<https://github.com/google-ai-edge/mediapipe-samples/blob/main/examples/pose_landmarker/python/%5BMediaPipe_Python_Tasks%5D_Pose_Landmarker.ipynb>

{

"cells": [

{

"cell\_type": "markdown",

"metadata": {

"id": "h2q27gKz1H20"

},

"source": [

"##### Copyright 2023 The MediaPipe Authors. All Rights Reserved."

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"cellView": "form",

"id": "TUfAcER1oUS6"

},

"outputs": [],

"source": [

"#@title Licensed under the Apache License, Version 2.0 (the \"License\");\n",

"# you may not use this file except in compliance with the License.\n",

"# You may obtain a copy of the License at\n",

"#\n",

"# https://www.apache.org/licenses/LICENSE-2.0\n",

"#\n",

"# Unless required by applicable law or agreed to in writing, software\n",

"# distributed under the License is distributed on an \"AS IS\" BASIS,\n",

"# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.\n",

"# See the License for the specific language governing permissions and\n",

"# limitations under the License."

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "L\_cQX8dWu4Dv"

},

"source": [

"# Pose Landmarks Detection with MediaPipe Tasks\n",

"\n",

"This notebook shows you how to use MediaPipe Tasks Python API to detect pose landmarks from images."

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "O6PN9FvIx614"

},

"source": [

"## Preparation\n",

"\n",

"Let's start with installing MediaPipe.\n"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "gxbHBsF-8Y\_l"

},

"outputs": [],

"source": [

"!pip install -q mediapipe"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "a49D7h4TVmru"

},

"source": [

"Then download an off-the-shelf model bundle. Check out the [MediaPipe documentation](https://developers.google.com/mediapipe/solutions/vision/pose\_landmarker#models) for more information about this model bundle."

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "OMjuVQiDYJKF"

},

"outputs": [],

"source": [

"!wget -O pose\_landmarker.task -q https://storage.googleapis.com/mediapipe-models/pose\_landmarker/pose\_landmarker\_heavy/float16/1/pose\_landmarker\_heavy.task"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "YYKAJ5nDU8-I"

},

"source": [

"## Visualization utilities"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "s3E6NFV-00Qt"

},

"outputs": [],

"source": [

"#@markdown To better demonstrate the Pose Landmarker API, we have created a set of visualization tools that will be used in this colab. These will draw the landmarks on a detect person, as well as the expected connections between those markers.\n",

"\n",

"from mediapipe import solutions\n",

"from mediapipe.framework.formats import landmark\_pb2\n",

"import numpy as np\n",

"\n",

"\n",

"def draw\_landmarks\_on\_image(rgb\_image, detection\_result):\n",

" pose\_landmarks\_list = detection\_result.pose\_landmarks\n",

" annotated\_image = np.copy(rgb\_image)\n",

"\n",

" # Loop through the detected poses to visualize.\n",

" for idx in range(len(pose\_landmarks\_list)):\n",

" pose\_landmarks = pose\_landmarks\_list[idx]\n",

"\n",

" # Draw the pose landmarks.\n",

" pose\_landmarks\_proto = landmark\_pb2.NormalizedLandmarkList()\n",

" pose\_landmarks\_proto.landmark.extend([\n",

" landmark\_pb2.NormalizedLandmark(x=landmark.x, y=landmark.y, z=landmark.z) for landmark in pose\_landmarks\n",

" ])\n",

" solutions.drawing\_utils.draw\_landmarks(\n",

" annotated\_image,\n",

" pose\_landmarks\_proto,\n",

" solutions.pose.POSE\_CONNECTIONS,\n",

" solutions.drawing\_styles.get\_default\_pose\_landmarks\_style())\n",

" return annotated\_image"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "83PEJNp9yPBU"

},

"source": [

"## Download test image\n",

"\n",

"To demonstrate the Pose Landmarker API, you can download a sample image using the follow code. The image is from [Pixabay](https://pixabay.com/photos/girl-woman-fitness-beautiful-smile-4051811/)."

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "tzXuqyIBlXer"

},

"outputs": [],

"source": [

"!wget -q -O image.jpg https://cdn.pixabay.com/photo/2019/03/12/20/39/girl-4051811\_960\_720.jpg\n",

"\n",

"import cv2\n",

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

"\n",

"img = cv2.imread(\"image.jpg\")\n",

"cv2\_imshow(img)"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "u-skLwMBmMN\_"

},

"source": [

"Optionally, you can upload your own image. If you want to do so, uncomment and run the cell below."

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "etBjSdwImQPw"

},

"outputs": [],

"source": [

"# from google.colab import files\n",

"# uploaded = files.upload()\n",

"\n",

"# for filename in uploaded:\n",

"# content = uploaded[filename]\n",

"# with open(filename, 'wb') as f:\n",

"# f.write(content)\n",

"\n",

"# if len(uploaded.keys()):\n",

"# IMAGE\_FILE = next(iter(uploaded))\n",

"# print('Uploaded file:', IMAGE\_FILE)"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "Iy4r2\_ePylIa"

},

"source": [

"## Running inference and visualizing the results\n",

"\n",

"The final step is to run pose landmark detection on your selected image. This involves creating your PoseLandmarker object, loading your image, running detection, and finally, the optional step of displaying the image with visualizations.\n",

"\n",

"Check out the [MediaPipe documentation](https://developers.google.com/mediapipe/solutions/vision/pose\_landmarker/python) to learn more about configuration options that this solution supports.\n"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "\_JVO3rvPD4RN"

},

"outputs": [],

"source": [

"# STEP 1: Import the necessary modules.\n",

"import mediapipe as mp\n",

"from mediapipe.tasks import python\n",

"from mediapipe.tasks.python import vision\n",

"\n",

"# STEP 2: Create an PoseLandmarker object.\n",

"base\_options = python.BaseOptions(model\_asset\_path='pose\_landmarker.task')\n",

"options = vision.PoseLandmarkerOptions(\n",

" base\_options=base\_options,\n",

" output\_segmentation\_masks=True)\n",

"detector = vision.PoseLandmarker.create\_from\_options(options)\n",

"\n",

"# STEP 3: Load the input image.\n",

"image = mp.Image.create\_from\_file(\"image.jpg\")\n",

"\n",

"# STEP 4: Detect pose landmarks from the input image.\n",

"detection\_result = detector.detect(image)\n",

"\n",

"# STEP 5: Process the detection result. In this case, visualize it.\n",

"annotated\_image = draw\_landmarks\_on\_image(image.numpy\_view(), detection\_result)\n",

"cv2\_imshow(cv2.cvtColor(annotated\_image, cv2.COLOR\_RGB2BGR))"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "\_BwzFvaxwtPX"

},

"source": [

"Visualize the pose segmentation mask."

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "3jAIFzw9M3JJ"

},

"outputs": [],

"source": [

"segmentation\_mask = detection\_result.segmentation\_masks[0].numpy\_view()\n",

"visualized\_mask = np.repeat(segmentation\_mask[:, :, np.newaxis], 3, axis=2) \* 255\n",

"cv2\_imshow(visualized\_mask)"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "QipRi2ozw7cg"

},

"outputs": [],

"source": []

}

],

"metadata": {

"colab": {

"provenance": []

},

"kernelspec": {

"display\_name": "Python 3 (ipykernel)",

"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": "3.8.10"

}

},

"nbformat": 4,

"nbformat\_minor": 0

}