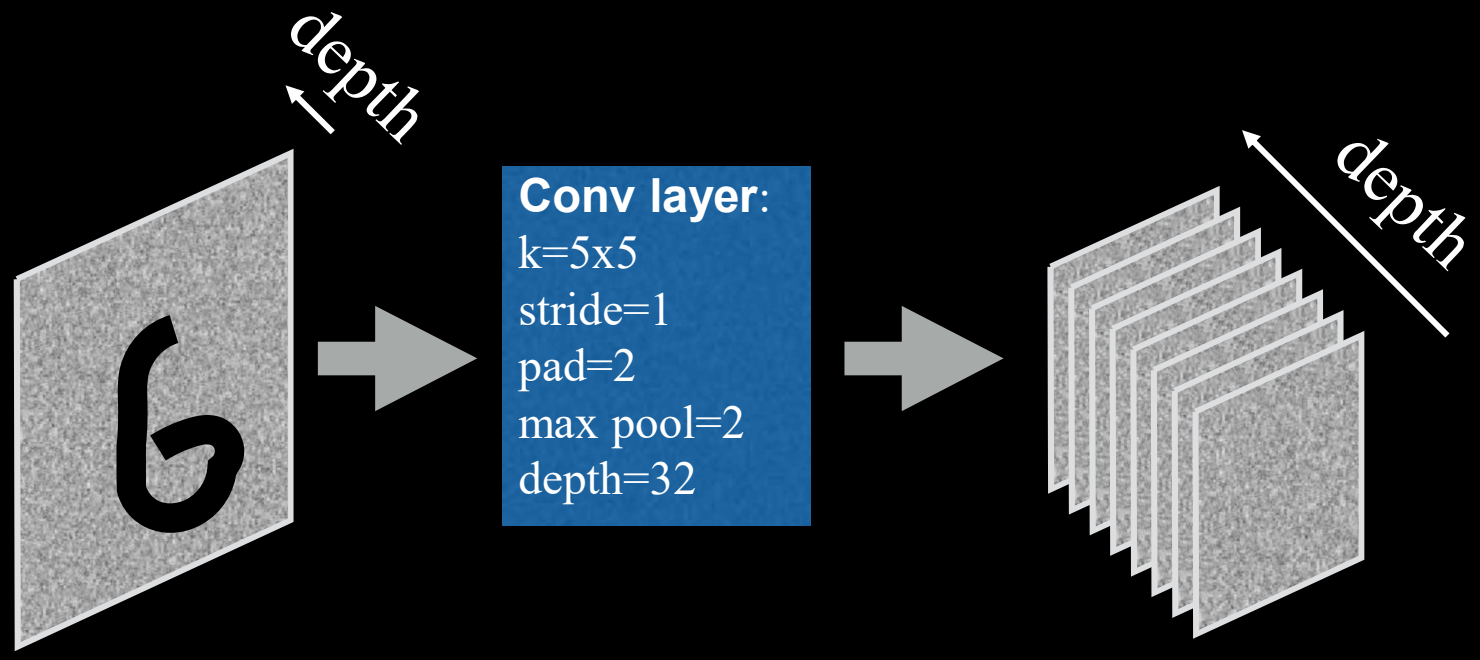
An example deep convolution network

- Input: 28x28 grayscale image
- Output: 10 classes. One output per class.

A Convolutional Net

- Let's assume we have 28x28 grayscale images as input to our conv net. So we will input 28x28x1 samples into the net.
- Let's fix our kernel size at 5x5 and, to make this simple, pad our images with zeros and use a stride = 1.
- Let's use max pooling on the output, with a 2x2 pooling region and a stride of 2.
- Let's extract 32 features after the first layer.
- So the output from this layer will be 14x14x32.

