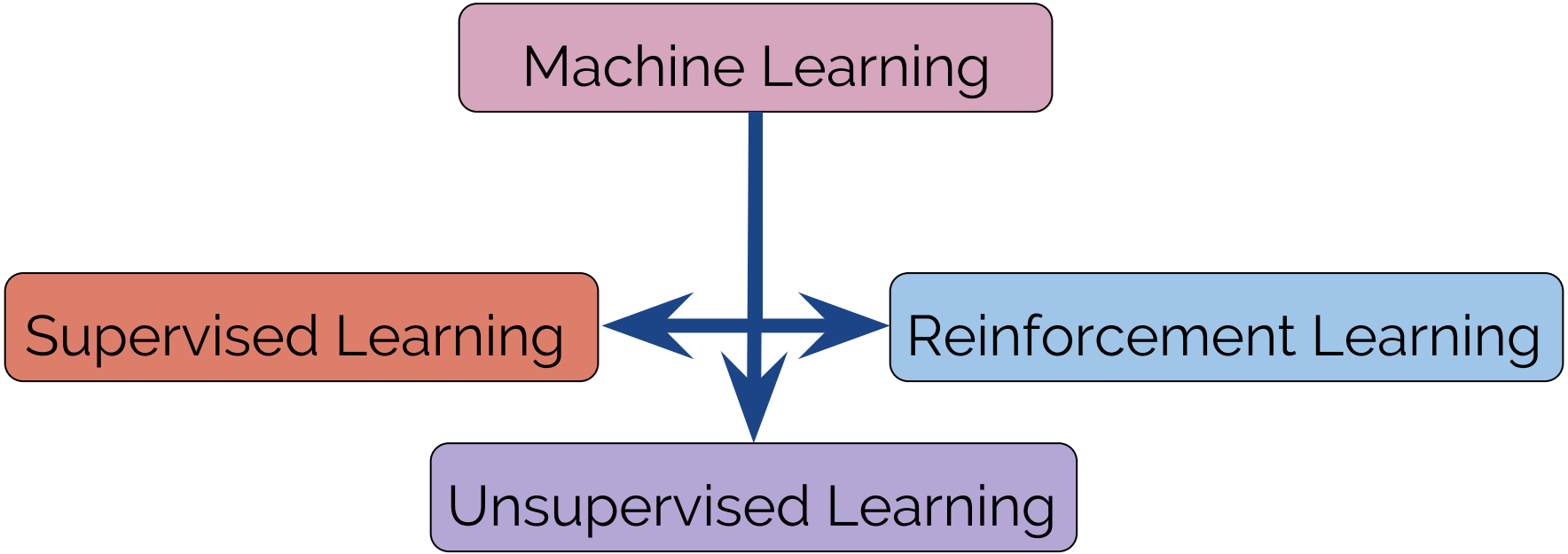
#TMLSS 2018
Introduction to ML/DL

Razvan Pascanu
razp@google.com

Lecture overview

Managing expectations:

- It will be kept relatively at an intuitive level
- Context for following lectures
- Somewhat condensed, lots of ground to cover
- Will go from very basic stuff to more complex topics, possibly quite quickly
- Interrupt me if you have questions


Machine Learning
Supervised Learning
Reinforcement Learning
Unsupervised Learning

Supervised Learning

- Relies on a **dataset**

  $\langle x_1, y_1 \rangle, \langle x_2, y_2 \rangle, \ldots \langle x_n, y_n \rangle$

- Underlying assumption of a ***teacher*** who knows ground truth
- The goal is to learn a **function f** that maps from x_i to y_i
- This is done by minimizing the distance between $f(x_i)$ and y_i for all **i**

Supervised Learning

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

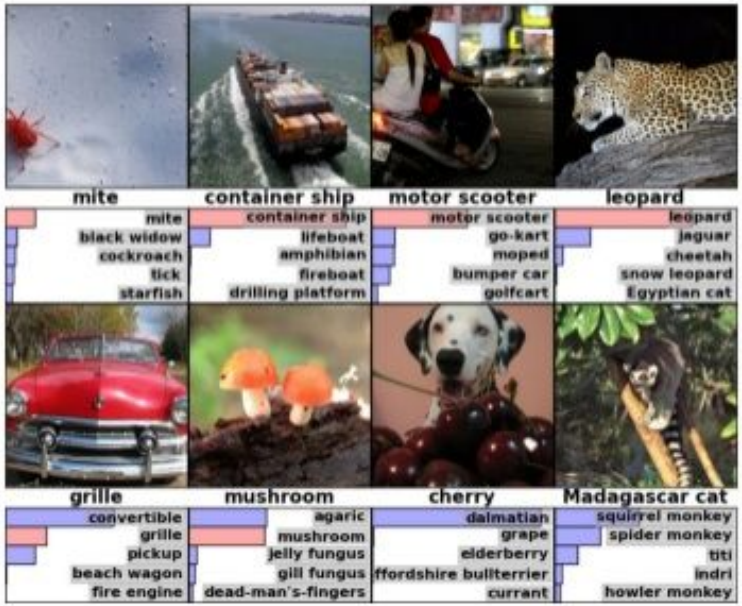
https://arxiv.org/pdf/1502.03044.pdf

Supervised Learning

ImageNet Challenge

IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.



Unsupervised Learning

- Relies on an **unlabelled dataset**

  $x_1, x_2, \dots x_n$

- The goal is to discover **structure** in data. For example:
  - Clustering
  - Density estimation
  - Dimensionality reduction

Unsupervised Learning

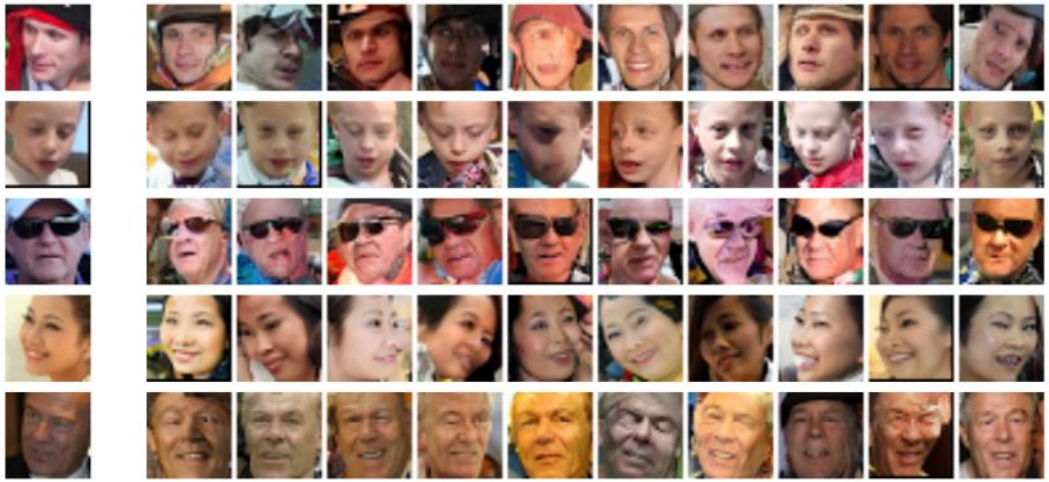
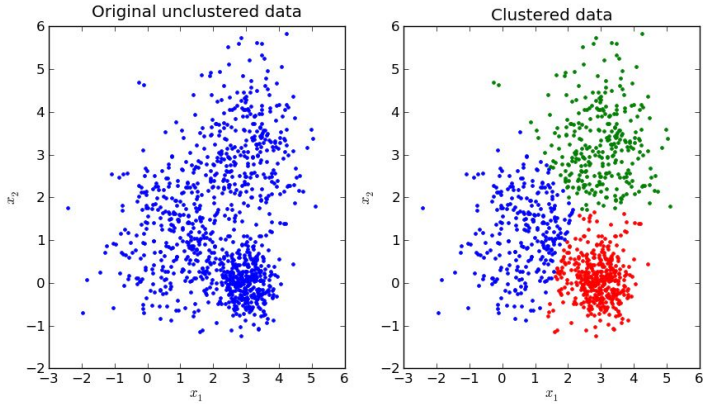
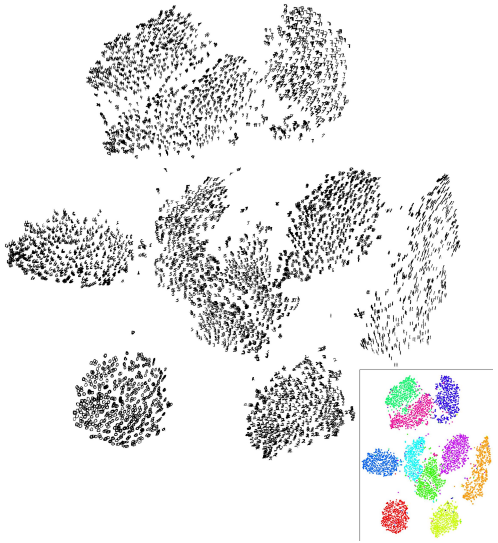


Figure 4: **Left**: source image. **Right**: new portraits generated from high-level latent representation.

