Deep Learning

Basics

Train the Trainer Workshop

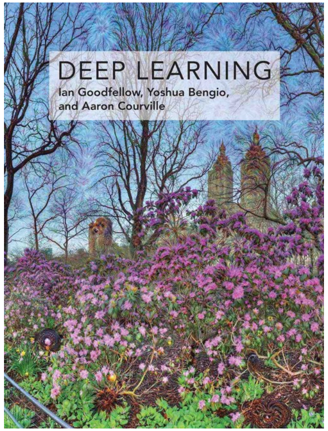
Thomas Kuhr
LMU Munich

Aachen, 30.03.2022

Bundesministerium
für Bildung
und Forschung

LMU
TUM
ORIGINS
Excellence Cluster
lrz

Literature, Material

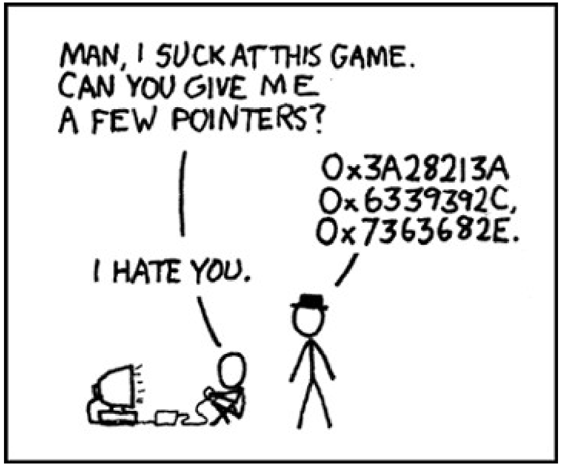


- Deep Learning, Goodfellow et al.
  deeplearningbook.org
- Deep Learning for Physics Research,
  Erdmann et al., deeplearningphysics.org
- Introduction to Machine Learning with Python,
  https://github.com/amueller/introduction_to_ml_with_python
- HSF: Introduction to Machine Learning,
  https://hsf-training.github.io/hsf-training-ml-webpage/
- ...

Human vs. Computer

$591342.46^{0.724}$

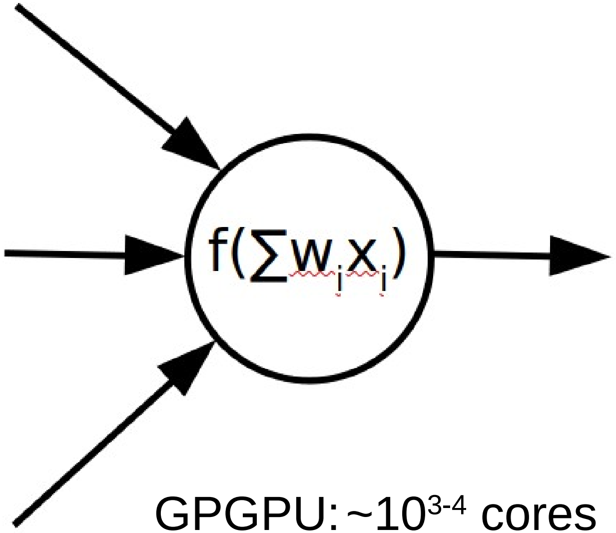
xkcd.com

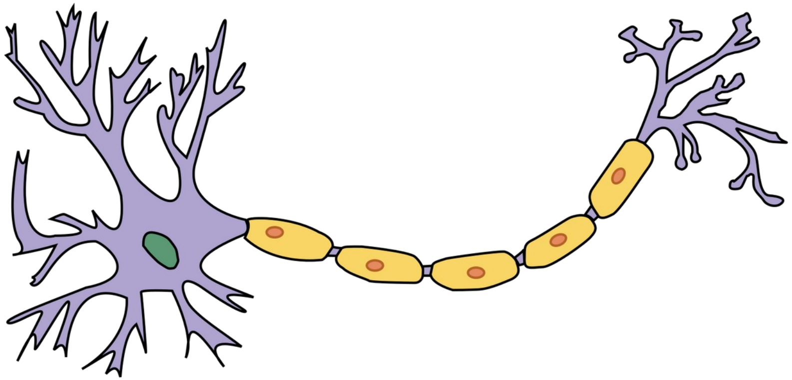


Easy for computers,
hard for humans

Easy for humans,
hard for computers

Neurons



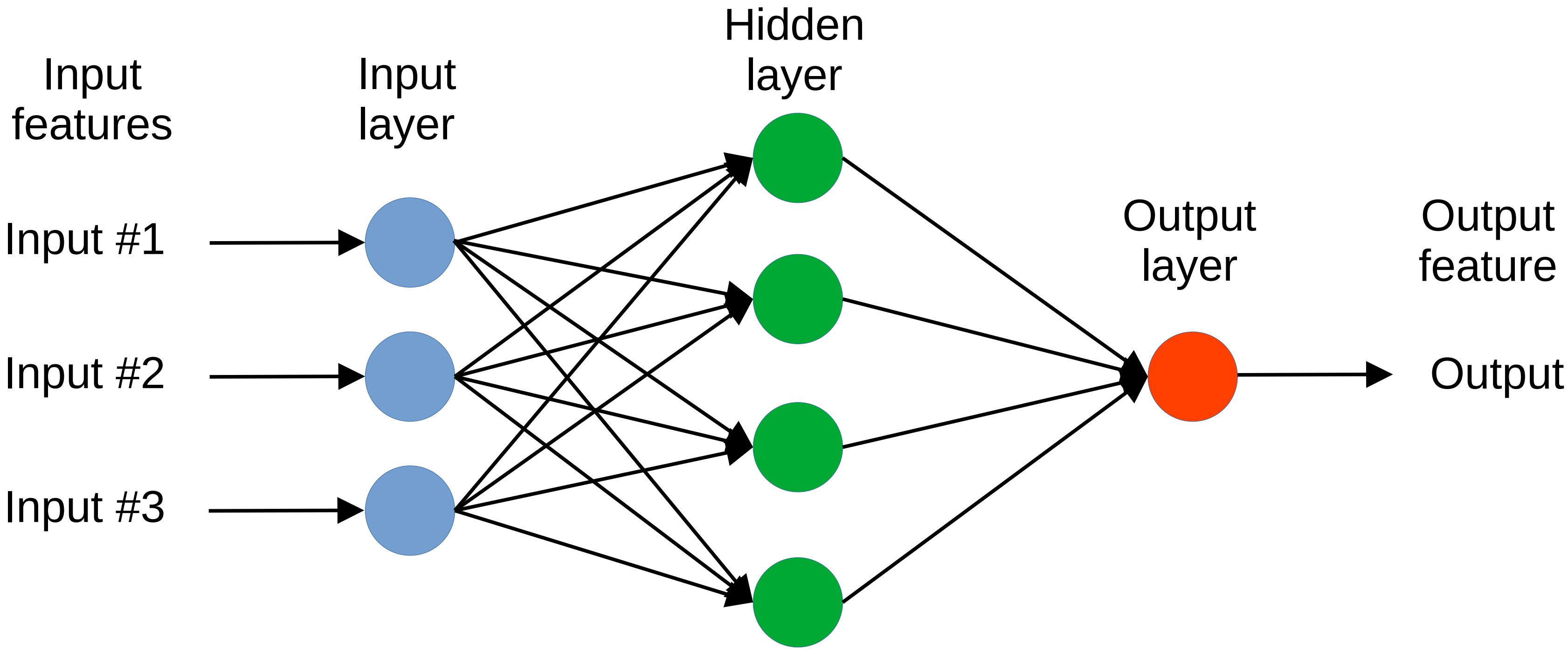


GPGPU: ~$10^{3\text{-}4}$ cores
~200 watt
FPGA: ~10^7 cells

Human: ~10^{11} neurons with
~10^4 connections each,
~12 watt
Insects: ~10^6 neurons with
~10^3 connections each

