PyTorch-YOLOv3

YOLOv3最小的PyTorch实现，支持培训、推理和评估。

安装

复制和安装需求

```
$ git clone https://github.com/eriklindernoren/PyTorch-YOLOv3
$ cd PyTorch-YOLOv3/
$ sudo pip3 install -r requirements.txt
```

下载预训练权重

```
$ cd weights/
$ bash download_weights.sh
```

下载COCO

```
$ cd data/
$ bash get_coco_dataset.sh
```

测试

对模型进行COCO试验评估。

```
$ python3 test.py --weights_path weights/yolov3.weights
```

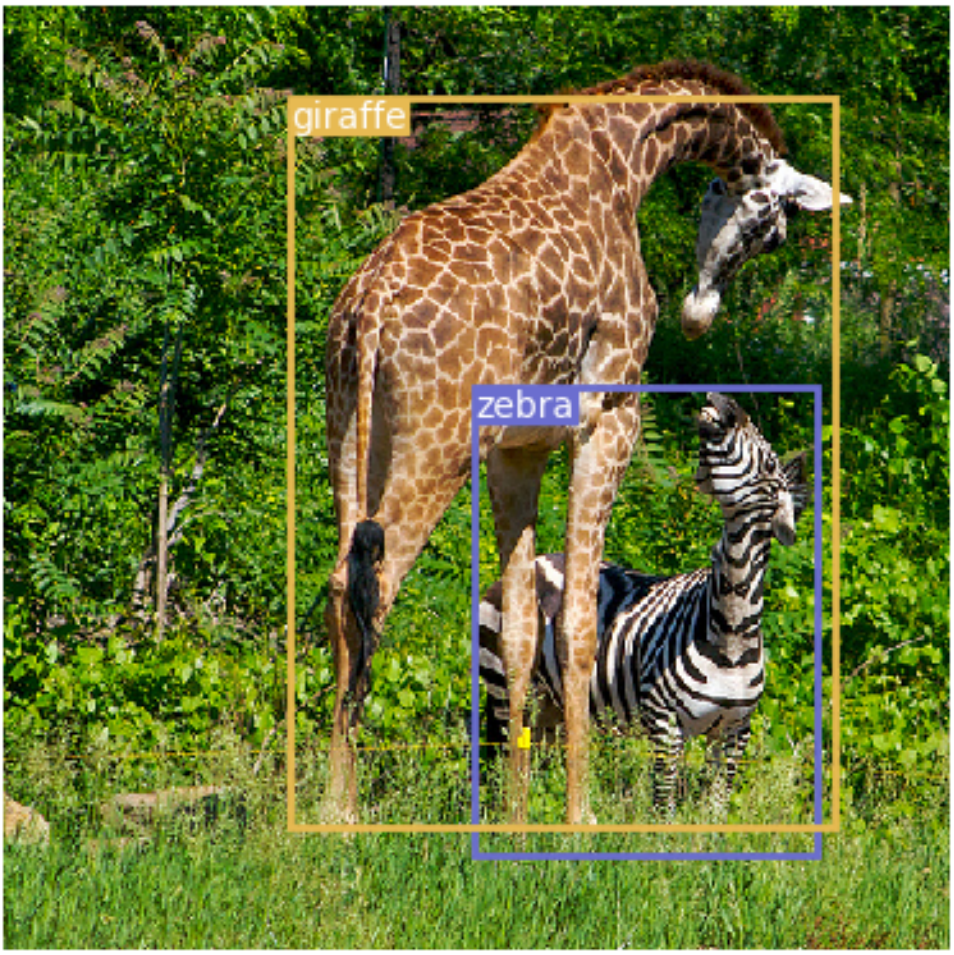
Model	mAP (min. 50 IoU)
YOLOv3 608 (paper)	57.9
YOLOv3 608 (this impl.)	57.3
YOLOv3 416 (paper)	55.3
YOLOv3 416 (this impl.)	55.5

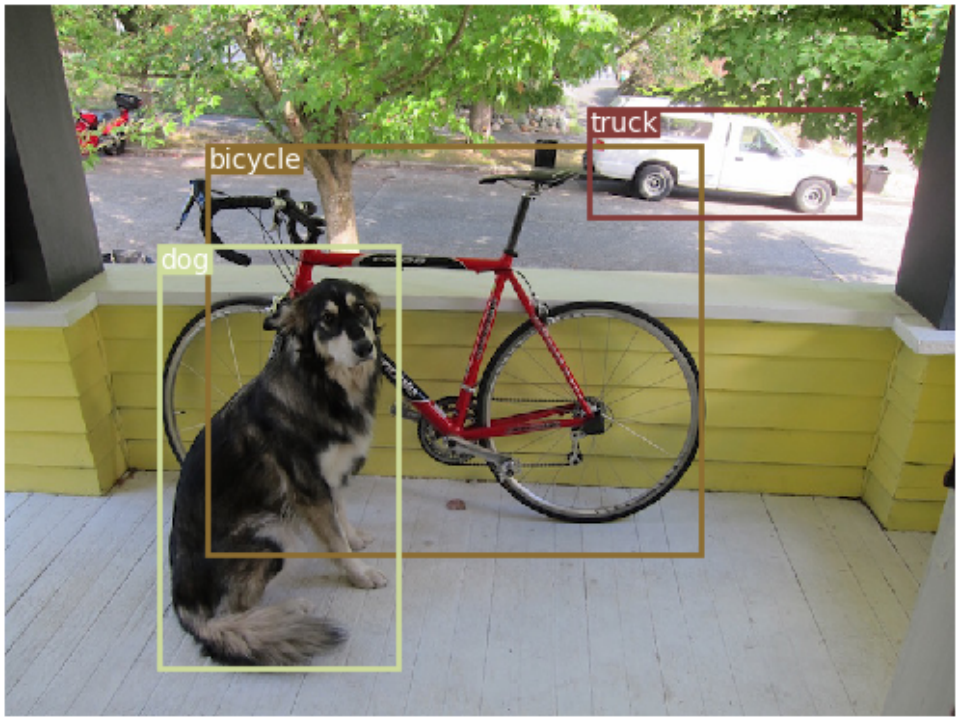
推理

使用预先训练好的权重对图像进行预测。
下表显示了使用缩放到256x256的作为输入图像的推断时间。
ResNet主干的测量取自YOLOv3的论文。
标记的Darknet-53测量显示了在我的1080ti卡上实现此实现的推理时间。

