Lec 26: Convolutional Neural Network (CNN, ConvNet)

Ailin Zhang

Agenda

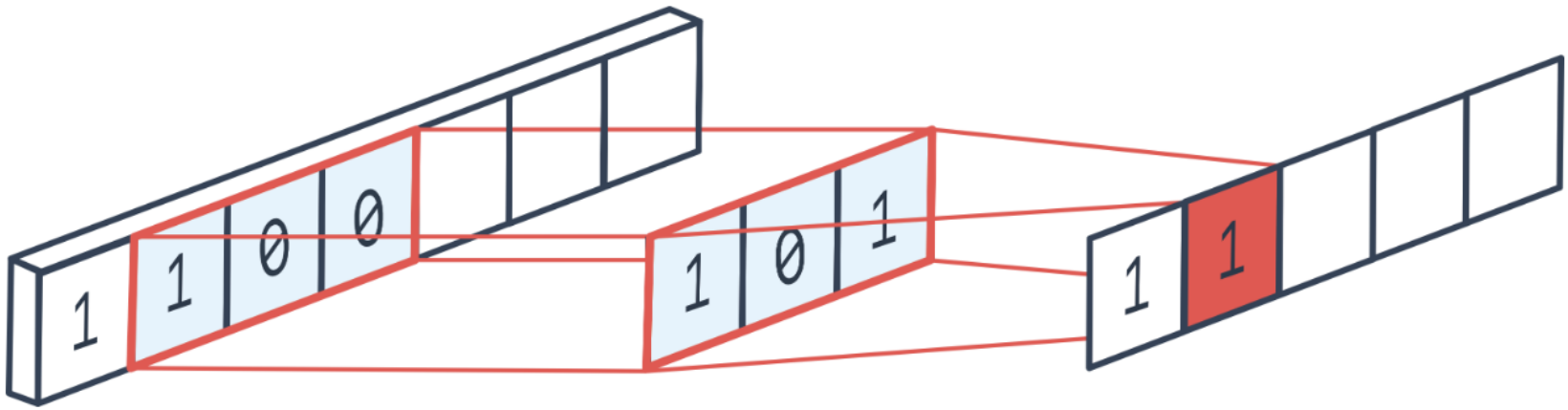
- Convolution, kernels, filters
- Subsampling, Batch Normalization
- Alex net, VGG net, inception net
- Object detection, semantic segmentation, deformable CNN

1D Convolution

- In the neural network, $h_l = f_l(W_l h_{l-1} + b_l)$, the linear transformation $s_l = W_l h_{l-1} + b_l$ that maps h_{l-1} to s_l can be highly structured.
- One important structure is the convolutional neural network, where W_l and b_l have a convolutional structure. (Start from 1d operation)

1D Convolution

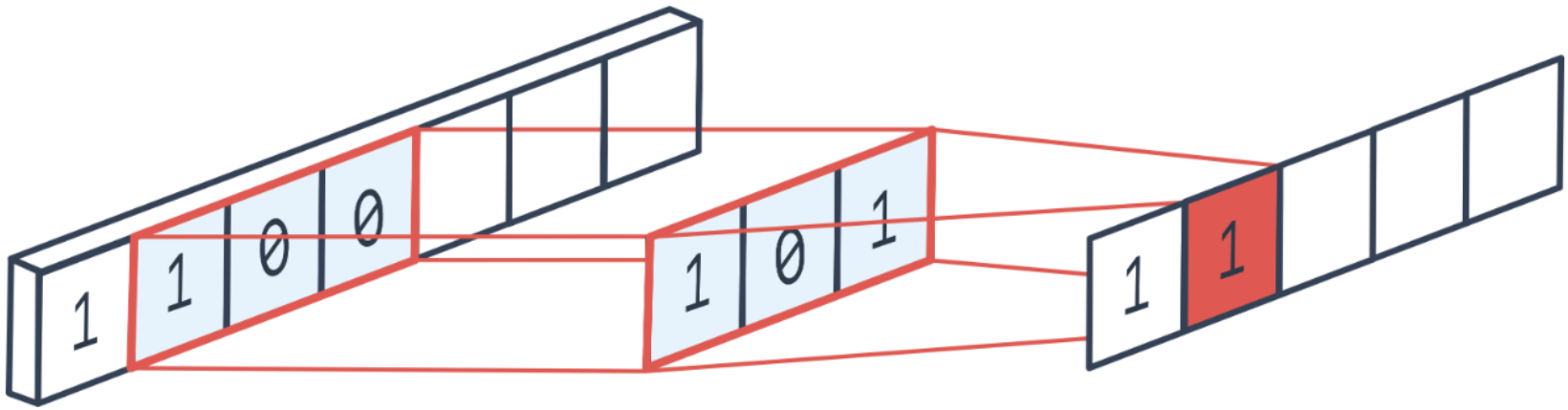
- In the neural network, $h_l = f_l(W_l h_{l-1} + b_l)$, the linear transformation $s_l = W_l h_{l-1} + b_l$ that maps h_{l-1} to s_l can be highly structured.
- One important structure is the convolutional neural network, where W_l and b_l have a convolutional structure. (Start from 1d operation)



Question: How can the output have the same length as the input?

