

The History of Intelligence and Mathematical Principles of Deep Learning

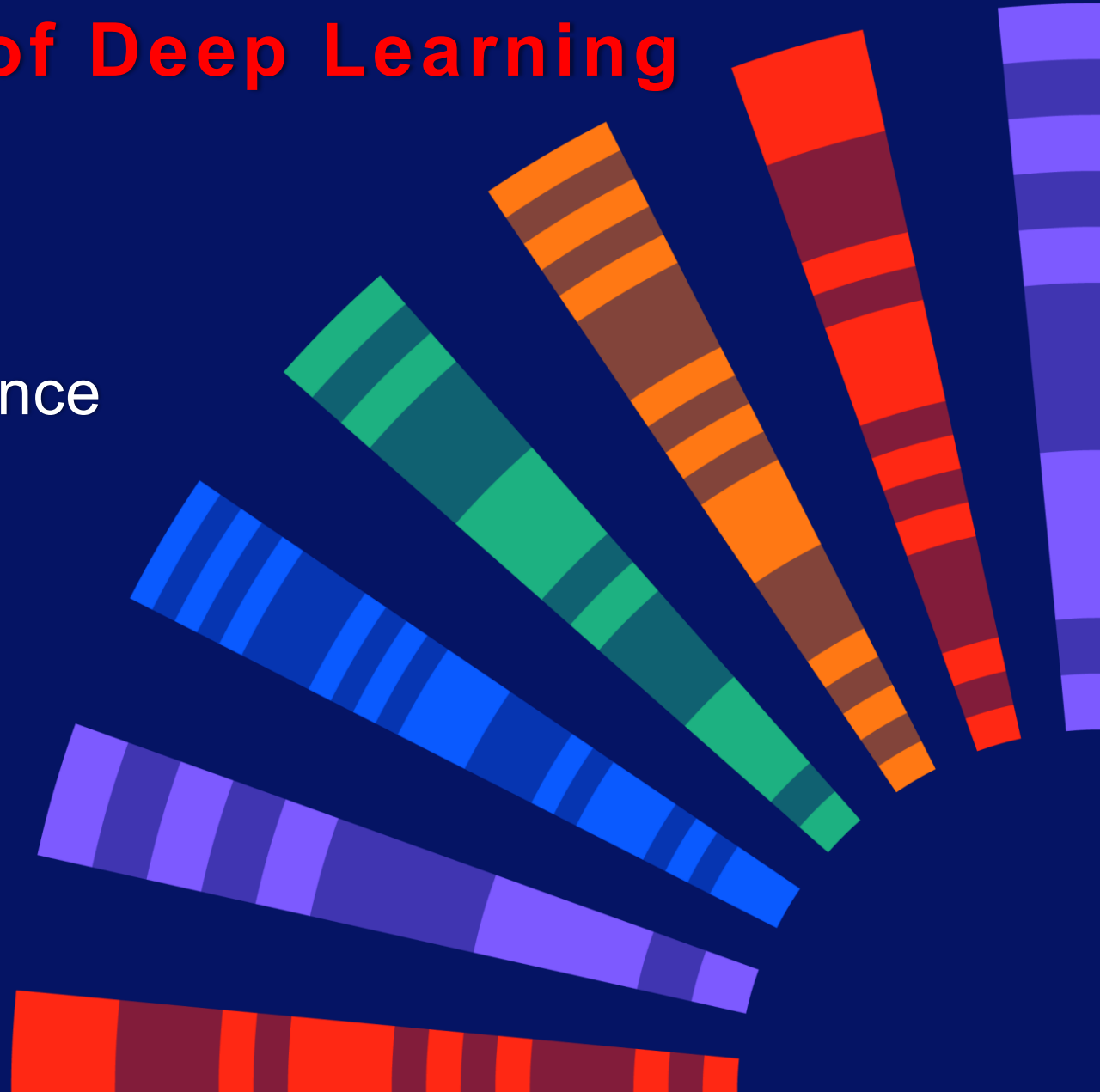
Professor Yi Ma

School of Computing and Data Science

The University of Hong Kong



SCHOOL OF
**COMPUTING &
DATA SCIENCE**
The University of Hong Kong



Prologue

Seek a scientific and theoretical foundation for **Intelligence**:

- what to learn?
- how to learn?
- why correct?

“What I cannot create, I do not understand.”

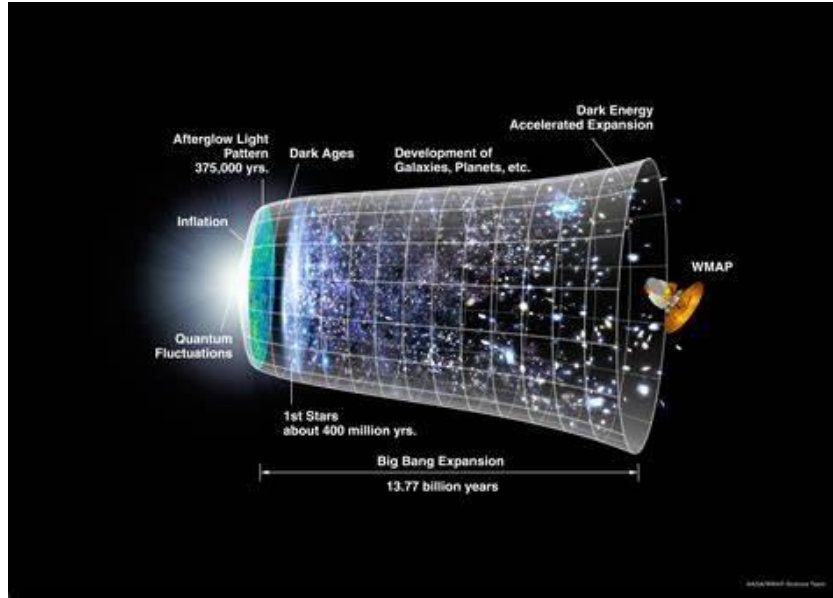
-- Richard Feynman

知致物格



Evolution of Life and Intelligence

Evolution of the Universe is **Physics** at work



Evolution of Life is **Intelligence** at work



*“Just as the constant **increase of entropy** is the basic law of the universe, so it is the basic law of life to be ever more highly structured and to struggle **against entropy**.”*

-- Vaclav Havel

Evolution of Life and Intelligence: From DNA to Brain

From the first DNA to the emergence of life with Brain: **3.6 Billion Years**
From the first Brain to the explosion of lives in the Cambrian period: **50 Million Years**

DNAs: Pretrained “Large Models”
Random Mutation & Natural Selection
Reinforcement Learning

Emergence of Brain & Senses
Individual Models: Memory
Learn from Feedback

Explosion of Lives
The Cambrian period

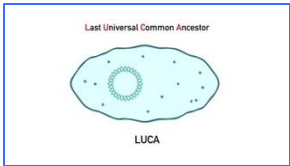
DNA appeared near
some volcano in the
ocean



40B Years

First DNA

Common ancestor
of all lives: **LUCA**



3.5B Years

First Life Form

First life form with a brain
(**Nematode**)



550M Years

First Brain



500M Years

Cambrian Period

Evolution of Life and Intelligence: From Species to Individuals

Emergence and evolution of life are mechanisms of intelligence at work!

Life depends on intelligence to continuously acquire more knowledge to better predict the world.

Phylogenetic Intelligence: DNA inheritance, random mutation, and natural selection

Ontogenetic Intelligence: Individual memory, perception & feedback, and error correction.



3.7B years ago
Life begins

500M years ago
Cambrian period

400M years ago
fish

360M years ago
amphibian

250M years ago
reptile

200M years ago
bird and mammal

310T years ago
neanderthal

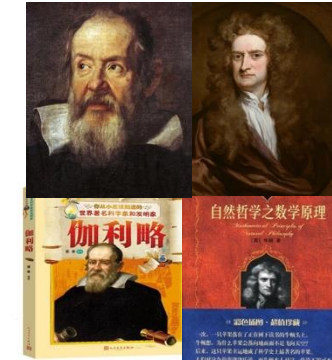
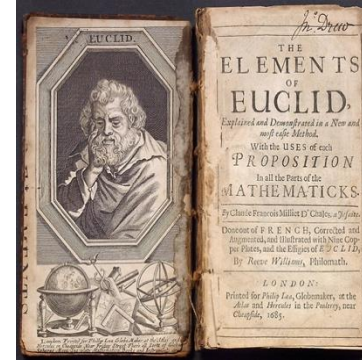
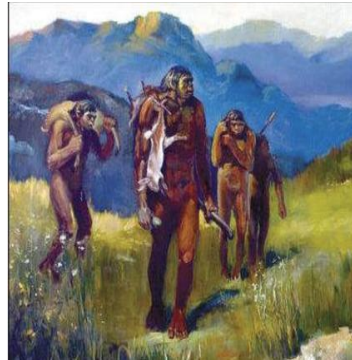
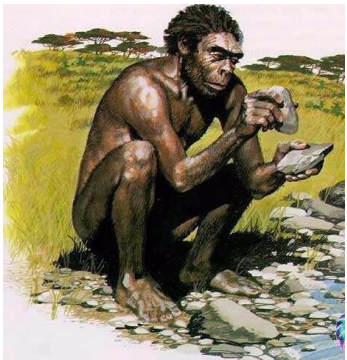
Evolution of Intelligence: From Societal to Artificial Intelligence

Emergence and evolution of life are mechanisms of intelligence at work!

Life depends on intelligence to continuously acquire more knowledge to better predict the world.

Societal Intelligence: Languages and texts, empirical knowledge, trial and error

Artificial Intelligence: Scientific facts, theorize, hypothesis testing & falsification.



310T years ago
neanderthal

Tools and
group hunting

70T years ago
Societal intelligence

Languages
Information sharing

3500 BC
written language

Knowledge
accumulation

600-300 BC
Artificial Intelligence

Abstraction, formal
logic, and mathematics

14-18th Century
Renaissance

Science
Hypothesis Testing

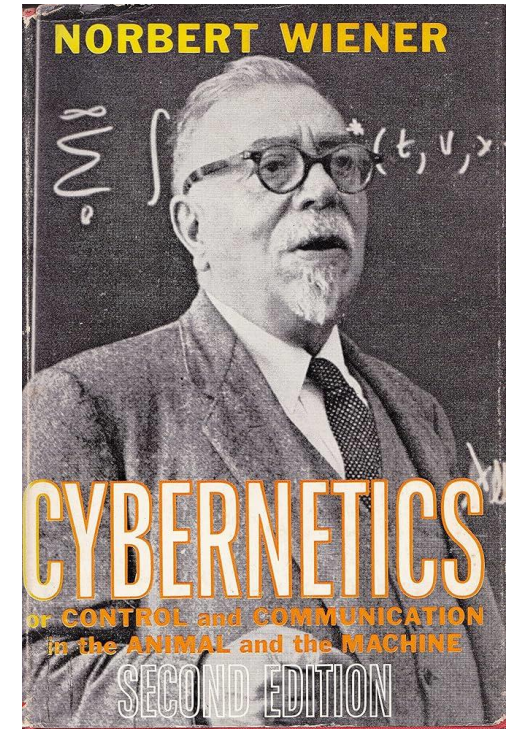
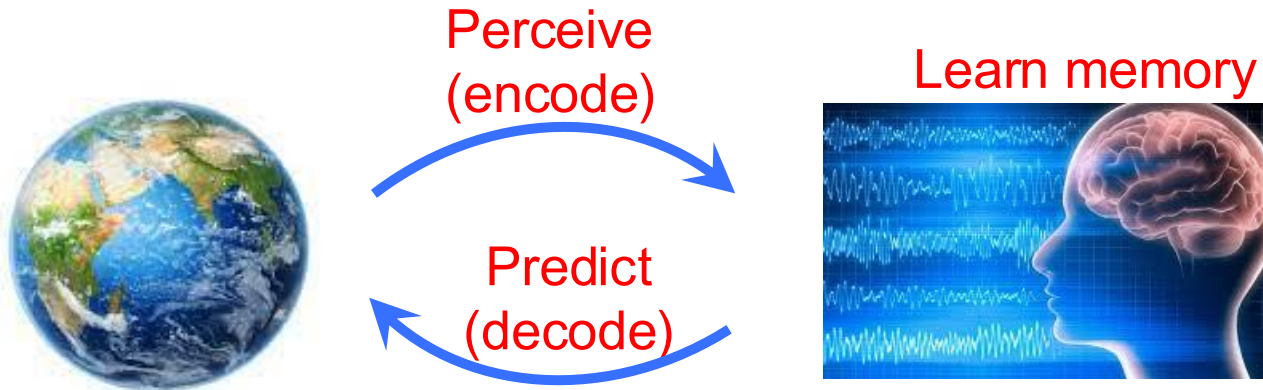
The 1940s
Machine Intelligence

Computing machines

True Origin of Machine Intelligence (the magic era!)

