

# (An Almost) No Math Introduction to Deep Learning

Abhijat Vatsyayan <sup>1</sup>

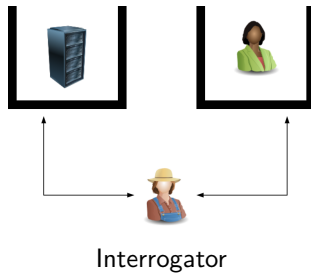
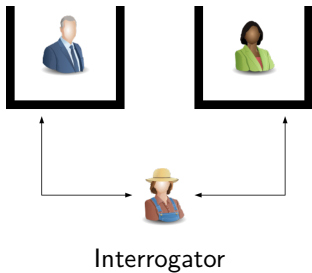
November 20, 2020

# Summary

- 1 Introduction
- 2 A little bit of math
- 3 Image processing
- 4 Machine learning
- 5 Fitting functions
- 6 References

# What is Artificial Intelligence?

# Imitation game



# Artificial intelligence

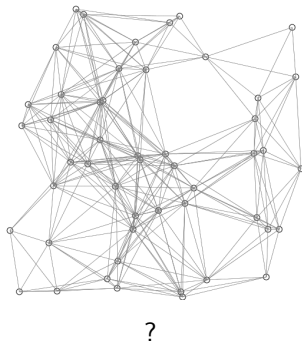
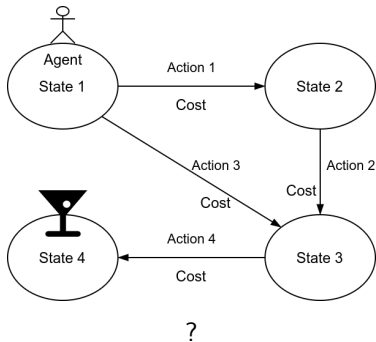
- Logic machines (50s)
- Knowledge based expert systems (80s)
- Language translation (60s) , 2000s, 2014 and later.
- Machine learning
  - Neural networks including deep learning (started in 1943)
  - Support vector machines
  - Bayesian learning
- Graphs
- Genetic algorithms and genetic programs.

# Expert systems

- Database of formally described "facts" or "knowledge".
- A reasoning engine for answering questions or solving problems.

Not to be confused with a true natural language processing and question-answering system.

# Search

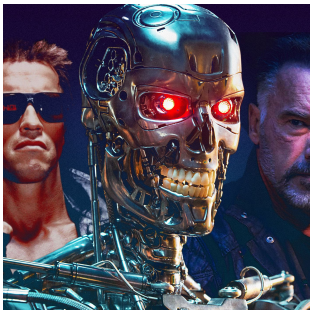


# Neural Networks

- 1943: Warren McCulloch and Walter Pits connected neurons, computation, logic and learning.
- 1950: Minsky and Dean Edmonds build first neural network computer. 3000 vacuum tubes, surplus auto-pilot parts from B-24 bomber. 40 Neurons.
- 1969: Minsky and Papert publishobama-funny perceptron - simple linear networks could not learn basic functions.
- 1980s: David Rumelhart, Jeff Hinton and Ronald Williams applied back propagation (again) for training multi-layer neural networks. Rumelhart's work also created the foundations for Recurrent Neural Networks.
- 1990s: LSTM networks by Hochreiter and Schmidhuber 1997. CNN for handwritten digit recognition - Yann LeCun.
- 2000s: LSTMs show promise in speech recognition
- 2012: Deep learning