1D Convolution

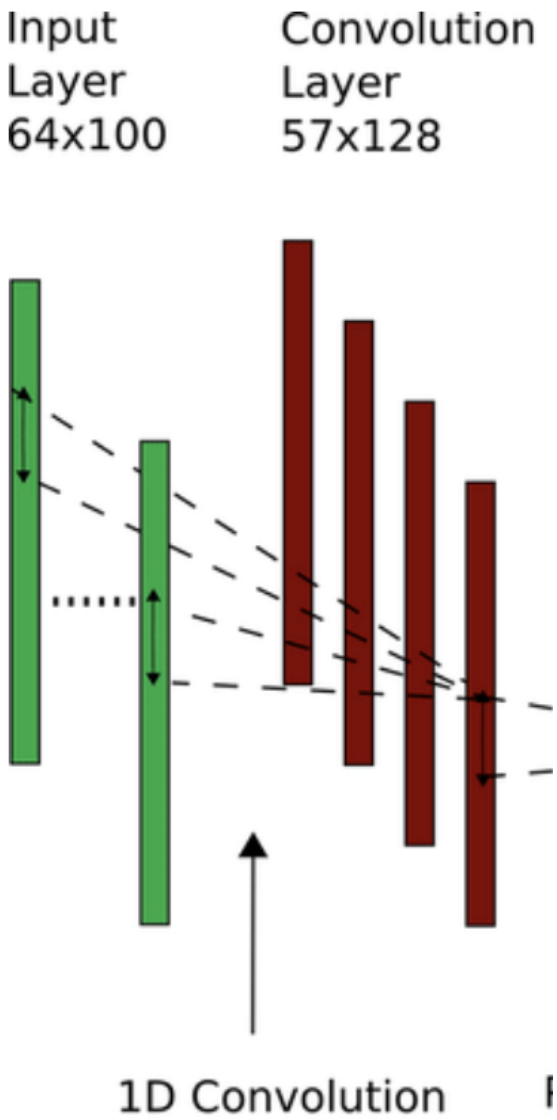
- In the neural network, $h_l = f_l(W_l h_{l-1} + b_l)$, the linear transformation $s_l = W_l h_{l-1} + b_l$ that maps h_{l-1} to s_l can be highly structured.
- One important structure is the convolutional neural network, where W_l and b_l have a convolutional structure. (Start from 1d operation)



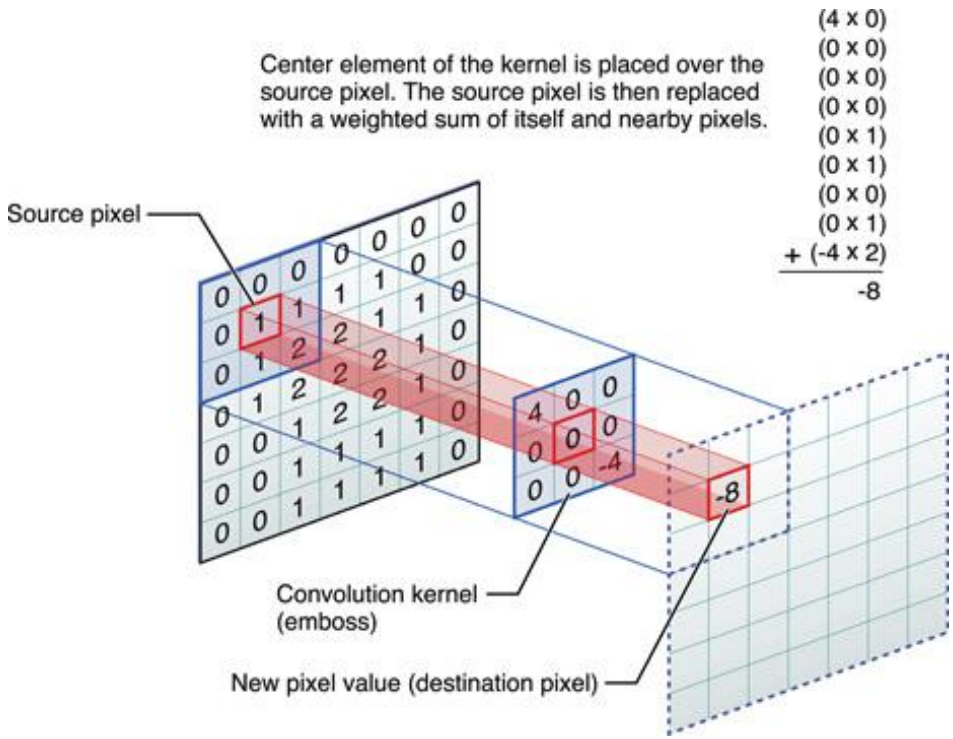
Question: How can the output have the same length as the input?

