OpenVIN-OJupyter

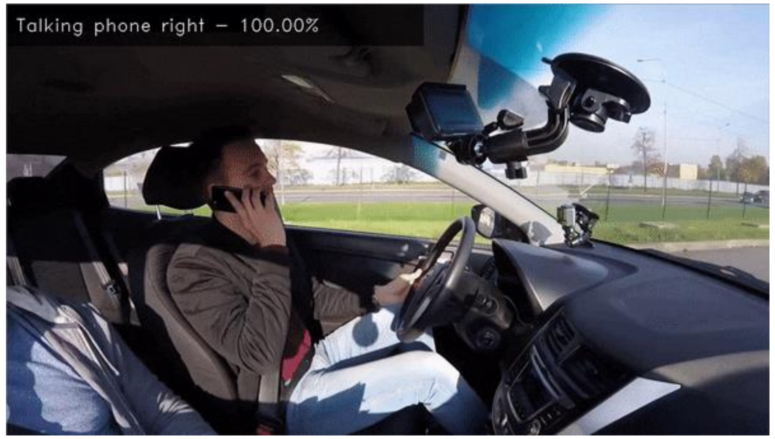
OpenVINO is an open source tool platform for optimizing the speed of AI inference, improving the performance of common deep learning in computer vision, speech recognition, natural language processing, and more. After converting those models using Tensorflow, PyTorch, Caffe and other frameworks to OpenVINO models, you can experience performance optimization and performance improvements on multiple different types of accelerators (CPU, GPU, NCS2).

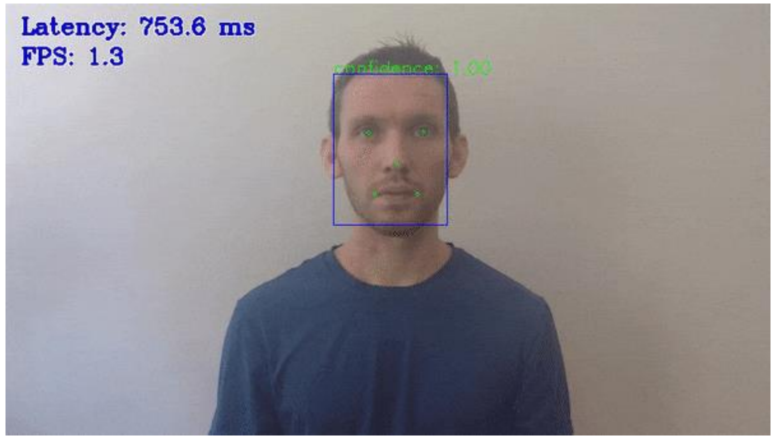
We rewrote the OpenVINO example into an .ipynb file that can be easily and quickly executed by JupyterLab.

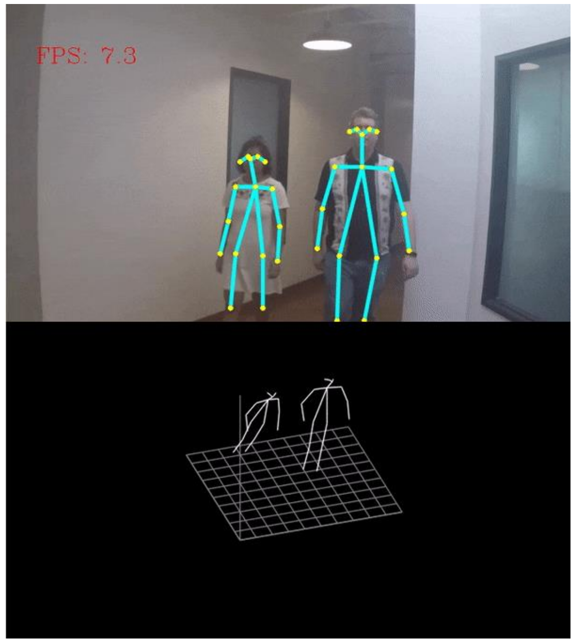
Version 20230223

Applications

Applications provided by OpenVINO: medical image segmentation, human pose recognition, Bert question answering, image classification, image deblurring, face recognition, location recognition, object detection, driving behavior recognition, translation, image restoration, MRI reconstruction, handwriting recognition, Image background removal, image synthesis, image segmentation, image parallax, noise reduction, speech recognition, text-to-speech.



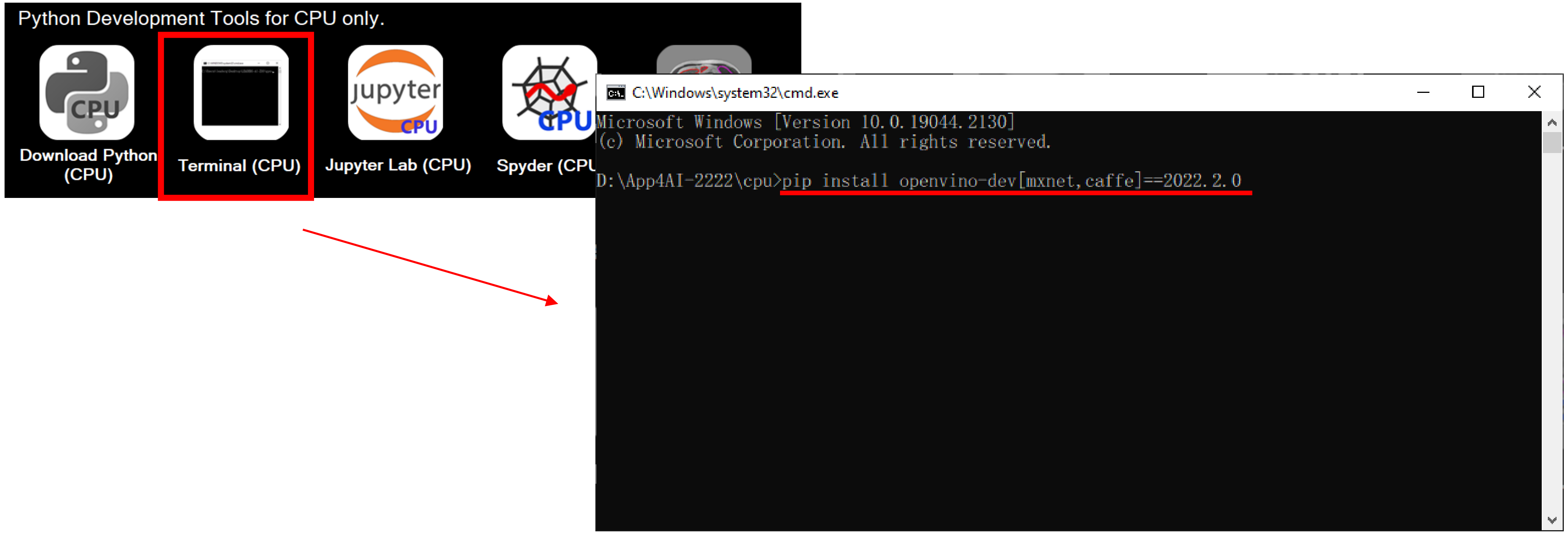




Before Use

- Please paste the following command in the terminal and press enter:

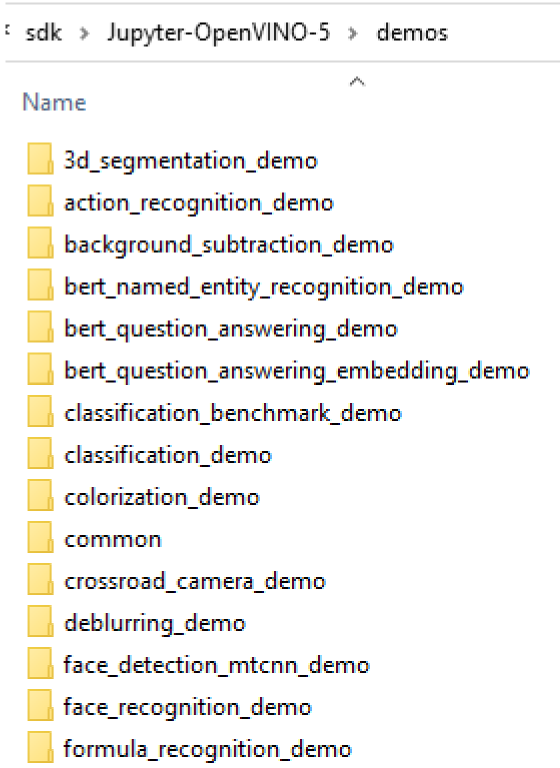
pip install openvino-dev[mxnet,caffe]==2022.2.0


Python Development Tools for CPU only.
Download Python (CPU)
Terminal (CPU)
Jupyter Lab (CPU)
Spyder (CPU
C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

D:\App4AI-2222\cpu>pip install openvino-dev[mxnet,caffe]==2022.2.0
```

function catalog

- 3d_segmentation_demo: segmentation of brain tumor images
- action_recognition_demo: action recognition
- background subtraction demo: remove the background
- bert_named_entity_recognition_demo: read text from english webpages for recognition
- bert_question_answering_demo : read text from an english webpage for question and answer
- bert_question_answering_embedding_demo: english webpage text question and answer embedding
- classification_demo: image classification
- colorization_demo: image colorization
- deblurring_demo: image deblurring
- face_detection_mtcnn_demo: face feature point detection
- face_recognition_demo: face recognition
- formula_recognition_demo: formula recognition



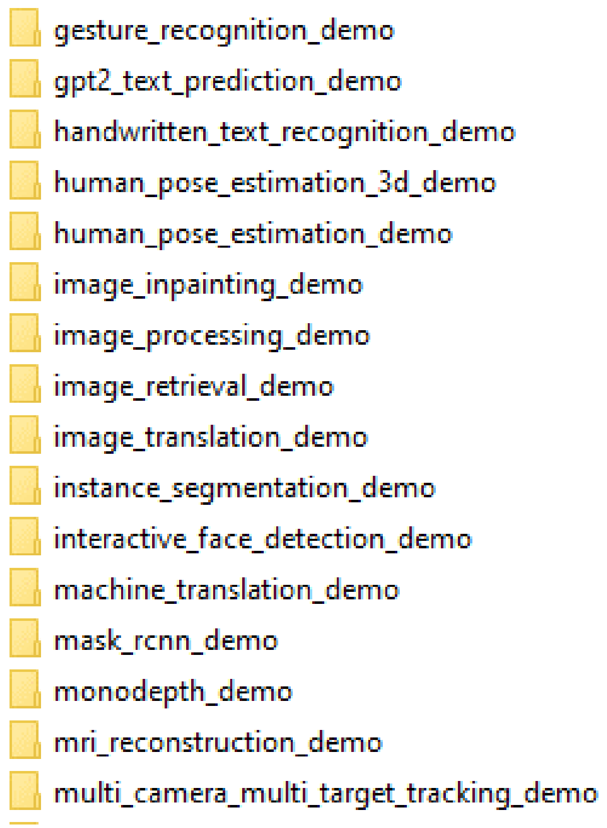
sdk › Jupyter-OpenVINO-5 › demos

Name

3d_segmentation_demo
action_recognition_demo
background_subtraction_demo
bert_named_entity_recognition_demo
bert_question_answering_demo
bert_question_answering_embedding_demo
classification_benchmark_demo
classification_demo
colorization_demo
common
crossroad_camera_demo
deblurring_demo
face_detection_mtcnn_demo
face_recognition_demo
formula_recognition_demo

function catalog

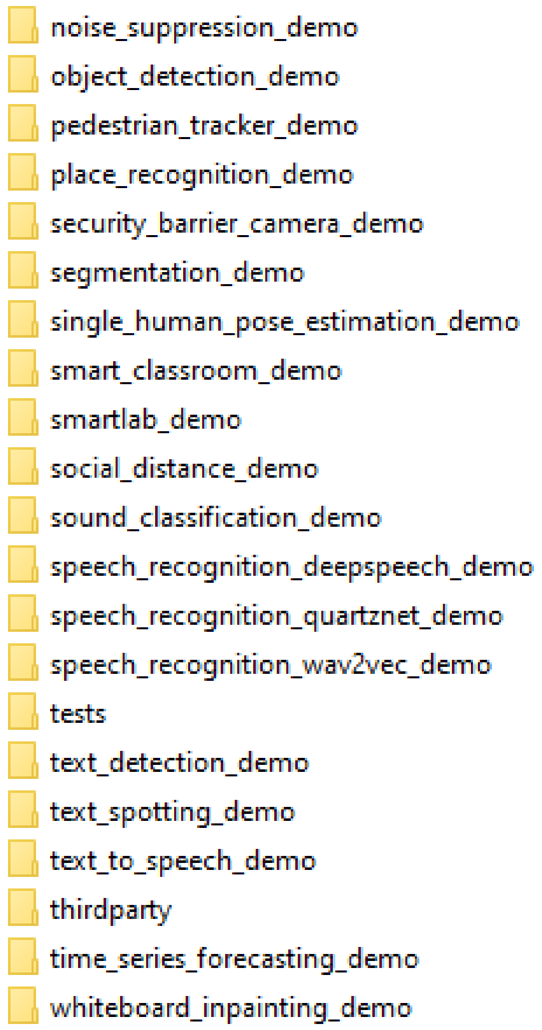
- gesture_recognition_demo: sign language recognition
- gpt2_text_prediction_demo: GPT2 prediction text
- handwritten_text_recognition_demo: handwriting recognition
- human_pose_estimation_3d_demo: 3D human pose detection
- human_pose_estimation_demo: human pose detection
- image_inpainting_demo: image inpainting
- image_retrieval_demo: image retrieval
- image_translation_demo: image synthesis
- instance_segmentation_demo: image segmentation
- machine_translation_demo: English-Russian translation, English-German translation
- monodepth_demo: disparity map from image
- mri_reconstruction_demo : MRI image reconstruction
- multi_camera_multi_target_tracking_demo: multi-camera object tracking



gesture_recognition_demo
gpt2_text_prediction_demo
handwritten_text_recognition_demo
human_pose_estimation_3d_demo
human_pose_estimation_demo
image_inpainting_demo
image_processing_demo
image_retrieval_demo
image_translation_demo
instance_segmentation_demo
interactive_face_detection_demo
machine_translation_demo
mask_rcnn_demo
monodepth_demo
mri_reconstruction_demo
multi_camera_multi_target_tracking_demo

function catalog

- noise_suppression_demo: sound file noise suppression
- object_detection_demo: object detection
- place_recognition_demo: place recognition
- segmentation_demo: image segmentation
- single_human_pose_estimation_demo: single human pose detection
- smartlab_demo: action recognition
- sound_classification_demo: sound classification
- speech_recognition_deepspeech_demo: DeepSpeech speech recognition
- speech_recognition_quartznet_demo: QuartzNet speech recognition
- speech_recognition_wav2vec_demo: Wav2Vec speech recognition
- text_spotting_demo: text recognition
- text_to_speech_demo: text-to-speech
- time_series_forecasting_demo: time series forecasting
- whiteboard_inpainting_demo: whiteboard text display



noise_suppression_demo
object_detection_demo
pedestrian_tracker_demo
place_recognition_demo
security_barrier_camera_demo
segmentation_demo
single_human_pose_estimation_demo
smart_classroom_demo
smartlab_demo
social_distance_demo
sound_classification_demo
speech_recognition_deepspeech_demo
speech_recognition_quartznet_demo
speech_recognition_wav2vec_demo
tests
text_detection_demo
text_spotting_demo
text_to_speech_demo
thirdparty
time_series_forecasting_demo
whiteboard_inpainting_demo

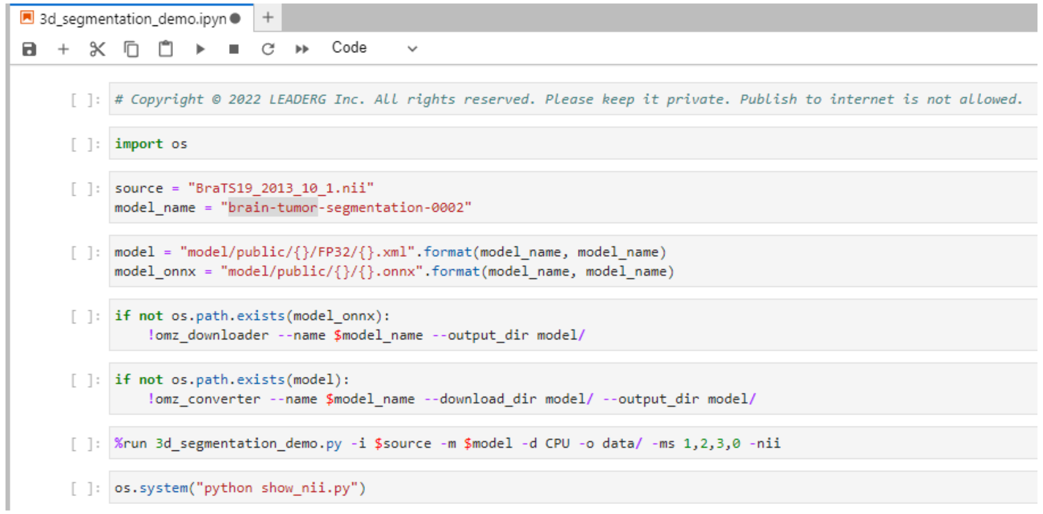
3d_segmentation_demo

Function: Segmentation of 3D brain tumor images

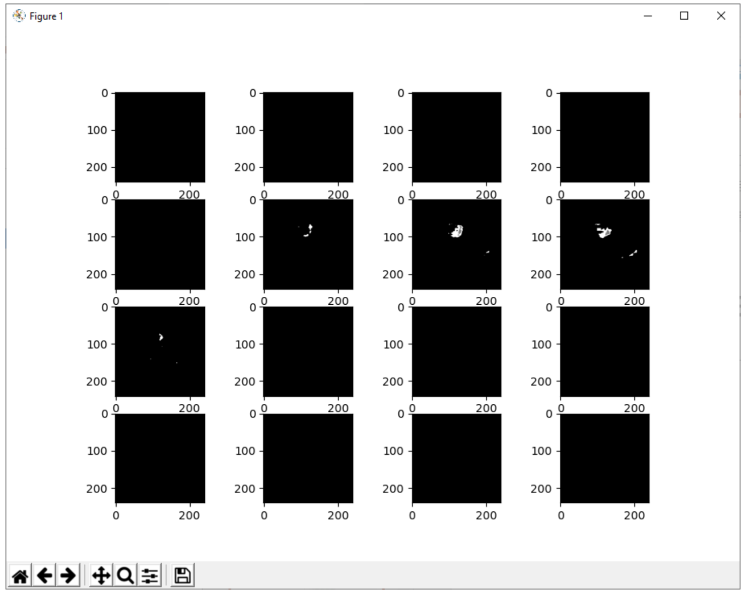
Introduction: Segmentation of BraTS2019 medical images into blocks with brain tumors

Source: Medical Imaging from BraTS2019

Inference file source: https://www.med.upenn.edu/cbica/brats2019/data.html



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: source = "BraTS19_2013_10_1.nii"
     model_name = "brain-tumor-segmentation-0002"

[ ]: model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
     model_onnx = "model/public/{}/{}.onnx".format(model_name, model_name)

[ ]: if not os.path.exists(model_onnx):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if not os.path.exists(model):
         !omz_converter --name $model_name --download_dir model/ --output_dir model/

[ ]: %run 3d_segmentation_demo.py -i $source -m $model -d CPU -o data/ -ms 1,2,3,0 -nii

[ ]: os.system("python show_nii.py")
```



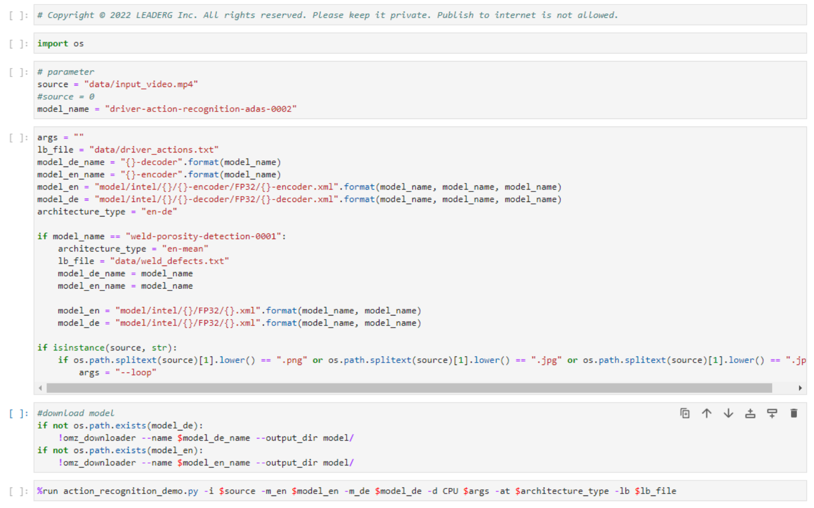
action_recognition_demo

Function: action recognition

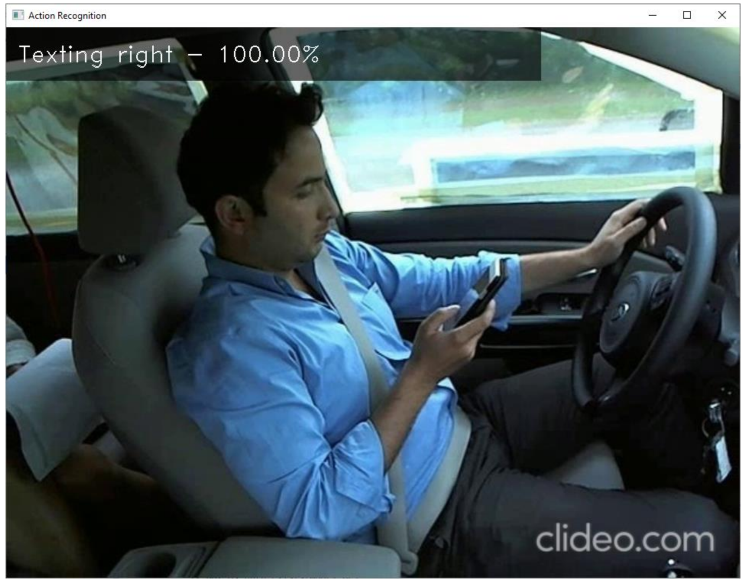
Introduction: Take ipynb as an example to identify people's behaviors while driving, such as making calls, driving safely, sending messages, etc., as shown in driver_actions.txt

Source: video, image, webcam

Inference file source:
https://www.kaggle.com/datasets/kunalrawat/test-video



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/input_video.mp4"
     #source = 0
     model_name = "driver-action-recognition-adas-0002"

[ ]: args = ""
     lb_file = "data/driver_actions.txt"
     model_de_name = "{}-decoder".format(model_name)
     model_en_name = "{}-encoder".format(model_name)
     model_en = "model/intel/{}/{}-encoder/FP32/{}-encoder.xml".format(model_name, model_name, model_name)
     model_de = "model/intel/{}/{}-decoder/FP32/{}-decoder.xml".format(model_name, model_name, model_name)
     architecture_type = "en-de"

     if model_name == "weld-porosity-detection-0001":
         architecture_type = "en-mean"
         lb_file = "data/weld_defects.txt"
         model_de_name = model_name
         model_en_name = model_name

         model_en = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
         model_de = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)

     if isinstance(source, str):
         if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
             args = "--loop"

[ ]: #download model
     if not os.path.exists(model_de):
         !omz_downloader --name $model_de_name --output_dir model/
     if not os.path.exists(model_en):
         !omz_downloader --name $model_en_name --output_dir model/

