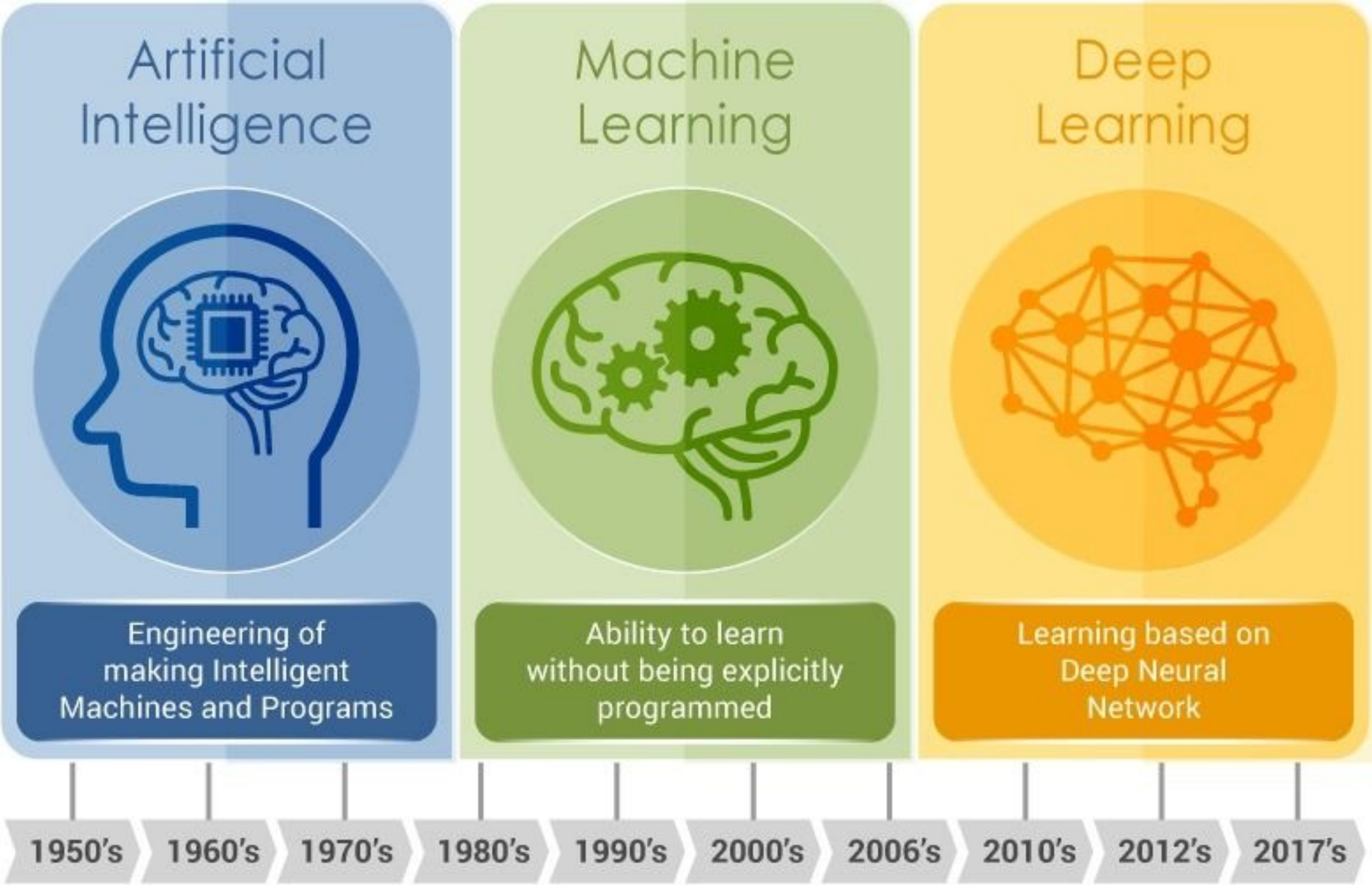
Deep Learning

PhD. Msc. David C. Baldears S.
PhD(s). Msc. Diego López Bernal

TC3007C


Artificial Intelligence
Engineering of making Intelligent Machines and Programs
Machine Learning
Ability to learn without being explicitly programmed
Deep Learning
Learning based on Deep Neural Network
1950's
1960's
1970's
1980's
1990's
2000's
2006's
2010's
2012's
2017's

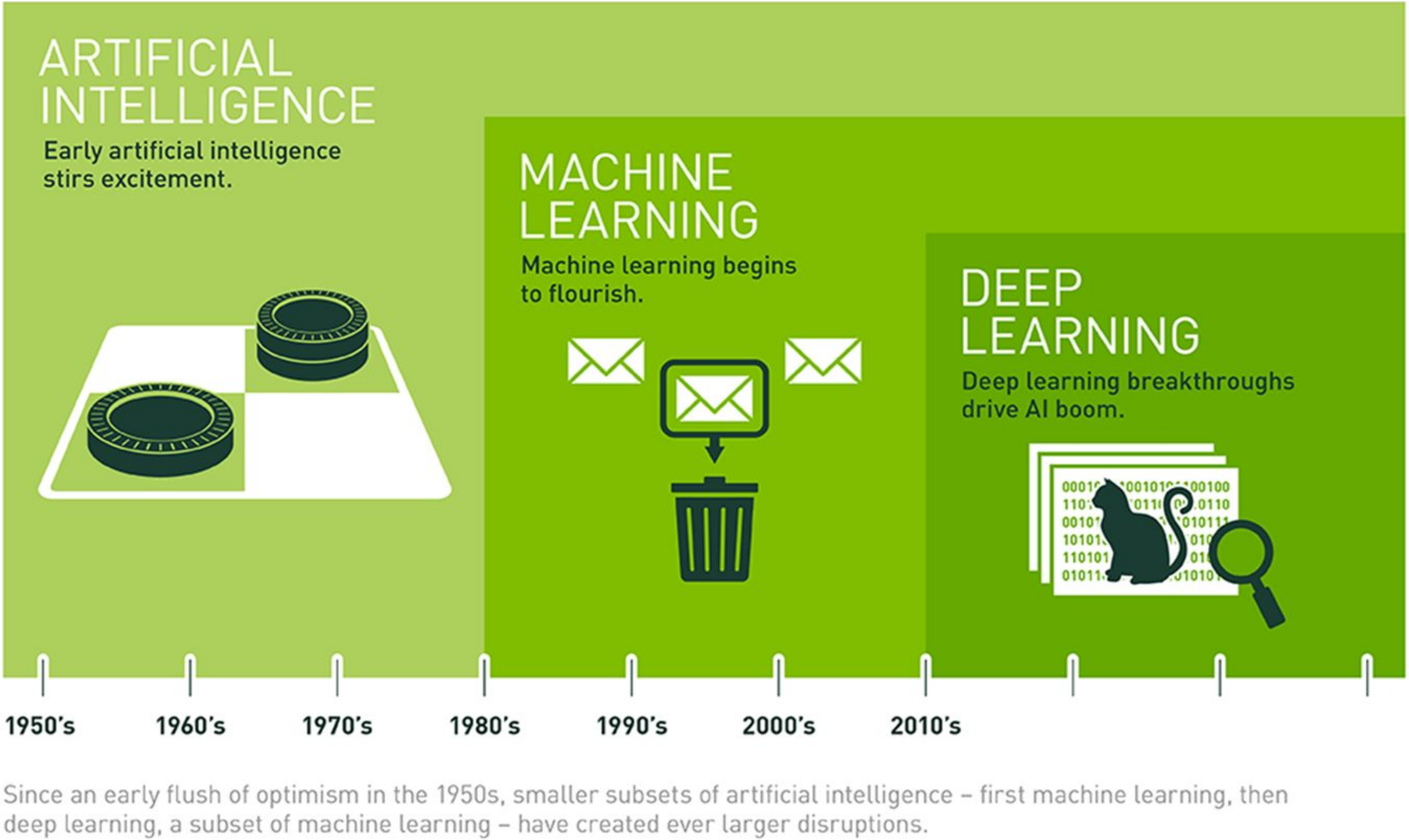
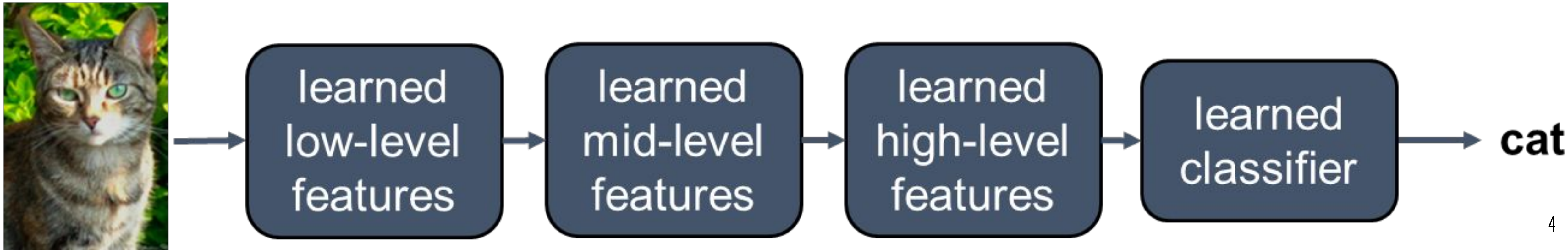


