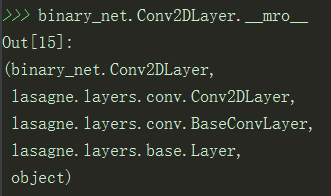
在super机制里可以保证公共父类仅被执行一次，至于执行的顺序，是按照mro进行的（E.\_\_mro\_\_）。  
注意super继承只能用于新式类，用于经典类时就会报错。  
新式类：必须有继承的类，如果没什么想继承的，那就继承object  
经典类：没有父类，如果此时调用super就会出现错误：『super() argument 1 must be type, not classobj』

关于super用法的详细研究可参考「<http://www.jb51.net/article/66912.htm>」



*class* lasagne.layers.Conv2DLayer(*incoming*, *num\_filters*, *filter\_size*, *stride=(1*, *1)*, *pad=0*, *untie\_biases=False*, *W=lasagne.init.GlorotUniform()*, *b=lasagne.init.Constant(0.)*, *nonlinearity=lasagne.nonlinearities.rectify*, *flip\_filters=True*, *convolution=theano.tensor.nnet.conv2d*, *\*\*kwargs*)[[source]](https://github.com/Lasagne/Lasagne/blob/master/lasagne/layers/conv.py#L488-L609)

2D convolutional layer

Performs a 2D convolution on its input and optionally adds a bias and applies an elementwise nonlinearity.

|  |  |
| --- | --- |
| **Parameters:** | **incoming** : a [Layer](http://lasagne.readthedocs.io/en/latest/modules/layers/base.html#lasagne.layers.Layer) instance or a tuple  The layer feeding into this layer, or the expected input shape. The output of this layer should be a 4D tensor, with shape (batch\_size, num\_input\_channels, input\_rows, input\_columns).  *def \_\_init\_\_*(self, *incoming*, *name*=None):  *if* isinstance(*incoming*, tuple):  self.input\_shape = *incoming* self.input\_layer = None  *else*:  self.input\_shape = *incoming*.output\_shape  self.input\_layer = *incoming* self.name = *name* self.params = OrderedDict()  self.get\_output\_kwargs = []   *if* any(d *is not* None *and* d <= 0 *for* d *in* self.input\_shape):  *raise* ValueError((  "Cannot create Layer with a non-positive input\_shape "  "dimension. input\_shape=%r, self.name=%r") % (  self.input\_shape, self.name))  incoming决定的是本层输入的形状。  **num\_filters** : int  The number of learnable convolutional filters this layer has.  **filter\_size** : int or iterable of int  An integer or a 2-element tuple specifying the size of the filters.  **stride** : int or iterable of int  An integer or a 2-element tuple specifying the stride of the convolution operation.  **pad** : int, iterable of int, ‘full’, ‘same’ or ‘valid’ (default: 0)  By default, the convolution is only computed where the input and the filter fully overlap (a valid convolution). When stride=1, this yields an output that is smaller than the input by filter\_size - 1. The *pad* argument allows you to implicitly pad the input with zeros, extending the output size.  A single integer results in symmetric zero-padding of the given size on all borders, a tuple of two integers allows different symmetric padding per dimension.  'full' pads with one less than the filter size on both sides. This is equivalent to computing the convolution wherever the input and the filter overlap by at least one position.  'same' pads with half the filter size (rounded down) on both sides. When stride=1 this results in an output size equal to the input size. Even filter size is not supported.  'valid' is an alias for 0 (no padding / a valid convolution).  Note that 'full' and 'same' can be faster than equivalent integer values due to optimizations by Theano.  **untie\_biases** : bool (default: False)  If False, the layer will have a bias parameter for each channel, which is shared across all positions in this channel. As a result, the *b* attribute will be a vector (1D).  If True, the layer will have separate bias parameters for each position in each channel. As a result, the *b* attribute will be a 3D tensor.  **W** : Theano shared variable, expression, numpy array or callable  Initial value, expression or initializer for the weights. These should be a 4D tensor with shape (num\_filters, num\_input\_channels, filter\_rows, filter\_columns). See [lasagne.utils.create\_param()](http://lasagne.readthedocs.io/en/latest/modules/utils.html#lasagne.utils.create_param) for more information.  **b** : Theano shared variable, expression, numpy array, callable or None  Initial value, expression or initializer for the biases. If set to None, the layer will have no biases. Otherwise, biases should be a 1D array with shape (num\_filters,) if *untied\_biases* is set to False. If it is set to True, its shape should be (num\_filters, output\_rows, output\_columns) instead. See [lasagne.utils.create\_param()](http://lasagne.readthedocs.io/en/latest/modules/utils.html#lasagne.utils.create_param) for more information.  **nonlinearity** : callable or None  The nonlinearity that is applied to the layer activations. If None is provided, the layer will be linear.  **flip\_filters** : bool (default: True)  Whether to flip the filters before sliding them over the input, performing a convolution (this is the default), or not to flip them and perform a correlation. Note that for some other convolutional layers in Lasagne, flipping incurs an overhead and is disabled by default – check the documentation when using learned weights from another layer.  **convolution** : callable  The convolution implementation to use. Usually it should be fine to leave this at the default value.  **\*\*kwargs**  Any additional keyword arguments are passed to the *Layer* superclass. |

