Frontiers in Computer Science and Technology

计算机科学与技术前沿

Spring 2026

Tailin Wu, Westlake University

Website: https://ai4s.lab.westlake.edu.cn/course/frontiers-2026/

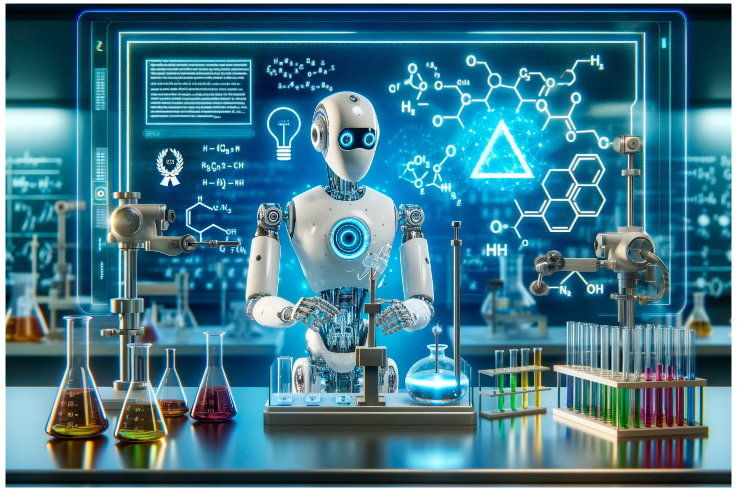


Image from: SciTechDaily

Outline

- **Course logistics**
- Why study this course?
- Course introduction
    - Tasks
    - Neural Architecture
    - Learning paradigm
    - Application in AI and Science

Course logistics

We are going to learn important branches of **AI techniques** and their **application in science**, both the **fundamentals** and **the frontiers**:

AI techniques:

- Deep Learning
- Generative models
- Foundation models
- AI Agents for learning and research
- Reinforcement learning
- Robotics
- Graph Neural Network
- Evolutionary Machine Learning and Multi-objective Optimization

AI + Science:

- AI + PDEs/scientific computing
- AI + life science
- Science for AI

Course timeline

All teachers are PIs in the AI direction in Westlake University

#	Topic	Date	Teacher
1	Course introduction	Fri, 3/6	Tailin Wu
2	Frontiers in Deep Learning	Fri, 3/13	Tailin Wu
3	Frontiers in Generative Modeling	Fri, 3/20	Tailin Wu
4	Foundation models	Fri, 3/27	Zhengzhong Lan
5	AI Agents for learning and research	Fri, 4/3	Tailin Wu
6	Introduction to Reinforcement Learning	Fri, 4/10	Tailin Wu
7	Reinforcement Learning: Advanced techniques and applications	Fri, 5/08	Tailin Wu
10	Frontiers in Graph Neural Networks	Fri, 5/09	Tailin Wu
11	AI + Life Sciences	Fri, 5/15	Tailin Wu
12	Evolutionary Machine Learning and Multi-objective optimization	Fri, 5/22	Tailin Wu
13	AI + PDE/Scientific computing	Fri, 5/29	Tailin Wu
14	Science for AI	Fri, 6/5	Tailin Wu

*For project timeline see the slides later.

Course arrangement

Time: Every Friday 13:30pm – 15:05pm

Typical time split:

- 13:30 – 14:15pm (45min): First part
- 14:15 – 14:20pm (5min): Discussion
- 14:20 – 15:40pm (45min): Second part

No prerequisite for the course

But would be good to have a basic understanding of neural network, its training, and write basic neural network with PyTorch

- If you don’t have any of the above background, that is fine. You only need **half a day** to master them, using the materials provided in this lecture.

Course grading

Assessment Criteria	Percentage
Attendance	5%
Project proposal and discussion	25%
Small project	15%
Project conclusion presentation	30%
Project conclusion report	25%

Grade	Assessment Standard
A	90-100 points
B	80-89 points
C	70-79 points
D	60-69 points
F	Below 60 points

The grading will be **generous**. The important thing is that you learn **useful AI techniques** that you can use for your own **research**.

Course Project proposal (presentation, 25 points)

Choose a **problem related to your research**, and **use AI to solve it.**

Team size: 2-4 people, encouraging **interdisciplinary** collaboration (+5 points if a team consists of both AI and non-AI students)

Course project design @ midterm:

- Give a presentation (10min) that formulates the problem for the 5 questions, each with one slide:
  1. What is the problem?
  2. Why is it important
  3. Why is it hard?
  4. What is the limitation of the prior method?
  5. What is the main component of the proposed method?

  Then detail the proposed method (3-4 slides) that uses an AI technique to solve the problem

#	Topic	Date	Teacher
8	Course project design (1)	Fri, 4/17	Tailin Wu
9	Course project design (2)	Fri, 4/24	Tailin Wu

Course Project presentation (30 points) and final report (25 points)

Course project report and discussion @ final:

Give a presentation (15min) that

- Formulates the problem in terms of the 5 questions before, each with one slide.
- Then detail the proposed method (3-4 slides) that uses an AI technique to solve the problem.
- Then report the main experiment results (3-4 slides)

Submit a project report that summarize the project.

Each person should **play important role** in the project, and the presentation and project report should **detail what each person has done**

#	Topic	Date	Teacher
15	Course Project Reporting and Discussion (1)	Fri, 6/12	Tailin Wu
16	Course Project Reporting and Discussion (2)	Fri, 6/24	Tailin Wu

Small Project (15%)

Most lectures are accompanied with a **minimal notebook** for the AI technique introduced in the class
The small project requires **replacing the dataset** from **one of the notebook** by one of your own dataset in your research, try to reach a good performance by tuning the hyperparameters, and report the results.

Submit a *small project* report including:

- A pdf with no less than 2 pages
  - Introducing the new dataset and the task (-5 points if ≥2 people have the same dataset)
  - What hyperparameters you have tuned, and which yields the best performance
  - Analysis of the results and visualizations
- A zip for the code.

Outline

- Course logistics
- **Why study this course?**
- Course introduction
  - Tasks
  - Neural Architecture
  - Learning paradigm
  - Application in AI and Science

Your research interest

Self-introduction:

Department & lab

Research interest

Familiarity (1-10) with deep learning

Why study this course?

If you are from science/engineering background, not so familiar with AI, you will:

- Know major AI techniques, their application areas, and limitations
- Able to use state-of-the-art AI techniques for your own research

If you are from AI background, you will:

- Learn state-of-the-art AI techniques in different subfields
- Know the open research problems for each subfield
- Collaborate and explore interdisciplinary research

Outline

- Course logistics
- Why study this course?
- **Course introduction**
  - Tasks
  - Neural Architecture
  - Learning paradigm
  - Application in AI and Science

Course introduction

Tasks

- Classification/ regression
- Simulation
- Inverse design/ inverse problem
- Control/planning

×

Neural architecture

- Multilayer perceptron
- Graph Neural Networks
- Convolutional Neural Networks
- Transformers

×

Learning paradigm

- Supervised learning
- Generative modeling
- Foundation models
- Reinforcement learning
- Evolutionary and multi-objective optimization

Application (AI & Science)

- Robotics
- Games (e.g., Go, atari)
- Autonomous Driving
- PDEs
- Life science
- Materials science

Course introduction: tasks

Tasks

- Classification/ regression
- Simulation
- Inverse design/ inverse problem
- Control/planning

×

Neural architecture

- Multilayer perceptron
- Graph Neural Networks
- Convolutional Neural Networks
- Transformers

×

Learning paradigm

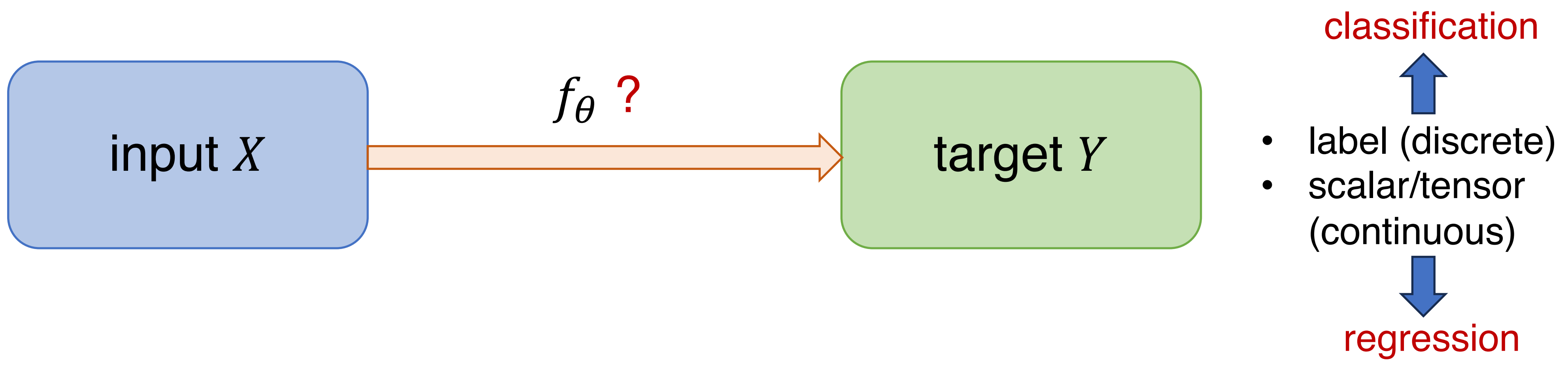
- Supervised learning
- Generative modeling
- Foundation models
- Reinforcement learning
- Evolutionary and multi-objective optimization

Application (AI & Science)

- Robotics
- Games (e.g., Go, atari)
- Autonomous Driving
- PDEs
- Life science
- Materials science

Task 1: Classification & regression

- image
- video
- graph
- time series
- natural language
- …



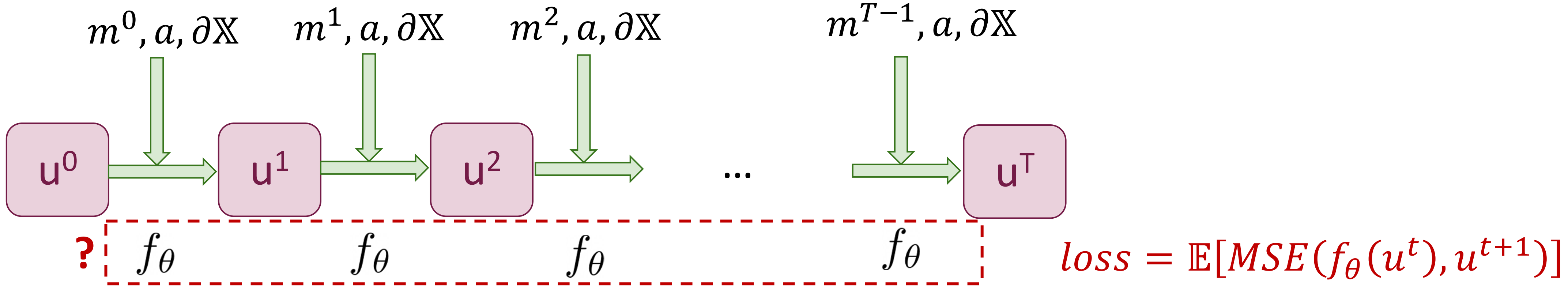
Given many examples of (X, Y) pairs, learn a neural network (NN) f_θ that minimizes the prediction loss:

$$\theta^* = \text{argmin}_\theta\, \mathbb{E}_{(X,Y)\sim P(X,Y)}[\ell(f_\theta(X), Y)]$$

f_θ: neural network to be learned
ℓ: loss function

Task 2: (Learning) simulation

Goal: learn the mapping f_θ from u^t to u^{t+1}:


$m^0, a, \partial\mathbb{X}$
$m^1, a, \partial\mathbb{X}$
$m^2, a, \partial\mathbb{X}$
$m^{T-1}, a, \partial\mathbb{X}$
u0
u1
u2
...
uT
?
f_θ
f_θ
f_θ
f_θ
$loss = \mathbb{E}[MSE(f_\theta(u^t), u^{t+1})]$

u^t: original **state**（状态）of the system. Can be a graph (e.g., mesh, particle-based systems, molecules), a tensor, or an infinite-dimensional function $u(t, x)$ as solution to a PDE

f_θ: **neural surrogate models**（神经网络代理模型）

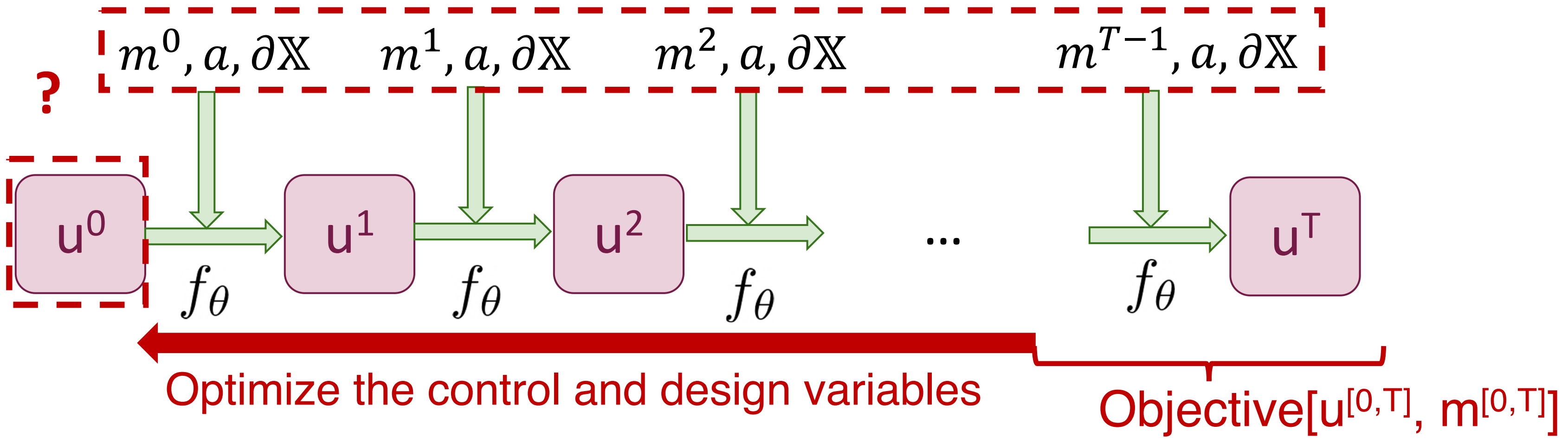
m^t: **external control**（外界控制）

a: **static parameters**（静态参数） of the system that does not change with time (e.g. parameters of PDE, spatially varying diffusion coefficient)

$\partial\mathbb{X}$: **boundary condition**（边界条件） of the system

PDE: partial differential equation
ODE: ordinary differential equation

Tasks 3 & 4: Inverse design, inverse problem, and control



$m^0, a, \partial\mathbb{X}$
$m^1, a, \partial\mathbb{X}$
$m^2, a, \partial\mathbb{X}$
$m^{T-1}, a, \partial\mathbb{X}$
?
u^0
u^1
u^2
...
u^T
f_θ
f_θ
f_θ
f_θ
Optimize the control and design variables
Objective[$u^{[0,T]}$, $m^{[0,T]}$]

u^t: original **state** of the system. Can be an infinite-dimensional function $u(t, x)$ as solution to a PDE, or a graph (e.g., mesh, particle-based systems, molecules)

f_θ: neural surrogate models

m^t: external **control（外界控制）**

a: **static parameters（静态参数）** of the system that does not change with time (e.g. parameters of PDE, spatially varying diffusion coefficient)

