智能、设计、体验与审美
Intelligence, Design, Experience, & Aesthetics

IDEA LAB
浙江大学-阿里巴巴联合实验室

GAN组进展汇报

玩具图像识别子系统

时间 **2020.7.27**
地点 **钉钉(online)**
汇报人 **实习生-张喆**

阶段工作汇总

工作概述		负责人	完成情况
环境搭建	①配置服务器环境 ②安装深度学习相关依赖 ③yolo网络搭建	张喆	• 连接服务器ssh和ftp服务 • 配置opencv环境并整理文档 • 配置yolo环境进行测试并整理文档
实验	①数据集拍摄 ②数据集扩充 ③数据集标注 ④数据集汇总	张喆	• 拍摄93张数据集 • 使用11种方法进行数据集扩充并整理文档 • 对超过500张图片进行手工标注，其余采用脚本自动标注 • 整理之前留下的数据集，并汇总得到1551张有效图片
模型训练	①yolov3 ②yolov4 ③模型测试	张喆	• 使用yolov3网络训练2000epochs • 使用yolov4网络训练2000epochs • 测试两种网络的mAP和实际效果 • 使用图片和视频对网络进行验收测试

汇报大纲

- 项目计划与目标
- 本周进展
  - 环境搭建
  - 数据集
  - 模型训练
- 项目管理与实践

项目计划与目标

第一周

- 购买玩具
- 构建数据集
- 进行初步模型搭建

第二周

- 模型训练优化
- 多种模型进行比较

第三周

- 场景图谱图

7月22日

本周进展

环境搭建

数据集

模型训练

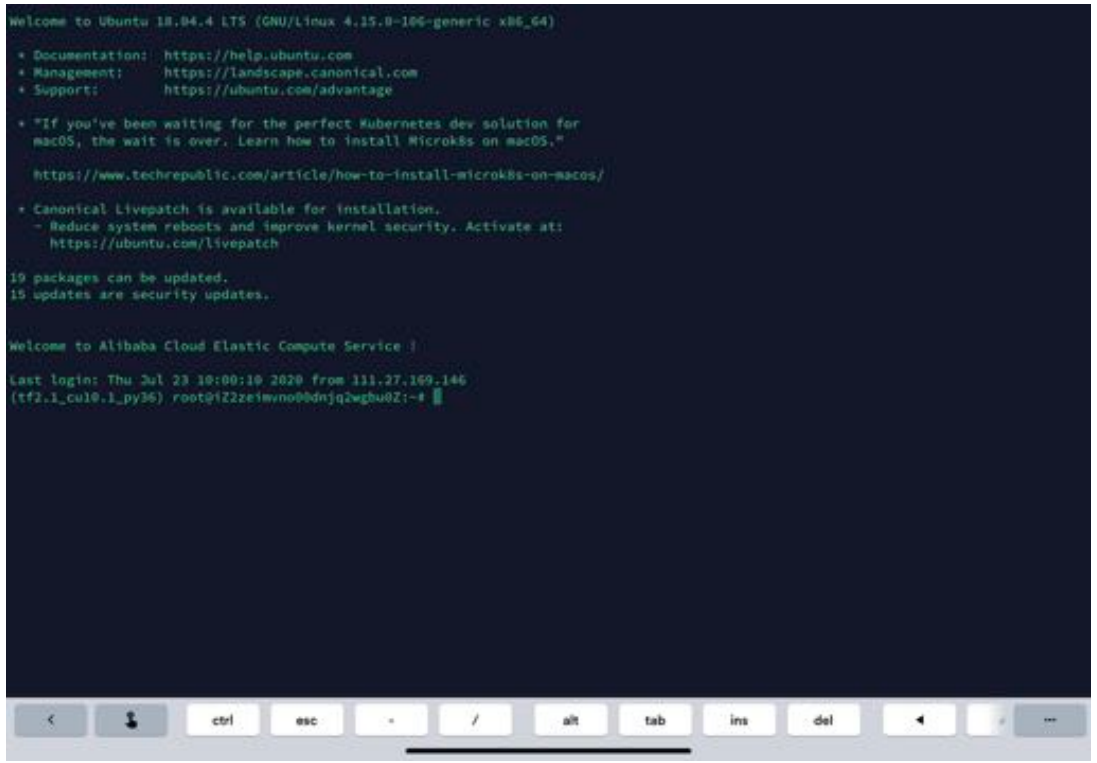
环境搭建

IDEA LAB
浙江大学-阿里巴巴联合实验室

服务器环境

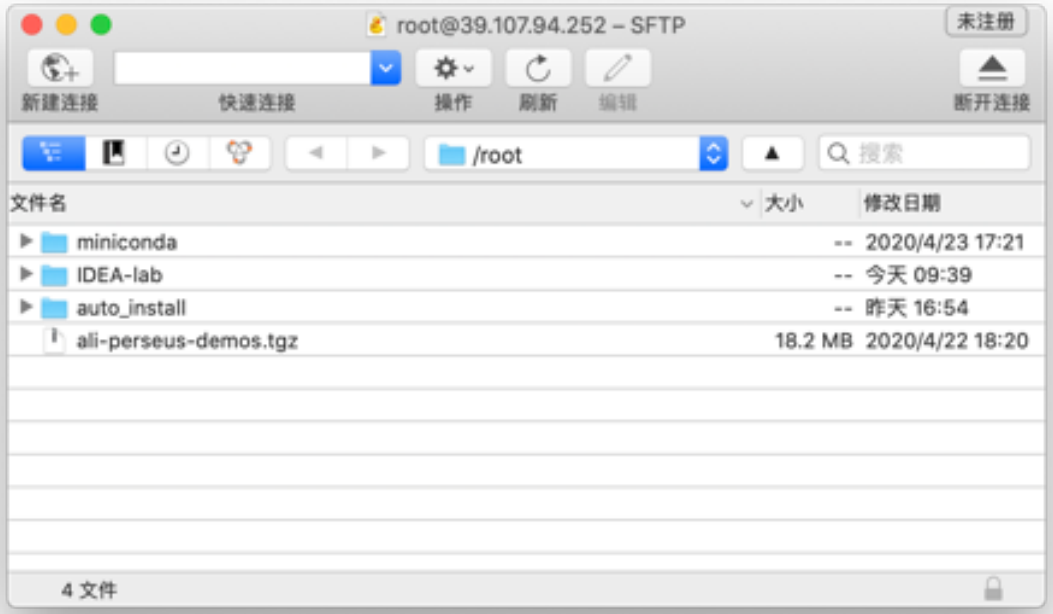
【ssh】

- ✓ macOS使用「终端」app进行连接
- ✓ iPadOS使用「Termius」app进行连接



【ftp】

- ✓ 使用「Cyberduck」app进行连接



环境搭建

IDEA LAB
浙江大学-阿里巴巴联合实验室

opencv

- Ubuntu下配置opencv环境
- 安装准备
  - 安装cmake
  - 安装依赖环境
  - 下载opencv
  - 安装
    - 解压
    - cmake
    - 编译
    - 安装
    - 配置环境
    - 检验

Ubuntu下配置opencv环境

Table of Contents

- **操作系统**：Ubuntu18.04.4
- **版本**：opencv3.2.0

> 【reference】
>
> https://zhuanlan.zhihu.com/p/76737748
>
> https://blog.csdn.net/public669/article/details/99044895
>
> https://blog.csdn.net/weixin_41851439/article/details/88712465

opencv

- Ubuntu下配置opencv环境
- 安装准备
  - 安装cmake
  - 安装依赖环境
  - 下载opencv
  - 安装
    - 解压
    - cmake
    - 编译
    - 安装
    - 配置环境
    - 检验

1. 无法定位软件包libjasper-dev
2. 由于网络问题无法下载 ippicv_linux_20151201.tgz 文件
3. The following variables are used in this project, but they are set to NOTFOUND.
4. cuda9不再支持2.0架构
5. Unsupported gpu architecture 'compute_20'
6. Package opencv was not found in the pkg-config search path.

yolo

Cloning and Building Darknet

```
# clone darknet repo
git clone https://github.com/AlexeyAB/darknet
cd darknet/

# change makefile to h
sed -i 's/OPENCV=0/OPE
sed -i 's/GPU=0/GPU=1/
sed -i 's/CUDNN=0/CUDN

make
```

