Module: tf.sets

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets#top_of_page)
* [Functions](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets#functions)

Tensorflow set operations.

Functions

[difference(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/difference): Compute set difference of elements in last dimension of a and b.

[intersection(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/intersection): Compute set intersection of elements in last dimension of a and b.

[size(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/size): Compute number of unique elements along last dimension of a.

[union(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/union): Compute set union of elements in last dimension of a and b.

# tf.sets.difference

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/difference#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/difference#aliases)

Compute set difference of elements in last dimension of a and b.

### Aliases:

* tf.compat.v1.sets.difference
* tf.compat.v1.sets.set\_difference
* tf.compat.v2.sets.difference
* tf.sets.difference

tf.sets.difference(  
    a,  
    b,  
    aminusb=True,  
    validate\_indices=True  
)

Defined in [python/ops/sets\_impl.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/sets_impl.py).

All but the last dimension of a and b must match.

#### Example:

  import tensorflow as tf  
  import collections  
  
  # Represent the following array of sets as a sparse tensor:  
  # a = np.array([[{1, 2}, {3}], [{4}, {5, 6}]])  
  a = collections.OrderedDict([  
      ((0, 0, 0), 1),  
      ((0, 0, 1), 2),  
      ((0, 1, 0), 3),  
      ((1, 0, 0), 4),  
      ((1, 1, 0), 5),  
      ((1, 1, 1), 6),  
  ])  
  a = tf.SparseTensor(list(a.keys()), list(a.values()), dense\_shape=[2, 2, 2])  
  
  # np.array([[{1, 3}, {2}], [{4, 5}, {5, 6, 7, 8}]])  
  b = collections.OrderedDict([  
      ((0, 0, 0), 1),  
      ((0, 0, 1), 3),  
      ((0, 1, 0), 2),  
      ((1, 0, 0), 4),  
      ((1, 0, 1), 5),  
      ((1, 1, 0), 5),  
      ((1, 1, 1), 6),  
      ((1, 1, 2), 7),  
      ((1, 1, 3), 8),  
  ])  
  b = tf.SparseTensor(list(b.keys()), list(b.values()), dense\_shape=[2, 2, 4])  
  
  # `set\_difference` is applied to each aligned pair of sets.  
  tf.sets.difference(a, b)  
  
  # The result will be equivalent to either of:  
  #  
  # np.array([[{2}, {3}], [{}, {}]])  
  #  
  # collections.OrderedDict([  
  #     ((0, 0, 0), 2),  
  #     ((0, 1, 0), 3),  
  # ])

#### Args:

* **a**: Tensor or SparseTensor of the same type as b. If sparse, indices must be sorted in row-major order.
* **b**: Tensor or SparseTensor of the same type as a. If sparse, indices must be sorted in row-major order.
* **aminusb**: Whether to subtract b from a, vs vice versa.
* **validate\_indices**: Whether to validate the order and range of sparse indices in a and b.

#### Returns:

A SparseTensor whose shape is the same rank as a and b, and all but the last dimension the same. Elements along the last dimension contain the differences.

# tf.sets.intersection

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/intersection#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/intersection#aliases)

Compute set intersection of elements in last dimension of a and b.

### Aliases:

* tf.compat.v1.sets.intersection
* tf.compat.v1.sets.set\_intersection
* tf.compat.v2.sets.intersection
* tf.sets.intersection

tf.sets.intersection(  
    a,  
    b,  
    validate\_indices=True  
)

Defined in [python/ops/sets\_impl.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/sets_impl.py).

All but the last dimension of a and b must match.

#### Example:

  import tensorflow as tf  
  import collections  
  
  # Represent the following array of sets as a sparse tensor:  
  # a = np.array([[{1, 2}, {3}], [{4}, {5, 6}]])  
  a = collections.OrderedDict([  
      ((0, 0, 0), 1),  
      ((0, 0, 1), 2),  
      ((0, 1, 0), 3),  
      ((1, 0, 0), 4),  
      ((1, 1, 0), 5),  
      ((1, 1, 1), 6),  
  ])  
  a = tf.SparseTensor(list(a.keys()), list(a.values()), dense\_shape=[2,2,2])  
  
  # b = np.array([[{1}, {}], [{4}, {5, 6, 7, 8}]])  
  b = collections.OrderedDict([  
      ((0, 0, 0), 1),  
      ((1, 0, 0), 4),  
      ((1, 1, 0), 5),  
      ((1, 1, 1), 6),  
      ((1, 1, 2), 7),  
      ((1, 1, 3), 8),  
  ])  
  b = tf.SparseTensor(list(b.keys()), list(b.values()), dense\_shape=[2, 2, 4])  
  
