Module: tf.quantization

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

Public API for tf.quantization namespace.

Functions

[dequantize(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/dequantize): Dequantize the 'input' tensor into a float Tensor.

[fake\_quant\_with\_min\_max\_args(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/fake_quant_with_min_max_args): Fake-quantize the 'inputs' tensor, type float to 'outputs' tensor of same type.

[fake\_quant\_with\_min\_max\_args\_gradient(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/fake_quant_with_min_max_args_gradient): Compute gradients for a FakeQuantWithMinMaxArgs operation.

[fake\_quant\_with\_min\_max\_vars(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/fake_quant_with_min_max_vars): Fake-quantize the 'inputs' tensor of type float via global float scalars min

[fake\_quant\_with\_min\_max\_vars\_gradient(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/fake_quant_with_min_max_vars_gradient): Compute gradients for a FakeQuantWithMinMaxVars operation.

[fake\_quant\_with\_min\_max\_vars\_per\_channel(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/fake_quant_with_min_max_vars_per_channel): Fake-quantize the 'inputs' tensor of type float and one of the shapes: [d],

[fake\_quant\_with\_min\_max\_vars\_per\_channel\_gradient(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/fake_quant_with_min_max_vars_per_channel_gradient): Compute gradients for a FakeQuantWithMinMaxVarsPerChannel operation.

[quantize(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/quantize): Quantize the 'input' tensor of type float to 'output' tensor of type 'T'.

[quantize\_and\_dequantize(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/quantize_and_dequantize): Quantizes then dequantizes a tensor.

[quantized\_concat(...)](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/quantization/quantized_concat): Concatenates quantized tensors along one dimension.

# tf.quantization.dequantize

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

Dequantize the 'input' tensor into a float Tensor.

### Aliases:

* tf.compat.v1.dequantize
* tf.compat.v1.quantization.dequantize
* tf.compat.v2.quantization.dequantize
* tf.quantization.dequantize

tf.quantization.dequantize(  
    input,  
    min\_range,  
    max\_range,  
    mode='MIN\_COMBINED',  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

[min\_range, max\_range] are scalar floats that specify the range for the 'input' data. The 'mode' attribute controls exactly which calculations are used to convert the float values to their quantized equivalents.

In 'MIN\_COMBINED' mode, each value of the tensor will undergo the following:

if T == qint8: in[i] += (range(T) + 1)/ 2.0  
out[i] = min\_range + (in[i]\* (max\_range - min\_range) / range(T))

here range(T) = numeric\_limits<T>::max() - numeric\_limits<T>::min()

MIN\_COMBINED Mode Example

If the input comes from a QuantizedRelu6, the output type is quint8 (range of 0-255) but the possible range of QuantizedRelu6 is 0-6. The min\_range and max\_range values are therefore 0.0 and 6.0. Dequantize on quint8 will take each value, cast to float, and multiply by 6 / 255. Note that if quantizedtype is qint8, the operation will additionally add each value by 128 prior to casting.

If the mode is 'MIN\_FIRST', then this approach is used:

num\_discrete\_values = 1 << (# of bits in T)  
range\_adjust = num\_discrete\_values / (num\_discrete\_values - 1)  
range = (range\_max - range\_min) \* range\_adjust  
range\_scale = range / num\_discrete\_values  
const double offset\_input = static\_cast<double>(input) - lowest\_quantized;  
result = range\_min + ((input - numeric\_limits<T>::min()) \* range\_scale)

SCALED mode Example

SCALED mode matches the quantization approach used in QuantizeAndDequantize{V2|V3}.

If the mode is SCALED, we do not use the full range of the output type, choosing to elide the lowest possible value for symmetry (e.g., output range is -127 to 127, not -128 to 127 for signed 8 bit quantization), so that 0.0 maps to 0.

We first find the range of values in our tensor. The range we use is always centered on 0, so we find m such that

  m = max(abs(input\_min), abs(input\_max))

Our input tensor range is then [-m, m].

