Module: tf.compat.v1.ragged / tf.ragged

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged#top_of_page)
* [Additional ops that support RaggedTensor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged#additional_ops_that_support_raggedtensor)
* [Classes](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged#classes)
* [Functions](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged#functions)

Ragged Tensors.

This package defines ops for manipulating ragged tensors ([tf.RaggedTensor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/RaggedTensor)), which are tensors with non-uniform shapes. In particular, each RaggedTensor has one or more *ragged dimensions*, which are dimensions whose slices may have different lengths. For example, the inner (column) dimension of rt=[[3, 1, 4, 1], [], [5, 9, 2], [6], []] is ragged, since the column slices (rt[0, :], ..., rt[4, :]) have different lengths. For a more detailed description of ragged tensors, see the [tf.RaggedTensor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/RaggedTensor) class documentation and the [Ragged Tensor Guide](https://www.tensorflow.org/guide/ragged_tensors).

Additional ops that support RaggedTensor

Arguments that accept RaggedTensors are marked in **bold**.

* tf.batch\_gather(**params**, **indices**, name=None)
* [tf.bitwise.bitwise\_and](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/bitwise/bitwise_and)(**x**, **y**, name=None)
* [tf.bitwise.bitwise\_or](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/bitwise/bitwise_or)(**x**, **y**, name=None)
* [tf.bitwise.bitwise\_xor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/bitwise/bitwise_xor)(**x**, **y**, name=None)
* [tf.bitwise.invert](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/bitwise/invert)(**x**, name=None)
* [tf.bitwise.left\_shift](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/bitwise/left_shift)(**x**, **y**, name=None)
* [tf.bitwise.right\_shift](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/bitwise/right_shift)(**x**, **y**, name=None)
* [tf.clip\_by\_value](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/clip_by_value)(**t**, clip\_value\_min, clip\_value\_max, name=None)
* [tf.concat](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/concat)(**values**, axis, name='concat')
* [tf.debugging.check\_numerics](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/debugging/check_numerics)(**tensor**, message, name=None)
* [tf.dtypes.cast](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/cast)(**x**, dtype, name=None)
* [tf.dtypes.complex](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/complex)(**real**, **imag**, name=None)
* [tf.dtypes.saturate\_cast](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/saturate_cast)(**value**, dtype, name=None)
* [tf.expand\_dims](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/expand_dims)(**input**, axis=None, name=None, dim=None)
* [tf.gather\_nd](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/gather_nd)(**params**, **indices**, name=None, batch\_dims=0)
* [tf.gather](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/gather)(**params**, **indices**, validate\_indices=None, name=None, axis=None, batch\_dims=0)
* [tf.identity](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/identity)(**input**, name=None)
* [tf.io.decode\_base64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/io/decode_base64)(**input**, name=None)
* [tf.io.decode\_compressed](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/io/decode_compressed)(**bytes**, compression\_type='', name=None)
* [tf.io.encode\_base64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/io/encode_base64)(**input**, pad=False, name=None)
* [tf.math.abs](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/abs)(**x**, name=None)
* [tf.math.acos](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/acos)(**x**, name=None)
* [tf.math.acosh](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/acosh)(**x**, name=None)
* [tf.math.add\_n](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/add_n)(**inputs**, name=None)
* [tf.math.add](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/add)(**x**, **y**, name=None)
* [tf.math.angle](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/angle)(**input**, name=None)
* [tf.math.asin](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/asin)(**x**, name=None)
* [tf.math.asinh](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/asinh)(**x**, name=None)
* [tf.math.atan2](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/atan2)(**y**, **x**, name=None)
* [tf.math.atan](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/atan)(**x**, name=None)
* [tf.math.atanh](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/atanh)(**x**, name=None)
* [tf.math.ceil](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/ceil)(**x**, name=None)
* [tf.math.conj](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/conj)(**x**, name=None)
* [tf.math.cos](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/cos)(**x**, name=None)
* [tf.math.cosh](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/cosh)(**x**, name=None)
* [tf.math.digamma](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/digamma)(**x**, name=None)
* [tf.math.divide\_no\_nan](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/divide_no_nan)(**x**, **y**, name=None)
* [tf.math.divide](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/divide)(**x**, **y**, name=None)
* [tf.math.equal](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/equal)(**x**, **y**, name=None)
* [tf.math.erf](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/erf)(**x**, name=None)
* [tf.math.erfc](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/erfc)(**x**, name=None)
* [tf.math.exp](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/exp)(**x**, name=None)
* [tf.math.expm1](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/expm1)(**x**, name=None)
* [tf.math.floor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/floor)(**x**, name=None)
* [tf.math.floordiv](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/floordiv)(**x**, **y**, name=None)
* [tf.math.floormod](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/floormod)(**x**, **y**, name=None)
* [tf.math.greater\_equal](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/greater_equal)(**x**, **y**, name=None)
* [tf.math.greater](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/greater)(**x**, **y**, name=None)
* [tf.math.imag](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/imag)(**input**, name=None)
* [tf.math.is\_finite](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/is_finite)(**x**, name=None)
* [tf.math.is\_inf](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/is_inf)(**x**, name=None)
* [tf.math.is\_nan](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/is_nan)(**x**, name=None)
* [tf.math.less\_equal](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/less_equal)(**x**, **y**, name=None)
* [tf.math.less](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/less)(**x**, **y**, name=None)
* [tf.math.lgamma](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/lgamma)(**x**, name=None)
* [tf.math.log1p](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/log1p)(**x**, name=None)