Image from https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

“Traditional” machine learning:

Deep, “end-to-end” learning:



Year	Contributer	Contribution
300 BC	Aristotle	introduced Associationism, started the history of human's attempt to understand brain.
1873	Alexander Bain	introduced Neural Groupings as the earliest models of neural network, inspired Hebbian Learning Rule.
1943	McCulloch & Pitts	introduced MCP Model, which is considered as the ancestor of Artificial Neural Model.
1949	Donald Hebb	considered as the father of neural networks, introduced Hebbian Learning Rule, which lays the foundation of modern neural network.
1958	Frank Rosenblatt	introduced the first perceptron, which highly resembles modern perceptron.
1974	Paul Werbos	introduced Backpropagation
1980	Teuvo Kohonen	introduced Self Organizing Map
	Kunihiko Fukushima	introduced Neocogitron, which inspired Convolutional Neural Network
1982	John Hopfield	introduced Hopfield Network
1985	Hilton & Sejnowski	introduced Boltzmann Machine
1986	Paul Smolensky	introduced Harmonium, which is later known as Restricted Boltzmann Machine
	Michael I. Jordan	defined and introduced Recurrent Neural Network
1990	Yann LeCun	introduced LeNet, showed the possibility of deep neural networks in practice
1997	Schuster & Paliwal	introduced Bidirectional Recurrent Neural Network
	Hochreiter & Schmidhuber	introduced LSTM, solved the problem of vanishing gradient in recurrent neural networks
2006	Geoffrey Hinton	introduced Deep Belief Networks, also introduced layer-wise pretraining technique, opened current deep learning era.
2009	Salakhutdinov & Hinton	introduced Deep Boltzmann Machines
2012	Geoffrey Hinton	introduced Dropout, an efficient way of training neural networks

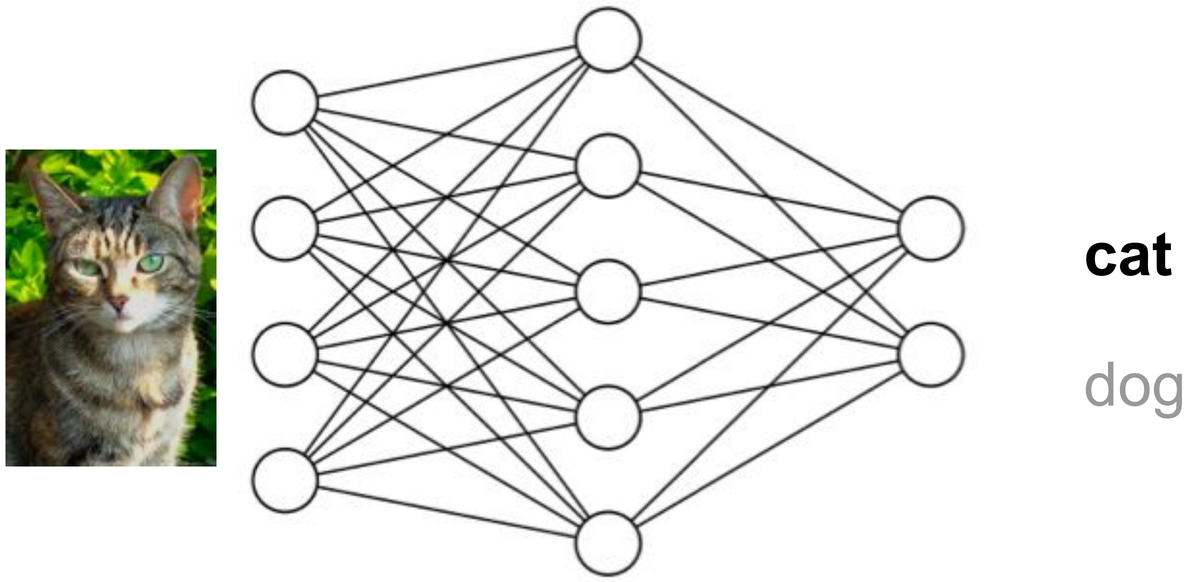
From: Wang & Raj: On the Origin of Deep Learning (2017)

Main types of machine learning

Main types of machine learning

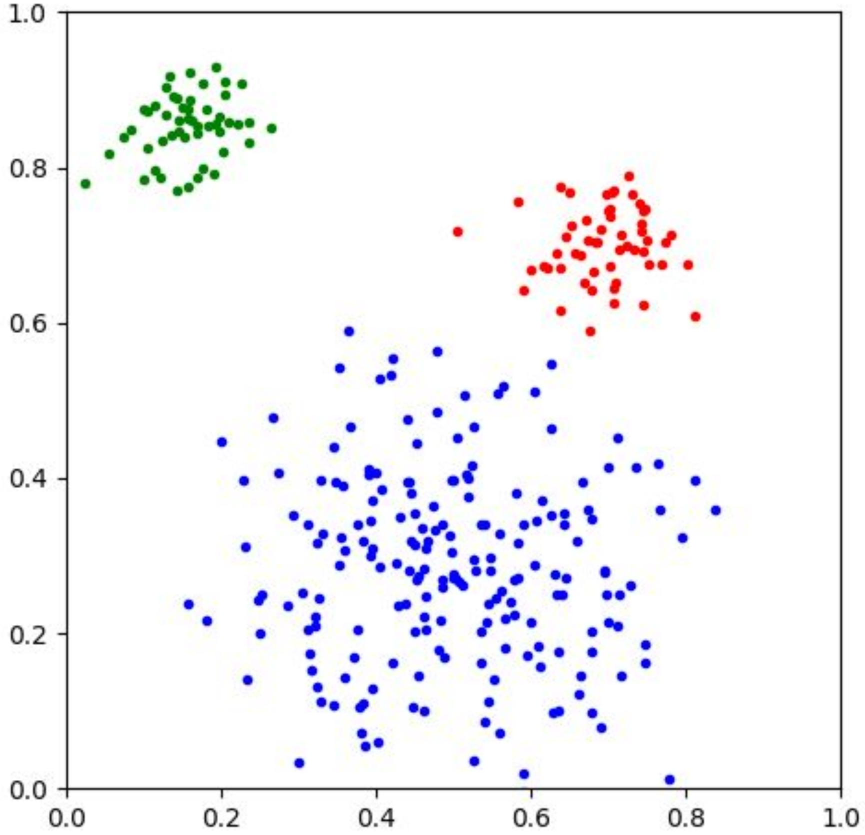
- **Supervised learning**

- Unsupervised learning
- Self-supervised learning
- Reinforcement learning



Main types of machine learning

- Supervised learning
- **Unsupervised learning**
- Self-supervised learning
- Reinforcement learning



