Soliton

Vision for a Better World

Deep Learning in Computer Vision

Anthill 2018

Soliton
Vision for a Better World

The team

Sumod

Founder, AutoInfer
&
CV & ML Architect,
Soliton Technologies

Shivaraj

3D Vision Lead,
Computer Vision and
Machine Learning,
Soliton Technologies

Dhivakar

Senior R&D Engineer,
Computer Vision and
Machine Learning,
Soliton Technologies

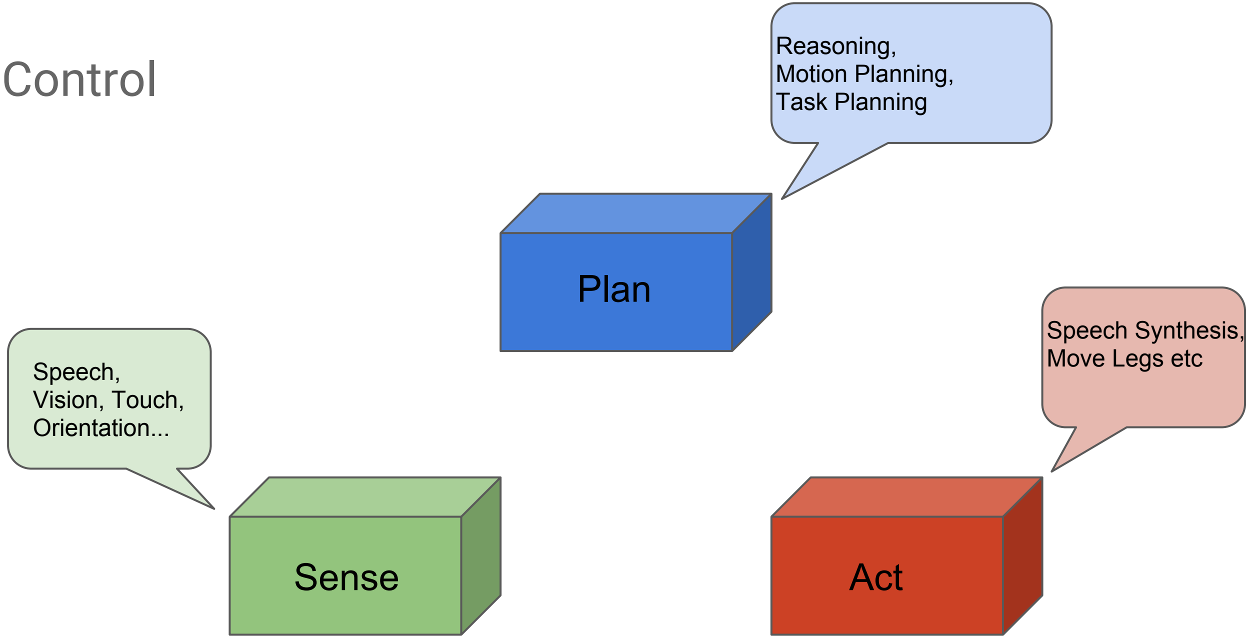
Senthil

R&D Engineer,
Computer Vision and
Machine Learning,
Soliton Technologies

Soliton
Vision for a Better World

Humans in the World

- Perceive -> Reason -> Control



Soliton
Vision for a Better World

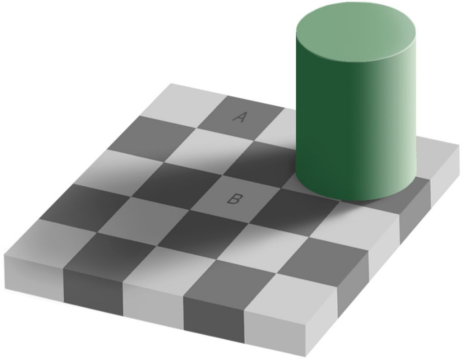
Human Vision

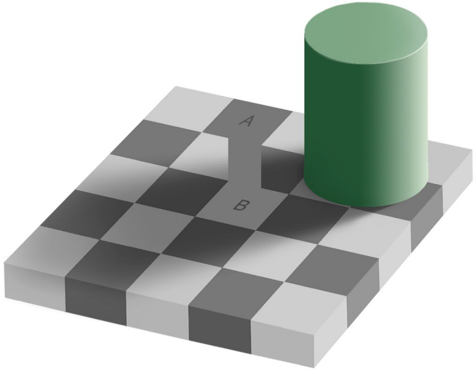
- Importance of vision for humans
    - 30% of cortex used in vision
    - McGurks Effect
- Arguable one of the most Complex Machines
    - Color Stereo Pair: 768 x 576 @ 30 fps : 80 MB/s
    - 10^{11} Neurons
    - 10^{14}-10^{15} Synapses
    - Eons of evolution, learning right priors
    - Continuous flow of perception (with very few glitches)

Soliton
Vision for a Better World

Human Vision

- MultiSensory, High Resolution, Immersive Inner Movie
- Arguable one of the most Complex Machines
  - Color Stereo Pair: 768 x 576 @ 30 fps : 80 MB/s
  - 10^{11} Neurons
  - 10^{14}-10^{15} Synapses
  - Eons of evolution, learning right priors
  - Continuous flow of perception (with very few glitches)
- Perception
  - No eye inside an eye: just stream of electrical impulses :*Shades*
  - Brain combines them all to create our reality based on past experiences: *Sound Prediction & McGurks*
  - It is Brain's best guess of what is out there: *Ames Room*





Ref: https://en.wikipedia.org/wiki/Checker_shadow_illusion

Soliton
Vision for a Better World

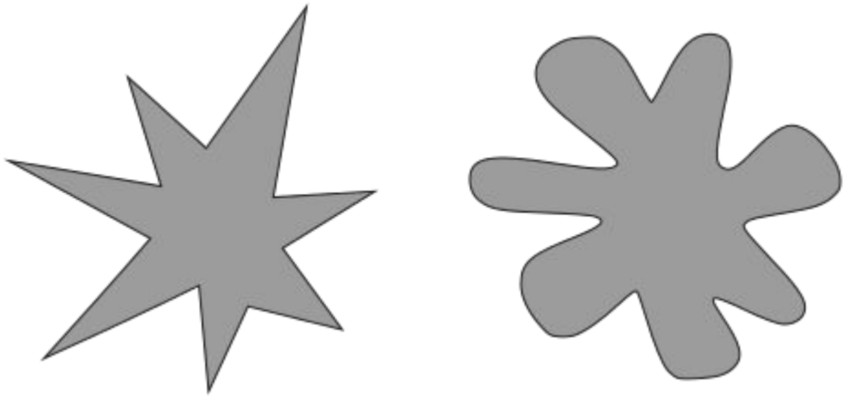
Perception as Controlled Hallucination

- Hallucination: Uncontrolled perception
  - **Perception: Controlled Hallucination**
- Perception: Not Passive
  - **Actively Generate a Model**
- Not just our external perception
  - **Even Internal**

Soliton
Vision for a Better World

*Ref - https://www.youtube.com/watch?v=DgPaCWJL7XI

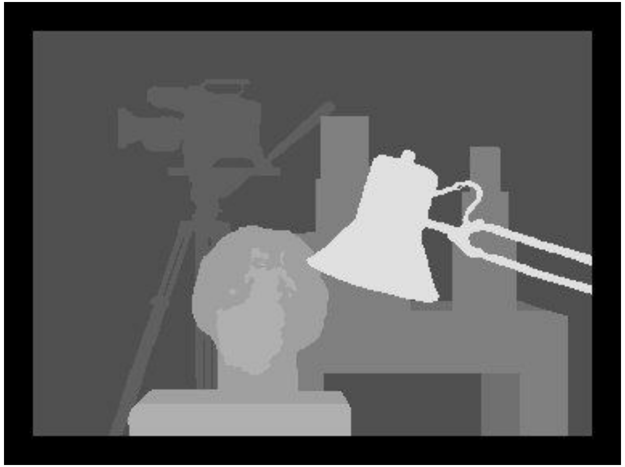
Interconnections



https://en.wikipedia.org/wiki/Bouba/kiki_effect

Computer Vision

- Image Processing, Computer Vision, Machine Vision
- Low Level: Edge Detection, Filtering..
- Mid Level: Segmentation..
- High Level: Object Detection, Stereo



http://vision.middlebury.edu/stereo/data/scenes2001/

Soliton
Vision for a Better World

What is Computer Vision?

Ability of Computers to See and Understand from the Scene

- Saying what are all the things present in the image
- Describe the scene

Computer Vision in Games

http://blog.playosmo.com/post/94655155207/a-peek-into-osmos-reflective-ai-technology

Image Captioning

A person riding a motorcycle on a dirt road.

A group of young people playing a game of frisbee.

A herd of elephants walking across a dry grass field.

http://blog.agi.io/2015/02/the-real-risks-of-ai_17.html

Seam Carving

https://en.wikipedia.org/wiki/Seam_carving

Soliton
Vision for a Better World

Vision Applications

- Bullet Time special-effect in The Matrix is an example of the application of SFM ideas.
- Linear array of cameras replaces moving camera.
- Green screen makes segmentation easy.

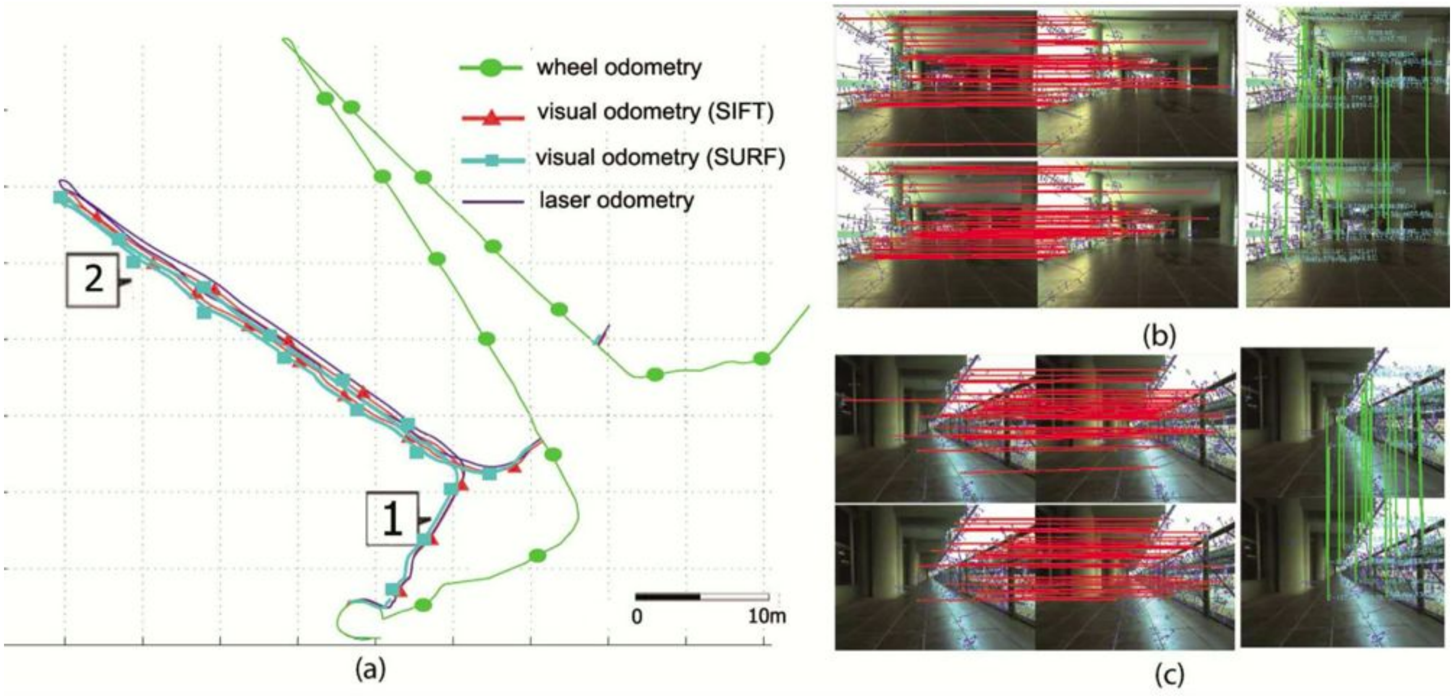
Soliton
Vision for a Better World

https://www.youtube.com/watch?v=Kjcv-JtUOgA

Vision Applications: Synthetic Aperture Imaging

https://graphics.stanford.edu/papers/plane+parallax_calib/

Vision Applications: Visual Odometry



http://www.mdpi.com/1424-8220/11/7/7262/htm

Soliton
Vision for a Better World

Vision Applications: Navigation

https://www.quora.com/How-does-one-train-a-self-driving-car-to-drive-safely

Soliton
Vision for a Better World

Goals

Help building an understanding of How to solve real world problems using Computer Vision with ML and DL

Understand the Math and appreciate the beauty of it as opposed to simply Applying the technology

We can’t teach you Vision in a 6 hour Workshop. But we will try our best to Motivate you towards learning

Computer Vision

Digital image capture, storage and process.
Traditional Object detection and Limitations

Machine Learning

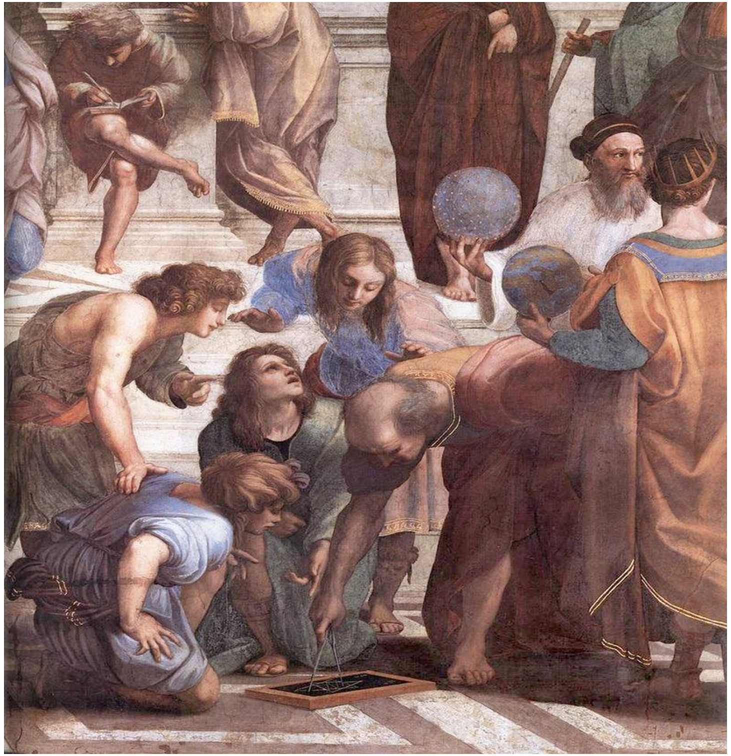
Evolution of ML. Understanding how Machines learn.
Limitations

Deep Learning

History and Biological motivation. Dive into Black Box (DL). Best practices

Soliton
Vision for a Better World

What are the problems with these images?



https://en.wikipedia.org/

Perspective | Projective Geometry

https://orlandotutor.wordpress.com/2010/10/28/a-perspective-on-sunbeams/
http://gaudi.fa.ulisboa.pt/~20141046/cenario.html

Soliton
Vision for a Better World

Equation of Line

- Equation of line cartesian coordinate system:

  $y = mx + c$ where m is slope and c is intercept

- How can we represent the vertical line with above equation?

- We can use polar coordinate system to represent line

  $a\sin\theta + b\cos\theta = \rho$

  where a and b are multiples of r, θ is the angle and ρ is the constant

- Can we use linear algebra with above equation?
  - No as we are dealing with nonlinear functions

Lines

- A better parameterization can represent all lines:

$$ax + by + c = 0$$

- Here the line is represented by 3 parameters:

$$u = [a, b, c]^T$$

- But nonzero scalar multiple does not change the equation:

$$\alpha a x + \alpha b y + \alpha c = 0, \qquad \alpha \neq 0$$

- So we have only 2 degrees of freedom
- To make this work, we have to introduce a non-intuitive definition:

$$\mathbf{u} \equiv \alpha \mathbf{u}, \qquad \alpha \neq 0$$

- I.e., the vector u and its scalar multiple are the same

Soliton
Vision for a Better World

Points

- While we are at it, let us put the point into a vector, too:

$$\mathbf{p} = [x \quad y \quad 1]^T$$

- Which leads to the beautiful expression:

$$\mathbf{p}^T\mathbf{u} = \mathbf{u}^T\mathbf{p} = 0$$

- Nonzero scalar multiple also does not change the point:

$$\mathbf{p}^T\mathbf{u} = \mathbf{u}^T\mathbf{p} = \mathbf{p}^T(\alpha\mathbf{u}) = (\alpha\mathbf{p})^T\mathbf{u} = 0$$

- So we introduce an analogous non-intuitive definition:

$$\mathbf{p} \equiv \alpha\mathbf{p}, \qquad \alpha \neq 0$$

Soliton
Vision for a Better World

Homogeneous Coordinates

- Homogeneous representation of Point

Normal coordinates $(x, y)^T$

Homogeneous coordinates $(x_1, x_2, x_3)^T$ but only 2DOF

- Conversion to normal representation as follows

$$\mathbf{x} = \mathbf{x_1} / \mathbf{x_3} \text{ and } \mathbf{y} = \mathbf{x_2} / \mathbf{x_3}$$

- Convert to Homogeneous coordinates **(2, 3) ? ->**

- Convert to inhomogeneous coordinates **(4, 6, 2) ? ->**

Soliton
Vision for a Better World

Homogeneous Coordinates

- Homogeneous representation of lines

$3x + 4y + 2 = 0 \qquad (3, 4, 2)^T$

$$ax + by + c = 0 \qquad (a,b,c)^T$$

$$(ka)x + (kb)y + kc = 0, \forall k \neq 0 \qquad (a,b,c)^T \sim k(a,b,c)^T$$

- Homogeneous representation of points on a line

$$\mathrm{x} = (x, y, 1)^T \text{ on } \mathrm{l} = (a,b,c)^T \text{ if and only if } ax + by + c = 0$$

$$(x,y,1)(a,b,c)^T = (x,y,1)\mathrm{l} = 0 \qquad (x, y, 1)^T \sim k(x, y, 1)^T, \forall k \neq 0$$

The point **x** lies on the line l if and only if $\mathbf{x}^T\mathbf{l} = \mathbf{l}^T\mathbf{x} = 0$

Soltion
Vision for a Better World

Example

- Question: What does the vector [4, 6, 2]T represent?

  Ans: It depends.

  - If the vector is a 2D point, then the point is (4/2, 6/2) = (2, 3) -- divide by 3rd coordinate
  - If the vector is a 2D line, then the line is $4x + 6y + 2 = 0$, or $2x + 3y + 1 = 0$

- Points and lines are represented in the same way. Context determines which.

Soliton
Vision for a Better World

Points from Lines and vice-versa

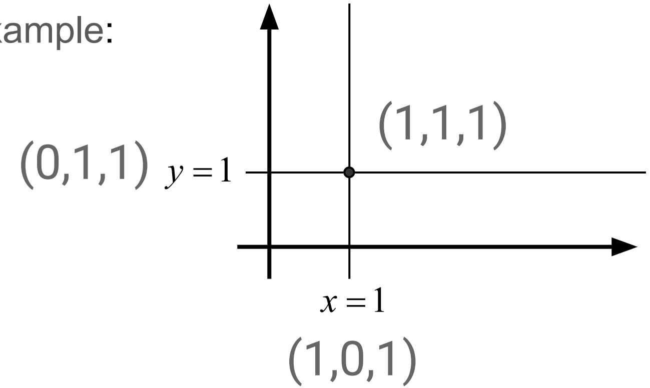
- Ques.: Which points lies at the intersection of two lines?

  The intersection of two lines $\mathbf{l}$ and $\mathbf{l}'$ is $\mathbf{x} = \mathbf{l} \times \mathbf{l}'$

- Ques.: Which line passes through two points **x** and **x'**?

  The line through two points $\mathbf{x}$ and $\mathbf{x}'$ is $\mathbf{l} = \mathbf{x} \times \mathbf{x}'$

Example:



- Ques: What will be the intersection point when two lines are parallel?

  l = $(1,0,1)^T$ and l' = $(1,0,2)^T$?

Soliton

Vision for a Better World

Duality

$$\mathbf{x} \longleftrightarrow \mathbf{l}$$

$$\mathbf{x}^{\mathsf{T}}\mathbf{l} = 0 \longleftrightarrow \mathbf{l}^{\mathsf{T}}\mathbf{x} = 0$$

$$\mathbf{x} = \mathbf{l} \times \mathbf{l}' \longleftrightarrow \mathbf{l} = \mathbf{x} \times \mathbf{x}'$$

Duality principle:

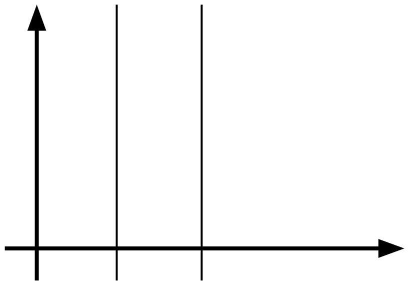
To any theorem of 2-dimensional projective geometry there corresponds a dual theorem, which may be derived by interchanging the role of points and lines in the original theorem

Ideal Points and The Line at Infinity

- Intersections of parallel lines

  l = $(1,0,1)^T$ and l' = $(1,0,2)^T$ the intersection point -> lxl' = $(0,-1,0)^T$ this is the point at infinity where these two lines meet.

Example:



$$\mathbf{l} = (a, b, c)^{\mathsf{T}} \text{ and } \mathbf{l}' = (a, b, c')^{\mathsf{T}}$$

$$\mathbf{l} \times \mathbf{l}' = (b, -a, 0)^{\mathsf{T}}$$

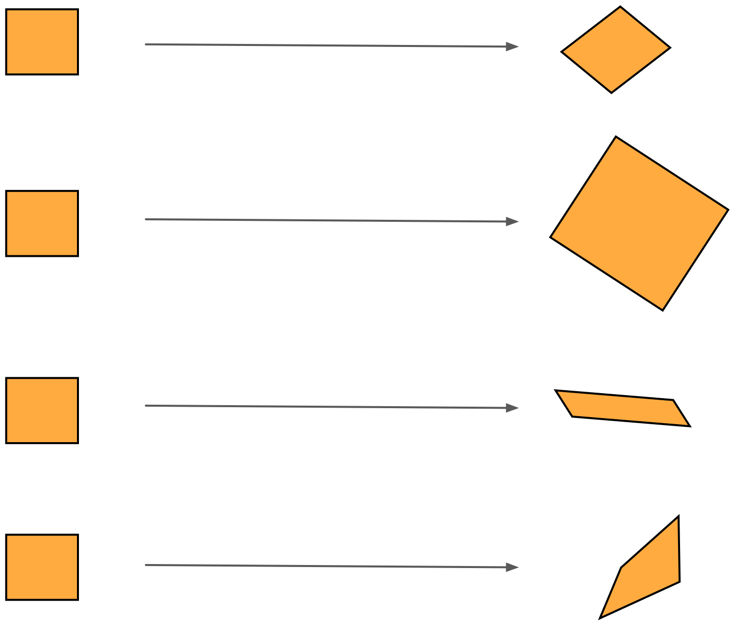
Line formed by points at infinity?

Ideal points $(x_1, x_2, 0)^{\mathsf{T}}$

Line at infinity $\mathbf{l}_\infty = (0,0,1)^{\mathsf{T}}$

Soliton
Vision for a Better World

How to achieve?



Soliton
Vision for a Better World

Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard H, Andrew Zisserman*

Class I: Euclidean

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \varepsilon\cos\theta & -\sin\theta & t_x \\ \varepsilon\sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

(*iso*=same, *metric*=measure)

$\varepsilon = \pm 1$

orientation preserving: $\varepsilon = 1$

orientation reversing: $\varepsilon = -1$

$$\mathrm{x}' = \mathbf{H}_E\,\mathrm{x} = \begin{bmatrix} \mathbf{R} & \mathrm{t} \\ 0^\mathsf{T} & 1 \end{bmatrix} \mathrm{x} \qquad \mathbf{R}^\mathsf{T}\mathbf{R} = \mathbf{I}$$

3DOF (1 rotation, 2 translation)

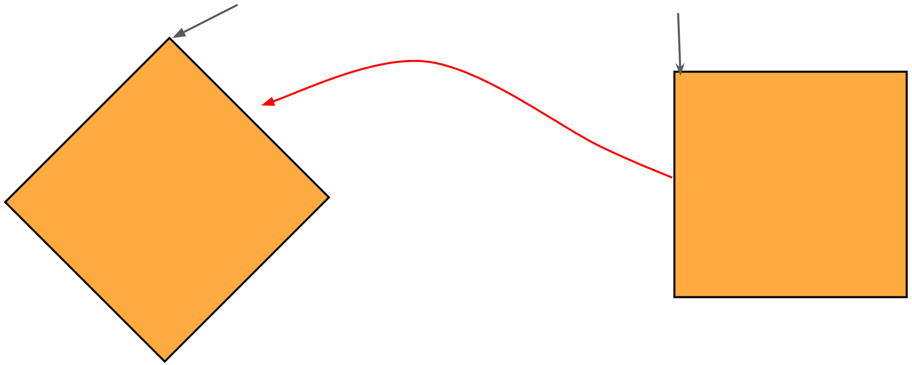
special cases: pure rotation, pure translation

Invariants: length, angle, area

Soliton
Vision for a Better World

Class I: Euclidean

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \varepsilon\cos\theta & -\sin\theta & t_x \\ \varepsilon\sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$



Θ = - 45 degrees

Class II: Similarities

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

(*isometry* + *scale*)

also known as *equi-form* (shape preserving)

$$\mathrm{x}' = \mathbf{H}_S\,\mathrm{x} = \begin{bmatrix} s\mathbf{R} & \mathrm{t} \\ 0^\mathsf{T} & 1 \end{bmatrix} \mathrm{x}$$

$$\mathbf{R}^\mathsf{T}\mathbf{R} = \mathbf{I}$$

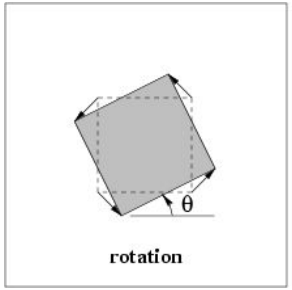
4DOF (1 scale, 1 rotation, 2 translation)

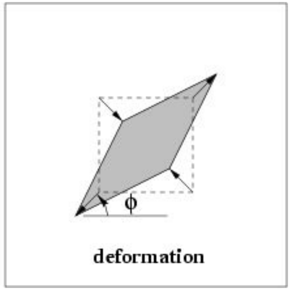
metric structure = structure up to similarity (in literature)

Invariants: ratios of length, angle, ratios of areas, parallel lines

Class III: Affine transformations

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$





$$\mathbf{x}' = \mathbf{H}_A\,\mathbf{x} = \begin{bmatrix} \mathbf{A} & \mathbf{t} \\ 0^\mathsf{T} & 1 \end{bmatrix} \mathbf{x}$$

$$\mathbf{A} = \mathbf{R}(\theta)\mathbf{R}(-\phi)\mathbf{D}\mathbf{R}(\phi) \quad \mathbf{D} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

6DOF (2 scale, 2 rotation, 2 translation)

non-isotropic scaling! (2DOF: scale ratio and orientation)

Invariants: parallel lines, ratios of parallel lengths, ratios of areas

Soliton
Vision for a Better World

Class VI: Projective transformations

$$\mathrm{x}' = \mathbf{H}_P \, \mathrm{x} = \begin{bmatrix} \mathbf{A} & \mathrm{t} \\ \mathrm{v}^{\mathsf{T}} & v \end{bmatrix} \mathrm{x}$$

$$\mathrm{v} = (v_1, v_2)^{\mathsf{T}}$$

8DOF (2 scale, 2 rotation, 2 translation, 2 line at infinity)

Action non-homogeneous over the plane

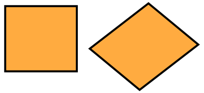
Invariants: cross-ratio of four points on a line (ratio of ratio)

Soliton
Vision for a Better World

Overview Transformations

Euclidean
3dof

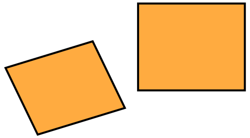
$$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$



lengths, areas.

Similarity
4dof

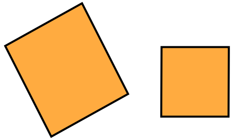
$$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$



Ratios of lengths, angles.
The circular points I,J

Affine
6dof

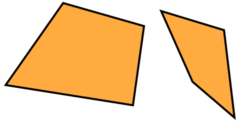
$$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$



Parallelism, ratio of areas, ratio of lengths on parallel lines (e.g midpoints), linear combinations of vectors (centroids). **The line at infinity l_∞**

Projective
8dof

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$



Concurrency, collinearity, order of contact (intersection, tangency, inflection, etc.), cross ratio

Soliton
Vision for a Better World

Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard H, Andrew Zisserman*

What are these Transformations?

Projective transformations

Definition:

A *projectivity* is an invertible mapping h from P^2 to itself such that three points x_1,x_2,x_3 lie on the same line if and only if $h(x_1),h(x_2),h(x_3)$ do.

