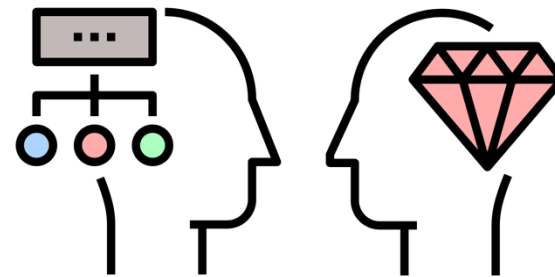


Data-Driven Modelling and Artificial Intelligence

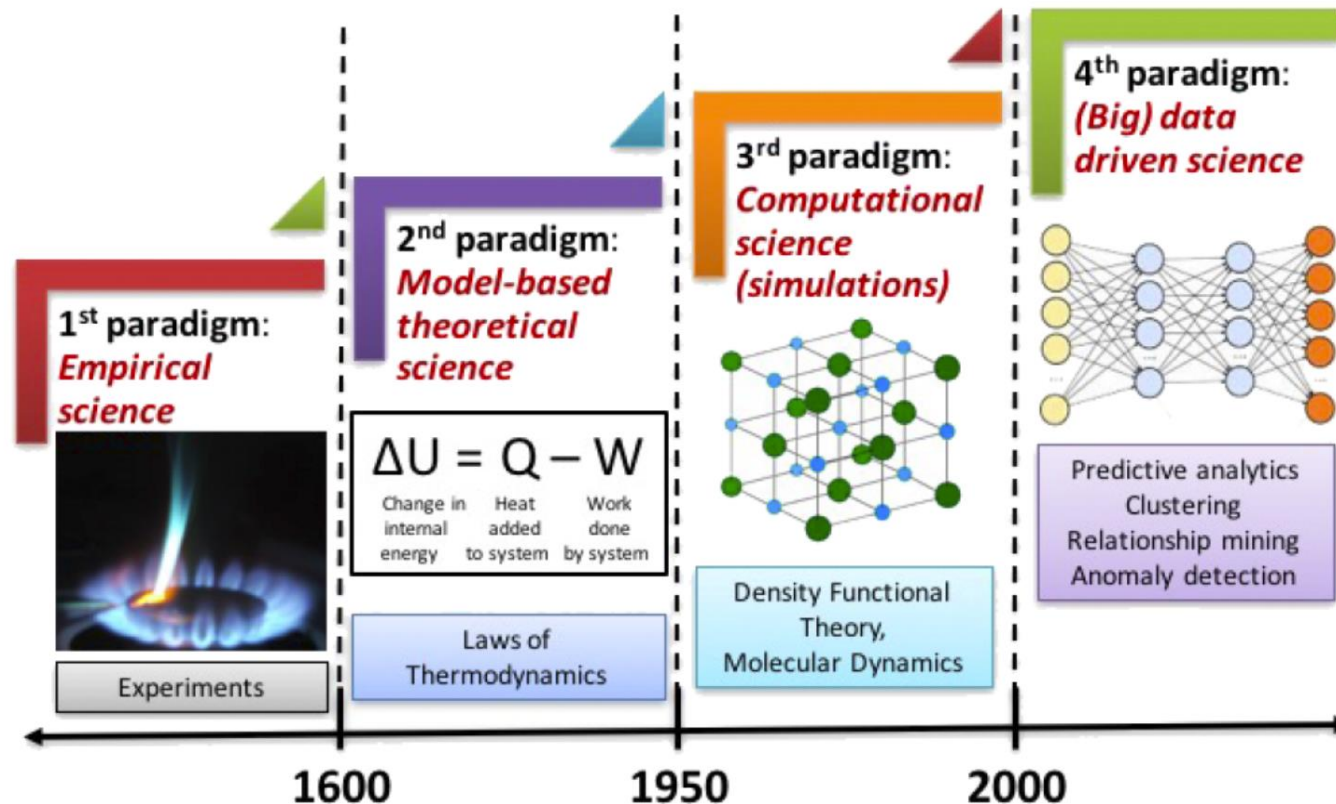
Aron Walsh

TYC Materials Modelling
Course (2025)



New Era of Materials Research

The research toolkit for materials science now includes powerful data-driven approaches



A. Agrawal and A. Choudhary, APL Materials 4, 053208 (2016)

Workflows in Materials Modelling

Input

Composition
and structure

Quantum Mechanics

$$\hat{H}|\Psi\rangle = E|\Psi\rangle$$

electronic
wavefunction

Output
Properties

Input

Composition
and structure

Machine Learning

$$y = f(\mathbf{X}, \Theta)$$

learned
weights

Output
Properties

Input

Target
properties

Inverse Design

$$O(\sigma) = \sum_{\alpha} \omega_{\alpha} |P_{\alpha}(\sigma) - P_{\alpha}^{target}|$$

configuration
property

Output
Composition
and structure

Artificial Intelligence

Computational techniques that mimic human intelligence

ARTIFICIAL INTELLIGENCE (AI)

(entire knowledge field)

MACHINE LEARNING (ML)

(data-driven statistical models)

Supervised

Unsupervised

Reinforcement

DEEP LEARNING

(multi-layered neural networks)

Nobel Prize in Physics (2024) to G. Hinton and J. Hopfield

Lecture Contents

1. Machine Learning Basics
 2. Deep Learning Essentials
 3. Models of Materials
 4. Advances in AI for Science
-

What is Machine Learning (ML)?

Statistical algorithms that learn from training data and build a model to make predictions

Learning types

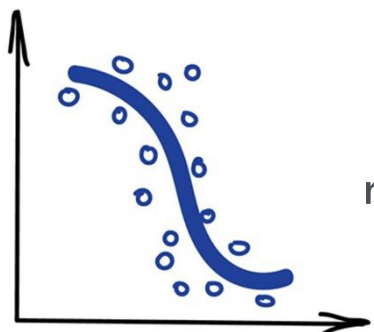
Unsupervised (**identify patterns**), supervised (**use patterns**), reinforcement (**maximise reward**)

Data types

Materials features can be binary (e.g. **stability**), categorical (e.g. **symmetry**), integer (e.g. **stoichiometry**), continuous (e.g. **rate**)

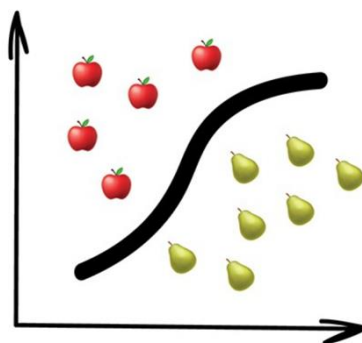
What is Machine Learning (ML)?

Statistical algorithms that identify and use patterns in multi-dimensional datasets



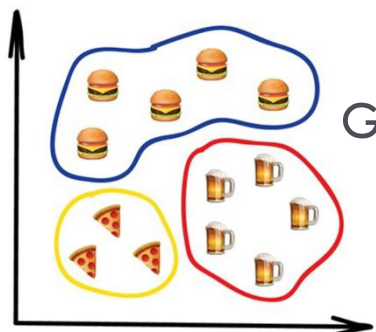
Regression

Predict a value, e.g. regression to extract a reaction rate



Classification

Predict a category, e.g. decision trees to predict reaction outcome



Clustering

Group by similarity, e.g. high-throughput crystallography



Reinforcement Learning

Maximise reward, e.g. reaction conditions to optimise yield

ML Model Map

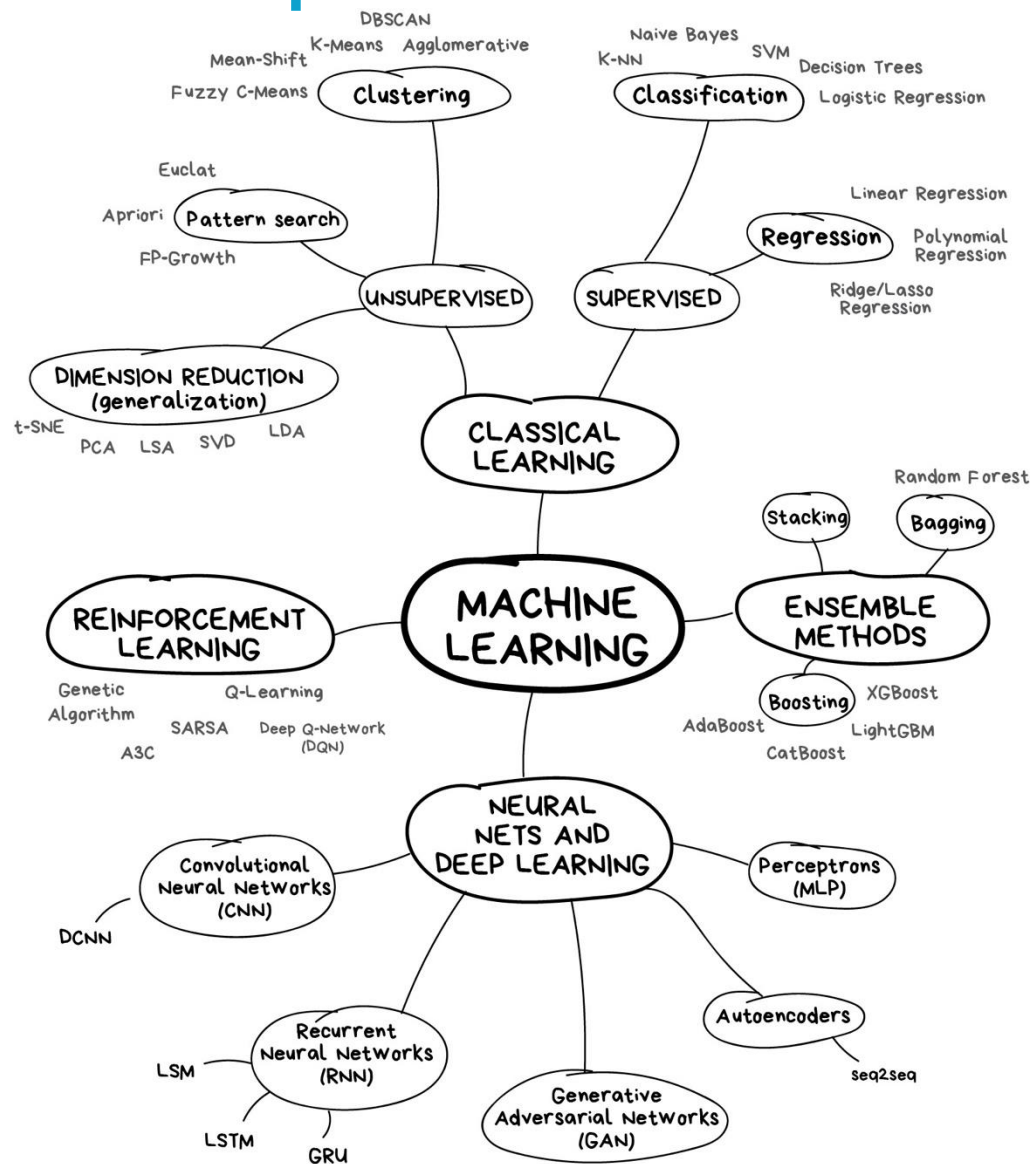
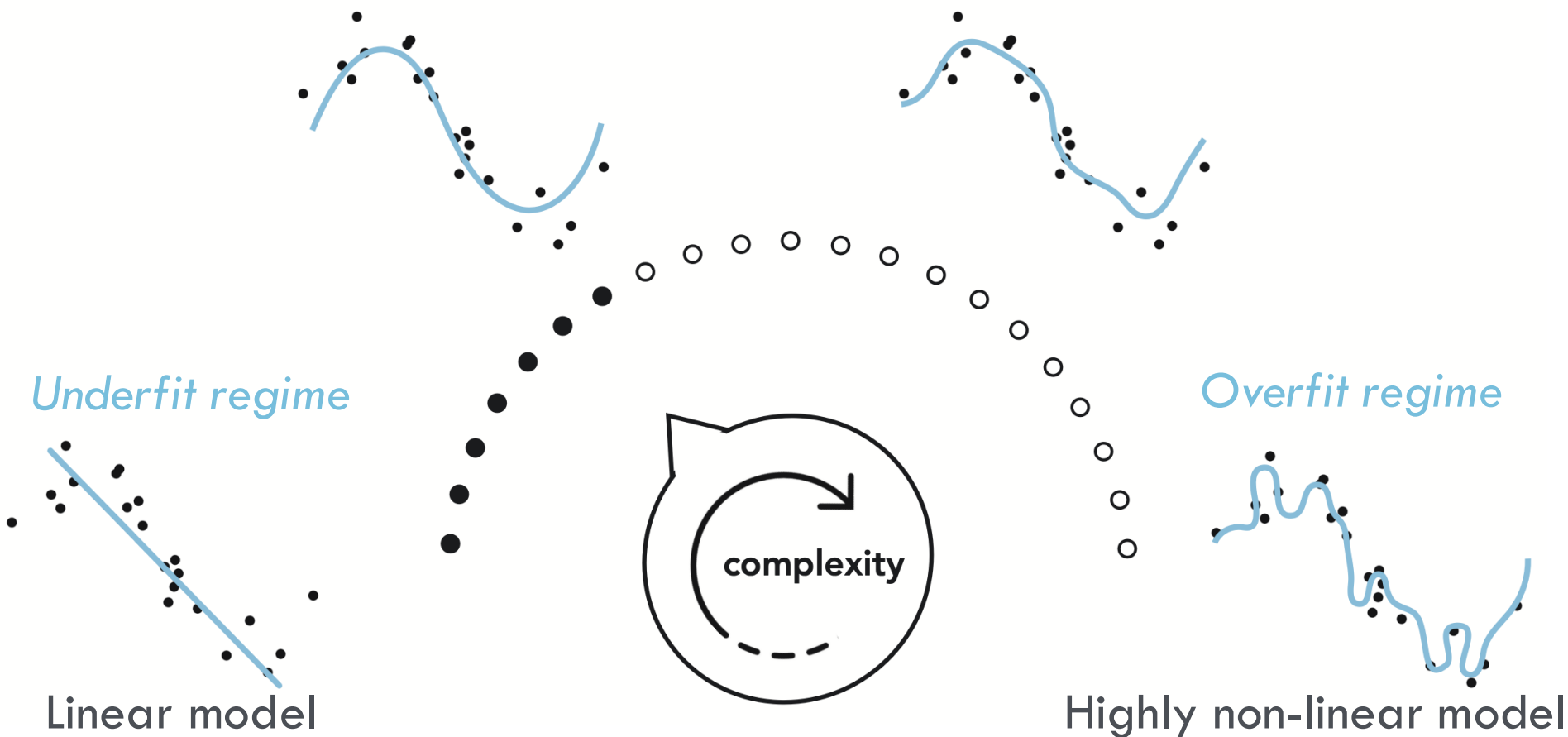


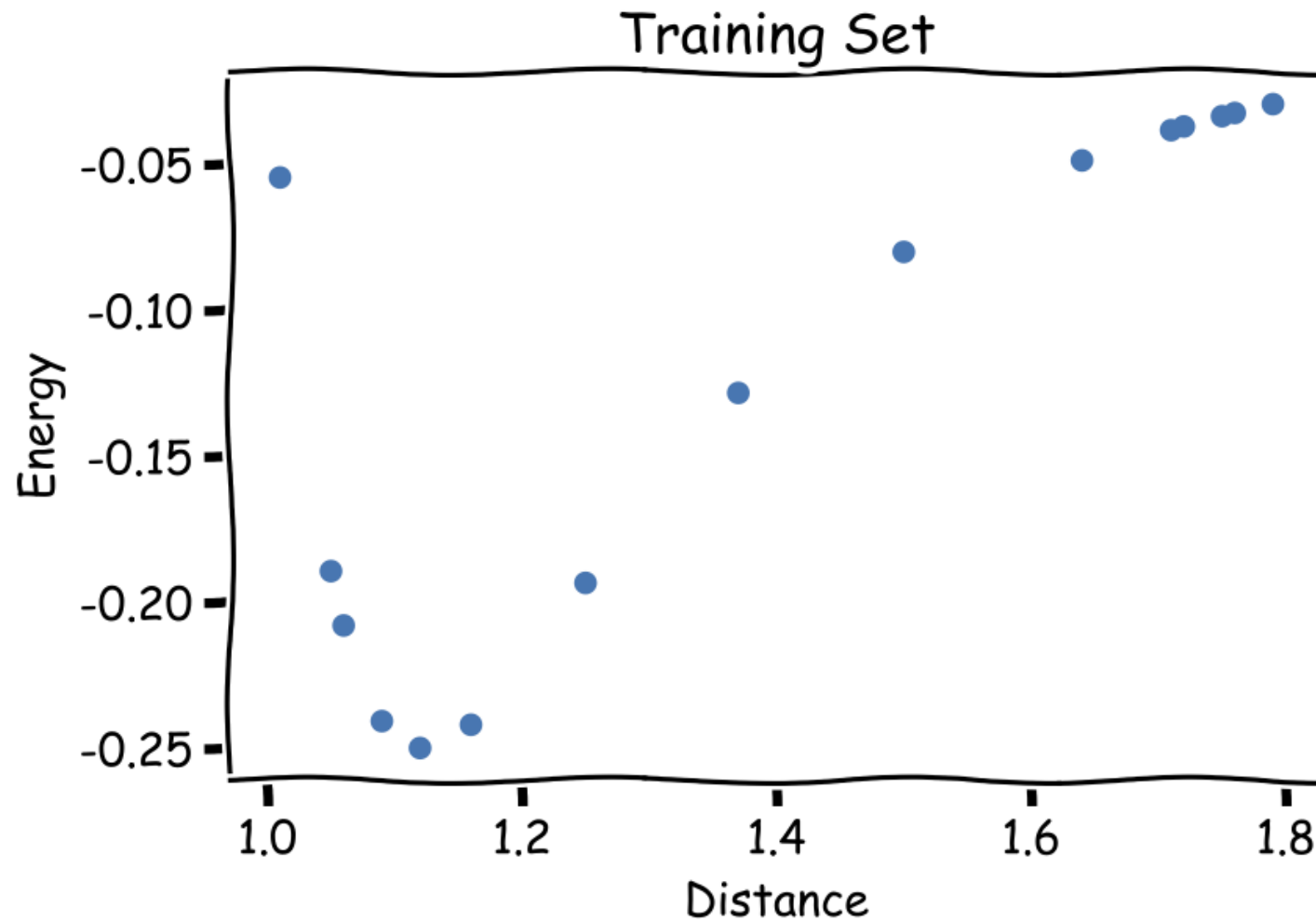
Image from https://vas3k.com/blog/machine_learning

ML ~ Function Approximation

Model selection, training, and testing tunes a “complexity dial” for your problem of interest



Function Approximation



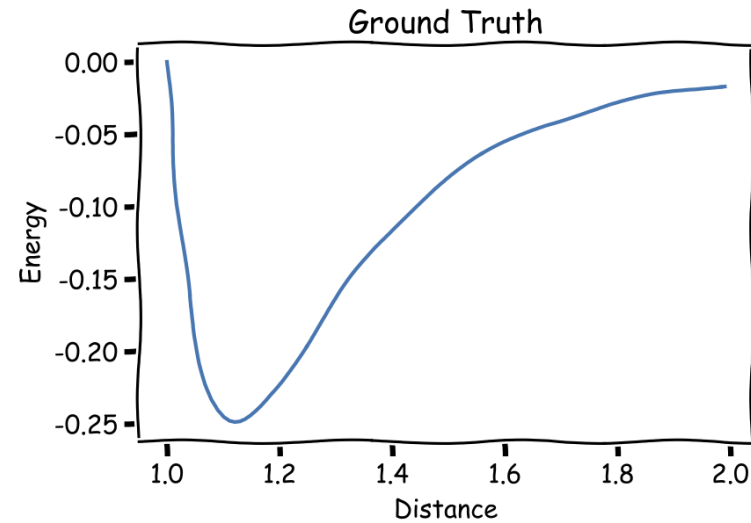
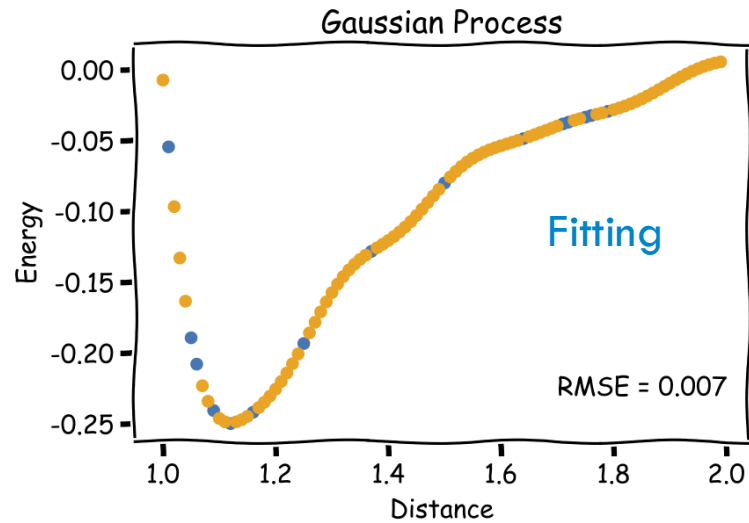
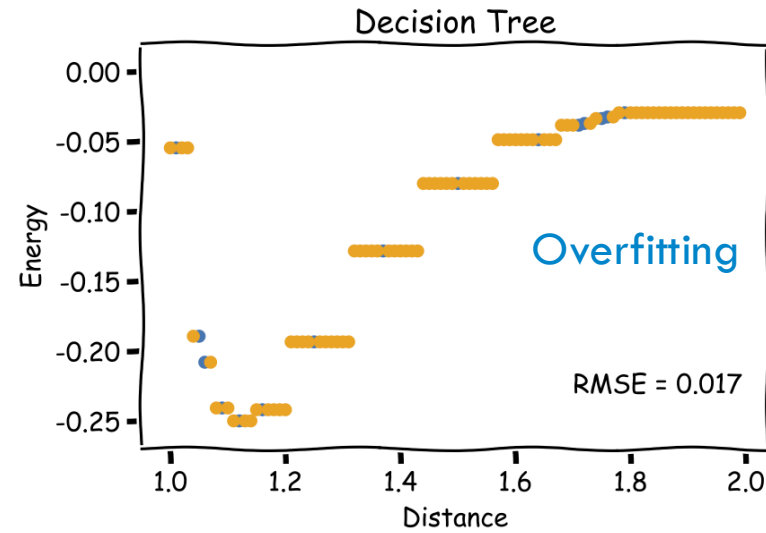
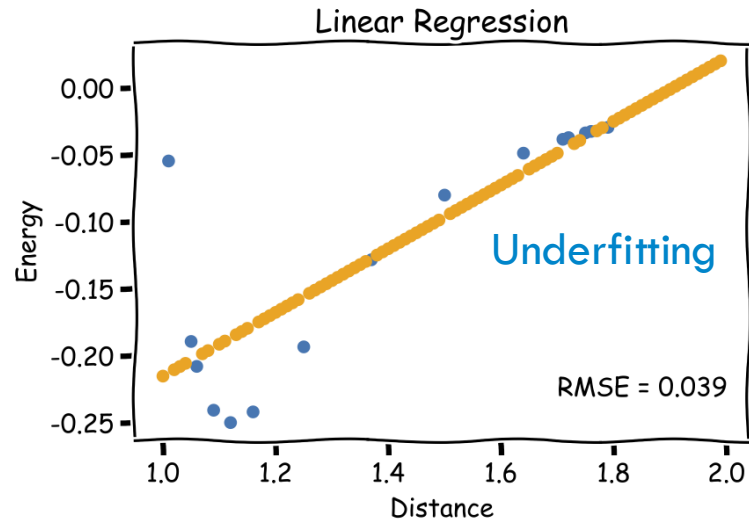
You should recognise the underlying function from undergraduate classes

