工程流程

本文档实现了R-CNN算法进行目标检测的完整过程，包括

1. 数据集创建
2. 卷积神经网络训练
3. 分类器训练
4. 边界框回归器训练
5. 目标检测器实现

**本仓库最终实现一个汽车类别目标检测器**

模块构成[¶](https://r-cnn.readthedocs.io/zh_CN/latest/#_2)

1. 区域建议生成：selectivesearch算法实现，生成类别独立的区域建议
2. 特征提取：卷积神经网络AlexNet实现，从每个区域建议中提取固定长度的特征向量
3. 线性SVM实现，输入特征向量，输出每类成绩
4. 使用类指定的边界框回归器计算候选建议的坐标偏移
5. 非最大抑制方法实现，得到最终的候选建议

*关于区域建议算法selectivesearch实现，在训练阶段使*用快速模式.

├── docs # 说明文档

├── imgs # 测试图像

├── LICENSE

├── mkdocs.yml

├── py

│   ├── bbox\_regression.py # 边界框回归器训练

│   ├── car\_detector.py # 检测器实现

│   ├── finetune.py # 卷积神经网络微调训练

│   ├── \_\_init\_\_.py

│   ├── linear\_svm.py # 分类器训练

│   ├── requirements.txt # python工程依赖

│   ├── selectivesearch.py # 选择性搜索算法实现

│   └── utils

│   ├── data # 创建数据/自定义数据处理类

│   └── util.py # 辅助函数

本工程最终目标是实现一个汽车检测器，在训练过程中需要依次实现以下数据集

1. 从VOC数据集中提取汽车类别
2. 创建微调数据集
3. 创建分类器数据集

pascal\_voc\_car.py

创建得到数据集voc\_car，其格式如下：

── train

│   ├── Annotations

│   ├── car.csv

│   └── JPEGImages

└── val

├── Annotations

├── car.csv

└── JPEGImages

* create\_classifier\_data.py

通过选择性搜索算法的质量模式获取候选建议，然后计算候选建议与标注边界框的IoU

1. 正样本：标注边界框
2. 负样本：IoU小于0.3的候选区域

在代码实现中，为了进一步减小负样本数据集，我修改了负样本判定规则

* 副样本（修改后）：IoU大于0，小于0.3。为了进一步限制负样本数目，其大小必须大于标注框的1/5

# 微调模型

本工程通过PyTorch库进行卷积网络训练，其微调实现参考[迁移学习](https://blog.zhujian.life/posts/c7511b44.html)

## 预训练模型

PyTorch提供了AlexNet的预训练模型

* 微调实现：py/finetune.py
* 自定义微调数据集类：py/utils/data/custom\_finetune\_dataset.py
* 自定义批量采样器类：py/utils/data/custom\_batch\_sampler.py
* 辅助函数：py/utils/util.py
* 微调实现：py/finetune.py
* 自定义微调数据集类：py/utils/data/custom\_finetune\_dataset.py
* 自定义批量采样器类：py/utils/data/custom\_batch\_sampler.py
* 辅助函数：py/utils/util.py

模型在训练过程中呈现过拟合现象，可考虑调整学习率、添加权重衰减以及更换优化器的方式：

1. 学习率从1e-3调整为1e-4
2. 添加L2权重衰减，衰减因子为1e-4
3. 使用Adam替换SGD

实际训练中很少有网络能够拥有足够大的数据集进行训练，所以迁移学习是实际卷积网络训练过程中非常重要的步骤.

简介

首先将模型在大数据集（比如ImageNet，包含120万张共1000类的图像）上进行预训练，然后将训练后的模型作为指定数据集的初始化或者固定特征提取器，这一操作称为迁移学习（Transfer Learning）

## 适用场景

迁移学习主要有2个适用场景：

1. 将卷积网络作为**固定特征提取器**。除了最后的全连接层外，将会冻结所有网络的权重。最后的全连接层将会被一个新的随机初始化的全连接层替代，并且仅训练该层
2. **微调**卷积网络。不使用随机初始化而是用一个预训练网络来初始化网络，就像在ImageNet 1000数据集上训练网络一样，剩下的训练和往常一样。可以微调卷积网络的所有层，或者可以保持一些早期的层固定不变(由于过度拟合的问题)，并且只微调网络的一些较高层部分。这是因为观察到卷积网络的早期特征包含更多通用特征(例如，边缘检测器或颜色斑点检测器)，这些特征对许多任务都很有用，但是卷积网络的顶层对于原始数据集中包含的类的细节变得越来越具体。例如，在包含许多犬种的ImageNet的情况下，卷积网络的很大一部分表示能力可以用于区分犬种的特定特性