Zero padding at the boundary

Multiple Channels

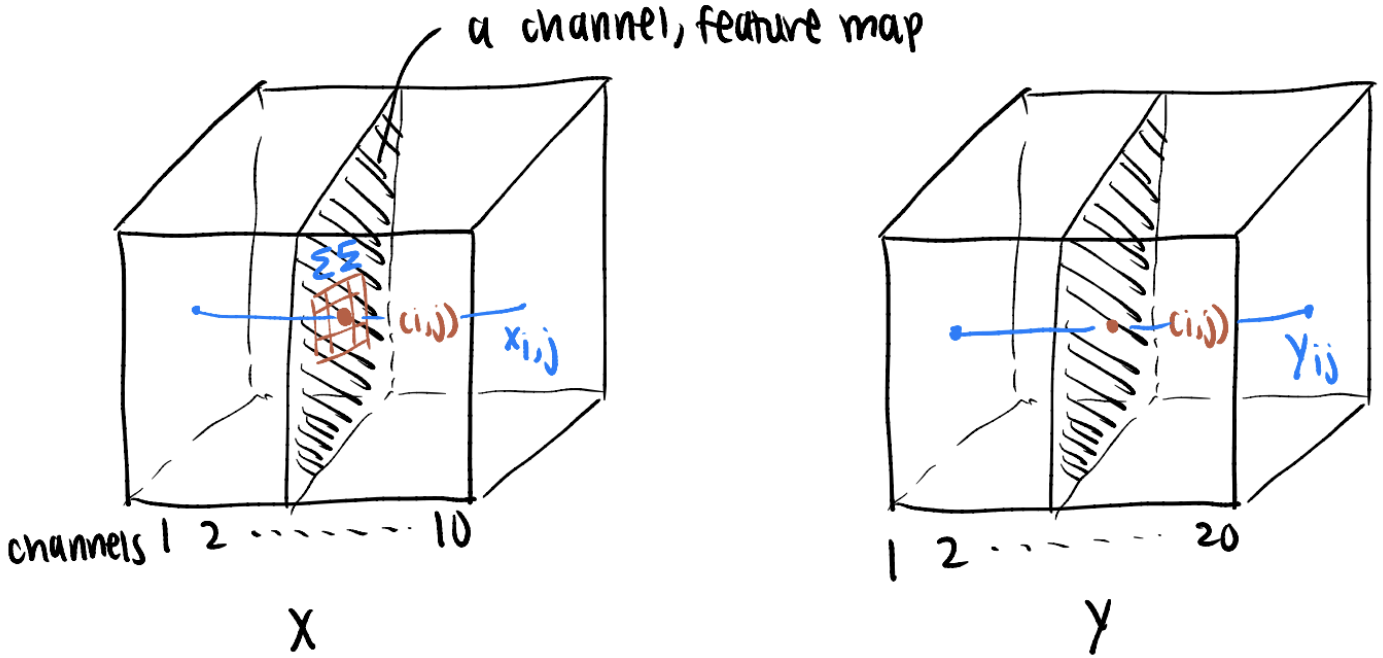


2D Convolution



Local weighted summation. Here the filter or kernel is 3×3. It slides over the whole input image or feature map. At each pixel, we compute the weighted sum of the 3×3 patch of the input image, where the weights are given by the filter or kernel. This gives us an output image or filtered image or feature map.