* [tf.math.log\_sigmoid](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/log_sigmoid)(**x**, name=None)
* [tf.math.log](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/log)(**x**, name=None)
* [tf.math.logical\_and](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/logical_and)(**x**, **y**, name=None)
* [tf.math.logical\_not](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/logical_not)(**x**, name=None)
* [tf.math.logical\_or](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/logical_or)(**x**, **y**, name=None)
* [tf.math.logical\_xor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/logical_xor)(**x**, **y**, name='LogicalXor')
* [tf.math.maximum](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/maximum)(**x**, **y**, name=None)
* [tf.math.minimum](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/minimum)(**x**, **y**, name=None)
* [tf.math.multiply](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/multiply)(**x**, **y**, name=None)
* [tf.math.negative](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/negative)(**x**, name=None)
* [tf.math.not\_equal](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/not_equal)(**x**, **y**, name=None)
* [tf.math.pow](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/pow)(**x**, **y**, name=None)
* [tf.math.real](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/real)(**input**, name=None)
* [tf.math.reciprocal](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/reciprocal)(**x**, name=None)
* [tf.math.reduce\_any](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/reduce_any)(**input\_tensor**, axis=None, keepdims=False, name=None)
* [tf.math.reduce\_max](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/reduce_max)(**input\_tensor**, axis=None, keepdims=False, name=None)
* [tf.math.reduce\_mean](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/reduce_mean)(**input\_tensor**, axis=None, keepdims=False, name=None)
* [tf.math.reduce\_min](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/reduce_min)(**input\_tensor**, axis=None, keepdims=False, name=None)
* [tf.math.reduce\_prod](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/reduce_prod)(**input\_tensor**, axis=None, keepdims=False, name=None)
* [tf.math.reduce\_sum](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/reduce_sum)(**input\_tensor**, axis=None, keepdims=False, name=None)
* [tf.math.rint](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/rint)(**x**, name=None)
* [tf.math.round](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/round)(**x**, name=None)
* [tf.math.rsqrt](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/rsqrt)(**x**, name=None)
* [tf.math.sign](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/sign)(**x**, name=None)
* [tf.math.sin](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/sin)(**x**, name=None)
* [tf.math.sinh](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/sinh)(**x**, name=None)
* [tf.math.sqrt](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/sqrt)(**x**, name=None)
* [tf.math.square](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/square)(**x**, name=None)
* [tf.math.squared\_difference](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/squared_difference)(**x**, **y**, name=None)
* [tf.math.subtract](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/subtract)(**x**, **y**, name=None)
* [tf.math.tan](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/tan)(**x**, name=None)
* [tf.math.truediv](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/truediv)(**x**, **y**, name=None)
* [tf.math.unsorted\_segment\_max](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/unsorted_segment_max)(**data**, **segment\_ids**, num\_segments, name=None)
* [tf.math.unsorted\_segment\_mean](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/unsorted_segment_mean)(**data**, **segment\_ids**, num\_segments, name=None)
* [tf.math.unsorted\_segment\_min](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/unsorted_segment_min)(**data**, **segment\_ids**, num\_segments, name=None)
* [tf.math.unsorted\_segment\_prod](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/unsorted_segment_prod)(**data**, **segment\_ids**, num\_segments, name=None)
* [tf.math.unsorted\_segment\_sqrt\_n](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/unsorted_segment_sqrt_n)(**data**, **segment\_ids**, num\_segments, name=None)
* [tf.math.unsorted\_segment\_sum](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/math/unsorted_segment_sum)(**data**, **segment\_ids**, num\_segments, name=None)
* [tf.ones\_like](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ones_like)(**tensor**, dtype=None, name=None, optimize=True)
* [tf.rank](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/rank)(**input**, name=None)
* [tf.realdiv](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/realdiv)(**x**, **y**, name=None)
* [tf.reduce\_all](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/reduce_all)(**input\_tensor**, axis=None, keepdims=False, name=None)
* [tf.size](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/size)(**input**, name=None, out\_type=[tf.int32](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int32))
* [tf.squeeze](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/squeeze)(**input**, axis=None, name=None, squeeze\_dims=None)
* [tf.stack](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/stack)(**values**, axis=0, name='stack')
* [tf.strings.as\_string](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/as_string)(**input**, precision=-1, scientific=False, shortest=False, width=-1, fill='', name=None)
* [tf.strings.join](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/join)(**inputs**, separator='', name=None)
* [tf.strings.length](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/length)(**input**, name=None, unit='BYTE')
* [tf.strings.regex\_full\_match](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/regex_full_match)(**input**, pattern, name=None)
* [tf.strings.regex\_replace](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/regex_replace)(**input**, pattern, rewrite, replace\_global=True, name=None)
* [tf.strings.strip](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/strip)(**input**, name=None)
* [tf.strings.substr](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/substr)(**input**, pos, len, name=None, unit='BYTE')
* [tf.strings.to\_hash\_bucket\_fast](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/to_hash_bucket_fast)(**input**, num\_buckets, name=None)
* [tf.strings.to\_hash\_bucket\_strong](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/to_hash_bucket_strong)(**input**, num\_buckets, key, name=None)
* [tf.strings.unicode\_script](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/strings/unicode_script)(**input**, name=None)
* [tf.tile](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/tile)(**input**, multiples, name=None)
* [tf.truncatediv](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/truncatediv)(**x**, **y**, name=None)
* [tf.truncatemod](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/truncatemod)(**x**, **y**, name=None)
* [tf.where](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/where)(**condition**, **x**=None, **y**=None, name=None)
* [tf.zeros\_like](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/zeros_like)(**tensor**, dtype=None, name=None, optimize=True)n

