Convolutional Neural Network

Dr. Jawwad A. Shamsi
FAST NUCES

Convolutional Neural Networks(ConvNet) are the optimal choice for image classification, visual recognition, and computer vision problems. The pre-processing or number of parameters required to train a ConvNet is much less than the conventional fully connected networks. Another important feature is the translational invariance. Although primitive methods require hand-engineering of filters, ConvNets can learn these filters/characteristics with enough preparation.

The architecture of a ConvNet is inspired by the organisation of the Visual Cortex and is similar to the communication pattern of Neurons in the Human Brain. Individual neurons can only respond to stimulus in a small area of the visual field called the Receptive Field. A set of such fields can be stacked on top of each other to fill the entire visual field.

Through the application of relevant filters, a ConvNet can successfully capture the Spatial and Temporal dependencies in an image. Owing to the reduced number of parameters involved and the reusability of weights, the architecture performs better fitting to the image dataset. In other words, the network can be conditioned to better understand the image's sophistication.

- Hand digit recognition
- Medical imaging
- Face recognition
- Facial expression recognition
- Object detection
- Segmentation
- Remote sensing / Environment / Satellite image classification
- Driverless cars and many more

Q. How to represent images ?

An image (gray scale) in the digital form is a two-dimensional matrix of numbers, where each element of the matrix represents the pixel values in the 2-dimensional spatial space. In an RGB image (3D) there are three color channels (Red, Green and Blue), usually 2D ConvNet filters are used for 3D images.

Classification vs Regression

Classification: O/P variable takes class label.

Regression: O/P variable takes continuous value

Image classification:

Bunch of people. We need to identify them. O/P with a probability

Q. What is unique about picture of Mike. (Distinguished Feature)

Q. What is unique about picture of someone else.

Q. What is unique about picture of each

Probabilistic O/P

Jawwad	.7 (In softmax classification, the output class is the one with the highest probability).
Osama	.2
Sara	.05
Michael	.05

High-level : features in each class

Classes

class 1: Face: Nose, eyes, mouth

Class 2: Automobile: wheels, license plate, headinglights

Class 3: Home: doors,windows,steps

How to extract features:

1) Manual Feature extraction:

Domain Knowledge → Define Features → Detect features to classify

Example: Human faces: first detect eyes, ears, nose, etc.

Problems:

(a) preleminary detection pipeline?

(b) Hierarchy?

Remember: images are three dimensional brightness values.

Much variations

a) Deformation
b) Orientation / Different viewss
c) background clutter
d) intra-class variation
e) Occlusion

We need to cater all these variations. Sensitive to inter-class variations and invariant to variations within a class

Manual extraction is challenging

Detection is difficult.

Solution:

Detect features and their presence simultaneously and in hierarchical fashion.

Neural Networks:

- low -level features : Edges, dark spots, (Layer 1)

- **Mid-level features**: Eyes, ears, nose (Layer 2)

- **High -level features**: Facial Structure (Layer 3)

reconstruct label

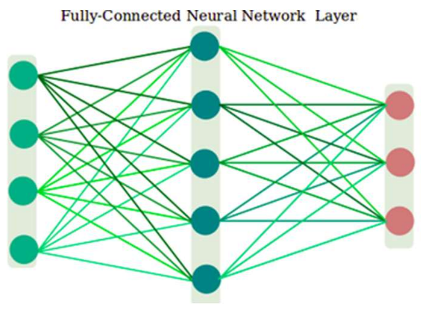
Learning visual features

Recap: dense layer: multiple layers.

<u>Option 1:</u>

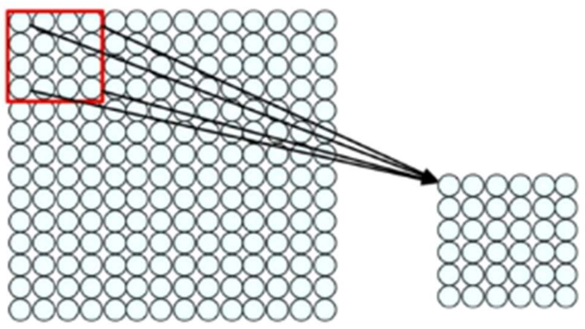
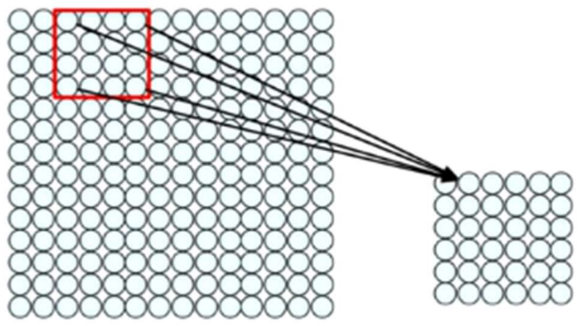
How about if we use a fully connected NN?