Download pretrained YOLOv3 weights

YOLOv3 has been trained already on the coco dataset which has 80 classes that it can predict.

```
# get yolov3 pretrai
wget https://pjreddi
```

Run Detections with Darknet and YOLOv3

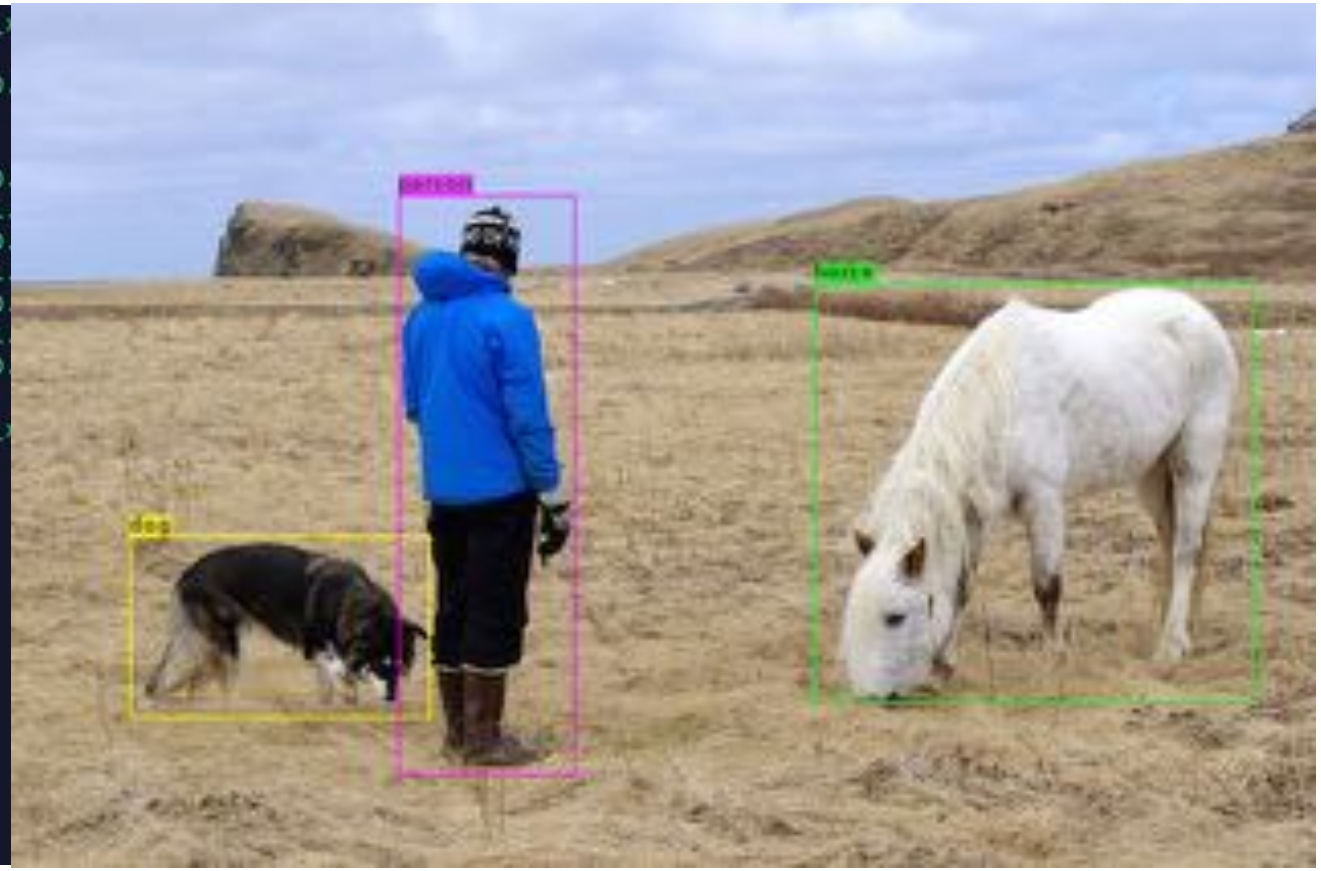
```
# run darknet detection
!./darknet detect cfg/y
```

Yolo command line flags

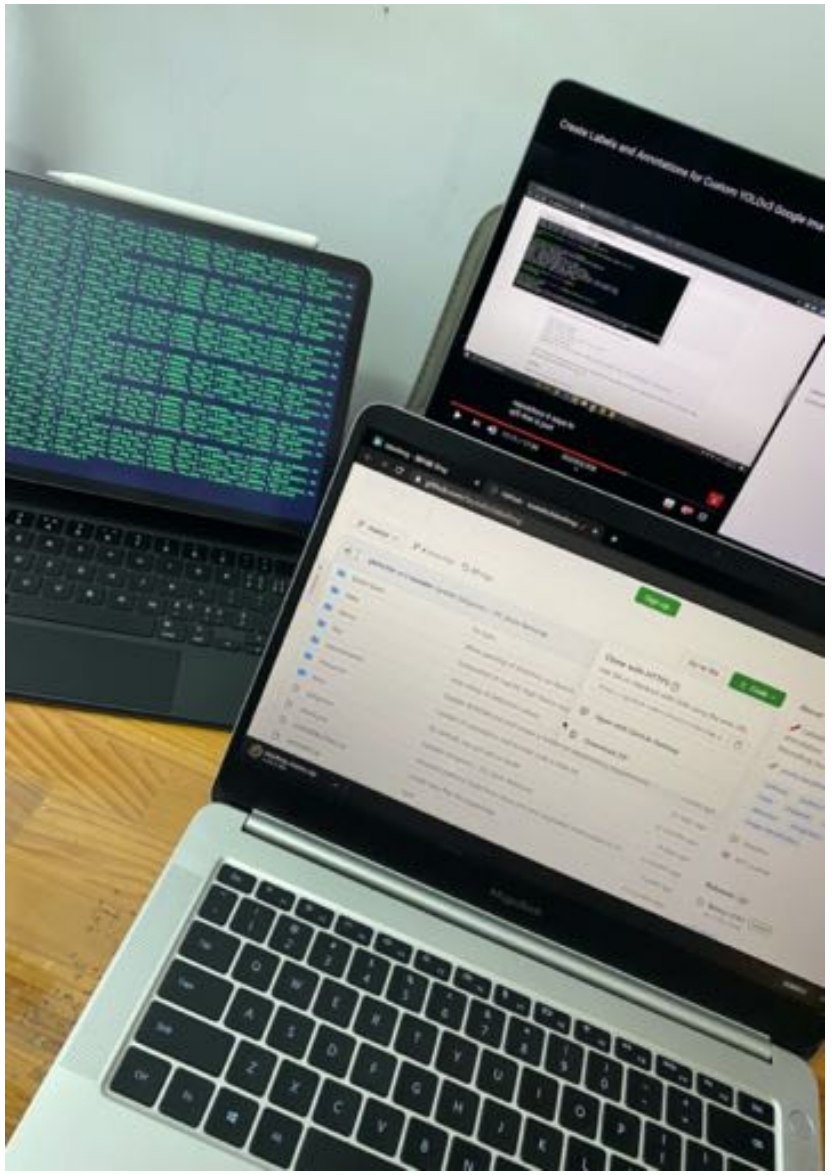
- `-thresh`: add a threshold for confidences on the detections, only detections with a confidence level above the threshold will be returned
- `-dont_show`: not have the image outputted after running darknet
- `-ext_output`: output bounding box coordinates

yolo

```
[yolo] params: iou loss: mse (2), iou_norm: 0.75, cls_norm: 1.00, scale_
  95 route  91                                     ->   26 x  26 x 256
  96 conv    128       1 x 1/ 1     26 x  26 x 256 ->   26 x  26 x 128 0
  97 upsample                 2x    26 x  26 x 128 ->   52 x  52 x 128
  98 route  97 36                                  ->   52 x  52 x 384
  99 conv    128       1 x 1/ 1     52 x  52 x 384 ->   52 x  52 x 128 0
 100 conv    256       3 x 3/ 1     52 x  52 x 128 ->   52 x  52 x 256 1
 101 conv    128       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 128 0
 102 conv    256       3 x 3/ 1     52 x  52 x 128 ->   52 x  52 x 256 1
 103 conv    128       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 128 0
 104 conv    256       3 x 3/ 1     52 x  52 x 128 ->   52 x  52 x 256 1
 105 conv    255       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 255 0
 106 yolo
[yolo] params: iou loss: mse (2), iou_norm: 0.75, cls_norm: 1.00, scale_
Total BFLOPS 65.879
avg_outputs = 532444
 Allocate additional workspace_size = 52.43 MB
Loading weights from yolov3.weights...
 seen 64, trained: 32013 K-images (500 Kilo-batches_64)
Done! Loaded 107 layers from weights-file
 Detection layer: 82 - type = 27
 Detection layer: 94 - type = 27
 Detection layer: 106 - type = 27
data/person.jpg: Predicted in 41.381000 milli-seconds.
dog: 99%
person: 100%
horse: 100%
```