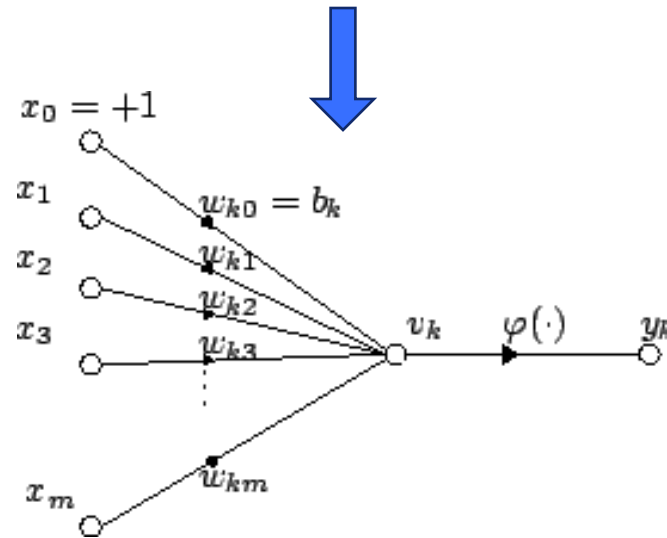
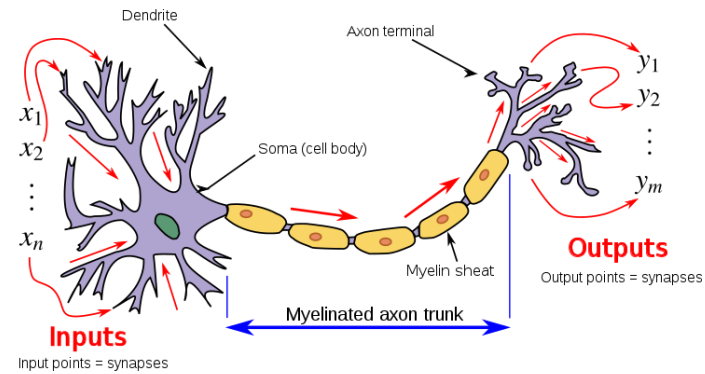
1940s, people started to make machines **imitate intelligence** (of animals).

- 1948, **Cybernetics & System Theory**, **Nobert Wiener**
- 1943, **Artificial Neural Networks**, Warren McCulloch and Walter Pitts
- 1948, **Information Theory**, Claude Shannon
- 1944, **Game Theory**, John von Neumann
- 1940's, **Turing Machine and Turing Test**, Alan Turing



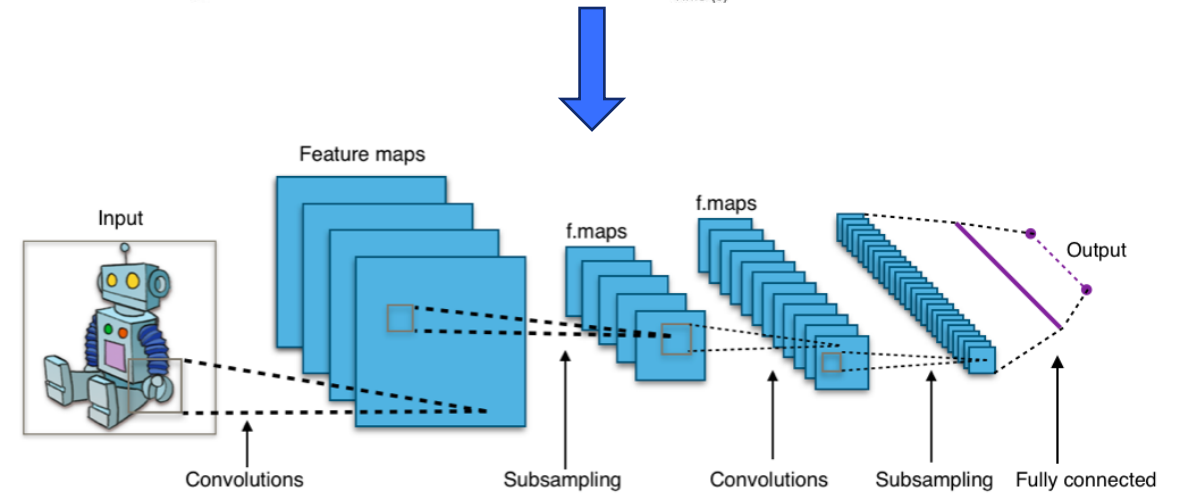
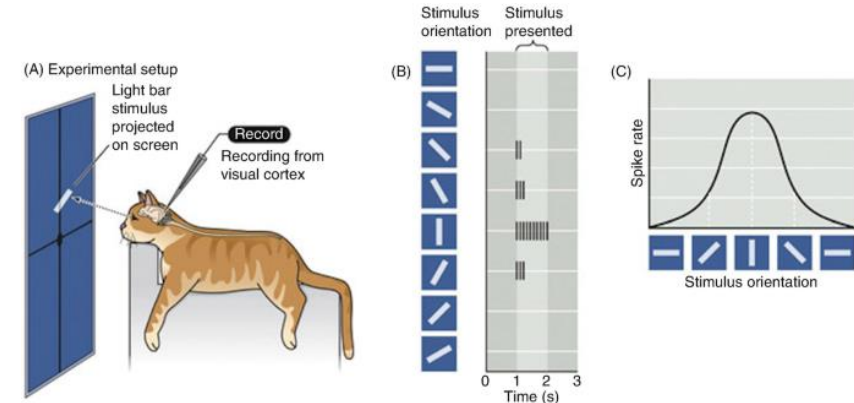
Artificial Neurons and Neural Networks: Learn from Nature

Golgi and Cajal 1888 (1901 Nobel Prize)



Warren McCulloch & Walter Pitts 1948

Hubel and Wiesel 1959 (1981 Nobel Prize)



Fukushima 1980 & LeCun 1989 (Turing Award)

History of Machine Intelligence (Artificial Neural Networks)

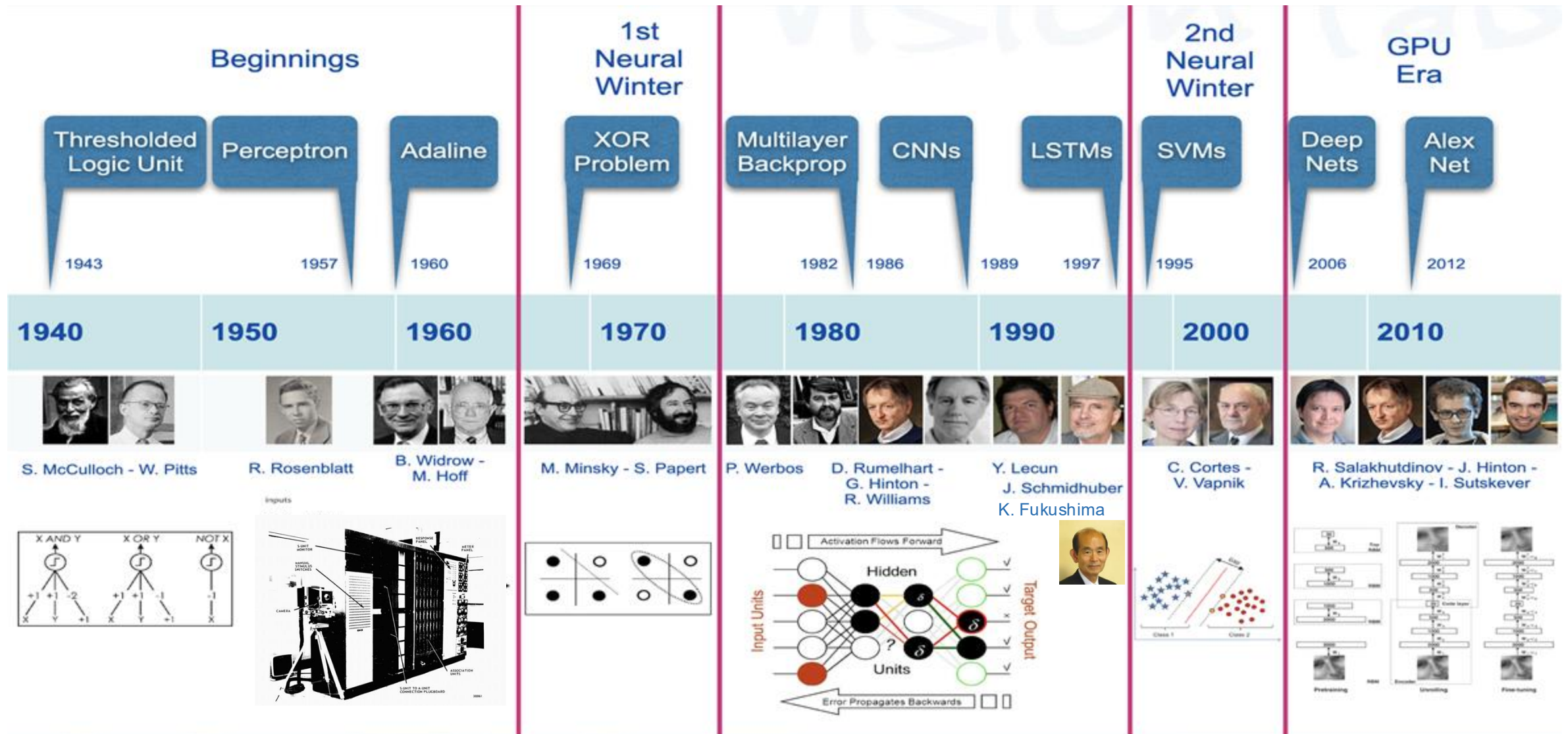
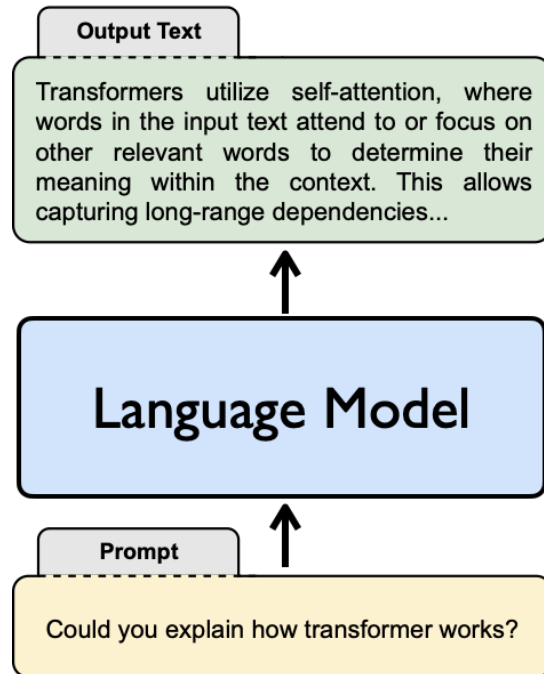


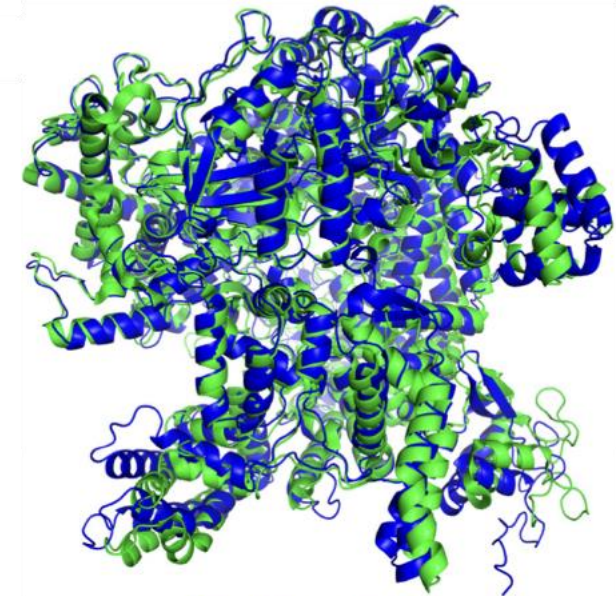
Figure courtesy of Professor [Rene Vidal](#)

Modern Evolution of Deep Neural Networks

Deep
Neural
Nets



(2024 Nobel Prize Chemistry)



AlphaFold Experiment
r.m.s.d.₉₅ = 2.2 Å; TM-score = 0.96

Why Must Turn Blackbox to Whitebox?

- Modern AI systems all based on empirically designed deep networks (alchemy?)
- Blackbox is difficult to explain, impossible to guarantee, costly to improve, ...

It is high time to develop a principled approach!



