Folder Data

Contents

Custom Folder Dataset.

This script creates a custom dataset from a folder.

class **anomalib.data.image.folder.Folder**(***name***, ***normal_dir***, ***root***=*None*, ***abnormal_dir***=*None*, ***normal_test_dir***=*None*, ***mask_dir***=*None*, ***normal_split_ratio***=*0.2*, ***extensions***=*None*, ***train_batch_size***=*32*, ***eval_batch_size***=*32*, ***num_workers***=*8*, ***task***=*TaskType.SEGMENTATION*, ***image_size***=*None*, ***transform***=*None*, ***train_transform***=*None*, ***eval_transform***=*None*, ***test_split_mode***=*TestSplitMode.FROM_DIR*, ***test_split_ratio***=*0.2*, ***val_split_mode***=*ValSplitMode.FROM_TEST*, ***val_split_ratio***=*0.5*, ***seed***=*None*)

Bases: `AnomalibDataModule`

Folder DataModule.

Parameters:

- **name** (*str*) – Name of the dataset. This is used to name the datamodule, especially when logging/saving.
- **normal_dir** (*str | Path | Sequence*) – Name of the directory containing normal images.
- **root** (*str | Path | None*) – Path to the root folder containing normal and abnormal dirs. Defaults to `None`.
- **abnormal_dir** (*str | Path | None | Sequence*) – Name of the directory containing abnormal images. Defaults to `None`.
- **normal_test_dir** (*str | Path | Sequence | None, optional*) – Path to the directory containing normal images for the test dataset. Defaults to `None`.
- **mask_dir** (*str | Path | Sequence | None, optional*) – Path to the directory containing the mask annotations. Defaults to `None`.
- **normal_split_ratio** (*float, optional*) – Ratio to split normal training images and add to the test set in case test set doesn't contain any normal images. Defaults to 0.2.
- **extensions** (*tuple[str, ...] | None, optional*) – Type of the image extensions to read from the directory. Defaults to `None`.
- **train_batch_size** (*int, optional*) – Training batch size. Defaults to `32`.
- **eval_batch_size** (*int, optional*) – Validation, test and predict batch size. Defaults to `32`.
- **num_workers** (*int, optional*) – Number of workers. Defaults to `8`.
- **task** (*TaskType, optional*) – Task type. Could be `classification`, `detection` or `segmentation`. Defaults to `segmentation`.
- **image_size** (*tuple[int, int], optional*) – Size to which input images should be resized. Defaults to `None`.
- **transform** (*Transform, optional*) – Transforms that should be applied to the input images. Defaults to `None`.
- **train_transform** (*Transform, optional*) – Transforms that should be applied to the input images during training. Defaults to `None`.
- **eval_transform** (*Transform, optional*) – Transforms that should be applied to the input images during evaluation. Defaults to `None`.
- **test_split_mode** (*TestSplitMode*) – Setting that determines how the testing subset is obtained. Defaults to `TestSplitMode.FROM_DIR`.
- **test_split_ratio** (*float*) – Fraction of images from the train set that will be

- **test_split_ratio** (*float*) – Fraction of images from the train set that will be reserved for testing. Defaults to `0.2`.
- **val_split_mode** (*ValSplitMode*) – Setting that determines how the validation subset is obtained. Defaults to `ValSplitMode.FROM_TEST`.
- **val_split_ratio** (*float*) – Fraction of train or test images that will be reserved for validation. Defaults to `0.5`.
- **seed** (*int | None, optional*) – Seed used during random subset splitting. Defaults to `None`.

Examples

The following code demonstrates how to use the `Folder` datamodule. Assume that the dataset is structured as follows:

```
$ tree sample_dataset
sample_dataset
├── colour
│   ├── 00.jpg
│   ├── ...
│   └── x.jpg
├── crack
│   ├── 00.jpg
│   ├── ...
│   └── y.jpg
├── good
│   ├── ...
│   └── z.jpg
├── LICENSE
└── mask
    ├── colour
    │   ├── ...
    │   └── x.jpg
    └── crack
        ├── ...
        └── y.jpg
```

```
folder_datamodule = Folder(
    root=dataset_root,
    normal_dir="good",
    abnormal_dir="crack",
    task=TaskType.SEGMENTATION,
    mask_dir=dataset_root / "mask" / "crack",
    image_size=256,
    normalization=InputNormalizationMethod.NONE,
)
folder_datamodule.setup()
```

To access the training images,

```
>> i, data = next(enumerate(folder_datamodule.train_dataloader()))
>> print(data.keys(), data["image"].shape)
```

To access the test images,

```
>> i, data = next(enumerate(folder_datamodule.test_dataloader()))
>> print(data.keys(), data["image"].shape)
```

property* name: *str

Name of the datamodule.

