{

"nbformat": 4,

"nbformat\_minor": 0,

"metadata": {

"colab": {

"provenance": [],

"gpuType": "T4"

},

"kernelspec": {

"name": "python3",

"display\_name": "Python 3"

},

"language\_info": {

"name": "python"

},

"accelerator": "GPU"

},

"cells": [

{

"cell\_type": "code",

"execution\_count": 1,

"metadata": {

"colab": {

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

},

"id": "vUwkZf2bjqPb",

"outputId": "9dc57777-c55f-44b8-ec40-bcb57776d88a"

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Path to dataset files: /kaggle/input/the-wildfire-dataset\n"

]

}

],

"source": [

"import kagglehub\n",

"\n",

"# Download latest version\n",

"path = kagglehub.dataset\_download(\"elmadafri/the-wildfire-dataset\")\n",

"\n",

"print(\"Path to dataset files:\", path)"

]

},

{

"cell\_type": "code",

"source": [

"# Importing necessary libraries\n",

"import os\n",

"import numpy as np\n",

"import matplotlib.pyplot as plt\n",

"import tensorflow as tf\n",

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

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

"from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, Input"

],

"metadata": {

"id": "GkqLuDexkgOI"

},

"execution\_count": 2,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"# TO ensure the GPU is present\n",

"physical\_devices = tf.config.experimental.list\_physical\_devices('GPU')\n",

"if len(physical\_devices) > 0:\n",

" tf.config.experimental.set\_memory\_growth(physical\_devices[0], True)\n",

" print('GPU is Available!')\n",

"else:\n",

" print('GPU is Unavailable!')"

],

"metadata": {

"colab": {

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

},

"id": "luezXzibma6T",

"outputId": "7d863a0b-d1d3-4110-acc4-fe57949f07a0"

},

"execution\_count": 3,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"GPU is Available!\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"# Load and explore the dataset\n",

"train\_dir = '/root/.cache/kagglehub/datasets/elmadafri/the-wildfire-dataset/versions/3/the\_wildfire\_dataset\_2n\_version/train'\n",

"val\_dir = '/root/.cache/kagglehub/datasets/elmadafri/the-wildfire-dataset/versions/3/the\_wildfire\_dataset\_2n\_version/val'\n",

"test\_dir = '/root/.cache/kagglehub/datasets/elmadafri/the-wildfire-dataset/versions/3/the\_wildfire\_dataset\_2n\_version/test'"

],

"metadata": {

"id": "31b12RAXnYBq"

},

"execution\_count": 4,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"# List all the classes\n",

"classes = os.listdir(train\_dir)\n",

"num\_classes = len(classes)\n",

"\n",

"# Display the class names\n",

"print(f'Number of Classes: {num\_classes}')\n",

"print(f'Classes: {classes}')"

],

"metadata": {

"colab": {

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

},

"id": "micmxs0en-vp",

"outputId": "fe9e20af-75b6-4a96-a1a4-2ac1d45f9b19"

},

"execution\_count": 5,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Number of Classes: 2\n",

"Classes: ['fire', 'nofire']\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"# Let's visualize the images in the dataset\n",

"plt.figure(figsize=(12, 10))\n",

"for i in range(5):\n",

" class\_path = os.path.join(train\_dir, classes[0])\n",

" img\_name = os.listdir(class\_path)[i]\n",

" img\_path = os.path.join(class\_path, img\_name)\n",

" img = plt.imread(img\_path)\n",

"\n",

" plt.subplot(1, 5, i+1)\n",

" plt.imshow(img)\n",

" plt.title(f'{classes[0]} \\n shape: {img.shape}')\n",

" plt.axis('off')\n",

"plt.show()"

],

"metadata": {

"colab": {

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

"height": 200

},

"id": "TE35M2ZWobj1",

"outputId": "cf64f46f-43bc-413f-e7ac-62b12b2e36a4"

},

"execution\_count": 6,

"outputs": [

{

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

"data": {

"text/plain": [

"<Figure size 1200x1000 with 5 Axes>"

],

"image/png": "\n"

},

"metadata": {}

}

]

},

{

"cell\_type": "code",

"source": [

"plt.figure(figsize=(12, 10))\n",

"for i in range(5):\n",

" class\_path = os.path.join(train\_dir, classes[1])\n",

" img\_name = os.listdir(class\_path)[i]\n",

" img\_path = os.path.join(class\_path, img\_name)\n",

" img = plt.imread(img\_path)\n",

"\n",

" plt.subplot(1, 5, i+1)\n",

" plt.imshow(img)\n",

" plt.title(f'{classes[1]} \\n shape: {img.shape}')\n",

" plt.axis('off')\n",

"plt.show()"

],

"metadata": {

"colab": {

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

"height": 320

},

"id": "U4jjfWG8pq81",

"outputId": "2c71262c-e6e6-460b-c6a2-e5296ceea41e"