Classes

[class RaggedTensorValue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/RaggedTensorValue): Represents the value of a RaggedTensor.

Functions

[boolean\_mask(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/boolean_mask): Applies a boolean mask to data without flattening the mask dimensions.

[constant(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/constant): Constructs a constant RaggedTensor from a nested Python list.

[constant\_value(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/constant_value): Constructs a RaggedTensorValue from a nested Python list.

[map\_flat\_values(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/map_flat_values): Applies op to the values of one or more RaggedTensors.

[placeholder(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/placeholder): Creates a placeholder for a [tf.RaggedTensor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/RaggedTensor) that will always be fed.

[range(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/range): Returns a RaggedTensor containing the specified sequences of numbers.

[row\_splits\_to\_segment\_ids(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/row_splits_to_segment_ids): Generates the segmentation corresponding to a RaggedTensor row\_splits.

[segment\_ids\_to\_row\_splits(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/segment_ids_to_row_splits): Generates the RaggedTensor row\_splits corresponding to a segmentation.

# tf.compat.v1.ragged.constant\_value

Constructs a RaggedTensorValue from a nested Python list.

tf.compat.v1.ragged.constant\_value(  
    pylist,  
    dtype=None,  
    ragged\_rank=None,  
    inner\_shape=None,  
    row\_splits\_dtype='int64'  
)

Defined in [python/ops/ragged/ragged\_factory\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/ragged_factory_ops.py).

