

Image-YOLOv5-Pose-PyTorch- GPL-Jupyter

Use YOLOv5 Pose to detect the human position, key point (eyes, ears, nose, shoulders, elbows, wrists, hips, knees, ankles), and achieve fast human pose detection.

Version 20230223

Applications

- YOLO Pose can be applied to medical image analysis, biological image analysis, advanced driver assistance systems, autonomous vehicle analysis, factory security systems, rehabilitation systems, etc.



How to use

The main process is:

Prepare files for training -> Training -> Inference

Name	Last Modified
data	24 minutes ago
src	24 minutes ago
1_coco_json_convert_yolo_format.ipynb	2 days ago
2_prepare_txt.ipynb	2 days ago
3_train.ipynb	2 days ago
4_inference_image.ipynb	2 days ago
5_inference_folder.ipynb	2 days ago
6_inference_webcam.ipynb	2 days ago
readme.txt	2 days ago
version.txt	a day ago

Dataset format

In the Data folder:

model: the folder where the training model is stored

train: divided into three folders: images, annotations, labels

val: divided into three folders: images, annotations, labels

The images, annotations, labels folders are:

images: coco2017 dataset images of people

annotations: coco2017 dataset contains person keypoint annotation file

labels: 1_coco_json_convert_yolo_format.ipynb The converted yolo format annotation file

If you need to annotate images, please find an annotation software that supports the coco keypoints format. You can also refer to the Annotation URL of readme.txt.

1_coco_json_convert_yolo_format.ipynb

Convert from coco annotation file with keypoints format to yolo format.