},

"execution\_count": 7,

"outputs": [

{

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

"data": {

"text/plain": [

"<Figure size 1200x1000 with 5 Axes>"

],

"image/png": "\n"

},

"metadata": {}

}

]

},

{

"cell\_type": "code",

"source": [

"# Preprocessing\n",

"# Image dimensions and batch size\n",

"img\_width, img\_height = 150, 150\n",

"batch\_size = 32 # there will be 32 images in a batch!\n",

"\n",

"# Data generators\n",

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

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

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

"\n",

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

" train\_dir,\n",

" target\_size = (img\_width, img\_height),\n",

" batch\_size = batch\_size,\n",

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

" shuffle = True\n",

")\n",

"\n",

"val\_generator = val\_datagen.flow\_from\_directory(\n",

" val\_dir,\n",

" target\_size = (img\_width, img\_height),\n",

" batch\_size = batch\_size,\n",

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

" shuffle = True\n",

")\n",

"\n",

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

" test\_dir,\n",

" target\_size = (img\_width, img\_height),\n",

" batch\_size = batch\_size,\n",

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

" shuffle = True\n",

")"

],

"metadata": {

"colab": {

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

},

"id": "2asrofIGdO-O",

"outputId": "90220a0a-166f-4ce1-d56e-21ff688cd097"

},

"execution\_count": 8,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Found 1887 images belonging to 2 classes.\n",

"Found 402 images belonging to 2 classes.\n",

"Found 410 images belonging to 2 classes.\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"# map the indices\n",

"class\_mapping = train\_generator.class\_indices\n",

"# print(class\_mapping)\n",

"# Extract the class names\n",

"class\_names = list(class\_mapping.keys())\n",

"print(\"Class Names:\", class\_names)"

],

"metadata": {

"colab": {

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

},

"id": "9eaPD4OhgKdP",

"outputId": "82ed877d-3de1-46aa-ed7a-bb68e3d7b32b"

},

"execution\_count": 9,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Class Names: ['fire', 'nofire']\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"# Let's build the CNN Model\n",

"model = Sequential([\n",

" Input(shape=(img\_width, img\_height, 3)),\n",

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

" MaxPooling2D(pool\_size=(2,2)),\n",

"\n",

" Conv2D(64, (3, 3), activation='relu'),\n",

" MaxPooling2D(pool\_size=(2, 2)),\n",

"\n",

" Conv2D(128, (3, 3), activation='relu'),\n",

" MaxPooling2D(pool\_size=(2, 2)),\n",

"\n",

" Flatten(),\n",

" Dense(512, activation='relu'),\n",

" Dropout(0.5),\n",

" Dense(1, activation='sigmoid')\n",

"]\n",

")"

],

"metadata": {

"id": "nhTuXuW2gUQK"

},

"execution\_count": 10,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"model.compile(optimizer='adam',\n",

" loss='binary\_crossentropy',\n",

" metrics=['accuracy'])\n",

"model.summary()"

],

"metadata": {

"colab": {

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

"height": 449

},

"id": "VAR7uR21dMl0",

"outputId": "2955f75f-859a-44f4-b537-985951546024"

},

"execution\_count": 11,

"outputs": [

{

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

"data": {

"text/plain": [

"\u001b[1mModel: \"sequential\"\u001b[0m\n"

],

"text/html": [

"<pre style=\"white-space:pre;overflow-x:auto;line-height:normal;font-family:Menlo,'DejaVu Sans Mono',consolas,'Courier New',monospace\"><span style=\"font-weight: bold\">Model: \"sequential\"</span>\n",

"</pre>\n"

]

},

"metadata": {}

},

{

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

"data": {

"text/plain": [

"┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓\n",

"┃\u001b[1m \u001b[0m\u001b[1mLayer (type) \u001b[0m\u001b[1m \u001b[0m┃\u001b[1m \u001b[0m\u001b[1mOutput Shape \u001b[0m\u001b[1m \u001b[0m┃\u001b[1m \u001b[0m\u001b[1m Param #\u001b[0m\u001b[1m \u001b[0m┃\n",

"┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩\n",

"│ conv2d (\u001b[38;5;33mConv2D\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m148\u001b[0m, \u001b[38;5;34m148\u001b[0m, \u001b[38;5;34m32\u001b[0m) │ \u001b[38;5;34m896\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ max\_pooling2d (\u001b[38;5;33mMaxPooling2D\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m74\u001b[0m, \u001b[38;5;34m74\u001b[0m, \u001b[38;5;34m32\u001b[0m) │ \u001b[38;5;34m0\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ conv2d\_1 (\u001b[38;5;33mConv2D\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m72\u001b[0m, \u001b[38;5;34m72\u001b[0m, \u001b[38;5;34m64\u001b[0m) │ \u001b[38;5;34m18,496\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ max\_pooling2d\_1 (\u001b[38;5;33mMaxPooling2D\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m36\u001b[0m, \u001b[38;5;34m36\u001b[0m, \u001b[38;5;34m64\u001b[0m) │ \u001b[38;5;34m0\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ conv2d\_2 (\u001b[38;5;33mConv2D\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m34\u001b[0m, \u001b[38;5;34m34\u001b[0m, \u001b[38;5;34m128\u001b[0m) │ \u001b[38;5;34m73,856\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ max\_pooling2d\_2 (\u001b[38;5;33mMaxPooling2D\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m17\u001b[0m, \u001b[38;5;34m17\u001b[0m, \u001b[38;5;34m128\u001b[0m) │ \u001b[38;5;34m0\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ flatten (\u001b[38;5;33mFlatten\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m36992\u001b[0m) │ \u001b[38;5;34m0\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ dense (\u001b[38;5;33mDense\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m512\u001b[0m) │ \u001b[38;5;34m18,940,416\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ dropout (\u001b[38;5;33mDropout\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m512\u001b[0m) │ \u001b[38;5;34m0\u001b[0m │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ dense\_1 (\u001b[38;5;33mDense\u001b[0m) │ (\u001b[38;5;45mNone\u001b[0m, \u001b[38;5;34m1\u001b[0m) │ \u001b[38;5;34m513\u001b[0m │\n",

"└─────────────────────────────────┴────────────────────────┴───────────────┘\n"

],

"text/html": [

"<pre style=\"white-space:pre;overflow-x:auto;line-height:normal;font-family:Menlo,'DejaVu Sans Mono',consolas,'Courier New',monospace\">┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓\n",

"┃<span style=\"font-weight: bold\"> Layer (type) </span>┃<span style=\"font-weight: bold\"> Output Shape </span>┃<span style=\"font-weight: bold\"> Param # </span>┃\n",

"┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩\n",

"│ conv2d (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">Conv2D</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">148</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">148</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">32</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">896</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ max\_pooling2d (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">MaxPooling2D</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">74</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">74</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">32</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">0</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ conv2d\_1 (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">Conv2D</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">72</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">72</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">64</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">18,496</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ max\_pooling2d\_1 (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">MaxPooling2D</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">36</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">36</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">64</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">0</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ conv2d\_2 (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">Conv2D</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">34</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">34</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">128</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">73,856</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ max\_pooling2d\_2 (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">MaxPooling2D</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">17</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">17</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">128</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">0</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ flatten (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">Flatten</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">36992</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">0</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ dense (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">Dense</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">512</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">18,940,416</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ dropout (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">Dropout</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">512</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">0</span> │\n",

"├─────────────────────────────────┼────────────────────────┼───────────────┤\n",

"│ dense\_1 (<span style=\"color: #0087ff; text-decoration-color: #0087ff\">Dense</span>) │ (<span style=\"color: #00d7ff; text-decoration-color: #00d7ff\">None</span>, <span style=\"color: #00af00; text-decoration-color: #00af00\">1</span>) │ <span style=\"color: #00af00; text-decoration-color: #00af00\">513</span> │\n",

"└─────────────────────────────────┴────────────────────────┴───────────────┘\n",

"</pre>\n"

]

},

"metadata": {}

},

{

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

"data": {

"text/plain": [

"\u001b[1m Total params: \u001b[0m\u001b[38;5;34m19,034,177\u001b[0m (72.61 MB)\n"

],

"text/html": [

"<pre style=\"white-space:pre;overflow-x:auto;line-height:normal;font-family:Menlo,'DejaVu Sans Mono',consolas,'Courier New',monospace\"><span style=\"font-weight: bold\"> Total params: </span><span style=\"color: #00af00; text-decoration-color: #00af00\">19,034,177</span> (72.61 MB)\n",

"</pre>\n"

]

},

"metadata": {}

},

{

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

"data": {

"text/plain": [

"\u001b[1m Trainable params: \u001b[0m\u001b[38;5;34m19,034,177\u001b[0m (72.61 MB)\n"

],

"text/html": [

"<pre style=\"white-space:pre;overflow-x:auto;line-height:normal;font-family:Menlo,'DejaVu Sans Mono',consolas,'Courier New',monospace\"><span style=\"font-weight: bold\"> Trainable params: </span><span style=\"color: #00af00; text-decoration-color: #00af00\">19,034,177</span> (72.61 MB)\n",

"</pre>\n"

]

},

"metadata": {}

},

{

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

"data": {

"text/plain": [

"\u001b[1m Non-trainable params: \u001b[0m\u001b[38;5;34m0\u001b[0m (0.00 B)\n"

],

"text/html": [

"<pre style=\"white-space:pre;overflow-x:auto;line-height:normal;font-family:Menlo,'DejaVu Sans Mono',consolas,'Courier New',monospace\"><span style=\"font-weight: bold\"> Non-trainable params: </span><span style=\"color: #00af00; text-decoration-color: #00af00\">0</span> (0.00 B)\n",

"</pre>\n"

]

},

"metadata": {}

}

]

}

]

}