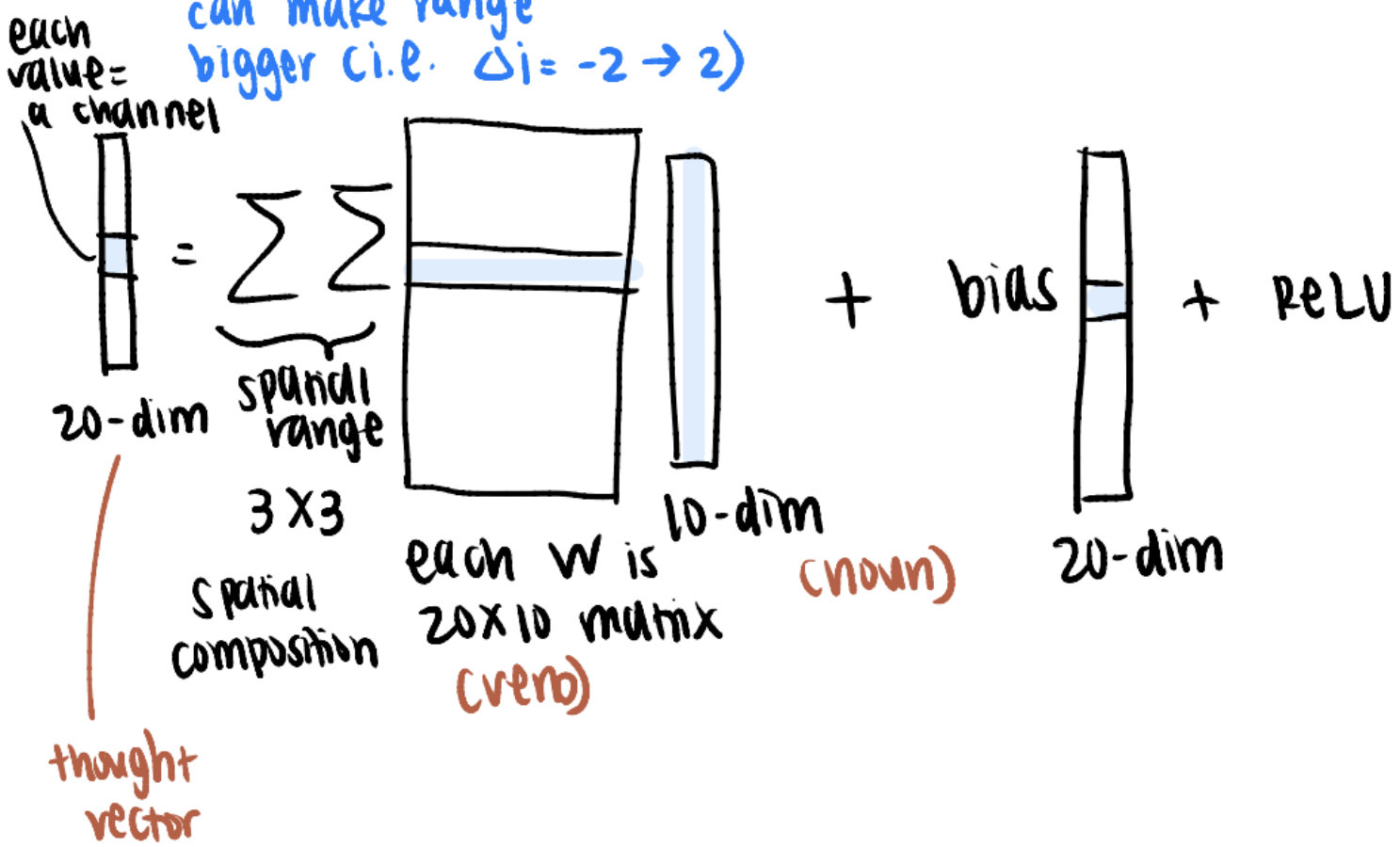
Multi-channels in 2D



$$y_{ij} = \sum_{\Delta i=-1}^{1} \sum_{\Delta j=-1}^{1} w_{\Delta i, \Delta j} x_{i+\Delta i, j+\Delta j}$$

Multi-channels in 2D

$$y_{ij} = \sum_{\Delta i=-1}^{1} \sum_{\Delta j=-1}^{1} w_{\Delta i, \Delta j} \, x_{i+\Delta i, j+\Delta j}$$



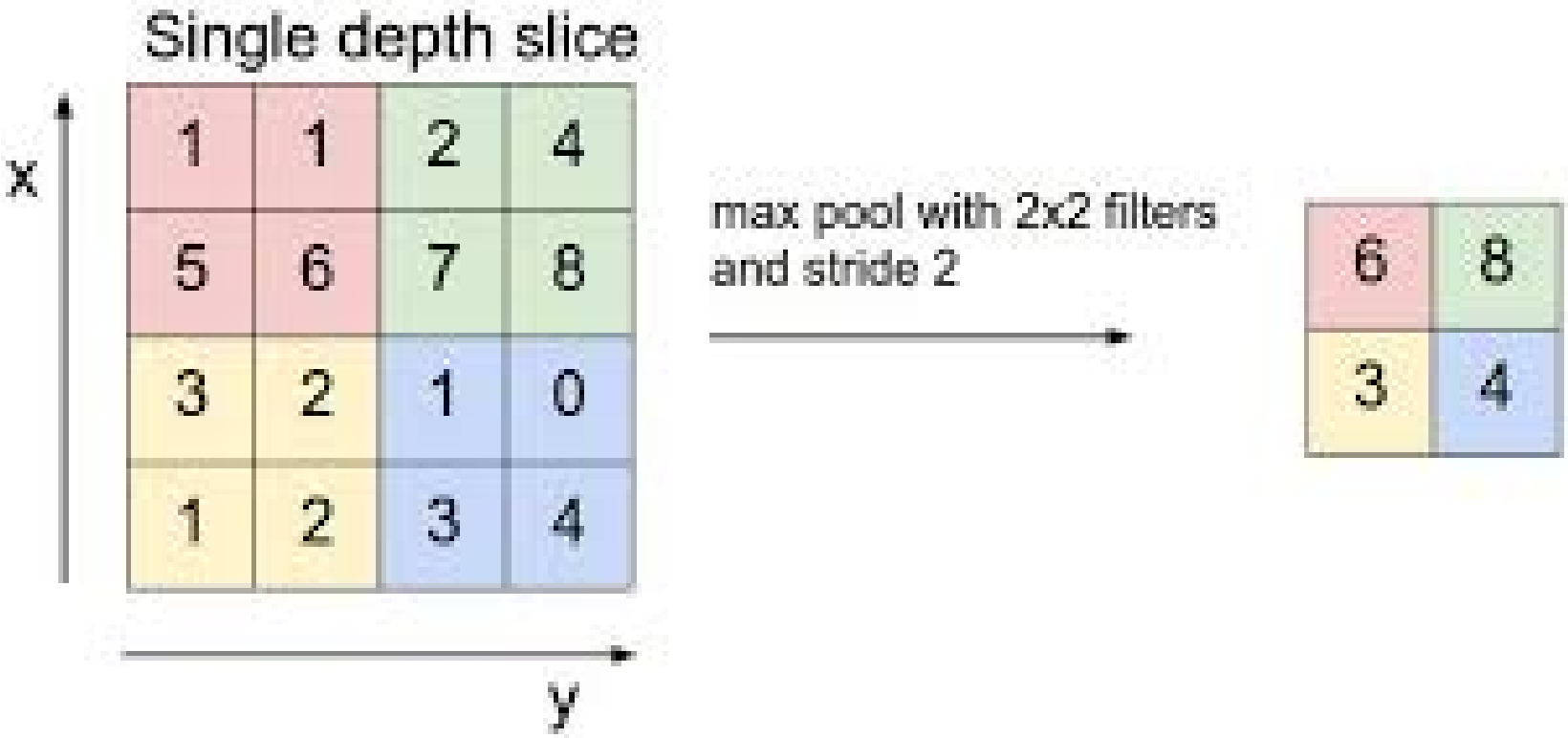
Spatial Arrangement: Depth, Stride, and Zero Padding

- Depth: it corresponds to the number of filters we would like to use, which is also the depth of output volume
- Stride: is the number of pixels that the filters jump as we slide them around
- Zero Padding: pad the input volume with zeros around the border. We will use it to exactly preserve the spatial size of the input volume so the input and output width and height are the same.