Main types of machine learning

- Supervised learning
- Unsupervised learning

- **Self-supervised learning**

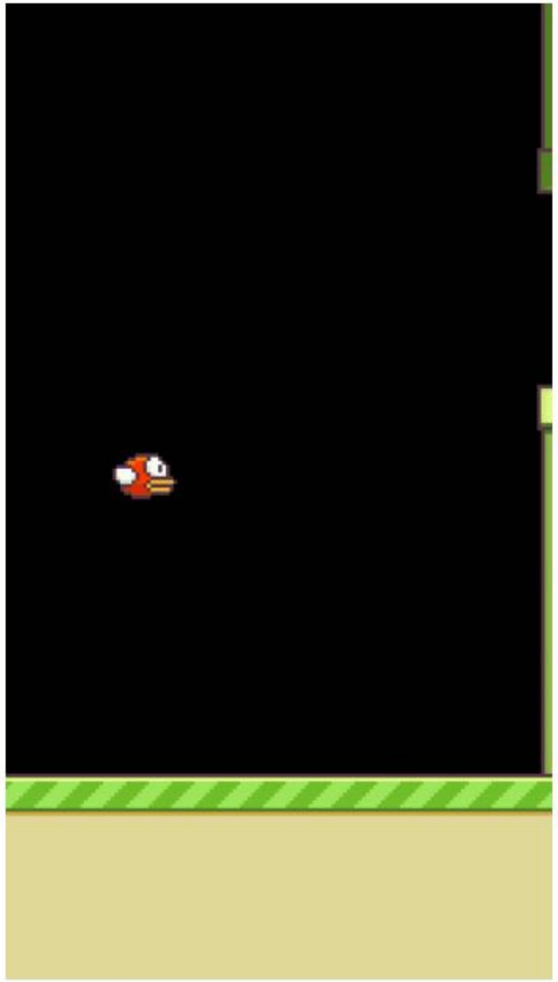
- Reinforcement learning

Image from https://arxiv.org/abs/1710.10196

Main types of machine learning

- Supervised learning
- Unsupervised learning
- Self-supervised learning

- **Reinforcement learning**

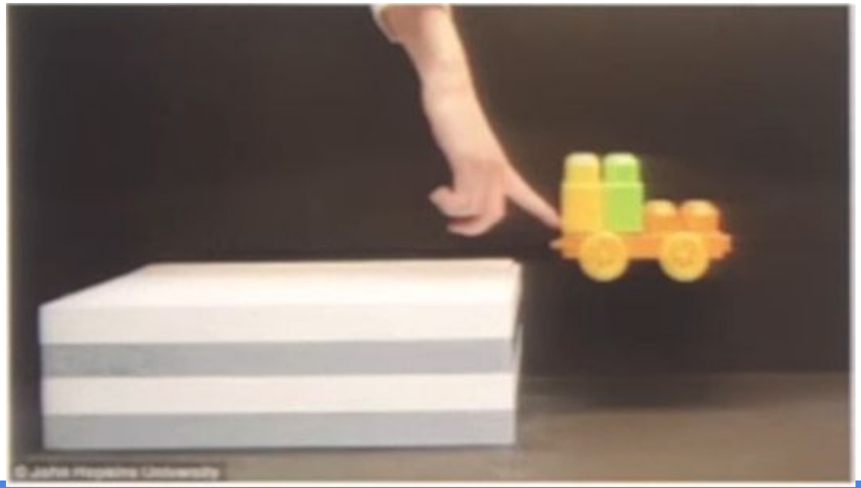


Animation from https://yanpanlau.github.io/2016/07/10/FlappyBird-Keras.html

Fundamentals of machine learning

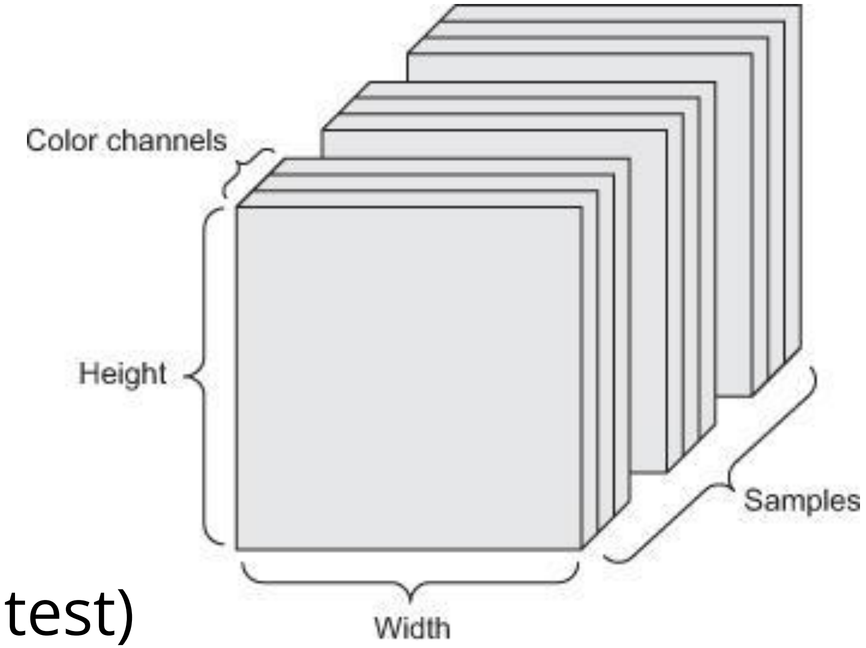
Data

- Humans learn by observation and unsupervised learning
  - model of the world / common sense reasoning
- Machine learning needs lots of (labeled) data to compensate



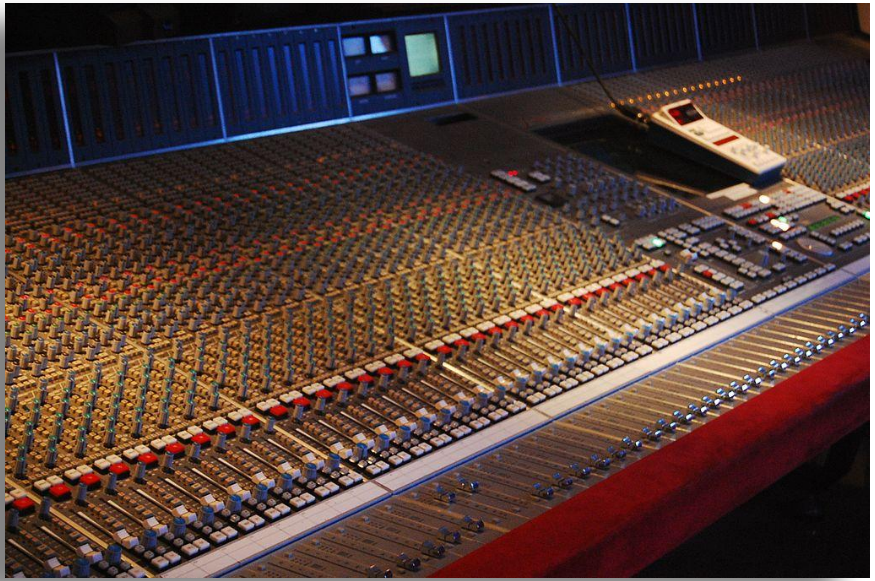
Data

- Tensors: generalization of matrices
  to n dimensions (or rank, order, degree)
  - 1D tensor: vector
  - 2D tensor: matrix
  - 3D, 4D, 5D tensors
  - numpy.ndarray(shape, dtype)
- Training – validation – test split (+ adversarial test)
- Minibatches
  - small sets of input data used at a time
  - usually processed independently