Before running, please make sure that the paths of train_json_file and val_json_file in #parameters are correct.



```
1_coco_json_convert_yolo_fo + Python 3 (ipykerne
+ × 🗑️ 📄 ▶️ ⏪ ⏩ Code
b_y = json_value['annotations'][i]['bbox'][1]
b_w = json_value['annotations'][i]['bbox'][0] + json_value['annotations'][i]['bbox'][2]
b_h = json_value['annotations'][i]['bbox'][1] + json_value['annotations'][i]['bbox'][3]

b = (float(b_x), float(b_w), float(b_y), float(b_h))

bb = convert((images_dict[json_value['annotations'][i]['image_id']]['width'], images_dict[json_value['annotations'][i]['image_id']]['height']), b)

convert_value = str(category_id)
if isOnlyLabel:
    convert_value = "0"

# 4 + 17
if json_value['annotations'][i]['num_keypoints'] == 0:
    continue
for j in range(21):
    if j < 4:
        convert_value += " %.6f" % (bb[j])
    else:
        keypoints = json_value['annotations'][i]['keypoints']
        keypoint_x = float(keypoints[(j-4) * 3] / images_dict[json_value['annotations'][i]['image_id']]['width'])
        keypoint_y = float(keypoints[(j-4) * 3 + 1] / images_dict[json_value['annotations'][i]['image_id']]['height'])
        convert_value += " %.6f %.6f %.6f" % (keypoint_x, keypoint_y, float(keypoints[(j-4) * 3 + 2]))

filename = images_dict[json_value['annotations'][i]['image_id']]['file_name']
with open(os.path.join(save_label_path, os.path.splitext(filename)[0] + '.txt'), 'a+', newline='\n') as f:
    f.write(convert_value + "\n")

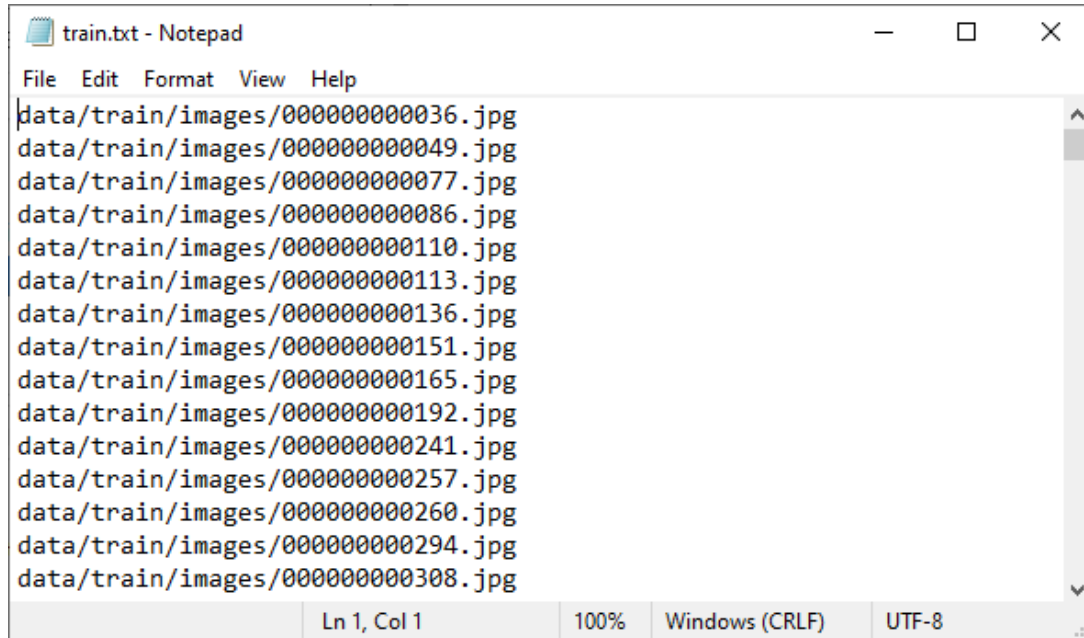
[*]: print('Converting, please wait a few minutes... ')
convert_function(train_json_file, train_save_label_path)

Converting, please wait a few minutes...

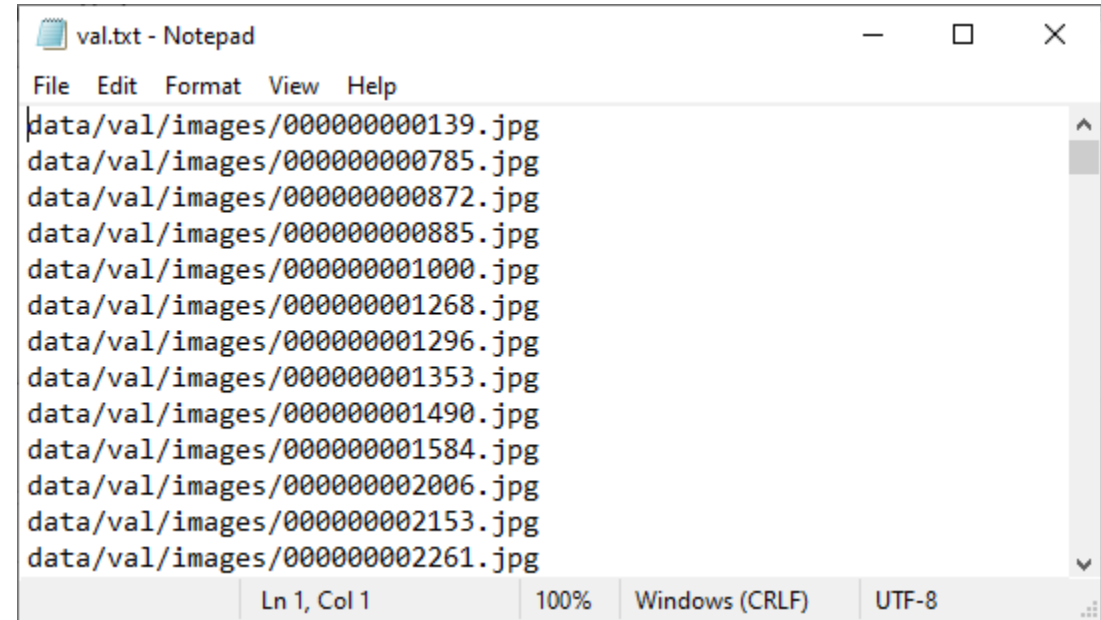
[*]: convert_function(val_json_file, val_save_label_path)
```