What to Learn?
(**Parsimony**)

- Learn what is predictable
- Low-dim structures
- Information gain

How to Learn?
(**Compression**)

- Unroll iterative optimization
- Iterative compressing
- Make DNN a whitebox

Why Correct?
(**Consistency**)

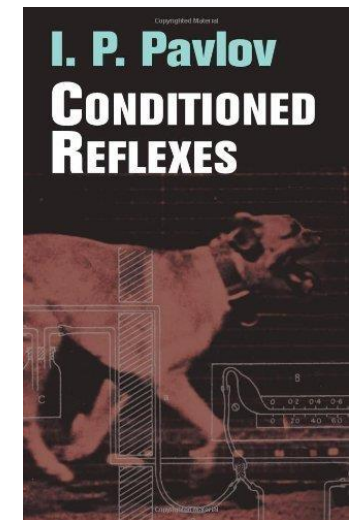
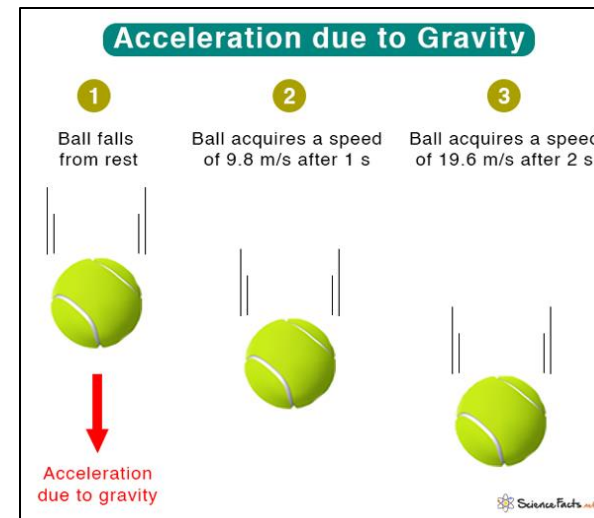
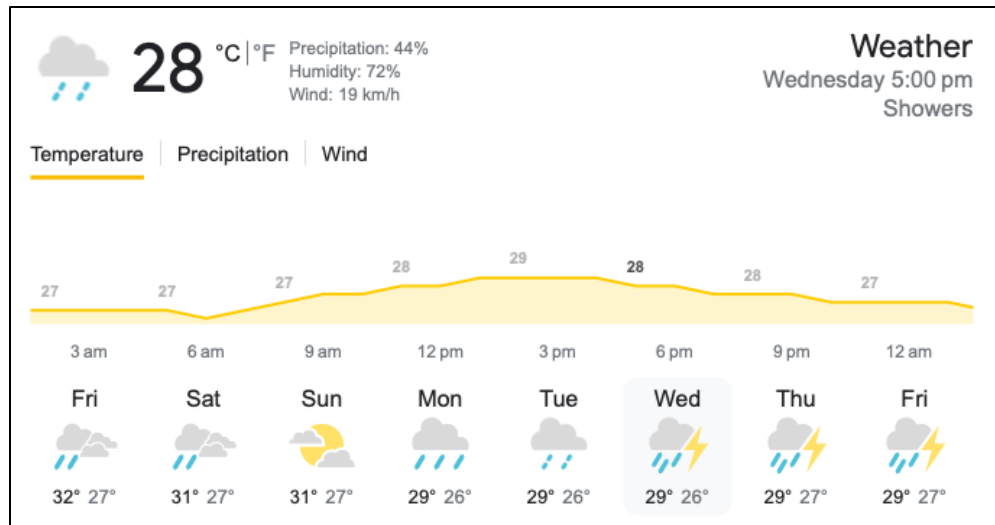
- Encoding & decoding
- Continuous learning
- Closed-loop feedback

What to Learn?

The fundamental reason why intelligence exists and evolves:

The world is not entirely random yet, and it is still largely predictable.

Intelligence and Science learn what is predictable from sensed data of external world (so every animal is Newton and has learned an accurate “world model.”)

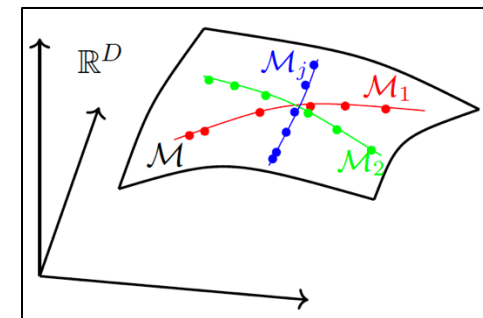
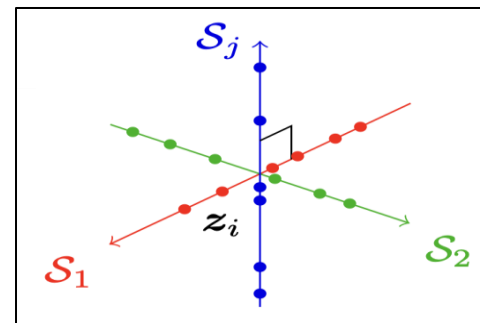
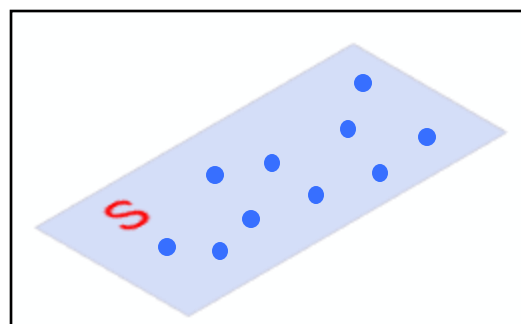
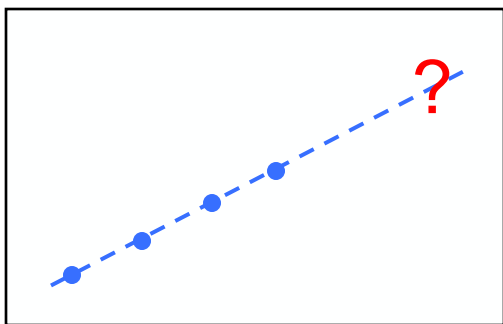


What to Learn?

The fundamental reason why intelligence exists and evolves:

The world is not entirely random yet, and it is still largely predictable.

Mathematically, all predictable information is encoded as a distribution $p(\mathbf{x})$ of **low-dimensional** supports in observed **high-dimensional** data space.



This is the only “inductive bias” necessary!

What to Learn: Low-dimensionality

Important properties of low-dimensional structures: **Completion**

$$\text{Bayes inference: } y = \mathcal{P}_{\Omega}(x) \rightarrow \underbrace{\hat{x} \sim p(x | y)}$$

Conditional sampling

Partial observations

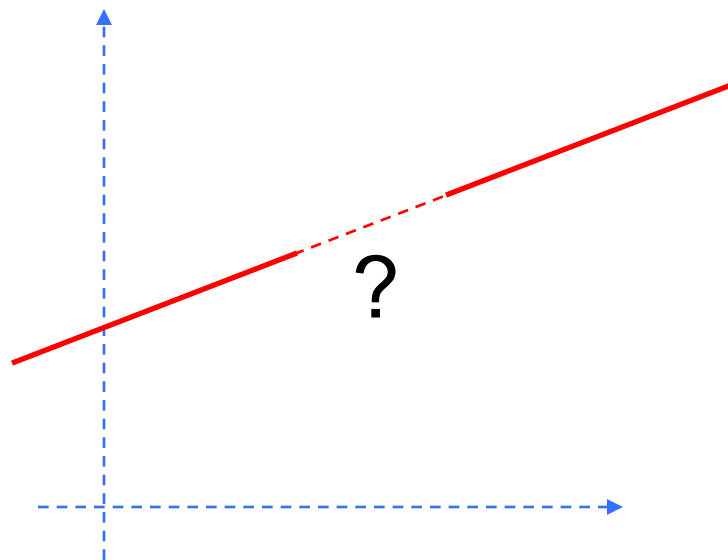
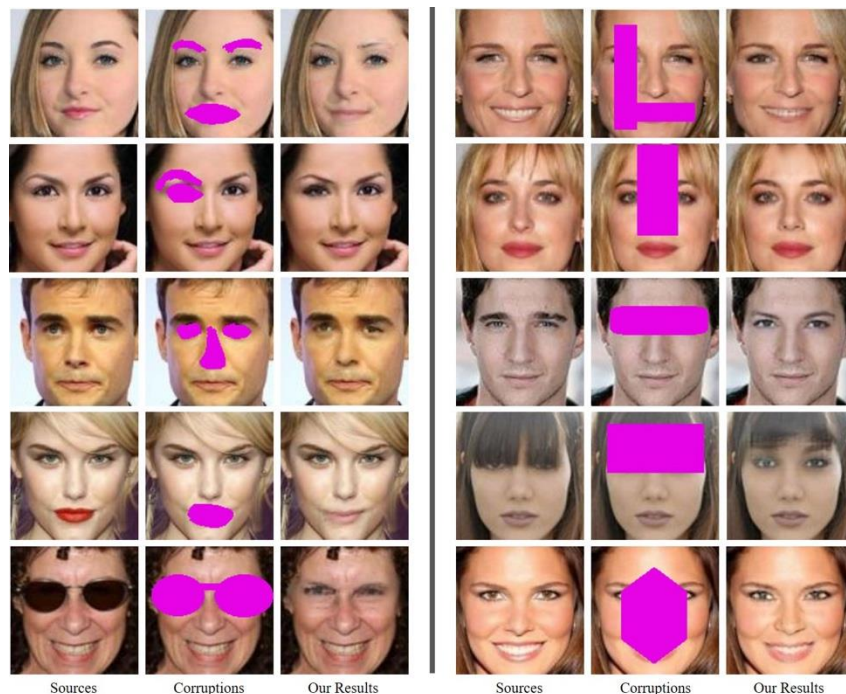
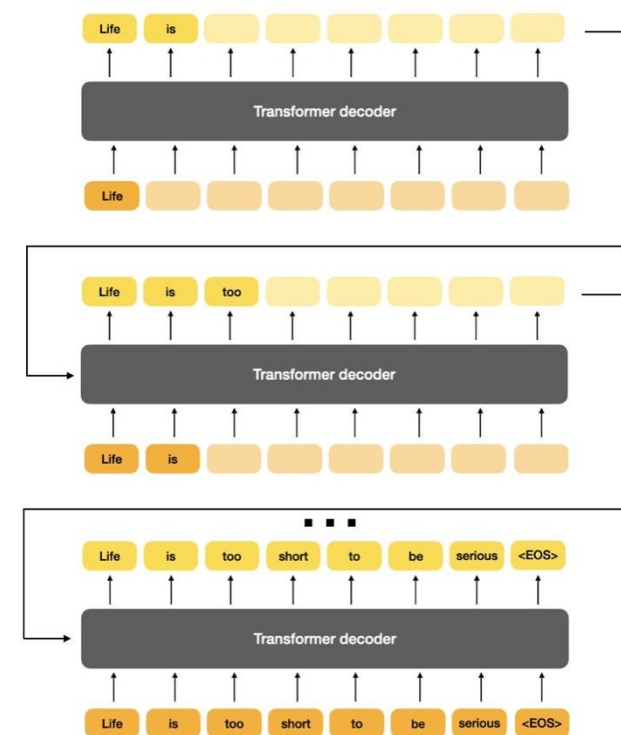


Image completion



Text prediction (GPT)



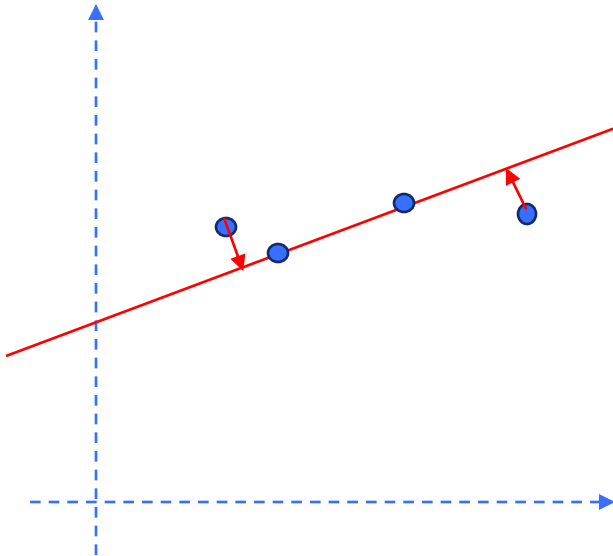
What to Learn: Low-dimensionality

Important properties of low-dimensional structures: **Denoising**

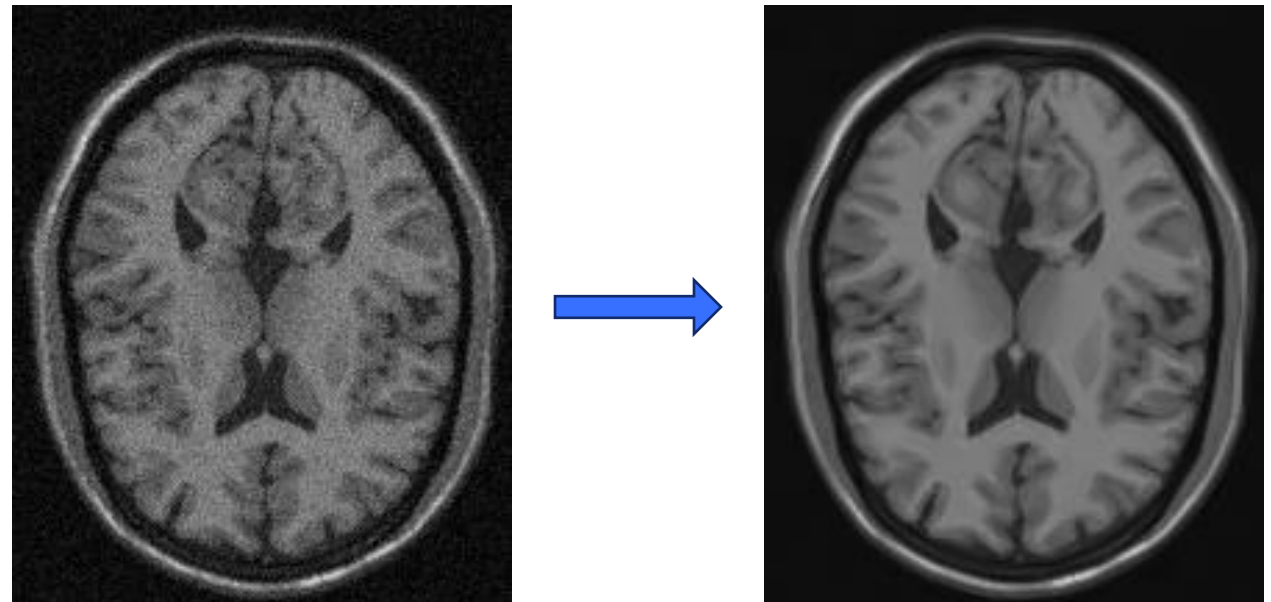
Empirical Bayes: $y = x + \sigma n \Rightarrow \hat{x} \sim \mathbb{E}(x | y) = \underbrace{y + \sigma^2 \nabla \log p(x)}_{\text{Tweedie's formula}}$

Tweedie's formula

Noisy observations



Natural image denoising (or generation)



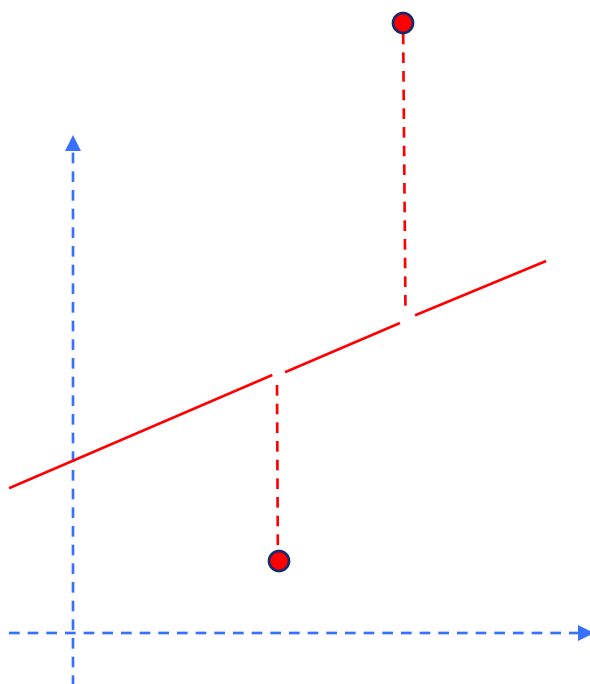
What to Learn?