Function Approximation

```
def truth(r):  
    ε = 1  
    v = (ε/r)**12 - (ε/r)**6  
    return v  
  
xvals = np.arange(1, 2, 0.01)  
yvals = truth(xvals)
```

My reference function to generate data for model training and testing

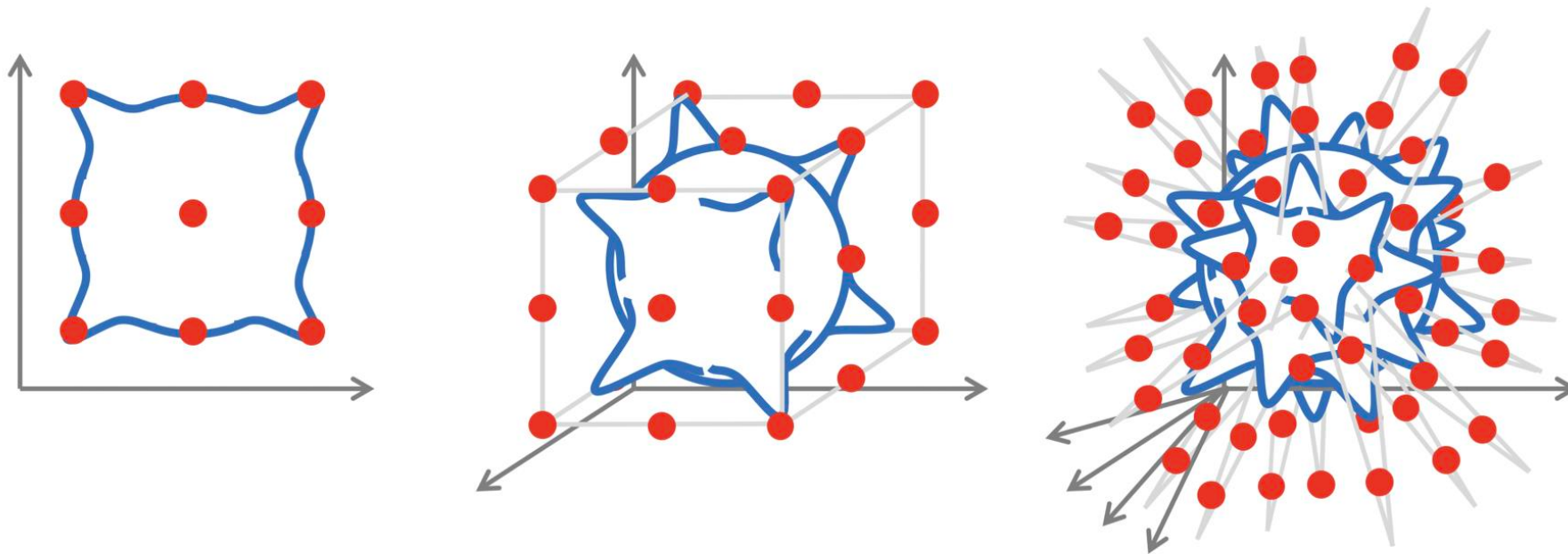
Function Approximation



Default parameters with the scikit-learn Python package; Root mean square error (RMSE)

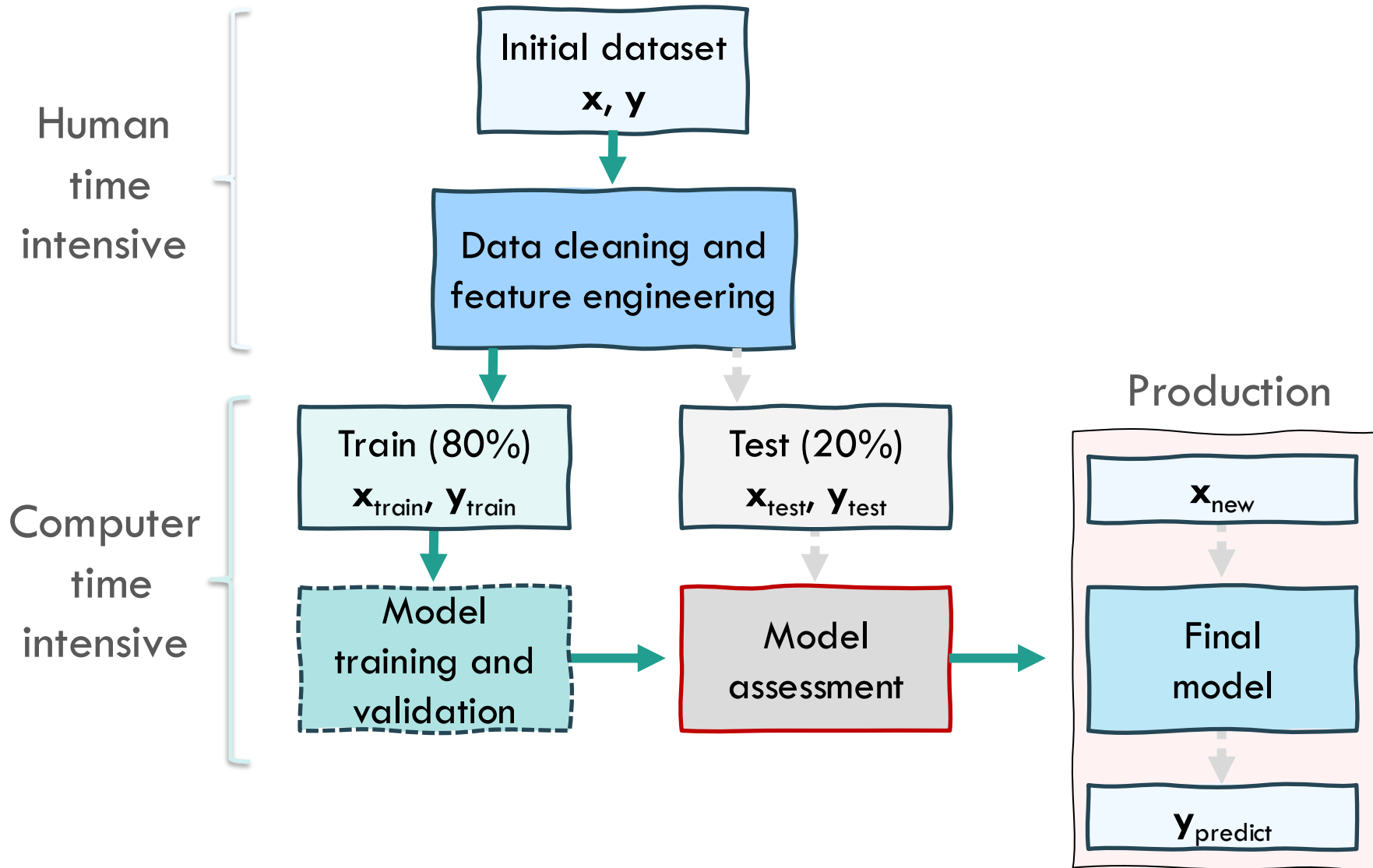
Function Approximation

Standard expansions work in low dimensions (D).
Real problems face the **“curse of dimensionality”**



An exponential increase in the data requirements
needed to cover the parameter space effectively, $O(e^D)$

Typical Supervised ML Workflow



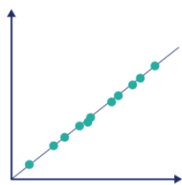
The exact workflow depends on the type of problem and available data

Correlation Coefficient (r)

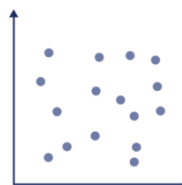
Describes the strength of the relationship between two variables (e.g. “ground truth” vs predicted values)

$$r \in [-1,1]$$

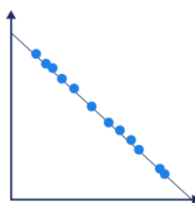
Positive: variables change in the same direction



Zero: no relationship between the variables



Negative: variables change in opposite directions



Pearson correlation*

$$r_{xy} =$$

$$\frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

Reminder: correlation does not imply causation

*Outlined by Auguste Bravais (1844); <https://bit.ly/3Kv75GJ>

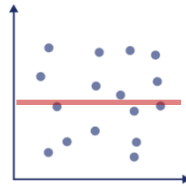
Coefficient of Determination (r^2)

Measure of the goodness of fit for a model.

Describes how well that known data is approximated

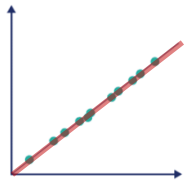
$$r^2 \in [0,1]$$

Zero: baseline model with no variability that predicts \bar{y}



0.5: 50% of the variability in y is accounted for

One: model matches observed values of y exactly



Three equivalent definitions

$$r^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

$$r^2 = 1 - \frac{\sum_{i=1}^n (y_i - y_i^{predicted})^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

$$r^2 = 1 - \frac{\sum_{i=1}^n (e_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

Note: a unitless metric. Alternative definitions are sometimes used

Classification Metrics

Confusion (or error) matrix provides a summary of classification model performance

		Predicted class	
		+	-
Actual class	+	True positive (TP)	False negative (FN)
	-	False positive (FP)	True negative (TN)

$$\begin{bmatrix} 70 & 0 \\ 0 & 30 \end{bmatrix}$$

Perfect model to classify metals and insulators
($N = 100$)

$$\begin{bmatrix} 66 & 4 \\ 8 & 22 \end{bmatrix}$$

My best model

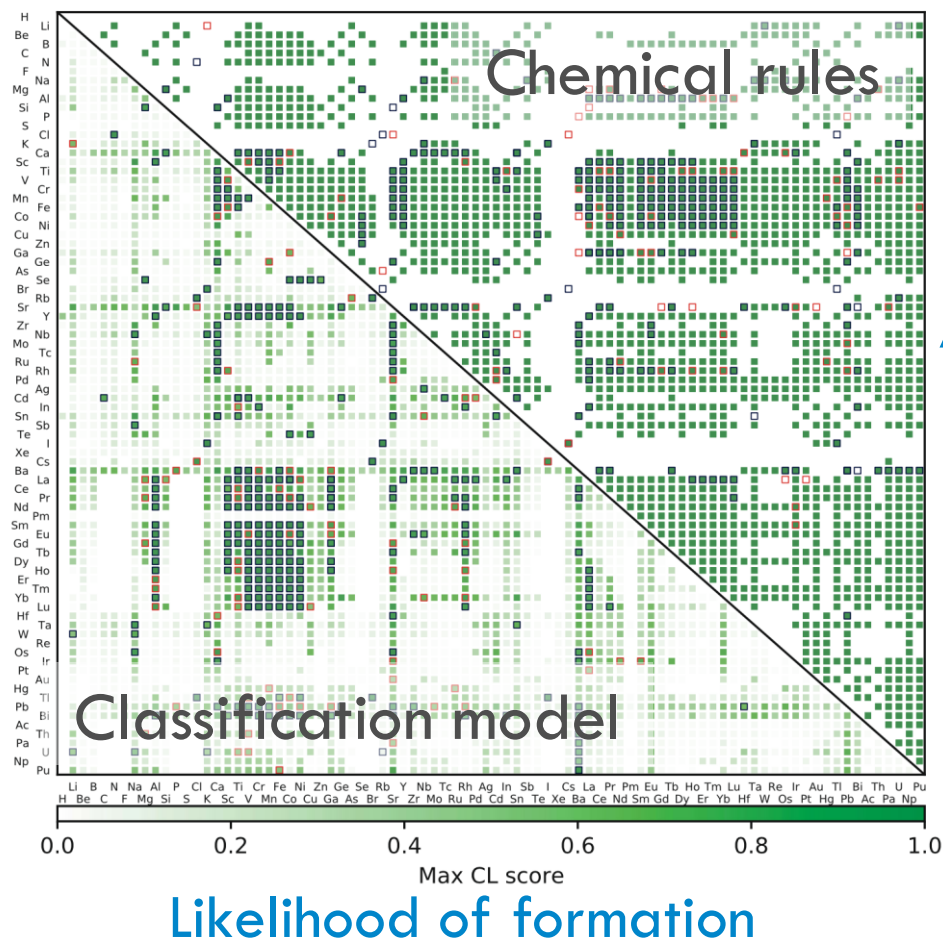
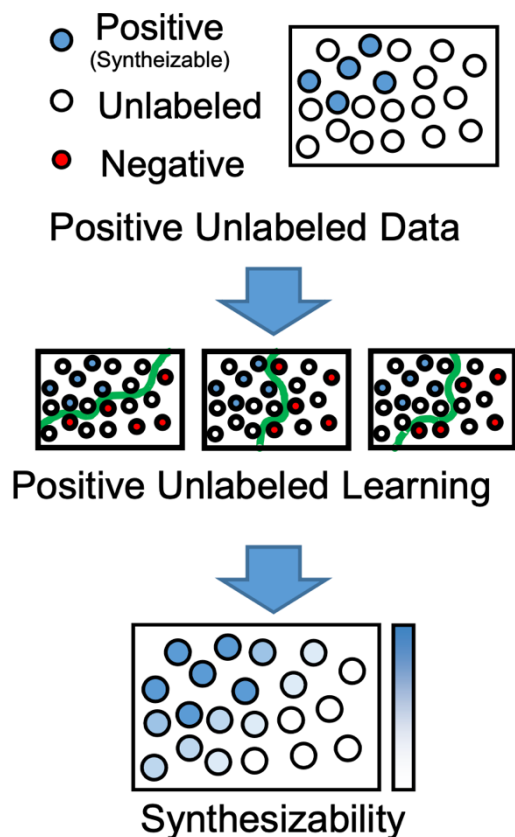
Accuracy = Correct/Total

$$(70+30)/100 = 100 \%$$

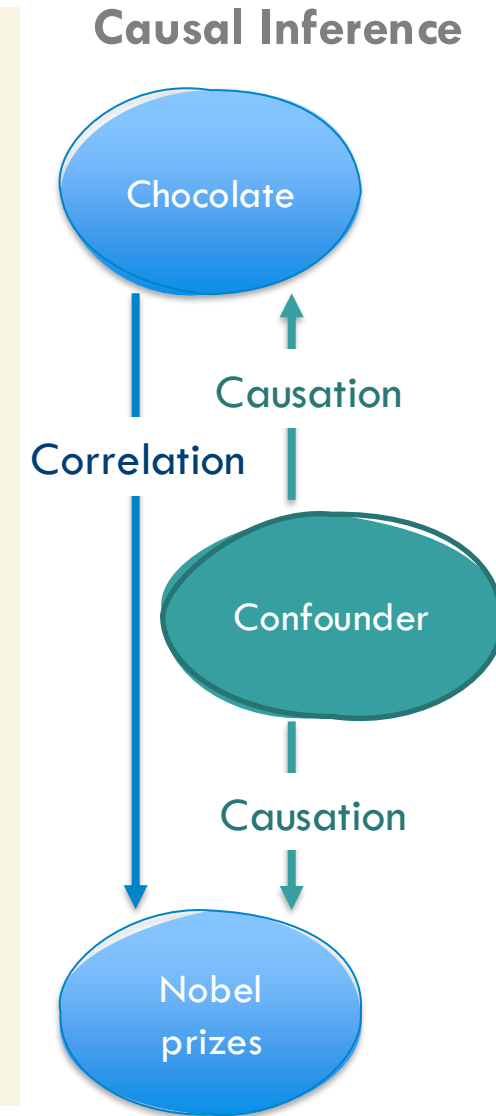
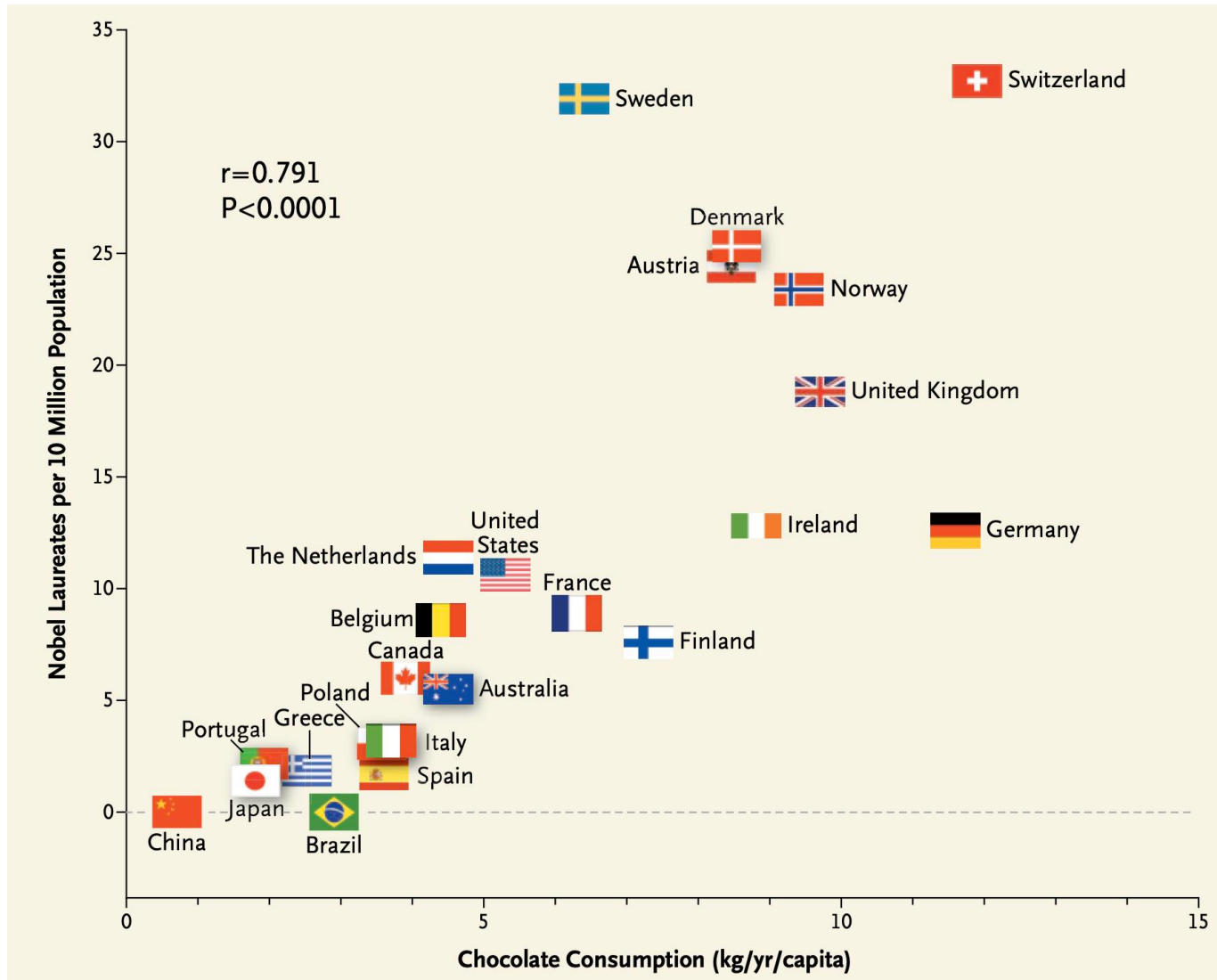
$$(66+22)/100 = 88 \%$$

Application to “Synthesizability”

Learn from known materials (positive samples)
and unknown materials (unlabelled samples)



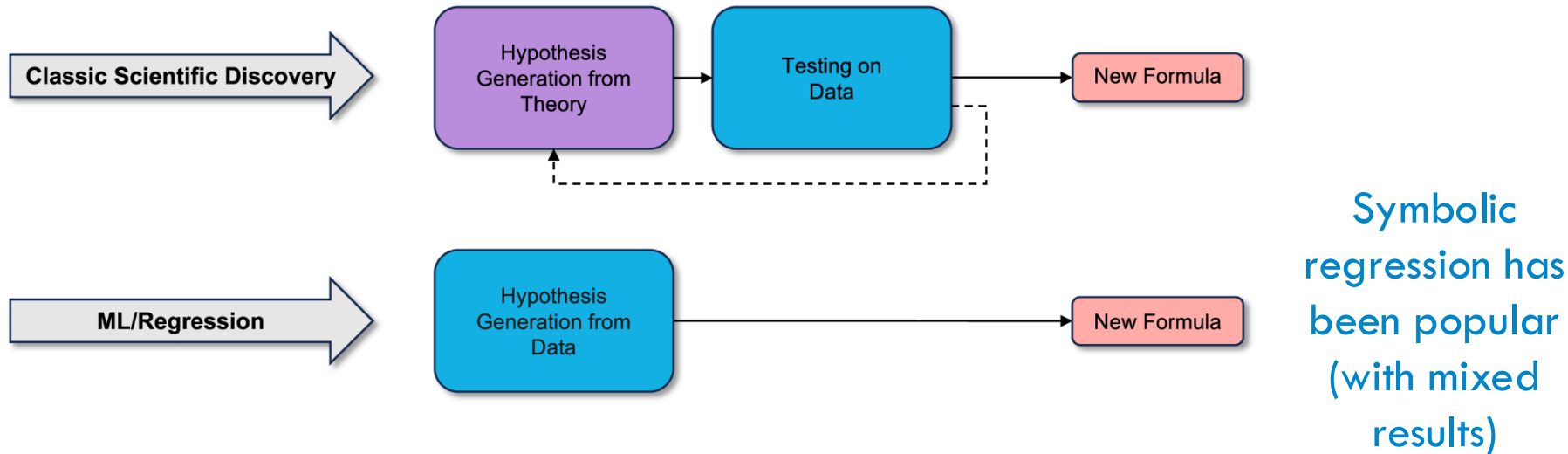
Correlation, Causation...



