数据增强Data Augmentation

实验背景

深度学习训练过程中如果遇到过拟合或者在测试集中泛化能力不足的问题的时候，你可能会想到更多的新数据、添加正则项等，数据增强也是其中一种，特别是对于机器视觉的任务，数据增强技术尤为重要。

为什么需要数据增强呢? 深度学习虽然是公认的人工智能中一个里程碑式的工具， 但是并不是我们想象中的那么智能；深度学习模型只是用大量的参数去拟合训练数据中对于不同类别的区分，模型性能强依赖于训练数据的多样性；有时候找到区分不同类别的关键特征并不是我们想象的事物的本质特征，而只是训练数据中的特征。举一个极端的例子，我们要识别不同品牌的汽车A和B，如果训练数据集中品牌A都是黑色，品牌B都是红色，模型拟合结果可能只是识别颜色，如果测试集中有一辆红色品牌A的车，就会出现误识的情况。为了正确的识别车的品牌，我们可以增加不同颜色，不同状态等各种各样的车的图片，直观上让模型可以通过比较，摒弃不重要的特征，真正找到区分事物的本质特征。

所谓数据增强就是在现有数据基础上通过，通过一些方式扩大数据集的方式。本文的数据增强技术主要指图像处理方面的。

数据增强的作用：

- 增加数据量，增强模型的泛化能力
- 引入一些数据噪声，增强模型鲁棒性

数据增强的种类：

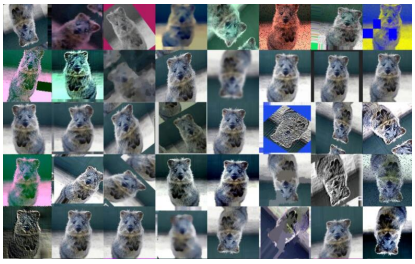
- 离线增强：在训练前对数据集进行处理，喂入模型的训练集是源数据的若干倍；比较适合小数据集
- 在线增强：如果是比较大的数据集，如果直接扩大数据集，训练的资源使用就会变大和效率就会降低。在线增强是指在训练的时候对数据进行变化，一般是以batch为单位进行增强操作。

实验内容

常用的数据增强操作：imgaug库

imgaug介绍

imgaug是提供了各种图像增强操作的python库https://github.com/aleju/imgaug



imgaug几乎包含了所有主流的数据增强的图像处理操作， 增强方法详见github

- 通道混淆
- 添加噪声
- 模糊：高斯、中值、运动模糊

- 遮罩cutout， dropout
- 颜色和亮度、对比度、饱和度变换
- 图像卷积操作：锐化，边缘检测等
- 左右翻转flip
- 几何变换：仿射变换
- 大小变换：resize、裁剪crop

imgaug还提供这些图像处理方法的随机组合方式。

imgaug API使用

安装

```
pip install imgaug
```

执行流程

- 定义增强方法sequential: 图片增强方法的集合， 通过`random_order`来控制是否按顺序执行集合中的操作
- 读取图片
- 执行增强操作

```
from imgaug import augmenters as iaa
import matplotlib.pyplot as plt
import cv2

# 定义增强操作的序列
seq = iaa.Sequential([
        iaa.CoarseDropout(0.05, size_percent=(0.02, 0.05)),
        ], random_order=False)

# 读取图片
file = 'test.jpg'
image = cv2.cvtColor(cv2.imread(str(file)), cv2.COLOR_BGR2RGB)

# 执行增强操作
aug_imgs = seq.augment_images([image])

# 测试展示
plt.imshow(image)
seq.show_grid(image, cols=10, rows=3)
```

对于每一个增强操作，`imgaug`提供执行的概率`Sometimes`和个数`SomeOf`和`OneOf`等选择， 选择操作是可嵌套

```
#  50%的概率执行这个Crop裁剪操作
 iaa.Sometimes(0.5, iaa.Crop(percent=(0.1, 0.05)))
```

```
# SomeOf 执行其中的若干个(0和5之间的随机数)；
# OneOf 执行其中的一个
iaa.SomeOf((0, 5), [
                iaa.OneOf([
                    iaa.GaussianBlur((0, 2.0)),
                    iaa.AverageBlur(k=(2, 5)),
                    iaa.MedianBlur(k=(3, 7)),
                ]), ## 模糊处理
```

自定义增强操作：Lambda增强器

```
# images 为img列表
def img_func(images, random_state, parents, hooks):
    for img in images:
        img[::4] = 0
    return images

def keypoint_func(keypoints_on_images, random_state, parents, hooks):
    return keypoints_on_images

seq = iaa.Sequential([
        iaa.Lambda(img_func, keypoint_func),
        iaa.CoarseDropout(0.05, size_percent=(0.02, 0.05)),
        ], random_order=False)
```

tensorflow中使用数据增强tf.py_func

```
# py_func可以支持tensor输入，tensor输出，而执行的函数中可以使用非tensor的操作
images = tf.py_func(seq.augment_images, [images], Tout=tf.uint8)
```

https://imgaug.readthedocs.io/en/latest/

mixup

batch augmentation

pair samples

实验结语

本实验和大家描述了数据增强的几种方法：

- imgaug库
- mixup方法
- batch augmentation方法
- pair samples方法

希望对大家有帮助。