Important properties of low-dimensional structures: **Error Correction**

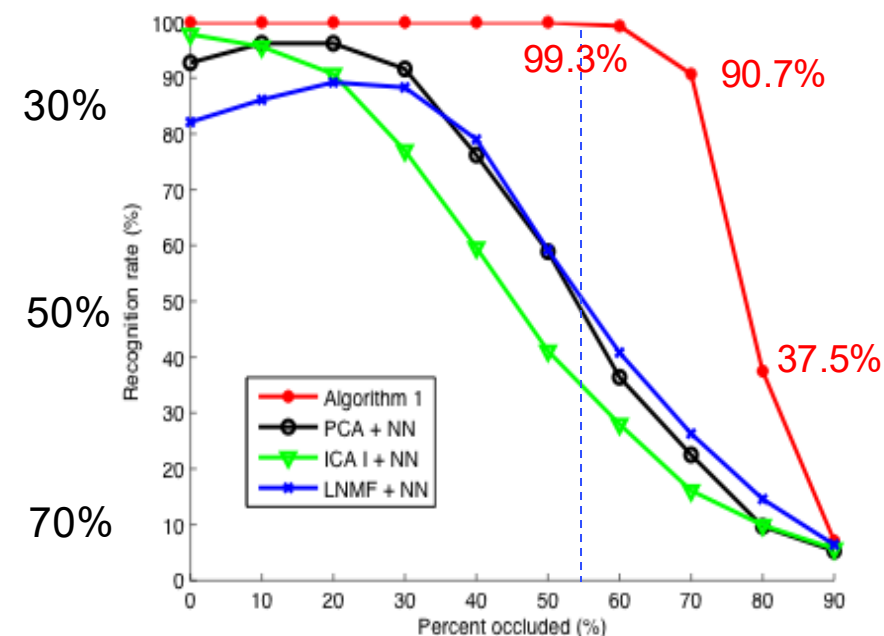
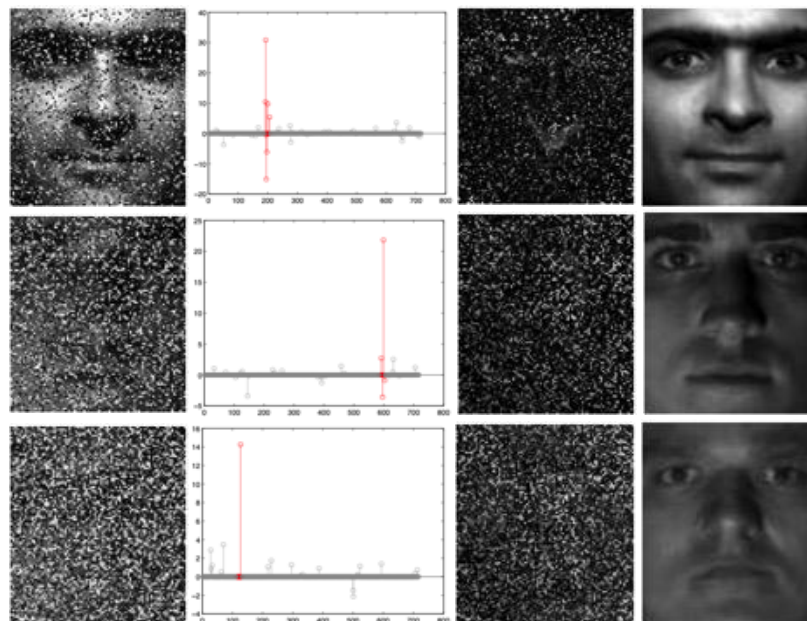
Robust Bayes inference: $y = x + e \rightarrow \hat{x} \sim \operatorname{argmin} \underbrace{||x||_1 + ||e||_1}_{\text{Exploiting sparsity prior}}$

Exploiting sparsity prior

Corrupted observations



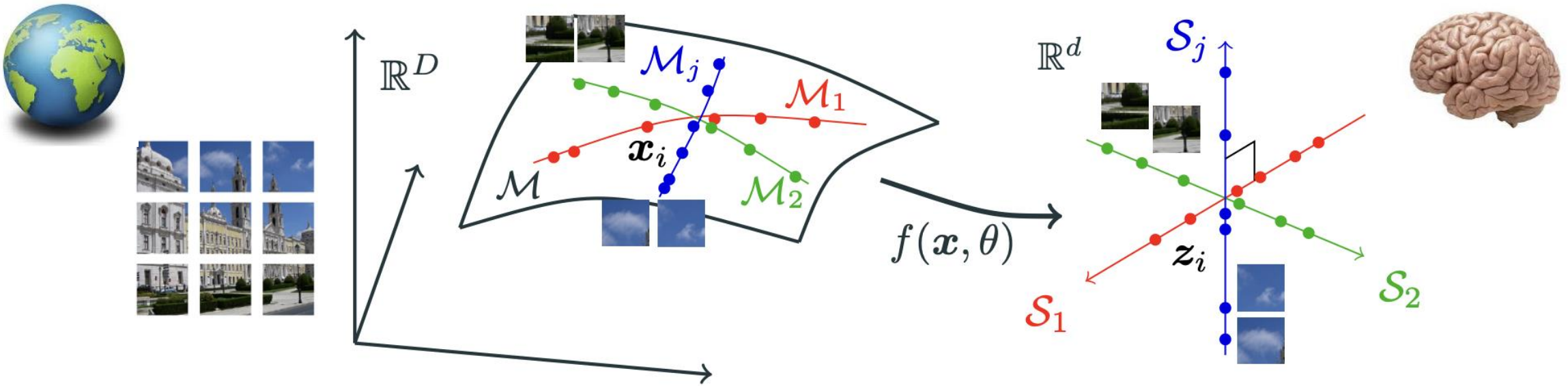
Robust (face) recognition via sparsity [WY+Ma, TPAMI'09]



What to Learn: Seeking Parsimony

The main objective of learning:

Identify a distribution with **low-dimensional structures** from sensed data x and transform them to a **compact and structured** representation z .

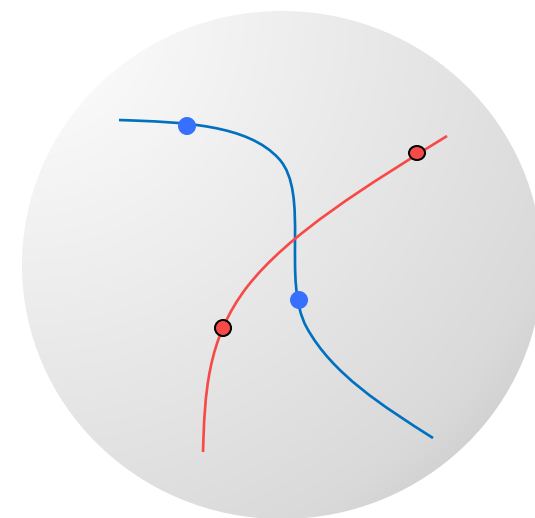


How to Learn: Measure of Parsimony

For distributions with **low-dim** structures, **entropy** (or “volume”) of the underlying data distribution should be very small (or “zero”):

(Discrete) Entropy: $H(X) = \sum_{x \in X} -p(x) \log p(x)$

Differential Entropy: $h(x) = \int -p(x) \log p(x) dx$



Technical Caveats with low-dim “distributions”:

Volume: $\text{vol}(x) = 0$

Differential entropy: $h(x) = -\infty$

Log Likelihood: $\log p(x) = +\infty$

KL divergence: $D(p||q)$ not defined

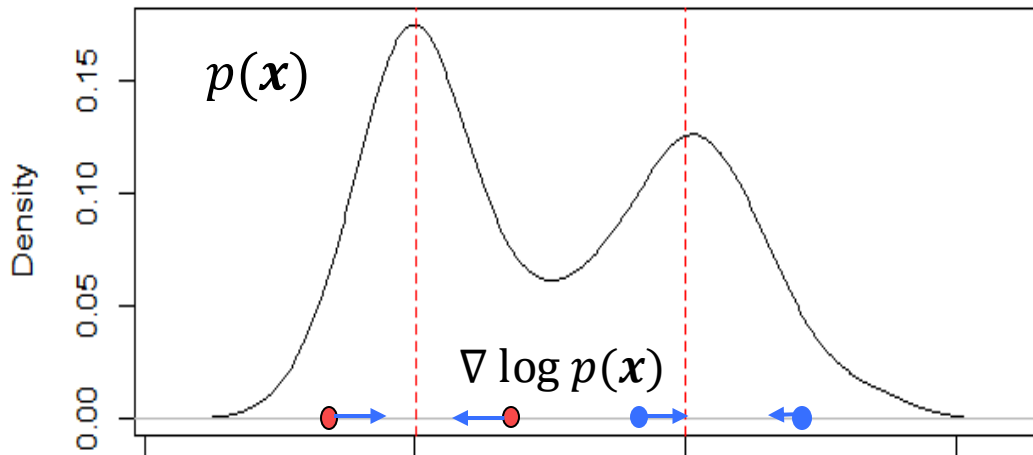


How to Learn: Compress

A fundamental and unifying mechanism to learn **low-dim** structures:
compress to reduce entropy of the observed (noisy) data distribution.

$$\text{minimize } h(\mathbf{x}) = \int -p(\mathbf{x}) \log p(\mathbf{x}) d\mathbf{x}$$

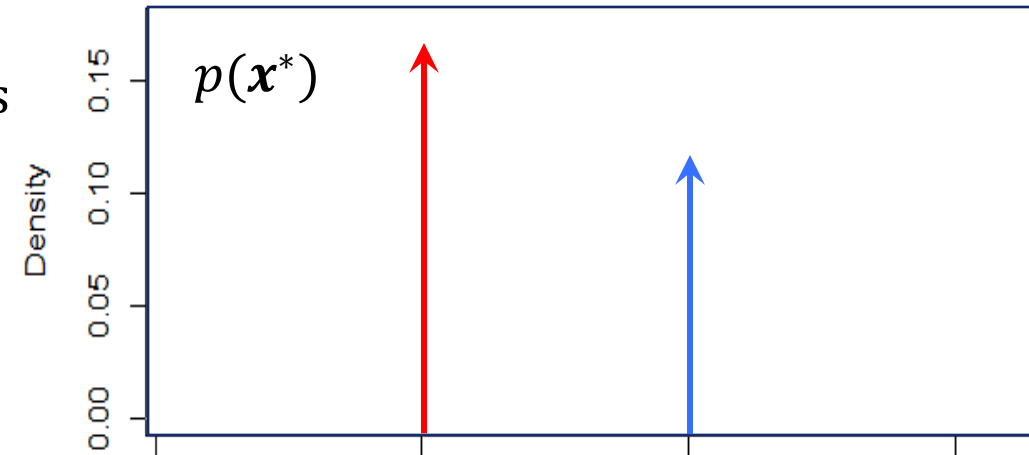
Noisy observations



$h(\mathbf{x})$ decreases



Converged law!