工作环境



数据集

IDEA LAB
浙江大学-阿里巴巴联合实验室

数据集拍摄

- 拍摄设备：iPhone 11
- 辅助设备：DJI OSMO Mobile3

- ✓ 云台目标追踪
- ✓ 大致控制iso、曝光

数据集拍摄

- 拍摄设备：iPhone 11
- 辅助设备：DJI OSMO Mobile3

- ✓ 云台目标追踪
- ✓ 大致控制iso、曝光

- ✓ 偏暖色调的沙发
- ✓ 偏冷色调的墙壁

数据集拍摄

- 拍摄设备：iPhone 11
- 辅助设备：DJI OSMO Mobile3

- ✓ 云台目标追踪
- ✓ 大致控制iso、曝光

- ✓ 偏暖色调的沙发
- ✓ 偏冷色调的墙壁

最终筛除掉一些比较劣质的数据，共得到93张有效数据，数量并不是很多，
这是因为想通过学习数据集扩充的方法减少人力劳动。

IDEA LAB
浙江大学-阿里巴巴联合实验室

数据集拍摄

- 拍摄设备：iPhone 11
- 辅助设备：DJI OSMO Mobile3

- ✓ 云台目标追踪
- ✓ 大致控制iso、曝光

- ✓ 偏暖色调的沙发
- ✓ 偏冷色调的墙壁

- ✓ 图像压缩

```python
def compress(img):
    w, h = img.size
    compress_img = img.resize((int(w/1.2), int(h/1.2)), Image.ANTIALIAS)
    return compress_img
```

5M -> 500K

数据集扩充

在训练图像识别的深度神经网络时，使用大量更多的训练数据，可能会使网络得到更好的性能，例如提高网络的分类准确率，防止过拟合等。**人为扩展训练数据时对数据的操作最好能反映真实世界的变化**。人为扩充数据集之后如果分类准确率有明显的提升，说明我们对数据所做的拓展操作是良性的，能够“**反映真实世界的变化**”，就会被用到整个数据集的扩展。反之，则说明不能用此操作对数据集进行拓展。

数据集扩充

人为扩展训练数据时对数据的操作最好能反映真实世界的变化
“反映真实世界的变化”

数据集扩充

图像整体加上一个随机偏差，或整体进行尺度的放缩

- 图像强度变换
  - 亮度变化： lightness
  - 对比度变化：contrast
- 图像滤波
  - 锐化：sharpen
  - 高斯模糊：blur
- 透视变换
  - 镜像翻转：flip
  - 图像裁剪：crop
  - 图像拉伸：deform
  - 镜头畸变：distortion
- 注入噪声
  - 椒盐噪声：noise
  - 渐晕：vignetting
- 其他
  - 随机抠除：cutout

```
'''
_brightness
    图像亮度增强
'''
def brightness(img):
    img = Image.fromarray(img)

    brightness = 1 + np.random.randint(1, 9) / 10
    brightness_img = img.point(lambda p: p * brightness)

    return Image.fromarray(np.uint8(brightness_img))
```

```
'''
_darkness
    图像亮度降低
'''
def darkness(img):
    darkness = np.random.randint(1, 9) / 10
    darkness_img = img * darkness
    return Image.fromarray(np.uint8(darkness_img))

def saveDarknessLabel(name):
    shutil.copyfile(name + ".txt", name + "_darkness.txt")
```

数据集扩充

图像整体加上一个随机偏差，或整体进行尺度的放缩

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

扩展图像灰度级动态范围，对两极的像素进行压缩，对中间范围的像素进行扩展

```
'''
_contrast
    对比度变换
'''
def contrast(img):
    img = Image.fromarray(img)
    range_contrast=(-50, 50)

    contrast = np.random.randint(*range_contrast)

    contrast_img = img.point(lambda p: p * (contrast / 127 + 1) - contrast)

    return Image.fromarray(np.uint8(contrast_img))
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

扩展图像灰度级动态范围，对两极的像素进行压缩，对中间范围的像素进行扩展

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

增强图像边缘信息

```
'''
_sharpen
    图像锐化
'''
def sharpen(img):
    img = cv2.cvtColor(np.asarray(img),cv2.COLOR_RGB2BGR)

    identity = np.array([[0, 0, 0],
                         [0, 1, 0],
                         [0, 0, 0]])
    sharpen = np.array([[ 0, -1,  0],
                        [-1,  4, -1],
                        [ 0, -1,  0]]) / 4
    max_center = 4

    sharp = sharpen * np.random.random() * max_center
    kernel = identity + sharp

    sharpen_img = cv2.filter2D(img, -1, kernel)
    return Image.fromarray(cv2.cvtColor(sharpen_img,cv2.COLOR_BGR2RGB))
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

增强图像边缘信息

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

图像平滑

```
'''
_blur
    高斯模糊
'''
def blur(img):
    img = cv2.cvtColor(np.asarray(img),cv2.COLOR_RGB2BGR)

    kernel_size = (7, 7)
    blur_img = cv2.GaussianBlur(img,kernel_size,0)

    return Image.fromarray(cv2.cvtColor(blur_img,cv2.COLOR_BGR2RGB))
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

图像平滑

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

使图像沿长轴进行翻转

```
'''
_flip
    图像左右翻转
'''
def flip(img):
    flip_img = cv2.flip(cv2.cvtColor(np.asarray(img),cv2.COLOR_RGB2BGR), 1)

    return Image.fromarray(cv2.cvtColor(flip_img,cv2.COLOR_BGR2RGB))
```

```
def saveFlipLabel(name):
    with open(name + "_flip.txt", "w") as outfile:
        with open(name + ".txt", "r") as infile:
            for line in infile.readlines():
                words = line.split(" ")
                horizontal_coord = float(words[1])
                outfile.write(words[0] + " " + str(format(1-horizontal_coord, ".6f")) + " " + words[2] + " " +
                words[3] + " " + words[4])
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

使图像沿长轴进行翻转

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

裁剪原图80%大小的中心图像，并进行随机移动

```
'''
_crop
    裁剪
    取中心的80%部分，并进行随机便宜
    (可能将目标对象裁减掉，因此采用手工重新标注)
'''
def crop(img):
    size = img.shape[:2]
    kernel_size = list(map(lambda x: int(x*0.8), size))
    shift_min, shift_max = -50, 50
    shift_size = [np.random.randint(shift_min, shift_max), np.random.randint(shift_min, shift_max)]

    crop_img = img[
        (size[0]-kernel_size[0])//2+shift_size[0]:(size[0]-kernel_size[0])//2+kernel_size[0]+shift_size[0],
        (size[1]-kernel_size[1])//2+shift_size[1]:(size[1]-kernel_size[1])//2+kernel_size[1]+shift_size[1]
    ]

    return Image.fromarray(np.uint8(crop_img))
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

裁剪原图80%大小的中心图像，并进行随机移动

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

拉伸成长宽为原始宽的正方形图像

```
'''
_deform
    图像拉伸
    拉伸成长宽为原始宽的正方形图像
    (需要重新手工标注)
'''
def deform(img):
    img = Image.fromarray(img)
    w, h = img.size[:2]