[ ]: %run action_recognition_demo.py -i $source -m_en $model_en -m_de $model_de -d CPU $args -at $architecture_type -lb $lb_file
```

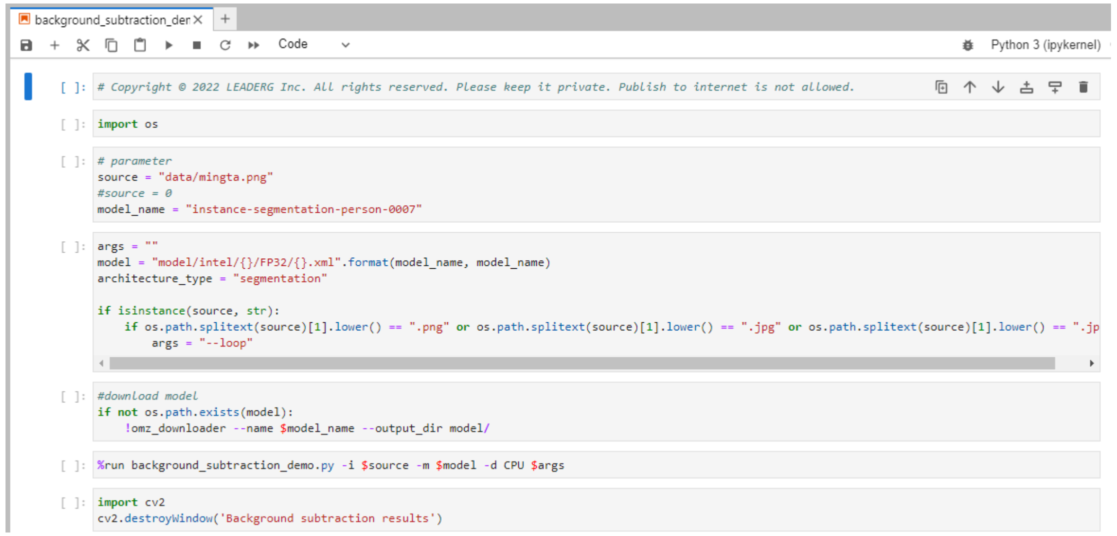


background_subtraction_demo

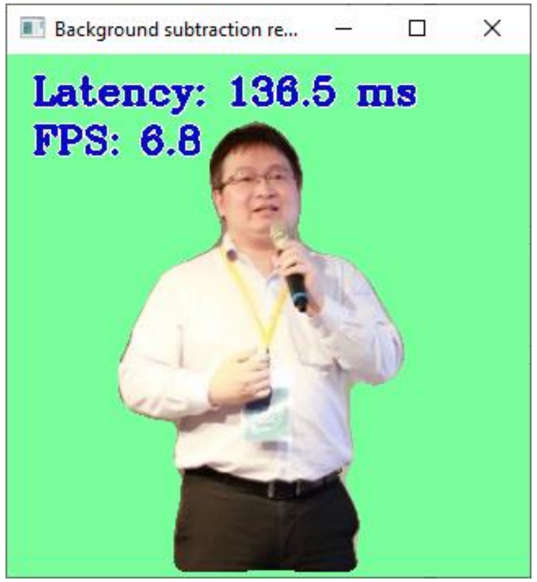
Function: remove background

Introduction: When someone appears, remove the background other than the person, and fill the removed background with light green

Source: video, image, webcam



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/mingta.png"
     #source = 0
     model_name = "instance-segmentation-person-0007"

[ ]: args = ""
     model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
     architecture_type = "segmentation"

     if isinstance(source, str):
         if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
             args = "--loop"

[ ]: #download model
     if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/

[ ]: %run background_subtraction_demo.py -i $source -m $model -d CPU $args

[ ]: import cv2
     cv2.destroyWindow('Background subtraction results')
```



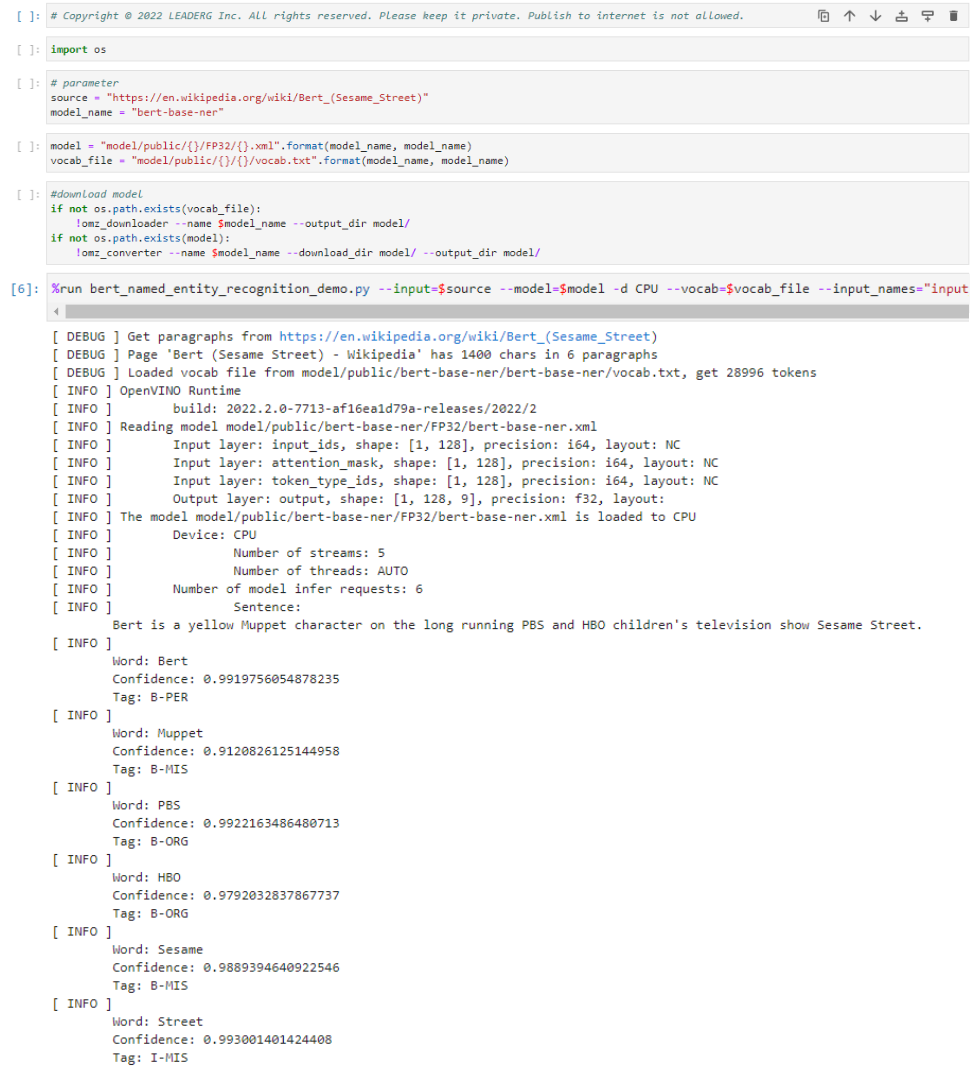
bert_named_entity_recognition_demo

Function: Read text from English web pages for identification

Introduction: After reading the text parsed by the English web page, output the state represented by the words in each sentence, LOC represents location, PER represents person, ORG represents organization, MISC represents miscellaneous

Source: URL

Note: Too much text on the webpage may cause the recognition failure



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "https://en.wikipedia.org/wiki/Bert_(Sesame_Street)"
     model_name = "bert-base-ner"

[ ]: model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
     vocab_file = "model/public/{}/{}/vocab.txt".format(model_name, model_name)

[ ]: #download model
     if not os.path.exists(vocab_file):
         !omz_downloader --name $model_name --output_dir model/
     if not os.path.exists(model):
         !omz_converter --name $model_name --download_dir model/ --output_dir model/

[6]: %run bert_named_entity_recognition_demo.py --input=$source --model=$model -d CPU --vocab=$vocab_file --input_names="input
```

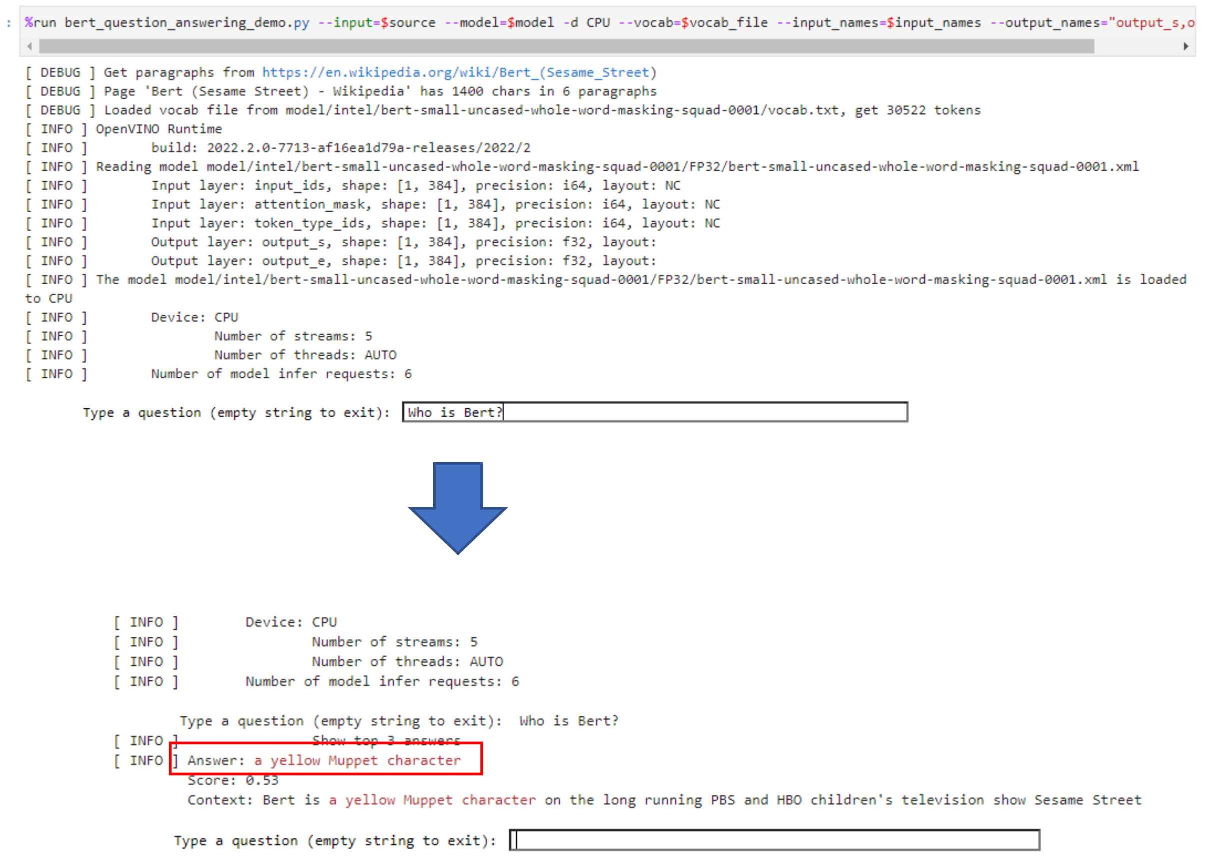
```
[ DEBUG ] Get paragraphs from https://en.wikipedia.org/wiki/Bert_(Sesame_Street)
[ DEBUG ] Page 'Bert (Sesame Street) - Wikipedia' has 1400 chars in 6 paragraphs
[ DEBUG ] Loaded vocab file from model/public/bert-base-ner/bert-base-ner/vocab.txt, get 28996 tokens
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.2.0-7713-af16ea1d79a-releases/2022/2
[ INFO ] Reading model model/public/bert-base-ner/FP32/bert-base-ner.xml
[ INFO ]        Input layer: input_ids, shape: [1, 128], precision: i64, layout: NC
[ INFO ]        Input layer: attention_mask, shape: [1, 128], precision: i64, layout: NC
[ INFO ]        Input layer: token_type_ids, shape: [1, 128], precision: i64, layout: NC
[ INFO ]        Output layer: output, shape: [1, 128, 9], precision: f32, layout:
[ INFO ] The model model/public/bert-base-ner/FP32/bert-base-ner.xml is loaded to CPU
[ INFO ]        Device: CPU
[ INFO ]                Number of streams: 5
[ INFO ]                Number of threads: AUTO
[ INFO ]        Number of model infer requests: 6
[ INFO ]                Sentence:
        Bert is a yellow Muppet character on the long running PBS and HBO children's television show Sesame Street.
[ INFO ]
        Word: Bert
        Confidence: 0.9919756054878235
        Tag: B-PER
[ INFO ]
        Word: Muppet
        Confidence: 0.9120826125144958
        Tag: B-MIS
[ INFO ]
        Word: PBS
        Confidence: 0.9922163486480713
        Tag: B-ORG
[ INFO ]
        Word: HBO
        Confidence: 0.9792032837867737
        Tag: B-ORG
[ INFO ]
        Word: Sesame
        Confidence: 0.9889394640922546
        Tag: B-MIS
[ INFO ]
        Word: Street
        Confidence: 0.993001401424408
        Tag: I-MIS
```

bert_question_answering_demo

Function: Read text from English web pages for question and answer

Introduction: After reading the text parsed by the English web page, enter the questions related to the web page, and the most likely answers, scores and sources will appear.

Source: URL



```
: %run bert_question_answering_demo.py --input=$source --model=$model -d CPU --vocab=$vocab_file --input_names=$input_names --output_names="output_s,o

[ DEBUG ] Get paragraphs from https://en.wikipedia.org/wiki/Bert_(Sesame_Street)
[ DEBUG ] Page 'Bert (Sesame Street) - Wikipedia' has 1400 chars in 6 paragraphs
[ DEBUG ] Loaded vocab file from model/intel/bert-small-uncased-whole-word-masking-squad-0001/vocab.txt, get 30522 tokens
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.2.0-7713-af16ea1d79a-releases/2022/2
[ INFO ] Reading model model/intel/bert-small-uncased-whole-word-masking-squad-0001/FP32/bert-small-uncased-whole-word-masking-squad-0001.xml
[ INFO ]        Input layer: input_ids, shape: [1, 384], precision: i64, layout: NC
[ INFO ]        Input layer: attention_mask, shape: [1, 384], precision: i64, layout: NC
[ INFO ]        Input layer: token_type_ids, shape: [1, 384], precision: i64, layout: NC
[ INFO ]        Output layer: output_s, shape: [1, 384], precision: f32, layout:
[ INFO ]        Output layer: output_e, shape: [1, 384], precision: f32, layout:
[ INFO ] The model model/intel/bert-small-uncased-whole-word-masking-squad-0001/FP32/bert-small-uncased-whole-word-masking-squad-0001.xml is loaded
to CPU
[ INFO ]        Device: CPU
[ INFO ]                Number of streams: 5
[ INFO ]                Number of threads: AUTO
[ INFO ]        Number of model infer requests: 6

        Type a question (empty string to exit): Who is Bert?
```

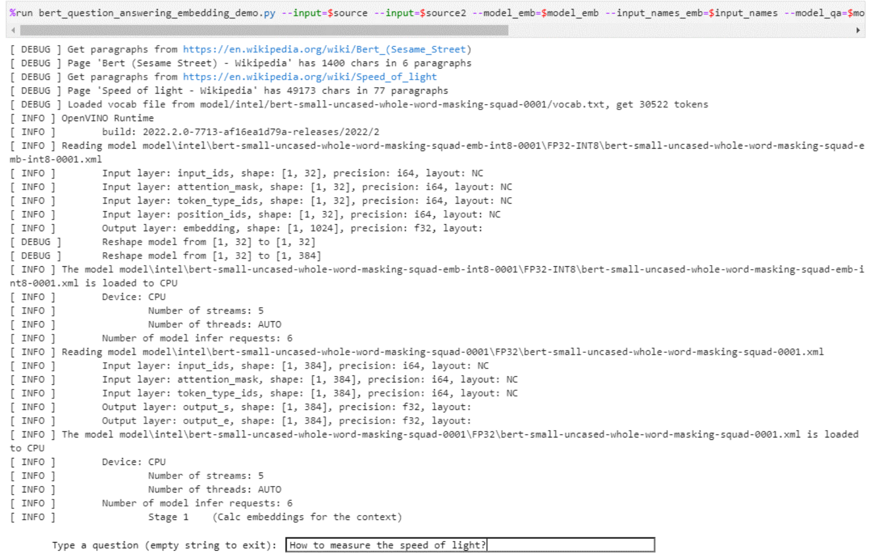
```
[ INFO ]        Device: CPU
[ INFO ]                Number of streams: 5
[ INFO ]                Number of threads: AUTO
[ INFO ]        Number of model infer requests: 6

        Type a question (empty string to exit):  Who is Bert?
[ INFO ]                Show top 3 answers
[ INFO ] Answer: a yellow Muppet character
         Score: 0.53
         Context: Bert is a yellow Muppet character on the long running PBS and HBO children's television show Sesame Street

        Type a question (empty string to exit):
```

bert_question_answering_embedding_demo

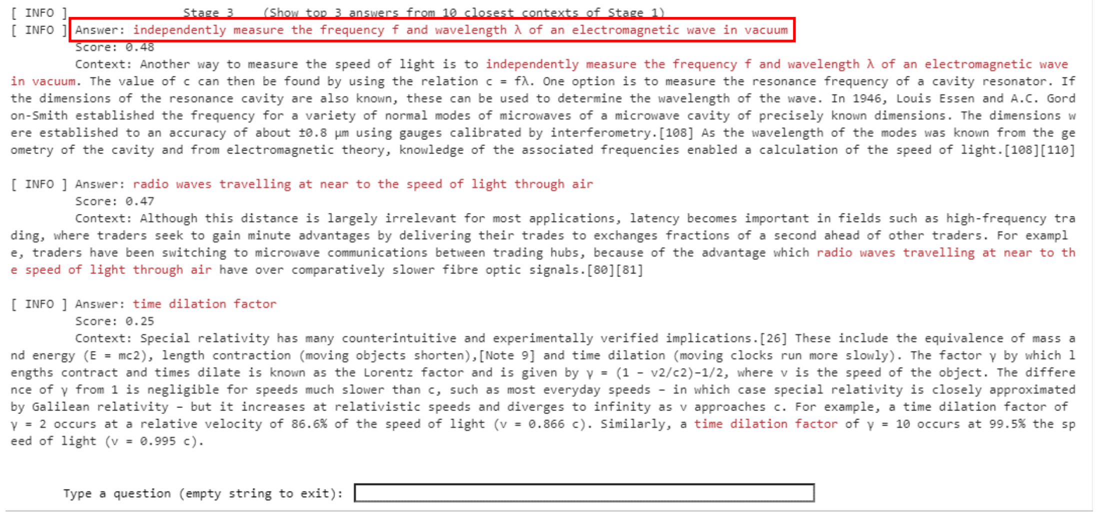
Function: Read text from English web pages for question and answer

Introduction: After reading the text parsed by two English web pages, enter a question, and extract the most likely answer, score and source from the article

Source: two URLs



```
%run bert_question_answering_embedding_demo.py --input=$source --input=$source2 --model_emb=$model_emb --input_names_emb=$input_names --model_qa=$mo
[ DEBUG ] Get paragraphs from https://en.wikipedia.org/wiki/Bert_(Sesame_Street)
[ DEBUG ] Page 'Bert (Sesame Street) - Wikipedia' has 1400 chars in 6 paragraphs
[ DEBUG ] Get paragraphs from https://en.wikipedia.org/wiki/Speed_of_light
[ DEBUG ] Page 'Speed of light - Wikipedia' has 49173 chars in 77 paragraphs
[ DEBUG ] Loaded vocab file from model/intel/bert-small-uncased-whole-word-masking-squad-0001/vocab.txt, get 30522 tokens
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.2.0-7713-af16ea1d79a-releases/2022/2
[ INFO ] Reading model model\intel\bert-small-uncased-whole-word-masking-squad-emb-int8-0001\FP32-INT8\bert-small-uncased-whole-word-masking-squad-e
mb-int8-0001.xml
[ INFO ]        Input layer: input_ids, shape: [1, 32], precision: i64, layout: NC
[ INFO ]        Input layer: attention_mask, shape: [1, 32], precision: i64, layout: NC
[ INFO ]        Input layer: token_type_ids, shape: [1, 32], precision: i64, layout: NC
[ INFO ]        Input layer: position_ids, shape: [1, 32], precision: i64, layout: NC
[ INFO ]        Output layer: embedding, shape: [1, 1024], precision: f32, layout:
[ DEBUG ]       Reshape model from [1, 32] to [1, 32]
[ DEBUG ]       Reshape model from [1, 32] to [1, 384]
[ INFO ] The model model\intel\bert-small-uncased-whole-word-masking-squad-emb-int8-0001\FP32-INT8\bert-small-uncased-whole-word-masking-squad-emb-i
nt8-0001.xml is loaded to CPU
[ INFO ]        Device: CPU
[ INFO ]                Number of streams: 5
[ INFO ]                Number of threads: AUTO
[ INFO ]        Number of model infer requests: 6
[ INFO ] Reading model model\intel\bert-small-uncased-whole-word-masking-squad-0001\FP32\bert-small-uncased-whole-word-masking-squad-0001.xml
[ INFO ]        Input layer: input_ids, shape: [1, 384], precision: i64, layout: NC
[ INFO ]        Input layer: attention_mask, shape: [1, 384], precision: i64, layout: NC
[ INFO ]        Input layer: token_type_ids, shape: [1, 384], precision: i64, layout: NC
[ INFO ]        Output layer: output_s, shape: [1, 384], precision: f32, layout:
[ INFO ]        Output layer: output_e, shape: [1, 384], precision: f32, layout:
[ INFO ] The model model\intel\bert-small-uncased-whole-word-masking-squad-0001\FP32\bert-small-uncased-whole-word-masking-squad-0001.xml is loaded
to CPU
[ INFO ]        Device: CPU
[ INFO ]                Number of streams: 5
[ INFO ]                Number of threads: AUTO
[ INFO ]        Number of model infer requests: 6
[ INFO ]                Stage 1    (Calc embeddings for the context)

      Type a question (empty string to exit): How to measure the speed of light?
```



```
[ INFO ]                Stage 3    (Show top 3 answers from 10 closest contexts of Stage 1)
[ INFO ] Answer: independently measure the frequency f and wavelength λ of an electromagnetic wave in vacuum
         Score: 0.48
         Context: Another way to measure the speed of light is to independently measure the frequency f and wavelength λ of an electromagnetic wave
in vacuum. The value of c can then be found by using the relation c = fλ. One option is to measure the resonance frequency of a cavity resonator. If
the dimensions of the resonance cavity are also known, these can be used to determine the wavelength of the wave. In 1946, Louis Essen and A.C. Gord
on-Smith established the frequency for a variety of normal modes of microwaves of a microwave cavity of precisely known dimensions. The dimensions w
ere established to an accuracy of about ±0.8 μm using gauges calibrated by interferometry.[108] As the wavelength of the modes was known from the ge
ometry of the cavity and from electromagnetic theory, knowledge of the associated frequencies enabled a calculation of the speed of light.[108][110]

[ INFO ] Answer: radio waves travelling at near to the speed of light through air
         Score: 0.47
         Context: Although this distance is largely irrelevant for most applications, latency becomes important in fields such as high-frequency tra
ding, where traders seek to gain minute advantages by delivering their trades to exchanges fractions of a second ahead of other traders. For exampl
e, traders have been switching to microwave communications between trading hubs, because of the advantage which radio waves travelling at near to th
e speed of light through air have over comparatively slower fibre optic signals.[80][81]

[ INFO ] Answer: time dilation factor
         Score: 0.25
         Context: Special relativity has many counterintuitive and experimentally verified implications.[26] These include the equivalence of mass a
nd energy (E = mc2), length contraction (moving objects shorten),[Note 9] and time dilation (moving clocks run more slowly). The factor γ by which l
engths contract and times dilate is known as the Lorentz factor and is given by γ = (1 − v2/c2)-1/2, where v is the speed of the object. The differe
nce of γ from 1 is negligible for speeds much slower than c, such as most everyday speeds – in which case special relativity is closely approximated
by Galilean relativity – but it increases at relativistic speeds and diverges to infinity as v approaches c. For example, a time dilation factor of
γ = 2 occurs at a relative velocity of 86.6% of the speed of light (v = 0.866 c). Similarly, a time dilation factor of γ = 10 occurs at 99.5% the sp
eed of light (v = 0.995 c).