https://arxiv.org/pdf/1606.05328.pdf

Unsupervised Learning



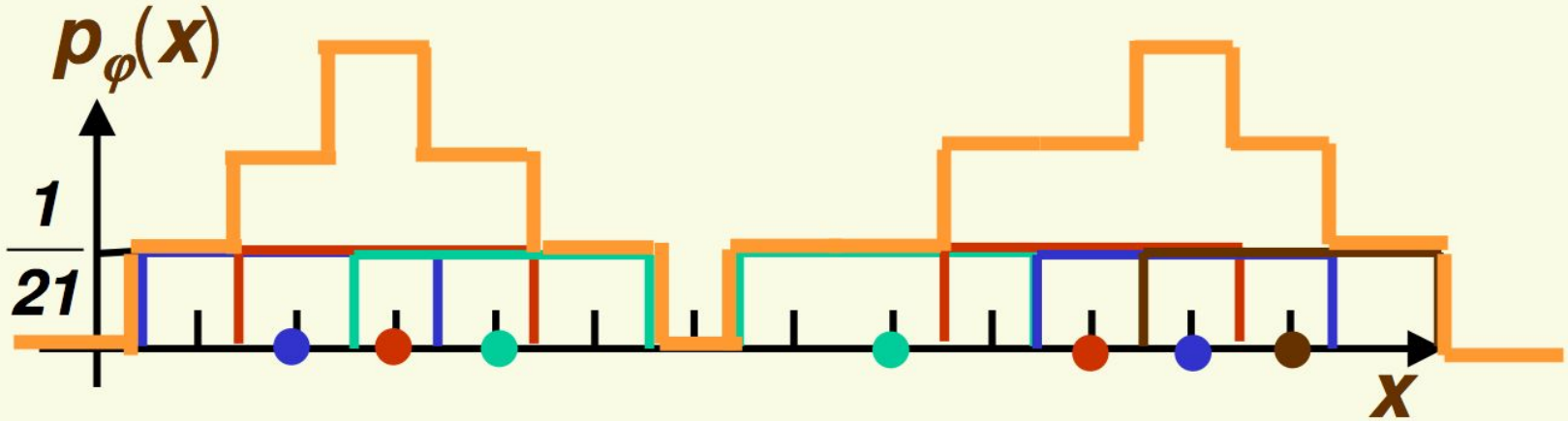


http://pypr.sourceforge.net/kmeans.html#k-means-example

https://lvdmaaten.github.io/tsne/#examples

Case study: parzen windows

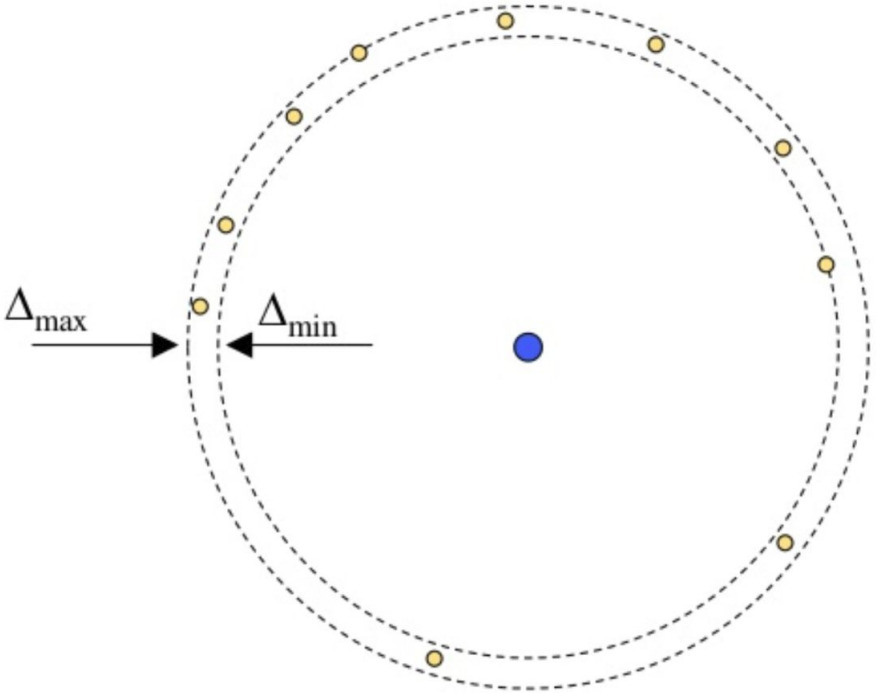
7 samples D={2,3,4,8,10,11,12}, h=3



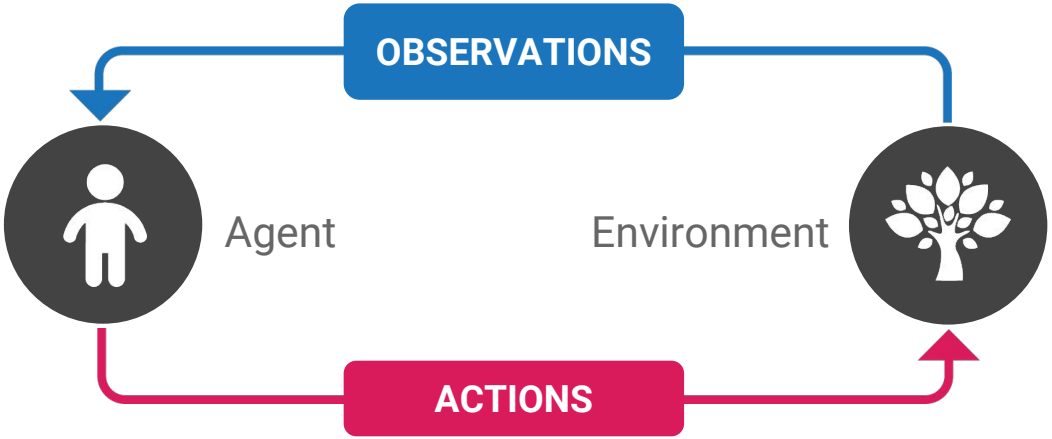
http://jxieeducation.com/2016-10-06/Nonparametric-Density-Estimation-Parzen-Windows-And-Beyond/

Curse of dimensionality

- High dimension is hard to find neighbours within the window
- Volume concentrates on the surface of the sphere

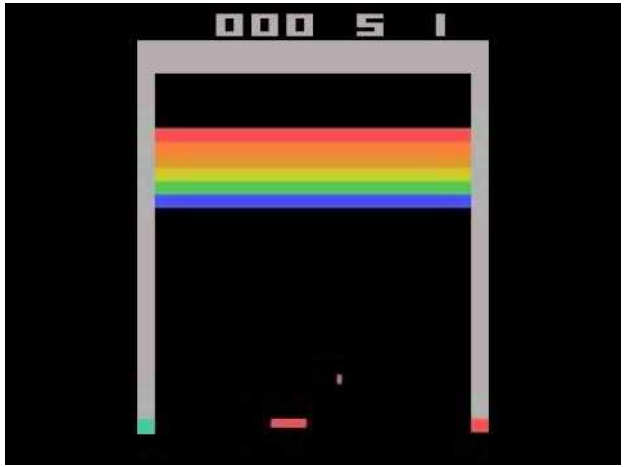


Reinforcement Learning

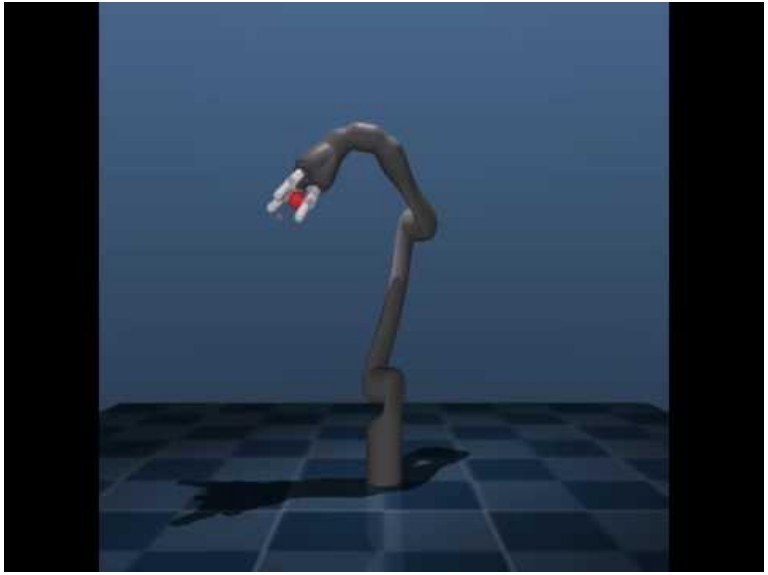


- General Purpose Framework for AI
- Agent interacts with the environment
- Select actions to maximise reward

Reinforcement Learning



Reinforcement Learning

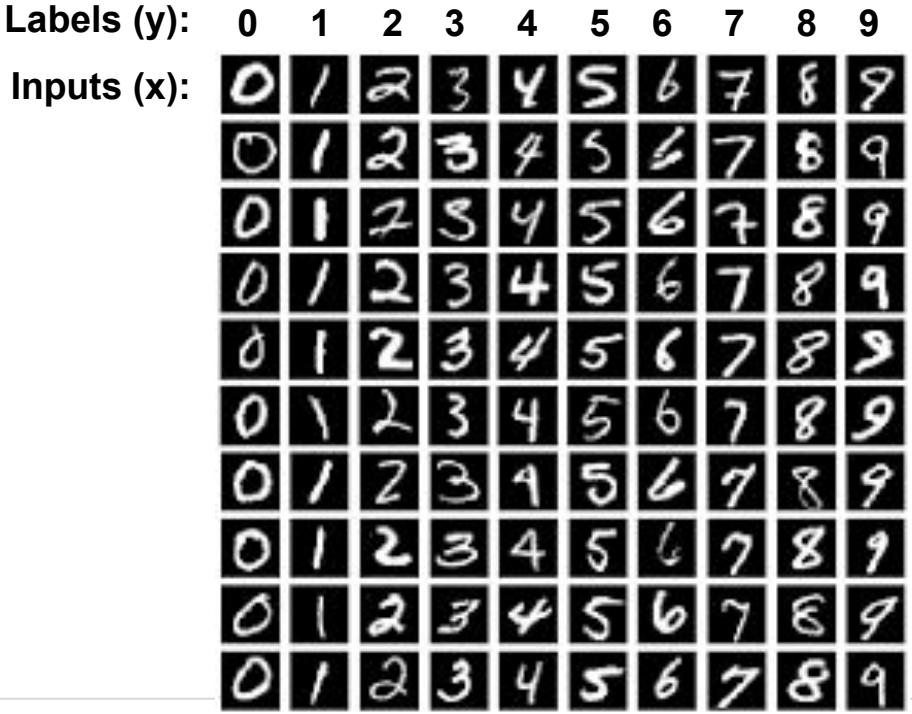


More on Supervised Learning

Example of a dataset

Terminology:

- labels/targets
- Output/predictions
- input/input features
- Instance or example
- datasets



More on Supervised Learning

- Let $\mathcal{X}$ denote the space of input values
- Let $\mathcal{Y}$ denote the space of output values
- Given a data set $D \subset \mathcal{X} \times \mathcal{Y}$, find a function:

$$h : \mathcal{X} \to \mathcal{Y}$$

  such that $h(\mathbf{x})$ is a *"good predictor"* for the value of y.
- h is called a *hypothesis*
- Problems are categorized by the type of output domain
  - If $\mathcal{Y} = \mathbb{R}$, this problem is called *regression*
  - If $\mathcal{Y}$ is a categorical variable (i.e., part of a finite discrete set), the problem is called *classification*
  - In general, $\mathcal{Y}$ could be a lot more complex (graph, tree, etc), which is called *structured prediction*

Parametrized functions

A parameterized function is a function:

$$h : \theta \times \mathcal{X} \rightarrow \mathcal{Y}$$

for example a linear function of the form

$$h(w, x) = wx$$

Learning then boils down to finding *the best* θ to minimize the distance between prediction and targets

$$\arg\min_{\theta} L(\theta) = \arg\min_{\theta} \mathbb{E}\left[dist(h(\theta, x_i), y_i)\right]$$

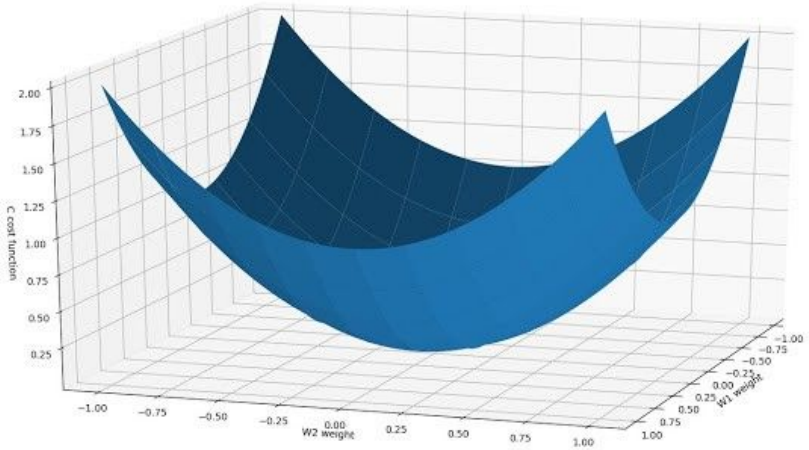
Loss functions

- Matching loss functions:
  - E.g. regression with mean square error
  - E.g. classification with negative log likelihood

Test case: Mean Square Error (MSE)

$$\arg\min_{\theta} L(\theta) = \arg\min_{\theta} \frac{1}{N} \sum_{1}^{N} \left[(h(\theta, x_i) - y_i)^2 \right]$$

Mean Square Error

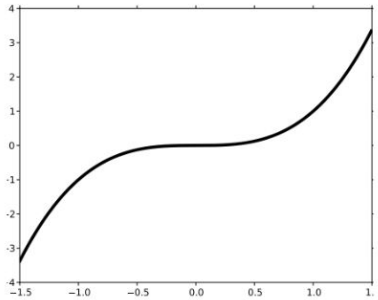


Finding the right parameters

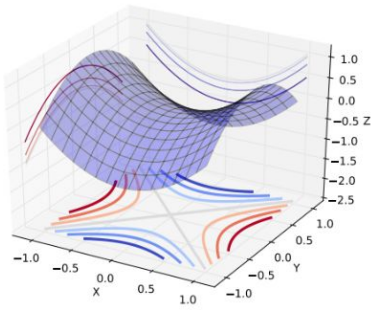
- One approach is following the gradient (gradient descent)

$$\min_{\Delta\theta} L(\theta + \Delta\theta) \approx L(\theta) + \Delta\theta \frac{\partial L}{\partial \theta}$$
$$\text{s.t. } |\Delta\theta| < \epsilon$$

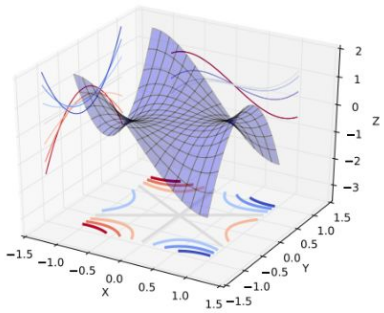
Possible pathological structures



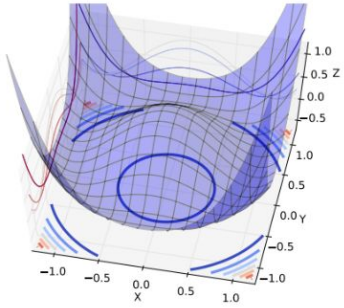
(a)



(b)



(c)

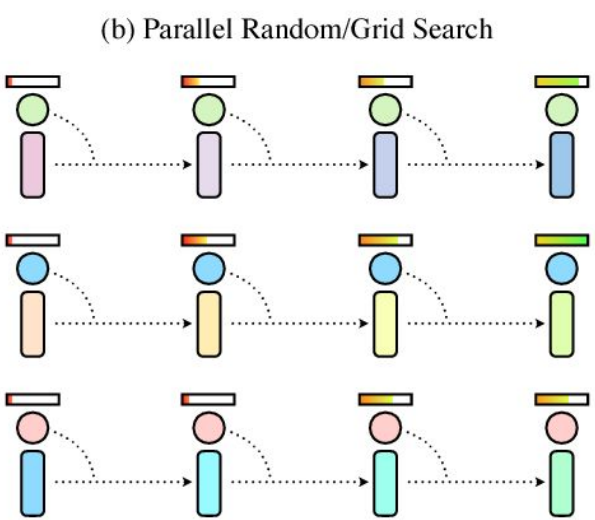


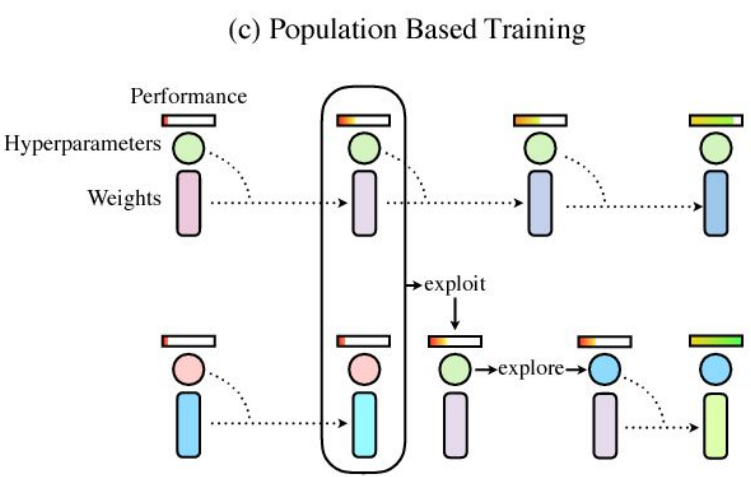
(d)

https://arxiv.org/pdf/1406.2572.pdf

Finding the right parameters

- Alternative learning scheme exists:
  - Random search
  - Evolution





https://arxiv.org/abs/1711.09846

Probabilistic interpretation of MSE

Slides borrowed from Doina

- Assume y_i is a noisy target value, generated from a hypothesis $h_{\mathbf{w}}(\mathbf{x})$
- More specifically, assume that there exists $\mathbf{w}$ such that:

$$y_i = h_{\mathbf{w}}(\mathbf{x}_i) + \epsilon_i$$

where ϵ_i is random variable (noise) drawn independently for each $\mathbf{x}_i$ according to some Gaussian (normal) distribution with mean zero and variance σ.

- How should we choose the parameter vector $\mathbf{w}$?

