tf.layers.conv2d_transpose

```
conv2d_transpose(
    inputs,
    filters,
    kernel_size,
    strides=(1, 1),
    padding='valid',
    data_format='channels_last',
    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 transposed 2D convolution layer.

The need for transposed convolutions generally arises from the desire to use a transformation going in the opposite direction of a normal convolution, i.e., from something that has the shape of the output of some convolution to something that has the shape of its input while maintaining a connectivity pattern that is compatible with said convolution.

Arguments:

- `inputs`: Input tensor.
- `filters`: Integer, the dimensionality of the output space (i.e. the number of filters in the convolution).
- `kernel_size`: A tuple or list of 2 positive integers specifying the spatial dimensions of the filters. Can be a single integer to specify the same value for all spatial dimensions.
- `strides`: A tuple or list of 2 positive integers specifying the strides of the convolution. Can be a single integer to specify the same value for all spatial dimensions.
- `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, height, width, channels)` while `channels_first` corresponds to inputs with shape `(batch, channels, height, width)`.
- `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`, then 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.