Theorem:

A mapping $h:P^2\rightarrow P^2$ is a projectivity if and only if there exist a non-singular 3x3 matrix **H** such that for any point in P^2 represented by a vector x it is true that $h(x)=\mathbf{H}x$

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

or

$$\mathbf{x}' = \mathbf{H}\,\mathbf{x}$$

8DOF

Projectivity = collineation = projective transformation = homography

A hierarchy of transformations

Projective linear group

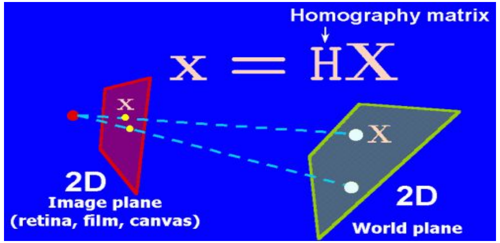
- Affine group (last row (0,0,1))
  - Euclidean group (upper left 2x2 orthogonal)
    - Oriented Euclidean group (upper left 2x2 det 1)

Alternative, characterize transformation in terms of elements or quantities that are preserved or *invariant*

- e.g. Euclidean transformations leave distances unchanged

Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann*

Projective Distortion



Ref: *ECE 661: Computer Vision, Avinash Kak, https://plus.maths.org/content/getting-picture*

Projective transformation

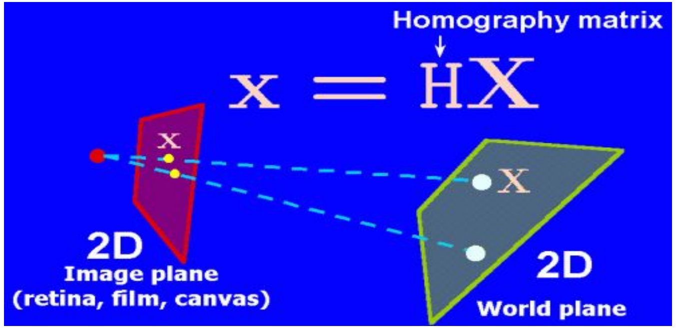
$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

or

$$\mathbf{x}' = \mathbf{H}\,\mathbf{x}$$

8DOF

Projectivity = projective transformation = homography



Removing projective distortion

select four points in a plane with known coordinates

$$x' = \frac{x'_1}{x'_3} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \qquad y' = \frac{x'_2}{x'_3} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

$$x'\left(h_{31}x + h_{32}y + h_{33}\right) = h_{11}x + h_{12}y + h_{13}$$
$$y'\left(h_{31}x + h_{32}y + h_{33}\right) = h_{21}x + h_{22}y + h_{23}$$

(linear in h_{ij})

(2 constraints/point, 8DOF ⇒ 4 points needed)

Remark: no calibration at all necessary

Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann*

Camera Model

Lens configuration (internal parameter)

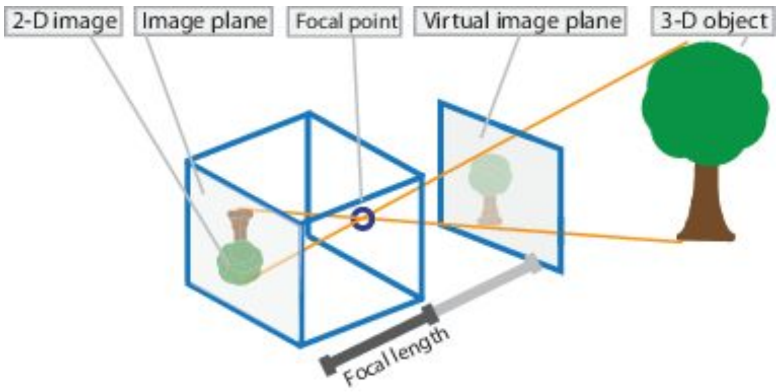
$$\begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = L\left(\mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} \right)$$

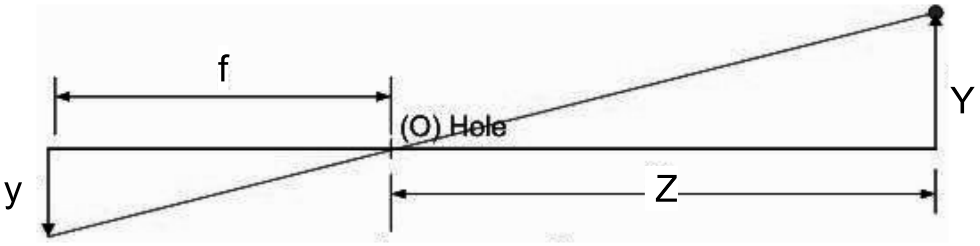
Spatial relationship between sensor and pinhole (internal parameter)

Camera body configuration (extrinsic parameter)

Pinhole Camera Model

- Simplest model of imaging process



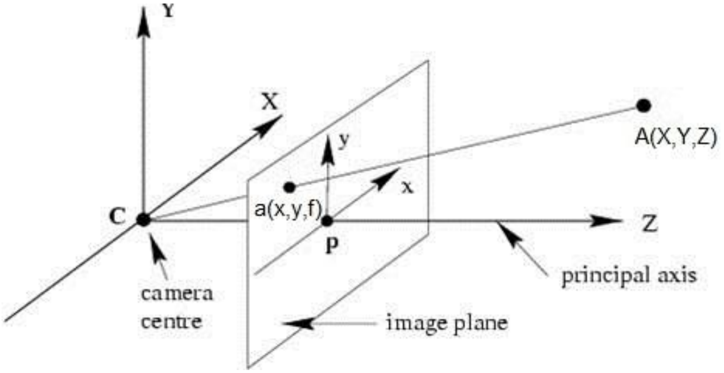


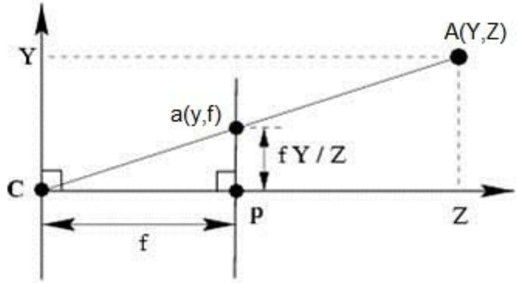
$$\frac{y}{f} = \frac{Y}{Z} \longrightarrow y = \frac{fY}{Z}$$

$$\frac{x}{f} = \frac{X}{Z} \longrightarrow x = \frac{fX}{Z}$$

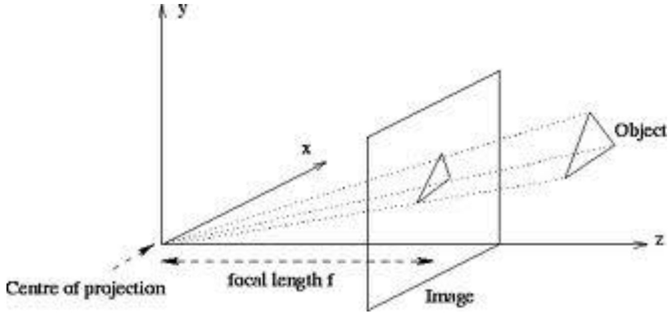
Ref : 1. *“A Flexible New Technique for Camera Calibration”, Zhengyou Zhang*
2. *https://in.mathworks.com/help/vision/ug/camera-calibration.html*
3. *https://jordicenzano.name/front-test/2d-3d-paradigm-overview-2011/camera-model/*

Pinhole Camera Model- Another Representation





View - YZ plane



Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann*

Homogeneous Representation

3D World Point $(X, Y, Z)^T \mapsto (fX/Z, fY/Z)^T$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Homogeneous form of 3D World Point

$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Thin Lens modeling matrix

Soliton
Vision for a Better World

Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann*

Modeling Camera Sensor Offset

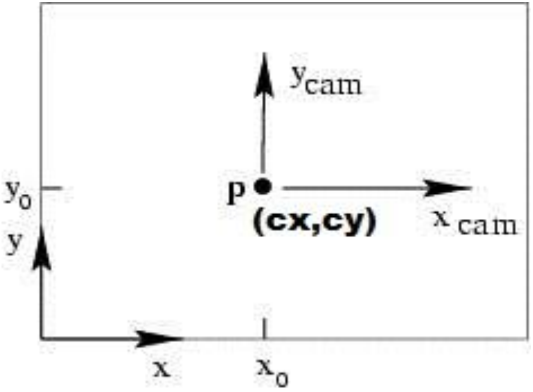


Image Plane

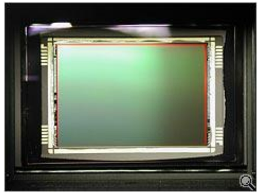
$$(X, Y, Z)^T \mapsto (fX/Z + p_x, fY/Z + p_y)^T$$

$(p_x, p_y)^T$ principal point

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_x \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann*

Modeling Camera Sensor Offset



$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{bmatrix} f_x & & p_x & 0 \\ & f_y & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

- If pixel is skewed

Homogeneous form of point in image plane

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{bmatrix} f_x & s & p_x & 0 \\ & f_y & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

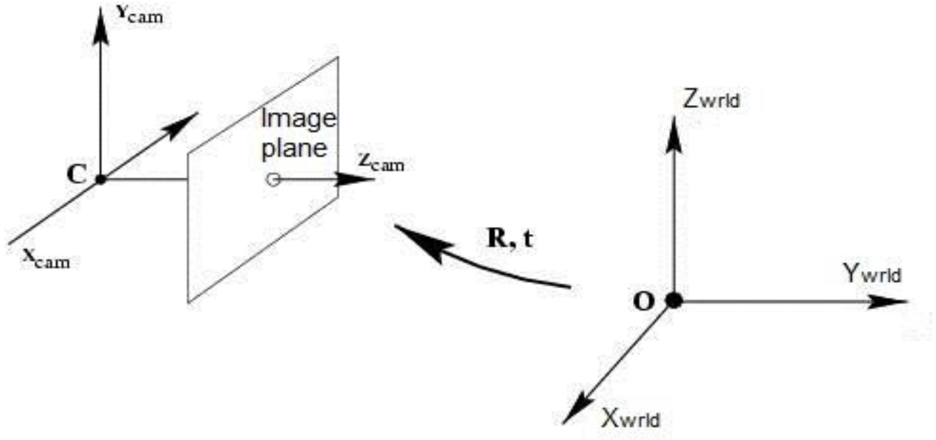
Homogeneous form of 3D World Point

***Ref:** CS766, Fall 2008, Computer Vision, Li Zhang*

Soliton
Vision for a Better World

Conversion of Coordinate System

- The pinhole model considers object points in camera coordinate system and the real world coordinate system might be different



- Transformation between two co-ordinate system is given by two factors – Rotation and Translation

Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann*

Conversion of Coordinate System

- Point in camera coordinate system to point in world coordinate system

$$P_c = R_{3x3} P_W + T_{3x1}$$

$$\begin{pmatrix} P_c \\ 1 \end{pmatrix} = \begin{bmatrix} R_{3x3} & T_{3x1} \end{bmatrix} \begin{pmatrix} P_W \\ 1 \end{pmatrix}$$

K is 3x3 matrix which defines internal parameters of the camera. It has 5 DOF

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = K_{3x3} \begin{bmatrix} R_{3x3} & T_{3x1} \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

[R T] define rotation and translation of camera these are called extrinsic parameters. It has 6 DOF

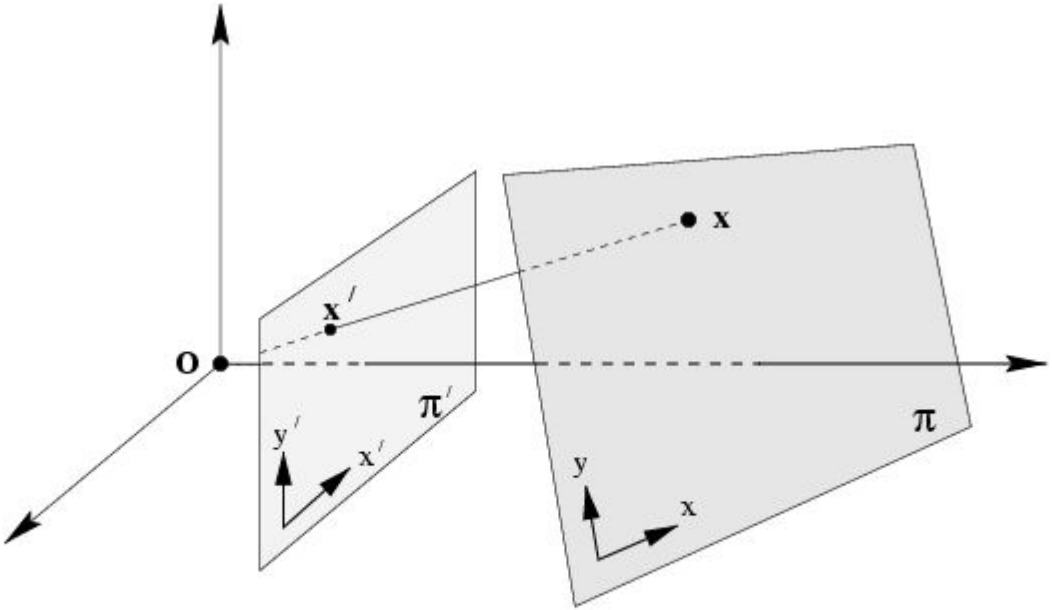
Application of Homography

- This equation can be solved if we know 3D points in real world and its corresponding 2D points in image
- Error chances are high when we use 3D points and ‘ease of use’ is low
- If all points are in single plane, it will become plane to plane transformation eliminating one of the dimension

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = K_{3x3} \begin{bmatrix} R_{3x3} & T_{3x1} \end{bmatrix} \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix}$$

Soliton
Vision for a Better World

Mapping between planes



Projection from one plane to another may be expressed by x'=Hx

Ref: Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann

Application of Homography

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = K_{3x3}\begin{bmatrix} R_{3x2} & T_{3x1} \end{bmatrix}\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

Plane to plan transformation (H)

$$p_{cam} = H_{3x3} P_{World}$$

$$H = K_{3x3}\begin{bmatrix} R_{3x2} & T_{3x1} \end{bmatrix}$$

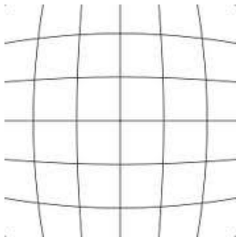
Given set of corresponding points in real world plane (checkerboard) and point in image we can find the H and decompose H into K, R and T

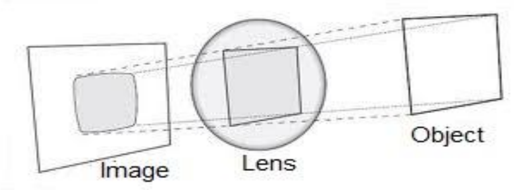
Soliton
Vision for a Better World

Lens effect

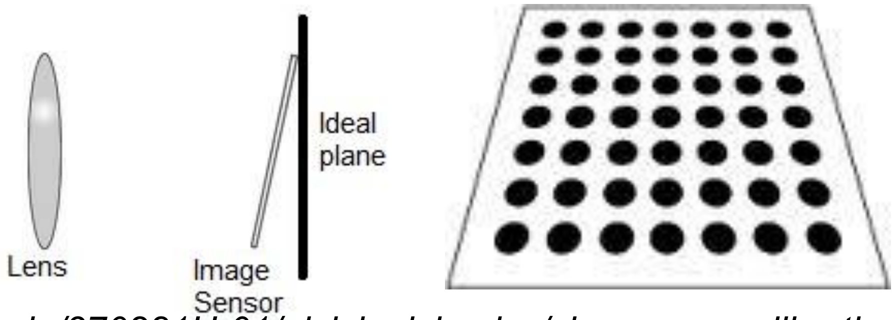
Camera model doesn't consider lens effects

- Lens – to focus light and converge
- Distortions
  - Radial Distortion – shape of lens





  - Tangential Distortion – image sensor not parallel to lens



Ref:http://zone.ni.com/reference/en-XX/help/370281U-01/nivisionlvbasics/choose_a_calibration_type/

Overview of Camera Calibration

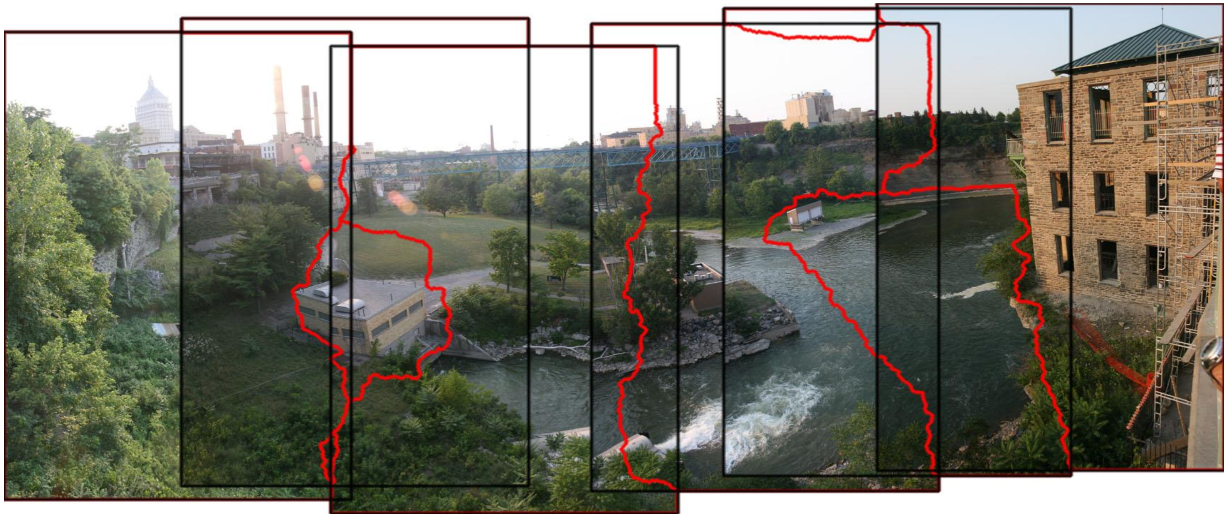
- Object points - known object plane
- Image points - Detection of feature points in image
- Homography matrix using correspondence between image points and object points.
- Decompose homography matrix to K, R and T
- Follow the above procedure for large samples
- Result : Intrinsic matrix K

Image Stitching

EE 596 Autumn 2015: HW3 KeyPoint Matching and Stitching

Image Stitching

1. Repeat for all images
    a. Detect 2D Features points in both the images
    b. Find match between 2D features of both images
    c. Find Homography between two images using matched points (Do RANSAC to reduce effect of outliers)
    d. Warp the image w.r.t any particular one image
    e. Merge the images
2. Color Blend the final image



https://en.wikipedia.org/wiki/Image_stitching

Decomposition of projective transformations

$$\mathbf{H} = \mathbf{H}_S \mathbf{H}_A \mathbf{H}_P = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{v}^\mathsf{T} & v \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{t} \\ \mathbf{v}^\mathsf{T} & v \end{bmatrix}$$

$$\mathbf{A} = s\mathbf{R}\mathbf{K} + \mathbf{t}\mathbf{v}^\mathsf{T}$$

decomposition unique (if chosen s>0)

$\mathbf{K}$ upper-triangular, $\det \mathbf{K} = 1$

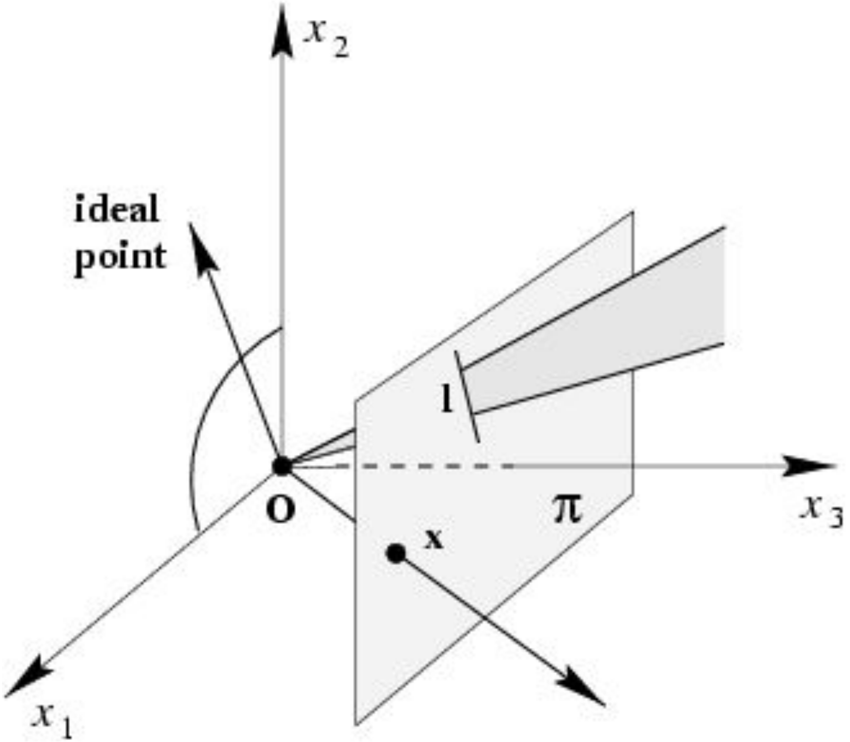
Example:

$$\mathbf{H} = \begin{bmatrix} 1.707 & 0.586 & 1.0 \\ 2.707 & 8.242 & 2.0 \\ 1.0 & 2.0 & 1.0 \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} 2\cos 45^\circ & -2\sin 45^\circ & 1.0 \\ 2\sin 45^\circ & 2\cos 45^\circ & 2.0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Soliton
Vision for a Better World

A model for the projective plane

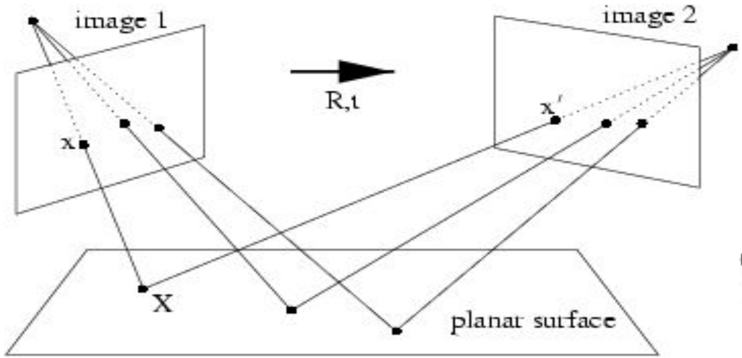


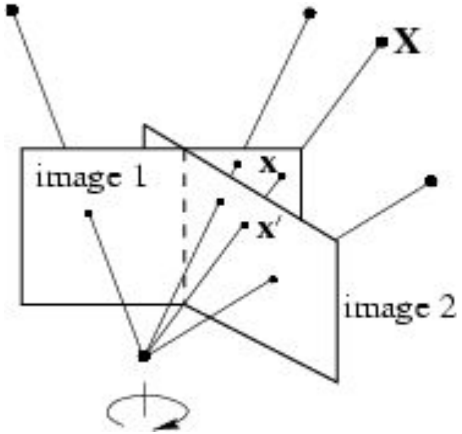
exactly one line through two points

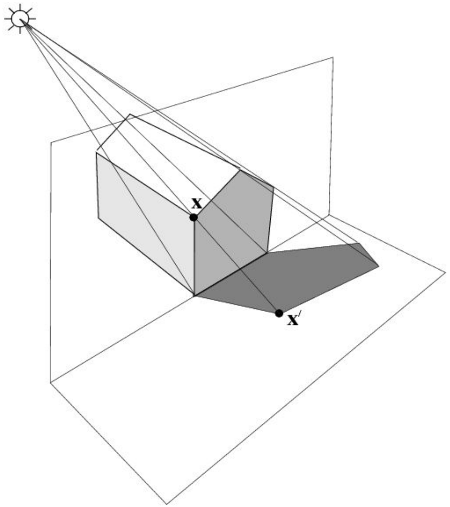
exactly one point at intersection of two lines

Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann*

More examples







Ref: *Multiple View Geometry in Computer Vision (Second Edition) : Richard Hartley, Andrew Zissermann*

Action of affinities and projectivities on line at infinity

$$\begin{bmatrix} \mathbf{A} & \mathbf{t} \\ 0^{\mathsf{T}} & v \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \mathbf{A}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ 0 \end{pmatrix}$$

Line at infinity stays at infinity,
but points move along line

$$\begin{bmatrix} \mathbf{A} & \mathbf{t} \\ \mathrm{v}^{\mathsf{T}} & v \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \mathbf{A}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ v_1 x_1 + v_2 x_2 \end{pmatrix}$$

Line at infinity becomes finite,
allows to observe vanishing points, horizon,

Session II (ML)

- ➔ Object Classification
- ➔ Classification: Simple Version
- ➔ Features
- ➔ Classification: CV based
- ➔ Linear classifiers
- ➔ SVM

http://lodgiq.com/improved-pricing-with-machine-learning/

Soliton
Vision for a Better World

What is Machine Learning?

- Arthur Samuel (1959): Field of study that gives computers the ability to learn without being explicitly programmed
- A well-posed learning problem (1998): A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance measure P, as measured by P, improves with experience E.

Soliton
Vision for a Better World

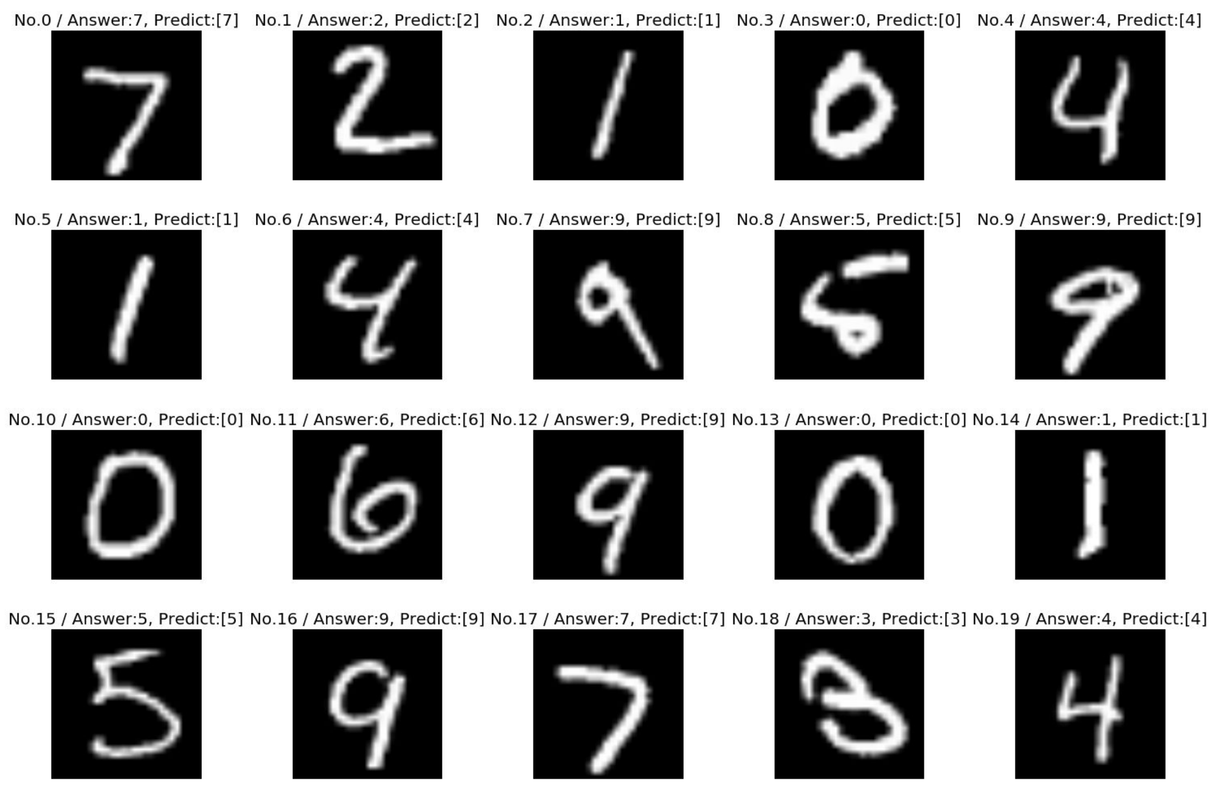
Credits: cs229, Andrew Y. Ng

Classification

Handwritten character Classification

Q: As a Human, how do I learn to classify?

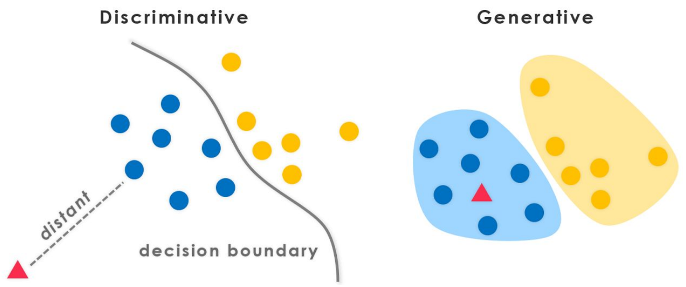
1. Learn Patterns
2. Learn Rules



http://corochann.com/mnist-inference-code-1202.html

Soliton
Vision for a Better World

What is Machine Learning?