Next, we choose our fixed-point quantization buckets, [min\_fixed, max\_fixed]. If T is signed, this is

  num\_bits = sizeof(T) \* 8  
  [min\_fixed, max\_fixed] =  
      [-(1 << (num\_bits - 1) - 1), (1 << (num\_bits - 1)) - 1]

Otherwise, if T is unsigned, the fixed-point range is

  [min\_fixed, max\_fixed] = [0, (1 << num\_bits) - 1]

From this we compute our scaling factor, s:

  s = (2 \* m) / (max\_fixed - min\_fixed)

Now we can dequantize the elements of our tensor:

result = input \* s

#### Args:

* **input**: A Tensor. Must be one of the following types: qint8, quint8, qint32, qint16, quint16.
* **min\_range**: A Tensor of type float32. The minimum scalar value possibly produced for the input.
* **max\_range**: A Tensor of type float32. The maximum scalar value possibly produced for the input.
* **mode**: An optional string from: "MIN\_COMBINED", "MIN\_FIRST", "SCALED". Defaults to "MIN\_COMBINED".
* **name**: A name for the operation (optional).

#### Returns:

A Tensor of type float32.

# tf.quantization.fake\_quant\_with\_min\_max\_args

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

Fake-quantize the 'inputs' tensor, type float to 'outputs' tensor of same type.

### Aliases:

* tf.compat.v1.fake\_quant\_with\_min\_max\_args
* tf.compat.v1.quantization.fake\_quant\_with\_min\_max\_args
* tf.compat.v2.quantization.fake\_quant\_with\_min\_max\_args
* tf.quantization.fake\_quant\_with\_min\_max\_args

tf.quantization.fake\_quant\_with\_min\_max\_args(  
    inputs,  
    min=-6,  
    max=6,  
    num\_bits=8,  
    narrow\_range=False,  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

Attributes [min; max] define the clamping range for the inputs data. inputs values are quantized into the quantization range ([0; 2^num\_bits - 1] when narrow\_range is false and [1; 2^num\_bits - 1] when it is true) and then de-quantized and output as floats in [min; max] interval.num\_bits is the bitwidth of the quantization; between 2 and 16, inclusive.

Before quantization, min and max values are adjusted with the following logic. It is suggested to have min <= 0 <= max. If 0 is not in the range of values, the behavior can be unexpected: If 0 < min < max: min\_adj = 0 and max\_adj = max - min. If min < max < 0: min\_adj = min - max and max\_adj = 0. If min <= 0 <= max: scale = (max - min) / (2^num\_bits - 1), min\_adj = scale \* round(min / scale) and max\_adj = max + min\_adj - min.

Quantization is called fake since the output is still in floating point.

#### Args:

* **inputs**: A Tensor of type float32.
* **min**: An optional float. Defaults to -6.
* **max**: An optional float. Defaults to 6.
* **num\_bits**: An optional int. Defaults to 8.
* **narrow\_range**: An optional bool. Defaults to False.
* **name**: A name for the operation (optional).

#### Returns:

A Tensor of type float32.

# tf.quantization.fake\_quant\_with\_min\_max\_args\_gradient

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

Compute gradients for a FakeQuantWithMinMaxArgs operation.

### Aliases:

* tf.compat.v1.fake\_quant\_with\_min\_max\_args\_gradient
* tf.compat.v1.quantization.fake\_quant\_with\_min\_max\_args\_gradient
* tf.compat.v2.quantization.fake\_quant\_with\_min\_max\_args\_gradient
* tf.quantization.fake\_quant\_with\_min\_max\_args\_gradient

tf.quantization.fake\_quant\_with\_min\_max\_args\_gradient(  
    gradients,  
    inputs,  
    min=-6,  
    max=6,  
    num\_bits=8,  
    narrow\_range=False,  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

#### Args:

* **gradients**: A Tensor of type float32. Backpropagated gradients above the FakeQuantWithMinMaxArgs operation.
* **inputs**: A Tensor of type float32. Values passed as inputs to the FakeQuantWithMinMaxArgs operation.
* **min**: An optional float. Defaults to -6.
* **max**: An optional float. Defaults to 6.
* **num\_bits**: An optional int. Defaults to 8.
* **narrow\_range**: An optional bool. Defaults to False.
* **name**: A name for the operation (optional).

#### Returns:

A Tensor of type float32.

