Week 5 Recitation

Krushna Shah
CS 190I Deep Learning

UC SANTA BARBARA

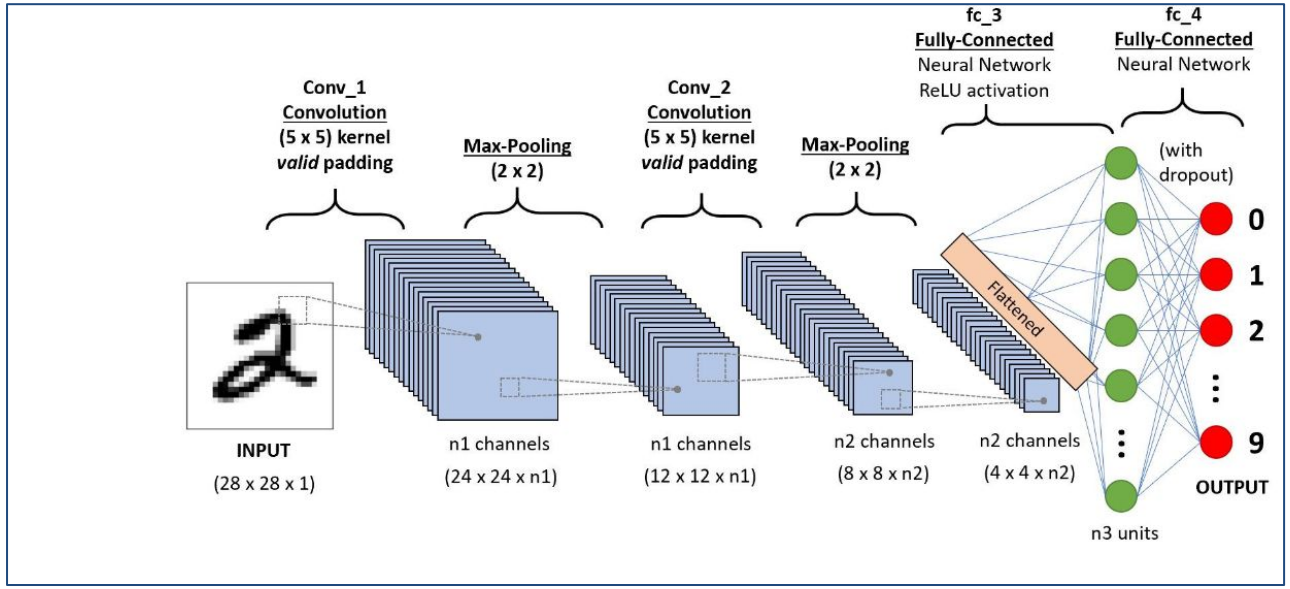
UC SANTA BARBARA
Computer Science

Convolutional Neural Network (CNN)

- Convolutional Neural Networks, also known as CNNs, are a popular type of deep learning neural network commonly used for image classification and computer vision tasks.
- The architecture of CNNs is inspired by the way the visual cortex of the brain processes information, where the neurons are sensitive to specific areas of the visual field.
- CNNs consist of multiple layers, including Convolutional Layers, Pooling Layers, and fully connected layers.

Why CNN?

- Grayscale images have 8-bit pixel values, with input vector size NxM.
- RGB images have input vector size NxMx3.
- Traditional artificial neural networks (ANNs) require a large number of parameters, leading to high computation and memory requirements.
- Convolutional Neural Networks (CNNs) exploit the structure of images by applying filters, reducing connections and leading to sparse input-output connections.
- CNNs allow for parameter sharing, where useful features can be applied to multiple parts of an image



