Convolution NN

Computer Vision
- Rapid growth → data → features
  - computer specification (RAM, CPU) → weight/parameter

Application
- Image classification
- Object detection
  - single
  - multiple → position
- Neural style transfer

Image Classification

Edge detection
- vertical
- horizontal
- others

Convolution
- matrix * filter/kernal
- ↓
- Σ element wise multiplication
- strided convolution - iteratively jump # steps ↔ - stride
- [shrinking output - # of convolution; pixel on edges are use less # of time]

filter
- $\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$ vertical edge
- $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$ horizontal edge
- $\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$ Sobel filter
- $\begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix}$ Scharr filter

- shades
  - light to dark
  - dark to light
  - detect - absolute
- learning parameter
- central position - natural dimension

Padding → border # of pixel → symmetrical padding
- Same Convolution → size (input = output)
- valid Convolution → no padding

dim

Size of output $= \left\lfloor \frac{\text{input} + \text{pad} \times 2 - \text{filter}}{\text{stride}} - 1 \right\rfloor$ floor

Convolution → Cross-correlation
- sliding dot product
- sliding inner product

\+ diagonal flip of filter → convolution

double mirroring operation - associative property - Signal processing

Convolutions over volume

2D grayscale image → 3D RGB channel image

filter → same no. of channel

# of filter
- 1 → 2D output
- n → 2D × n output (stacked)

activation
- normal
- z transformation
- non linear ftn

applied on each in layer

Edge detector
- Vertical/Horizontal
  - 1 specific pattern
    - 1 channel - dummy layer in filter
    - other channel
  - multiple pattern - multiple filter

1 filter

Type of Layers in Convolution NN

Conv Net
- Convolution
- Pooling
- Fully Connected

Pooling layer

Max Pooling
Average Pooling
- feature detected over region gets
  - max
  - average
- preserved

3D apply on each channel independently

data ↓ → efficient
- robust feature

type parameter
- filter size
- stride

no learning while backward prop

Convolution NN

activation size ↓
- Convolution (time) ↓ height, width ↑ depth
- Pooling (time) ↓ height, width
- Flatten
- Fully connected layers
- Prediction

Why convolution

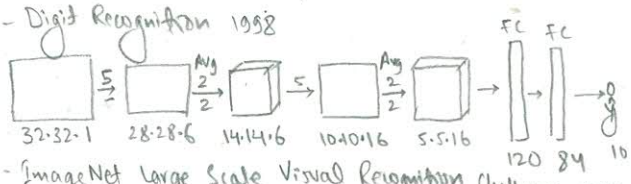
no. of parameter ↓ → overfitting
- Parameter sharing - filters
- Sparsity of connection - filter

CASE STUDIES

NN Architecture

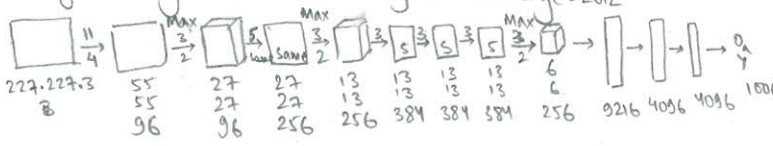
Classic Networks

LeNet5 - Digit Recognition 1998



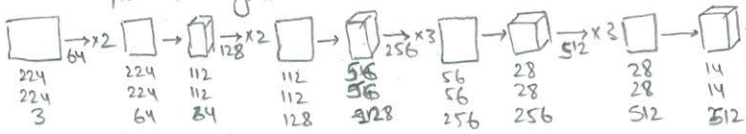
60,000 parameters
Pooling → average
Activation function → Sigmoid, tanh
Classifier → ↓softmax
Valid convolution

AlexNet - ImageNet Large Scale Visual Recognition Challenge - 2012



60 million params
Image Net dataset
ReLU
Local Response Normalization

VGG-16 16 layers with weights 2015



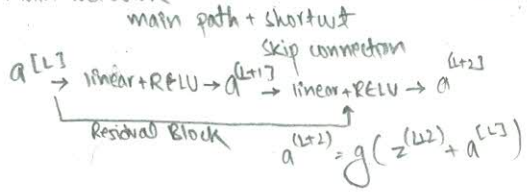
Convolution filter $f=3\times3$ stride = 1 same
Max Pool $f=2\times2$ $s=2$
138 million params
Easy architecture
- filter

ResNets Networks 2015

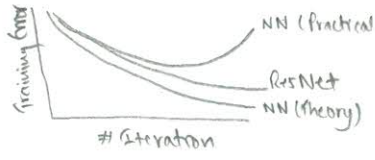
Deep NN - Vanishing / Exploding } Gradients

Plain Network

Residual Network
main path + shortcut
skip connection



$a^{(l+2)} = g(z^{(l+2)} + a^{[l]})$



Parameter - Easy to learn Identity function
↓ hurt performance

Condition { same dimension - same convolution
↓ same dimension - weight transform

Inception Networks 2014

1×1 Convolution
Network in Network

>1 channel → reduce channel - filters
non-trivial + non linearity
can ↓ computational ~~cost~~

choose { convolution / pooling } - layer ? Stack - computational cost - 1×1 convolution - bottle neck layer
Channel concat

side branch → prediction
↓ overfitting
- regularization
- hidden layer → prediction

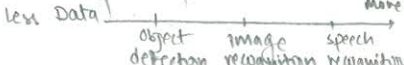
Advice

Transfer learning → Open source Implimentation - github
↓ Codes & weight
↓ Architecture
→ use & change - { small data - softmax function
medium data - last few layer - { change / re tune
large data - full network - re train
} freeze trainable parameters
transform input → save