➔ Connectionism: Solving complex problems by combining many simple, generic elements

Multi Layer Perceptron

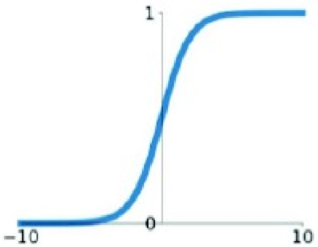


- Each node: $x_i \rightarrow f(\sum w_i x_i)$, weights w_i, activation function f

Activation Functions

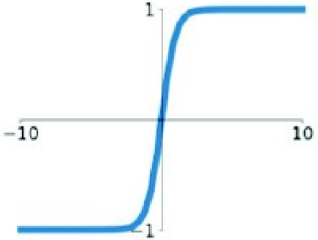
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



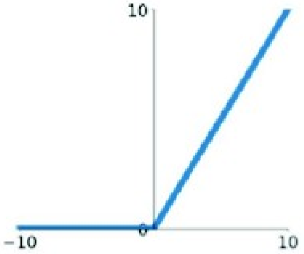
tanh

$\tanh(x)$



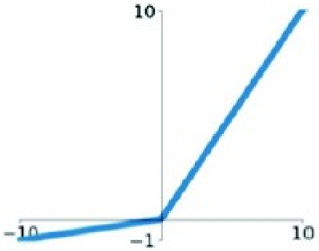
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

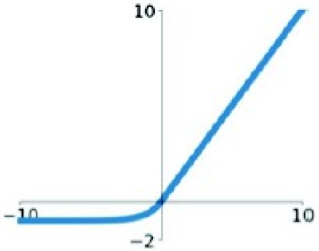


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

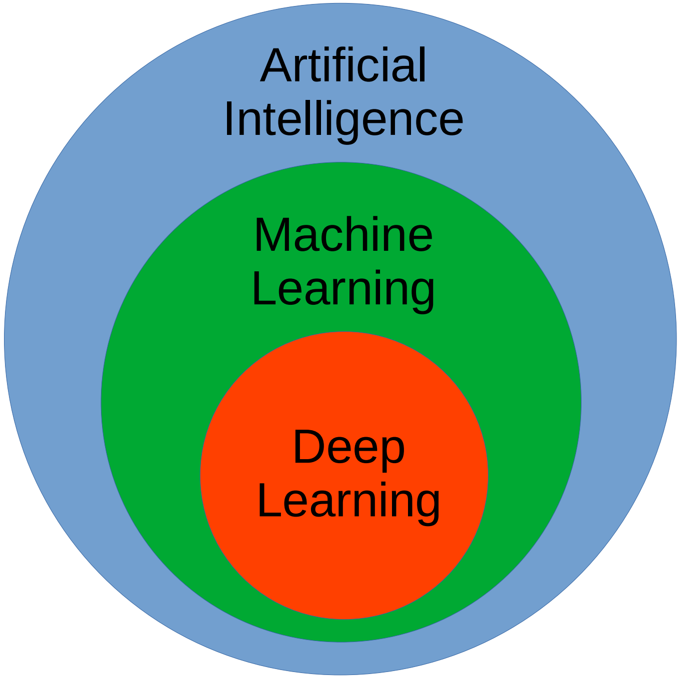
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Machine Learning

- Task defines desired relation between input and output
- Network implements a model
- Model is trained on the experience of a data set
  - Supervised with labeled data
  - Unsupervised with unlabeled data
- Choice of model parameters?



Loss Function

- Deviation of actual network output o from desired target value t (label) measured by loss (or error, or cost, or objective) function L for all data points with index i=1..N

- Common loss function for regression:
  - Mean squared error (MSE): $L = 1/N \sum_i (o_i - t_i)^2$
- Common loss function for classification (with estimated probability $q^k = o^k$ and true probability $p^k = t^k$, usually 0 or 1, for class k):
  - Cross entropy for m classes: $L = -\sum_i \sum_{k=1..m} p_i^k \ln(q_i^k)$
  - Cross entropy for two classes: $L = -\sum_i p_i \ln(q_i) + (1-p_i) \ln(1-q_i)$

- Minimization of loss function → optimization problem

Information Content of Events

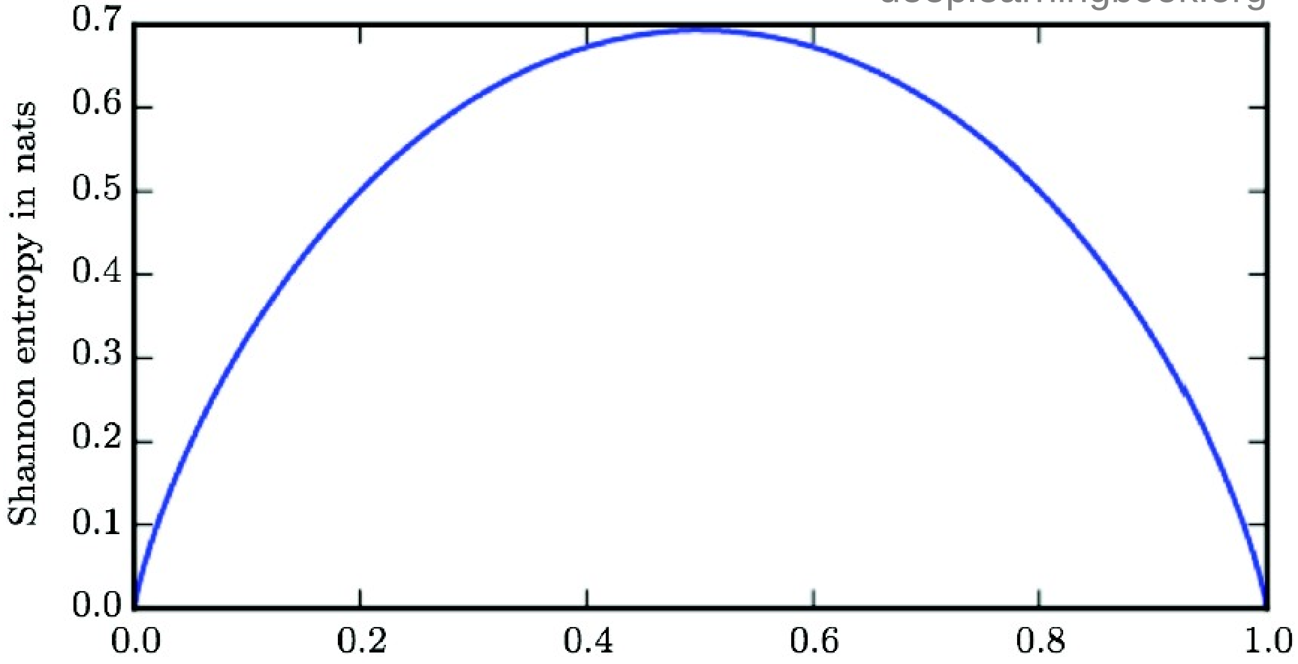
- Should be higher for less likely events