- Set of training samples : $[x_1,....x_n]$ & $[y_1.....y_n]$
- Learn discriminatively or generativity
- Optimizing the loss function
  - Update / change the coefficients such that they minimize the error
  - Perform till convergence
- Getting optimal values for ‘*w*’ is called learning
- Now for a new x, you predict it’s y

Soliton
Vision for a Better World

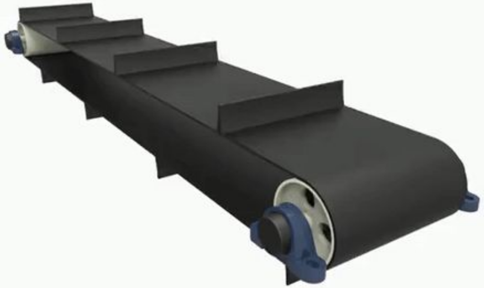
Credits: cs229, Andrew Y. Ng
Img Ref - https://datawarrior.wordpress.com/2016/05/08/generative-discriminative-pairs/

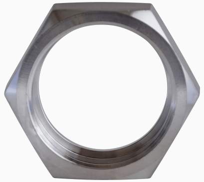
Simple Problem

How to classify a new image into any of the two categories?

Simplified Problem

- Conveyor Belt: 2D Image of Nut & Bolt
- Supervised Learning Problem
  - Train on available data
  - Test on new data





Nut

Bolt

Nut: https://www.toplineonline.com/product/25h-1-wrench-aluminum-single-end-for-hex-nut/
Bolt: http://www.armafix.com/products/threaded-fasteners/hex-screws-bolts

Rules

We can classify a new image into any of these two categories by forming a set of rules.

Q: List down few rules by which we can classify

Bolts	Nuts
1. Longer 2. Thinner 3. Cylindrical in shape 4. More compact	1. Circular 2. Has cavity in the centre 3. Less area

Soliton
Vision for a Better World

Data

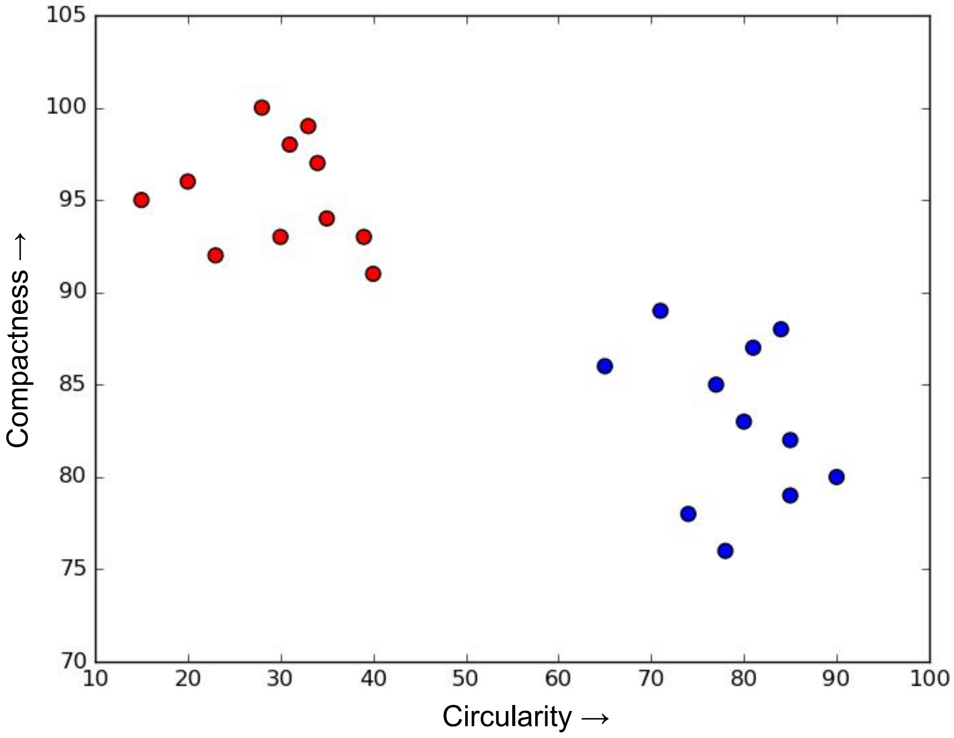
The graph shows the distribution of data with respect to the below features

Features

1. Circularity
2. Compactness

Now, How can you classify the data?

● Bolt ● Nut



Soliton
Vision for a Better World

Classification

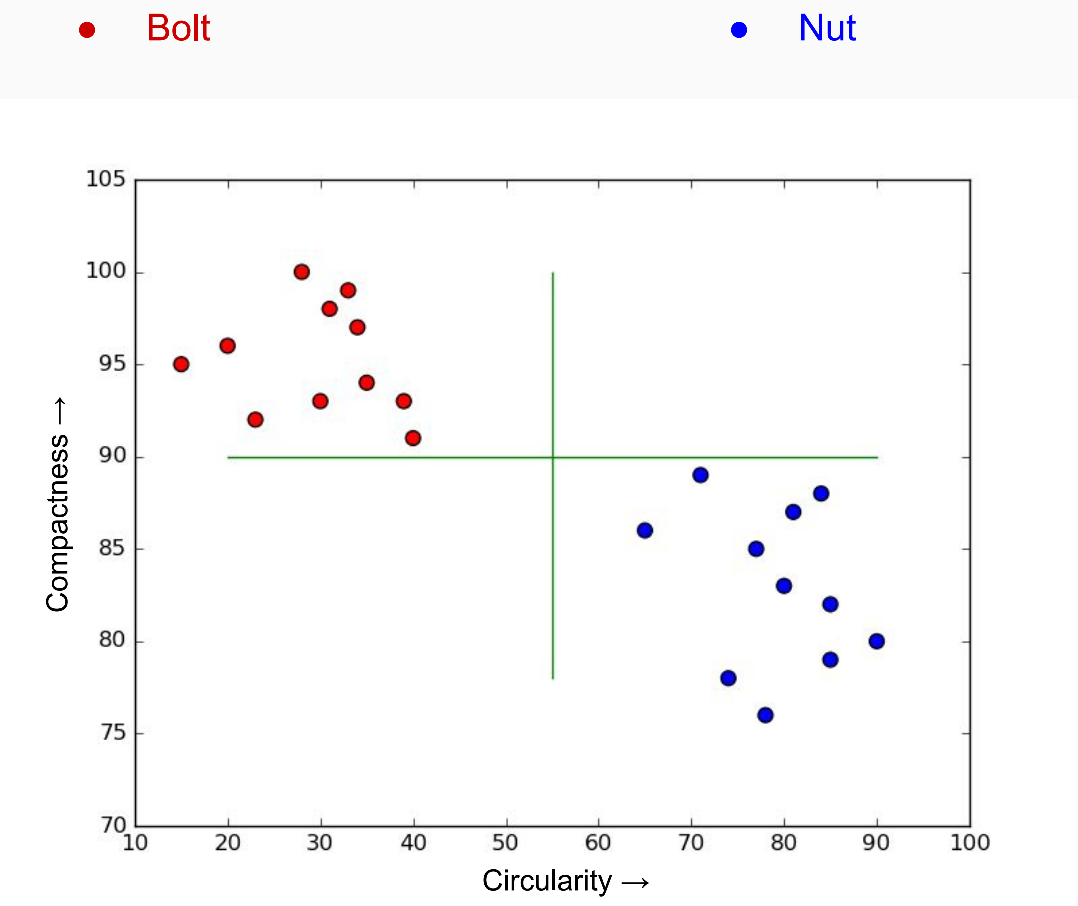
It is a simple if with two conditions

```
if Circularity > 55 and
Compactness < 90:

     Nut

elif Circularity < 55 and
Compactness > 90:

     Bolt
```

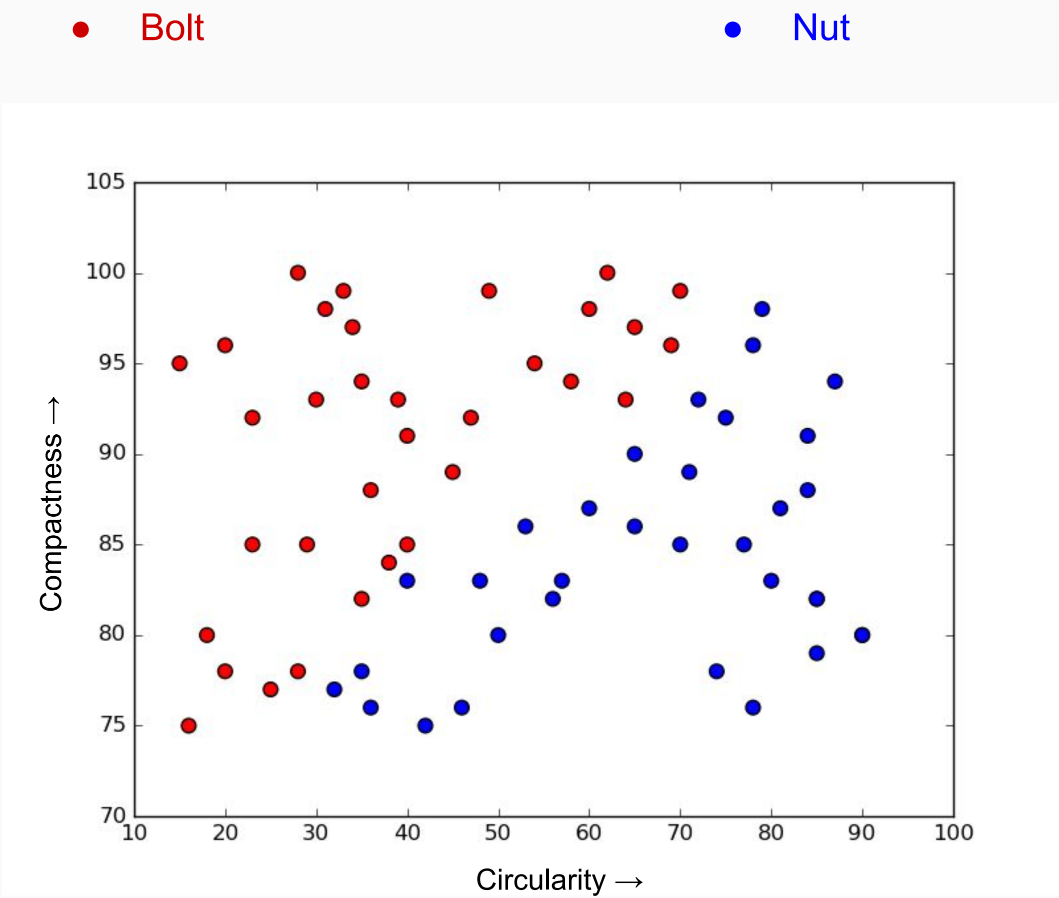


Soliton
Vision for a Better World

Complex Data

The Data becomes more complex to classify using a simple If Classifier

Now, How can you classify the data?

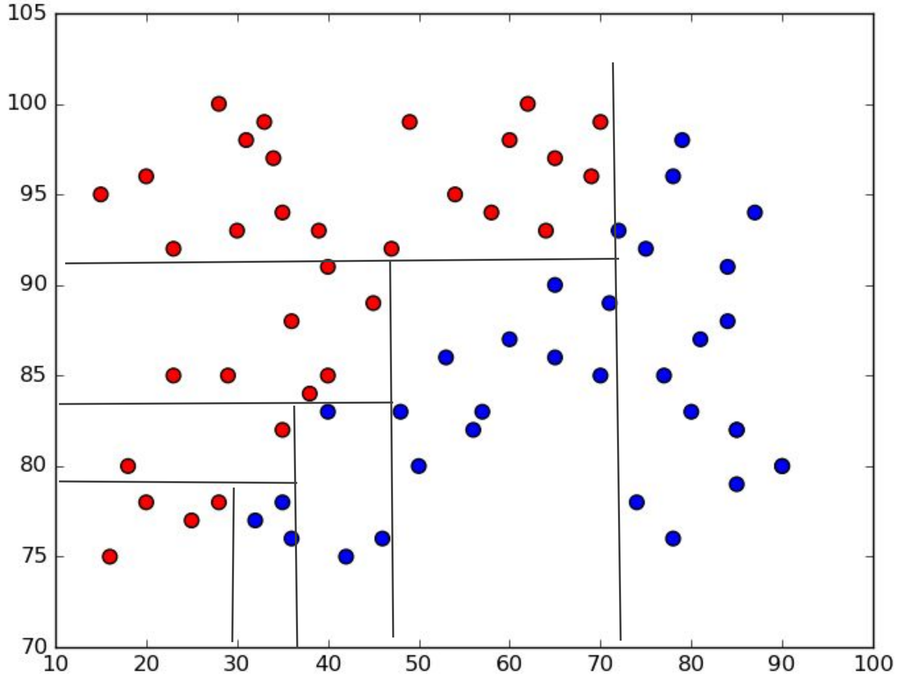


Soliton
Vision for a Better World

Decision Tree

- Non-parametric Supervised Learning
- Approximating data by a set of if-then-else decision rules
- Greedy algorithm
- Whitebox Model :)

● Bolt ● Nut



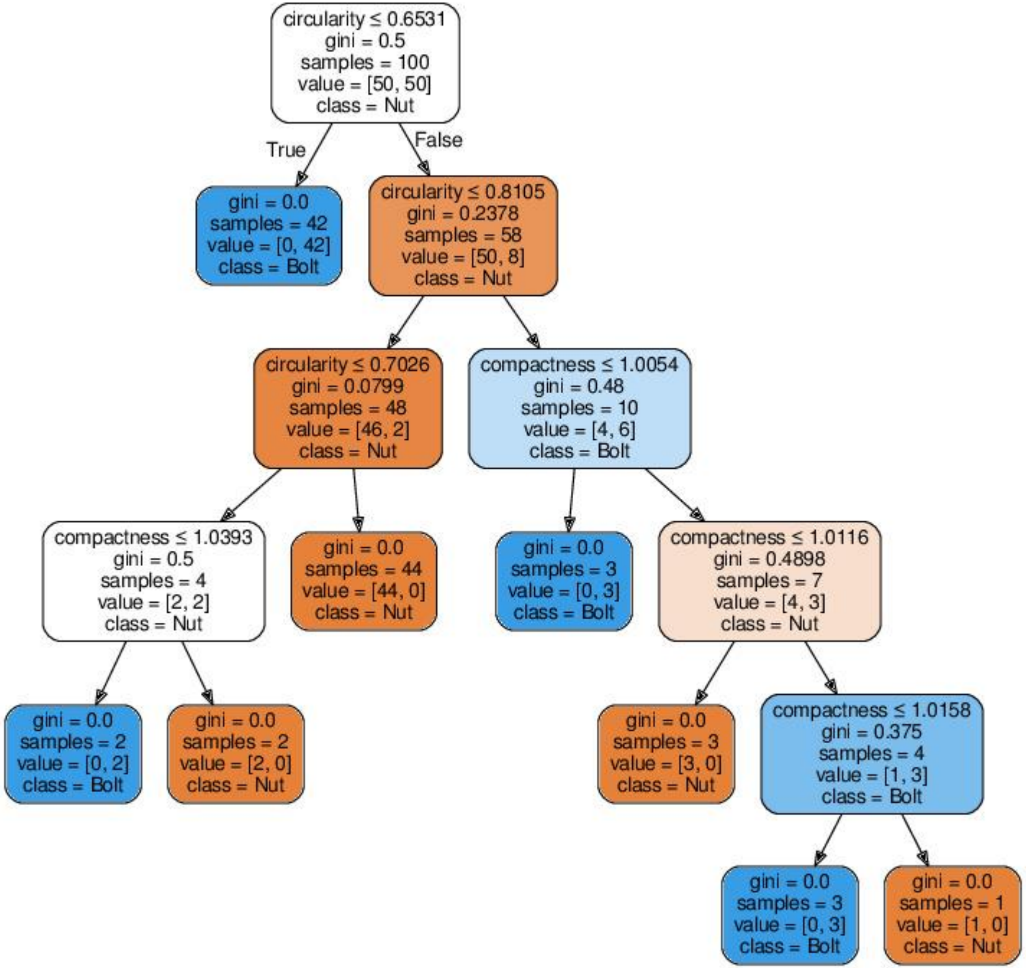
Soliton
Vision for a Better World

Decision Tree

Decision Tree generated for the Nuts and Bolts classification problem

Cons

- May grow huge and hard to manage / visualize
- Problem of overfitting

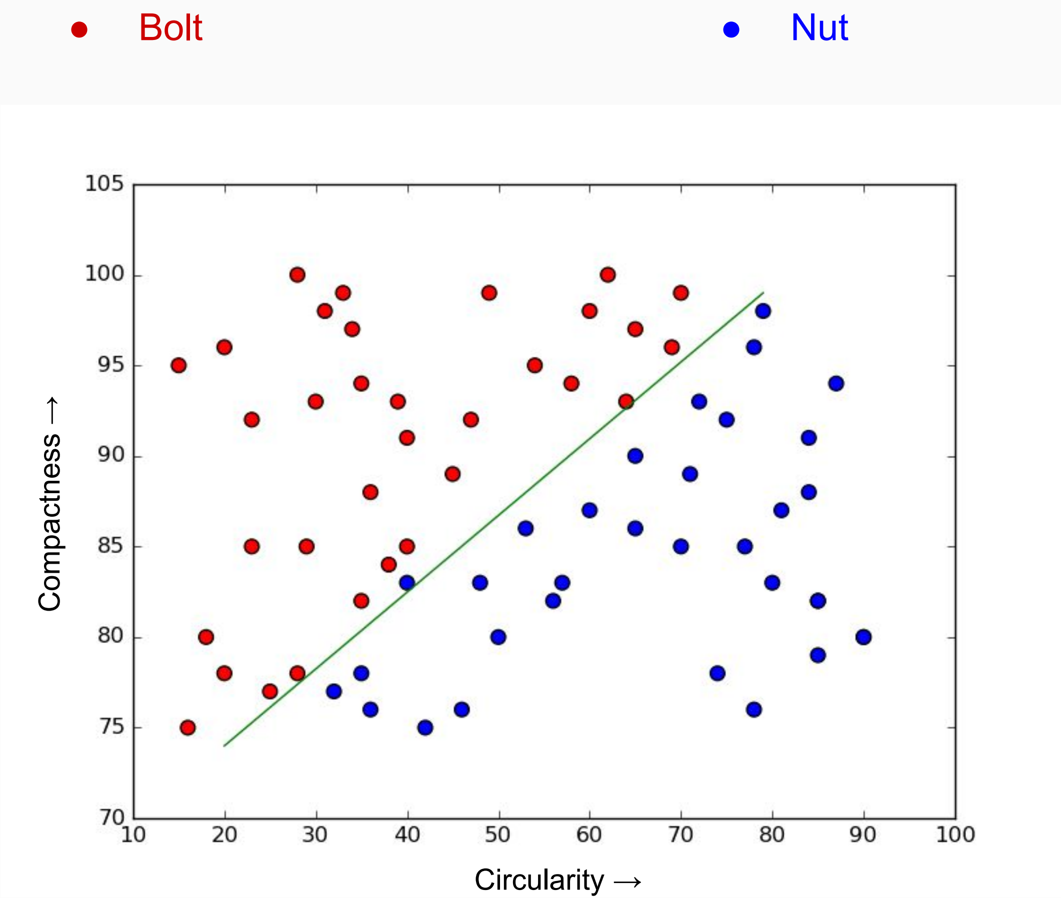


Soliton
Vision for a Better World

Linear Classifier

A line should classify the data

But, How do I come up with a Line which separates both the classes of data optimally?

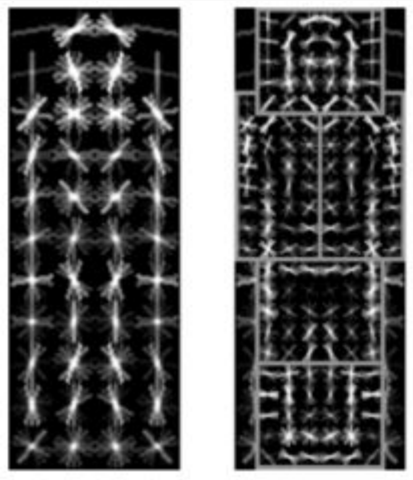


Soliton
Vision for a Better World

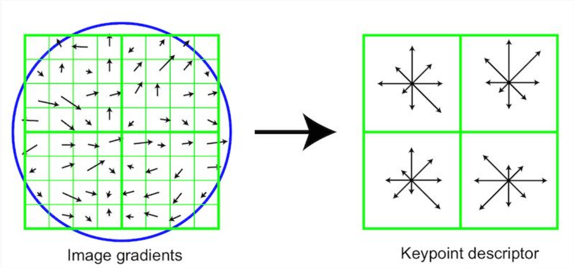
Real World features

In Machine Learning terms, we call them as features

Low-level Features



HoG

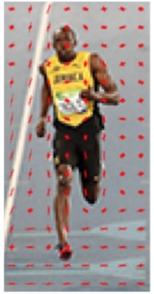


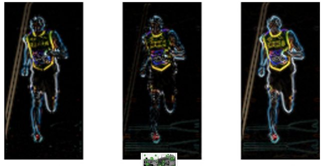
SIFT

https://www.pinterest.com/pin/348747564886840675/
http://www.computervisionblog.com/2015/01/from-feature-descriptors-to-deep.html

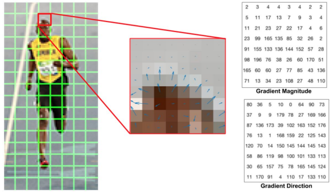
Soliton
Vision for a Better World

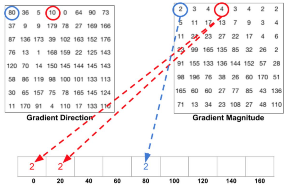
HoG



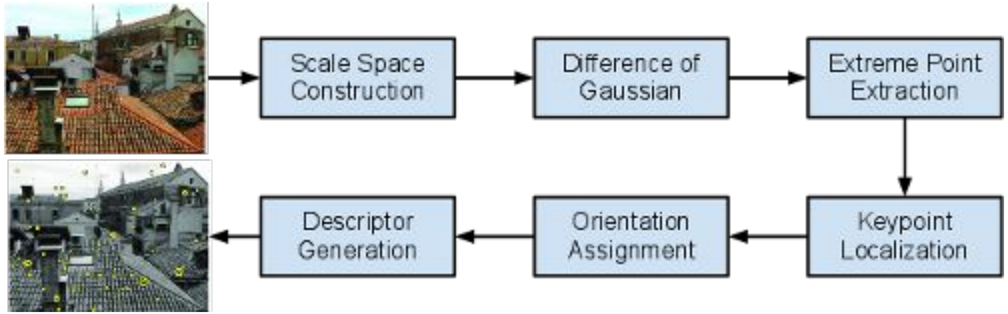


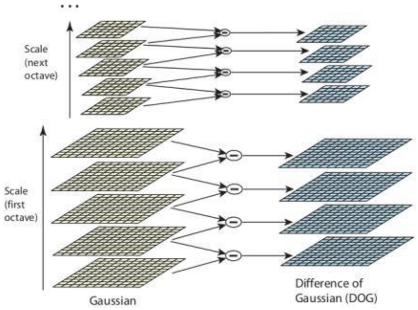
Left : Absolute value of x-gradi[...]er : Absolute value of y-gradient.
Right : Magnitude of gradient.

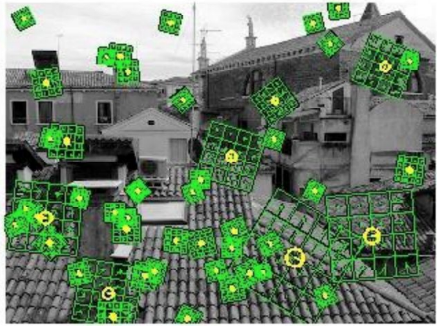




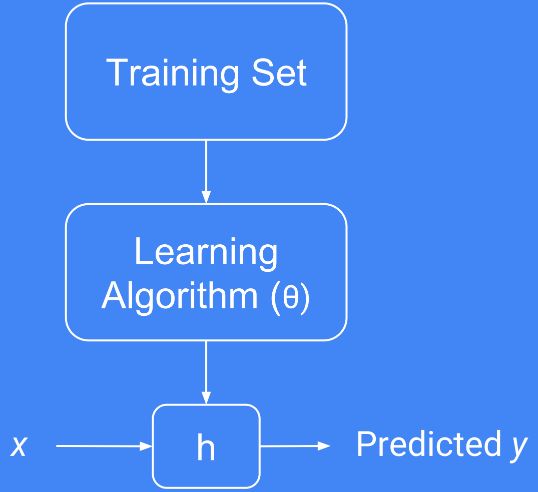
SIFT







Parts of an ML Algo



Inference

- Hypothesis space, the set of possible hypotheses it can come up with in order to model the unknown target function by formulating the final hypothesis

- An example (linear function) $h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$

Learning

X - Training data   Y - Labels   h(θ) = Predicted label

- Cost function => J(θ) = h(θ) - y   [Predicted - Actual]

- Update Rule $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta).$

Soliton
Vision for a Better World

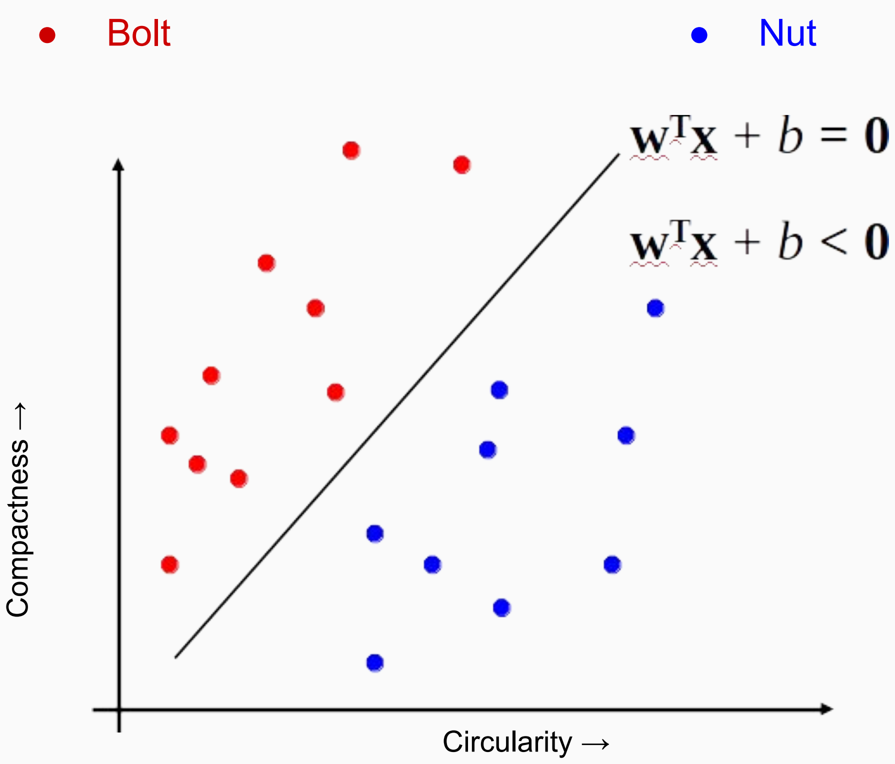
Credits: cs229, Andrew Y. Ng

Logistic Regression

The Line can be drawn from the learned parameters

$$\theta_0 + \theta_1 X_1 + \theta_2 X_2 = 0$$

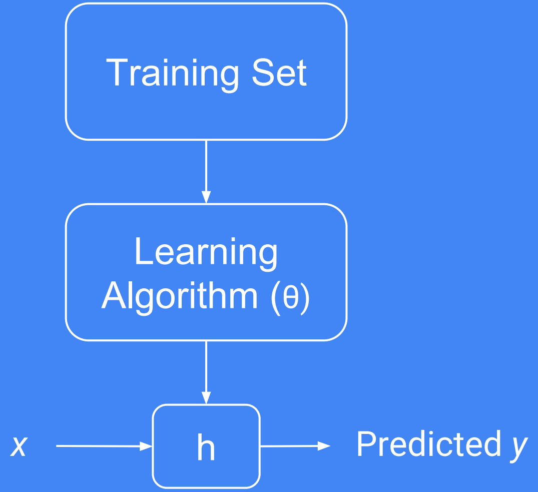
$$X_1 = \theta_0/\theta_1 + \theta_2 X_2 / \theta_1$$



Soliton
Vision for a Better World

Credits: cs229, Andrew Y. Ng

Logistic Regression



Learning

X - Training data   Y - Labels   h(θ) = Predicted label

- This can be simplified as $h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$
- Cost function $J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2.$
- Update Rule $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta).$

Gradient Descent: Starts with some initial value of θ and repeatedly performs update

Soliton
Vision for a Better World

Credits: cs229, Andrew Y. Ng

Gradient Descent

- The magnitude of the update is proportional to the error term $(y(i) - h_\theta(x(i)))$
- If a training example on which our prediction nearly matches the actual y (i), there is little need to change the parameters

Batch Gradient Descent

Repeat until convergence {

$$\theta_j := \theta_j + \alpha \sum_{i=1}^{m} \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)} \qquad \text{(for every } j\text{)}.$$

}

Stochastic Gradient Descent (SGD)