Optimization

- Mathematical optimization:
  "the selection of a best element (with regard to some criterion) from some set of available alternatives" (Wikipedia)
- Main types:
  finite-step, iterative, heuristic
- Learning as an optimization problem
  - cost function:



By Rebecca Wilson (originally posted to Flickr as Vicariously) [CC BY 2.0], via Wikimedia Commons

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \underbrace{L(f(\mathbf{x}_i;\theta), y_i)}_{\text{loss}} + \underbrace{R(\theta)}_{\text{regularization}}$$

Gradient descent

- Derivative and minima/maxima of functions
- Gradient: the derivative of a multivariable function
- Gradient descent:

$$\theta_{t+1} = \theta_t - \alpha \frac{\partial J(\theta)}{\partial \theta}$$

- (Mini-batch) stochastic gradient descent (and its variants)

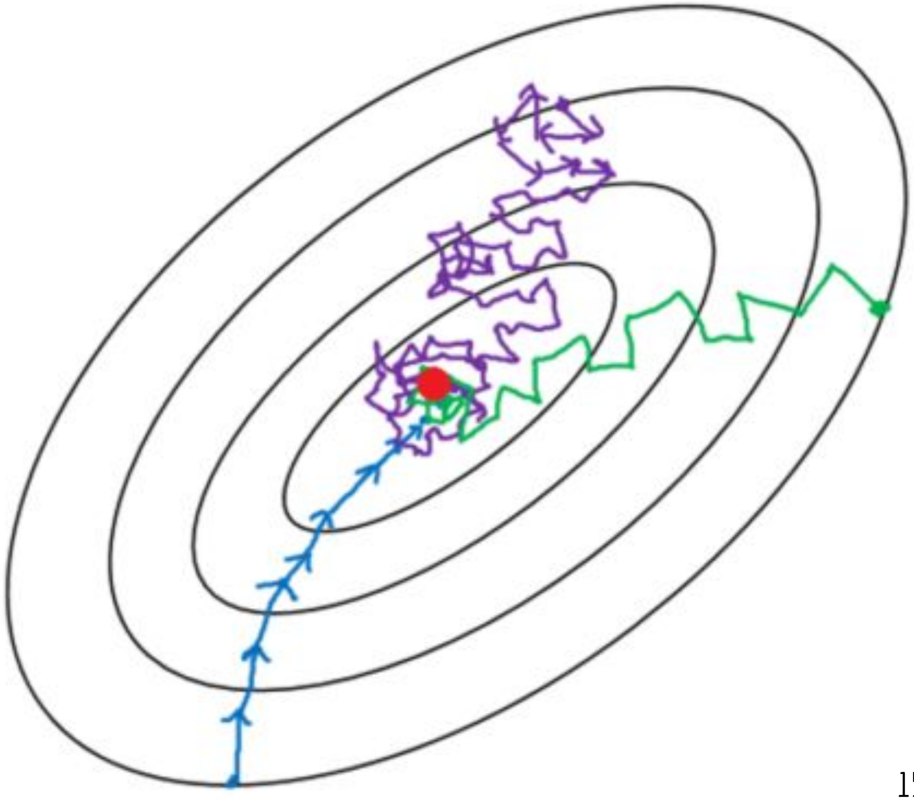
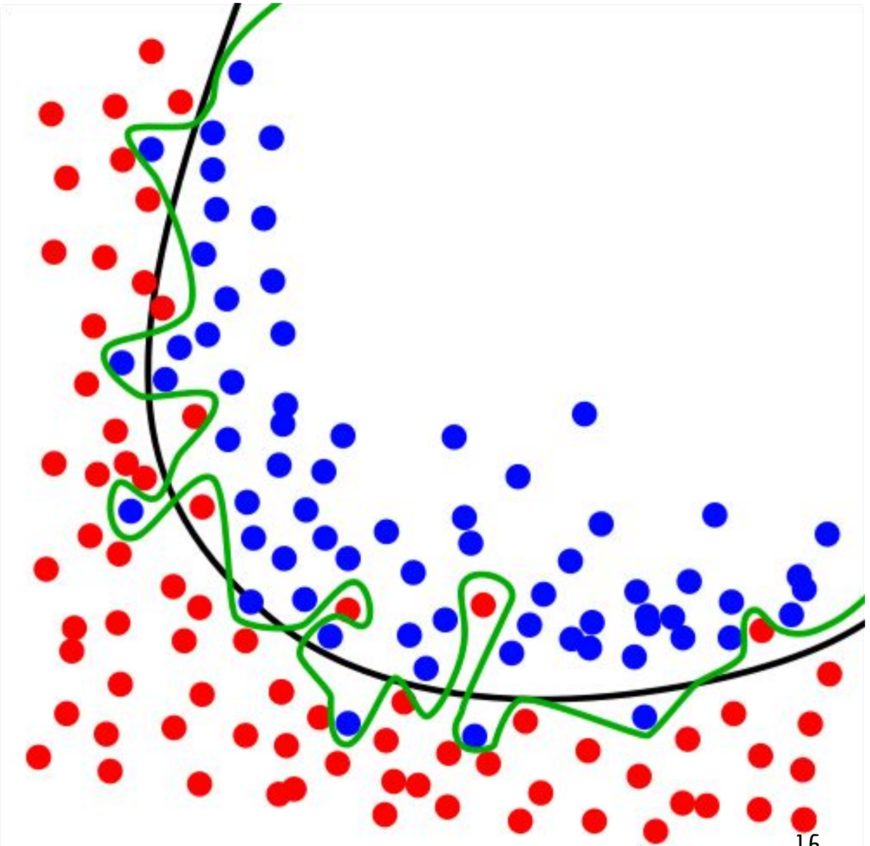