# Different views



Agent



Tool

General

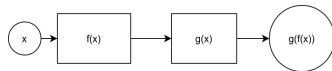
Narrow

# Composition of functions

$$f(x) = ax + b \quad (1)$$

$$g(x) = \frac{1}{e^{-x} + 1} \quad (2)$$

$$g(f(x)) = \frac{1}{e^{-(ax+b)} + 1} \quad (3)$$



Composite

# Derivatives I

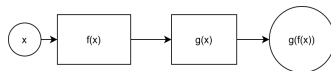
$$f(x) = ax + b \quad (4)$$

$$\frac{df(x)}{dx} = a \quad (5)$$

$$\sigma(z) = \frac{1}{e^{-z} + 1} \quad (6)$$

$$\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z)) \quad (7)$$

$$\frac{d\sigma(f(x))}{dx} = ? \quad (8)$$



Composite

# Derivatives II

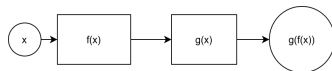
$$z = ax + b \quad (9)$$

$$\frac{dz}{dx} = a \quad (10)$$

$$\sigma(z) = \frac{1}{e^{-z} + 1} \quad (11)$$

$$\frac{d\sigma}{dz} = \sigma(z)(1 - \sigma(z)) \quad (12)$$

$$\frac{d\sigma}{dx} = \frac{d\sigma}{dz} \frac{dz}{dx} \quad (13)$$



Composite

# Derivatives III

$$z = ax + b \quad (14)$$

$$\frac{dz}{dx} = a \quad (15)$$

$$\sigma(z) = \frac{1}{e^{-z} + 1} \quad (16)$$

$$\frac{d\sigma}{dz} = \sigma(z)(1 - \sigma(z)) \quad (17)$$

$$\frac{d\sigma}{dx} = \frac{d\sigma}{dz} \frac{dz}{dx} = \frac{dz}{dx} \frac{d\sigma}{dz} \quad (18)$$



Back propagation of gradients

# Datasets

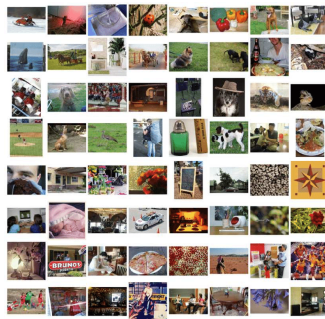
- Modified National Institute of Standards and Technology - MNIST (60k/10k)
- Canadian Institute For Advanced Research - CIFAR-10 (50k/10k) and CIFAR-100 (2 level, 500/100)
- Pascal Visual Object Classes (VOC) - 22k images, 20 classes
- ...
- ImageNet

# ImageNet Large Scale Visual Recognition Challenge I

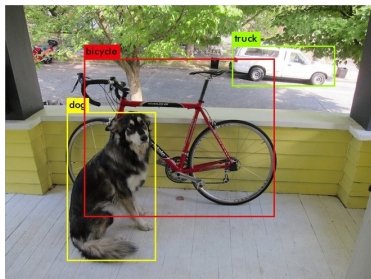
MNIST Dataset (60k, 10k)



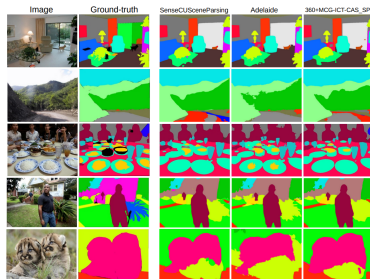
ImageNet (14M+, 22k)



# ImageNet Large Scale Visual Recognition Challenge II



Localization

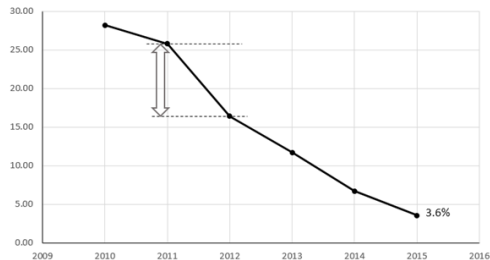


Segmentation



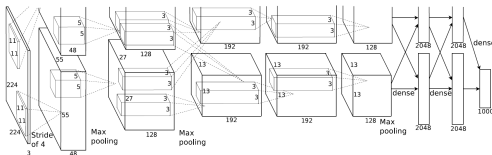
# ImageNet Large Scale Visual Recognition Challenge III

- Publicly available dataset - ImageNet (14M+, 22k categories)
- Annual competition
  - Image classification
  - Object detection and localization
- Increasing depth
  - 8 layer AlexNet
  - 19 layer GoogLeNet
  - 152 layer ResNet



Top 5 classification error rate

# ImageNet Large Scale Visual Recognition Challenge IV



Alex Krizhevsky, Sutskever, Ilya and Hinton, Geoffrey E., "ImageNet Classification with Deep Convolutional Neural Networks", 2012