    # 拉伸成宽为w的正方形
    deform_img = img.resize((int(w), int(w)))

    return deform_img
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

拉伸成长宽为原始宽的正方形图像

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

对图像进行透视变化，模拟鱼眼镜头的镜头畸变
通过播放径向系数k1，k2，k3和切向系数p1，p2实现

```
'''
_distortion
    镜头畸变
    (需要重新手工标注)
'''
def distortion(img):
    d_coef= np.array((0.15, 0.15, 0.1, 0.1, 0.05))

    # get the height and the width of the image
    h, w = img.shape[:2]

    # compute its diagonal
    f = (h ** 2 + w ** 2) ** 0.5

    # set the image projective to carrtesian dimension
    K = np.array([[f, 0, w / 2],
                  [0, f, h / 2],
                  [0, 0,   1  ]])

    d_coef = d_coef * np.random.random(5) # value
    d_coef = d_coef * (2 * (np.random.random(5) < 0.5) - 1) # sign

    # Generate new camera matrix from parameters
    M, _ = cv2.getOptimalNewCameraMatrix(K, d_coef, (w, h), 0)

    # Generate look-up tables for remapping the camera image
    remap = cv2.initUndistortRectifyMap(K, d_coef, None, M, (w, h), 5)

    # Remap the original image to a new image
    distortion_img = cv2.remap(img, *remap, cv2.INTER_LINEAR)

    return Image.fromarray(np.uint8(distortion_img))
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

对图像进行透视变化，模拟鱼眼镜头的镜头畸变
通过播放径向系数k1，k2，k3和切向系数p1，p2实现

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

在图像中随机添加白/黑像素

```
'''
_noise
    为图像添加噪声
    随机生成5000个椒盐
'''
def addNoise(img):
    rows,cols,dims = img.shape
    noise_img = img
    for i in range(5000):
        x = np.random.randint(0,rows)
        y = np.random.randint(0,cols)
        noise_img[x,y,:] = 255
    noise_img.flags.writeable = True  # 将数组改为读写模式

    return Image.fromarray(np.uint8(noise_img))
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

在图像中随机添加白/黑像素

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

对图像添加一个圆范围内的噪声模拟光晕

```
'''
_vignetting
    渐晕
'''
def vignetting(img):
    ratio_min_dist=0.2
    range_vignette=np.array((0.2, 0.8))
    random_sign=False

    h, w = img.shape[:2]
    min_dist = np.array([h, w]) / 2 * np.random.random() * ratio_min_dist

    # create matrix of distance from the center on the two axis
    x, y = np.meshgrid(np.linspace(-w/2, w/2, w), np.linspace(-h/2, h/2, h))
    x, y = np.abs(x), np.abs(y)

    # create the vignette mask on the two axis
    x = (x - min_dist[0]) / (np.max(x) - min_dist[0])
    x = np.clip(x, 0, 1)
    y = (y - min_dist[1]) / (np.max(y) - min_dist[1])
    y = np.clip(y, 0, 1)

    # then get a random intensity of the vignette
    vignette = (x + y) / 2 * np.random.uniform(*range_vignette)
    vignette = np.tile(vignette[..., None], [1, 1, 3])

    sign = 2 * (np.random.random() < 0.5) * (random_sign) - 1
    vignetting_img = img * (1 + sign * vignette)

    return Image.fromarray(np.uint8(vignetting_img))
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

对图像添加一个圆范围内的噪声模拟光晕

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

随机抠出四个位置，并用黑色/彩色矩形填充

```
'''
_cutout
    随机抠除四个位置
    用黑色/彩色矩形填充
'''
def cutout(img):
    min_size_ratio, max_size_ratio = 0.1, 0.3
    channel_wise = False
    max_crop = 4
    replacement=0

    size = np.array(img.shape[:2])
    mini, maxi = min_size_ratio * size, max_size_ratio * size
    cutout_img = img
    for _ in range(max_crop):
        # random size
        h = np.random.randint(mini[0], maxi[0])
        w = np.random.randint(mini[1], maxi[1])
        # random place
        shift_h = np.random.randint(0, size[0] - h)
        shift_w = np.random.randint(0, size[1] - w)

        if channel_wise:
            c = np.random.randint(0, img.shape[-1])
            cutout_img[shift_h:shift_h+h, shift_w:shift_w+w, c] = replacement
        else:
            cutout_img[shift_h:shift_h+h, shift_w:shift_w+w] = replacement

    return Image.fromarray(np.uint8(cutout_img))
```

数据集扩充

图像强度变换
- 亮度变化： lightness
- 对比度变化：contrast

图像滤波
- 锐化：sharpen
- 高斯模糊：blur

透视变换
- 镜像翻转：flip
- 图像裁剪：crop
- 图像拉伸：deform
- 镜头畸变：distortion

注入噪声
- 椒盐噪声：noise
- 渐晕：vignetting

其他
- 随机抠除：cutout

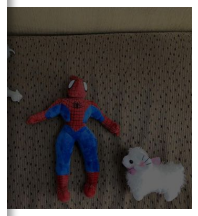
随机抠出四个位置，并用黑色/彩色矩形填充

数据集扩充

数据集扩充

背景引入

在训练图像识别的深度神经网络时，使用大量更多的训练数据，可能会使网络得到更好的性能，例如提高网络的分类准确率，防止过拟合等。人为扩展训练数据时对数据的操作最好能反映真实世界的变化。人为扩充数据集之后如果分类准确率有明显的提升，说明我们对数据所做的拓展操作是良性的，能够“反映真实世界的变化”，就会被用到整个数据集的扩展。反之，则说明不能用此操作对数据集进行拓展。



```
n.randint(1, 9) / 10
(lambda p: p * brightness)
```

```
int(*range_contrast)
lambda p: p * (contrast / 127 + 1)
```

```
center
```

数据集标注

labelImg

安装步骤

1. download from github repo
2. install dependencies

```
pip3 install pyqt5 lxml
```

3. build from resource
   - **mac**

```
make qt5py3
```

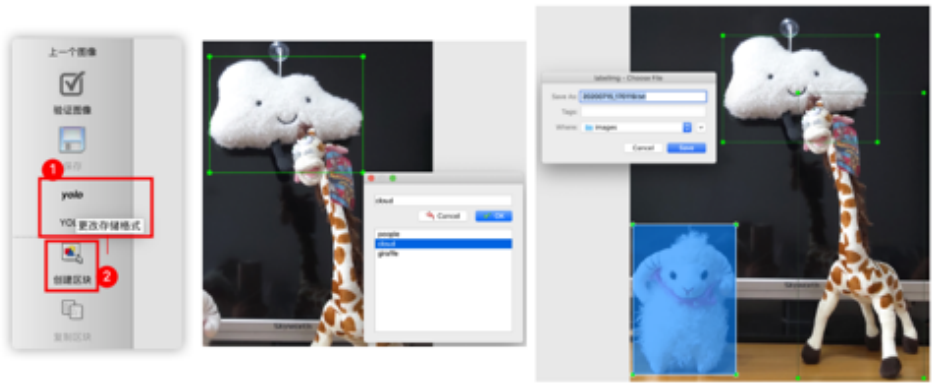
   - **win10**

```
pyqt5, pyrcc5 -o libs/resources.py resou
```

4. 准备`classes.txt`和`images/`
5. 运行qt工程

```
python labelImg.py images/ classes.txt
```

使用方法



1. 切换存储格式为yolo
2. 创建区块
3. 拖拽区域进行标注并选择种类
4. 保存label到`.txt`文件

数据集标注

labelImg

标注结果

```
2 0.516246 0.489583 0.967508 0.353107
```

- pos0: 所标目标的类别

  > 2对应第三个类别

- pos1: 目标垂直方向起始位置 / 图像总高度
- pos2: 目标水平方向起始位置 / 图像总宽度
- pos3: 目标水平宽度 / 图像总宽度
- pos4: 目标垂直高度 / 图像总高度

数据集汇总

拍摄的两个场景原始照片

将main进行其他处理后的图片并自动进行重标注

之前同学留下来的照片（筛选）

将main进行crop, deform, distortion处理后的图片进行手工标注

共1551张标注图片数据

模型训练

Yolo训练自己的数据集

- Labeled Custom Dataset
- Custom .cfg file
- obj.data and obj.names files
- train.txt file

Using Google's Open Images Dataset

> 由于实验室的任务针对特定几种玩具，且在扩展性上没有太强硬的要求。因此谷歌的开源数据集仅做学习使用，还是采用自己标注数据集的方法进行数据集的构建

- Google's Open Images Dataset
- OIDv4 toolkit

Manually Labeling Images with labelImg(Annotation Tool)

- labelImg工具使用

Yolo训练自己的数据集

- Labeled Custom Dataset
- Custom .cfg file
- obj.data and obj.names files
- train.txt file

edit the `yolov4.cfg` to fit the needs based on the object detector

- `bash=64` & `subdivisions=16`：网上比较推荐的参数

  > 这里受限于服务器容量 将subdivisions设为32，但是速度仍然很慢

- `classes=4` in the three YOLO layers
- `filters=(classes + 5) * 3`: three convolutional layers before the YOLO layers
- `width=416` & `height=416`: any multiple of 32, 416 is standard
  - improve results by making value larger like 608 but will slow down training
- `max_batches=(# of classes) * 2000`: but no less than 6000
- `steps=(80% of max_batches), (90% of max_batches)`
- `random=1`: if run into memory issues or find the training taking a super long time, change three yolo layers from 1 to 0 to speed up training but slightly reduce accurancy of model

- **train**:
  - $classes$ = 待分类的类别数
  - $filters = (classes + 5) \times 3$

