

Deep Learning

Lecture 1 – Introduction

Sebastian Bordt

Theory of Machine Learning Group

University of Tübingen

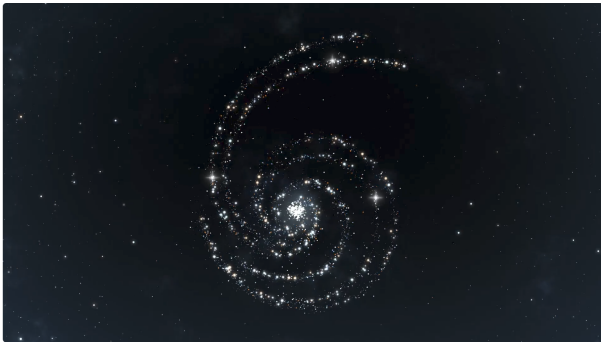
EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN



Tübingen AI Center
tuebingen.ai



e l i a s
European Laboratory for Learning and Intelligent Systems



GPT-6 Astra: A new generation of intelligence

Research 28 min read



The Hugging Face incident and the road ahead

Security 22 min read





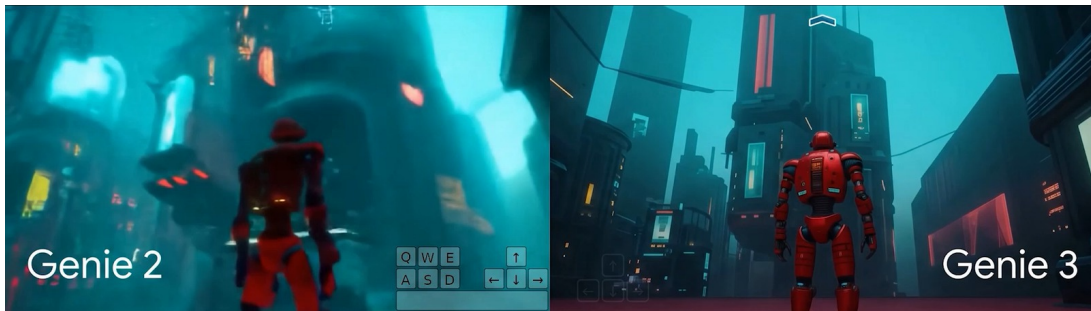
Deep Learning for Images and Video



create an image of a lecture hall at the university of tübingen with students who want to learn about deep learning

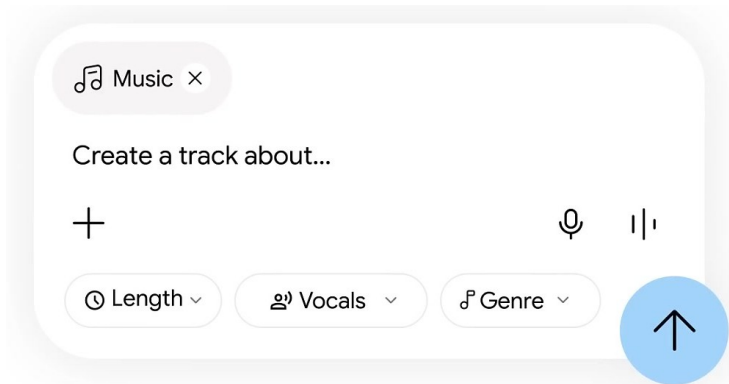


The same prompt with Stable Diffusion 3.5 Large



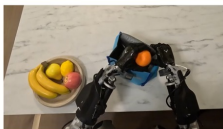
A text prompt becomes an interactive 3D world

Deep Learning for Audio

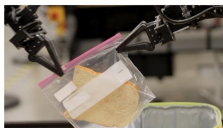


Describe a song in a sentence

Deep Learning for Robotics



Apollo



Aloha2



Bi-arm Franka

Robot arms follow instructions in natural language



“put the watering can into the green bin”

Deep Learning for Science

FINITE TIME BLOWUP FOR NAVIER-STOKES

OPENAI

ABSTRACT. For every positive viscosity, we construct a solution of the three-dimensional incompressible Navier–Stokes equations that starts from rest and develops unbounded velocity in finite time while maintaining uniformly bounded kinetic energy.

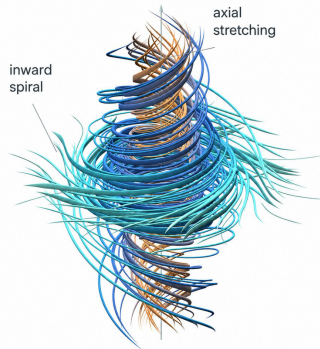
Theorem 1.1. For every $\nu > 0$ there exist a force $f \in C_c^\infty(\mathbb{R}^3 \times (0, \infty); \mathbb{R}^3)$, a compact set $K \subset \mathbb{R}^3$, and smooth velocity and pressure fields u, p on $\mathbb{R}^3 \times [0, 1)$ satisfying

$$\partial_t u + (u \cdot \nabla)u - \nu \Delta u + \nabla p = f, \quad \nabla \cdot u = 0, \quad u(\cdot, 0) = 0, \quad (1.1)$$

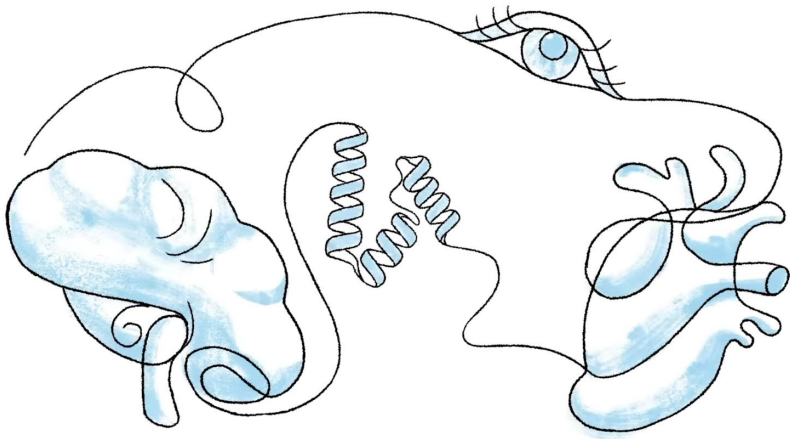
such that $\text{supp } u(\cdot, t) \cup \text{supp } p(\cdot, t) \subset K$ for every $0 \leq t < 1$,

$$\sup_{0 \leq t < 1} \|u(t)\|_{L^2(\mathbb{R}^3)} < \infty, \quad \limsup_{t \uparrow 1} \|u(t)\|_{L^\infty(\mathbb{R}^3)} = \infty.$$

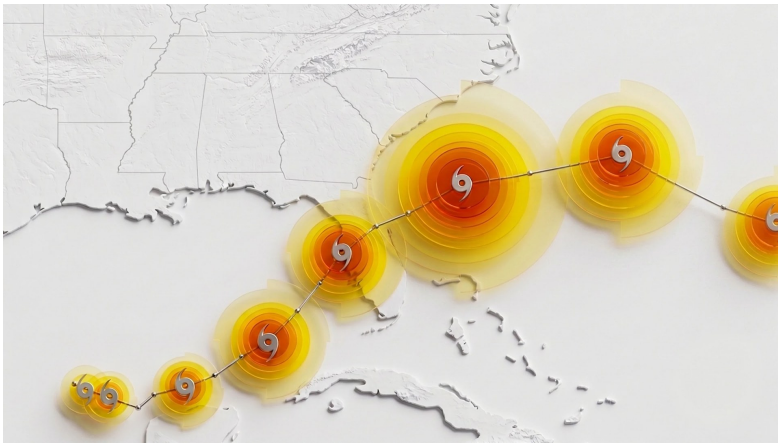
Consequently, there is no smooth solution (u, p) on $\mathbb{R}^3 \times [0, \infty)$ with the same force and initial datum whose kinetic energy is uniformly bounded $\sup_{t \geq 0} \frac{1}{2} \int_{\mathbb{R}^3} |u(x, t)|^2 dx < \infty$.



According to OpenAI, 10,000 AI agents worked on this for 88 hours



AlphaFold predicts the 3D structure of a protein from its amino-acid sequence



Hurricane forecasts

What will you learn in this lecture?

- ▶ This lecture is about deep learning, the technology behind the showcased AI applications.
- ▶ Deep learning is a **flexible** and **scalable** recipe for fitting a function to data by gradient descent.
- ▶ In deep learning, we learn a differentiable **neural network** from **data**, using a **loss function** and an **optimization algorithm**.
- ▶ The gradients of the neural network are computed with **backpropagation**.
- ▶ In this lecture, you will learn about these individual components, and how they fit together in a language model or image generator.



“Deep learning is just a buzzword for neural nets, and neural nets are just a stack of matrix-vector multiplications, interleaved with some non-linearities. No magic there.”

Ronan Collobert (Torch), 2011

Agenda

1.1 Organization

1.2 History of Deep Learning

1.3 Machine Learning Basics

1.1

Organization

Team



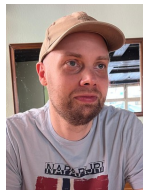
Dr.
Sebastian Bordt



Dr.
Yong Cao



Naama
Pearl



Glenn
Angrabeit

- ▶ **Lecture:** Sebastian Bordt
- ▶ **Tutorials and exercises:** Yong Cao, Naama Pearl, Glenn Angrabeit

Contents

Goal: Students gain an understanding of the theoretical and practical concepts of deep neural networks including, optimization, inference, architectures and applications. After this course, students should be able to develop and train deep neural networks, reproduce research results and conduct original research.

- ▶ History of deep learning
- ▶ Linear/logistic regression
- ▶ Multi-layer perceptrons
- ▶ Backpropagation
- ▶ Loss and Activation Functions
- ▶ Optimization and Regularization
- ▶ Convolutional Neural Networks
- ▶ Sequence Models
- ▶ Natural Language Processing
- ▶ Transformers
- ▶ Autoencoders
- ▶ Diffusion Models

Related Lectures

- ▶ Introduction to ML (BSc): 4 lectures on DL basics, lighter treatment
- ▶ Mathematics for ML: Repetition of BSc math and probability / statistics
- ▶ Mathematics for DL (online): 4h, only the basics necessary to follow this course
- ▶ Data Literacy: Hands-on data science, classic tools, complementary to DL
- ▶ Statistical ML: Little overlap (e.g., PCA), theoretical focus
- ▶ Probabilistic ML: 3 lectures on DL, but probabilistic and more theoretical focus
- ▶ Reinforcement Learning: Builds upon deep learning, modern RL applications
- ▶ Computer Vision: Builds upon deep learning, modern CV applications
- ▶ Self-Driving Cars: Builds upon deep learning, modern SDC applications

Organization

Organization

- ▶ **Lecture** (Wednesdays: 14:15–15:45, HS N05, Hörsaalzentrum Morgenstelle)
 - ▶ Held in person, not hybrid and not recorded
 - ▶ Ask (many) questions during the lecture (dare to ask!)
- ▶ **Tutorial** (Wednesdays: 16:15–17:45, same room)
 - ▶ Exercise hand-out, joint problem solving and discussion of solutions
- ▶ **No lecture and no tutorial** on 18.11.2026 and 09.12.2026
- ▶ We further provide:
 - ▶ Practice via **exercise sheets** with solutions (solve them!)
 - ▶ Individual help via the **helpdesk** (make use of it!)
 - ▶ Fast feedback via the **Ilias forum** (ask and reply!)
- ▶ **Exam:** Written, at the end of the semester
- ▶ **Register in Ilias today** – needed for announcements and the quizzes
- ▶ **Course Website** (Ilias link, slides, exercise sheets):

https://www.tml.cs.uni-tuebingen.de/teaching/2026_deep_learning/

Recordings

- ▶ This lecture is **not recorded**
- ▶ There are recordings of the **Deep Learning** lecture by **Andreas Geiger** (winter term 2020/21)
<https://www.youtube.com/playlist?list=PL05umP7R6ij3NTWIdtMbfvX7Z-4WEXRqD>
- ▶ Up to **Convolutional Neural Networks** (02.12.): similar material, **but not the same!**
- ▶ After that, the content of this year's lecture is **completely different**
- ▶ **We recommend that you attend the lecture**
- ▶ If this is not possible: the recordings can be useful in the **first half** of the semester, but not after that

Exercises

- ▶ Exercises are offered every 2 weeks, **6 assignments** in total [Slides / Exercises](#)
- ▶ Exercises deepen the understanding with **pen & paper and coding tasks**
- ▶ Solving the exercises is not a prerequisite to participate in the exam
- ▶ Solving the exercise sheets yourself is the **best preparation for the exam**

Exercises

- ▶ **Sheets** are introduced in the tutorial and made available on the **course website**
- ▶ Each sheet runs for **two weeks**; the first one goes out **today**
- ▶ **Nothing has to be handed in.** We publish the solutions and discuss them in the tutorial
- ▶ You can work on the exercise sheets **in groups!**
- ▶ Solving the exercise sheets yourself is the **best preparation for the exam**

The First Five Weeks

Week	Date	Lecture	Tutorial
1	14.10.	Introduction	Sheet 1 out
2	21.10.	Computation Graphs	work on sheet 1
3	28.10.	Deep Neural Networks I	Solutions 1; sheet 2 out
4	04.11.	Deep Neural Networks II	work on sheet 2
5	11.11.	Regularization	Solutions 2; sheet 3 out

► ...and so on, for **6 sheets** over the semester

A Note on AI Assistants

- ▶ **Spending time** on the exercises is the **most effective** way to prepare for the exam
- ▶ Having an AI assistant solve a sheet for you will effectively **harm you**
- ▶ It will not help you in the exam: there you need to understand the topic yourself
- ▶ **Simple rule to success:** never show an assistant the exercise sheets!
- ▶ Asking an assistant to **explain a concept** is a different matter – but solve the sheets yourself, then compare with our solutions

Exercise Helpdesk

- ▶ We offer a **weekly helpdesk** where our TAs provide individual support
- ▶ **Just show up** (no registration)
- ▶ Ask **any question** about the exercises
- ▶ Bring your laptop and show us the problem
- ▶ Start working on your sheet **early**
- ▶ **When should it be?** Scan the code and tell us which slot suits you



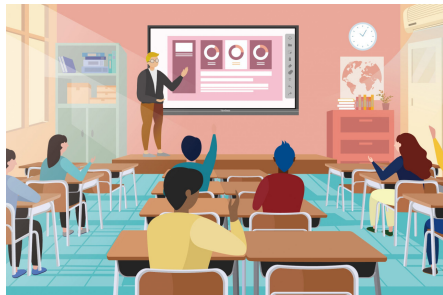
vote for a time

Forum

- ▶ We use the **Ilias forum** for questions about the lecture and the exercises
- ▶ **Ask** (many) questions! **Answer** questions if you think you know the answer!
- ▶ Questions asked in the forum help everyone – prefer it over private mail
- ▶ Or simply ask **in person** in the break after the lecture

A Typical Wednesday

1. **Lecture** (14:15–15:45)
 - ▶ New material, examples and discussion
2. **Break** (30 minutes, 15:45–16:15)
3. **Tutorial** (16:15–17:45), hands-on:
 - ▶ Live-coding sessions
 - ▶ Quizzes
 - ▶ Old exam exercises
 - ▶ Programming tips and tricks
 - ▶ A new sheet, or the solutions to the last one



Questions? I am there in the break after the lecture.

What will the exam look like?

- ▶ **Written** main and make-up exam
- ▶ You may **choose freely** (but no 3rd exam)
- ▶ Registration via **Alma**
- ▶ Only **pen and ruler** allowed (no notes)
- ▶ Duration: 90 minutes (can be solved in 60)
- ▶ 5 Tasks, 10 points each
- ▶ Tasks cover both **lectures** and **exercises**
- ▶ Mix of knowledge, calculation, multiple choice



Deep Learning

Exam, Winter 2020/2021

Tübingen, March 5, 2021

Student Number:	<input type="text"/>
Seat Number:	<input type="text"/>
First Name:	<input type="text"/>
Last Name:	<input type="text"/>
Date of Birth:	<input type="text"/>
Program of Study (e.g., "Informatik"):	<input type="text"/>
Master ☐ Bachelor ☐	

Task	1	2	3	4	5	Σ
Maximal Score	10	10	10	10	10	50
Attained Score						
Correction						

Remarks:

- Leave your bag and jacket at the front desk. Only take your **pen, ruler, student ID** with you.
- Notes, books, printouts, smartphones or calculators are not permitted.
- Have your student ID ready for inspection.
- Write your full name at the top of **every** page.
- Use the space provided for your answers. You can use the back side if you need more space.
- Additional paper can be obtained at the front desk.
- Use black or blue pen. Pencil or red pens are not allowed and will lead to a score of 0.
- A total score of **25** is sufficient for passing the exam. The duration of the exam is **90 minutes**.

Good Luck!

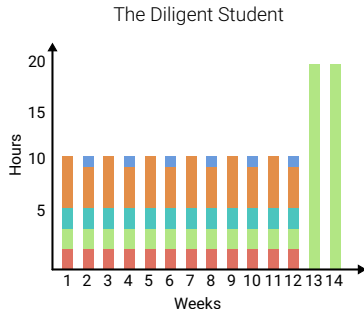
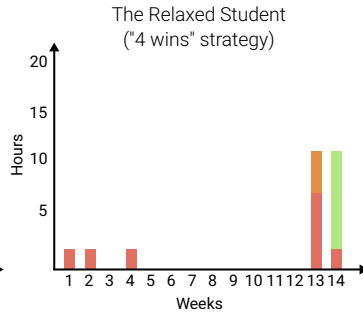
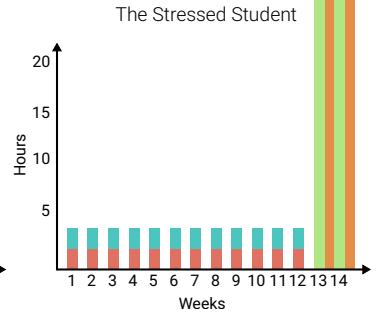
Exam Registration and Admission

- ▶ **Registration** for the exam is in **Alma**
- ▶ No Alma account by the deadline? Email your name, matriculation number and field of study to Patrizia Balloch (tml-sekretariat@inf.uni-tuebingen.de)
- ▶ The **exam date** is not yet known – it will be published on the course website
- ▶ Throughout the semester: **5 short quizzes on Ilias** about the lecture
- ▶ **To take the exam, you must participate in at least 3 of the 5 quizzes!**
- ▶ The quizzes are announced in the lecture and on the course website
- ▶ The quizzes need an **Ilias account**. No account? Send me an email

Work Ethics

This lecture has **6 ECTS**, corresponding to a total workload of **180 hours** (MHB)

Lecture Videos Self-Study Live Sessions Exercises Quizzes



Course Materials and Prerequisites

Where This Class Comes From

- ▶ The content of this class is based on the **Deep Learning** class by **Prof. Andreas Geiger** (Autonomous Vision Group, Tübingen)
<https://uni-tuebingen.de/de/175884>
- ▶ The lectures of that class are available as **videos on YouTube**
<https://www.youtube.com/playlist?list=PL05umP7R6ij3NTWIdtMbfvX7Z-4WEXRqD>
- ▶ They are a useful companion, but **we do not cover exactly the same material**



Andreas Geiger
University of Tübingen

Course Materials

Books:

- ▶ Goodfellow, Bengio, Courville: Deep Learning
<http://www.deeplearningbook.org>
- ▶ Bishop: Pattern Recognition and Machine Learning
<http://www.springer.com/gp/book/9780387310732>
- ▶ Zhang, Lipton, Li, Smola: Dive into Deep Learning
<http://d2l.ai>
- ▶ Deisenroth, Faisal, Ong: Mathematics for Machine Learning
<https://mml-book.github.io>
- ▶ Petersen, Pedersen: The Matrix Cookbook
http://cs.toronto.edu/~bonner/courses/2018s/csc338/matrix_cookbook.pdf

Course Materials

Courses:

- ▶ McAllester (TTI-C): Fundamentals of Deep Learning
<http://mcallester.github.io/ttic-31230/Fall12020/>
- ▶ Kolterl, Chen (CMU): Deep Learning Systems
<https://dlsyscourse.org/lectures/>
- ▶ Leal-Taixe, Niessner (TUM): Introduction to Deep Learning
<http://niessner.github.io/I2DL/>
- ▶ Grosse (UoT): Intro to Neural Networks and Machine Learning
http://www.cs.toronto.edu/~rgrosse/courses/csc321_2018/
- ▶ Li (Stanford): Convolutional Neural Networks for Visual Recognition
<http://cs231n.stanford.edu/>
- ▶ Abbeel, Chen, Ho, Srinivas (Berkeley): Deep Unsupervised Learning
<https://sites.google.com/view/berkeley-cs294-158-sp20/home>

Course Materials

Tutorials:

- ▶ The Python Tutorial

<https://docs.python.org/3/tutorial/>

- ▶ NumPy Quickstart

<https://numpy.org/devdocs/user/quickstart.html>

- ▶ PyTorch Tutorial

<https://pytorch.org/tutorials/>

Frameworks / IDEs:

- ▶ Visual Studio Code

<https://code.visualstudio.com/>

- ▶ Google Colab

<https://colab.research.google.com>

Prerequisites

Computer Science:

- ▶ Variables, functions, loops, classes, algorithms

Python:

- ▶ <https://docs.python.org/3/tutorial/>

Math:

- ▶ Linear algebra, probability and information theory. If unsure, have a look at:
Luxburg: [Mathematics for Machine Learning \(Lecture\)](#)
Deisenroth: [Mathematics for Machine Learning \(Book\)](#)
Geiger: [Mathematics for Deep Learning \(4h Micro Lecture\)](#)

Prerequisites

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN



$$\mathbf{A} = \begin{pmatrix} 3 & 1 \\ 2 & 1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 1 & 2 \\ 3 & 1 \end{pmatrix} \Rightarrow \mathbf{AB} = \begin{pmatrix} 6 & 7 \\ 5 & 5 \end{pmatrix}$$

$$\mathbf{Ax} = \mathbf{b}$$

$$\mathbf{A}^{-1}\mathbf{Ax} = \mathbf{A}^{-1}\mathbf{b}$$

$$\mathbf{I}_N \mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

$$\hat{\mathbf{w}}_{ML} = \underset{\mathbf{w}}{\operatorname{argmax}} \underbrace{\sum_{i=1}^N \log p_{\text{model}}(y_i | \mathbf{x}_i, \mathbf{w})}_{\text{Log-Likelihood}}$$

$$\sum_{i=1}^M \frac{\partial f}{\partial g_i} \frac{dg_i}{dx}$$

$$\mathbf{x}^T \mathbf{y} = (\mathbf{x}^T \mathbf{y})^T = \mathbf{y}^T \mathbf{x} = \mathbf{x}^T \mathbf{y}$$

Math for Deep Learning

Prof. Dr. Andreas Geiger

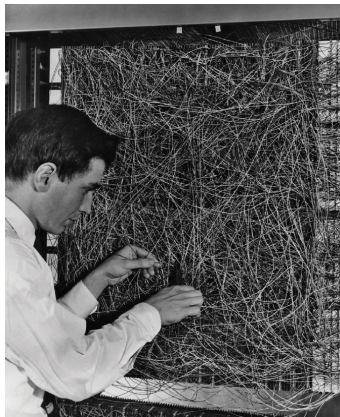


Stretching Break

5 Minutes

The History of Deep Learning

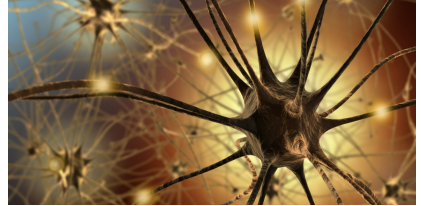
- ▶ 1943: McCulloch and Pitts propose the first mathematical model of a neuron
- ▶ 1958: Rosenblatt's Perceptron
- ▶ 1986: Backpropagation algorithm
- ▶ 1997: Long Short-Term Memory (LSTM)
- ▶ 1998: Convolutional Neural Networks (MNIST, LeNet)
- ▶ 2009-2012: Image Recognition Breakthroughs (CIFAR-10, ImageNet, AlexNet)
- ▶ 2012-now: Modern era of deep learning



[The Mark I Perceptron]

Inspiration from the Brain

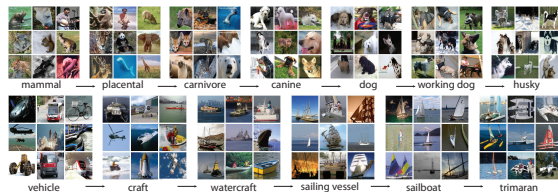
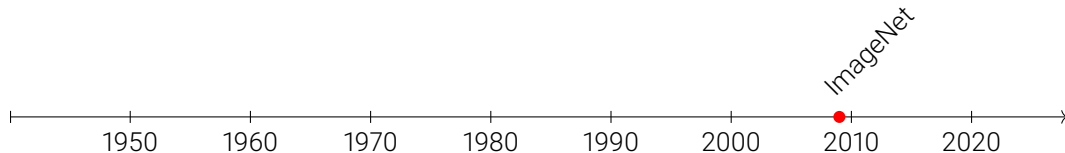
- ▶ Neural networks began as models of **neurons in the brain** (McCulloch and Pitts, Rosenblatt)
- ▶ Building models that are like the brain has **inspired many** (including Geoffrey Hinton)
- ▶ The degree to which modern neural networks are actually like human or animal brains is a **research question**, and there is **no consensus** on the matter
- ▶ What is definitely true: historically, deep learning has been **inspired** by the idea of building computers that work like the brain



2009: ImageNet

ImageNet, introduced in 2009, was the first large-scale image recognition dataset.

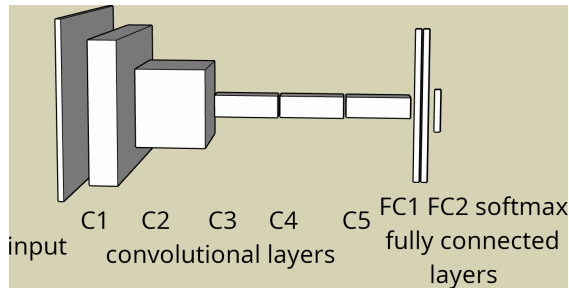
- Recognition benchmark (ILSVRC)
- 1 million annotated images
- 1000 categories



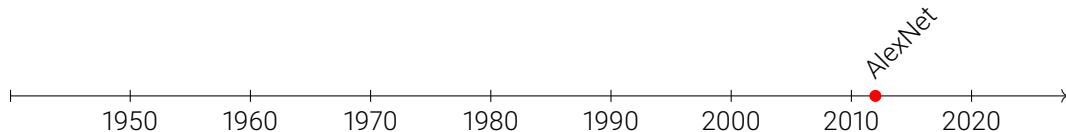
2012: AlexNet

AlexNet, published at NeurIPS in 2012, was a breakthrough in image recognition.

- ▶ The first neural network to win ILSVRC
- ▶ A **deep model** trained on **GPUs**
- ▶ Sparked the deep learning revolution



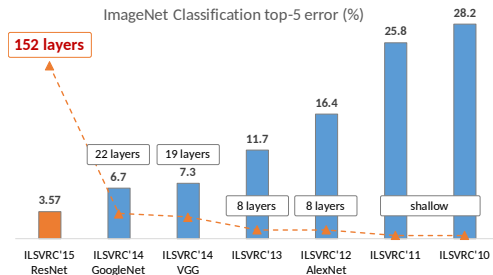
[Trebol6, Wikimedia Commons, CC BY-SA 4.0]



2012-2017: The ImageNet Era

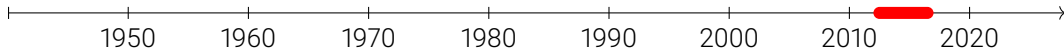
In five years, the top-5 error on ILSVRC fell from 25.8% to below 4%.

- Networks became much **deeper**
- AlexNet (8) → ResNet (152 layers)
- Surpassed human-level performance



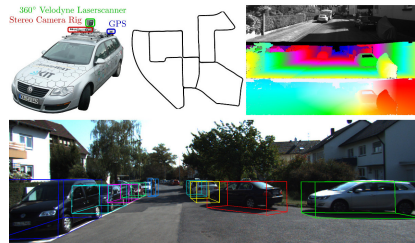
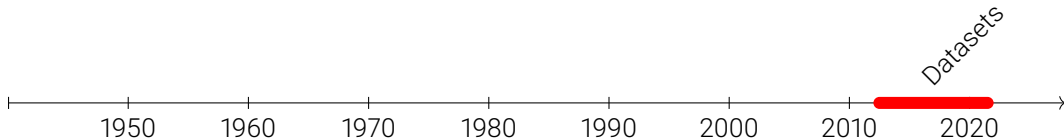
[He et al., CVPR 2016]

ImageNet Era



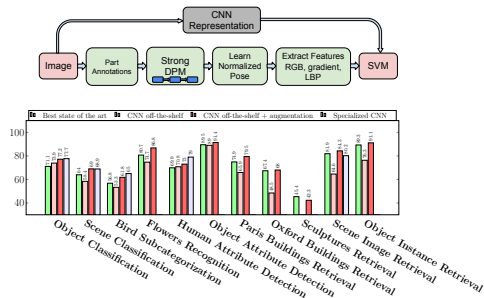
2012-now: The Golden Age of Datasets

- ▶ KITTI, Cityscapes: Self-driving
- ▶ PASCAL, MS COCO: Recognition
- ▶ ShapeNet, ScanNet: 3D DL
- ▶ GLUE: Language understanding
- ▶ Visual Genome: Vision/Language
- ▶ VisualQA: Question Answering

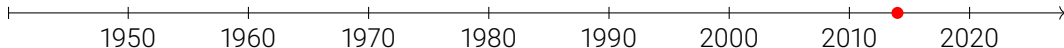


2014: Generalization

- Empirical demonstration that deep representations generalize well despite large number of parameters
- Pre-train CNN on large amounts of data on a generic task (e.g., ImageNet)
- Fine-tune only last layers on few data of a new task

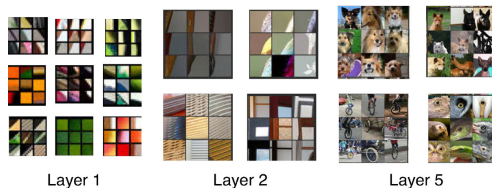
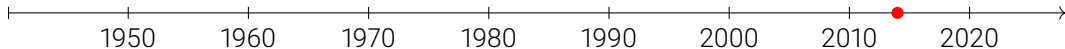


Generalization



2014: Visualization

- ▶ Goal: provide insights into what the network (black box) has learned
- ▶ Visualized image regions that most strongly activate neurons at different layers
- ▶ Higher levels capture more abstract semantic information

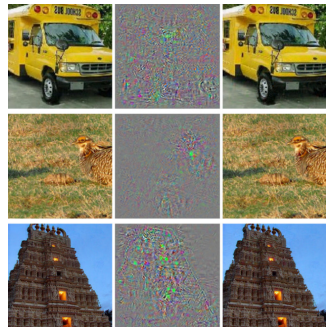
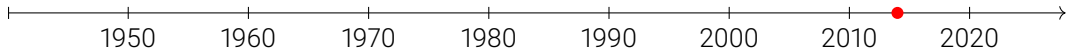


2014: Adversarial Examples

- ▶ Accurate image classifiers can be fooled by imperceptible changes
- ▶ Adversarial example:

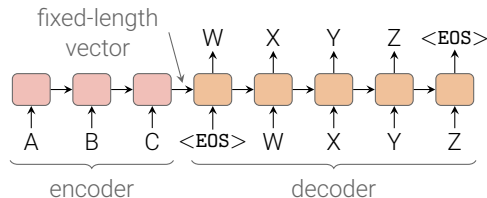
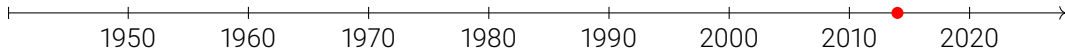
$$x + \underset{\Delta x}{\operatorname{argmin}} \{ \|\Delta x\|_2 : f(x + \Delta x) \neq f(x) \}$$

- ▶ All images classified as “ostrich”



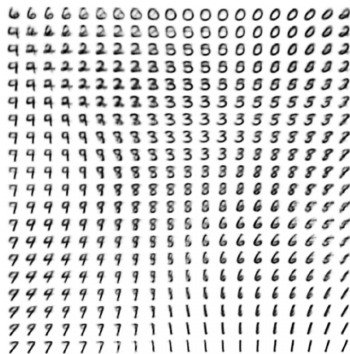
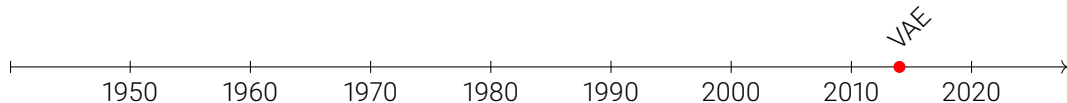
2014: Sequence to Sequence Learning

- ▶ An encoder RNN maps the input sequence to a fixed-length vector
- ▶ A decoder RNN generates the output sequence from that vector
- ▶ First end-to-end neural machine translation



2013-2014: Variational Autoencoders

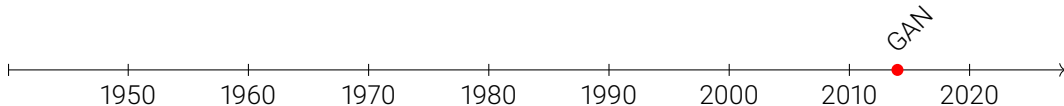
- ▶ Learn a **latent representation** together with a decoder that turns it back into data
- ▶ Trained by maximising a lower bound on the likelihood
- ▶ Sampling the latent space produces new data



2014: Generative Adversarial Networks

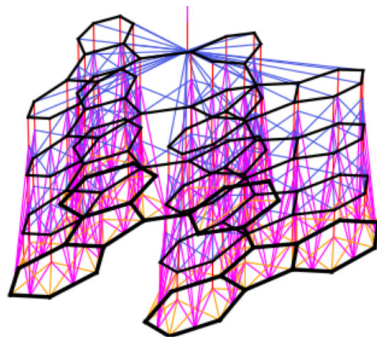
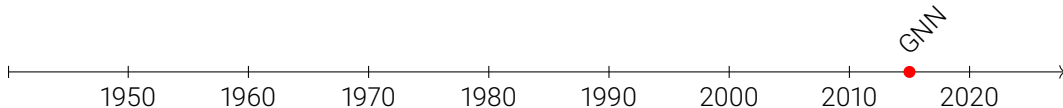
- ▶ A **generator** and a **discriminator** are trained against each other
- ▶ No likelihood is needed, only a discriminator that cannot tell real from fake
- ▶ Produced the first photorealistic synthetic faces

Moore's Law of AI
4.5 years of progress on faces



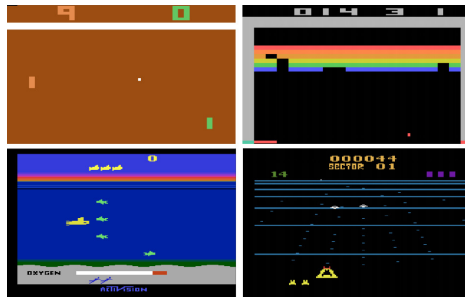
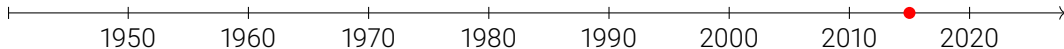
2015: Graph Neural Networks

- ▶ Many kinds of data are **graphs**: molecules, meshes, social and traffic networks
- ▶ Convolution is generalised from grids to arbitrary neighbourhoods
- ▶ Strong results in predicting molecular properties



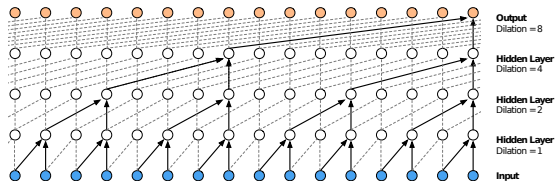
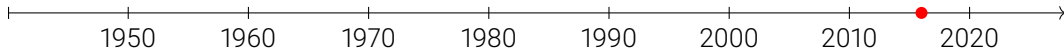
2015: Deep Reinforcement Learning

- ▶ Deep Q-Networks learn to play Atari games from raw pixels
- ▶ Reached human-level performance on a range of games
- ▶ Reward signal only, no supervision



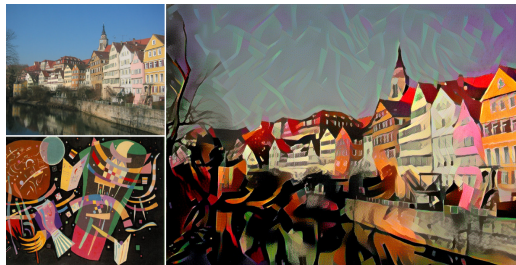
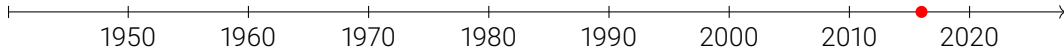
2016: WaveNet

- ▶ Autoregressive model of raw audio waveforms
- ▶ Dilated convolutions capture long-range structure
- ▶ Clear improvement in text-to-speech quality



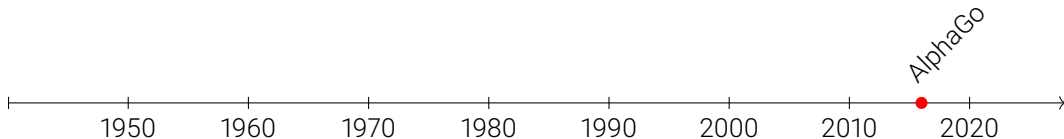
2016: Style Transfer

- ▶ Separates the content of one image from the style of another
- ▶ Uses the feature statistics of a pre-trained CNN
- ▶ An early sign that learned features are broadly reusable



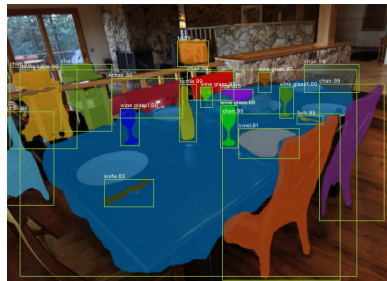
2016: AlphaGo defeats Lee Sedol

- ▶ Go was long considered out of reach for computers
- ▶ Deep networks combined with Monte Carlo tree search
- ▶ AlphaGo Zero later learned entirely from self-play

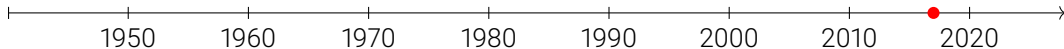


2017: Mask R-CNN

- ▶ Detects objects and segments them at the same time
- ▶ One model for detection, segmentation and keypoints
- ▶ Became the standard baseline for instance segmentation

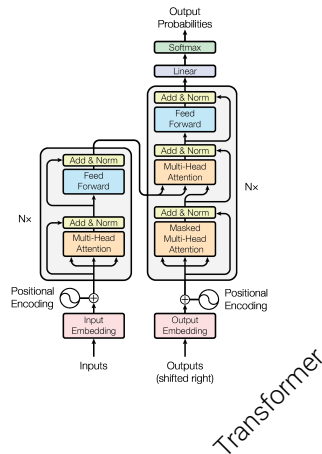
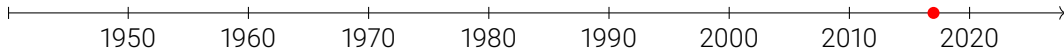


Mask R-CNN



2017: The Transformer Architecture

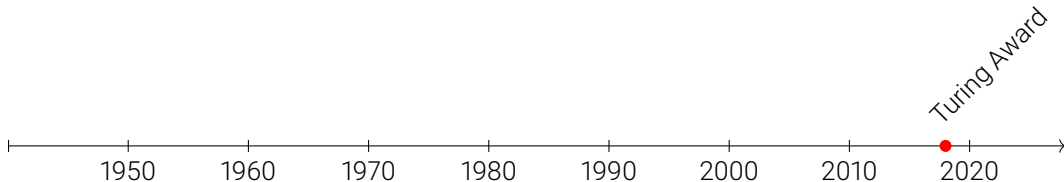
- Attention replaces recurrence and convolution
- Every position attends to every other
- Parallelises far better than an RNN
- Backbone of all modern large models



2018: Turing Award

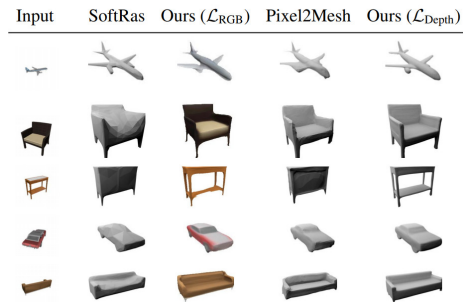
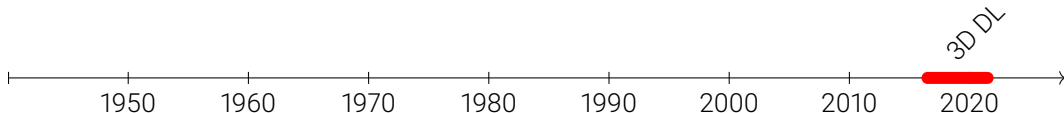
In 2018, the “Nobel Prize of computing” was awarded to:

- ▶ Yoshua Bengio
- ▶ Geoffrey Hinton
- ▶ Yann LeCun

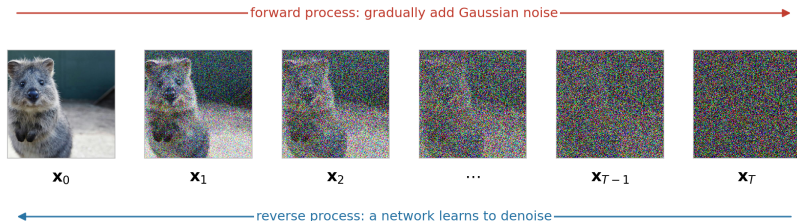


2016-2020: 3D Deep Learning

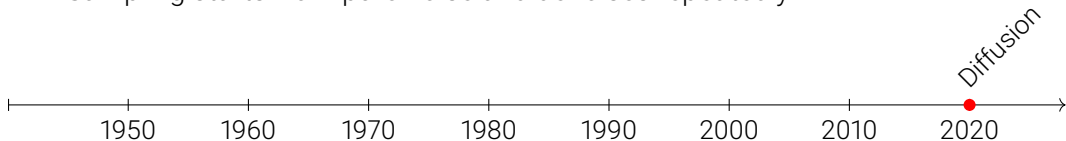
- ▶ First models to successfully output 3D representations
- ▶ Voxels, point clouds, meshes, implicit representations
- ▶ Prediction of 3D models even from a single image
- ▶ Geometry, materials, light, motion



2020: Diffusion Models

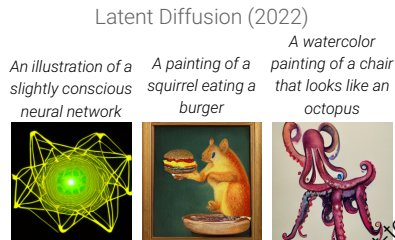


- ▶ The neural network is trained to predict the denoised version of a noisy input
- ▶ Sampling starts from pure noise and denoises repeatedly

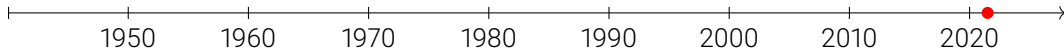


2021-2022: Text-to-Image Generation

- Diffusion models conditioned on text
- DALL·E, Imagen, Stable Diffusion
- Latent diffusion made it cheap enough to run on a single GPU
- Reached a very wide audience outside research

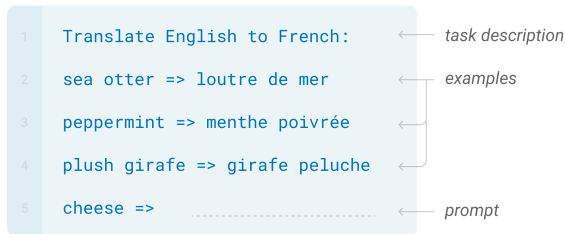
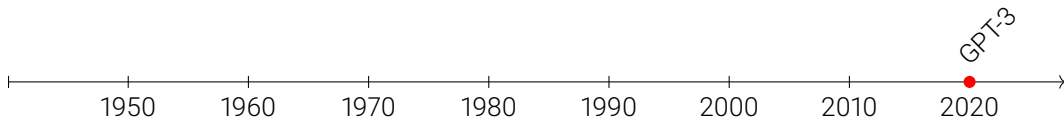


Text-to-Image



2020: GPT-3

- ▶ Language model by OpenAI, 175 billion parameters
- ▶ Text-in / text-out interface
- ▶ Many use cases: coding, poetry, news articles, chatbots
- ▶ Solves new tasks from a few examples in the prompt



2022-2023: ChatGPT / GPT-4

- ▶ **ChatGPT** (2022): human feedback turned a language model into something people could talk to
- ▶ Incredibly popular outside the machine learning community
- ▶ **GPT-4** (2023): "Sparks of AGI", performs on par with the average student on academic exams
- ▶ Architecture and training data undisclosed

training with human feedback (RLHF)

1. Supervised fine-tuning

on answers written by humans



2. Reward model

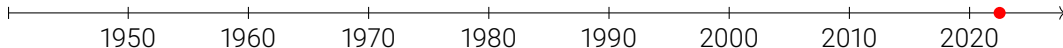
trained on human rankings



3. Reinforcement learning

maximise the learned reward

ChatGPT / GPT-4



2024: Two Nobel Prizes

- ▶ **Physics:** John Hopfield and Geoffrey Hinton, “for foundational discoveries and inventions that enable machine learning with artificial neural networks”
- ▶ **Chemistry:** David Baker, Demis Hassabis and John Jumper, for computational protein design and protein structure prediction (AlphaFold)
- ▶ Deep learning is recognised as a tool of the natural sciences, not only of computer science



Hopfield



Hinton



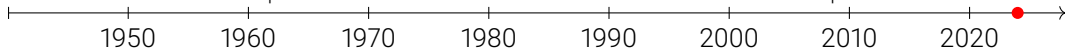
Baker



Hassabis



Jumper



1.3

Machine Learning Basics

Goodfellow et al.: Deep Learning, Chapter 5

<http://www.deeplearningbook.org/contents/ml.html>

Learning Problems

▶ **Supervised learning**

- ▶ Learn model parameters using dataset of data-label pairs $\{(x_i, y_i)\}_{i=1}^N$
- ▶ Examples: Classification, regression, structured prediction

▶ **Unsupervised learning**

- ▶ Learn model parameters using dataset without labels $\{x_i\}_{i=1}^N$
- ▶ Examples: Clustering, dimensionality reduction, generative models

▶ **Self-supervised learning**

- ▶ Learn model parameters using dataset of data-data pairs $\{(x_i, x'_i)\}_{i=1}^N$
- ▶ Examples: Self-supervised stereo/flow, contrastive learning

▶ **Reinforcement learning**

- ▶ Learn model parameters using active exploration from sparse rewards
- ▶ Examples: deep q learning, gradient policy, actor critique

Supervised Learning

Classification, Regression, Structured Prediction

Classification / Regression:

$$f : \mathcal{X} \rightarrow \mathbb{N} \quad \text{or} \quad f : \mathcal{X} \rightarrow \mathbb{R}$$

- ▶ Inputs $x \in \mathcal{X}$ can be any kind of objects
 - ▶ images, text, audio, sequence of amino acids, ...
- ▶ Output $y \in \mathbb{N}/y \in \mathbb{R}$ is a discrete or real number
 - ▶ classification, regression, density estimation, ...

Structured Output Learning:

$$f : \mathcal{X} \rightarrow \mathcal{Y}$$

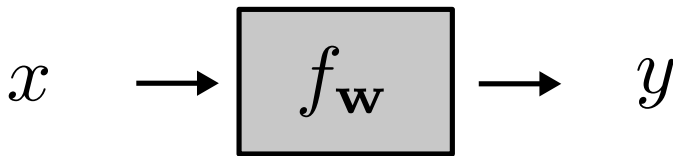
- ▶ Inputs $x \in \mathcal{X}$ can be any kind of objects
- ▶ Outputs $y \in \mathcal{Y}$ are complex (structured) objects
 - ▶ images, text, parse trees, folds of a protein, computer programs, ...

Supervised Learning

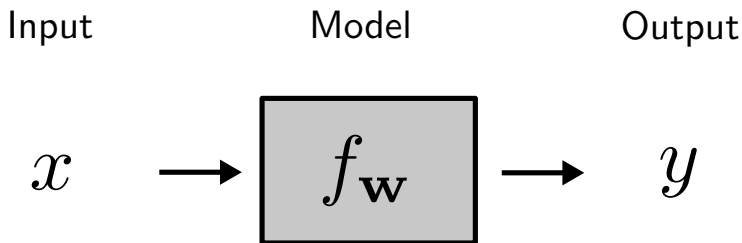
Input

Model

Output

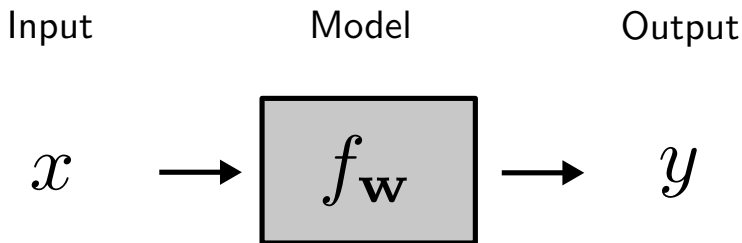


Supervised Learning



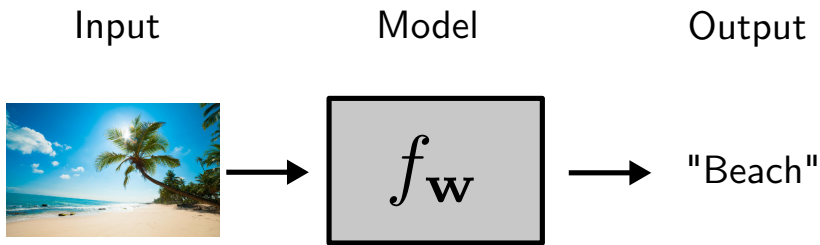
- **Learning:** Estimate parameters $\mathbf{w}$ from training data $\{(x_i, y_i)\}_{i=1}^N$

Supervised Learning



- **Learning:** Estimate parameters $\mathbf{w}$ from training data $\{(x_i, y_i)\}_{i=1}^N$
- **Inference:** Make novel predictions: $y = f_{\mathbf{w}}(x)$

Classification



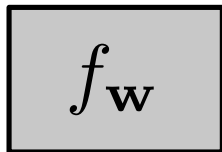
► **Mapping:** $f_{\mathbf{w}} : \mathbb{R}^{W \times H} \rightarrow \{\text{"Beach"}, \text{"No Beach"}\}$

Regression

Input



Model

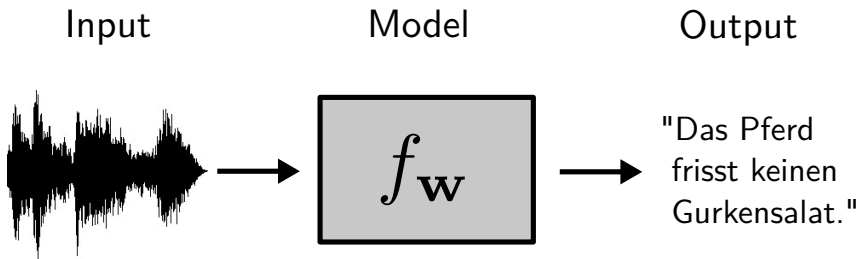


Output

143,52 €

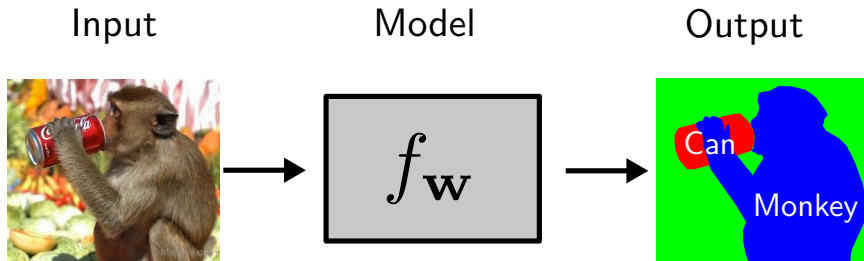
► Mapping: $f_{\mathbf{w}} : \mathbb{R}^N \rightarrow \mathbb{R}$

Structured Prediction



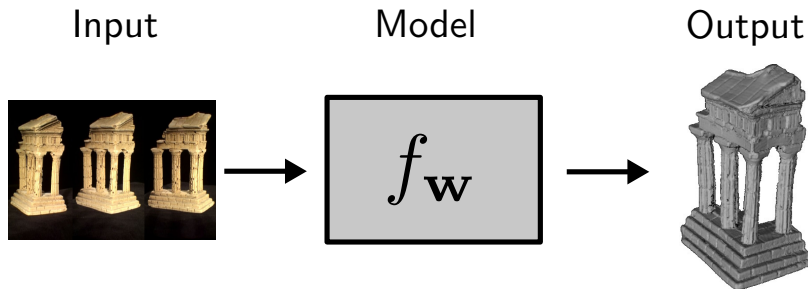
► **Mapping:** $f_{\mathbf{w}} : \mathbb{R}^N \rightarrow \{1, \dots, C\}^M$

Structured Prediction



► **Mapping:** $f_{\mathbf{w}} : \mathbb{R}^{W \times H} \rightarrow \{1, \dots, C\}^{W \times H}$

Structured Prediction



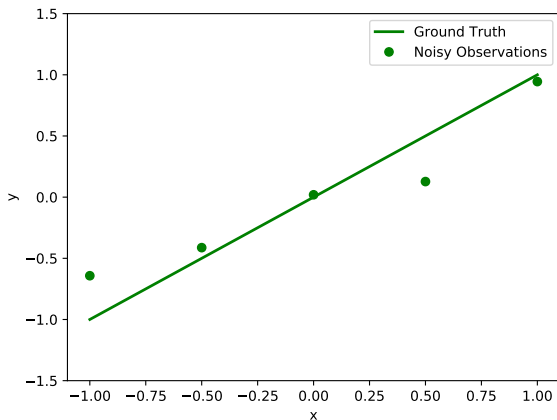
- **Mapping:** $f_{\mathbf{w}} : \mathbb{R}^{W \times H \times N} \rightarrow \{0, 1\}^{M^3}$
- **Suppose:** 32^3 voxels, binary variable per voxel (occupied/free)
- **Question:** How many different reconstructions? $2^{32^3} = 2^{32768}$
- **Comparison:** Number of atoms in the universe? $\sim 2^{273}$

Linear Regression

Linear Regression

Let $\mathcal{X}$ denote a dataset of size N and let $(\mathbf{x}_i, y_i) \in \mathcal{X}$ denote its elements ($y_i \in \mathbb{R}$).

Goal: Predict y for a previously unseen input $\mathbf{x}$. The input $\mathbf{x}$ may be multidimensional.

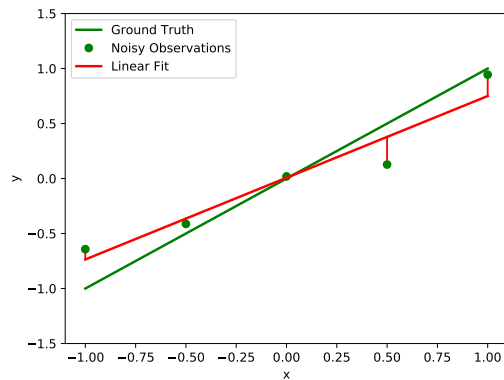


Linear Regression

The **loss function** $E(\mathbf{w})$ measures the displacement along the y dimension between the data points (green) and the model $f(\mathbf{x}, \mathbf{w})$ (red) specified by the parameters $\mathbf{w}$.

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^\top \mathbf{x}$$

$$\begin{aligned} E(\mathbf{w}) &= \sum_{i=1}^N (f(\mathbf{x}_i, \mathbf{w}) - y_i)^2 \\ &= \sum_{i=1}^N (\mathbf{x}_i^\top \mathbf{w} - y_i)^2 \\ &= \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \end{aligned}$$



Here: $\mathbf{x} = [1, x]^\top \Rightarrow f(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x$

Linear Regression

The **gradient of the loss function** with respect to the parameters $\mathbf{w}$ is given by:

$$\begin{aligned}\nabla_{\mathbf{w}} E(\mathbf{w}) &= \nabla_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \\ &= \nabla_{\mathbf{w}} (\mathbf{X}\mathbf{w} - \mathbf{y})^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) \\ &= \nabla_{\mathbf{w}} \left(\mathbf{w}^\top \mathbf{X}^\top \mathbf{X} \mathbf{w} - 2\mathbf{w}^\top \mathbf{X}^\top \mathbf{y} + \mathbf{y}^\top \mathbf{y} \right) \\ &= 2\mathbf{X}^\top \mathbf{X} \mathbf{w} - 2\mathbf{X}^\top \mathbf{y}\end{aligned}$$

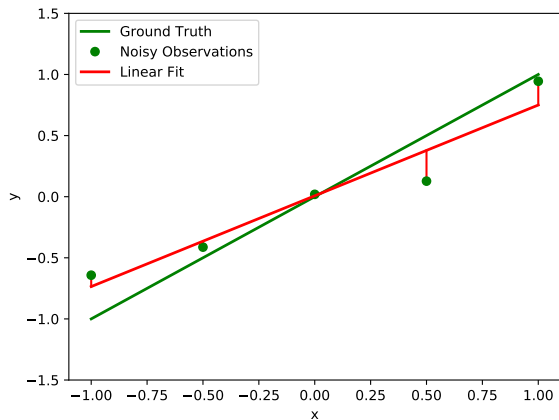
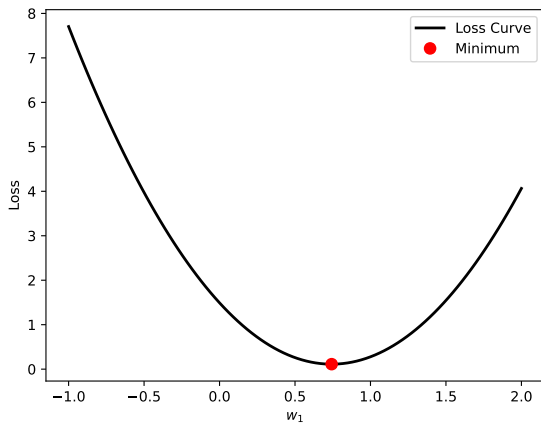
As $E(\mathbf{w})$ is quadratic and convex in $\mathbf{w}$, its minimizer (wrt. $\mathbf{w}$) is given in closed form:

$$\begin{aligned}\nabla_{\mathbf{w}} E(\mathbf{w}) &= 0 \\ \Rightarrow \mathbf{w} &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}\end{aligned}$$

The matrix $(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$ is also called **Moore-Penrose inverse** or pseudoinverse.

Example: Line Fitting

Line Fitting



Linear least squares fit of model $f(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x$ (red) to data points (green). Errors are also shown in red. Right: Loss function $E(\mathbf{w})$ wrt. parameter w_1 .

Break

- ▶ **Now: 30 minute break**
- ▶ **After that: Tutorial** about derivatives
 - ▶ Jacobian matrix, gradient
 - ▶ How to compute derivatives!



Please complete this short quiz!

forms.gle/WM7gaKEVW4H6FanJ7