tf.stack

```
stack(
    values,
    axis=0,
    name='stack'
)
```

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

See the guides: Layers (contrib) > Higher level ops for building neural network layers, Tensor Transformations > Slicing and Joining

Stacks a list of rank-`R` tensors into one rank-`(R+1)` tensor.

Packs the list of tensors in `values` into a tensor with rank one higher than each tensor in `values`, by packing them along the `axis` dimension. Given a list of length `N` of tensors of shape `(A, B, C)`;

if `axis == 0` then the `output` tensor will have the shape `(N, A, B, C)`. if `axis == 1` then the `output` tensor will have the shape `(A, N, B, C)`. Etc.

For example:

```
x = tf.constant([1, 4])
y = tf.constant([2, 5])
z = tf.constant([3, 6])
tf.stack([x, y, z])  # [[1, 4], [2, 5], [3, 6]] (Pack along first dim.)
tf.stack([x, y, z], axis=1)  # [[1, 2, 3], [4, 5, 6]]
```

This is the opposite of unstack. The numpy equivalent is

```
tf.stack([x, y, z]) = np.stack([x, y, z])
```

Args:

- `values`: A list of `Tensor` objects with the same shape and type.
- `axis`: An `int`. The axis to stack along. Defaults to the first dimension. Negative values wrap around, so the valid range is `[-(R+1), R+1)`.
- `name`: A name for this operation (optional).

Returns:

- `output`: A stacked `Tensor` with the same type as `values`.

Raises:

- `ValueError`: If `axis` is out of the range [-(R+1), R+1).

Last updated November 2, 2017.