$$\mathbf{x}^{l+1} = \mathbf{x}^l + \beta \nabla \log p(\mathbf{x}^l)$$

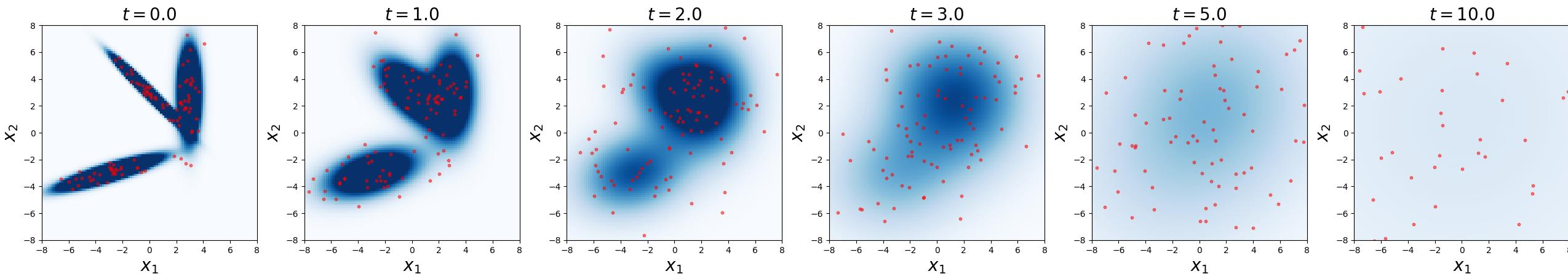
How to Learn: Compress via Denoising

Theorem **[Diffusion]** Consider the diffusion process:

$$\mathbf{x}_t = \mathbf{x}_o + t\mathbf{n}, \quad \mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

Under natural technical assumptions, the **entropy of the process increases**:

$$\frac{d}{dt} h(\mathbf{x}_t) > 0, \quad \forall t > 0.$$



How to Learn: Compress via Denoising

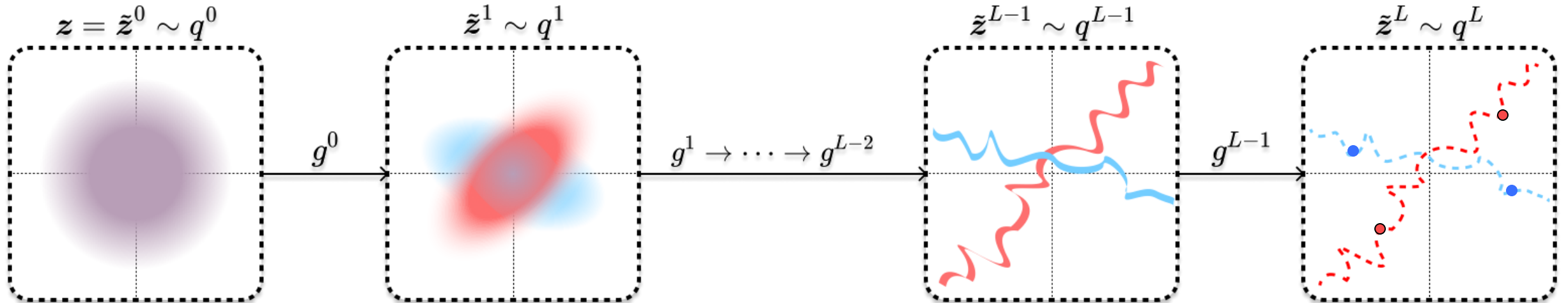
Theorem **[Denoising]** Consider the inverse denoising process:

$$\hat{\mathbf{x}}_{t-s} = \mathbb{E}[\mathbf{x}_{t-s} \mid \mathbf{x}_t] = \mathbf{x}_t + st \nabla \log p(\mathbf{x}_t)$$

Under natural technical assumptions, the **entropy of the process decreases**:

$$\frac{d}{ds} h(\hat{\mathbf{x}}_{t-s}) < 0, \quad \forall s > 0.$$

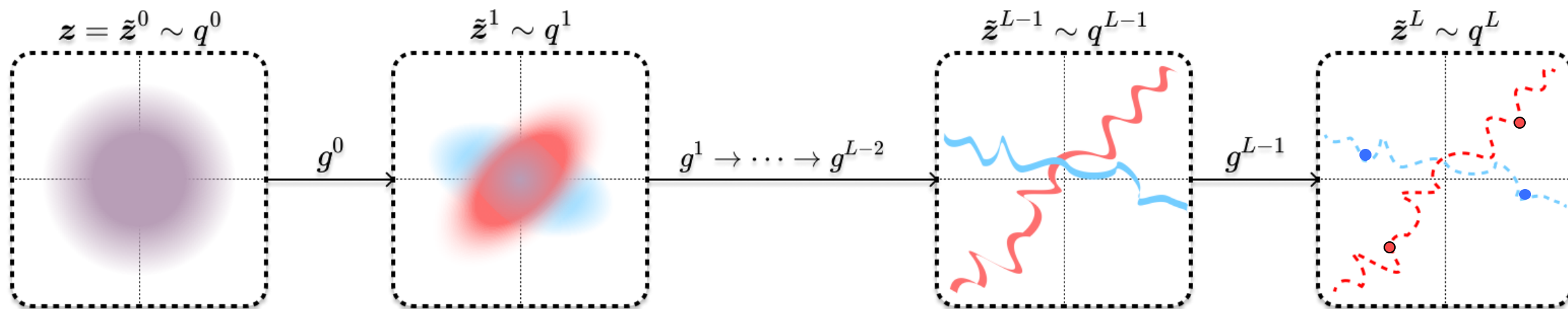
$$\tilde{\mathbf{z}}^{l+1} = \tilde{\mathbf{z}}^l + \beta \nabla \log p(\tilde{\mathbf{z}}^l)$$



How to Learn: Empirical Approaches

Compression via iterative denoising

$$\tilde{\mathbf{z}}^{l+1} = \tilde{\mathbf{z}}^l + \beta \nabla \log p(\tilde{\mathbf{z}}^l)$$

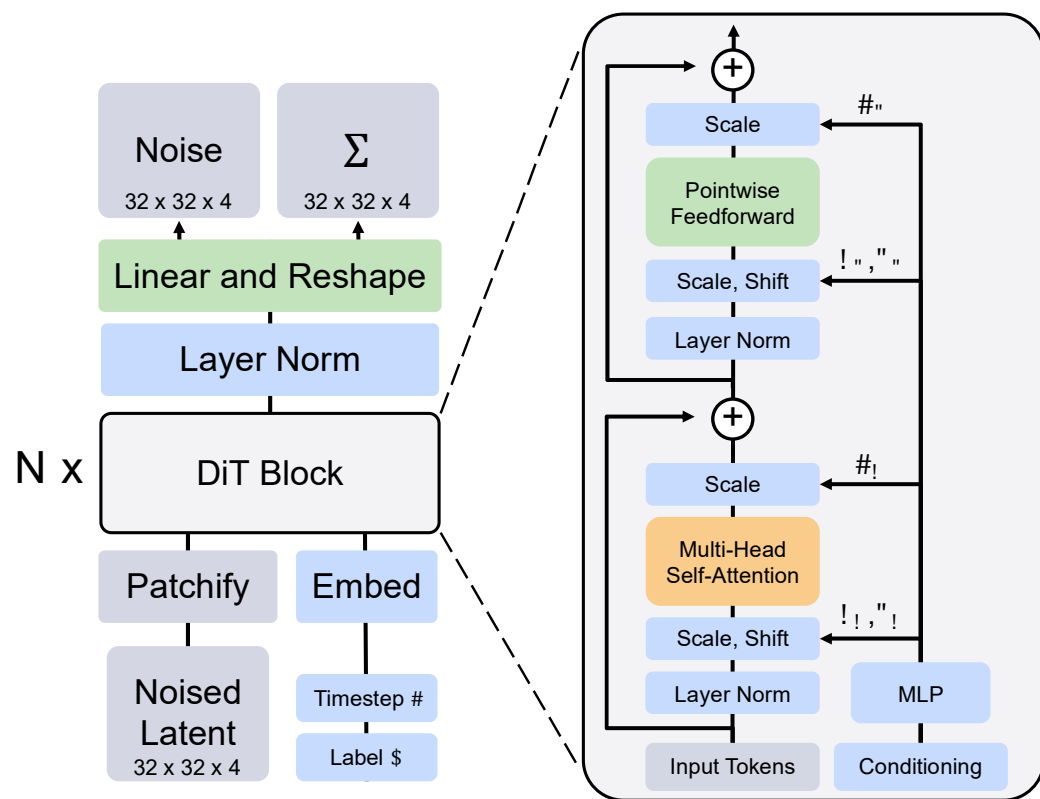


How to Learn: Empirical Approaches

Empirically designed networks to realize the denoising operator:

$$\tilde{\mathbf{z}}^{l+1} = \tilde{\mathbf{z}}^l + \beta \nabla \log p(\tilde{\mathbf{z}}^l) \text{ how to realize?}$$

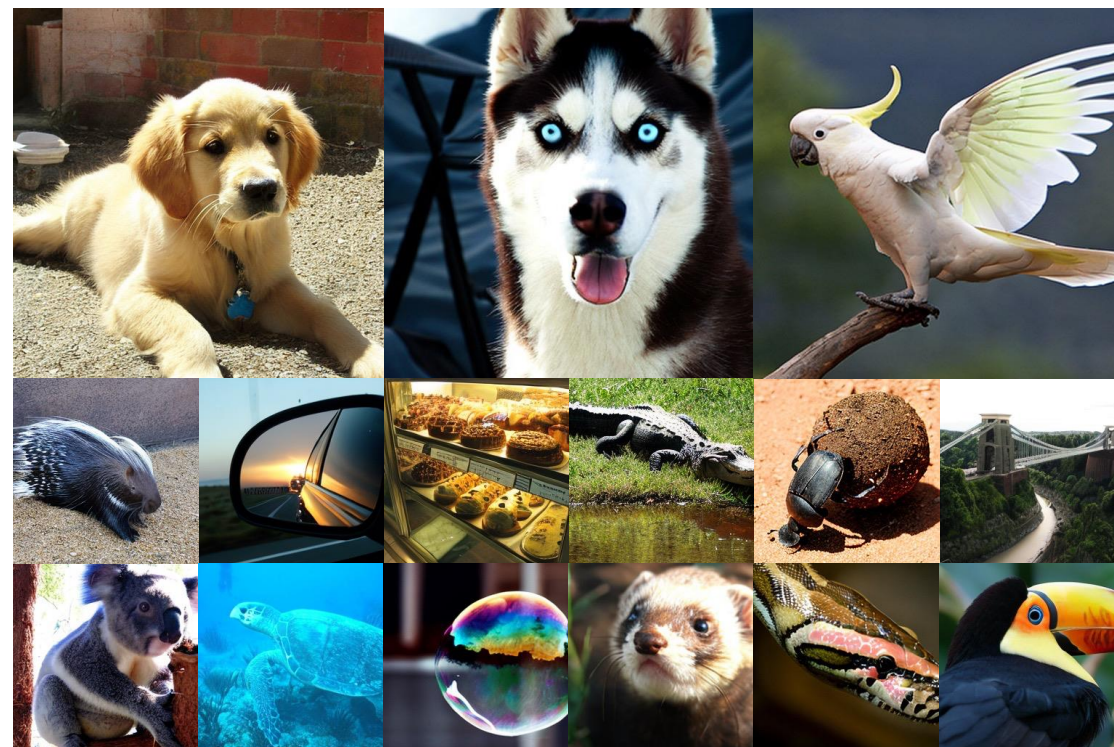
Diffusion Transformer (DiT)



Latent Diffusion Transformer

DiT Block with adaLN-Zero

Image or video generation



How to Learn: An Analytical Case

Analytically derived operation to realize the denoising operator:

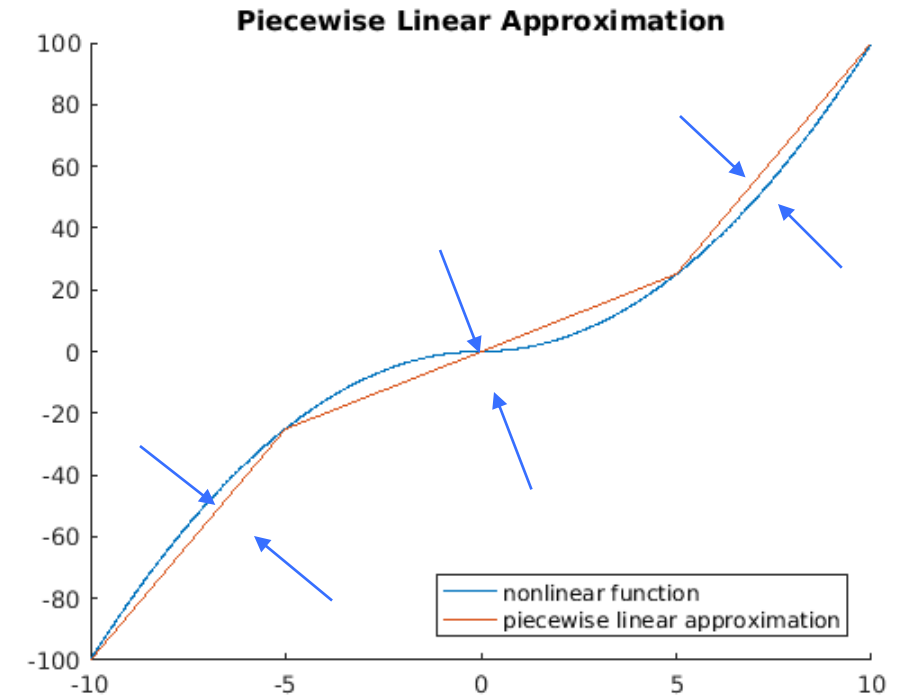
$$\tilde{\mathbf{z}}^{l+1} = \tilde{\mathbf{z}}^l + \beta \nabla \log p(\tilde{\mathbf{z}}^l) \quad \text{how to realize?}$$

If we approximate a general distribution $p(\tilde{\mathbf{z}})$ with a **mixture of subspaces or low-dim Gaussians**, e.g. PCA, ICA, GPCA, Sparse Coding [W+Ma, 2022],

$$\tilde{\mathbf{z}}^l \sim \frac{1}{K} \sum_{k=1}^K \mathcal{N}(\mathbf{0}, \mathbf{U}_k \mathbf{U}_k^\top), \quad \mathbf{U}_k \in O(D, d),$$

then

$$\tilde{\mathbf{z}}^{l+1} \propto \frac{1}{K} \sum_{k=1}^K \text{softmax}\{\alpha \|\mathbf{U}_k^\top \tilde{\mathbf{z}}^l\|^2\} \mathbf{U}_k \mathbf{U}_k^\top \tilde{\mathbf{z}}^l.$$



How to Learn: Measure of Compression/Information

How to measure information in a distribution with a low-dimensional support?