# tf.quantization.fake\_quant\_with\_min\_max\_vars

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

Fake-quantize the 'inputs' tensor of type float via global float scalars min

### Aliases:

* tf.compat.v1.fake\_quant\_with\_min\_max\_vars
* tf.compat.v1.quantization.fake\_quant\_with\_min\_max\_vars
* tf.compat.v2.quantization.fake\_quant\_with\_min\_max\_vars
* tf.quantization.fake\_quant\_with\_min\_max\_vars

tf.quantization.fake\_quant\_with\_min\_max\_vars(  
    inputs,  
    min,  
    max,  
    num\_bits=8,  
    narrow\_range=False,  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

and max to 'outputs' tensor of same shape as inputs.

[min; max] define the clamping range for the inputs data. inputs values are quantized into the quantization range ([0; 2^num\_bits - 1] when narrow\_range is false and [1; 2^num\_bits - 1] when it is true) and then de-quantized and output as floats in [min; max] interval. num\_bits is the bitwidth of the quantization; between 2 and 16, inclusive.

Before quantization, min and max values are adjusted with the following logic. It is suggested to have min <= 0 <= max. If 0 is not in the range of values, the behavior can be unexpected: If 0 < min < max: min\_adj = 0 and max\_adj = max - min. If min < max < 0: min\_adj = min - max and max\_adj = 0. If min <= 0 <= max: scale = (max - min) / (2^num\_bits - 1), min\_adj = scale \* round(min / scale) and max\_adj = max + min\_adj - min.

This operation has a gradient and thus allows for training min and max values.

#### Args:

* **inputs**: A Tensor of type float32.
* **min**: A Tensor of type float32.
* **max**: A Tensor of type float32.
* **num\_bits**: An optional int. Defaults to 8.
* **narrow\_range**: An optional bool. Defaults to False.
* **name**: A name for the operation (optional).

#### Returns:

A Tensor of type float32.

# tf.quantization.fake\_quant\_with\_min\_max\_vars\_gradient

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

Compute gradients for a FakeQuantWithMinMaxVars operation.

### Aliases:

* tf.compat.v1.fake\_quant\_with\_min\_max\_vars\_gradient
* tf.compat.v1.quantization.fake\_quant\_with\_min\_max\_vars\_gradient
* tf.compat.v2.quantization.fake\_quant\_with\_min\_max\_vars\_gradient
* tf.quantization.fake\_quant\_with\_min\_max\_vars\_gradient

tf.quantization.fake\_quant\_with\_min\_max\_vars\_gradient(  
    gradients,  
    inputs,  
    min,  
    max,  
    num\_bits=8,  
    narrow\_range=False,  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

#### Args:

* **gradients**: A Tensor of type float32. Backpropagated gradients above the FakeQuantWithMinMaxVars operation.
* **inputs**: A Tensor of type float32. Values passed as inputs to the FakeQuantWithMinMaxVars operation. min, max: Quantization interval, scalar floats.
* **min**: A Tensor of type float32.
* **max**: A Tensor of type float32.
* **num\_bits**: An optional int. Defaults to 8. The bitwidth of the quantization; between 2 and 8, inclusive.
* **narrow\_range**: An optional bool. Defaults to False. Whether to quantize into 2^num\_bits - 1 distinct values.
* **name**: A name for the operation (optional).

#### Returns:

A tuple of Tensor objects (backprops\_wrt\_input, backprop\_wrt\_min, backprop\_wrt\_max).

* **backprops\_wrt\_input**: A Tensor of type float32.
* **backprop\_wrt\_min**: A Tensor of type float32.
* **backprop\_wrt\_max**: A Tensor of type float32.

# tf.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel

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

Fake-quantize the 'inputs' tensor of type float and one of the shapes: [d],

### Aliases:

* tf.compat.v1.fake\_quant\_with\_min\_max\_vars\_per\_channel
* tf.compat.v1.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel
* tf.compat.v2.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel
* tf.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel

tf.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel(  
    inputs,  
    min,  
    max,  
    num\_bits=8,  
    narrow\_range=False,  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

[b, d] [b, h, w, d] via per-channel floats min and max of shape [d] to 'outputs' tensor of same shape as inputs.

[min; max] define the clamping range for the inputs data. inputs values are quantized into the quantization range ([0; 2^num\_bits - 1] when narrow\_range is false and [1; 2^num\_bits - 1] when it is true) and then de-quantized and output as floats in [min; max] interval. num\_bits is the bitwidth of the quantization; between 2 and 16, inclusive.