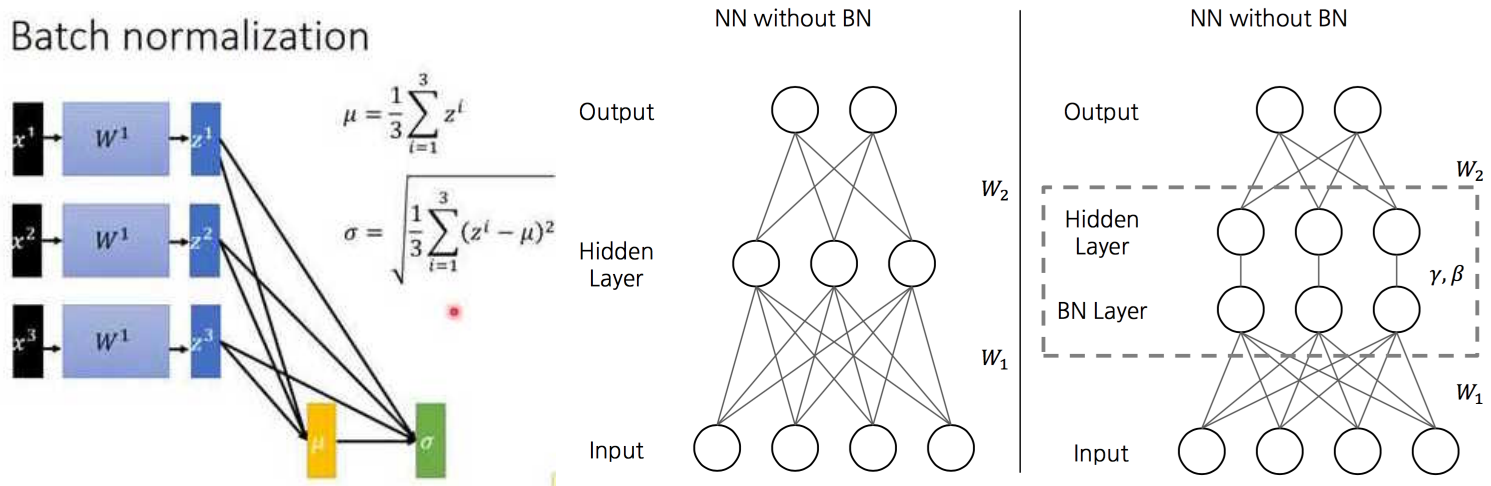
Subsampling

Make a smaller map for faster computation

Question: How to reduce a 4×4 input to a 2×2 output?

