实战案例9：交通标识自动识别

作者： Robin
日期： 2018/10
提问： 小象问答
数据集来源： https://btsd.ethz.ch/shareddata/

1. 案例描述

自动驾驶已经迎来发展的热潮。自动驾驶车在行驶时，需要正确识别路上的交通标识。本案例通过搭建卷积神经网络对比利时的62种交通标志进行自动识别。

2. 数据集描述

- 训练集下载地址：https://btsd.ethz.ch/shareddata/BelgiumTSC/BelgiumTSC_Training.zip
- 测试集下载地址：https://btsd.ethz.ch/shareddata/BelgiumTSC/BelgiumTSC_Testing.zip
- 将下载的数据集，解压后放到`data`目录中
- class的个数：62

3. 任务描述

- 搭建深度神经网络CNN，并进行交通标识分类

4. 主要代码解释

- 代码结构

```
lect09_proj
├── output  # 程序输出结果保存的目录
├── data  # 训练集和测试集存放的目录
├── main.py     # 主程序
├── utils.py     # 工具文件，包含数据加载、构建CNN等
├── config.y    # 配置文件
├── lect09_proj_readme.pdf  # 案例讲解文档
```

- **utils.py**

读取数据时，需要对图片进行格式转化，用于CNN的输入。并且将像素值其归一化到0-1。

```
def load_data(data_dir):
    ...
          # 以指定的尺寸读取图像数据
          image_data = image.load_img(f, target_size=(config.img_rows,
config.img_cols))
          # 将图片像素值转换为0-1
          image_data = image.img_to_array(image_data) / 255
    ...
```

- **utils.py**

数据矩阵`X`的格式为`[n_samples, n_rows, n_cols, n_channels]`，即`[样本个数, 图片高度, 图片宽度, 通道个数]`。对于标签可以通过`keras`的`to_categotical()`方法转换成独热编码的形式。

```
def process_data(images, labels):
    """
        处理加载的图像数据和标签，用于CNN的输入
    """
    X = np.array(images)
    y = np.array(labels)
    y = keras.utils.to_categorical(y, config.n_classes)

    return X, y
```

- **config.py**

可以通过设置`config.load_model`选择加载训练好的模型，还是重新训练模型；设置`config.data_augmentation`是否对样本进行增强处理。

```
# 是否加载已经训练好的模型
load_model = False

# 是否对样本进行增强
data_augmentation = True
```

- **utils.py**

CNN的第一层中：

- 卷积层使用了`32`个卷积核，每个卷积核的大小为`3 * 3`，激活函数为`relu`；
- 池化层，核的大小为`2 * 2`；
- Dropout层，避免过拟合，Dropout率为`0.25`；

```
def build_cnn():
    ...
    # 第一层
    model.add(Conv2D(filters=32, kernel_size=(3, 3), padding='same',
input_shape=input_shape))
    model.add(Activation('relu'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Dropout(0.25))
    ...
```

- **utils.py**

CNN的第二层中：

- 卷积层使用了`64`个卷积核，每个卷积核的大小为`3 * 3`，激活函数为`relu`；
- 池化层，核的大小为`2 * 2`；
- Dropout层，避免过拟合，Dropout率为`0.25`；

```
def build_cnn():
    ...
    # 第二层
    model.add(Conv2D(filters=64, kernel_size=(3, 3), padding='same'))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Dropout(0.25))
    ...
```

- **utils.py**

CNN的最后一层：

- 全连接层，`512`个结点，激活函数为`relu`；
- Dropout层，避免过拟合，Dropout率为`0.5`；
- 全连接层，`10`个结点，激活函数为`softmax`，用于预测值的输出；

```
def build_cnn():
    ...
    # 全连接层
    model.add(Flatten())
    model.add(Dense(512))
    model.add(Activation('relu'))
    model.add(Dropout(0.5))
    model.add(Dense(config.n_classes))
    model.add(Activation('softmax'))
    ...
```

5. 案例总结

- 该项目通过使用深度学习的卷积神经网络（CNN）进行交通标识的分类，包含了如下内容:
    - CNN的基本概念及框架
    - 使用keras搭建CNN用于图像分类

6. 课后练习

- 尝试将激活函数替换成`sigmoid`或`tanh`，观察对结果的影响。
- 尝试去掉Dropout层，观察对结果的影响

参考资料

1. Keras教程
2. Keras CNN案例