- Should add up for independent events

→ Information content of event with probability p: *$I = -\ln(p)$*

- Unit: nat (or nit) for ln, bit for $\log_2$

deeplearningbook.org



coin flip probability

Cross Entropy

- Average information content for a distribution p of events:

  $lim_{n\to\infty}\ 1/n \sum_{events} I_j = -\sum_{states} p_i\ ln(p_i) =: H(p)$

- Shannon entropy H(p):
  - Average number of nats of a message about an event for optimal encoding for distribution p

- Cross entropy for distribution q:

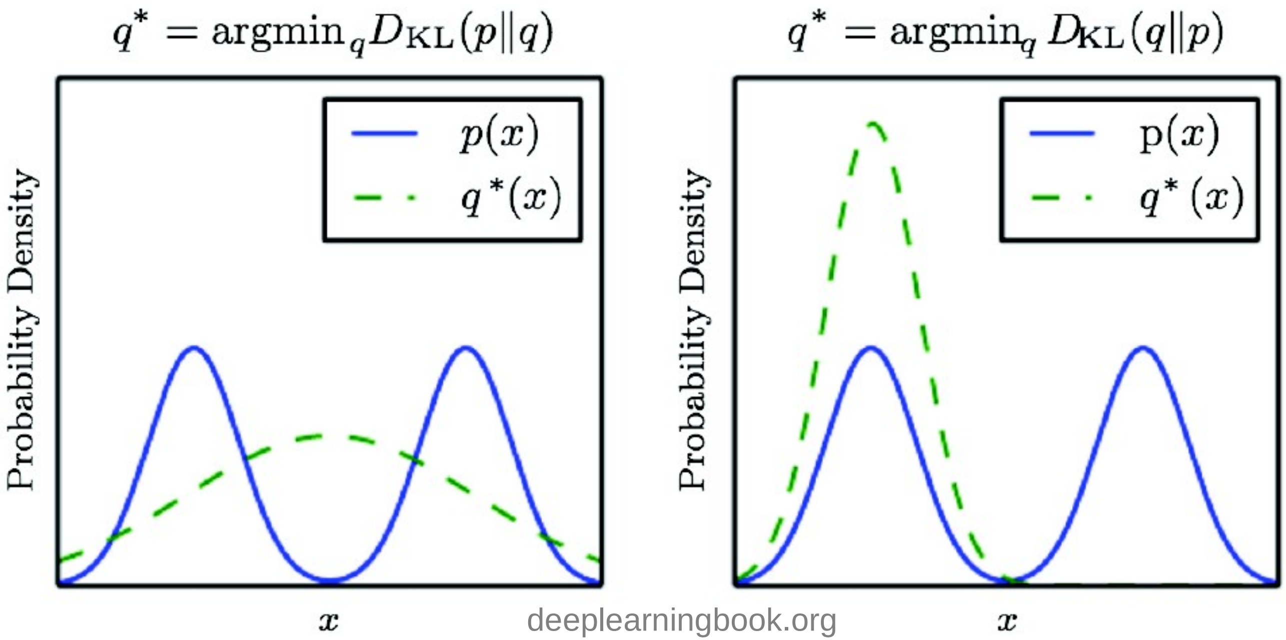
  $H(p,q) = -\sum p_i\ ln(q_i)$

  $= -\sum p_i\ [ln(q_i) + ln(p_i) - ln(p_i)] = H(p) - \sum p_i\ ln(q_i/p_i)$

- Kullback-Leibler divergence: $D_{KL}(p,q) = -\sum p_i\ ln(q_i/p_i)$
  - Average additional number of nats needed for a message about an event for optimal encoding assuming distribution q

Kullback-Leibler Divergence

- Measure of difference between distributions
- Minimum of 0 if and only if q = p
- Not symmetric (→ not a distance measure)



deeplearningbook.org

Maximum Likelihood

- Probability (density) of single event given by q_j
- Total likelihood: $L = \prod_{events} q_j$

➢ Find model parameters that maximize L
by minimizing negative log likelihood:

$$-\ln(L) = -\sum_{events} \ln(q_j) = -\sum_{states} p_i \ln(q_i)$$

➔ Cross-entropy loss function ↔ maximum likelihood fit

Backpropagation

- $o = f(\sum_i w_i^{(x)} x_i) = f(\sum_i w_i^{(x)} g(\sum_j w_{ij}^{(y)} y_j)$
  $= f(\sum_i w_i^{(x)} g(\sum_j w_{ij}^{(y)} h(\sum_k w_{jk}^{(z)} z_k)) = \ldots$

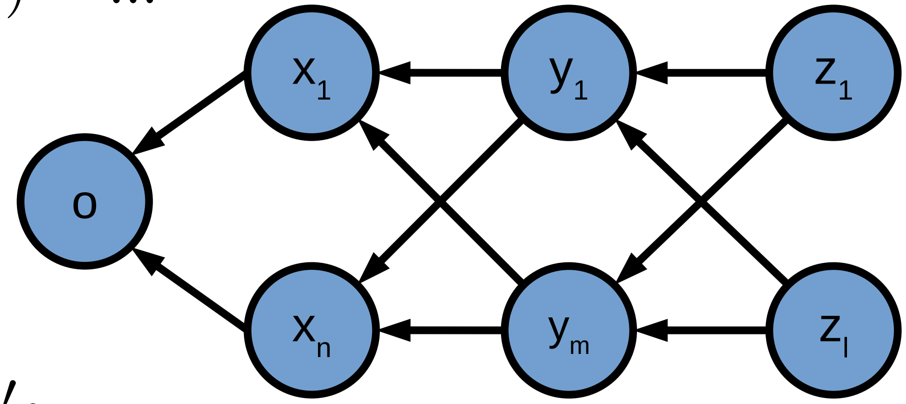
➔ Derivative of loss function L(o) with respect to weights?