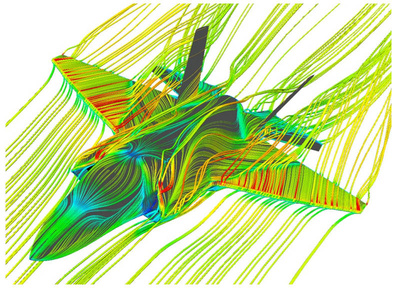
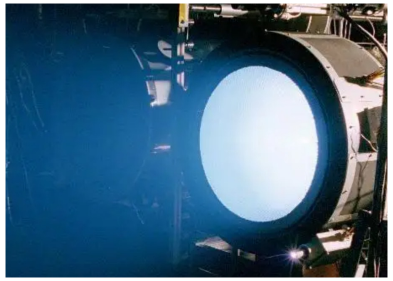
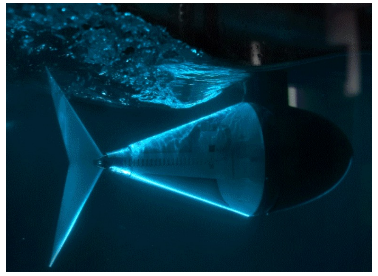
$\partial\mathbb{X}$: **boundary condition（边界条件）** of the system

} control（控制）

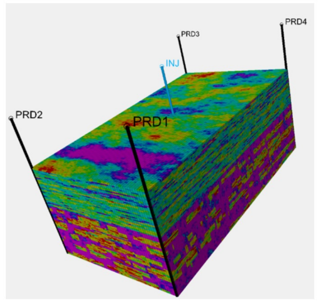
} inverse design（反向设计）

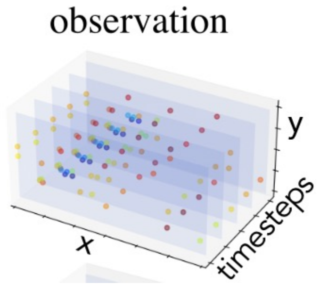
Tasks 3 & 4: Inverse design, inverse problem, and control

- **Inverse design:** boundary $\partial\mathbb{X}$, initial condition u^0, parameter a to **optimize design objective:** plane design, rocket shape, underwater robot shape

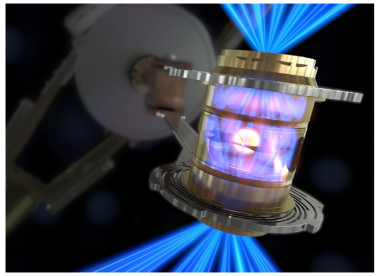




- **Inverse problem : infer initial condition** u^0 or parameter a to **match prediction with observation**





- **Control: optimize control** m^t to **optimize control objectives**: controlled nuclear fusion, robotics



Tasks 2 & 3: Steady-state simulation and inverse design

Simulation:

$$\begin{tikzcd} \boxed{a, \partial\mathbb{X}} \arrow[r, "f_\theta\ ?"] & \boxed{u} \end{tikzcd}$$

parameter and boundary

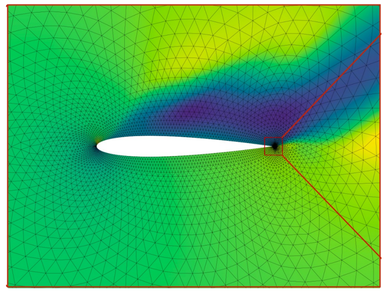
steady state of the system

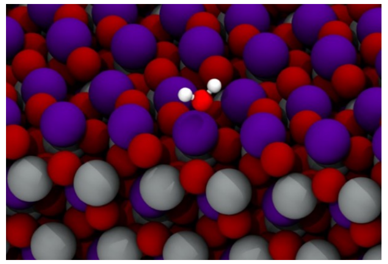
Inverse design/inverse problem:

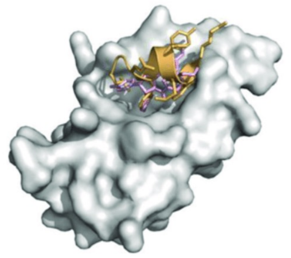
$$\begin{tikzcd} \boxed{?\ a, \partial\mathbb{X}} \arrow[r, "f"] & \boxed{u} \end{tikzcd}$$

parameter and boundary

steady state of the system


Aerodynamics simulation


Materials design


Protein design

Course introduction: Neural architecture

Tasks

- Classification/ regression
- Simulation
- Inverse design/ inverse problem
- Control/planning

×

Neural architecture

- Multilayer perceptron
- Graph Neural Networks
- Convolutional Neural Networks
- Transformers

×

Learning paradigm

- Supervised learning
- Generative modeling
- Foundation models
- Reinforcement learning
- Evolutionary and multi-objective optimization

Application (AI & Science)

- Robotics
- Games (e.g., Go, atari)
- Autonomous Driving
- PDEs
- Life science
- Materials science

Neural architecture: overview

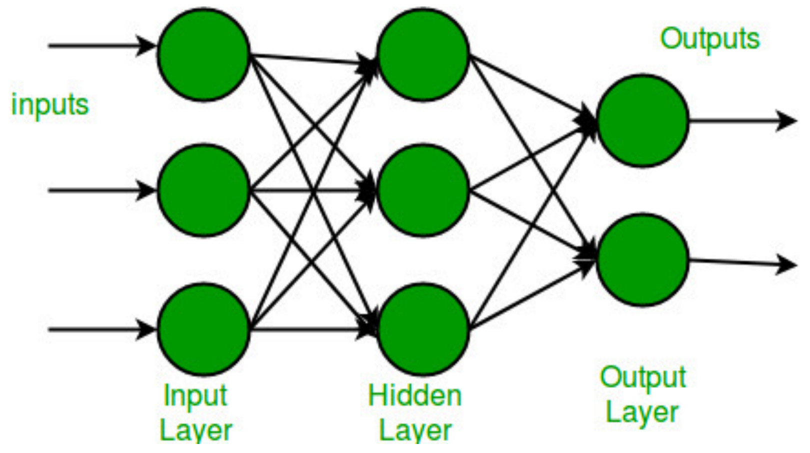
The choice of neural architecture depend on the data structure:

Data structure	Examples	Suitable neural architecture
Vector	simple vectors	Multilayer Perceptron (MLP)
Graph	molecules, irregular mesh	Graph Neural Network (GNN)
Grid	image, videos	Convolutional Neural Network (CNN)
Sequence	time series, natural language	Transformer

Neural architecture 1: Multilayer Perceptron (MLP)

f_θ

input $x \in R^d$ (vector)

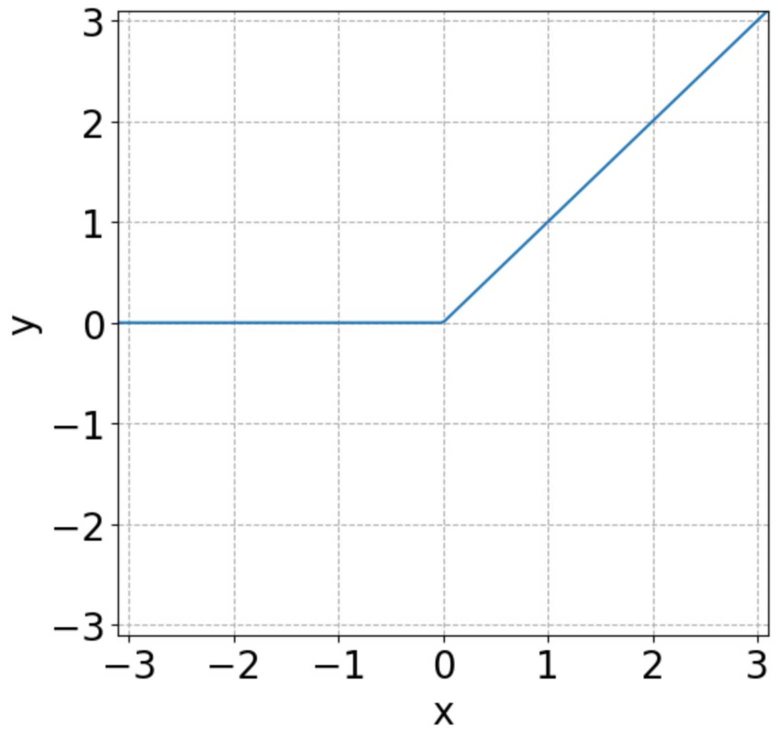


prediction $\hat{y}$

An MLP f_θ with n layers: $\hat{y} = W_n\sigma(\ldots\sigma(W_2\sigma(W_1x + b_1) + b_2) + b_n$

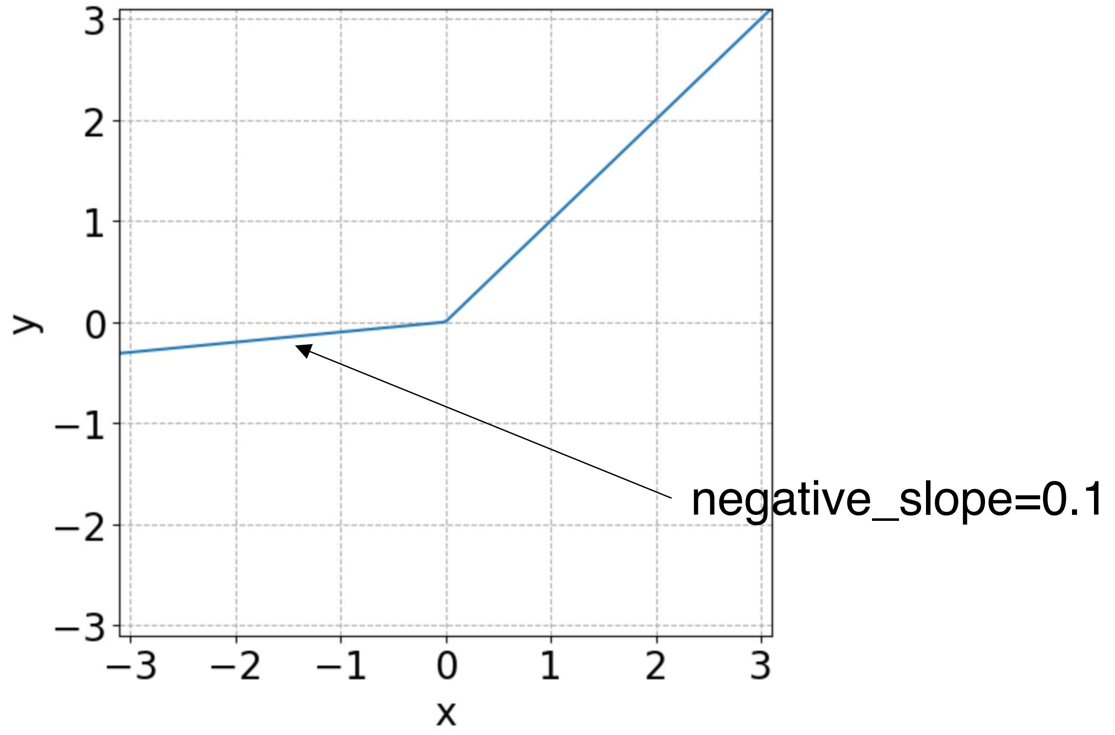
W_i: weight matrix to be learned
b_i: bias vector to be learned
σ: (nonlinear) activation function, e.g., ReLU, softplus, ELU

Activation function



ReLU (Rectified Linear Unit)

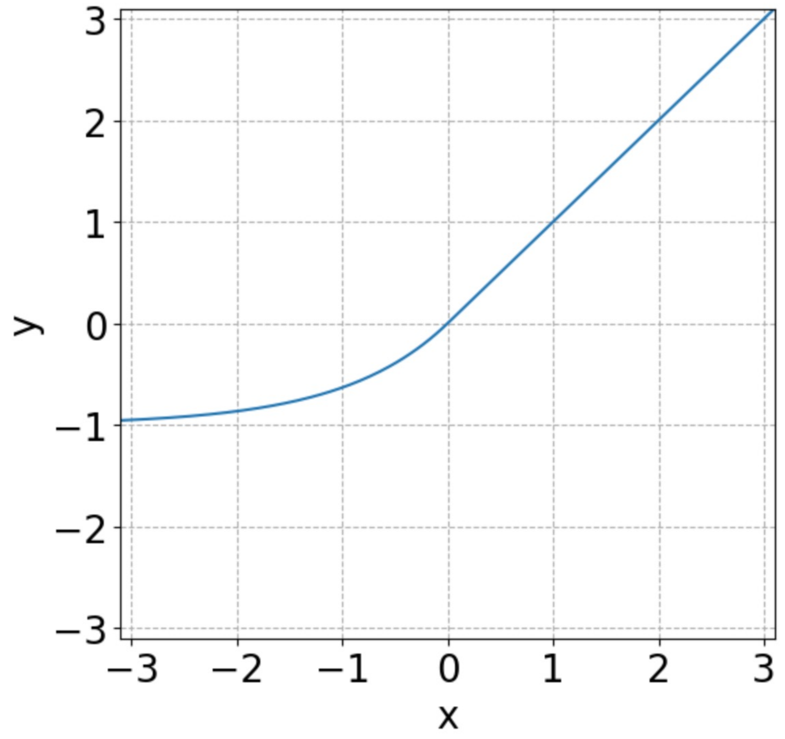
$$\mathrm{ReLU}(x) = \max(0, x)$$



LeakyReLU

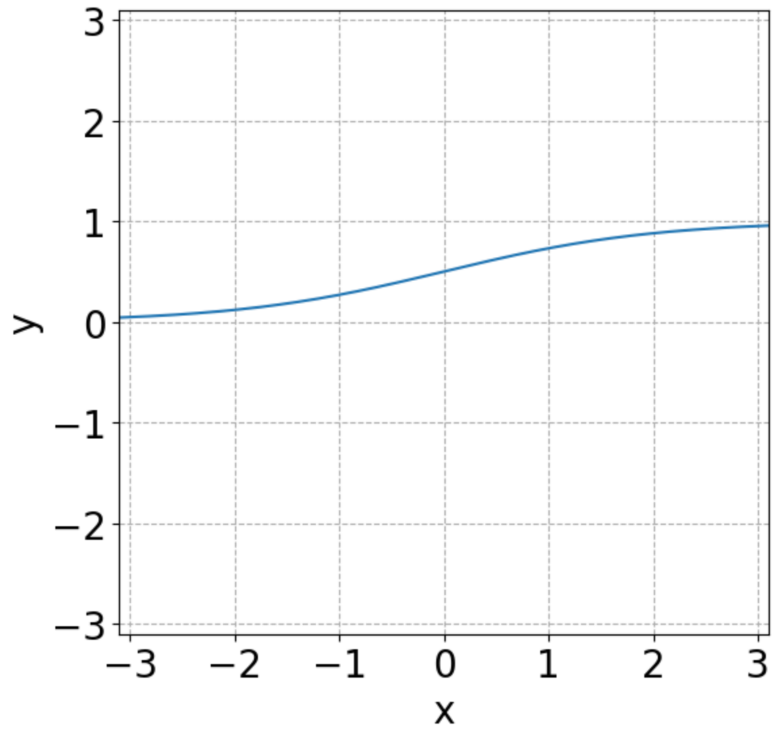
$$\mathrm{LeakyReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \text{negative\_slope} \times x, & \text{otherwise} \end{cases}$$

Activation function



ELU (Exponential Linear Unit)

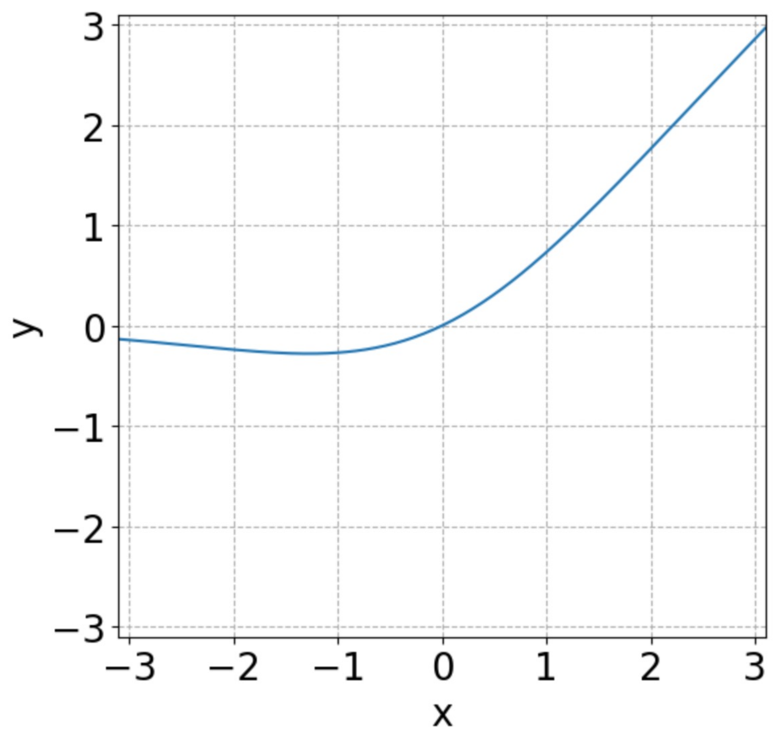
$$\text{ELU}(x) = \begin{cases} x, & x > 0 \\ e^x - 1, & x \leq 0 \end{cases}$$