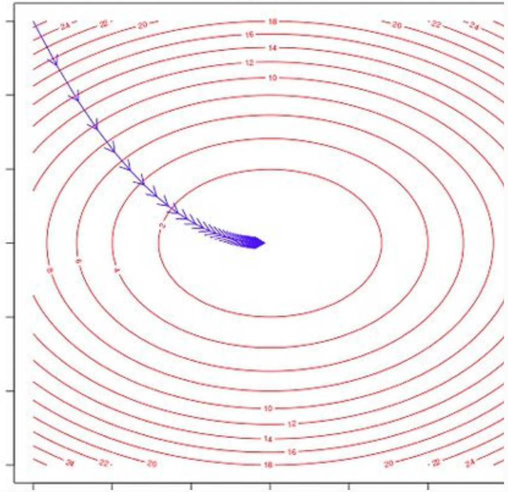
Loop {

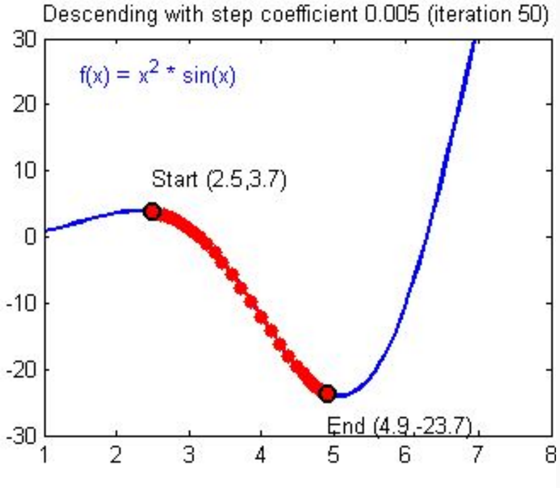
for i=1 to m, {

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)} \qquad \text{(for every } j\text{)}.$$

}

}





Credits: cs229, Andrew Y. Ng

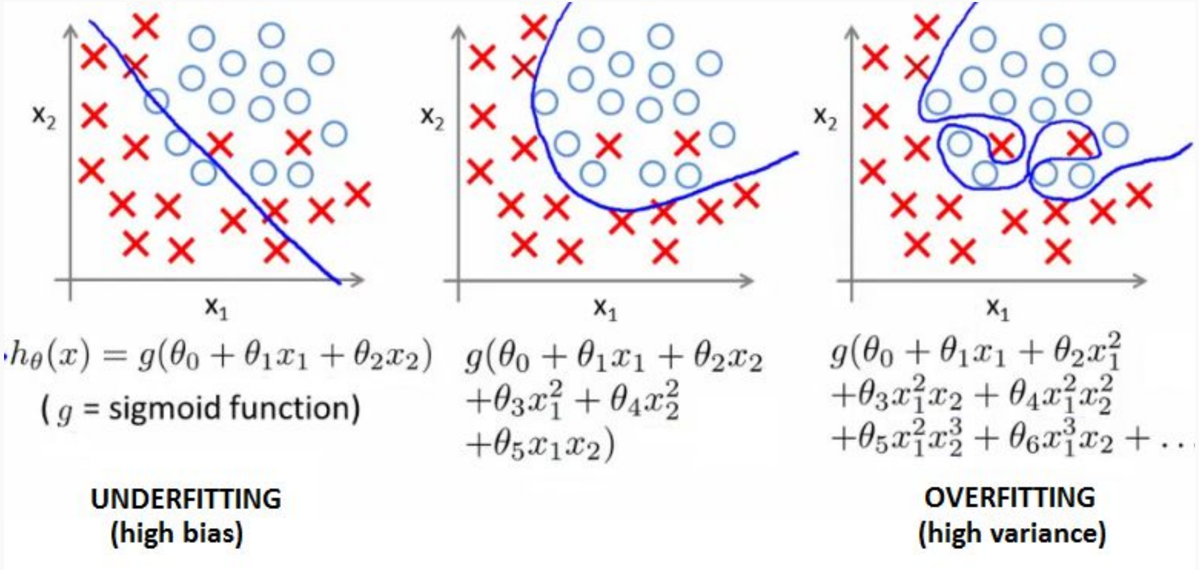
https://giphy.com/gifs/gradient-6QlTwkigqg4yk

https://hackernoon.com/life-is-gradient-descent-880c60ac1be8

Overfitting & Underfitting

How to Fix

1. Cross Validation
2. Feature selection
3. Regularization



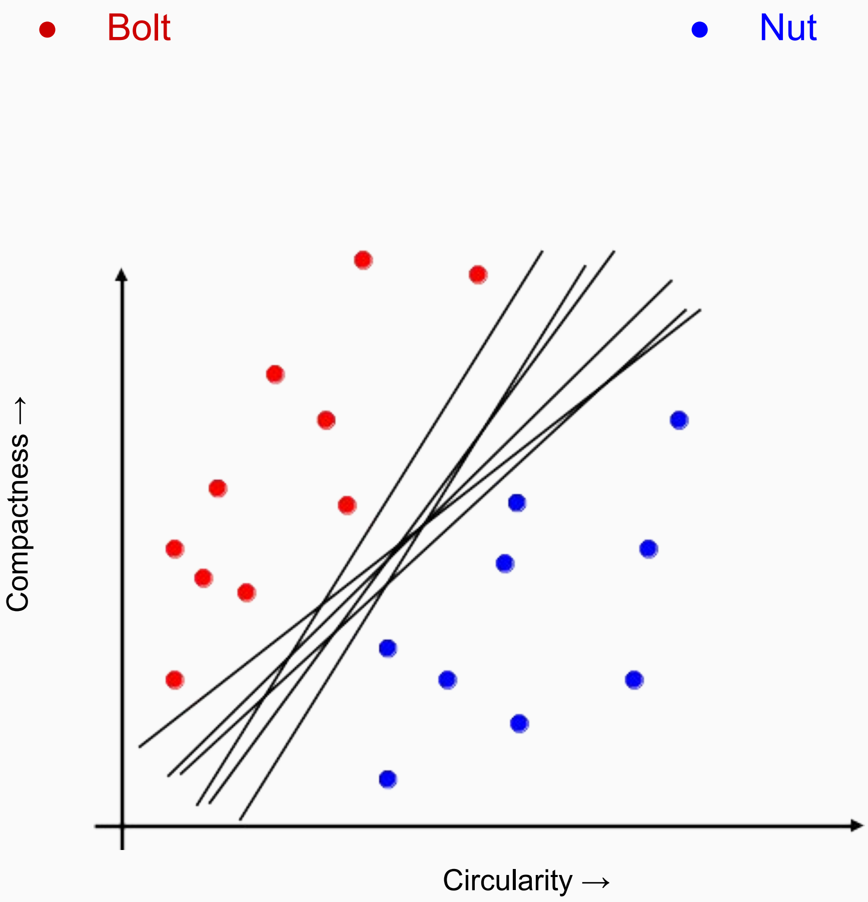
Regularization

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Regularization parameter which penalizes higher order θ

More intelligent classifier

Which of the linear separators is optimal?



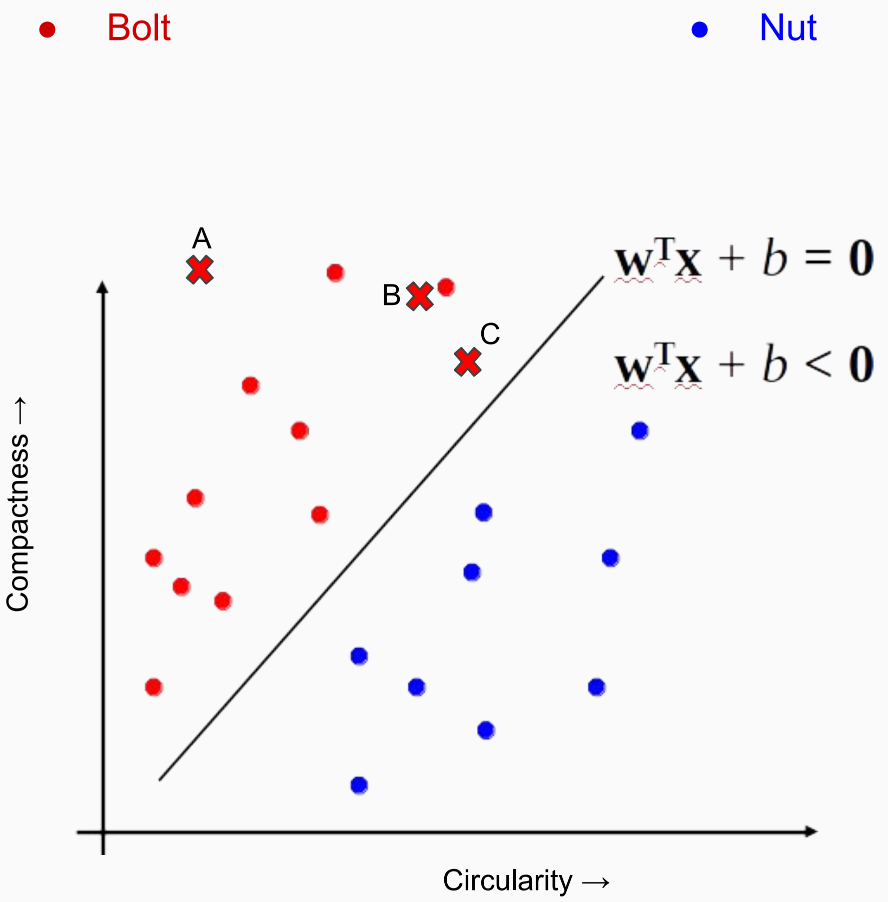
Margins

Let's take the Logistic regression example

Among A, B and C which data point would you confidently classify as Bolt?

If the point is far from the separating hyperplane, we may be significantly more confident in our predictions

Find a decision boundary that allows us to make all CORRECT and CONFIDENT predictions



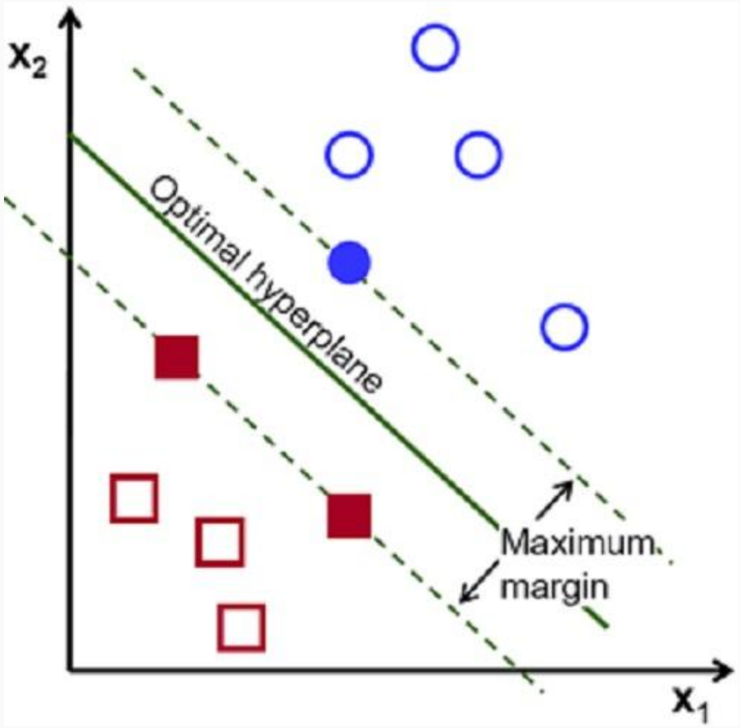
Soliton
Vision for a Better World

SVM

Optimal Margin Classifier

- Examples closest to the separator are support vectors.
- Margin **ρ** of the separator is the distance between support vectors
  $r = (W^TX + b) / ||W||$
- Functional Margin
  $\hat{\gamma}^{(i)} = y^{(i)} (w^T x + b)$
-

- Bolt
- Nut



Compactness →

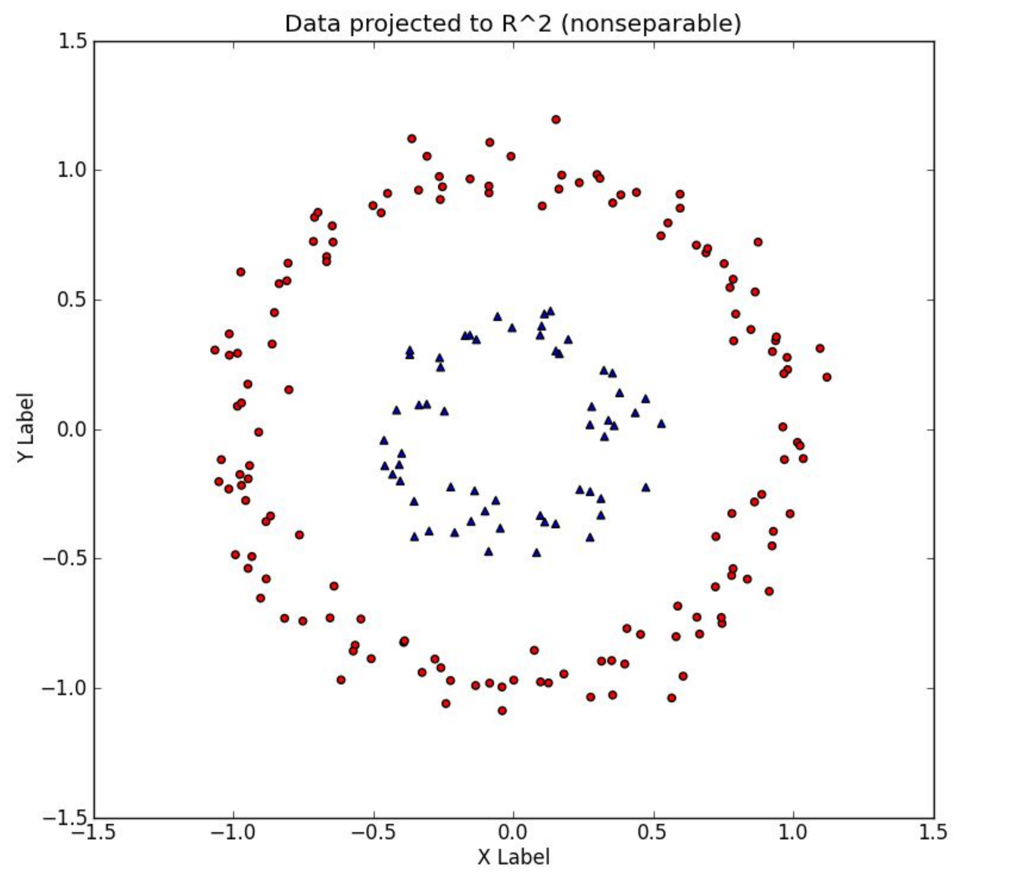
Circularity →

Credits: cs229, Andrew Y. Ng

Nonseparable Data

We learned that SVM is a Linear Classifier

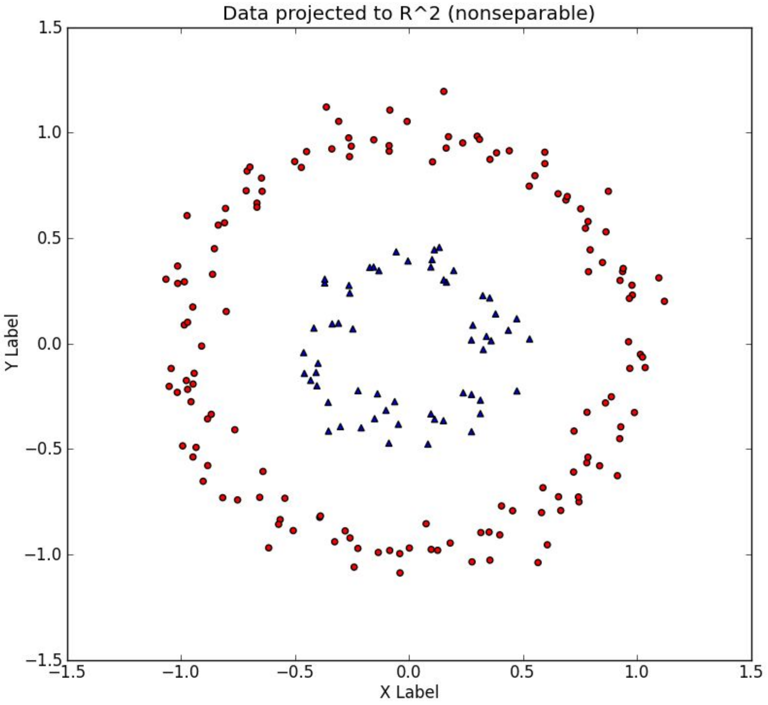
Can SVM classify the given data?

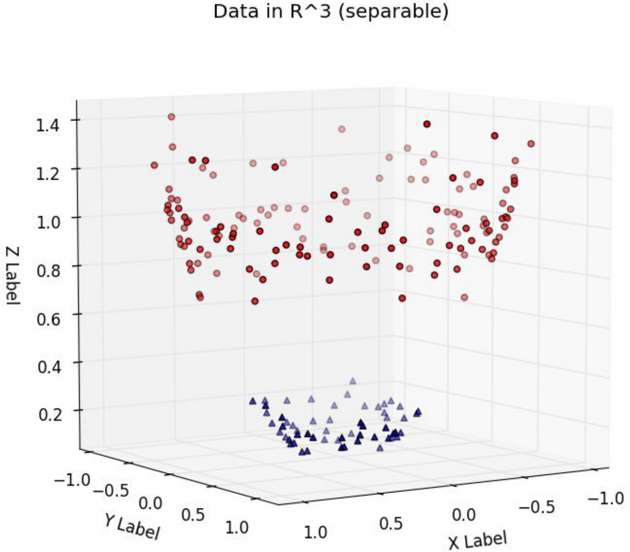


Ref: http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html

Soliton
Vision for a Better World

Project data to Higher Dimension

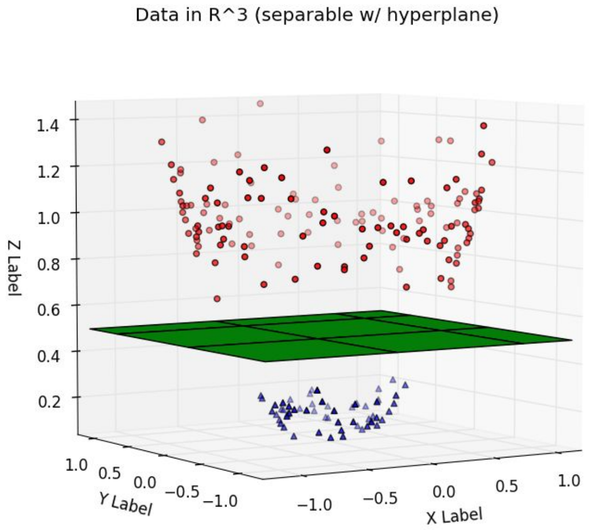


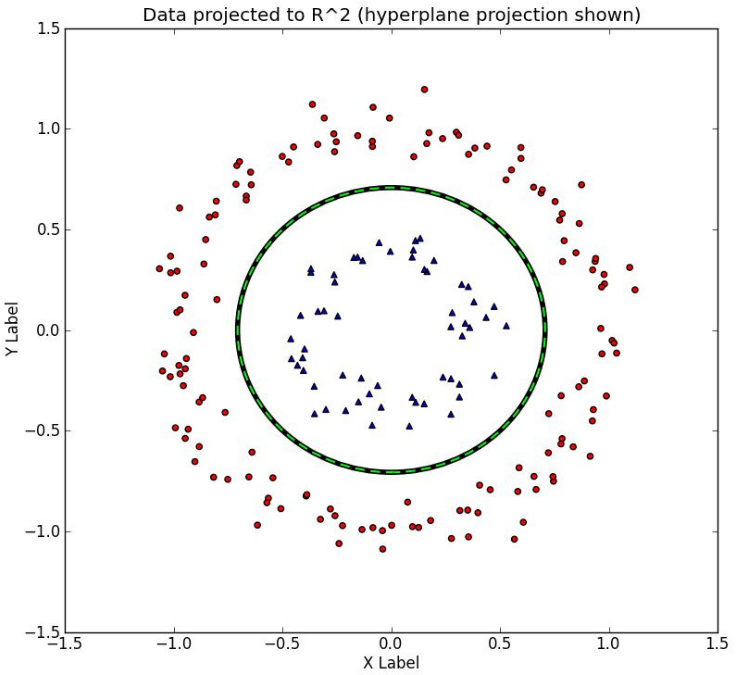


$[x_1, x_2] = [x_1, x_2, x_1^2+x_2^2]$

Ref: http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html

Hyperplane re-projection in 2D





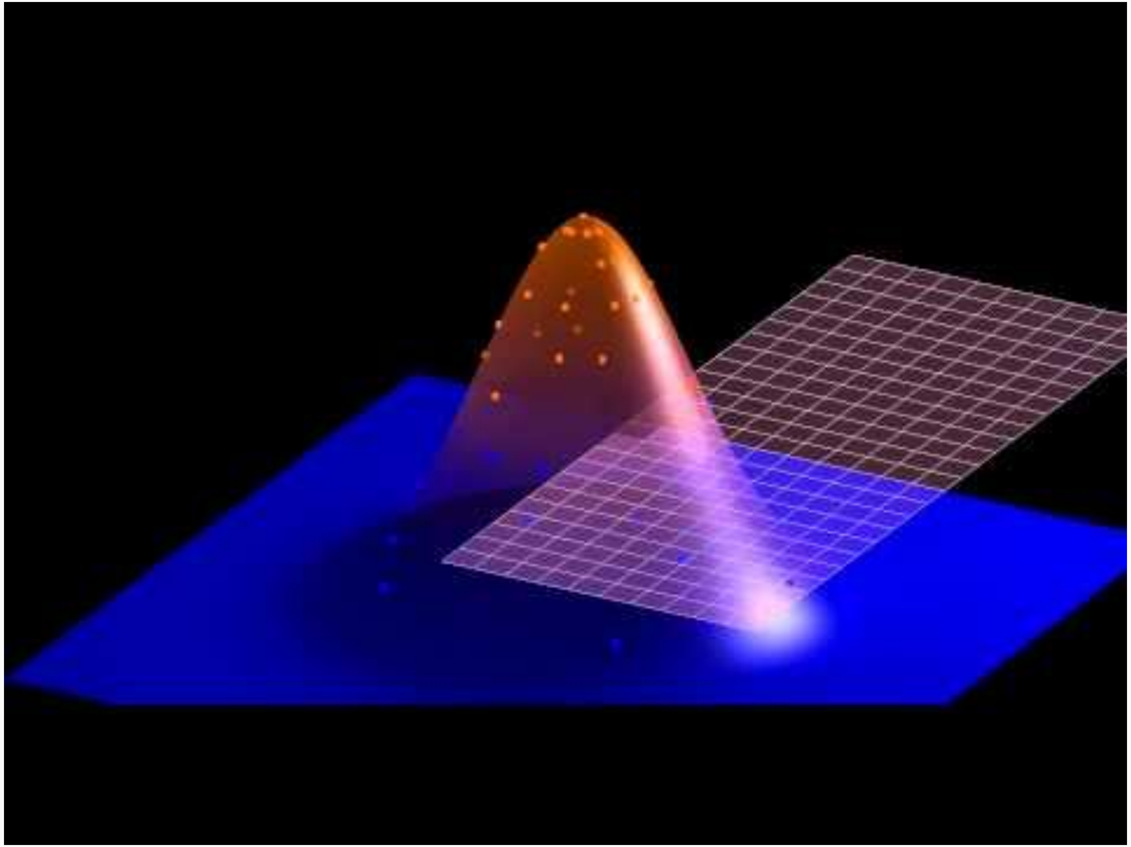
Soliton
Vision for a Better World

Ref: http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html

Demo

Just,

- Project the data into higher Dimension
- Find Hyperplane
- Re-project the Hyperplane back to original dimension



https://www.youtube.com/watch?v=9NrALgHFwTo

Soliton
Vision for a Better World

Kernel Trick

It is so simple. Is it True?

Higher Dimensional Projection is Expensive

- Impractical for Large Dimensions
- Huge memory and Computation are required
- Transformation from N dimension to M Dimension is **$O(N^2)$** expensive

We only need Dot Products..! Not the High Dimension Data

$$w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}.$$

$$\begin{aligned} w^T x + b &= \left(\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}\right)^T x + b \\ &= \sum_{i=1}^{m} \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b. \end{aligned}$$

$$\phi(x) = \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_1x_3 \\ x_2x_1 \\ x_2x_2 \\ x_2x_3 \\ x_3x_1 \\ x_3x_2 \\ x_3x_3 \end{bmatrix}.$$

$O(N^2)$

$$K(x, z) = \varphi(x)^T \varphi(z)$$

$$K(x, z) = (x^T z)^2.$$

$$\begin{aligned} K(x,z) &= \left(\sum_{i=1}^{n} x_i z_i\right)\left(\sum_{j=1}^{n} x_j z_j\right) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j z_i z_j \\ &= \sum_{i,j=1}^{n} (x_i x_j)(z_i z_j) \end{aligned}$$

$O(N)$

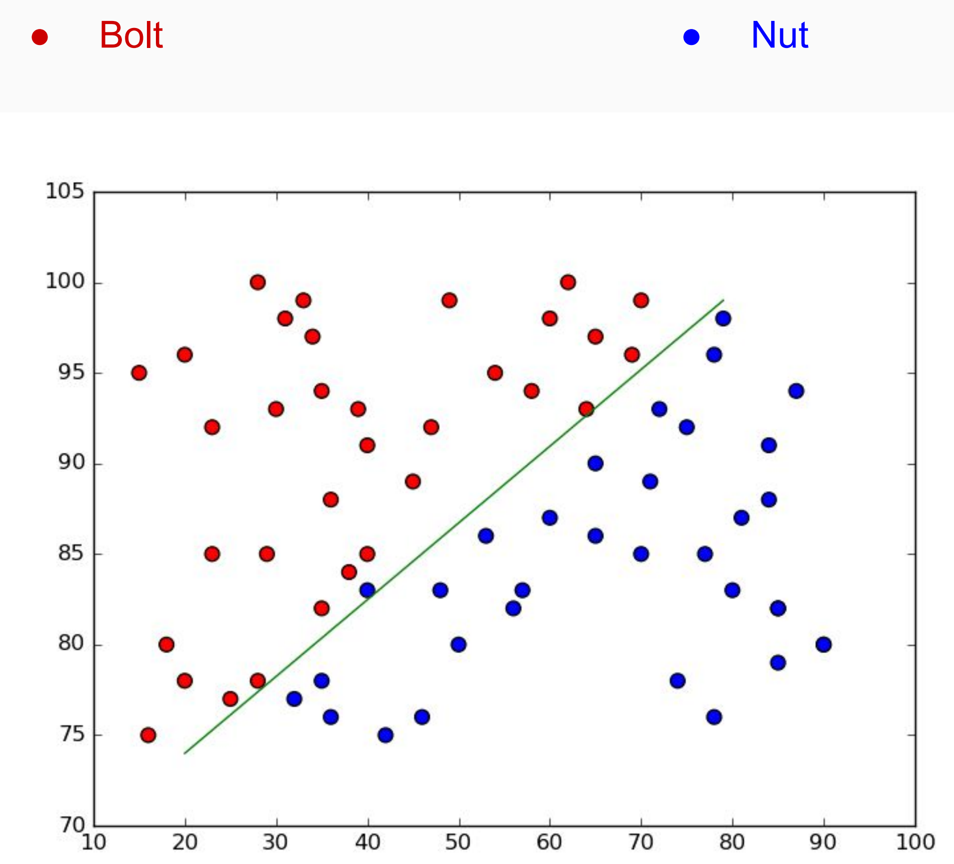
Computationally Faster and No extra memory needed

Soliton
Vision for a Better World

Credits: cs229, Andrew Y. Ng

SVM

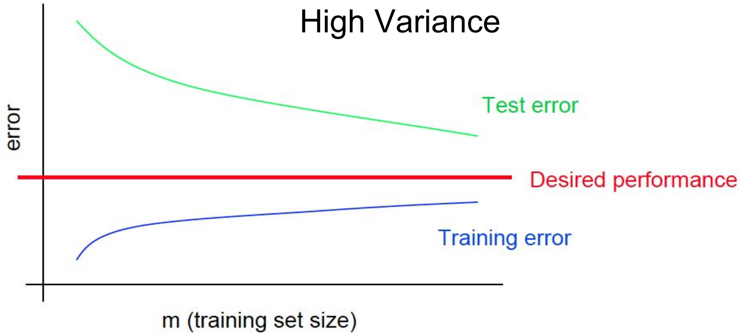
Find the Hyperplane which optimizes the Geometric margin iteratively

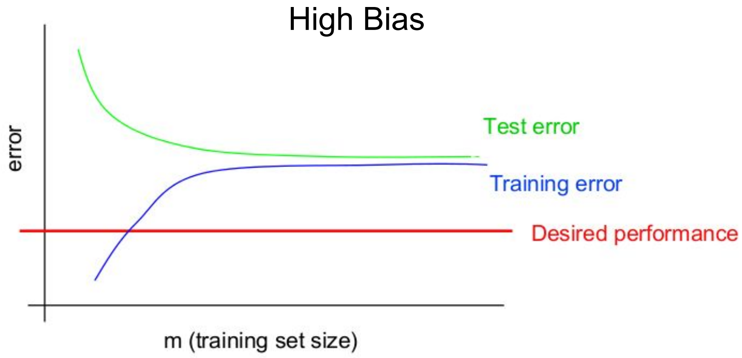


Soliton
Vision for a Better World

ML Advice - Diagnostics

Try getting more training examples	Fixes high Variance
Try a smaller set of features	Fixes high Variance
Try a larger set of features	Fixes high Bias
Try changing features (e.g, email header features)	Fixes high Bias
Run gradient descent for more iterations	Fixes optimization algorithm
Try Newton's method	Fixes optimization algorithm
Use a different value for λ	Fixes optimization objective
Try using an SVM	Fixes optimization objective





Soliton
Vision for a Better World

Credits: cs229, Andrew Y. Ng

ML Advice

Rule #1: Plot the Data

Questions to ask:

Is the Algorithm converging?