- Chain rule:

$$\frac{\partial L}{\partial w_i^{(x)}} = L' f' x_i$$

$$\frac{\partial L}{\partial w_{ij}^{(y)}} = L' f' \sum_i w_i^{(x)} g' y_j$$

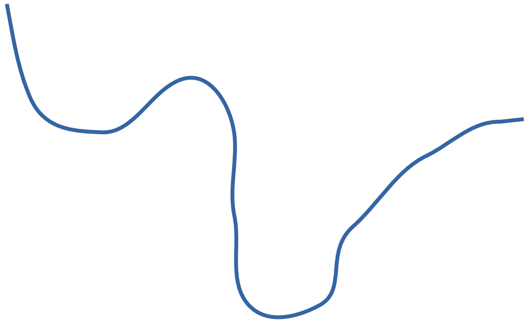
$$\frac{\partial L}{\partial w_{jk}^{(z)}} = L' f' \sum_i w_i^{(x)} g' \sum_j w_{ij}^{(y)} h' z_k$$



➔ Go backward in net and reuse already calculated values

Optimizers

- Task: find global minimum of loss function in model parameter space: L(w) → min



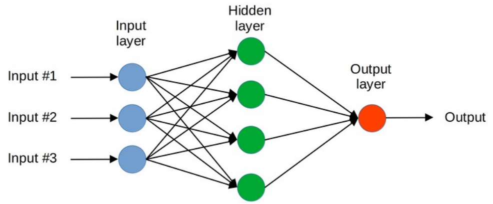
- ➔ Gradient descent → learning rate η: $\Delta\vec{w} = -\eta\frac{\partial L}{\partial\vec{w}}$
  - Learning rate usually decreased
  - Exploding gradient problem → gradient clipping
  - Vanishing gradient problem → momentum term $\Delta\vec{w}_{t+1} = -\eta\frac{\partial L}{\partial\vec{w}} + \alpha\Delta\vec{w}_t$
  - Line search
- ➔ AdaGrad, RMSProp, Adam: dynamic adjustment of algorithm parameters
- ➔ Newton, BFGS: step length determined from second derivative

Model Training

- Choice of architecture and training parameters (hyper-parameters) often based on problem-specific experience
- Preprocessing of input variables (features)
  → reasonable range, decorrelation
- Model parameter starting values → random, reasonable range

- Iterative training process
  → reuse data multiple times (epochs)
- Updates with chunks of partial data (mini batches)
  → stochastic learning

Theorems



Universal Approximation Theorem:

Hornik et al., 1989; Cybenko, 1989

- A feed-forward network with linear output and at least one hidden layer with a finite number of nodes can approximate any continuous function on closed or bound subsets of $\mathbb{R}^n$ to arbitrary precision.

No Free Lunch Theorem:

Wolpert, 1996

- Averaged over all possible data-generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points

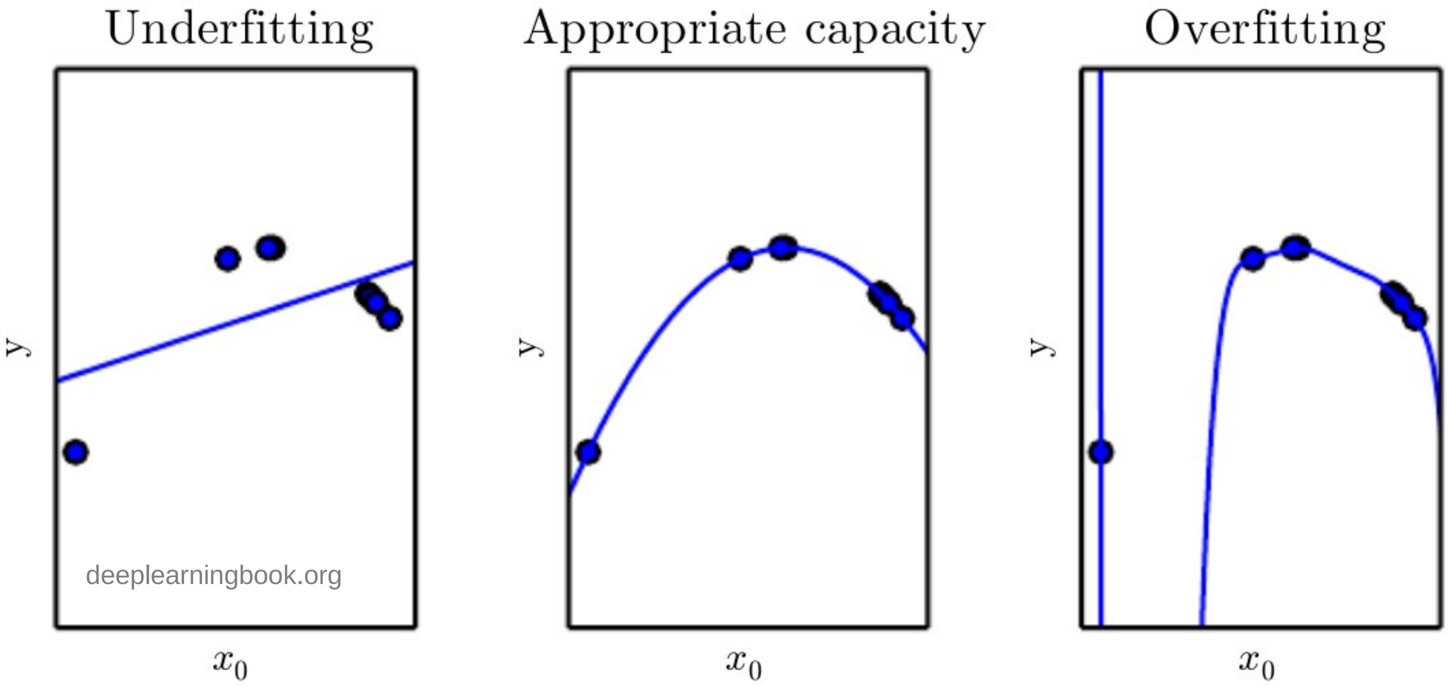
→ No machine learning algorithm is universally any better than any other.

Generalization

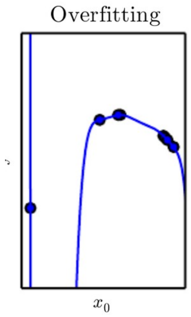
- Aim: Minimize loss with respect to true distribution instead of samples that are drawn from it