Convert a two-dimension image into a 1-dimension
send every pixel to each neuron
Lots of information if we use fully connected NN.



<u>Option 2:</u>
Construct a spatial structure

instead of connecting every pixel, connect a spatial structure
only a region of that image is influencing that neuron.

sliding the patch, learn visual features

Example:
- filter of size 4x4: 16 different weights
apply the same filter to 4x4 patches input.

each patch needs to be weighted

This process is called **convolution**.

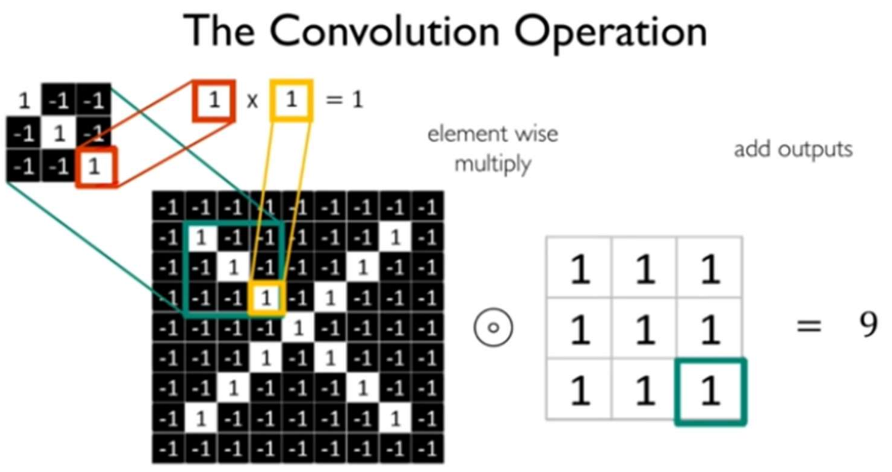
1) Apply a set of weights a filter to extract local features
2) Use multiple features to extract different features
3) spatially share parameters of each filter

Feature Extraction:

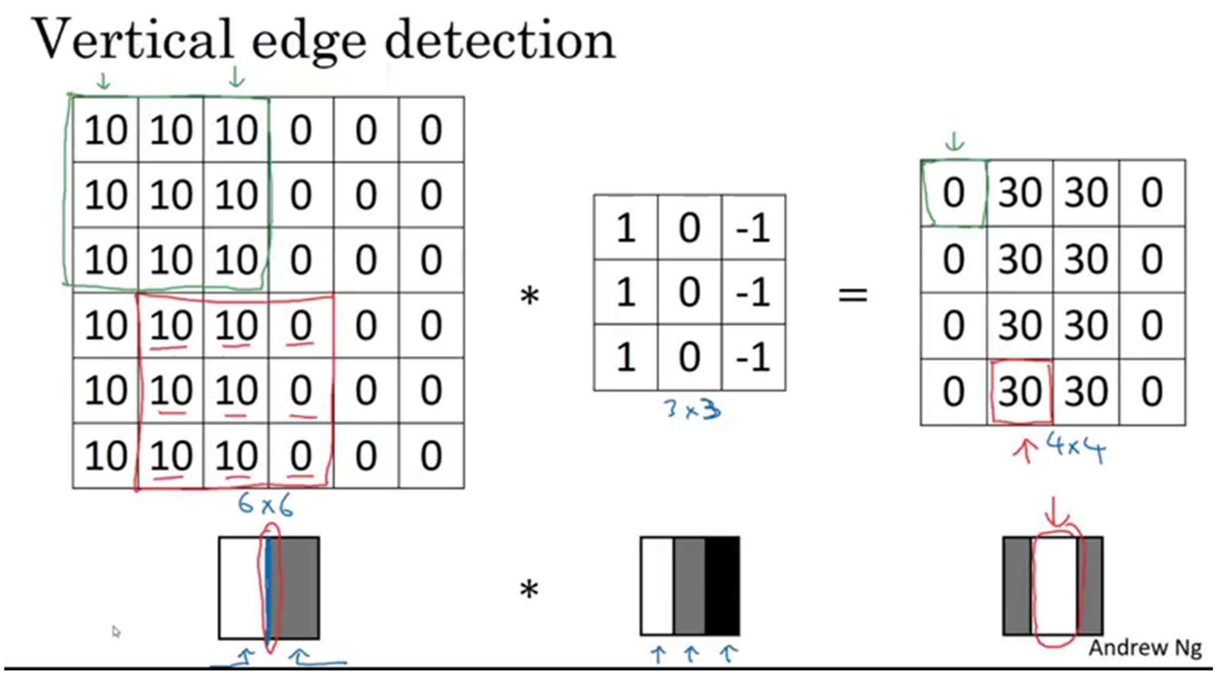
filter or kernel : they scan over an image and filter out the location – producing high activations when they came out across the pattern.