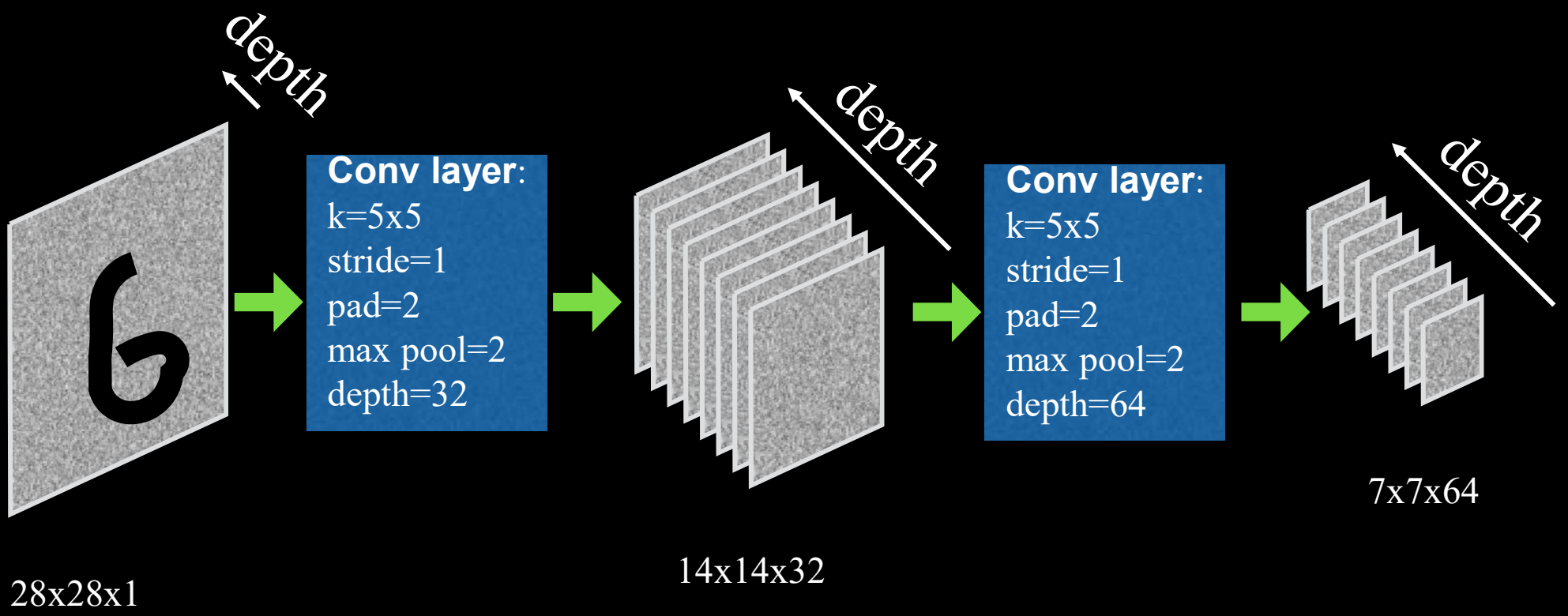
A Convolutional Net



A Convolutional Net

- Now let's make a second layer, also convolutional.
- Let's fix our kernel size at 5x5, pad our images with zeros and use a stride = 1.
- Let's use max pooling on the output again, with a 2x2 pooling region and a stride of 2.
- Let's extract 64 features after the second layer.
- So the output from this layer will be 7x7x64.

