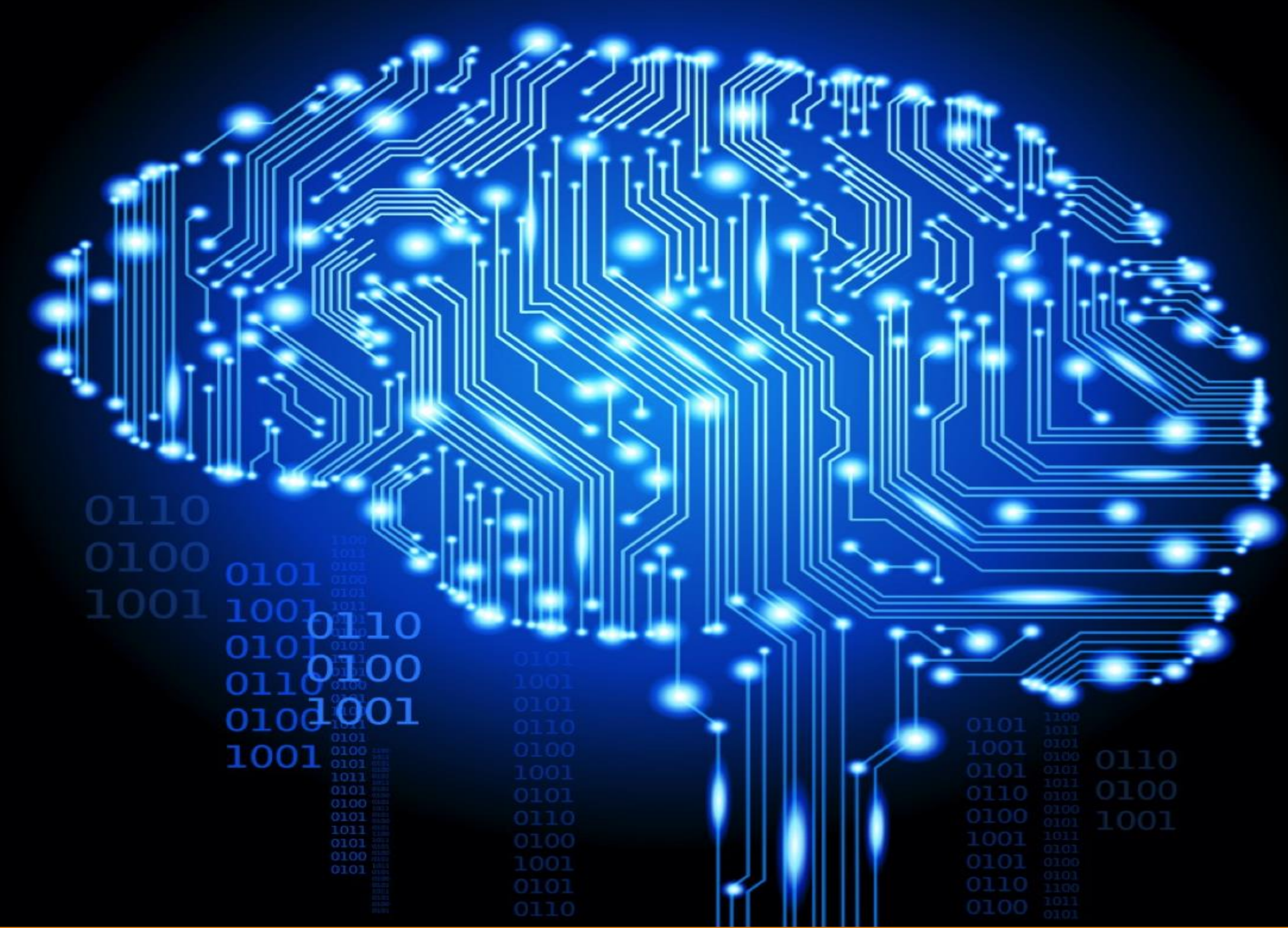


Lecture 1: Introduction to Deep Learning

Efstratios Gavves

Prerequisites

- Machine Learning 1
- Calculus, Linear Algebra
  - Derivatives, integrals
  - Matrix operations
  - Computing lower bounds, limits
- Probability Theory, Statistics
- Advanced programming
- Time, patience & drive

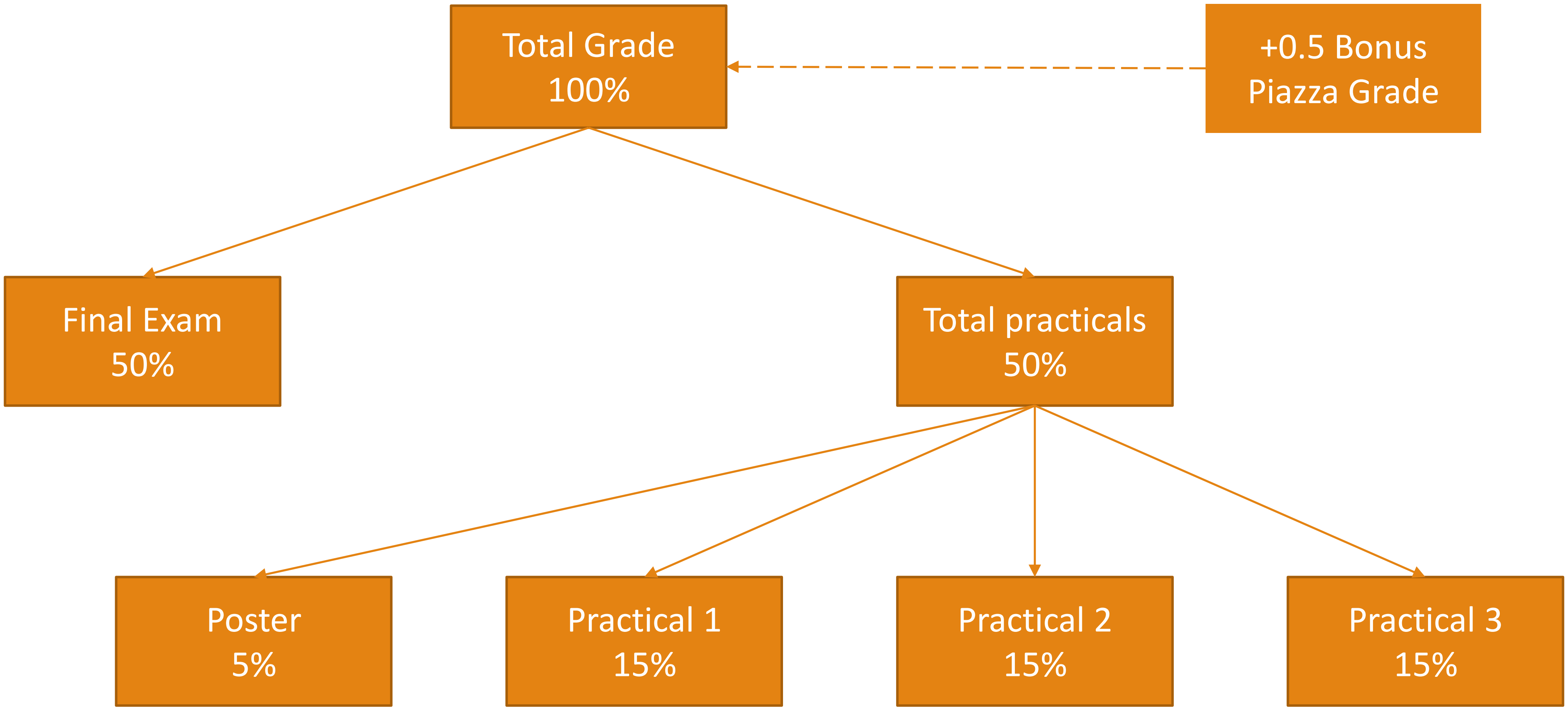
Learning Goals

- Design and Program Deep Neural Networks
- Advanced Optimizations (SGD, Nestorov's Momentum, RMSprop, Adam) and Regularizations
- Convolutional and Recurrent Neural Networks (feature invariance and equivariance)
- Unsupervised Learning and Autoencoders
- Generative models (RBMs, Variational Autoencoders, Generative Adversarial Networks)
- Bayesian Neural Networks and their Applications
- Advanced Temporal Modelling, Credit Assignment, Neural Network Dynamics
- Biologically-inspired Neural Networks
- Deep Reinforcement Learning

Practicals

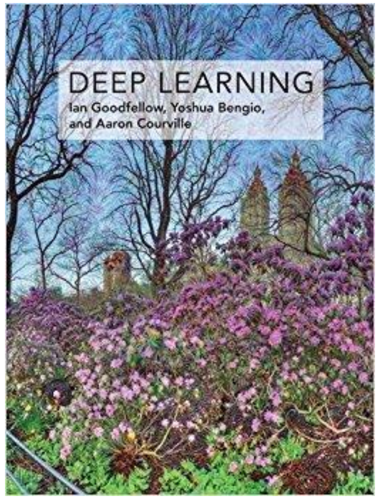
- 3 individual practicals (PyTorch)
  - Practical 1: Convnets and Optimizations
  - Practical 2: Recurrent Networks
  - Practical 3: Generative Models
- 1 group presentation of an existing paper (1 group=3 persons)
  - We'll provide a list of papers or choose another paper (your own?)
  - <u>By next Monday make your team:</u> we will prepare a Google Spreadsheet

Grading


Total Grade
100%
+0.5 Bonus
Piazza Grade
Final Exam
50%
Total practicals
50%
Poster
5%
Practical 1
15%
Practical 2
15%
Practical 3
15%

Overview

- Course: Theory (4 hours per week) + Labs (4 hours per week)
  - All material on http://uvadlc.github.io
  - Book: *Deep Learning* by I. Goodfellow, Y. Bengio, A. Courville (available online)
- Live interactions via Piazza. Please, subscribe today!
  - **Link:** https://piazza.com/university_of_amsterdam/fall2018/uvadlc/home



- Practicals are **individual!**
  - More than encouraged to cooperate but not copy
    The top 3 Piazza contributors get +0.5 grade
  - Plagiarism checks on reports and code → **Do not cheat!**

Who we are and how to reach us

- Efstratios Gavves
  - Assistant Professor, QUVA Deep Vision Lab (C3.229)
  - Temporal Models, Spatiotemporal Deep Learning, Video Analysis
- Teaching Assistants
  - Kirill Gavrilyuk, Berkay Kicanaoglu, Tom Runia, Jorn Peters, Maurice Weiler

@egavves

Efstratios Gavves

Me :P

Kirill

Berkay

Tom

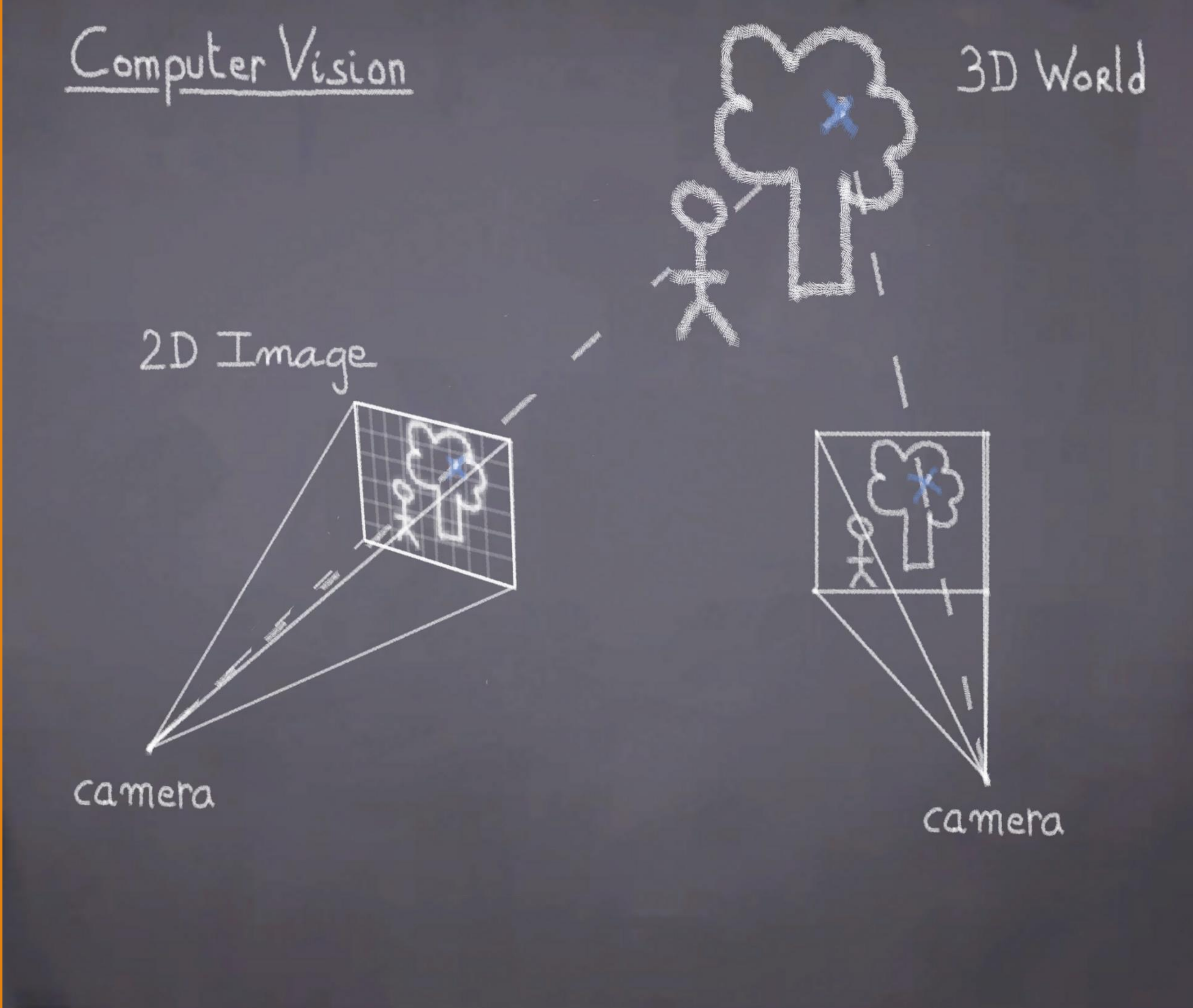
Jorn

Maurice

Lecture Overview

- Applications of Deep Learning in Vision, Robotics, Game AI, NLP
- A brief history of Neural Networks and Deep Learning
- Neural Networks as modular functions

Applications of Deep Learning



Deep Learning in practice

[YouTube](#) | [Youtube](#) | [Website](#)

Large-scale Video Classification with Convolutional Neural Networks, CVPR 2014

Microsoft Deep Learning Semantic Image Segmentation

[Youtube](#) | [Youtube](#)

Deep Sensorimotor Learning

Google DeepMind's Deep Q-learning playing Atari Breakout

Newspapers			
New York San Jose	New York Times San Jose Mercury News	Baltimore Cincinnati	Baltimore Sun Cincinnati Enquirer
NHL Teams			
Boston Phoenix	Boston Bruins Phoenix Coyotes	Montreal Nashville	Montreal Canadiens Nashville Predators
NBA Teams			
Detroit Oakland	Detroit Pistons Golden State Warriors	Toronto Memphis	Toronto Raptors Memphis Grizzlies
Airlines			
Austria Belgium	Austrian Airlines Brussels Airlines	Spain Greece	Spainair Aegean Airlines
Company executives			
Steve Ballmer Samuel J. Palmisano	Microsoft IBM	Larry Page Werner Vogels	Google Amazon

Table 2: Examples of the analogical reasoning task for phrases (the full test set has 3218 examples). The goal is to compute the fourth phrase using the first three. Our best model achieved an accuracy of 72% on this dataset.

Why should we be impressed?

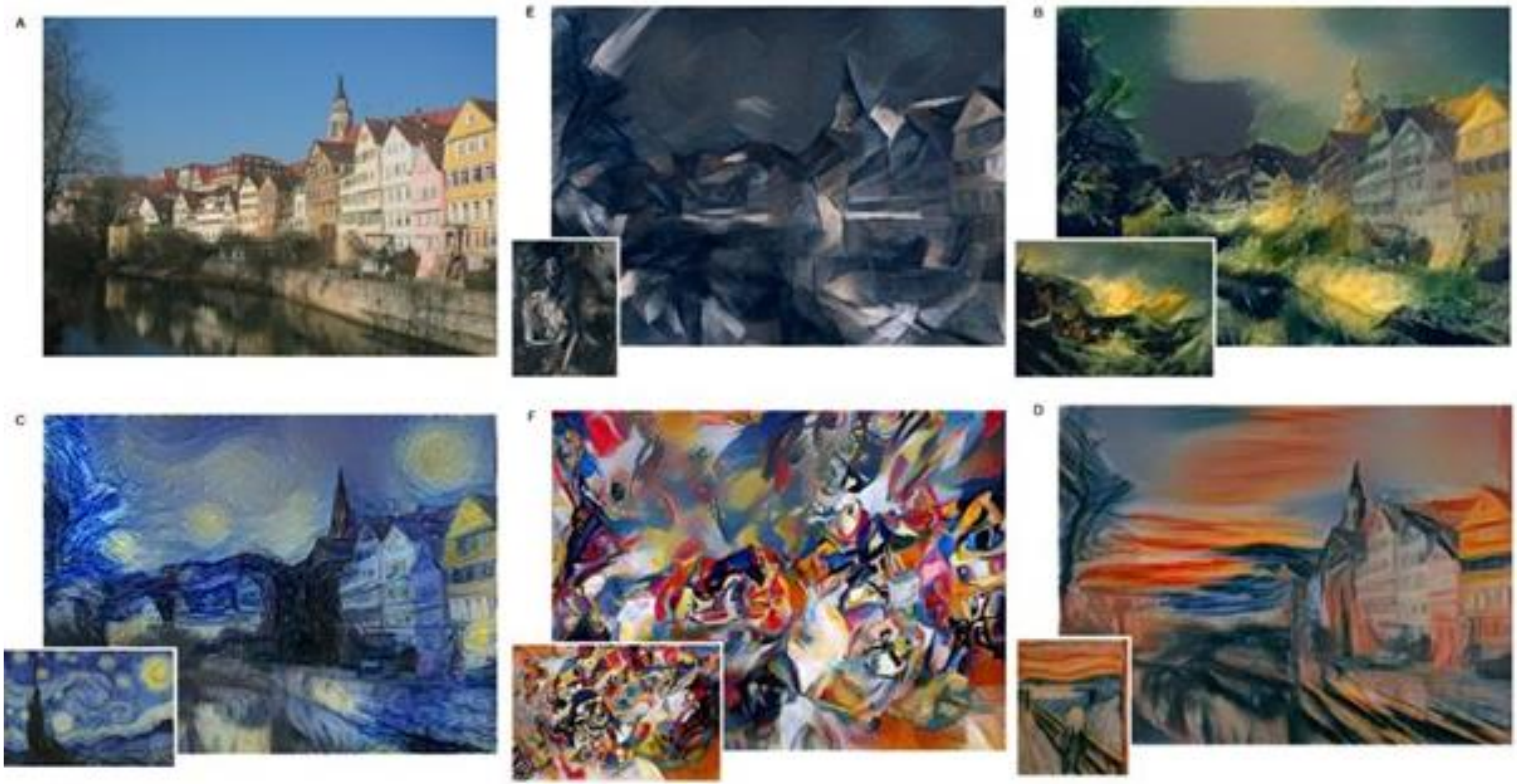
- **Vision** is ultra challenging!
  - For 256x256 resolution → $2^{524,288}$ of possible images (10^{24} stars in the universe)
  - Large visual object variations (viewpoints, scales, deformations, occlusions)
  - Large semantic object variations

Inter-class variation

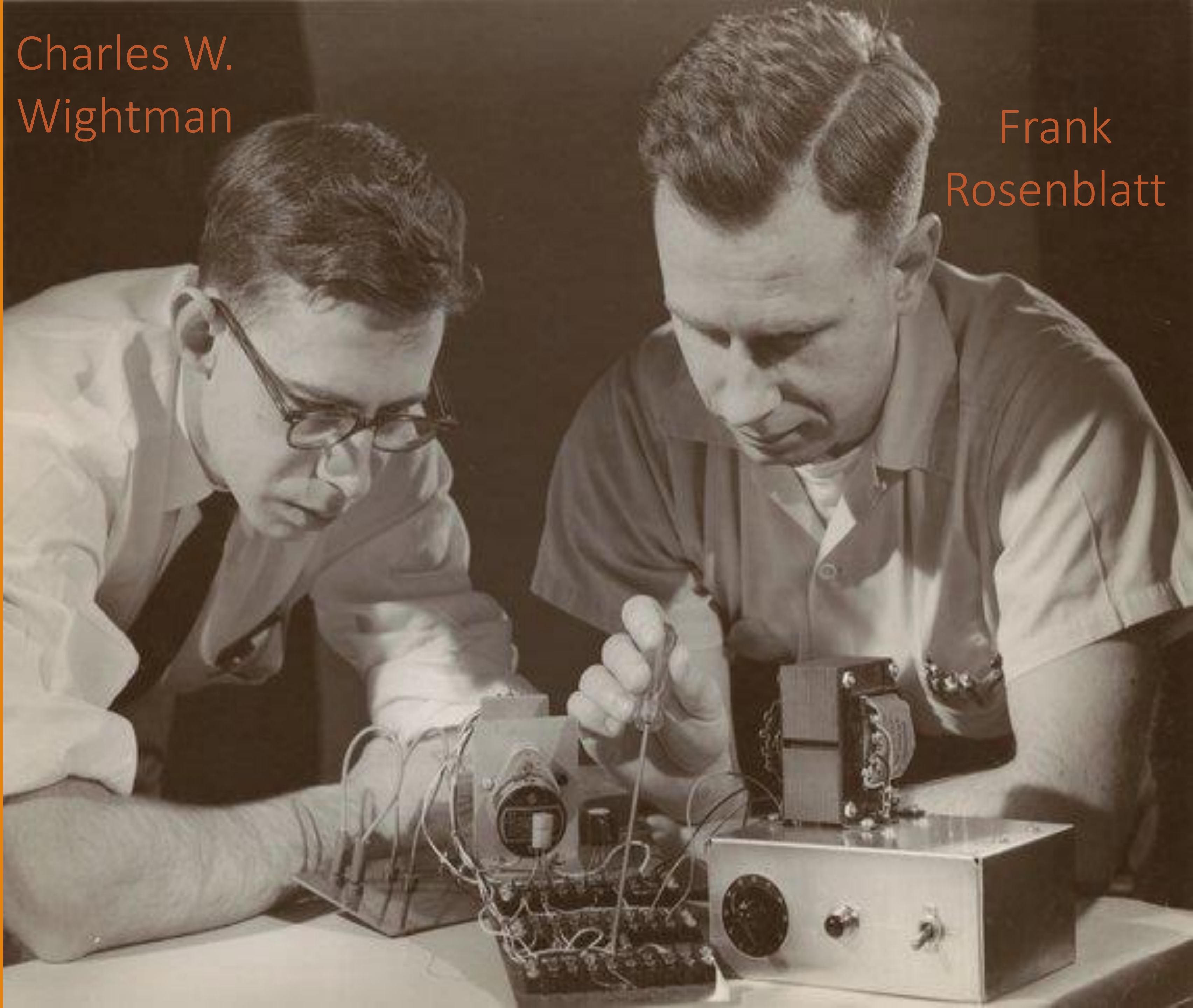
- **Robotics** is typically considered in controlled environme
- **Game AI** involves extreme number of possible games states ($10^{10^{48}}$ possible GO games)
- **NLP** is extremely high dimensional and vague (just for English: 150K words)

Intra-class overlap

Deep Learning even for the arts



A brief history of Neural Networks & Deep Learning

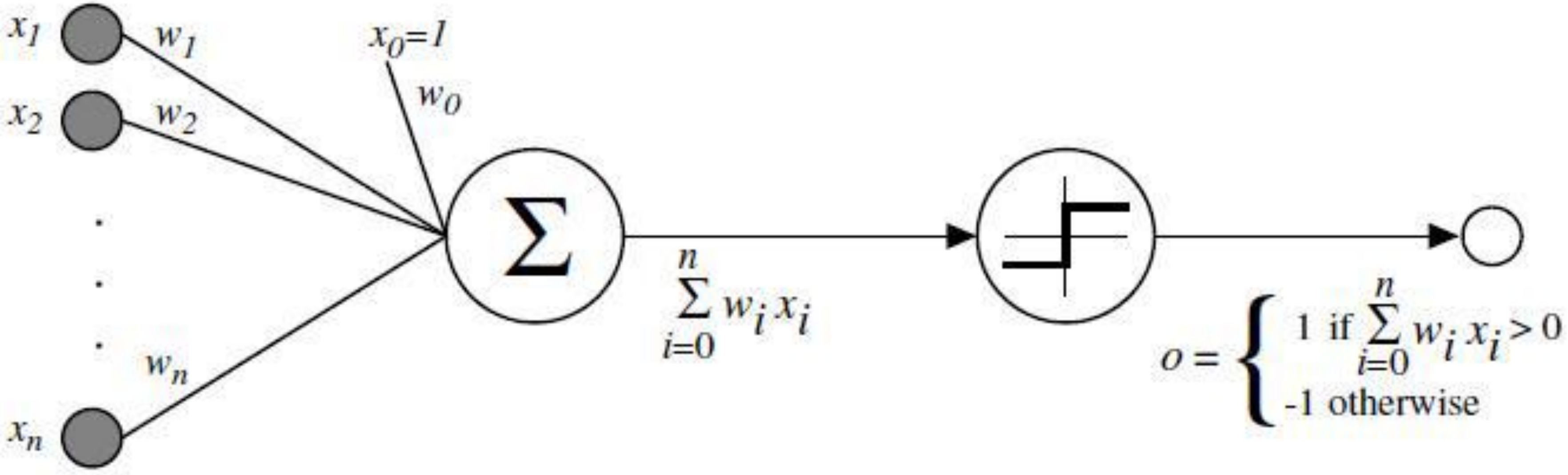


First appearance (roughly)

- 1958 — Perceptrons, Rosenblatt
- 1960 — Adaline, Widrow and Hoff
- 1969 — Perceptrons, Minsky and Papert
- 1970 — Backpropagation, Linnainmaa
- 1974 — Backpropagation, Werbos
- 1986 — Backpropagation, Rumelhart, Hinton and Williams
- 1997 — LSTM, Hochreiter and Schmidhuber
- 1998 — OCR, LeCun, Bottou, Bengio and Haffner
- 2006 — Deep Learning, Hinton, Osindero, Teh
- 2009 — Imagenet, Deng et al.
- 2013 — Alexnet, LeCun, Bottou, Bengio and Haffner
- 2015 — Resnet (154 layers), MSRA
- today — GO, Deepmind

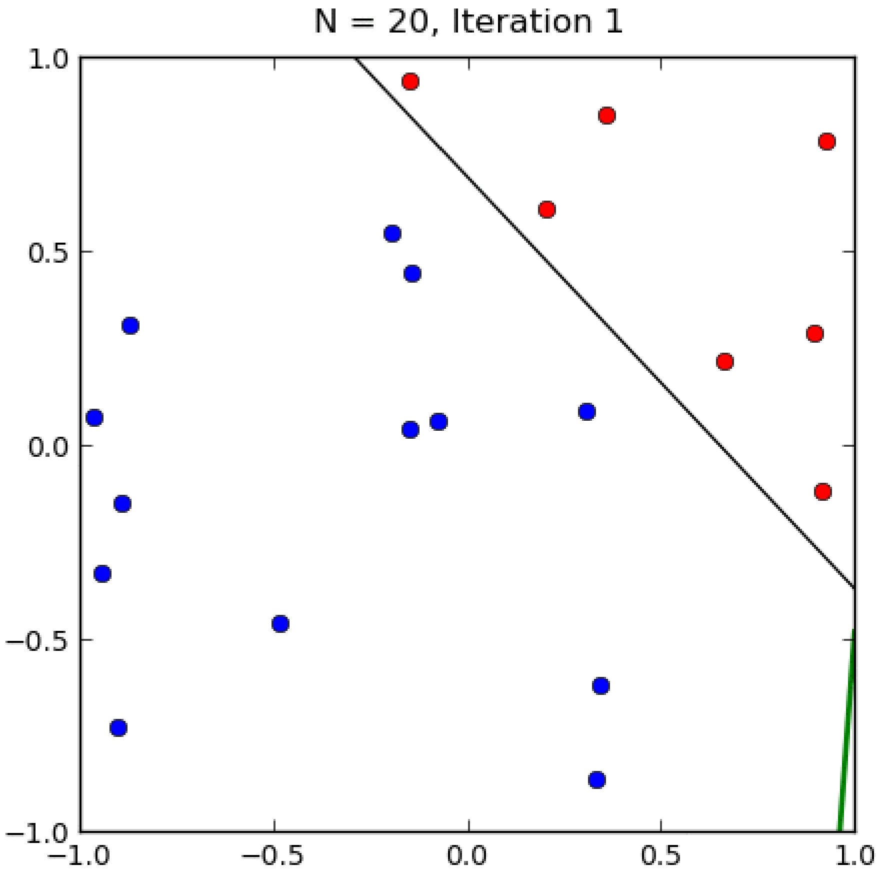
Perceptrons

- Rosenblatt proposed **Perceptrons** for binary classifications
  - One weight w_i per input x_i
  - Multiply weights with respective inputs and add bias x_0 =+1
  - If result larger than threshold return 1, otherwise 0



Training a perceptron

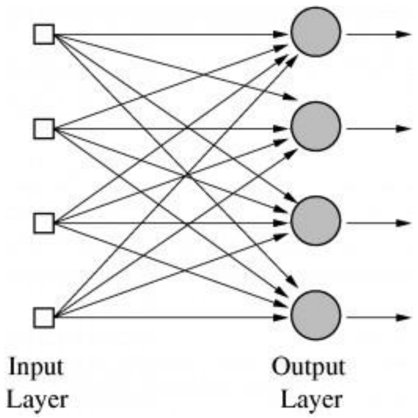
- Rosenblatt's innovation was mainly the learning algorithm for perceptrons
- Learning algorithm
  - Initialize weights randomly
  - Take one sample x_i and predict y_i
  - For erroneous predictions update weights
    - If prediction $\widehat{y}_i = 0$ and ground truth $y_i = 1$, increase weights
    - If prediction $\widehat{y}_i = 1$ and ground truth $y_i = 0$, decrease weights
  - Repeat until no errors are made



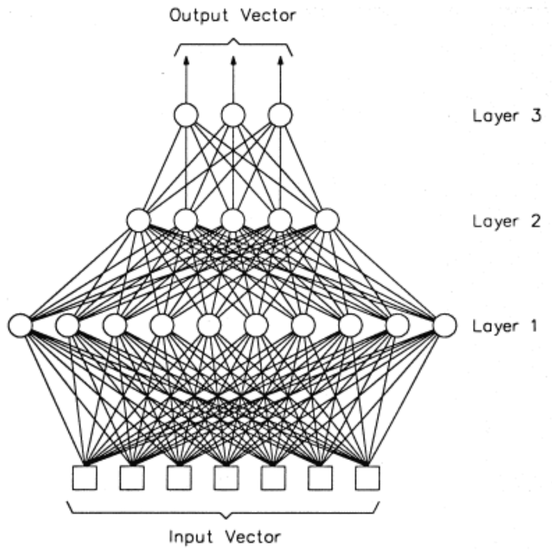
From a single layer to multiple layers

- 1 perceptron == 1 decision
- What about multiple decisions?
  - E.g. digit classification
- Stack as many outputs as the possible outcomes into a layer
  - Neural network
- Use one layer as input to the next layer
  - Add nonlinearities between layers
  - Multi-layer perceptron (MLP)

1-layer neural network



Multi-layer perceptron



What could be a problem with perceptrons?

A. They can only return one output, so only work for binary problems
B. They are linear machines, so can only solve linear problems
C. They can only work for vector inputs
D. They are too complex to train, so they can work with big computers only

The question will open when you start your session and slideshow.

Time: 60s

What could be a problem with perceptrons?

A. They can only return one output, so only work for binary problems

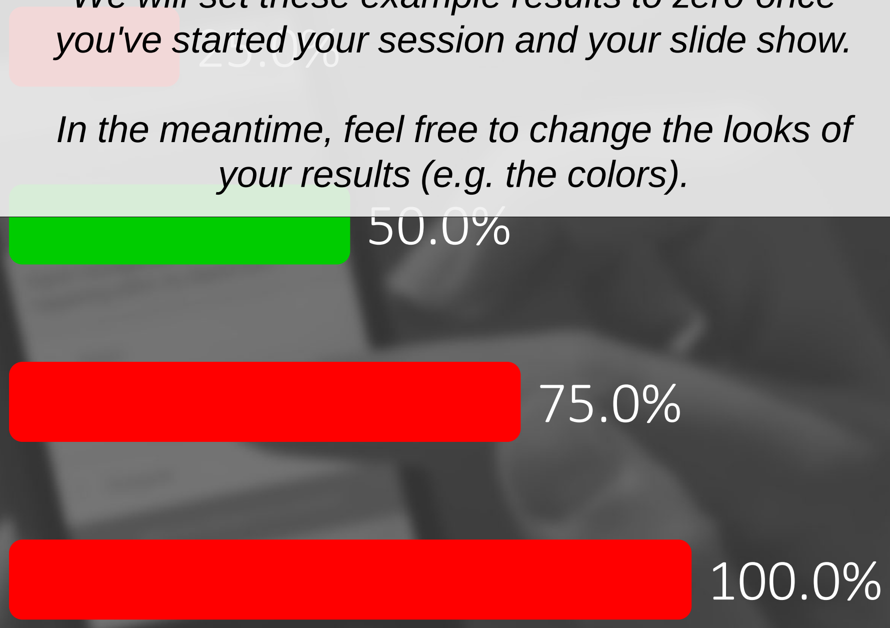
B. They are linear machines, so can only solve linear problems

C. They can only work for vector inputs

D. They are too complex to train, so they can work with big computers only

We will set these example results to zero once you've started your session and your slide show.

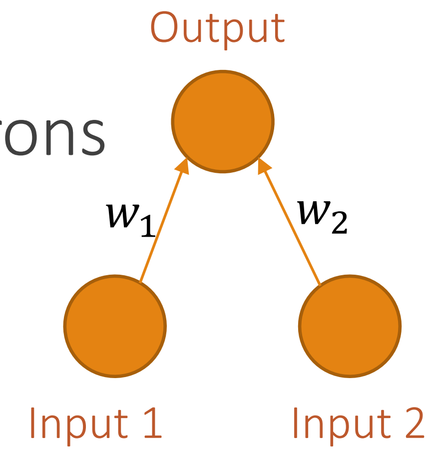
In the meantime, feel free to change the looks of your results (e.g. the colors).



Closed

shakespeak

XOR & Single-layer Perceptrons



- However, the exclusive or (XOR) cannot be solved by perceptrons
  - [Minsky and Papert, “Perceptrons”, 1969]

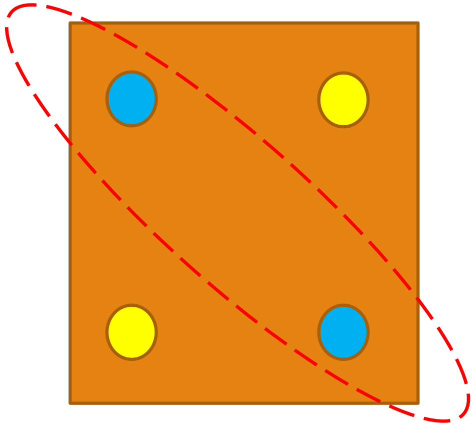
Input 1	Input 2	Output
1	1	0
1	0	1
0	1	1
0	0	0

- $0\,w_1 + 0w_2 < \theta \rightarrow 0 < \theta$
- $0\,w_1 + 1w_2 > \theta \rightarrow w_2 > \theta$
- $1\,w_1 + 0w_2 > \theta \rightarrow w_1 > \theta$
- $1\,w_1 + 1w_2 < \theta \rightarrow w_1 + w_2 < \theta$

Inconsistent!!

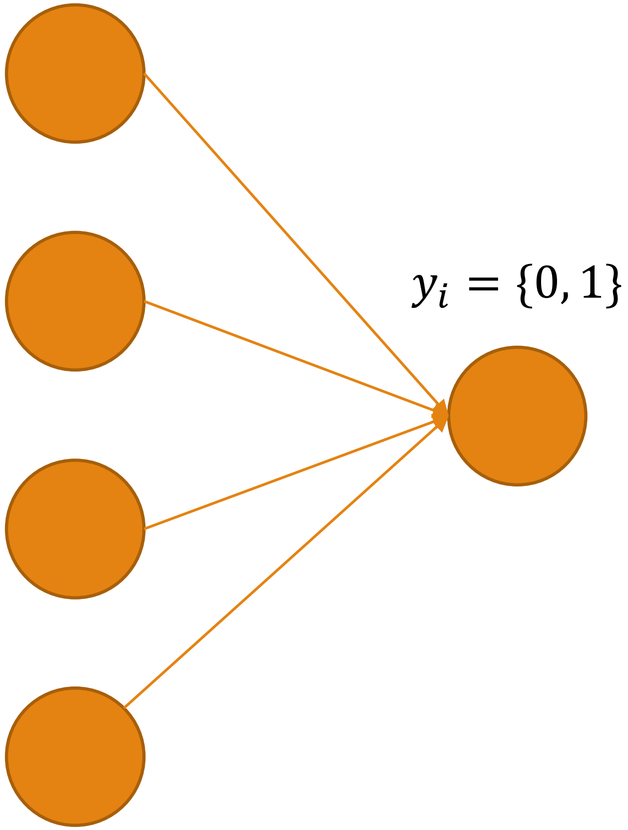
Graphically

The classification boundary to solve XOR is not a line!!



Minsky & Multi-layer perceptrons

- Interestingly, Minksy **never said** XOR cannot be solved by neural networks
  - Only that XOR cannot be solved with <u>**1 layer**</u> perceptrons
- <u>Multi</u>-layer perceptrons <u>can solve</u> XOR
  - 9 years earlier Minsky built such a multi-layer perceptron
- However, how to train a multi-layer perceptron?
- Rosenblatt's algorithm not applicable
  - It expects to know the desired target

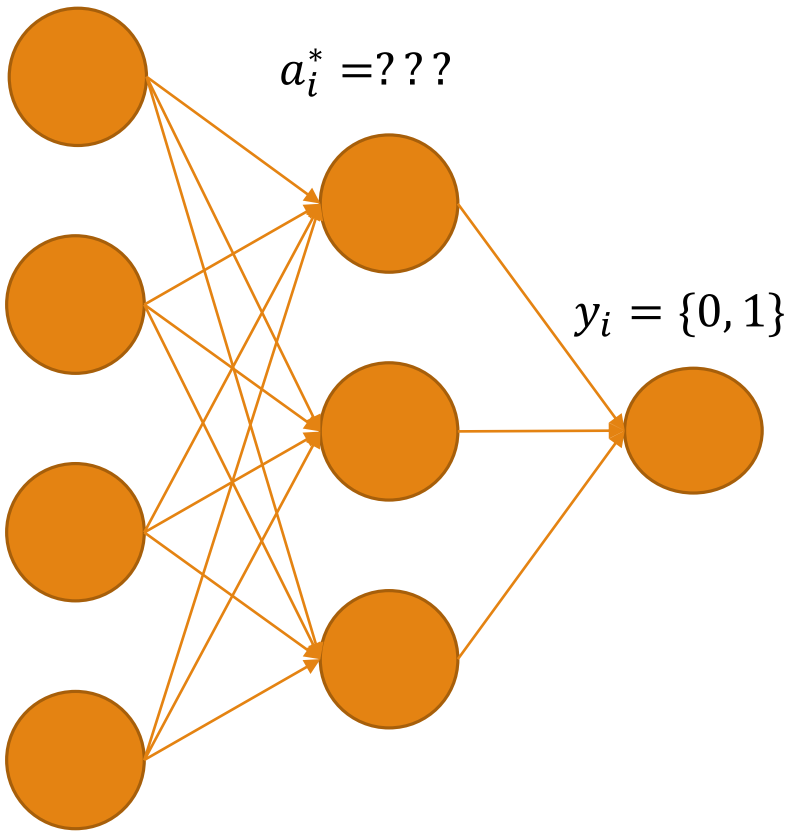


Minsky & Multi-layer perceptrons

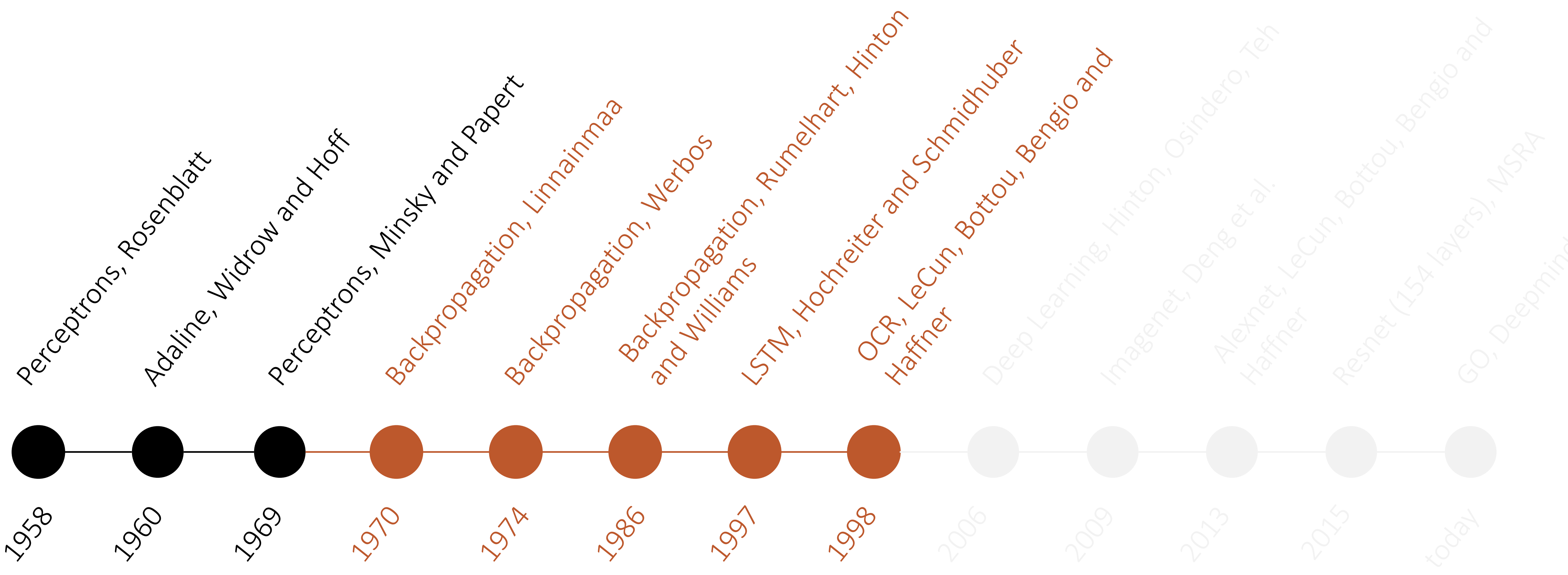
- Minksy **never said** XOR is unsolvable by multi-layer perceptrons

- <u>Multi</u>-layer perceptrons <u>can solve</u> XOR

- Problem: how to train a multi-layer perceptron?
  - Rosenblatt’s algorithm not applicable
  - It expects to know the ground truth a_i^* for a variable a_i
  - For the output layers we have the ground truth labels
  - For intermediate hidden layers we don’t



The “AI winter” despite notable successes



The first “AI winter”

- What everybody thought: “If a perceptron cannot even solve XOR, why bother?
- Results not as promised (too much hype!) → no further funding → **AI Winter**
- Still, significant discoveries were made in this period
  - Backpropagation → Learning algorithm for MLPs (Lecture 2)
  - Recurrent networks → Neural Networks for infinite sequences (Lecture 5)

The second "AI winter"

- Concurrently with Backprop and Recurrent Nets, new and promising Machine Learning models were proposed

- Kernel Machines & Graphical Models
  - Similar accuracies with better math and proofs and fewer heuristics
  - Neural networks could not improve beyond a few layers

Interim Announcements

- We have invited the PyTorch developers to give a tutorial on how to use PyTorch
- 3 slots
  - Tuesday (today), 11-13, Turingzaal
  - Tuesday (today), 15-17, C0.110
  - Wednesday (today), 15-17, C0.110
- Next Friday at the practical, 11-12, presentation by SURFSara
- If you are not an MSc student and you want to follow the course and get updates, send me an email to subscribe you

Prepare to vote

Internet

1 Go to shakespeak.me

2 Log in with uva507

This presentation has been loaded without the Shakespeak add-in.
Want to download the add-in for free? Go to http://shakespeak.com/en/free-download/.

TXT

1 Text to 06 4250 0030

2 Type uva507 <space> your choice (e.g. uva507 b)

Voting is anonymous

shakespeak

In this edition we will try for a more interactive course. Would you like to try this out?

A. Yes, why not?
B. Nope!
C. Yes, under conditions.

The question will open when you start your session and slideshow.

Time: 60s

shakespeak

Internet

TXT

Votes: 92

This presentation has been loaded without the Shakespeak add-in.
Want to download the add-in for free? Go to http://shakespeak.com/en/free-download/.

In this edition we will try for a more interactive course. Would you like to try this out?

A. Yes, why not? 79.3%

B. Nope! 0.0%

C. Yes, under conditions. 20.7%

Closed

shakespeak

The thaw of the "AI winter"

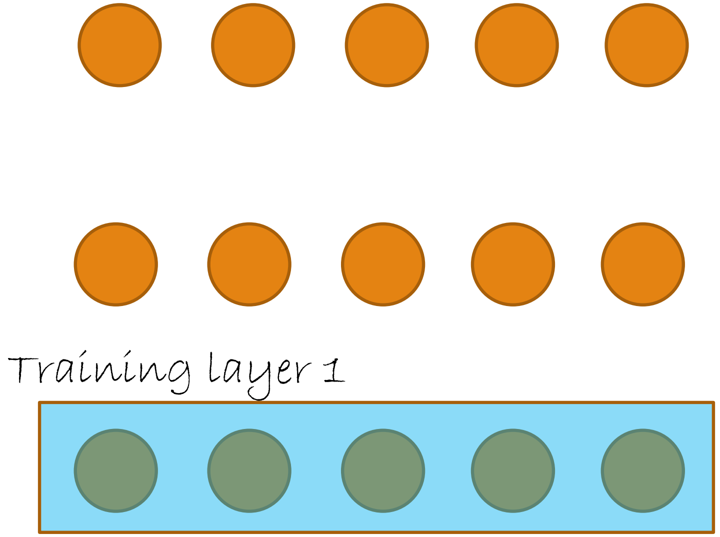
- 1958 — Perceptrons, Rosenblatt
- 1960 — Adaline, Widrow and Hoff
- 1969 — Perceptrons, Minsky and Papert
- 1970 — Backpropagation, Linnainmaa
- 1974 — Backpropagation, Werbos
- 1986 — Backpropagation, Rumelhart, Hinton and Williams
- 1997 — LSTM, Hochreiter and Schmidhuber
- 1998 — OCR, LeCun, Bottou, Bengio and Haffner
- 2006 — Deep Learning, Hinton, Osindero, Teh
- 2009 — Imagenet, Deng et al.
- 2013 — Alexnet, LeCun, Bottou, Bengio and Haffner
- 2015 — Resnet (154 layers), MSRA
- today — GO, Deepmind

Neural Network problems a decade ago

- Lack of processing power
- Lack of data
- Overfitting
- Vanishing gradients
- Experimentally, training multi-layer perceptrons was not that useful
  - Accuracy didn't improve with more layers
  - Are 1-2 hidden layers the best neural networks can do?

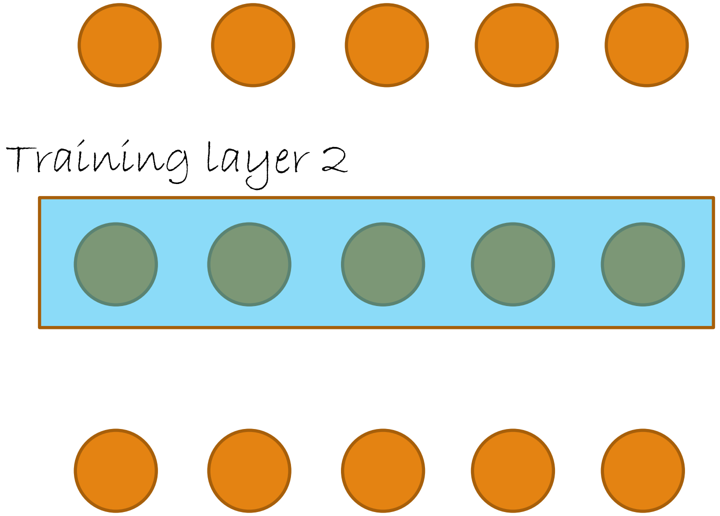
Deep Learning arrives

- Layer-by-layer training
  - The training of each layer individually is an easier undertaking
- Training multi-layered neural networks became easier
- Per-layer trained parameters initialize further training using contrastive divergence



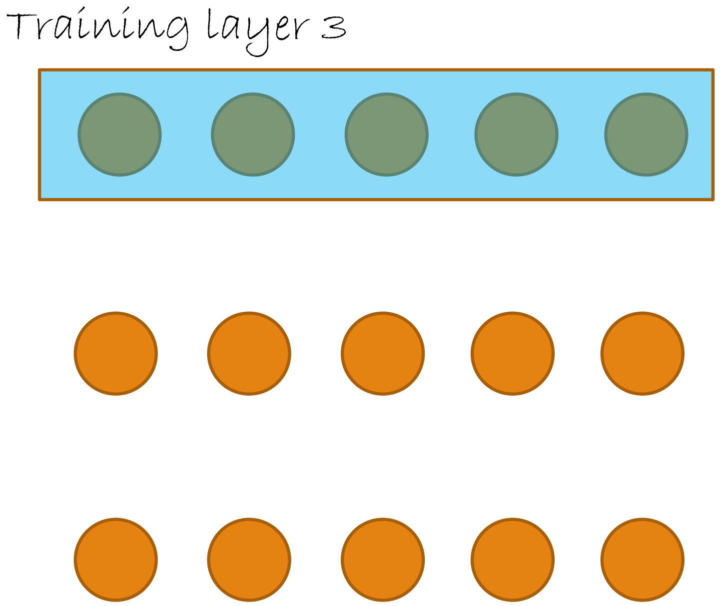
Deep Learning arrives

- Layer-by-layer training
  - The training of each layer individually is an easier undertaking
- Training multi-layered neural networks became easier
- Per-layer trained parameters initialize further training using contrastive divergence



Deep Learning arrives

- Layer-by-layer training
  - The training of each layer individually is an easier undertaking
- Training multi-layered neural networks became easier
- Per-layer trained parameters initialize further training using contrastive divergence



Deep Learning Renaissance

Perceptrons, Rosenblatt	1958
Adaline, Widrow and Hoff	1960
Perceptrons, Minsky and Papert	1969
Backpropagation, Linnainmaa	1970
Backpropagation, Werbos	1974
Backpropagation, Rumelhart, Hinton and Williams	1986
LSTM, Hochreiter and Schmidhuber	1997
OCR, LeCun, Bottou, Bengio and Haffner	1998
Deep Learning, Hinton, Osindero, Teh	2006
Imagenet, Deng et al.	2009
Alexnet, LeCun, Bottou, Bengio and Haffner	2013
Resnet (154 layers), MSRA	2015
GO, Deepmind	today

Alexnet architecture

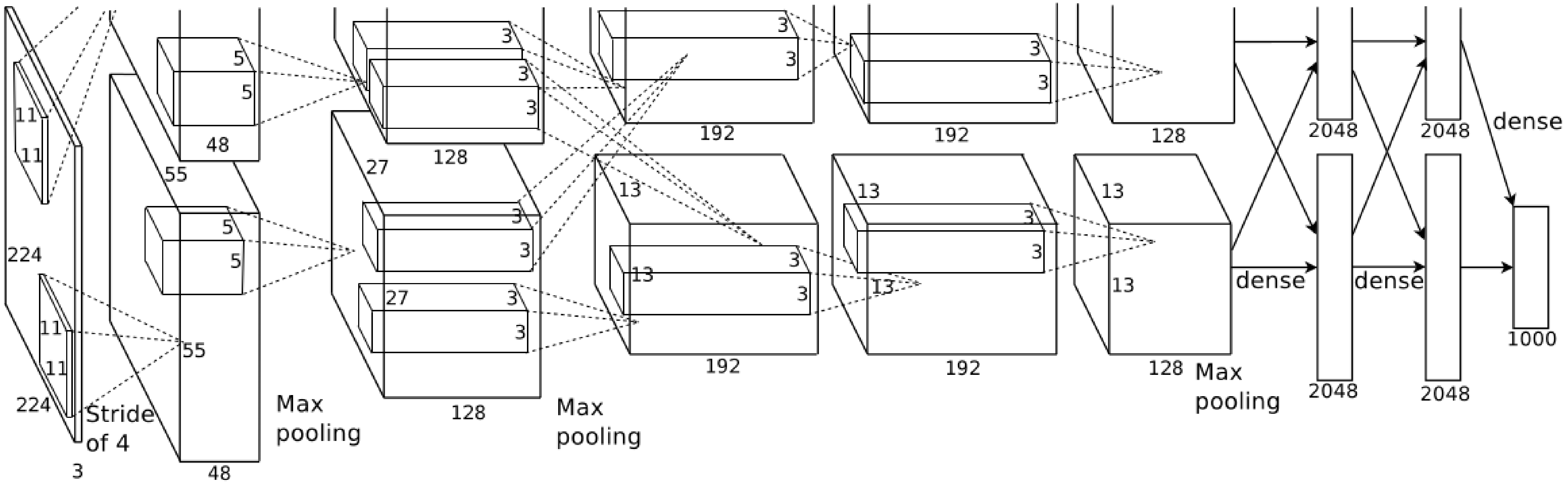


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

Deep Learning is **Big Data Hungry!**

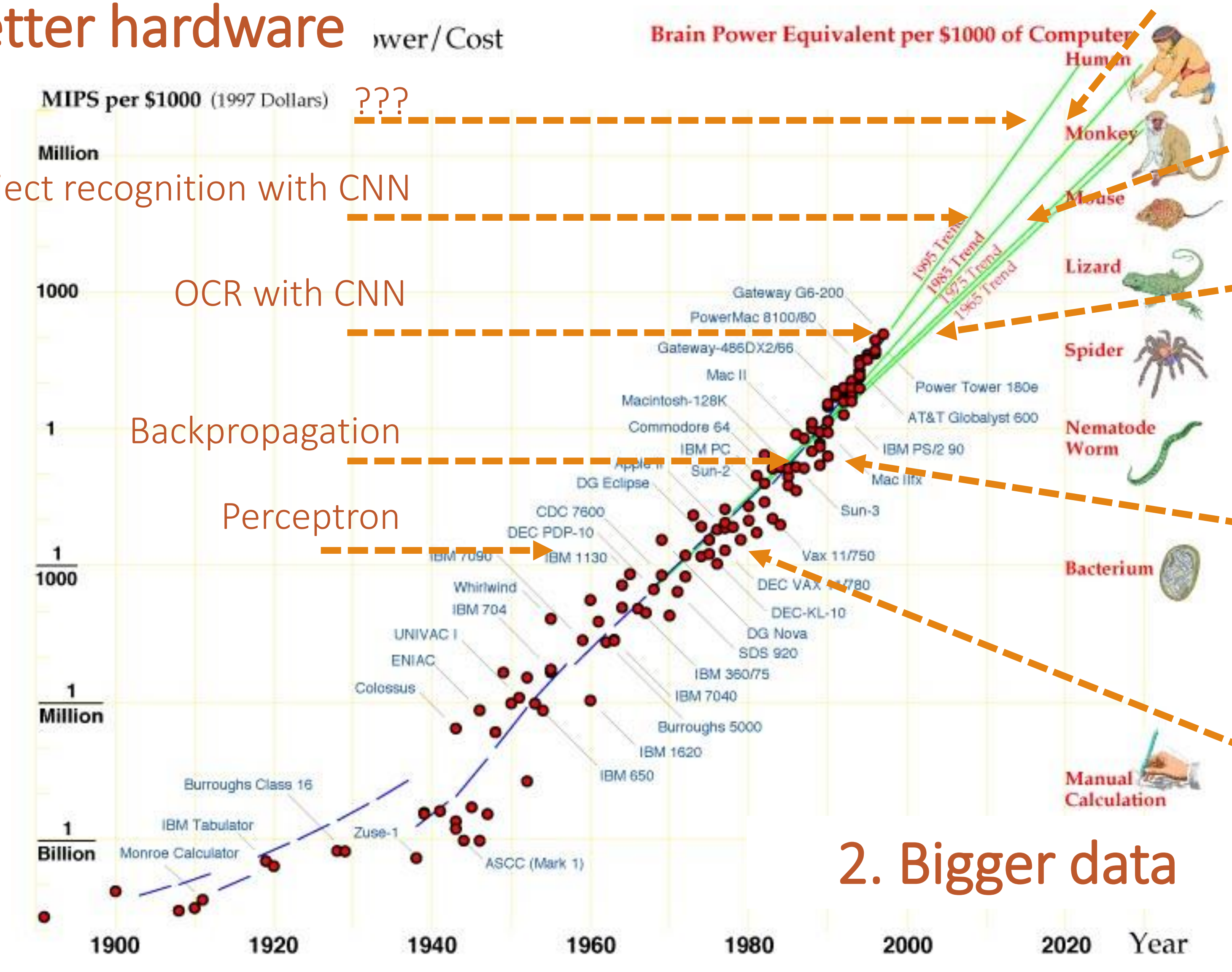
- In 2009 the Imagenet dataset was published [Deng et al., 2009]
  - Collected images for each of the 100K terms in Wordnet (16M images in total)
  - Terms organized hierarchically: “Vehicle”→“Ambulance”

- Imagenet Large Scale Visual Recognition Challenge (ILSVRC)
  - 1 million images
  - 1,000 classes
  - Top-5 and top-1 error measured

Why now?

Datasets of everything (captions, question-answering, …), reinforcement learning, ???

1. Better hardware



???

Object recognition with CNN

OCR with CNN

Backpropagation

Perceptron

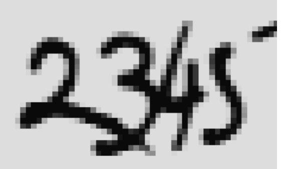
2. Bigger data

Imagenet: 1,000 classes from real images, 1,000,000 images

Results:

- Persian cat: 0.35211
- Egyptian cat: 0.23635
- hamster: 0.20282
- tiger cat: 0.05896
- lynx: 0.05759

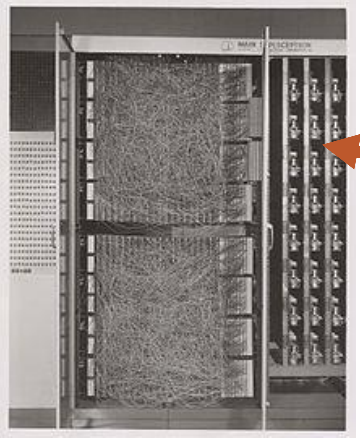
Bank cheques



Parity, negation problems

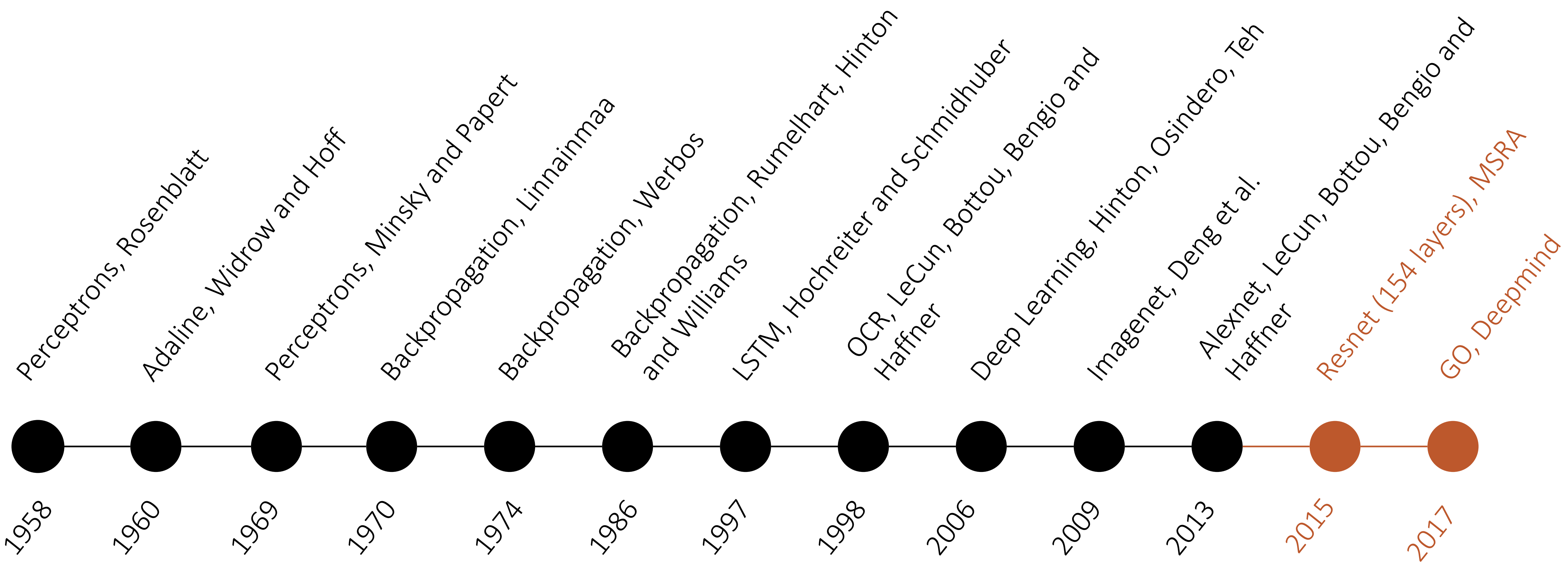
D1	D2	D3	Even-Parity
0	0	0	True
0	0	1	False
0	1	0	False
0	1	1	True
1	0	0	False
1	0	1	True
1	1	0	True
1	1	1	False

Mark I Perceptron

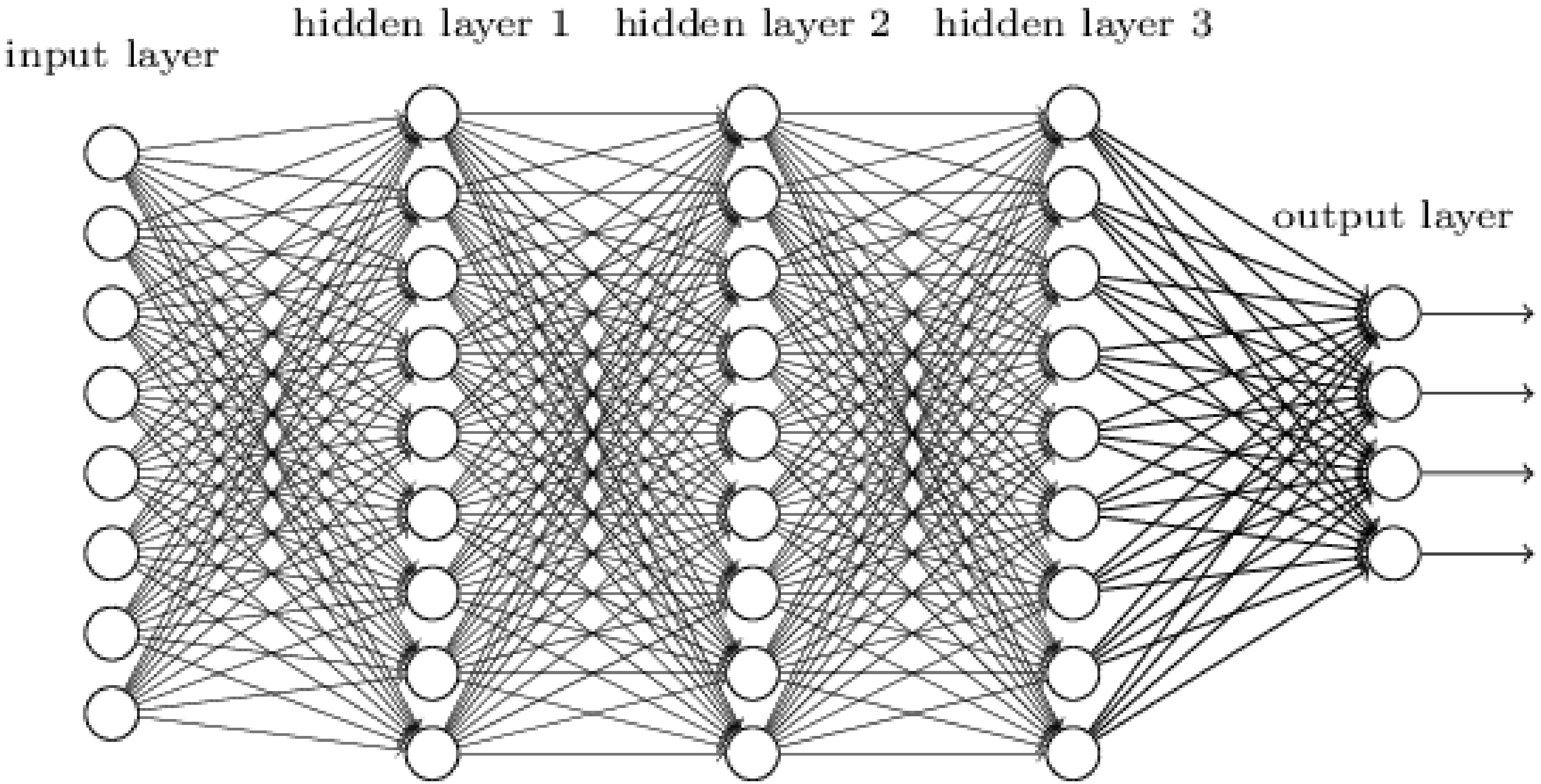


Potentiometers implement perceptron weights

Deep Learning Golden Era



Deep Learning: The *What* and *Why*



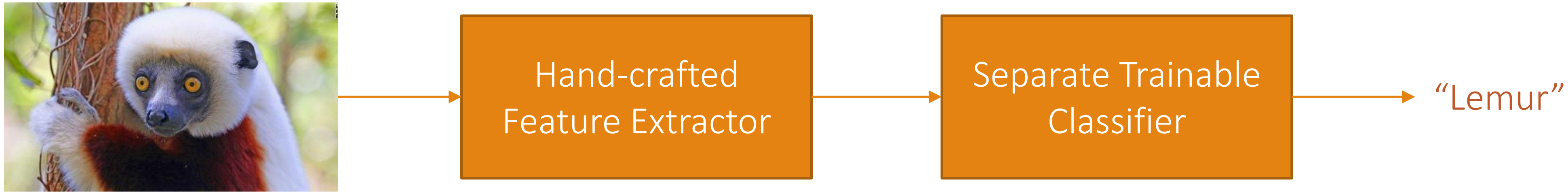
Long story short

- A family of parametric, non-linear and hierarchical representation learning functions, which are massively optimized with stochastic gradient descent to encode domain knowledge, i.e. domain invariances, stationarity.
- $a_L(x; \theta_{1,\ldots,\mathrm{L}}) = h_L\, (h_{L-1}(\ldots h_1(x, \theta_1), \theta_{L-1}),\ \theta_L)$
  - x:input, θ_l: parameters for layer l, $a_l = h_l(x, \theta_l)$: (non-)linear function
- Given training corpus $\{X, Y\}$ find optimal parameters

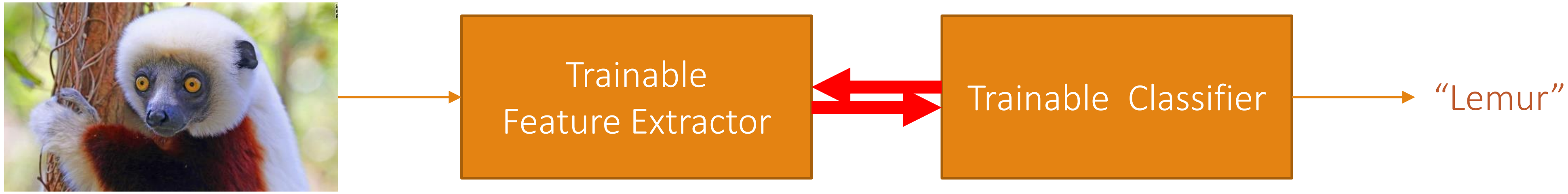
$$\theta^* \leftarrow \arg\min_\theta \sum_{(x,y)\subseteq(X,Y)} \ell(y, a_L(x; \theta_{1,\ldots,\mathrm{L}}))$$

Learning Representations & Features

- Traditional pattern recognition

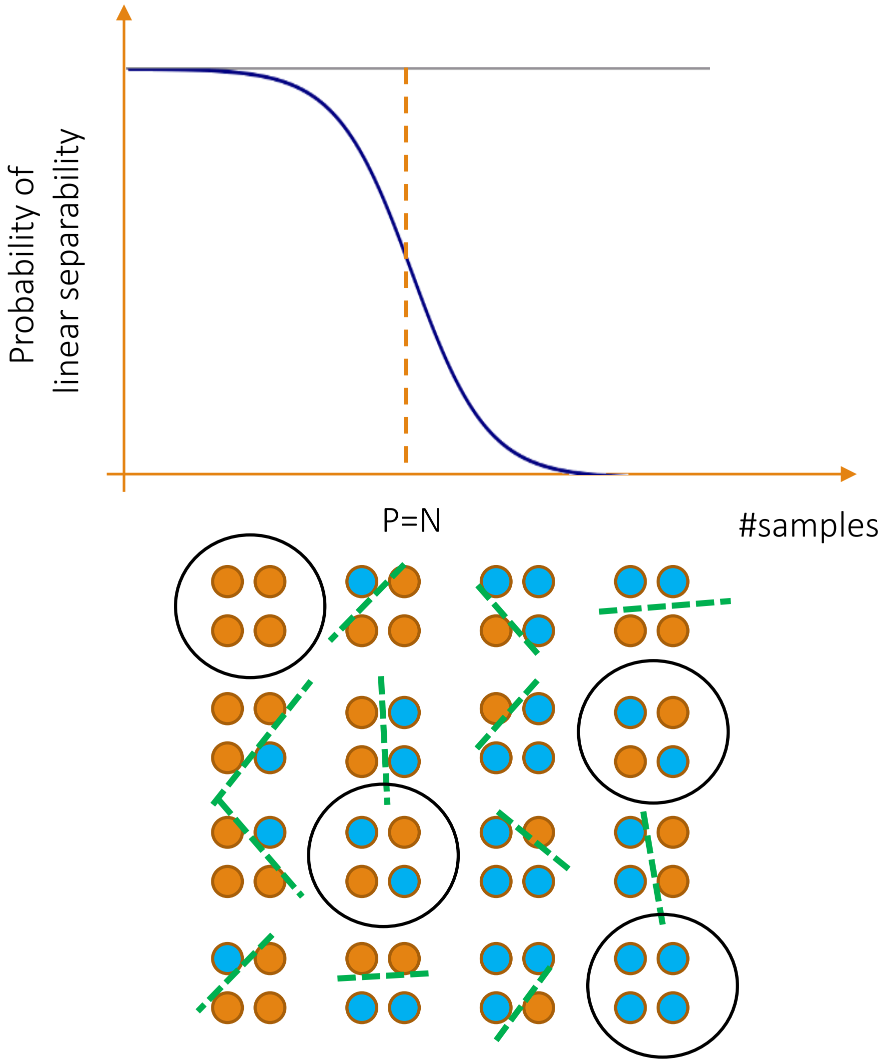


- End-to-end learning → Features are also learned from data



Non-separability of linear machines

- $X = \{x_1, x_2, \dots, x_n\} \in \mathcal{R}^d$
- Given the n points there are in total 2^n dichotomies
- Only about d are linearly separable
- With $n > d$ the probability X is linearly separable converges to 0 very fast
- The chances that a dichotomy is linearly separable is very small



How can we solve the non-separability of linear machines?

A. Apply SVM
B. Use non-linear features
C. Use non-linear kernels
D. Use advanced optimizers, like Adam or Nesterov's Momentum

The question will open when you start your session and slideshow.

Time: 60s

How can we solve the non-separability of linear machines?

A.	Apply SVM	6.1%
B.	Use non-linear features	24.4%
C.	Use non-linear kernels	69.5%
D.	Use advanced optimizers, like Adam or Nesterov's Momentum	0.0%

shakespeak

Closed

Non-linearizing linear machines

- Most data distributions and tasks are non-linear
- A linear assumption is often convenient, but not necessarily truthful
- **Problem:** How to get non-linear machines without too much effort?

Non-linearizing linear machines

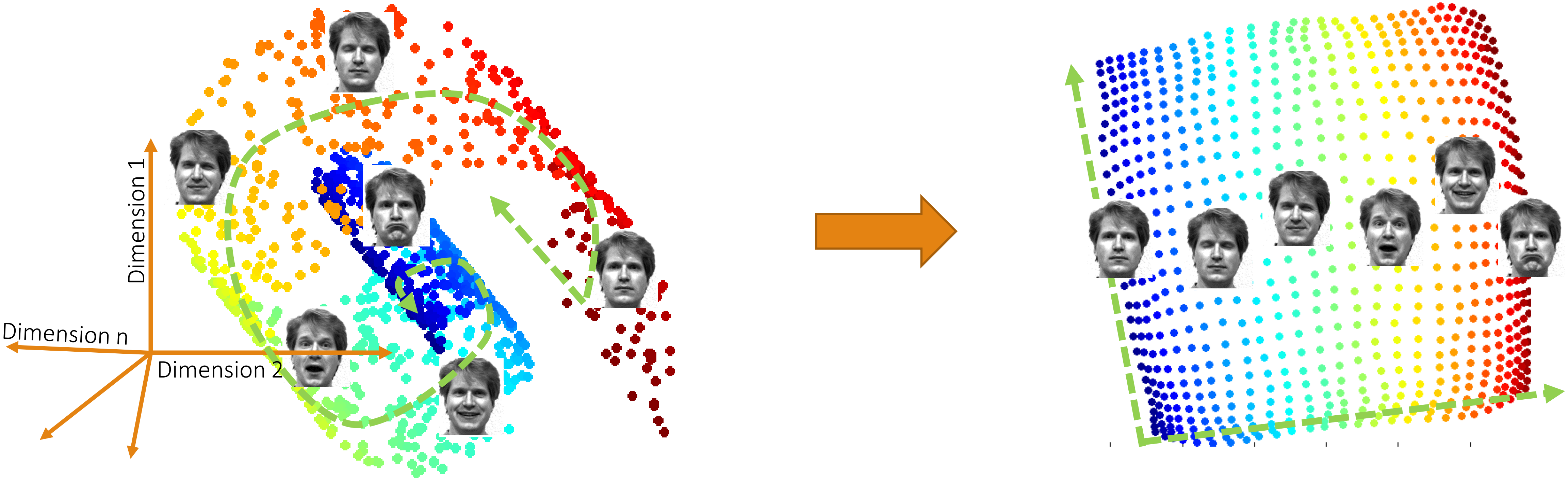
- Most data distributions and tasks are non-linear
- A linear assumption is often convenient, but not necessarily truthful
- **Problem:** How to get non-linear machines without too much effort?
- **Solution:** Make features non-linear
- What is a good non-linear feature?
  - Non-linear kernels, e.g., polynomial, RBF, etc
  - Explicit design of features (SIFT, HOG)?

Good features

- Invariant … but not too invariant
- Repeatable … but not bursty
- Discriminative … but not too class-specific
- Robust … but sensitive enough

Manifolds

- Raw data live in huge dimensionalities
- But, effectively lie in lower dimensional manifolds
- Can we discover this manifold to embed our data on?



How to get good features?

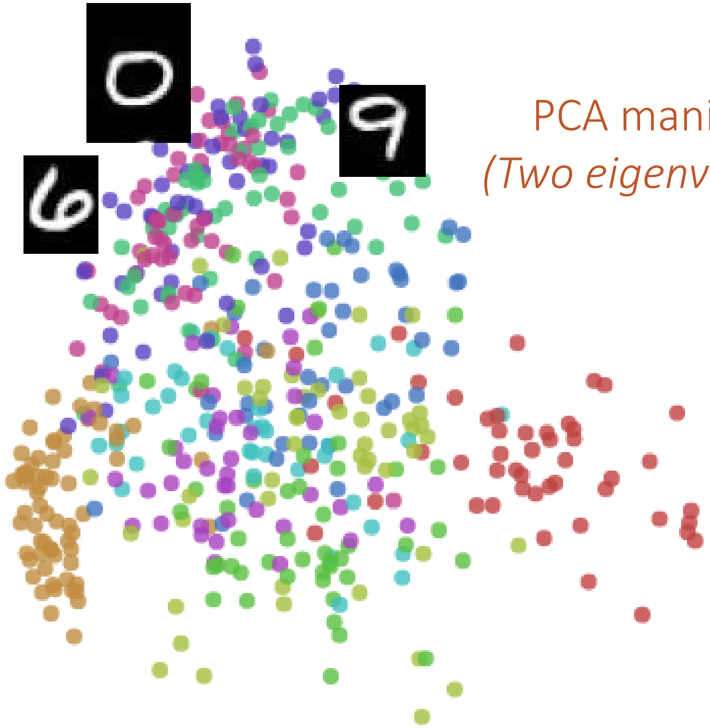
- **Goal:** discover these lower dimensional manifolds
  - These manifolds are most probably highly non-linear

- **First hypothesis:** Semantically similar things lie closer together than semantically dissimilar things

- **Second hypothesis:** A face (or any other image) is a point on the manifold
  → Compute the coordinates of this point and use them as a feature
  → Face features will be separable

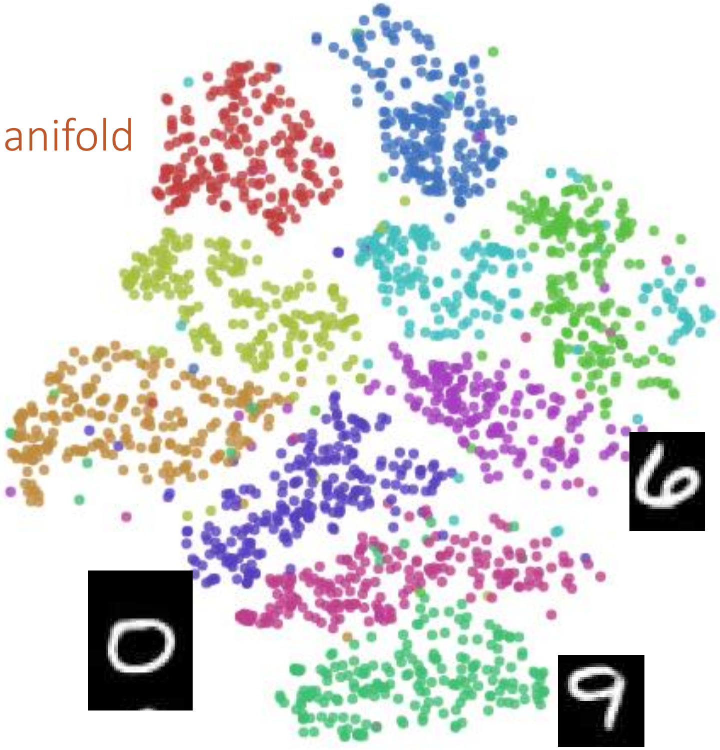
The digits manifolds

- There are good features (manifolds) and bad features
- 28 pixels x 28 pixels = 784 dimensions



PCA manifold
(Two eigenvectors)

t-SNE manifold

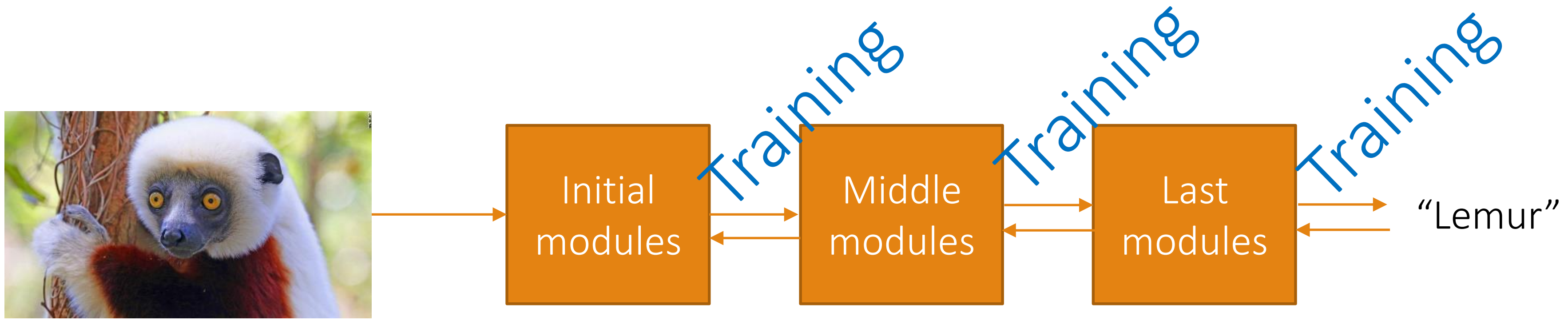


End-to-end learning of feature hierarchies

- A pipeline of successive, differentiable modules
  - Each module's output is the input for the next module

- Each subsequent module produce higher abstraction features

- Preferably, input <u>as raw as possible</u>



Why learn the features and not just design them?

A. Designing features manually is too time consuming and requires expert knowledge
B. Learned features give us a better understanding of the data
C. Learned features are more compact and specific for the task at hand
D. Learned features are easy to adapt
E. Features can be learnt in a plug-n-play fashion, ease for the layman

The question will open when you start your session and slideshow.

Time: 60s

Why learn the features and not just design them?

A. Designing features manually is too time consuming and requires expert knowledge — 48.1%

B. Learned features give us a better understanding of the data — 13.6%

C. Learned features are more compact and specific for the task at hand — 28.4%

D. Learned features are easy to adapt — 8.6%

E. Features can be learnt in a plug-n-play fashion, ease for the layman — 1.2%

Closed

Why learn the features?

- Manually designed features
  - Expensive to research & validate

- Learned features
  - If data is enough, easy to learn, compact and specific

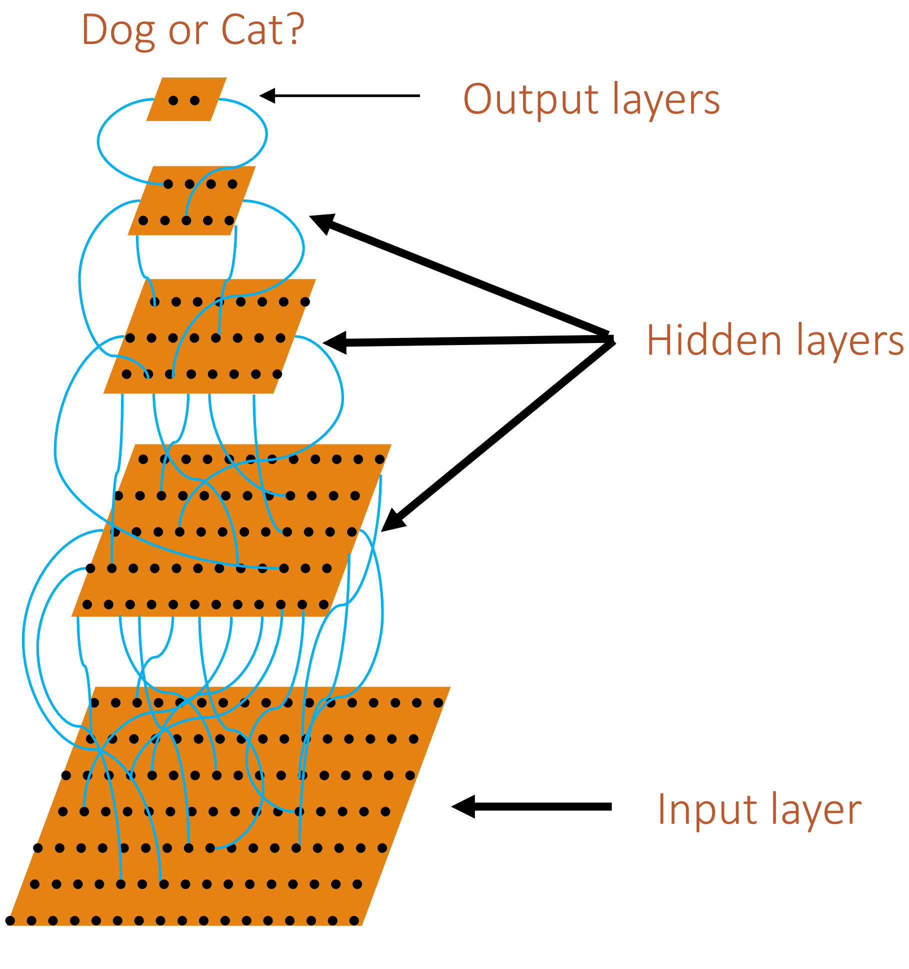
- Time spent for designing features now spent for designing architectures

Types of learning

- Supervised learning, e.g. Convolutional Networks

Convolutional networks

Is this a dog or a cat?

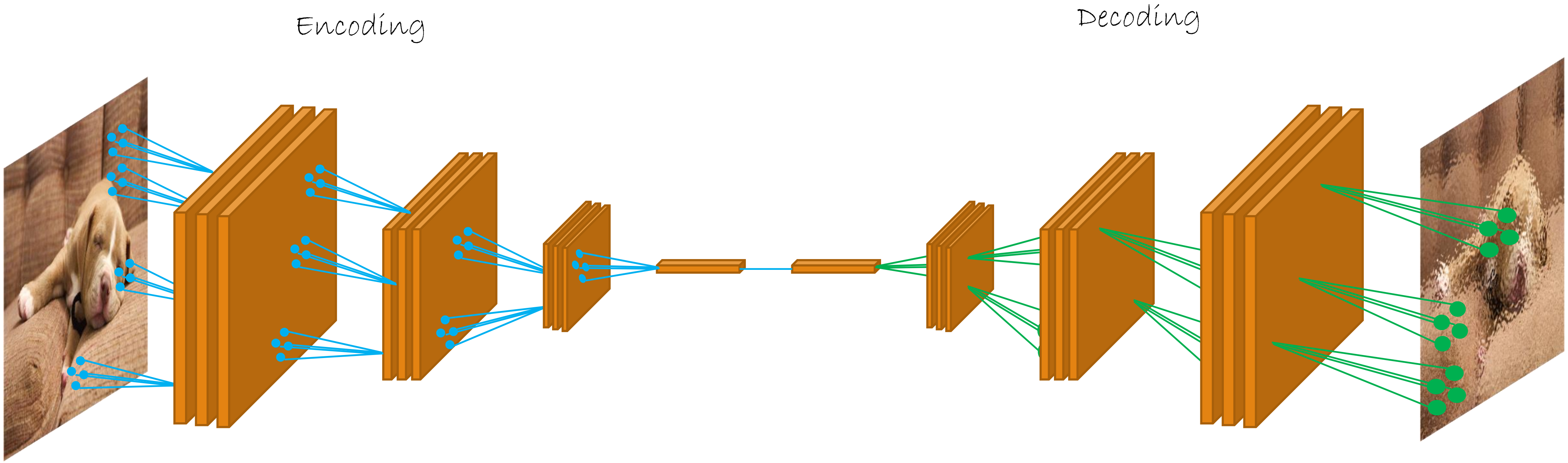


Types of learning

- Supervised learning, e.g. Convolutional Networks

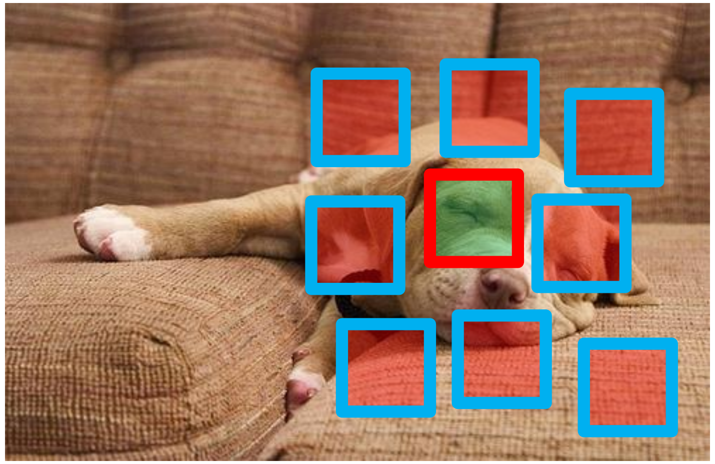
- Unsupervised learning, e.g. Autoencoders

Autoencoders



Types of learning

- Supervised learning, e.g. Convolutional Networ



- Unsupervised learning, e.g. Autoencoders
- Self-supervised learning
  - A mix of supervised and unsupervised learning

Types of learning

- Supervised learning, e.g. Convolutional Networks
- Unsupervised learning, e.g. Autoencoders
- Self-supervised learning
  - A mix of supervised and unsupervised learning
- Reinforcement learning
  - Agent perform actions in an environment and gets rewards

Philosophy of the course

The bad news ☹

- We only have 2 months = 14 lectures
- Lots of material to cover
- Hence, no time to lose
  - Basic neural networks, learning PyTorch, learning to program on a server, advanced optimization techniques, convolutional neural networks, recurrent neural networks, generative models
- This course is hard
  - But is optional
  - From previous student evaluations, it has been very useful for everyone

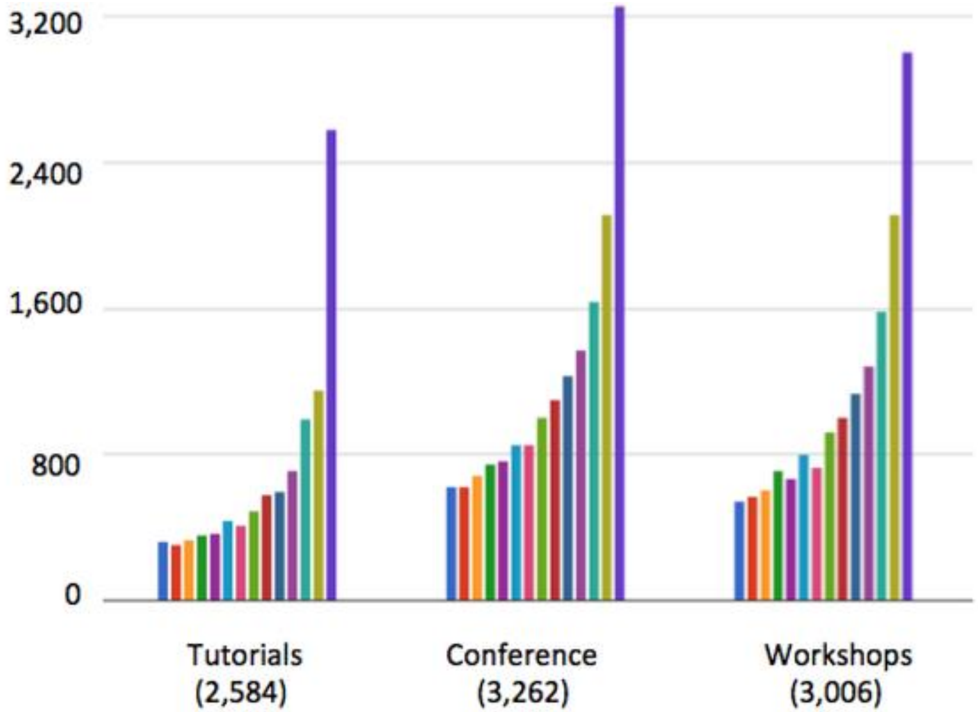
The good news ☺

- We are here to help
  - Last year we got a great evaluation score, so people like it and learn from it
- We have agreed with SURF SARA to give you access to the Dutch Supercomputer Cartesius with a bunch of (very) expensive GPUs
- You'll get to know some of the hottest stuff in AI today
- You'll get to present your own work to an interesting/ed crowd

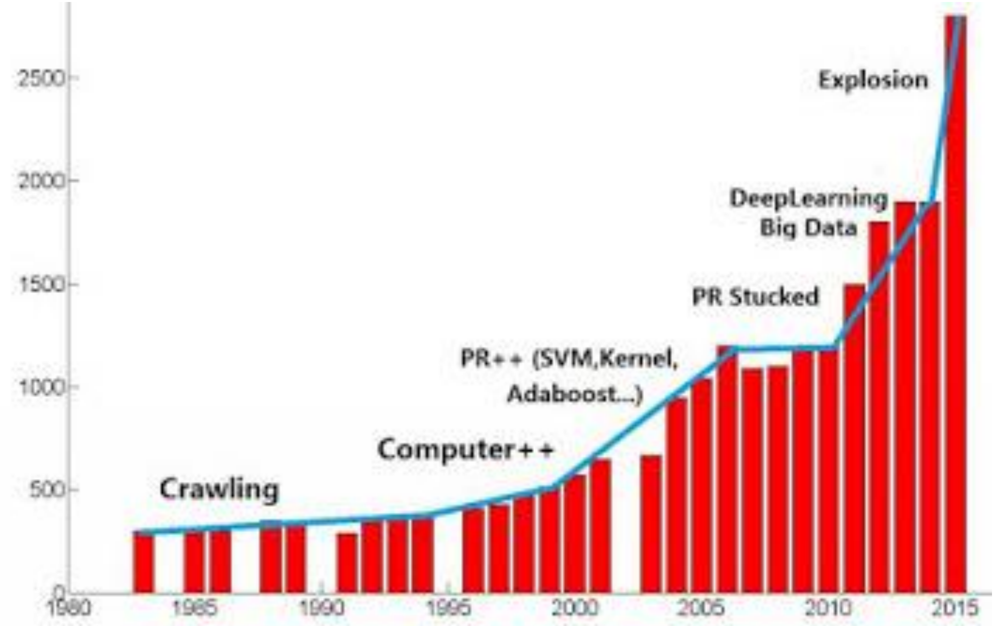
The good news ☺

- You'll get to know some of the hottest stuff in AI today
  - in academia

NIPS



CVPR



The good news ☺

- You will get to know some of the hottest stuff in AI today
  - in academia & in industry

The even better news ☺☺☺

- In the end of the course we might give a few MSc Thesis Projects in collaboration with Qualcomm/QUVA Lab
  - Students will become interns in the QUVA lab and get paid during thesis
- Requirements
  - Work hard enough and be motivated
  - Have top performance in the class
  - And interested in working with us
- Come and find me later

Code of conduct

- We encourage you to help each other, actively participate, give feedback
  - 3 students with **highest participation** in Q&A in Piazza get **+0.5 grade**
  - Your grade depends on **<u>what you do</u>**, not what others do
  - You have plenty of chances to collaborate for your poster and paper presentation
- However, we do not tolerate **blind** copy
  - Not from each other
  - Not from the internet
  - We use TurnitIn for plagiarism detection

Summary

- A brief history of Deep Learning
- Why is Deep Learning happening now?
- What types of Deep Learning exist?

Reading material

- http://www.deeplearningbook.org/
- Chapter 1: Introduction, p.1-28

Also, enroll in Deep Vision Seminars

Next lecture

- Neural networks as layers and modules
- Build your own modules
- Backprop
- Stochastic Gradient Descend