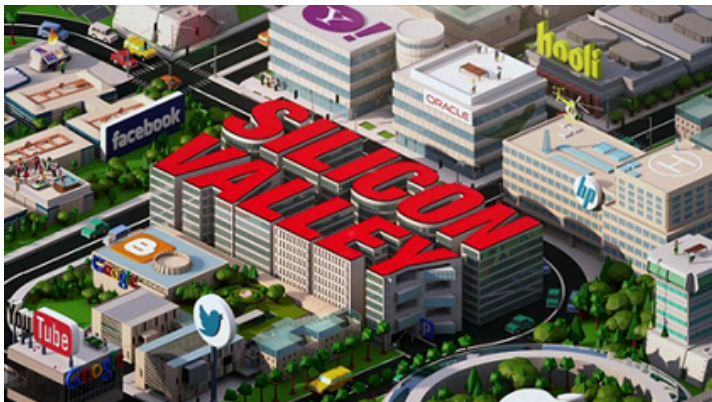
AlexNet:] 61m parameters, 8 layers

GoogLeNet: 6.7m parameter, 22 layers

ResNet: ResNet-50, ResNet-101, ResNet-152

Question: Are deeper networks always better?

# Not hotdog



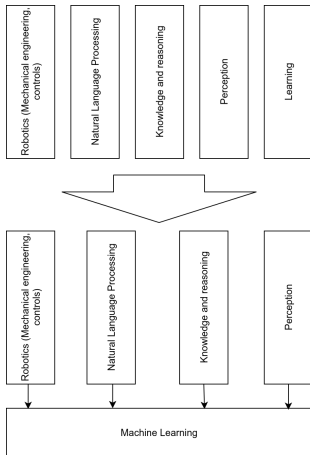
► [https://www.youtube.com/watch\\_popup?v=vlci3C4JkL0](https://www.youtube.com/watch_popup?v=vlci3C4JkL0)

# Why is this picture is funny?



Credit: <http://karpathy.github.io/2012/10/22/state-of-computer-vision/>

# Late 2000s - machine learning dominates



- Image classification, localization and segmentation
- Neural machine translation, question answering, summary.
- Game playing, helicopter flying (stunts)
- Planning, self driving cars
- Text, audio and video processing, generation
- ...

# Machine learning I

## Models

- Build a model of the world
- Infer/predict using the model.

## Machine learning

- Supervised learning
- Unsupervised learning
- Reinforcement learning

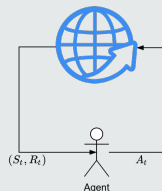
# Machine learning II

## Unsupervised learning

- Training a model to find patterns in a dataset, typically an unlabeled dataset.
- Learning how to extract *interesting* features.
- Learning data distribution for generating data.

## Reinforcement learning

A family of algorithms that learn an optimal policy, whose goal is to maximize return when interacting with an environment.



# Supervised learning I

Detail Compact Column

# sepal_length	# sepal_width	# petal_length	# petal_width	Δ species
5.7	2.9	4.2	1.3	Iris-versicolor
6.2	2.9	4.3	1.3	Iris-versicolor
5.1	2.5	3	1.1	Iris-versicolor
5.7	2.8	4.1	1.3	Iris-versicolor
6.3	3.3	6	2.5	Iris-virginica
5.8	2.7	5.1	1.9	Iris-virginica
7.1	3	5.9	2.1	Iris-virginica
6.3	2.9	5.6	1.8	Iris-virginica
6.5	3	6.8	2.2	Iris-virginica

Detail Compact Column

5 of 5 columns

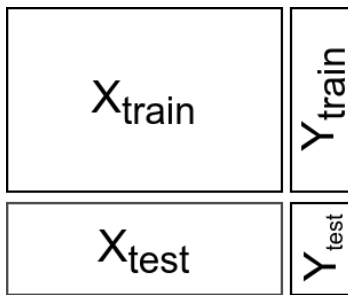




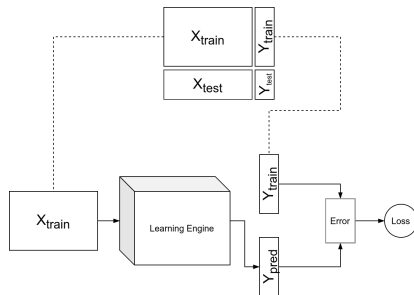
# Supervised learning II

Source: <https://www.kaggle.com/arshid/iris-flower-dataset?select=IRIS.csv>

150 rows, 5 attributes (columns), 4 numerical and 1 categorical.

