CSST_photo_dl 说明文档

Github link: https://github.com/xczhou-astro/CSST_photo_z_dl
文章1：Extracting Photometric Redshift from Galaxy Flux and Image Data using Neural Networks in the CSST Survey；
文章2：Photometric redshift estimates using Bayesian neural networks in the CSST survey。

依赖

Python版本：
`python==3.9.20`

依赖库：
`numpy==1.26.0`
`scipy==1.13.1`
`astropy==6.0.1`
`matplotlib==3.9.2`
`tensorflow==2.14.0`
`tensorflow-probability==0.22.1`
`tqdm==4.66.5`

推荐新建环境运行：
`conda create -n photz_csst python=3.9.20`
`conda activate photz_csst`
安装依赖：
`pip install <package_name>`

使用说明

导入类

```
from dataProcess import DataProcess
from photzEstimator import PhotzEstimator
```

DataProcess的构造函数

```
dataloader = DataProcess(data_type, mode='train', augmentation=True,
aug_n=50, batch_size=256)
```

`data_type`：`str`：`'photometry'`，`'image'` 或 `'photometry_and_image'`，数据类型
`mode`：`str`：`'train'`，`'evaluate'` 或 `'inference'`，工作模式

`augmentation`: `bool`, 只在 `mode='train'` 时生效, 是否采用数据增强
`aug_n` : `int`: 只在 `mode='train'` 时生效, num augments for photometry data
`batch_size`: `int`, 批处理数目

DataProcess的类函数

```
dataloader.load_catalogue(catalogue_filename)
```

用于导入星表
`catalogue_filename`: `str`: 星表文件名

```
dataloader.load_photometry(flux_data=None, flux_keys=None,
flux_error_keys=None)
```

用于导入测光数据
`flux_data`: `list`: `[flux, err_flux]`, 流量和流量误差
`flux_keys`: `list`: 星表中7个波段流量的关键字
`flux_error_keys`: `list`: 星表中7个波段流量误差的关键字
注意:
`flux_data`会覆盖 `flux_data` 和 `flux_error_keys`

```
dataloader.load_images(images=None, imgnames=None)
```

用于导入图像数据
`images`: `numpy.array`: 以`(num, 32, 32, 7)`为大小的图像数据
`imgnames`: `list`: 以FITS为格式的图像文件名
注意:

1. `images` 会覆盖 `imgnames`
2. 如果提供文件名，图像会标准化为以`(32, 32, 7)`为大小，更耗时。

```
dataloader.load_specz(specz=None, specz_key=None)
```

用于导入光谱红移
`specz`: `numpy.array`: 以 `(num,)` 为大小的光谱红移数据
`specz_key`: `str`: 星表中光谱红移的关键字
注意:

1. `specz` 会覆盖 `specz_key`
2. 只在`mode=='train'` 或 `'evaluate'`调用此函数

如果想在训练时实时监视测试数据的精度:

```
dataloader.load_test_catalogue(test_catalogue_filename)
dataloader.load_test_images(images=None, imgnames=None)
dataloader.load_test_photometry(flux_data=None, flux_keys=None,
flux_error_keys=None)
dataloader.load_test_specz(specz=None, specz_key=None)
```

```
loaded_data = dataloader.get_dataset()
```

获得TensorFlow数据集和其他信息
`tfds, datasize = loaded_data` 如果 `mode='train'` 或 `'inference'`
`tfds, tfds_specz = loaded_data` 如果 `mode='evaluate'`

```
loaded_test_data = dataloader.get_test_dataset()
```

获得测试用TensorFlow数据集和其他信息
`test_tfds, test_tfds_specz = loaded_data` 如果`mode='train'`

PhotzEstimator的构造函数

```
estimator = PhotzEstimator(model_type, data_type, transfer=False,
outDir='outputs')
```

`model_type`: `str`: `'NN'` 或 `'BNN'`, 模型类型
`data_type`: `str`: `'photometry'`, `'image'` 或 `'photometry_and_image'`, 输入数据类型
`transfer`: `bool`: 只在 `datatype='photometry_and_image'` 时生效, 是否使用迁移学习
`outDir`: `bool`: 输出文件夹

```
estimator.get_model(datasize=50000, weights=None, cnn_weights=None,
mlp_weights=None, alpha_file=None)
```

获得模型
`datasize`: `int`: 载入的数据数目，当`model_type='BNN'`时必须给出
`weights`: `str`: 以`.h5`为格式的权重文件
`cnn_weights`: `str`: 以`.h5`为格式的CNN权重文件，只在
`data_type='photometry_and_image'`和`transfer=True`时需要给出
`mlp_weights`: `str`: 以`.h5`为格式的MLP权重文件，只在
`data_type='photometry_and_image'`和`transfer=True`时需要给出
`alpha_file`: `str`: 修正参数的文件，只在`model_type='BNN'` 和 `mode='inference'`时需要给出

```
estimator.train(train_ds, test_ds=None, learning_rate=2e-4, epochs=200)
```