Before quantization, min and max values are adjusted with the following logic. It is suggested to have min <= 0 <= max. If 0 is not in the range of values, the behavior can be unexpected: If 0 < min < max: min\_adj = 0 and max\_adj = max - min. If min < max < 0: min\_adj = min - max and max\_adj = 0. If min <= 0 <= max: scale = (max - min) / (2^num\_bits - 1), min\_adj = scale \* round(min / scale) and max\_adj = max + min\_adj - min.

This operation has a gradient and thus allows for training min and max values.

#### Args:

* **inputs**: A Tensor of type float32.
* **min**: A Tensor of type float32.
* **max**: A Tensor of type float32.
* **num\_bits**: An optional int. Defaults to 8.
* **narrow\_range**: An optional bool. Defaults to False.
* **name**: A name for the operation (optional).

#### Returns:

A Tensor of type float32.

# tf.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel

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

Fake-quantize the 'inputs' tensor of type float and one of the shapes: [d],

### Aliases:

* tf.compat.v1.fake\_quant\_with\_min\_max\_vars\_per\_channel
* tf.compat.v1.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel
* tf.compat.v2.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel
* tf.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel

tf.quantization.fake\_quant\_with\_min\_max\_vars\_per\_channel(  
    inputs,  
    min,  
    max,  
    num\_bits=8,  
    narrow\_range=False,  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

[b, d] [b, h, w, d] via per-channel floats min and max of shape [d] to 'outputs' tensor of same shape as inputs.

[min; max] define the clamping range for the inputs data. inputs values are quantized into the quantization range ([0; 2^num\_bits - 1] when narrow\_range is false and [1; 2^num\_bits - 1] when it is true) and then de-quantized and output as floats in [min; max] interval. num\_bits is the bitwidth of the quantization; between 2 and 16, inclusive.

Before quantization, min and max values are adjusted with the following logic. It is suggested to have min <= 0 <= max. If 0 is not in the range of values, the behavior can be unexpected: If 0 < min < max: min\_adj = 0 and max\_adj = max - min. If min < max < 0: min\_adj = min - max and max\_adj = 0. If min <= 0 <= max: scale = (max - min) / (2^num\_bits - 1), min\_adj = scale \* round(min / scale) and max\_adj = max + min\_adj - min.

This operation has a gradient and thus allows for training min and max values.

#### Args:

* **inputs**: A Tensor of type float32.
* **min**: A Tensor of type float32.
* **max**: A Tensor of type float32.
* **num\_bits**: An optional int. Defaults to 8.
* **narrow\_range**: An optional bool. Defaults to False.
* **name**: A name for the operation (optional).

#### Returns:

A Tensor of type float32.

# tf.quantization.quantize

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

Quantize the 'input' tensor of type float to 'output' tensor of type 'T'.

### Aliases:

* tf.compat.v1.quantization.quantize
* tf.compat.v1.quantize
* tf.compat.v2.quantization.quantize
* tf.quantization.quantize

tf.quantization.quantize(  
    input,  
    min\_range,  
    max\_range,  
    T,  
    mode='MIN\_COMBINED',  
    round\_mode='HALF\_AWAY\_FROM\_ZERO',  
    name=None  
)

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

[min\_range, max\_range] are scalar floats that specify the range for the 'input' data. The 'mode' attribute controls exactly which calculations are used to convert the float values to their quantized equivalents. The 'round\_mode' attribute controls which rounding tie-breaking algorithm is used when rounding float values to their quantized equivalents.

In 'MIN\_COMBINED' mode, each value of the tensor will undergo the following:

out[i] = (in[i] - min\_range) \* range(T) / (max\_range - min\_range)  
if T == qint8: out[i] -= (range(T) + 1) / 2.0

here range(T) = numeric\_limits<T>::max() - numeric\_limits<T>::min()

MIN\_COMBINED Mode Example

Assume the input is type float and has a possible range of [0.0, 6.0] and the output type is quint8 ([0, 255]). The min\_range and max\_range values should be specified as 0.0 and 6.0. Quantizing from float to quint8 will multiply each value of the input by 255/6 and cast to quint8.

If the output type was qint8 ([-128, 127]), the operation will additionally subtract each value by 128 prior to casting, so that the range of values aligns with the range of qint8.

