tf.contrib.rnn.LSTMBlockWrapper

Class `LSTMBlockWrapper`

Inherits From: `FusedRNNCell`

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

See the guide: RNN and Cells (contrib) > Core RNN Cell wrappers (RNNCells that wrap other RNNCells)

This is a helper class that provides housekeeping for LSTM cells.

This may be useful for alternative LSTM and similar type of cells. The subclasses must implement `_call_cell` method and `num_units` property.

Properties

`num_units`

Number of units in this cell (output dimension).

Methods

`__call__`

```
__call__(
    inputs,
    initial_state=None,
    dtype=None,
    sequence_length=None,
    scope=None
)
```

Run this LSTM on inputs, starting from the given state.

Args:

- `inputs`: `3-D` tensor with shape `[time_len, batch_size, input_size]` or a list of `time_len` tensors of shape `[batch_size, input_size]`.
- `initial_state`: a tuple `(initial_cell_state, initial_output)` with tensors of shape `[batch_size,`

`self._num_units]`. If this is not provided, the cell is expected to create a zero initial state of type `dtype`.

- `dtype`: The data type for the initial state and expected output. Required if `initial_state` is not provided or RNN state has a heterogeneous dtype.
- `sequence_length`: Specifies the length of each sequence in inputs. An `int32` or `int64` vector (tensor) size `[batch_size]`, values in `[0, time_len).` Defaults to `time_len` for each element.
- `scope`: `VariableScope` for the created subgraph; defaults to class name.

Returns:

A pair containing:

- Output: A `3-D` tensor of shape `[time_len, batch_size, output_size]` or a list of time_len tensors of shape `[batch_size, output_size]`, to match the type of the `inputs`.
- Final state: a tuple `(cell_state, output)` matching `initial_state`.

Raises:

- `ValueError`: in case of shape mismatches
