Named Tensors

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 1)

Unknown directive type "currentmodule".

```
.. currentmodule:: torch
```

Named Tensors allow users to give explicit names to tensor dimensions. In most cases, operations that take dimension parameters will accept dimension names, avoiding the need to track dimensions by position. In addition, named tensors use names to automatically check that APIs are being used correctly at runtime, providing extra safety. Names can also be used to rearrange dimensions, for example, to support "broadcasting by name" rather than "broadcasting by position".

Warning

The named tensor API is a prototype feature and subject to change.

Creating named tensors

Factory functions now take a new :attr:`names` argument that associates a name with each dimension.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 23); *backlink*

Unknown interpreted text role "attr".

```
>>> torch.zeros(2, 3, names=('N', 'C'))
tensor([[0., 0., 0.],
        [0., 0., 0.]], names=('N', 'C'))
```

Named dimensions, like regular Tensor dimensions, are ordered. `tensor.names[i]` is the name of dimension `i` of `tensor`.

The following factory functions support named tensors:

- :func:`torch.empty`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 37); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.rand`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 38); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.randn`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 39); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.ones`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 40); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.tensor`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 41); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.zeros`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 42); *backlink*

  Unknown interpreted text role "func".

Named dimensions

See :attr:`~Tensor.names` for restrictions on tensor names.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 47); *backlink*

Unknown interpreted text role "attr".

Use :attr:`~Tensor.names` to access the dimension names of a tensor and :meth:`~Tensor.rename` to rename named dimensions.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 49); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 49); *backlink*

Unknown interpreted text role "meth".

```
>>> imgs = torch.randn(1, 2, 2, 3 , names=('N', 'C', 'H', 'W'))
>>> imgs.names
('N', 'C', 'H', 'W')

>>> renamed_imgs = imgs.rename(H='height', W='width')
>>> renamed_imgs.names
('N', 'C', 'height', 'width)
```

Named tensors can coexist with unnamed tensors; named tensors are instances of :class:`torch.Tensor`. Unnamed tensors have `None`-named dimensions. Named tensors do not require all dimensions to be named.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 63); *backlink*

Unknown interpreted text role "class".

```
>>> imgs = torch.randn(1, 2, 2, 3 , names=(None, 'C', 'H', 'W'))
>>> imgs.names
(None, 'C', 'H', 'W')
```

Name propagation semantics

Named tensors use names to automatically check that APIs are being called correctly at runtime. This occurs in a process called *name inference*. More formally, name inference consists of the following two steps:

- **Check names**: an operator may perform automatic checks at runtime that check that certain dimension names must match.
- **Propagate names**: name inference propagates names to output tensors.

All operations that support named tensors propagate names.

```
>>> x = torch.randn(3, 3, names=('N', 'C'))
>>> x.abs().names
('N', 'C')
```

match semantics

Two names *match* if they are equal (string equality) or if at least one is `None`. Nones are essentially a special "wildcard" name.

`unify(A, B)` determines which of the names `A` and `B` to propagate to the outputs. It returns the more *specific* of the two names, if they match. If the names do not match, then it errors.

Note

In practice, when working with named tensors, one should avoid having unnamed dimensions because their handling can be complicated. It is recommended to lift all unnamed dimensions to be named dimensions by using :meth:`~Tensor.refine_names`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 106); *backlink*

Unknown interpreted text role "meth".

Basic name inference rules

Let's see how `match` and `unify` are used in name inference in the case of adding two one-dim tensors with no broadcasting.

```
x = torch.randn(3, names=('X',))
y = torch.randn(3)
z = torch.randn(3, names=('Z',))
```

Check names: check that the names of the two tensors *match*.

For the following examples:

```
>>> # x + y  # match('X', None) is True
>>> # x + z  # match('X', 'Z') is False
>>> # x + x  # match('X', 'X') is True

>>> x + z
Error when attempting to broadcast dims ['X'] and dims ['Z']: dim 'X' and dim 'Z' are at the same position from the right
```

Propagate names: *unify* the names to select which one to propagate. In the case of `x + y`, `unify('X', None) = 'X'` because `'X'` is more specific than `None`.

```
>>> (x + y).names
('X',)
>>> (x + x).names
('X',)
```

For a comprehensive list of name inference rules, see :ref:`name_inference_reference-doc`. Here are two common operations that may be useful to go over:

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 147); *backlink***
>
> Unknown interpreted text role "ref".

- Binary arithmetic ops: :ref:`unifies_names_from_inputs-doc`

  > **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 150); *backlink***
  >
  > Unknown interpreted text role "ref".

- Matrix multiplication ops: :ref:`contracts_away_dims-doc`

  > **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 151); *backlink***
  >
  > Unknown interpreted text role "ref".

Explicit alignment by names

Use :meth:`~Tensor.align_as` or :meth:`~Tensor.align_to` to align tensor dimensions by name to a specified ordering. This is useful for performing "broadcasting by names".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 156); *backlink***
>
> Unknown interpreted text role "meth".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 156); *backlink***
>
> Unknown interpreted text role "meth".

```
# This function is agnostic to the dimension ordering of `input`,
# as long as it has a `C` dimension somewhere.
def scale_channels(input, scale):
    scale = scale.refine_names('C')
    return input * scale.align_as(input)

>>> num_channels = 3
>>> scale = torch.randn(num_channels, names=('C',))
>>> imgs = torch.rand(3, 3, 3, num_channels, names=('N', 'H', 'W', 'C'))
>>> more_imgs = torch.rand(3, num_channels, 3, 3, names=('N', 'C', 'H', 'W'))
>>> videos = torch.randn(3, num_channels, 3, 3, 3, names=('N', 'C', 'H', 'W', 'D')

>>> scale_channels(imgs, scale)
>>> scale_channels(more_imgs, scale)
>>> scale_channels(videos, scale)
```