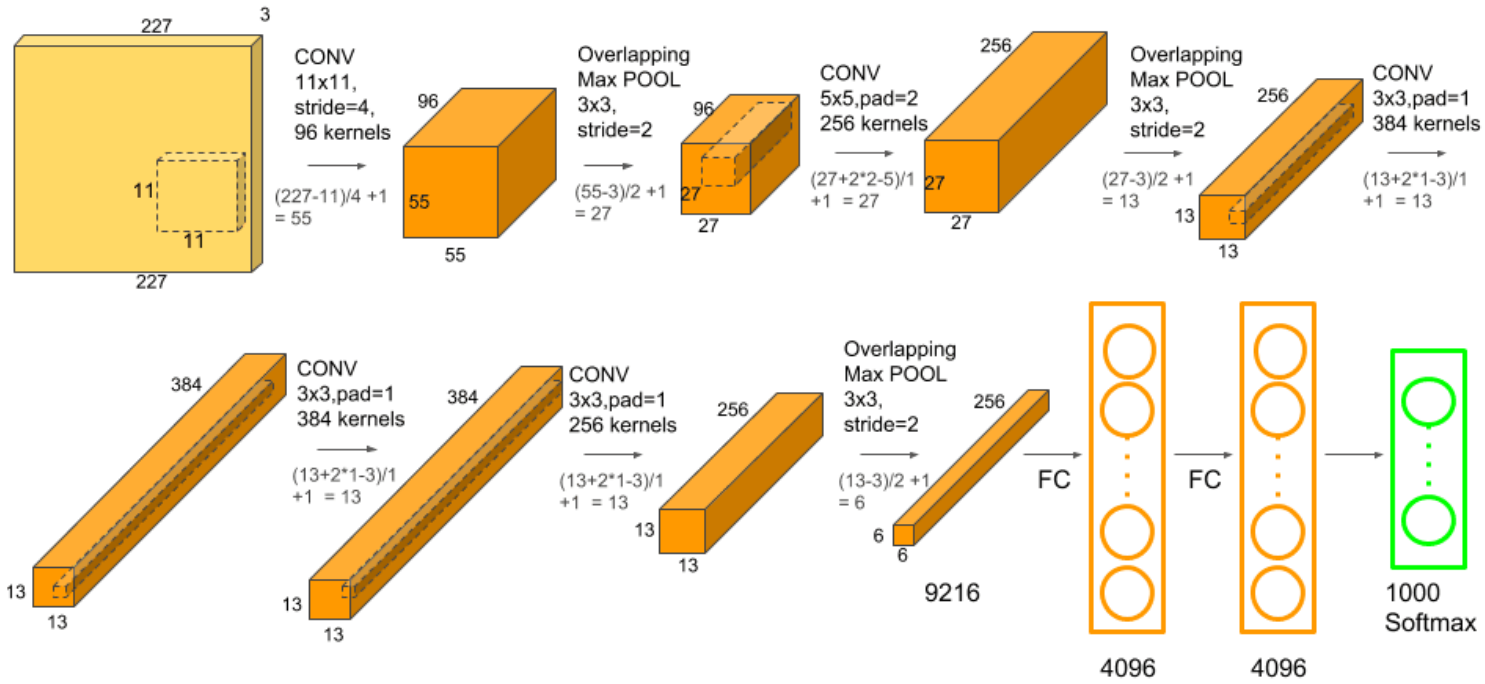


Batch Normalization



- Suppose we have a batch of 3 examples. We compute the mean μ and standard deviation σ by pooling over the 3 examples.
- We then normalize the element for each example, followed by a linear transformation to be learned from the data.
- In back-propagation, we need to treat μ and σ as a layer and compute the derivatives of μ and σ.

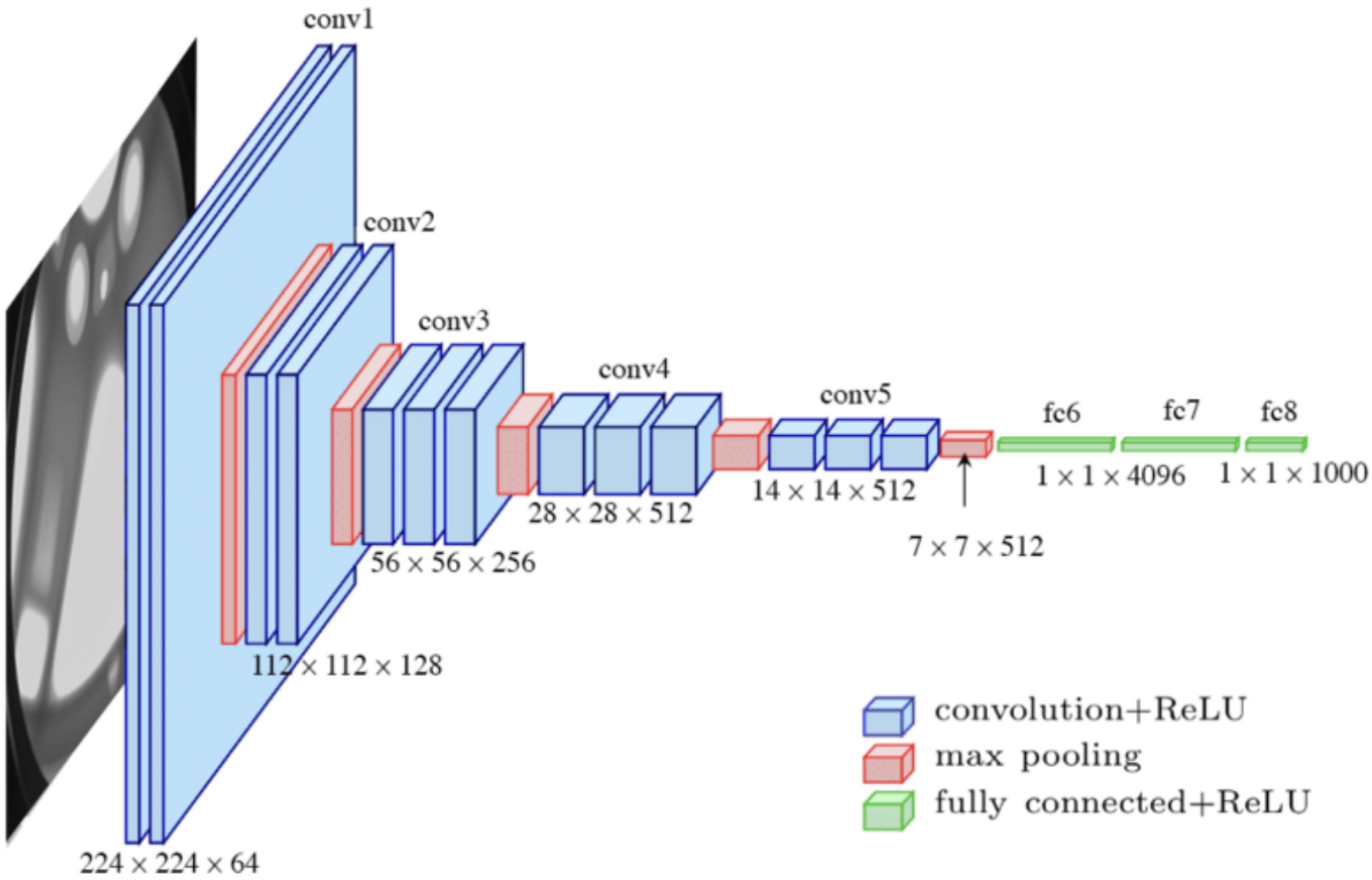
Alex net



It consisted of 11×11, 5×5,3×3 convolutions, max pooling, dropout, data augmentation, ReLU activations, SGD with momentum. The Alex net achieved the initial breakthrough on object recognition for the ImageNet dataset.