- Requires appropriate capacity of model

- More data helps



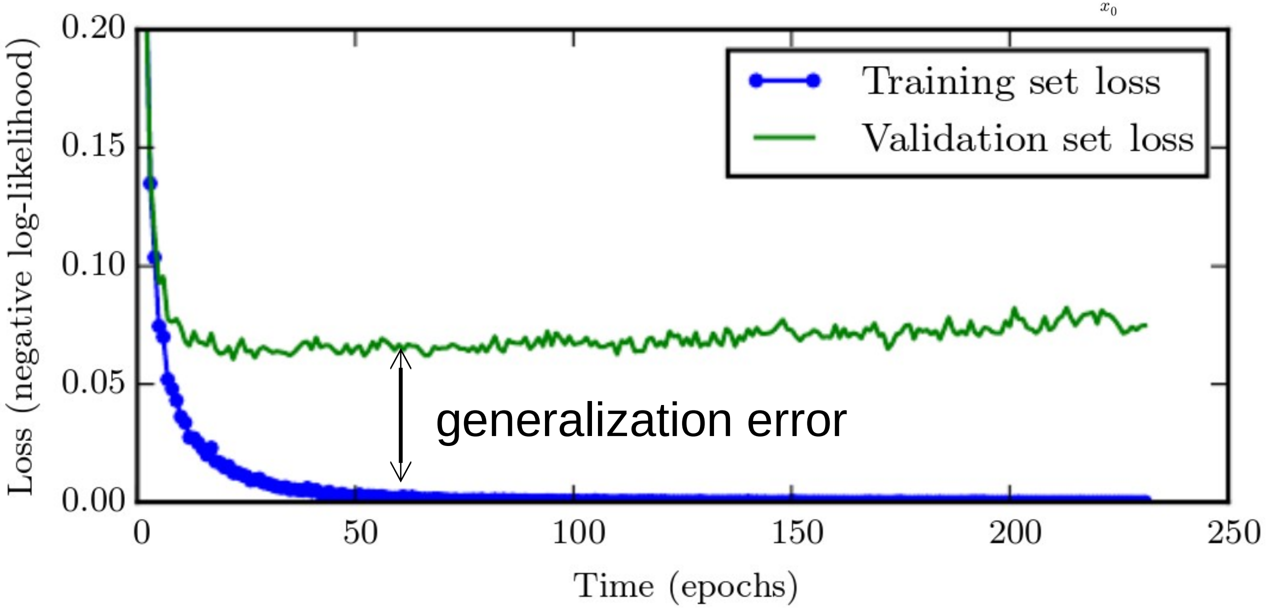
Detection of Overfitting



deeplearningbook.org

- Test for overfitting with independent test dataset

  → no overfitting if $L_{test} = L_{train}$

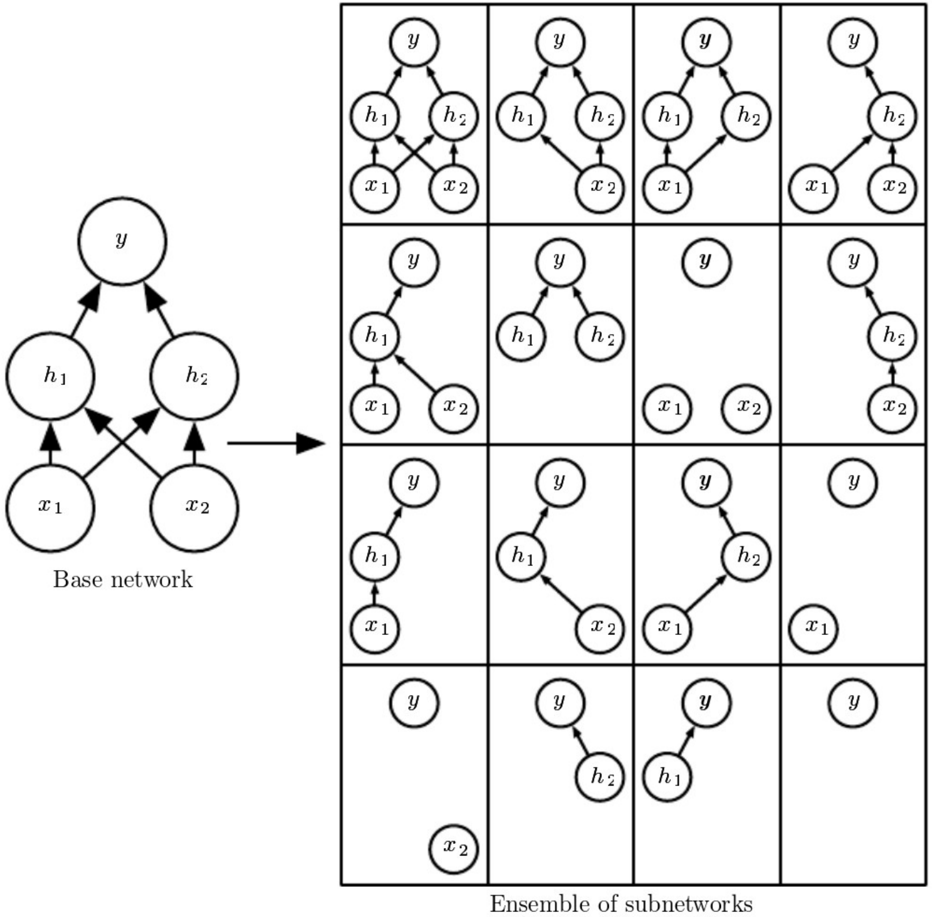


- Split in training, validation, and test datasets if hyperparameters are tuned

Regularization

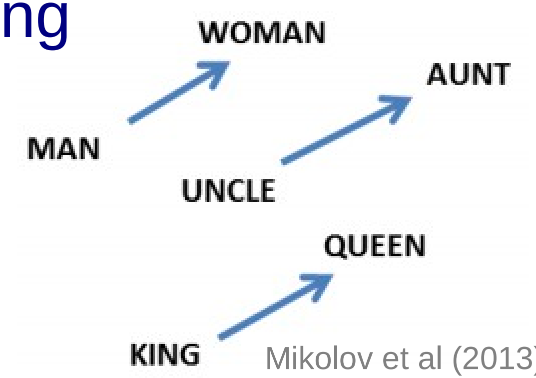
deeplearningbook.org

- Early stopping
- Penalty terms in loss function
- Addition of noise
- Reduction/sharing of parameters
- Dropout