Manipulating dimensions

Use :meth:`~Tensor.align_to` to permute large amounts of dimensions without mentioning all of them as in required by :meth:`~Tensor.permute`.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 180); *backlink***
>
> Unknown interpreted text role "meth".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 180); *backlink***
>
> Unknown interpreted text role "meth".

```
>>> tensor = torch.randn(2, 2, 2, 2, 2, 2)
>>> named_tensor = tensor.refine_names('A', 'B', 'C', 'D', 'E', 'F')

# Move the F (dim 5) and E dimension (dim 4) to the front while keeping
# the rest in the same order
>>> tensor.permute(5, 4, 0, 1, 2, 3)
>>> named_tensor.align_to('F', 'E', ...)
```

Use :meth:`~Tensor.flatten` and :meth:`~Tensor.unflatten` to flatten and unflatten dimensions, respectively. These methods are more verbose than :meth:`~Tensor.view` and :meth:`~Tensor.reshape`, but have more semantic meaning to someone reading the code.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 193); *backlink***
>
> Unknown interpreted text role "meth".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 193); *backlink***
>
> Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 193); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 193); *backlink*

Unknown interpreted text role "meth".

```
>>> imgs = torch.randn(32, 3, 128, 128)
>>> named_imgs = imgs.refine_names('N', 'C', 'H', 'W')

>>> flat_imgs = imgs.view(32, -1)
>>> named_flat_imgs = named_imgs.flatten(['C', 'H', 'W'], 'features')
>>> named_flat_imgs.names
('N', 'features')

>>> unflattened_imgs = imgs.view(32, 3, 128, 128)
>>> unflattened_named_imgs = named_flat_imgs.unflatten(
        'features', [('C', 3), ('H', 128), ('W', 128)])
```

Autograd support

Autograd currently supports named tensors in a limited manner: autograd ignores names on all tensors. Gradient computation is still correct but we lose the safety that names give us.

```
>>> x = torch.randn(3, names=('D',))
>>> weight = torch.randn(3, names=('D',), requires_grad=True)
>>> loss = (x - weight).abs()
>>> grad_loss = torch.randn(3)
>>> loss.backward(grad_loss)
>>> weight.grad  # Unnamed for now. Will be named in the future
tensor([-1.8107, -0.6357,  0.0783])

>>> weight.grad.zero_()
>>> grad_loss = grad_loss.refine_names('C')
>>> loss = (x - weight).abs()
# Ideally we'd check that the names of loss and grad_loss match but we don't yet.
>>> loss.backward(grad_loss)
>>> weight.grad
tensor([-1.8107, -0.6357,  0.0783])
```

Currently supported operations and subsystems

Operators

See :ref:`name_inference_reference-doc` for a full list of the supported torch and tensor operations. We do not yet support the following that is not covered by the link:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 244); *backlink*

Unknown interpreted text role "ref".

- indexing, advanced indexing.

For `torch.nn.functional` operators, we support the following:

- :func:`torch.nn.functional.relu`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 251); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.nn.functional.softmax`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 252); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.nn.functional.log_softmax`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 253); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.nn.functional.tanh`

  System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 254); *backlink*

  Unknown interpreted text role "func".

- :func:`torch.nn.functional.sigmoid`

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 255); *backlink*

Unknown interpreted text role "func".

- :func:`torch.nn.functional.dropout`

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 256); *backlink*

Unknown interpreted text role "func".

Subsystems

Autograd is supported, see :ref:`named_tensors_autograd-doc`. Because gradients are currently unnamed, optimizers may work but are untested.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 261); *backlink*

Unknown interpreted text role "ref".

NN modules are currently unsupported. This can lead to the following when calling modules with named tensor inputs:

- NN module parameters are unnamed, so outputs may be partially named.
- NN module forward passes have code that don't support named tensors and will error out appropriately.

We also do not support the following subsystems, though some may work out of the box:

- distributions
- serialization (:func:`torch.load`, :func:`torch.save`)

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 275); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 275); *backlink*

Unknown interpreted text role "func".

- multiprocessing
- JIT
- distributed
- ONNX

If any of these would help your use case, please search if an issue has already been filed and if not, file one.

Named tensor API reference

In this section please find the documentation for named tensor specific APIs. For a comprehensive reference for how names are propagated through other PyTorch operators, see :ref:`name_inference_reference-doc`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 288); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 295)

Unknown directive type "autoattribute".

```
.. autoattribute:: names
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 296)

Unknown directive type "automethod".

```
.. automethod:: rename
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 297)

Unknown directive type "automethod".

```
.. automethod:: rename_
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 298)

Unknown directive type "automethod".

```
.. automethod:: refine_names
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 300)

Unknown directive type "automethod".

```
.. automethod:: align_as
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 301)

Unknown directive type "automethod".

```
.. automethod:: align_to
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 303)

Unknown directive type "automethod".

```
.. automethod:: unflatten
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\pytorch-master\docs\source\(pytorch-master)(docs)(source)named_tensor.rst, line 304)

Unknown directive type "py:method".

```
.. py:method:: flatten(dims, out_dim) -> Tensor
   :noindex:

   Flattens :attr:`dims` into a single dimension with name :attr:`out_dim`.

   All of `dims` must be consecutive in order in the :attr:`self` tensor,
   but not necessary contiguous in memory.

   Examples::

       >>> imgs = torch.randn(32, 3, 128, 128, names=('N', 'C', 'H', 'W'))
       >>> flat_imgs = imgs.flatten(['C', 'H', 'W'], 'features')
       >>> flat_imgs.names, flat_imgs.shape
       (('N', 'features'), torch.Size([32, 49152]))

   .. warning::
       The named tensor API is experimental and subject to change.
```