Sigmoid

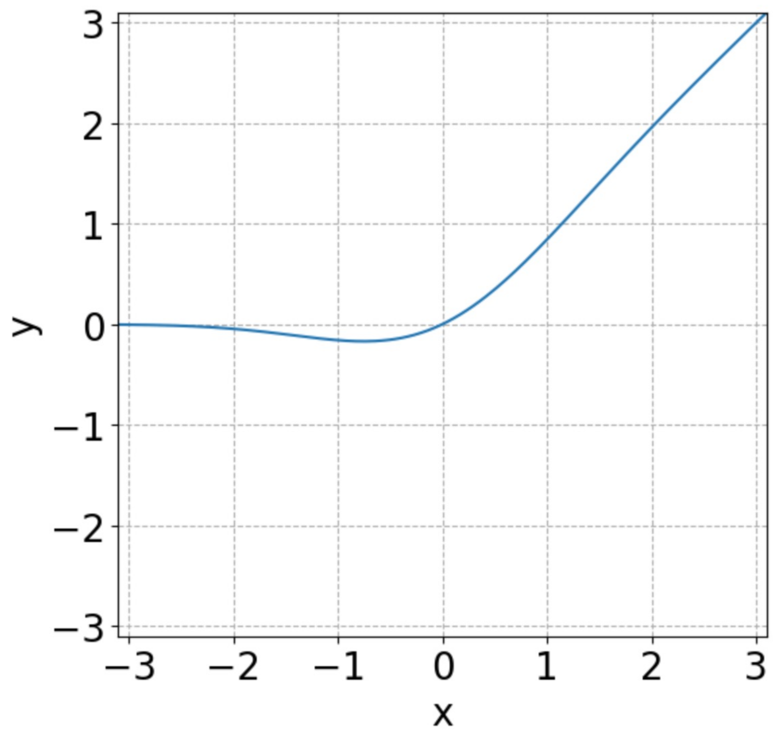
$$\text{Sigmoid}(x) = \frac{1}{1+\exp(-x)}$$

Activation function



SiLU (Sigmoid Linear Unit)

SiLU$(x) = x \cdot \mathrm{Sigmoid}(x)$



GELU (Gaussian Error Linear Unit)

SiLU$(x) = x \cdot \Phi(x)$

$\Phi(x)$ is the cumulative distribution function for Gaussian distribution

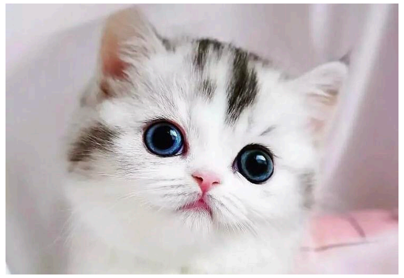
Activation function

Activation function	Advantages	Drawbacks
ReLU	Simple, suitable for classification	Can have some “dead neurons” The network is piecewise linear
LeakyReLU	Does not have dead neurons	The network is piecewise linear
ELU	Typically useful for regression	
Sigmoid	Output contrained to [0,1]	If input is far from 0, then have saturation (vanishing gradient)
SiLU	Typically useful for regression	
GELU	Typically useful for regression	

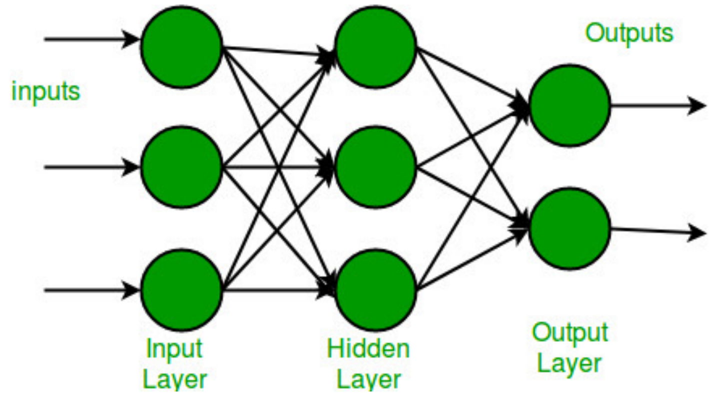
Typically try ReLU, LeakyReLU, ELU, and SiLU in hyperparameter search

MLP: universal approximation theorem

input



f_θ



target

“cat”

An MLP f_θ that has 1 hidden layer (with arbitrary width) and a nonlinear activation function can approximate any function to arbitrary precision [1][2].

Here $f_\theta(x) = W_2\sigma(W_1 x + b_1)$

- With one hidden layer, may need exponential number of neurons w.r.t. input size
- With more layers, the neurons needed may be polynomial [3]

[1] Funahashi, Ken-Ichi. "On the approximate realization of continuous mappings by neural networks." Neural networks 2.3 (1989): 183-192.
[2] Hornik, Kurt, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators." Neural networks 2.5 (1989): 359-366.
[3] Rolnick, David, and Max Tegmark. "The power of deeper networks for expressing natural functions." ICLR 2018

Learning with gradient descent

$$f_\theta(x) = \sigma(W_n\sigma(\dots\sigma(W_2\sigma(W_1x + b_1) + b_2)\dots + b_n)$$

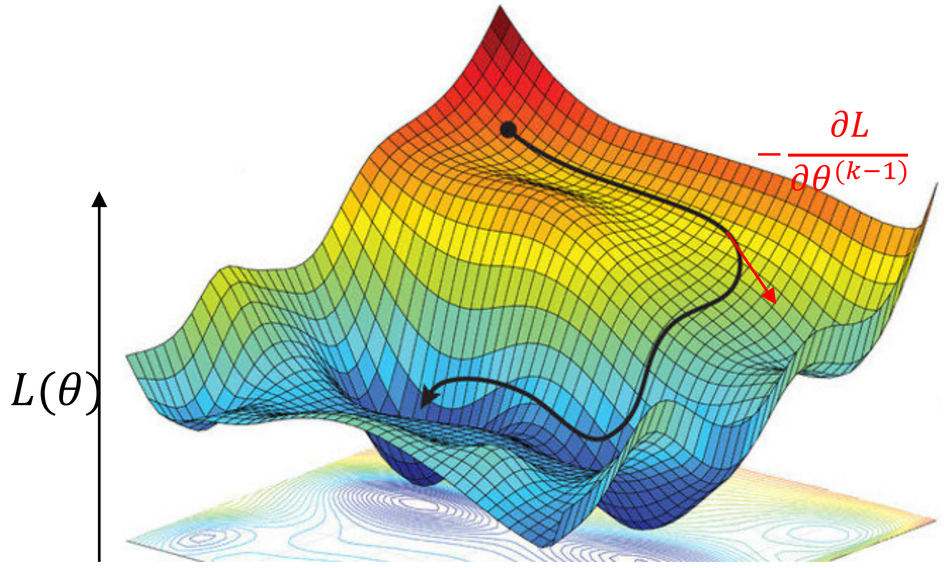
To fit dataset $\{(x_i, y_i)\}, i = 1,2,\dots N$, we can use Mean Squared Error (MSE):

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - f_\theta(x_i)\right)^2$$

How can we optimize the parameter $\theta = (W_1,,\dots W_n, b_1,\dots b_n)$?

Answer: compute $\frac{\partial L}{\partial \theta}$, then we can perform gradient descent :

$$\theta^{(k)} \leftarrow \theta^{(k-1)} - \eta\,\frac{\partial L}{\partial \theta^{(k-1)}}$$



θ (typically high dimensional)

η: learning rate

Backpropagation

Consider:

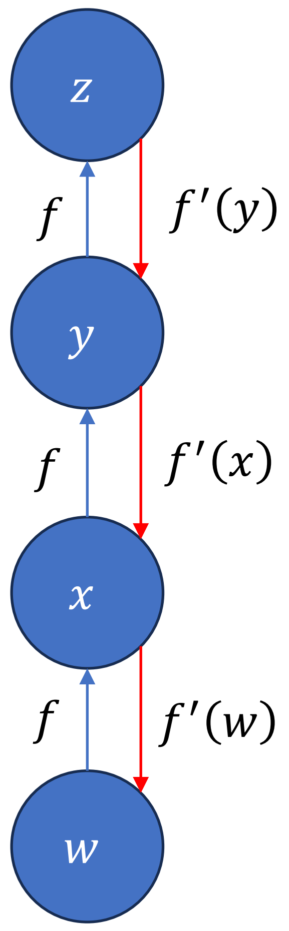
$$z = f(y), y = f(x), x = f(w)$$
$$z = f\left(f(f(w))\right)$$

Chain rule:

$$\frac{\partial z}{\partial w} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}\frac{\partial x}{\partial w} = f'(y)f'(x)f'(w)$$

Observation:

1. We need to store intermediate result x, y to avoid recomputing them.
2. Goes layer-by-layer from output to input.



Backpropagation

Let's take a two layer MLP $f_\theta(x) = W_2\sigma(W_1x + b_1)$ as an example:

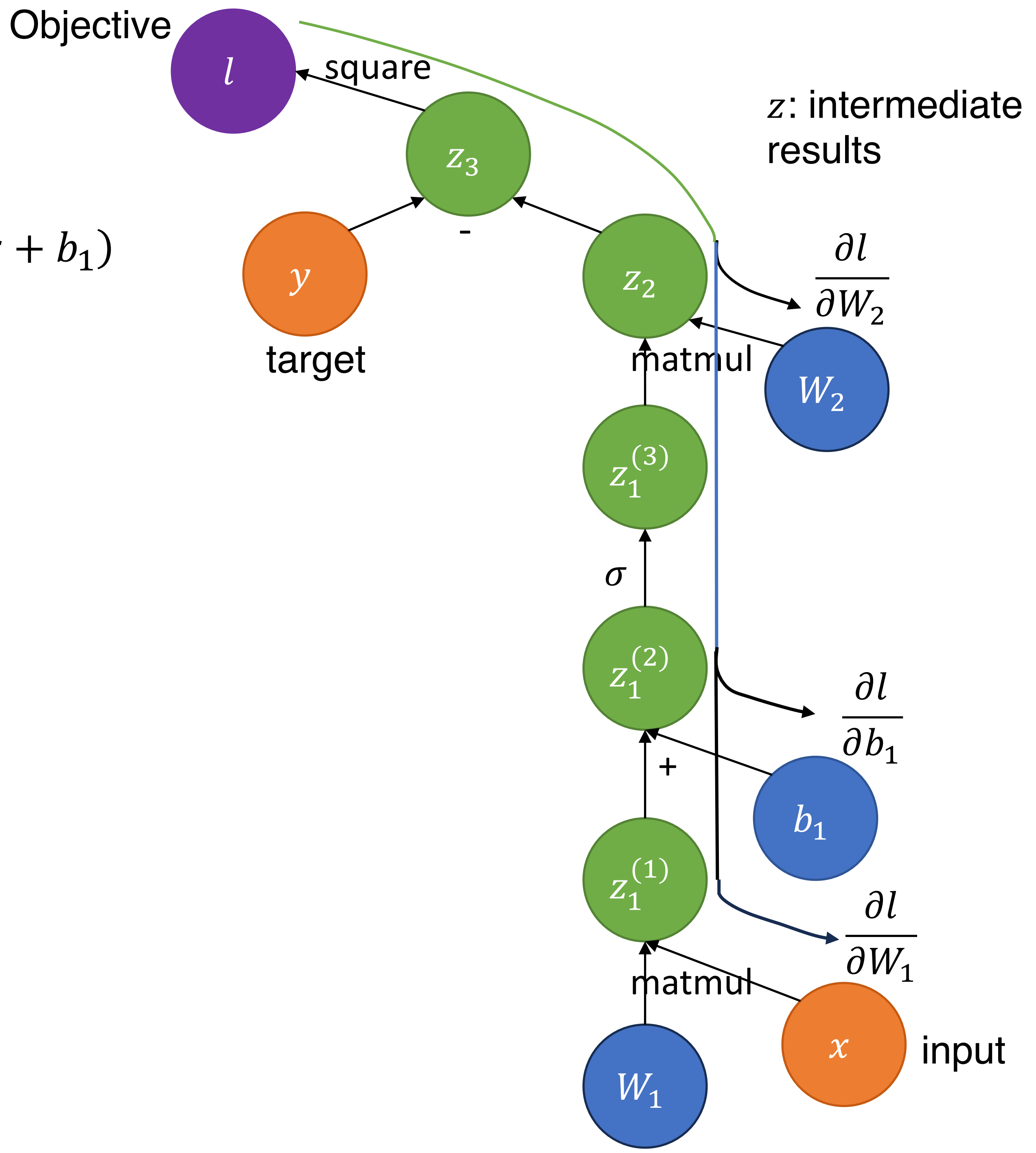
Objective: $l = \left(y - f_\theta(x)\right)^2$

$$\frac{\partial l}{\partial W_2} = \boxed{\frac{\partial l}{\partial z_3} \cdot \frac{\partial z_3}{\partial z_2}} \cdot \frac{\partial z_2}{\partial W_2}$$

$$\frac{\partial l}{\partial b_1} = \boxed{\frac{\partial l}{\partial z_3} \cdot \frac{\partial z_3}{\partial z_2}} \cdot \boxed{\frac{\partial z_2}{\partial z_1^{(3)}} \cdot \frac{\partial z_1^{(3)}}{\partial z_1^{(2)}}} \cdot \frac{\partial z_1^{(2)}}{\partial b_1}$$

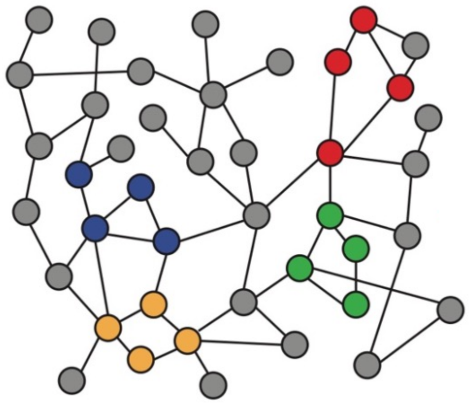
$$\frac{\partial l}{\partial W_1} = \boxed{\frac{\partial l}{\partial z_3} \cdot \frac{\partial z_3}{\partial z_2}} \cdot \boxed{\frac{\partial z_2}{\partial z_1^{(3)}} \cdot \frac{\partial z_1^{(3)}}{\partial z_1^{(2)}}} \cdot \frac{\partial z_1^{(2)}}{\partial z_1^{(1)}} \cdot \frac{\partial z_1^{(1)}}{\partial W_1}$$

shared, no need to recompute



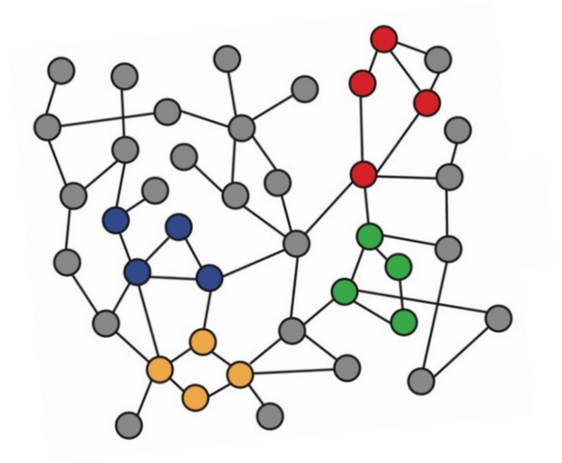
Objective
l
square
z_3
y
target
-
z_2
z: intermediate results
$\frac{\partial l}{\partial W_2}$
matmul
W_2
$z_1^{(3)}$
σ
$z_1^{(2)}$
$\frac{\partial l}{\partial b_1}$
+
b_1
$z_1^{(1)}$
$\frac{\partial l}{\partial W_1}$
matmul
W_1
x
input