VGG net

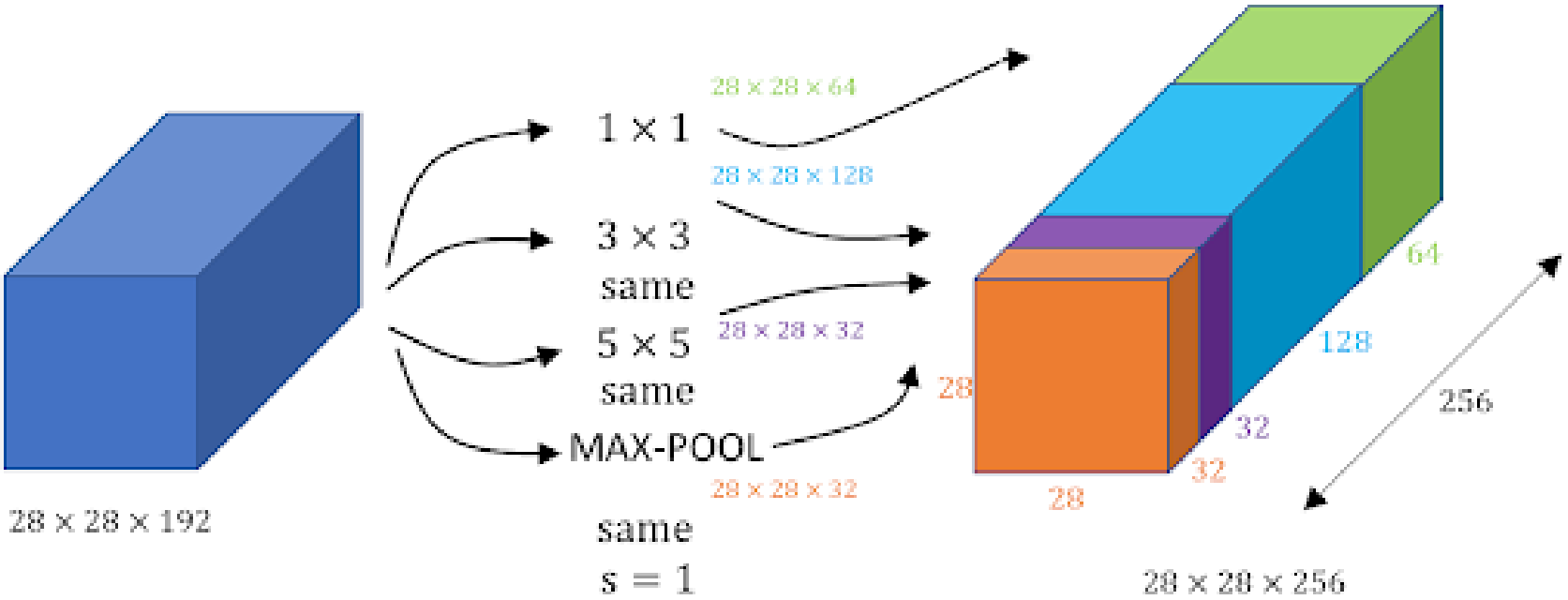
The VGG net is an improvement on the Alex net. There are two versions, VGG16 and VGG19, which consist of 16 hidden layers and 19 hidden layers respectively.



Inception net

- The inception net took its name from the movie: "Inception"
- Instead of picking one of the filter sizes or pooling we want, we can do them all and by the end, we only need to concatenate all the outputs and let the Network learn whatever combinations it wants to use.
- Extensive use of 1×1 filters, i.e., for each feature map in h_l, each pixel value is a weighted summation of the pixel values of all the feature maps in h_{l-1} at the same pixel, plus a bias and a non-linear transformation.
- The 1×1 filters serve to fuse the channels in h_{l-1} at each pixel.
- The feature maps at each layer of the inception net are obtained by filters of sizes 1×1, 3×3 and 5×5, as well as max pooling.

