```python
import torch.utils.data as data
from torch import Tensor
from os import listdir
from os.path import join
import numpy as np
import h5py


def is_image_file(filename):
    return any(filename.endswith(extension) for extension in [".hdf5", ".h5"])

def load_img(filepath):
    img = None
    with h5py.File(filepath, "r") as f:
        # breakpoint()
        try:
            img = f['data'][()]
        except:
            breakpoint()
    torch_img = Tensor(img)
    return torch_img


def collate_img(filepath):
    img = None
    data_list = []
    with h5py.File(filepath, "r") as f:
        try:
            img = f['data'][()]
        except:
            breakpoint()

        target = f['target'][()]
    return img, target

class HDF5Dataset(data.Dataset):
    def __init__(self, image_dir, input_transform=None, target_transform=None):
        super(HDF5Dataset, self).__init__()
        self.image_filenames = [join(image_dir, x) for x in listdir(image_dir) if is_image_file(x)]
        self.input_transform = input_transform
        self.target_transform = target_transform

    def __getitem__(self, index):

        data = load_img(self.image_filenames[index])
        return data
    def __getnumpyitem__(self, index):
        input, target = collate_img(self.image_filenames[index])
        return input, target
    def __len__(self):
        return len(self.image_filenames)

```