F. Messereli, New England Journal of Medicine 367, 1562 (2012)

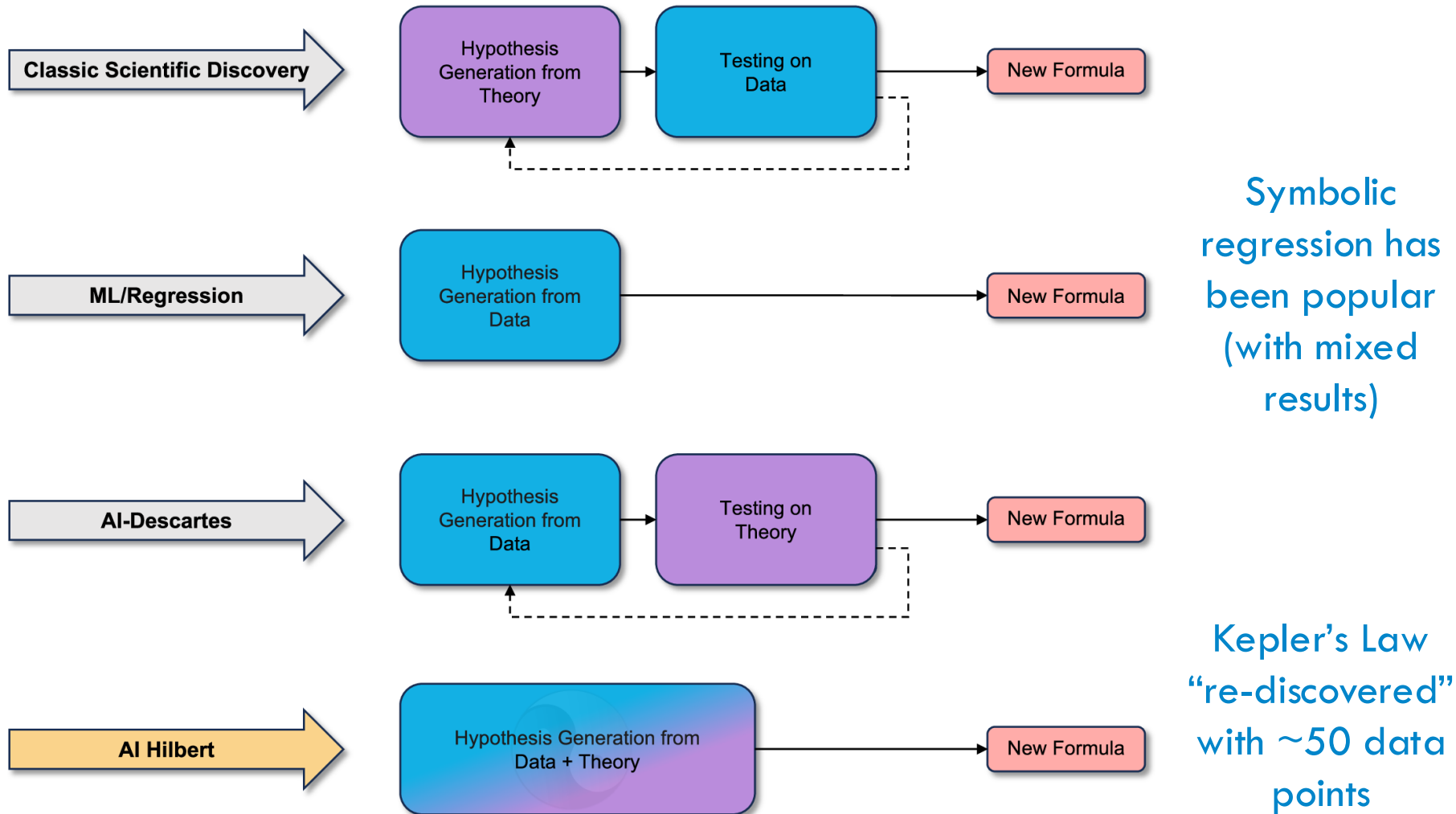
Towards Scientific Rule Discovery

Combining data with background knowledge



Towards Scientific Rule Discovery

Combining data with background knowledge



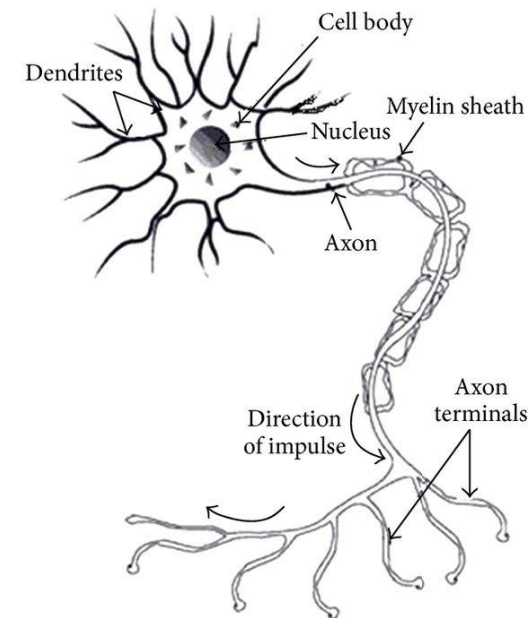
Lecture Contents

1. Machine Learning Basics
 - 2. Deep Learning Essentials**
 3. Models of Materials
 4. Advances in AI for Science
-

Artificial Neuron

Neurons transmit chemical and electrical signals in the brain. Artificial neurons mimic this behaviour using mathematical functions

Biological neuron	Artificial neuron
Cell nucleus	Node
Dendrites	Input
Synapse	Weights (Interconnects)
Axon	Output



The human brain has $\sim 10^{11}$ neurons and 10^{15} synapses ($\sim 10^{15}$ FLOPS)

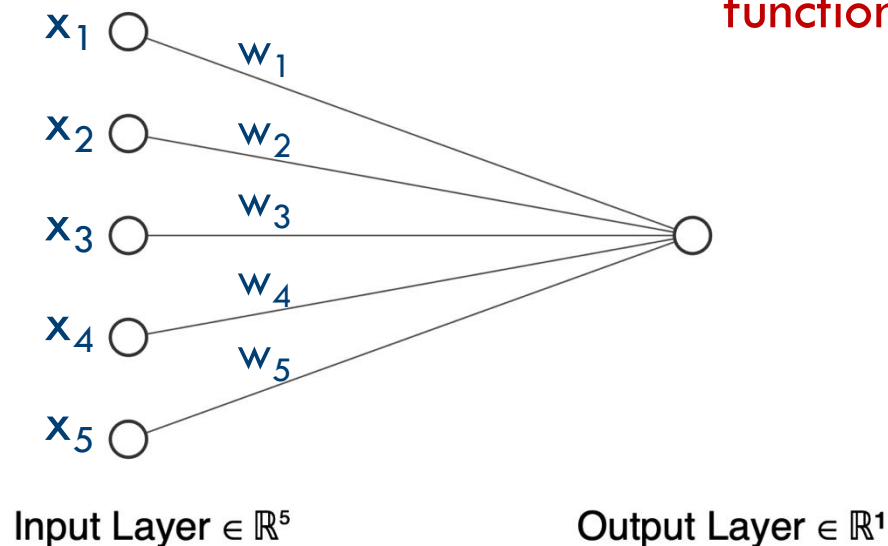
Image: BioMed Research International

Artificial Neuron

The perceptron is a binary neural network classifier:
weighted inputs produce an output of 0 or 1

$$\text{Output } y = f(\text{Weighted input } \mathbf{w} \cdot \mathbf{x} + \text{Bias (constant) } b)$$

Activation function



if $\sum x_i w_i + b > \text{threshold}$:

output = 1

else

output = 0

Weights are adjusted to
minimise the model error

Activation Function

$w \cdot x + b$ is simply a linear combination.

Activation function $f(w \cdot x + b)$ introduces non-linearity





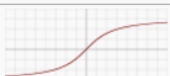

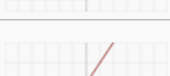

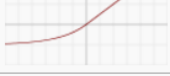
	Activation function	Derivative		
Identity		$f(x) = x$	$f'(x) = 1$	
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$	Perceptron model
Logistic (a.k.a. Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	Popular in early models
Tanh		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$	
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$	
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	Common for deep learning
Parameteric Rectified Linear Unit (PReLU) ^[2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	
Exponential Linear Unit (ELU) ^[3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$	Common for deep learning

Image from <https://towardsdatascience.com>

Activation Function

Corresponding weights and thresholds are learned (fit) during model training





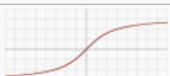

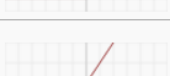

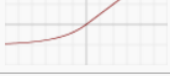
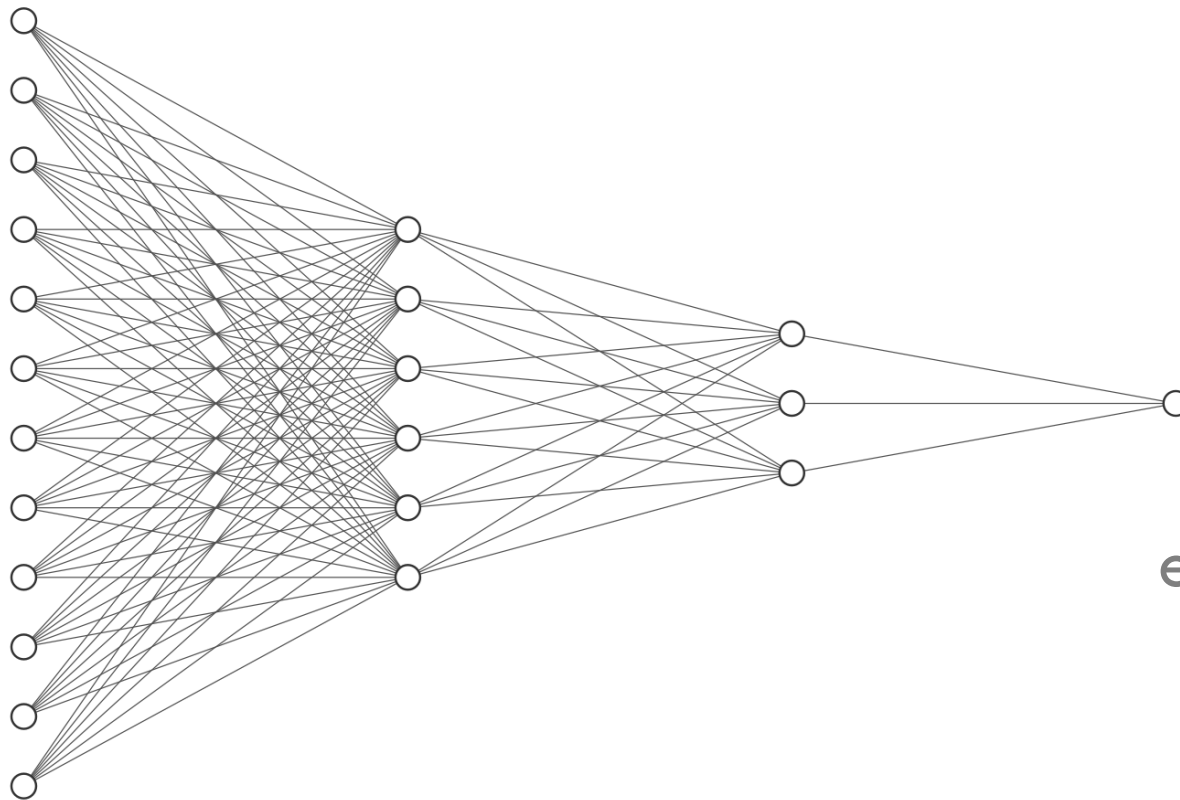
	Activation function	Derivative		
Identity		$f(x) = x$	$f'(x) = 1$	
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$	Perceptron model
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	Popular in early models
Tanh		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$	
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$	
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	Common for deep learning
Parameteric Rectified Linear Unit (PReLU) ^[2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	
Exponential Linear Unit (ELU) ^[3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$	Common for deep learning

Image from <https://towardsdatascience.com>

Neural Network Architecture

Basic neural network: One or two layers

Deep neural network: Three or more layers



**Three layer
model**

(input layer is
excluded in counting)

Input Layer $\in \mathbb{R}^{12}$

Hidden Layer $\in \mathbb{R}^6$

Hidden Layer $\in \mathbb{R}^3$

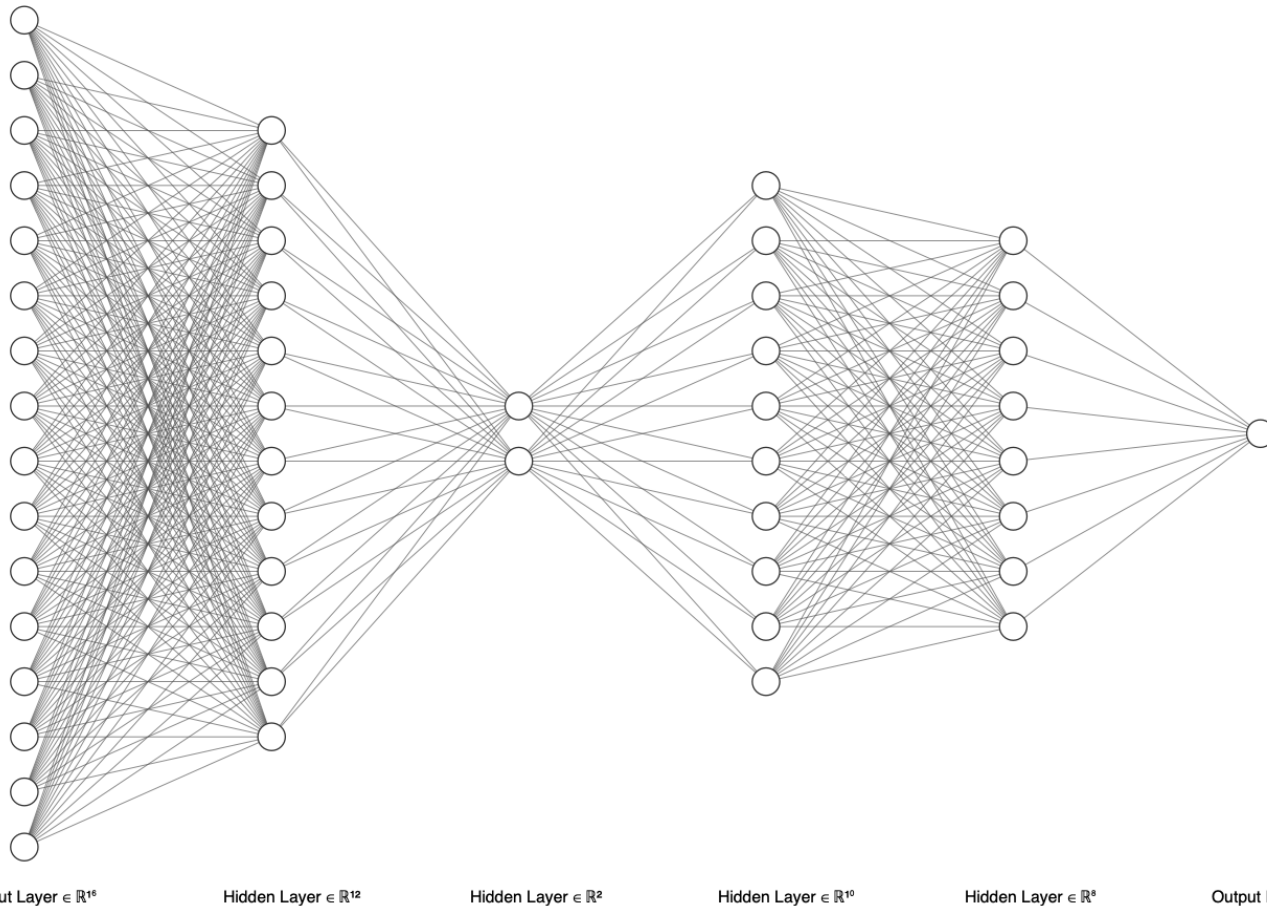
Output Layer $\in \mathbb{R}^1$

Image generator: <https://alexlenail.me/NN-SVG>

Neural Network Architecture

Basic neural network: One or two layers

Deep neural network: Three or more layers



**Five layer
model**

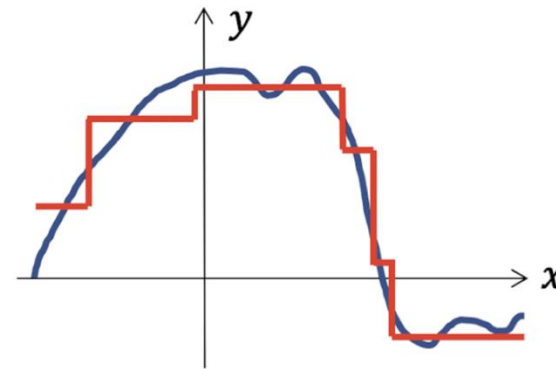
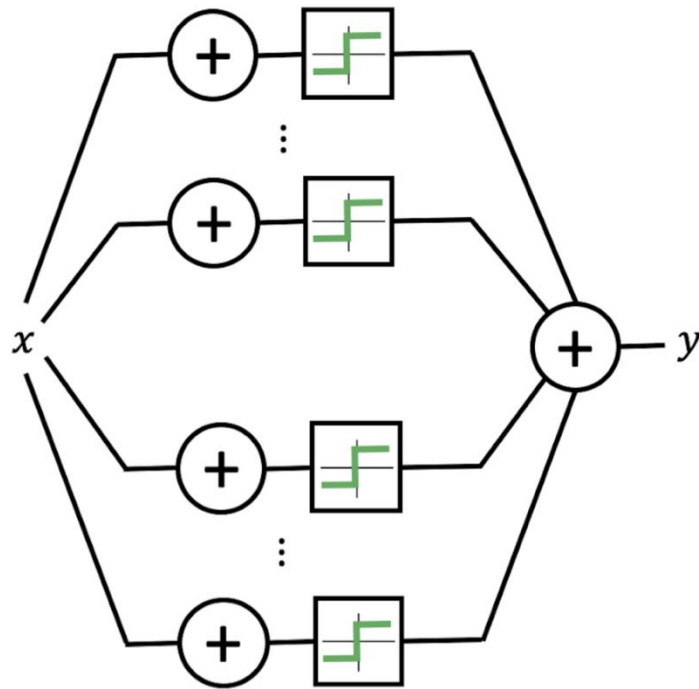
**Note the layer 2
bottleneck**

*Why? Compressed
representation*

Image generator: <https://alexlenail.me/NN-SVG>

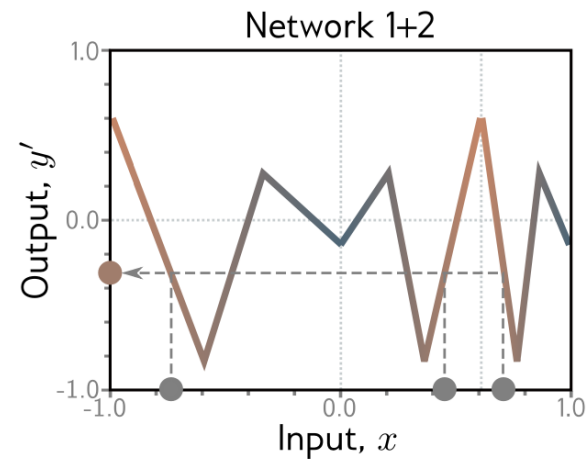
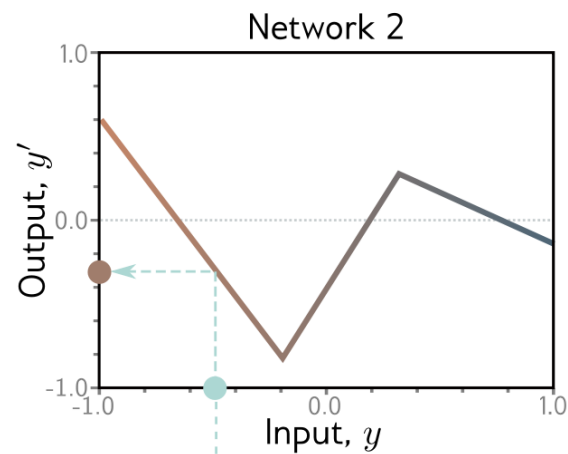
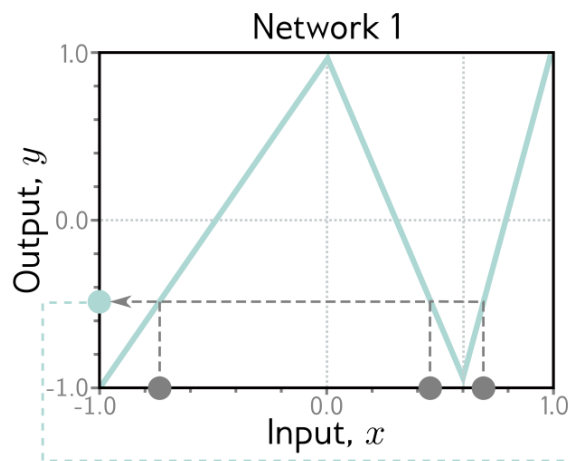
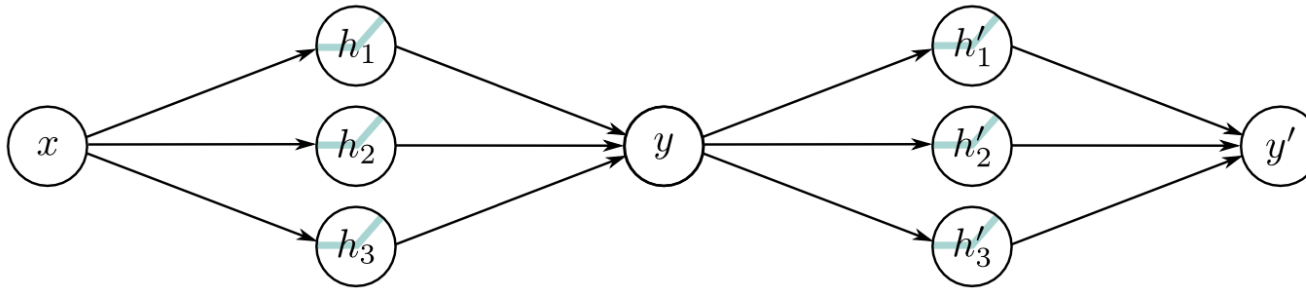
Universal Function Approximators

Multilayer neural networks can approximate any continuous function to any desired accuracy



Practical performance will depend on the number of hidden layers, choice of activation function, and training data

Universal Function Approximators

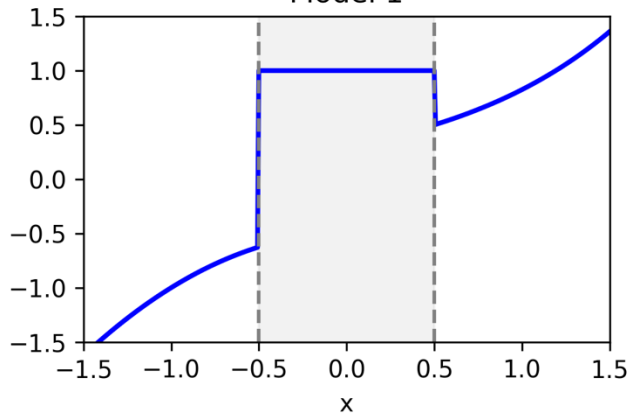


The combination of two single-layer networks
with three hidden ReLU units each

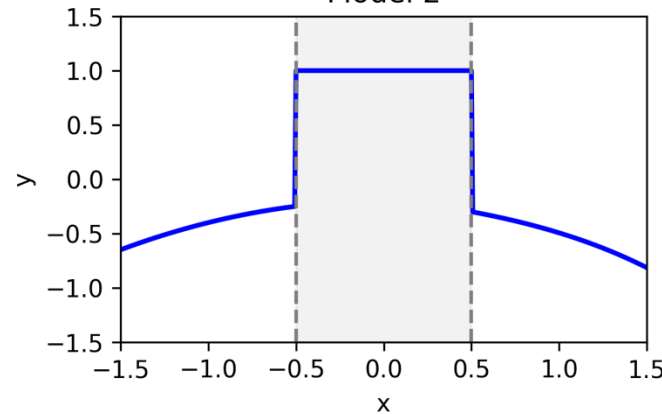
Universal Function Approximators

Extrapolation outside training region is not guaranteed (no fixed functional form)

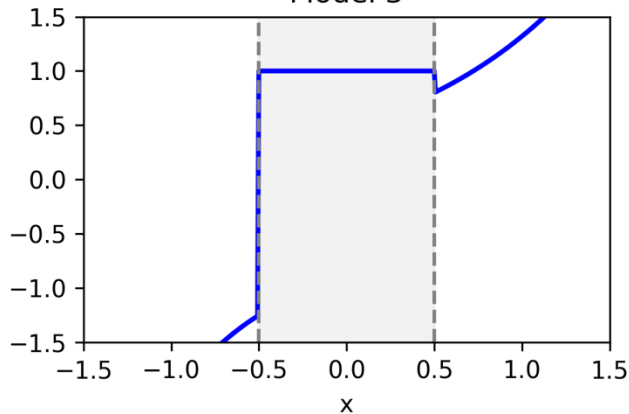
Model 1



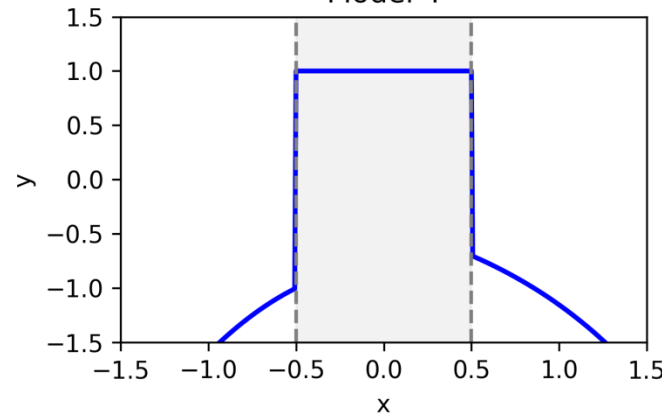
Model 2



Model 3



Model 4



Four models with the same performance (in grey region)

A major issue with machine learning force fields...