- Large number of kernels , facilitates identification of complex features
- for multiple conv layers, the optimal number of kernels a could vary quite a bit from layer to layer.
- Try to minimize layers wherever possible.

Filter for Edge



Vertical Detection



Horizontal edge detection:

used the filter [[1, 1, 1], [0, 0, 0], [-1, -1, -1]]

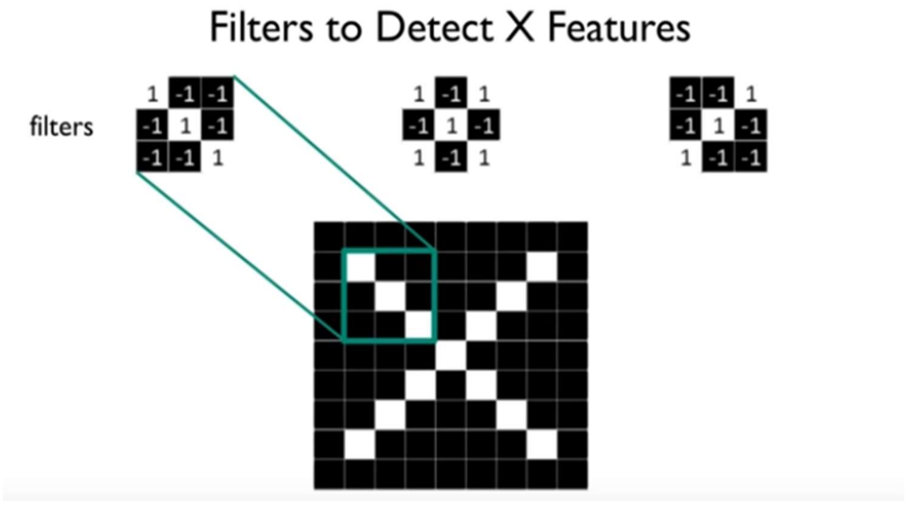
Conv layer output: convolutional output consisting of 0

Another filter for vertical edge:

[[-1, 1], [-1, 1]]

bright pixels on the left and dark pixels on the right (or vice-versa).

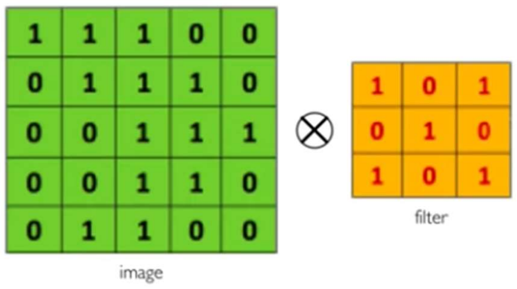
Diagonal Edge

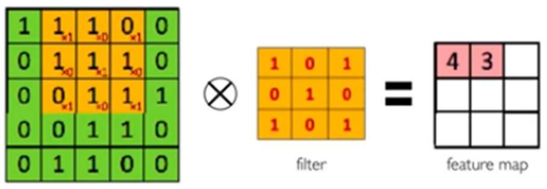


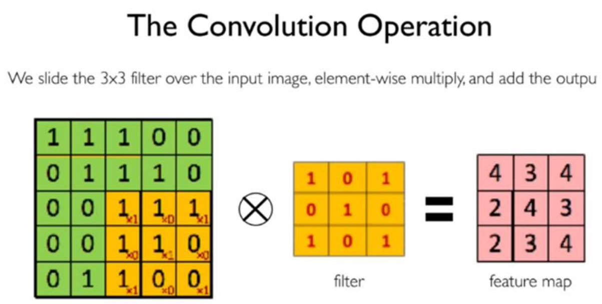
Thumb rule:

The sum of values of the filter should be 0 *else* the resultant image will become brighter or darker.

