# tf.contrib.layers.conv2d



## Aliases:

- `tf.contrib.layers.conv2d`
- `tf.contrib.layers.conv3d`
- `tf.contrib.layers.convolution2d`
- `tf.contrib.layers.convolution3d`

```
conv2d(
    inputs,
    num_outputs,
    kernel_size,
    stride=1,
    padding='SAME',
    data_format=None,
    rate=1,
    activation_fn=tf.nn.relu,
    normalizer_fn=None,
    normalizer_params=None,
    weights_initializer=initializers.xavier_initializer(),
    weights_regularizer=None,
    biases_initializer=tf.zeros_initializer(),
    biases_regularizer=None,
    reuse=None,
    variables_collections=None,
    outputs_collections=None,
    trainable=True,
    scope=None
)
```

Defined in `tensorflow/contrib/layers/python/layers/layers.py`.

See the guide: Layers (contrib) > Higher level ops for building neural network layers

Adds an N-D convolution followed by an optional batch_norm layer.

It is required that 1 <= N <= 3.

`convolution` creates a variable called `weights`, representing the convolutional kernel, that is convolved (actually cross-correlated) with the `inputs` to produce a `Tensor` of activations. If a `normalizer_fn` is provided (such as `batch_norm`), it is then applied. Otherwise, if `normalizer_fn` is None and a `biases_initializer` is provided then a `biases` variable would be created and added the activations. Finally, if `activation_fn` is not `None`, it is applied to the activations as well.

Performs atrous convolution with input stride/dilation rate equal to `rate` if a value > 1 for any dimension of `rate` is specified. In this case `stride` values != 1 are not supported.

## Args:

- `inputs`: A Tensor of rank N+2 of shape `[batch_size] + input_spatial_shape + [in_channels]` if data_format does not start with "NC" (default), or `[batch_size, in_channels] + input_spatial_shape` if data_format starts with "NC".
- `num_outputs`: Integer, the number of output filters.
- `kernel_size`: A sequence of N positive integers specifying the spatial dimensions of the filters. Can be a single integer to specify the same value for all spatial dimensions.
- `stride`: A sequence of N positive integers specifying the stride at which to compute output. Can be a single integer to specify the same value for all spatial dimensions. Specifying any `stride` value != 1 is incompatible with specifying any `rate` value != 1.
- `padding`: One of `"VALID"` or `"SAME"`.
- `data_format`: A string or None. Specifies whether the channel dimension of the `input` and output is the last dimension (default, or if `data_format` does not start with "NC"), or the second dimension (if `data_format` starts with "NC"). For N=1, the valid values are "NWC" (default) and "NCW". For N=2, the valid values are "NHWC" (default) and "NCHW". For N=3, the valid values are "NDHWC" (default) and "NCDHW".
- `rate`: A sequence of N positive integers specifying the dilation rate to use for atrous convolution. Can be a single integer to specify the same value for all spatial dimensions. Specifying any `rate` value != 1 is incompatible with specifying any `stride` value != 1.
- `activation_fn`: Activation function. The default value is a ReLU function. Explicitly set it to None to skip it and maintain a linear activation.
- `normalizer_fn`: Normalization function to use instead of `biases`. If `normalizer_fn` is provided then `biases_initializer` and `biases_regularizer` are ignored and `biases` are not created nor added. default set to None for no normalizer function
- `normalizer_params`: Normalization function parameters.
- `weights_initializer`: An initializer for the weights.
- `weights_regularizer`: Optional regularizer for the weights.
- `biases_initializer`: An initializer for the biases. If None skip biases.
- `biases_regularizer`: Optional regularizer for the biases.
- `reuse`: Whether or not the layer and its variables should be reused. To be able to reuse the layer scope must be given.
- `variables_collections`: Optional list of collections for all the variables or a dictionary containing a different list of collection per variable.
- `outputs_collections`: Collection to add the outputs.
- `trainable`: If `True` also add variables to the graph collection `GraphKeys.TRAINABLE_VARIABLES` (see tf.Variable).
- `scope`: Optional scope for `variable_scope`.

## Returns:

A tensor representing the output of the operation.

## Raises:

- `ValueError`: If `data_format` is invalid.
- `ValueError`: Both 'rate' and `stride` are not uniformly 1.

*Last updated November 2, 2017.*