

## Loading Image Data:

dataset = datasets.ImageFolder('path',  
transform = transforms)

image folder structure:

root/dog/n.png

root/dog/~.png  
:

root/cat/n.png

root/cat/~.png  
:

one folder  
for each  
class

## Transforms:

transforms = transforms.Compose([  
transforms.Resize(255),  
transforms.CenterCrop(224),  
transforms.ToTensor()])

Converting to tensor



## Data Loaders:

```
data_loader = torch.utils.data.DataLoader  
              (dataset, batch_size=32,  
               shuffle=True)
```

```
next(iter(data_loader))
```

The diagram illustrates the relationship between the components in the code snippet above. A blue bracket under the argument `data_loader` is labeled "generator". Another blue bracket under the entire expression `iter(data_loader)` is labeled "iterator".

## Data Augmentation:

Introducing randomness in input data itself.

transforms  $\leftarrow$  RandomRotation, RandomResizedCrop, etc.

## Normalization:

transforms.Normalize

$$\text{input}[\text{channel}] = (\text{input}[\text{channel}] - \text{mean}[\text{channel}]) / \text{std}[\text{channel}]$$

Helps keep data near zero, making backpropagation more stable.