     Type a question (empty string to exit):
```

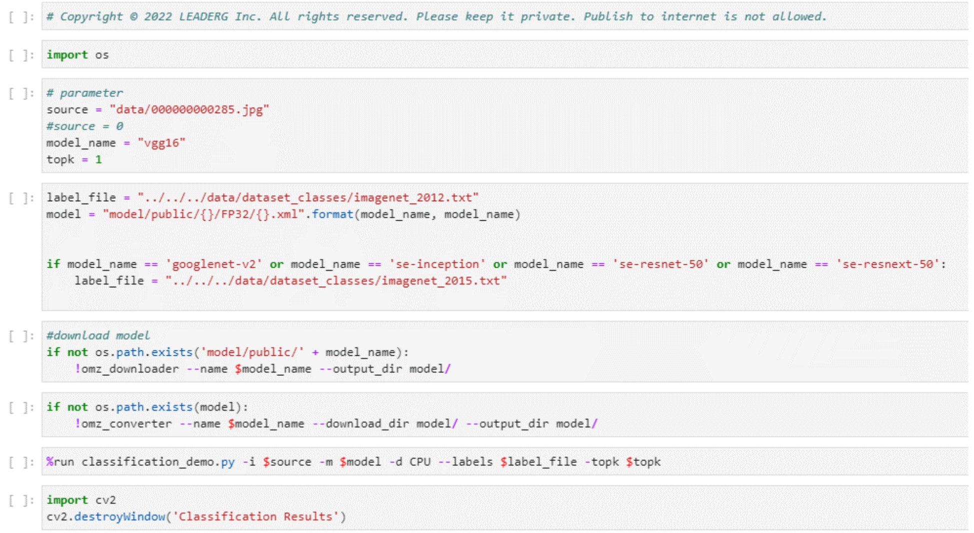
classification_demo

Function: image classification

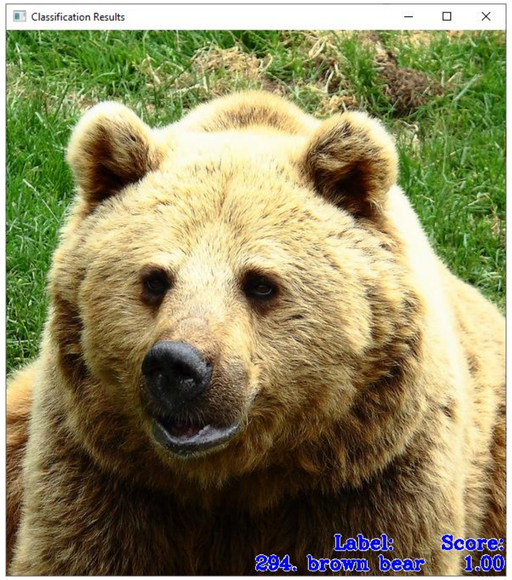
Introduction: Take ipynb as an example, find the most similar category in the image, the category name that can be classified is shown in data/dataset_classes/imagenet_2012.txt

Source: video, image, webcam

Number of categories (topk): the most like n categories



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/000000000285.jpg"
     #source = 0
     model_name = "vgg16"
     topk = 1

[ ]: label_file = "../../../data/dataset_classes/imagenet_2012.txt"
     model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)


     if model_name == 'googlenet-v2' or model_name == 'se-inception' or model_name == 'se-resnet-50' or model_name == 'se-resnext-50':
         label_file = "../../../data/dataset_classes/imagenet_2015.txt"

[ ]: #download model
     if not os.path.exists('model/public/' + model_name):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if not os.path.exists(model):
         !omz_converter --name $model_name --download_dir model/ --output_dir model/

[ ]: %run classification_demo.py -i $source -m $model -d CPU --labels $label_file -topk $topk

[ ]: import cv2
     cv2.destroyWindow('Classification Results')
```

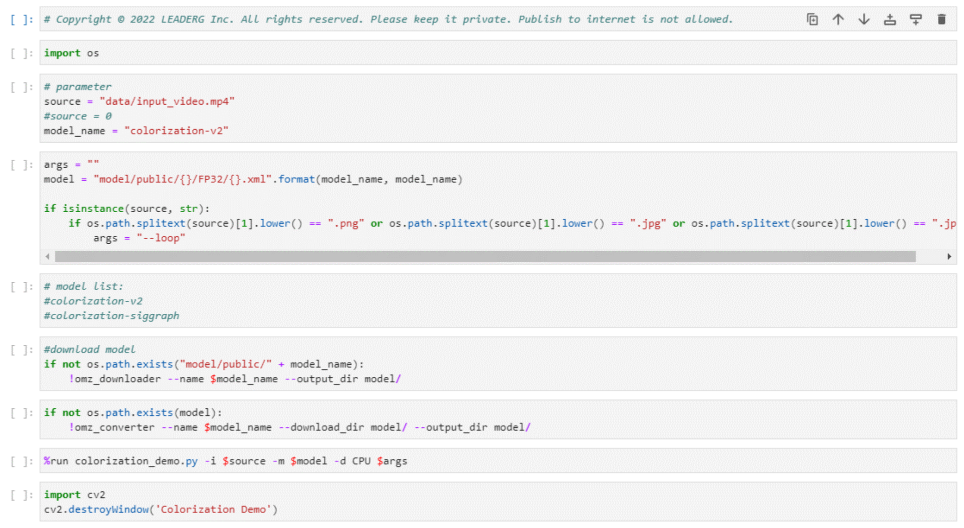


colorization_demo

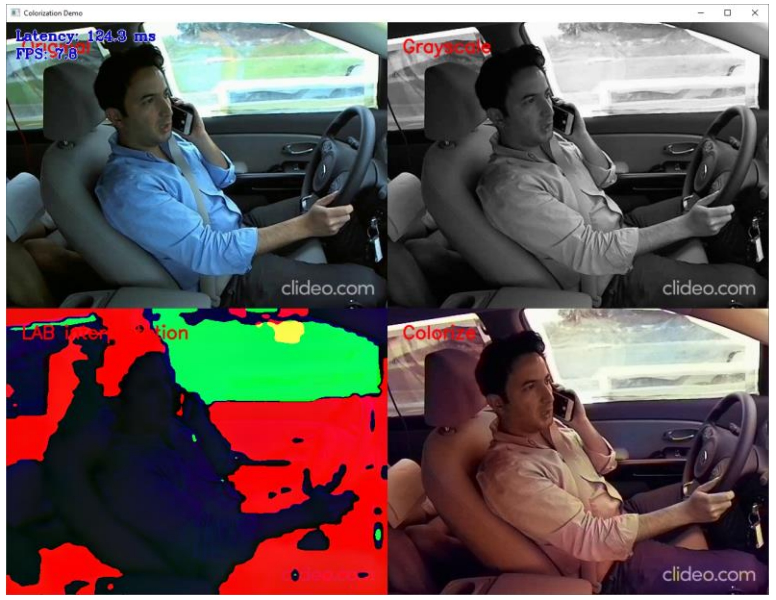
Function: Image colorization

Introduction: using neural networks to colorize a grayscale image or video.

Source: video, image, webcam



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/input_video.mp4"
     #source = 0
     model_name = "colorization-v2"

[ ]: args = ""
     model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)

     if isinstance(source, str):
         if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
             args = "--loop"

[ ]: # model list:
     #colorization-v2
     #colorization-siggraph

[ ]: #download model
     if not os.path.exists("model/public/" + model_name):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if not os.path.exists(model):
         !omz_converter --name $model_name --download_dir model/ --output_dir model/

[ ]: %run colorization_demo.py -i $source -m $model -d CPU $args

[ ]: import cv2
     cv2.destroyWindow('Colorization Demo')
```



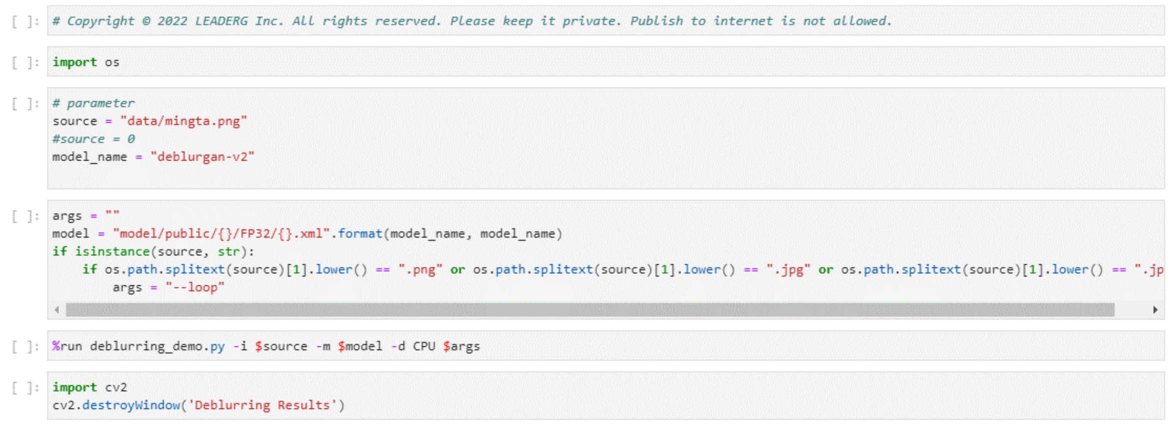
Inference video source:
https://www.kaggle.com/datasets/kunalrawat/test-video

deblurring_demo

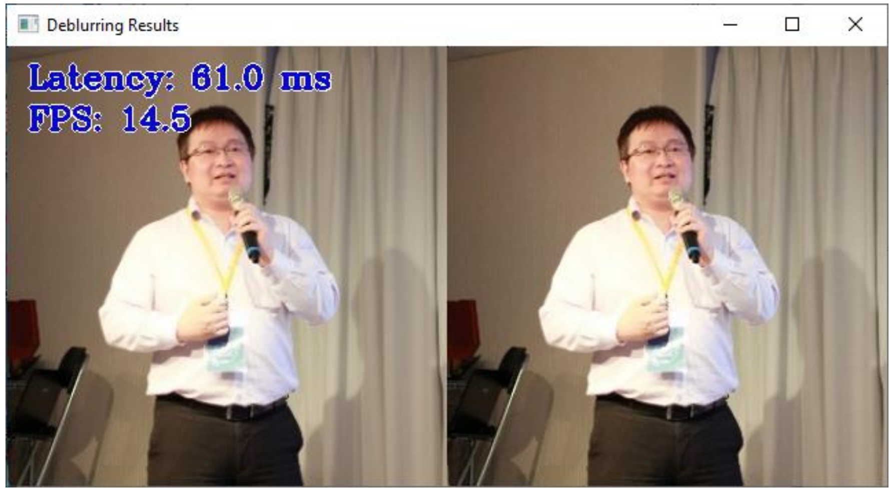
Function: Image deblurring

Introduction: Deblurring an image

Source: video, image, webcam



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/mingta.png"
     #source = 0
     model_name = "deblurgan-v2"

[ ]: args = ""
     model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
     if isinstance(source, str):
         if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
             args = "--loop"

[ ]: %run deblurring_demo.py -i $source -m $model -d CPU $args

[ ]: import cv2
     cv2.destroyWindow('Deblurring Results')
```

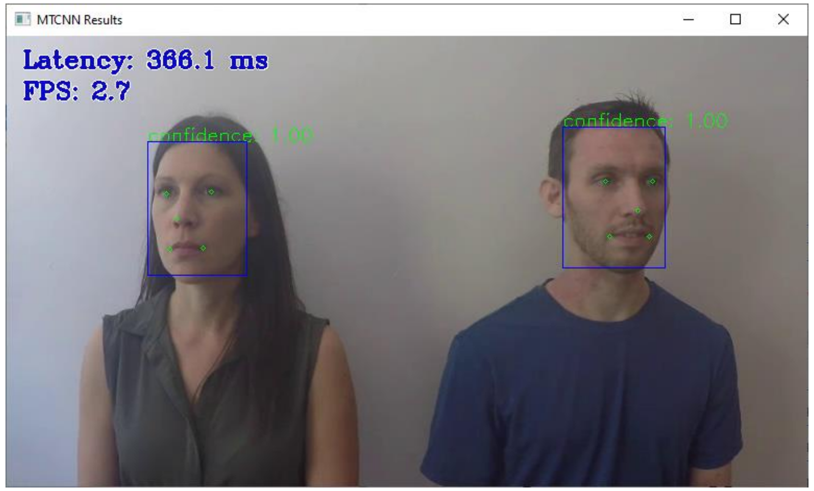


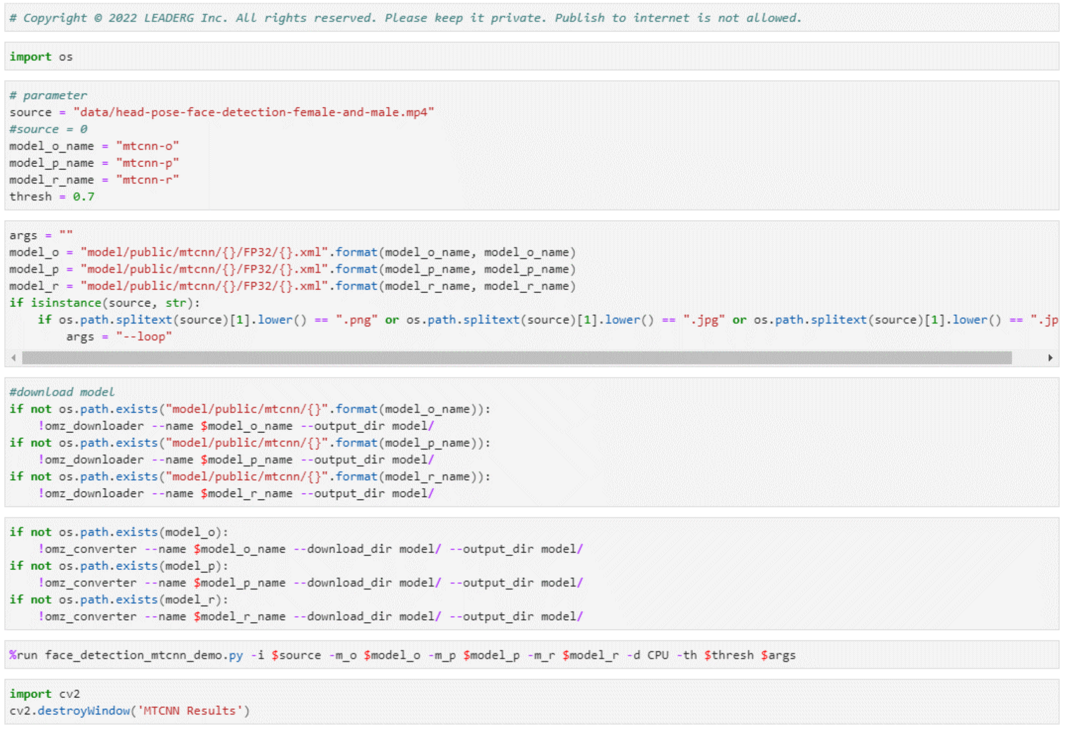
face_detection_mtcnn_demo

Function: face feature point detection

Introduction: Detect the position of the face and the position of the eyes, the tip of the nose, and the corners of the mouth

Source: video, image, webcam





```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source = "data/head-pose-face-detection-female-and-male.mp4"
#source = 0
model_o_name = "mtcnn-o"
model_p_name = "mtcnn-p"
model_r_name = "mtcnn-r"
thresh = 0.7
```

```
args = ""
model_o = "model/public/mtcnn/{}/FP32/{}.xml".format(model_o_name, model_o_name)
model_p = "model/public/mtcnn/{}/FP32/{}.xml".format(model_p_name, model_p_name)
model_r = "model/public/mtcnn/{}/FP32/{}.xml".format(model_r_name, model_r_name)
if isinstance(source, str):
    if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
        args = "--loop"
```

```
#download model
if not os.path.exists("model/public/mtcnn/{}".format(model_o_name)):
    !omz_downloader --name $model_o_name --output_dir model/
if not os.path.exists("model/public/mtcnn/{}".format(model_p_name)):
    !omz_downloader --name $model_p_name --output_dir model/
if not os.path.exists("model/public/mtcnn/{}".format(model_r_name)):
    !omz_downloader --name $model_r_name --output_dir model/
```

```
if not os.path.exists(model_o):
    !omz_converter --name $model_o_name --download_dir model/ --output_dir model/
if not os.path.exists(model_p):
    !omz_converter --name $model_p_name --download_dir model/ --output_dir model/
if not os.path.exists(model_r):
    !omz_converter --name $model_r_name --download_dir model/ --output_dir model/
```

```
%run face_detection_mtcnn_demo.py -i $source -m_o $model_o -m_p $model_p -m_r $model_r -d CPU -th $thresh $args
```

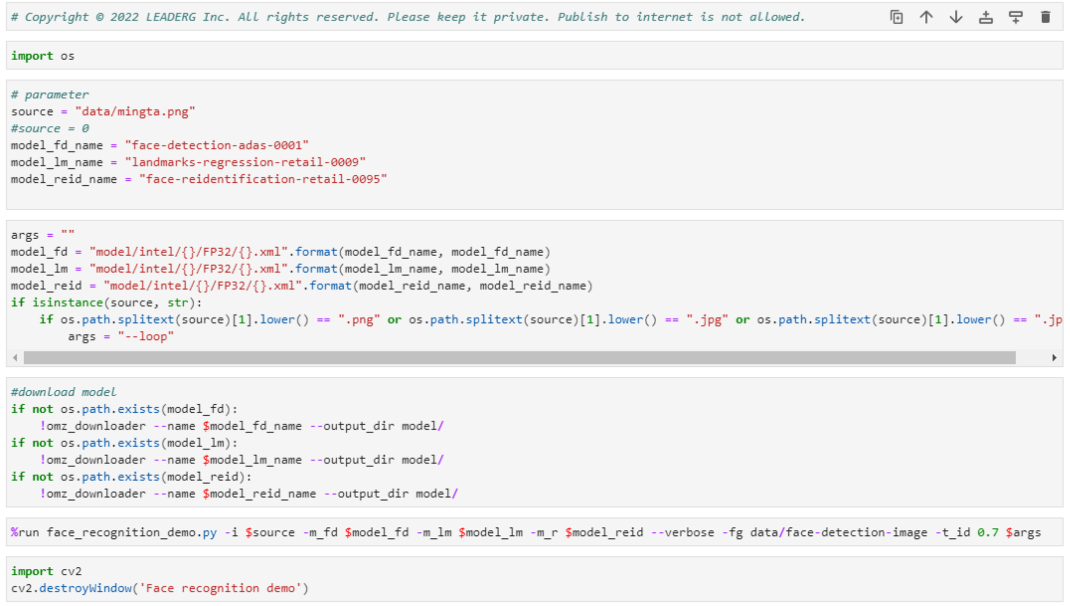
```
import cv2
cv2.destroyWindow('MTCNN Results')
```

face_recognition_demo

Function: face recognition

Introduction: Recognize face position, feature point position and perform face recognition based on the images in the “data/face-detection-image” folder

Source: video, image, webcam



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

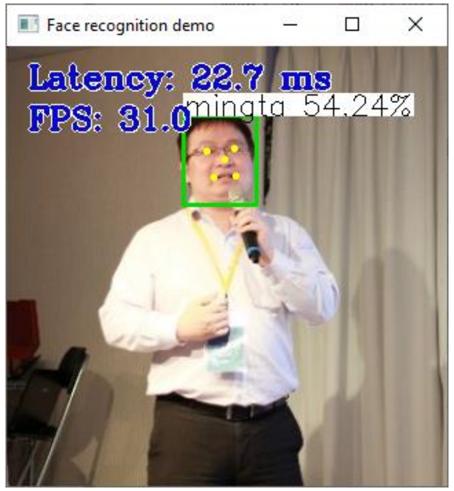
```
# parameter
source = "data/mingta.png"
#source = 0
model_fd_name = "face-detection-adas-0001"
model_lm_name = "landmarks-regression-retail-0009"
model_reid_name = "face-reidentification-retail-0095"
```

```
args = ""
model_fd = "model/intel/{}/FP32/{}.xml".format(model_fd_name, model_fd_name)
model_lm = "model/intel/{}/FP32/{}.xml".format(model_lm_name, model_lm_name)
model_reid = "model/intel/{}/FP32/{}.xml".format(model_reid_name, model_reid_name)
if isinstance(source, str):
    if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
        args = "--loop"
```

```
#download model
if not os.path.exists(model_fd):
    !omz_downloader --name $model_fd_name --output_dir model/
if not os.path.exists(model_lm):
    !omz_downloader --name $model_lm_name --output_dir model/
if not os.path.exists(model_reid):
    !omz_downloader --name $model_reid_name --output_dir model/
```

```
%run face_recognition_demo.py -i $source -m_fd $model_fd -m_lm $model_lm -m_r $model_reid --verbose -fg data/face-detection-image -t_id 0.7 $args
```

```
import cv2
cv2.destroyWindow('Face recognition demo')
```



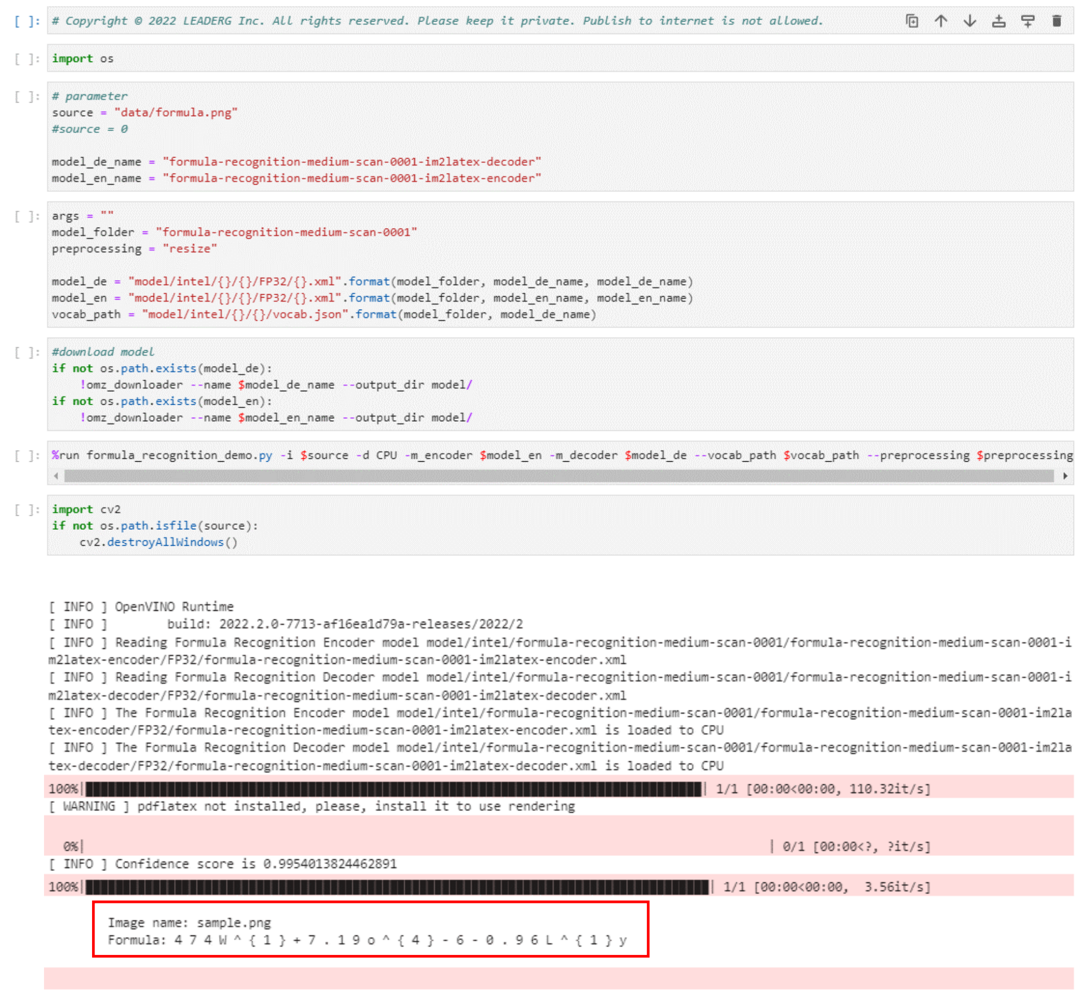
formula_recognition_demo

Function: formula recognition

Introduction: If it is a webcam, parse the formula in the red box, if it is an image, parse the formula in the image

Source: video, image, webcam

$$474W^{1}+7.19o^{4}-6-0.96L^{1}y$$



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/formula.png"
     #source = 0

     model_de_name = "formula-recognition-medium-scan-0001-im2latex-decoder"
     model_en_name = "formula-recognition-medium-scan-0001-im2latex-encoder"

[ ]: args = ""
     model_folder = "formula-recognition-medium-scan-0001"
     preprocessing = "resize"

     model_de = "model/intel/{}/{}/FP32/{}.xml".format(model_folder, model_de_name, model_de_name)
     model_en = "model/intel/{}/{}/FP32/{}.xml".format(model_folder, model_en_name, model_en_name)
     vocab_path = "model/intel/{}/{}/vocab.json".format(model_folder, model_de_name)

[ ]: #download model
     if not os.path.exists(model_de):
         !omz_downloader --name $model_de_name --output_dir model/
     if not os.path.exists(model_en):
         !omz_downloader --name $model_en_name --output_dir model/

[ ]: %run formula_recognition_demo.py -i $source -d CPU -m_encoder $model_en -m_decoder $model_de --vocab_path $vocab_path --preprocessing $preprocessing