**Warning:** This function returns a **RaggedTensorValue**, not a **RaggedTensor**. If you wish to construct a constant **RaggedTensor**, use [**ragged.constant(...)**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/constant) instead.

#### Example:

>>> ragged.constant\_value([[1, 2], [3], [4, 5, 6]])  
RaggedTensorValue(values=[1, 2, 3, 4, 5, 6], splits=[0, 2, 3, 6])

All scalar values in pylist must have the same nesting depth K, and the returned RaggedTensorValue will have rank K. If pylist contains no scalar values, then K is one greater than the maximum depth of empty lists in pylist. All scalar values in pylist must be compatible with dtype.

#### Args:

* **pylist**: A nested list, tuple or np.ndarray. Any nested element that is not a list or tuple must be a scalar value compatible with dtype.
* **dtype**: numpy.dtype. The type of elements for the returned RaggedTensor. If not specified, then a default is chosen based on the scalar values in pylist.
* **ragged\_rank**: An integer specifying the ragged rank of the returned RaggedTensorValue. Must be nonnegative and less than K. Defaults to max(0, K - 1) if inner\_shape is not specified. Defaults to `max(0, K
  + 1 - len(inner\_shape))ifinner\_shape` is specified.
* **inner\_shape**: A tuple of integers specifying the shape for individual inner values in the returned RaggedTensorValue. Defaults to () if ragged\_rank is not specified. If ragged\_rank is specified, then a default is chosen based on the contents of pylist.
* **row\_splits\_dtype**: data type for the constructed RaggedTensorValue's row\_splits. One of numpy.int32 or numpy.int64.

#### Returns:

A tf.RaggedTensorValue or numpy.array with rank K and the specified ragged\_rank, containing the values from pylist.

#### Raises:

* **ValueError**: If the scalar values in pylist have inconsistent nesting depth; or if ragged\_rank or inner\_shape are incompatible with pylist.

# tf.compat.v1.ragged.placeholder

Creates a placeholder for a [tf.RaggedTensor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/RaggedTensor) that will always be fed.

tf.compat.v1.ragged.placeholder(  
    dtype,  
    ragged\_rank,  
    value\_shape=None,  
    name=None  
)

Defined in [python/ops/ragged/ragged\_factory\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/ragged_factory_ops.py).

**Important**: This ragged tensor will produce an error if evaluated. Its value must be fed using the feed\_dict optional argument to Session.run(), Tensor.eval(), or Operation.run().

@compatibility{eager} Placeholders are not compatible with eager execution.

#### Args:

* **dtype**: The data type for the RaggedTensor.
* **ragged\_rank**: The ragged rank for the RaggedTensor
* **value\_shape**: The shape for individual flat values in the RaggedTensor.
* **name**: A name for the operation (optional).

#### Returns:

A RaggedTensor that may be used as a handle for feeding a value, but not evaluated directly.

#### Raises:

* **RuntimeError**: if eager execution is enabled

# tf.compat.v1.ragged.RaggedTensorValue

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/RaggedTensorValue#top_of_page)
* [Class RaggedTensorValue](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/RaggedTensorValue#class_raggedtensorvalue)
* [\_\_init\_\_](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/RaggedTensorValue#__init__)
* [Properties](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/RaggedTensorValue#properties)
  + [dtype](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/compat/v1/ragged/RaggedTensorValue#dtype)

## Class RaggedTensorValue

Represents the value of a RaggedTensor.

Defined in [python/ops/ragged/ragged\_tensor\_value.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/ragged_tensor_value.py).

**Warning:** **RaggedTensorValue** should only be used in graph mode; in eager mode, the [**tf.RaggedTensor**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/RaggedTensor) class contains its value directly.

See [tf.RaggedTensor](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/RaggedTensor) for a description of ragged tensors.

## \_\_init\_\_

\_\_init\_\_(  
    values,  
    row\_splits  
)

Creates a RaggedTensorValue.

#### Args:

* **values**: A numpy array of any type and shape; or a RaggedTensorValue.
* **row\_splits**: A 1-D int32 or int64 numpy array.

## Properties

### dtype

The numpy dtype of values in this tensor.

### flat\_values

The innermost values array for this ragged tensor value.

### nested\_row\_splits

The row\_splits for all ragged dimensions in this ragged tensor value.

### ragged\_rank

The number of ragged dimensions in this ragged tensor value.