Data Augmentation
- Shape & size - mirroring, random cropping, rotation, shearing, local wrapping
- Color shifting - PCA color augmentation

Implementation
CPU thread

State of Computer Vision



- simple Algorithm
- ↓ hand engineering features

Source of knowledge
- labelled data
- Hand engineering

Benchmark
Ensembling
Multi crop at test time

Deployment
Production
Computation cost

OBJECT DETECTION

Image classification $\hat{y} = P(\text{class})$ (each) → softmax function

Landmark Detection (Snapchat)

boundary / location of box / bbox → bool: interested class or not

b_x, b_y (center) b_h (height) b_w (width) → location coordinate (0,0) to (1,1)

$c_1, c_2 \dots c_n$ interested class → class selected

→ other class not interested

probability of class

Loss function

$L(\hat{y}, y)$

$= \begin{cases} \sum_{j=0}^{n} (\hat{y} - y)^2 & \text{if object} = 1 \\ (\hat{y}_1 - y_1) & \text{if object} = 0 \end{cases}$

all

only bool

Object detection with localization

Classified Images → search → canvas

Sliding windows detection algorithm 2014

window → stride/step size → size - multiple sequentially

Simple classifier → linear regression + hand engineered features

computation cost → CNN

Convolutional Implementation - share computation

CNN fully connected layer → convolution layer (1x1 convolution) → multiple layers

convolution - windows

pooling - stride

Object Detection

YOLO (You look only once) algorithm

Image → grides(s) → each target label

P_c - one object → one grid (center of object)

$0 \le b_x, b_y \le 1$

b_h, b_w relative to grid cell

Bounding box accuracy ↓ → position, stride, shape

- Boundary box of any aspect ratio explicitly incorporate
- convolutional implementation - real time

Multiple detection of same object

clean up detection - Non Max Supression

- Select boxes (cutt off of P_c)
- Pick box with largest P_c
- Discard boxes with high similarity - Intersection over Union (IoU) = intersection / union

Multiple Objects - Anchor Box

predefined shapes → stack in target label

each object (training) - grid cell - mid point → anchor box - with - highest IOU

- Multiple obj in anchor boxes - tie breaker
- Different obj - same grid cells, anchor box × rare chances
- box shape - predefined hand engineered, Kmeans to detect obj shape

Region proposals - ignore non interesting regions

Subset few windows → run classifier

segmentation algorithms - find blub - run classifier

Box: shape, scale

slow

R-CNN Region - label - boundary box

Fast R-CNN convolutional implementation

Faster R-CNN

Face Recognition + (Liveness)
Image → Database
↓
Generate ID

Face Verification
Image → ID
↓
Check

Learn from one example - One shot learning problem
CNN - accuracy ↓ scalable
+ extra person → update model

Learning Similarity function
- degree of differences → cutoff

Siamese Network Architecture - 2014 DeepFace
Image $x^{(i)}$ → CNN encoding → Vector $F(x^{(i)})$
representation of image

$d(x^{(i)}, x^{(j)}) = \| f(x^{(i)}) - f(x^{(j)}) \|_2^2$ - L_2 difference in encoding
same param from CNN/encoding → find encoding - degree of diff - small → same person; degree of diff - large → diff person

2015 - FaceNet
Objective
Triplet Image → Anchor(A) Positive(P) Negative(N)
same person | different person

$\| f(A) - f(P) \|^2 - \| f(A) - f(N) \|^2 + \alpha = 0$
margin - hyperparameter, +ve number

Loss function → Triplet
$L(A,P,N) = \max(\| f(A) - f(P) \|^2 - \| f(A) - f(N) \|^2 + \alpha , 0)$
$J = \sum L(A,P,N)$

Choose Triplet Image - hard to train - CNN
Random

Binary Classification Problem
Image → CNN → Embedding → Logistic regression
- encoding
- difference in encoding - absolute diff $|f(x^{(i)}) - f(x^{(j)})|$
- χ^2 similarity $\frac{(f(x^{(i)}) - f(x^{(j)}))^2}{f(x^{(i)}) + f(x^{(j)})}$

Neural style Transfer
Content(C) Style(S)
Generate Image (G) - random - assign - keep updating

Visualization of NN
Image → Patches → Activation Unit - given layer - Repeat
Find patches with high activation → Visualize
max
shallow layer - simple feature - edges, colors
deep layer - complex pattern

Cost function
$J(G) = \alpha J_{content}(C,G) + \beta J_{style}(S,G)$
Content Cost function - apply pretrained ConvNet - C, G - select layer
obj → C & G - similar
$J_C = \| a^{[l](C)} - a^{[l](G)} \|^2$

Generate Image
$G = G - \frac{\partial}{\partial G} J(G)$

Style Cost function - correlation across channels
Style matrix
$a^{[l]}_{i,j,k}$ - activation, height, width, channel
layer
gram matrix - G
$G^{[l]}_{kk'} = \sum_i \sum_j a^{[l]}_{ijk} a^{[l]}_{ijk'}$
unnormalized cross covariance

given layer - $J_S^{[l](S,G)} = \| G^{[l](S)} - G^{[l](G)} \|^2$ frobenium norm
multiple layers - $J_S = \sum_l \lambda^{[l]} J_S^{[l]}(S,G)$
hyperparameter

Convolution - Data
		Convolution
1D	ECG Report	1D
2D	picture	2D
3D	CAT scan Move with Time	3D

- channel