Image from: https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3

Over- and underfitting, generalization, regularization

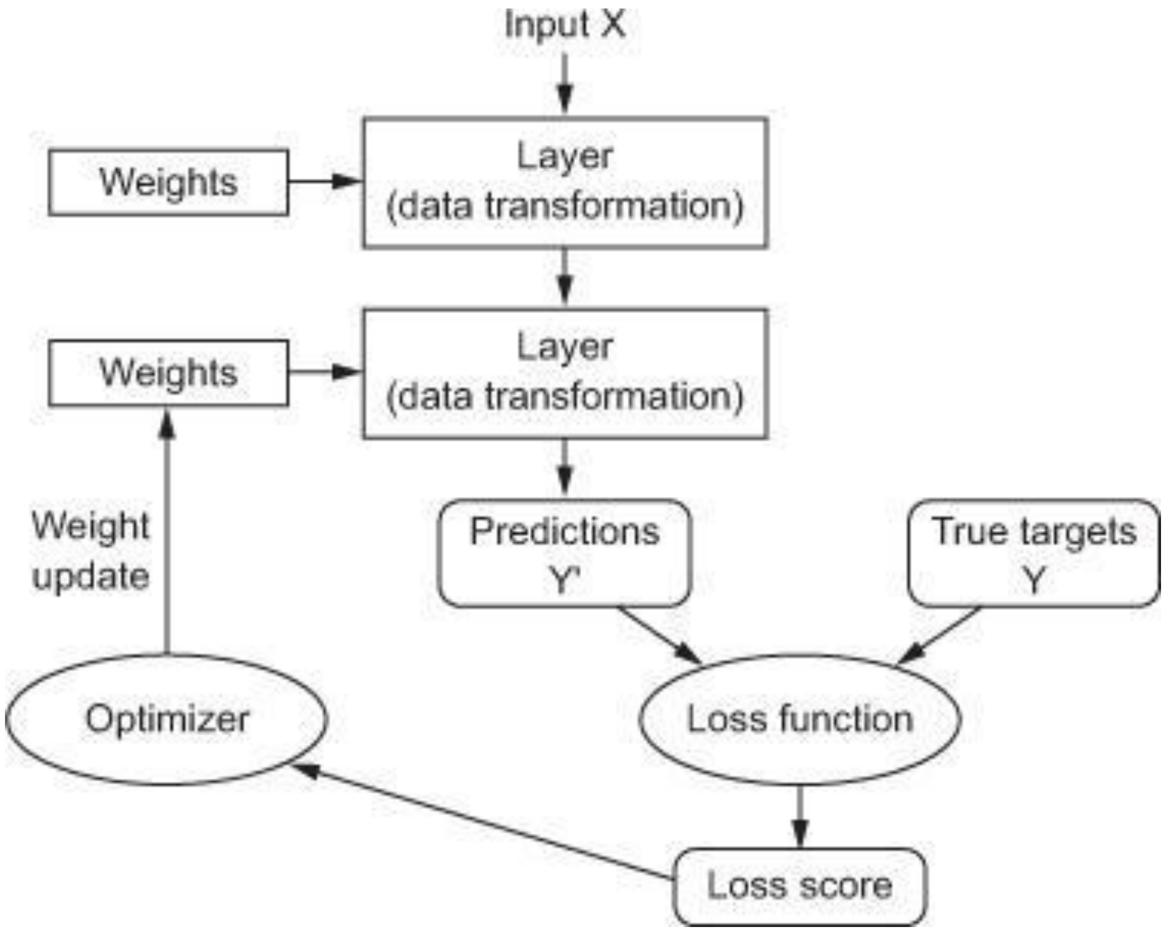
- Models with lots of parameters can easily overfit to training data
- Generalization: the quality of ML model is measured on new, unseen samples
- Regularization: any method* to prevent overfitting
  - simplicity, sparsity, dropout, early stopping
  - *other than adding more data



By Chabacano [GFDL or CC BY-SA 4.0], from Wikimedia Commons

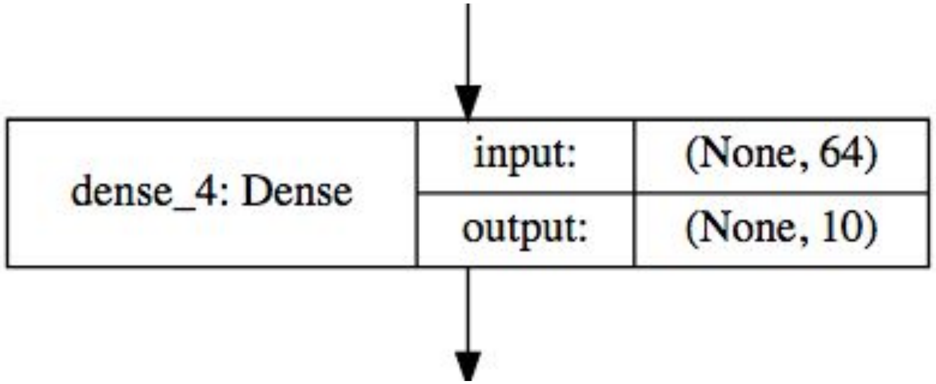
Anatomy of a deep neural network

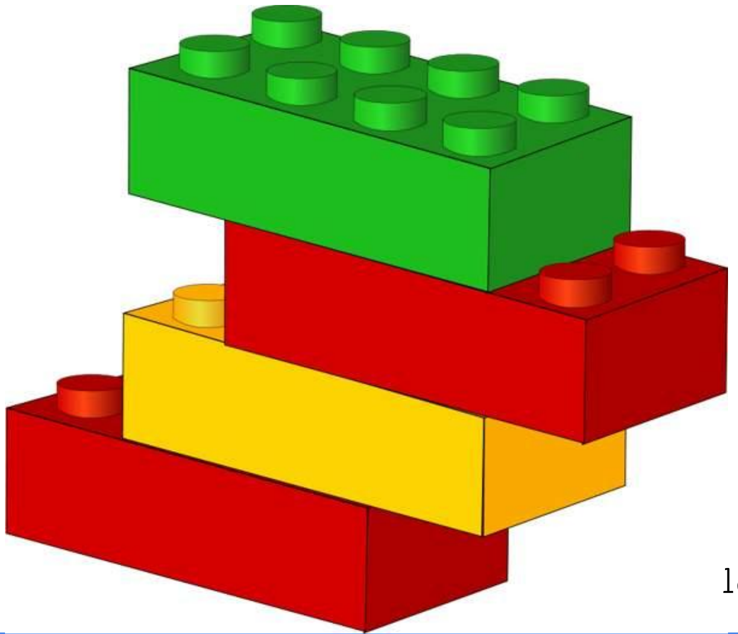
- Layers
- Input data and targets
- Loss function
- Optimizer



Layers

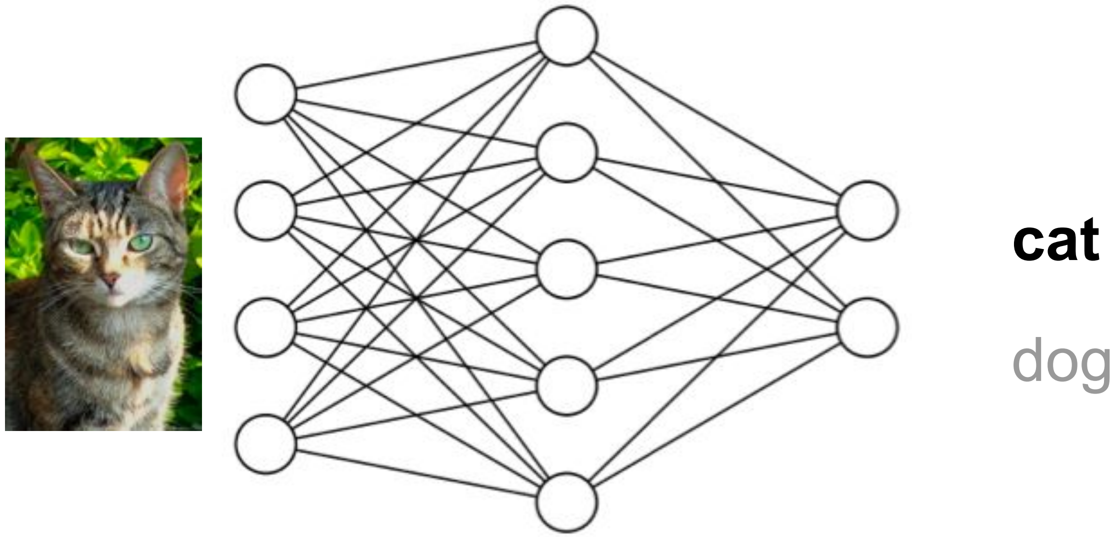
- Data processing modules
- Many different kinds exist
  - densely connected
  - convolutional
  - recurrent
  - pooling, flattening, merging, normalization, etc.
- Input: one or more tensors
- output: one or more tensors
- Usually have a state, encoded as weights
  - learned, initially random
  - When combined, form a network or a model