Neural architecture 2: Graph Neural Networks (GNN)



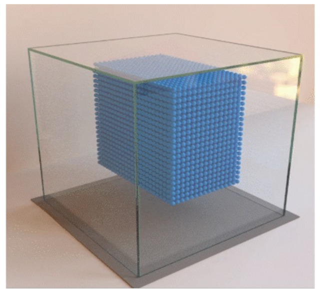
GNN f_θ $\xrightarrow{\hspace{6em}}$

Predictions on the node/edge with updated features

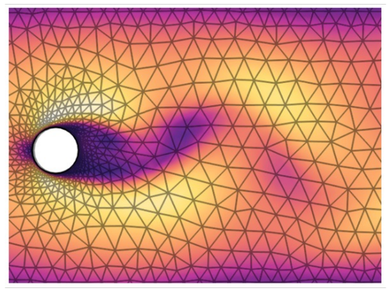


input graph $G = (V, E)$

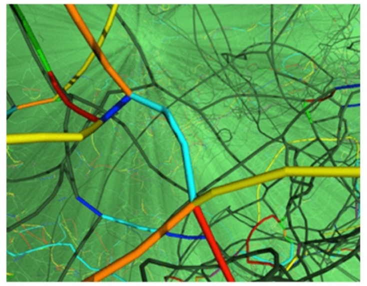
V: set of nodes with node features
E: set of edges with edge features



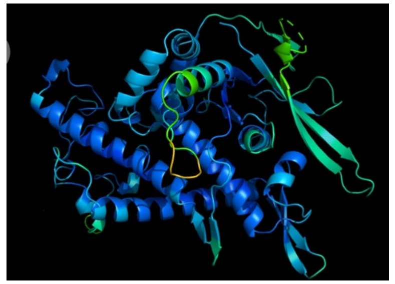
Fluid dynamics, computer graphics



Mesh-based simulation for PDEs

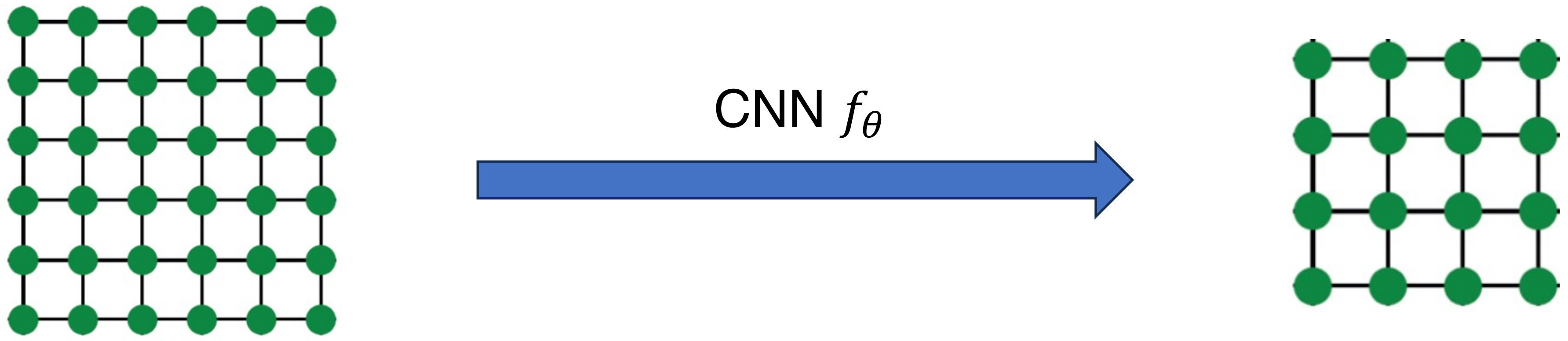


Dislocation in materials



Proteins and small molecules

Neural architecture 3: Convolutional Neural Networks (CNN)

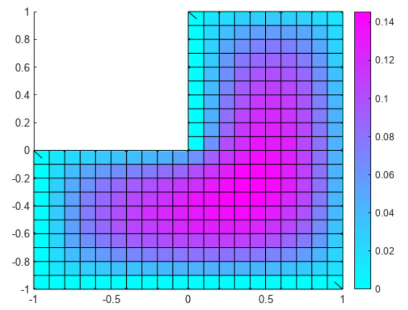


input tensor $X \in R^{d_1 \times d_2 \times \cdots \times d_n}$

output tensor $\hat{Y} \in R^{d'_1 \times d'_2 \times \cdots \times d'_n}$

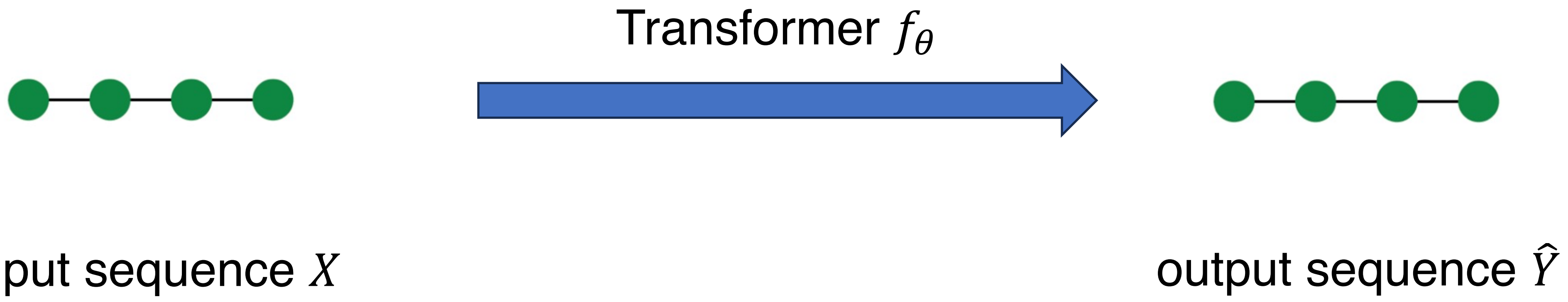
image

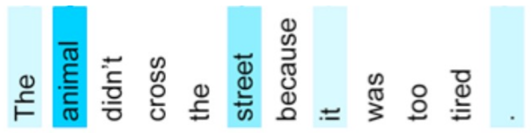
video



PDE discretized on a regular grid

Neural architecture 4: Transformer





Natural language

DNA sequence

Protein sequence

Neural architecture: Summary

Data structure	Suitable neural architecture	Course #
Vector	Multilayer Perceptron (MLP)	2 (Tailin Wu)
Graph	Graph Neural Network (GNN)	10 (Tailin Wu)
Grid	Convolutional Neural Network (CNN)	3 (Tailin Wu)
Sequence	Transformer	4 (Zhenzhong Lan)

For each neural architecture (same goes for topics in the course), we will introduce its:

- Motivation
- Architecture
- Typical tasks
- Research frontiers

Course introduction: Learning paradigm

Tasks

- Classification/ regression
- Simulation
- Inverse design/ inverse problem
- Control/planning

×

Neural architecture

- Multilayer perceptron
- Graph Neural Networks
- Convolutional Neural Networks
- Transformers

×

Learning paradigm

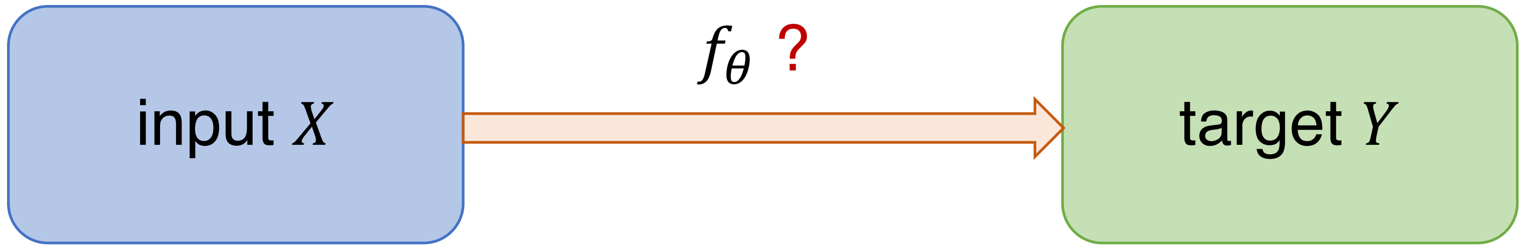
- Supervised learning
- Generative modeling
- Foundation models
- Reinforcement learning
- Evolutionary and multi-objective optimization
- In-context learning

Application (AI & Science)

- Robotics
- Games (e.g., Go, atari)
- Autonomous Driving
- PDEs
- Life science
- Materials science

Learning paradigm 1: Supervised learning

- image
- video
- graph
- time series
- natural language
- …



classification

- label (discrete)
- scalar/tensor (continuous)

regression

Given many examples of $(X, Y) = \{(x_i, y_i)\}_{i=1}^{N}$ pairs, learn a neural network (NN) f_θ that minimizes the prediction loss:

$$\theta^* = \text{argmin}_\theta \, \mathbb{E}_{(X,Y)\sim P(X,Y)}[\ell(f_\theta(X), Y)]$$

f_θ: neural network to be learned
ℓ: loss function

Learning paradigm 2: Generative modeling

Images and shapes generated by diffusion models:

By DallE 2

By MeshDiffusion [1]

[1] Liu, Zhen, et al. "Meshdiffusion: Score-based generative 3d mesh modeling." *ICLR 2023*

Learning paradigm 2: Generative modeling

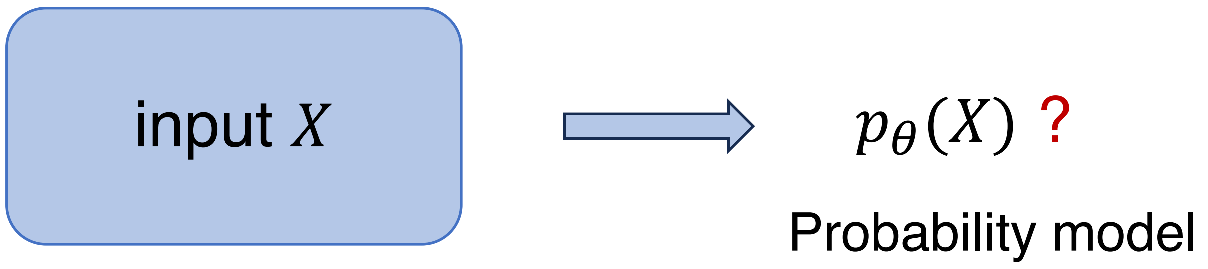
Robotic policy by diffusion models [1]

Text to video generation by Sora [2]

[1] Fu, Zipeng, Tony Z. Zhao, and Chelsea Finn. "Mobile ALOHA: Learning Bimanual Mobile Manipulation with Low-Cost Whole-Body Teleoperation." *arXiv preprint arXiv:2401.02117* (2024).
[2] OpenAI team. “Video generation models as world simulators”, 2024

Learning paradigm 2: Generative modeling

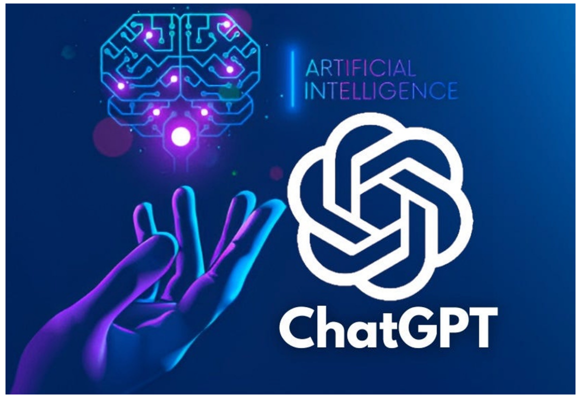
- image
- video
- graph
- time series
- natural language
- …



Given many examples of the input X, learn a probability model $p_\theta(X)$ that can **sample** new instances of X that conform to the data distribution

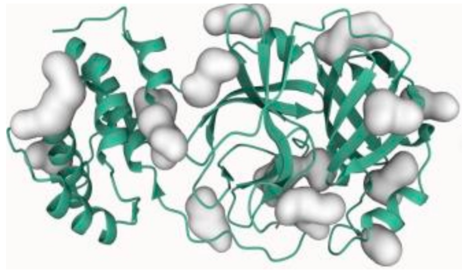
Major generative models:
- Diffusion models
- Flow
- Generative adversarial network (GAN)
- Variational autoencoder (VAE)

Learning paradigm 3: Foundation models



chatGPT

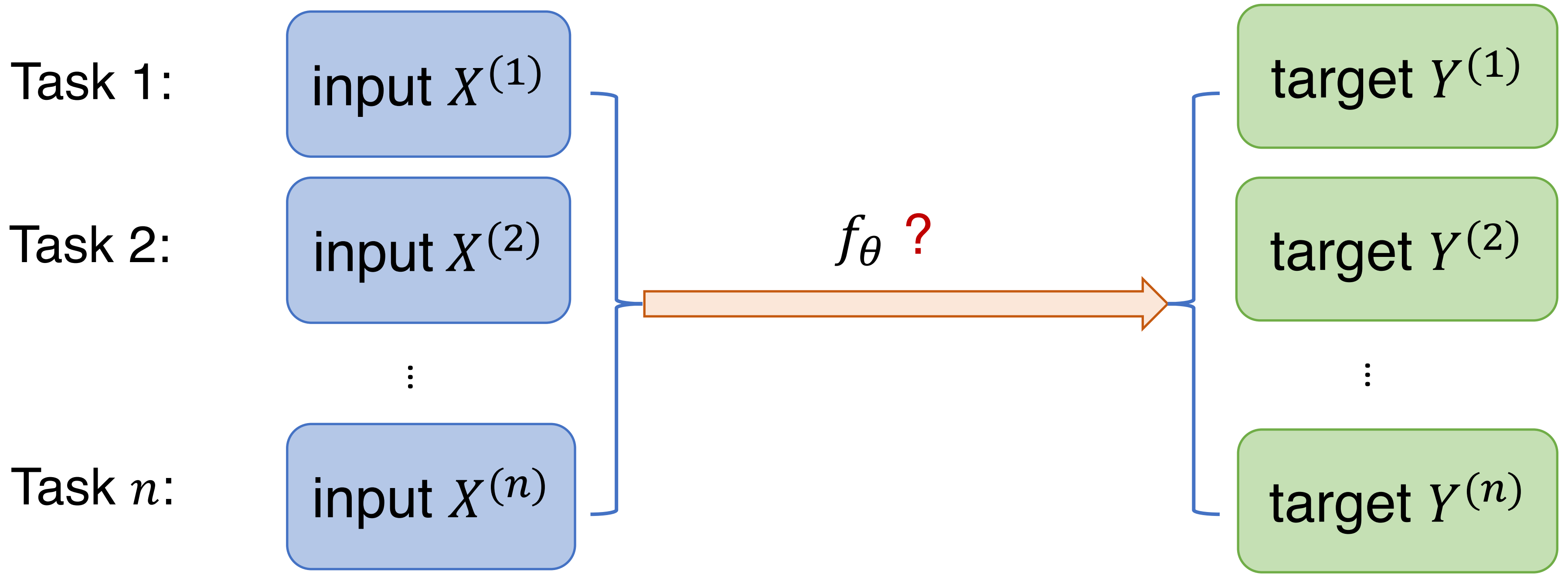
Sora [1]



uniMol [2]

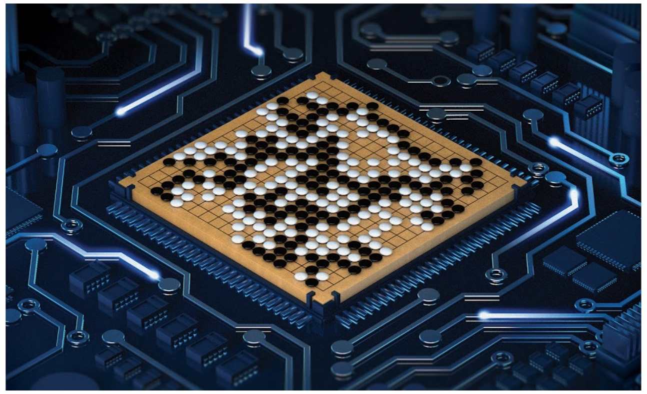
[1] OpenAI team. “Video generation models as world simulators”, 2024
[2] Zhou, Gengmo, et al. "Uni-Mol: a universal 3D molecular representation learning framework." ICLR 2023

Learning paradigm 3: Foundation models

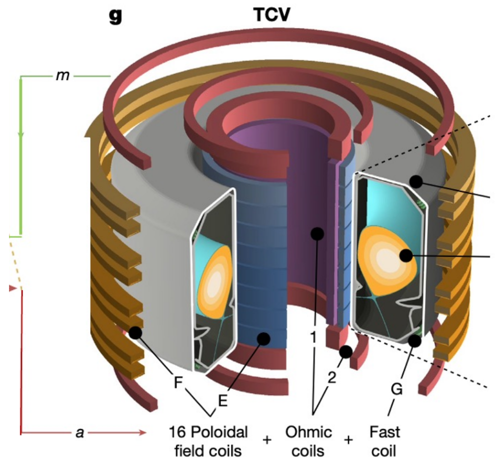


Given many diverse tasks, each consists of its massive number of examples $\left(X^{(n)}, Y^{(n)}\right) = \left\{\left(x_i^{(n)}, y_i^{(n)}\right)\right\}_{i=1}^{N^{(n)}}$, learn a single foundation model f_θ that can faithfully predict the target from the input.