### row\_splits

The split indices for the ragged tensor value.

### shape

A tuple indicating the shape of this RaggedTensorValue.

### values

The concatenated values for all rows in this tensor.

## Methods

### to\_list

to\_list()

Returns this ragged tensor value as a nested Python list.

# tf.ragged.boolean\_mask

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

Applies a boolean mask to data without flattening the mask dimensions.

### Aliases:

* tf.compat.v1.ragged.boolean\_mask
* tf.compat.v2.ragged.boolean\_mask
* tf.ragged.boolean\_mask

tf.ragged.boolean\_mask(  
    data,  
    mask,  
    name=None  
)

Defined in [python/ops/ragged/ragged\_array\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/ragged_array_ops.py).

Returns a potentially ragged tensor that is formed by retaining the elements in data where the corresponding value in mask is True.

* output[a1...aA, i, b1...bB] = data[a1...aA, j, b1...bB]

Where j is the ith True entry of mask[a1...aA].

Note that output preserves the mask dimensions a1...aA; this differs from [tf.boolean\_mask](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/boolean_mask), which flattens those dimensions.

#### Args:

* **data**: A potentially ragged tensor.
* **mask**: A potentially ragged boolean tensor. mask's shape must be a prefix of data's shape.rank(mask) must be known statically.
* **name**: A name prefix for the returned tensor (optional).

#### Returns:

A potentially ragged tensor that is formed by retaining the elements in data where the corresponding value in mask is True.

* rank(output) = rank(data).
* output.ragged\_rank = max(data.ragged\_rank, rank(mask) - 1).

#### Raises:

* **ValueError**: if rank(mask) is not known statically; or if mask.shape is not a prefix of data.shape.

#### Examples:

>>> # Aliases for True & False so data and mask line up.  
>>> T, F = (True, False)  
  
>>> tf.ragged.boolean\_mask(  # Mask a 2D Tensor.  
...     data=[[1, 2, 3], [4, 5, 6], [7, 8, 9]],  
...     mask=[[T, F, T], [F, F, F], [T, F, F]]).tolist()  
[[1, 3], [], [7]]  
  
>>> tf.ragged.boolean\_mask(  # Mask a 2D RaggedTensor.  
...     tf.ragged.constant([[1, 2, 3], [4], [5, 6]]),  
...     tf.ragged.constant([[F, F, T], [F], [T, T]])).tolist()  
[[3], [], [5, 6]]  
  
>>> tf.ragged.boolean\_mask(  # Mask rows of a 2D RaggedTensor.  
...     tf.ragged.constant([[1, 2, 3], [4], [5, 6]]),  
...     tf.ragged.constant([True, False, True])).tolist()  
[[1, 2, 3], [5, 6]]

# tf.ragged.constant

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/constant#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/constant#aliases)
* [Used in the guide:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/constant#used_in_the_guide)

Constructs a constant RaggedTensor from a nested Python list.

### Aliases:

* tf.compat.v1.ragged.constant
* tf.compat.v2.ragged.constant
* tf.ragged.constant

tf.ragged.constant(  
    pylist,  
    dtype=None,  
    ragged\_rank=None,  
    inner\_shape=None,  
    name=None,  
    row\_splits\_dtype=tf.dtypes.int64  
)

Defined in [python/ops/ragged/ragged\_factory\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/ragged_factory_ops.py).

### Used in the guide:

* [Ragged Tensors](https://www.tensorflow.org/beta/guide/ragged_tensors)

#### Example:

>>> ragged.constant([[1, 2], [3], [4, 5, 6]]).eval()  
RaggedTensorValue(values=[1, 2, 3, 4, 5, 6], splits=[0, 2, 3, 6])

All scalar values in pylist must have the same nesting depth K, and the returned RaggedTensorwill have rank K. If pylist contains no scalar values, then K is one greater than the maximum depth of empty lists in pylist. All scalar values in pylist must be compatible with dtype.

