Changes:

Architectural:

Created a class for each layer type (sub of nn.Module): convolutions, activations, batchnorms, dropouts, pooling and upsample. This enabled separate parameter checking in each class and leaving only the structural testing in conv\_block. It will also enable adding functionality and changing behavior per layer.

Parameter testing:

Added what I could think about. All failures raised exceptions, except in the case of adding BN to the block with setting the batch\_norm flag to False. I thought this was done on purpose to allow easy testing, so left it as a warning.

Fixes:

Pooling.py

sending parameters to Max/AvgPool2d was in the wrong order, fixed the order and added parameters names.

convolutions.py

padding calculation in case of not causal and same is wrong. Should be (d\*(k-1))//2

It was also wrong in the causal case, both dimensions should be the same, d\*(k-1) and should consider kernel length. No reason why not to do a similar thing to the non-causal case.

After I fixed the second-dimension padding, there was also a need to fix the trimming of the output on both dimensions in the forward function

Other (better?) solution exists like F.pad for padding only one side of the data.

For conv2d a kaiming\_normal was done as a default, but it broke the code in case separable was True (since Sequential doesn’t have weights), and then it checked for the conv\_init and ignored it. Which was strange.

Batchnorms.py

The weight initialization code was wrong. For BN the default was ‘ones’ and then another initialization was done.

Activations.py

Fixed called to wrong api with leaky relu

Assumptions:

Took into consideration that convolution can be used on it’s own. This led to adding some tests and parameter handling

I assumed tensor type is ‘2d’ so did not put any checks on it

I considered the api as something I could not change, there were things I did not like in it but left it as it is (for example padding should have been padding\_mode). I did remove one unused parameter to convolutions.py

I also assumed groups is always 1 so did not check divisibility of in/out\_channels

Unit testing:

Added two modules both convblock and convolutions

Did not add unit testing for the case of non-square kernels, also not for all activations.

.