Bayes theorem in learning

Let h be a hypothesis and D be the set of training data.
Using Bayes theorem, we have:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)},$$

where:

- $P(h)$ is the *prior probability of hypothesis h*
- $P(D) = \int_h P(D|h)P(h)$ is the probability of training data D (normalization, independent of h)
- $P(h|D)$ is the probability of h given D
- $P(D|h)$ is the probability of D given h (*likelihood of the data*)

Choosing hypothesis

- What is the most probable hypothesis given the training data?
- *Maximum a posteriori (MAP)* hypothesis h_{MAP}:

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in \mathcal{H}} P(h|D) \\ &= \arg\max_{h \in \mathcal{H}} \frac{P(D|h)P(h)}{P(D)} \text{(using Bayes theorem)} \\ &= \arg\max_{h \in \mathcal{H}} P(D|h)P(h) \end{aligned}$$

Last step is because $P(D)$ is independent of h (so constant for the maximization)

- This is the Bayesian answer (more in a minute)

Maximum Likelihood estimation

$$h_{MAP} = \arg\max_{h \in \mathcal{H}} P(D|h)P(h)$$

- If we assume $P(h_i) = P(h_j)$ (all hypotheses are equally likely a priori) then we can further simplify, and choose the *maximum likelihood (ML) hypothesis*:

$$h_{ML} = \arg\max_{h \in \mathcal{H}} P(D|h) = \arg\max_{h \in \mathcal{H}} L(h)$$

- Standard assumption: the training examples are *independently identically distributed (i.i.d.)*
- This alows us to simplify $P(D|h)$:

$$P(D|h) = \prod_{i=1}^{m} P(\langle \mathbf{x_i}, y_i \rangle | h) = \prod_{i=1}^{m} P(y_i|\mathbf{x}_i; h)P(\mathbf{x}_i)$$

The **log** trick

- We want to maximize:

$$L(h) = \prod_{i=1}^{m} P(y_i|\mathbf{x}_i; h)P(\mathbf{x}_i)$$

  This is a product, and products are hard to maximize!

- Instead, we will maximize $\log L(h)$! (the log-likelihood function)

$$\log L(h) = \sum_{i=1}^{m} \log P(y_i|\mathbf{x}_i; h) + \sum_{i=1}^{m} \log P(\mathbf{x}_i)$$

- The second sum depends on D, but not on h, so it can be ignored in the search for a good hypothesis

Maximum likelihood for regression

- Adopt the assumption that:

$$y_i = h_{\mathbf{w}}(\mathbf{x}_i) + \epsilon_i,$$

  where $\epsilon_i \sim \mathcal{N}(0, \sigma)$.
- The best hypothesis maximizes the likelihood of $y_i - h_{\mathbf{w}}(\mathbf{x}_i) = \epsilon_i$
- Hence,

$$L(\mathbf{w}) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{y_i - h_{\mathbf{w}}(\mathbf{x}_i)}{\sigma}\right)^2}$$

  because the noise variables ϵ_i are from a Gaussian distribution

Applying the **log** trick

$$\begin{aligned}\log L(\mathbf{w}) &= \sum_{i=1}^{m} \log \left(\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(y_i - h_{\mathbf{w}}(\mathbf{x}_i))^2}{\sigma^2}} \right) \\ &= \sum_{i=1}^{m} \log \left(\frac{1}{\sqrt{2\pi\sigma^2}} \right) - \sum_{i=1}^{m} \frac{1}{2} \frac{(y_i - h_{\mathbf{w}}(\mathbf{x}_i))^2}{\sigma^2}\end{aligned}$$

Maximizing the right hand side is the same as minimizing:

$$\sum_{i=1}^{m} \frac{1}{2} \frac{(y_i - h_{\mathbf{w}}(\mathbf{x}_i))^2}{\sigma^2}$$

This is our old friend, the sum-squared-error function! (the constants that are independent of h can again be ignored)

Maximum likelihood hypothesis

- Under the assumption that the training examples are i.i.d. and that we have *Gaussian target noise*, the maximum likelihood parameters $\mathbf{w}$ are those minimizing the sum squared error:

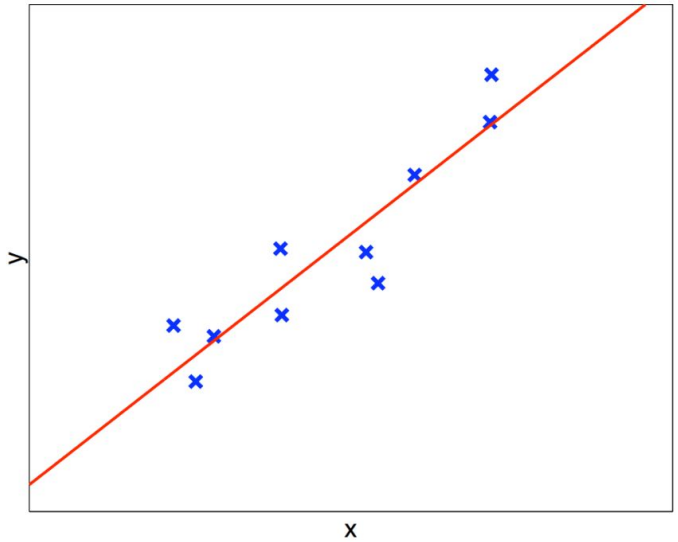
$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_{i=1}^{m} (y_i - h_{\mathbf{w}}(\mathbf{x_i}))^2$$

- This makes explicit the hypothesis behind minimizing the sum-squared error
- If the noise is not normally distributed, maximizing the likelihood will not be the same as minimizing the sum-squared error
- In practice, different loss functions are used depending on the noise assumption

Picking the right hypothesis class

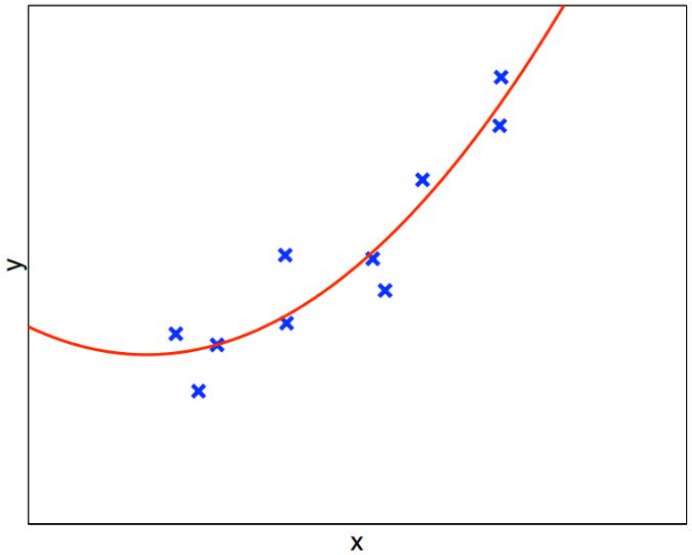
Example: Data and best linear hypothesis

$$y = 1.60x + 1.05$$



Picking the right hypothesis class

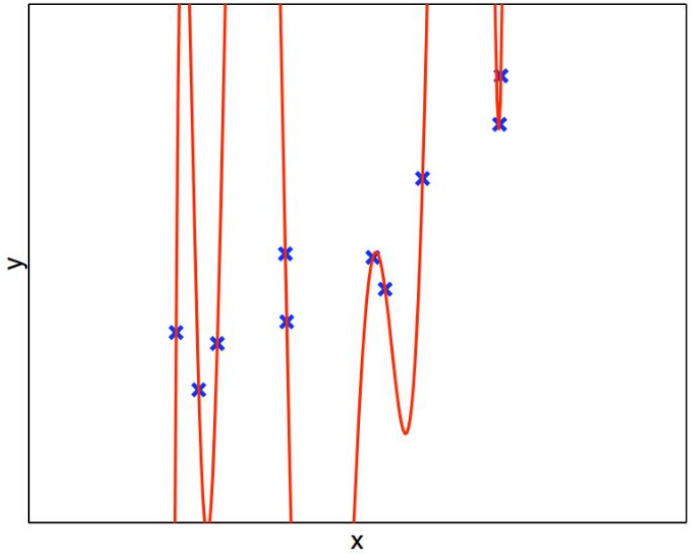
Order-2 fit



Is this a better fit to the data?

Picking the right hypothesis class

Order-9 fit



Is this a better fit to the data?

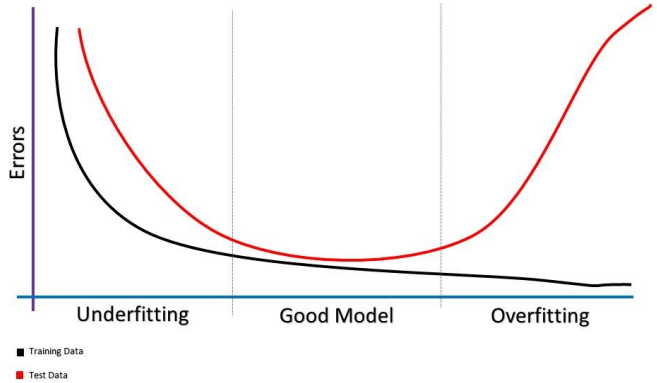
Overfitting

- *Very important* as we need the models **to generalize**

Training set: data used for finding the right parameters

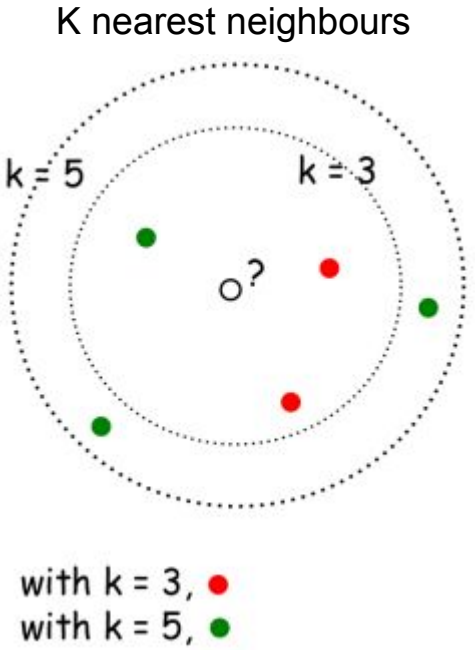
Validation set: data used to estimate true loss on unseen data

Learning is about minimizing an intractable function via optimizing a tractable approximation of it