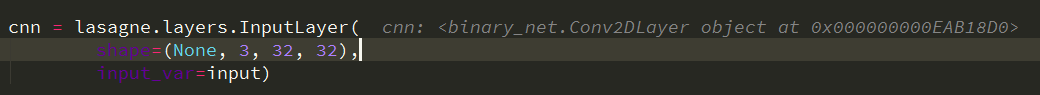
Attributes

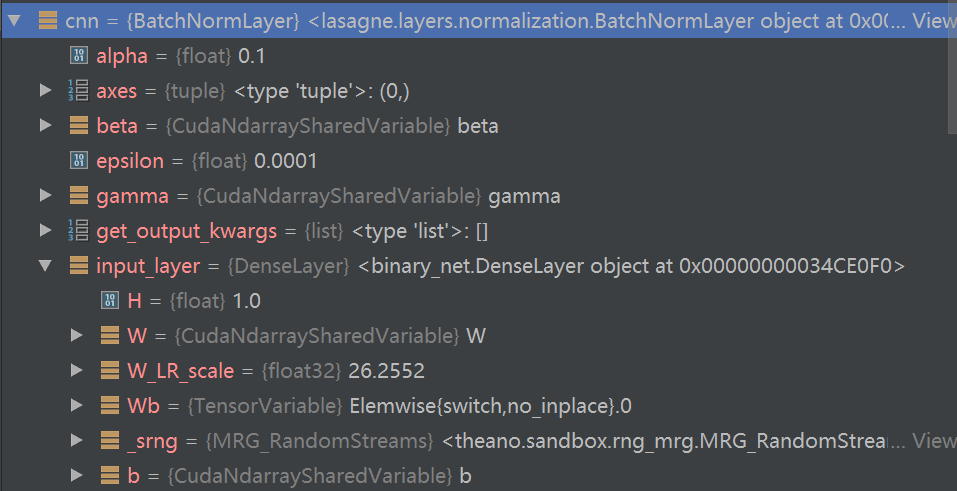
|  |  |
| --- | --- |
| W | (Theano shared variable or expression) Variable or expression representing the filter weights. |
| b | (Theano shared variable or expression) Variable or expression representing the biases. |

Note

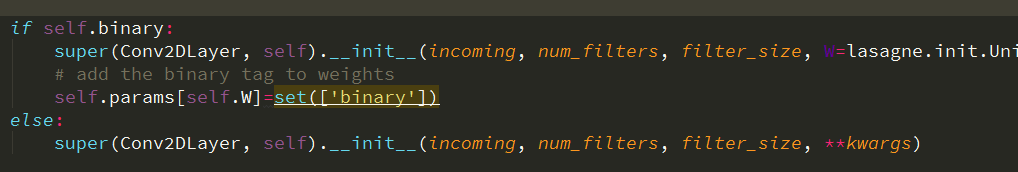
For experts: Conv2DLayer will create a convolutional layer using T.nnet.conv2d, Theano’s default convolution. On compilation for GPU, Theano replaces this with a [cuDNN](https://developer.nvidia.com/cudnn)-based implementation if available, otherwise falls back to a gemm-based implementation. For details on this, please see the [Theano convolution documentation](http://deeplearning.net/software/theano/library/tensor/nnet/conv.html).

Lasagne also provides convolutional layers directly enforcing a specific implementation: [lasagne.layers.dnn.Conv2DDNNLayer](http://lasagne.readthedocs.io/en/latest/modules/layers/dnn.html#lasagne.layers.dnn.Conv2DDNNLayer) to enforce cuDNN, [lasagne.layers.corrmm.Conv2DMMLayer](http://lasagne.readthedocs.io/en/latest/modules/layers/corrmm.html#lasagne.layers.corrmm.Conv2DMMLayer) to enforce the gemm-based one, [lasagne.layers.cuda\_convnet.Conv2DCCLayer](http://lasagne.readthedocs.io/en/latest/modules/layers/cuda_convnet.html#lasagne.layers.cuda_convnet.Conv2DCCLayer) for Krizhevsky’s [cuda-convnet](https://code.google.com/p/cuda-convnet/).

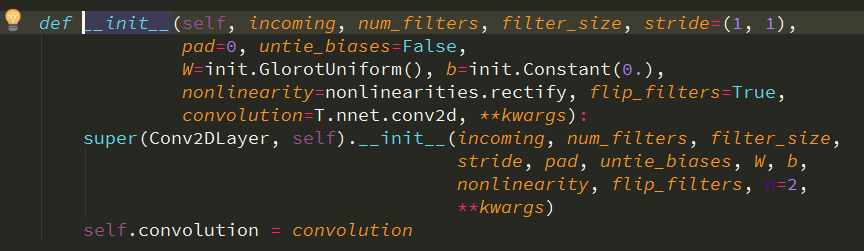




整个网络结构是靠“input\_layer”堆叠起来的。



Incoming传下去



最后归结到这里：

Incoming变成了这个实例的一个属性：input\_layer