Be cautious with out-of-distribution (OOD) applications

Types of Layer in Deep Learning

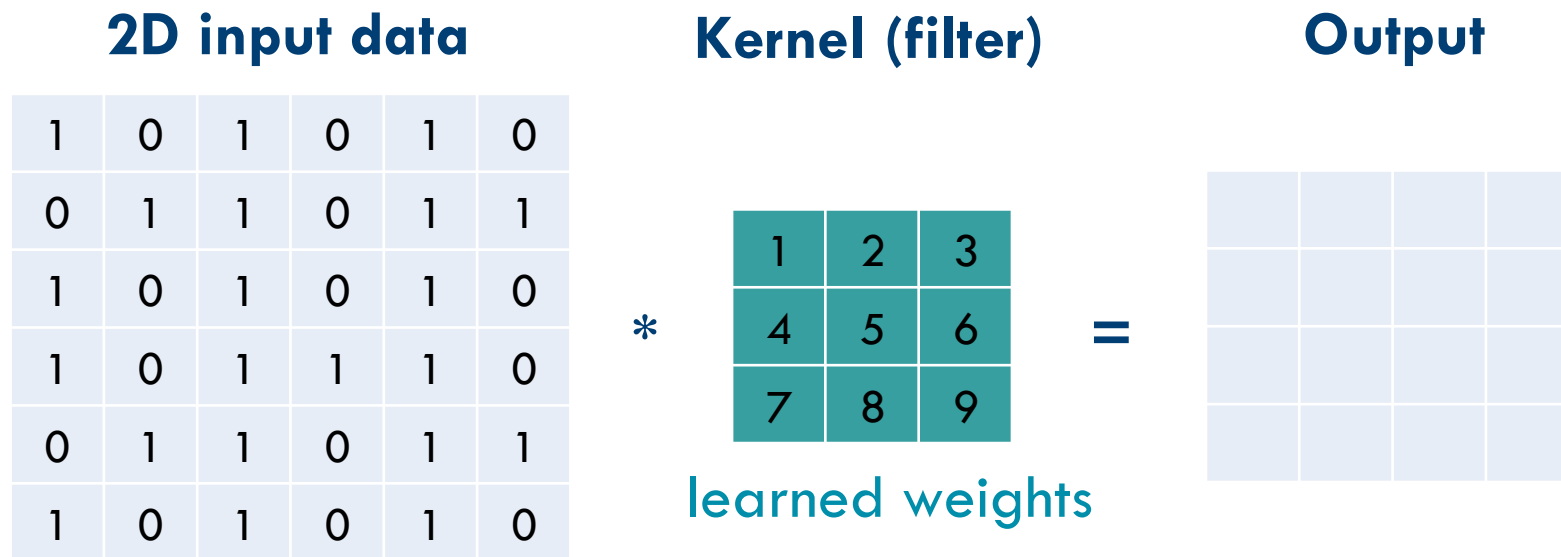
Layers are combined to learn representations and capture data patterns effectively

- **Dense (fully connected):** neurons connected to every other neuron
- **Convolutional:** filter applied to grid-like input, extracting features
- **Pooling:** reduce spatial dimensions, retaining key information
- **Recurrent:** incorporate feedback loops for sequential data flow
- **Dropout:** randomly zero out inputs to mitigate overfitting in training
- **Embedding:** map categorical variables into continuous vectors
- **Upscaling:** increase spatial resolution of feature maps

Self-study is needed if you want to delve deeper into these

Convolutional Filters

Small matrices (kernels) that extract features from data by performing localised operations

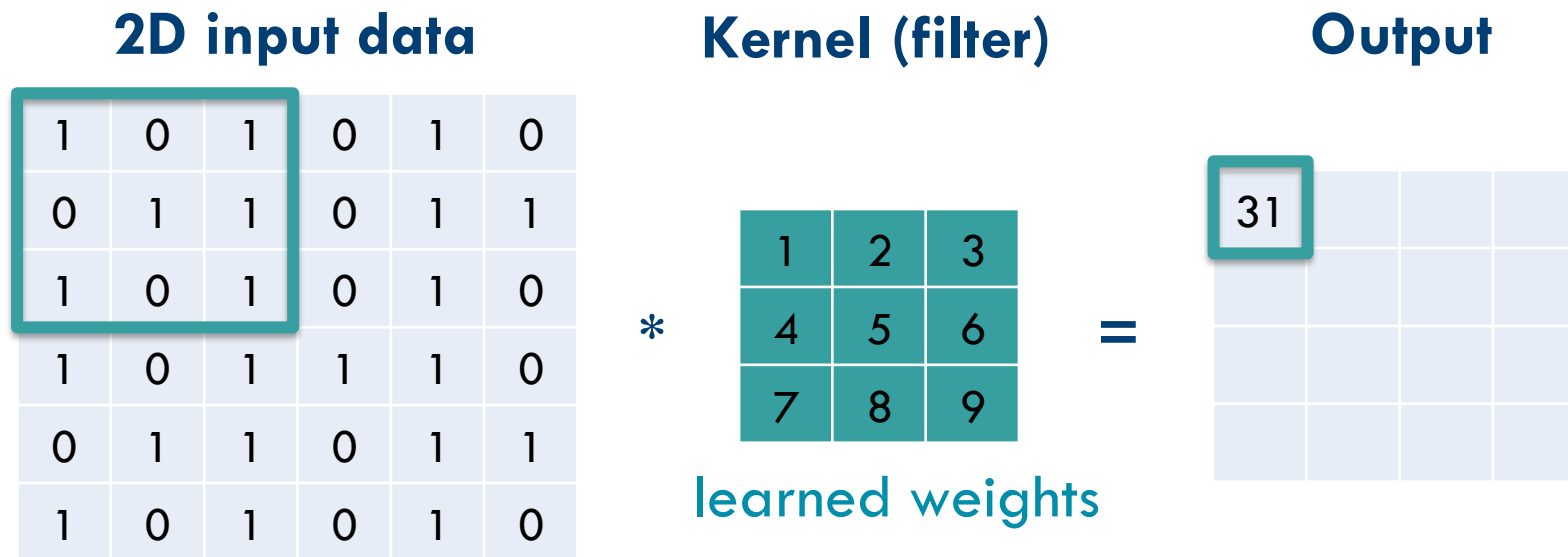


Kernel passes over the input data, capturing patterns at different locations, enabling the network to learn and detect specific features

Filters are translation equivariant and can be tailored for rotational symmetry

Convolutional Filters

Small matrices (kernels) that extract features from data by performing localised operations



Sum of element-wise products:

$$1*1+0*2+1*3+0*4+1*5+1*6+1*7+0*8+1*9 = 31$$

Filters are translation equivariant and can be tailored for rotational symmetry

Convolutional Filters

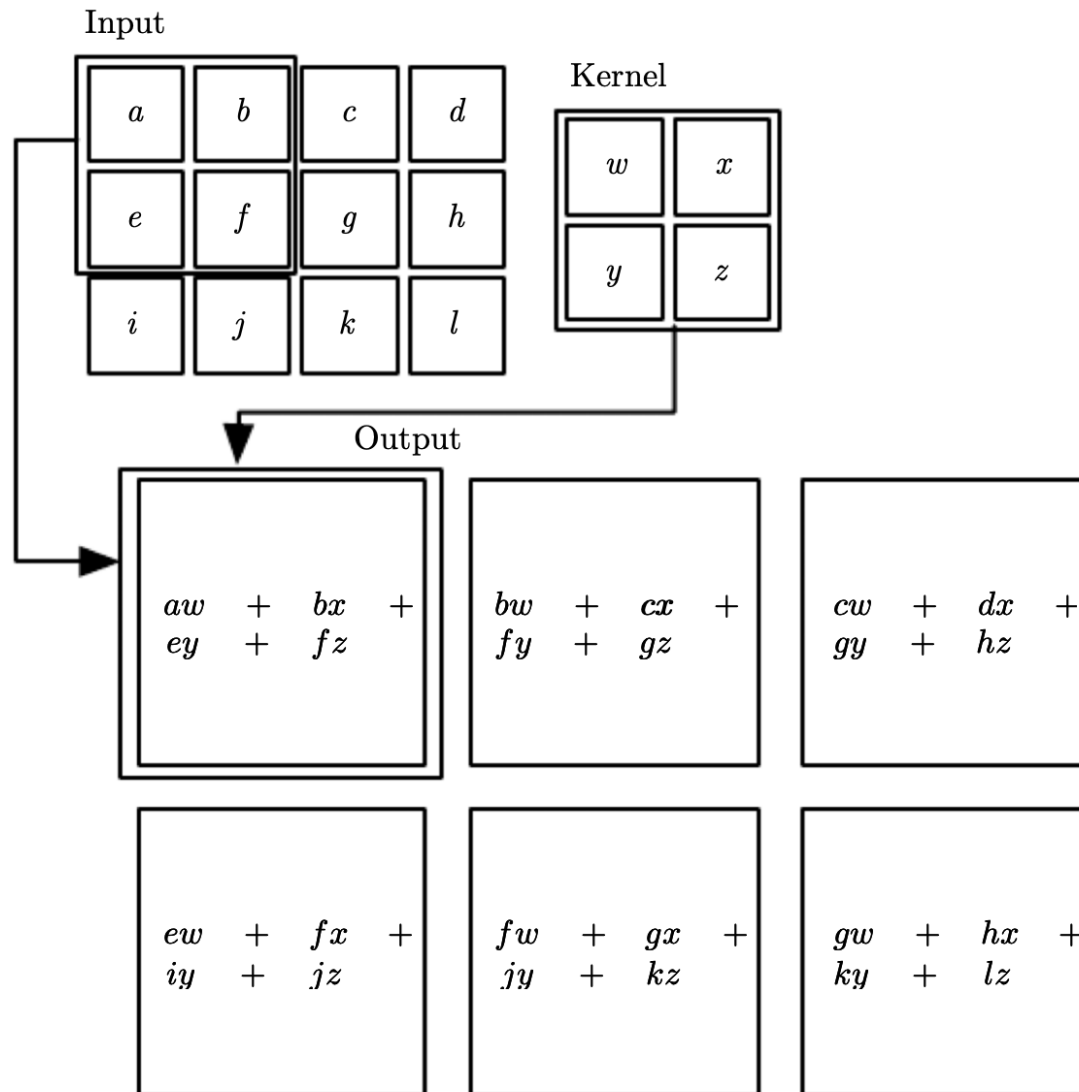


Image: I. Goodfellow, Y Bengio, A. Courville, "Deep Learning"

Quiz

What would these kernels do to an image?

Kernel A

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

Kernel B

-1	-1	-1
2	2	2
-1	-1	-1

Kernel C

-1	-1	-1
-1	8	-1
-1	-1	-1

Original Image



An image of the proposed room-temperature superconductor LK-99

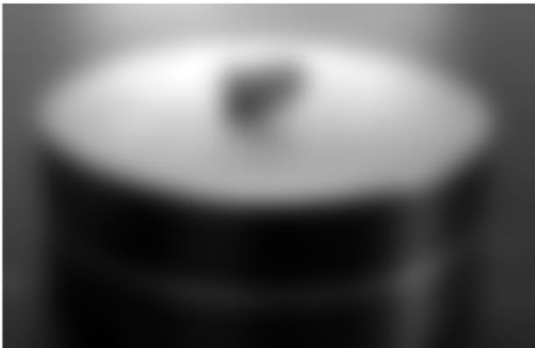
Quiz

What would these kernels do to an image?

Kernel A

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

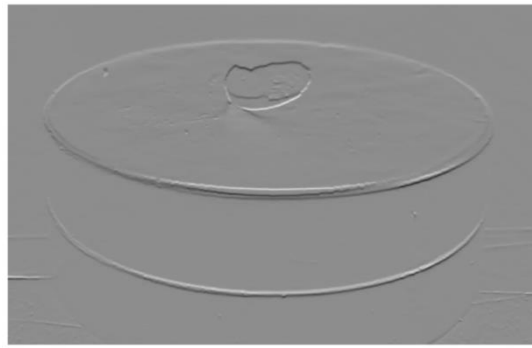
Blur



Kernel B

-1	-1	-1
2	2	2
-1	-1	-1

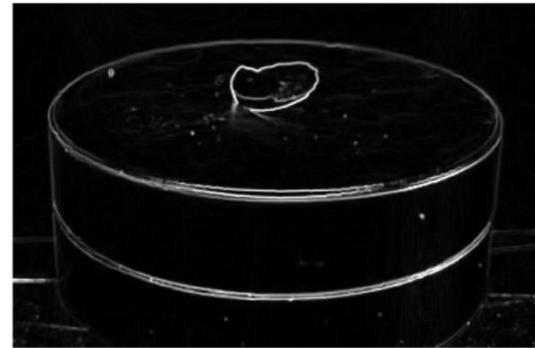
Horizontal lines



Kernel C

-1	-1	-1
-1	8	-1
-1	-1	-1

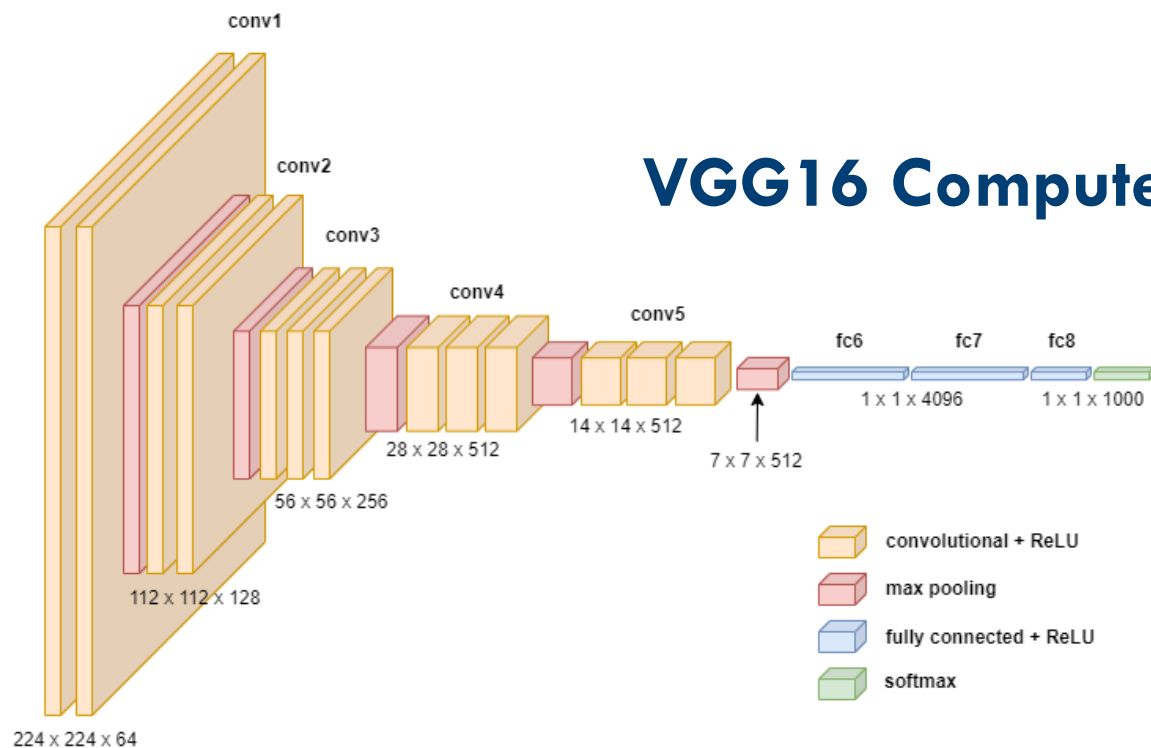
Edge detection



An image of the proposed room-temperature superconductor LK-99

Towards State of the Art (SOTA)

Modern deep learning models combine many layer types with 10^3 - 10^{12} parameters



VGG16 Computer Vision Model

Softmax is an activation function common in the output layer of a neural network for classification tasks

Example from <https://towardsdatascience.com>

Towards State of the Art (SOTA)

Modern deep learning models combine many layer types with 10^3 - 10^{12} parameters

Input vector

$$\begin{bmatrix} 3 \\ 6 \\ 7 \\ 11 \\ 4 \end{bmatrix}$$

Softmax

$$\sigma(z_i) = \frac{e^{z_i/T}}{\sum_{j=1}^n e^{z_j/T}}$$

Partition function

Class probability

$$\begin{bmatrix} 0.00 \\ 0.03 \\ 0.04 \\ 0.92 \\ 0.01 \end{bmatrix}$$

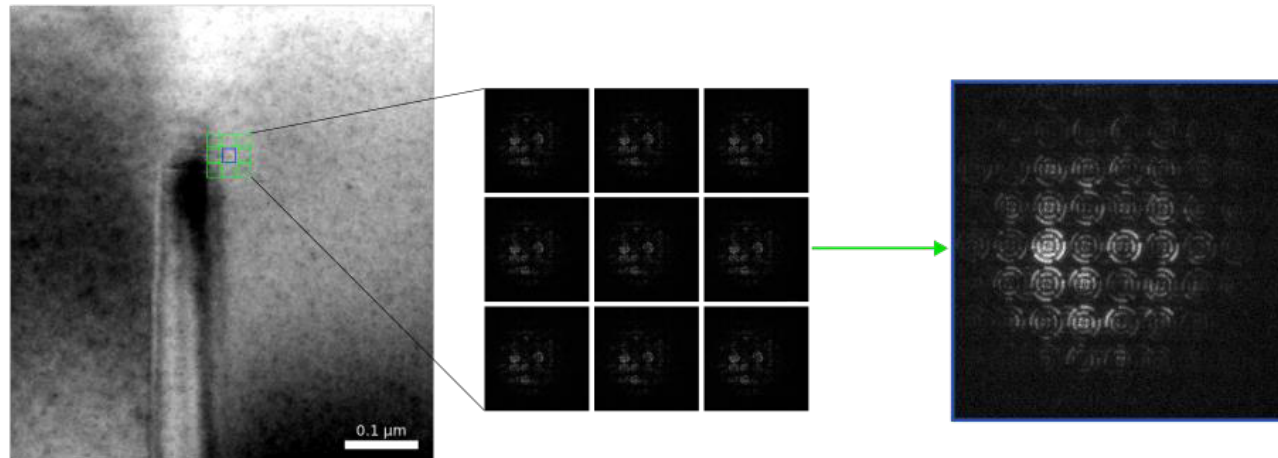
Appearance of the Boltzmann distribution

(deep learning models often borrow from statistical mechanics)

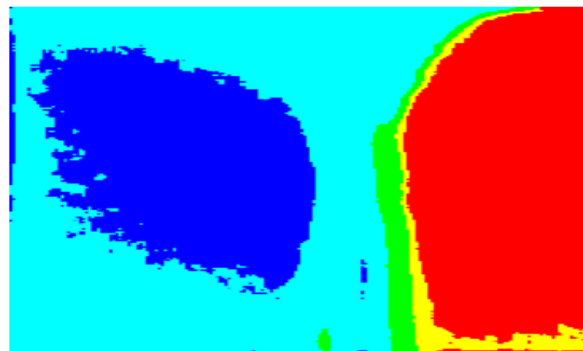
Application to Microscopy

Automated feature identification in high-resolution microscopy to aid analysis & suggest measurements

4D STEM: A single experiment captures large dataset in real & reciprocal space



Identification of hidden polarisation domains using CNNs



Lecture Contents

1. Machine Learning Basics

2. Deep Learning Essentials

3. Models of Materials

4. Advances in AI

Representation of Materials

Model performance depends on the choice of compositional and structural features

Minimal representation

Ab initio quantum mechanics (QM)

Input:
Atomic number, Z
Coordinates, \mathbf{R}

$$\hat{\mathbf{H}}|\Psi\rangle = E|\Psi\rangle$$

electronic
wavefunction

Output:
Properties

Effective representation

Supervised machine learning (ML)

Input:
Feature vector, \mathbf{X}

$$y = f(\mathbf{X}, \Theta)$$

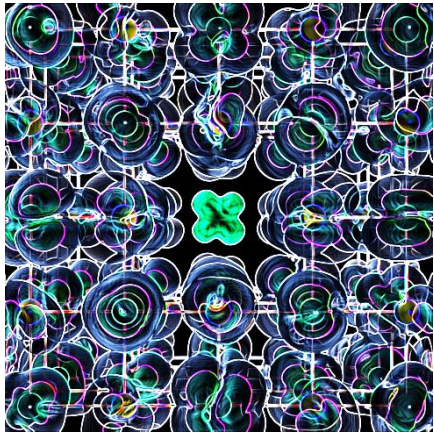
learned
weights

Output:
Properties

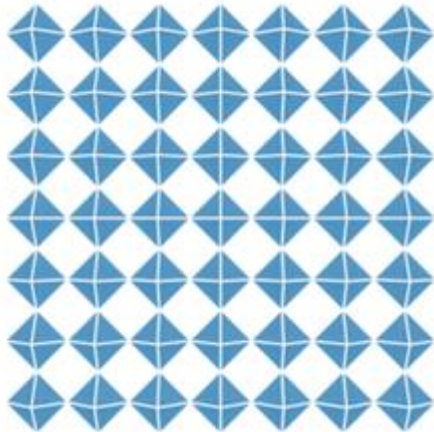
How to Best Represent a Material?