[ ]: import cv2
     if not os.path.isfile(source):
         cv2.destroyAllWindows()
```

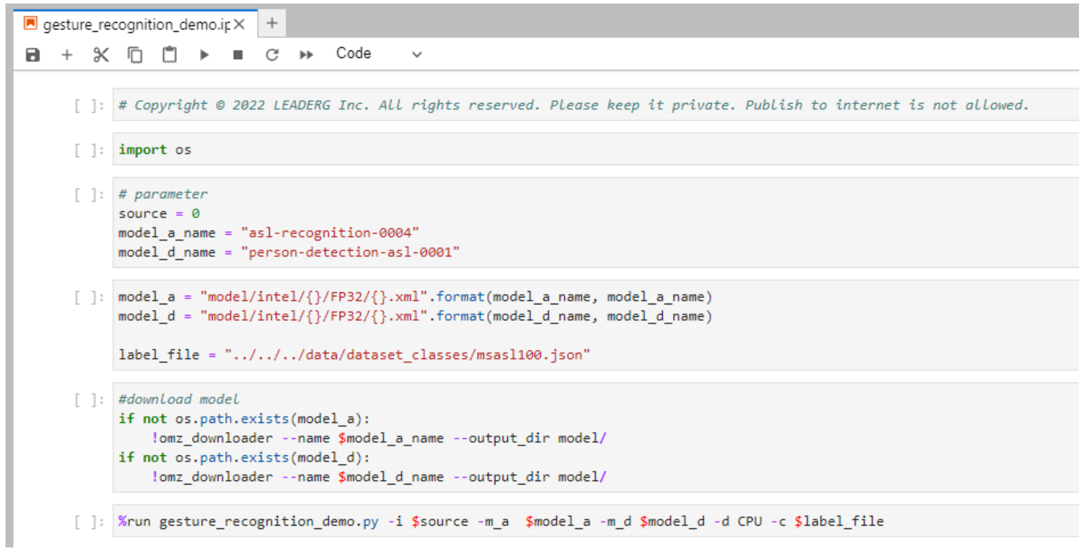
```
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.2.0-7713-af16ea1d79a-releases/2022/2
[ INFO ] Reading Formula Recognition Encoder model model/intel/formula-recognition-medium-scan-0001/formula-recognition-medium-scan-0001-i
m2latex-encoder/FP32/formula-recognition-medium-scan-0001-im2latex-encoder.xml
[ INFO ] Reading Formula Recognition Decoder model model/intel/formula-recognition-medium-scan-0001/formula-recognition-medium-scan-0001-i
m2latex-decoder/FP32/formula-recognition-medium-scan-0001-im2latex-decoder.xml
[ INFO ] The Formula Recognition Encoder model model/intel/formula-recognition-medium-scan-0001/formula-recognition-medium-scan-0001-im2la
tex-encoder/FP32/formula-recognition-medium-scan-0001-im2latex-encoder.xml is loaded to CPU
[ INFO ] The Formula Recognition Decoder model model/intel/formula-recognition-medium-scan-0001/formula-recognition-medium-scan-0001-im2la
tex-decoder/FP32/formula-recognition-medium-scan-0001-im2latex-decoder.xml is loaded to CPU
100%|██████████████████████████████████████████████████████████████████████| 1/1 [00:00<00:00, 110.32it/s]
[ WARNING ] pdflatex not installed, please, install it to use rendering

  0%|                                                                      | 0/1 [00:00<?, ?it/s]
[ INFO ] Confidence score is 0.9954013824462891
100%|██████████████████████████████████████████████████████████████████████| 1/1 [00:00<00:00,  3.56it/s]
    Image name: sample.png
    Formula: 4 7 4 W ^ { 1 } + 7 . 1 9 o ^ { 4 } - 6 - 0 . 9 6 L ^ { 1 } y
```

gesture_recognition_demo

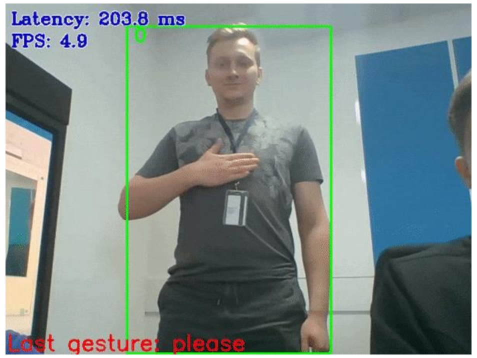
Function: sign language recognition

Introduction: Take ipynb as an example to identify the American Sign Language meaning represented by human hand movements. Please refer to “data/dataset_classes/msasl100.json” for the recognizable American Sign Language meaning

Source: video, image, webcam



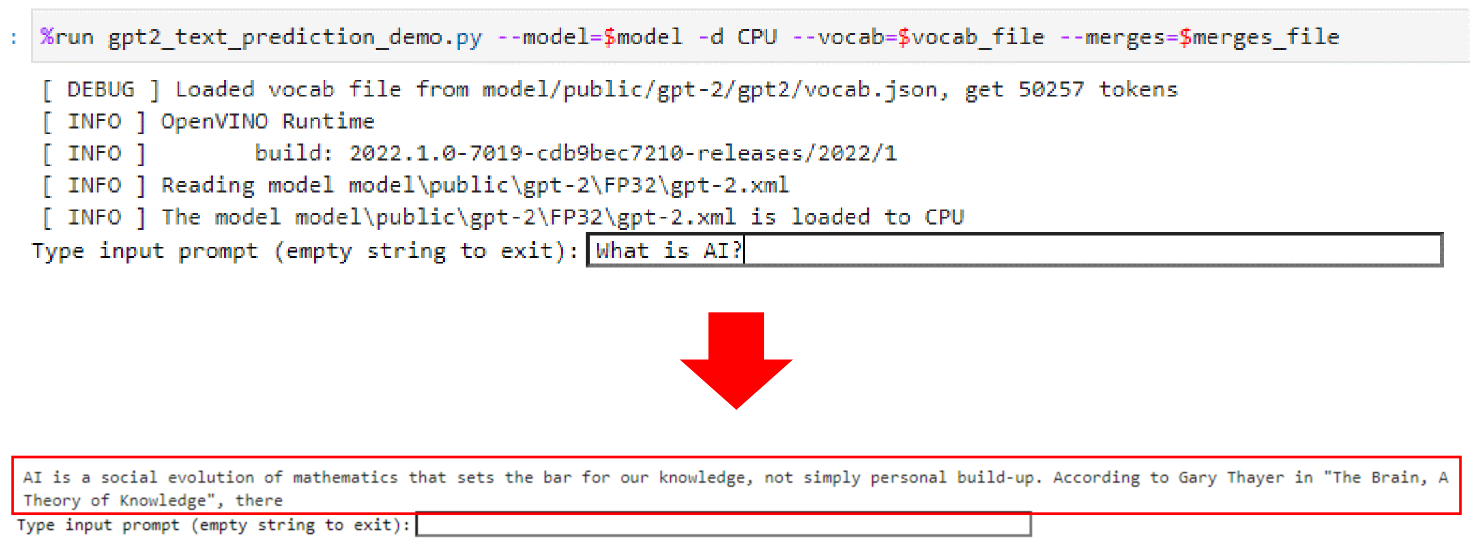
```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = 0
     model_a_name = "asl-recognition-0004"
     model_d_name = "person-detection-asl-0001"

[ ]: model_a = "model/intel/{}/FP32/{}.xml".format(model_a_name, model_a_name)
     model_d = "model/intel/{}/FP32/{}.xml".format(model_d_name, model_d_name)

     label_file = "../../../data/dataset_classes/msasl100.json"

[ ]: #download model
     if not os.path.exists(model_a):
         !omz_downloader --name $model_a_name --output_dir model/
     if not os.path.exists(model_d):
         !omz_downloader --name $model_d_name --output_dir model/

[ ]: %run gesture_recognition_demo.py -i $source -m_a  $model_a -m_d $model_d -d CPU -c $label_file
```



gpt2_text_prediction_demo

Feature: GPT2 Predicted Text

Introduction: Parse the input sentence and output articles related to the sentence



```
%run gpt2_text_prediction_demo.py --model=$model -d CPU --vocab=$vocab_file --merges=$merges_file
```

```
[ DEBUG ] Loaded vocab file from model/public/gpt-2/gpt2/vocab.json, get 50257 tokens
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
[ INFO ] Reading model model\public\gpt-2\FP32\gpt-2.xml
[ INFO ] The model model\public\gpt-2\FP32\gpt-2.xml is loaded to CPU
Type input prompt (empty string to exit): What is AI?
```

```
AI is a social evolution of mathematics that sets the bar for our knowledge, not simply personal build-up. According to Gary Thayer in "The Brain, A
Theory of Knowledge", there
Type input prompt (empty string to exit):
```

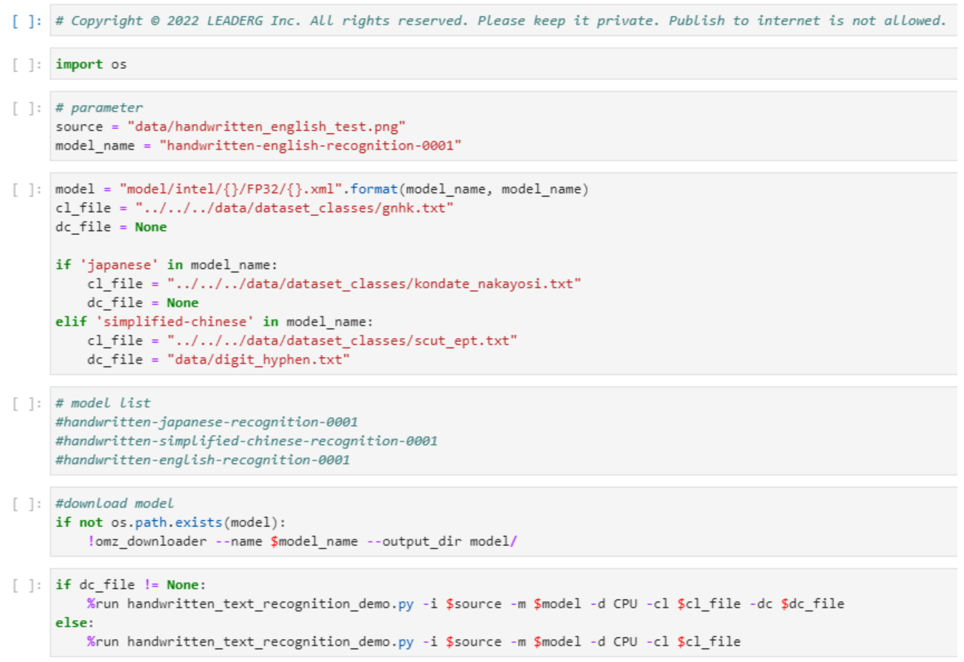
handwritten_text_recognition_demo

Function: handwriting recognition

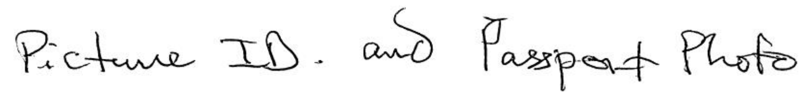
Introduction: Recognizing handwritten English

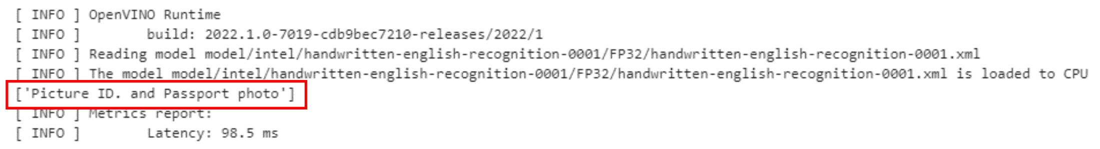
Source: image

Note: If you want to recognize handwritten Japanese, please change the model_name parameter to "handwritten-japanese-recognition-0001" and run it; if you want to recognize handwritten simplified Chinese, please change the model_name parameter to "handwritten-simplified-chinese-recognition-0001" and run it.



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/handwritten_english_test.png"
     model_name = "handwritten-english-recognition-0001"

[ ]: model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
     cl_file = "../../../data/dataset_classes/gnhk.txt"
     dc_file = None

     if 'japanese' in model_name:
         cl_file = "../../../data/dataset_classes/kondate_nakayosi.txt"
         dc_file = None
     elif 'simplified-chinese' in model_name:
         cl_file = "../../../data/dataset_classes/scut_ept.txt"
         dc_file = "data/digit_hyphen.txt"

[ ]: # model list
     #handwritten-japanese-recognition-0001
     #handwritten-simplified-chinese-recognition-0001
     #handwritten-english-recognition-0001

[ ]: #download model
     if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if dc_file != None:
         %run handwritten_text_recognition_demo.py -i $source -m $model -d CPU -cl $cl_file -dc $dc_file
     else:
         %run handwritten_text_recognition_demo.py -i $source -m $model -d CPU -cl $cl_file
```





```
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
[ INFO ] Reading model model/intel/handwritten-english-recognition-0001/FP32/handwritten-english-recognition-0001.xml
[ INFO ] The model model/intel/handwritten-english-recognition-0001/FP32/handwritten-english-recognition-0001.xml is loaded to CPU
['Picture ID. and Passport photo']
[ INFO ] Metrics report:
[ INFO ]        Latency: 98.5 ms
```

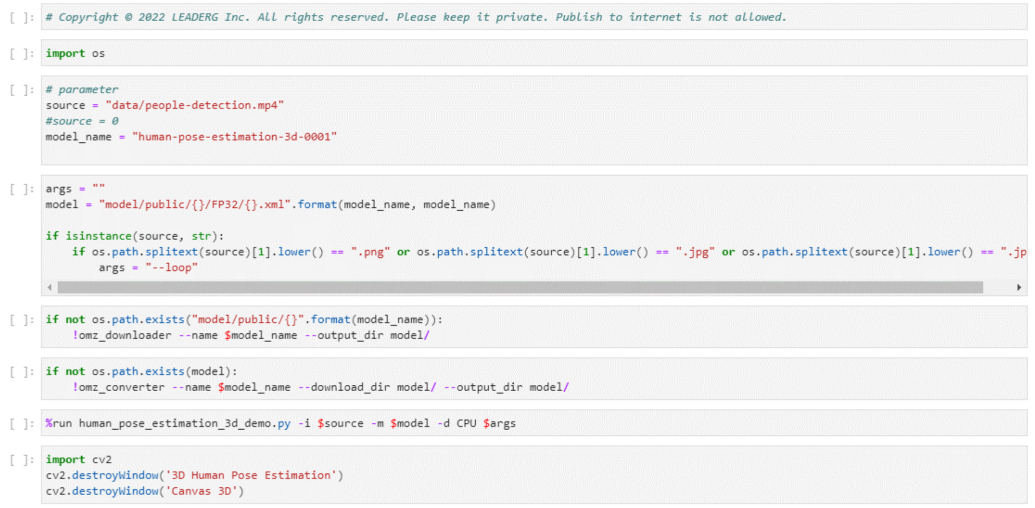
human_pose_estimation_3d_demo

Function: 3D human posture detection

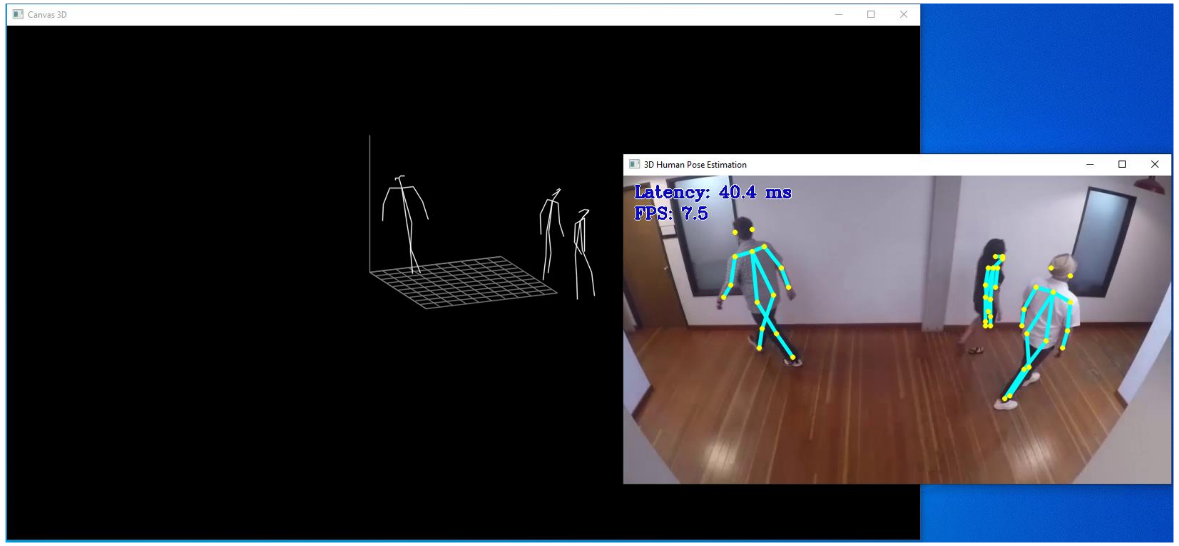
Introduction: Identify the position and posture of a person in 3D space

Source: video, image, webcam

Remarks: The black coordinate axis can be dragged with the mouse to change the angle of the coordinate axis for viewing



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/people-detection.mp4"
     #source = 0
     model_name = "human-pose-estimation-3d-0001"

[ ]: args = ""
     model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)

     if isinstance(source, str):
         if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
             args = "--loop"

[ ]: if not os.path.exists("model/public/{}".format(model_name)):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if not os.path.exists(model):
         !omz_converter --name $model_name --download_dir model/ --output_dir model/

[ ]: %run human_pose_estimation_3d_demo.py -i $source -m $model -d CPU $args

[ ]: import cv2
     cv2.destroyWindow('3D Human Pose Estimation')
     cv2.destroyWindow('Canvas 3D')
```

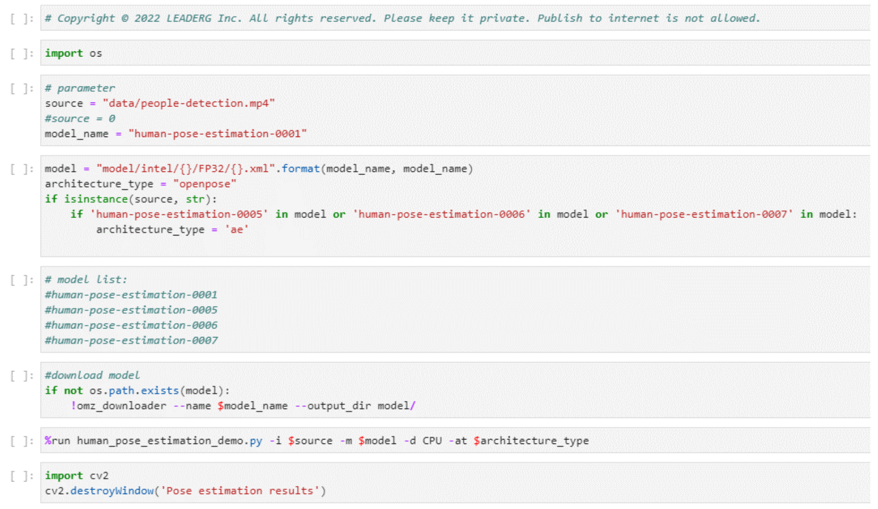


human_pose_estimation_demo

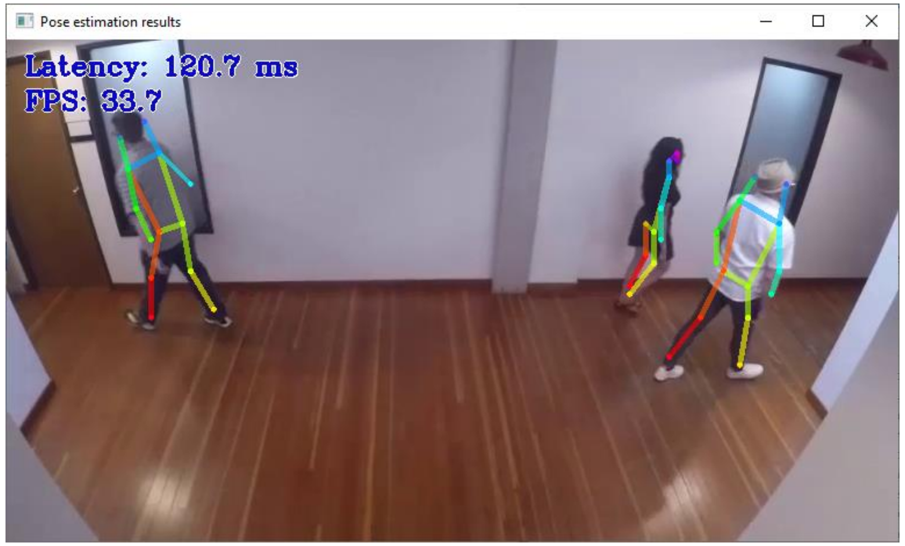
Function: human pose detection

Introduction: Identify the position of people's eyes, nose, ears, neck, shoulders, elbows, wrists, hips, knees, ankles

Source: video, image, webcam



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/people-detection.mp4"
     #source = 0
     model_name = "human-pose-estimation-0001"

[ ]: model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
     architecture_type = "openpose"
     if isinstance(source, str):
         if 'human-pose-estimation-0005' in model or 'human-pose-estimation-0006' in model or 'human-pose-estimation-0007' in model:
             architecture_type = 'ae'

[ ]: # model list:
     #human-pose-estimation-0001
     #human-pose-estimation-0005
     #human-pose-estimation-0006
     #human-pose-estimation-0007

[ ]: #download model
     if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/

[ ]: %run human_pose_estimation_demo.py -i $source -m $model -d CPU -at $architecture_type

[ ]: import cv2
     cv2.destroyWindow('Pose estimation results')
```



image_inpainting_demo

Function: image inpainting

Introduction: Doodle the image and press the spacebar or Enter key to inpainting it

Source: video, image, webcam

Remark:

The brush size slider can change the thickness of the graffiti pen

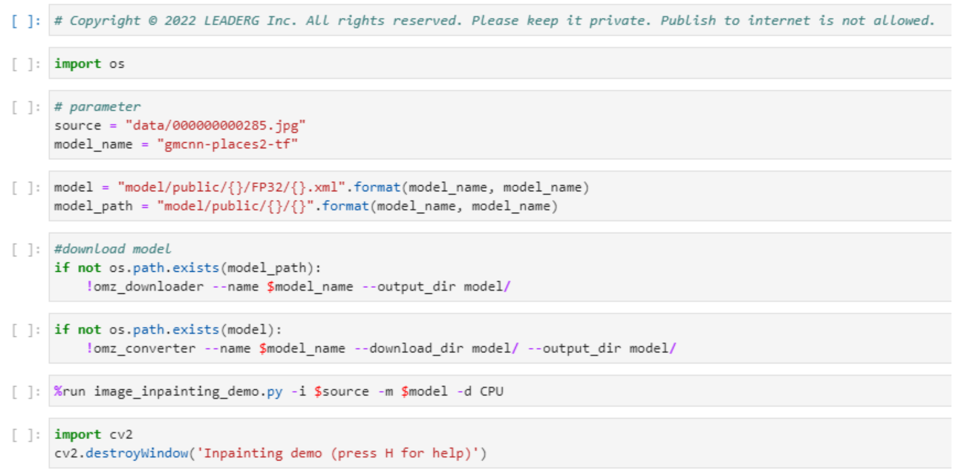
Press Backspace or c to clear graffiti

Press Spacebar or Enter to inpainting

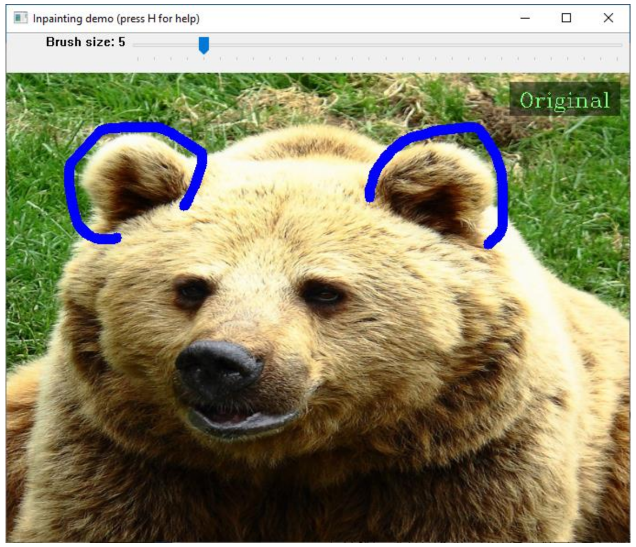
Press r to go back to the original doodle page