Dimension
(points)

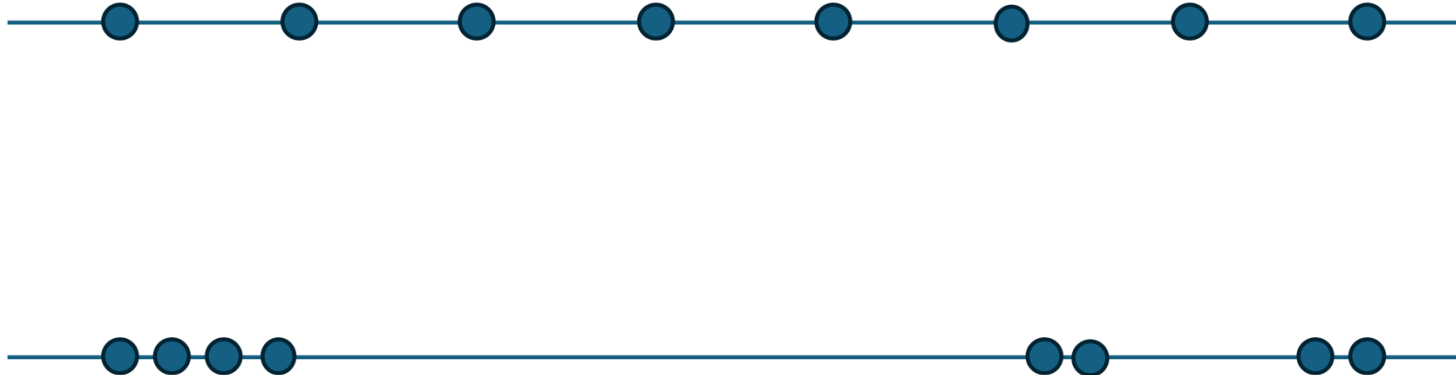
$$\dim(\mathbf{x}) = 0$$

Volume
(points/lines)

$$\text{vol}(\mathbf{x}) = 0$$

Differential entropy
(points/lines)

$$h(\mathbf{x}) = -\infty$$



How to Learn: Measure of Compression/Information

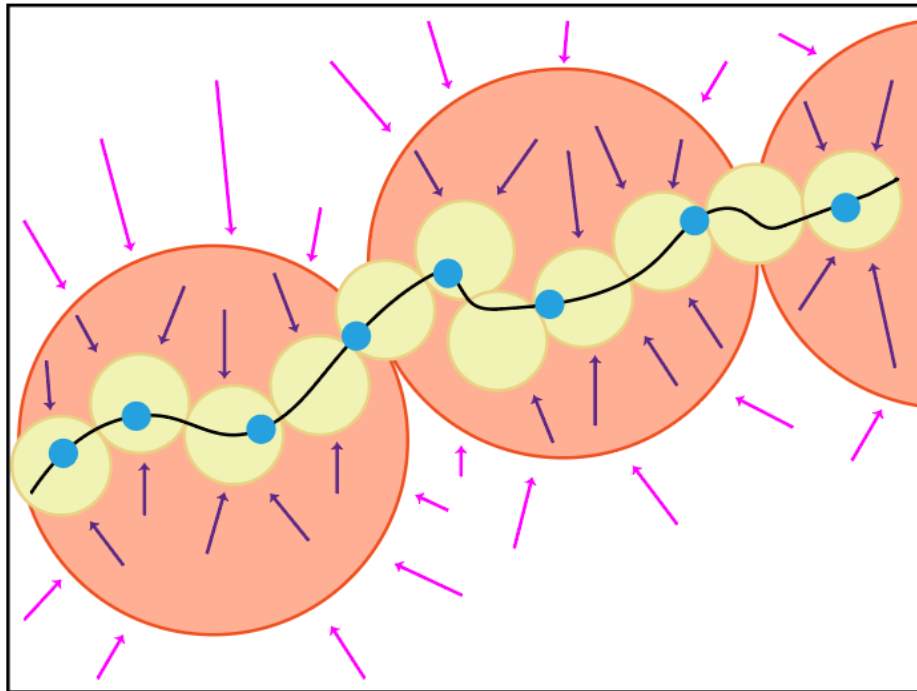
How to measure information in a distribution with a low-dimensional support?

Dimension
 $\dim(\mathbf{x}) = 0$

Volume
 $\text{vol}(\mathbf{x}) = 0$

Differential entropy
 $h(\mathbf{x}) = -\infty$

Construct *a finite codebook* by packing the support of the distribution with ϵ -balls.



rate distortion

$$R(\mathbf{x}, \epsilon) = \min_{\mathbb{E} \|\hat{\mathbf{x}} - \mathbf{x}\| \leq \epsilon} h(\hat{\mathbf{x}}) - h(\hat{\mathbf{x}} | \mathbf{x})$$



How to Learn: Measure of Information Gain

How to explicitly represent a distribution with a low-dimensional support?

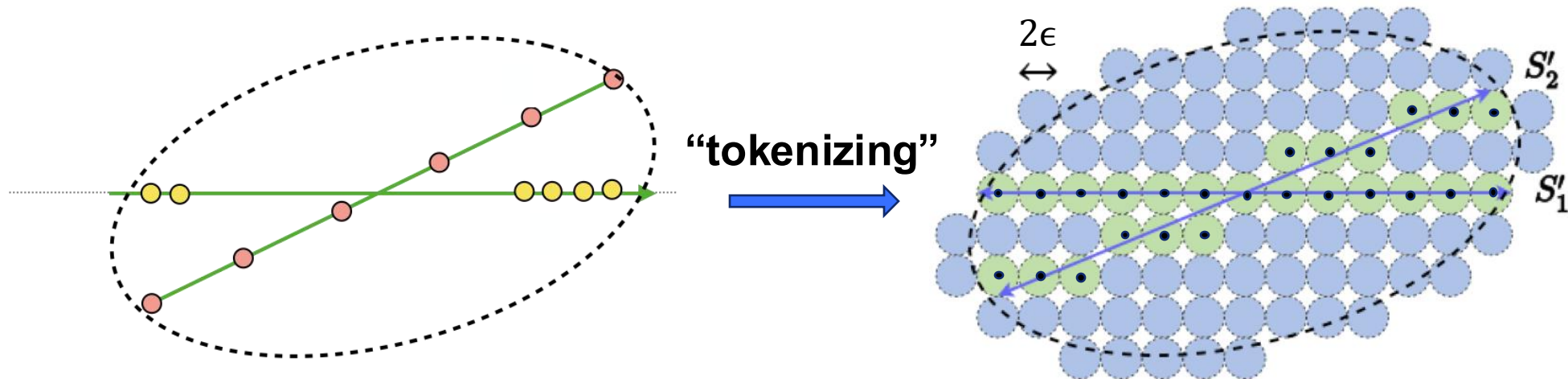
A key idea: **lossy encoding and decoding**

construct *a finite codebook* by packing the support of the distribution with ϵ -balls.

volume differential entropy
 $\text{vol}(\mathbf{x}) = 0$ $h(\mathbf{x}) = -\infty$

rate distortion

$$R(\mathbf{x}, \epsilon) = \min_{\mathbb{E}[\|\hat{\mathbf{x}} - \mathbf{x}\| \leq \epsilon} h(\hat{\mathbf{x}}) - h(\hat{\mathbf{x}} | \mathbf{x})$$



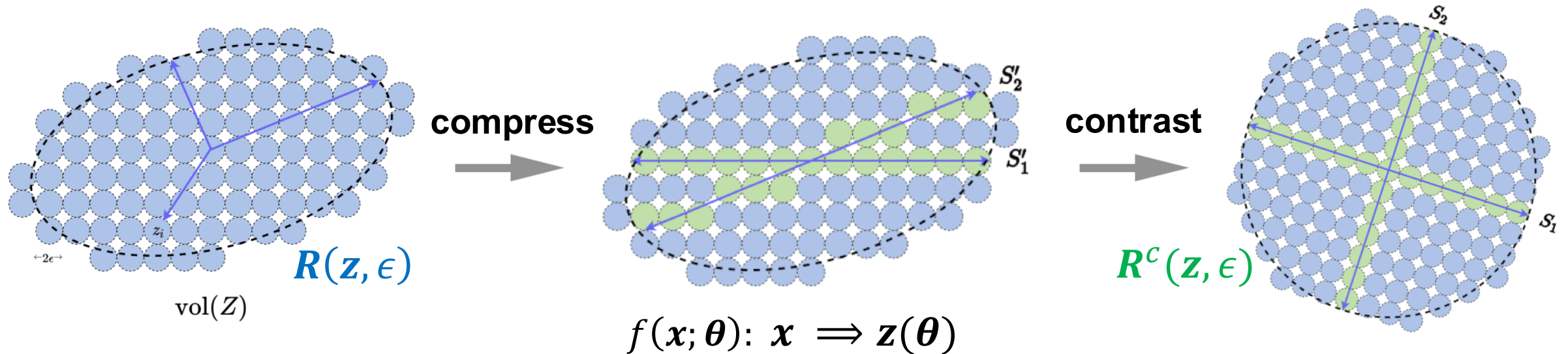
Segmentation of Mixed Data via Lossy Coding and Compression [Ma+DHW, TPAMI2007].



How to Learn: Organize Information

How to make the resulting representation the most informative?

compress what is similar; **contrast** what is dissimilar.



Mathematically: maximize **information gain** or **reduce coding rate**:

$$\max \Delta R(z, \epsilon) = R(z, \epsilon) - R^c(z, \epsilon)$$

ReduNet: A White-box Deep Network from the Principle of MCR^2 [YCY+Ma, JMLR2022].

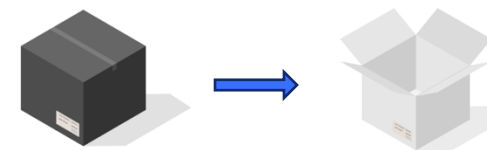
How to Learn: Maximal Coding Rate Reduction (ReduNet)

When $\mathbf{Z} = f(\mathbf{X}; \boldsymbol{\theta})$ is a mixture of K Gaussians with $\boldsymbol{\Pi}_k$ encodes membership of samples in the k th class.

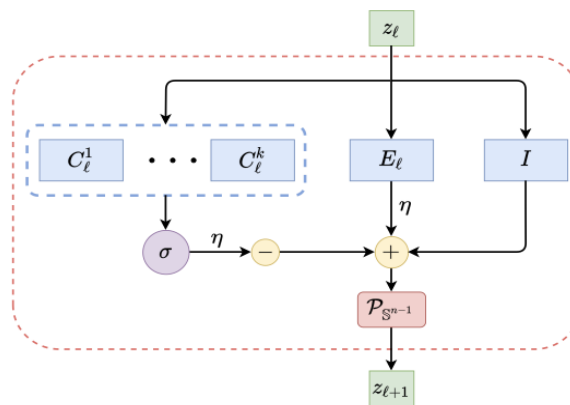
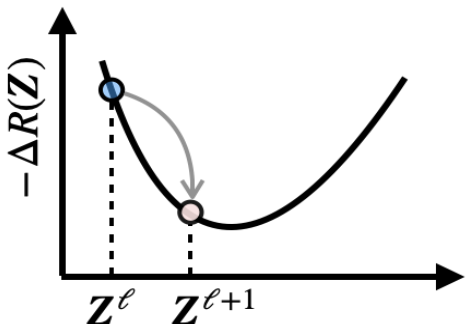
Maximal Coding Rate Reduction (MCR²)

$$\max_f \Delta R(\mathbf{Z} | \boldsymbol{\Pi}) = \underbrace{\frac{1}{2} \log \det \left(\mathbf{I} + \frac{d}{\varepsilon^2} \cdot \frac{\mathbf{Z}\mathbf{Z}^\top}{m} \right)}_{R(\mathbf{Z})} - \underbrace{\sum_{k=1}^K \frac{\text{tr}(\boldsymbol{\Pi}_k)}{2m} \log \det \left(\mathbf{I} + \frac{d}{\varepsilon^2} \cdot \frac{\mathbf{Z}\boldsymbol{\Pi}_k\mathbf{Z}^\top}{\text{tr}(\boldsymbol{\Pi}_k)} \right)}_{R^c(\mathbf{Z}|\boldsymbol{\Pi})}$$

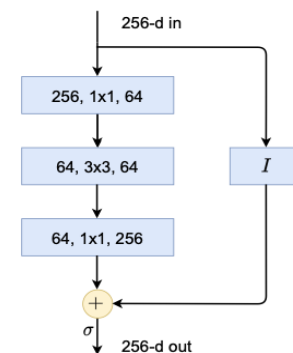
Unrolled optimization: gradient ascent on the MCR² objective,



$$\mathbf{Z}^{\ell+1} = \mathbf{Z}^\ell + \eta \cdot \frac{\partial \Delta R}{\partial \mathbf{Z}} \Big|_{\mathbf{Z}=\mathbf{Z}^\ell}$$



ReduNet (MoE?)