#### Args:

* **pylist**: A nested list, tuple or np.ndarray. Any nested element that is not a list, tuple or np.ndarray must be a scalar value compatible with dtype.
* **dtype**: The type of elements for the returned RaggedTensor. If not specified, then a default is chosen based on the scalar values in pylist.
* **ragged\_rank**: An integer specifying the ragged rank of the returned RaggedTensor. Must be nonnegative and less than K. Defaults to max(0, K - 1) if inner\_shape is not specified. Defaults to `max(0, K
  + 1 - len(inner\_shape))ifinner\_shape` is specified.
* **inner\_shape**: A tuple of integers specifying the shape for individual inner values in the returned RaggedTensor. Defaults to () if ragged\_rank is not specified. If ragged\_rank is specified, then a default is chosen based on the contents of pylist.
* **name**: A name prefix for the returned tensor (optional).
* **row\_splits\_dtype**: data type for the constructed RaggedTensor's row\_splits. One of [tf.int32](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int32) or [tf.int64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int64).

#### Returns:

A potentially ragged tensor with rank K and the specified ragged\_rank, containing the values from pylist.

#### Raises:

* **ValueError**: If the scalar values in pylist have inconsistent nesting depth; or if ragged\_rank or inner\_shape are incompatible with pylist.

# tf.ragged.map\_flat\_values

* [**Contents**](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/map_flat_values#top_of_page)
* [Aliases:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/map_flat_values#aliases)
* [Used in the guide:](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/ragged/map_flat_values#used_in_the_guide)

Applies op to the values of one or more RaggedTensors.

### Aliases:

* tf.compat.v1.ragged.map\_flat\_values
* tf.compat.v2.ragged.map\_flat\_values
* tf.ragged.map\_flat\_values

tf.ragged.map\_flat\_values(  
    op,  
    \*args,  
    \*\*kwargs  
)

Defined in [python/ops/ragged/ragged\_functional\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/ragged_functional_ops.py).

### Used in the guide:

* [Ragged Tensors](https://www.tensorflow.org/beta/guide/ragged_tensors)

Replaces any RaggedTensor in args or kwargs with its flat\_values tensor, and then calls op. Returns a RaggedTensor that is constructed from the input RaggedTensors' nested\_row\_splitsand the value returned by the op.

If the input arguments contain multiple RaggedTensors, then they must have identical nested\_row\_splits.

#### Examples:

>>> rt = ragged.constant([[1, 2, 3], [], [4, 5], [6]])  
>>> ragged.map\_flat\_values(tf.ones\_like, rt).eval().tolist()  
[[1, 1, 1], [], [1, 1], [1]]  
>>> ragged.map\_flat\_values(tf.multiply, rt, rt).eval().tolist()  
[[1, 4, 9], [], [16, 25], [36]]  
>>> ragged.map\_flat\_values(tf.add, rt, 5).eval().tolist()  
[[6, 7, 8], [], [9, 10], [11]]

#### Args:

* **op**: The operation that should be applied to the RaggedTensor flat\_values. op is typically an element-wise operation (such as math\_ops.add), but any operation that preserves the size of the outermost dimension can be used. I.e., shape[0] of the value returned by op must matchshape[0] of the RaggedTensors' flat\_values tensors.
* **\*args**: Arguments for op.
* **\*\*kwargs**: Keyword arguments for op.

#### Returns:

A RaggedTensor whose ragged\_rank matches the ragged\_rank of all input RaggedTensors.

#### Raises:

* **ValueError**: If args contains no RaggedTensors, or if the nested\_splits of the input RaggedTensors are not identical.

# tf.ragged.range

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

Returns a RaggedTensor containing the specified sequences of numbers.

### Aliases:

* tf.compat.v1.ragged.range
* tf.compat.v2.ragged.range
* tf.ragged.range

tf.ragged.range(  
    starts,  
    limits=None,  
    deltas=1,  
    dtype=None,  
    name=None,  
    row\_splits\_dtype=tf.dtypes.int64  
)

Defined in [python/ops/ragged/ragged\_math\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/ragged_math_ops.py).

Each row of the returned RaggedTensor contains a single sequence:

ragged.range(starts, limits, deltas)[i] ==  
    tf.range(starts[i], limits[i], deltas[i])

If start[i] < limits[i] and deltas[i] > 0, then output[i] will be an empty list. Similarly, if start[i] > limits[i] and deltas[i] < 0, then output[i] will be an empty list. This behavior is consistent with the Python range function, but differs from the [tf.range](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/range) op, which returns an error for these cases.