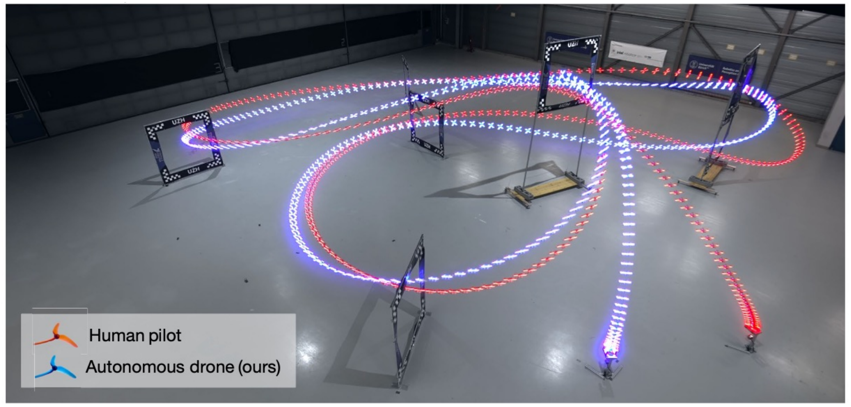
Learning paradigm 4: Reinforcement learning



AlphaGo [1]



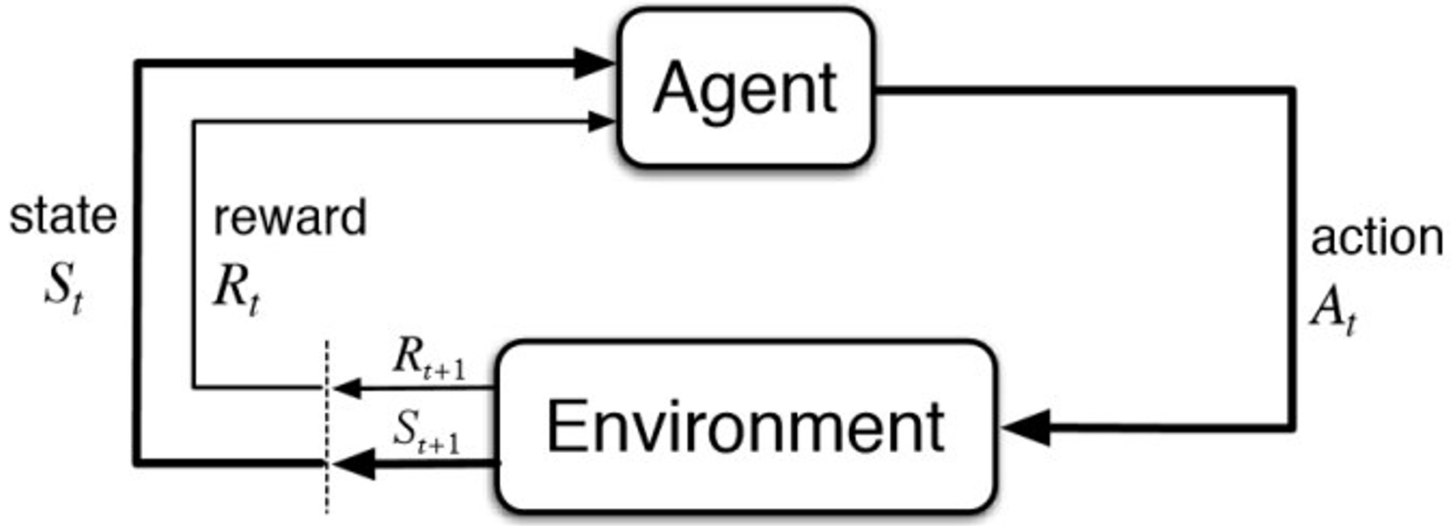
Controlled nuclear fusion [2]



Drone racing [3]

[1] Silver, David, et al. "Mastering the game of Go with deep neural networks and tree search." *Nature* 529.7587 (2016): 484-489.
[2] Degrave, Jonas, et al. "Magnetic control of tokamak plasmas through deep reinforcement learning." *Nature* 602.7897 (2022): 414-419.
[3] Kaufmann, Elia, et al. "Champion-level drone racing using deep reinforcement learning." *Nature* 620.7976 (2023): 982-987.

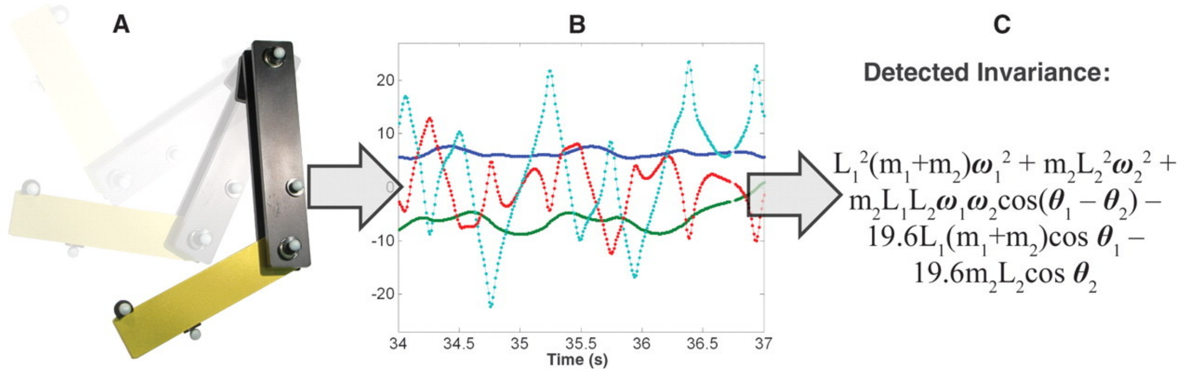
Learning paradigm 4: Reinforcement learning



Goal: maximize the long-term expected reward w.r.t. to the policy $\pi(A_t|S_t)$

$$\max_{\pi(A_t|S_t)} \mathbb{E}_t[R_t]$$

Learning paradigm 5: Evolutionary and multi-objective learning

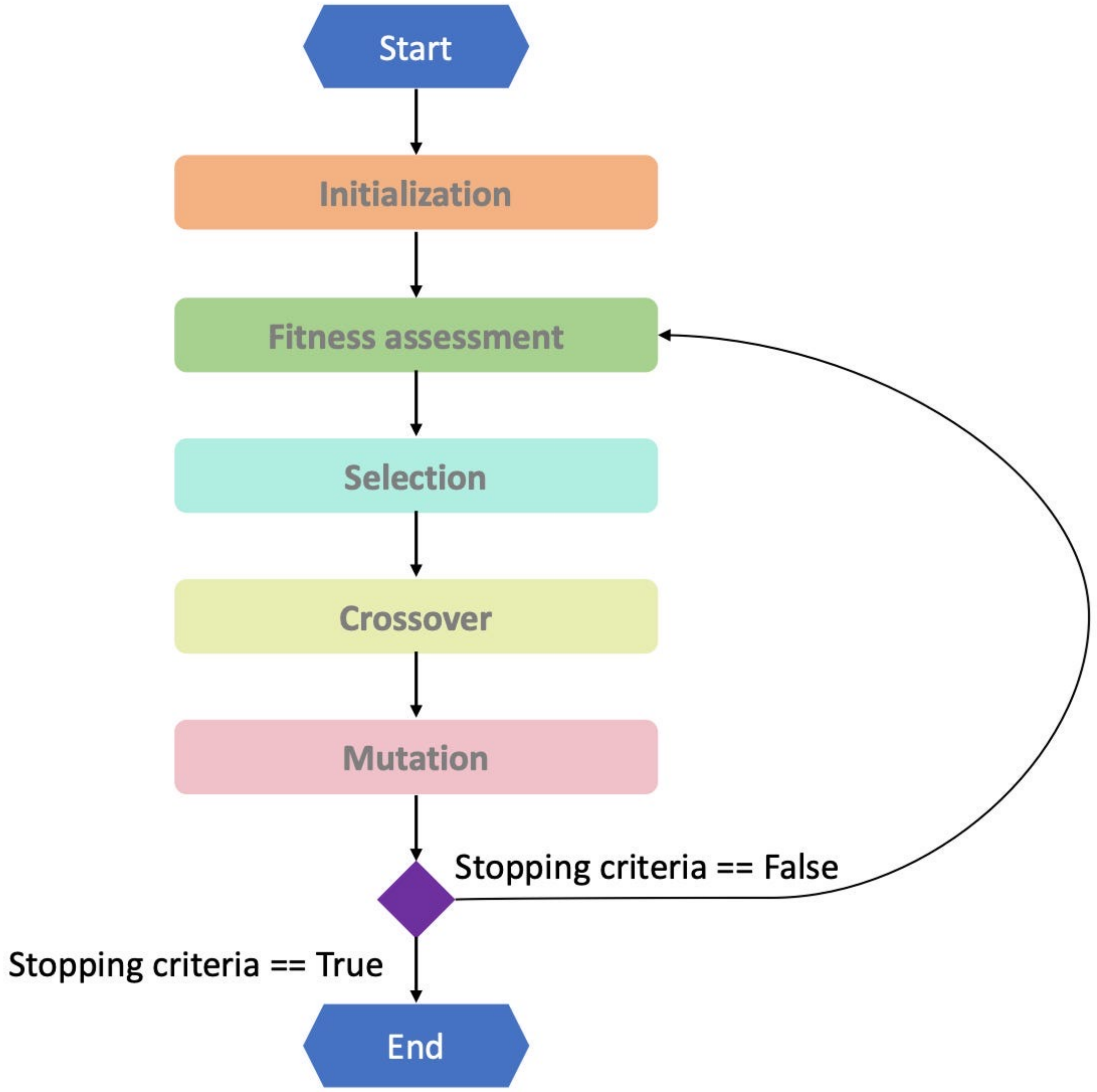

A
B
C
Detected Invariance:
$L_1^2(m_1+m_2)\omega_1^2 + m_2L_2^2\omega_2^2 + m_2L_1L_2\omega_1\omega_2\cos(\theta_1-\theta_2) - 19.6L_1(m_1+m_2)\cos\theta_1 - 19.6m_2L_2\cos\theta_2$
20
10
0
-10
-20
34
34.5
35
35.5
36
36.5
37
Time (s)

Discovering equations from data [1]

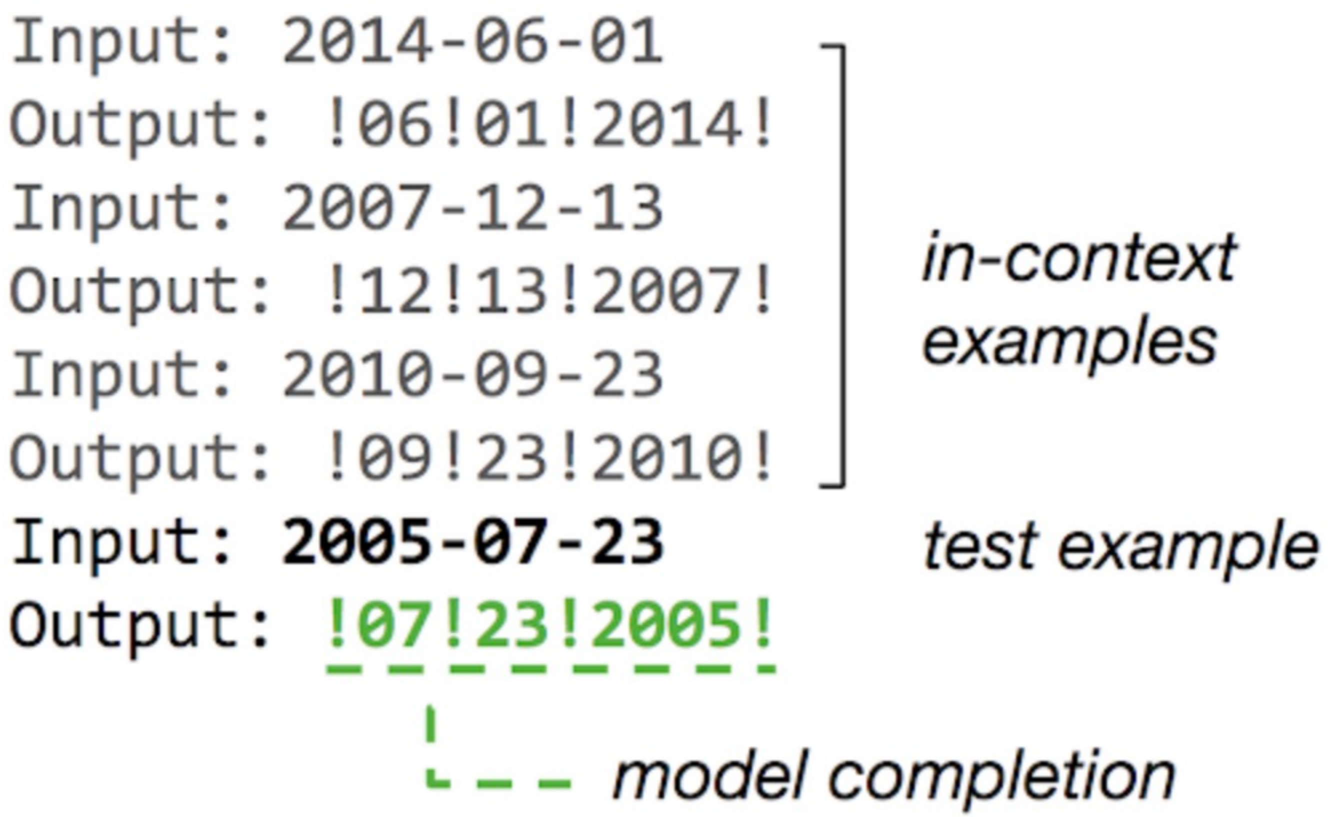
Optimization in engineering

[1] Schmidt, Michael, and Hod Lipson. "Distilling free-form natural laws from experimental data." *science* 324.5923 (2009): 81-85.