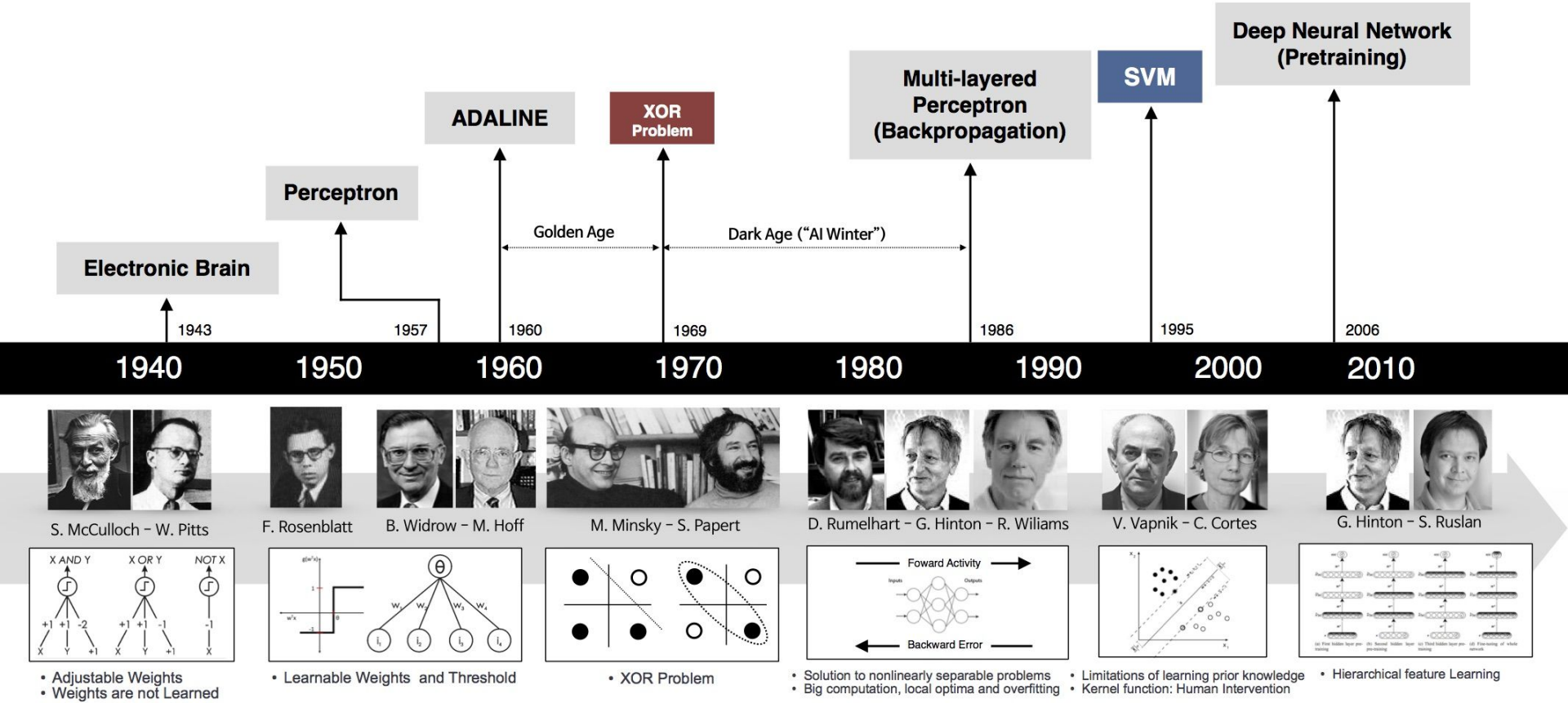


Non-parametric models and other things

- K nearest neighbours
- Bayesian nonparametrics
- Kernel methods
- Support Vector Machines
- Random Forests
- Self-organizing maps
- ...

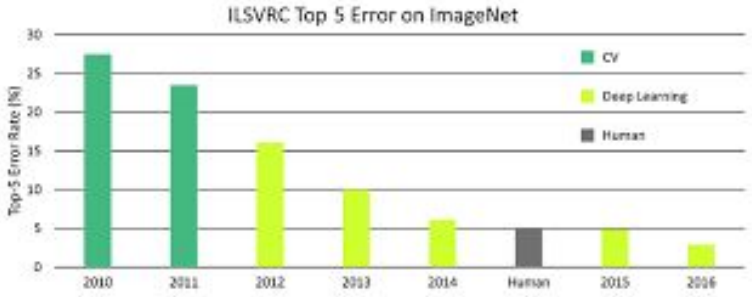


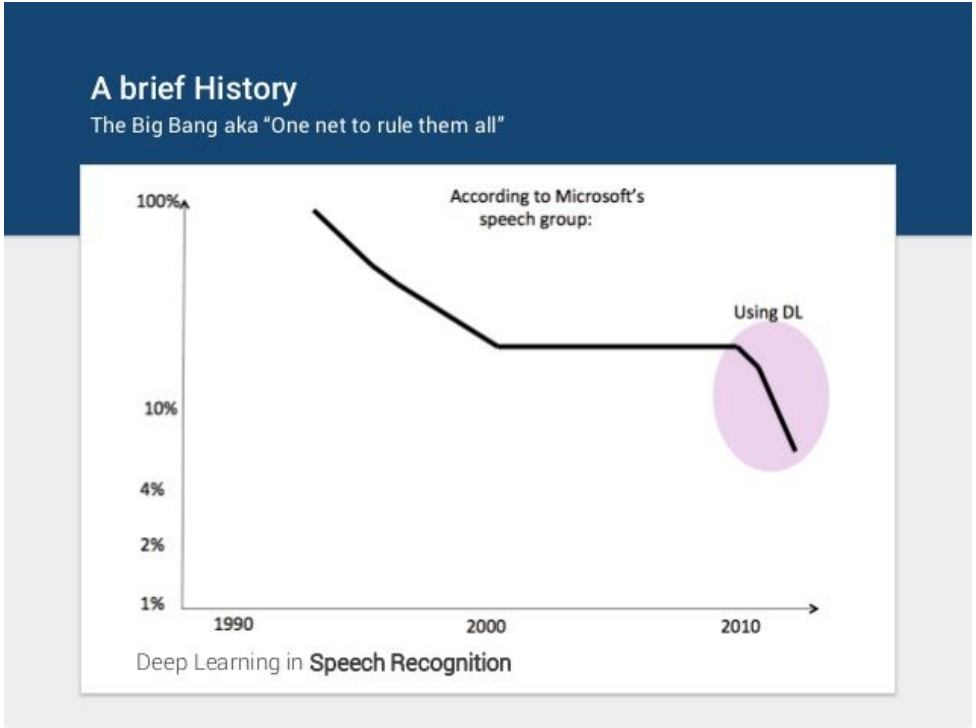
Neural Networks & Deep Learning



https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

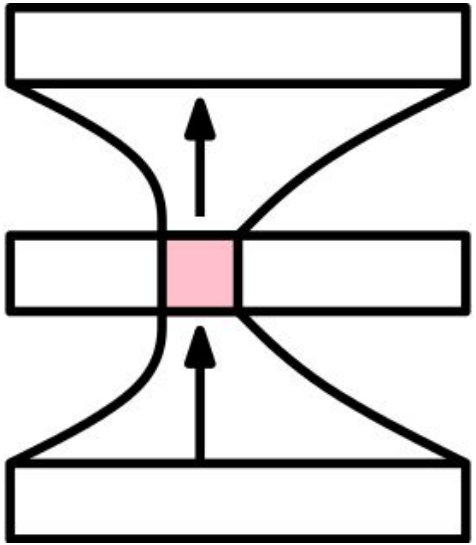
Some of DL's success



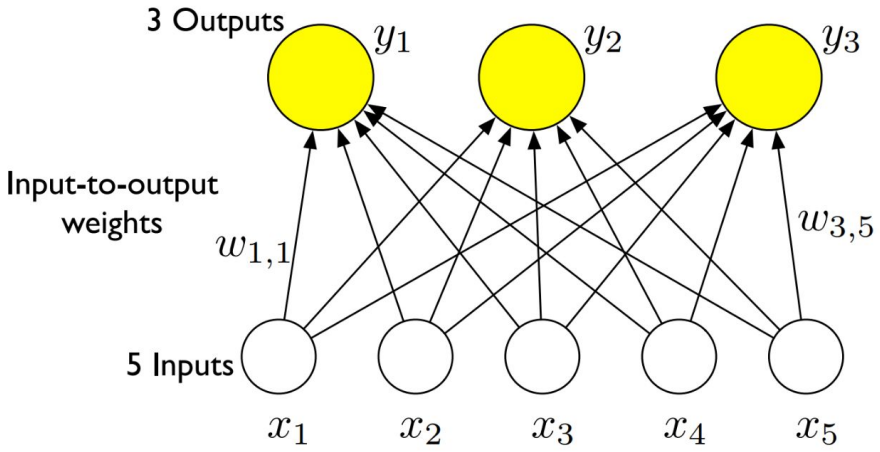


What is a neural network

$$\mathbf{y}(\mathbf{x}) = l_3(l_2(l_1(\mathbf{x})))$$



What is an MLP

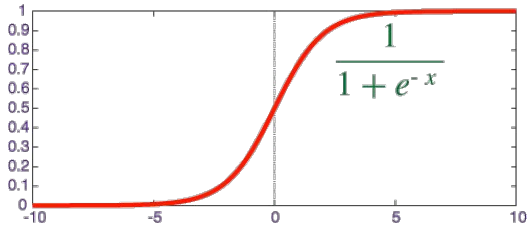


$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b} \qquad y_k = \sum_{i=1}^{d} W_{ki} x_i + b_k$$