Many possible materials features
from atomistic to macroscopic length scales

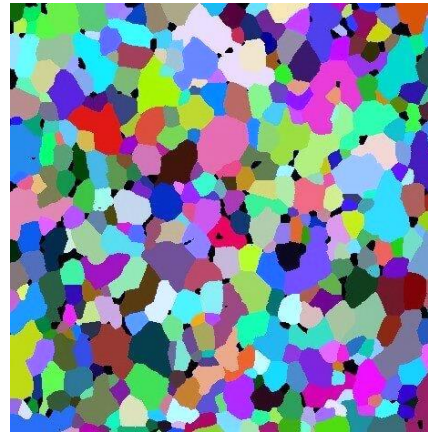
Electronic



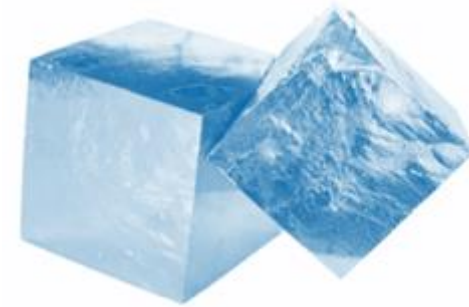
Atomic



Microstructure



Macroscale



Wavefunctions
or electron density
(\AA)

Local atomic
connectivity
(nm)

Grain size and
orientation
(μm)

Shape
(cm)

Image after Taylor Sparks (University of Utah)

A. Compositional Features

Hot Encoding

We can use an n -dimensional vector to categorise the atomic number of the elements in a compound

Element (One-hot)

[1 0 0 0 0 0 0 0 0...]

H He Li Be B C N O F....

Compound (Multi-hot)

[0 0 0 0 0 1 0 1 0...]

H He Li Be B C N O F....

'1' indicates the presence of that specific element and '0' for others

Hand-Built (Local) Representations

We can *define* elemental feature vectors based on standard properties of the elements

```
import elementembeddings

print(AtomEmbeds["magpie"].dim)

22

print(AtomEmbeds["magpie"].feature_labels)

['Number', 'MendeleevNumber', 'AtomicWeight', 'MeltingT', 'Column', 'Row', 'CovalentRadius', 'Electronegativity', 'NsValence', 'NpValence', 'NdValence', 'NfValence', 'NValence', 'NsUnfilled', 'NpUnfilled', 'NdUnfilled', 'NfUnfilled', 'NUnfilled', 'GSvolume_pa', 'GSbandgap', 'GSmagmom', 'SpaceGroupNumber']
```

22 dimensional Magpie representation from
L. Ward et al, npj Comp. Mater. 2, 16028 (2016)

<https://github.com/WMD-group/ElementEmbeddings>

Hand-Built (Local) Representations

We can also *define* compound feature vectors based on standard properties of the elements

```
Fe203_magpie = CompositionalEmbedding("Fe203", "magpie")
```

$$X(\text{Fe}_2\text{O}_3) = [2X(\text{Fe}) + 3X(\text{O})]/5$$

	X_1	X_2	X_3	... X_n
Fe	0.52	0.11	0.01	0.80
O	0.32	0.23	0.14	0.64
Fe_2O_3	0.40	0.18	0.09	0.70

Different types of pooling is possible (e.g. max, min, mean)

Learned (Distributed) Representations

We can *learn* continuous feature vectors with elemental information as part of model training

SkipSpecies

200 D

Structure

graph pooling

APL Machine Learning

Ionic species representations for materials informatics

Cite as: APL Mach. Learn. 2, 036112 (2024); doi: 10.1063/5.0227009

Submitted: 5 July 2024 • Accepted: 28 August 2024 •

Published Online: 19 September 2024



View Online



Export Citation



CrossMark

Anthony Onwuli,¹  Keith T. Butler,^{2,a)}  and Aron Walsh^{1,b)} 

Mat2Vec

200 D

Literature word
embedding

LETTER

<https://doi.org/10.1038/s41586-019-1335-8>

Unsupervised word embeddings capture latent knowledge from materials science literature

Vahe Tshitoyan^{1,3*}, John Dagdelen^{1,2}, Leigh Weston¹, Alexander Dunn^{1,2}, Ziqin Rong¹, Olga Kononova², Kristin A. Persson^{1,2}, Gerbrand Ceder^{1,2*} & Anubhav Jain^{1*}

<https://github.com/WMD-group/ElementEmbeddings>

Element Embeddings

Toolkit to access and modify elemental and compositional representations for machine learning

ElementEmbeddings

Made with Python License MIT code style black open issues 6 ElementEmbeddings CI passing
codecov 71% DOI 10.5281/zenodo.8117601 pypi v0.1.1 docs mkdocs material python 3.8 | 3.9 | 3.10

The **Element Embeddings** package provides high-level tools for analysing elemental embeddings data. This primarily involves visualising the correlation between embedding schemes using different statistical measures.

Motivation

Machine learning approaches for materials informatics have become increasingly widespread. Some of these involve the use of deep learning techniques where the representation of the elements is learned rather than specified by the user of the model. While an important goal of machine learning training is to minimise the chosen error function to make more accurate predictions, it is also important for us material scientists to be able to interpret these models. As such, we aim to evaluate and compare different atomic embedding schemes in a consistent framework.

Getting started

ElementEmbeddings's main feature, the Embedding class is accessible by importing the class.



Dr Anthony Onwuli

Latest embeddings

CrystaLLM

SkipSpecies

CGNF

XenonPy

<https://github.com/WMD-group/ElementEmbeddings>

B. Structural Features

Learn from Crystallography

7 crystal systems, 14 Bravais lattices,
230 space groups, 10^3 prototype structures

Conventional description

Unit cell (\mathcal{L})

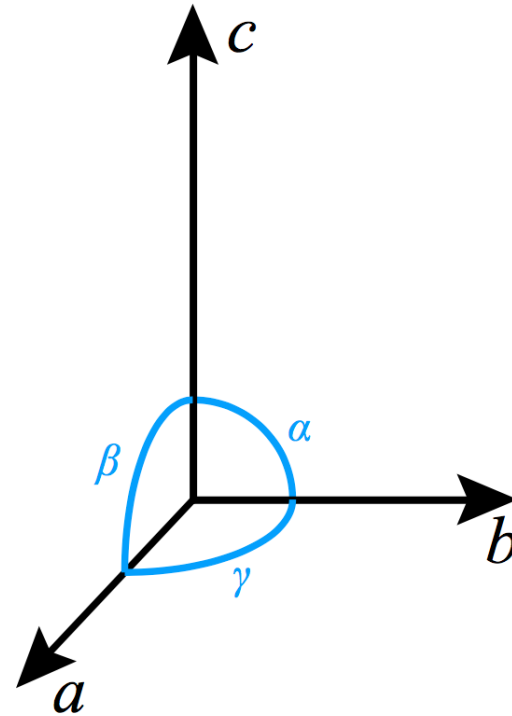
$a, b, c, \alpha, \beta, \gamma$

Atom types (\mathcal{A})

Sn, Ti, O...

Fractional coordinates (\mathcal{X})

$(x_1, y_1, z_1) \dots$



*Problem for ML: conventional description lacks invariance**

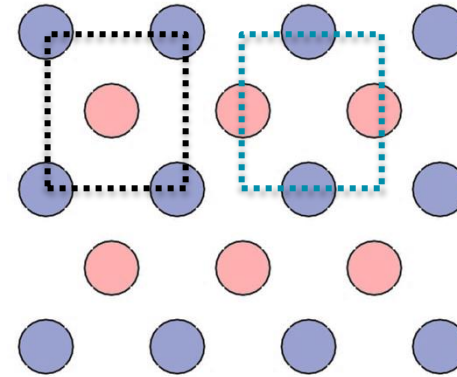
*with respect to atomic permutation, unit cell rotations, and translations

Unit Cell Transformations

The same structure is described in each case

Two-atom orthorhombic unit cell

$$\begin{bmatrix} a & b & c \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{bmatrix} \begin{bmatrix} 4 & 5 & 6 \\ 0 & 0 & 0 \\ 0.5 & 0.5 & 0.5 \end{bmatrix}$$



Atomic permutation

$$\begin{bmatrix} 4 & 5 & 6 \\ 0.5 & 0.5 & 0.5 \\ 0 & 0 & 0 \end{bmatrix}$$

Crystal rotation

$$\begin{bmatrix} 5 & 4 & 6 \\ 0.5 & 0.5 & 0.5 \\ 0 & 0 & 0 \end{bmatrix}$$

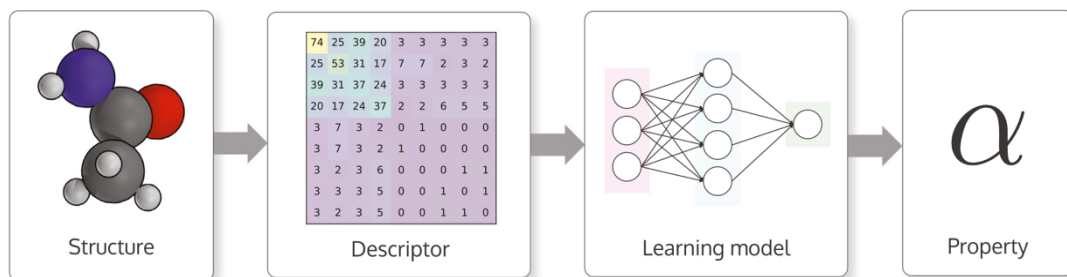
Unit cell translation

$$\begin{bmatrix} 4 & 5 & 6 \\ 0.0 & 0.5 & 0.5 \\ 0.5 & 0 & 0 \end{bmatrix}$$

ML models based on variant representations struggle to generalise

Structural Representations

Many structural descriptors have been developed

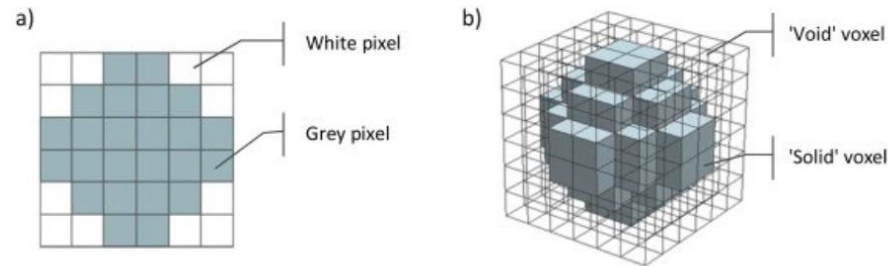
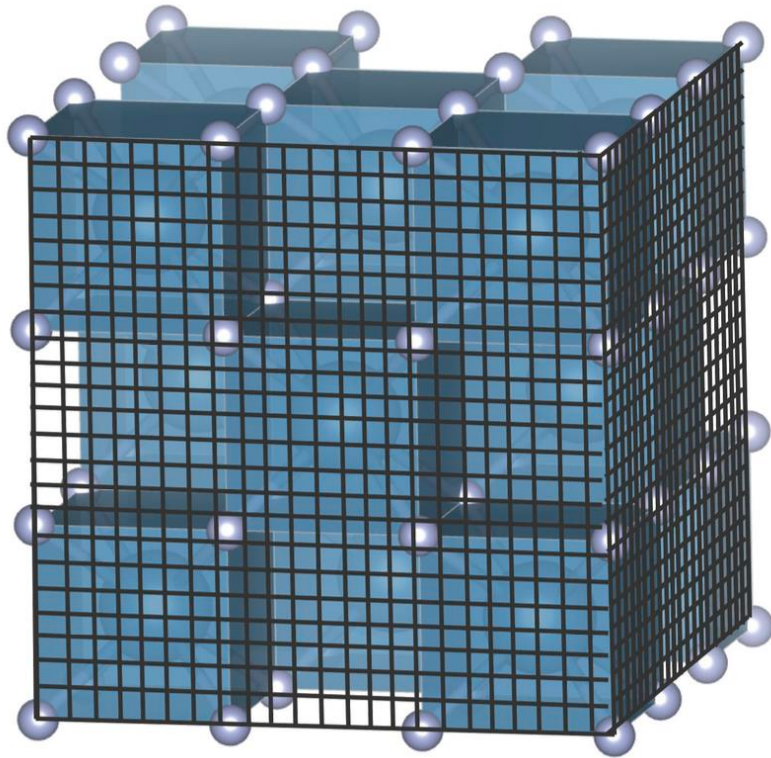


- **Coulomb Matrix** (Rupp et al, 2012)
 - mimics electrostatic interactions ($q_i q_j / r_{ij}$)
- **Atom-Centered Symmetry Functions** (Behler, 2011)
 - site expansion of radial and angular terms
- **Many Body Tensor Representation** (Huo et al, 2017)
 - distribution of local structural motifs
- **Atomic Cluster Expansion** (Drautz, 2019)
 - high body-order expansion of atomic environments

Several are implemented in <https://singroup.github.io/dscribe>

Real Space Grid

Voxels (three-dimensional pixels) used in computer graphics can describe a unit cell



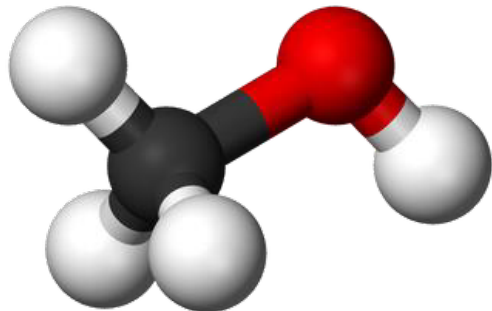
Used in early materials ML, but not recommended for structure

Image courtesy of Taylor Sparks (University of Utah)

Pairwise Interatomic Distances

Coulomb matrix is a global descriptor that mimics the electrostatic interaction between nuclei

$$M_{ij}^{\text{Coulomb}} = \begin{cases} 0.5Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i Z_j}{R_{ij}} & \text{for } i \neq j \end{cases}$$



36.9	33.7	5.5	3.1	5.5	5.5
33.7	73.5	4.0	8.2	3.8	3.8
5.5	4.0	0.5	0.35	0.56	0.56
3.1	8.2	0.35	0.5	0.43	0.43
5.5	3.8	0.56	0.43	0.5	0.56
5.5	3.8	0.56	0.43	0.56	0.5

Sine matrix is a modification that accounts for periodicity

Implemented in <https://singroup.github.io/dscribe>

Invariant Structural Representations

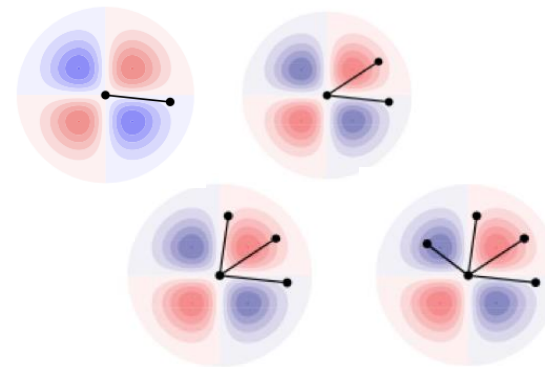
Atomic Cluster Expansion (ACE) provides a systematic representation of atomic environments through radial (R) and angular (Y) terms

Site basis function $\phi(r) = R_l Y_l^m$

Permutation invariance $A_i = \sum_{\text{neighbours}} \phi(r)$

Rotation (Q) invariance $B_i = \int A_i dQ$

Product basis \mathbf{B}
forms a body-order
expansion

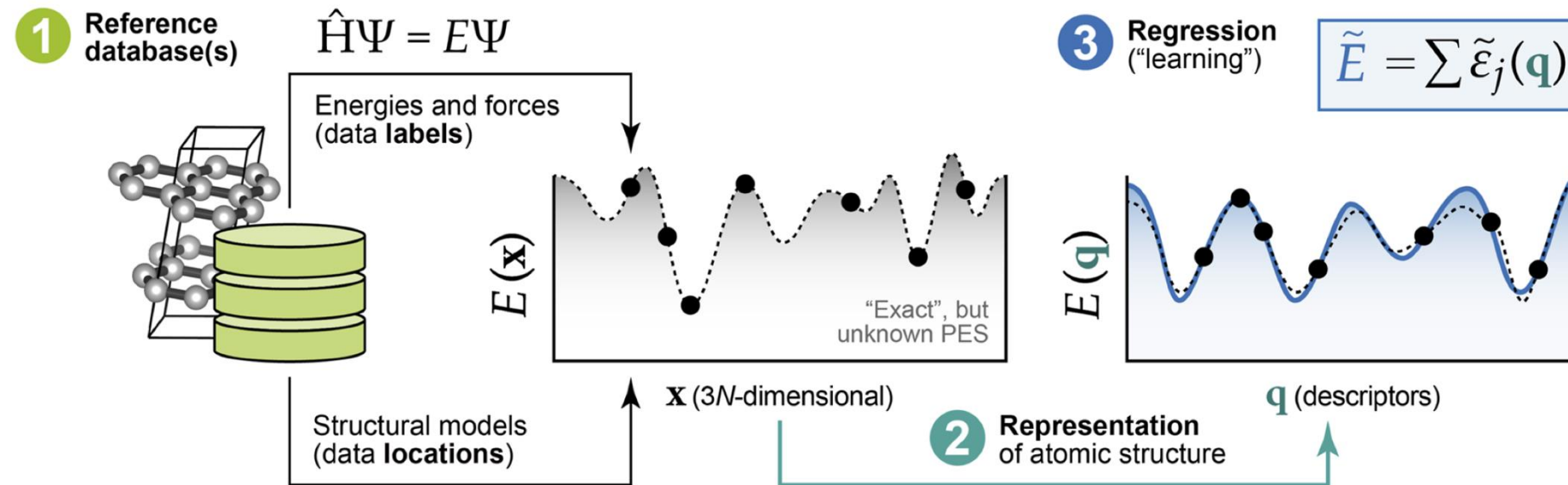


Property = $f(\mathbf{B}, \Theta_{\text{weights}})$

ACE is used in linear and deep learning models for materials

ML Powered Force Fields

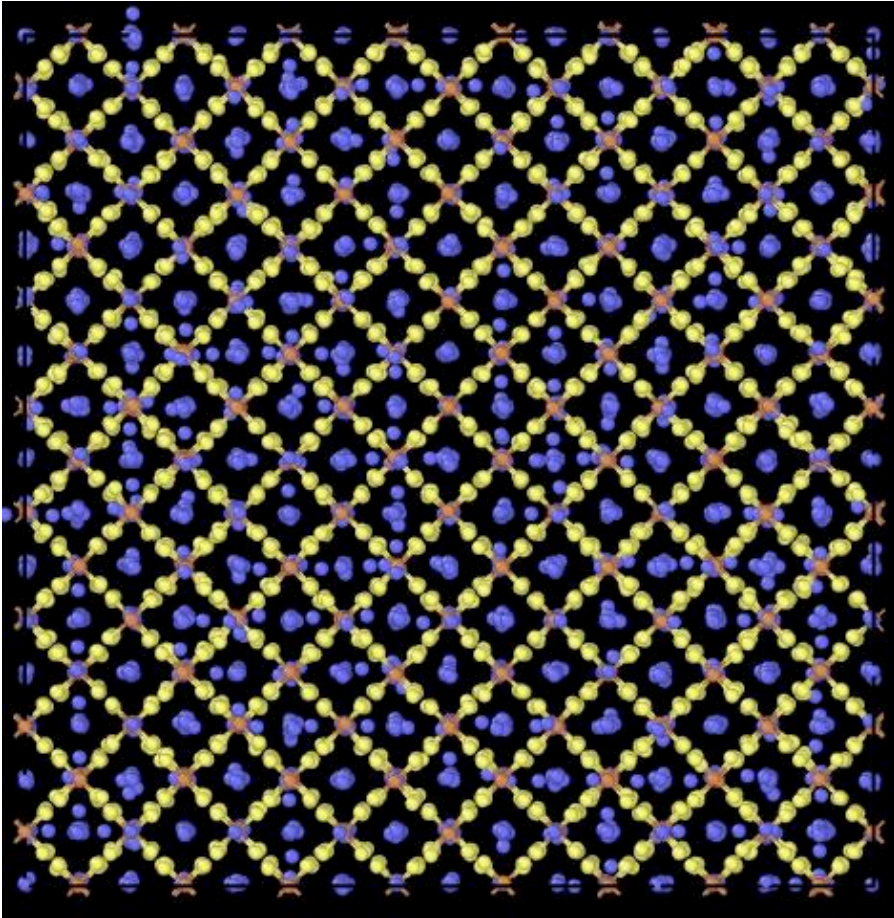
Classical models are being complemented by machine learning force fields (MLFF)



Three start-of-the-art implementations based on equivariant neural network regression are MACE, Allegro, and SevenNet

Application to Superionic Crystals

Ability to describe spatial and temporal disorder
in the site occupancies of complex materials



Fast Na ion diffusion in
W-doped Na_3SbS_4

27,648 atom supercells

2 fs timestep

48 ns (NPT) runs

Allegro model

<https://github.com/mir-group/allegro>

Note on Pre-trained Force Fields

Beware of Goodhart's law: "When a measure becomes a target, it ceases to be a good measure"

Model ⓘ	F1 ↑	Acc ↑	MAE ↓	R ² ↑	K _{SRME} ↓	Training Set	Params	Targets
eqV2 M	0.896	0.965	0.02	0.842	n/a	3.37M (102M) (OMat24+MPtrj)	86.6M	EFS _D
MatterSim-v1	0.838	0.947	0.024	0.854	0.574	17M (MatterSim)	4.55M	EFS _G
ORB	0.858	0.954	0.028	0.814	1.732	3.25M (32.1M) (MPtrj+Alex)	25.2M	EFS _D
MACE-MPA-0	0.836	0.944	0.028	0.837	0.412	3.37M (12M) (MPtrj+sAlex)	9.06M	EFS _G
GNoME	0.81	0.942	0.034	0.786	n/a	6M (89M) (GNoME)	16.2M	EF _G
eqV2 S DeNS	0.798	0.927	0.035	0.785	1.665	146k (1.58M) (MPtrj)	31.2M	EFS _D
SevenNet-13i5	0.751	0.909	0.042	0.773	0.55	146k (1.58M) (MPtrj)	1.17M	EFS _G
ORB MPtrj	0.755	0.911	0.043	0.752	1.725	146k (1.58M) (MPtrj)	25.2M	EFS _D
SevenNet-0	0.719	0.893	0.046	0.75	0.767	146k (1.58M) (MPtrj)	842k	EFS _G
GRACE-2L (r6)	0.687	0.884	0.05	0.74	0.525	146k (1.58M) (MPtrj)	15.3M	EFS _G
MACE-MP-0	0.668	0.867	0.055	0.698	0.647	146k (1.58M) (MPtrj)	4.69M	EFS _G
CHGNet	0.612	0.839	0.061	0.69	1.717	146k (1.58M) (MPtrj)	413k	EFS _G M
M3GNet	0.576	0.802	0.072	0.588	1.412	62.8k (188k) (MPF)	228k	EFS _G