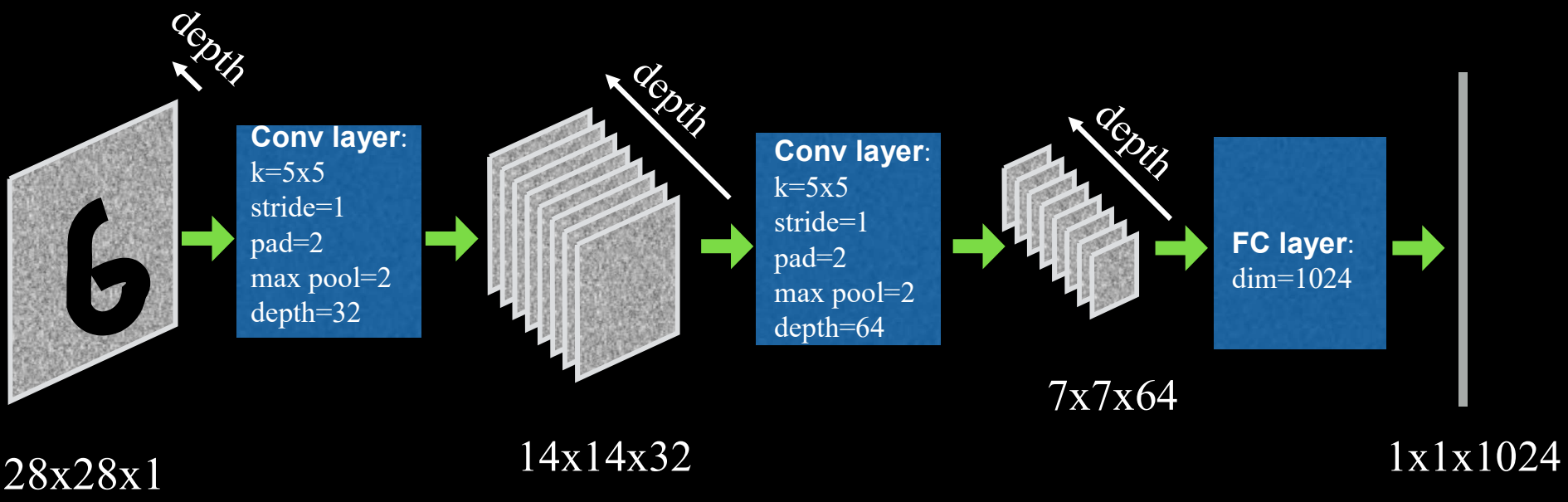
A Convolutional Net



A Convolutional Net

- Our third layer will be a fully connected layer mapping our convolutional features to a 1024 dimensional feature space.
- This layer is just like any of the hidden layers you've seen before. It is a linear transformation followed by ReLU.
- So the output from this layer will be 1x1x1024.