Activation functions for MLP

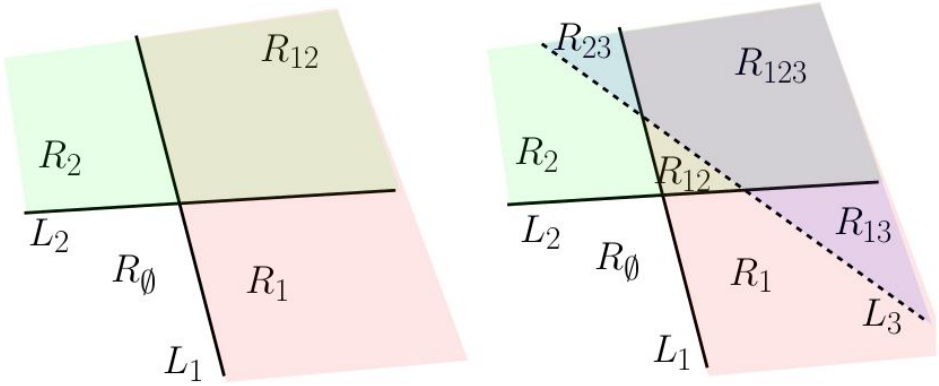
$$ReLU(x) = \begin{cases} x & x > 0 \\ 0 & \text{otherwise} \end{cases}$$

sigmoid

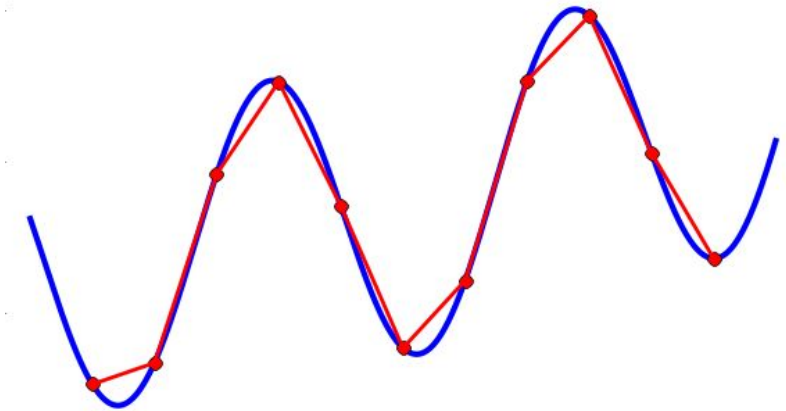


$$l_k(l_{k-1}) = \sigma(W_k l_{k-1} + b_k)$$

ReLU: Division of input space into regions



ReLU:piecewise linear approximation

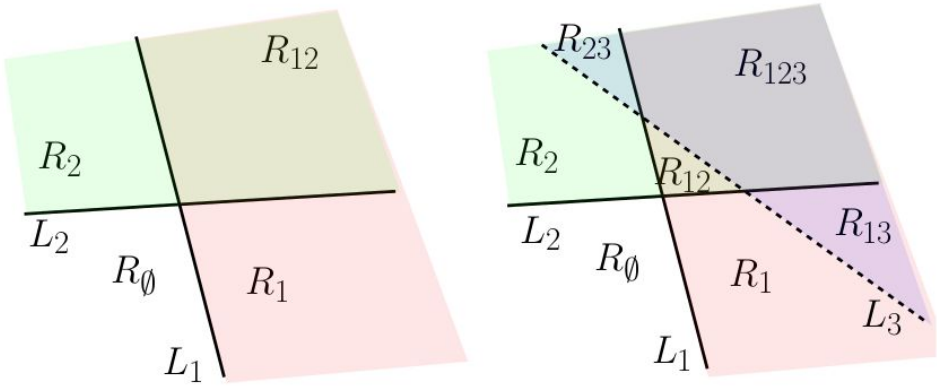


We know Neural Nets are universal approximators of any functions!

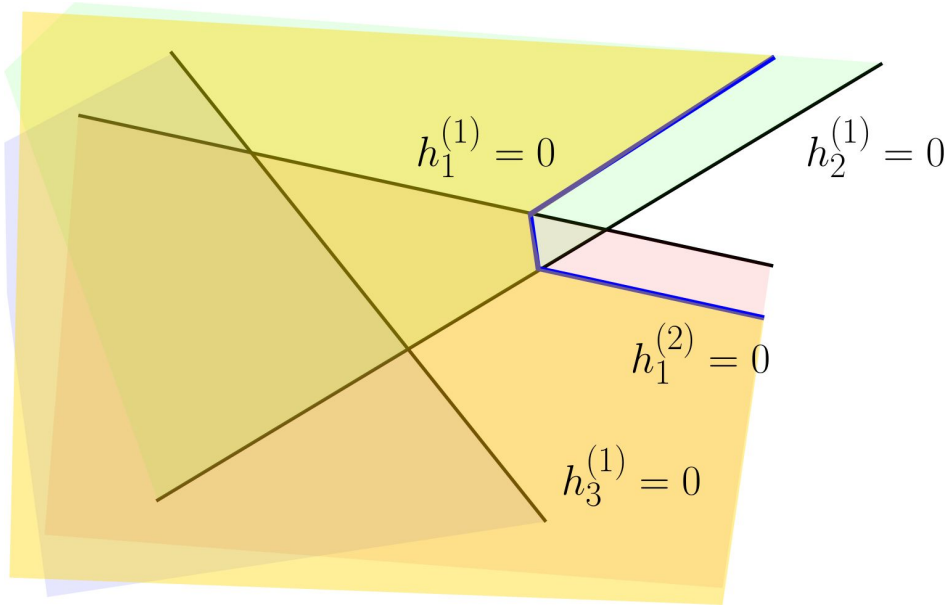
Understanding depth

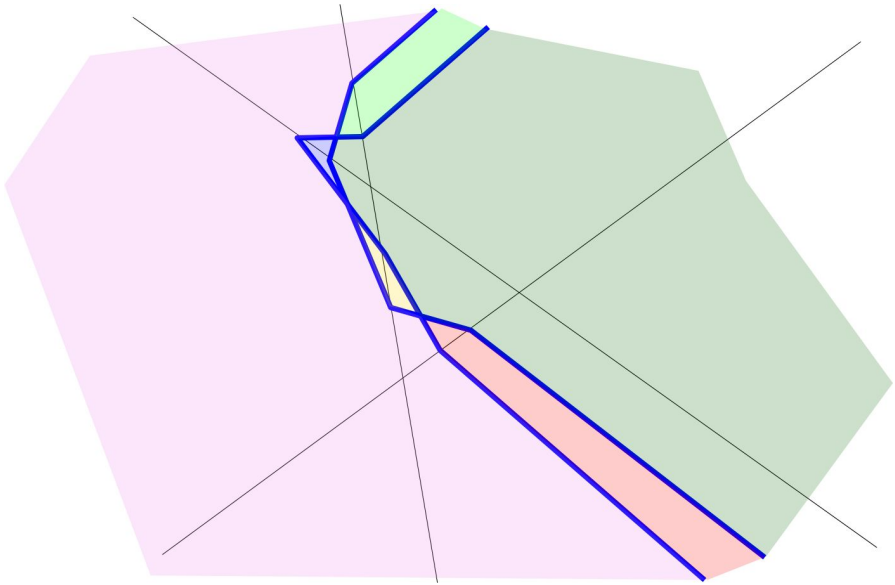
Zalasky theorem (1975):

$$r(\mathcal{A}_m) = \binom{m}{2} + m + 1.$$



Understanding depth





Understanding depth

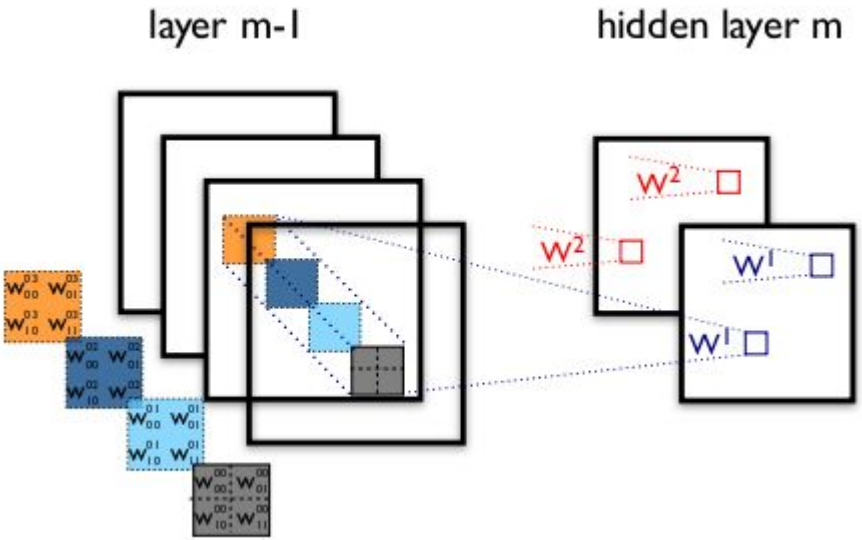
Other hypothesis:

- Optimization becomes easier (what about learning?)
- Higher control over variations between different linear regions?