Learning paradigm 5: Evolutionary and multi-objective learning



Learning paradigm 6: In-context learning



```
Input: 2014-06-01
Output: !06!01!2014!
Input: 2007-12-13
Output: !12!13!2007!
Input: 2010-09-23
Output: !09!23!2010!
Input: 2005-07-23
Output: !07!23!2005!
```

Learning paradigm: Summary

Learning paradigm	Suitable scenarios	Course #
Supervised learning	Most standard	2 (Tailin Wu)
Generative modeling	High-dimensional data, can also be used in any tasks in regression	3 (Tailin Wu)
Foundation models	Large diverse tasks	4 (Zhenzhong Lan)
Reinforcement learning	Agent interacting with environment, cannot pass gradient through	6、9 (Tailin Wu)
Evolutionary and multi-objective learning	Gradient-free, discrete optimization	11 (Tailin Wu)
In-context learning	With language specifications	5 (Tailin Wu)

Course introduction: Application in AI and Science

Tasks × Neural architecture × Learning paradigm

Tasks

- Classification/ regression
- Simulation
- Inverse design/ inverse problem
- Control/planning

×

Neural architecture

- Multilayer perceptron
- Graph Neural Networks
- Convolutional Neural Networks
- Transformers

×

Learning paradigm

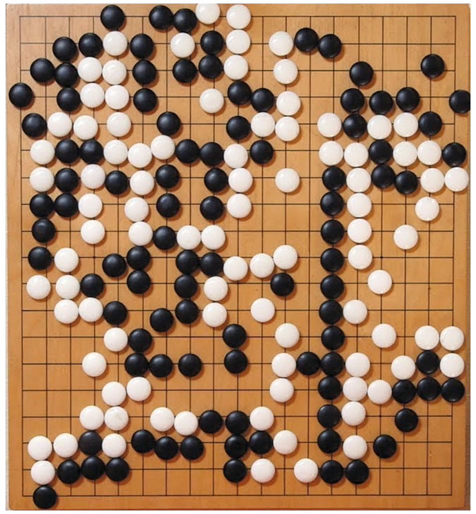
- Supervised learning
- Generative modeling
- Foundation models
- Reinforcement learning
- Evolutionary and multi-objective optimization

Application (AI & Science)

- Robotics
- Games (e.g., Go, atari)
- Autonomous Driving
- PDEs
- Life science
- Materials science

Application in AI

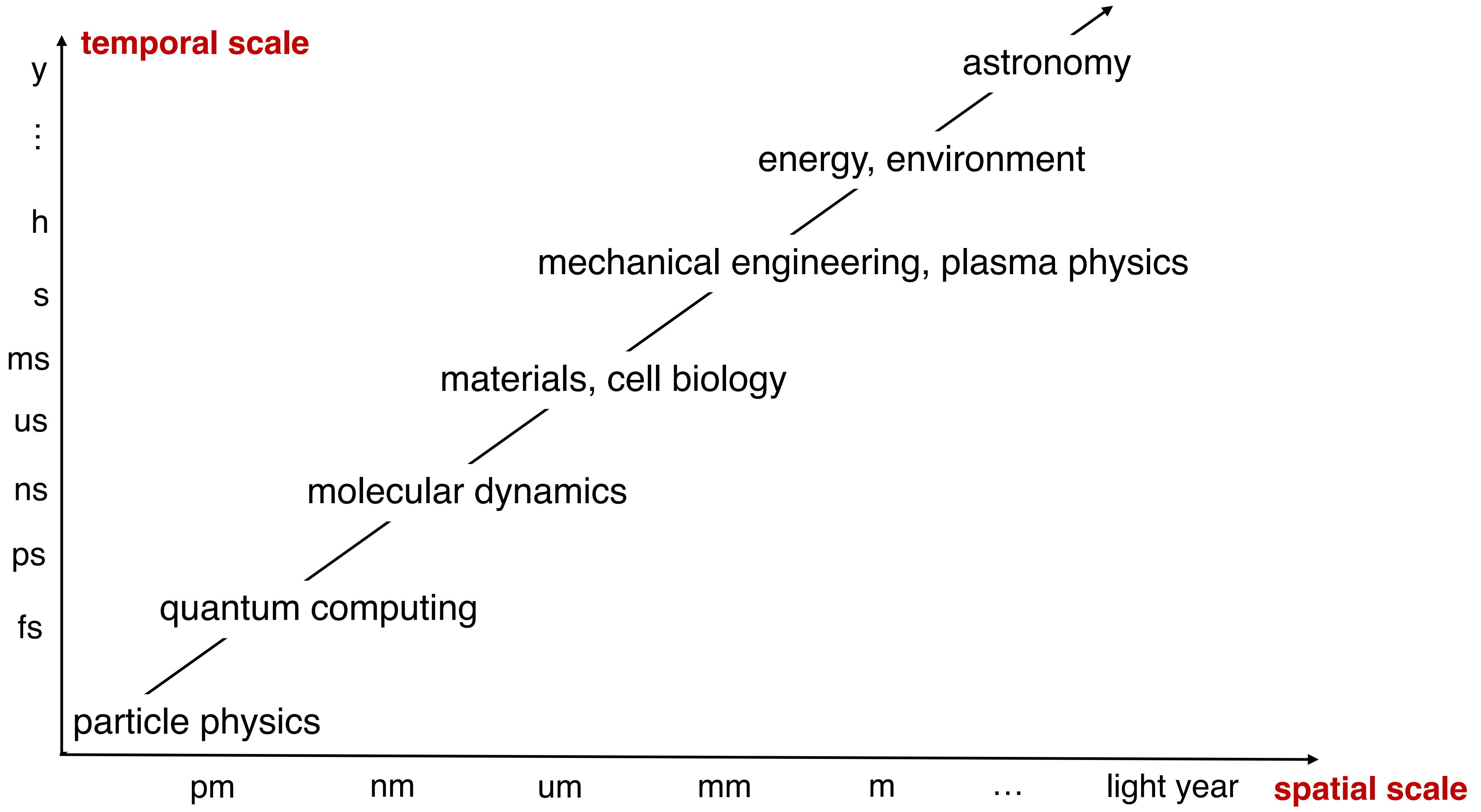
robotics



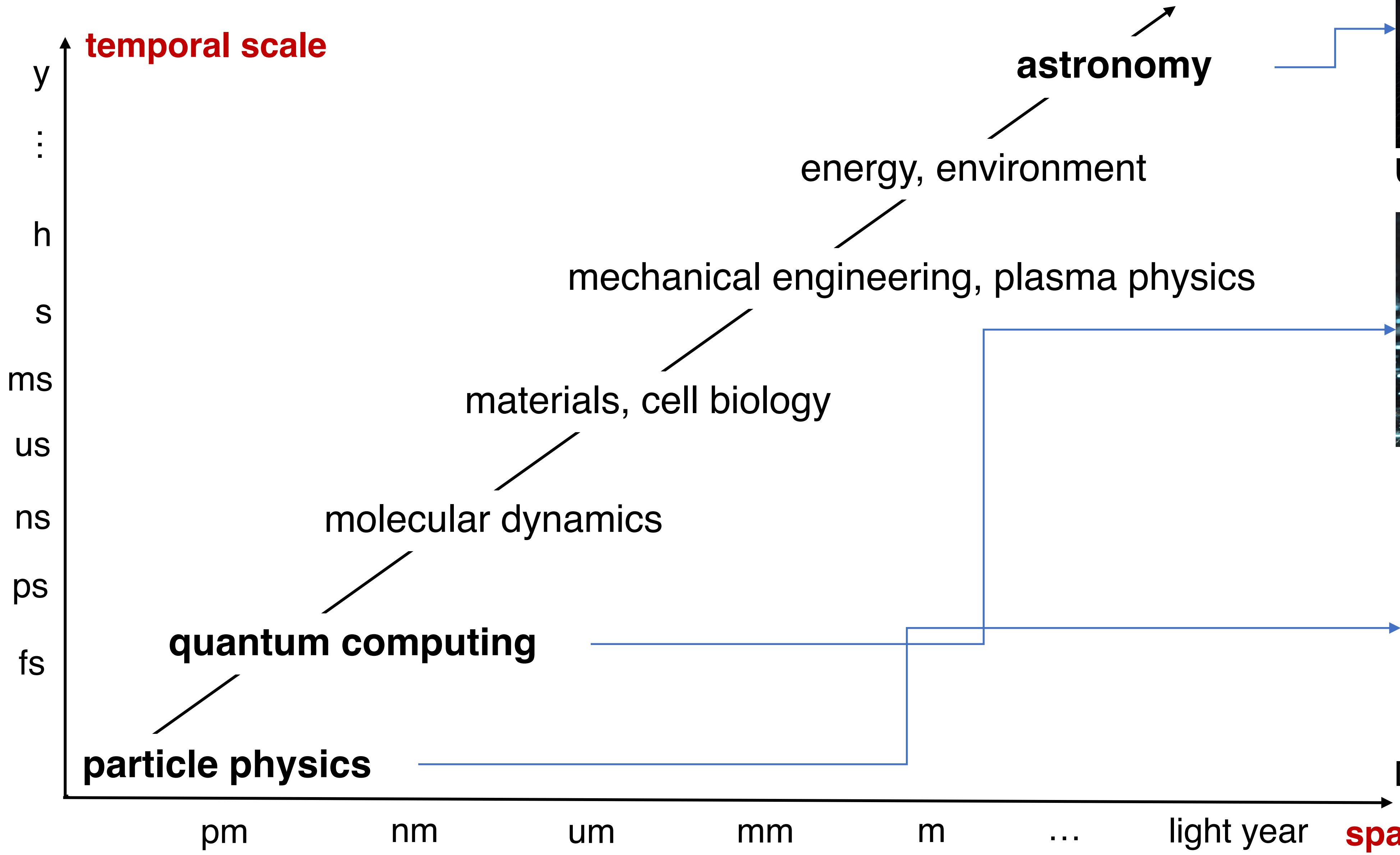
games

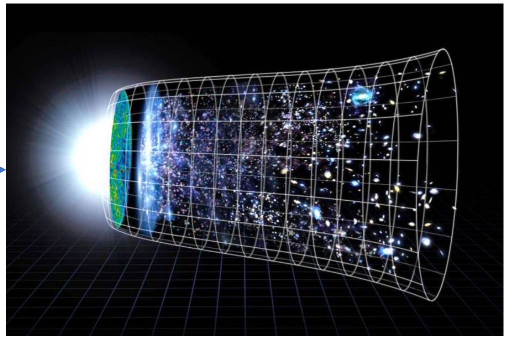
self-driving

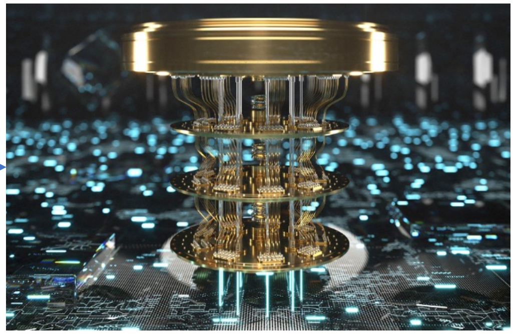
Application in AI for Science: from microscopic to macroscopic