```
[convolutional]
size=1
stride=1
pad=1
filters=27
activation=linear


[yolo]
mask = 6,7,8
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,
116,90,  156,198,  373,326
classes=4
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=1
```

- **test**
  - batch: 1
  - subdivisions: 1

```
[net]
# Testing
batch=1
subdivisions=1
```

Yolo训练自己的数据集

- Labeled Custom Dataset
- Custom .cfg file
- obj.data and obj.names files
- train.txt file

obj.data

包含待分类的种类数，训练集、验证集等

> 由于本次这是做模型测试，故指定train和valid为同一列表

```
classes= 4
train  = data/train.txt
valid  = data/train.txt
names = data/obj.names
backup = backup
```

obj.names

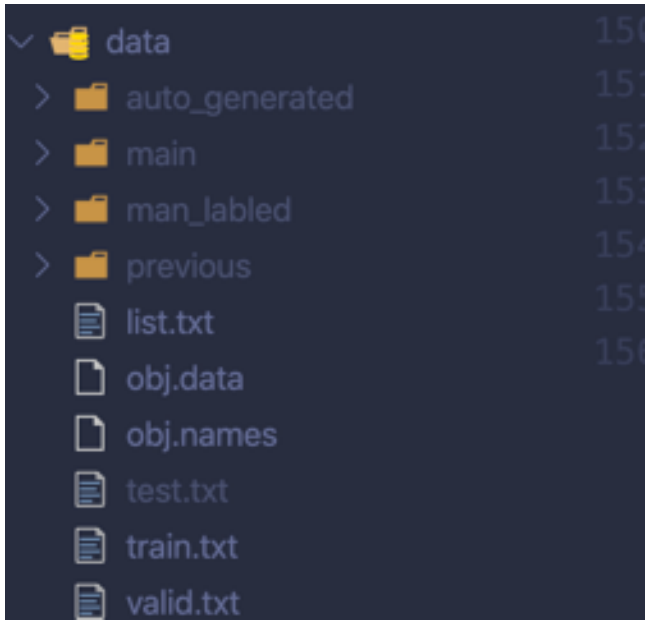
待分类的四类对象的名称

```
sheep
giraffe
cloud
people
```

Yolo训练自己的数据集

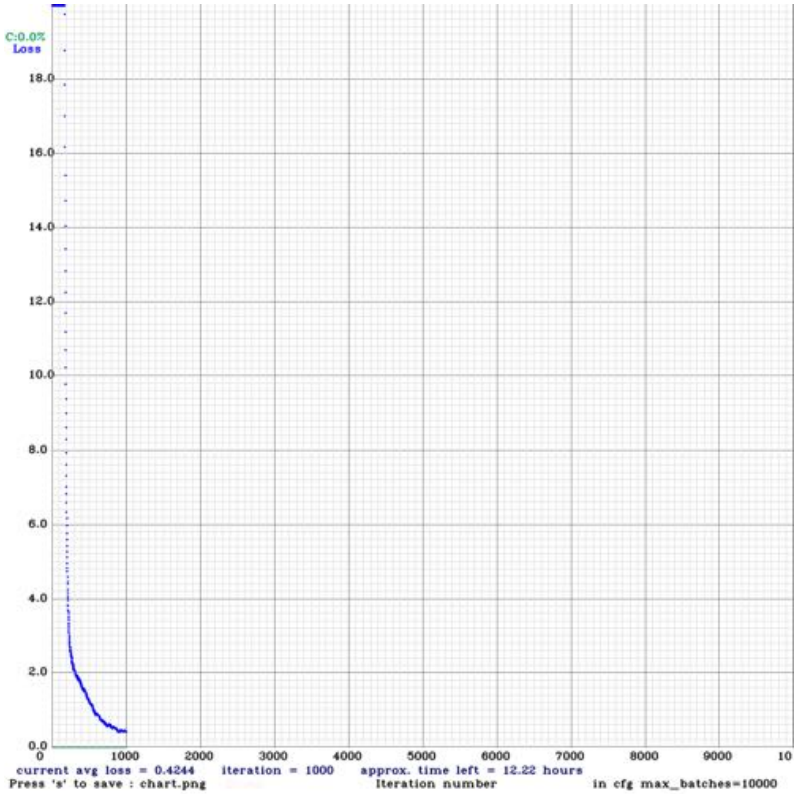
- Labeled Custom Dataset
- Custom .cfg file
- obj.data and obj.names files
- train.txt file

Train	8
Valid	1
Test	1

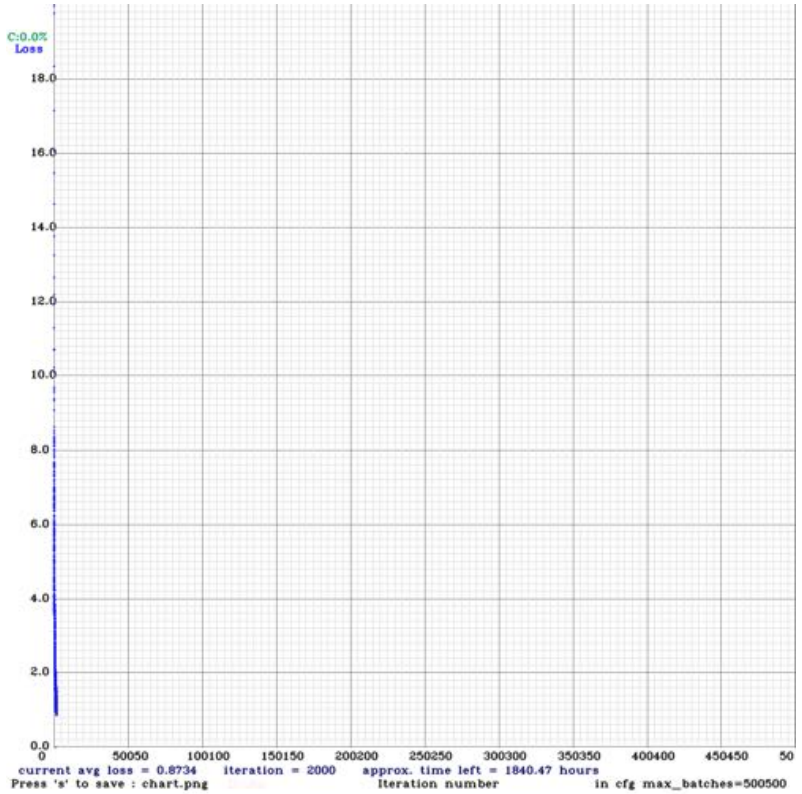


训练