General CNN Architecture

CNN Layers

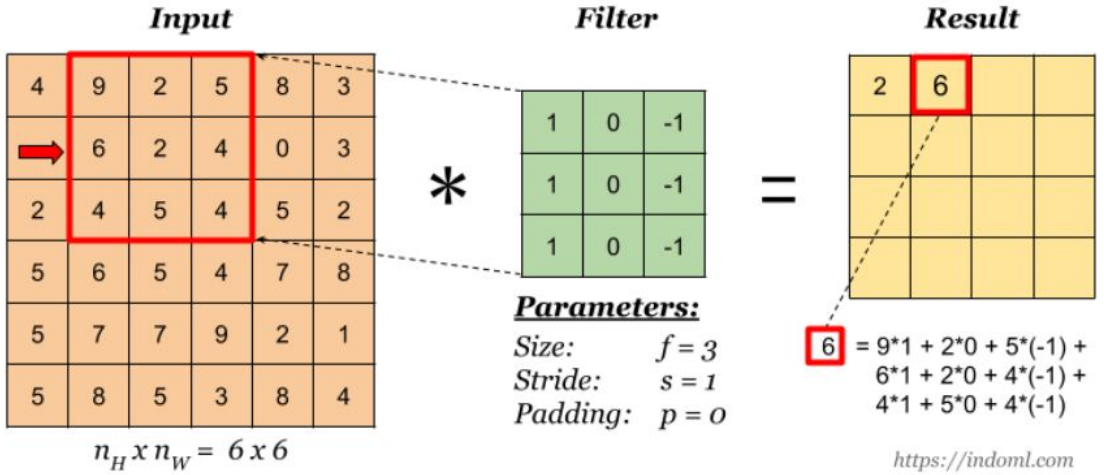
Convolutional Neural Networks (CNNs) are composed of multiple layers that each perform a specific function. The following are the main layers in a typical CNN:

1. Convolutional layer
2. Pooling layer
3. ReLU layer
4. Fully connected layer
5. Dropout layer

These are the main layers in a typical CNN architecture, but there may be variations depending on the specific task and requirements. In addition, modern CNN architectures may include additional layers such as normalization layers, attention layers, or recurrent layers.

Convolutional Layer

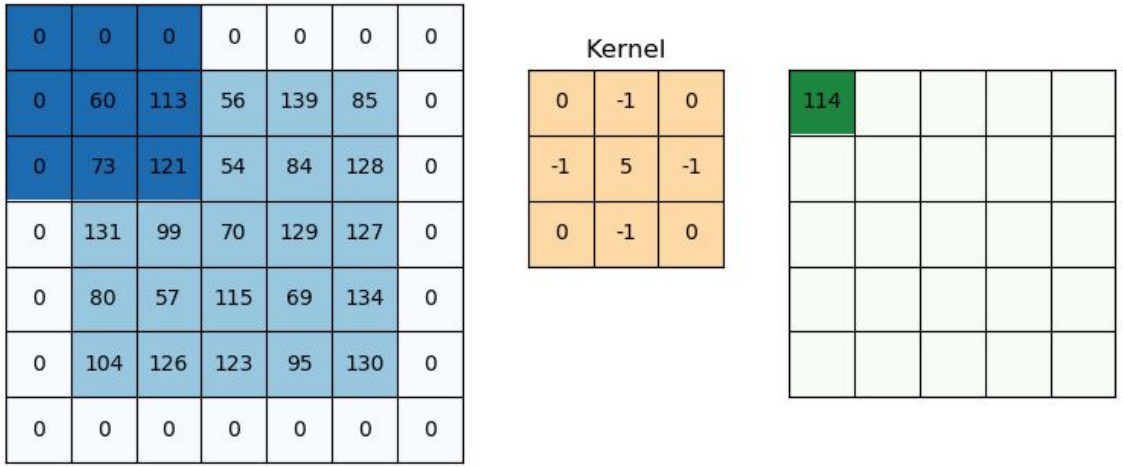
- A convolutional layer in a CNN contains multiple filters applied to a small portion of the input image.
- Each filter produces an output, and all outputs are stacked to form an output volume.
- Matrix values of the input image are multiplied with the corresponding values of the kernel filter and a summation operation is performed to produce the final output.
- The kernel filter slides over the input image to produce the output vector.
- The dimensions of the final output depend on the dimensions of the input image and the kernel filter.
- Weights in a CNN represent the kernel filters, with K kernel maps producing K kernel features.