2_prepare_txt.ipynb

Generate training and validation image path files train.txt and val.txt.



```
train.txt - Notepad
File Edit Format View Help
data/train/images/00000000036.jpg
data/train/images/00000000049.jpg
data/train/images/00000000077.jpg
data/train/images/00000000086.jpg
data/train/images/00000000110.jpg
data/train/images/00000000113.jpg
data/train/images/00000000136.jpg
data/train/images/00000000151.jpg
data/train/images/00000000165.jpg
data/train/images/00000000192.jpg
data/train/images/00000000241.jpg
data/train/images/00000000257.jpg
data/train/images/00000000260.jpg
data/train/images/00000000294.jpg
data/train/images/00000000308.jpg
Ln 1, Col 1 100% Windows (CRLF) UTF-8
```



```
val.txt - Notepad
File Edit Format View Help
data/val/images/00000000139.jpg
data/val/images/000000000785.jpg
data/val/images/000000000872.jpg
data/val/images/000000000885.jpg
data/val/images/000000001000.jpg
data/val/images/000000001268.jpg
data/val/images/000000001296.jpg
data/val/images/000000001353.jpg
data/val/images/000000001490.jpg
data/val/images/000000001584.jpg
data/val/images/000000002006.jpg
data/val/images/000000002153.jpg
data/val/images/000000002261.jpg
Ln 1, Col 1 100% Windows (CRLF) UTF-8
```


3_train.ipynb

Start training.

ipynb parameter:

- `cfg_file` is the yolo yaml file.
- `pretrained_model` is the pretrained model.
- `image_size` is the training image size.
- `epochs` is the number of training epochs.

```
[1]: # parameter
     cfg_file = "data/yolov5m6_kpts_ti_lite.yaml"
     pretrained_model = "data/model/Yolov5m6_pose_960_ti_lite_pretrained_weight.pt"
     batch_size = 16
     image_size = 960
     hyp_file = "data/Yolov5m6_pose_960_ti_lite_hyp.yaml"
     epochs = 100
     save_model_path = "data/model"

[ ]: %run src/train.py --data data/coco_kpts.yaml --cfg $cfg_file --weights $pretrained_model --project $save_model_path --batch-size $batch_size --img $image_size
```

YOLOv5 2022-6-6 torch 1.12.0+cu113 CUDA:0 (NVIDIA TITAN RTX, 24575.6875MB)

Namespace(weights='data/model/Yolov5m6_pose_960_ti_lite_pretrained_weight.pt', cfg='data/yolov5m6_kpts_ti_lite.yaml', data='data/coco_kpts.yaml', hyp='data/Yolov5m6_pose_960_ti_lite_hyp.yaml', epochs=100, batch_size=16, img_size=[960, 960], rect=False, resume=False, nosave=False, notest=False, noautoanchor=False, evolve=False, bucket='', cache_images=False, image_weights=False, device='', multi_scale=False, single_cls=False, adam=False, sync_bn=False, local_rank=-1, workers=8, project='data/model', entity=None, name='exp', exist_ok=True, quad=False, linear_lr=False, label_smoothing=0.0, upload_dataset=False, bbox_interval=-1, save_period=-1, artifact_alias='latest', kpt_label=True, log_folder='data/logs', world_size=1, global_rank=-1, save_dir='data\\model', total_batch_size=16)

tensorboard: Start with 'tensorboard --logdir data/model', view at <http://localhost:6006/>

github: skipping check (not a git repository)

requirements: requirements.txt not found, check failed.