#### Examples:

>>> ragged.range([3, 5, 2]).eval().tolist()  
[[0, 1, 2], [0, 1, 2, 3, 4], [0, 1]]  
>>> ragged.range([0, 5, 8], [3, 3, 12]).eval().tolist()  
[[0, 1, 2], [], [8, 9, 10, 11]]  
>>> ragged.range([0, 5, 8], [3, 3, 12], 2).eval().tolist()  
[[0, 2], [], [8, 10]]

The input tensors starts, limits, and deltas may be scalars or vectors. The vector inputs must all have the same size. Scalar inputs are broadcast to match the size of the vector inputs.

#### Args:

* **starts**: Vector or scalar Tensor. Specifies the first entry for each range if limits is not None; otherwise, specifies the range limits, and the first entries default to 0.
* **limits**: Vector or scalar Tensor. Specifies the exclusive upper limits for each range.
* **deltas**: Vector or scalar Tensor. Specifies the increment for each range. Defaults to 1.
* **dtype**: The type of the elements of the resulting tensor. If not specified, then a value is chosen based on the other args.
* **name**: A name for the operation.
* **row\_splits\_dtype**: dtype for the returned RaggedTensor's row\_splits tensor. One of [tf.int32](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int32) or [tf.int64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int64).

#### Returns:

A RaggedTensor of type dtype with ragged\_rank=1.

# tf.ragged.row\_splits\_to\_segment\_ids

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

Generates the segmentation corresponding to a RaggedTensor row\_splits.

### Aliases:

* tf.compat.v1.ragged.row\_splits\_to\_segment\_ids
* tf.compat.v2.ragged.row\_splits\_to\_segment\_ids
* tf.ragged.row\_splits\_to\_segment\_ids

tf.ragged.row\_splits\_to\_segment\_ids(  
    splits,  
    name=None,  
    out\_type=None  
)

Defined in [python/ops/ragged/segment\_id\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/segment_id_ops.py).

Returns an integer vector segment\_ids, where segment\_ids[i] == j if splits[j] <= i < splits[j+1]. Example:

>>> ragged.row\_splits\_to\_segment\_ids([0, 3, 3, 5, 6, 9]).eval()  
[ 0 0 0 2 2 3 4 4 4 ]

#### Args:

* **splits**: A sorted 1-D integer Tensor. splits[0] must be zero.
* **name**: A name prefix for the returned tensor (optional).
* **out\_type**: The dtype for the return value. Defaults to splits.dtype, or [tf.int64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int64) if splitsdoes not have a dtype.

#### Returns:

A sorted 1-D integer Tensor, with shape=[splits[-1]]

#### Raises:

* **ValueError**: If splits is invalid.

# tf.ragged.segment\_ids\_to\_row\_splits

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

Generates the RaggedTensor row\_splits corresponding to a segmentation.

### Aliases:

* tf.compat.v1.ragged.segment\_ids\_to\_row\_splits
* tf.compat.v2.ragged.segment\_ids\_to\_row\_splits
* tf.ragged.segment\_ids\_to\_row\_splits

tf.ragged.segment\_ids\_to\_row\_splits(  
    segment\_ids,  
    num\_segments=None,  
    out\_type=None,  
    name=None  
)

Defined in [python/ops/ragged/segment\_id\_ops.py](https://github.com/tensorflow/tensorflow/tree/r2.0/tensorflow/python/ops/ragged/segment_id_ops.py).

Returns an integer vector splits, where splits[0] = 0 and splits[i] = splits[i-1] + count(segment\_ids==i). Example:

>>> ragged.segment\_ids\_to\_row\_splits([0, 0, 0, 2, 2, 3, 4, 4, 4]).eval()  
[ 0 3 3 5 6 9 ]

#### Args:

* **segment\_ids**: A 1-D integer Tensor.
* **num\_segments**: A scalar integer indicating the number of segments. Defaults to max(segment\_ids) + 1 (or zero if segment\_ids is empty).
* **out\_type**: The dtype for the return value. Defaults to segment\_ids.dtype, or [tf.int64](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf#int64) if segment\_ids does not have a dtype.
* **name**: A name prefix for the returned tensor (optional).

#### Returns:

A sorted 1-D integer Tensor, with shape=[num\_segments + 1].