If the mode is 'MIN\_FIRST', then this approach is used:

num\_discrete\_values = 1 << (# of bits in T)  
range\_adjust = num\_discrete\_values / (num\_discrete\_values - 1)  
range = (range\_max - range\_min) \* range\_adjust  
range\_scale = num\_discrete\_values / range  
quantized = round(input \* range\_scale) - round(range\_min \* range\_scale) +  
  numeric\_limits<T>::min()  
quantized = max(quantized, numeric\_limits<T>::min())  
quantized = min(quantized, numeric\_limits<T>::max())

The biggest difference between this and MIN\_COMBINED is that the minimum range is rounded first, before it's subtracted from the rounded value. With MIN\_COMBINED, a small bias is introduced where repeated iterations of quantizing and dequantizing will introduce a larger and larger error.

SCALED mode Example

SCALED mode matches the quantization approach used in QuantizeAndDequantize{V2|V3}.

If the mode is SCALED, we do not use the full range of the output type, choosing to elide the lowest possible value for symmetry (e.g., output range is -127 to 127, not -128 to 127 for signed 8 bit quantization), so that 0.0 maps to 0.

We first find the range of values in our tensor. The range we use is always centered on 0, so we find m such that

  m = max(abs(input\_min), abs(input\_max))

Our input tensor range is then [-m, m].

Next, we choose our fixed-point quantization buckets, [min\_fixed, max\_fixed]. If T is signed, this is

  num\_bits = sizeof(T) \* 8  
  [min\_fixed, max\_fixed] =  
      [-(1 << (num\_bits - 1) - 1), (1 << (num\_bits - 1)) - 1]

Otherwise, if T is unsigned, the fixed-point range is

  [min\_fixed, max\_fixed] = [0, (1 << num\_bits) - 1]

From this we compute our scaling factor, s:

  s = (max\_fixed - min\_fixed) / (2 \* m)

Now we can quantize the elements of our tensor:

result = round(input \* s)

One thing to watch out for is that the operator may choose to adjust the requested minimum and maximum values slightly during the quantization process, so you should always use the output ports as the range for further calculations. For example, if the requested minimum and maximum values are close to equal, they will be separated by a small epsilon value to prevent ill-formed quantized buffers from being created. Otherwise, you can end up with buffers where all the quantized values map to the same float value, which causes problems for operations that have to perform further calculations on them.

#### Args:

* **input**: A Tensor of type float32.
* **min\_range**: A Tensor of type float32. The minimum scalar value possibly produced for the input.
* **max\_range**: A Tensor of type float32. The maximum scalar value possibly produced for the input.
* **T**: A [tf.DType](https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/dtypes/DType) from: tf.qint8, tf.quint8, tf.qint32, tf.qint16, tf.quint16.
* **mode**: An optional string from: "MIN\_COMBINED", "MIN\_FIRST", "SCALED". Defaults to "MIN\_COMBINED".
* **round\_mode**: An optional string from: "HALF\_AWAY\_FROM\_ZERO", "HALF\_TO\_EVEN". Defaults to "HALF\_AWAY\_FROM\_ZERO".
* **name**: A name for the operation (optional).

#### Returns:

A tuple of Tensor objects (output, output\_min, output\_max).

* **output**: A Tensor of type T.
* **output\_min**: A Tensor of type float32.
* **output\_max**: A Tensor of type float32.

# tf.quantization.quantized\_concat

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

Concatenates quantized tensors along one dimension.

### Aliases:

* tf.compat.v1.quantization.quantized\_concat
* tf.compat.v1.quantized\_concat
* tf.compat.v2.quantization.quantized\_concat
* tf.quantization.quantized\_concat

tf.quantization.quantized\_concat(  
    concat\_dim,  
    values,  
    input\_mins,  
    input\_maxes,  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

#### Args:

* **concat\_dim**: A Tensor of type int32. 0-D. The dimension along which to concatenate. Must be in the range [0, rank(values)).
* **values**: A list of at least 2 Tensor objects with the same type. The N Tensors to concatenate. Their ranks and types must match, and their sizes must match in all dimensions except concat\_dim.
* **input\_mins**: A list with the same length as values of Tensor objects with type float32. The minimum scalar values for each of the input tensors.
* **input\_maxes**: A list with the same length as values of Tensor objects with type float32. The maximum scalar values for each of the input tensors.
* **name**: A name for the operation (optional).