# 分类器训练

R-CNN在完成卷积模型的微调后，额外使用了线性SVM分类器，采用负样本挖掘方法进行训练，参考[Hard Negative Mining](https://blog.zhujian.life/posts/bc29003.html)

## HNM实现

1. 设置初始训练集，正负样本数比值为1:1（或者1:3，*具体比例根据实际训练效果决定*）
2. 每轮训练完成后，使用分类器对剩余负样本进行检测，如果检测为正，则加入到训练集中
3. 重新训练分类器，重复第二步，直到检测精度开始收敛

## 为什么要使用HNM

该方法应用在正负样本的数目相差很大的情况。通过HNM，选出对训练分类器有帮助的负样本，能够提高分类器性能

**注意：由于初始训练集的数目会很小，所以降低学习率有利于训练结果**

## R-CNN中的HNM实现

在训练特征提取时，训练参数如下：

1. 学习率：1e-3
2. 动量：0.9
3. 随步长衰减：每隔7轮衰减一次，参数因子α=0.1α=0.1
4. 迭代次数：25

在训练分类器时，使用HNM，训练参数如下：

1. 学习率：1e-4
2. 动量：0.9
3. 随步长衰减：每隔4轮衰减一次，参数因子α=0.1α=0.1
4. 迭代次数：10

线性SVM分类器包含了线性回归+折页损失，其中自定义PyTorch的折页损失实现：

def hinge\_loss(outputs, labels):

"""

折页损失计算

:param outputs: 大小为(N, num\_classes)

:param labels: 大小为(N)

:return: 损失值

"""

num\_labels = len(labels)

corrects = outputs[range(num\_labels), labels].unsqueeze(0).T

# 最大间隔

margin = 1.0

margins = outputs - corrects + margin

loss = torch.sum(torch.max(margins, 1)[0]) / len(labels)

# # 正则化强度

# reg = 1e-3

# loss += reg \* torch.sum(weight \*\* 2)

return loss

## 负样本挖掘[¶](https://r-cnn.readthedocs.io/zh_CN/latest/%E5%88%86%E7%B1%BB%E5%99%A8%E8%AE%AD%E7%BB%83/#_2)

实现流程如下：

1. 设置初始训练集，正负样本数比值为1:1（以正样本数目为基准）
2. 每轮训练完成后，使用分类器对剩余负样本进行检测，如果检测为正，则加入到训练集中
3. 重新训练分类器，重复第二步，直到检测精度开始收敛

## 训练参数[¶](https://r-cnn.readthedocs.io/zh_CN/latest/%E5%88%86%E7%B1%BB%E5%99%A8%E8%AE%AD%E7%BB%83/#_3)

1. 学习率：1e-4
2. 动量：0.9
3. 随步长衰减：每隔4轮衰减一次，参数因子α=0.1
4. 迭代次数：10
5. 批量处理：每次训练128个图像，其中32个正样本，96个负样本

# 边界框回归器训练

使用SVM分类器对候选建议进行分类后，使用对应类别的边界框回归器（bounding-box regression）预测其坐标偏移值，这一操作能够进一步提高检测精度

在R-CNN算法中，使用SVM分类器对候选建议进行分类后，使用对应类别的边界框回归器（bounding-box regression）预测其坐标偏移值，这一操作能够进一步提高检测精度

## 边界框回归器训练

* 在R-CNN算法中，使用AlexNet作为CNN模型，其第5个池化层输出256\*6\*6大小特征向量，所以权重ww大小为9216×49216×4
* 文章设置了候选建议和标注边界框的IoU阈值为>0.6
* 文章设置了超参数λ=1000

通过提高IoU阈值（>0.6）过滤正样本候选建议，将候选建议和标注边界框之间的转换看成线性回归问题，并通过岭回归（ridge regression）来训练权重w

# 检测器实现

## 实现流程[¶](https://r-cnn.readthedocs.io/zh_CN/latest/%E6%A3%80%E6%B5%8B%E5%99%A8%E5%AE%9E%E7%8E%B0/#_2)

目标检测的实现流程如下：

1. 输入图像
2. 使用选择性搜索算法计算得到候选建议
3. 逐个计算候选建议
   1. 使用AlexNet模型计算特征
   2. 使用线性SVM分类器计算得到分类结果
4. 对所有分类为汽车的候选建议执行非最大抑制

## 非最大抑制

在目标检测中，非最大值抑制（Non-Maximum Suppression, NMS）能够有效去除冗余的候选边界框.

## 实现流程

1. 输入一组候选边界框以及对应的置信度（通常为分类概率）
2. 通过置信度进行排序
3. 保留置信度最高的候选边界框
4. 计算保留的候选边界框和剩余边界框的IoU
5. 去除IoU超过**阈值**的剩余边界框
6. 对剩余的边界框重复2-5步，直至剩余边界框清空

## 阈值设置

通常为0.3-0.5