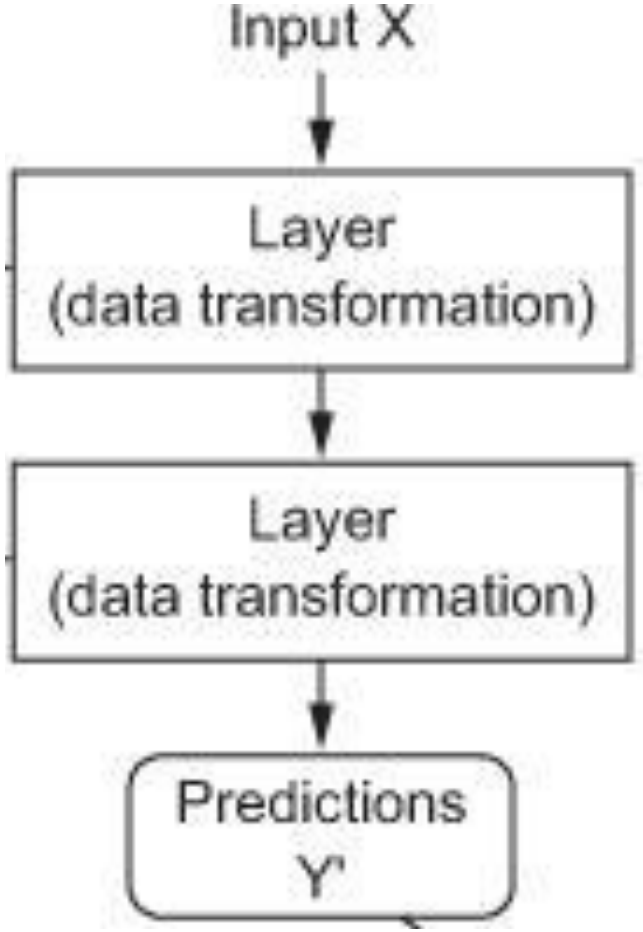




Input data and targets

- The network maps the input data X to predictions Y′
- During training, the predictions Y′ are compared to true targets Y using the loss function



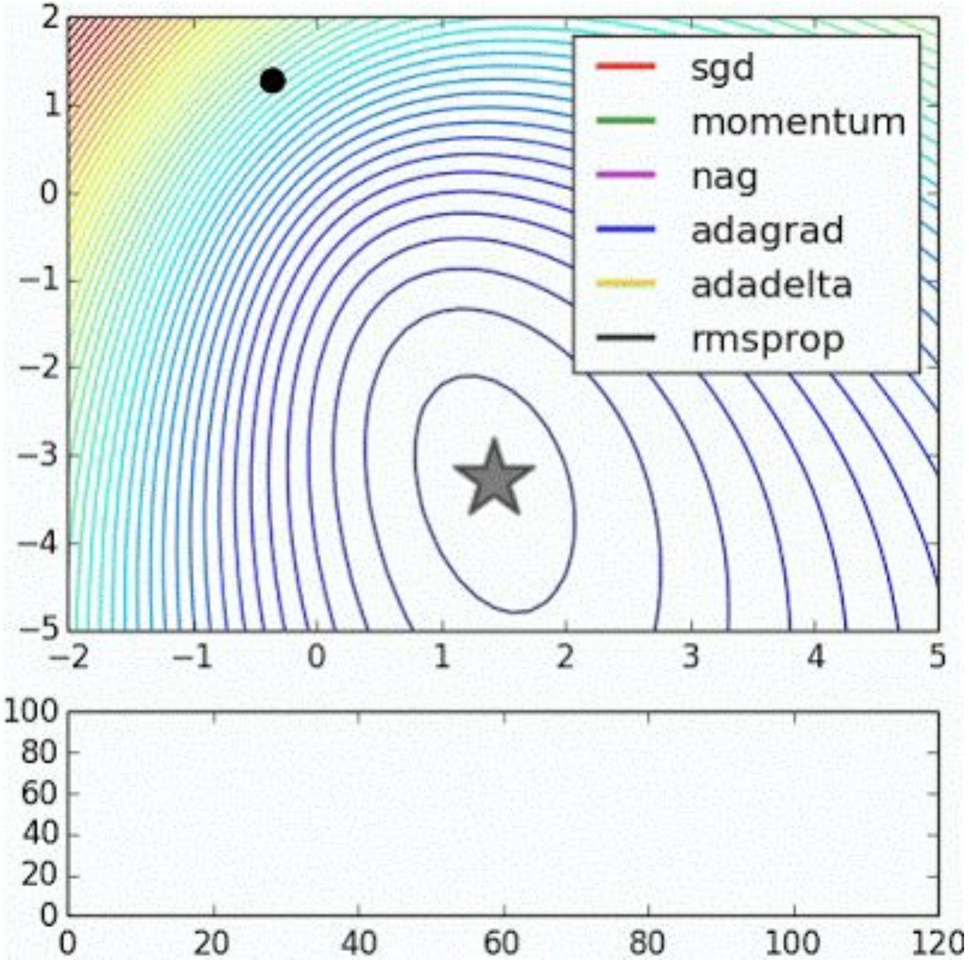


Loss function

- The quantity to be minimized (optimized) during training
    - the only thing **the network** cares about
    - there might also be other metrics **you** care about
- Common tasks have “standard” loss functions:
    - *mean squared error* for regression
    - *binary cross-entropy* for two-class classification
    - *categorical cross-entropy* for multi-class classification
    - etc.
- https://lossfunctions.tumblr.com/

Optimizer

- How to update the weights based on the loss function
- *Learning rate (+scheduling)*
- Stochastic gradient descent, momentum, and their variants
  - RMSProp is usually a good first choice
  - more info: http://ruder.io/optimizing-gradient-descent/