Example: Suppose 5x5 image with a 3x3 filter.






The Convolution Operation
We slide the 3x3 filter over the input image, element-wise multiply, and add the outpu
filter
feature map
4 3 4
2 4 3
2 3 4

1) Apply a set of weights – a filter to extract local features
2) Use multiple filters to extract different features

Stride: Size of the step that the kernel takes.

Padding: Additional number of zeros which are added to the image. We use padding for two purpose:

a) The activation map is of the same size as of image. So the dimensions are not reduced.
b) Pixels on the corners and edges are used as much as the pixels in the middle.

The value in the filter. Higher value greater activation
Activation Map:

Size of Activation map= $\frac{D-F+2P}{S} + 1$

$$\frac{28 - 5 + 2x2}{1} + 1$$

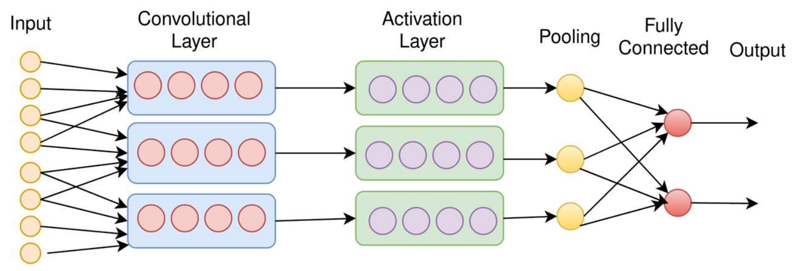
= 28

D is the size of the image (width or height depends if we are calculating the size or width of an image)
F is the size of filter
P is the amount of padding
S is the stride length

No. of filters

A conv. Layer can have any number of kernels. It is a hyper-parameter. Number of filters can be seen as number of features in the spatial dimensions. After the convolution operation a bias term is added to each element of the filter, and finally an activation function, which is normally relu is applied. This produces an activation or a feature map. The convolution layer is normally followed by a pooling layer to further reduce the feature map.

Training:
Learning the weights of the filters for image classification. Minimize error.

