tf.contrib.distributions.bijectors.ConditionalBijector

Contents

Class `ConditionalBijector`

Inherits From: `Bijector`

Defined in `tensorflow/contrib/distributions/python/ops/bijectors/conditional_bijector_impl.py`.

Conditional Bijector is a Bijector that allows intrinsic conditioning.

Properties

`dtype`

dtype of `Tensor`s transformable by this distribution.

`event_ndims`

Returns then number of event dimensions this bijector operates on.

`graph_parents`

Returns this `Bijector`'s graph_parents as a Python list.

`is_constant_jacobian`

Returns true iff the Jacobian is not a function of x.

> **Note:** Jacobian is either constant for both forward and inverse or neither.

Returns:

- `is_constant_jacobian`: Python `bool`.

`name`

Returns the string name of this `Bijector`.

`validate_args`

Returns True if Tensor arguments will be validated.

Methods

`__init__`

```
__init__(
    event_ndims=None,
    graph_parents=None,
    is_constant_jacobian=False,
    validate_args=False,
    dtype=None,
    name=None
)
```

Constructs Bijector.

A `Bijector` transforms random variables into new random variables.

Examples:

```
# Create the Y = g(X) = X transform which operates on vector events.
identity = Identity(event_ndims=1)

# Create the Y = g(X) = exp(X) transform which operates on matrices.
exp = Exp(event_ndims=2)
```

See `Bijector` subclass docstring for more details and specific examples.

Args:

- `event_ndims`: number of dimensions associated with event coordinates.
- `graph_parents`: Python list of graph prerequisites of this `Bijector`.
- `is_constant_jacobian`: Python `bool` indicating that the Jacobian is not a function of the input.
- `validate_args`: Python `bool`, default `False`. Whether to validate input with asserts. If `validate_args` is `False`, and the inputs are invalid, correct behavior is not guaranteed.
- `dtype`: `tf.dtype` supported by this `Bijector`. `None` means dtype is not enforced.
- `name`: The name to give Ops created by the initializer.

Raises:

- `ValueError`: If a member of `graph_parents` is not a `Tensor`.

`forward`

```
forward(
    *args,
    **kwargs
)
```

kwargs:

- `**condition_kwargs`: Named arguments forwarded to subclass implementation.

forward_event_shape

```
forward_event_shape(input_shape)
```

Shape of a single sample from a single batch as a `TensorShape`.

Same meaning as `forward_event_shape_tensor`. May be only partially defined.

Args:

- `input_shape`: `TensorShape` indicating event-portion shape passed into `forward` function.

Returns:

- `forward_event_shape_tensor`: `TensorShape` indicating event-portion shape after applying `forward`. Possibly unknown.

forward_event_shape_tensor

```
forward_event_shape_tensor(
    input_shape,
    name='forward_event_shape_tensor'
)
```

Shape of a single sample from a single batch as an `int32` 1D `Tensor`.

Args:

- `input_shape`: `Tensor`, `int32` vector indicating event-portion shape passed into `forward` function.
- `name`: name to give to the op

Returns:

- `forward_event_shape_tensor`: `Tensor`, `int32` vector indicating event-portion shape after applying `forward`.

forward_log_det_jacobian

```
forward_log_det_jacobian(
    *args,
    **kwargs
)
```

kwargs:

- `**condition_kwargs`: Named arguments forwarded to subclass implementation.

inverse

```
inverse(
    *args,
    **kwargs
)
```

kwargs:

- `**condition_kwargs` : Named arguments forwarded to subclass implementation.

inverse_event_shape

```
inverse_event_shape(output_shape)
```

Shape of a single sample from a single batch as a `TensorShape` .

Same meaning as `inverse_event_shape_tensor` . May be only partially defined.

Args:

- `output_shape` : `TensorShape` indicating event-portion shape passed into `inverse` function.

Returns:

- `inverse_event_shape_tensor` : `TensorShape` indicating event-portion shape after applying `inverse` . Possibly unknown.

inverse_event_shape_tensor

```
inverse_event_shape_tensor(
    output_shape,
    name='inverse_event_shape_tensor'
)
```

Shape of a single sample from a single batch as an `int32` 1D `Tensor` .

Args:

- `output_shape` : `Tensor` , `int32` vector indicating event-portion shape passed into `inverse` function.
- `name` : name to give to the op

Returns:

- `inverse_event_shape_tensor` : `Tensor` , `int32` vector indicating event-portion shape after applying `inverse` .

inverse_log_det_jacobian

```
inverse_log_det_jacobian(
    *args,
    **kwargs
)
```

kwargs:

- `**condition_kwargs`: Named arguments forwarded to subclass implementation.

Last updated November 2, 2017.