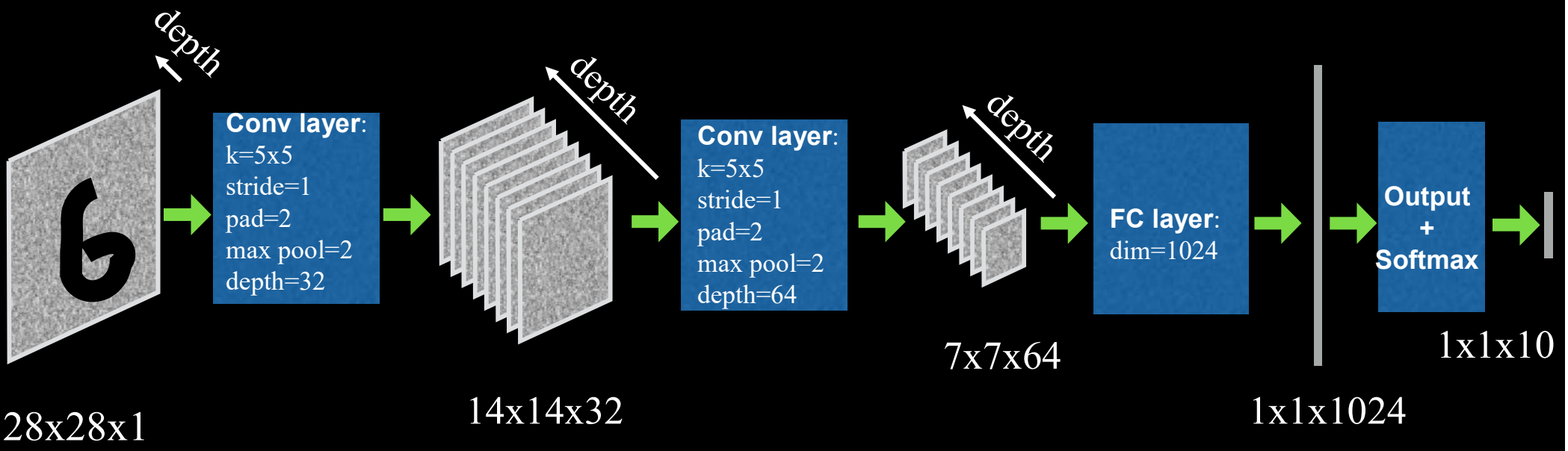
A Convolutional Net



A Convolutional Net

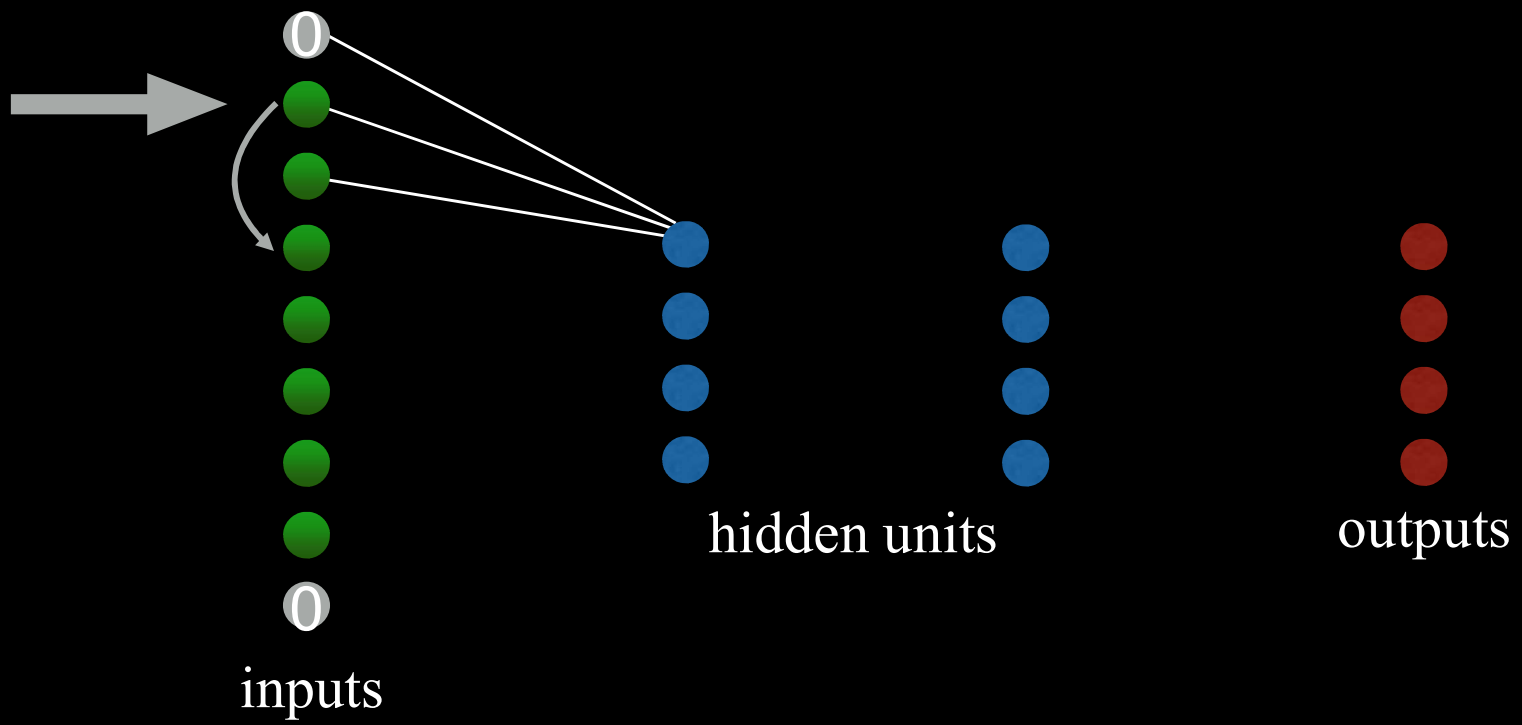
- Finally, we'll map this feature space to a 10 class output space and use a softmax with a MLE/cross entropy loss function.
- And…we're done!

A Convolutional Net



Parameters = (5x5x1x32+32) + (5x5x32x64+64) + (7x7x64x1024+1024) + (1024x10+10)

Convolutional Layer: Padding + Stride



Output dimension = (input dim - kernel size + 2 * padding) / stride + 1