#### Returns:

A tuple of Tensor objects (output, output\_min, output\_max).

* **output**: A Tensor. Has the same type as values.
* **output\_min**: A Tensor of type float32.
* **output\_max**: A Tensor of type float32.

# tf.quantization.quantize\_and\_dequantize

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

Quantizes then dequantizes a tensor.

### Aliases:

* tf.compat.v1.quantization.quantize\_and\_dequantize
* tf.compat.v2.quantization.quantize\_and\_dequantize
* tf.quantization.quantize\_and\_dequantize

tf.quantization.quantize\_and\_dequantize(  
    input,  
    input\_min,  
    input\_max,  
    signed\_input=True,  
    num\_bits=8,  
    range\_given=False,  
    round\_mode='HALF\_TO\_EVEN',  
    name=None  
)

Defined in generated file: python/ops/gen\_array\_ops.py.

This op simulates the precision loss from the quantized forward pass by:

1. Quantizing the tensor to fixed point numbers, which should match the target quantization method when it is used in inference.
2. Dequantizing it back to floating point numbers for the following ops, most likely matmul.

There are different ways to quantize. This version uses only scaling, so 0.0 maps to 0.

From the specified 'num\_bits' in the quantized output type, it determines minimum and maximum representable quantized values.

e.g.

* [-128, 127] for signed, num\_bits = 8, or
* [0, 255] for unsigned, num\_bits = 8.

If range\_given == False, the initial input\_min, input\_max will be determined automatically as the minimum and maximum values in the input tensor, otherwise the specified values of input\_min, input\_max are used.

**Note:** If the input\_min, input\_max are specified, they do not need to equal the actual minimum and maximum values in the tensor. e.g. in some cases it may be beneficial to specify these values such that the low probability extremes of the input distribution are clipped.

This op determines the maximum scale\_factor that would map the initial [input\_min, input\_max] range to a range that lies within the representable quantized range.

It determines the scale from one of input\_min and input\_max, then updates the other one to maximize the respresentable range.

e.g.

* if the output is signed, num\_bits = 8, [input\_min, input\_max] = [-10.0, 5.0]: it would use a scale\_factor of -128 / -10.0 = 12.8 In this case, it would update input\_max to be 127 / 12.8 = 9.921875
* if the output is signed, num\_bits = 8, [input\_min, input\_max] = [-10.0, 10.0]: it would use a scale\_factor of 127 / 10.0 = 12.7 In this case, it would update input\_min to be 128.0 / 12.7 = -10.07874
* if the output is unsigned, input\_min is forced to be 0, and only the specified input\_max is used.

After determining the scale\_factor and updating the input range, it applies the following to each value in the 'input' tensor.

output = round(clamp(value, input\_min, input\_max) \* scale\_factor) / scale\_factor.

The above round function rounds the value based on the given round\_mode.

#### Args:

* **input**: A Tensor. Must be one of the following types: bfloat16, half, float32, float64. Tensor to quantize and then dequantize.
* **input\_min**: A Tensor. Must have the same type as input. If range\_given == True, this specifies the minimum input value that needs to be represented, otherwise it is determined from the min value of the input tensor.
* **input\_max**: A Tensor. Must have the same type as input. If range\_given == True, this specifies the maximum input value that needs to be represented, otherwise it is determined from the max value of the input tensor.
* **signed\_input**: An optional bool. Defaults to True. Whether the quantization is signed or unsigned. (actually this parameter should have been called **signed\_output**)
* **num\_bits**: An optional int. Defaults to 8. The bitwidth of the quantization.
* **range\_given**: An optional bool. Defaults to False. Whether the range is given or should be determined from the input tensor.
* **round\_mode**: An optional string from: "HALF\_TO\_EVEN", "HALF\_UP". Defaults to "HALF\_TO\_EVEN". The 'round\_mode' attribute controls which rounding tie-breaking algorithm is used when rounding float values to their quantized equivalents. The following rounding modes are currently supported:
  + HALF\_TO\_EVEN: this is the default round\_mode.
  + HALF\_UP: round towards positive. In this mode 7.5 rounds up to 8 and -7.5 rounds up to -7.
* **name**: A name for the operation (optional).

#### Returns:

A Tensor. Has the same type as input.