1 安装

brine 是一个 pip 包，因此我们可以用下面的命令安装

```
pip install brine-io
```

2 下载数据集

在你的项目坐在目录运行 brine 命令行下载 CIFAR10 数据集。

```
brine install cifar10/train
```

3 载入数据集

用 load_dataset() 函数载入数据集

```
import brine
cifar_train = brine.load_dataset('cifar10/train')
```

用数据集的 columns 属性我们可以查看数据集的结构

```
cifar_train.columns
Column(image=Column(name=image, type=Image), label=Column(name=label, type=
    Category, categories=['dog', 'horse', 'frog', 'airplane', 'cat', 'ship
    ', ...]))
```

返回一个通过列的名字索引的元祖的 Column，因此 label 是一个类别列，我们可以查看列包含的分类，下面我们将分类保存在本地变量中

```
categories = cifar_train.columns.label.categories
categories
['dog', 'horse', 'frog', 'airplane', 'cat', 'ship', 'deer', 'bird', 'truck',
     'automobile']
```

通过 len 检查数据集的长度

```
>> len(cifar_train)
50000
```

从任何行访问数据集，我们简单的传递索引进去。它返回一个命名的元祖，我们可以访问单个的元素

```
>> cifar_train[20]
Row(image='46405_bird.png', label='bird')
>> cifar_train[20].image
'46405_bird.png'
```

```
>> cifar_trian[20].label
'bird'
```

image 返回一个图像路径。用数据集的 load_image() 方法载入图片

```
>> cifar_train.load_image(cifar_train[20].image)
<PIL.PngImagePlugin.PngImageFile image mode=RGB size=32x32 at 0x7F9BA7860D68
    >
```

我们可以用 create_folds() 分割我们的数据集为多个文件夹。这用于创建 train/validation 分割。brine 数据集中的每个文件夹（不占据额外的磁盘空间）因此你可以在原始数据集的文件夹伤执行相同的行为，下面分割 2000 个向本用于我们的验证文件夹，剩下的样本将进入训练文件夹

```
>> validation_fold, training_fold = cifar_train.create_folds([2000],
shuffle=True)
>> len(validation_fold)
2000
>> len(training_fold)
48000
```

4 PyTorch 接口

Brine 提供方法转换你的 Brine 数据集问流行的机器学习兼容数据，像 PyTorch 和 keras 这里是用 PyTorch 的例子

```
from torchvision import transforms
transform = transforms.Compose(
    [transforms.ToTensor(),
    transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))])

train_fold_pytorch = training_fold.to_pytorch(transform=transform,
    transform_columns='images')
validation_fold_pytorch = validation_fold.to_pytorch(transform=transform,
    transform_columns='images')
```

to_pytorch() 方法返回一个 PyTorch 数据集。转化调用用在数据集的每一行，因此我们指定 transform_columns='images', 转化将被用于图像列（这里是 image），图像列被传递前将被转化为 PIL 图像，你可以用这个 PyTorch 数据集正如我们用其它的 PyTorch 数据集一样，例如你可以用它创建一个 DataLoader。trainloader = torch.utils.data.DataLoader(train_fold_pytorch, batch_size=4, shuffle=True, num_workers=2)

5 结论

可以用相同的方法应用通过 Brine 应用到任何数据集，包括分割，分类，多类分类和对象检测，用 Keras 结合 Brine 分割图像的例子查看这篇博客