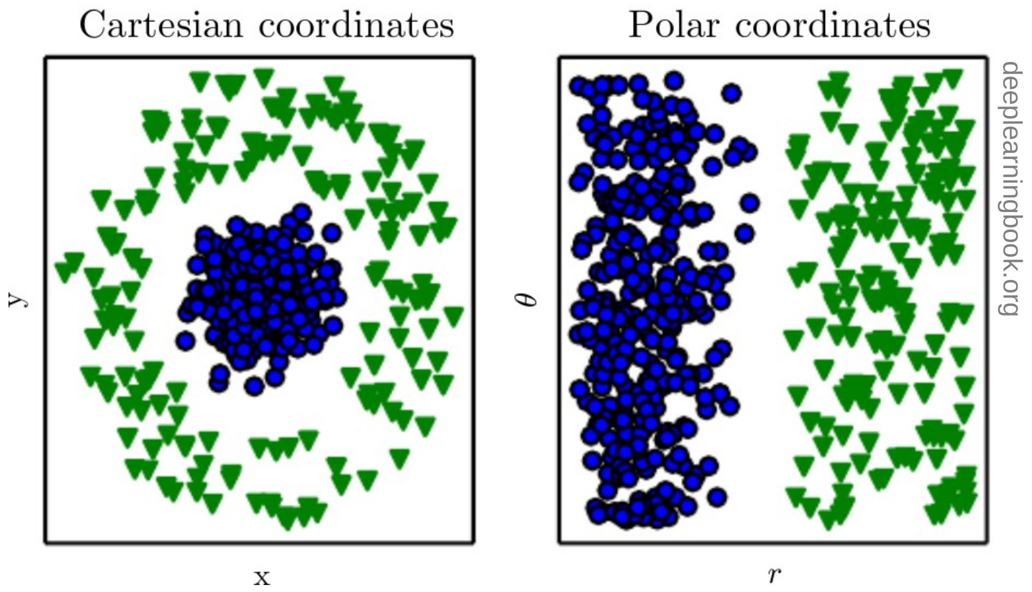
Representation

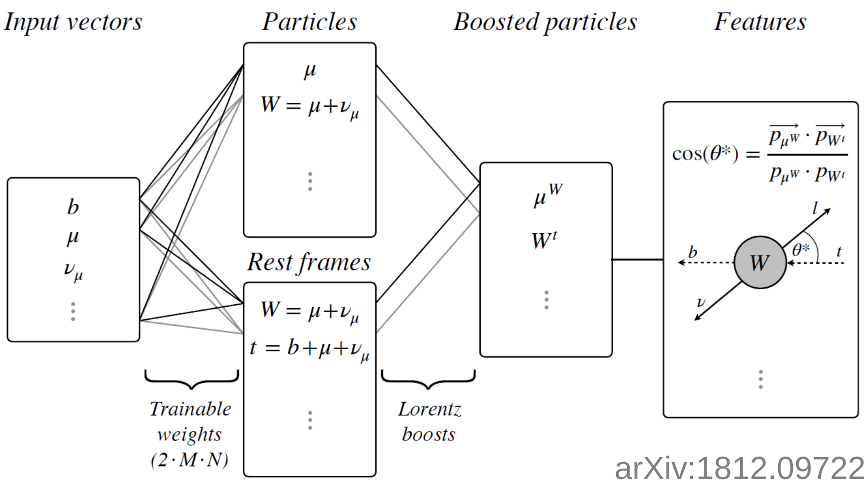
- Domain knowledge
- Pre-processing
- One-hot encoding for unordered categories
- Embedding



Mikolov et al (2013)

- Representation learning



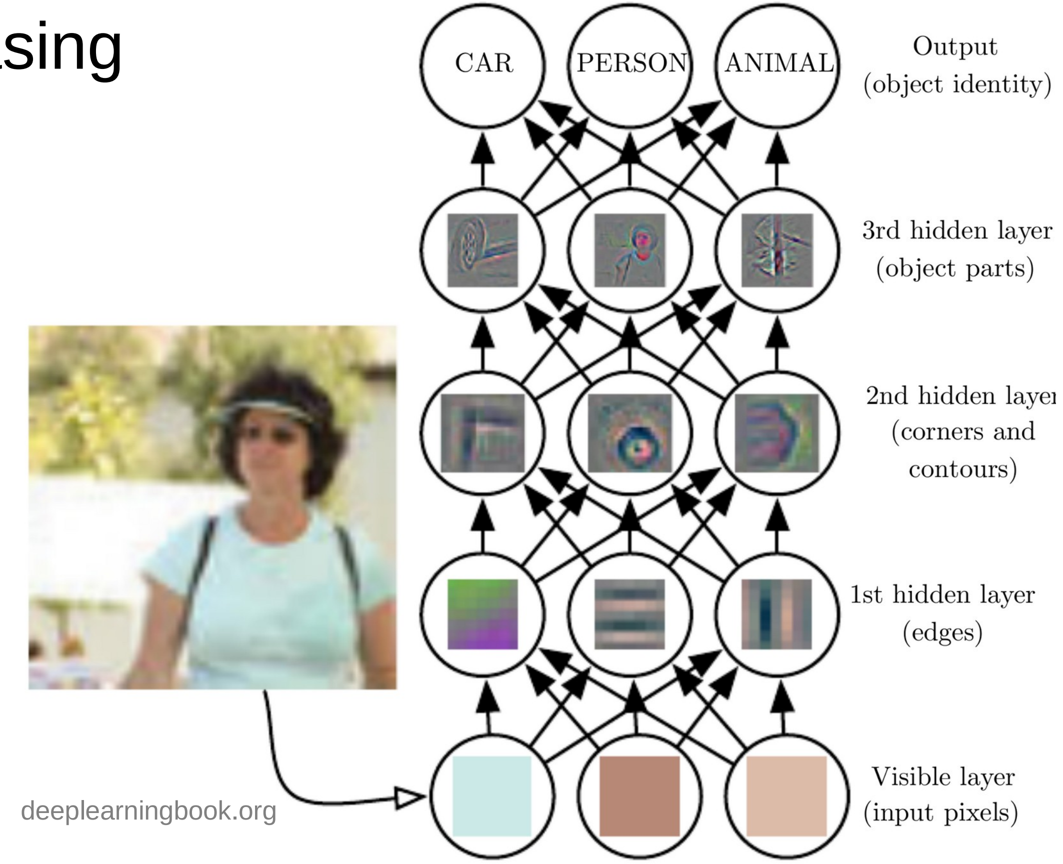


arXiv:1812.09722

Deep Learning

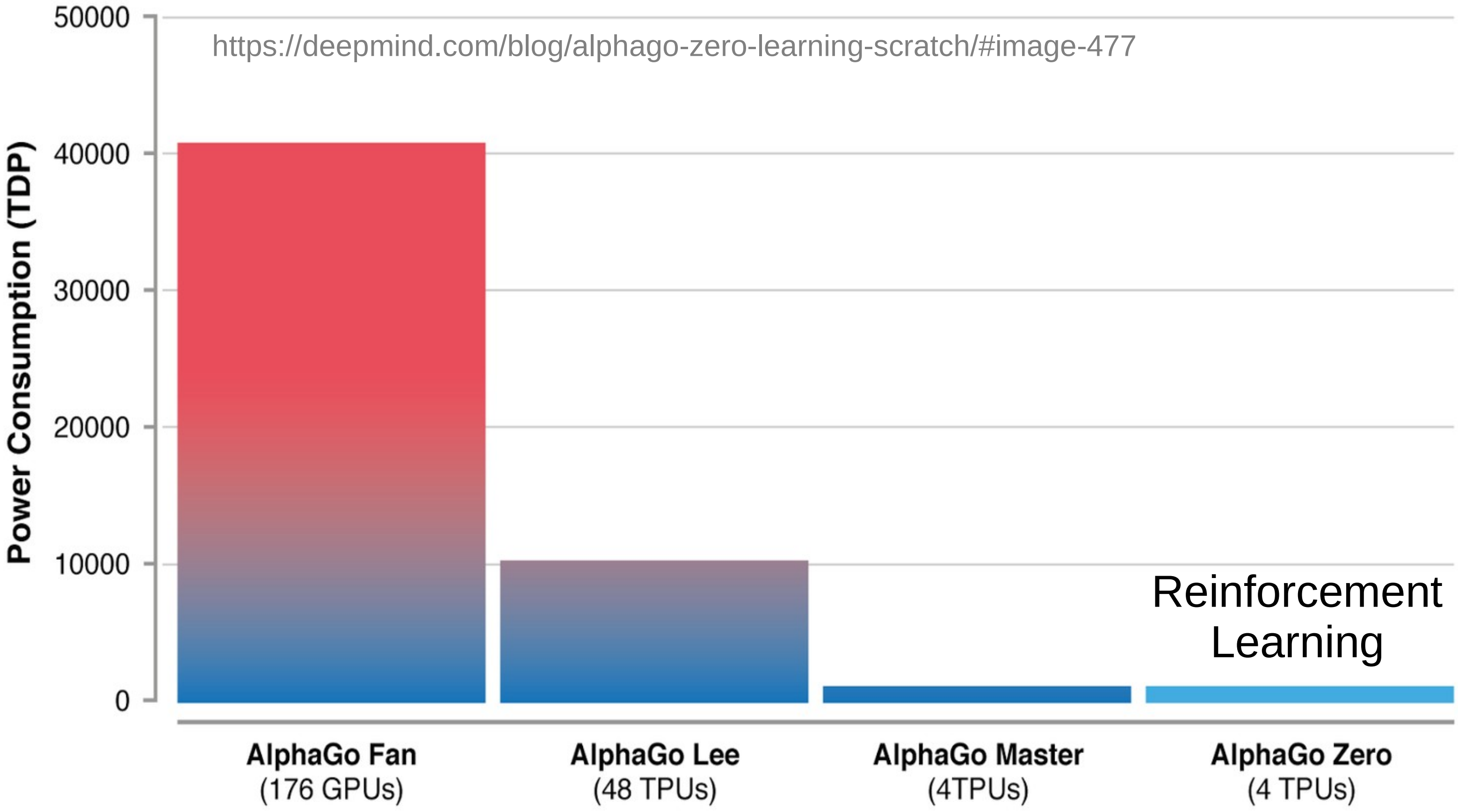
- Multiple layers (with increasing level of representation)
- High model capacity