Are you optimizing the right function?

Is the value for λ is correct?

Is the value for C is correct?

Are initial parameters correct?

Soliton
Vision for a Better World

Credits: cs229, Andrew Y. Ng

Error Analysis

- Helps to understand how much error is attributable to each component?
- Helps to identify Poor components by which we can improve performance
- List down accuracy **DROP** after introducing each component.
- Plug in ground-truth for each component, and see how accuracy changes

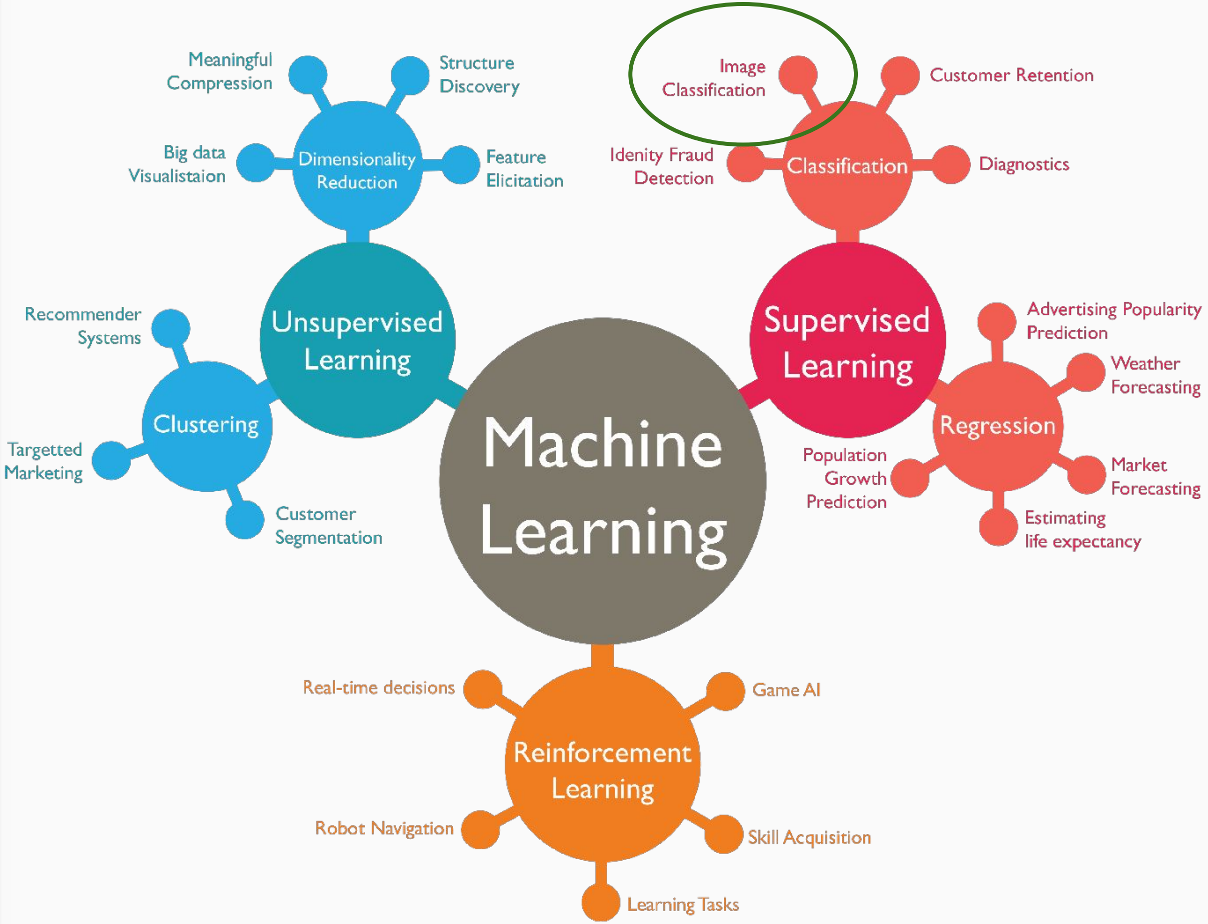
Ablative Analysis

- Helps to understand how each component in the system helps to achieve final better accuracy
- Helps to identify the less contributing component so they can be removed
- List down what is the accuracy **IMPROVEMENT** after each level starting from the basic model
- Remove one component at a time and see how accuracy drops

Types of ML

Classification

- **Logistic Regression**
- **Decision Tree**
- AdaBoost
- Naive Bayes Classification
- **SVM (Support Vector Machine)**



Ref: https://towardsdatascience.com/machine-learning-types-2-c1291d4f04b1

Soliton
Vision for a Better World

Types of Learning

Supervised: Learning by Labelled Ex

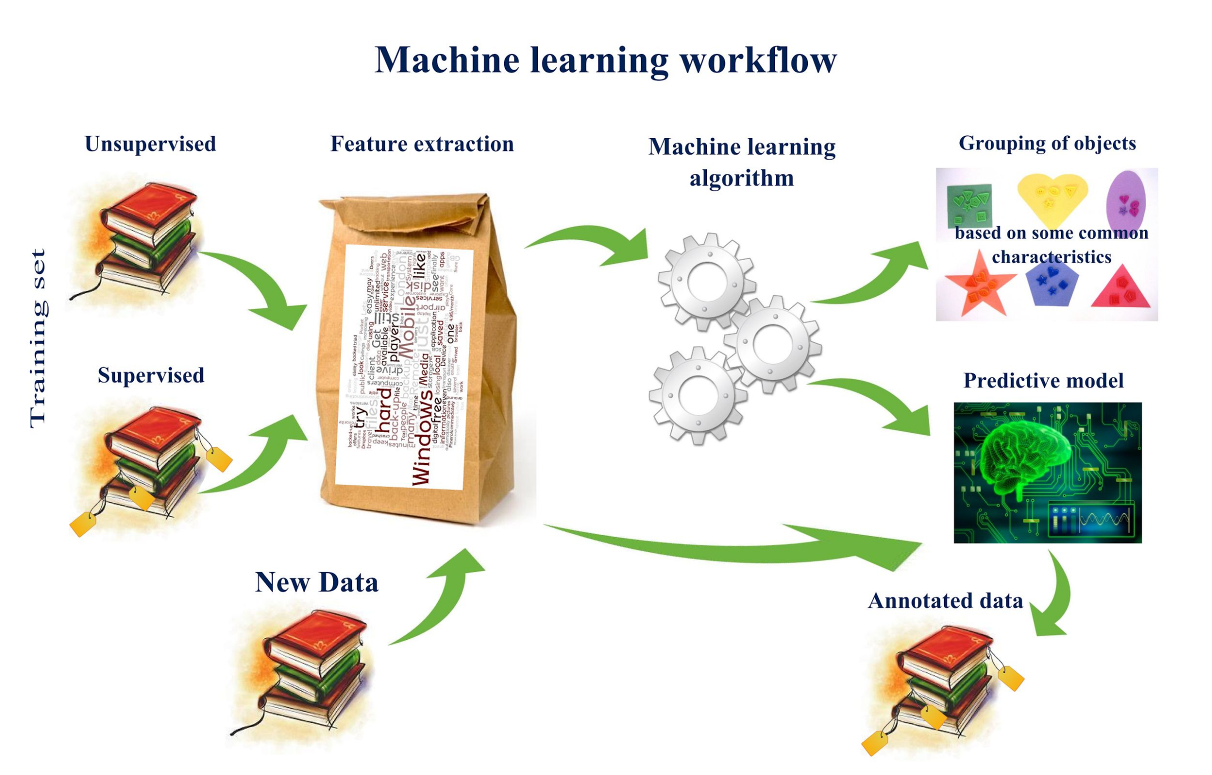
- Eg. Face Recognition
- Amazingly effective if you have labelled examples

Unsupervised: Discovering Patterns

- Eg. Google News - Data Clustering
- Useful if you lack labelled data

Reinforcement: Feedback right/wrong

- Eg. Playing chess by winning or losing
- Works well in some domains, becoming more important



Ref: http://www.datascienceassn.org/content/machine-learning-workflow

Soliton
Vision for a Better World

MOOC for ML

cs229 is good place to start

Do a lot of assignments

Work on pet projects

Contribute to ML Open source libraries

Soliton
Vision for a Better World

Courses

- ML: cs229 by Andrew Y. Ng
- RL: David Silver

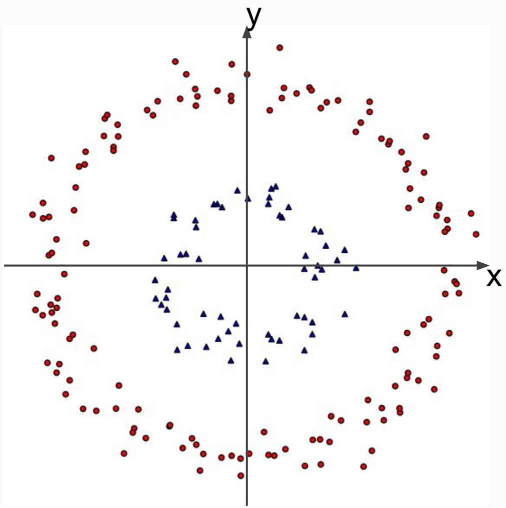
Blogs & Github:

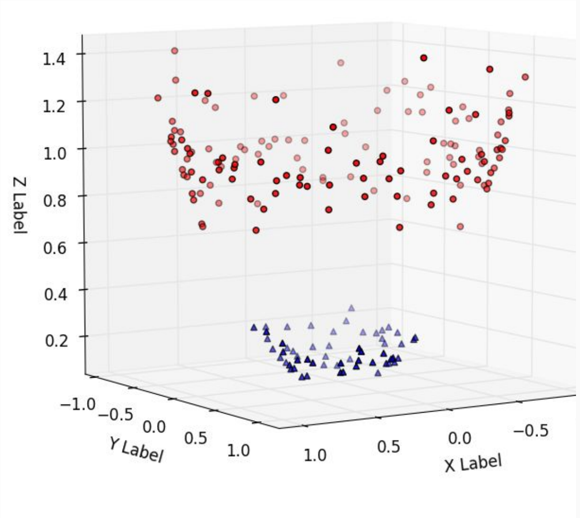
- Scikit-Learn examples
- ML Playground

How to Solve this?

We MAY be able to solve this by introducing one more feature which MAY separate them linearly.

Do you see any pattern?





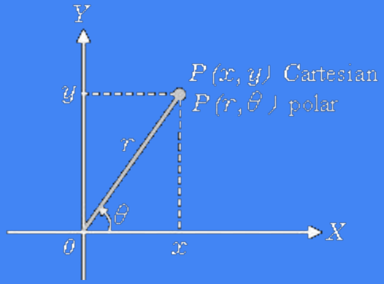
Ref: http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html

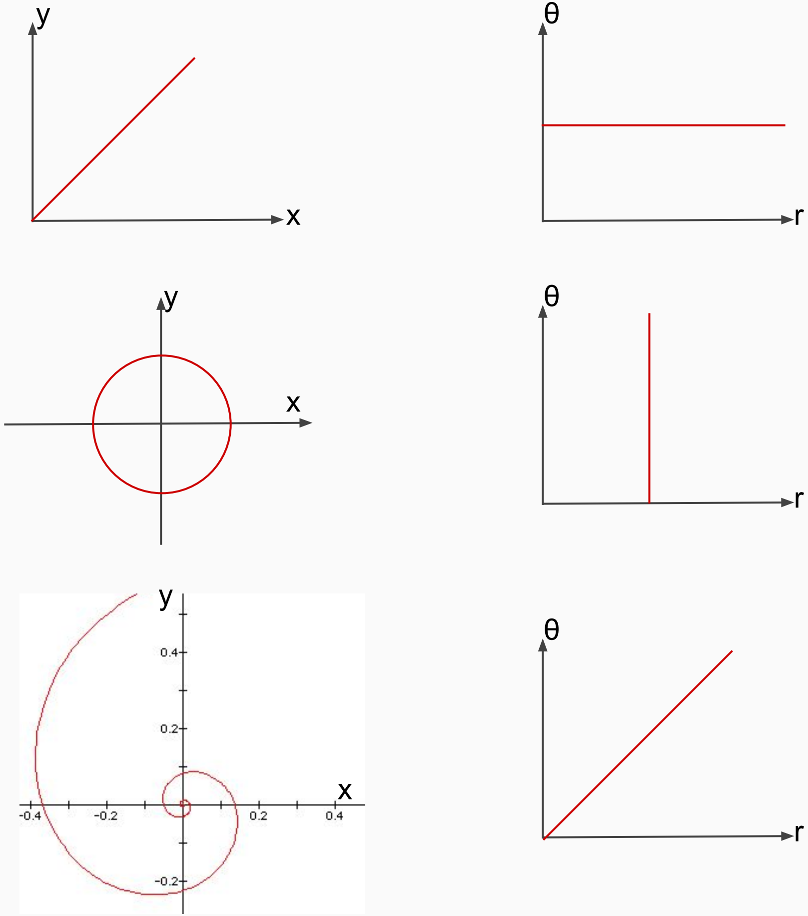
Soliton
Vision for a Better World

Cartesian -> Polar

r = √ (x2 + y2)

θ = tan-1 (y / x)



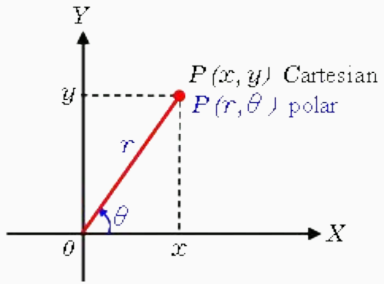


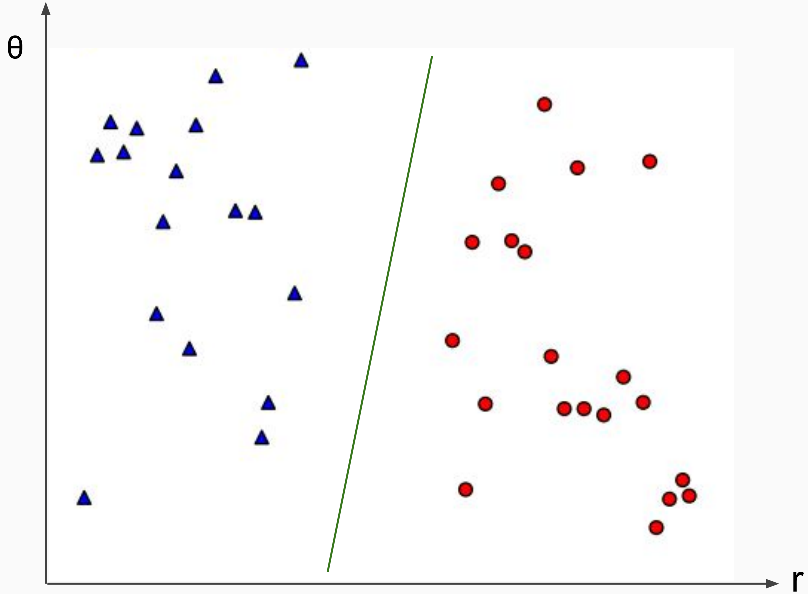
Soliton
Vision for a Better World

Cartesian -> Polar

$r = \sqrt{(x2 + y2)}$

$\theta = \tan^{-1}(y / x)$





Soliton
Vision for a Better World

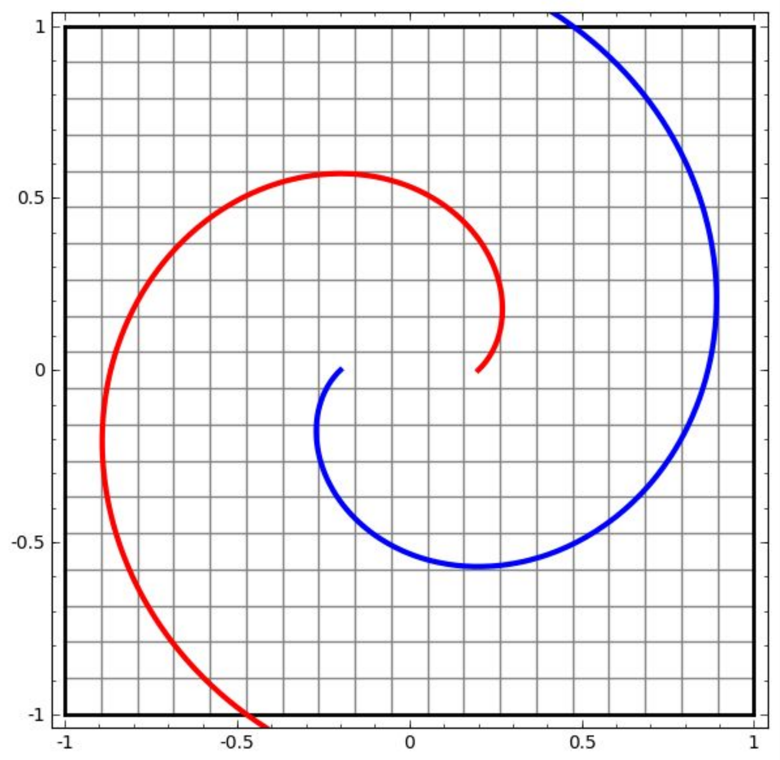
What is Next?

Let the Algorithm Learn these

- Functions
- Features

ON ITS OWN…!

Deep Learning



Ref: http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/

Soliton
Vision for a Better World

Session III (DL)

What is the limitation of simple Image Processing and why we need intelligent systems?

Session III

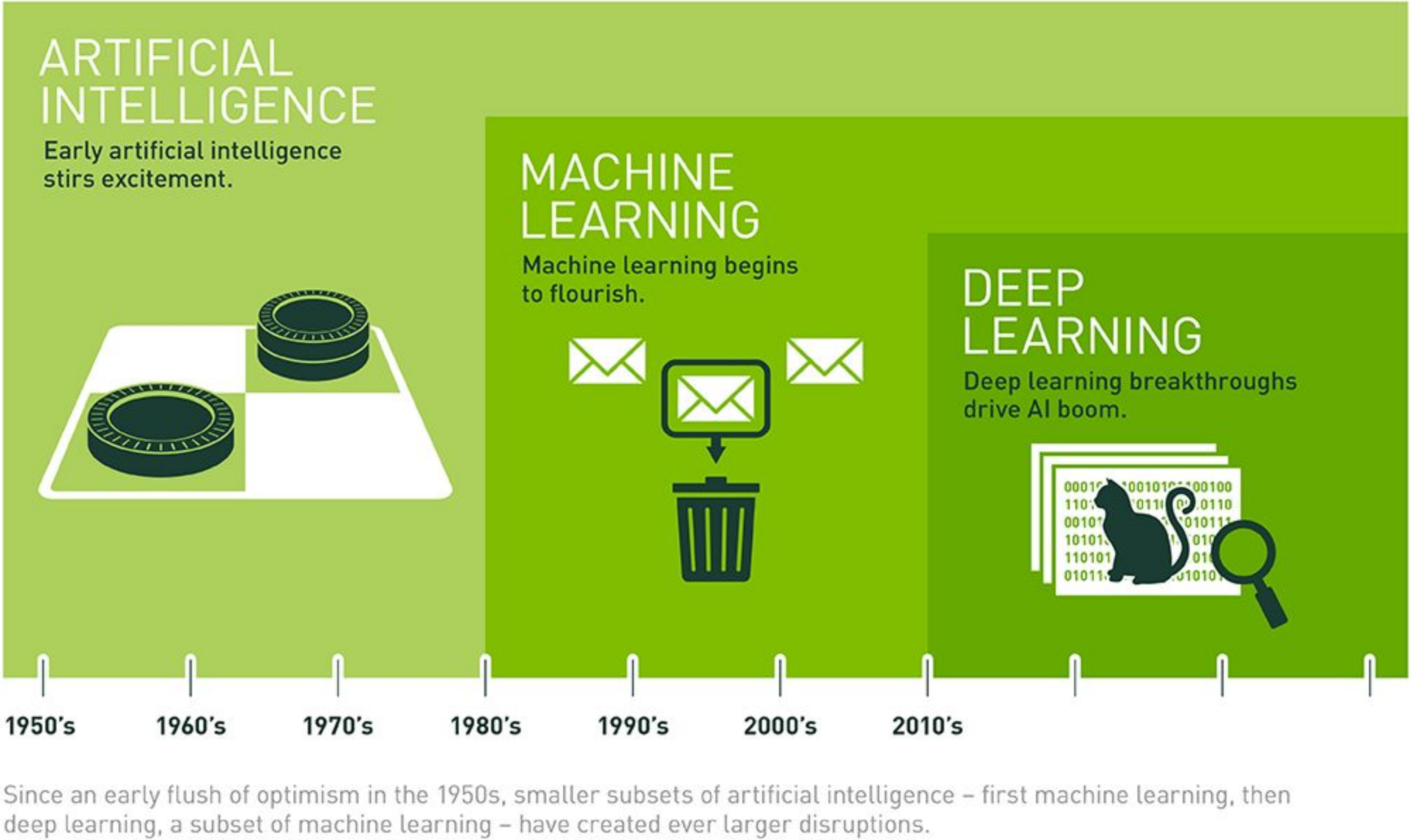
Deep Learning

Agenda

- ➔ What is AI
- ➔ Human Brain
- ➔ Limitation of ML
- ➔ Perceptron / MLP
- ➔ Backpropagation
- ➔ Deep learning
- ➔ CNN
- ➔ Architectures
- ➔ Applications

http://lodgiq.com/improved-pricing-with-machine-learning/

Soliton
Vision for a Better World



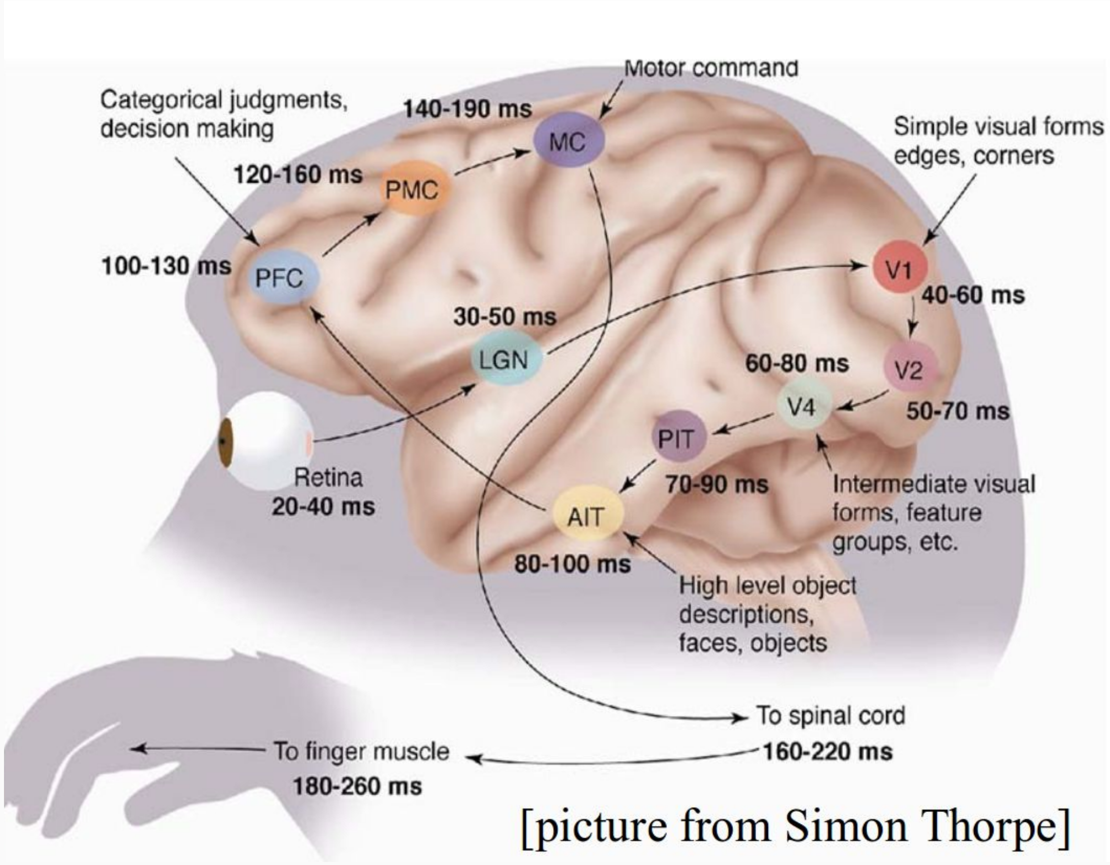
https://www.datasciencecentral.com/profiles/blogs/artificial-intelligence-vs-machine-learning-vs-deep-learning

Human Brain..!

100 Billion Neurons

Each Neuron has 7000 connections

All these are processed under 20 Watts



[picture from Simon Thorpe]

Soliton
Vision for a Better World

Few Questions

When you take a video in a mobile motioning your hand, the video has the same effect. But when you are watching, the world does not change. How?

How do you learn? How do you know the action that you performed is right? And repeat the action.

Why do you hear the Sound differently?

Soliton
Vision for a Better World

Case Study - 1

David lost his sight at 3. With the help of a surgery, he gets his sight back at the age of 35. But still he is not able to see things properly. Why?

- The occipital, the back part of the brain which is responsible for the Vision was used for other activities
- This part assignment happens when we are child
- If we don’t use a particular part, it will be assigned other tasks

Case Study - 2

John met with an accident. When he met his parents, he could not recognize them and treated them like strangers. When he spoke to them over phone, he could recognize and talk to them normally. What happened?

Three parts involved

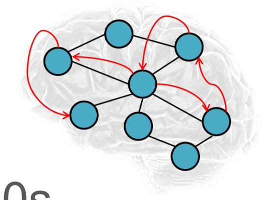
- Vision recognition
- Auditory
- Emotion

In the accident, the Vision recognition part of the brain lost its memory

But the auditory system still recognizes they are his parents

But for the emotional part of the brain to take decisions, Vision system has more weight and as it sees the conflict, it believes the Vision system

Deep Learning



- Modern reincarnation of Artificial Neural Networks from 1980s and 90s
- Idea: Most of Human Intelligence may be due to one Learning Algorithm
- For all ill-defined problems: Build Learning Algorithms that mimic Brain
- Deep Learning attempts to learn multiple levels of representation of increasing complexity or abstraction
- Higher (deeper) levels of brain (ANN) forms higher levels of abstractions

Limitations of Traditional ML

- Need to hand engineer features which can take a lot of time
- The limitations in its ability to represent complex features (Requires a lot of diligence and intelligence)
- People have spent years to collect data
- Models developed for one problem cannot be easily be utilised for a similar problem

*Img Ref - https://www.cebma.org/faq/limitations-evidence-based-practice/

Soliton
Vision for a Better World

Classification

- How do you classify Pencils and Erasers?
- Any distinguishing features?

Length

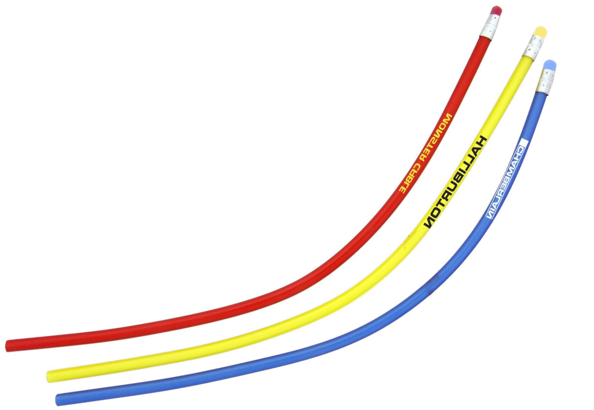
Rigidity

Soliton
Vision for a Better World

Classification

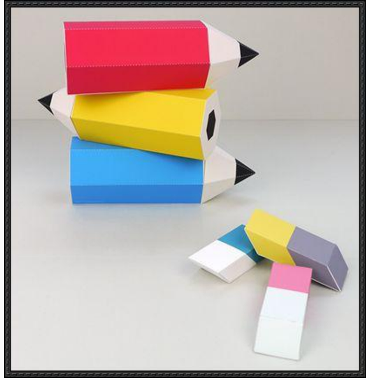
How to classify these?

