD08AA50>"

],

"image/png": "\n"

},

"metadata": {},

"execution\_count": 28

}

]

},

{

"cell\_type": "code",

"source": [

"x=image.img\_to\_array(img)#conversion image into array"

],

"metadata": {

"id": "vndVbu0q9O4P"

},

"execution\_count": 31,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"x"

],

"metadata": {

"colab": {

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

},

"id": "1u8piYkF9b9p",

"outputId": "83c2f78b-c54d-436f-eb3c-2712ae4799e8"

},

"execution\_count": 32,

"outputs": [

{

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

"data": {

"text/plain": [

"array([[[255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" ...,\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.]],\n",

"\n",

" [[255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" ...,\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.]],\n",

"\n",

" [[255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" ...,\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.]],\n",

"\n",

" ...,\n",

"\n",

" [[255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" ...,\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.]],\n",

"\n",

" [[255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" ...,\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.]],\n",

"\n",

" [[255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" ...,\n",

" [255., 255., 255.],\n",

" [255., 255., 255.],\n",

" [255., 255., 255.]]], dtype=float32)"

]

},

"metadata": {},

"execution\_count": 32

}

]

},

{

"cell\_type": "code",

"source": [

"x.ndim"

],

"metadata": {

"colab": {

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

},

"id": "yaVVqp4F9WTB",

"outputId": "3d2bdf0a-4754-4112-dae1-2d4b832eca36"

},

"execution\_count": 30,

"outputs": [

{

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

"data": {

"text/plain": [

"3"

]

},

"metadata": {},

"execution\_count": 30

}

]

},

{

"cell\_type": "code",

"source": [

"x=np.expand\_dims(x,axis=0) #expand the dimension"

],

"metadata": {

"id": "\_ksKd6z09WaM"

},

"execution\_count": 33,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"x.ndim"

],

"metadata": {

"colab": {

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

},

"id": "7H5wVKTE9Wgy",

"outputId": "1086f9fa-4f63-40b6-8d0a-dbae88a4bfc1"

},

"execution\_count": 34,

"outputs": [

{

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

"data": {

"text/plain": [

"4"

]

},

"metadata": {},

"execution\_count": 34

}

]

},

{

"cell\_type": "code",

"source": [

"pred = classifier.predict(x)"

],

"metadata": {

"colab": {

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

},

"id": "UPDDmAKg9Wmh",

"outputId": "52c35432-7500-4965-a7b1-910301bfb14f"

},

"execution\_count": 35,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"1/1 [==============================] - 0s 79ms/step\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"pred"

],

"metadata": {

"colab": {

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

},

"id": "\_7ZC16gb9q02",

"outputId": "599c97d3-5173-4a26-fc93-06c07c1672db"

},

"execution\_count": 36,

"outputs": [

{

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

"data": {

"text/plain": [

"array([[1., 0., 0., 0., 0.]], dtype=float32)"

]

},

"metadata": {},

"execution\_count": 36

}

]

},

{

"cell\_type": "code",

"source": [

"labels=['APPLES', 'BANANA', 'ORANGE','PINEAPPLE','WATERMELON']\n",

"labels[np.argmax(pred)]"

],

"metadata": {

"colab": {

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

"height": 36

},

"id": "F4SkklKo9q9n",

"outputId": "3708e2cb-81cb-48de-d729-db60097ce171"

},

"execution\_count": 37,

"outputs": [

{

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

"data": {

"text/plain": [

"'APPLES'"

],

"application/vnd.google.colaboratory.intrinsic+json": {

"type": "string"

}

},

"metadata": {},

"execution\_count": 37

}

]

}

],

"metadata": {

"colab": {

"collapsed\_sections": [],

"provenance": []

},

"kernelspec": {

"display\_name": "Python 3",

"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": "3.8.3"

}

},

"nbformat": 4,

"nbformat\_minor": 0

}