- Technological progress due to
  - Powerful hardware
  - Huge datasets
  - Available tools



deeplearningbook.org

Example: AlphaGo

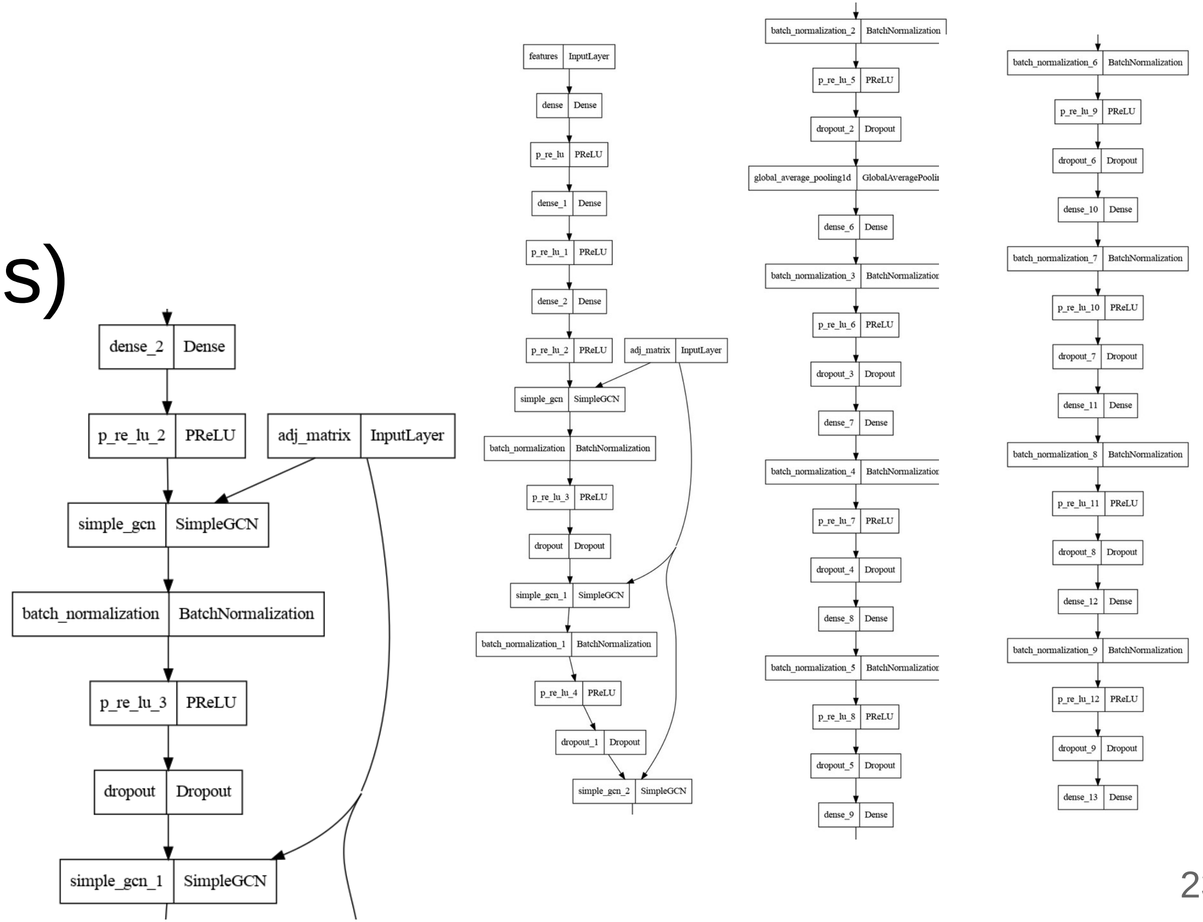


Network Architectures

- Pooling
- Softmax
  (for multiple classes)

$$f(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_i)}$$

- Convolution
- Recursion
- ...

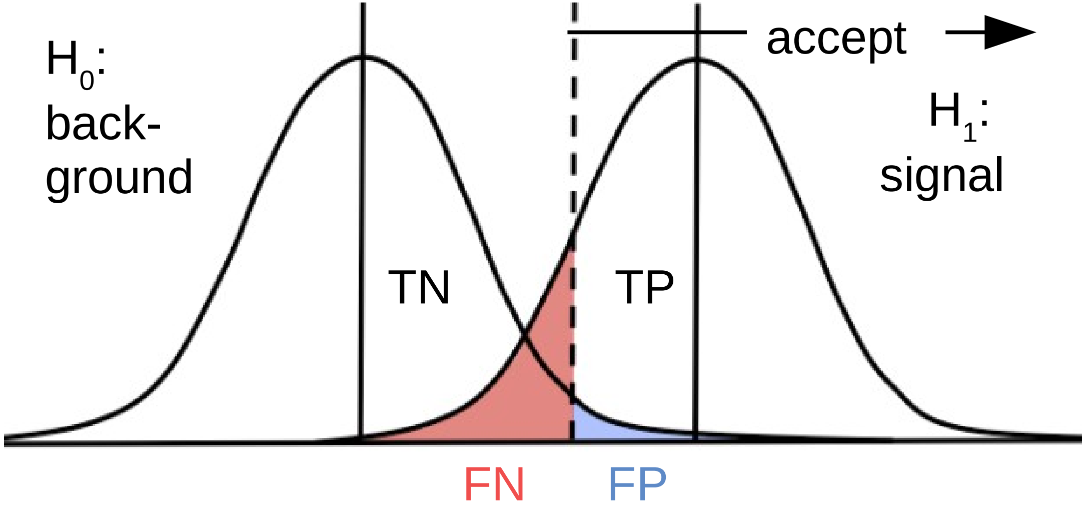


Classification Performance

- Type I error: false positive
  FP rate (FPR) = FP / (TN+FP)
  significance α, p-value
- ➔ Positive predictive value
  PPV = TP / (TP+FP)
  precision, purity