Convolutional Neural Networks

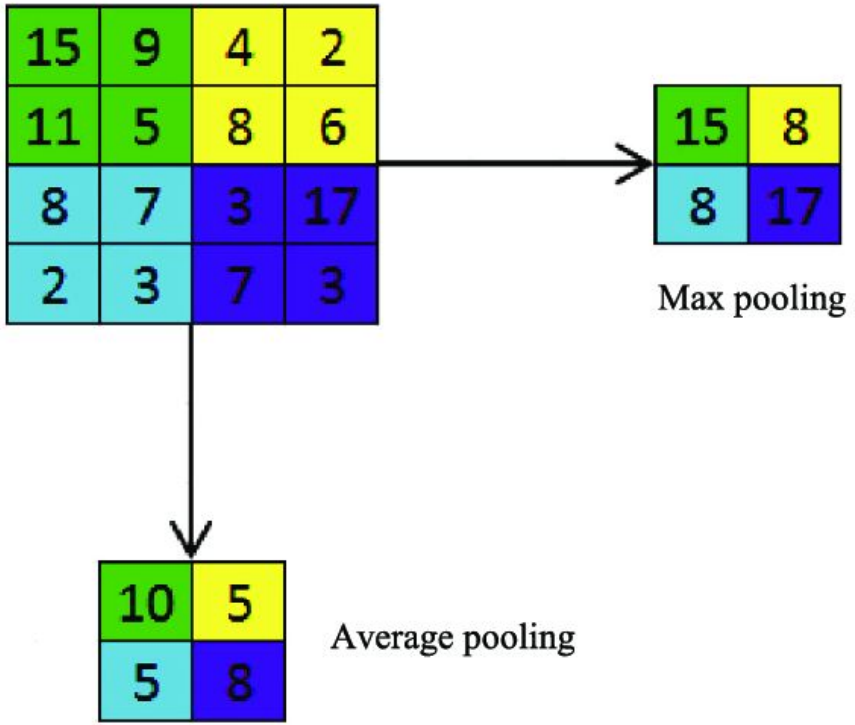
Convolutional Layer

- **Structural prior:** spatial neighbourhood defines the role of a pixel
- Apply same function at all position
- Induces translation invariance as features are computed independent of position

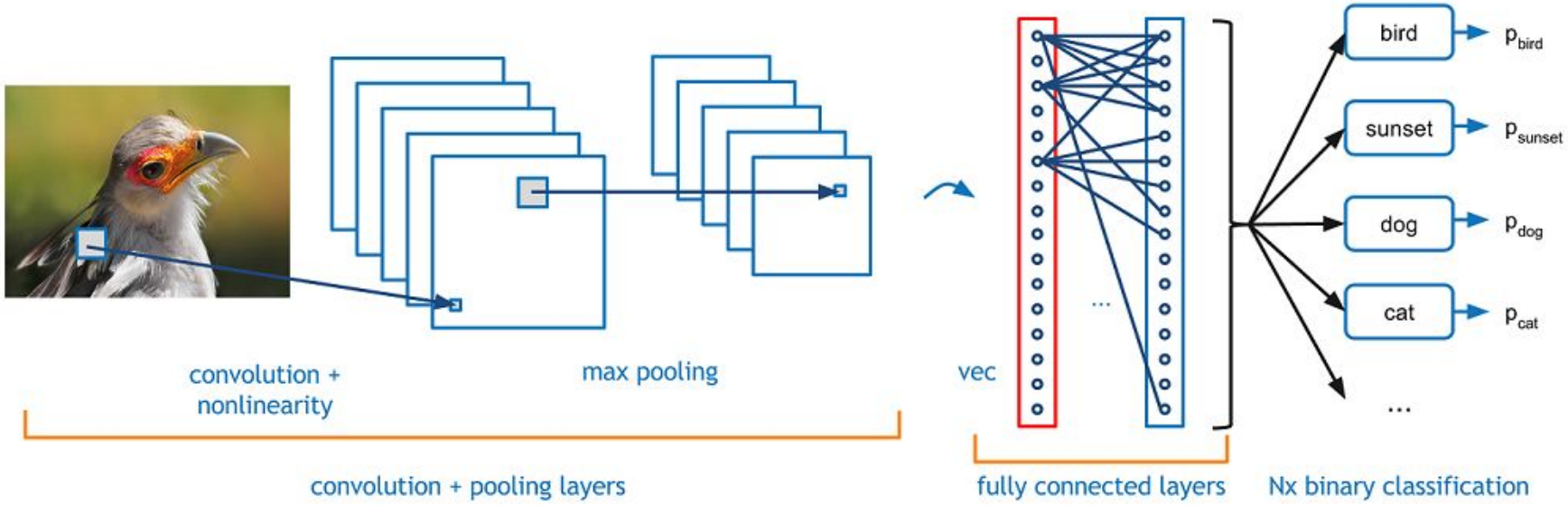


Convolutional Layer

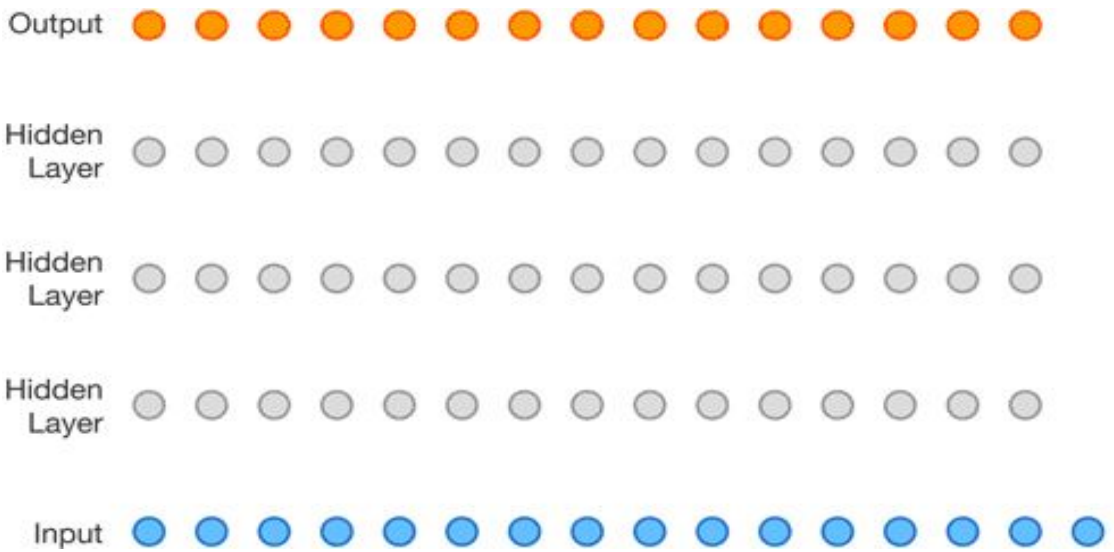
- Reduce dimensionality as you go up through the neural network
- Lose spatial resolution (robustness to translation)



Convolutional Network



Dilated convolutions



https://arxiv.org/abs/1711.10433

Residual connections

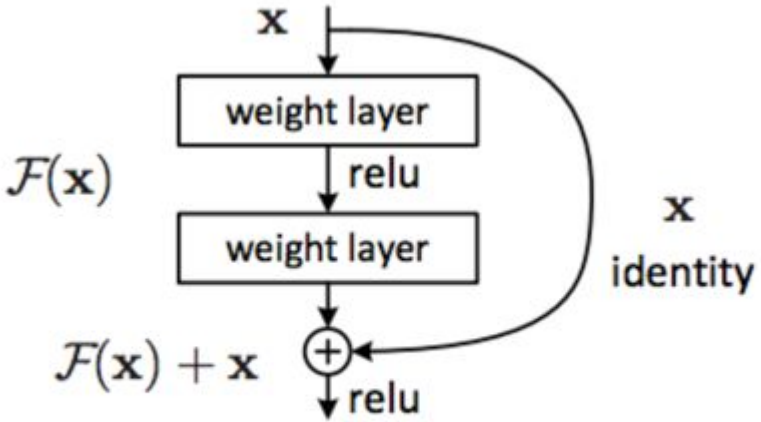
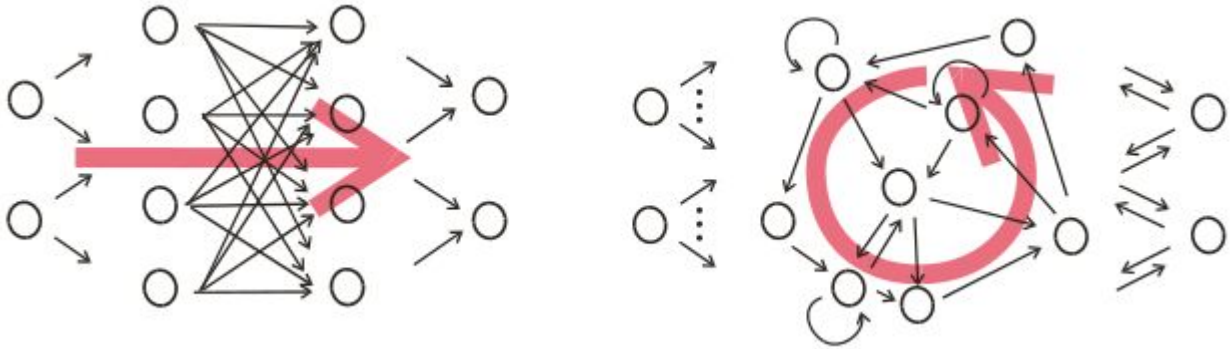


Figure 2. Residual learning: a building block.