  # `tf.sets.intersection` is applied to each aligned pair of sets.  
  tf.sets.intersection(a, b)  
  
  # The result will be equivalent to either of:  
  #  
  # np.array([[{1}, {}], [{4}, {5, 6}]])  
  #  
  # collections.OrderedDict([  
  #     ((0, 0, 0), 1),  
  #     ((1, 0, 0), 4),  
  #     ((1, 1, 0), 5),  
  #     ((1, 1, 1), 6),  
  # ])

#### Args:

* **a**: Tensor or SparseTensor of the same type as b. If sparse, indices must be sorted in row-major order.
* **b**: Tensor or SparseTensor of the same type as a. If sparse, indices must be sorted in row-major order.
* **validate\_indices**: Whether to validate the order and range of sparse indices in a and b.

#### Returns:

A SparseTensor whose shape is the same rank as a and b, and all but the last dimension the same. Elements along the last dimension contain the intersections.

# tf.sets.size

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/size#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/size#aliases)

Compute number of unique elements along last dimension of a.

### Aliases:

* tf.compat.v1.sets.set\_size
* tf.compat.v1.sets.size
* tf.compat.v2.sets.size
* tf.sets.size

tf.sets.size(  
    a,  
    validate\_indices=True  
)

Defined in [python/ops/sets\_impl.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/sets_impl.py).

#### Args:

* **a**: SparseTensor, with indices sorted in row-major order.
* **validate\_indices**: Whether to validate the order and range of sparse indices in a.

#### Returns:

int32 Tensor of set sizes. For a ranked n, this is a Tensor with rank n-1, and the same 1st n-1dimensions as a. Each value is the number of unique elements in the corresponding [0...n-1]dimension of a.

#### Raises:

* **TypeError**: If a is an invalid types.

# tf.sets.union

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/union#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/sets/union#aliases)

Compute set union of elements in last dimension of a and b.

### Aliases:

* tf.compat.v1.sets.set\_union
* tf.compat.v1.sets.union
* tf.compat.v2.sets.union
* tf.sets.union

tf.sets.union(  
    a,  
    b,  
    validate\_indices=True  
)

Defined in [python/ops/sets\_impl.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/sets_impl.py).

All but the last dimension of a and b must match.

#### Example:

  import tensorflow as tf  
  import collections  
  
  # [[{1, 2}, {3}], [{4}, {5, 6}]]  
  a = collections.OrderedDict([  
      ((0, 0, 0), 1),  
      ((0, 0, 1), 2),  
      ((0, 1, 0), 3),  
      ((1, 0, 0), 4),  
      ((1, 1, 0), 5),  
      ((1, 1, 1), 6),  
  ])  
  a = tf.SparseTensor(list(a.keys()), list(a.values()), dense\_shape=[2, 2, 2])  
  
  # [[{1, 3}, {2}], [{4, 5}, {5, 6, 7, 8}]]  
  b = collections.OrderedDict([  
      ((0, 0, 0), 1),  
      ((0, 0, 1), 3),  
      ((0, 1, 0), 2),  
      ((1, 0, 0), 4),  
      ((1, 0, 1), 5),  
      ((1, 1, 0), 5),  
      ((1, 1, 1), 6),  
      ((1, 1, 2), 7),  
      ((1, 1, 3), 8),  
  ])  
  b = tf.SparseTensor(list(b.keys()), list(b.values()), dense\_shape=[2, 2, 4])  
  
  # `set\_union` is applied to each aligned pair of sets.  
  tf.sets.union(a, b)  
  
  # The result will be a equivalent to either of:  
  #  
  # np.array([[{1, 2, 3}, {2, 3}], [{4, 5}, {5, 6, 7, 8}]])  
  #  
  # collections.OrderedDict([  
  #     ((0, 0, 0), 1),  
  #     ((0, 0, 1), 2),  
  #     ((0, 0, 2), 3),  
  #     ((0, 1, 0), 2),  
  #     ((0, 1, 1), 3),  
  #     ((1, 0, 0), 4),  
  #     ((1, 0, 1), 5),  
  #     ((1, 1, 0), 5),  
  #     ((1, 1, 1), 6),  
  #     ((1, 1, 2), 7),  
  #     ((1, 1, 3), 8),  
  # ])

#### Args:

* **a**: Tensor or SparseTensor of the same type as b. If sparse, indices must be sorted in row-major order.
* **b**: Tensor or SparseTensor of the same type as a. If sparse, indices must be sorted in row-major order.
* **validate\_indices**: Whether to validate the order and range of sparse indices in a and b.

#### Returns:

A SparseTensor whose shape is the same rank as a and b, and all but the last dimension the same. Elements along the last dimension contain the unions.