tf.quantize_v2

```
quantize_v2(
    input,
    min_range,
    max_range,
    T,
    mode='MIN_COMBINED',
    name=None
)
```

Defined in `tensorflow/python/ops/gen_array_ops.py`.

See the guide: Tensor Transformations > Slicing and Joining

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

[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:

```
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:

```
number_of_steps = 1 << (# of bits in T)
range_adjust = number_of_steps / (number_of_steps - 1)
range = (range_max - range_min) * range_adjust
range_scale = number_of_steps / 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 = (input * s).round_to_nearest()
```

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` from: `tf.qint8, tf.quint8, tf.qint16, tf.quint16, tf.qint32`.
- `mode`: An optional `string` from: `"MIN_COMBINED", "MIN_FIRST", "SCALED"`. Defaults to `"MIN_COMBINED"`.
- `name`: A name for the operation (optional).

Returns:

A tuple of `Tensor` objects (output, output_min, output_max).

- `output`: A `Tensor` of type `T`. The quantized data produced from the float input.
- `output_min`: A `Tensor` of type `float32`. The actual minimum scalar value used for the output.
- `output_max`: A `Tensor` of type `float32`. The actual maximum scalar value used for the output.

Last updated November 2, 2017.