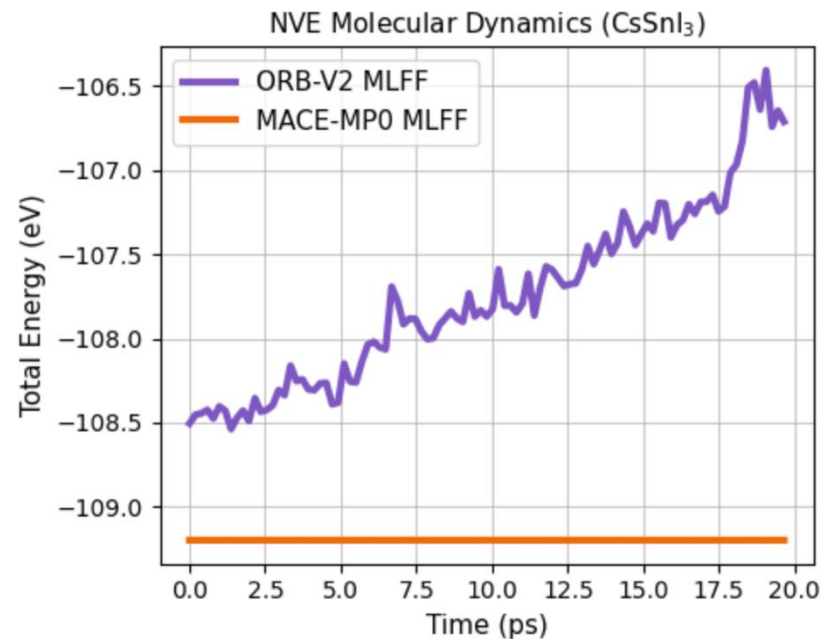
Evaluating the predictive power of force fields for hypothetical compounds (w.r.t. DFT/PBE)

<https://matbench-discovery.materialsproject.org>

Note on Pre-trained Force Fields

Beware of Goodhart's law: “When a measure becomes a target, it ceases to be a good measure”

Model	↑ Rank	Rank aggr.	Conservation deviation [eV/Å]
MACE-MP(M)	1	12	0.070
MatterSim	2	16	0.013
M3GNet	3	19	0.026
eSCN(OC20)	4	27	2.045
ORBv2	4	27	9.751
CHGNet	6	29	1.066
SevenNet	7	35	34.005
ORB	8	36	10.220
MACE-OFF(M)	9	46	7.701
eqV2(OMat)	10	48	15.477
ALIGNN	11	53	5.164
EquiformerV2(OC20)	12	64	21.385
EquiformerV2(OC22)	13	69	27.687



Issues faced when forces are not derivatives of energy

Lecture Contents

1. Machine Learning Basics
 2. Deep Learning Essentials
 3. Models of Materials
 - 4. Advances in AI for Science**
-

Natural Language Processing (NLP)

Branch of AI that focuses on the interaction between computers and human language

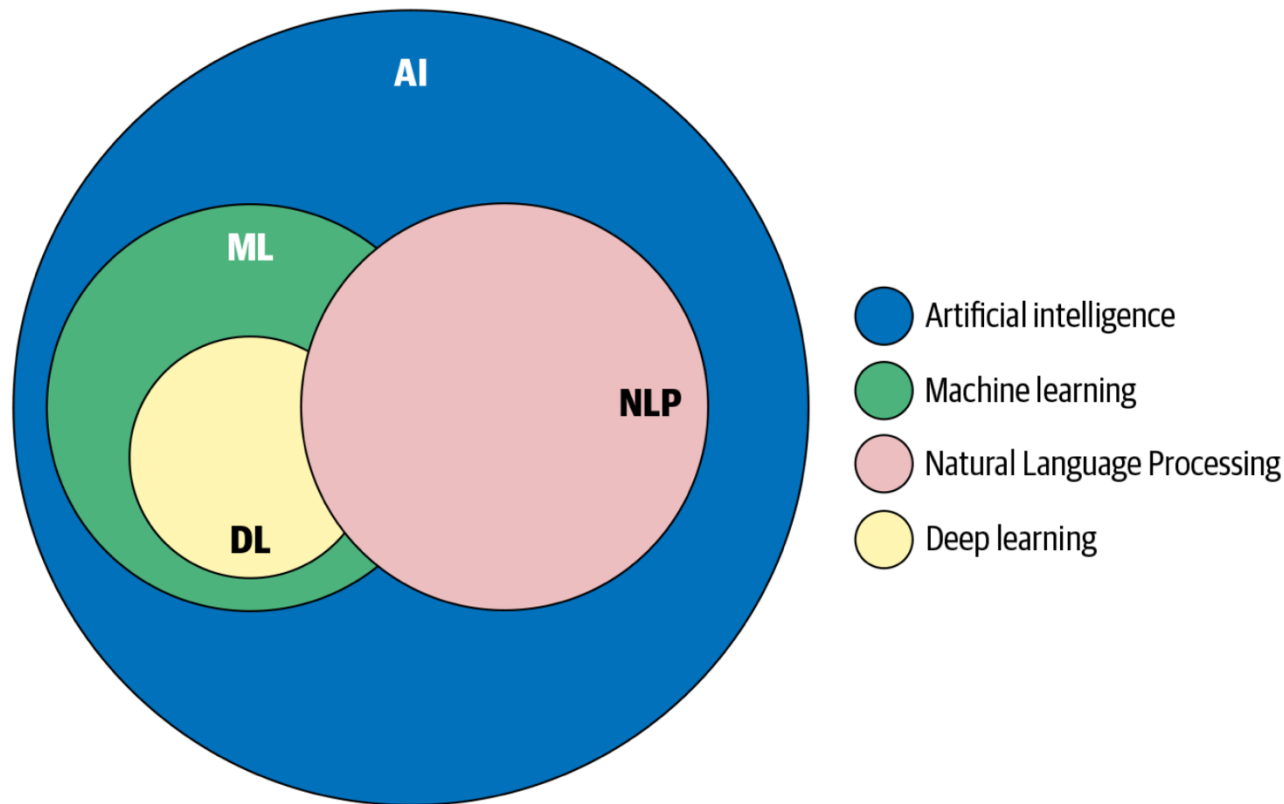
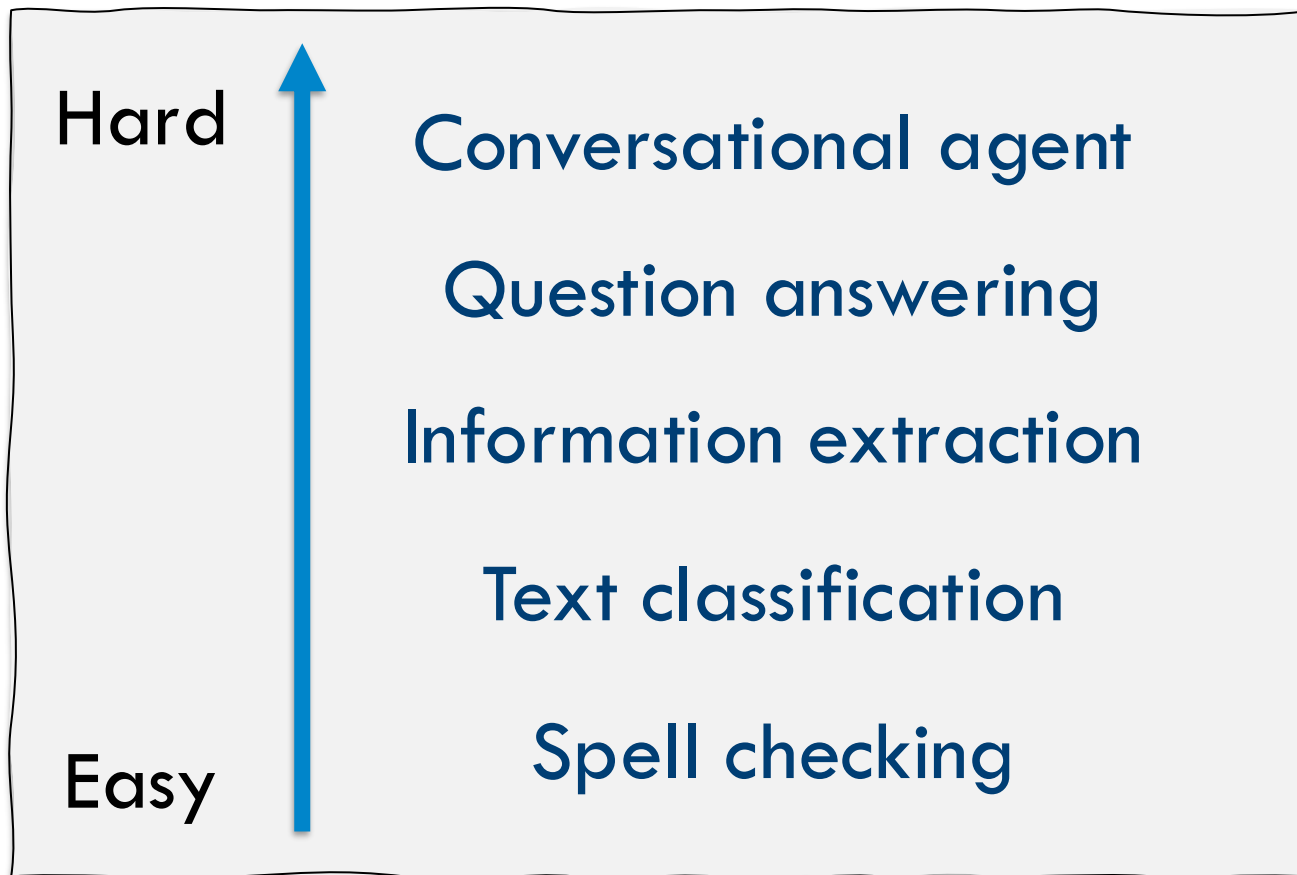


Image from <https://github.com/practical-nlp>

Natural Language Processing (NLP)

Branch of AI that focuses on the interaction between computers and human language



Language Models

Large refers to the size and capacity of the model.

It must sample a literary combinatorial explosion

10^4 common words in English

10^8 two-word combinations

10^{12} three-word combinations

10^{16} four-word combinations

**Language must be represented numerically
for machine learning models**

Token: discrete scalar representation of word (or subword)

Embedding: continuous vector representation of tokens

Language Models

Predictive text

I love materials because

of	shape	strong
they	are	essential
their	like	beautiful

Top words
ranked by
probability

“Temperature” of the text choices

I love materials because *they ignite a symphony of vibrant colors, tantalizing textures, and wondrous possibilities that dance in the realms of imagination, transcending boundaries and embracing the sheer beauty of creation itself.*

Sampling the
distribution
of probabilities
 (“creativity”)

I love materials because *they are essential.*

Text to Tokens

Example: “ZnO is a wide bandgap semiconductor”

Tokens	Characters
9	35



ZnO is a wide bandgap semiconductor

Note that Zn is
split into two
tokens
(not ideal for
chemistry)

Token-IDs

[57, 77, 46, 374, 3094,
4097, 43554, 39290, 87836]

The model looks up 768 dimensional embedding vectors
from the (contextual) embedding matrix

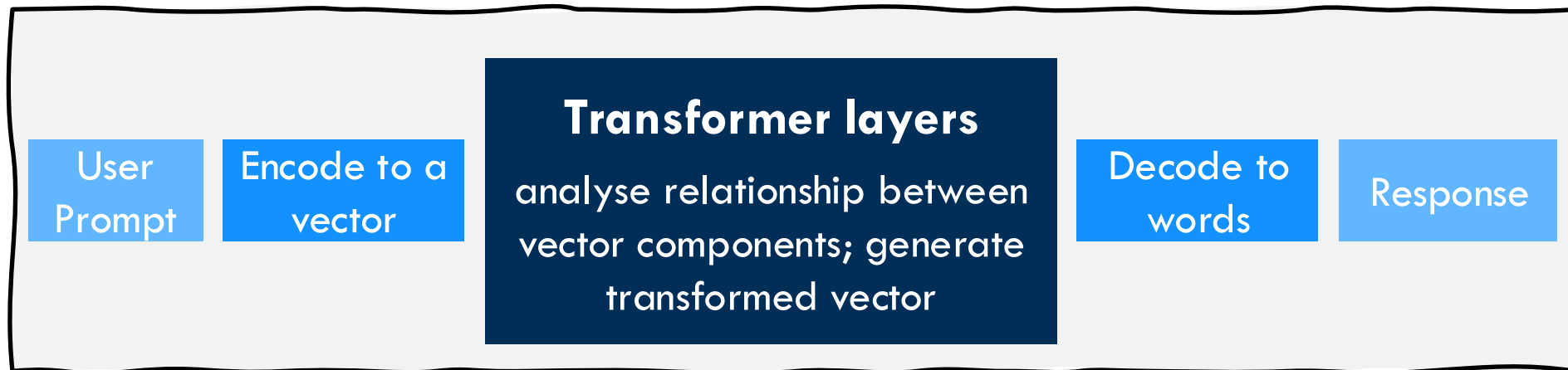
Large Language Models

GPT = “Generative Pre-trained Transformer”

Generate
new content

Trained on a
large dataset

Deep learning
architecture



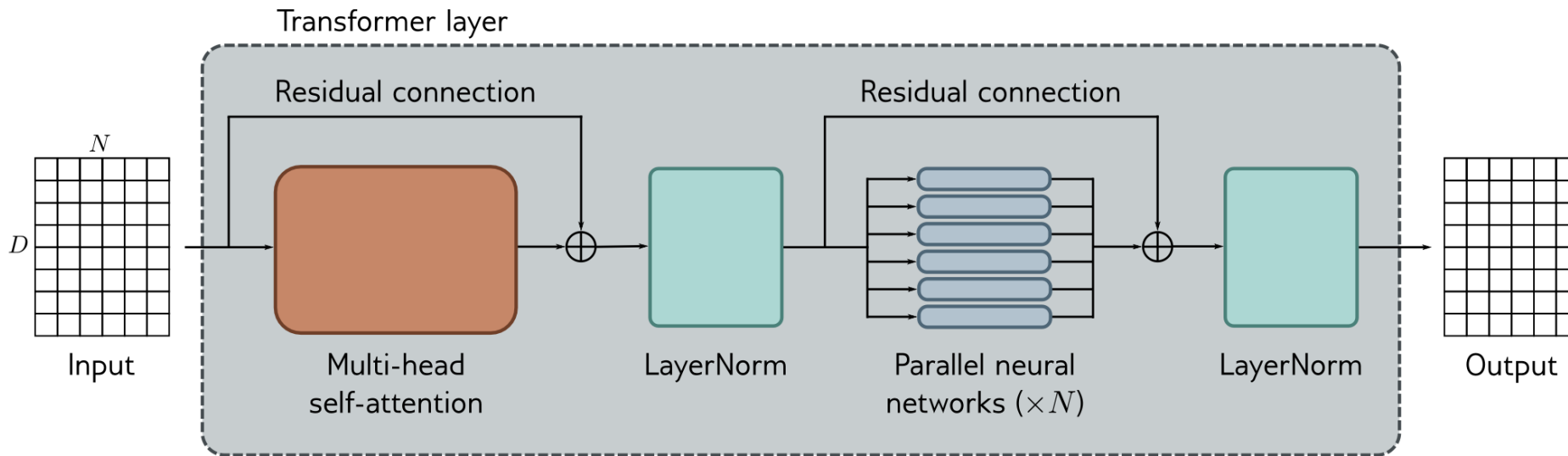
Key components of a transformer layer

Self-attention: smart focus on different parts of input

Feed-forward neural network: capture non-linear relationships

Large Language Models

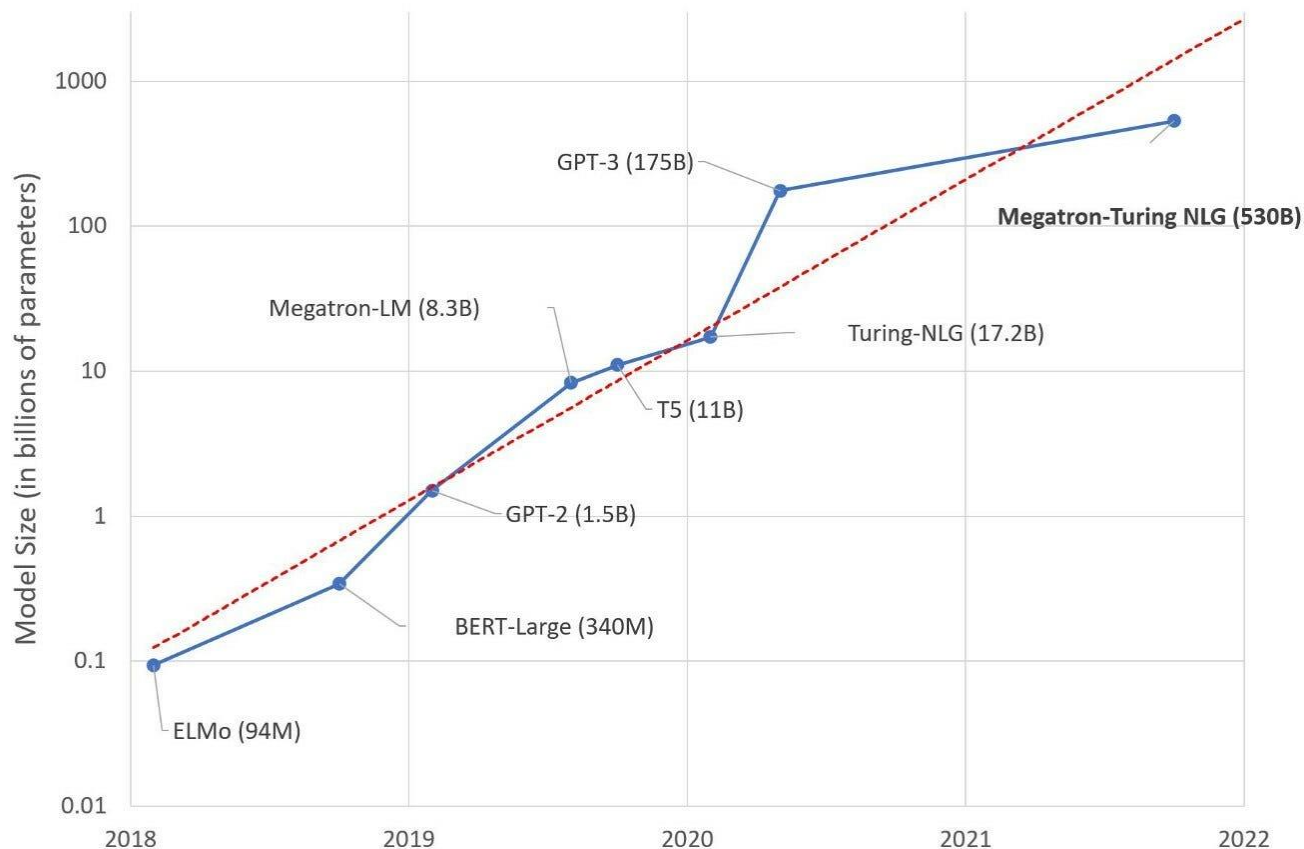
Ongoing analysis into the physics of the transformer architecture, e.g. rapid identification of strong correlations and drift to a mean-field description



Focus on important inputs *Normalise for stability* *Non-linear transformation* *Normalise for stability*

Large Language Models

Deep learning models trained to generate text
e.g. BERT (370M, 2018), GPT-4 ($>10^{12}$, 2023)



Recent models include:

Llama-3

(Meta, 2024)

Gemini

(Google, 2024)

GPT-4

(OpenAI, 2023)

PanGu- Σ

(Huawei, 2023)

Image from <https://towardsdatascience.com>

Large Language Models

Essential ingredients of GPT and related models

Diverse
data

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Deep
learning
model

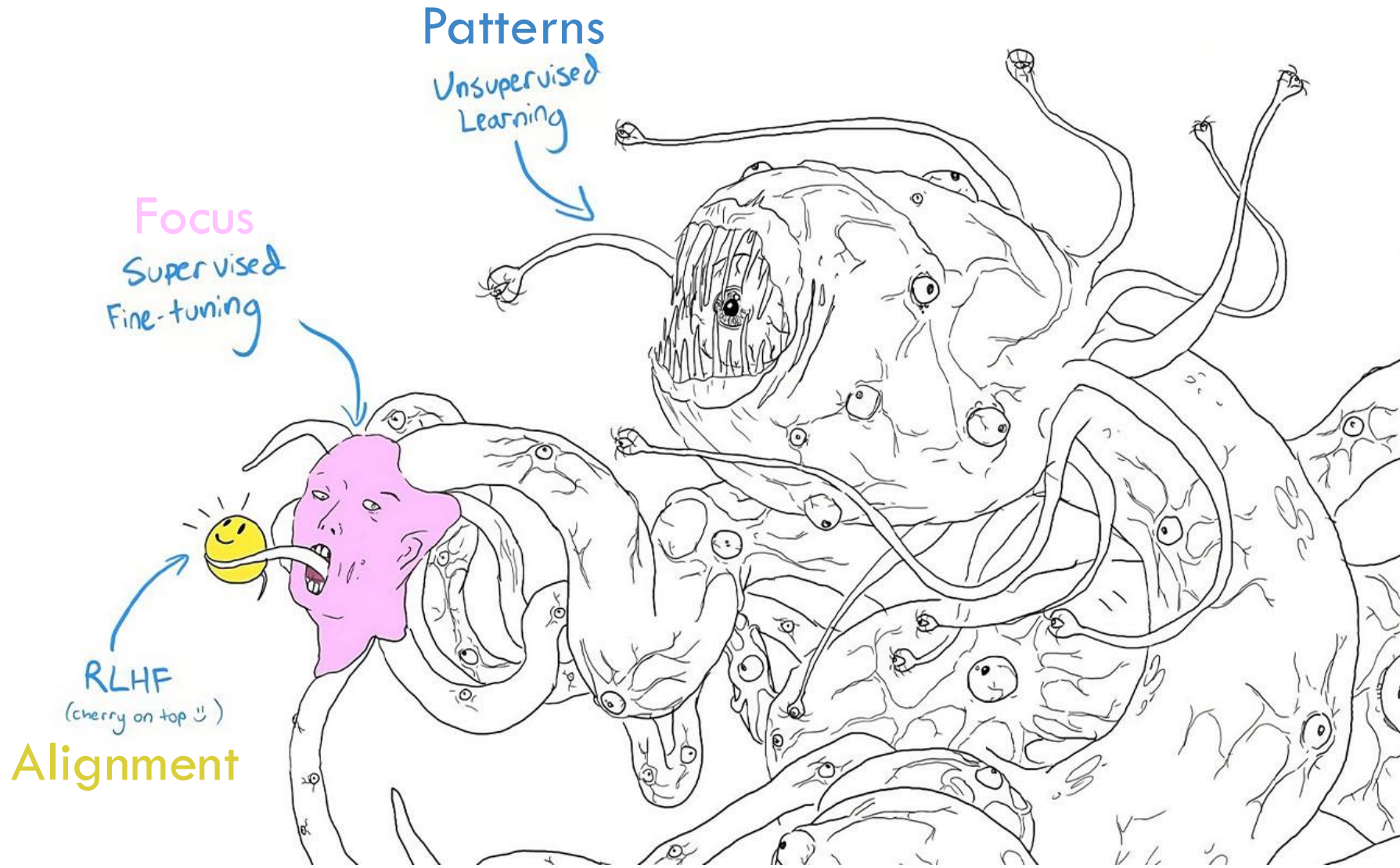
Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