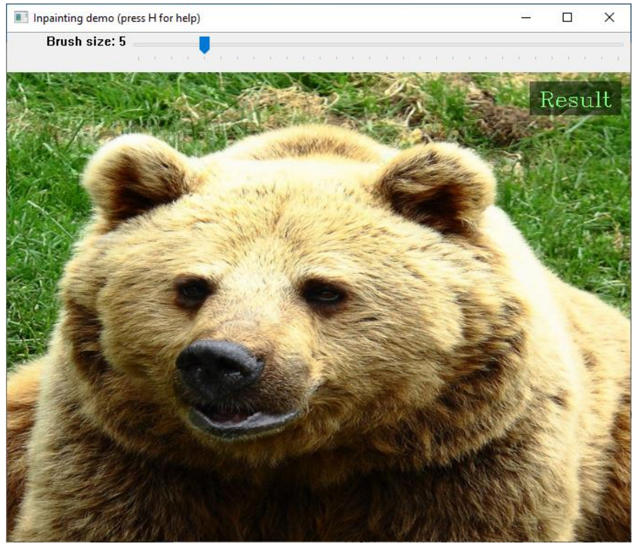
Press the Tab key to switch between the original image and the repaired result

Press Esc or q to leave



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/000000000285.jpg"
     model_name = "gmcnn-places2-tf"

[ ]: model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
     model_path = "model/public/{}/{}".format(model_name, model_name)

[ ]: #download model
     if not os.path.exists(model_path):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if not os.path.exists(model):
         !omz_converter --name $model_name --download_dir model/ --output_dir model/

[ ]: %run image_inpainting_demo.py -i $source -m $model -d CPU

[ ]: import cv2
     cv2.destroyWindow('Inpainting demo (press H for help)')
```





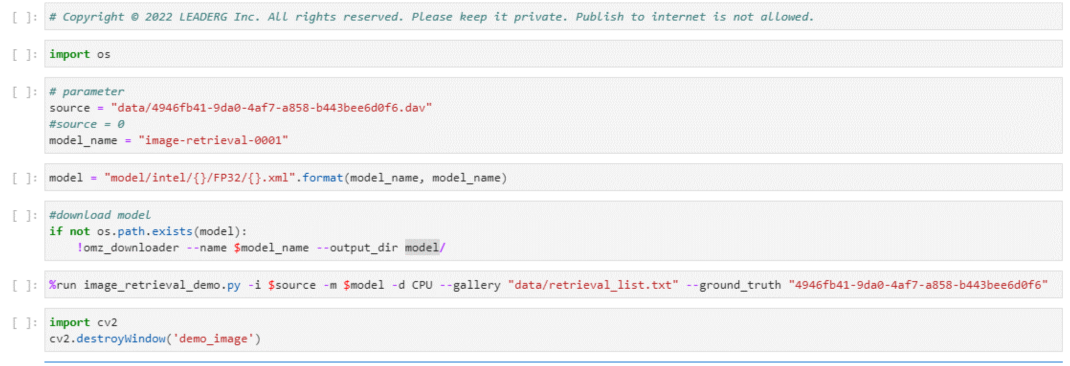
image_retrieval_demo

Function: image retrieval

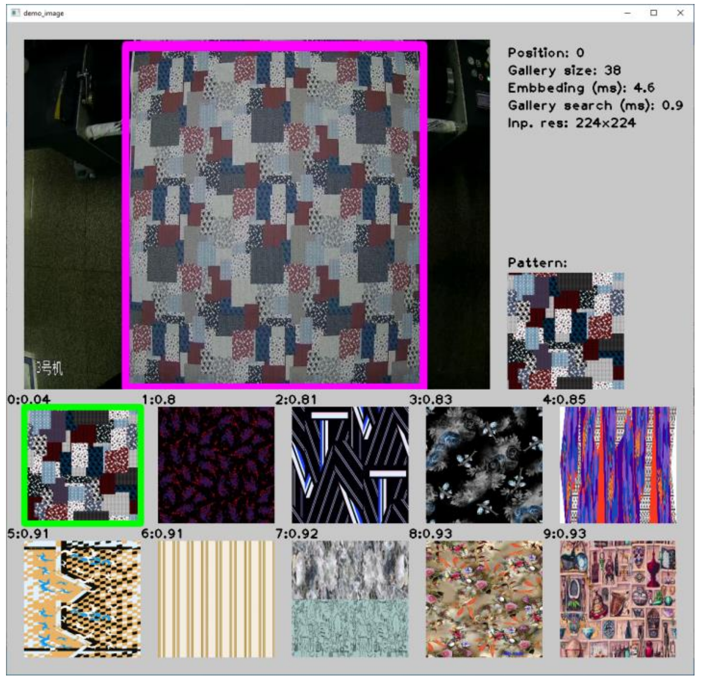
Introduction: Find out the pattern of the fabric in the video and list the most similar pattern

Source: video, image, webcam

Inference source: https://github.com/19900531/test



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/4946fb41-9da0-4af7-a858-b443bee6d0f6.dav"
     #source = 0
     model_name = "image-retrieval-0001"

[ ]: model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)

[ ]: #download model
     if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/

[ ]: %run image_retrieval_demo.py -i $source -m $model -d CPU --gallery "data/retrieval_list.txt" --ground_truth "4946fb41-9da0-4af7-a858-b443bee6d0f6"

[ ]: import cv2
     cv2.destroyWindow('demo_image')
```



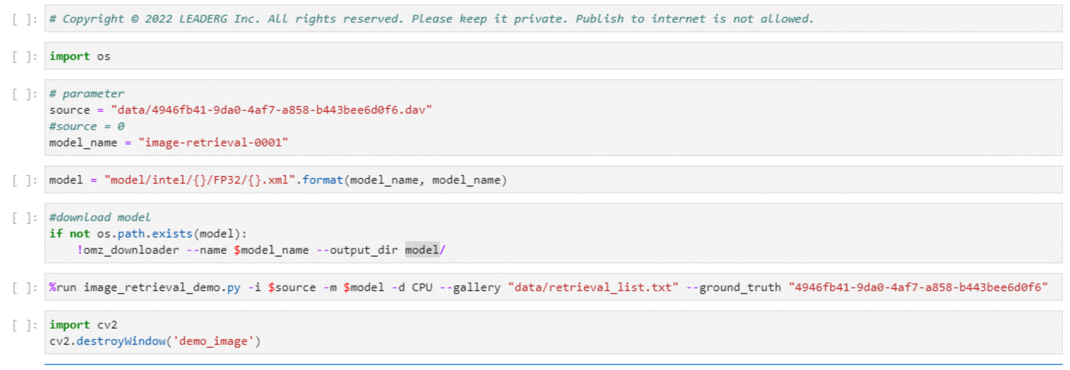
```
Position: 0
Gallery size: 38
Embbeding (ms): 4.6
Gallery search (ms): 0.9
Inp. res: 224x224
```

image_translation_demo

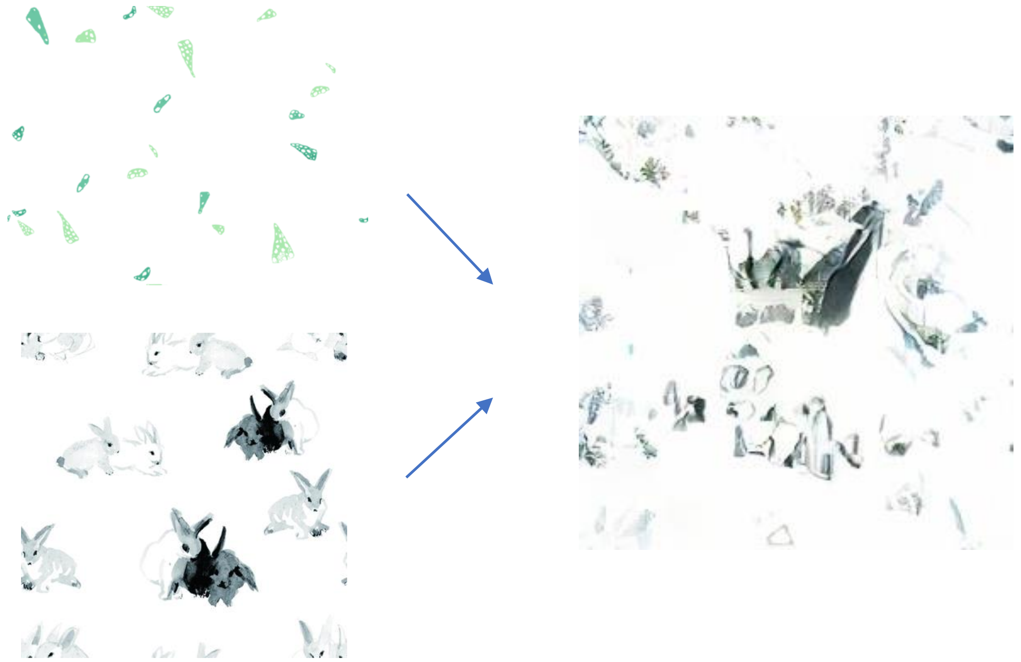
Function: image translation

Introduction: using neural networks to synthesize a photo-realistic image based on an exemplar image.

Source: video, image, webcam



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/4946fb41-9da0-4af7-a858-b443bee6d0f6.dav"
     #source = 0
     model_name = "image-retrieval-0001"

[ ]: model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)

[ ]: #download model
     if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/

[ ]: %run image_retrieval_demo.py -i $source -m $model -d CPU --gallery "data/retrieval_list.txt" --ground_truth "4946fb41-9da0-4af7-a858-b443bee6d0f6"

[ ]: import cv2
     cv2.destroyWindow('demo_image')
```



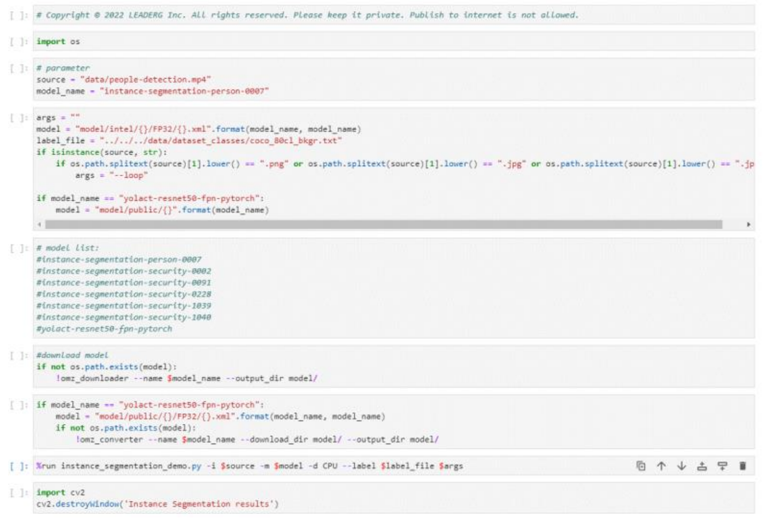
Inference source:
https://github.com/19900531/test

instance_segmentation_demo

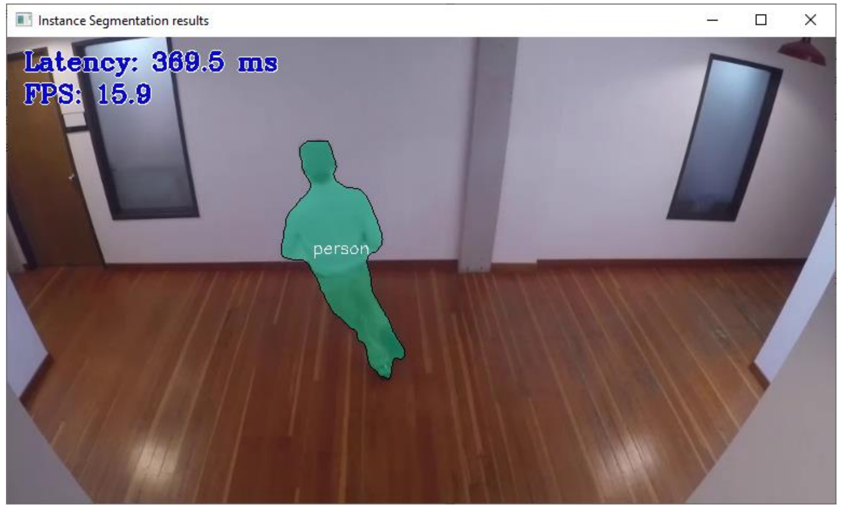
Function: image segmentation

Introduction: Segment each category in the image, please refer to the content of “data/dataset_classes/coco_80cl_bkgr.txt” for the categories that can be split

Source: video, image, webcam



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/people-detection.mp4"
     model_name = "instance-segmentation-person-0007"

[ ]: args = ""
     model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
     label_file = "../../../data/dataset_classes/coco_80cl_bkgr.txt"
     if isinstance(source, str):
         if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
             args = "--loop"

     if model_name == "yolact-resnet50-fpn-pytorch":
         model = "model/public/{}".format(model_name)

[ ]: # model list:
     #instance-segmentation-person-0007
     #instance-segmentation-security-0002
     #instance-segmentation-security-0091
     #instance-segmentation-security-0228
     #instance-segmentation-security-1039
     #instance-segmentation-security-1040
     #yolact-resnet50-fpn-pytorch

[ ]: #download model
     if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if model_name == "yolact-resnet50-fpn-pytorch":
         model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
         if not os.path.exists(model):
             !omz_converter --name $model_name --download_dir model/ --output_dir model/

[ ]: %run instance_segmentation_demo.py -i $source -m $model -d CPU --label $label_file $args

[ ]: import cv2
     cv2.destroyWindow('Instance Segmentation results')
```



machine_translation_demo

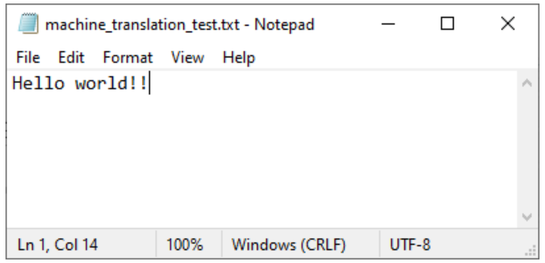
Function: English-Russian translation, English-German translation

Introduction: Take ipynb as an example to translate files from English to Russian

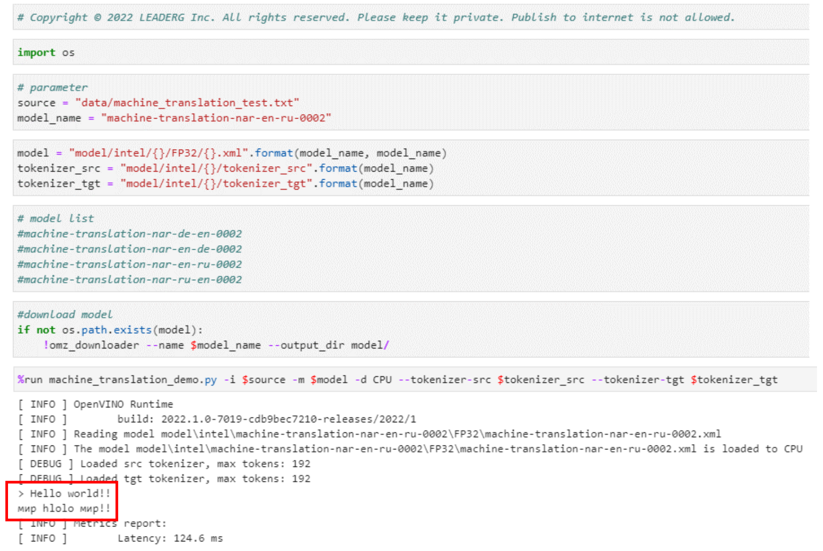
Source: txt file containing English sentences

Note: If you want to translate from English to German, please change the model_name parameter to

machine-translation-nar-en-de-0002


machine_translation_test.txt - Notepad
File Edit Format View Help
Ln 1, Col 14
100%
Windows (CRLF)
UTF-8

```
Hello world!!
```



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source = "data/machine_translation_test.txt"
model_name = "machine-translation-nar-en-ru-0002"
```

```
model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
tokenizer_src = "model/intel/{}/tokenizer_src".format(model_name)
tokenizer_tgt = "model/intel/{}/tokenizer_tgt".format(model_name)
```

```
# model list
#machine-translation-nar-de-en-0002
#machine-translation-nar-en-de-0002
#machine-translation-nar-en-ru-0002
#machine-translation-nar-ru-en-0002
```

```
#download model
if not os.path.exists(model):
    !omz_downloader --name $model_name --output_dir model/
```

```
%run machine_translation_demo.py -i $source -m $model -d CPU --tokenizer-src $tokenizer_src --tokenizer-tgt $tokenizer_tgt
```

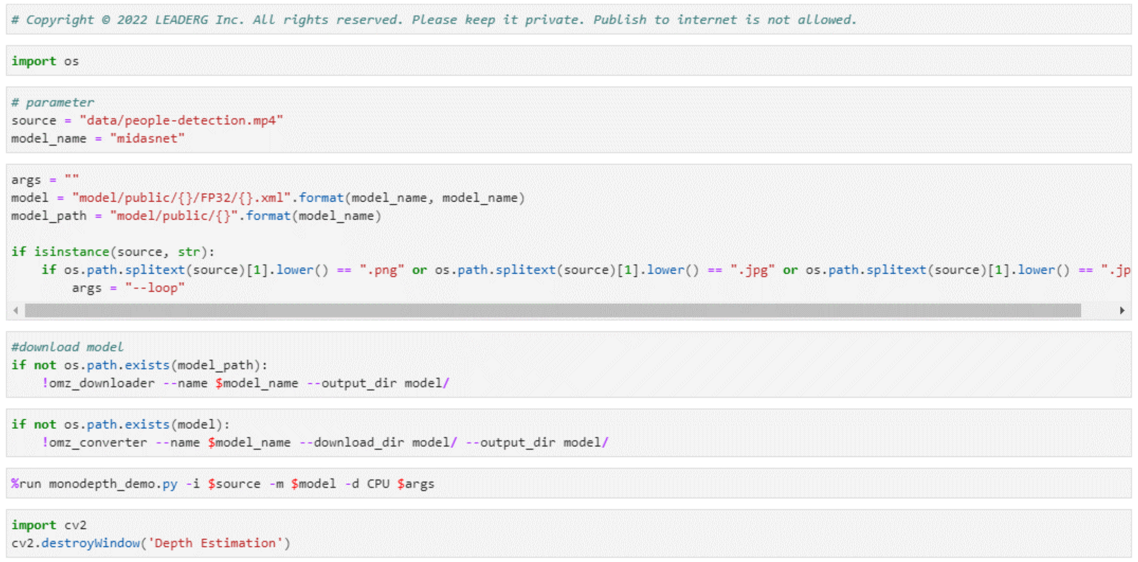
```
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
[ INFO ] Reading model model\intel\machine-translation-nar-en-ru-0002\FP32\machine-translation-nar-en-ru-0002.xml
[ INFO ] The model model\intel\machine-translation-nar-en-ru-0002\FP32\machine-translation-nar-en-ru-0002.xml is loaded to CPU
[ DEBUG ] Loaded src tokenizer, max tokens: 192
[ DEBUG ] Loaded tgt tokenizer, max tokens: 192
> Hello world!!
мир hlolo мир!!
[ INFO ] Metrics report:
[ INFO ]        Latency: 124.6 ms
```

monodepth_demo

Function: disparity map from image

Introduction: Generate disparity map from input image

Source: video, image, webcam



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source = "data/people-detection.mp4"
model_name = "midasnet"
```

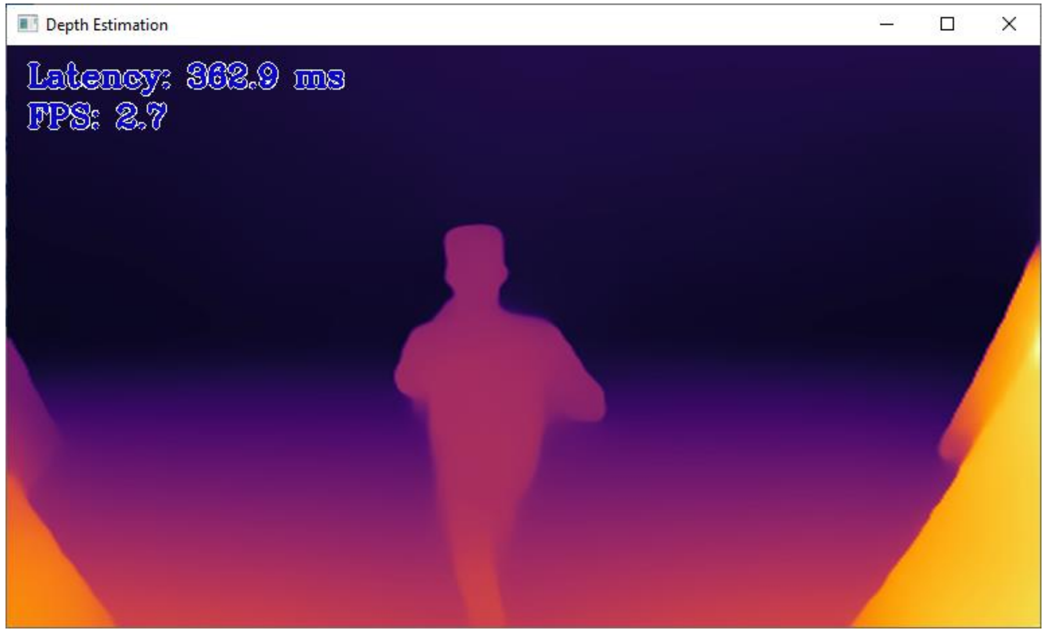
```
args = ""
model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
model_path = "model/public/{}".format(model_name)

if isinstance(source, str):
    if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
        args = "--loop"
```

```
#download model
if not os.path.exists(model_path):
    !omz_downloader --name $model_name --output_dir model/
```

```
if not os.path.exists(model):
    !omz_converter --name $model_name --download_dir model/ --output_dir model/
```

```
%run monodepth_demo.py -i $source -m $model -d CPU $args
```

```
import cv2
cv2.destroyWindow('Depth Estimation')
```



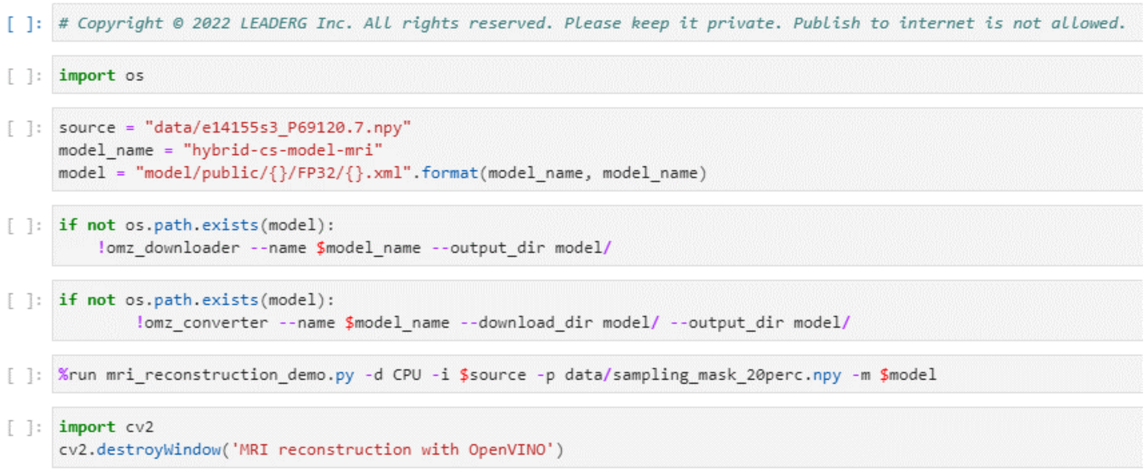
mri_reconstruction_demo

Function: MRI magnetic resonance imaging reconstruction

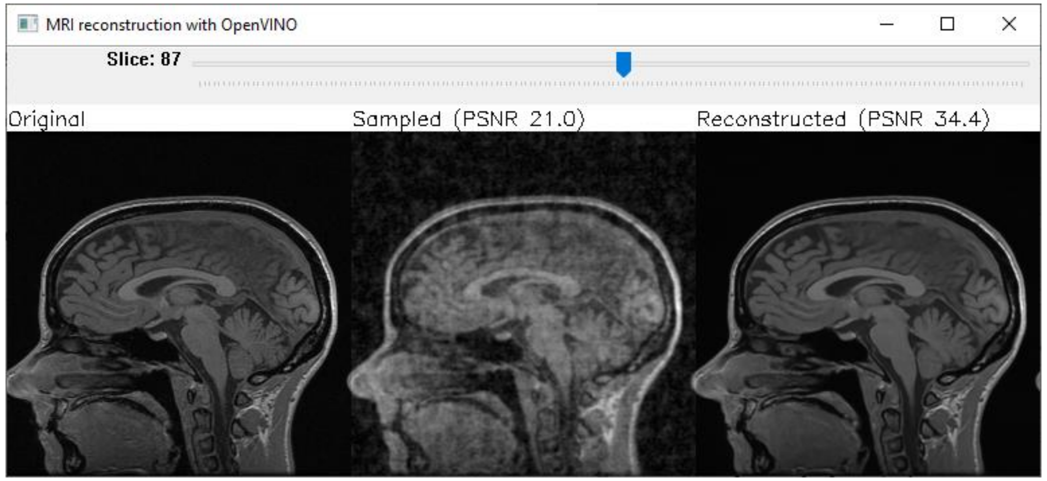
Introduction: Reconstructing MRI magnetic resonance imaging images, the upper pull bar can see different slices, the left is the original image, and the right is the reconstructed image, which is very useful for MRI with insufficient sampling