Animation from: https://imgur.com/s25RsOr

Deep learning frameworks

Caffe

torch

Chainer

PyTorch

DEEPLEARNING4J

Caffe2

The Microsoft Cognitive Toolkit

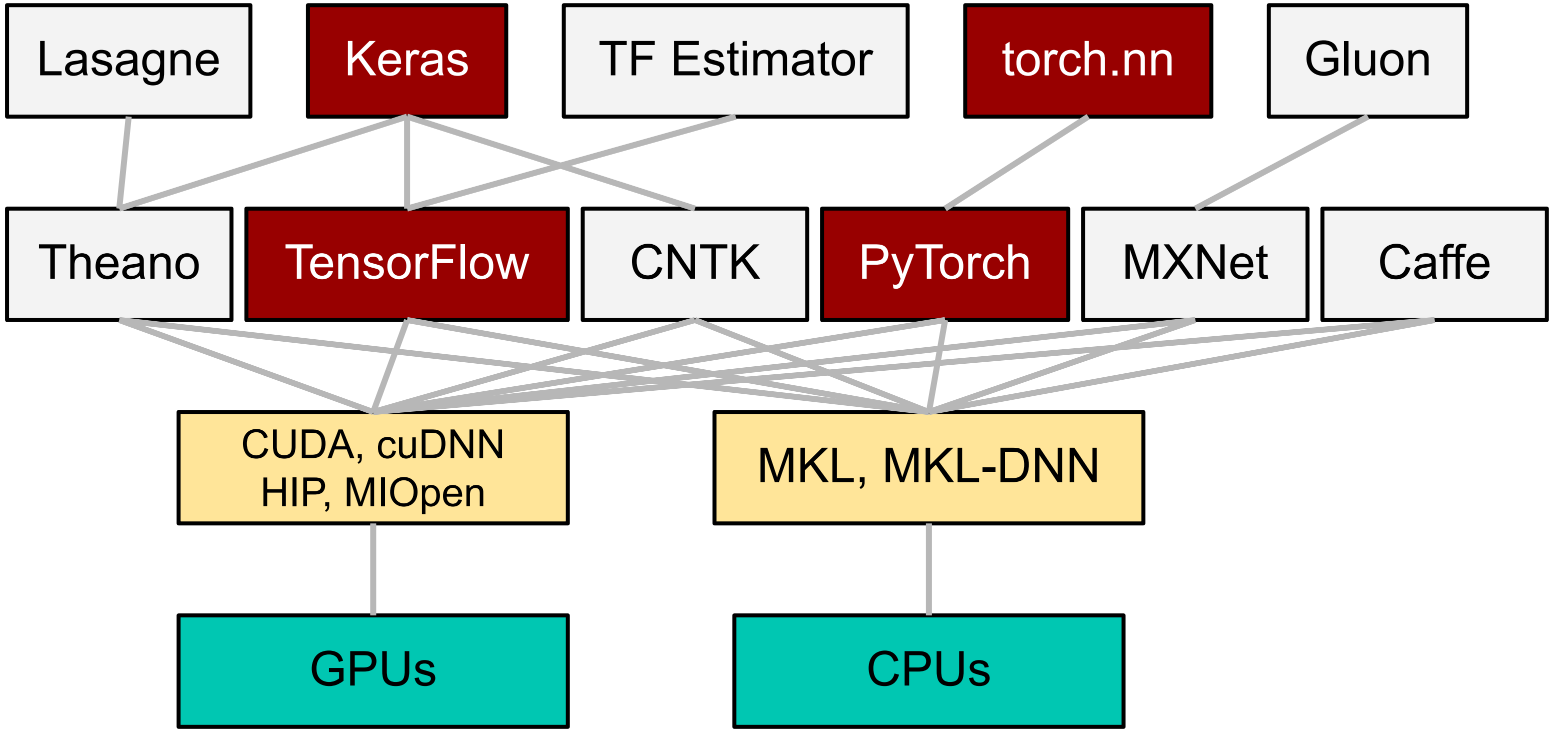
TensorFlow

K

dmlc mxnet

theano

Deep learning frameworks



- Keras is a high-level neural networks API
  - included in TensorFlow 2 as `tf.keras`
  - https://keras.io/ , https://www.tensorflow.org/guide/keras
- PyTorch is:
  - a GPU-based tensor library
  - an efficient library for dynamic neural networks
  - https://pytorch.org/

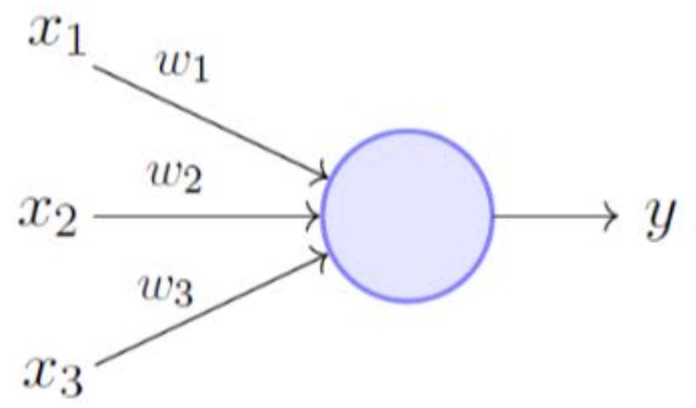
Important Concepts Used In Artificial Neural Network (ANN)

Perceptron

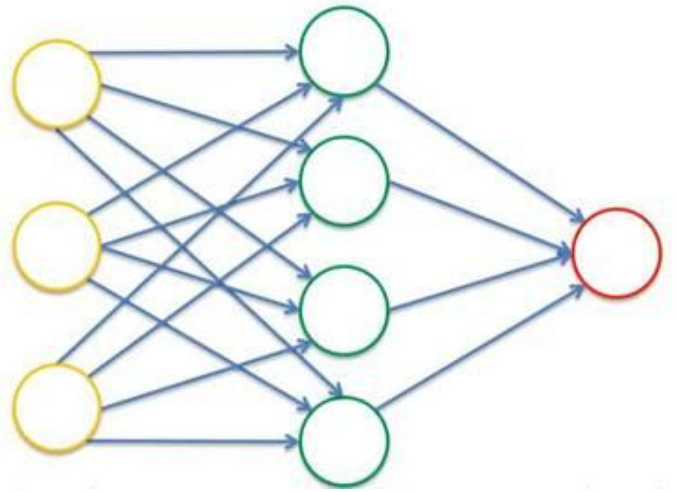
A perceptron is known as a single neuron model that is the basic building block to larger neural networks. Neurons

Neuron

Each neuron may have one or many inputs. Likewise, one neuron may yield single or multiple outputs to multiple neurons.



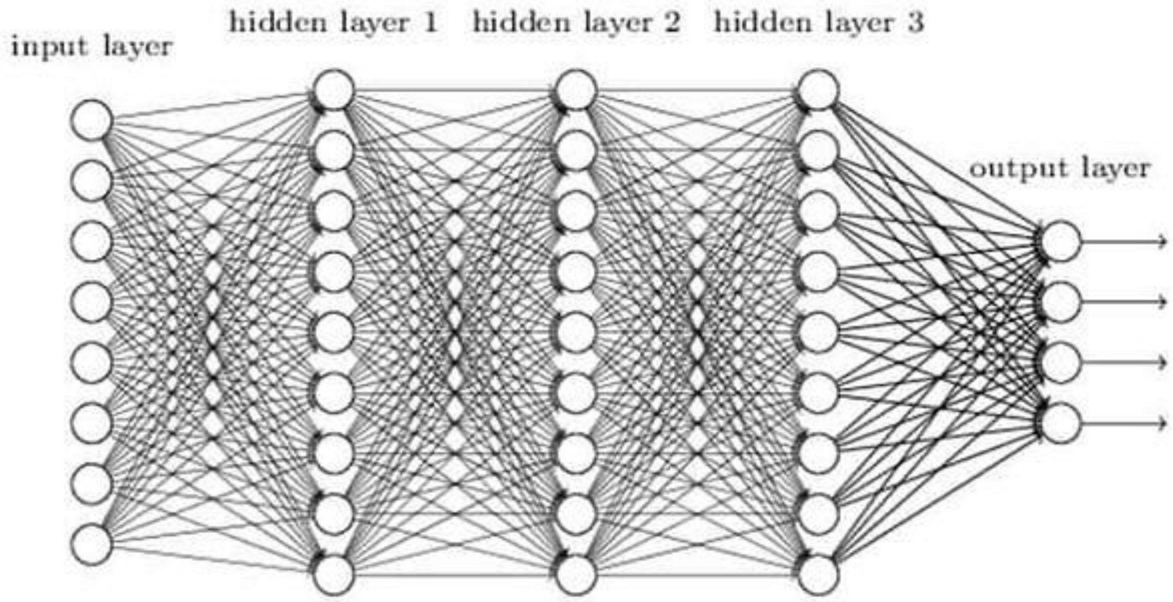
Perceptron Model (Minsky-Papert in 1969)



Important Concepts Used In Artificial Neural Network (ANN)

Synapse

ANN is made up of connections. These connections are more commonly known as **weights** or synapse. **Weights** have an important role, as they are used for a neural network to **learn**. Weights are supposed to **adjust** or **pass** the signal to the next neurons.



Important Concepts Used In Artificial Neural Network (ANN)

Hidden layer

The hidden layer is a layer of neuron in ANN which is present between the input and output layer. The input layer receives the data.