Convolution operation, Image by indoml

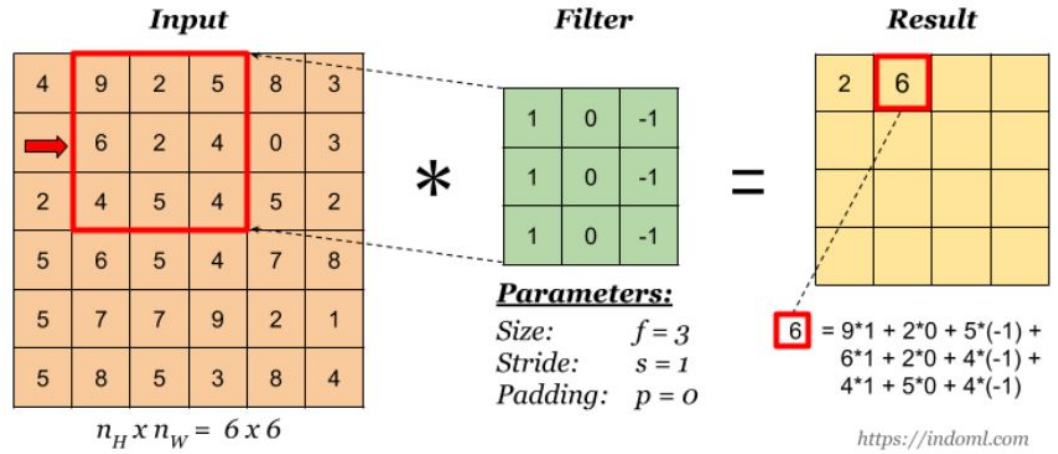
Padding

- Padding is used in CNNs to describe the number of pixels added to an image during processing.
- The padding can be set to a value of zero, meaning every added pixel will have a value of zero. If the padding is set to one, a one-pixel border with a pixel value of zero will be added to the image.
- Padding works by increasing the processing region of a CNN.
- The kernel is a filter that moves through an image, converting data into a smaller or larger format.
- Padding is added to the image frame to provide more room for the kernel to cover the image, aiding in its processing.
- The output feature map size is determined by the equation N-F+2P+1, where N is the size of the input map, F is the size of the kernel matrix, and P is the value of padding.
- To preserve the dimensions of the input matrix, N-F+2P+1 should equal N.

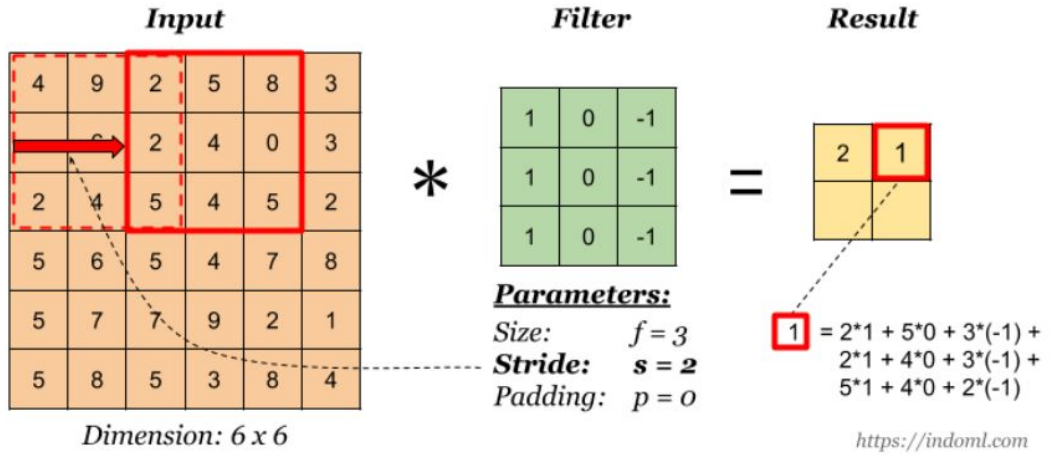


Stride

- Stride refers to the number of pixels the kernel filter will skip during the convolution operation.
- A stride of 2 means the kernel will skip 2 pixels before performing the convolution operation.
- In the figure above, the kernel filter is sliding over the input matrix by skipping one pixel at a time. A Stride of 2 would perform this skipping action twice before performing the convolution like in the image below.
- Increasing the stride from 1 to 2 reduces the size of the output feature map by 4 times.
- The dimension of the output feature map can be calculated using the formula: (N-F+2P)/S + 1, where N is the size of the input map, F is the size of the kernel matrix, P is the value of padding, and S is the stride value.