```
./darknet detector train ../../data/obj.data cfg/yolov4_custom.cfg
yolov4.conv.137 -dont_show
```



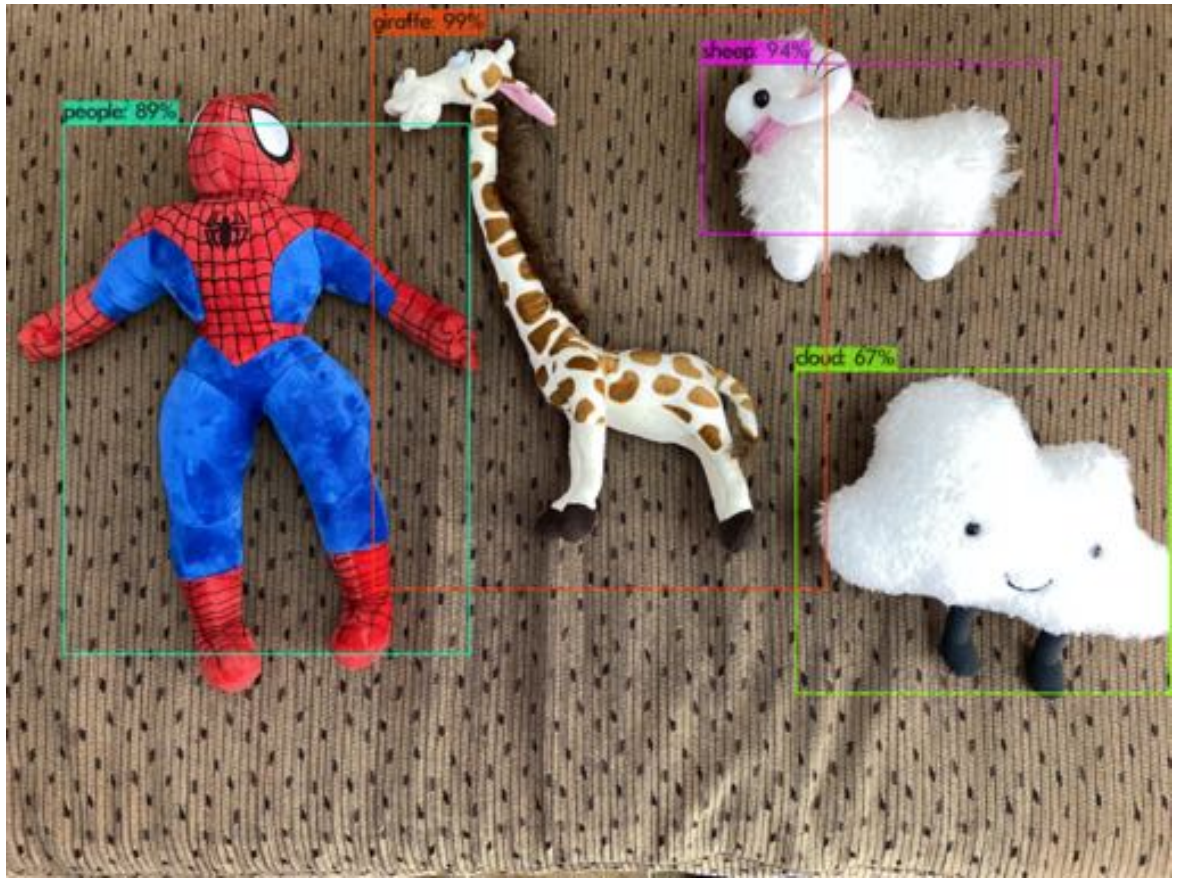
yolov3



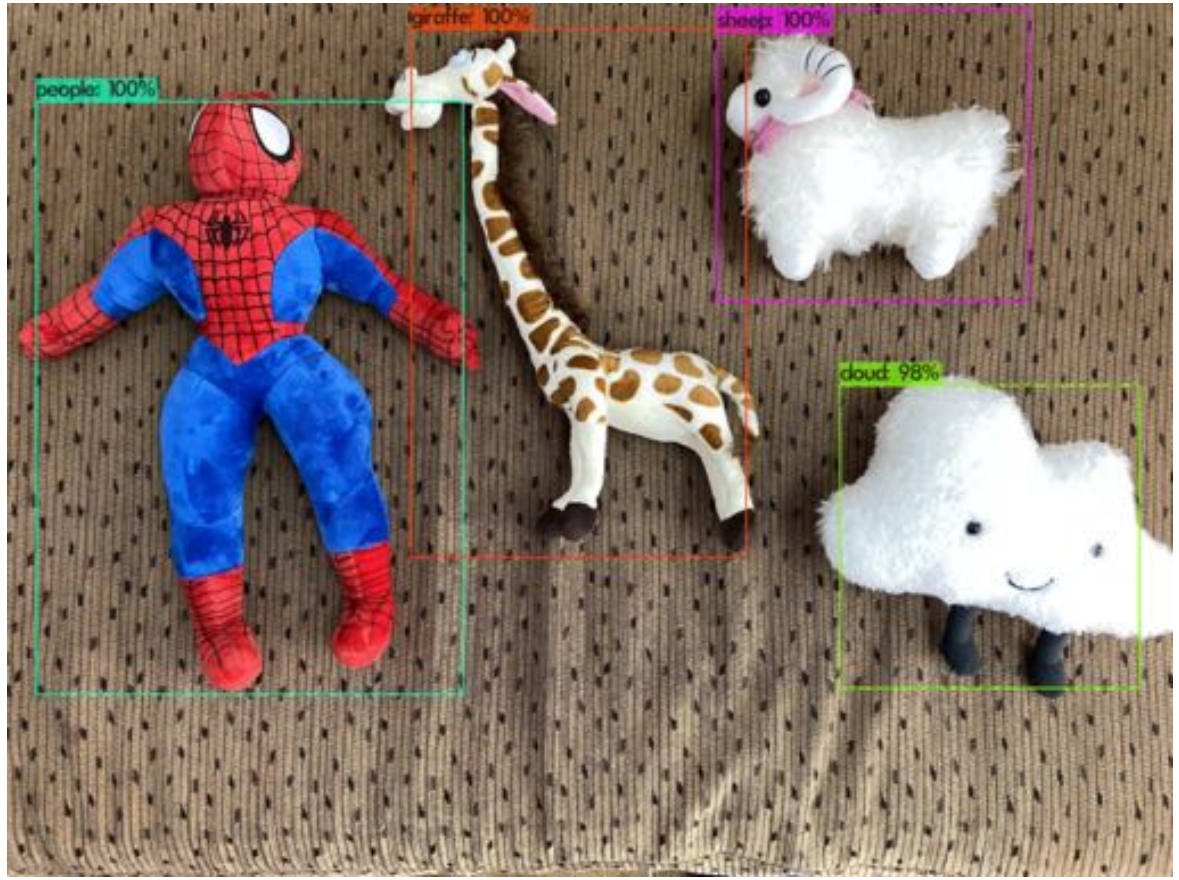
yolov4

测试