Backbone	GPU	FPS
ResNet-101	Titan X	53
ResNet-152	Titan X	37
Darknet-53 (paper)	Titan X	76
Darknet-53 (this impl.)	1080ti	74

```
$ python3 detect.py --image_folder data/samples/
```




truck
bicycle
dog


traffic light
truck
car
car
car
car
car
car


person
person
person

训练

```
$ train.py [-h] [--epochs EPOCHS] [--batch_size BATCH_SIZE]
               [--gradient_accumulations GRADIENT_ACCUMULATIONS]
               [--model_def MODEL_DEF] [--data_config DATA_CONFIG]
               [--pretrained_weights PRETRAINED_WEIGHTS] [--n_cpu N_CPU]
               [--img_size IMG_SIZE]
               [--checkpoint_interval CHECKPOINT_INTERVAL]
               [--evaluation_interval EVALUATION_INTERVAL]
               [--compute_map COMPUTE_MAP]
               [--multiscale_training MULTISCALE_TRAINING]
```

例子(COCO)

使用在ImageNet上预先训练过的Darknet-53后端对COCO进行培训:

```
$ python3 train.py --data_config config/coco.data  --pretrained_weights
weights/darknet53.conv.74
```

训练日志

```
---- [Epoch 7/100, Batch 7300/14658] ----
+------------+--------------+--------------+--------------+
| Metrics    | YOLO Layer 0 | YOLO Layer 1 | YOLO Layer 2 |
+------------+--------------+--------------+--------------+
| grid_size  | 16           | 32           | 64           |
| loss       | 1.554926     | 1.446884     | 1.427585     |
| x          | 0.028157     | 0.044483     | 0.051159     |
| y          | 0.040524     | 0.035687     | 0.046307     |
| w          | 0.078980     | 0.066310     | 0.027984     |
| h          | 0.133414     | 0.094540     | 0.037121     |
| conf       | 1.234448     | 1.165665     | 1.223495     |
| cls        | 0.039402     | 0.040198     | 0.041520     |
| cls_acc    | 44.44%       | 43.59%       | 32.50%       |
| recall50   | 0.361111     | 0.384615     | 0.300000     |
| recall75   | 0.222222     | 0.282051     | 0.300000     |
| precision  | 0.520000     | 0.300000     | 0.070175     |
| conf_obj   | 0.599058     | 0.622685     | 0.651472     |
| conf_noobj | 0.003778     | 0.004039     | 0.004044     |
+------------+--------------+--------------+--------------+
Total Loss 4.429395
---- ETA 0:35:48.821929
```

Tensorboard

用Tensorboard追踪训练进度:

- 初始化训练
- 运行下面命令
- Go to http://localhost:6006/

```
$ tensorboard --logdir='logs' --port=6006
```

训练自定义数据集

自定义模型

运行下面的命令来创建一个自定义模型定义，用数据集中的类数量替换`< number -classes>`。

```
$ cd config/                                # Navigate to config dir
$ bash create_custom_model.sh <num-classes> # Will create custom model 'yolov3-
custom.cfg'
```

类别

在`data/custom/classes.names`中添加类名。这个文件每个类名应该有一行。

图片文件夹

将数据集的图像移动到`data/custom/images/`。

标签文件夹

将注释移动到`data/custom/labels/`。`dataloader`期望对应于图像`data/custom/images/train.jpg`的注释文件具有路径`data/custom/labels/train.txt`。注释文件中的每一行应该使用`label_idx x_center y_center width height`语法定义一个边界框。坐标应该缩放为`[0,1]`，`label_idx`应该是零索引，并对应于`data/custom/classes.names`中类名的行号。

定义训练集和验证集

在`data/custom/train.txt`和`data/custom/valid.txt`。为图像添加路径，分别用作训练数据和验证数据。

训练

要训练自定义数据集运行:

```
$ python3 train.py --model_def config/yolov3-custom.cfg --data_config
config/custom.data
```

添加`--pretrained_weights weights/darknet53.conv.74`使用在ImageNet上预先训练过的后端进行培训。

引用

YOLOv3:增量的改进

Joseph Redmon, Ali Farhadi

摘要

我们向YOLO提供一些更新!我们做了一些设计上的小改变来让它更好。我们还训练了这个非常棒的新网络。它比上次大了一点，但更准确。不过它还是很快的，别担心。在320×320 YOLOv3运行在22 ms 28.2 mAP，与SSD一样准确，但三倍快。当我们看旧的。5 IOU地图检测度量YOLOv3是相当好的。它在Titan X上实现57.9 AP50在51毫秒，相比之下，RetinaNet的57.5 AP50在198毫秒，类似的性能，但3.8倍快。和往常一样，所有代码都可以在线访问https://pjreddie.com/yolo/。

[Paper] [Project Webpage] [Authors' Implementation]

```
@article{yolov3,
  title={YOLOv3: An Incremental Improvement},
  author={Redmon, Joseph and Farhadi, Ali},
  journal = {arXiv},
  year={2018}
}
```