Source file (source): MRI image npy file

Inference source: https://sites.google.com/view/calgary-campinas-dataset/home



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: source = "data/e14155s3_P69120.7.npy"
     model_name = "hybrid-cs-model-mri"
     model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)

[ ]: if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if not os.path.exists(model):
             !omz_converter --name $model_name --download_dir model/ --output_dir model/

[ ]: %run mri_reconstruction_demo.py -d CPU -i $source -p data/sampling_mask_20perc.npy -m $model

[ ]: import cv2
     cv2.destroyWindow('MRI reconstruction with OpenVINO')
```

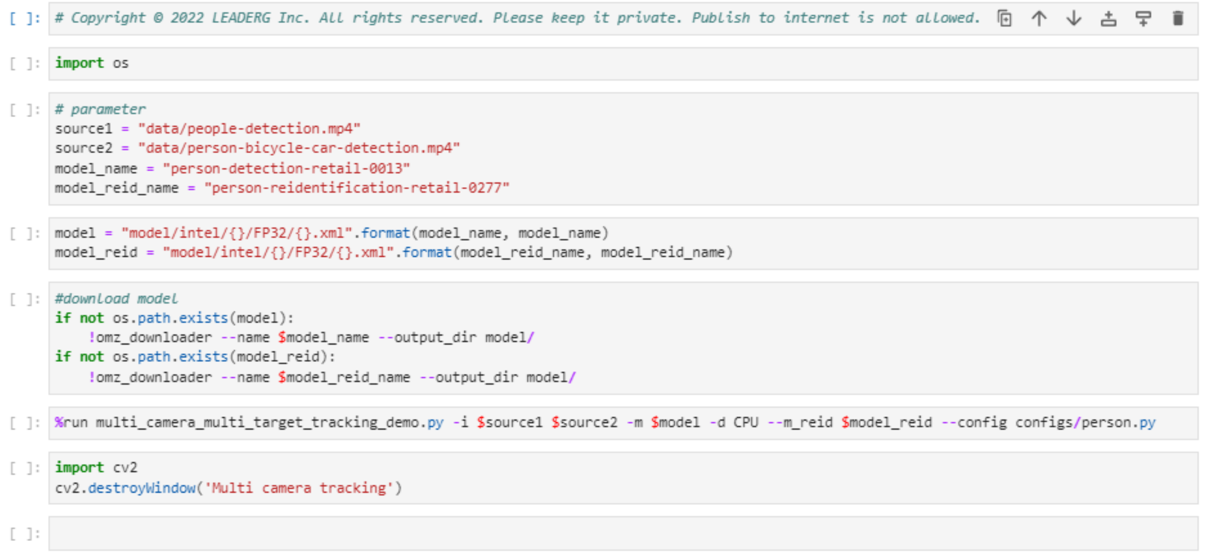


multi_camera_multi_target_tracking_demo

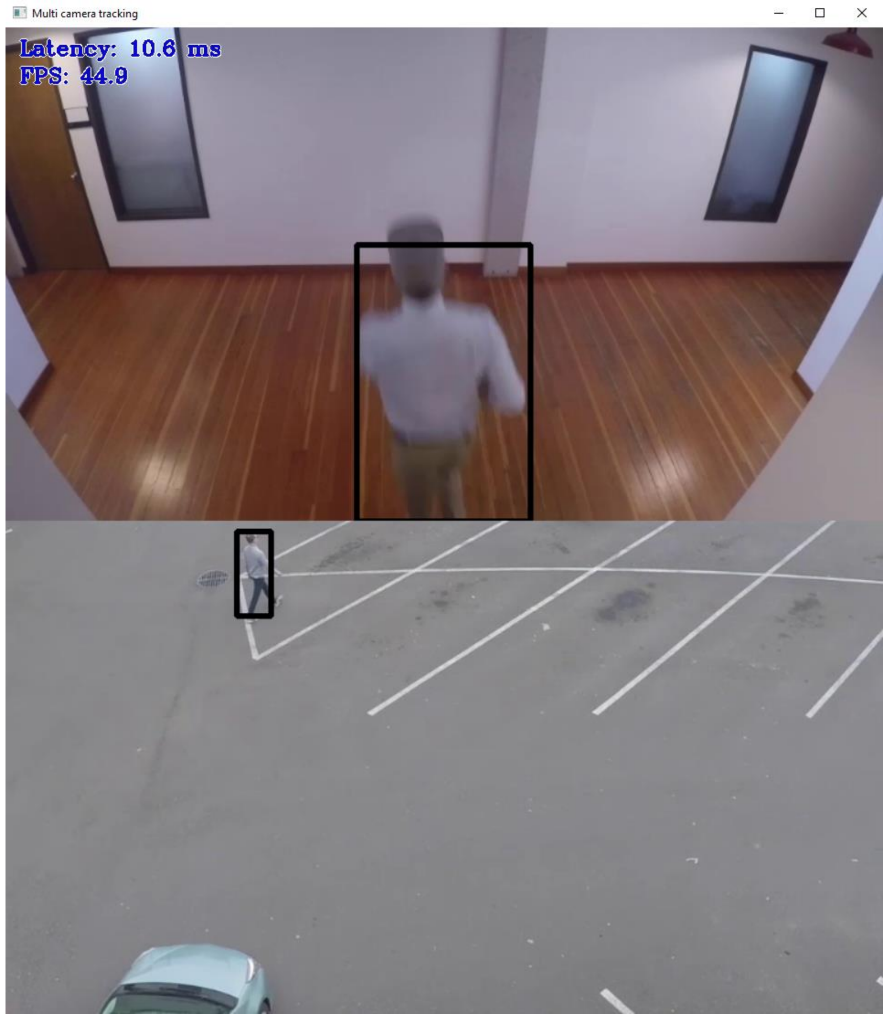
Function: multi-camera object tracking

Introduction: Track images from multiple cameras

Source: video



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source1 = "data/people-detection.mp4"
     source2 = "data/person-bicycle-car-detection.mp4"
     model_name = "person-detection-retail-0013"
     model_reid_name = "person-reidentification-retail-0277"

[ ]: model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
     model_reid = "model/intel/{}/FP32/{}.xml".format(model_reid_name, model_reid_name)

[ ]: #download model
     if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/
     if not os.path.exists(model_reid):
         !omz_downloader --name $model_reid_name --output_dir model/

[ ]: %run multi_camera_multi_target_tracking_demo.py -i $source1 $source2 -m $model -d CPU --m_reid $model_reid --config configs/person.py

[ ]: import cv2
     cv2.destroyWindow('Multi camera tracking')

[ ]:
```

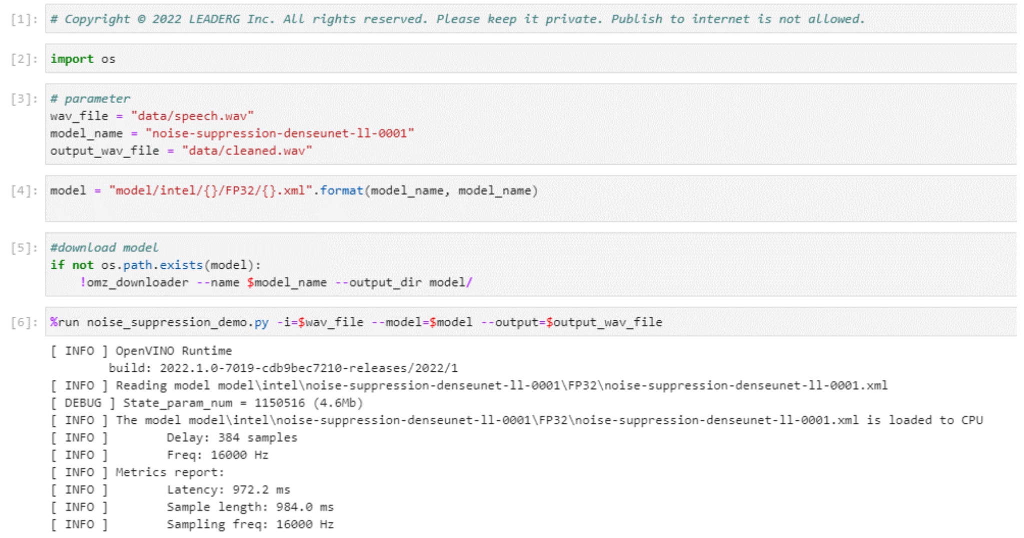


noise_suppression_demo

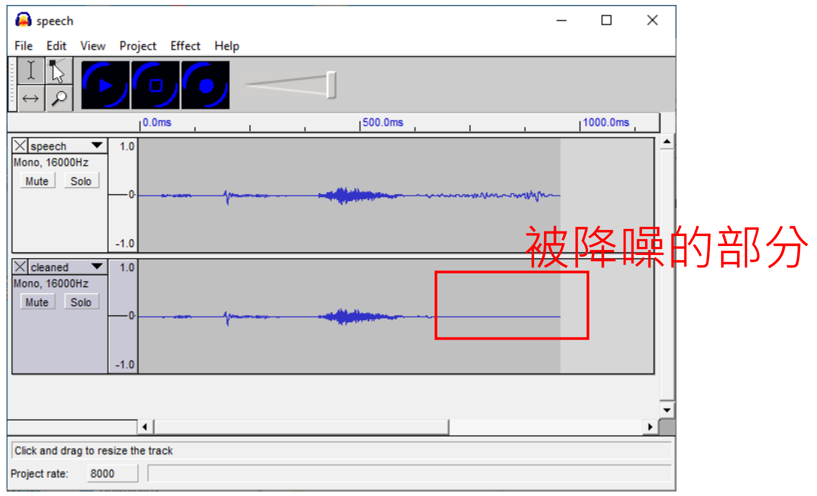
Function: noise suppression

Introduction: Noise reduction is performed on the source audio file, and the denoised audio file is stored in data/cleaned.wav

Source：wav file



```
[1]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[2]: import os

[3]: # parameter
     wav_file = "data/speech.wav"
     model_name = "noise-suppression-denseunet-ll-0001"
     output_wav_file = "data/cleaned.wav"

[4]: model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)

[5]: #download model
     if not os.path.exists(model):
         !omz_downloader --name $model_name --output_dir model/

[6]: %run noise_suppression_demo.py -i=$wav_file --model=$model --output=$output_wav_file
     [ INFO ] OpenVINO Runtime
             build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
     [ INFO ] Reading model model\intel\noise-suppression-denseunet-ll-0001\FP32\noise-suppression-denseunet-ll-0001.xml
     [ DEBUG ] State_param_num = 1150516 (4.6Mb)
     [ INFO ] The model model\intel\noise-suppression-denseunet-ll-0001\FP32\noise-suppression-denseunet-ll-0001.xml is loaded to CPU
     [ INFO ]         Delay: 384 samples
     [ INFO ]         Freq: 16000 Hz
     [ INFO ] Metrics report:
     [ INFO ]         Latency: 972.2 ms
     [ INFO ]         Sample length: 984.0 ms
     [ INFO ]         Sampling freq: 16000 Hz
```

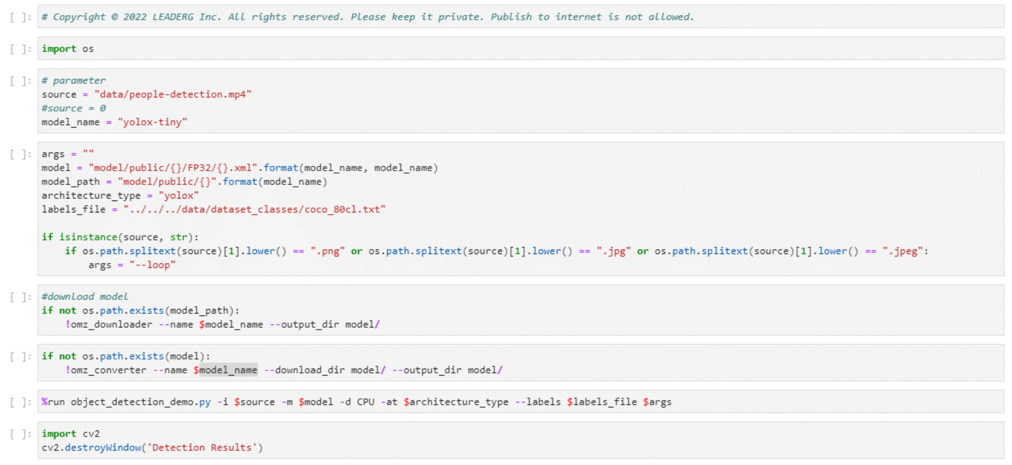


object_detection_demo

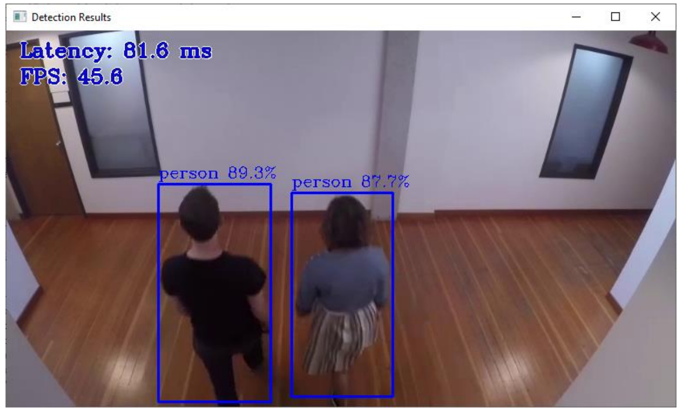
Function: object detection

Introduction: Detect objects in the image, please refer to the content of "data/dataset_classes/coco_80cl.txt" for the types that can be detected

Source: video, image, webcam



```
[ ]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[ ]: import os

[ ]: # parameter
     source = "data/people-detection.mp4"
     #source = 0
     model_name = "yolox-tiny"

[ ]: args = ""
     model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
     model_path = "model/public/{}".format(model_name)
     architecture_type = "yolox"
     labels_file = "../../../data/dataset_classes/coco_80cl.txt"

     if isinstance(source, str):
         if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jpeg":
             args = "--loop"

[ ]: #download model
     if not os.path.exists(model_path):
         !omz_downloader --name $model_name --output_dir model/

[ ]: if not os.path.exists(model):
         !omz_converter --name $model_name --download_dir model/ --output_dir model/

[ ]: %run object_detection_demo.py -i $source -m $model -d CPU -at $architecture_type --labels $labels_file $args

[ ]: import cv2
     cv2.destroyWindow('Detection Results')
```

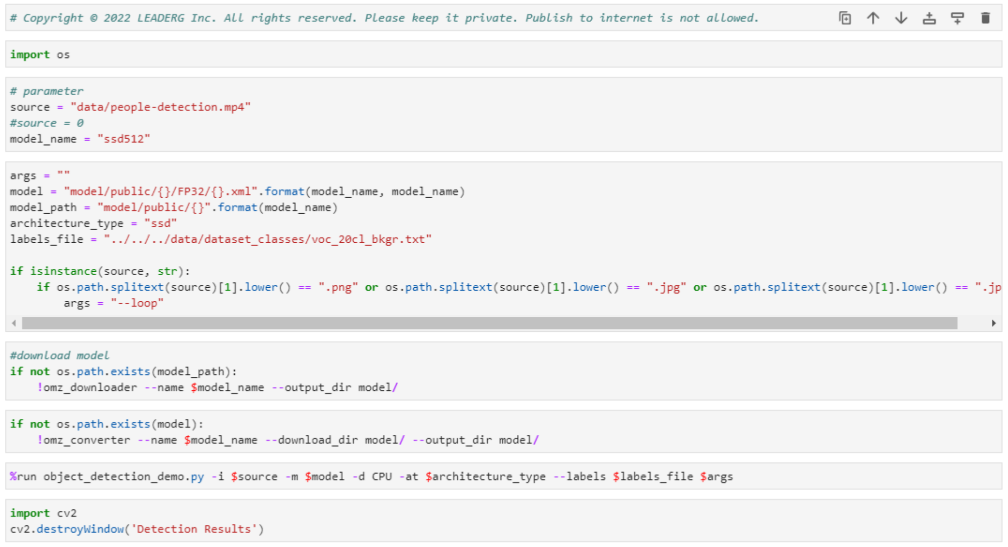


place_recognition_demo

Function: place recognition

Introduction: Identify locations based on the images in the “data/gallery_folder” folder, and find the closest top location images

Source: video, image, webcam



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source = "data/people-detection.mp4"
#source = 0
model_name = "ssd512"
```

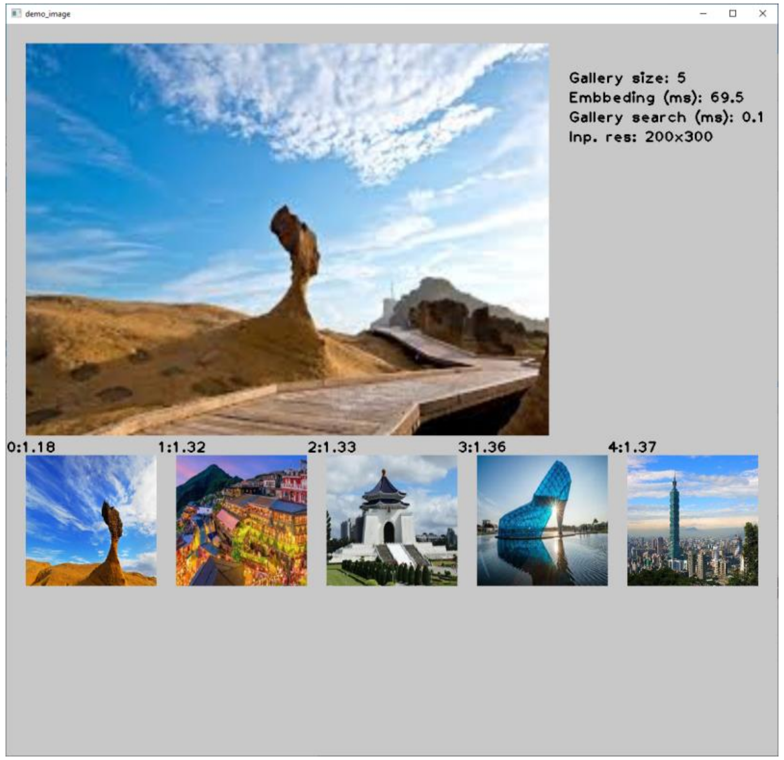
```
args = ""
model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
model_path = "model/public/{}".format(model_name)
architecture_type = "ssd"
labels_file = "../../../data/dataset_classes/voc_20cl_bkgr.txt"

if isinstance(source, str):
    if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
        args = "--loop"
```

```
#download model
if not os.path.exists(model_path):
    !omz_downloader --name $model_name --output_dir model/
```

```
if not os.path.exists(model):
    !omz_converter --name $model_name --download_dir model/ --output_dir model/
```

```
%run object_detection_demo.py -i $source -m $model -d CPU -at $architecture_type --labels $labels_file $args
```

```
import cv2
cv2.destroyWindow('Detection Results')
```



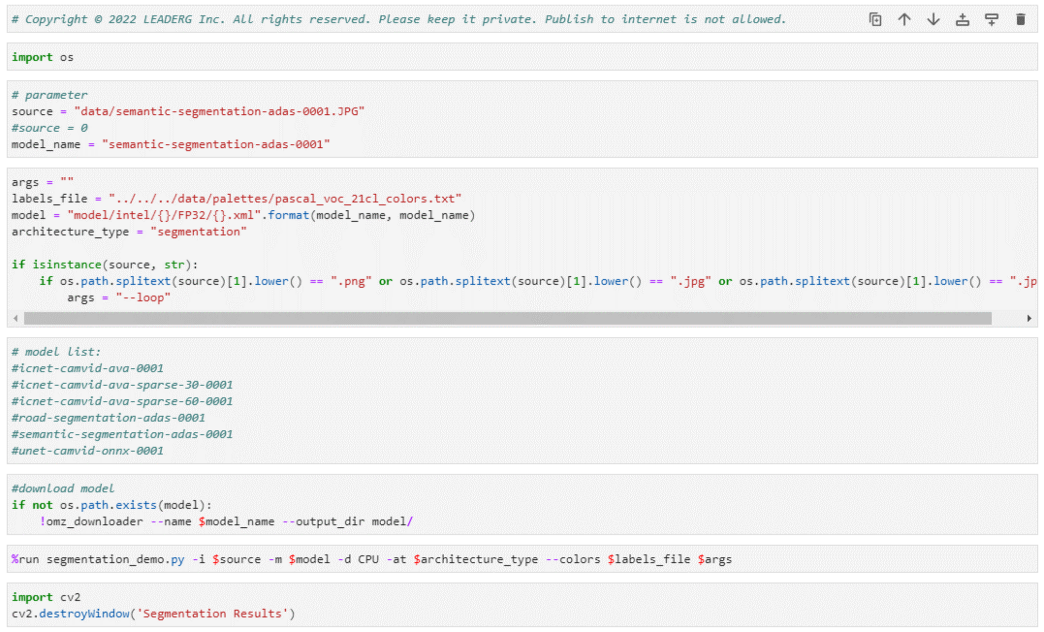
```
Gallery size: 5
Embbeding (ms): 69.5
Gallery search (ms): 0.1
Inp. res: 200x300
```

segmentation_demo

Function: image segmentation

Introduction: Segmentation of various categories of images

Source: video, image, webcam



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source = "data/semantic-segmentation-adas-0001.JPG"
#source = 0
model_name = "semantic-segmentation-adas-0001"
```

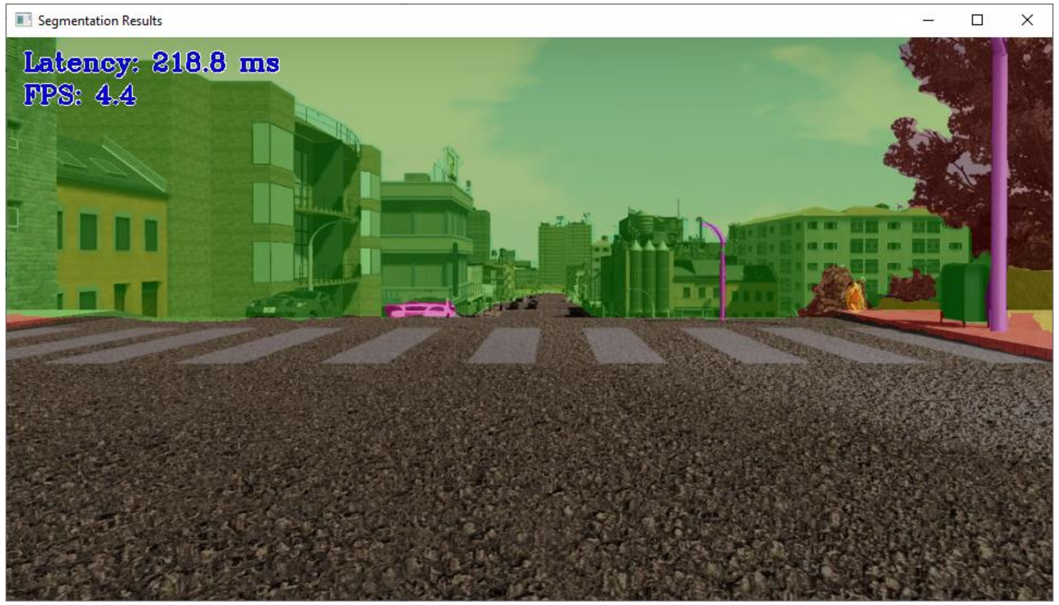
```
args = ""
labels_file = "../../../data/palettes/pascal_voc_21cl_colors.txt"
model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
architecture_type = "segmentation"

if isinstance(source, str):
    if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
        args = "--loop"
```

```
# model list:
#icnet-camvid-ava-0001
#icnet-camvid-ava-sparse-30-0001
#icnet-camvid-ava-sparse-60-0001
#road-segmentation-adas-0001
#semantic-segmentation-adas-0001
#unet-camvid-onnx-0001
```

```
#download model
if not os.path.exists(model):
    !omz_downloader --name $model_name --output_dir model/
```

```
%run segmentation_demo.py -i $source -m $model -d CPU -at $architecture_type --colors $labels_file $args
```

```
import cv2
cv2.destroyWindow('Segmentation Results')
```