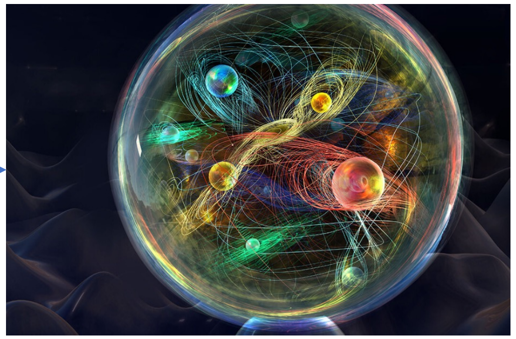


AI for Science: from microscopic to macroscopic

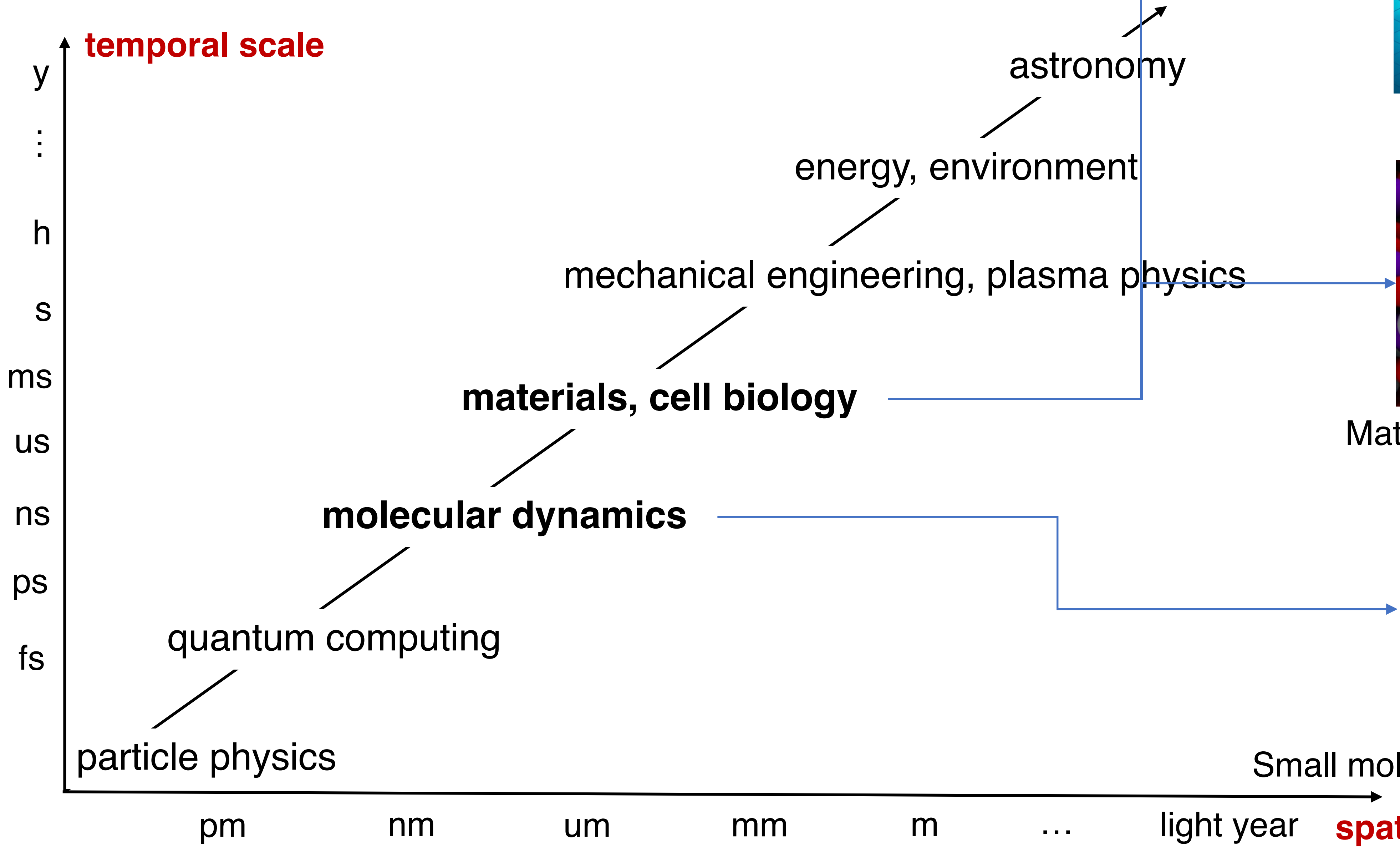



Understand our universe

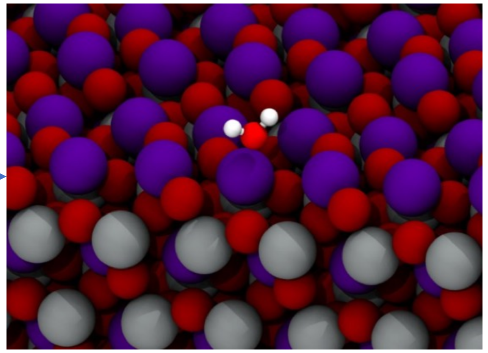

Speed up computation

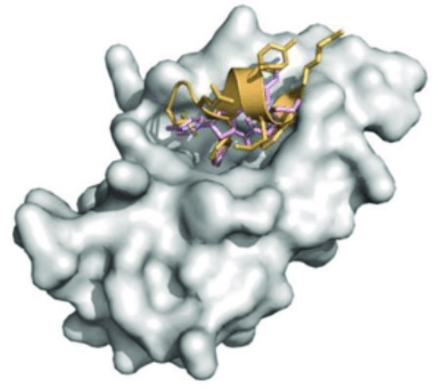

Discovering new physics

AI for Science: from microscopic to macroscopic

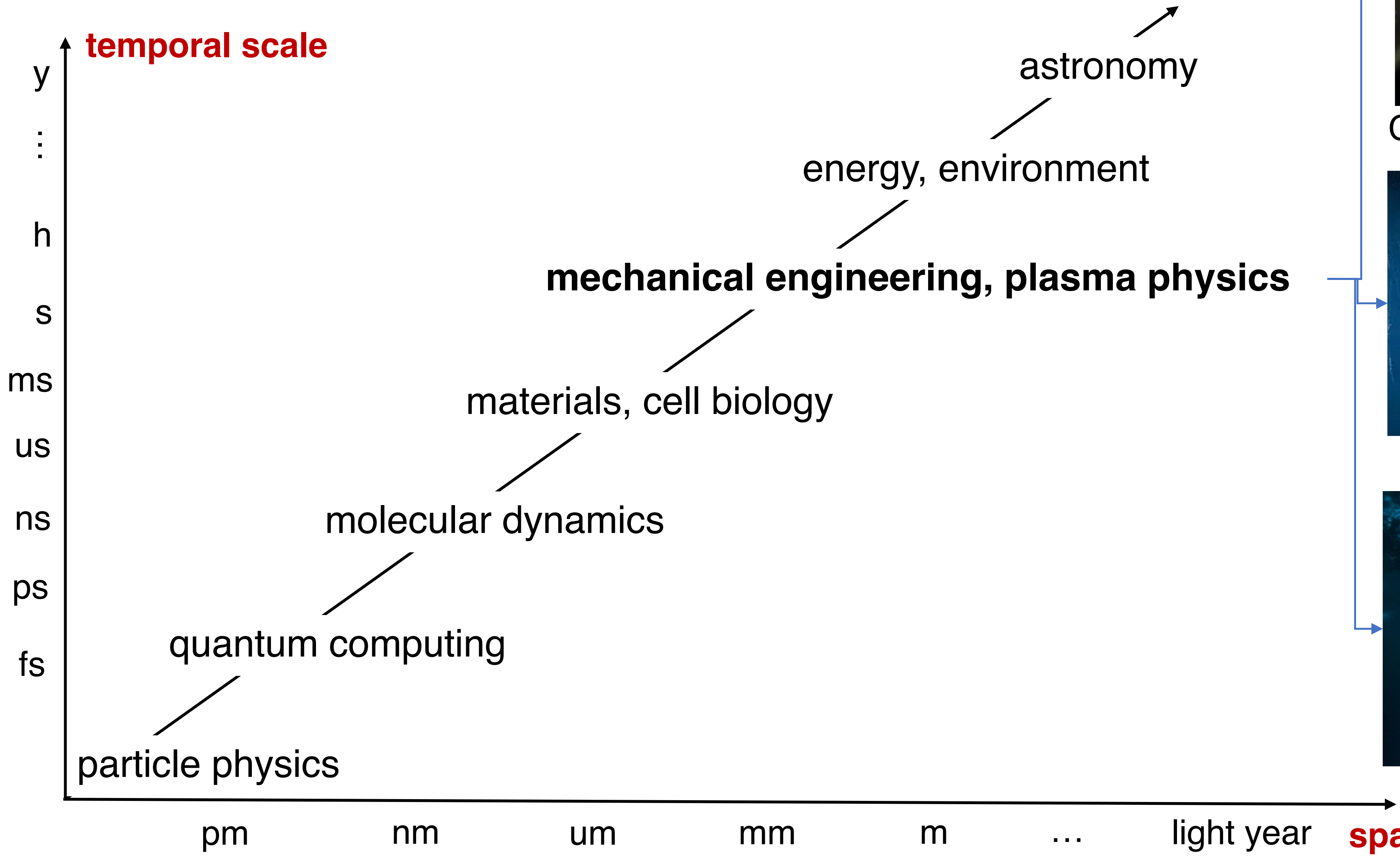


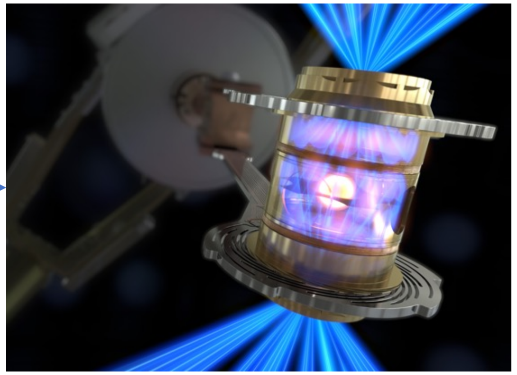
AI for genomics

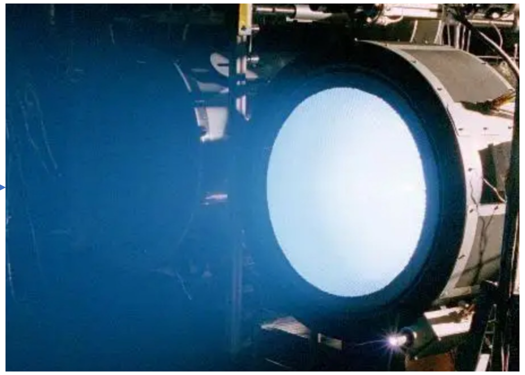

Material simulation & design

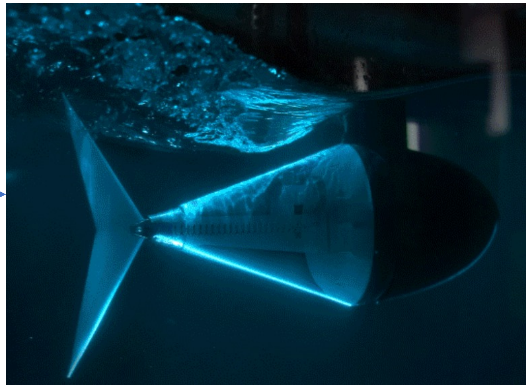

Small molecules & protein simulation

AI for Science: from microscopic to macroscopic

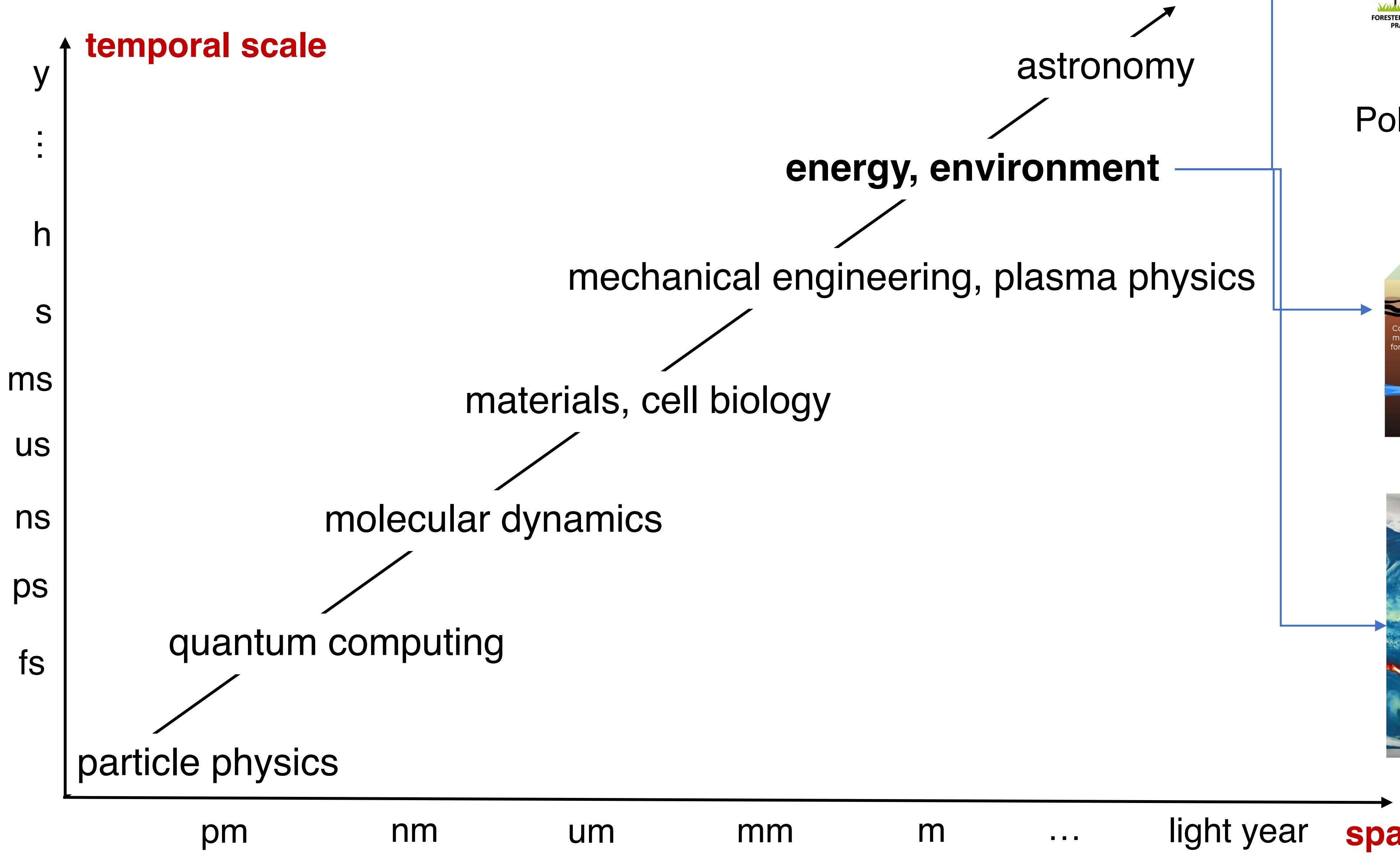


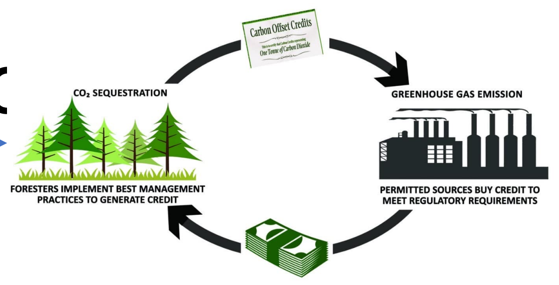

Controlled nuclear fusion


Ion thruster in space

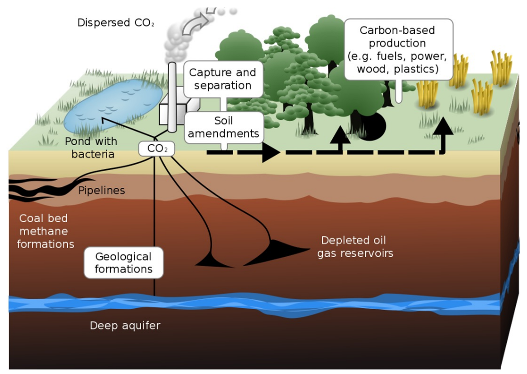

Underwater robots

AI for Science: from microscopic to macroscopic

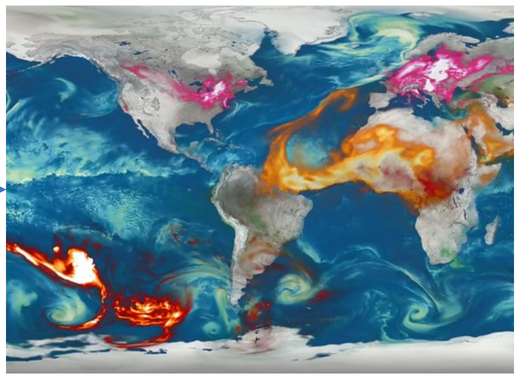



Carbon Offset Credits
CO₂ SEQUESTRATION
GREENHOUSE GAS EMISSION
FORESTERS IMPLEMENT BEST MANAGEMENT PRACTICES TO GENERATE CREDIT
PERMITTED SOURCES BUY CREDIT TO MEET REGULATORY REQUIREMENTS

Policy design for carbon credit

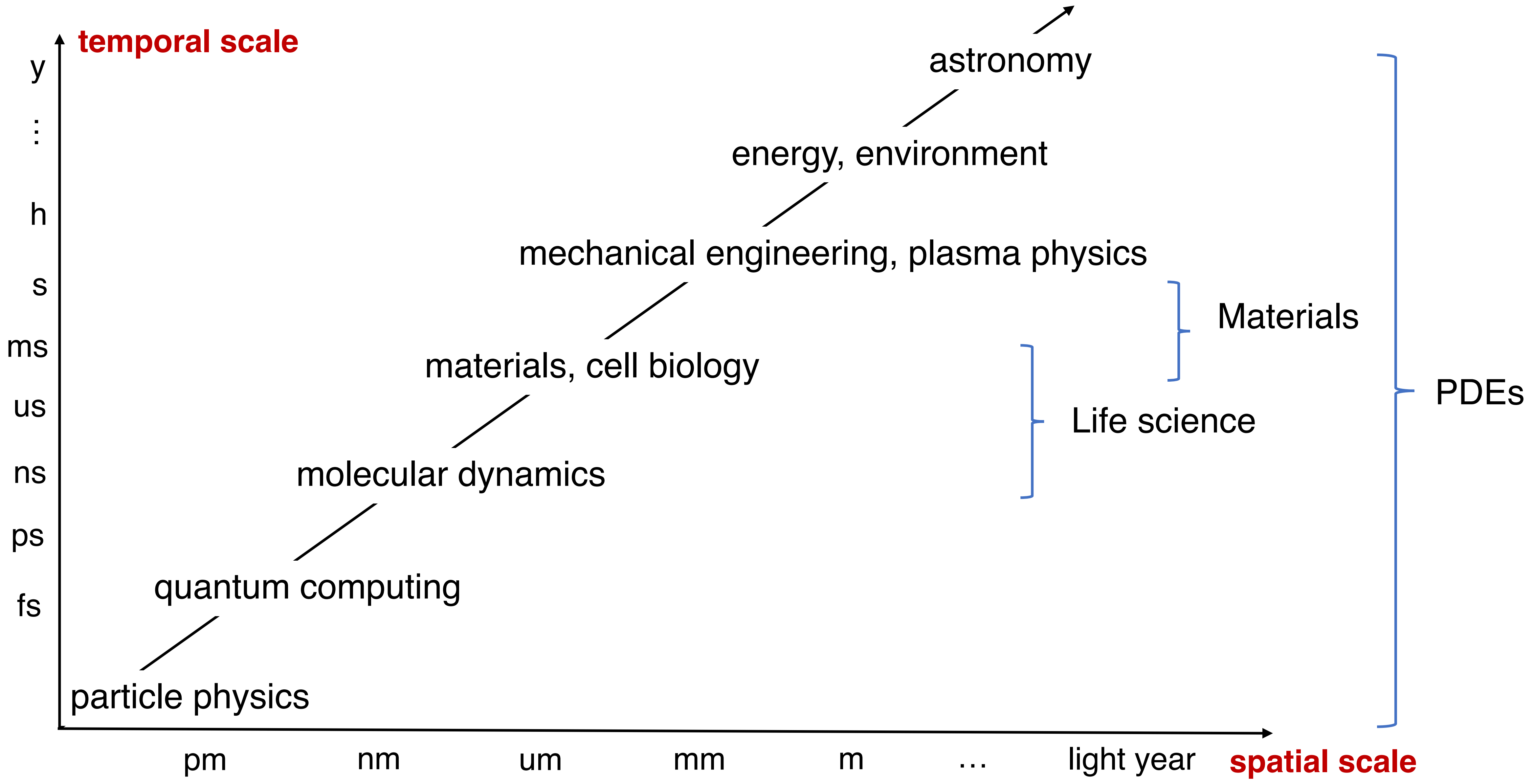

Dispersed CO₂
Capture and separation
Carbon-based production (e.g. fuels, power, wood, plastics)
Soil amendments
Pond with bacteria
CO₂
Pipelines
Coal bed methane formations
Geological formations
Depleted oil gas reservoirs
Deep aquifer