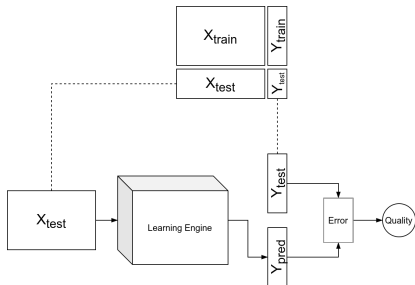


Data

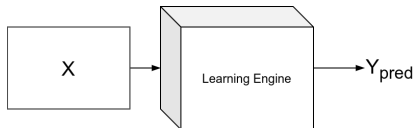


Training

# Supervised learning III



Testing



Predicting

# Fitting a function to data

## Description (supervised learning)

Given a set of data  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ , can we find a function  $y = f(x)$  that "fits" this data?

## Questions

- What is this function  $f(x)$ ?
- What does "fit" mean?
- How do we know this works?
- What kinds of problems can we solve?

# More about $f(x)$

## Class of functions

Starting with a function  $f(x; \theta_1, \theta_2, \dots, \theta_n)$  where  $x$  is the input to the function and  $\theta$ s are its parameters, we need to find the set of  $\theta$ s that best "fits" the give data  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$

## Class of linear functions

Consider  $f(x) = ax + b$ . If we can say, with some confidence, that our data is linearly related, we need to find  $\theta_1 = a$ ,  $\theta_2 = b$  that *fits* the given data. We can also write it as  $f(x; a, b) = ax + b$ .

Preferred,

$$f(x; \theta_1, \theta_2) = \theta_1 x + \theta_2 \quad (19)$$

# Fit

## Mean squared Euclidean distance as one possible measure of fit

Let  $L_i$  be the squared Euclidean distance between the predicted value,  $\hat{y}_i = f(x_i)$  and the actual,  $y_i$ . Then,

$$L_i = z_i^2 \quad (20)$$

$$z_i = y_i - f(x_i) \quad (21)$$

$$= y_i - \theta_1 x_i - \theta_2 \quad (22)$$

Minimizing  $L_i$  with respect to the parameters  $\theta_1$  and  $\theta_2$ ,

$$\frac{\partial L_i}{\partial \theta_1} = \frac{\partial z_i^2}{\partial z_i} \frac{\partial z_i}{\partial \theta_1} \quad (23)$$

$$\frac{\partial L_i}{\partial \theta_2} = \frac{\partial z_i^2}{\partial z_i} \frac{\partial z_i}{\partial \theta_2} \quad (24)$$

For  $n$  data points, mean loss is

$$L = \frac{1}{n} \sum_{i=1}^{i=n} L_i = \frac{1}{n} \sum_{i=1}^{i=n} (y_i - \theta_1 x_i - \theta_2)^2$$

In practice

- Choose a small (64 or 128) random subset of training data.
- Compute predicted values, then loss.
- Compute gradients of loss W.R.T. parameters then update parameters.

An **epoch** refers to a single iteration over all training data.

## Two different spaces

- Space spanned by  $x$  and  $y$ . Optimization tries to find the surface (model) in this space that best fits the data.
- Space spanned by  $\theta$ s. We minimize the loss function in this space.

# References I



V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” 2018.



Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov 1998.



I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in *International Conference on Learning Representations*, 2015.




A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” *ICLR Workshop*, 2017.



I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

# References II

 I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in Neural Information Processing Systems 27* (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds.), pp. 3104–3112, Curran Associates, Inc., 2014.

 S. J. Russell and P. Norvig, *Artificial Intelligence: a modern approach*. Pearson, 3 ed., 2009.