训练模型，完成后在输出文件夹给出模型权重文件
`train_ds`：`tf.data.Dataset`：训练用TensorFlow数据集
`test_ds`：`tf.data.Dataset`：测试用TensorFlow数据集
`learning_rate`：`float`：Adam优化器的学习率
`epochs`：`int`：网络迭代次数

```
estimator.evaluate(ds, ds_specz, n_runs=200)
```

评估模型，完成后在输出文件夹给出结果图和包含红移估计的`.npz`数据集
`ds`：`tf.data.Dataset`：评估用TensorFlow数据集
`ds_specz`：`tf.data.Dataset`：评估用光谱红移TensorFlow数据集
`n_runs`：`int`：评估用数据`ds`输入BNN模型的次数

```
estimator.inference(ds, datasize, catalogue=None, info_keys=['ra', 'dec'],
n_runs=200)
```

用模型预测测光红移，完成后在输出文件夹给出测光红移星表
`ds`：`tf.data.Dataset`：预测用TensorFlow数据集
`datasize`：`int`：数据集`ds`源的数目
`catalogue`：`str`：星表文件名
`info_keys`：`list`：星表关键字
`n_runs`：`int`：数据`ds`输入BNN模型的次数
注意：
如果catalogue没有给定，则输出的星表只包括测光红移预测；如果给定，则输出的星表还包含`info_keys`中关键字的信息

示例

1.如果想训练针对测光和图像数据的BNN模型，并检查对测试数据的结果：

```
from dataProcess import DataProcess
from photzEstimator import PhotzEstimator

dataloader = DataProcess(data_type='photometry_and_image', mode='train',
augmentation=True, aug_n=50, batch_size=1024)
dataloader.load_catalogue('catalogue_filename')

bands = ['NUV', 'u', 'g', 'r', 'i', 'z', 'y']
flux_keys = [f'flux_{bd}' for bd in bands]
flux_error_keys = [f'fluxerr_{bd}' for bd in bands]
dataloader.load_photometry(flux_keys=flux_keys,
flux_error_keys=flux_error_keys)
```

```
imgnames = ['stamp_000.fits', 'stamp_001.fits', 'stamp_002.fits' '..']
dataloader.load_images(imgnames=imgnames)

dataloader.load_specz(specz_key=['zspec'])

dataloader.load_test_catalogue('test_catalogue_filename')

images = np.load('image_arrays.npy')
dataloader.load_test_images(images=images)

fluxes = np.load('flux_data.npy')
err_fluxes = np.load('err_flux_data.npy')
dataloader.load_test_photometry(flux_data=[fluxes, err_fluxes])

specz = np.loadtxt('zspec.txt')
dataloader.load_test_specz(specz=specz)

loaded_data = dataloader.get_dataset()
tfds, datasize = loaded_data

loaded_test_data = dataloader.get_test_dataset()
test_tfds, test_tfds_specz = loaded_test_data

estimator = PhotzEstimator(model_type='BNN',
data_type='photometry_and_image', transfer=False, outDir='outputs')

estimator.get_model(datasize=datasize)

estimator.train(train_ds=tfds, test_ds=test_tfds, learning_rate=2e-4,
epochs=200)

estimator.evaluate(test_tfds, test_tfds_specz, n_runs=200)
```

数据产品：
权重文件 `outputs/BNN_models/Hybrid/Hybrid_weights.h5`
修正参数文件 `outputs/BNN_models/alpha.json`
图 `outputs/BNN_models/Hybrid/loss.png`
图 `outputs/BNN_models/Hybrid/acc.png`
图 `outputs/BNN_models/Hybrid/results.png`

2.如果想用给定权重的BNN模型从图像数据中估计红移：

```
from dataProcess import DataProcess
from photzEstimator import PhotzEstimator

dataloader = DataProcess(data_type='image', mode='inference',
batch_size=1024)
```

```
dataloader.load_catalogue('catalogue_filename')


imgnames = ['stamp_000.fits', 'stamp_001.fits', 'stamp_002.fits' '..']
dataloader.load_images(imgnames=imgnames)

loaded_data = dataloader.get_dataset()
tfds, datasize = loaded_data

estimator = PhotzEstimator(model_type='BNN', data_type='image',
transfer=False, outDir='outputs')

estimator.get_model(weights='Data/BNN/CNN_BNN_weights.h5',
alpha_file='Data/BNN/alpha.json')

estimator.inference(ds=tfds, datasize=datasize,
catalogue=dataloader.catalogue_filename, info_keys=['ra', 'dec'],
n_runs=200)
```

数据产品：
包含 `ra, dec, z_pred, z_err` 的星表 `outputs/BNN_models/CNN/photoz_catalogue.fits`.