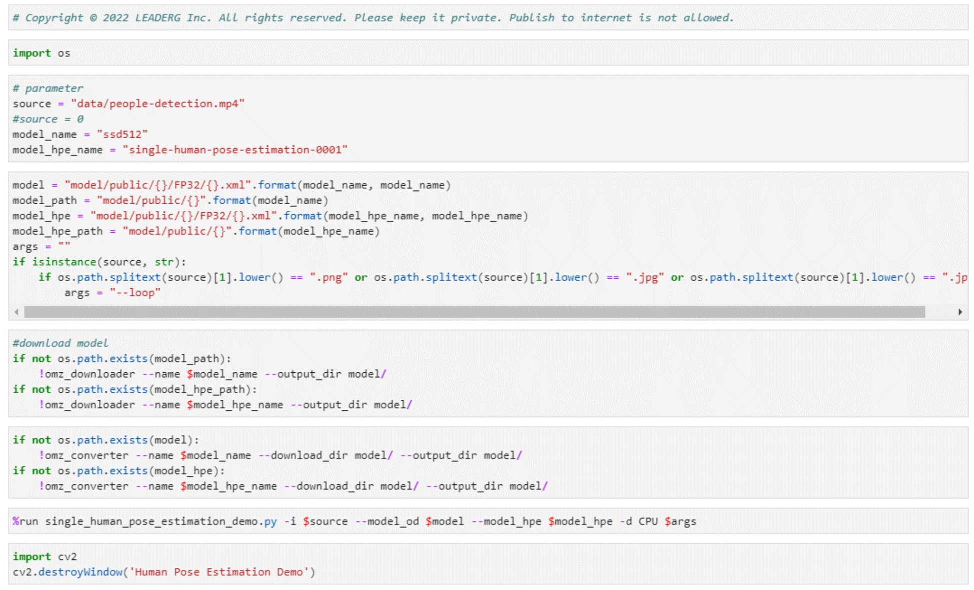


single_human_pose_estimation_demo

Function: single human pose detection

Introduction: Detect key points in each person's pose, ears, eyes, nose, shoulders, elbows, wrists, hips, knees and ankles

Source: video, image, webcam



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source = "data/people-detection.mp4"
#source = 0
model_name = "ssd512"
model_hpe_name = "single-human-pose-estimation-0001"
```

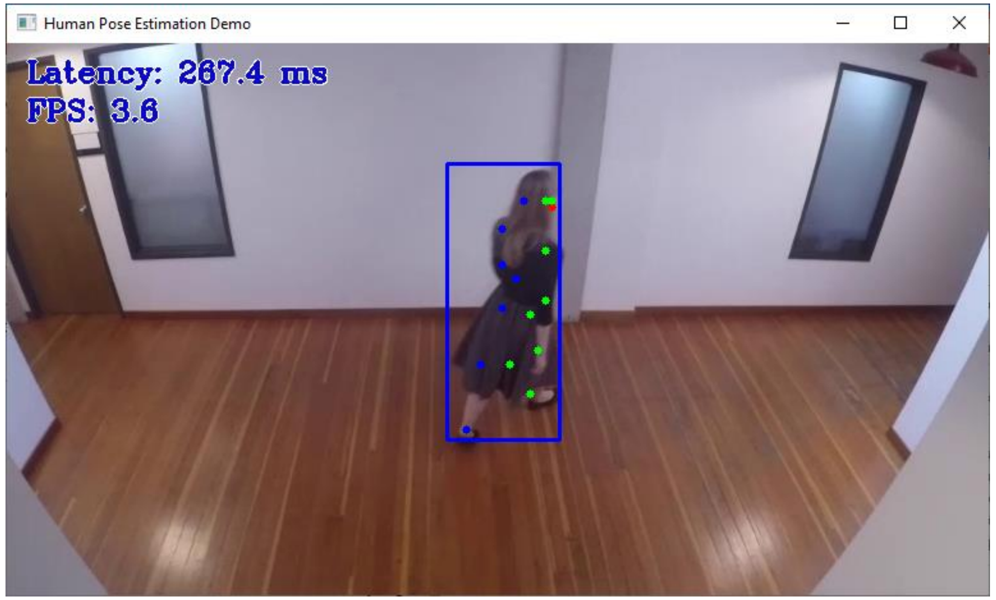
```
model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
model_path = "model/public/{}".format(model_name)
model_hpe = "model/public/{}/FP32/{}.xml".format(model_hpe_name, model_hpe_name)
model_hpe_path = "model/public/{}".format(model_hpe_name)
args = ""
if isinstance(source, str):
    if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
        args = "--loop"
```

```
#download model
if not os.path.exists(model_path):
    !omz_downloader --name $model_name --output_dir model/
if not os.path.exists(model_hpe_path):
    !omz_downloader --name $model_hpe_name --output_dir model/
```

```
if not os.path.exists(model):
    !omz_converter --name $model_name --download_dir model/ --output_dir model/
if not os.path.exists(model_hpe):
    !omz_converter --name $model_hpe_name --download_dir model/ --output_dir model/
```

```
%run single_human_pose_estimation_demo.py -i $source --model_od $model --model_hpe $model_hpe -d CPU $args
```

```
import cv2
cv2.destroyWindow('Human Pose Estimation Demo')
```

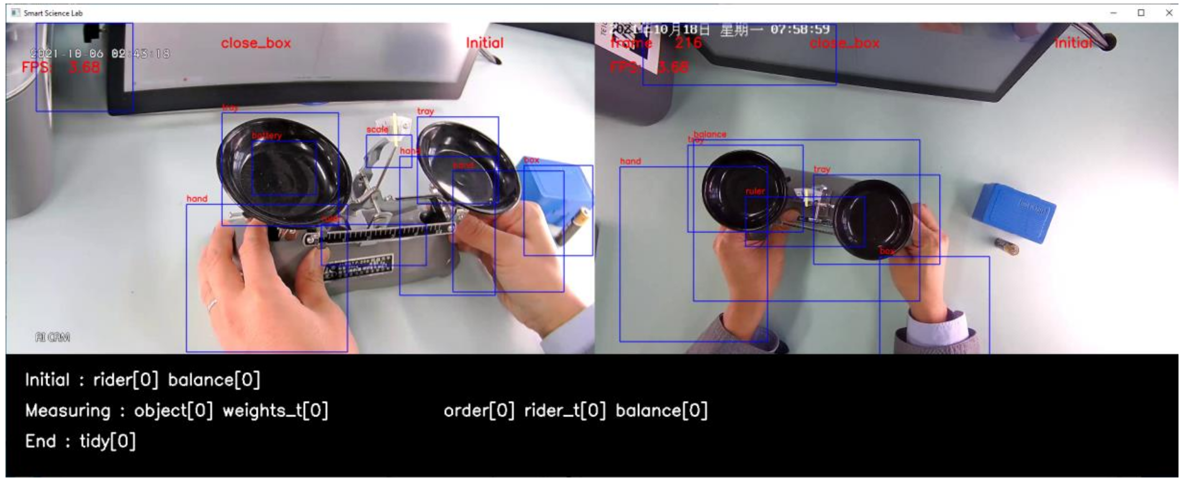


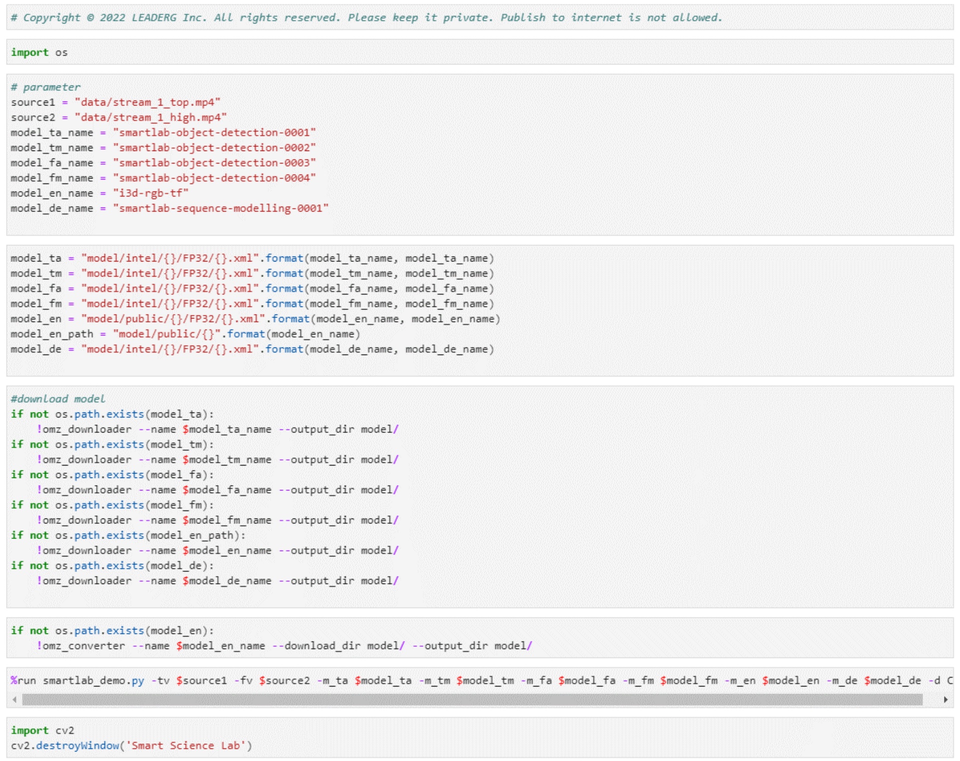
smartlab_demo

Function: action recognition

Introduction: Read two videos, one from front view and one from top view, identify balance, weight, tweezers, box, battery, tray, ruler, rider, scale, hand position in both videos and actions

Source: video





```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source1 = "data/stream_1_top.mp4"
source2 = "data/stream_1_high.mp4"
model_ta_name = "smartlab-object-detection-0001"
model_tm_name = "smartlab-object-detection-0002"
model_fa_name = "smartlab-object-detection-0003"
model_fm_name = "smartlab-object-detection-0004"
model_en_name = "i3d-rgb-tf"
model_de_name = "smartlab-sequence-modelling-0001"
```

```
model_ta = "model/intel/{}/FP32/{}.xml".format(model_ta_name, model_ta_name)
model_tm = "model/intel/{}/FP32/{}.xml".format(model_tm_name, model_tm_name)
model_fa = "model/intel/{}/FP32/{}.xml".format(model_fa_name, model_fa_name)
model_fm = "model/intel/{}/FP32/{}.xml".format(model_fm_name, model_fm_name)
model_en = "model/public/{}/FP32/{}.xml".format(model_en_name, model_en_name)
model_en_path = "model/public/{}".format(model_en_name)
model_de = "model/intel/{}/FP32/{}.xml".format(model_de_name, model_de_name)
```

```
#download model
if not os.path.exists(model_ta):
    !omz_downloader --name $model_ta_name --output_dir model/
if not os.path.exists(model_tm):
    !omz_downloader --name $model_tm_name --output_dir model/
if not os.path.exists(model_fa):
    !omz_downloader --name $model_fa_name --output_dir model/
if not os.path.exists(model_fm):
    !omz_downloader --name $model_fm_name --output_dir model/
if not os.path.exists(model_en_path):
    !omz_downloader --name $model_en_name --output_dir model/
if not os.path.exists(model_de):
    !omz_downloader --name $model_de_name --output_dir model/
```

```
if not os.path.exists(model_en):
    !omz_converter --name $model_en_name --download_dir model/ --output_dir model/
```

```
%run smartlab_demo.py -tv $source1 -fv $source2 -m_ta $model_ta -m_tm $model_tm -m_fa $model_fa -m_fm $model_fm -m_en $model_en -m_de $model_de -d C
```

```
import cv2
cv2.destroyWindow('Smart Science Lab')
```

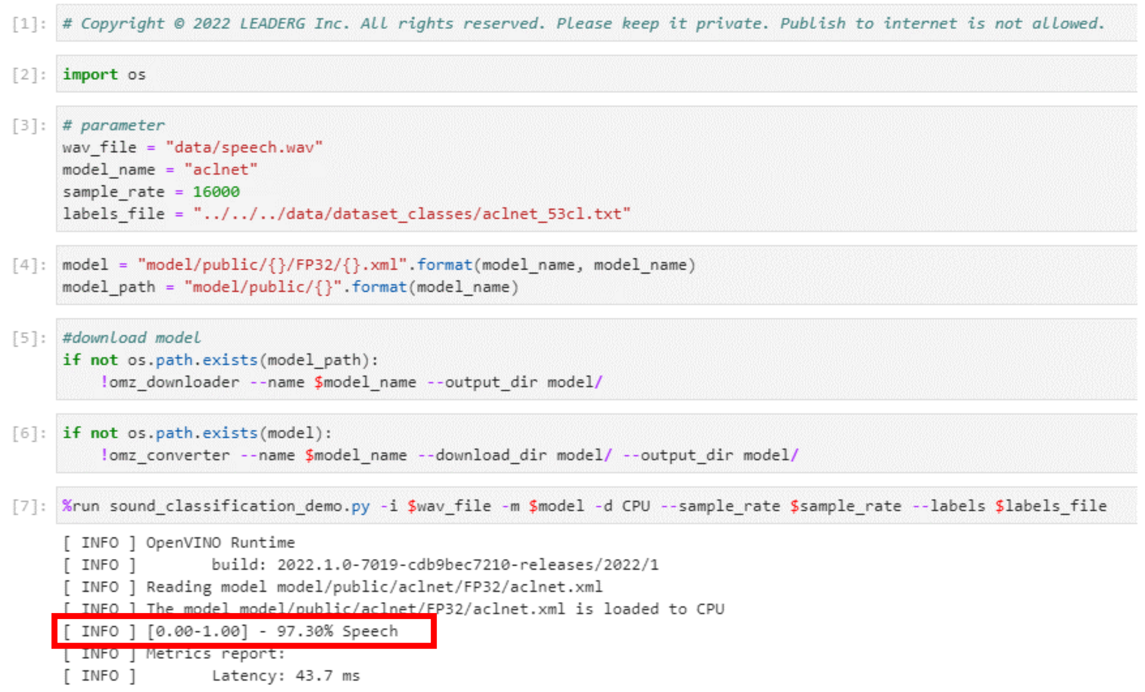
Inference video source:
https://storage.openvinotoolkit.org/data/test_data/videos/smartlab/

sound_classification_demo

Function: sound classification

Introduction: Read the wav audio file, classify which word the audio file is, please refer to the content of "data/dataset_classes/aclnet_53cl.txt" for the category name

Source: wav file



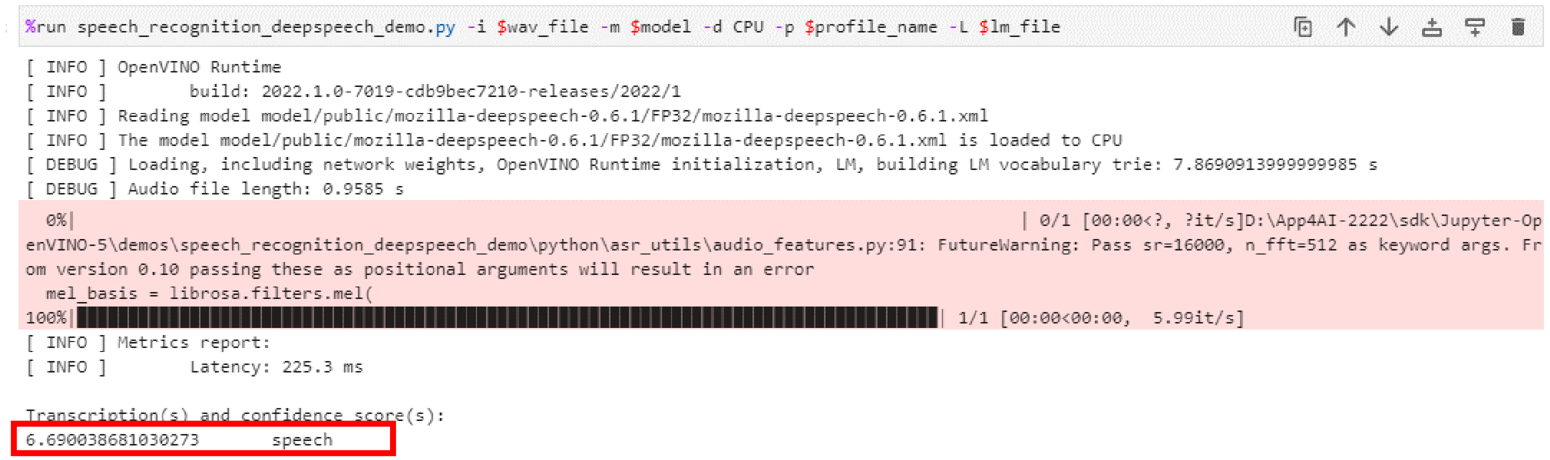
```
[1]: # Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.

[2]: import os

[3]: # parameter
     wav_file = "data/speech.wav"
     model_name = "aclnet"
     sample_rate = 16000
     labels_file = "../../../data/dataset_classes/aclnet_53cl.txt"

[4]: model = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
     model_path = "model/public/{}".format(model_name)

[5]: #download model
     if not os.path.exists(model_path):
         !omz_downloader --name $model_name --output_dir model/

[6]: if not os.path.exists(model):
         !omz_converter --name $model_name --download_dir model/ --output_dir model/

[7]: %run sound_classification_demo.py -i $wav_file -m $model -d CPU --sample_rate $sample_rate --labels $labels_file

     [ INFO ] OpenVINO Runtime
     [ INFO ]        build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
     [ INFO ] Reading model model/public/aclnet/FP32/aclnet.xml
     [ INFO ] The model model/public/aclnet/FP32/aclnet.xml is loaded to CPU
     [ INFO ] [0.00-1.00] - 97.30% Speech
     [ INFO ] Metrics report:
     [ INFO ]        Latency: 43.7 ms
```

speech_recognition_deepspeech_demo

Function: DeepSpeech speech recognition

Introduction: Read the wav audio file and identify the content of the audio file

Source: wav file



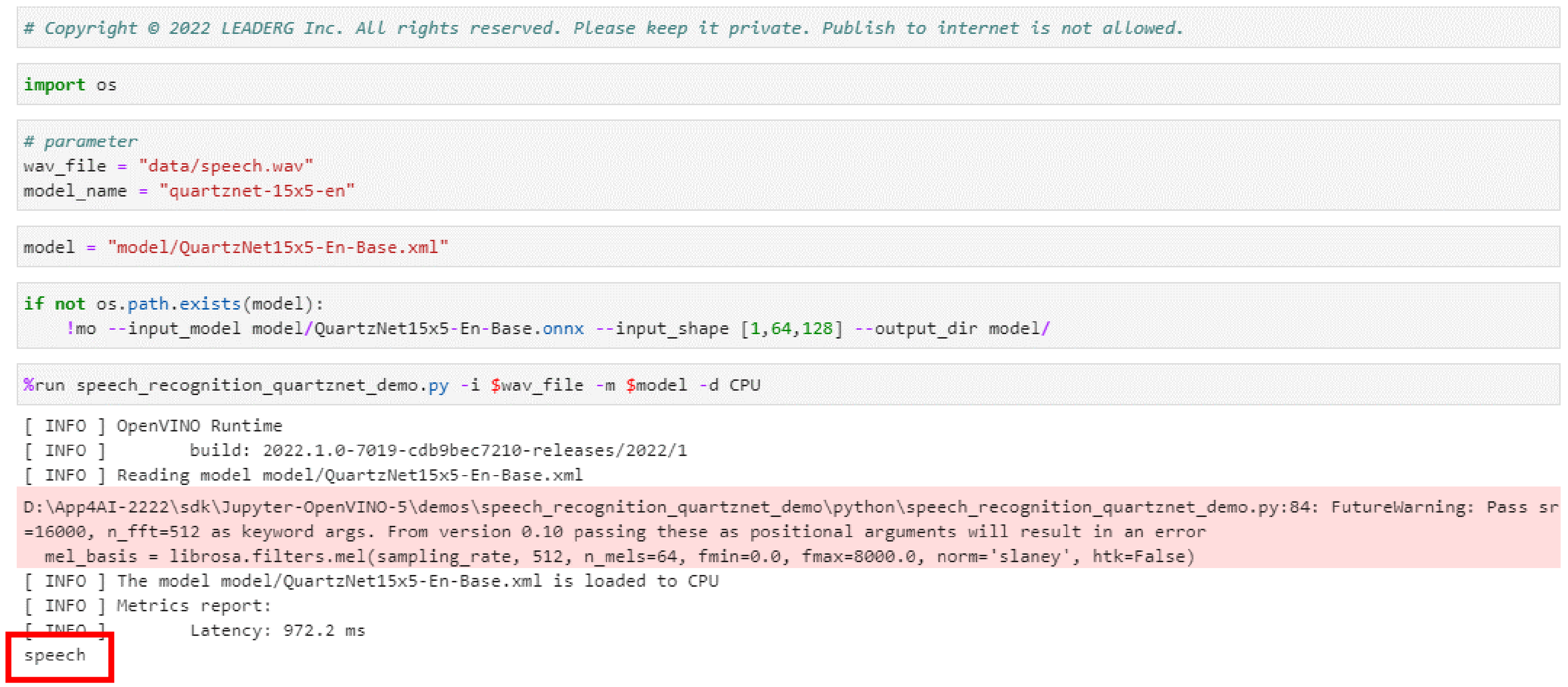
```
%run speech_recognition_deepspeech_demo.py -i $wav_file -m $model -d CPU -p $profile_name -L $lm_file
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
[ INFO ] Reading model model/public/mozilla-deepspeech-0.6.1/FP32/mozilla-deepspeech-0.6.1.xml
[ INFO ] The model model/public/mozilla-deepspeech-0.6.1/FP32/mozilla-deepspeech-0.6.1.xml is loaded to CPU
[ DEBUG ] Loading, including network weights, OpenVINO Runtime initialization, LM, building LM vocabulary trie: 7.8690913999999985 s
[ DEBUG ] Audio file length: 0.9585 s
  0%|                                                                                   | 0/1 [00:00<?, ?it/s]D:\App4AI-2222\sdk\Jupyter-Op
enVINO-5\demos\speech_recognition_deepspeech_demo\python\asr_utils\audio_features.py:91: FutureWarning: Pass sr=16000, n_fft=512 as keyword args. Fr
om version 0.10 passing these as positional arguments will result in an error
  mel_basis = librosa.filters.mel(
100%|██████████████████████████████████████████████████████████████████████████████| 1/1 [00:00<00:00,  5.99it/s]
[ INFO ] Metrics report:
[ INFO ]        Latency: 225.3 ms

Transcription(s) and confidence score(s):
6.690038681030273       speech
```

speech_recognition_quartznet_demo

Function: QuartzNet speech recognition

Introduction: Read the wav audio file and identify the content of the audio file

Source: wav file



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
wav_file = "data/speech.wav"
model_name = "quartznet-15x5-en"
```

```
model = "model/QuartzNet15x5-En-Base.xml"
```

```
if not os.path.exists(model):
    !mo --input_model model/QuartzNet15x5-En-Base.onnx --input_shape [1,64,128] --output_dir model/
```

```
%run speech_recognition_quartznet_demo.py -i $wav_file -m $model -d CPU
```