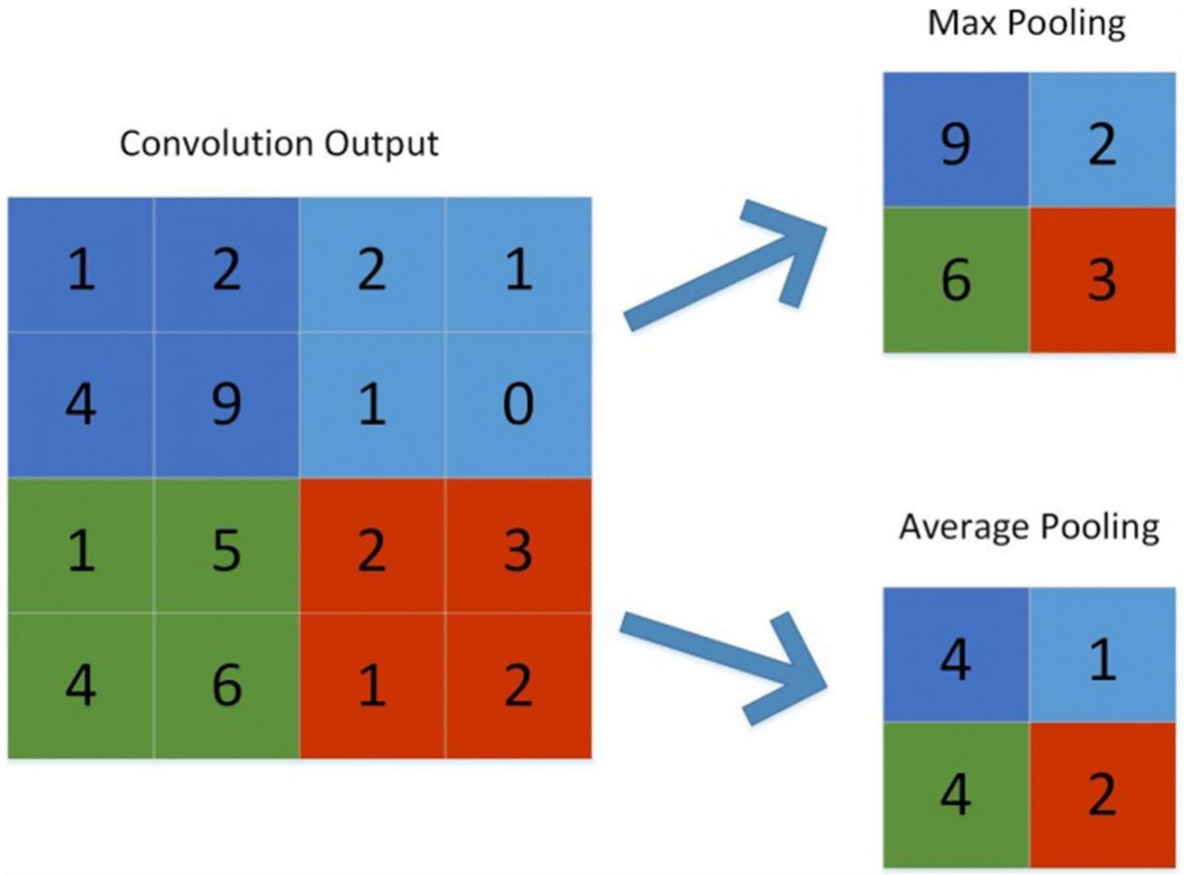
Stride demonstration, Image by indoml



Stride demonstration, Image by indoml

Pooling Layer

- Purpose of pooling layer: reduce number of parameters and computations, prevent overfitting
- Treats each feature map separately
- Gradually reduces spatial dimension of representation
- Output has same resolution as input without pooling.

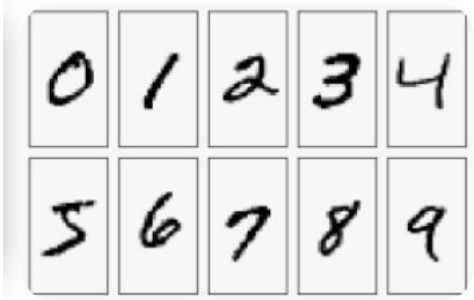


Fully Connected Layer

- The Fully Connected Layers are used to make predictions based on the features extracted by the Convolutional Layers and Pooling Layers.
- The Fully Connected Layers receive the output from the Pooling Layer, flatten it into a single vector, and then pass it through one or more dense layers to make the final prediction.
- Multiple fully connected layers can be present after multiple convolutional and pooling layers.
- Every neuron in the current layer is connected with all neurons in the previous layer.
- The last layer is an output layer that makes final predictions.
- Softmax function is used for multi-class classification, and Sigmoid function for binary classification. (generally)

Summarize

- Input layer – Convolutional Layer – Pooling Layer – FC layer
- LeNet - 5



Layer	#channels	kernel size	stride	activation	feature map size
Input					32 x 32 x 1
Conv 1	6	5 x 5	1	tanh	28 x 28 x 6
Avg Pooling 1		2 x 2	2		14 x 14 x 6
Conv 2	16	5 x 5	1	tanh	10 x 10 x 16
Avg Pooling 2		2 x 2	2		5 x 5 x 16
Conv 3	120	5 x 5	1	tanh	120
FC 1					84
FC 2					10

Implementation of CNN using PyTorch

https://colab.research.google.com/drive/1PDLEddCZKYC_z7U1TY3eIGSj3GUebYmn?usp=sharing

Any Questions?