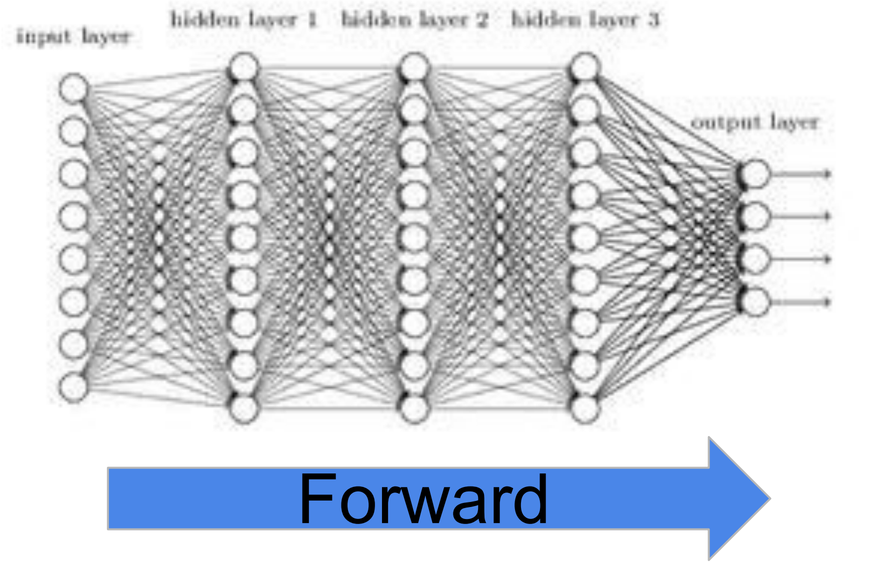
Hyperparameter

To run ANN, several constant parameters are set before the learning process begins. Some examples of hyperparameters are a number of hidden layers, epoch, batch size, optimization, etc.

Important Concepts Used In Artificial Neural Network (ANN)

Forward propagation

In this step, the neural network is fired in the forward direction, from left to right. Input layers move to the next hidden layer based on weight and this step moves forward as per the activation function, until getting the predicted results y.



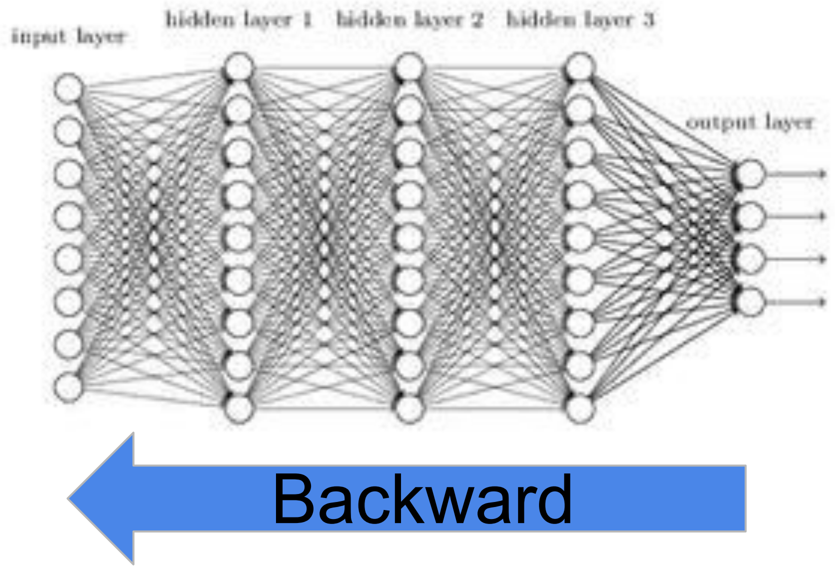
Cost Function

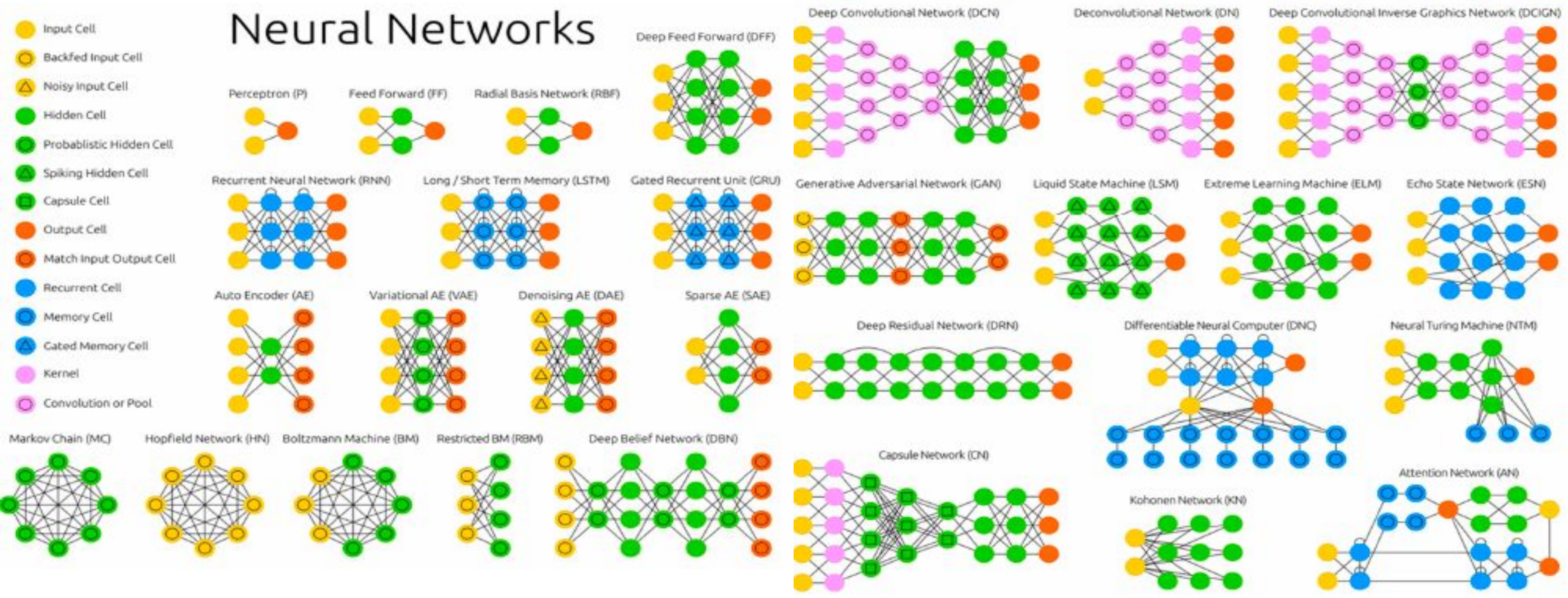
Cost function and loss function refers to nearly the same. It basically compares the predicted results to the actual result and measures the generated error.

Important Concepts Used In Artificial Neural Network (ANN)

Backward Propagation

Moves from right to left. This is an algorithm that uses gradient descent to calculate the gradients of the error function.




Neural Networks
Input Cell
Backfed Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Capsule Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Gated Memory Cell
Kernel
Convolution or Pool
Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Deep Feed Forward (DFF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)
Markov Chain (MC)
Hopfield Network (HN)
Boltzmann Machine (BM)
Restricted BM (RBM)
Deep Belief Network (DBN)
Deep Convolutional Network (DCN)
Deconvolutional Network (DN)
Deep Convolutional Inverse Graphics Network (DCIGN)
Generative Adversarial Network (GAN)
Liquid State Machine (LSM)
Extreme Learning Machine (ELM)
Echo State Network (ESN)
Deep Residual Network (DRN)
Differentiable Neural Computer (DNC)
Neural Turing Machine (NTM)
Capsule Network (CN)
Kohonen Network (KN)
Attention Network (AN)

Data Preparation

According to a survey in Forbes, data scientists spend 80% of their time on data preparation: