Introductie AI

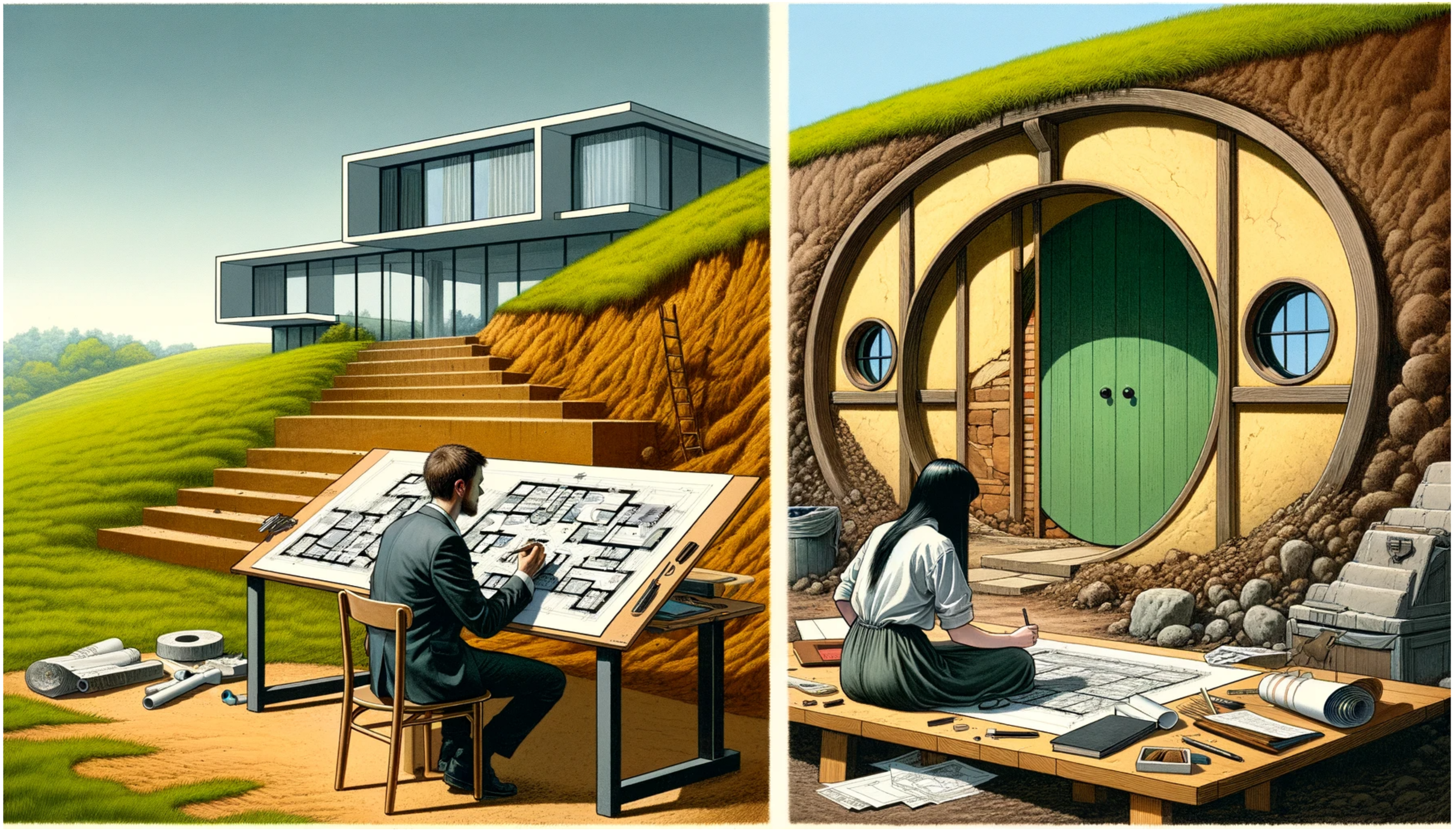
Raoul Grouls, 19-02-2024

Hello world

Raoul Grouls

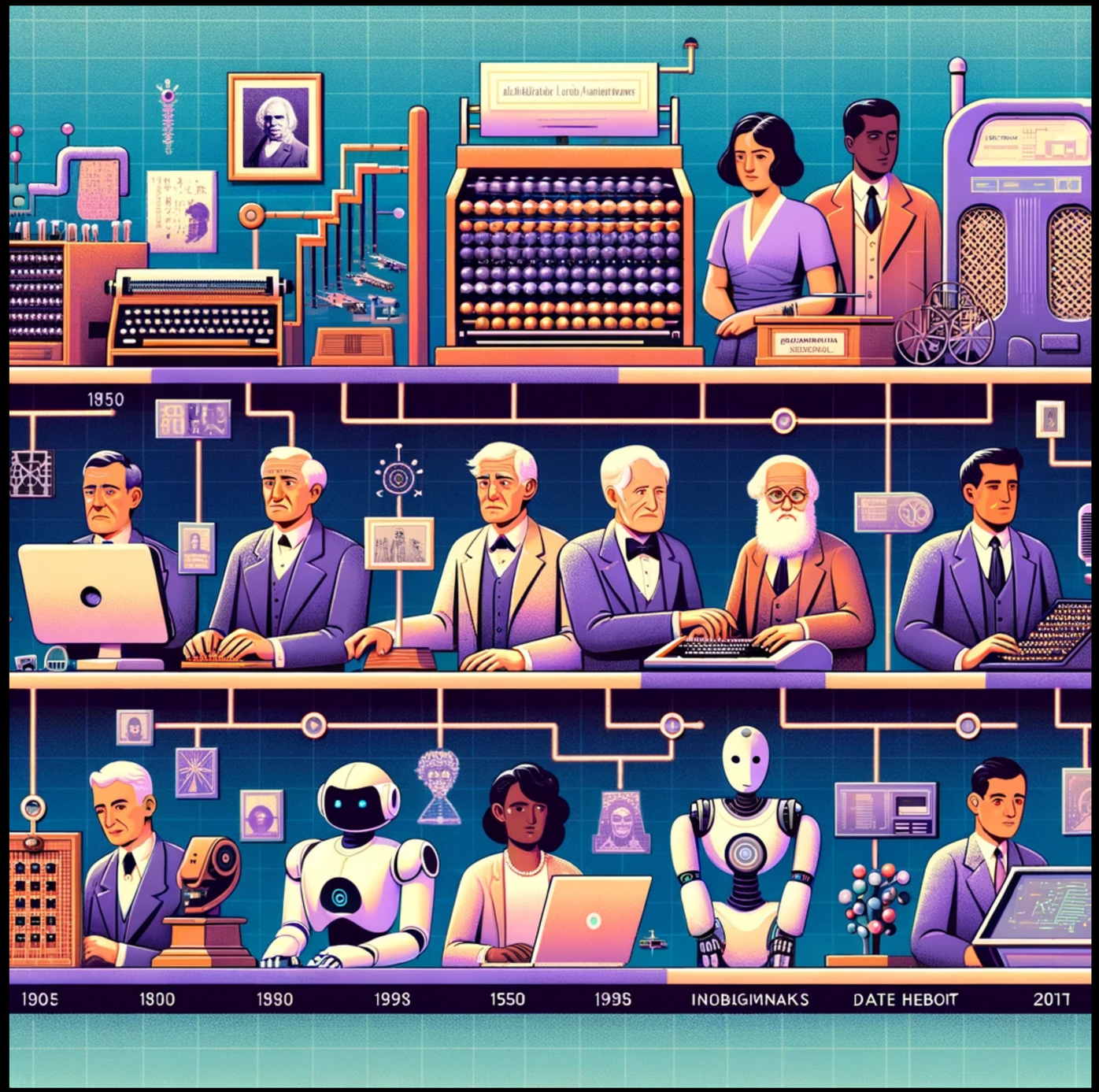
- AI onderzoeker & docent @ HAN
- AI docent @ HU
- data alchemist @ zzp the-pttrn.nl/
- raoul.grouls@han.nl

What is intelligence?



A very short history of AI

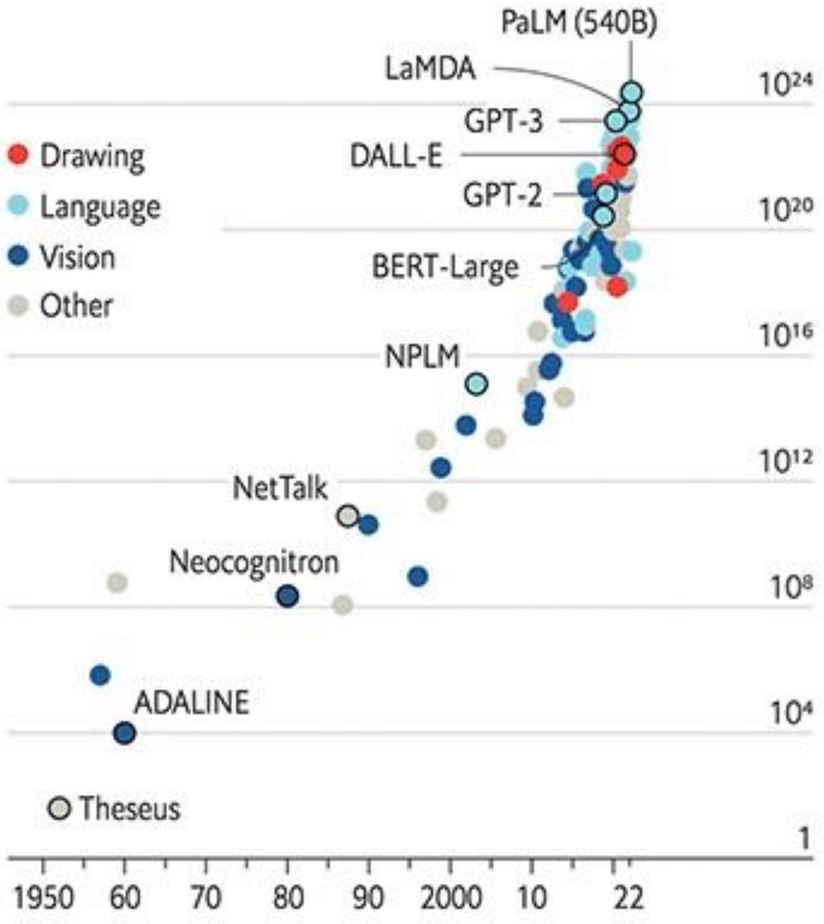
- 1842 Ada Lovelace & Charles Babbage work on the “analytical engine”
- 1950 Mathematics of Neural Networks
- 1970 Symbolic AI (expert systems)
- 1990-2010 statistical foundations
  - Swarm Intelligence
  - Support Vector Machines
  - Random Forest
- 2012 Deep learning
- 2017 Transformers
- 2023 chatGPT4



The blessings of scale

AI training runs, estimated computing resources used

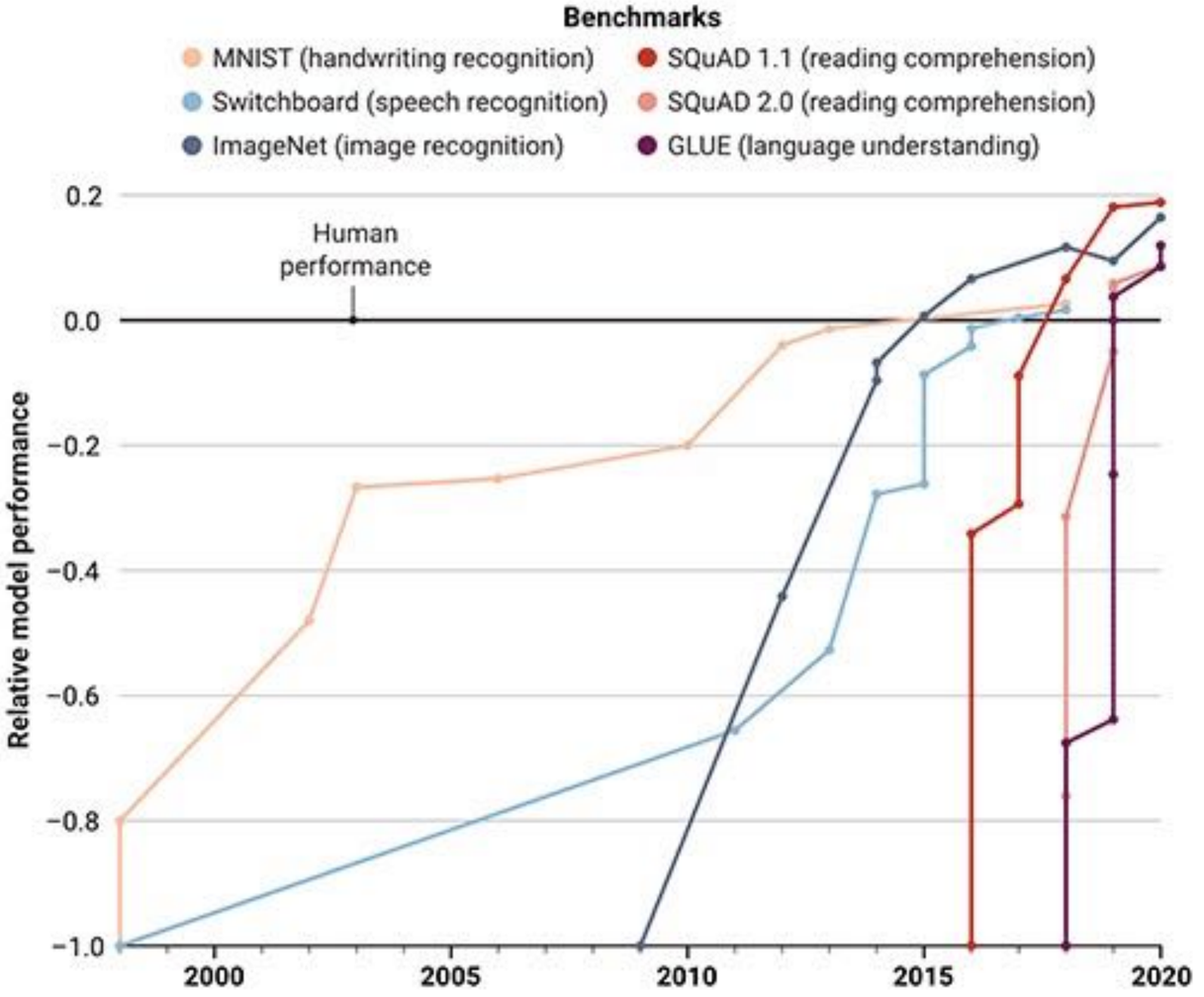
Floating-point operations, selected systems, by type, log scale



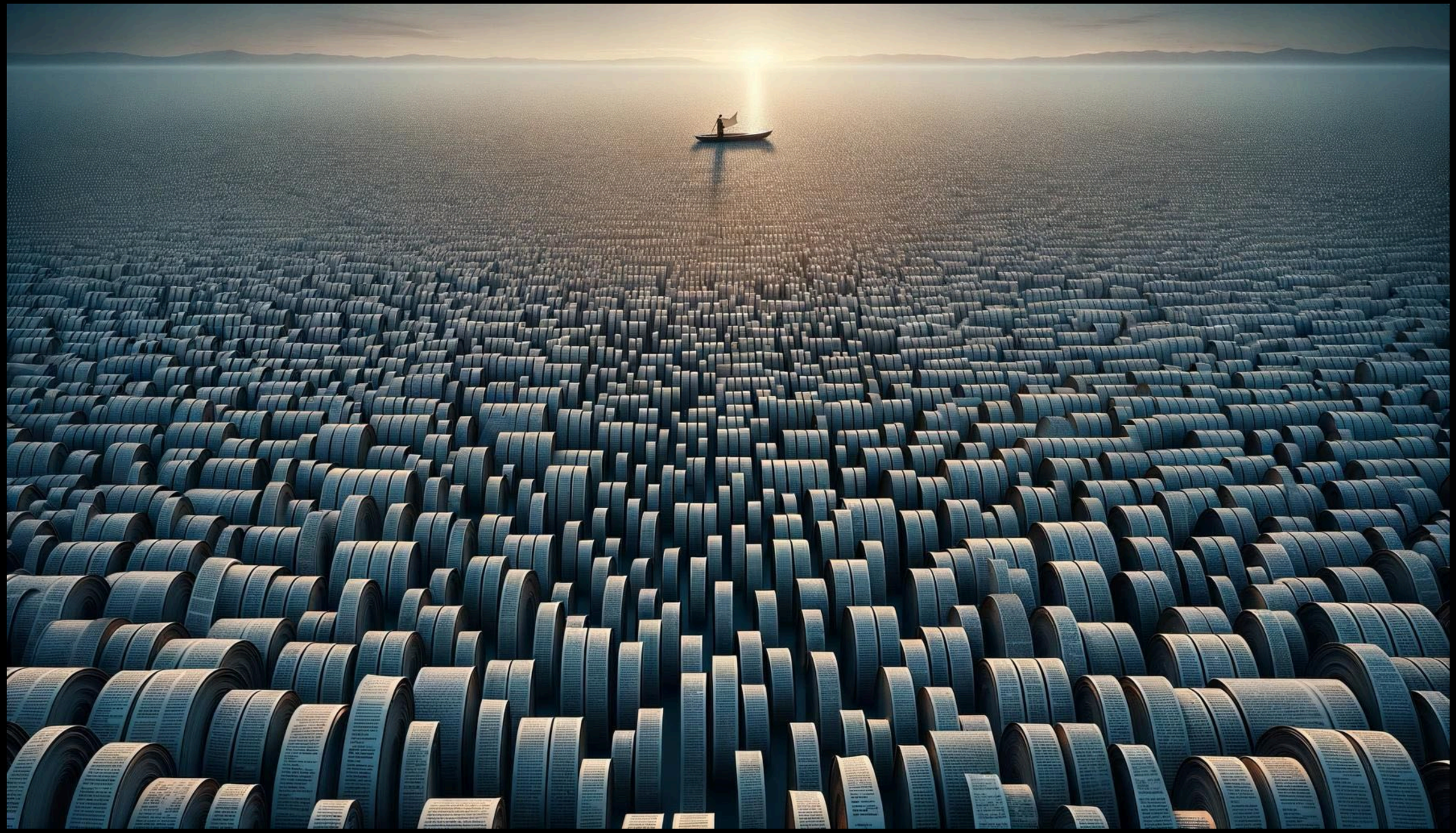
Sources: "Compute trends across three eras of machine learning", by J. Sevilla et al., arXiv, 2022; Our World in Data

Quick learners

The speed at which artificial intelligence models master benchmarks and surpass human baselines is accelerating. But they often fall short in the real world.



(GRAPHIC) K. FRANKLIN/*SCIENCE*; (DATA) D. KIELA *ET AL.*, DYNABENCH: RETHINKING BENCHMARKING IN NLP, DOI:10.48550/ARXIV.2104.14337

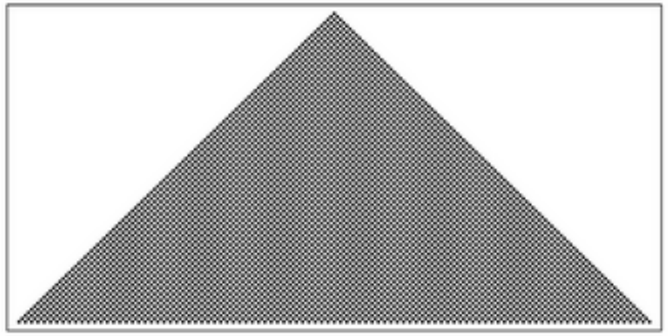


Different types of AI

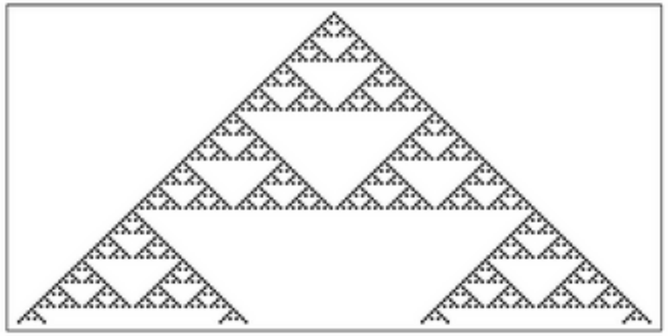
- Cellular Automata
- Genetic Algorithms
- Swarm Intelligence
- Machine Learning
- Deep Learning

simpele doelen, complex gedrag

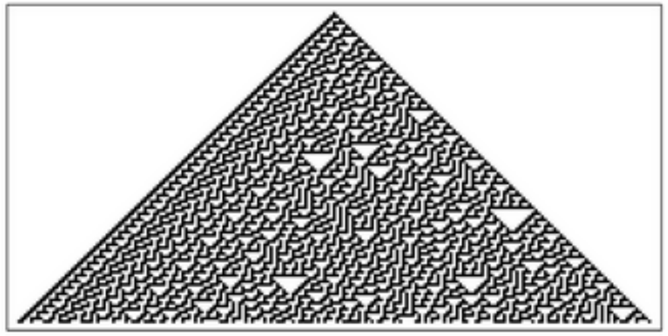
Cellular Automata



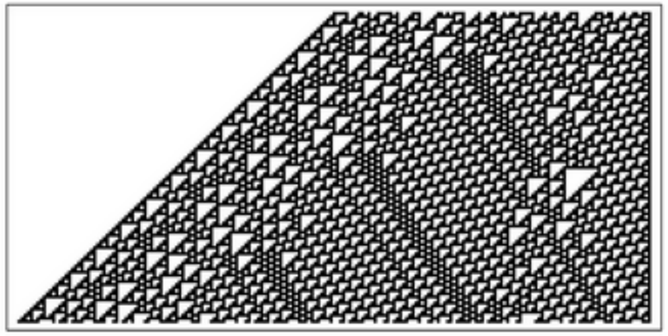
repetition (rule 250)



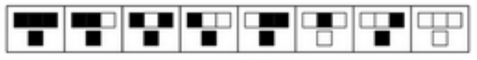
nesting (rule 90)



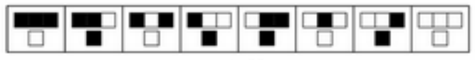
randomness (rule 30)



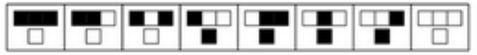
localized structures (rule 110)



rule 250



rule 90



rule 30

rule 110

Cellular Automata

- Recursive computational functions
- A grid world, where each “cell” takes some of the environment as input and gives a new output, based on simple rules
- Simple rules, complex behaviors
- You can’t always “jump” to the end of an evolution: you will need to simulate the full program to see the end.

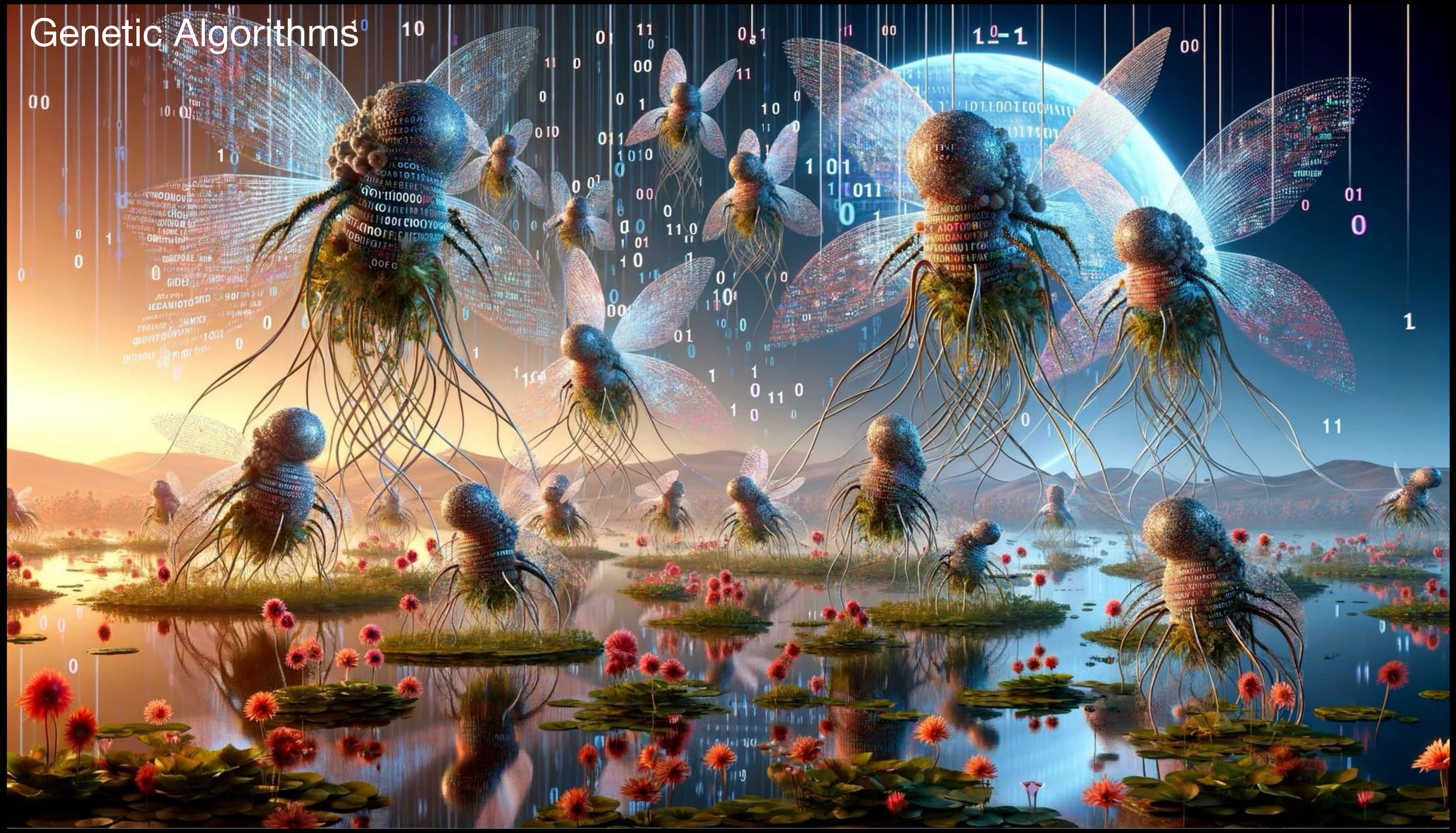
Cellular Automata applications

- Generative art
- Modelling growth processes
- Traffic flow simulations
- Disease spread simulations

Genetic Algorithms

1. Population, where each individual represents a solution
2. Fitness function: a measure of how good a solution performs
3. Selection: individuals with a high fitness have a higher chance of being selected for “breeding”
4. Recombination: pairs of succesful individuals are crossed over at random points to produce “offspring”
5. Mutation: with some probability, the solution of the offspring can have random mutations.
6. Replacement: some of the succesful individuals stay alive, some are replaced by new offspring.

Genetic Algorithms



Genetic Algorithm Applications

- The randomness of the process helps to explore the solutionspace
- You can even breed model “architectures”
- This can be great if you have no idea of how the structure of the solution looks
- GA can be adaptive to changing situations
- Often, specialised solutions perform better, but if you have no idea of what the “best” could be, or if this could change over time, GA are often a good second best.

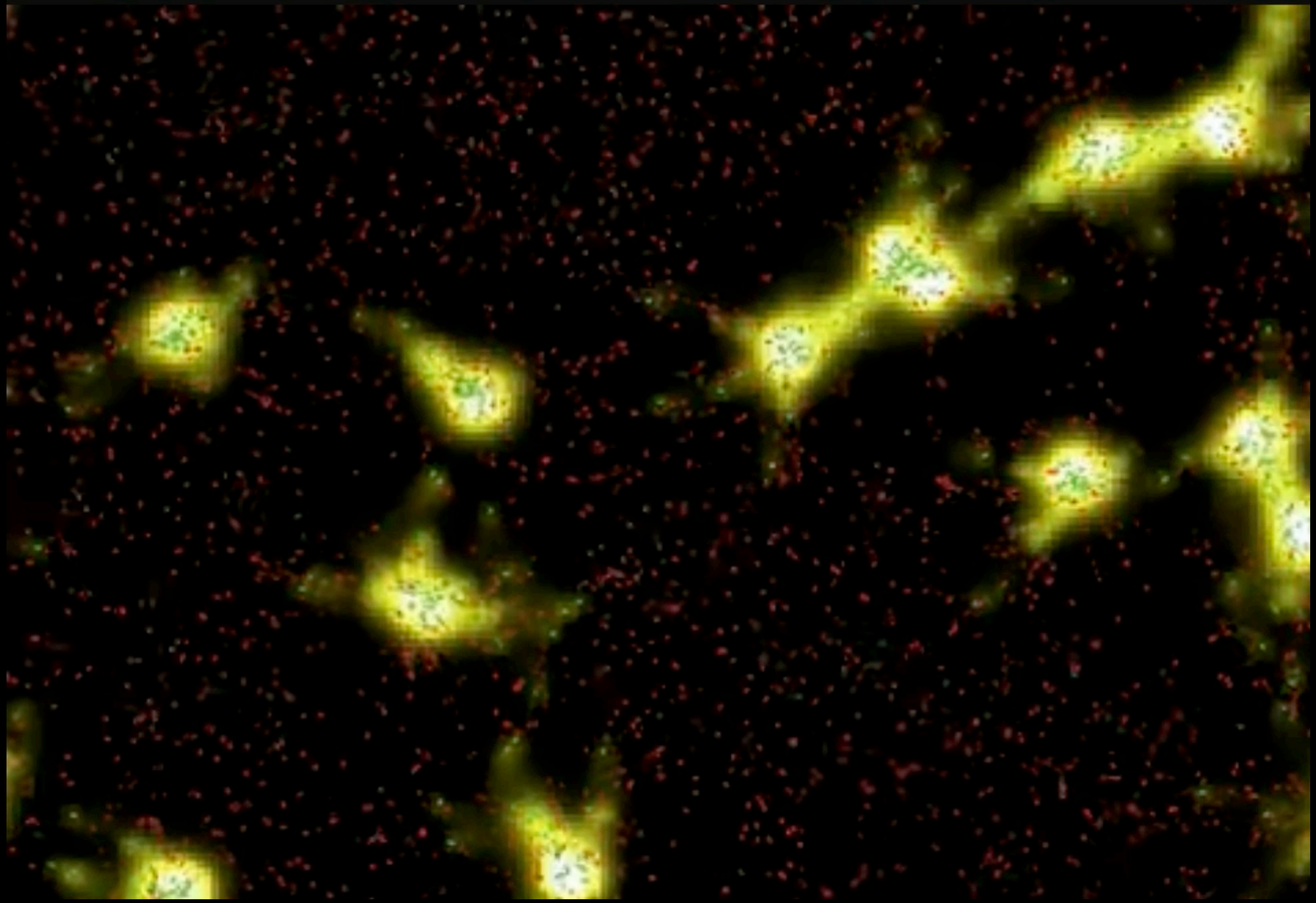
Swarm intelligence

Swarm Intelligence

- Swarm intelligence algorithms are a class of computational methods inspired by the collective behavior of social organisms, particularly insects, birds, and fish.
- These algorithms involve simulating the behavior of decentralized, self-organized systems to solve complex optimization and decision-making problems

Ant Colony Optimization

- Ant agents: can move to a problem space (typically a graph or grid)
- Pheromone trails: each edge or cell has a pheromone strenght, initialised at 0
- Ants deposit Pheromone after they are succesful (eg find “food”)
- Over time, Pheromone evaporates
- Pheromones indicate attractiveness of a path
- Decision policy: based on a combination of pheromone strength, some heuristics and randomness ants choose a direction



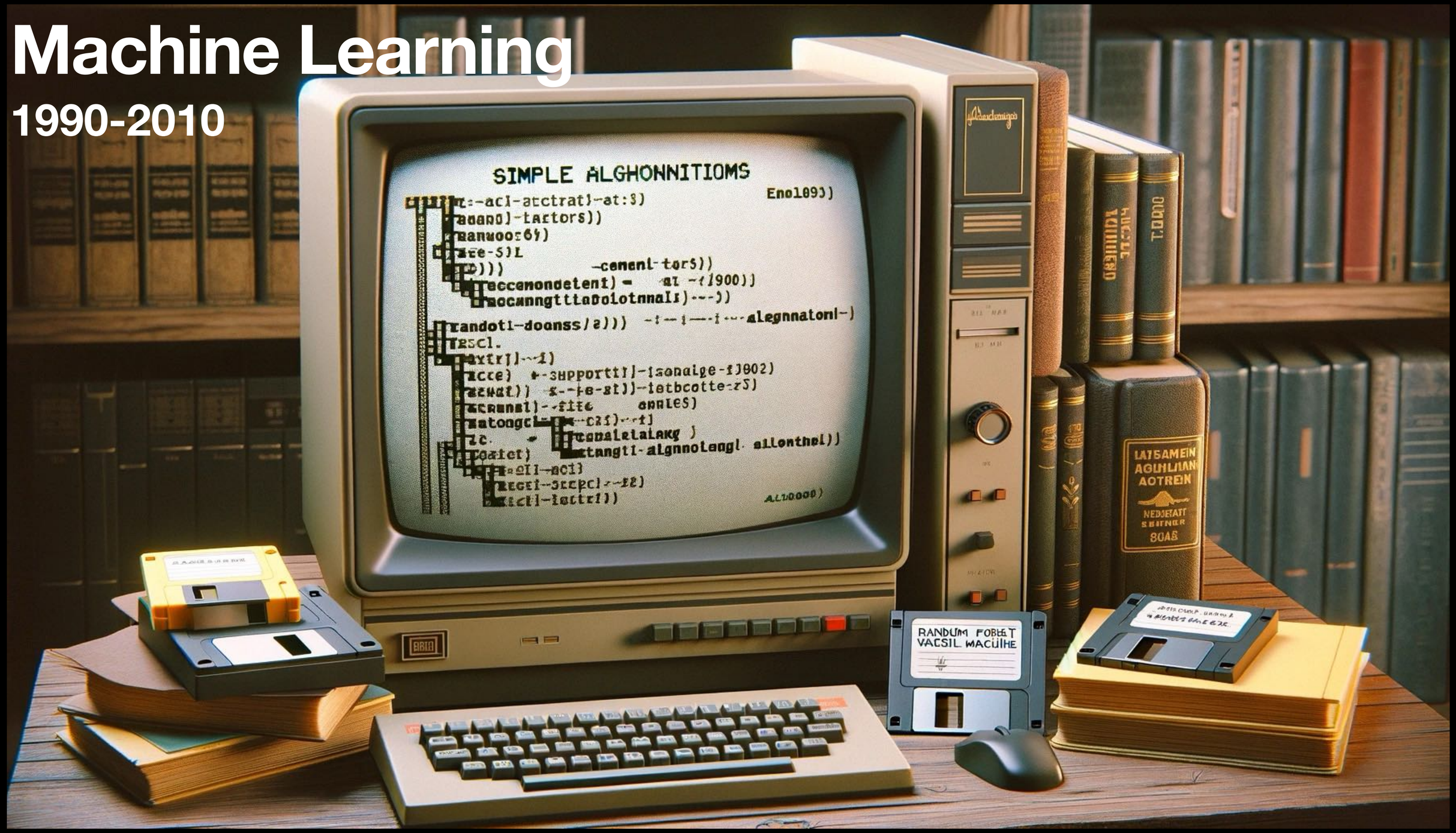
Swarm Intelligence Applications

Great for NP-hard optimisation problems

- Traveling salesman problem: single vehicle, minimize total distance
- Vehicle Routing problem: multiple vehicles, constraints like capacity and time windows
- Scheduling problems: minimize total completion time with constraints like job precedence and machine availability
- Knapsack Problem: selecting subsets that cover the full set as efficient as possible. Eg the placing of robot-chargers in a water distribution network.

Machine Learning

1990-2010



Machine Learning
1990-2010

- Algorithms that fit on a floppy disk an run on a commodore 64
- Support Vector Machines
- Random Forest

Complexity 1990 - SVM

$$max_a \sum_{i=1}^{m} a_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} a_i a_j y_i y_j K(x_i, x_j)$$

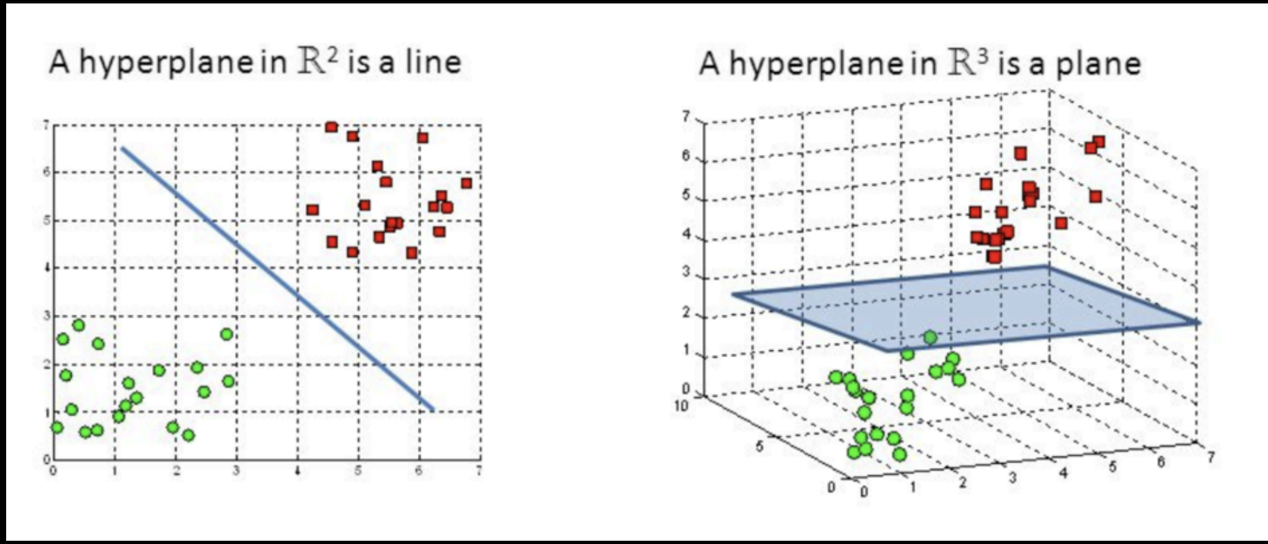
$$K(x_i, x_j) = exp(-\gamma ||x_i - x_j||^2)$$

$$J(w, b) = \frac{1}{2} w^T w + C \sum_{i=1}^{m} max(0, 1 - t^i(w^T x^i + b))$$

m	number of observations
a	counter for errors
x	observations
y	labels
γ, C	hyperparameters
w, b	weights we need to learn

Hyperplanes

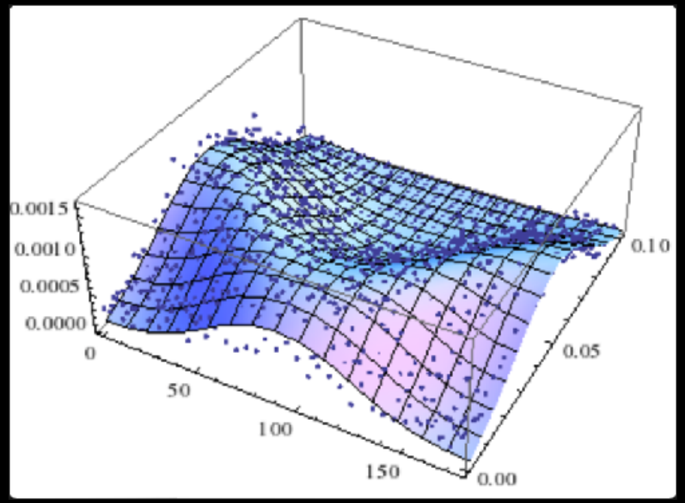
It’s a bird… It’s a plane…



- For classification, our outcome are discrete classes (e.g. “yes” or “no”)
- For regression, our outcome is a real number (e.g. 1.4 or 26.834)

We can use a linear model for both cases. We call these models “hyperplanes”: they have one dimension less than the ambient space.

With classification, we want the data to be separated by the hyperplane. With regression, we want to points to get as close as possible to the hyperplane



Linear models

The basic mathematical shape of a linear model is:

$$Y = WX + b$$

- $Y = \{y_1, \ldots, y_m\}$ are labels, so we have m labels.
- Every label Y_i has corresponding features $X_i = \{x_{i,1}, \ldots, x_{i,n}\}$, so we have n features.
- We can store the observations in an (m, n) matrix X
- We can store n weights for every feature in a matrix W
- b is an additional bias weight

The matrix notation is concise. We could write this out in full as:

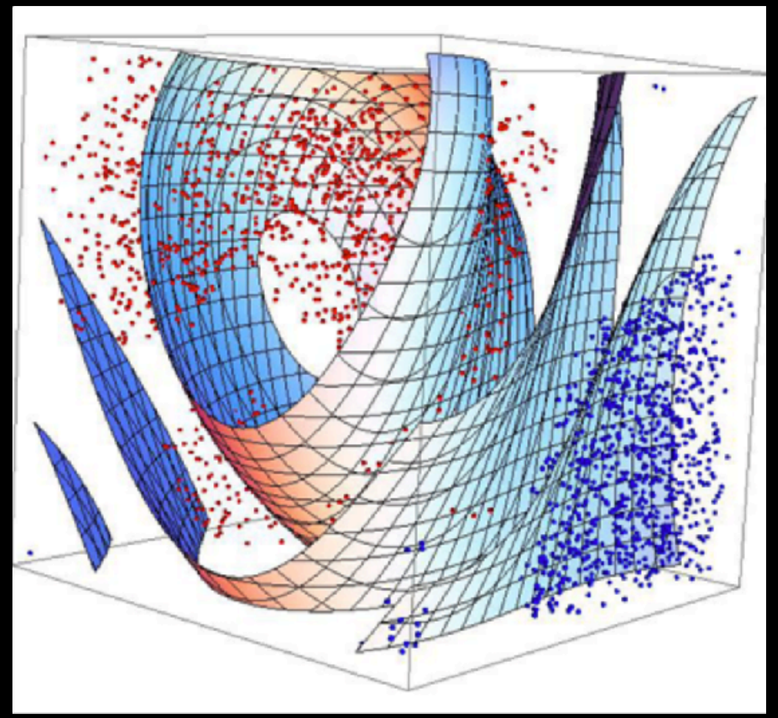
$$y_i = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$$

Non-linear models

A lot of data is non-linear. This means we would need a curved hyperplane.

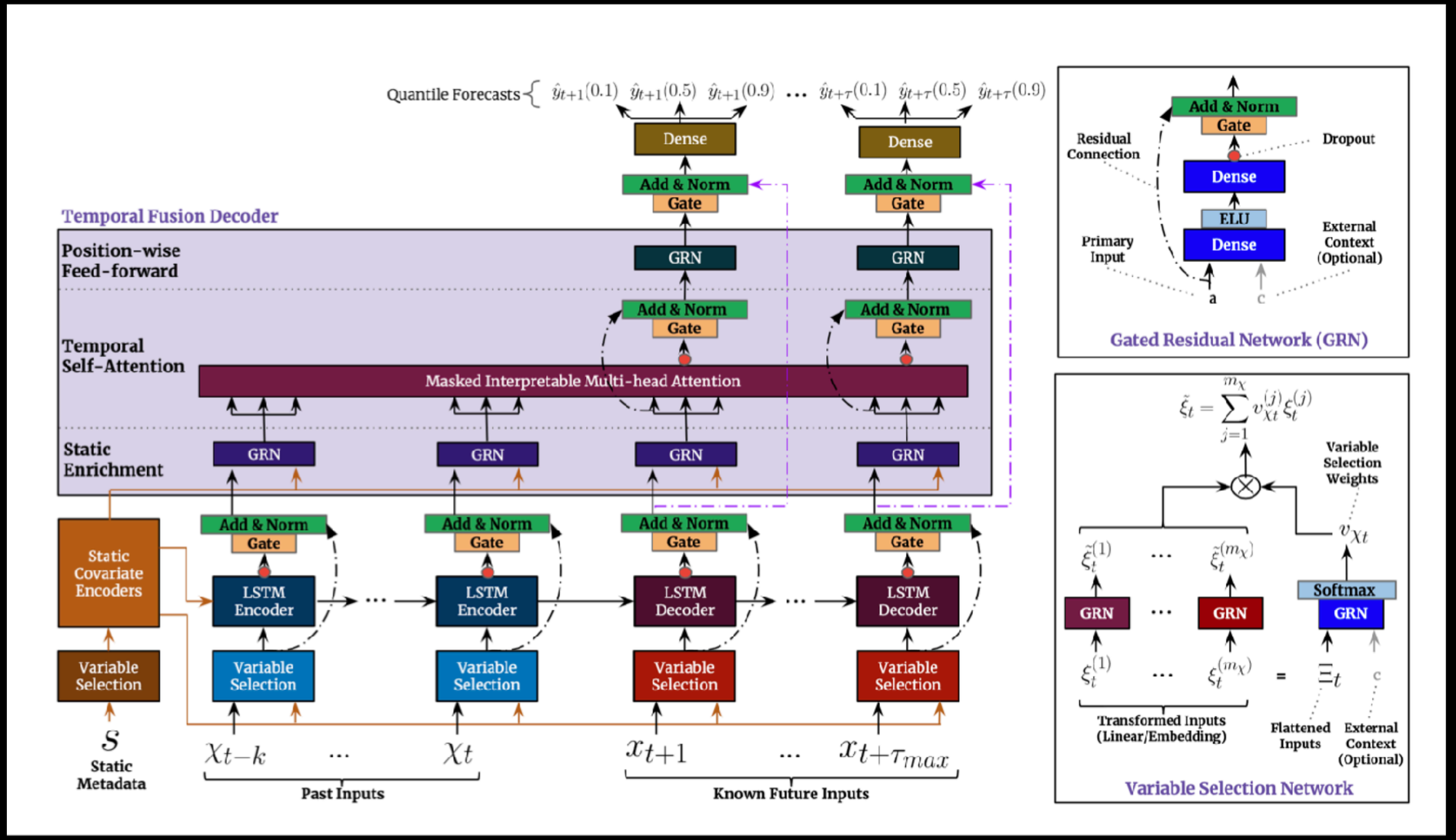
One trick to do this is the kernel-trick, which is commonly used with Support Vector Machines, which we touched upon in the short history.

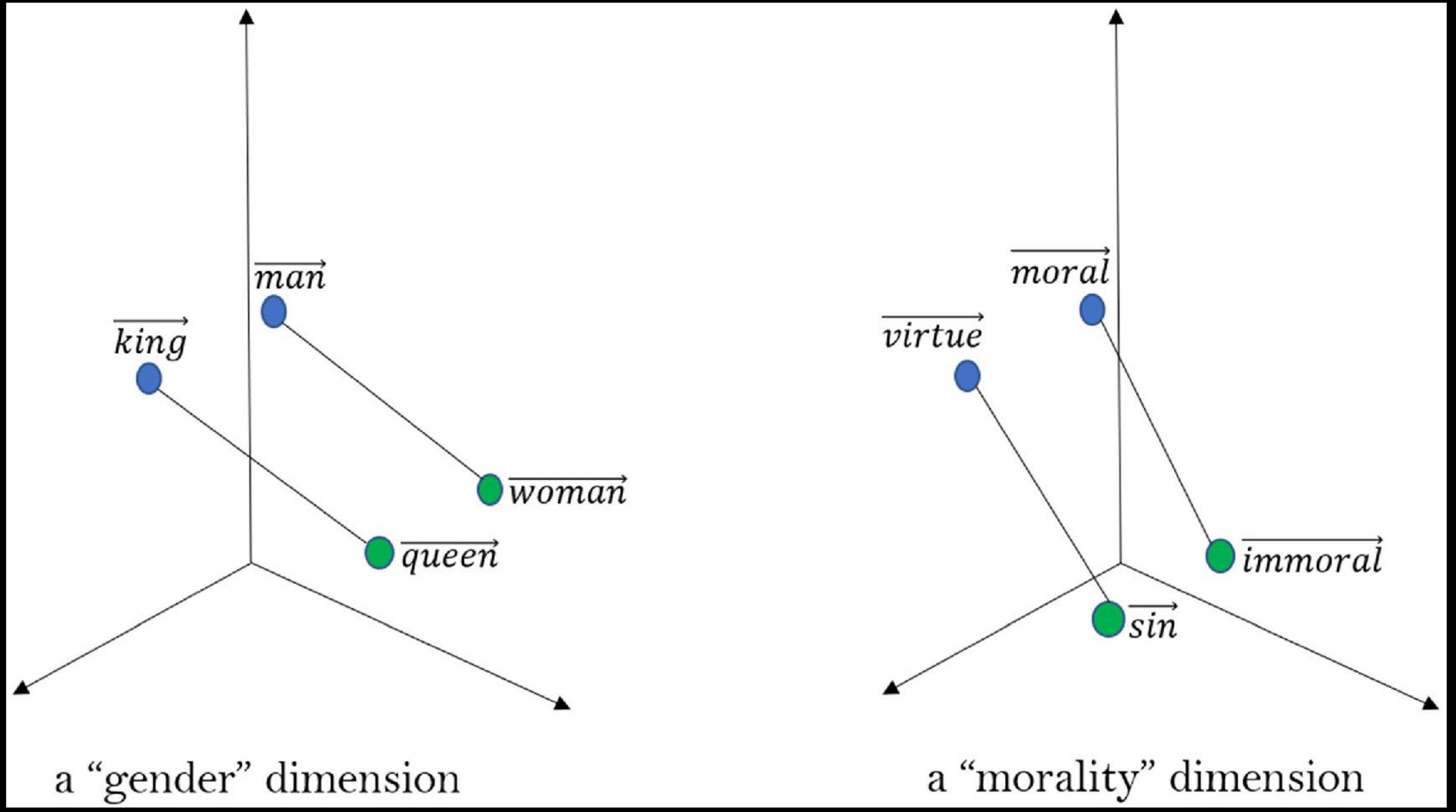
Deep learning has found another trick



Deep Learning

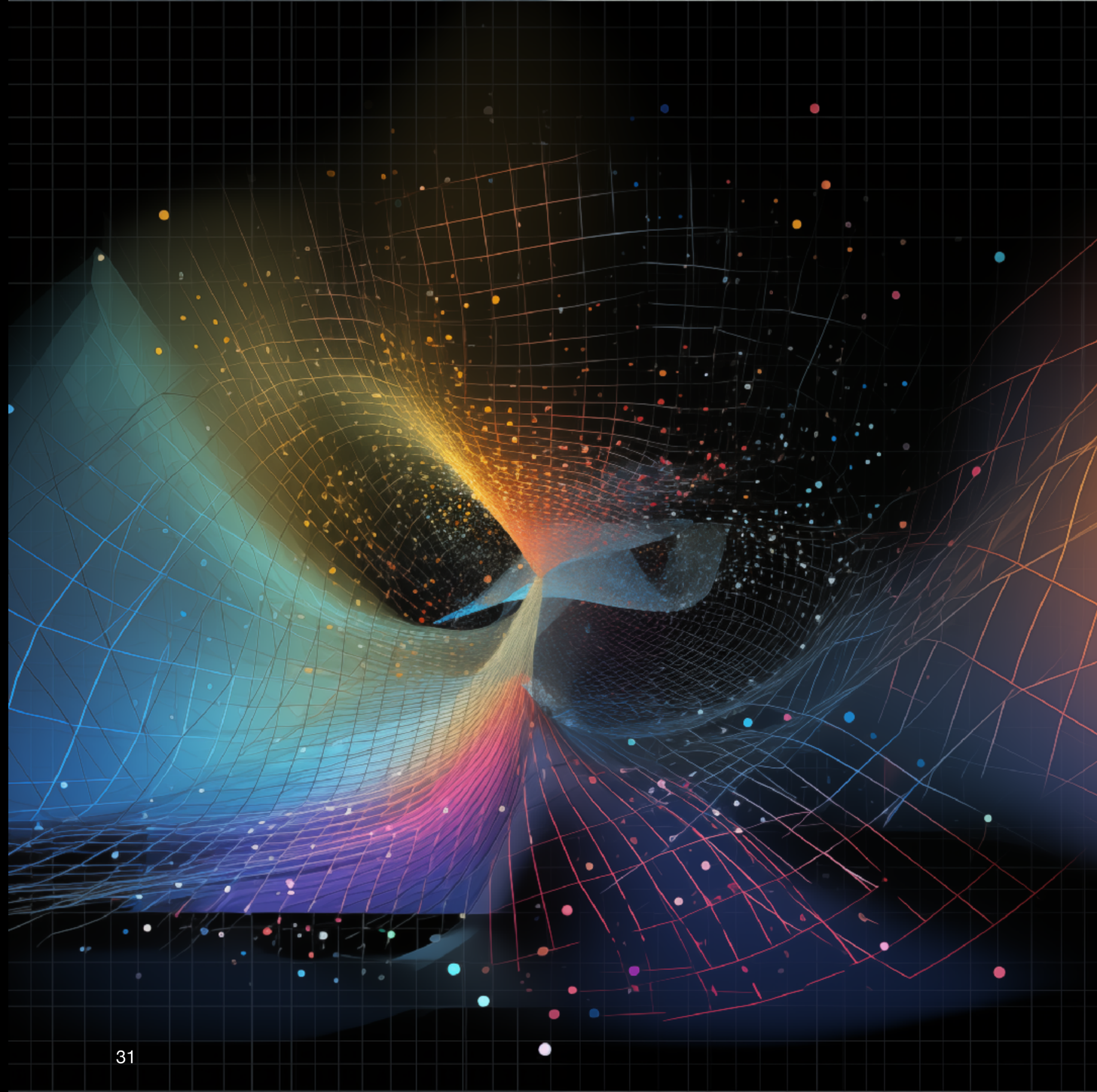
Complexity 2021 - TFT




$\overrightarrow{man}$
$\overrightarrow{king}$
$\overrightarrow{woman}$
$\overrightarrow{queen}$
a “gender” dimension
$\overrightarrow{moral}$
$\overrightarrow{virtue}$
$\overrightarrow{immoral}$
$\overrightarrow{sin}$
a “morality” dimension

LLMs use a semantic vectorspace with 768 dimensions : $\mathbb{R}^{768}$

Atoms in the visible universe : 10^{80}



The Machine Learning Canvas (v0.4)

Designed for: | Designed by: | Date: | Iteration:

Decisions

How are predictions used to make decisions that provide the proposed value to the end-user?

ML task

Input, output to predict, type of problem.

Value Propositions

What are we trying to do for the end-user(s) of the predictive system? What objectives are we serving?

Data Sources

Which raw data sources can we use (internal and external)?

Collecting Data

How do we get new data to learn from (inputs and outputs)?

how

how

PREDICT

LEARN

GOAL

(what, why, who)

Making Predictions

When do we make predictions on new inputs? How long do we have to featurize a new input and make a prediction?

Offline Evaluation

Methods and metrics to evaluate the system before deployment.

Features

Input representations extracted from raw data sources.

Building Models

When do we create/update models with new training data? How long do we have to featurize training inputs and create a model?

how well

Live Evaluation and Monitoring

Methods and metrics to evaluate the system after deployment, and to quantify value creation.

EVALUATE

Licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

Predict

- Decisions: descriptive, predictive, prescriptive
- Task: classification, regression, clustering?
- Evaluation: what are good metrics?

Learn

- Data sources: what is good data?
- Collecting: how do we get new data?
- Features: what kind of information do you need?

Machine learning

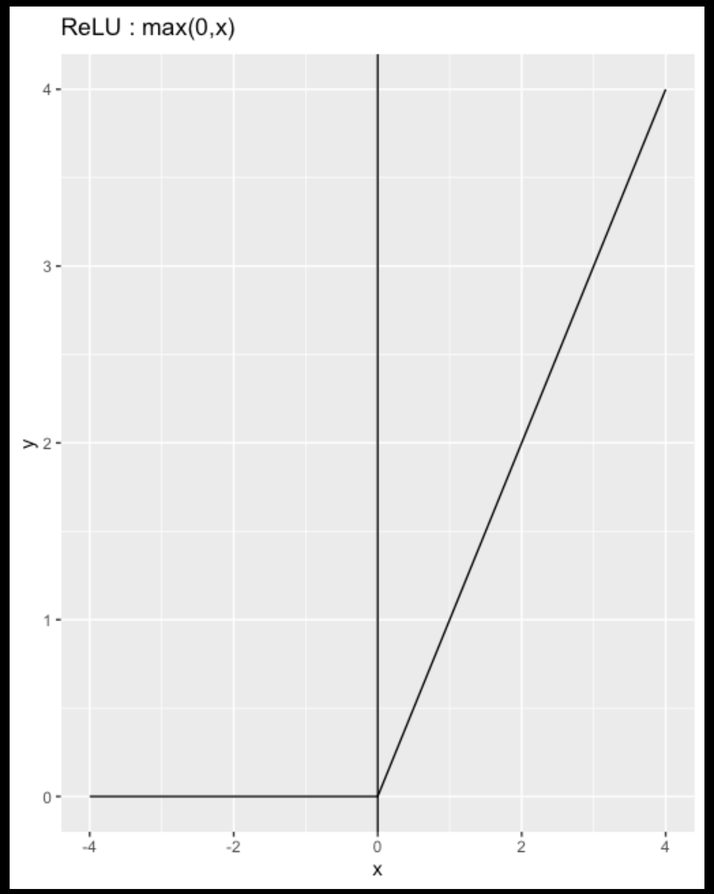
$$f\colon X \to y$$

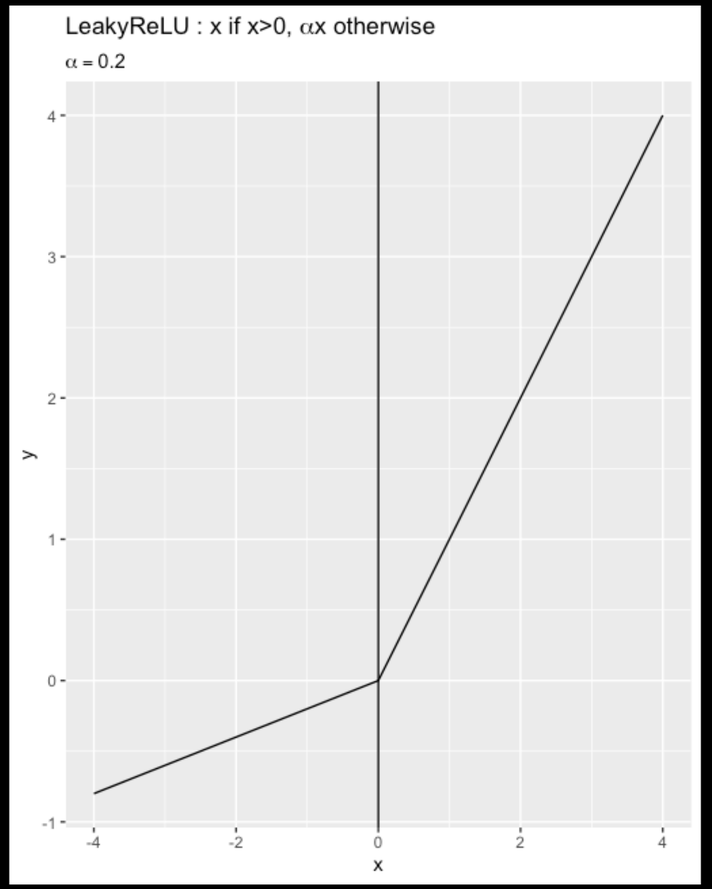
Deep Learning

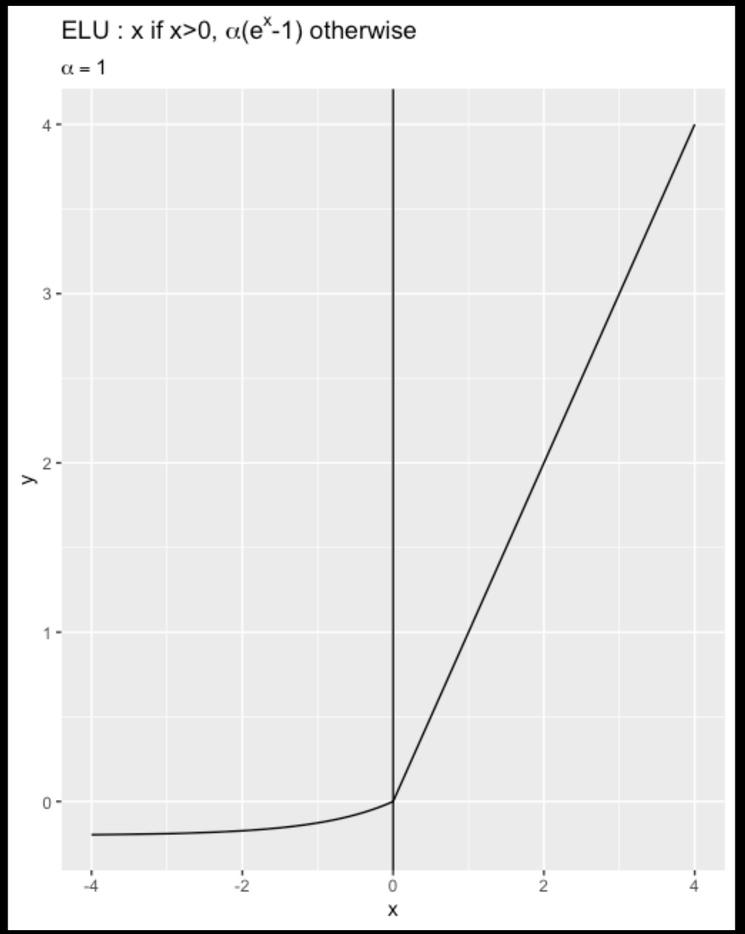
Data	$X = \{\overrightarrow{x_1}, \ldots, \overrightarrow{x_n} \mid \vec{x} \in \mathbb{R}^d\}$
	$y = \{y_1, \ldots, y_n \mid y \in \{0,1\}\}$
Trainable Weights	W, b
(Non)linearity	$f(X) = WX + b \quad \sigma(X) = max(0,X)$
Predict	$\hat{y} = f_n \circ \sigma \circ f_{n-1} \circ \ \ldots \ \circ \sigma \circ f_1$
Loss	$Loss(y, \hat{y})$
Optimize	$w \leftarrow w - \eta \frac{\partial Loss}{\partial W}$

Activations

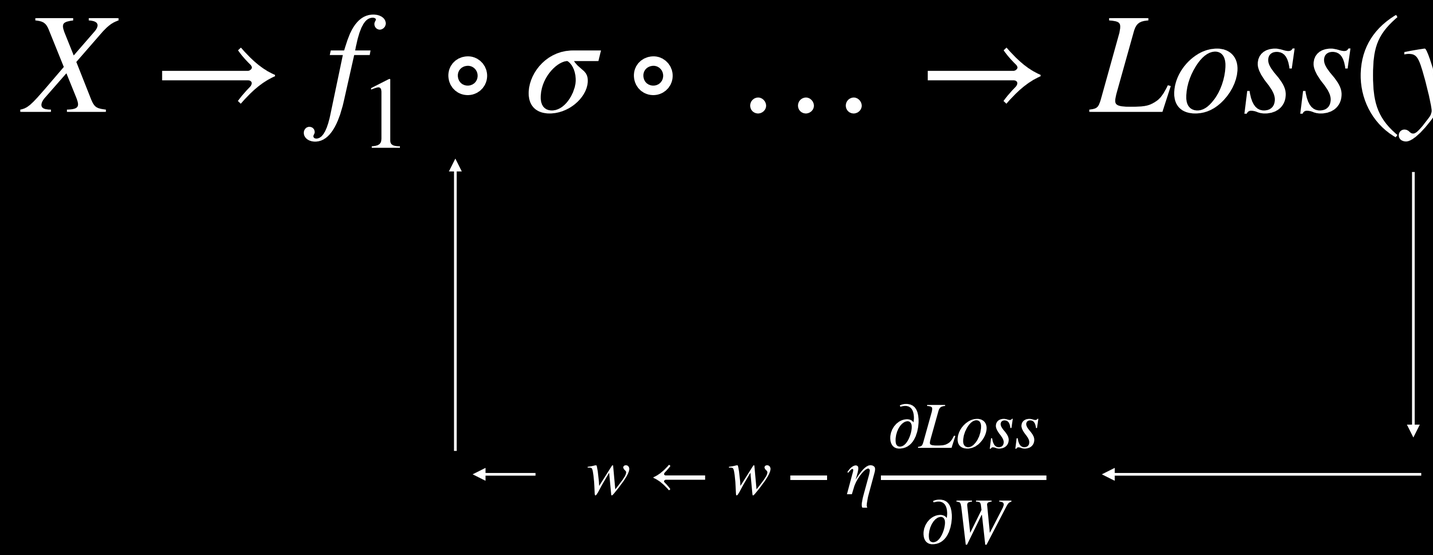
non-linearities







Training


$X \rightarrow f_1 \circ \sigma \circ \ \ldots \ \rightarrow Loss(y, \hat{y})$
$w \leftarrow w - \eta \frac{\partial Loss}{\partial W}$