LocallyConnected1D [source]

```
keras.layers.local.LocallyConnected1D(filters, kernel_size, strides=1, padding='valid', data
```

1次元入力に対応したLocally-connectedレイヤーです.

`LocallyConnected1D` は `Conv1D` と似た動作をします. しかし, 重みが共有されない, つまり入力のパッチごとに異なるフィルタが適用される点が違います.

例

```
# apply a unshared weight convolution 1d of length 3 to a sequence with
# 10 timesteps, with 64 output filters
model = Sequential()
model.add(LocallyConnected1D(64, 3, input_shape=(10, 32)))
# now model.output_shape == (None, 8, 64)
# add a new conv1d on top
model.add(LocallyConnected1D(32, 3))
# now model.output_shape == (None, 6, 32)
```

引数

- **filters**: 整数, 使用するカーネルの数を指定（出力の次元）.
- **kernel_size**: 整数, または一つの整数からなるタプル/リスト. 1次元畳み込みのウィンドウ長を指定します.
- **strides**: 整数, または一つの整数からなるタプル/リスト. 畳み込みのストライド長を指定します. dilation_rate value != 1 とすると, strides value != 1を指定することはできません.
- **padding**: 現在 `"valid"` （大文字, 小文字は区別されない）のみサポートされます. 将来 `"same"` がサポートされる予定です.
- **activation**: 使用する活性化関数の名前（activationsを参照）. 指定がない場合, 活性化は適用されない（つまり"線形"活性 `a(x) = x` となる）.
- **use_bias**: 真理値で, バイアスベクトルを加えるかどうかを指定します.
- **kernel_initializer**: カーネルの重み行列の初期値を指定します. （initializersを参照）
- **bias_initializer**: バイアスベクトルの初期値を指定します. （initializersを参照）.
- **kernel_regularizer**: カーネルの重み行列に適用させるRegularizerを指定します. （regularizerを参照）
- **bias_regularizer**: バイアスベクトルに適用させるRegularizerを指定します. （regularizerを参照）
- **activity_regularizer**: ネットワーク出力（同ネットワークの「活性化」）に適用させるRegularizerを指定します. （regularizerを参照）
- **kernel_constraint**: カーネルの重み行列に適用させるConstraintsを指定します. （constraintsを参照）
- **bias_constraint**: バイアスベクトルに適用させるConstraintsを指定します. （constraintsを参照）

入力のshape

入力は `(batch_size, steps, input_dim)` の3階テンソルです．

出力のshape

出力は `(batch_size, new_steps, filters)` の3階テンソルです． `steps` 値はパディングやストライドにより変わることがあります．

LocallyConnected2D [source]

```
keras.layers.local.LocallyConnected2D(filters, kernel_size, strides=(1, 1), padding='valid',
```

2次元入力に対応したLocally-connectedレイヤーです．

`LocallyConnected2D` は `Conv2D` と似た動作をします．しかし，重みが共有されない，つまり入力のパッチごとに異なるフィルタが適用される点が違います．

例

```
# apply a 3x3 unshared weights convolution with 64 output filters on a 32x32 image
# with `data_format="channels_last"`:
model = Sequential()
model.add(LocallyConnected2D(64, (3, 3), input_shape=(32, 32, 3)))
# now model.output_shape == (None, 30, 30, 64)
# notice that this layer will consume (30*30)*(3*3*3*64) + (30*30)*64 parameters

# add a 3x3 unshared weights convolution on top, with 32 output filters:
model.add(LocallyConnected2D(32, (3, 3)))
# now model.output_shape == (None, 28, 28, 32)
```

引数

- **filters**: 整数，使用するカーネルの数を指定（出力の次元）．
- **kernel_size**: 畳み込みカーネルの幅と高さを指定します．タプル/リストでカーネルの幅と高さをそれぞれ指定でき，整数の場合は正方形のカーネルになります．
- **strides**: カーネルのストライドを指定します. 二つの整数からなるタプル/リストで縦と横のストライドをそれぞれ指定でき，整数の場合は幅と高さが同一のストライドになります．
- **padding**: 現在 `"valid"` （大文字，小文字は区別されない）のみサポートされます．将来 `"same"` がサポートされる予定です．
- **data_format**: `channels_last` （デフォルト）か `channels_first` を指定します． `channels_last` の場合，入力のshapeは `(batch, height, width, channels)` となり， `channels_first` の場合は `(batch, channels, height, width)` となります．デフォルトはKerasの設定ファイル `~/.keras/keras.json` の `image_data_format` の値です．一度も値を変更していなければ， `channels_last` になります．
- **activation**: 使用する活性化関数の名前（activationsを参照）． 指定がない場合，活性化は適用されない（つまり"線形"活性 `a(x) = x` となる）．

- **use_bias**: 真理値で，バイアスベクトルを加えるかどうかを指定します．
- **kernel_initializer**: カーネルの重み行列の初期値を指定します．（initializersを参照）
- **bias_initializer**: バイアスベクトルの初期値を指定します．（initializersを参照）．
- **kernel_regularizer**: カーネルの重み行列に適用させるRegularizerを指定します．（regularizerを参照）
- **bias_regularizer**: バイアスベクトルに適用させるRegularizerを指定します．（regularizerを参照）
- **activity_regularizer**: ネットワーク出力（同ネットワークの「活性化」）に適用させるRegularizerを指定します．（regularizerを参照）
- **kernel_constraint**: カーネルの重み行列に適用させるConstraintsを指定します．（constraintsを参照）
- **bias_constraint**: バイアスベクトルに適用させるConstraintsを指定します．（constraintsを参照）

入力のshape

`data_format='channels_first'`の場合，入力は`(samples, channels, rows, cols)`の4階テンソルです．
`data_format='channels_last'`の場合，入力は`(samples, rows, cols, channels)`の4階テンソルです．

出力のshape

`data_format='channels_first'`の場合，出力は`(samples, filters, new_rows, new_cols)`の4階テンソルです．
`data_format='channels_last'`の場合，出力は`(samples, new_rows, new_cols, filters)`の4階テンソルです．
`rows`と`cols`値はパディングにより変わることがあります．