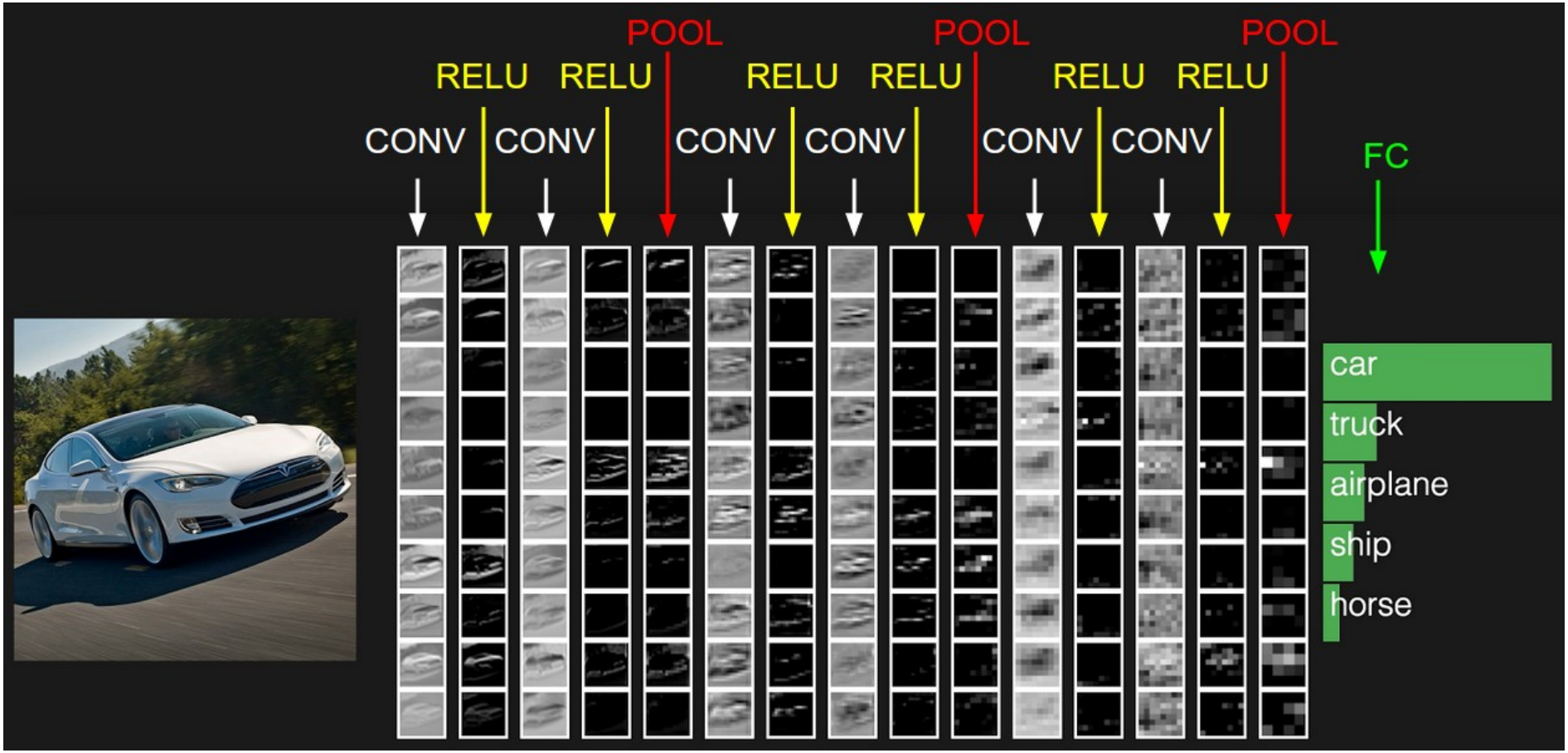
Inception net



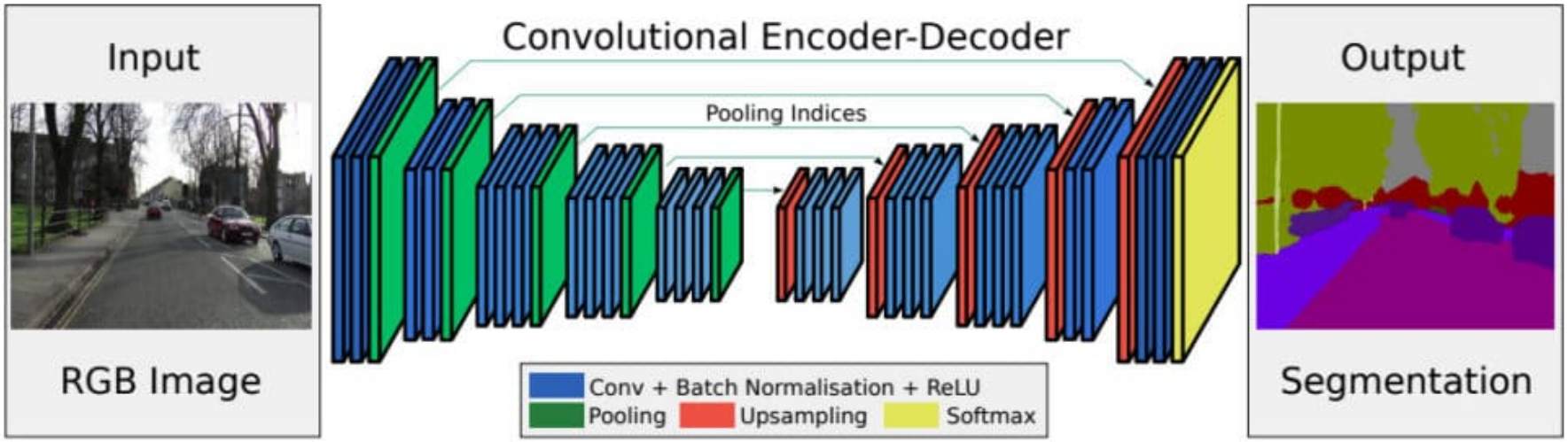
Object detection, semantic segmentation, deformable CNN

- CNN is commonly used for computer vision.
- Two prominent tasks are object detection and semantic segmentation.
  - Object detection is to impose bounding boxes on the objects in the image and output the object categories.
  - In object detection, we can use CNN to predict the possible bounding boxes and the shifts of the bounding boxes.
  - Semantic segmentation is to segment the image into different regions, with each region corresponding to an object whose category is labeled.
- In deformable CNN, we can use CNN to predict the deformations of the grids that support the convolutional kernels.

CNN for object detection



CNN for semantic segmentation



CNN for deformable convolution