```
./darknet detector test ../../data/obj.data
cfg/yolov3_custom_test.cfg backup/yolov3_custom_last.weights
../../data/demo.jpg -thresh 0.25 -dont-show
```

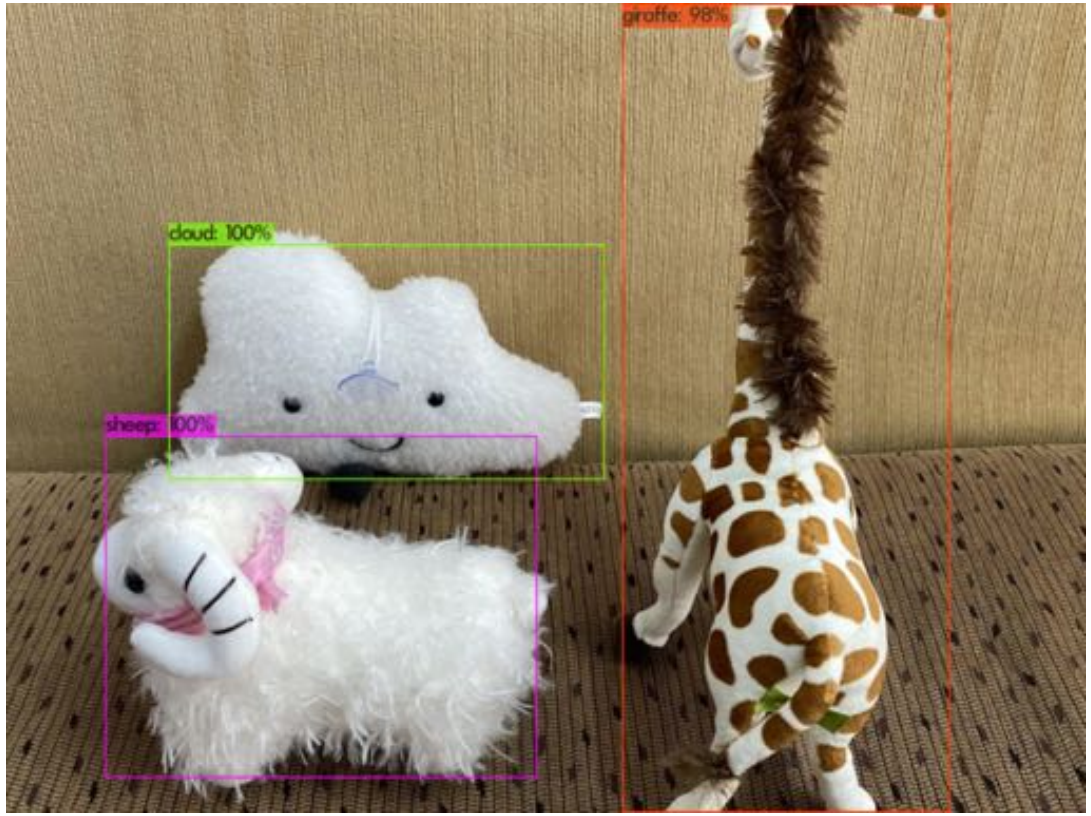


yolov3 1000 epochs

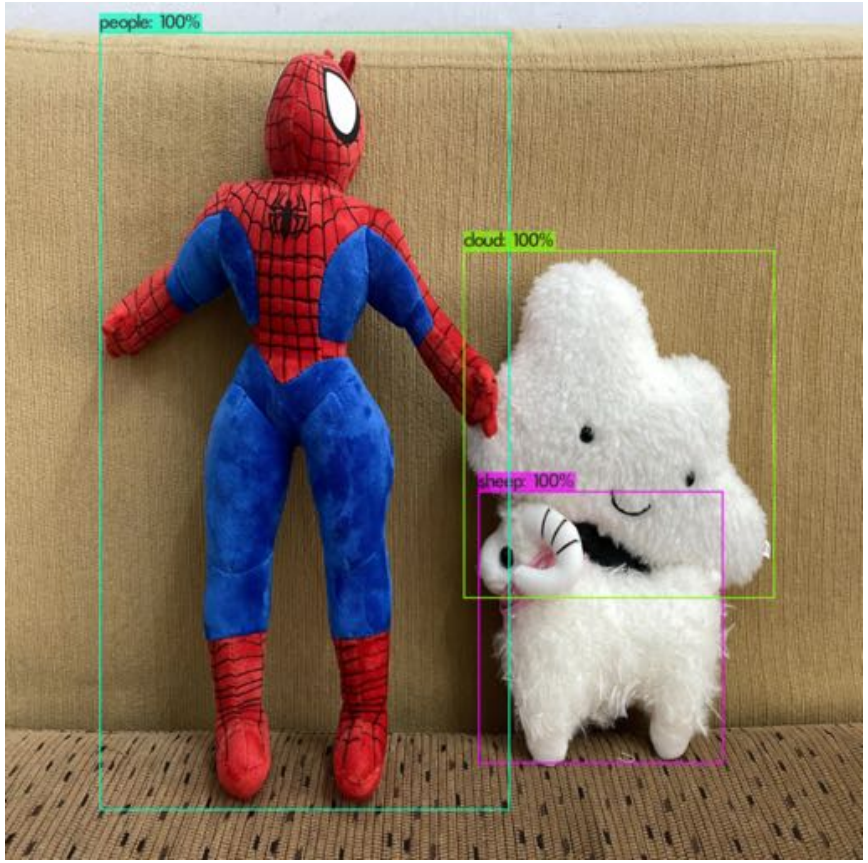


yolov3 2000 epochs

测试

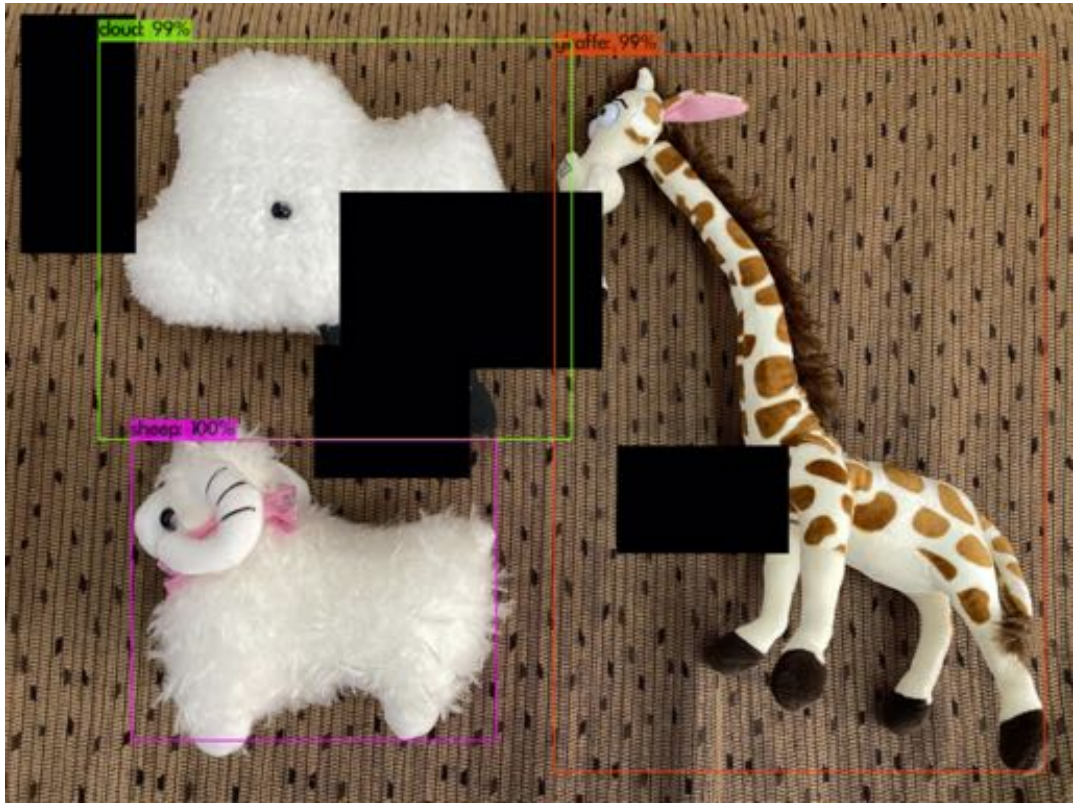


yolov4 2000 epochs - crop



yolov4 2000 epochs - deform

测试

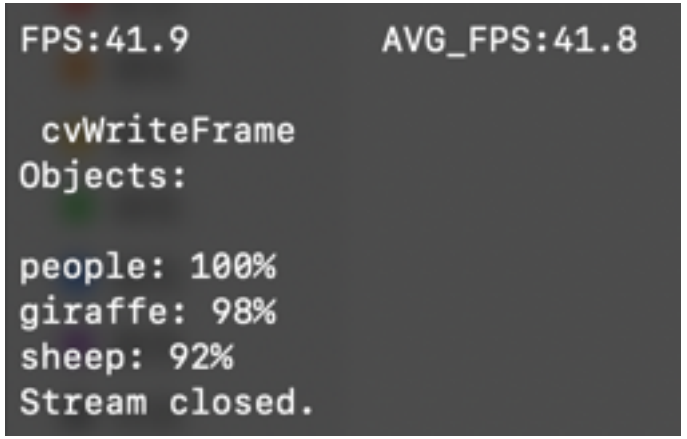


yolov4 2000 epochs - cutout

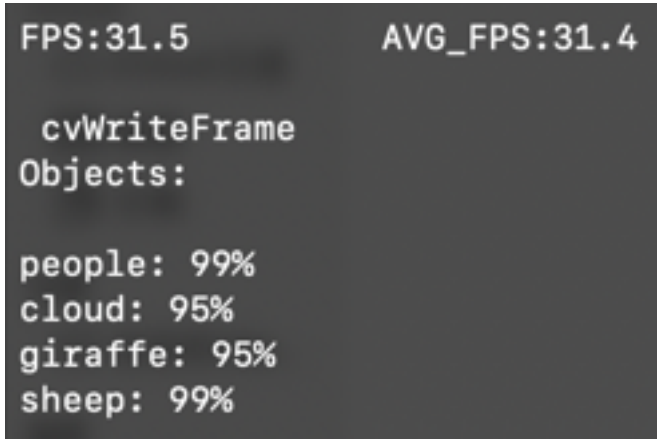
yolov4 2000 epochs - darkness

测试

```
./darknet detector test ../../data/obj.data
cfg/yolov3_custom_test.cfg backup/yolov3_custom_last.weights
../../data/demo.jpg -thresh 0.25 -dont-show
```



yolov3 2000 epochs



yolov4 2000 epochs

测试

yolov4 2000 epochs

mAP

Checking the mAP of the Model

mAP: mean average precision