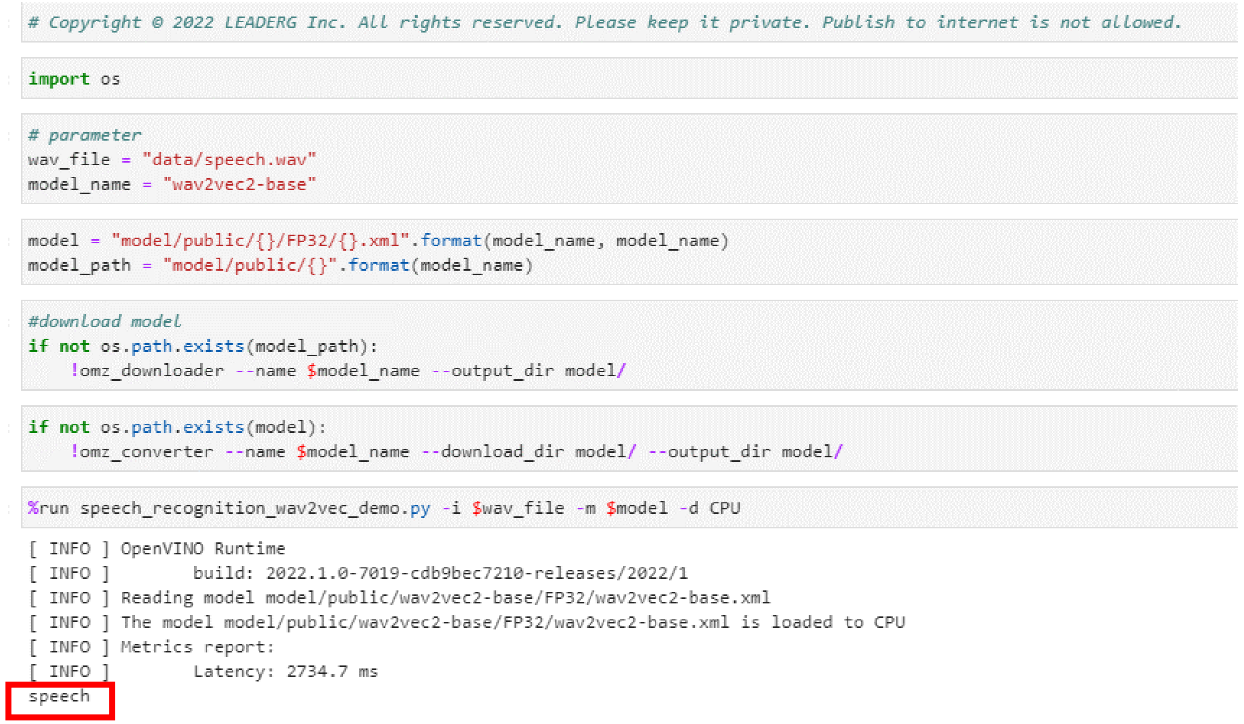
```
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
[ INFO ] Reading model model/QuartzNet15x5-En-Base.xml
D:\App4AI-2222\sdk\Jupyter-OpenVINO-5\demos\speech_recognition_quartznet_demo\python\speech_recognition_quartznet_demo.py:84: FutureWarning: Pass sr
=16000, n_fft=512 as keyword args. From version 0.10 passing these as positional arguments will result in an error
  mel_basis = librosa.filters.mel(sampling_rate, 512, n_mels=64, fmin=0.0, fmax=8000.0, norm='slaney', htk=False)
[ INFO ] The model model/QuartzNet15x5-En-Base.xml is loaded to CPU
[ INFO ] Metrics report:
[ INFO ]        Latency: 972.2 ms
speech
```

speech_recognition_wav2vec_demo

Function: Wav2Vec speech recognition

Introduction: Read the wav audio file and identify the content of the audio file

Source file (source): wav audio file



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
wav_file = "data/speech.wav"
model_name = "wav2vec2-base"
```

```
model  = "model/public/{}/FP32/{}.xml".format(model_name, model_name)
model_path = "model/public/{}".format(model_name)
```

```
#download model
if not os.path.exists(model_path):
    !omz_downloader --name $model_name --output_dir model/
```

```
if not os.path.exists(model):
    !omz_converter --name $model_name --download_dir model/ --output_dir model/
```

```
%run speech_recognition_wav2vec_demo.py -i $wav_file -m $model -d CPU
```

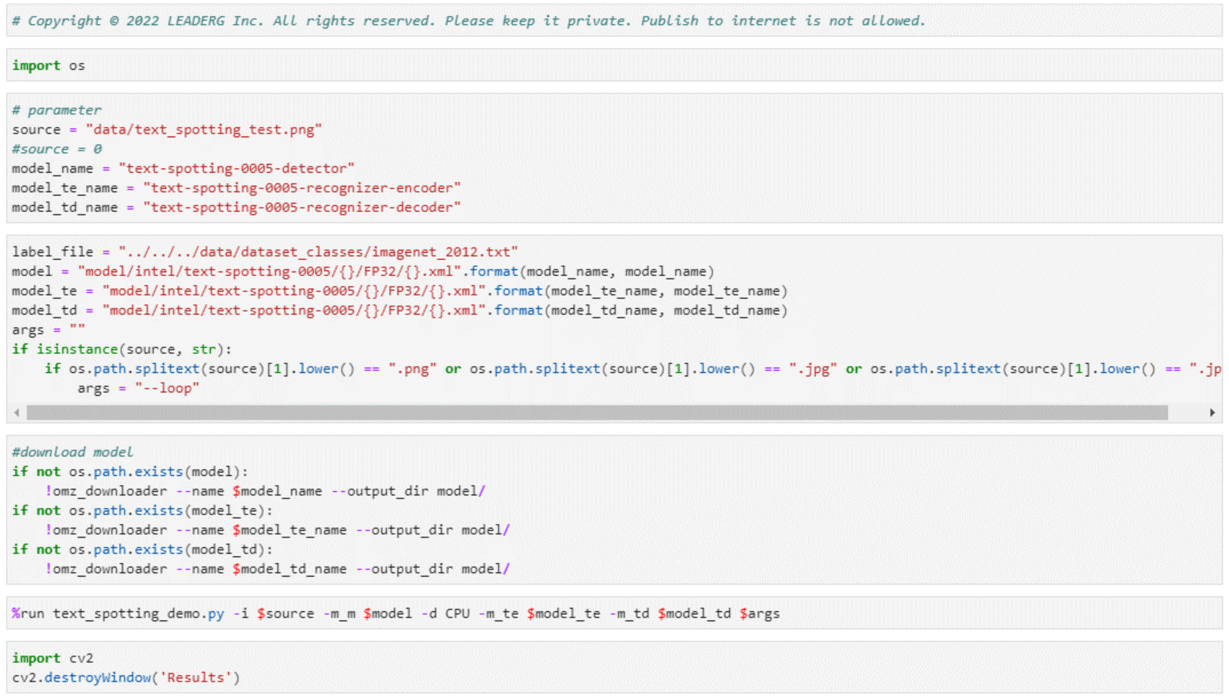
```
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
[ INFO ] Reading model model/public/wav2vec2-base/FP32/wav2vec2-base.xml
[ INFO ] The model model/public/wav2vec2-base/FP32/wav2vec2-base.xml is loaded to CPU
[ INFO ] Metrics report:
[ INFO ]        Latency: 2734.7 ms
speech
```

text_spotting_demo

Function: Text Recognition

Introduction: Recognize text in images

Source: video, image, webcam



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

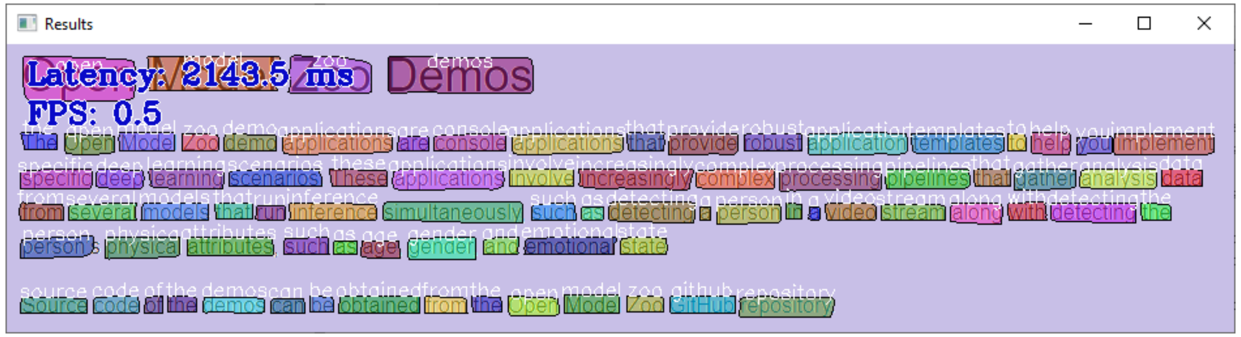
```
# parameter
source = "data/text_spotting_test.png"
#source = 0
model_name = "text-spotting-0005-detector"
model_te_name = "text-spotting-0005-recognizer-encoder"
model_td_name = "text-spotting-0005-recognizer-decoder"
```

```
label_file = "../../../data/dataset_classes/imagenet_2012.txt"
model = "model/intel/text-spotting-0005/{}/FP32/{}.xml".format(model_name, model_name)
model_te = "model/intel/text-spotting-0005/{}/FP32/{}.xml".format(model_te_name, model_te_name)
model_td = "model/intel/text-spotting-0005/{}/FP32/{}.xml".format(model_td_name, model_td_name)
args = ""
if isinstance(source, str):
    if os.path.splitext(source)[1].lower() == ".png" or os.path.splitext(source)[1].lower() == ".jpg" or os.path.splitext(source)[1].lower() == ".jp
        args = "--loop"
```

```
#download model
if not os.path.exists(model):
    !omz_downloader --name $model_name --output_dir model/
if not os.path.exists(model_te):
    !omz_downloader --name $model_te_name --output_dir model/
if not os.path.exists(model_td):
    !omz_downloader --name $model_td_name --output_dir model/
```

```
%run text_spotting_demo.py -i $source -m_m $model -d CPU -m_te $model_te -m_td $model_td $args
```

```
import cv2
cv2.destroyWindow('Results')
```

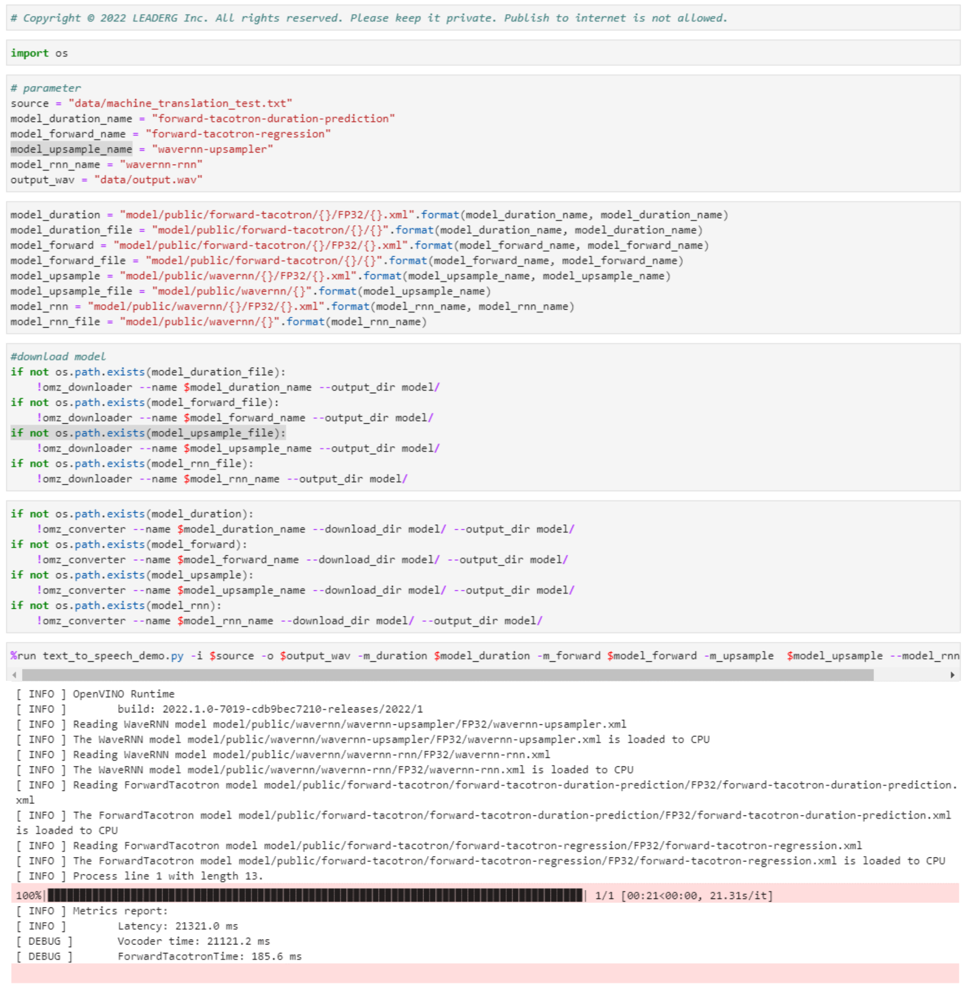

Results
Latency: 2143.5 ms
FPS: 0.5
open model zoo demos
Demos
the open model zoo demo applications are console applications that provide robust application templates to help you implement
specific deep learning scenarios these applications involve increasingly complex processing pipelines that gather analysis data
from several models that run inference such as detecting a person in a video stream along with detecting the
person physical attributes such as age gender and emotional state
source code of the demos can be obtained from the open model zoo github repository

text_to_speech_demo

Function: Text-to-speech

Introduction: Read in txt file, convert English words into voice and output to data/output.wav

Source: text file



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source = "data/machine_translation_test.txt"
model_duration_name = "forward-tacotron-duration-prediction"
model_forward_name = "forward-tacotron-regression"
model_upsample_name = "wavernn-upsampler"
model_rnn_name = "wavernn-rnn"
output_wav = "data/output.wav"
```

```
model_duration = "model/public/forward-tacotron/{}/FP32/{}.xml".format(model_duration_name, model_duration_name)
model_duration_file = "model/public/forward-tacotron/{}/{}".format(model_duration_name, model_duration_name)
model_forward = "model/public/forward-tacotron/{}/FP32/{}.xml".format(model_forward_name, model_forward_name)
model_forward_file = "model/public/forward-tacotron/{}/{}".format(model_forward_name, model_forward_name)
model_upsample = "model/public/wavernn/{}/FP32/{}.xml".format(model_upsample_name, model_upsample_name)
model_upsample_file = "model/public/wavernn/{}".format(model_upsample_name)
model_rnn = "model/public/wavernn/{}/FP32/{}.xml".format(model_rnn_name, model_rnn_name)
model_rnn_file = "model/public/wavernn/{}".format(model_rnn_name)
```

```
#download model
if not os.path.exists(model_duration_file):
    !omz_downloader --name $model_duration_name --output_dir model/
if not os.path.exists(model_forward_file):
    !omz_downloader --name $model_forward_name --output_dir model/
if not os.path.exists(model_upsample_file):
    !omz_downloader --name $model_upsample_name --output_dir model/
if not os.path.exists(model_rnn_file):
    !omz_downloader --name $model_rnn_name --output_dir model/
```

```
if not os.path.exists(model_duration):
    !omz_converter --name $model_duration_name --download_dir model/ --output_dir model/
if not os.path.exists(model_forward):
    !omz_converter --name $model_forward_name --download_dir model/ --output_dir model/
if not os.path.exists(model_upsample):
    !omz_converter --name $model_upsample_name --download_dir model/ --output_dir model/
if not os.path.exists(model_rnn):
    !omz_converter --name $model_rnn_name --download_dir model/ --output_dir model/
```

```
%run text_to_speech_demo.py -i $source -o $output_wav -m_duration $model_duration -m_forward $model_forward -m_upsample  $model_upsample --model_rnn
```

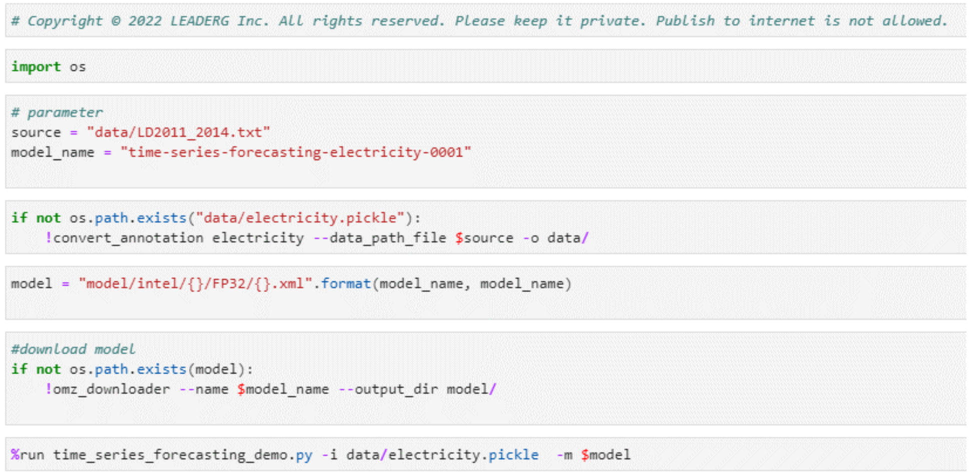
```
[ INFO ] OpenVINO Runtime
[ INFO ]        build: 2022.1.0-7019-cdb9bec7210-releases/2022/1
[ INFO ] Reading WaveRNN model model/public/wavernn/wavernn-upsampler/FP32/wavernn-upsampler.xml
[ INFO ] The WaveRNN model model/public/wavernn/wavernn-upsampler/FP32/wavernn-upsampler.xml is loaded to CPU
[ INFO ] Reading WaveRNN model model/public/wavernn/wavernn-rnn/FP32/wavernn-rnn.xml
[ INFO ] The WaveRNN model model/public/wavernn/wavernn-rnn/FP32/wavernn-rnn.xml is loaded to CPU
[ INFO ] Reading ForwardTacotron model model/public/forward-tacotron/forward-tacotron-duration-prediction/FP32/forward-tacotron-duration-prediction.
xml
[ INFO ] The ForwardTacotron model model/public/forward-tacotron/forward-tacotron-duration-prediction/FP32/forward-tacotron-duration-prediction.xml
is loaded to CPU
[ INFO ] Reading ForwardTacotron model model/public/forward-tacotron/forward-tacotron-regression/FP32/forward-tacotron-regression.xml
[ INFO ] The ForwardTacotron model model/public/forward-tacotron/forward-tacotron-regression/FP32/forward-tacotron-regression.xml is loaded to CPU
[ INFO ] Process line 1 with length 13.
100%|██████████████████████████████████████████████████████████████████████████| 1/1 [00:21<00:00, 21.31s/it]
[ INFO ] Metrics report:
[ INFO ]        Latency: 21321.0 ms
[ DEBUG ]       Vocoder time: 21121.2 ms
[ DEBUG ]       ForwardTacotronTime: 185.6 ms
```

time_series_forecasting_demo

Function: Time Series Forecasting

Introduction: Read in the power text file and output the time series forecast graph

Source: text file



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

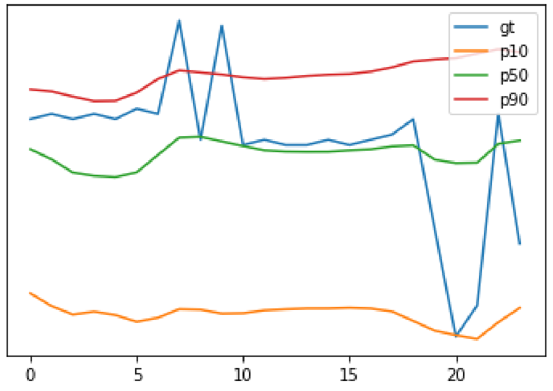
```
# parameter
source = "data/LD2011_2014.txt"
model_name = "time-series-forecasting-electricity-0001"
```

```
if not os.path.exists("data/electricity.pickle"):
    !convert_annotation electricity --data_path_file $source -o data/
```

```
model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
```

```
#download model
if not os.path.exists(model):
    !omz_downloader --name $model_name --output_dir model/
```

```
%run time_series_forecasting_demo.py -i data/electricity.pickle  -m $model
```



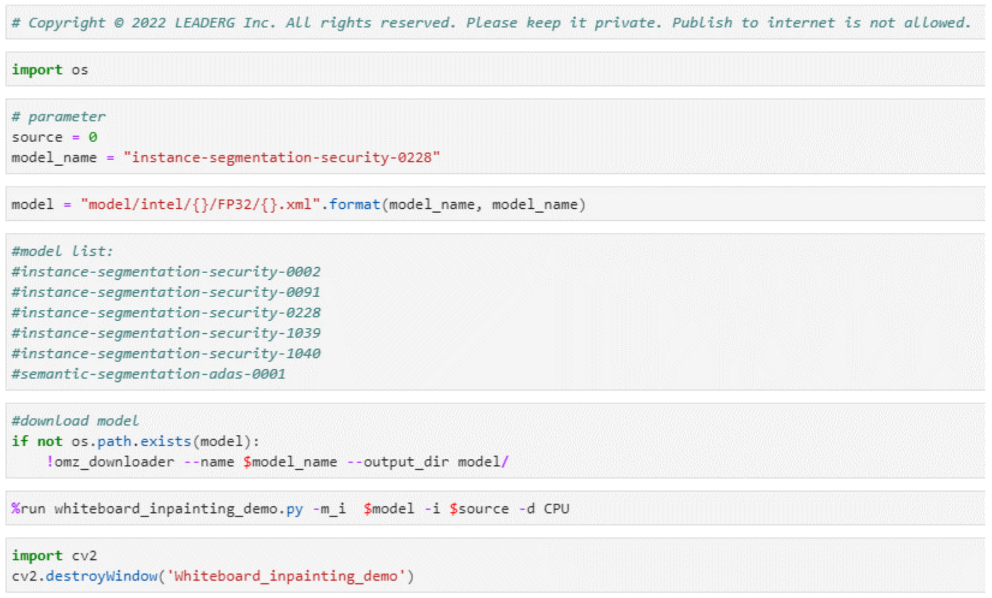
Inference source:
https://archive.ics.uci.edu/ml/machine-learning-databases/00321/LD2011_2014.txt.zip

whiteboard_inpainting_demo

Function: whiteboard inpainting

Introduction: Read the content of the whiteboard in the video and hide the person on a video

Source: video, webcam



```
# Copyright © 2022 LEADERG Inc. All rights reserved. Please keep it private. Publish to internet is not allowed.
```

```
import os
```

```
# parameter
source = 0
model_name = "instance-segmentation-security-0228"
```

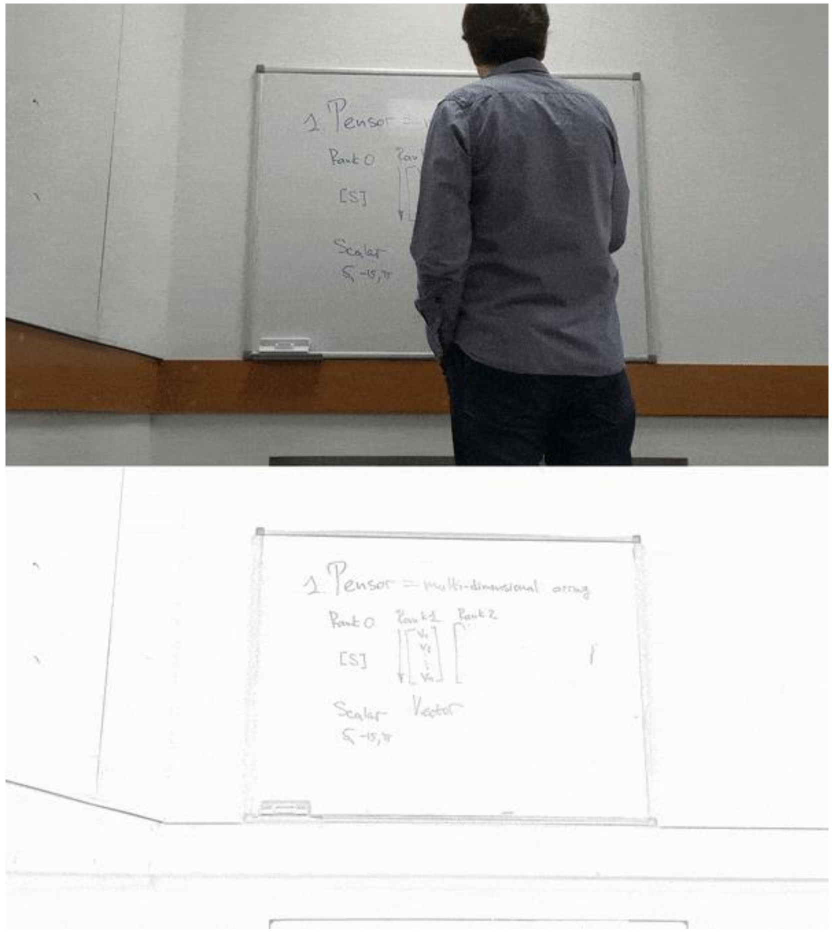
```
model = "model/intel/{}/FP32/{}.xml".format(model_name, model_name)
```

```
#model list:
#instance-segmentation-security-0002
#instance-segmentation-security-0091
#instance-segmentation-security-0228
#instance-segmentation-security-1039
#instance-segmentation-security-1040
#semantic-segmentation-adas-0001
```

```
#download model
if not os.path.exists(model):
    !omz_downloader --name $model_name --output_dir model/
```

```
%run whiteboard_inpainting_demo.py -m_i  $model -i $source -d CPU
```

```
import cv2
cv2.destroyWindow('Whiteboard_inpainting_demo')
```



Reference

- Please refer to the readme.txt in the SDK folder.
- LEADERG AppForAI: https://www.leaderg.com/appforai-windows
- Copyright © LEADERG INC. All rights reserved.