- Type II error: false negative
  FN rate (FNR) = FN / (TP+FN)
- ➔ TPR = TP / (TP+FN) = 1 – FNR
  recall, sensitivity,
  power β, efficiency ε

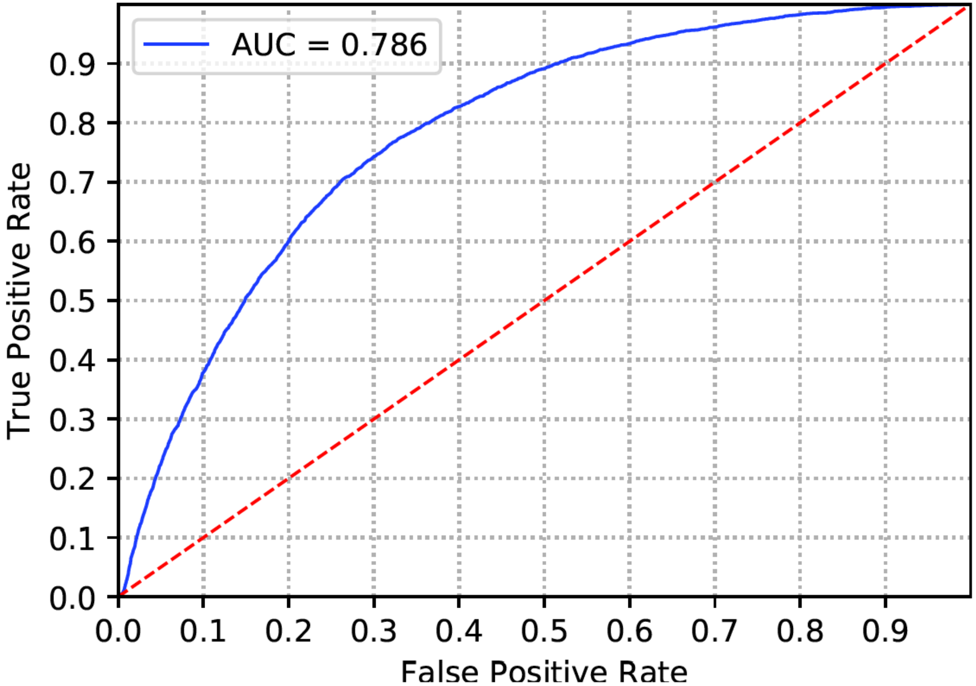
- Accuracy (TP+TN)/(TP+FP+TN+FN)



confusion matrix	hypothesis accepted	hypothesis rejected
signal	**T**rue **P**ositive	**F**alse **N**egative
background	**F**alse **P**ositive	**T**rue **N**egative

ROC Curve, Cross-Validation

Receiver **O**perator **C**haracteristic (ROC) curve
→ **A**rea **U**nder **C**urve

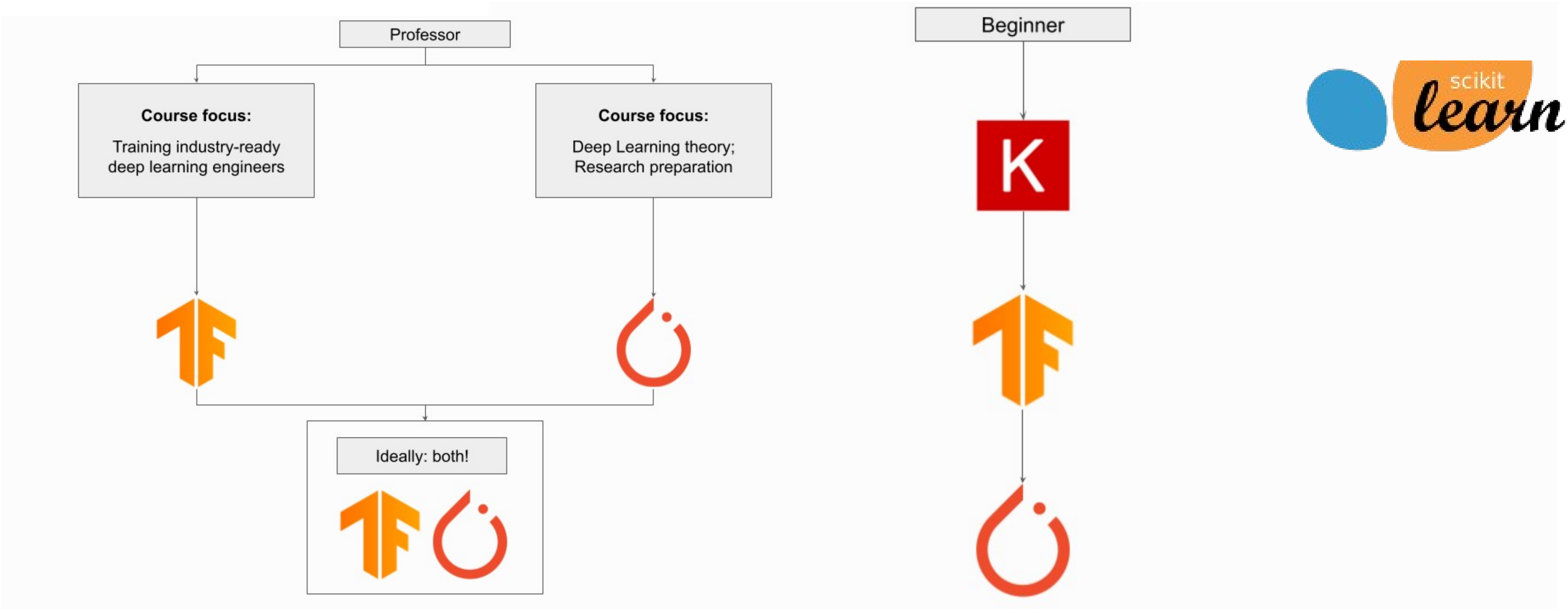


Cross-validation:

- Split dataset in N parts
- Loop i=1...N
  - Train model with all data except part i
  - Validate on part i
- Sum validations on partial data

Which Tool?

- https://www.assemblyai.com/blog/pytorch-vs-tensorflow-in-2022/



Documentation, Tensorboard

- https://www.tensorflow.org/guide/keras