```
./darknet detector map ../../data/obj.data cfg/yolov4_custom.cfg
backup/yolov4_custom_last.weights
```

the highest mAP, the most accurate is

```
 detections_count = 346, unique_truth_count = 290
class_id = 0, name = sheep, ap = 100.00%         (TP = 88, FP = 0)
class_id = 1, name = giraffe, ap = 100.00%       (TP = 78, FP = 0)
class_id = 2, name = cloud, ap = 100.00%         (TP = 77, FP = 0)
class_id = 3, name = people, ap = 100.00%        (TP = 47, FP = 0)

 for conf_thresh = 0.25, precision = 1.00, recall = 1.00, F1-score = 1.00
 for conf_thresh = 0.25, TP = 290, FP = 0, FN = 0, average IoU = 87.04 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 1.000000, or 100.00 %
Total Detection Time: 17 Seconds
```

yolo3 2000 epochs - mAP

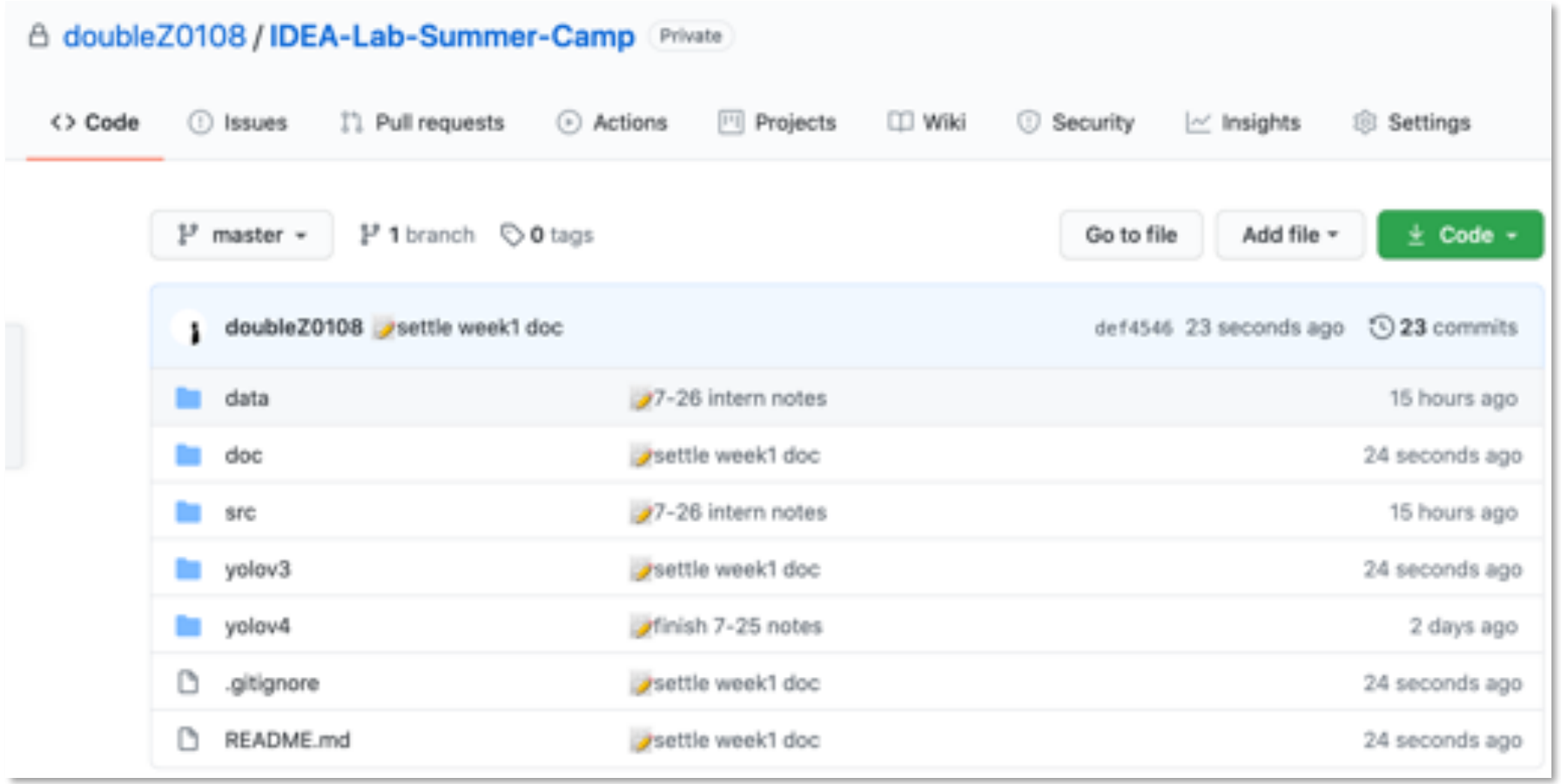
```
 detections_count = 392, unique_truth_count = 290
class_id = 0, name = sheep, ap = 99.97%          (TP = 88, FP = 1)
class_id = 1, name = giraffe, ap = 99.98%        (TP = 78, FP = 1)
class_id = 2, name = cloud, ap = 100.00%         (TP = 77, FP = 0)
class_id = 3, name = people, ap = 100.00%        (TP = 47, FP = 0)

 for conf_thresh = 0.25, precision = 0.99, recall = 1.00, F1-score = 1.00
 for conf_thresh = 0.25, TP = 290, FP = 2, FN = 0, average IoU = 88.30 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.999896, or 99.99 %
Total Detection Time: 19 Seconds
```

yolov4 2000 epochs - mAP

版本控制


doubleZ0108 / IDEA-Lab-Summer-Camp
Private
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
master
1 branch
0 tags
Go to file
Add file
Code
doubleZ0108 settle week1 doc
def4546 23 seconds ago
23 commits
data
7-26 intern notes
15 hours ago
doc
settle week1 doc
24 seconds ago
src
7-26 intern notes
15 hours ago
yolov3
settle week1 doc
24 seconds ago
yolov4
finish 7-25 notes
2 days ago
.gitignore
settle week1 doc
24 seconds ago
README.md
settle week1 doc
24 seconds ago

笔记

实习报告 | Internship Report

- 暑期实习任务清单

- IDEA Lab实习笔记 7-23
- IDEA Lab实习笔记 7-24
- IDEA Lab实习笔记 7-25
- IDEA Lab实习笔记 7-26
- IDEA Lab实习汇报

学习笔记 | Study Notes

- ubuntu下配置opencv环境
- labelImg工具使用
- 数据集扩充

Scrum过程模型

Product Owner

孙凌云教授

- Determine the function of the project.
- For each sprint, adjust features and priorities as needed.
- Inspection work of the development team.

Scrum Master

吴敬宇学长

- Link Team and Product owner.
- Organize Daily Scrum, Sprint Review and Sprint Planning meetings.
- Ensure good collaboration between members.
- Resolve obstacles in team development.

Team

张喆实习生

- Mainly responsible for product development.
- Deliver potentially deliverable product increments after each Iteration. Ensure that the goals of the Sprint are achieved.

Scrum过程模型

Planning meeting

7月22日第一次接到暑期实习的任务
每周组会时进行计划安排

Daily meeting

每天晚上19点~21点，与吴学长通过钉钉进行远程视频，汇报当天进度，并适当调整之后的进度安排

Review meeting

7月27日参加小组组会
会上学长对我这一周的进展进行评价并提出建议

Acceptance meeting

7月28日与老师同学分享这一周的进展
并听取老师点评，对接下来的工作进行改正

工作规划

	工作规划	负责人	完成时间
实验	①将后续的3～4个玩具进行数据集采集 ②重新训练具有8个目标识别能力的网络 ③尝试FastRCNN用于目标检测 ④实验不同的划分策略和训练策略对结果的影响	张喆	2020.8.3
算法	①阅读yolov3相关博客和论文了解原理 ②阅读yolov4相关博客和论文了解原理	张喆	2020.8.3
实验+算法	①阅读scene graph相关博客和论文了解思想 ②尝试利用识别出的目标构建scene graph	张喆	2020.8.10

智能、设计、体验与审美
Intelligence, Design, Experience, & Aesthetics

IDEA LAB
浙江大学-阿里巴巴联合实验室

GAN组进展汇报

玩具图像识别子系统

时间 **2020.7.27**
地点 **钉钉(online)**
汇报人 **实习生-张喆**