- ➔ Pencils are little flexible
- ➔ Erasers grew long



Classification

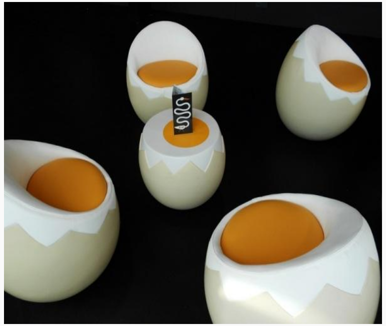
How to classify these? :-(



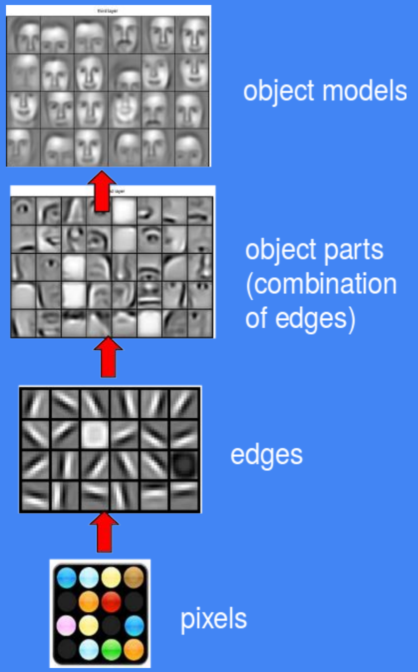
Soliton
Vision for a Better World

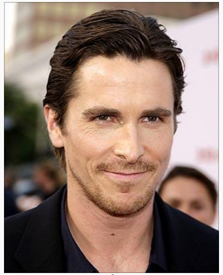
Features

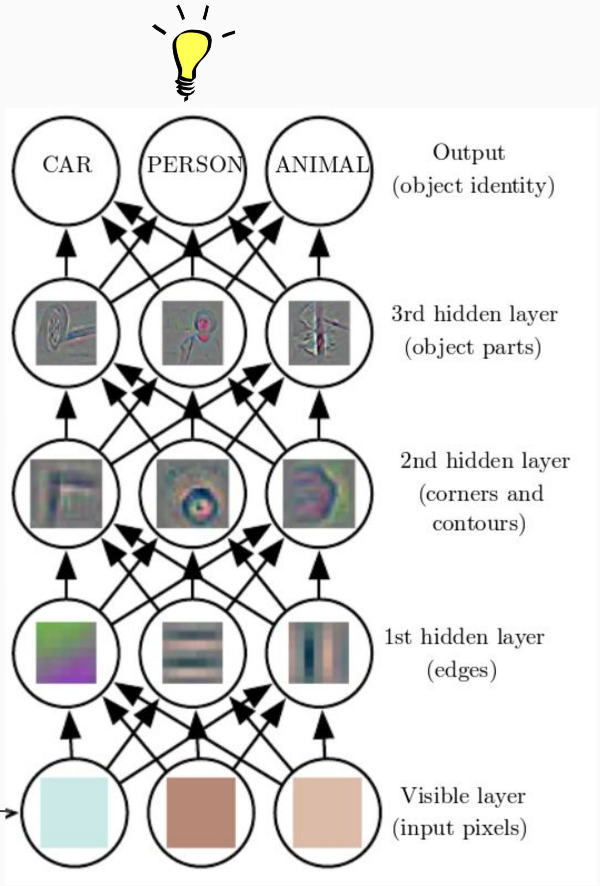
How does a chair look like?



What NN Learns?







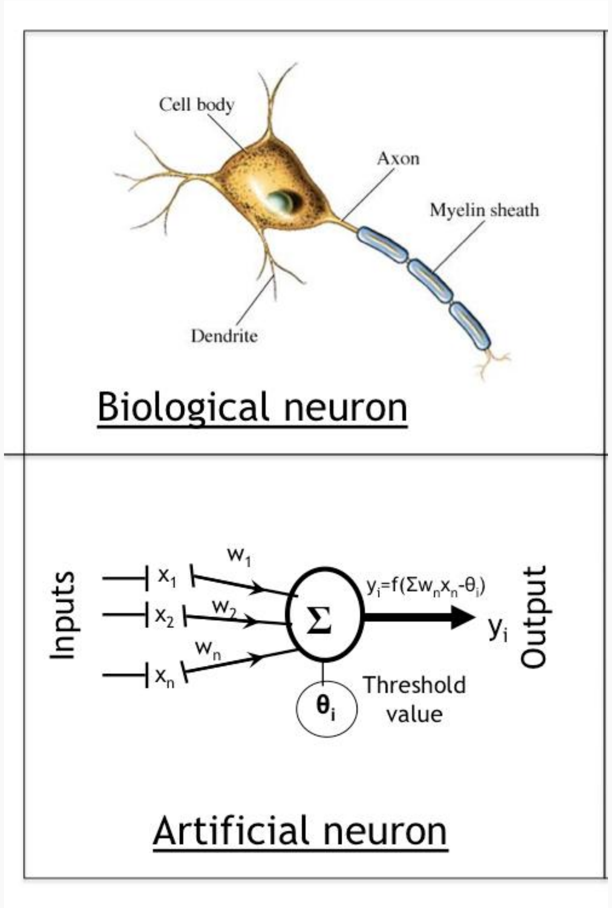
*Ref - https://www.kdnuggets.com/2016/04/deep-learning-book-finished.html
*Ref - https://www.mizuho-ir.co.jp/publication/column/2013/1119.html
*Ref - http://bale.forosactivos.net/t7p45-con-barba-el-hombre-como-el-oso

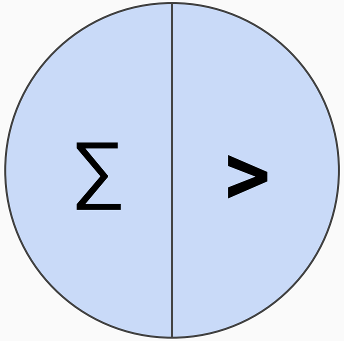
Soliton
Vision for a Better World

Neuron vs Perceptron

Incoming singals from Synapses are summed up at the Soma (A biological inner product)

On crossing a threshold, the neuron "fires" generating an action potential to pass on to the next neuron





Soliton
Vision for a Better World

Perceptron training

Include bias term as the third weight(w3) with its input always set to 1

Step 1: Initialization: $w_1 = 0$, $w_2 = 0$, bias = 0

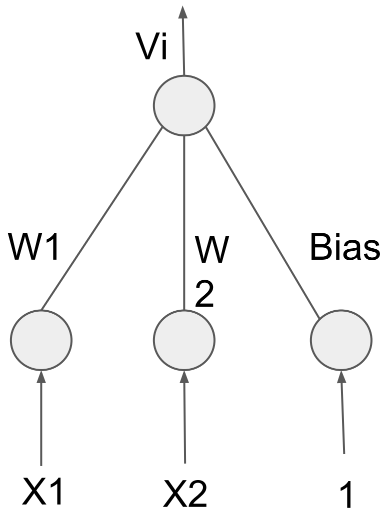
For each of the training sample do steps 2 -4

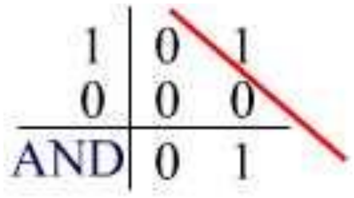
Step 2: Compute output by weighted linear combination of inputs
(Vi = w1 *x1 + w2 * x2 + 1 * bias)
Predicted output = Vi > Threshold

Step 3: Find the error (Error = Anticipated output - Predicted output)

Step 4: Update each weight based on the following
ΔWi = Error * λ * Xi
Wi = Wi + ΔWi
Where λ is the learning rate and its range is 0 <= λ < 1

Step 5: Repeat the procedure until no error results



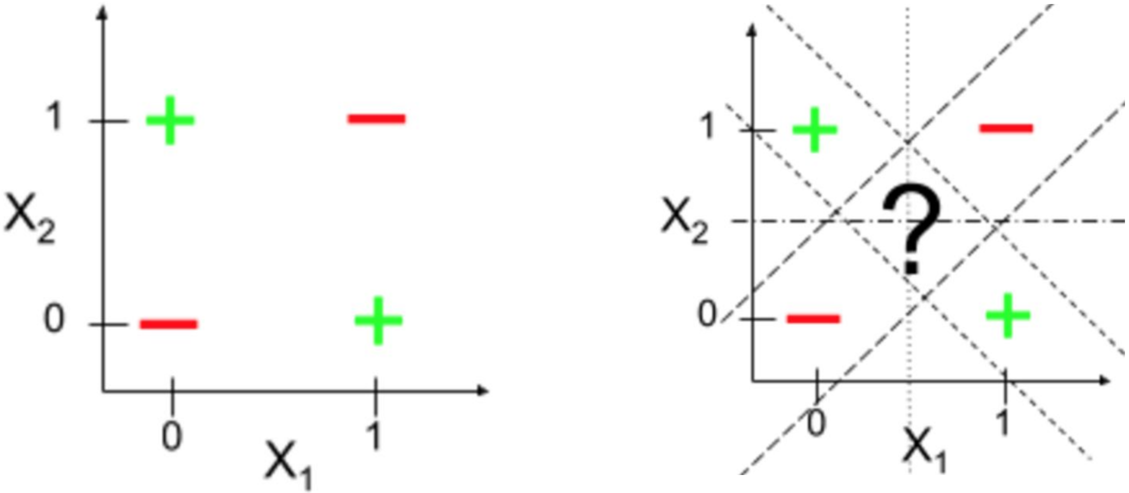


* Ref - http://wwwold.ece.utep.edu/research/webfuzzy/docs/kk-thesis/kk-thesis-html/node12.html

Limitations of Perceptron

- A simple perceptron cannot learn a classifier for a XOR gate
- How to draw a decision boundary in case of a XOR gate?

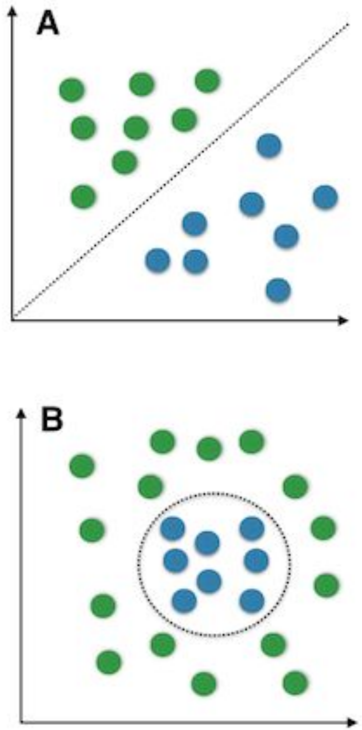
x_1	x_2	x_1 **XOR** x_2
0	0	0
0	1	1
1	0	1
1	1	0



*Ref - https://medium.com/@claude.coulombe/the-revenge-of-perceptron-learning-xor-with-tensorflow-eb52cbdf6c60

Problem - Linear Separability

- Can this kind of perceptron provide solutions to all kinds of data patterns we might encounter in practice? [Let's find out](#)
- This is because we don't have non-linear elements in our network. Hence, this kind of network can only learn linear functions of inputs.
- How can we improve the network to learn non-linear functions?
- 1. Include Nonlinearities in the network
- 2. Key observation - Cannot directly classify data. Convert the data to a new feature space to classify



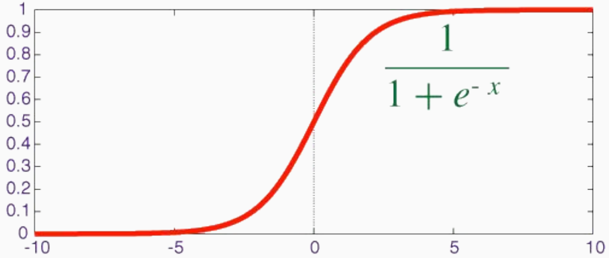
*Img ref - https://stackoverflow.com/questions/44606126/linear-svm-vs-nonlinear-svm-high-dimensional-data

Sigmoid activation fn

We can include non-linear functions in our to improve the representational power of the network.

Can we represent this kind of activation as continuous and smooth function

- The sigmoid function is shown as follows



- Here x is weighted combination of the neuron inputs.
- The neuron fires when x >> 0 and does not fire when x << 0. The neuron lies in a transition state when when x ≈ 0.
- The function is smooth and differentiable

$$\frac{1}{1+e^{-t}} = \sum_0^{\infty} (-1)^k e^{-kt}$$

Soliton
Vision for a Better World

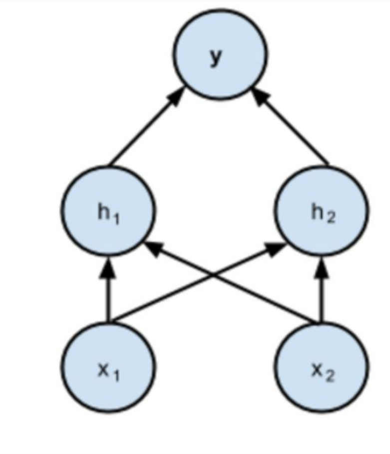
*Img Ref - http://mathworld.wolfram.com/HeavisideStepFunction.html

MLP

Convert the inputs into a new feature space where the data points are linearly separable

This necessitates the need at least two neurons with **Non-linearities**. This kind of architecture is called **Multilayer Perceptron (MLP)**

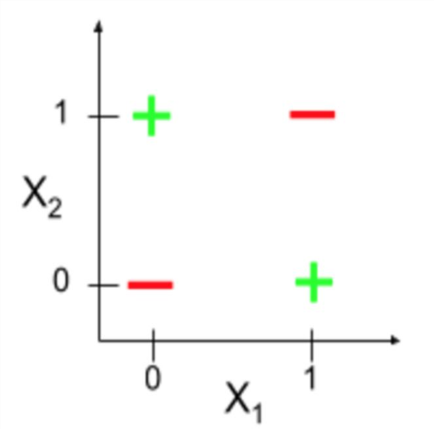
The first layer is called input layer, the layer at the last is called output layer. The layer / layers in between are called **Hidden layers**

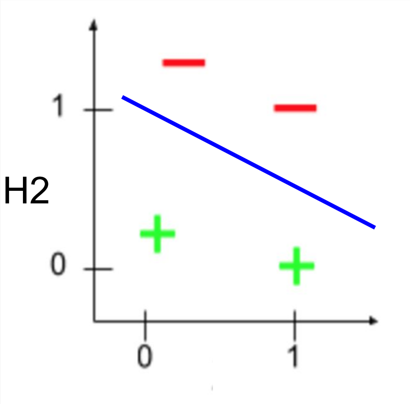


Output layer

Hidden layer

Input layer





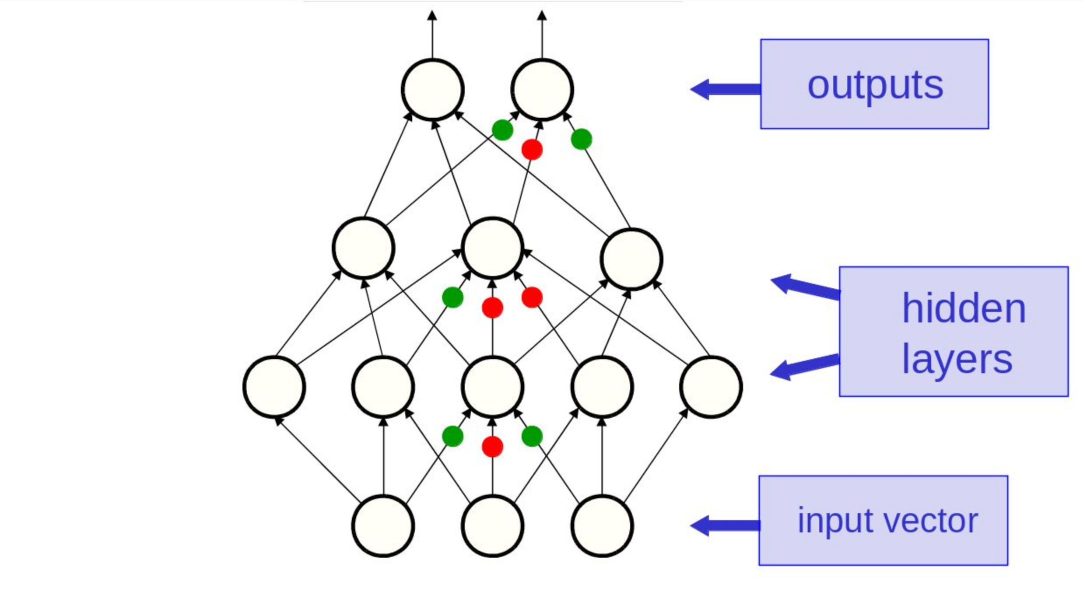
* Img Ref - https://medium.com/@claude.coulombe/the-revenge-of-perceptron-learning-xor-with-tensorflow-eb52cbdf6c60

Multi Layer Perceptron (MLP)

A simple Multilayer network will consist of:

- Input layer
- Output layer
- One or more hidden layers

It is not necessary that each hidden layer should contain same number of neurons. Each hidden layer usually contains a non-linear activation function



*Ref - http://slideplayer.com/slide/1545087/

Soliton
Vision for a Better World

What about training?

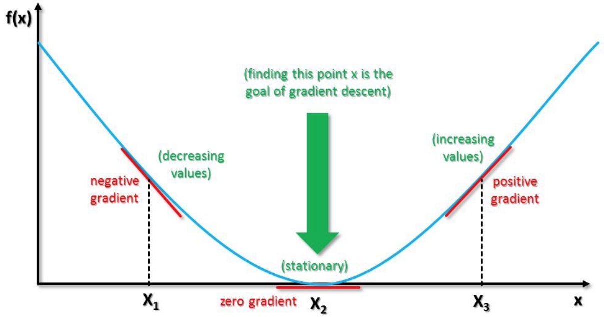
- Single layer perceptron - Direct interaction between input and output, hence update weights directly based on output and inputs.

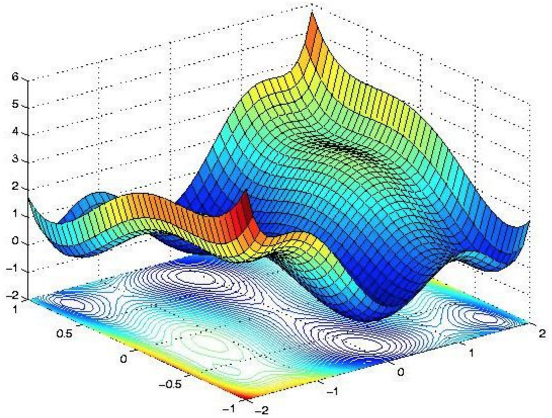
- Training is harder in MLP, since there are multiple layers of weights.

- Measure error committed by network → Objective function (Squared difference b/w the Anticipated o/p and predicted o/p - MSE)

- Find the minima of the objective function through gradient descent

- But how do we update weights of the network based on direction to move in gradient descent?



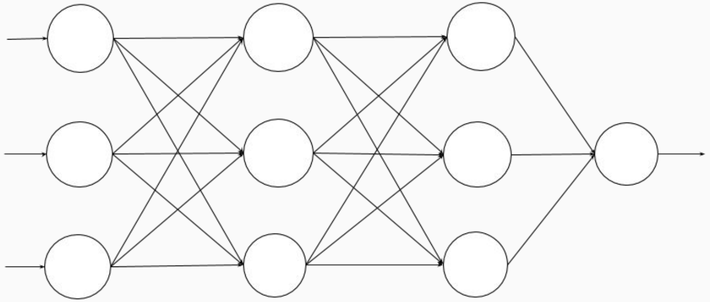


*Img reference - http://www.big-data.tips/gradient-descent
*Img reference - http://www.mit.edu/~amini/projects/

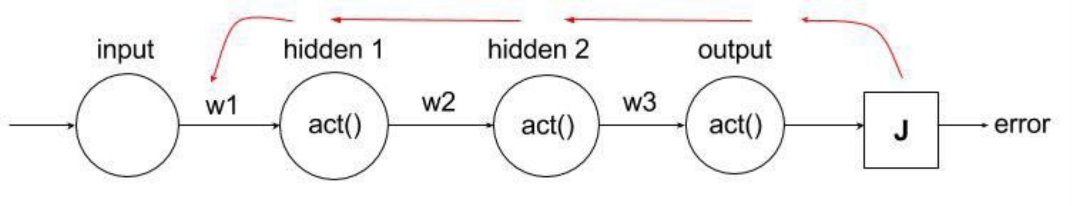
Backpropagation

Backpropagation provides a way to compute the gradient of the error function with respect to each of the weights in the network.

This provides a way to update the weights of the network based on the error function.



$$W := W - \alpha \frac{\partial J}{\partial W}$$



$$\frac{\partial error}{\partial w1} = \frac{\partial error}{\partial output} * \frac{\partial output}{\partial hidden2} * \frac{\partial hidden2}{\partial hidden1} * \frac{\partial hidden1}{\partial w1}$$

*Img Ref - http://mathworld.wolfram.com/HeavisideStepFunction.html

Backpropagation

Backpropagation provides a way to compute the gradient of the error function with respect to each of the weights in the network.

This provides a way to update the weights of the network based on the error function.

$$W \coloneqq W - \alpha \frac{\partial J}{\partial W}$$

$$error = J = \frac{1}{2}(\vec{p} - \vec{a})^2$$

$$\frac{d}{d\vec{p}} error = \frac{d}{d\vec{p}} J = 2 * \frac{1}{2}(\vec{p} - \vec{a})^{2-1} * 1 = (\vec{p} - \vec{a})$$

$$Sigmoid = S(\alpha) = \frac{1}{1 + e^{-\alpha}}$$

$$\frac{d}{d\alpha} S = S(1 - S)$$

$$\frac{\partial J}{\partial W^1_{ij}} = (p_j - \vec{a}) * p_j(1 - p_j) * \frac{\partial p_i}{\partial W^1_{ij}}$$

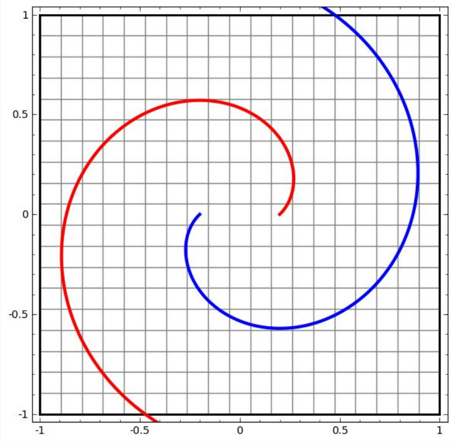
*Img Ref - http://mathworld.wolfram.com/HeavisideStepFunction.html

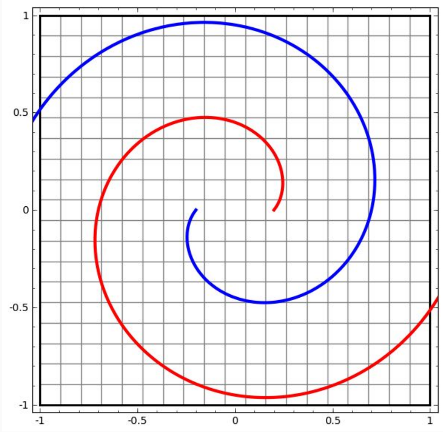
Soliton
Vision for a Better World

What does deep learning learn?

- MLPs without bias term model linear transformation
- MLPs with bias term model affine transformation
- The activation functions introduces further non-linearities
- Deep learning learns a transformation of a feature space that becomes linearly separable in a different topological space
- Link - [convnetjs]

Deep Learning





*Ref - http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/

Soliton
Vision for a Better World

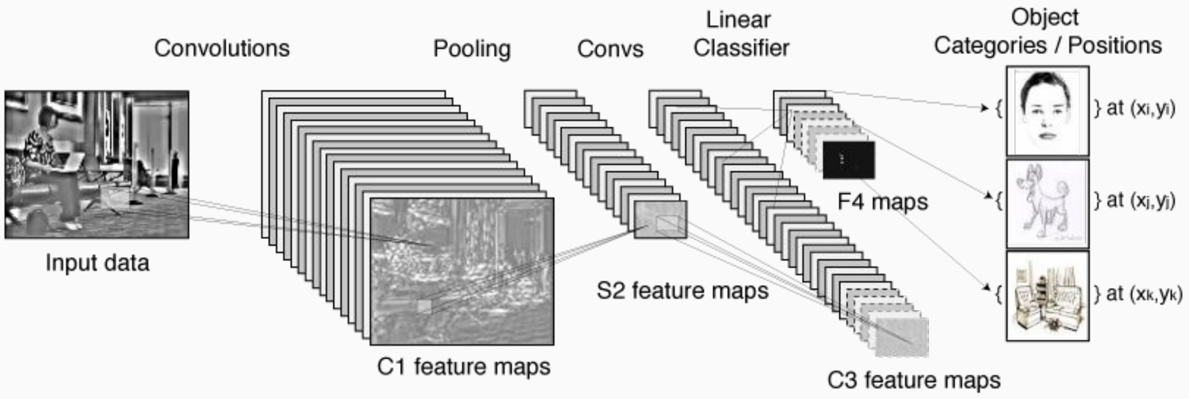
MLP - Limitations

- Can this kind of MLP learn any functions?
- Neural network with one hidden layer is a universal approximator
- Then why are many hidden layers required?
- It is practically difficult to learn the exact values of the parameters in such networks. Hence multiple layers make it practically possible to exploit the representational power of a neural network.
- Very expensive training process (Too many parameters to learn)
- Not scalable to a larger architecture. The number of neurons increases rapidly with the number of neurons in the network.
- Does not converge

CNN

- Multi Layered
  - Convolution
  - Pooling
  - Dropout
  - Dense (Fully Connected)
- Neurons are arranged in a 3D layer, unlike a MLP, where it is arranged in a 2D layer

- Each neuron views only a specific portion of the input
- Shared weights
- Encodes properties that are more desirable for images.
- Calculates the features from the input by repeated applying convolution operation.



*Img Ref - https://www.frontiersin.org/articles/10.3389/fnins.2012.00032/full

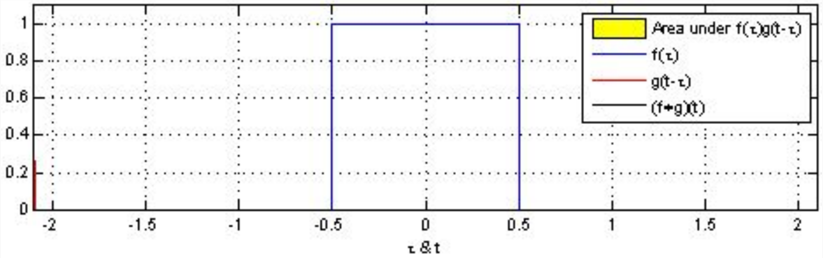
Soliton
Vision for a Better World

Convolution

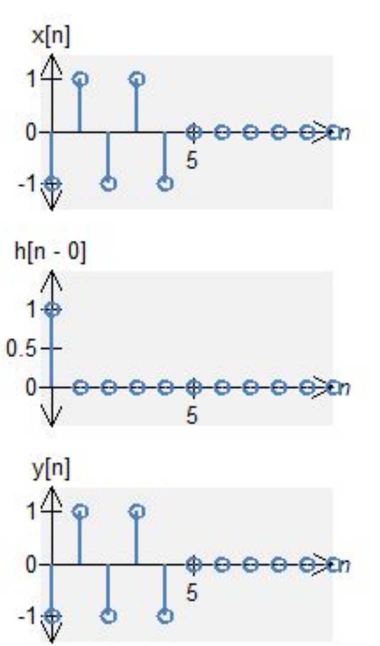
Convolution is a mathematical operation on two functions to produce a third function giving the summation of the pointwise multiplication of the two functions as one of the functions is translated throughout

$$
\begin{aligned}
(f * g)[n] &= \sum_{m=-\infty}^{\infty} f[m]g[n-m] \\
&= \sum_{m=-\infty}^{\infty} f[n-m]g[m].
\end{aligned}
$$

Continuous case



Discrete case

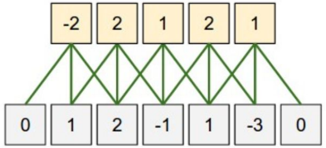


*Ref - https://en.wikipedia.org/wiki/Convolution
*Ref - http://www.ece.utah.edu/~ece3500/notes/class06.html