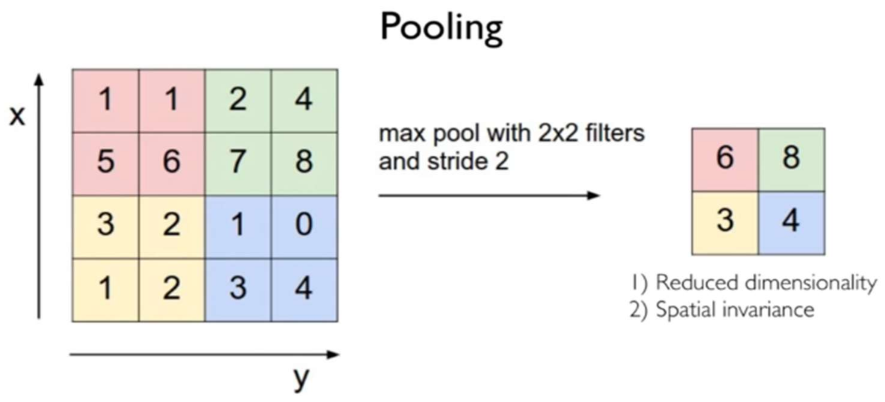


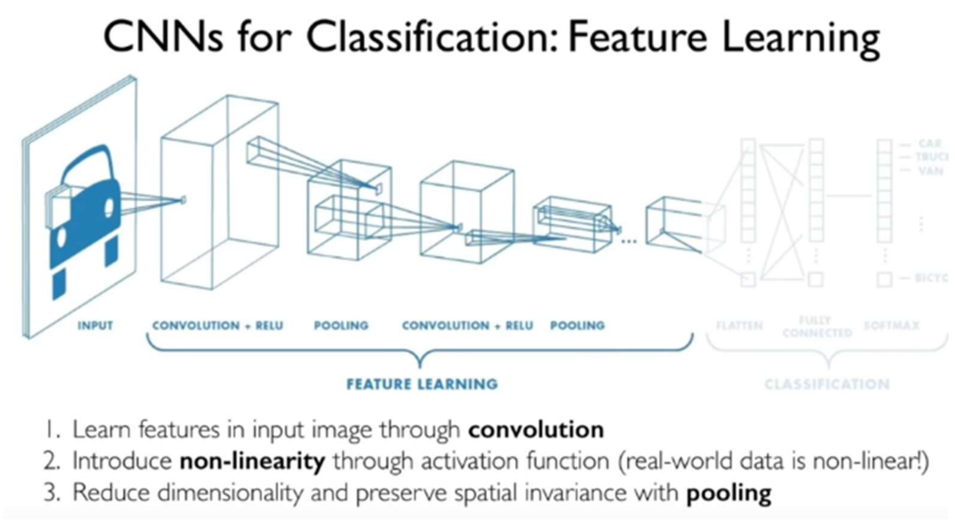
Three main layers of CNN

1) Convolution: Each neuron in the hidden layer will compute weighted sum of inputs (filter and image). Each neuron in the hidden layer is only seeing what comes before.
   a) Takes inputs from the sample
   b) Compute weighted sum
   c) Apply bias
   d) Non-linearity
2) Pooling: Down sampling to reduce the size of the feature map
3) Output through a fully connected layer at the end. No. of neurons resembles no. of classes.

Training: learning weights of filters. applying window of weights . computing linear combination activating with non linear function

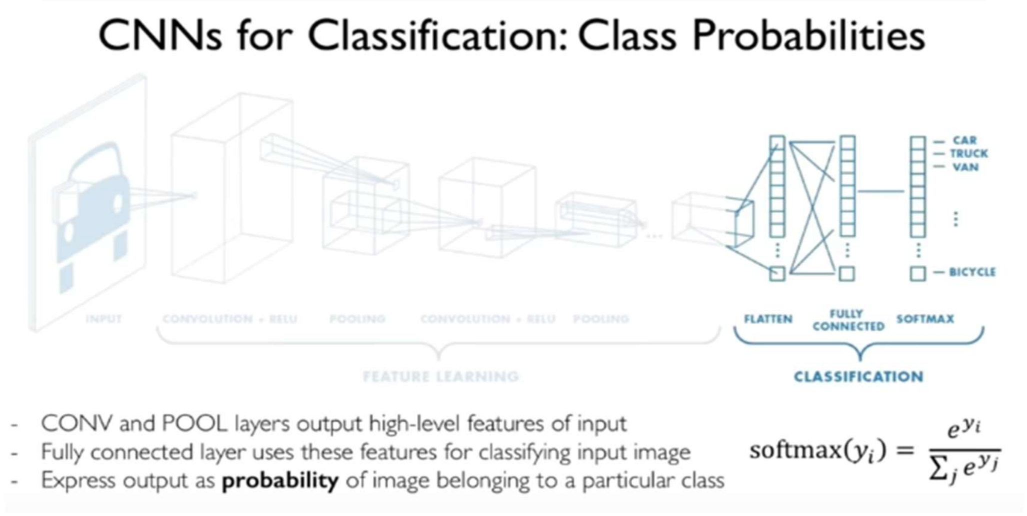
Pooling





CNNs for Classification: Feature Learning

1. Learn features in input image through **convolution**
2. Introduce **non-linearity** through activation function (real-world data is non-linear!)
3. Reduce dimensionality and preserve spatial invariance with **pooling**



CNNs for Classification: Class Probabilities

- CONV and POOL layers output high-level features of input
- Fully connected layer uses these features for classifying input image
- Express output as **probability** of image belonging to a particular class

$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

softmax output represents categorical probability distribution

Training:
BackPropagation: cross entropy loss.
We learn weights of convolution filters
weights of fully connected layer

There could be more than 1 conv layer in a CNN. We learn different features at each layer. For instance, for the image recognition example in the lab, 1st layer, recognize edges, the second layer recognizes objects (eyes, ears etc) , the third layer can recognize faces.

conv2d means, a 2 dimensional filter with height and width.

No. of parameters to learn
For the lab exercise given, following is the calculation of no. of parameters to learn.

<u>1st layer</u>
16 filters, each a 3x3 dimension and RGB scale
3x3 (dim) x3(RGB) =27
27x16(filters)=432
432+16 (bias)=448.

2nd layer

32 filters, each a 3x3 dimension and 16 depth (from the previous layer)

3x3 (dim) x16 =144

144x32(filters)=4608

432+32 (bias)=4640