O PyTorch

Q Search Docs

Notes [+]

Language Bindings [+]

Python API [-]

torch

torch.nn

torch.nn.functional

torch.Tensor

Tensor Attributes

Tensor Views

torch.autograd

torch.cuda

torch.cuda.amp

torch.backends

torch.distributed

torch.distributed.algorithms.join

torch.distributed.elastic

torch.distributed.fsdp

torch.distributed.optim

torch.distributions

torch.fft

torch.futures torch.fx

torch.hub

torch.jit

torch.linalg

torch.special

torch.monitor

torch.overrides

torch.package

torch.profiler

torch.nn.init

torch.onnx torch.optim

Get Started

Docs > torch.nn > BatchNorm2d

BATCHNORM2D

Ecosystem

track\_running\_stats=True, device=None, dtype=None) [SOURCE]

CLASS torch.nn.BatchNorm2d(num\_features, eps=1e-05, momentum=0.1, affine=True,

Mobile

Blog

Applies Batch Normalization over a 4D input (a mini-batch of 2D inputs with additional channel dimension) as described in the paper Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift .

**Tutorials** 

Docs 🗸

GitHub

>\_

Resources ~

$$y = rac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + eta$$

The mean and standard-deviation are calculated per-dimension over the mini-batches and  $\gamma$  and eta are learnable parameter vectors of size C (where C is the input size). By default, the elements of  $\gamma$  are set to 1 and the elements of etaare set to 0. The standard-deviation is calculated via the biased estimator, equivalent to torch.var(input, unbiased=False).

Also by default, during training this layer keeps running estimates of its computed mean and variance, which are then used for normalization during evaluation. The running estimates are kept with a default momentum of 0.1.

If track\_running\_stats is set to False, this layer then does not keep running estimates, and batch statistics are instead used during evaluation time as well.

• NOTE

This momentum argument is different from one used in optimizer classes and the conventional notion of momentum. Mathematically, the update rule for running statistics here is  $\hat{x}_{
m new} = (1-{
m momentum}) imes \hat{x} + 1$  $\mathrm{momentum} imes x_t$ , where  $\hat{x}$  is the estimated statistic and  $x_t$  is the new observed value.

Because the Batch Normalization is done over the C dimension, computing statistics on (N, H, W) slices, it's common terminology to call this Spatial Batch Normalization.

Parameters

- ullet num\_features C from an expected input of size (N,C,H,W)
- eps a value added to the denominator for numerical stability. Default: 1e-5
- momentum the value used for the running\_mean and running\_var computation. Can be set to None for cumulative moving average (i.e. simple average). Default: 0.1
- affine a boolean value that when set to True, this module has learnable affine parameters. Default: True
- track\_running\_stats a boolean value that when set to True, this module tracks the running mean and variance, and when set to False, this module does not track such statistics, and initializes statistics buffers running\_mean and running\_var as None. When these buffers are None, this module always uses batch statistics. in both training and eval modes. Default: True

Shape:

- Input: (N,C,H,W)
- ullet Output: (N,C,H,W) (same shape as input)

**Examples:** 

```
>>> # With Learnable Parameters
      = nn.BatchNorm2d(100)
>>> # Without Learnable Parameters
>>> m = nn.BatchNorm2d(100, affine=False)
>>> input = torch.randn(20, 100, 35, 45)
>>> output = m(input)
```

Previous Next >

© Copyright 2019, Torch Contributors.

Built with Sphinx using a theme provided by Read the Docs.

**Tutorials** Docs Resources

Access comprehensive developer documentation for

PyTorch

View Docs >

Get in-depth tutorials for beginners and advanced developers

View Tutorials >

**Features** 

Find development resources and get your questions answered

View Resources >

PyTorch Resources

Get Started **Tutorials** Docs

Ecosystem Discuss Github Issues Blog

Contributing **Brand Guidelines** 



Stay Connected

Email Address

 $\rightarrow$