
PixelCNN MNIST

Alex Pierron & Ambre Adjevi-Neglokpe

Mar 15, 2023

CONTENTS:

1	MNIST	1
1.1	MaskedConv module	1
1.2	PixelCNN_MNIST_model module	2
1.3	application module	2
1.4	test_MNIST module	2
1.5	training_loop module	3
2	Indices and tables	5
	Python Module Index	7
	Index	9

1.1 MaskedConv module

class MaskedConv.MaskedConv2D(*mask_type*, *args, **kwargs)

Bases: Conv2d

This class creates a 2d masked convolutions with a type A or type B mask

Parameters

mask_type (*string*) – the type of the mask (‘A’ or ‘B’)

bias: Optional[Tensor]

dilation: Tuple[int, ...]

forward(*x*)

Forward function for the masked convolution

groups: int

kernel_size: Tuple[int, ...]

out_channels: int

output_padding: Tuple[int, ...]

padding: Union[str, Tuple[int, ...]]

padding_mode: str

stride: Tuple[int, ...]

transposed: bool

weight: Tensor

1.2 PixelCNN_MNIST_model module

```
class PixelCNN_MNIST_model.PixelCNN_MNIST(in_channels=1, out_channels=1, nb_layer_block=12,  
                                           h_channels=32, device=None)
```

Bases: Module

Defines the architecture of the PixelCNN algorithm for the MNIST database

Parameters

- **in_channel** (*int*) – number of channel in the input images (set to 1 for MNIST images)
- **out_channel** (*int*) – number of channel in the output images (set to 1 for MNIST images)
- **nb_layer_block** (*int*) – number of layer of residual block used in the network
- **h_channels** (*int*) – number of feature map to use during the convolutions in the processing. This number will often be doubled during calculation.

```
forward(x, **kwargs)
```

Defines the computation performed at every call.

Should be overridden by all subclasses.

Note: Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

```
residual_block(x)
```

Defines the residual connection in the residual blocks :param tensor x: tensor we are processing through the residual block

Returns

the sum of x and what the residual block calculated for x

Return type

tensor

```
training: bool
```

1.3 application module

1.4 test_MNIST module

```
test_MNIST.test_MNIST(model, weight_path, no_images, n_col, n_row, user=True)
```

Function used to generate new images with a trained network.

Parameters

- **dataloader** (*tensor*) – data that will be used to train the network
- **model** (*torch*) – the architecture of the model we use
- **weight_path** (*path*) – relative path to the weight for the desired network to use
- **no_images** (*int*) – number of images to generate

- **n_col** (*int*) – the number of column to have in the final global image (merge of all the sub-images)
- **n_row** (*int*) – the number of row to have in the final global image (merge of all the sub-images)
- **user** (*bool*) – if the network used is custom made (True) or not (False)

Returns

stocks the generated image in `pages/generated_images` as a png file

1.5 training_loop module

`training_loop.train_loop(data_loader, model, loss_fn, optimizer, best_loss)`

Function used to train a network. Performs one epoch.

Parameters

- **data_loader** – data that will be used to train the network.
- **model** – the architecture for the model to be trained.
- **loss_fn** – the loss function chosen for the training.
- **optimizer** – the optimizer chosen for the training.
- **best_loss** – the minimal loss recorded during training.

Returns

two elements : the mean loss during the whole epoch and the minimum found

Return type

tuple of two elements

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

a

`application`, 2

m

`MaskedConv`, 1

p

`PixelCNN_MNIST_model`, 2

t

`test_MNIST`, 2

`training_loop`, 3

INDEX

A

application
 module, 2

B

bias (*MaskedConv.MaskedConv2D* attribute), 1

D

dilation (*MaskedConv.MaskedConv2D* attribute), 1

F

forward() (*MaskedConv.MaskedConv2D* method), 1
forward() (*PixelCNN_MNIST_model.PixelCNN_MNIST*
 method), 2

G

groups (*MaskedConv.MaskedConv2D* attribute), 1

K

kernel_size (*MaskedConv.MaskedConv2D* attribute),
 1

M

MaskedConv
 module, 1

MaskedConv2D (*class in MaskedConv*), 1
module

 application, 2
 MaskedConv, 1
 PixelCNN_MNIST_model, 2
 test_MNIST, 2
 training_loop, 3

O

out_channels (*MaskedConv.MaskedConv2D* attribute),
 1

output_padding (*MaskedConv.MaskedConv2D* at-
 tribute), 1

P

padding (*MaskedConv.MaskedConv2D* attribute), 1

padding_mode (*MaskedConv.MaskedConv2D* attribute),
 1

PixelCNN_MNIST (*class in PixelCNN_MNIST_model*), 2
PixelCNN_MNIST_model
 module, 2

R

residual_block() (*Pixel-*
 CNN_MNIST_model.PixelCNN_MNIST
 method), 2

S

stride (*MaskedConv.MaskedConv2D* attribute), 1

T

test_MNIST
 module, 2

test_MNIST() (*in module test_MNIST*), 2

train_loop() (*in module training_loop*), 3

training (*PixelCNN_MNIST_model.PixelCNN_MNIST*
 attribute), 2

training_loop
 module, 3

transposed (*MaskedConv.MaskedConv2D* attribute), 1

W

weight (*MaskedConv.MaskedConv2D* attribute), 1