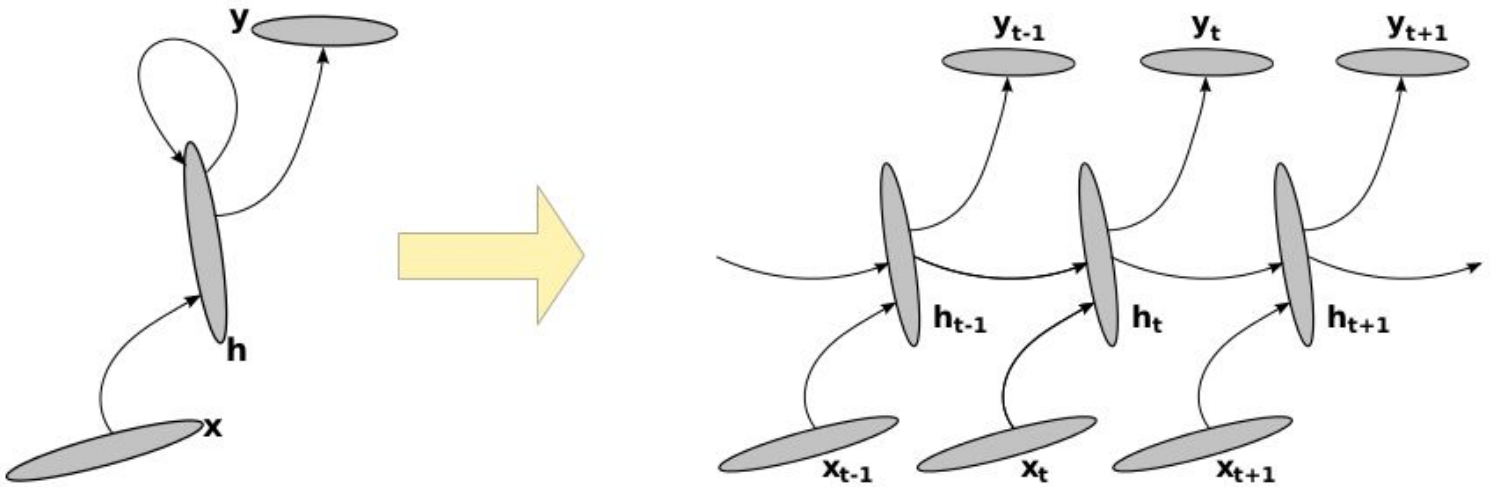
He et al. 2015

Recurrent models

What is an RNN

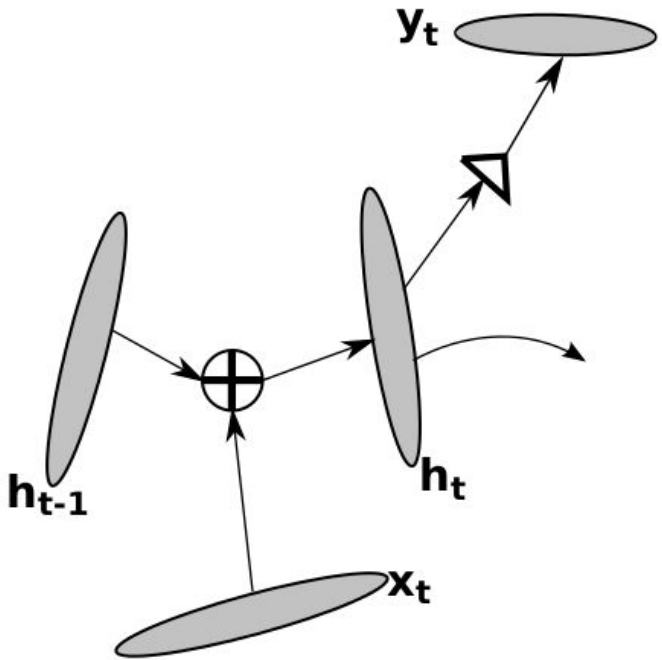


Depictions of RNNs



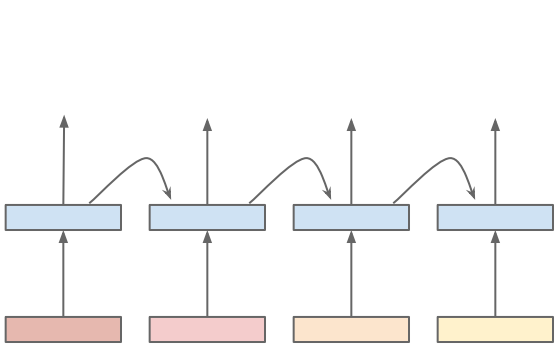
Understanding RNNs

Assign name of the operations involved in an RNN
(e.g. summarization, readout)

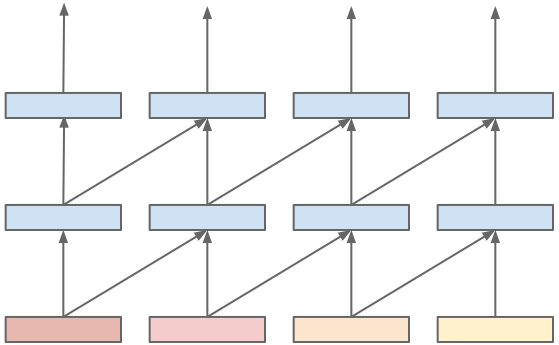


Pascanu et al. 2014

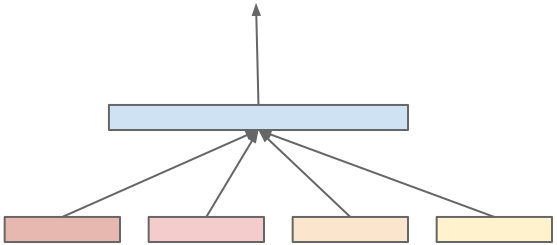
Understanding RNNs



Recurrent Network

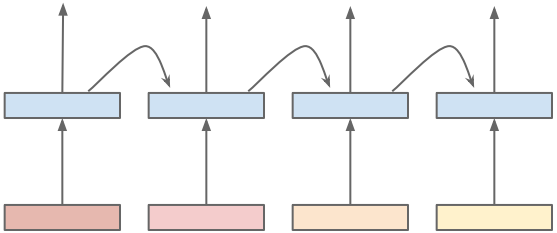


Convolutional Network

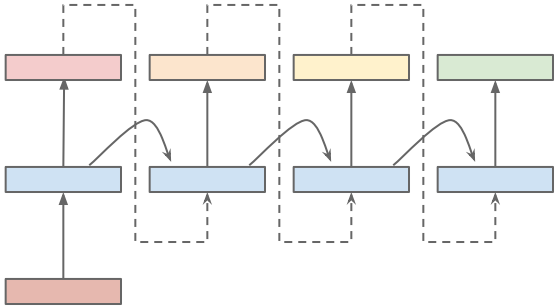


MLP

Understanding RNNs



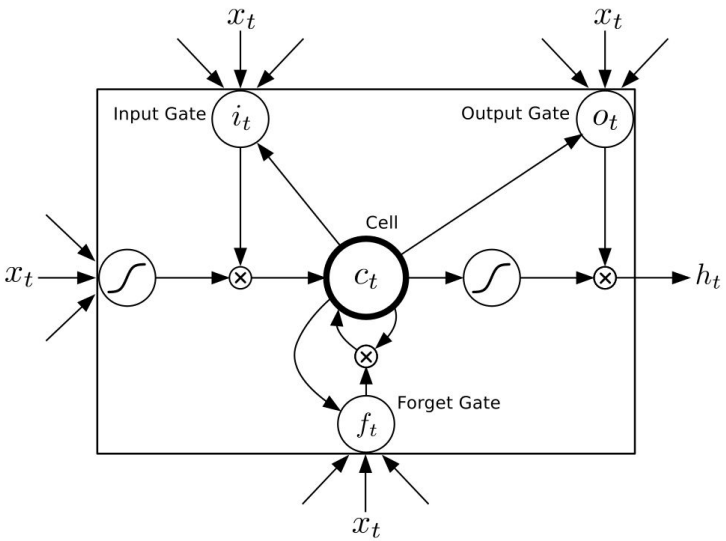
Teacher forcing



Unconstrained

LSTMs

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right) \quad (7)$$
$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \quad (8)$$
$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right) \quad (9)$$
$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right) \quad (10)$$
$$h_t = o_t \tanh(c_t) \quad (11)$$



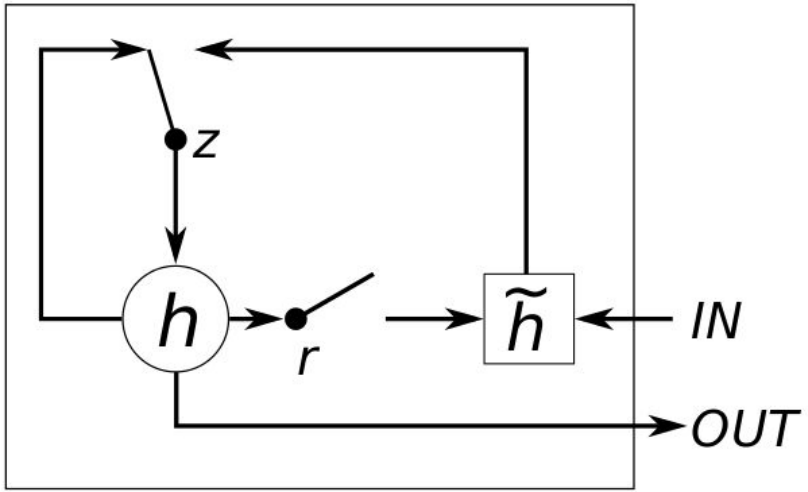
Hochreiter et al. 1997

Graves 2013

GRUs

Chung et al. 2015

$$z = \sigma(W_z x_t + U_z h_{t-1})$$
$$r = \sigma(W_r x_t + U_r h_{t-1})$$
$$\tilde{h} = tanh(W_h x_t + U_h(r \circ h_{t-1}))$$
$$h_t = (1 - z) \circ h_{t-1} + z \circ \tilde{h}$$



(b) Gated Recurrent Unit

Thank you for listening

Questions?