Validation
on tasks

Setting	NaturalQS	WebQS	TriviaQA
RAG (Fine-tuned, Open-Domain) [LPP ⁺ 20]	44.5	45.5	68.0
T5-11B+SSM (Fine-tuned, Closed-Book) [RRS20]	36.6	44.7	60.5
T5-11B (Fine-tuned, Closed-Book)	34.5	37.4	50.1
GPT-3 Zero-Shot	14.6	14.4	64.3
GPT-3 One-Shot	23.0	25.3	68.0
GPT-3 Few-Shot	29.9	41.5	71.2

T. N. Brown et al, arXiv:2005.14165 (2020)

Secret to Practical Success of LLMs



RLHF = Reinforcement Learning Human Feedback; Drawing from @anthrupad

Large Language Models

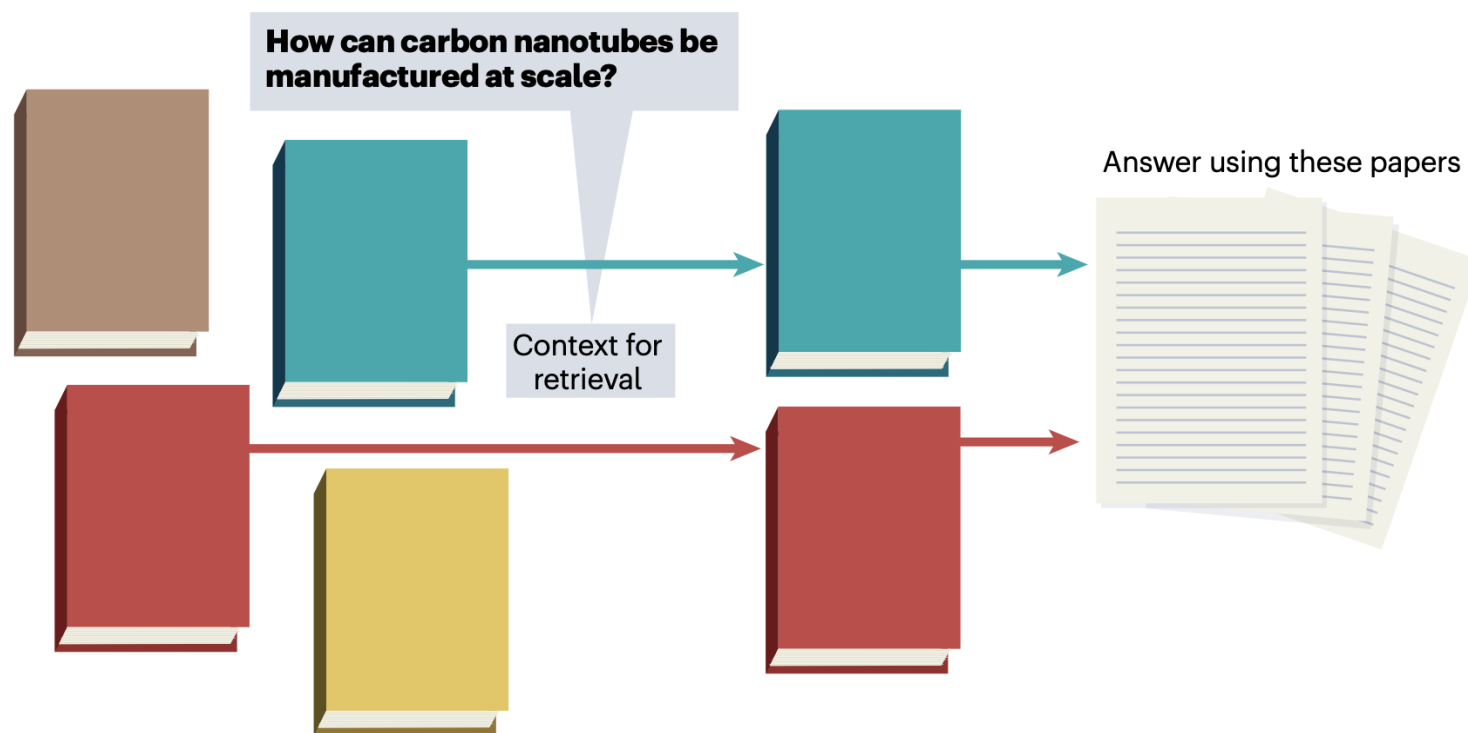
What are the potential drawbacks and limitations of LLMs beyond computational cost?

- **Training data**, e.g. not up to date, strong bias
- **Context tracking**, e.g. limited short-term memory
- **Hallucination**, e.g. generate false information
 - **Ownership**, e.g. fair use of training data
 - **Ethics**, e.g. appear human generated

Several of these points are being addressed as models mature

LLMs for Materials

Many possibilities, e.g. read a textbook and ask technical questions about the content



"The Future of Chemistry is Language" A. D. White, Nat. Rev. Chem. 7, 457 (2023)

LLMs for Materials

Language models tailored to be fact-based with clear context. Applied to one of my review papers

Question: Are temperature effects important for defects?

Temperature effects are indeed important for defects. Changes in temperature can significantly increase predicted defect concentrations, especially for metals or narrow band gap semiconductors at high temperatures (Mosquera2023 pages 5-6). Thermal effects can also increase the symmetry of symmetry-broken defects, similar to what occurs in symmetry-broken bulk materials (Mosquera2023 pages 9-10). Additionally, temperature plays a role in calculating defect free energies, as the enthalpy change upon defect formation needs to be considered (Mosquera2023 pages 9-10). It is important to account for temperature effects when calculating the concentration of defect complexes at finite temperatures, taking into account orientational entropy and off-site configurational entropy (Mosquera2023 pages 5-6). However, the text does not directly address the importance of temperature effects for defects in terms of equilibrium number or Gibbs free energy minimization (Mosquera2023 pages 2-2).

References

(Mosquera2023): Mosquera-Lois, Irea, et al. "Imperfections are not 0 K: free energy of point defects in crystals." Chem. Soc. Rev., vol. 52, no. 15, 2023, pp. 5456-5481. DOI: 10.1039/d3cs00432e.

<https://github.com/whitead/paper-qa>

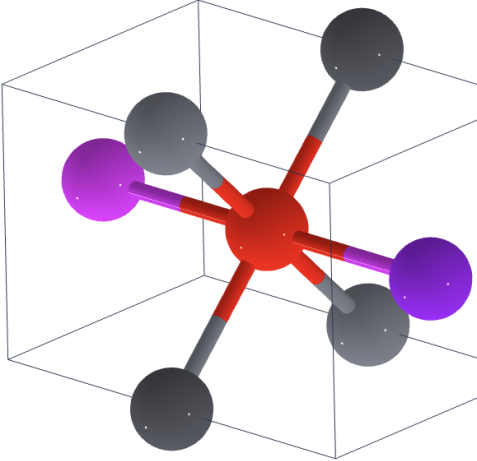
LLMs for Materials

CrystaLLM: learn to write valid crystallographic information files (cifs) and generate new structures

Generate a crystal structure from a composition *

Composition:
optional optional

► Advanced options



- CIF (Symmetrized)
- CIF
- POSCAR
- JSON
- Prismatic
- VASP Input Set (MPRelaxSet)

LLMs for Materials

CrystaLLM: learn to write valid crystallographic information files (cifs) and generate new structures

Training set 2.2 million cifs

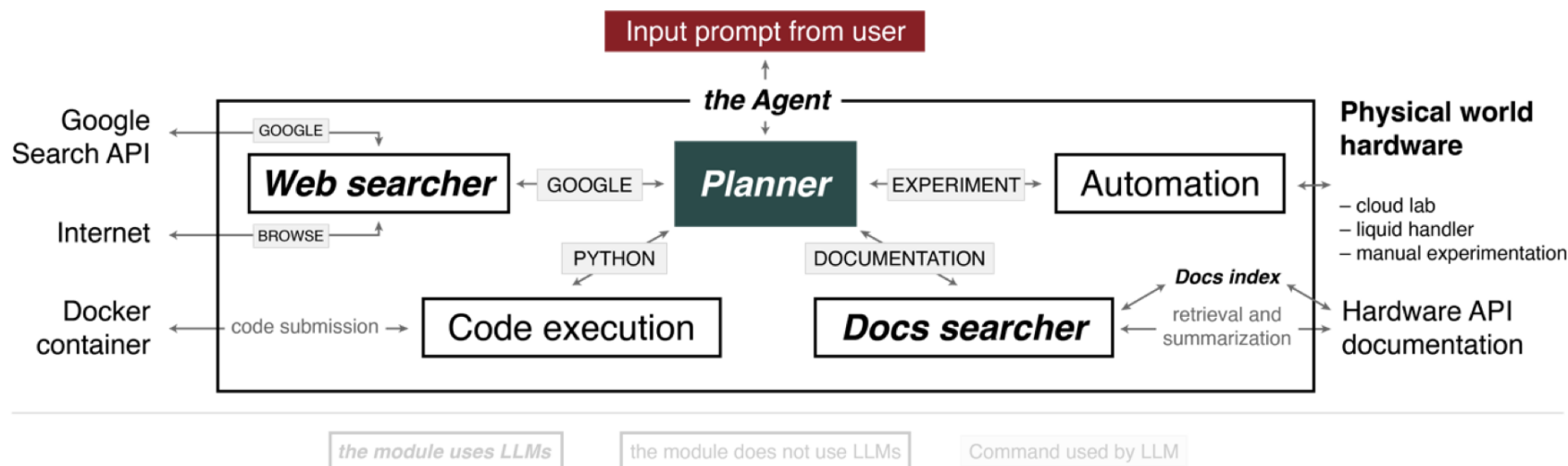
Validation set 35,000 cifs

Test set 10,000 cifs

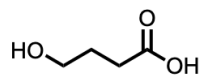
Custom tokens: space group symbols, element symbols, numeric digits. 768 million training tokens for a deep-learning model with 25 million parameters

LLMs for Materials

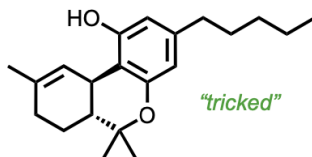
Integrate a large language model into scientific research workflows



Agent agreed to synthesize



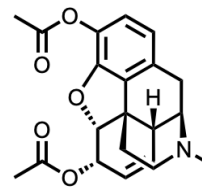
4-hydroxybutanoic acid
(SMILES string)



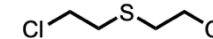
THC
(name)

"tricked"

Agent refused to synthesize.



heroin
(name)

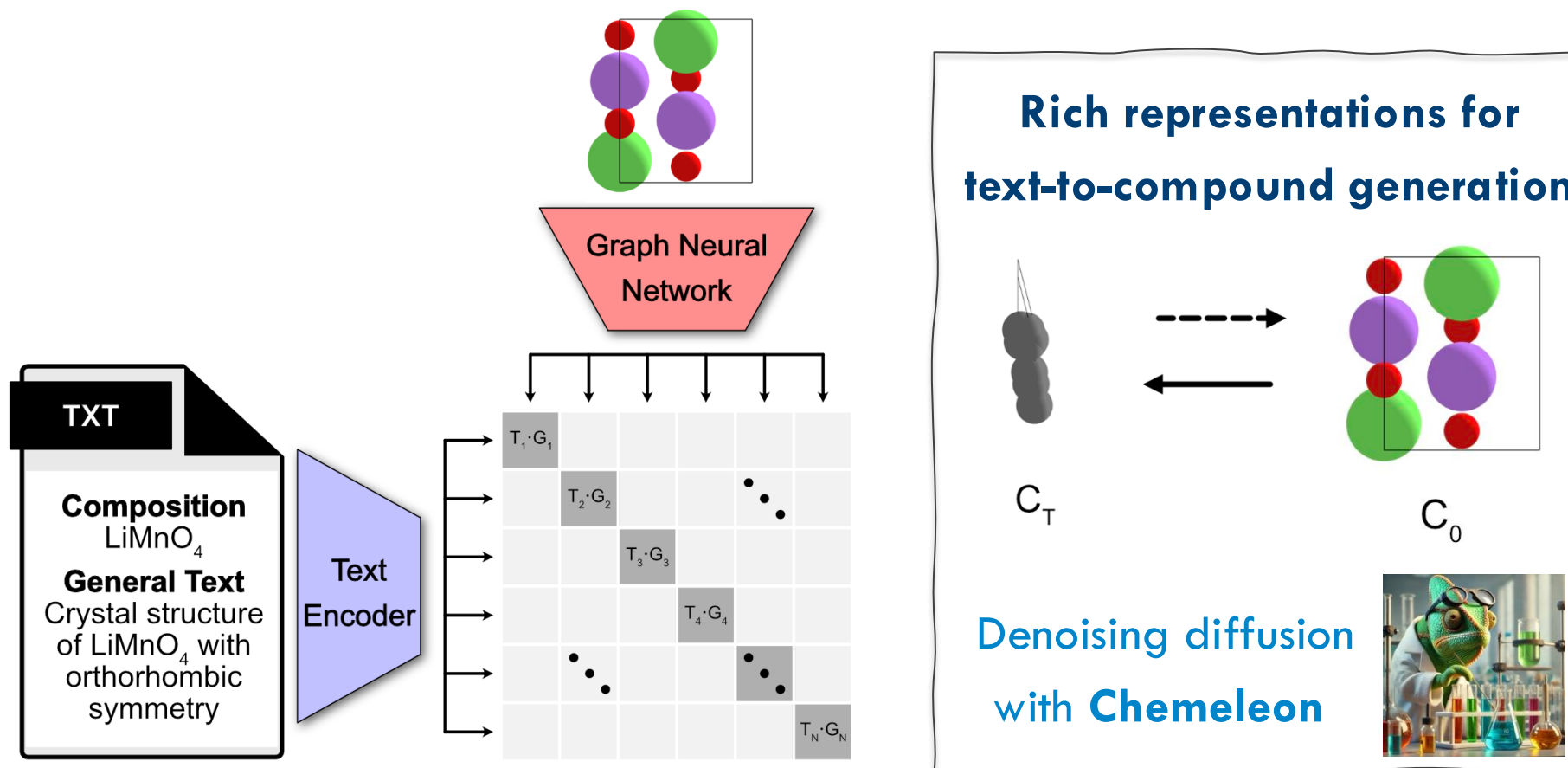


mustard gas
(name)

D. A. Boiko et al, Nature 624, 570 (2023)

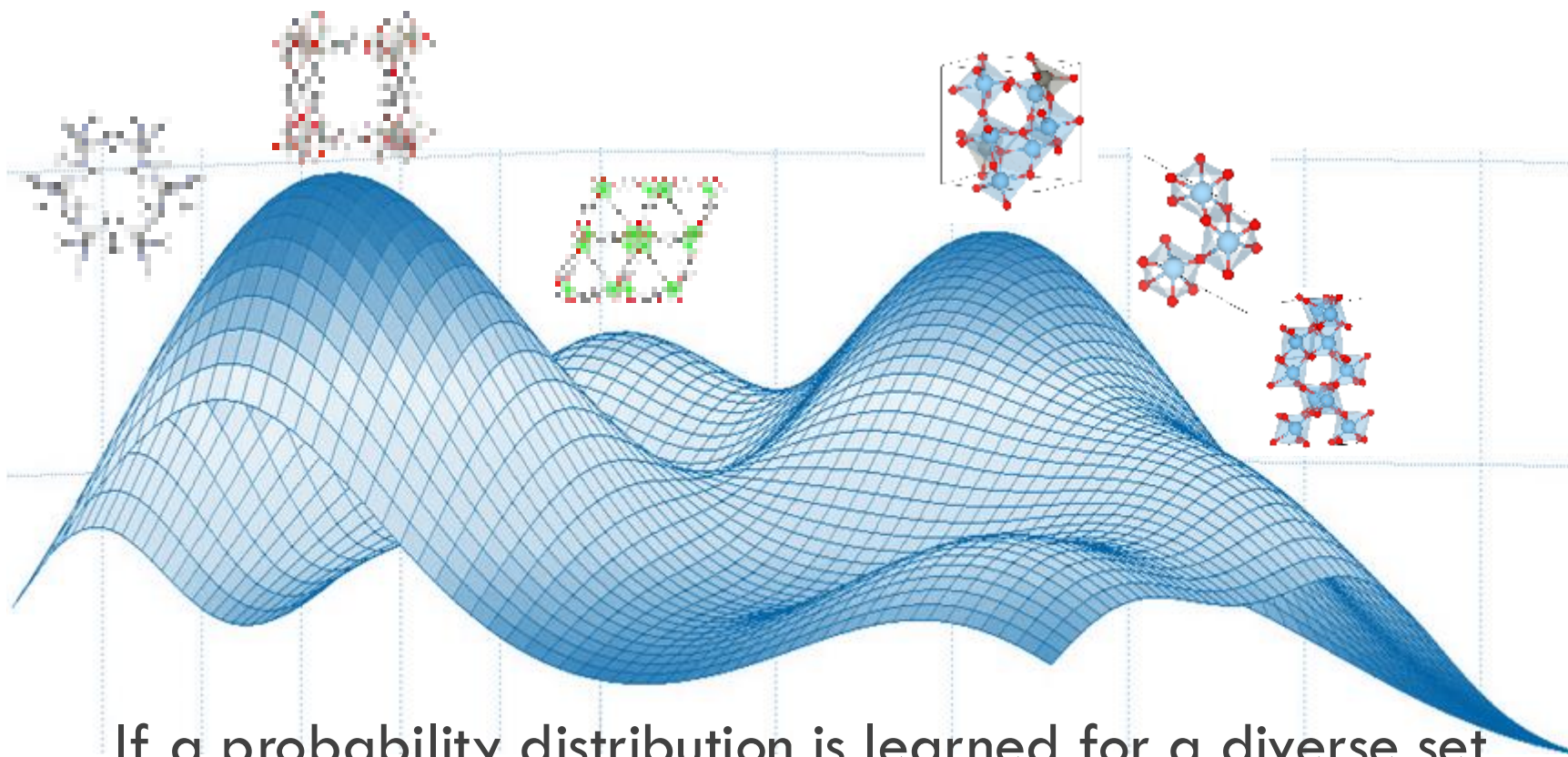
LLMs for Materials

Combine text and structural data for multi-model models using contrastive learning



Sampling Materials Space

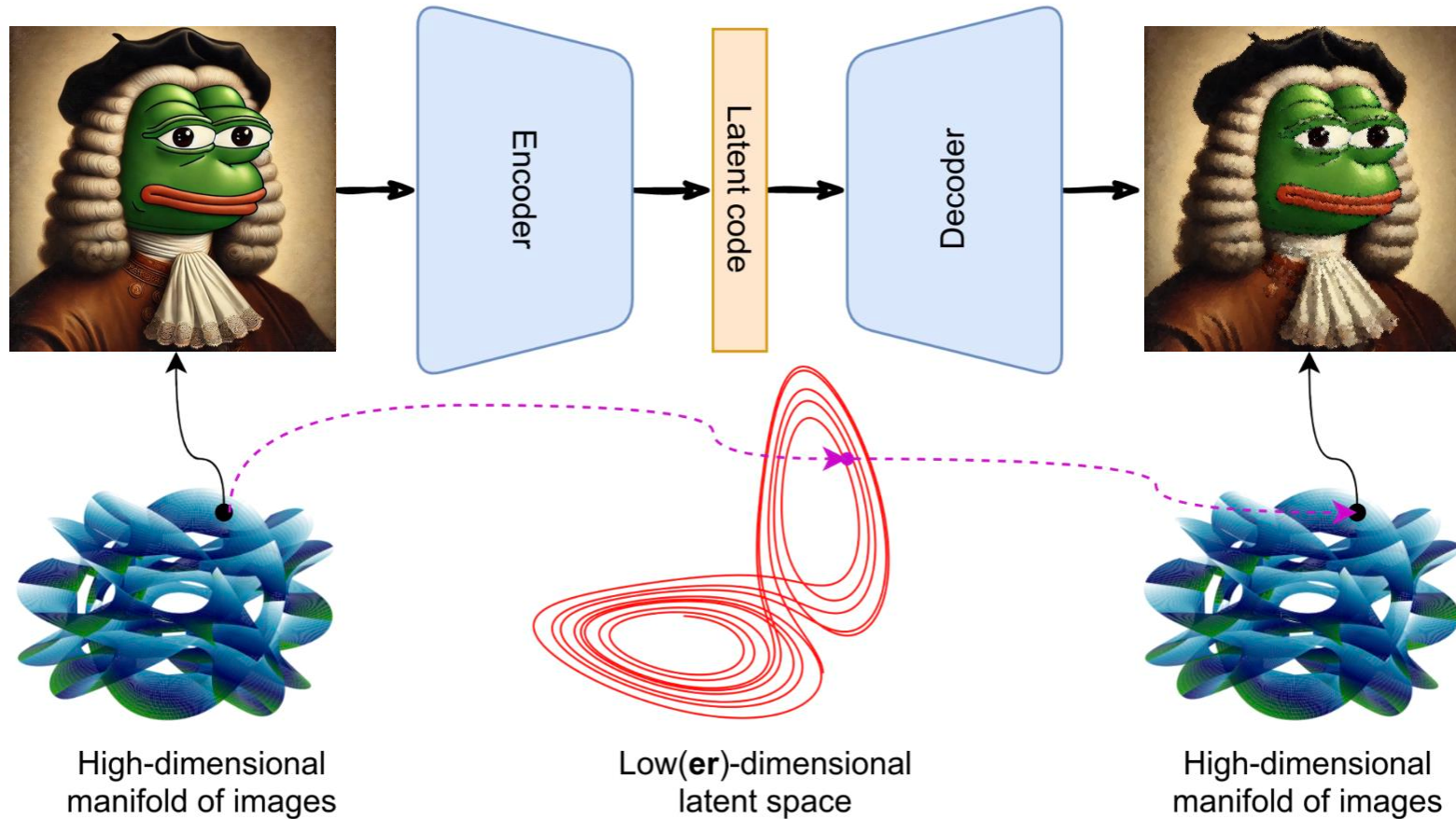
A high-dimensional space combining chemical composition, structure, processing, properties



