tf.layers.conv1d

```
conv1d(
    inputs,
    filters,
    kernel_size,
    strides=1,
    padding='valid',
    data_format='channels_last',
    dilation_rate=1,
    activation=None,
    use_bias=True,
    kernel_initializer=None,
    bias_initializer=tf.zeros_initializer(),
    kernel_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    bias_constraint=None,
    trainable=True,
    name=None,
    reuse=None
)
```

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

Functional interface for 1D convolution layer (e.g. temporal convolution).

This layer creates a convolution kernel that is convolved (actually cross-correlated) with the layer input to produce a tensor of outputs. If `use_bias` is True (and a `bias_initializer` is provided), a bias vector is created and added to the outputs. Finally, if `activation` is not `None`, it is applied to the outputs as well.

Arguments:

- `inputs`: Tensor input.
- `filters`: Integer, the dimensionality of the output space (i.e. the number of filters in the convolution).
- `kernel_size`: An integer or tuple/list of a single integer, specifying the length of the 1D convolution window.
- `strides`: An integer or tuple/list of a single integer, specifying the stride length of the convolution. Specifying any stride value != 1 is incompatible with specifying any `dilation_rate` value != 1.
- `padding`: One of `"valid"` or `"same"` (case-insensitive).
- `data_format`: A string, one of `channels_last` (default) or `channels_first`. The ordering of the dimensions in the inputs. `channels_last` corresponds to inputs with shape `(batch, length, channels)` while `channels_first` corresponds to inputs with shape `(batch, channels, length)`.
- `dilation_rate`: An integer or tuple/list of a single integer, specifying the dilation rate to use for dilated convolution. Currently, specifying any `dilation_rate` value != 1 is incompatible with specifying any `strides` value != 1.
- `activation`: Activation function. Set it to None to maintain a linear activation.
- `use_bias`: Boolean, whether the layer uses a bias.
- `kernel_initializer`: An initializer for the convolution kernel.
- `bias_initializer`: An initializer for the bias vector. If None, no bias will be applied.

- `kernel_regularizer`: Optional regularizer for the convolution kernel.
- `bias_regularizer`: Optional regularizer for the bias vector.
- `activity_regularizer`: Optional regularizer function for the output.
- `kernel_constraint`: Optional projection function to be applied to the kernel after being updated by an `Optimizer` (e.g. used to implement norm constraints or value constraints for layer weights). The function must take as input the unprojected variable and must return the projected variable (which must have the same shape). Constraints are not safe to use when doing asynchronous distributed training.
- `bias_constraint`: Optional projection function to be applied to the bias after being updated by an `Optimizer`.
- `trainable`: Boolean, if `True` also add variables to the graph collection `GraphKeys.TRAINABLE_VARIABLES` (see `tf.Variable`).
- `name`: A string, the name of the layer.
- `reuse`: Boolean, whether to reuse the weights of a previous layer by the same name.

Returns:

Output tensor.

Last updated November 2, 2017.