Carbon capture

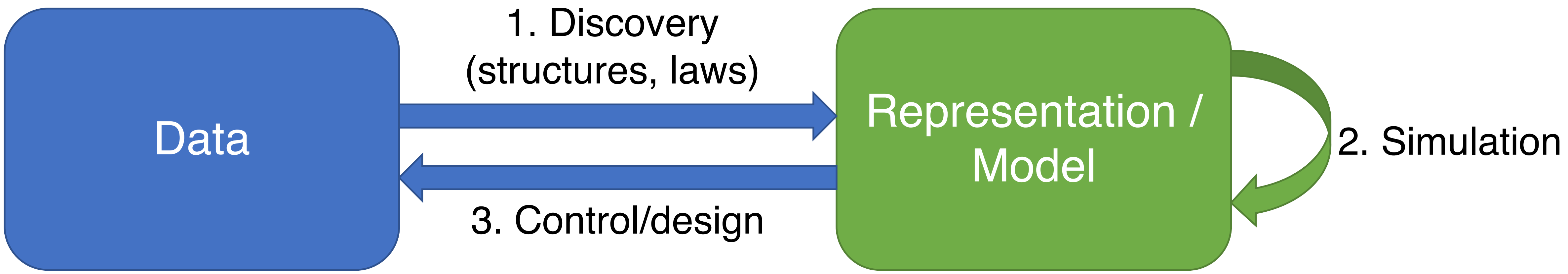


Weather forecasting

Application in AI for Science: from microscopic to macroscopic



AI for Science: universal tasks



These three tasks are fundamental in **science** and **engineering**

These three tasks are equally fundamental in **machine learning**

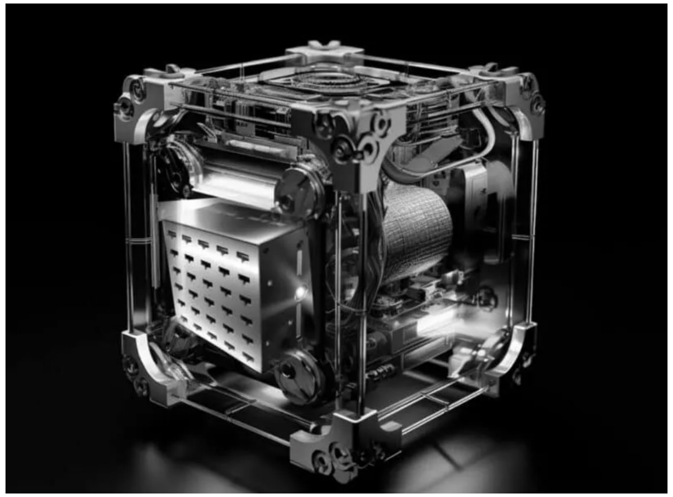
Application: Summary

Application	Area	Course #
Robotics	AI	6 (Tailin)
AI + PDEs/scientific computing	AI for science	13 (Tailin Wu)
AI + Life sciences	AI for science	12 (Tailin Wu)
Science for AI	Science for AI	14 (Tailin Wu)

Trend 1: Integration of technologies

- **Aerospace + Controlled nuclear fusion:** a spacecraft could theoretically reach 1/10 the speed of light (30,000 km/s) [1], which is 1,700 times the current maximum spacecraft speed of 17 km/s.

RocketStar is developing the FireStar Drive thruster based on controlled nuclear fusion technology.

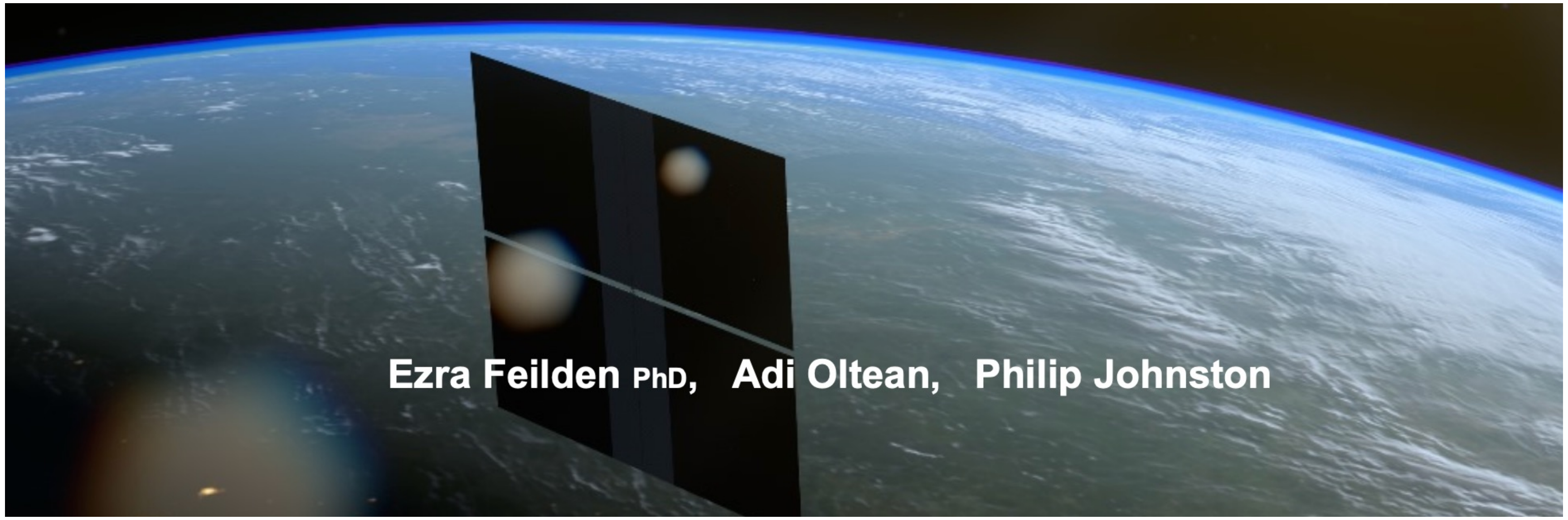


[1] 吴从军，火箭推进速度上限的估算，中国物理学会期刊网

Trend 1: Integration of technologies

- **Aerospace + GPU + Embodied AI:** Build GPU cluster in space

Lumen Orbit is planning to build the first GPU cluster in space [1].



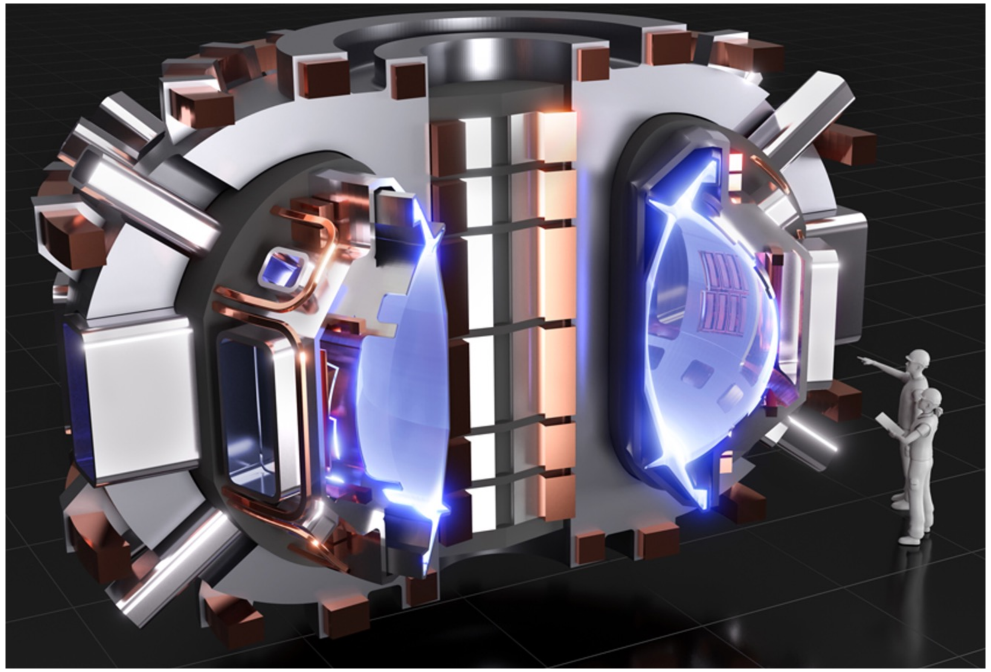
[1] Lumen Orbit et al., Why we should train AI in space, 2024

Trend 1: Integration of technologies

- **Aerospace + Mechanical engineering:** Gravity Jet Suit by gravity industries

Trend 1: Integration of technologies

- **Controlled nuclear fusion + materials:** The realization of high-temperature superconductors or room-temperature superconductors could significantly reduce the size of tokamaks.

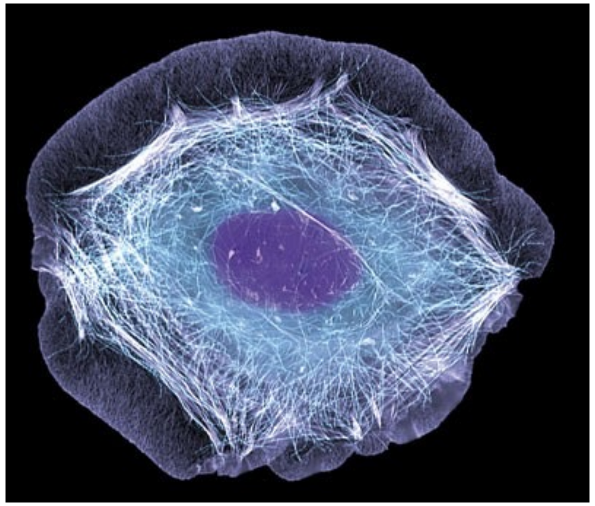


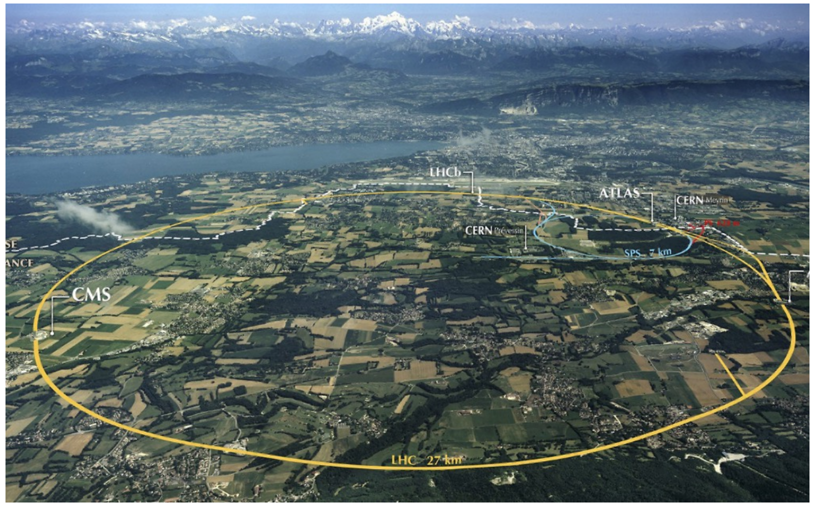
(image from Science)

Trend 2: Improvement of measurement technology

- **Better science needs better measurement**
  - Astronomy observation: New satellites and more accurate telescopes
  - Life science observation: Better single cell measurements
  - Particle collision measurement: Higher energy and better reconstruction

New telescopes


Single cell measurement

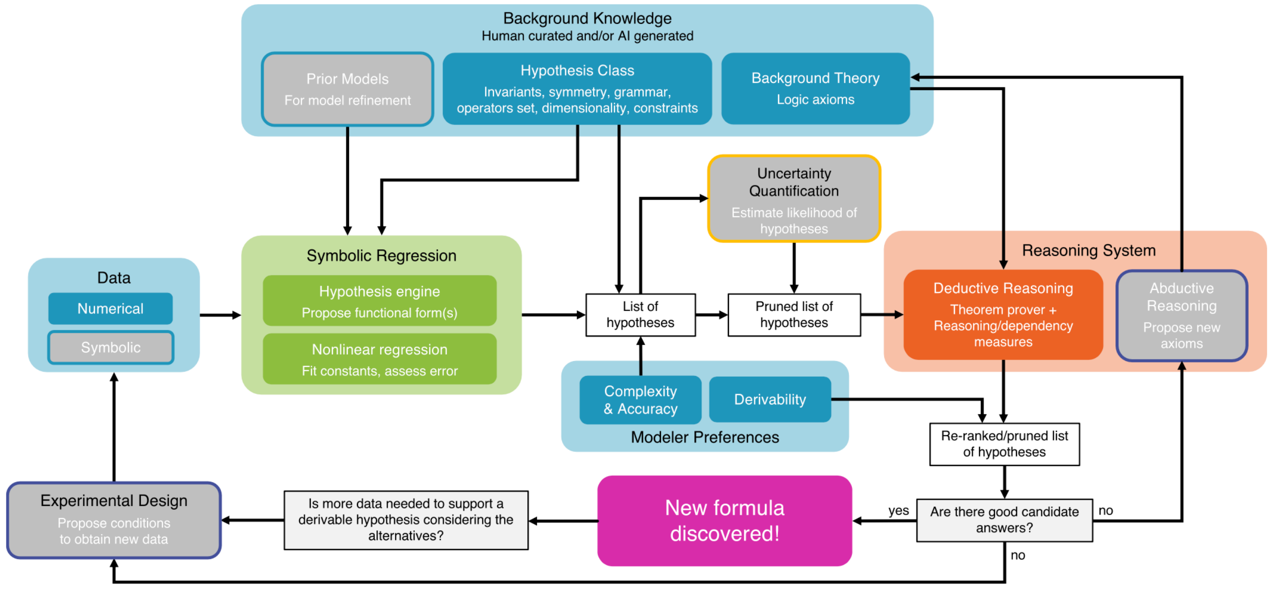


Large Hadron Collider (LHC)

Trend 3: Foundation models

[1]Cornelio, Cristina, et al. "Combining data and theory for derivable scientific discovery with AI-Descartes." *Nature Communications* 14.1 (2023): 1777.

- **AI Scientist** for *automatic* and *high-throughput* experiments
- **Foundation model** for solving diverse problems with a single model



Background Knowledge
Human curated and/or AI generated
Prior Models
For model refinement
Hypothesis Class
Invariants, symmetry, grammar, operators set, dimensionality, constraints
Background Theory
Logic axioms
Uncertainty Quantification
Estimate likelihood of hypotheses
Data
Numerical
Symbolic
Symbolic Regression
Hypothesis engine
Propose functional form(s)
Nonlinear regression
Fit constants, assess error
List of hypotheses
Pruned list of hypotheses
Reasoning System
Deductive Reasoning
Theorem prover + Reasoning/dependency measures
Abductive Reasoning
Propose new axioms
Complexity & Accuracy
Derivability
Modeler Preferences
Re-ranked/pruned list of hypotheses
Experimental Design
Propose conditions to obtain new data
Is more data needed to support a derivable hypothesis considering the alternatives?
New formula discovered!
yes
Are there good candidate answers?
no
no

AI-Descartes pipeline [1]

Foundation models

Course introduction: Summary

Tasks

- Classification/ regression
- Simulation
- Inverse design/ inverse problem
- Control/planning

×

Neural architecture

- Multilayer perceptron
- Graph Neural Networks
- Convolutional Neural Networks
- Transformers

×

Learning paradigm

- Supervised learning
- Generative modeling
- Foundation models
- Reinforcement learning
- Evolutionary and multi-objective optimization

Application (AI & Science)

- Robotics
- Games (e.g., Go, atari)
- Autonomous Driving
- PDEs
- Life science
- Materials science

Useful materials to get started in deep learning

- Learn PyTorch in 1h: https://pytorch.org/tutorials/beginner/introyt/introyt1_tutorial.html
- Book: Deep Learning book (https://www.deeplearningbook.org/)

Hope you all learn useful techniques to help your research!

Questions?