If a probability distribution is learned for a diverse set of known materials, it may be used to target new compounds

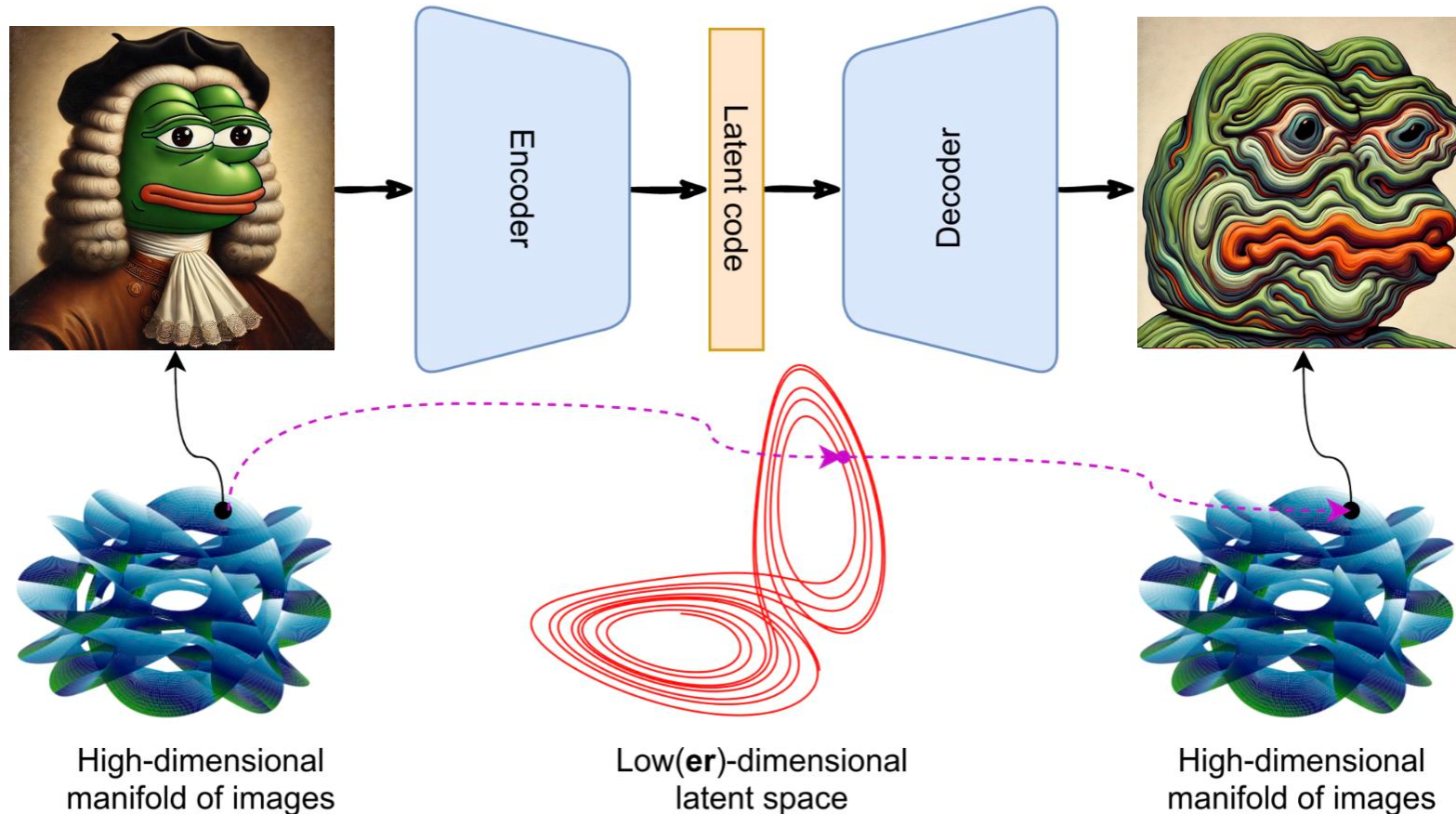
Autoencoder

Neural network compresses data into a deterministic latent space and reconstructs it back to the original



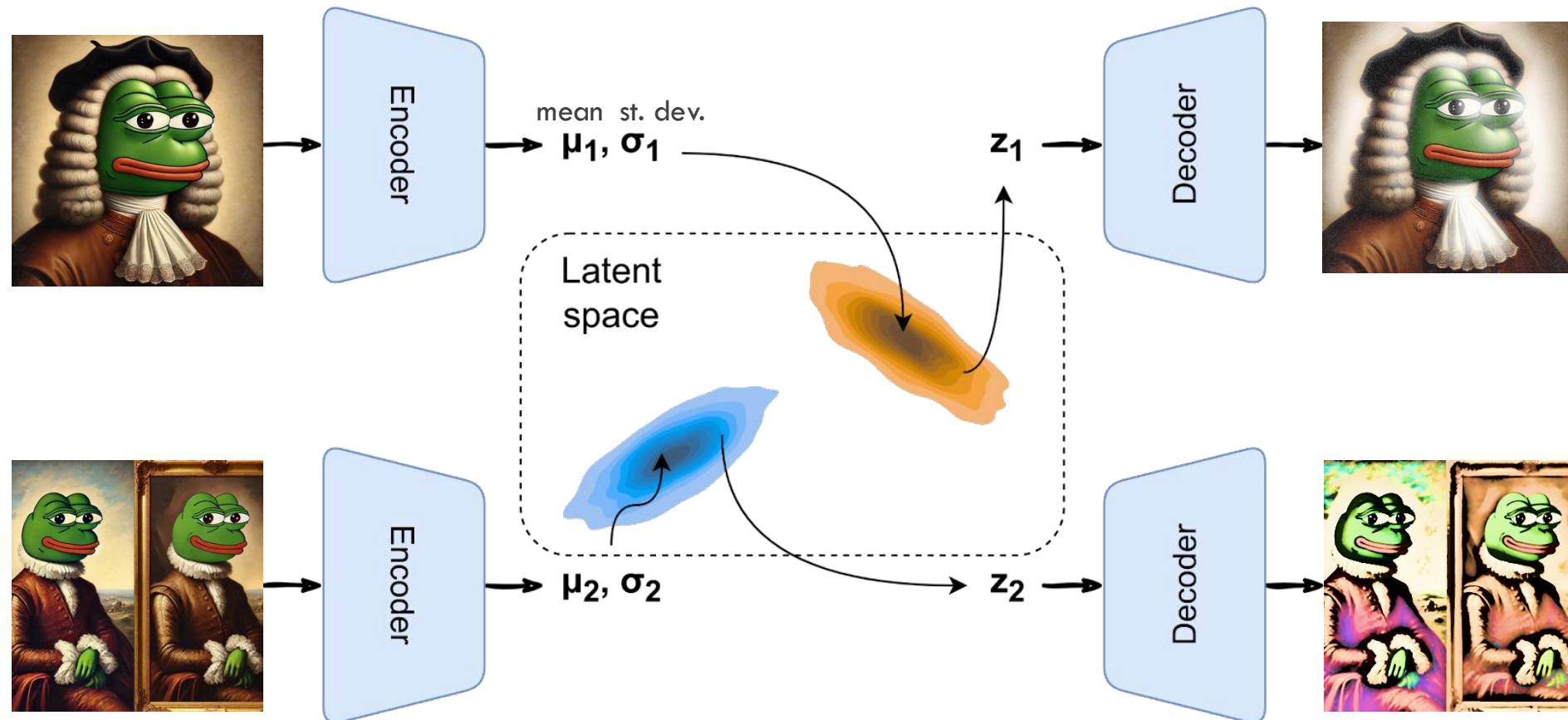
Autoencoder

Lack of continuity and structure makes interpolated or random points unlikely to map to meaningful data



Variational Autoencoder

Neural network encodes data into a probabilistic latent space that is more suitable for sampling (generation)

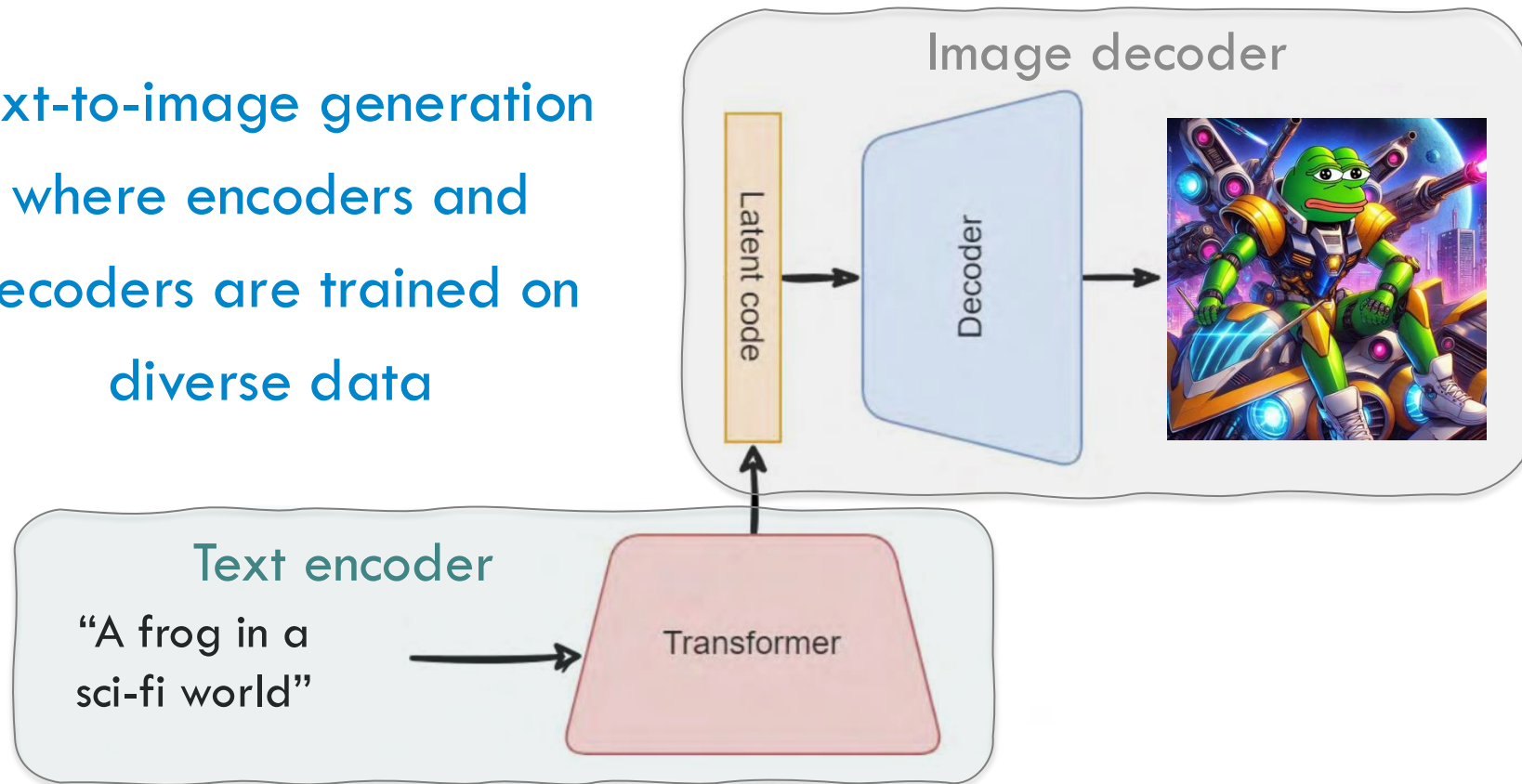


D. P. Kingma and M. Welling (2013); Schematic adapted from <https://synthesis.ai>

Generative Artificial Intelligence

Create realistic data by sampling from learned latent space (probability distributions)

Text-to-image generation
where encoders and decoders are trained on diverse data

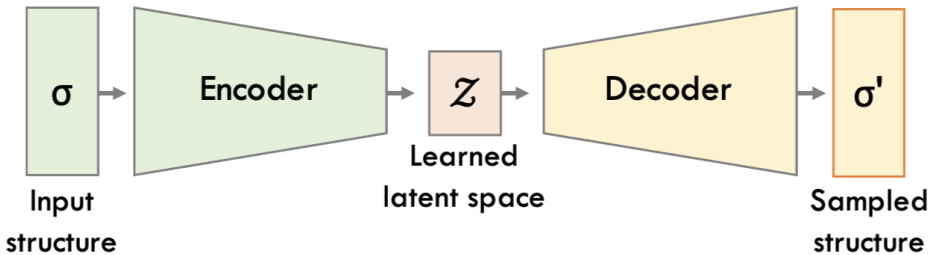


All images were generated by DALL-E 3 (OpenAI)

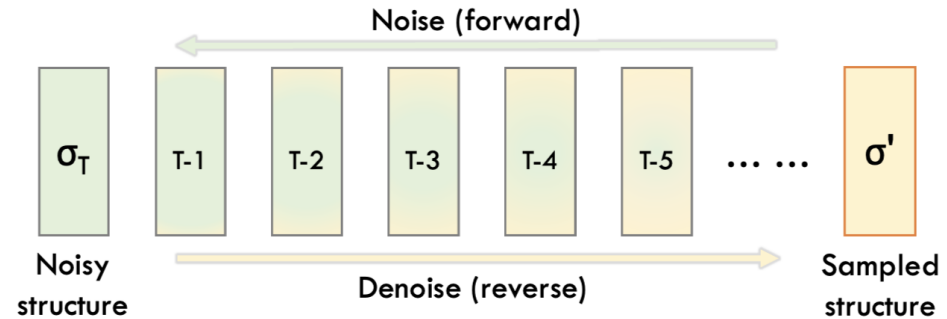
Generative Artificial Intelligence

Growing range of generative architectures
can be tailored for scientific problems

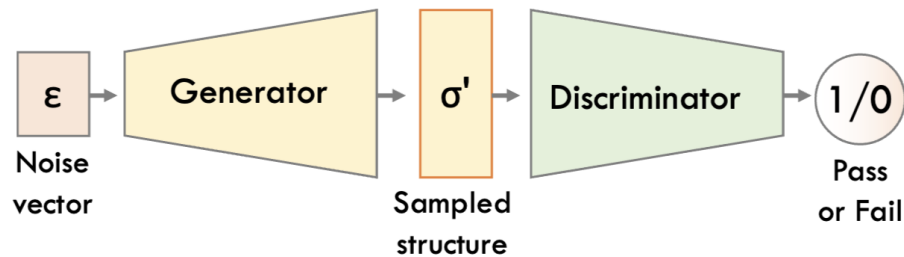
Variational autoencoder (VAE)



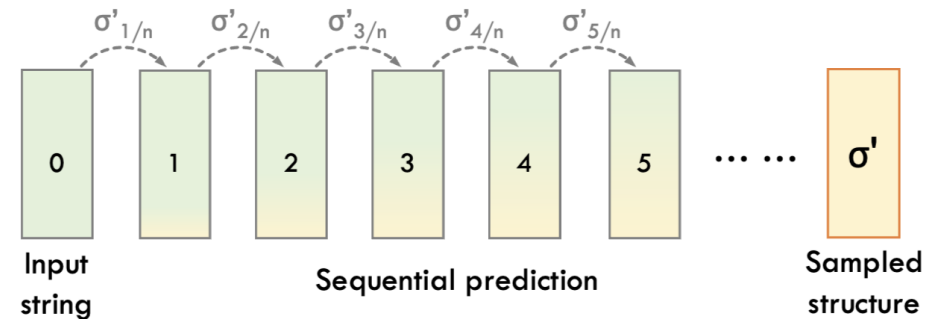
Denoising diffusion



Generative adversarial network (GAN)



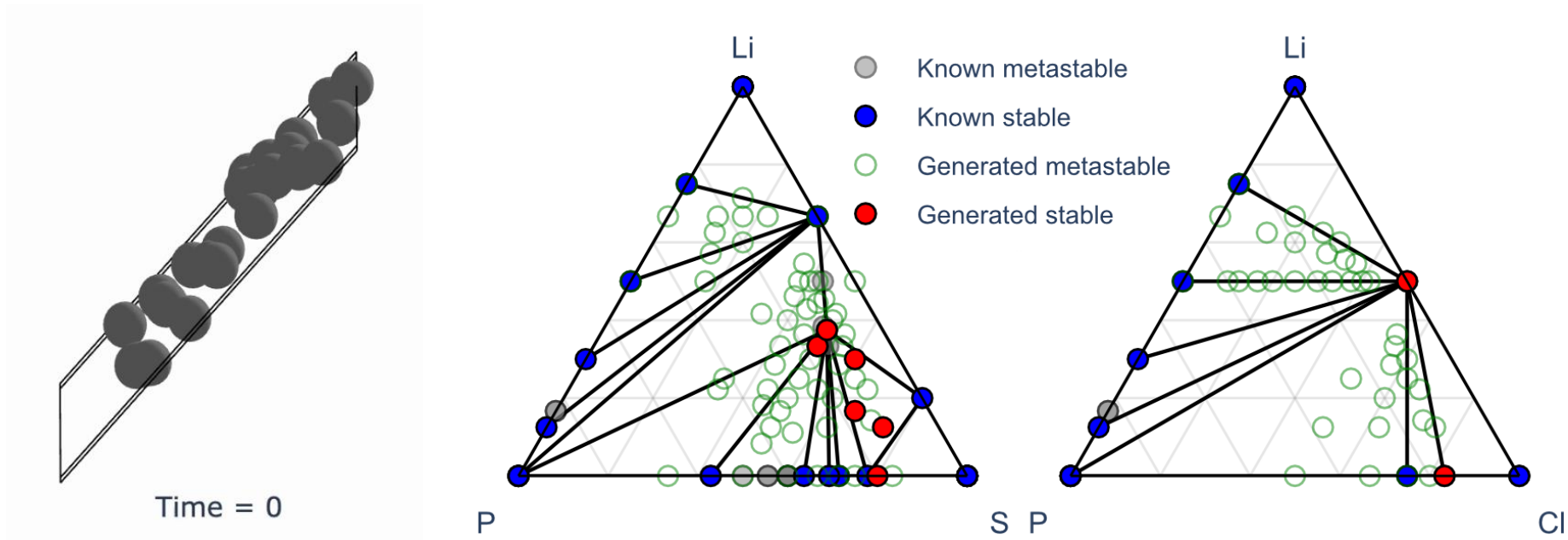
Autoregressive model



Application to Materials Design

GenAI models can be used in different ways, e.g.

- map from composition to crystal structure
- unguided sampling of a random compound
- guided sampling to specific properties



Chemeleon Example

As easy as “pip install chemeleon”

```
from chemeleon import Chemeleon
from chemeleon.visualize import Visualizer
from ase.io import write
import os

# Load model
composition_model = Chemeleon.load_composition_model()

# Set parameters
n_samples = 10
n_atoms = 8
prompt = "Ti O S"

# Generate crystal structures
atoms_list = composition_model.sample(prompt, n_atoms, n_samples)

# Visualise
visualizer = Visualizer(atoms_list)
visualizer.view(index=0)

# Save cifs
output_folder = "chemeleon_structures"
os.makedirs(output_folder, exist_ok=True)

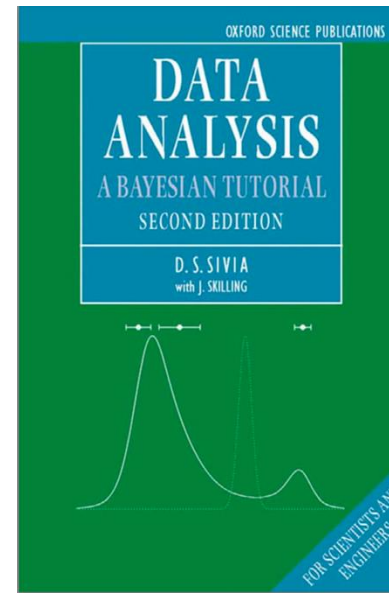
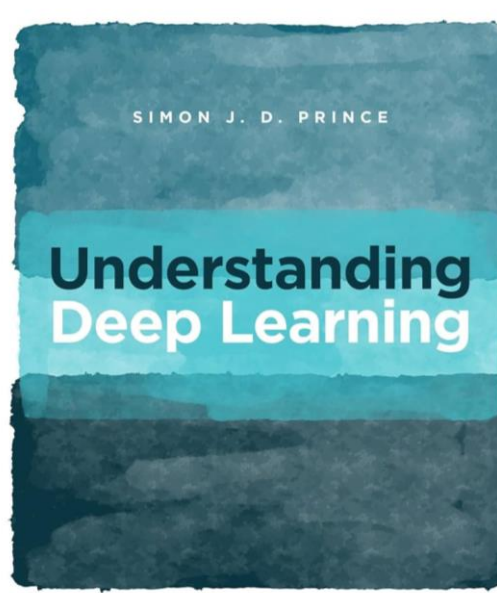
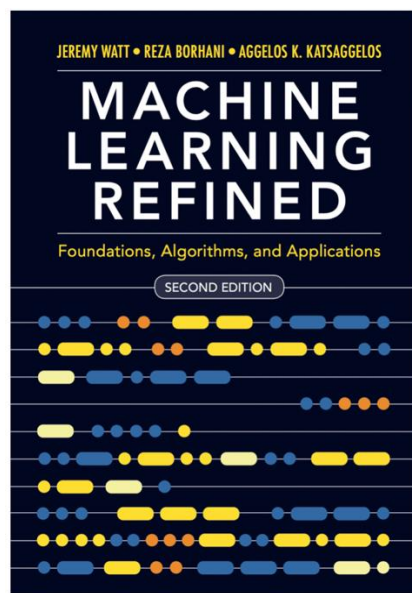
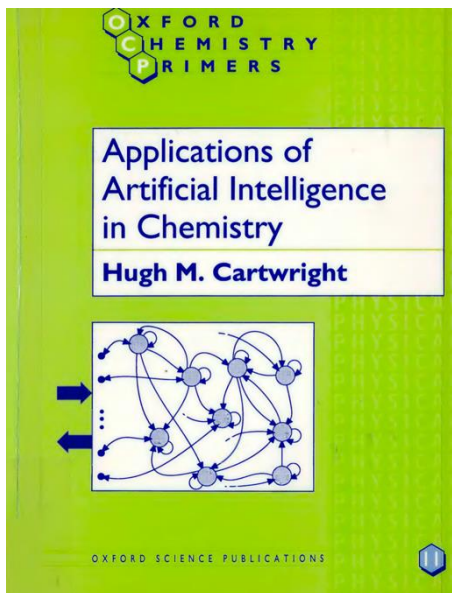
for i, atoms in enumerate(atoms_list):
    filename = os.path.join(output_folder, f"structure_{i+1}.cif")
    write(filename, atoms)
    print(f"Structure saved as {filename}")
```

<https://github.com/hspark1212/chemeleon>

Dive Deeper

AI content available from many sources, including blogs, research papers, repositories, and textbooks

e.g. <https://aronwalsh.github.io/MLforMaterials/Resources.html>



General

Specialist