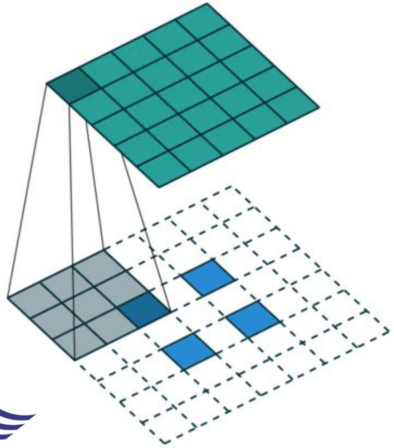
Soliton
Vision for a Better World

1D vs 2D vs 3D Convolution

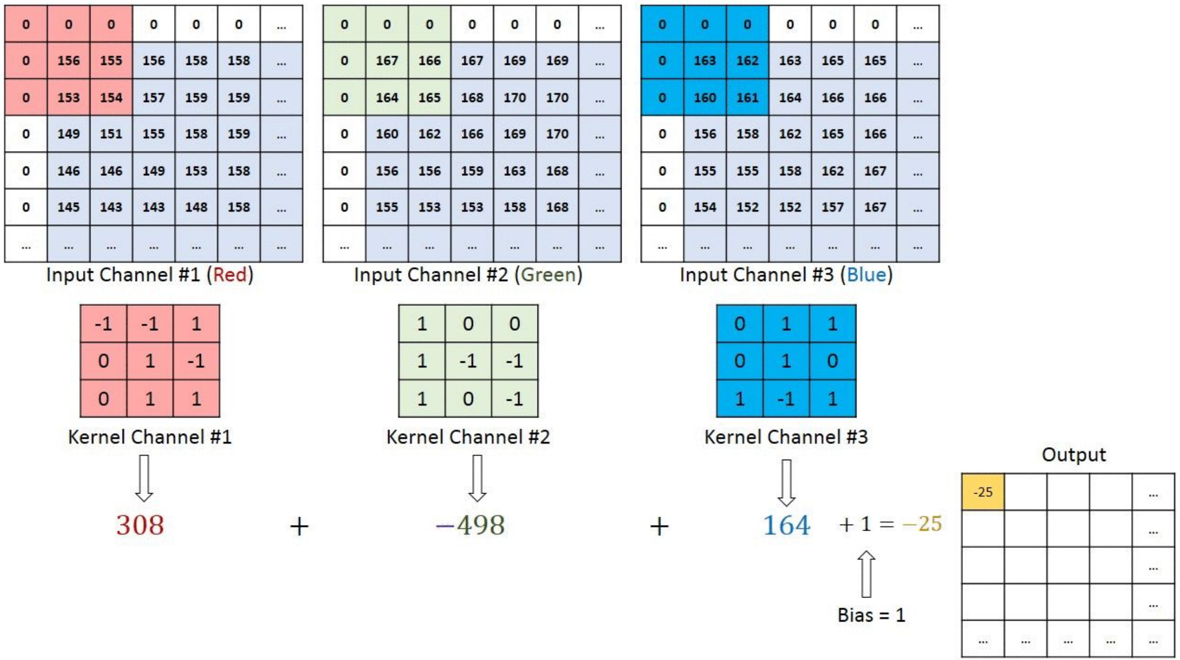
1D



2D



3D

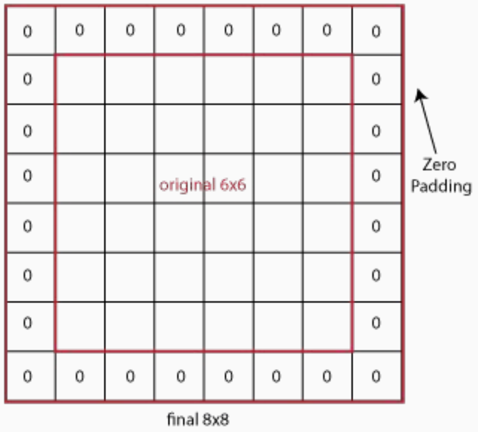


Convolution

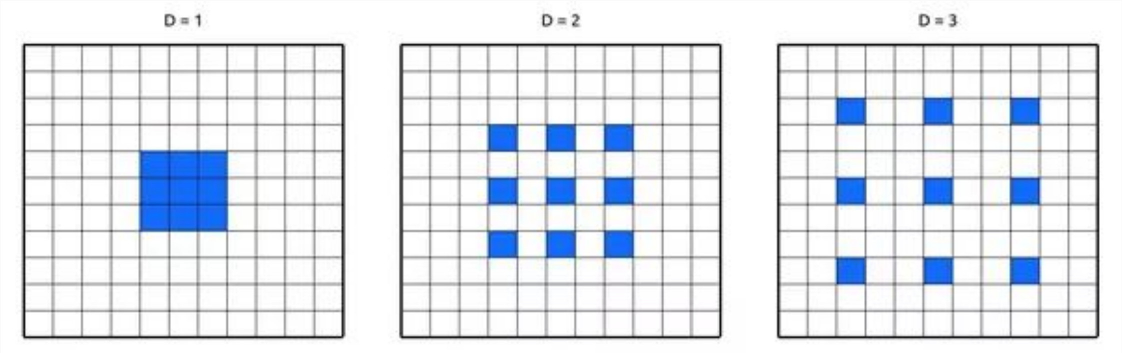
ZERO PADDING - zeros can be added to the feature map to increase the size of the feature map. The idea is to keep the size of the feature map the same throughout

STRIDES - Number shifts to next window in the feature map while doing convolution

Zero Padding



Convolution on a feature map with three different strides (Demo)



*Img Ref - https://www.analyticsvidhya.com/blog/2016/04/deep-learning-computer-vision-introduction-convolution-neural-networks/
*Img Ref - https://stats.stackexchange.com/questions/234692/how-to-work-multiple-filter-region-sizes-2-3-and-4-in-cnn

Soliton
Vision for a Better World

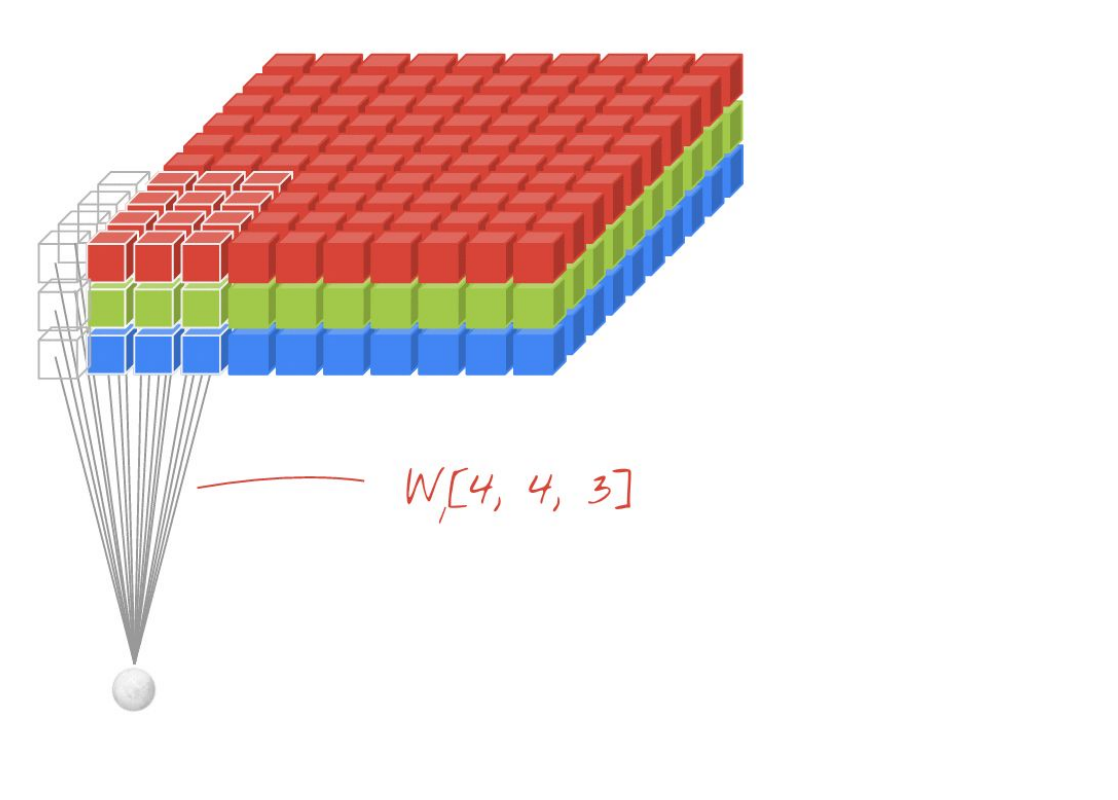
Quiz..!

Convoluted result size

- **Feat 9x9**
  - Conv 3x3
  - Stride 2
- **Feat 28x28**
  - Conv 5x5
  - Stride 1
  - Padding 1
- **Feat 224x224**
  - Conv 16x16
  - Stride 3

Soliton
Vision for a Better World

Answers: 4x4, 25x25, 69x69



Convolution Layer

Neurons:

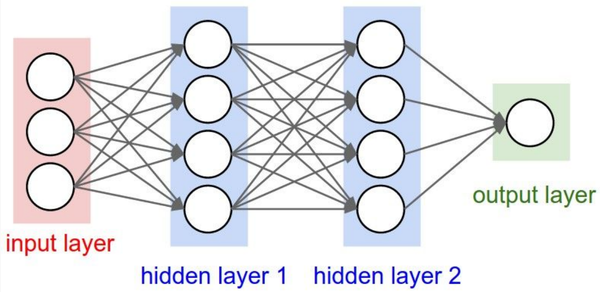
- Input pixels
- Hidden layer

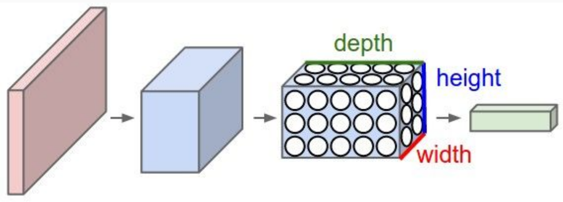
Weights:

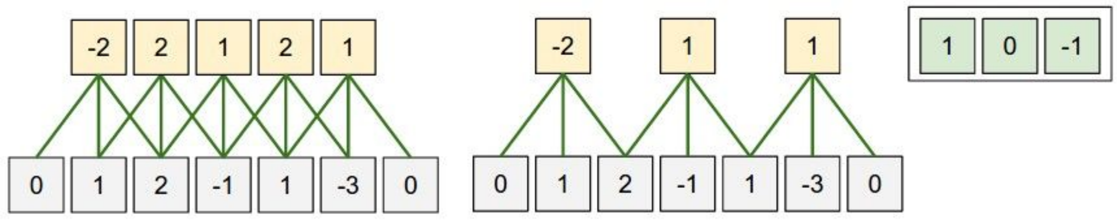
- Kernel weights for the convolution
- Shared among neurons

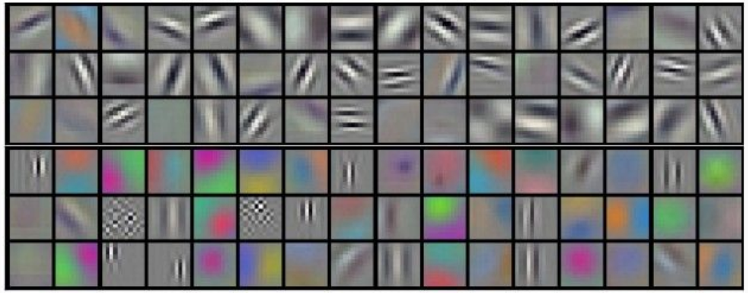
Features:

- A kernel decides what feature you are learning





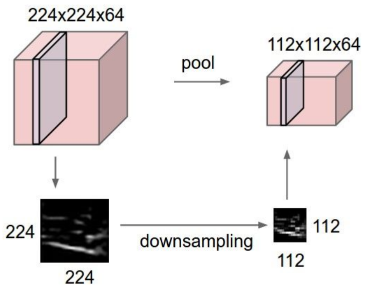


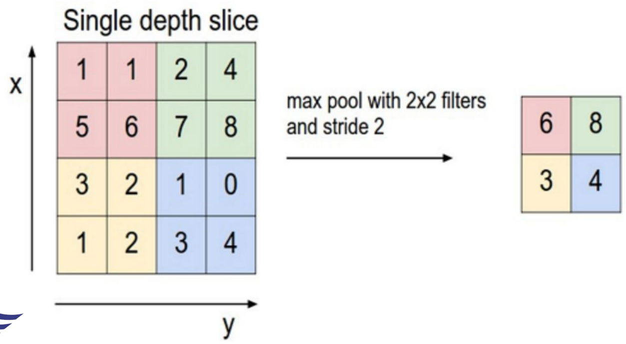


http://cs231n.github.io/convolutional-networks/

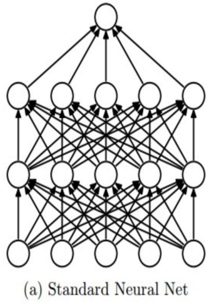
Other Layers

Pooling layer - Down samples the size of feature map. Promotes translation invariance

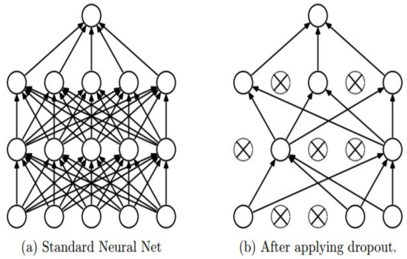




Fully connected - All neurons in a hidden layer is connected with all neurons in the next layer



Dropout - Randomly removes a few layers in the connection. Reduces overfitting



*Ref- http://cs231n.stanford.edu/

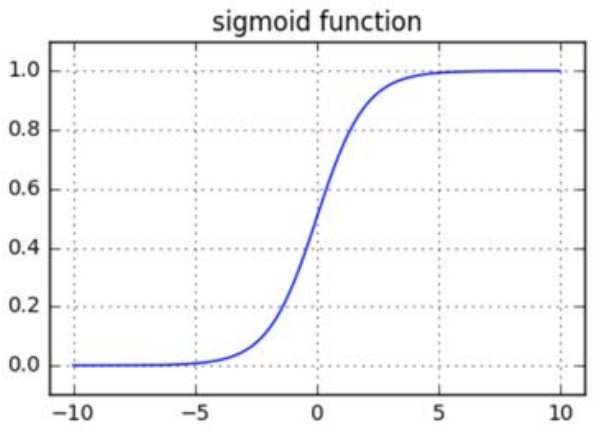
Problem with sigmoid

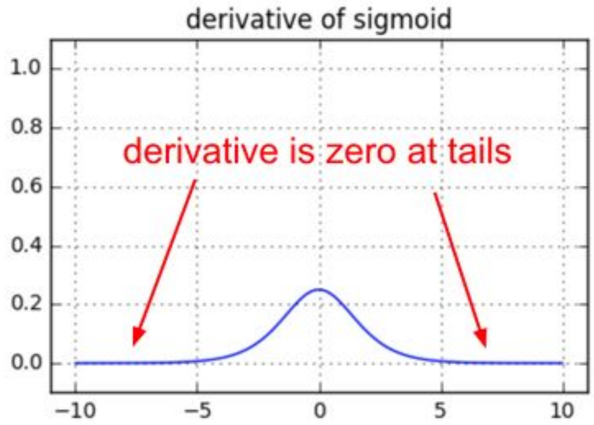
- Vanishing gradient: It’s output saturates at both ends, hence produces “vanishing gradient problem” i.e., the gradient becomes zero at these saturation points and the network cannot learn weights based on backpropagation

- Scales down the gradients to 0.25
    - z*(1-z) ⇒ Max = 0.25 at z=0.5

- It involves expensive operations, hence slows down the training process

```
z = 1/(1 + np.exp(-np.dot(W, x))) # forward pass
dW = (z*(1-z) * x  # backward pass
```



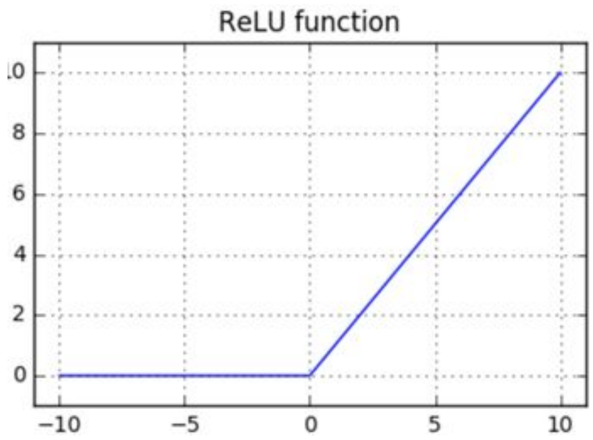


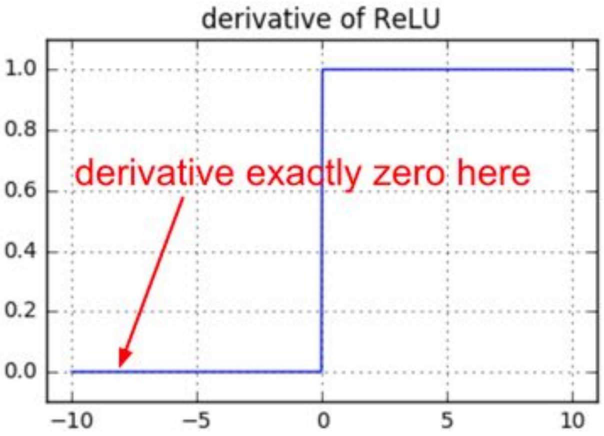
Soliton
Vision for a Better World

ReLU (Rectified Linear Unit)

- Sparse activations - A characteristic property of biological neurons
- Does not saturate at extremes, hence allow gradients to propagate through larger networks
- ReLUs are computationally inexpensive.
- Dead ReLU’s: Not a big Concern.
- Leaky ReLU: Instead of Zero, a small negative slope

```
z = np.maximum(0, np.dot(W, x)) # forward pass
dW = np.outer(z > 0, x)   # backward pass
```





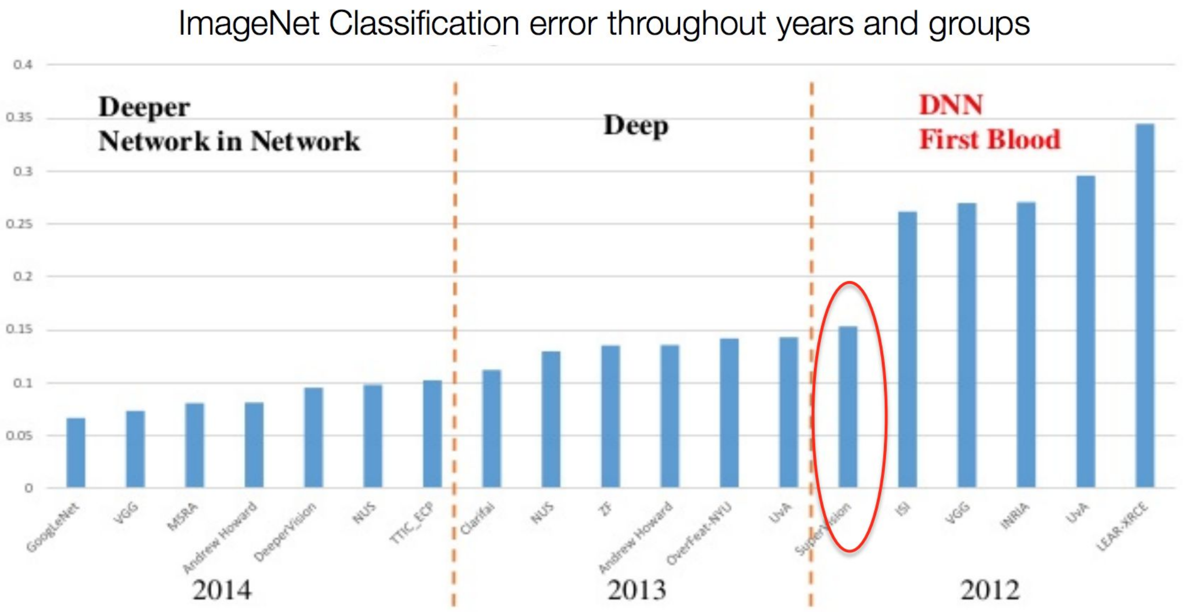
Soliton
Vision for a Better World

*Img Ref - https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/relu_layer.html

CNN: What Changed?

- ReLU
- Shared Weights
- Specialized Layers: CP
- GPGPU
- Availability of OSS Libraries/Datasets

ILSVRC

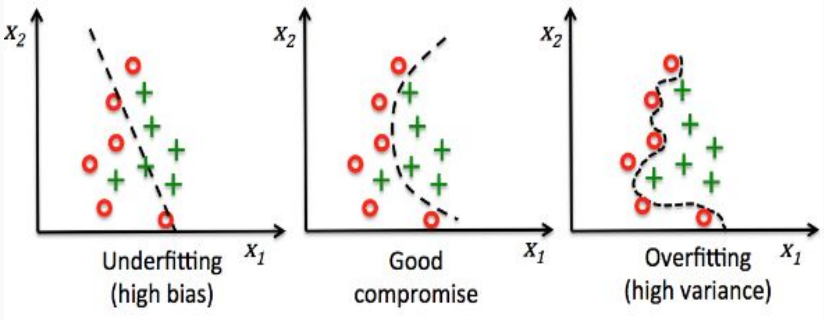


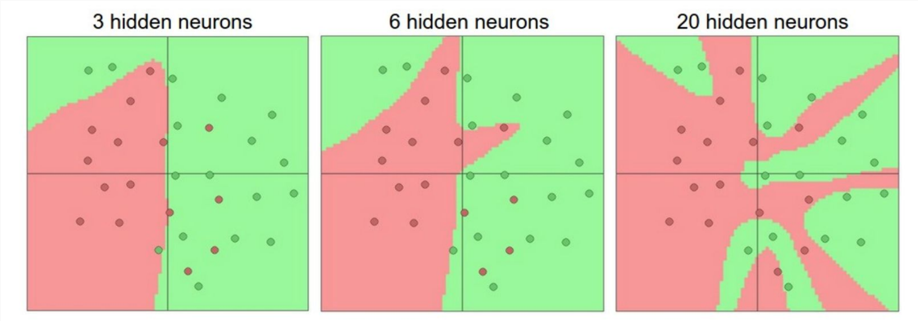
Li Fei-Fei: ImageNet Large Scale Visual Recognition Challenge, 2014 http://image-net.org/

Overfitting

Noise in the data is also fit by the model leading to overfitting

Overfitting increases with number of neurons. But it is not a good practice to decrease the number of neurons! Use Regularisation instead :)

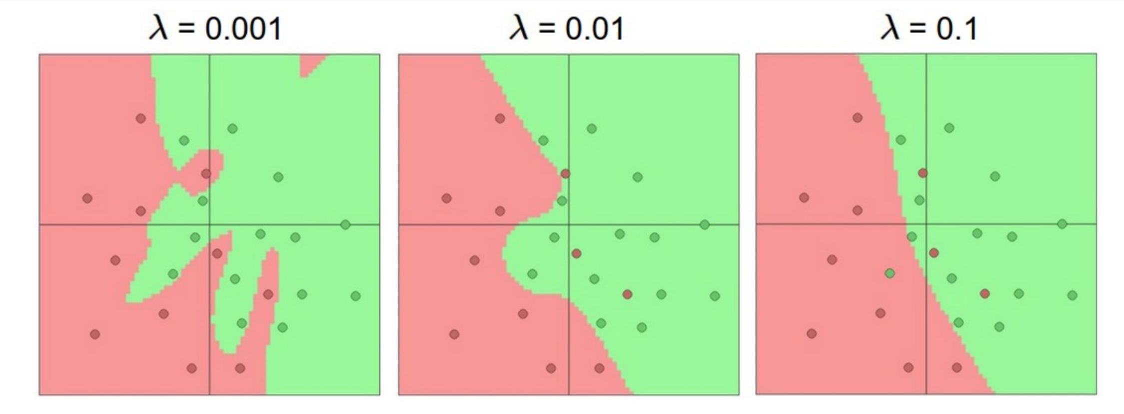




http://cs231n.github.io/convolutional-networks/

Regularization

Regularisation helps to generalise to the given data while maintaining the representational power of the network



Types of regularisation:-

- L1 regularisation - Prefers sparse distribution
- L2 regularisation - heavily penalizes peaky weight vectors and prefers more diffuse weight vectors.

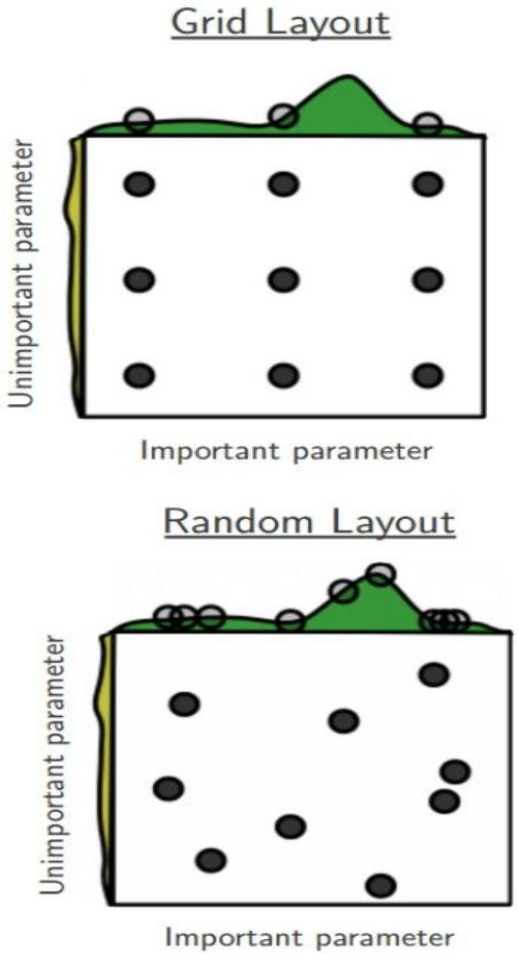
Cost function = Loss (say, binary cross entropy) + Regularization term

http://cs231n.github.io/convolutional-networks/

Soliton
Vision for a Better World

Hyperparameter Strategies

- Hyperparameters: learning rate, batch size, size of convolution, stride etc..
- First do a coarse search with small epochs and then fine search with larger epochs.
- **Don’t do grid search, prefer random search.**



*Ref- http://cs231n.stanford.edu/

Cross Entropy based loss

- Cross Entropy formula given two distributions over discrete variable x, where q(x) is the estimate for true distribution p(x) is given by

  H(p,q) = −∑p(x)log(q(x))

- For a neural network,it is denoted by L=−**y**·log(**ŷ**) where y is the ground truth label and **ŷ** is the predicted label

- Cross entropy loss rewards / penalises only the prediction for the ground truth class. All the other class predictions has no effect.

- Works better in practice than MSE for classification problems

$$\hat{\mathbf{y}} \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \mathbf{y}$$

$$\begin{bmatrix} 0.1 \\ 0.5 \\ 0.4 \end{bmatrix} \quad D(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_j y_j \ln \hat{y}_j \quad \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

The total cross entropy loss will be

D = [0 * log(0.1) + 1 * log(0.5) + 0 * log(0.4)]

Visualization Techniques

1. Layer activations
   a. Visualise the activations of the network. When we use ReLU, the activations starts out relatively blobby but spreads out during learning.
   b. Some activations may be all zero indicating high learning rate.
2. Visualise weights
   a. Weights are the most interpretable on the first layer, which is looking at the input pixels directly.
   b. Weights from other layers can be visualised too. They will usually form some smooth patterns. Noisy patterns indicate that the network has not probably learnt well and needs to trained longer.
3. Retrieving images that maximally excite a neuron.
   a. A large dataset of images is taken. The images which fired maximally for some neuron are recorded. Hence this will give us a good insight into what the neuron has learnt.
   b. One problem with this is that each Relu neuron might not learn something sematic. It is the combination of several Relu neurons that learn something semantic.

