tf.nn.static_bidirectional_rnn

Aliases:

- `tf.contrib.rnn.static_bidirectional_rnn`
- `tf.nn.static_bidirectional_rnn`

```
static_bidirectional_rnn(
    cell_fw,
    cell_bw,
    inputs,
    initial_state_fw=None,
    initial_state_bw=None,
    dtype=None,
    sequence_length=None,
    scope=None
)
```

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

See the guide: RNN and Cells (contrib) > Recurrent Neural Networks

Creates a bidirectional recurrent neural network.

Similar to the unidirectional case above (rnn) but takes input and builds independent forward and backward RNNs with the final forward and backward outputs depth-concatenated, such that the output will have the format [time][batch][cell_fw.output_size + cell_bw.output_size]. The input_size of forward and backward cell must match. The initial state for both directions is zero by default (but can be set optionally) and no intermediate states are ever returned – the network is fully unrolled for the given (passed in) length(s) of the sequence(s) or completely unrolled if length(s) is not given.

Args:

- `cell_fw`: An instance of RNNCell, to be used for forward direction.
- `cell_bw`: An instance of RNNCell, to be used for backward direction.
- `inputs`: A length T list of inputs, each a tensor of shape [batch_size, input_size], or a nested tuple of such elements.
- `initial_state_fw`: (optional) An initial state for the forward RNN. This must be a tensor of appropriate type and shape `[batch_size, cell_fw.state_size]`. If `cell_fw.state_size` is a tuple, this should be a tuple of tensors having shapes `[batch_size, s] for s in cell_fw.state_size`.
- `initial_state_bw`: (optional) Same as for `initial_state_fw`, but using the corresponding properties of `cell_bw`.
- `dtype`: (optional) The data type for the initial state. Required if either of the initial states are not provided.
- `sequence_length`: (optional) An int32/int64 vector, size `[batch_size]`, containing the actual lengths for each of the sequences.
- `scope`: VariableScope for the created subgraph; defaults to "bidirectional_rnn"

Returns:

A tuple (outputs, output_state_fw, output_state_bw) where: outputs is a length `T` list of outputs (one for each input), which are depth-concatenated forward and backward outputs. output_state_fw is the final state of the forward rnn. output_state_bw is the final state of the backward rnn.

Raises:

- `TypeError`: If `cell_fw` or `cell_bw` is not an instance of `RNNCell`.
- `ValueError`: If inputs is None or an empty list.