tf.SparseTensor

Contents

Class **SparseTensor**

Defined in `tensorflow/python/framework/sparse_tensor.py`.

See the guide: Sparse Tensors > Sparse Tensor Representation

Represents a sparse tensor.

TensorFlow represents a sparse tensor as three separate dense tensors: `indices`, `values`, and `dense_shape`. In Python, the three tensors are collected into a `SparseTensor` class for ease of use. If you have separate `indices`, `values`, and `dense_shape` tensors, wrap them in a `SparseTensor` object before passing to the ops below.

Concretely, the sparse tensor `SparseTensor(indices, values, dense_shape)` comprises the following components, where `N` and `ndims` are the number of values and number of dimensions in the `SparseTensor`, respectively:

- `indices`: A 2-D int64 tensor of dense_shape `[N, ndims]`, which specifies the indices of the elements in the sparse tensor that contain nonzero values (elements are zero-indexed). For example, `indices=[[1,3], [2,4]]` specifies that the elements with indexes of [1,3] and [2,4] have nonzero values.
- `values`: A 1-D tensor of any type and dense_shape `[N]`, which supplies the values for each element in `indices`. For example, given `indices=[[1,3], [2,4]]`, the parameter `values=[18, 3.6]` specifies that element [1,3] of the sparse tensor has a value of 18, and element [2,4] of the tensor has a value of 3.6.
- `dense_shape`: A 1-D int64 tensor of dense_shape `[ndims]`, which specifies the dense_shape of the sparse tensor. Takes a list indicating the number of elements in each dimension. For example, `dense_shape=[3,6]` specifies a two-dimensional 3x6 tensor, `dense_shape=[2,3,4]` specifies a three-dimensional 2x3x4 tensor, and `dense_shape=[9]` specifies a one-dimensional tensor with 9 elements.

The corresponding dense tensor satisfies:

```
dense.shape = dense_shape
dense[tuple(indices[i])] = values[i]
```

By convention, `indices` should be sorted in row-major order (or equivalently lexicographic order on the tuples `indices[i]`). This is not enforced when `SparseTensor` objects are constructed, but most ops assume correct ordering. If the ordering of sparse tensor `st` is wrong, a fixed version can be obtained by calling `tf.sparse_reorder(st)`.

Example: The sparse tensor

```
SparseTensor(indices=[[0, 0], [1, 2]], values=[1, 2], dense_shape=[3, 4])
```

represents the dense tensor

```
[[1, 0, 0, 0]
 [0, 0, 2, 0]
 [0, 0, 0, 0]]
```

Properties

`dense_shape`

A 1-D Tensor of int64 representing the shape of the dense tensor.

`dtype`

The `DType` of elements in this tensor.

`graph`

The `Graph` that contains the index, value, and dense_shape tensors.

`indices`

The indices of non-zero values in the represented dense tensor.

Returns:

A 2-D Tensor of int64 with dense_shape `[N, ndims]`, where `N` is the number of non-zero values in the tensor, and `ndims` is the rank.

`op`

The `Operation` that produces `values` as an output.

`values`

The non-zero values in the represented dense tensor.

Returns:

A 1-D Tensor of any data type.

Methods

`__init__`

```
__init__(
    indices,
    values,
    dense_shape
)
```

Creates a `SparseTensor`.

Args:

- `indices`: A 2-D int64 tensor of shape `[N, ndims]`.
- `values`: A 1-D tensor of any type and shape `[N]`.
- `dense_shape`: A 1-D int64 tensor of shape `[ndims]`.

Returns:

A `SparseTensor`.

__div__

```
__div__(
    sp_x,
    y
)
```

Component-wise divides a SparseTensor by a dense Tensor.

Limitation: this Op only broadcasts the dense side to the sparse side, but not the other direction.

Args:

- `sp_indices`: A `Tensor` of type `int64`. 2-D. `N x R` matrix with the indices of non-empty values in a SparseTensor, possibly not in canonical ordering.
- `sp_values`: A `Tensor`. Must be one of the following types: `float32`, `float64`, `int64`, `int32`, `uint8`, `uint16`, `int16`, `int8`, `complex64`, `complex128`, `qint8`, `quint8`, `qint32`, `half`. 1-D. `N` non-empty values corresponding to `sp_indices`.
- `sp_shape`: A `Tensor` of type `int64`. 1-D. Shape of the input SparseTensor.
- `dense`: A `Tensor`. Must have the same type as `sp_values`. `R`-D. The dense Tensor operand.
- `name`: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `sp_values`. 1-D. The `N` values that are operated on.

__mul__

```
__mul__(
    sp_x,
    y
)
```

Component-wise multiplies a SparseTensor by a dense Tensor.

The output locations corresponding to the implicitly zero elements in the sparse tensor will be zero (i.e., will not take up storage space), regardless of the contents of the dense tensor (even if it's +/-INF and that INF*0 == NaN).

Limitation: this Op only broadcasts the dense side to the sparse side, but not the other direction.

Args:

- `sp_indices`: A `Tensor` of type `int64`. 2-D. `N x R` matrix with the indices of non-empty values in a SparseTensor, possibly not in canonical ordering.
- `sp_values`: A `Tensor`. Must be one of the following types: `float32`, `float64`, `int64`, `int32`, `uint8`, `uint16`, `int16`, `int8`, `complex64`, `complex128`, `qint8`, `quint8`, `qint32`, `half`. 1-D. `N` non-empty values corresponding to `sp_indices`.
- `sp_shape`: A `Tensor` of type `int64`. 1-D. Shape of the input SparseTensor.
- `dense`: A `Tensor`. Must have the same type as `sp_values`. `R`-D. The dense Tensor operand.
- `name`: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `sp_values`. 1-D. The `N` values that are operated on.

`__truediv__`

```
__truediv__(
    sp_x,
    y
)
```

Internal helper function for 'sp_t / dense_t'.

`eval`

```
eval(
    feed_dict=None,
    session=None
)
```

Evaluates this sparse tensor in a `Session`.

Calling this method will execute all preceding operations that produce the inputs needed for the operation that produces this tensor.

N.B. Before invoking `SparseTensor.eval()`, its graph must have been launched in a session, and either a default session must be available, or `session` must be specified explicitly.

Args:

- `feed_dict`: A dictionary that maps `Tensor` objects to feed values. See `tf.Session.run` for a description of the valid feed values.
- `session`: (Optional.) The `Session` to be used to evaluate this sparse tensor. If none, the default session will be used.

Returns:

A `SparseTensorValue` object.

`from_value`

```
@classmethod
from_value(
    cls,
    sparse_tensor_value
)
```

get_shape

```
get_shape()
```

Get the `TensorShape` representing the shape of the dense tensor.

Returns:

A `TensorShape` object.

Last updated November 2, 2017.