hyperparameters: lr0=0.0032, lrf=0.2, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bias_lr=0.1, box=0.05, kpt=0.1, cls=0.5, cls_pw=1.0, obj=1.0, obj_pw=1.0, iou_t=0.2, anchor_t=4.0, fl_gamma=0.0, hsv_h=0.015, hsv_s=0.7, hsv_v=0.4, degrees=0.0, translate=0.1, scale=0.5, shear=0.0, perspective=0.0, flipud=0.0,fliplr=0.5, mosaic=1.0, mixup=0.0

wandb: Install Weights & Biases for YOLOv5 logging with 'pip install wandb' (recommended)

	from	n	params	module	arguments
0	-1	1	5628	models.common.ConvFocus	[3, 48, 3]
1	-1	1	41664	models.common.Conv	[48, 96, 3, 2]
2	-1	1	65280	models.common.C3	[96, 96, 2]
3	-1	1	166272	models.common.Conv	[96, 192, 3, 2]
4	-1	1	629760	models.common.C3	[192, 192, 6]
5	-1	1	664320	models.common.Conv	[192, 384, 3, 2]
6	-1	1	2512896	models.common.C3	[384, 384, 6]
7	-1	1	1991808	models.common.Conv	[384, 576, 3, 2]
8	-1	1	2327040	models.common.C3	[576, 576, 2]
9	-1	1	3982848	models.common.Conv	[576, 768, 3, 2]
10	-1	1	1476864	models.common.SPP	[768, 768, [3, 5, 7]]
11	-1	1	4134912	models.common.C3	[768, 768, 2, False]
12	-1	1	443520	models.common.Conv	[768, 576, 1, 1]
13	-1	1	0	torch.nn.modules.uopsampling.Upsample	[None, 2, 'nearest']

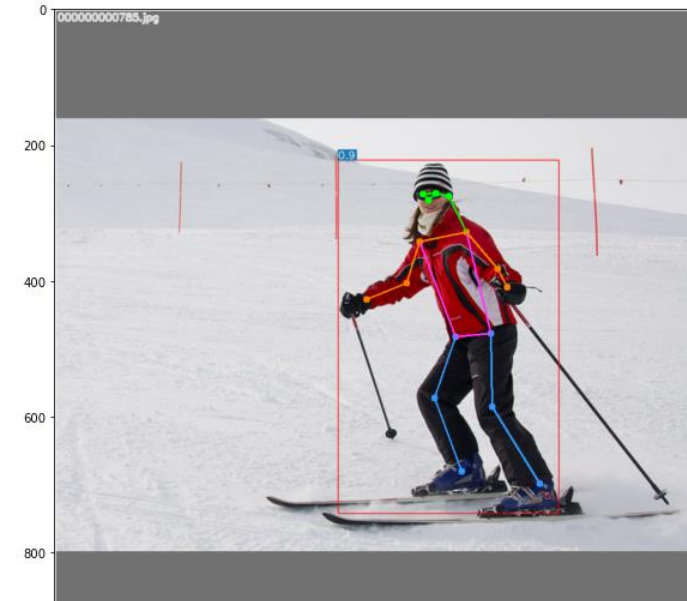
4_inference_image.ipynb

Infer a single image.

ipynb parameter:

- inference_image is the inference image path.
- image_size is the inferred image size and needs to be a multiple of 64.
- inference_model is the inference model path.
- save_result_path is the path to the folder where the inference results are stored.

```
%run src/test.py --data data/coco_kpts.yaml --task test --img-size $image_size --conf 0.001 --iou 0.65 --weights $inference_model --kpt-label --project $save_re
4
YOLOv5 2022-6-6 torch 1.12.0+cu113 CUDA:0 (NVIDIA TITAN RTX, 24575.6875MB)
Namespace(weights=['data/model/best.pt'], data='data/coco_kpts.yaml', batch_size=32, img_size=960, conf_thres=0.001, iou_thres=0.65, task='test', device='', sin
gle_cls=True, augment=False, verbose=False, save_txt=False, save_txt_tidl=False, tidl_load=False, dump_img=False, save_hybrid=False, save_single_image=True, sav
e_conf=False, save_json=False, save_json_kpt=True, project='data/inference_result', name='exp', exist_ok=True, kpt_label=True, flip_test=False)
Fusing layers...
C:\Users\ai\Desktop\App4AI-2225\gpu\python\lib\site-packages\torch\functional.py:478: UserWarning: torch.meshgrid: in an upcoming release, it will be required t
o pass the indexing argument. (Triggered internally at C:\actions-runner\work\pytorch\pytorch\builder\windows\pytorch\aten\src\ATen\native\TensorShape.cpp:289
5.)
return _VF.meshgrid(tensors, **kwargs) # type: ignore[attr-defined]
Model Summary: 407 layers, 35728332 parameters, 0 gradients, 52.0 GFLOPS
test: Scanning 'data/test' images and labels... 0 found, 1 missing, 0 empty, 0 corrupted: 100%|█ 1/1 [00:00<00:00, 33.
test: New cache created: data\test.cache
test: WARNING: No labels found in data\test.cache. See https://github.com/ultralytics/yolov5/wiki/Train-Custom-Data
Class      Images  Labels  P      R      mAP@.5  mAP@.5:.95:  0% | 0/1 [00:00<?, ?it
data\val\images\000000000785.jpg
```



5_inference_folder.ipynb

Infer all images in the folder.

ipynb parameter:

- inference_folder is the inference image folder path
- image_size is the inferred image size and needs to be a multiple of 64.
- inference_model is the inference model path.
- save_result_path is the path to the folder where the inference results are stored.

```
[*]: %run src/test.py --data data/coco_kpts.yaml --task test --img-size $image_size --conf 0.001 --iou 0.65 --weights $inference_model --kpt-label --project $save_re
```

```
YOLOv5 2022-6-6 torch 1.12.0+cu113 CUDA:0 (NVIDIA TITAN RTX, 24575.6875MB)
```

```
Namespace(weights=['data/model/best.pt'], data='data/coco_kpts.yaml', batch_size=32, img_size=960, conf_thres=0.001, iou_thres=0.65, task='test', device='', single_cls=True, augment=False, verbose=False, save_txt=False, save_txt_tidl=False, tidl_load=False, dump_img=False, save_hybrid=False, save_single_image=True, save_conf=False, save_json=False, save_json_kpt=True, project='data/inference_result', name='exp', exist_ok=True, kpt_label=True, flip_test=False)
```

```
Fusing layers...
```

```
C:\Users\ai\Desktop\App4AI-2225\gpu\python\lib\site-packages\torch\functional.py:478: UserWarning: torch.meshgrid: in an upcoming release, it will be required to pass the indexing argument. (Triggered internally at C:\actions-runner\work\pytorch\pytorch\builder\windows\pytorch\aten\src\ATen\native\TensorShape.cpp:2895.)
```

```
return _VF.meshgrid(tensors, **kwargs) # type: ignore[attr-defined]
```

```
Model Summary: 407 layers, 35728332 parameters, 0 gradients, 52.0 GFLOPS
```

```
test: Scanning 'data\test' images and labels... 0 found, 2346 missing, 0 empty, 0 corrupted: 100%|█ 2346/2346 [00:00<0
```

```
test: WARNING: No labels found in data\test.cache. See https://github.com/ultralytics/yolov5/wiki/Train-Custom-Data
```

```
test: New cache created: data\test.cache
```

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95:	0%	0/74	[00:00<?, ?i
-------	--------	--------	---	---	--------	-------------	----	------	--------------

```
data\val\images\000000000139.jpg
```

6_inference_webcam.ipynb

Infer the image of the webcam. Press “q” on the display to turn the webcam off.

ipynb parameter:

- `image_size` is the inferred image size and needs to be a multiple of 64.
- `inference_model` is the inference model path.

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.