*Ref- http://cs231n.stanford.edu/

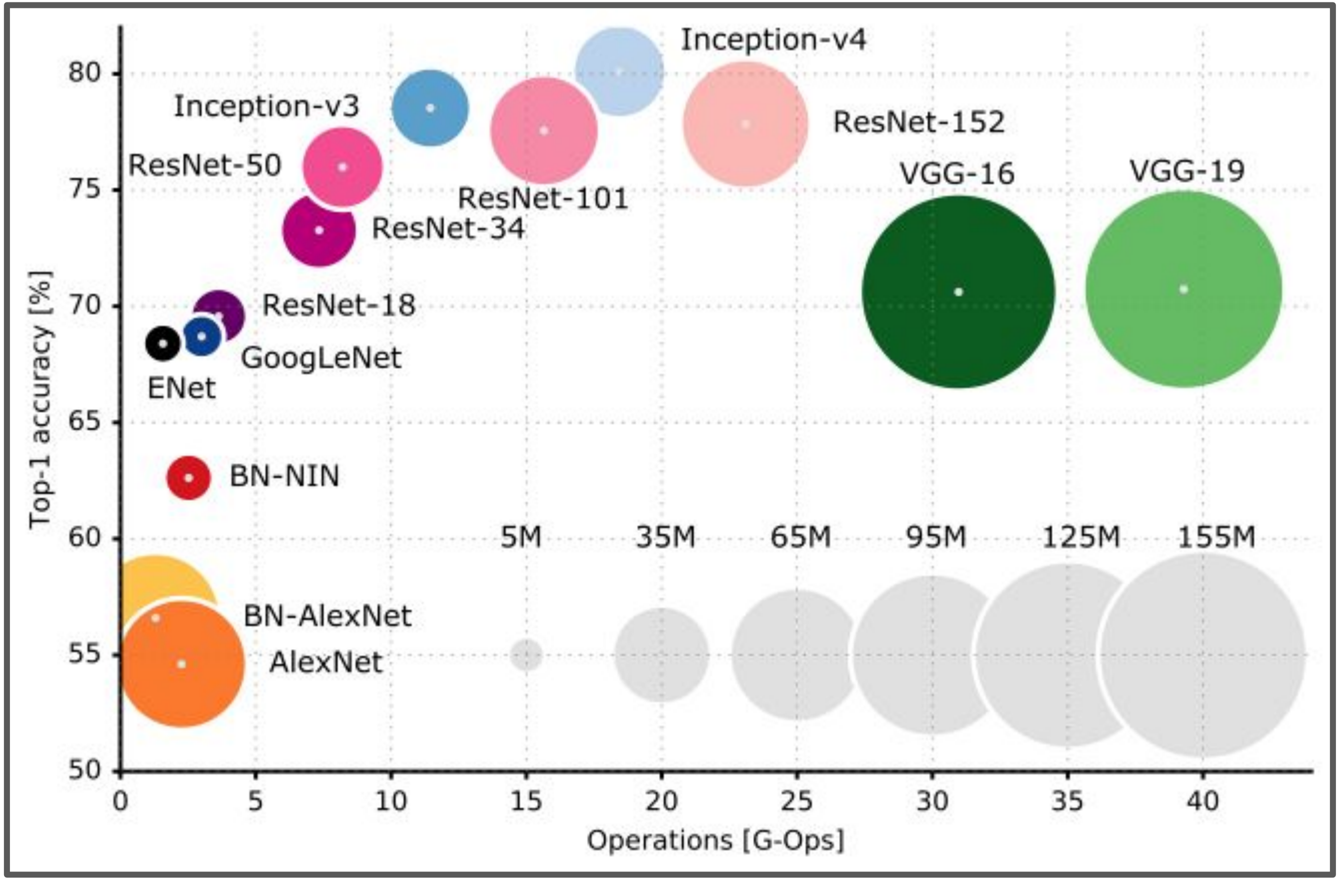
Visualization examples

1. Low dimensional embedding
    a. Several visualisation techniques have been proposed which convert the image vectors in high dimensional space to a 2-D space, preserving the pairwise distance between any two points. Eg: tsne! tsne_mnist, tsne_karpathy.
2. Occluding parts of image (can be used for classification):-
    a. We can set a patch of the image to be all zeros. We can iterate position of the patch throughout the image and record the probability of correct class label as a function of position.
    b. A 2-dimensional heat map can be produced through such procedure. The probability should reduce considerably at position where the actual object is placed in the image

Soliton
Vision for a Better World

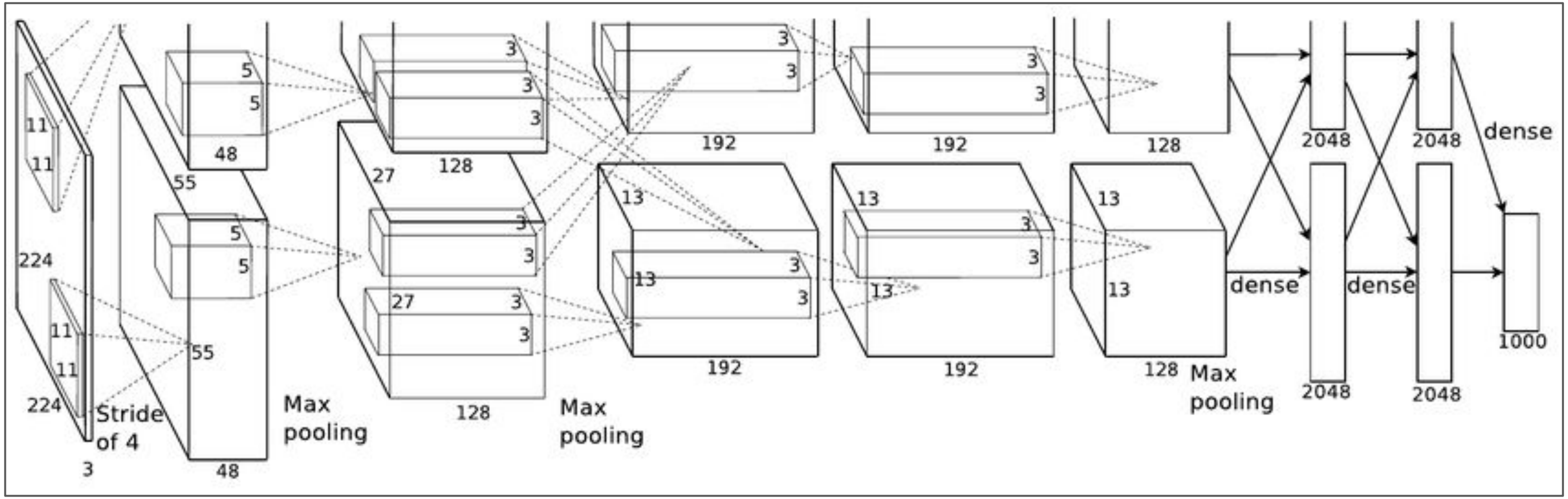
**Ref- http://cs231n.stanford.edu/

Network Architectures



Soliton
Vision for a Better World

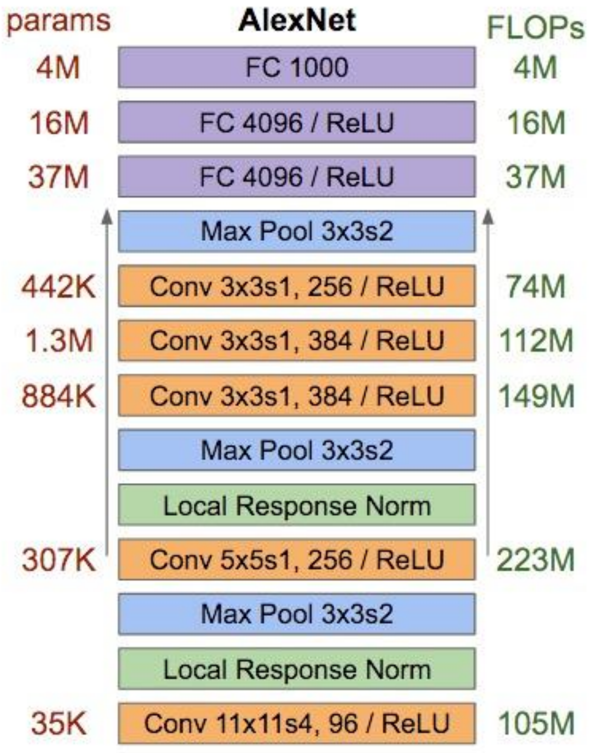
AlexNet: The game-changer



Soliton
Vision for a Better World

AlexNet

- Convolution and pooling layers on top of each other.
- It consists of 8 layers - 5 convolutional layers and 3 fully connected layers.
- Uses Relu, dropout, max pool, GPU's!



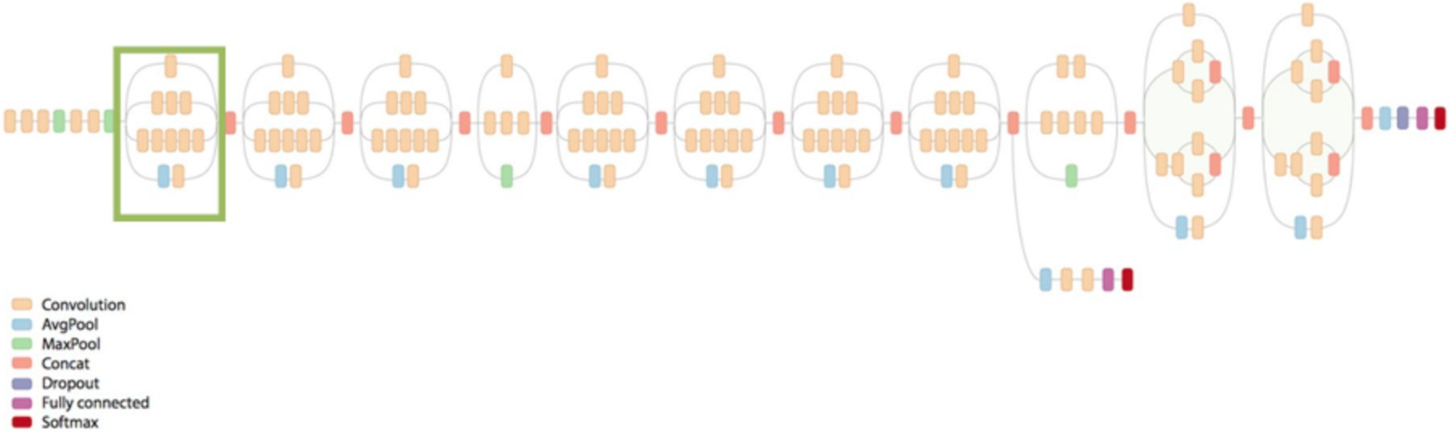
VGG

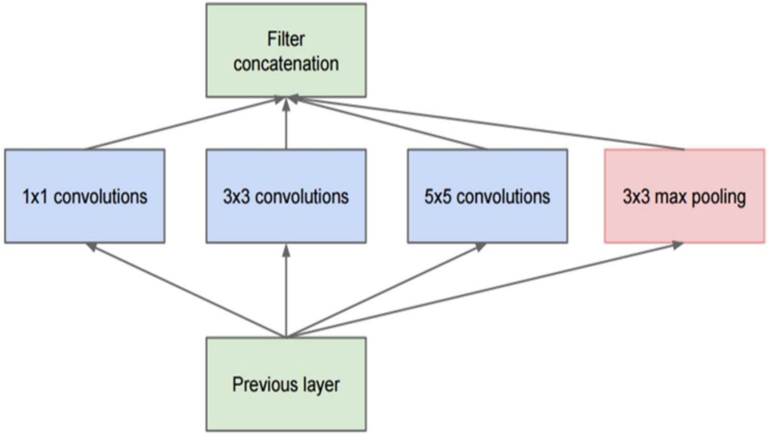
- Deeper, 19 layers.
- Uses 3*3 Conv filters, over 11*11 and 5*5 as in AlexNet

Soliton
Vision for a Better World

*Img Ref - https://groups.google.com/forum/#!topic/caffe-users/cUD3IF5NMOk
* Ref - http://vision.stanford.edu/teaching/cs231b_spring1415/slides/alexnet_tugce_kyunghee.pdf

Googlenet





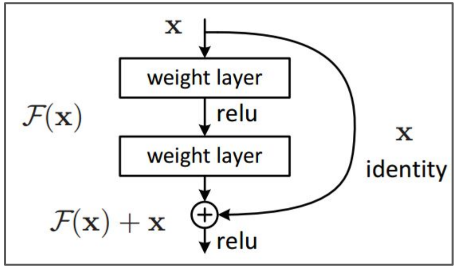
- **Inception module** - A module where features from multiple layers can be mixed together.
- Inception provides provides a way for combining local and global features.
- GoogleNet trains faster than VGG.
- It contains 22 layers but has less computations and consumes less memory.

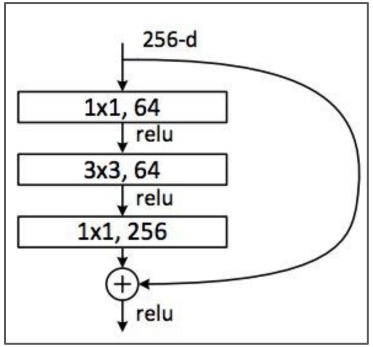
*Ref - https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Szegedy_Going_Deeper_With_2015_CVPR_paper.pdf
* Ref - https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html

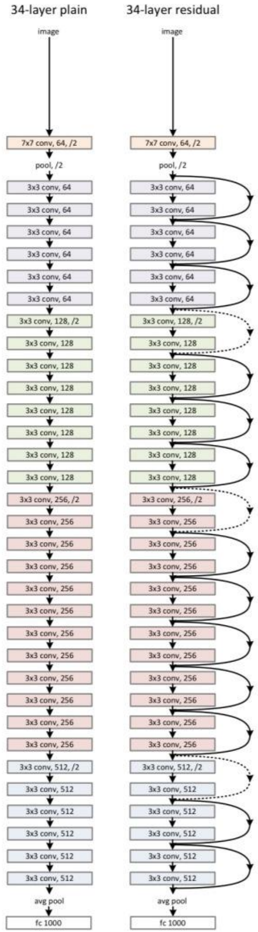
Soliton
Vision for a Better World

Residual Networks(ResNet)

The idea of skip connection suggests that it is easier to learn minor modifications to identity functions







*Ref - https://arxiv.org/pdf/1512.03385.pdf
*Ref - https://towardsdatascience.com/neural-network-architectures-156e5bad51ba

ResNeXt

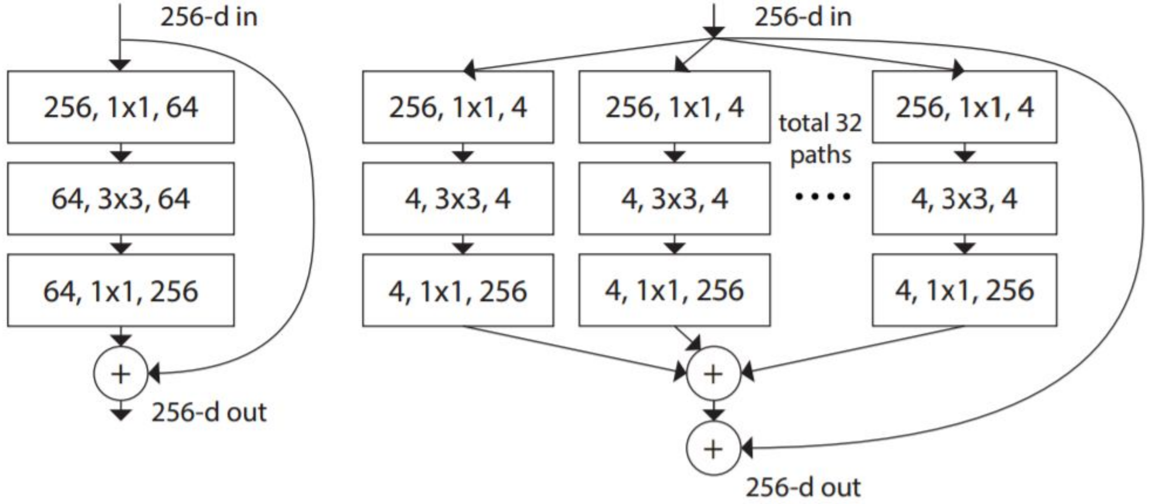


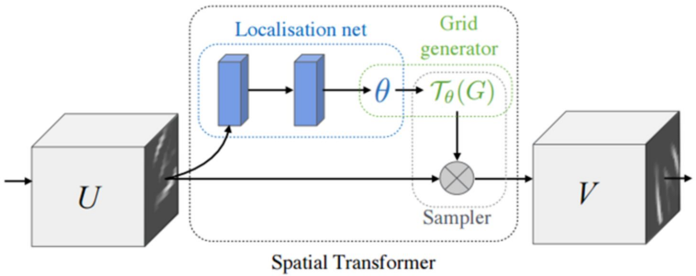
Figure 1. **Left**: A block of ResNet [14]. **Right**: A block of ResNeXt with cardinality = 32, with roughly the same complexity. A layer is shown as (# in channels, filter size, # out channels).

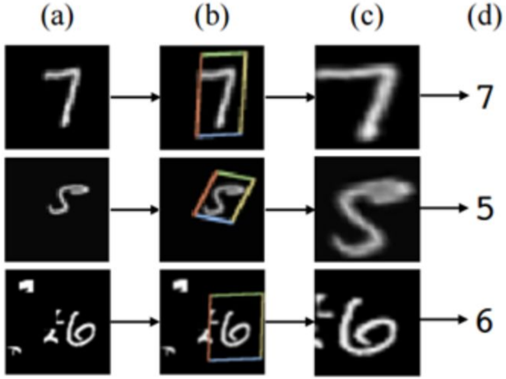
- Inception
- Residual Blocks
- Network-in-network(1*1 Convolutions)

*Ref - https://towardsdatascience.com/neural-network-architectures-156e5bad51ba
*Ref - Network in Network

Spatial Transformer Networks (STN)

- Introduces a network to make images invariant to rotations and translations
- It consists of a localisation network that computes the spatial transformation, creation of sampling grid through a grid generator and a sampler which warps the input based on the generated grid.
- An affine transformation followed by bilinear interpolation
- Eliminates the use of Pooling!
- [Demo]



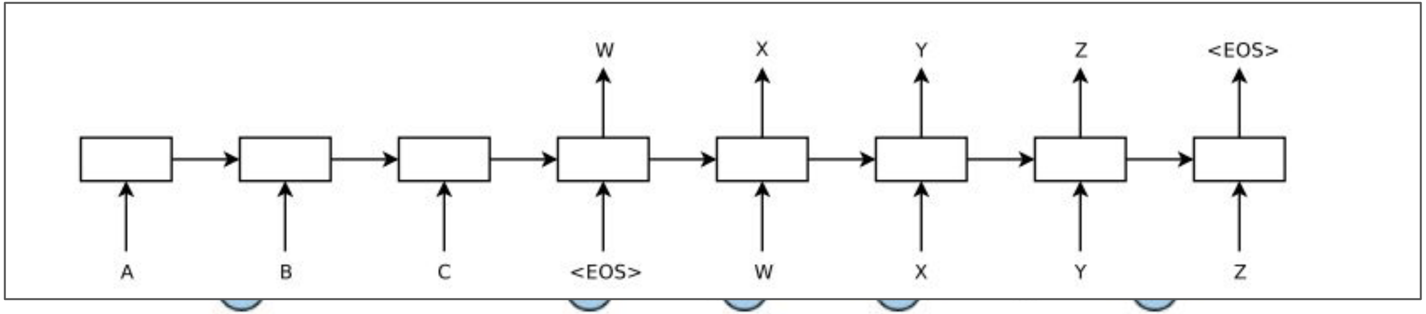


Soliton
Vision for a Better World

*Ref - https://arxiv.org/pdf/1506.02025.pdf
*Ref - https://kevinzakka.github.io/2017/01/18/stn-part2/

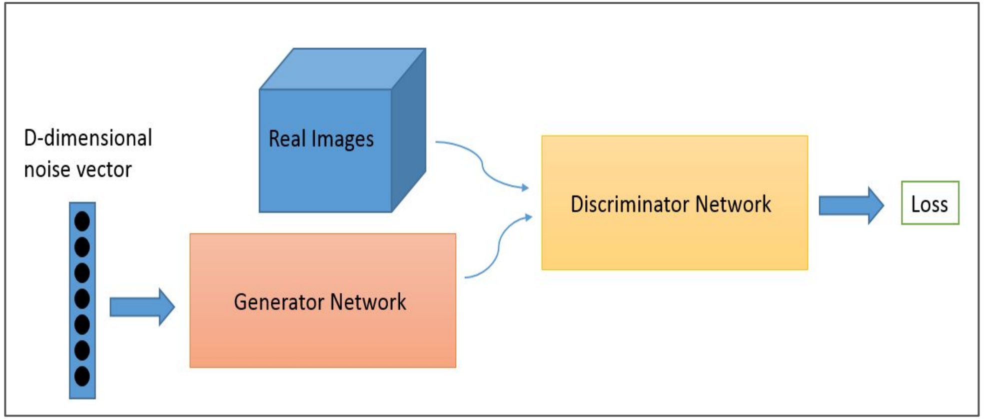
Recurrent Neural Network(RNN)

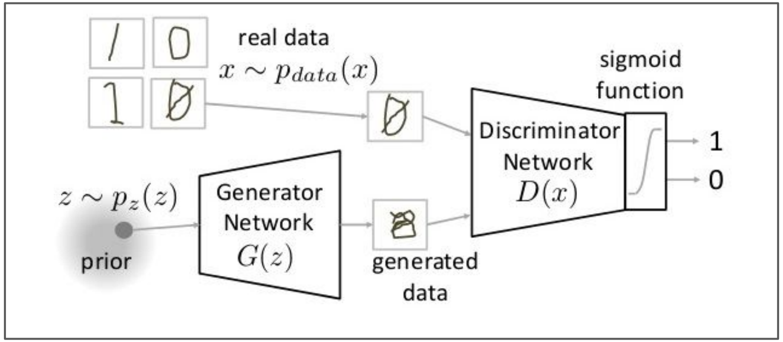
- Unlike feedforward neural network, RNN can have loops.
- Useful for Sequential inputs.
- Useful in applications where memory about the past output can play a role in predicting the current output.
- Very useful in text analytics.
- Inefficient for storing long seq, vanishing gradients, not hardware friendly, use hierarchical neural Attention encoder Instead!



*Ref - http://colah.github.io/posts/2015-08-Understanding-LSTMs/
*Ref - http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

GAN(Generative Adversarial Networks)





- Two Networks: Generator and Discriminator
- Both have CNN and DeConv
- The task of the generator is to create natural looking images that are similar to the original data distribution.
- Discriminator tries to tell if the image is generated artificially by Generator Network or a Real Image.
- Generator tries to fool the Discriminator and hence generate real-like images!

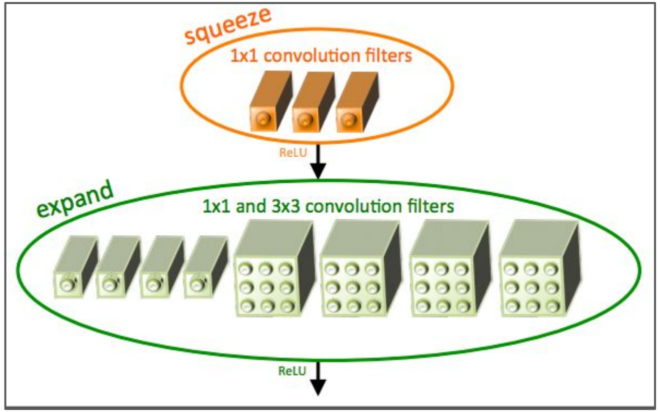
Soliton
Vision for a Better World

*Ref - https://github.com/uclaacmai/Generative-Adversarial-Network-Tutorial/blob/master/Generative%20Adversarial%20Networks%20Tutorial.ipynb

SqueezeNet

- 3x Faster, 500 times smaller than Alexnet with same Accuracy!
- How: Uses fire module with deep compression!
- Downsample late to keep a big feature map.

Fire Module



EIE: Efficient Inference Engine on Compressed Deep Neural Network

1. Trained Pruning
2. Trained Quantisation
3. Huffman Coding
4. Efficient Inference Engine(FPGA based Hardware)

Table 2: Comparing SqueezeNet to model compression approaches. By *model size*, we mean the number of bytes required to store all of the parameters in the trained model.

CNN architecture	Compression Approach	Data Type	Original → Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD (Denton et al., 2014)	32 bit	240MB → 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning (Han et al., 2015b)	32 bit	240MB → 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression (Han et al., 2015a)	5-8 bit	240MB → 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	**50x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB → 0.66MB	**363x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB → 0.47MB	**510x**	57.5%	80.3%

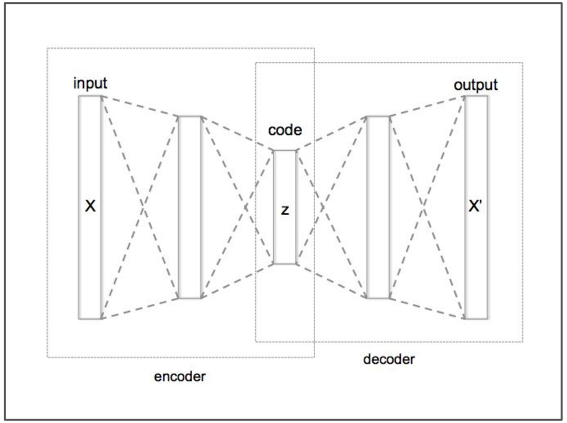
*Ref - https://arxiv.org/pdf/1602.01528.pdf
*Ref - https://arxiv.org/pdf/1602.07360.pdf

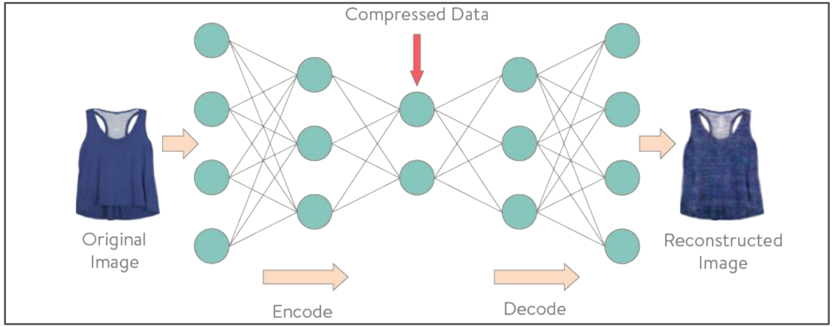
Autoencoders

- **Stage1(EnCode)** : Used to learn a fixed number of features that can best represent the data.
- **Stage2(Decode)** : Used to reconstruct input from the code(i.e Stage 1 output).
- Can be used for data compression.

Drawbacks:

- Data Specific (A model trained on human faces would not work on modern buildings!)
- Lossy Representation (Like JPEG compression)





*Ref - https://en.wikipedia.org/wiki/Autoencoder
*Ref - https://www.slideshare.net/TJTorres1/deep-style-using-variational-autoencoders-for-image-generation

DL in Computer vision Applications

Demos: ?

Image segmentation

Style Transfer

GAN

Face Fake

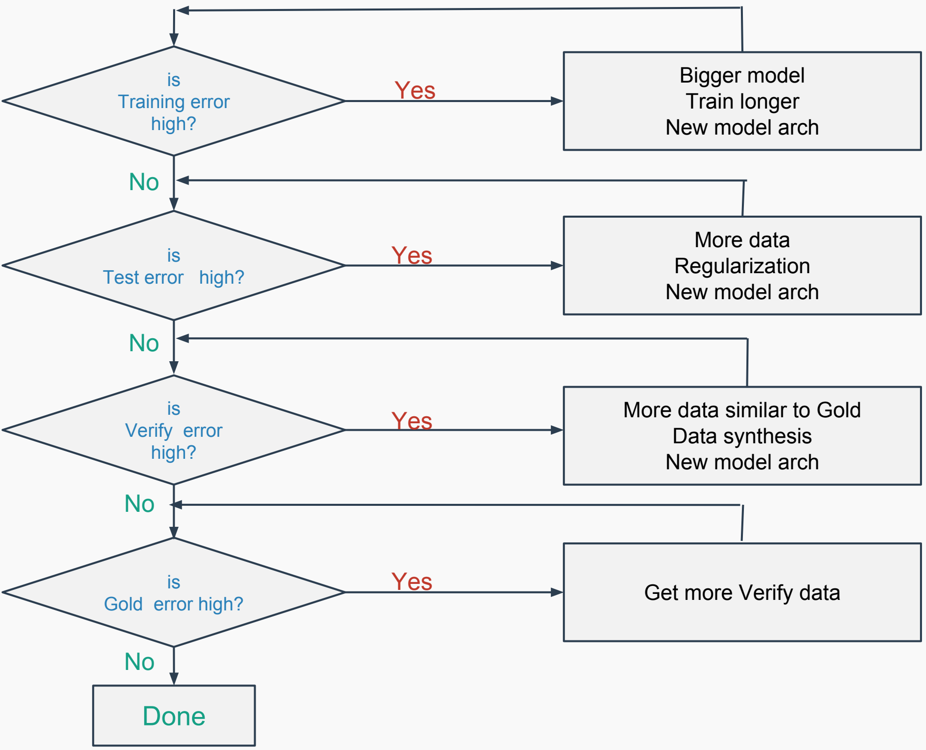
RL

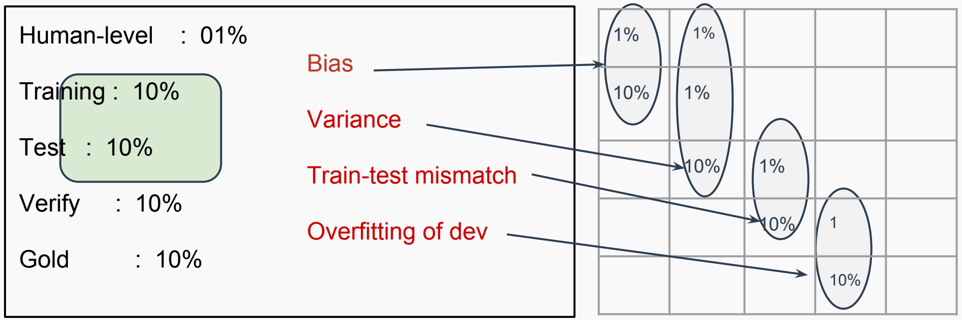
Soliton
Vision for a Better World

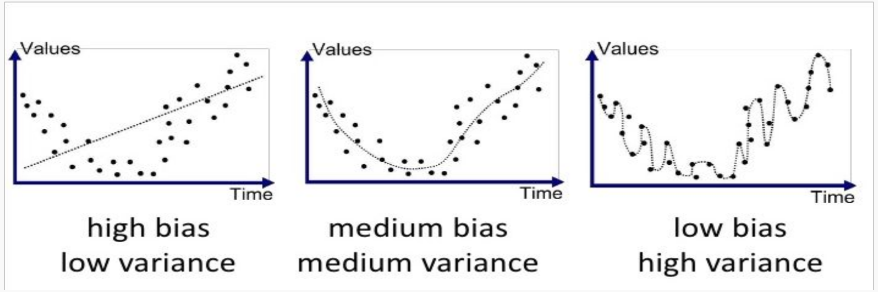
DL Advice

- Importance of **Clean Data & Representative Data**
- Can you **overfit** your model (Near Human level accuracy on subset of Training data)?
- Build **tools** to effectively **view large quantity of data**
- **Balance dataset** or find ways to Handle that!
    - Subsampling
    - Balance accuracy metric, ROC Curve, Precision/Recall
- Balance General Enough vs Over General (Over greedy is bad)
- Human in the loop
- Keep **up-to date with tools/libraries**

Disciplined ML/DL Training







Based on Andrew Ng ‘2013

Demos

- GAN
  - https://github.com/uclaacmai/Generative-Adversarial-Network-Tutorial/blob/master/Generative%20Adversarial%20Networks%20Tutorial.ipynb
  - https://github.com/erschmidt/Jupyter-GAN/blob/master/Labeled-GAN-MNIST.ipynb
  - https://github.com/Ujitoko/GAN/blob/master/vanilla_GAN/Vanilla_GAN_keras.ipynb
  -

Soliton
Vision for a Better World

Soliton
Vision for a Better World

*Img Ref- https://www.theodysseyonline.com/remember-always-body

References

1. http://cs231n.stanford.edu/
2. http://cs229.stanford.edu/
3. http://colah.github.io/
4. https://github.com/janishar/mit-deep-learning-book-pdf
5. Multiple View Geometry in Computer Vision (Second Edition) : Richard H, Andrew Z