LEC-3 Deep Computer Vision

↙ ↘

Regression (continous value)

Classification (belonging to a particular class)

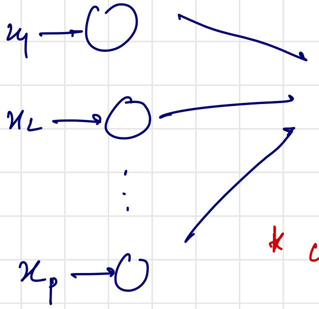
Key features in each image category - Feature Extraction

Domain knowledge → Define features → Detect features to classify

hierarchy features low → mid → high level features

Fully Connected Neural Network

Input 2-D



Fully connected

* no spatial information
* too many parameters

* connect patches of input toneurons in hidden layer

"sees" value of patch

Feature Extraction with Convolution

- Filter size 4x4 : 16 different weights
- Apply same filter to 4x4 patches in input
- Shift by 2 pixels for next patch

"patchy" operation is convolution

1) Apply set of weights — to extract local features

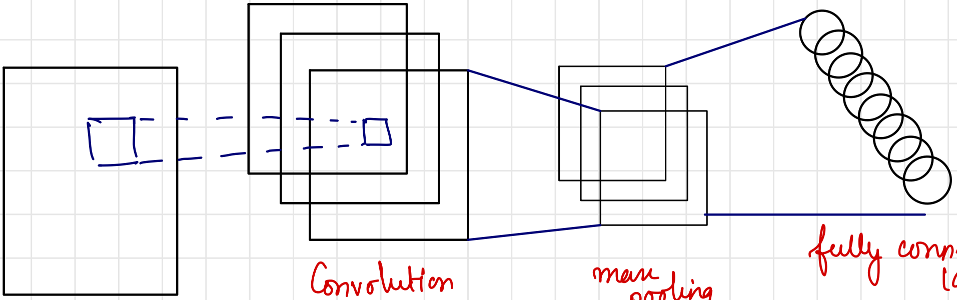
2) Use multiple features to extract different images

3) Spatially share parameters of each filter

Convolution Operation : element wise multiply

image x filter = feature map

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 3 & 4 \\ 2 & 4 & 3 \\ 2 & 3 & 4 \end{bmatrix}$$



1) Convolution — Apply filters to generate feature maps

2) Non-Linearity — often ReLU

3) Pooling — Downsampling operation on each feature map

tf. keras . layers . Conv2D

For a neuron in hidden layer

- Take inputs from patch
- Compute weighted sum
- Apply bias

CNNs : Spatial Arrangement of Output Volume

Layer dimension $h \times w \times d$

d = no. of filters

Stride : filter step size

Receptive field : Locations in input image that a node path is connected to

```
tf. keras. layers. Conv2D( filters = d, kernel_size = (h,w), strides = s)
```

Pooling

```
tf. keras. layers . MaxPool2D(
        pool_size = (2,2),
        strides = 2 )
```

① Feature Learning - giving high level features of input

② Class Probablities - fully connect layer to classify & give probablity of image belonging to a class

Input $\underbrace{\text{conv + relu} \quad \text{pooling}} \quad \underbrace{\text{conv + relu} \quad \text{pooling}}$

Object detection with R-CNN

① Input Image

② Extract region proposals

③ Compute CNN features

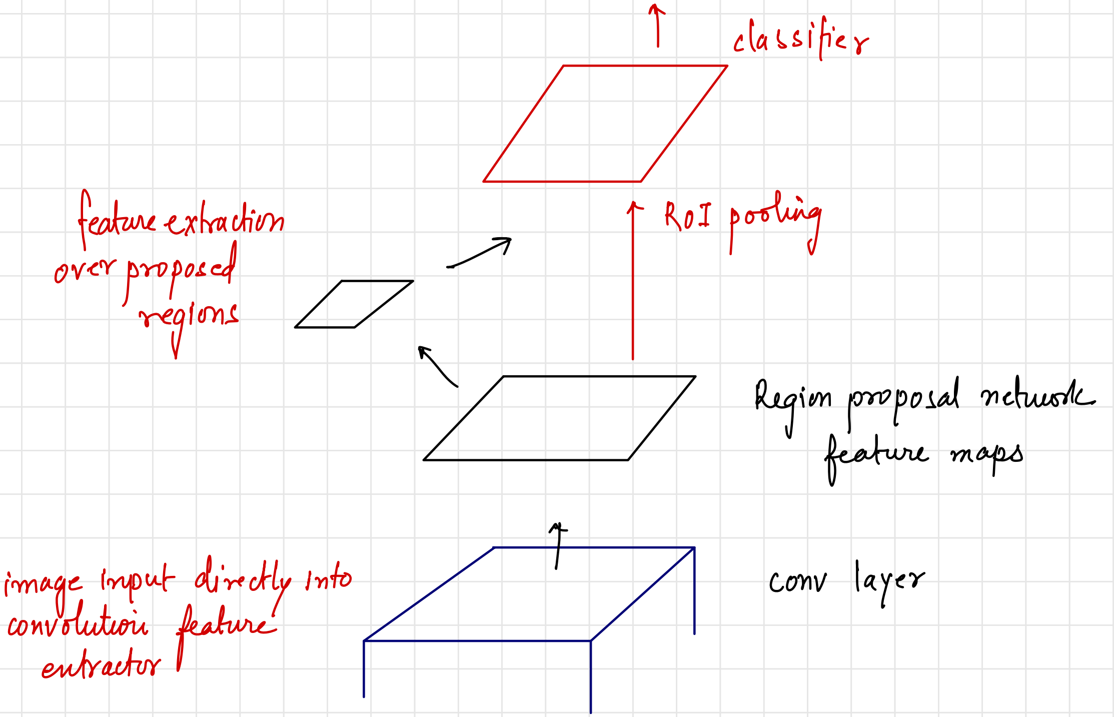
④ Classify Regions

issues

① slow; time intensive inference

② Brittle; manually defined region proposals

Faster R-CNN Region proposals


classifier
feature extraction over proposed regions
RoI pooling
Region proposal network feature maps
image input directly into convolution feature extractor
conv layer