Folder datamodule overrides the name property to provide a custom name.

***class* anomalib.data.image.folder.FolderDataset(*name*, *task*, *normal_dir*, *transform=None*, *root=None*, *abnormal_dir=None*, *normal_test_dir=None*, *mask_dir=None*, *split=None*, *extensions=None*)**

Bases: `AnomalibDataset`

Folder dataset.

This class is used to create a dataset from a folder. The class utilizes the Torch Dataset class.

Parameters:

- **name** (*str*) – Name of the dataset. This is used to name the datamodule, especially when logging/saving.
- **task** (*TaskType*) – Task type. (`classification`, `detection` or `segmentation`).
- **transform** (*Transform, optional*) – Transforms that should be applied to the input images. Defaults to `None`.
- **normal_dir** (*str | Path | Sequence*) – Path to the directory containing normal images.
- **root** (*str | Path | None*) – Root folder of the dataset. Defaults to `None`.
- **abnormal_dir** (*str | Path | Sequence | None, optional*) – Path to the directory containing abnormal images. Defaults to `None`.
- **normal_test_dir** (*str | Path | Sequence | None, optional*) – Path to the directory containing normal images for the test dataset. Defaults to `None`.
- **mask_dir** (*str | Path | Sequence | None, optional*) – Path to the directory containing the mask annotations. Defaults to `None`.
- **split** (*str | Split | None*) – Fixed subset split that follows from folder structure on file system. Choose from [Split.FULL, Split.TRAIN, Split.TEST] Defaults to `None`.
- **extensions** (*tuple[str, ...] | None, optional*) – Type of the image extensions to read from the directory. Defaults to `None`.

Raises:

ValueError – When task is set to classification and *mask_dir* is provided. When *mask_dir* is provided, *task* should be set to *segmentation*.

Examples

Assume that we would like to use this `FolderDataset` to create a dataset from a folder for a classification task. We could first create the transforms,

```
>>> from anomalib.data.utils import InputNormalizationMethod, get_transforms
>>> transform = get_transforms(image_size=256, normalization=InputNormalization
```

We could then create the dataset as follows,

```
folder_dataset_classification_train = FolderDataset(
    normal_dir=dataset_root / "good",
    abnormal_dir=dataset_root / "crack",
```

```
    split="train",
    transform=transform,
    task=TaskType.CLASSIFICATION,
)
```

property **name**: *str*

Name of the dataset.

Folder dataset overrides the name property to provide a custom name.

anomalib.data.image.folder.make_folder_dataset(*normal_dir*, *root=None*, *abnormal_dir=None*, *normal_test_dir=None*, *mask_dir=None*, *split=None*, *extensions=None*)

Make Folder Dataset.

Parameters:

- **normal_dir** (*str* | *Path* | *Sequence*) – Path to the directory containing normal images.
- **root** (*str* | *Path* | *None*) – Path to the root directory of the dataset. Defaults to `None`.
- **abnormal_dir** (*str* | *Path* | *Sequence* | *None, optional*) – Path to the directory containing abnormal images. Defaults to `None`.
- **normal_test_dir** (*str* | *Path* | *Sequence* | *None, optional*) – Path to the directory containing normal images for the test dataset. Normal test images will be a split of *normal_dir* if *None*. Defaults to `None`.
- **mask_dir** (*str* | *Path* | *Sequence* | *None, optional*) – Path to the directory containing the mask annotations. Defaults to `None`.
- **split** (*str* | *Split* | *None, optional*) – Dataset split (ie., Split.FULL, Split.TRAIN or Split.TEST). Defaults to `None`.
- **extensions** (*tuple[str, ...]* | *None, optional*) – Type of the image extensions to read from the directory. Defaults to `None`.

Returns:

an output dataframe containing samples for the requested split (ie., train or test).

Return type:

DataFrame

Examples

Assume that we would like to use this `make_folder_dataset` to create a dataset from a folder. We could then create the dataset as follows,

```
folder_df = make_folder_dataset(
    normal_dir=dataset_root / "good",
    abnormal_dir=dataset_root / "crack",
    split="train",
)
folder_df.head()
```

	image_path	label	label_index	mask_path	split
0	./toy/good/00.jpg	DirType.NORMAL	0		Split.TRAIN
1	./toy/good/01.jpg	DirType.NORMAL	0		Split.TRAIN
2	./toy/good/02.jpg	DirType.NORMAL	0		Split.TRAIN
3	./toy/good/03.jpg	DirType.NORMAL	0		Split.TRAIN
4	./toy/good/04.jpg	DirType.NORMAL	0		Split.TRAIN

Previous
BTech Data

Next
Kolektor Data