ResNet

ReduNet: A White-box Deep Network from the Principle of MCR² [YCY+Ma, JMLR2022].

How to Learn: Maximal Coding Rate Reduction (ReduNet)

Benign local and global optimization landscape of MCR^2

Maximal Coding Rate Reduction (MCR^2)

$$\max_f \Delta R(\mathbf{Z} | \Pi) = R(\mathbf{Z}) - R^c(\mathbf{Z} | \Pi)$$

Theorem [YCY+M NeurIPS2020].

The global optimal solution $\mathbf{Z}^* = [\mathbf{Z}_1^*, \mathbf{Z}_2^*, \dots, \mathbf{Z}_K^*]$ of MCR^2 satisfies:

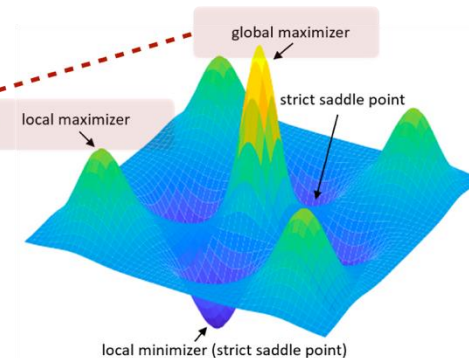
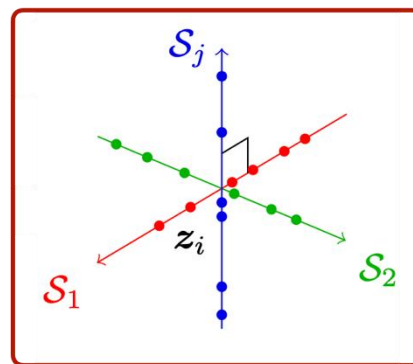
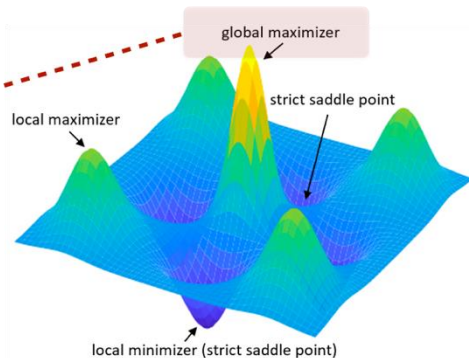
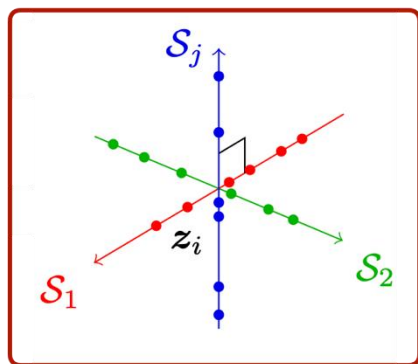
- Subspaces of different classes are orthogonal to each other, $(\mathbf{Z}_i^*)^\top \mathbf{Z}_j^* = \mathbf{0}$ for $i \neq j$;
- Each subspace achieves its maximal dimension, i.e., $\text{rank}(\mathbf{Z}_k^*) = d_k$, and $d = \sum_{k=1}^K d_k$.

Regularized Maximal Coding Rate Reduction (MCR^2)

$$\max_f F(\mathbf{Z}) := \Delta R_\lambda(\mathbf{Z} | \Pi) = R(\mathbf{Z}) - R^c(\mathbf{Z} | \Pi) - \lambda \cdot \|\mathbf{Z}\|_F^2$$

Theorem [WLY+M 2024].

Every critical point $\{\mathbf{Z} : \nabla F(\mathbf{Z}) = \mathbf{0}\}$ is either a local maximizer or a strict saddle point: each local maximizer corresponds to a feature representation that consists of a family of orthogonal subspaces.

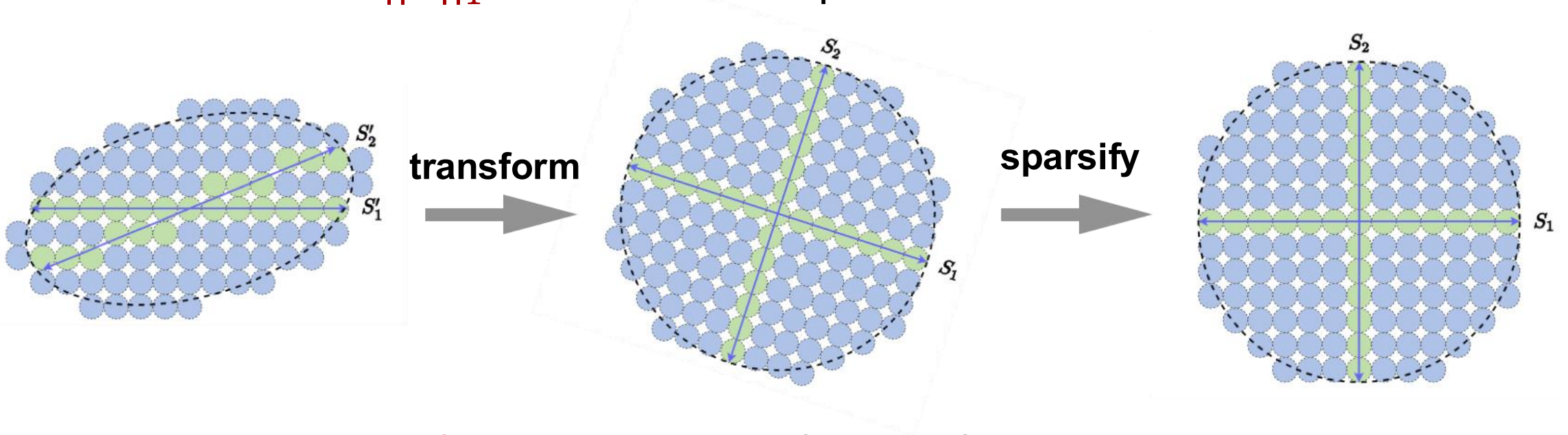


How to Learn: Sparse Rate Reduction (SRR)

Maximize the difference between the coding rate of all features and the coding rate of features within each subspace, and promote sparsity:

$$\max_{\mathbf{Z}} \text{SRR}(\mathbf{Z} | \mathbf{U}_{[K]}) = R(\mathbf{Z}) - R^c(\mathbf{Z} | \mathbf{U}_{[K]}) - \lambda \|\mathbf{Z}\|_1$$

$-\lambda \|\mathbf{Z}\|_1$: measure how sparse all features are.



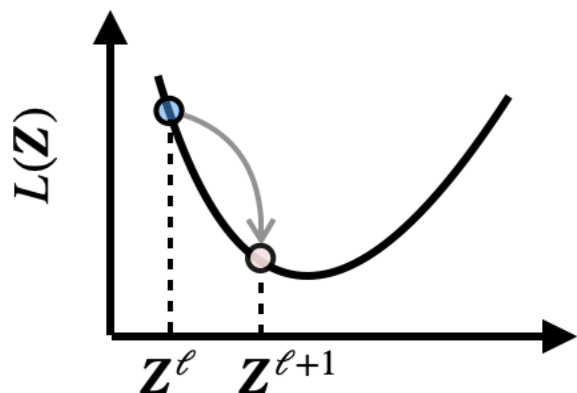
CRATE: White-box Transformer via Sparse Rate Reduction [YBP+Ma, JMLR2024].

How to Learn: DNN to Realize Iterative Optimization

Optimize Sparse Rate Reduction via **gradient descent (GD)**

$$\min L_{SRR}(\mathbf{Z}) = -\text{SRR}(\mathbf{Z} \mid \mathbf{U}_{[K]})$$

gradient descent $\mathbf{Z}^{\ell+1} = f^{\ell}(\mathbf{Z}^{\ell}) \approx \text{Prox}[\mathbf{Z}^{\ell} - \eta \cdot \nabla L_{\text{srr}}(\mathbf{Z}^{\ell})]$



Sparse Rate Reduction (SRR):

$$\min_f L_{\text{srr}}(\mathbf{Z} \mid \mathbf{U}_{[K]}) = \underbrace{R^c(\mathbf{Z} \mid \mathbf{U}_{[K]})}_{\text{compression}} + \underbrace{\lambda \|\mathbf{Z}\|_1 - R(\mathbf{Z})}_{\text{sparsification}}$$

$$\mathbf{Z}^{\ell+1} = \mathbf{Z}^{\ell} - \eta \cdot \nabla L(\mathbf{Z}) \Big|_{\mathbf{Z}=\mathbf{Z}^{\ell}}$$

f^{ℓ}

Design the ℓ -th layer $f^{\ell} = f_2^{\ell} \circ f_1^{\ell}$ via an **Alternating Minimization Scheme**:

- **Compression Step**: $\mathbf{Z}^{\ell+1/2} = f_1^{\ell}(\mathbf{Z}^{\ell}) \approx \mathbf{Z}^{\ell} - \eta \cdot \nabla R^c(\mathbf{Z}^{\ell}; \mathbf{U}_{[K]})$;
- **Sparsification Step**: $\mathbf{Z}^{\ell+1} = f_2^{\ell}(\mathbf{Z}^{\ell+1/2}) \approx \text{Prox}_{\lambda \|\cdot\|_1} [\mathbf{Z}^{\ell+1/2} - \eta \cdot \text{grad}(\mathbf{Z}^{\ell+1/2})]$.

How to Learn: Interpretation of Each Layer

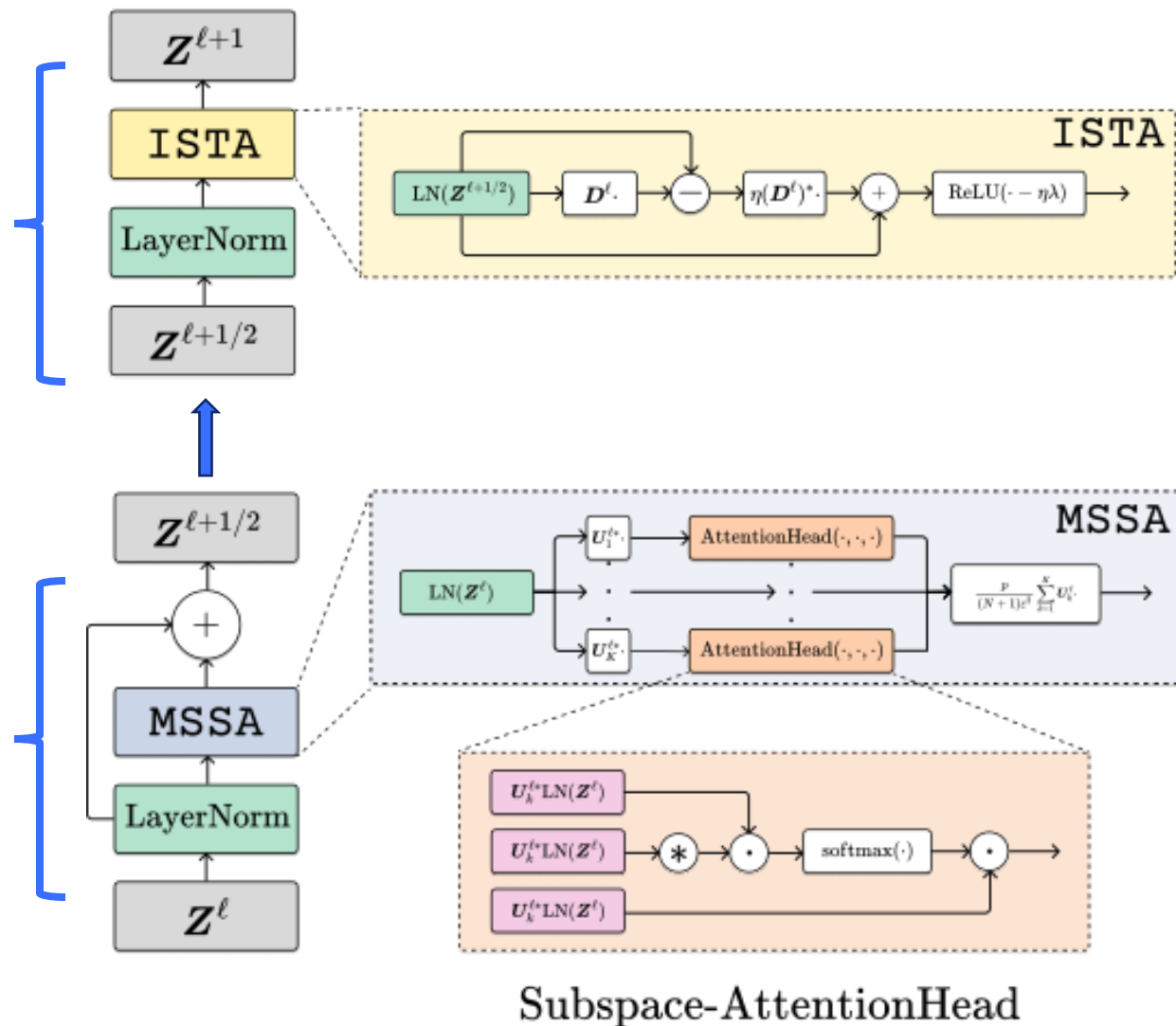
Gradient Descent (GD) operator:

Sparsification Step:

$$\begin{aligned} \mathbf{Z}^{\ell+1} &= \text{Prox}_{\lambda \|\cdot\|_1} [\mathbf{Z}^{\ell+1/2} - \eta \cdot \text{grad}(\mathbf{Z}^{\ell+1/2})] \\ &= \underbrace{\text{ReLU}(\mathbf{Z}^{\ell+1/2} + \eta \cdot \mathbf{D}^\top (\mathbf{Z}^{\ell+1/2} - \mathbf{D} \mathbf{Z}^{\ell+1/2}) - \eta \cdot \lambda \cdot \mathbf{1})}_{\text{ISTA}(\mathbf{Z}^{\ell+1/2})} \end{aligned}$$

Compression Step:

$$\begin{aligned} \mathbf{Z}^{\ell+1/2} &= \mathbf{Z}^\ell - \eta \cdot \nabla R^c(\mathbf{Z}^\ell; \mathbf{U}_{[K]}) \\ &\approx (1 - \eta \cdot \beta) \mathbf{Z}^\ell + \eta \cdot \beta \cdot \text{MSSA}(\mathbf{Z}^\ell | \mathbf{U}_{[K]}) \end{aligned}$$

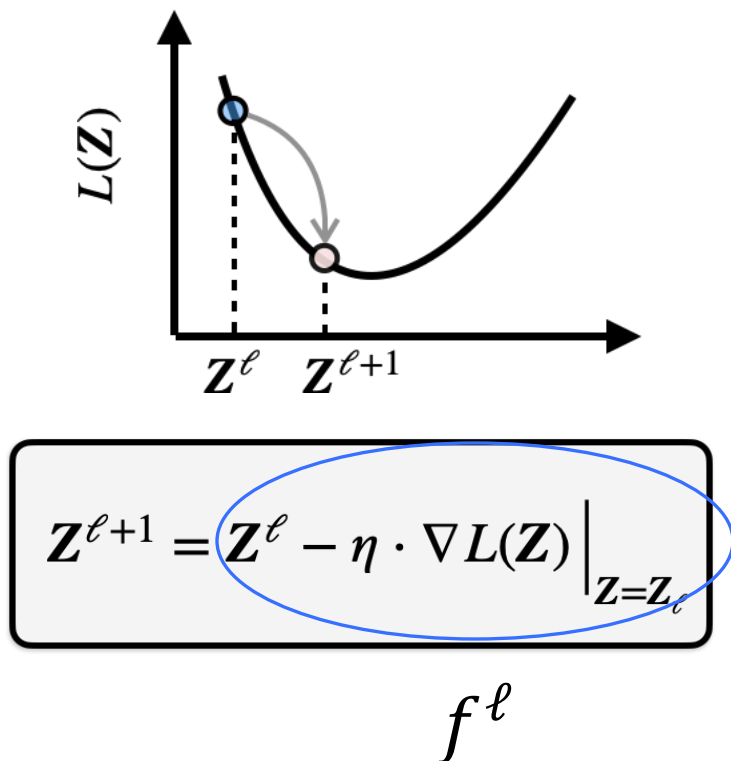


CRATE: White-box Transformer via Sparse Rate Reduction [YBP+Ma, JMLR2024].

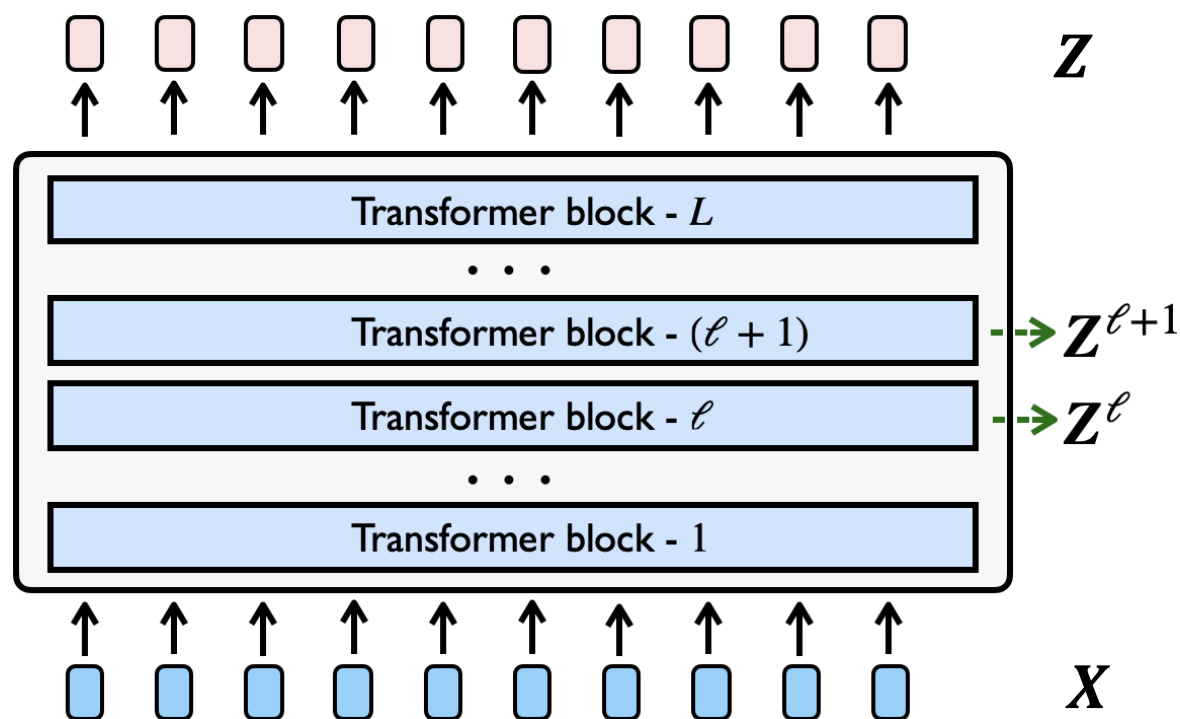
How to Learn: DNN to Realize Iterative Optimization

Optimize Rate Reduction via **iterative gradient descent (GD)**

$$\min L_{SRR}(\mathbf{Z}) = -\Delta R(\mathbf{Z} \mid \mathbf{U}_{[K]})$$



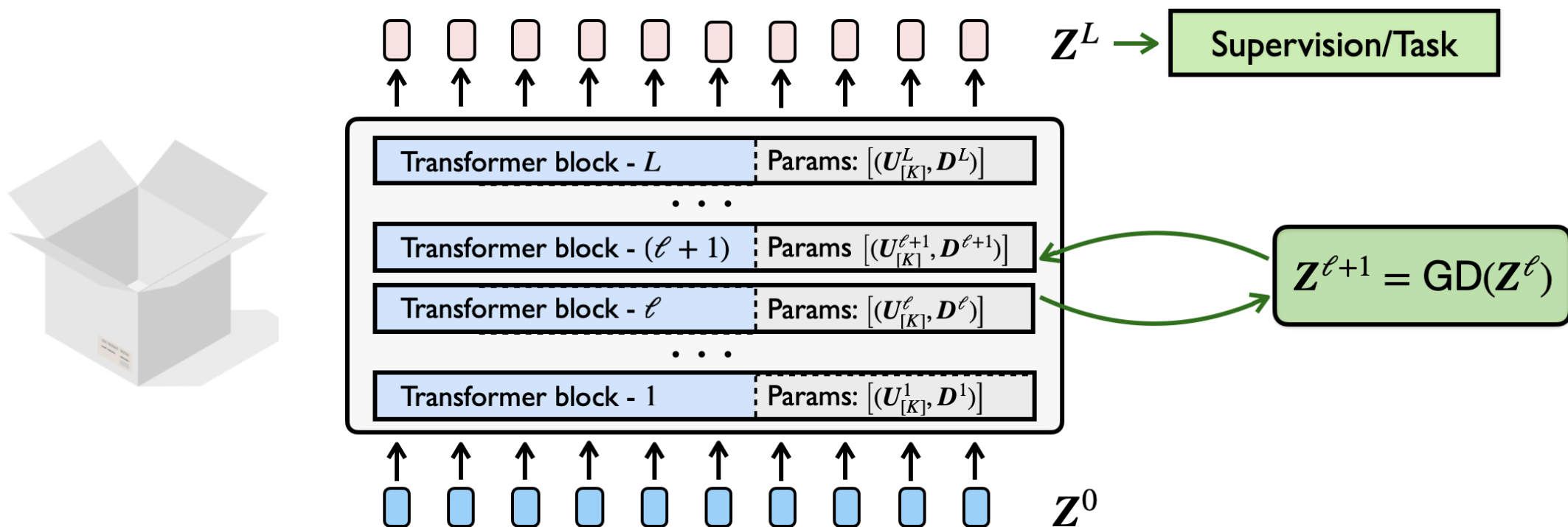
$$f: \mathbf{X} = \mathbf{Z}^0 \xrightarrow{f^1} \mathbf{Z}^1 \xrightarrow{f^2} \mathbf{Z}^2 \longrightarrow \dots \xrightarrow{f^L} \mathbf{Z}^L = \mathbf{Z}$$



Each layer of a deep network (e.g., Transformer) realizes a GD operator.

How to Learn: Interpretation of the Whole Network

- **Forward encoding**: given fixed subspaces and dictionaries $(U_{[K]}^\ell, D^\ell)_{\ell \in [L]}$, each layer performs compression and sparsification on representations.
- **Backward learning the “codebook”**: backpropagation to learn subspaces and dictionaries $(U_{[K]}^\ell, D^\ell)_{\ell \in [L]}$ from data.

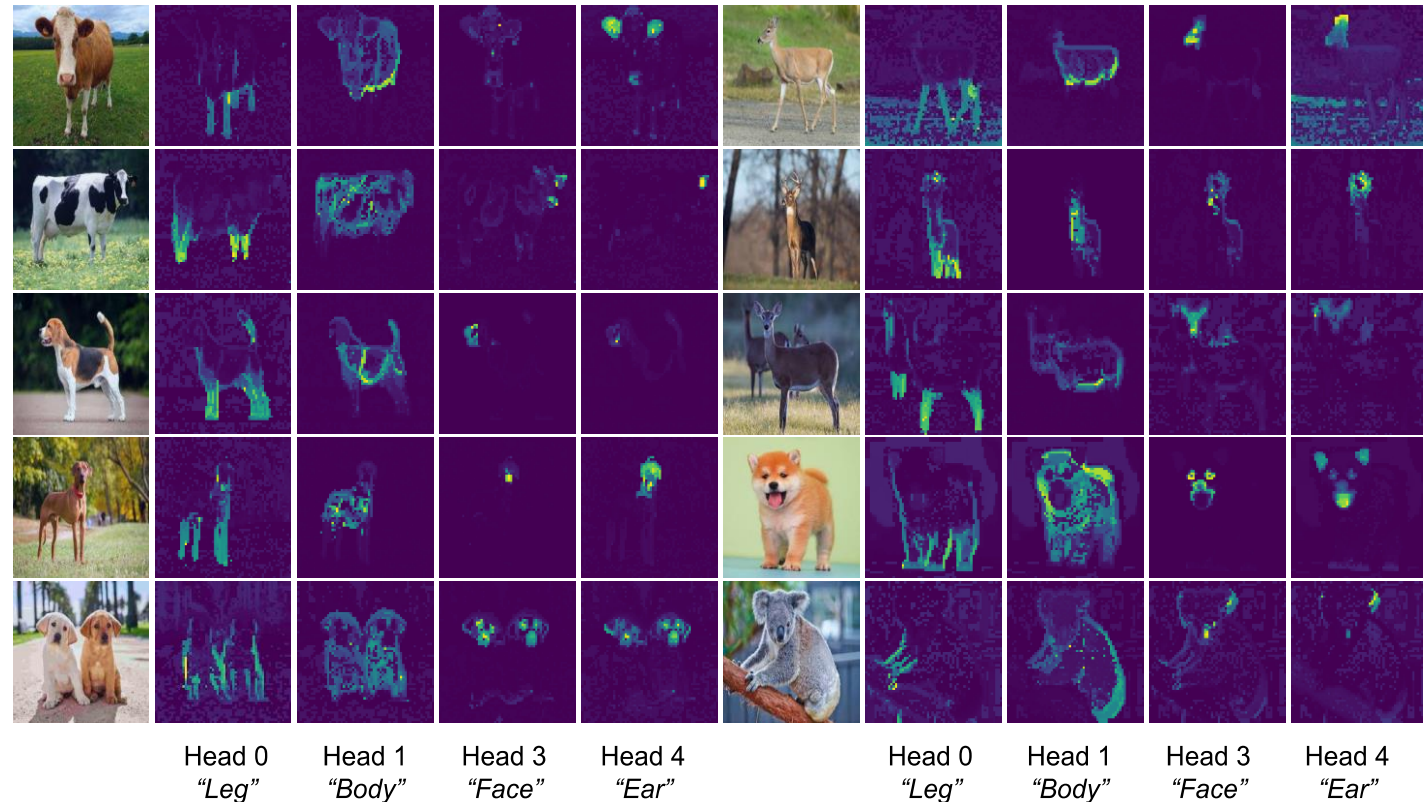
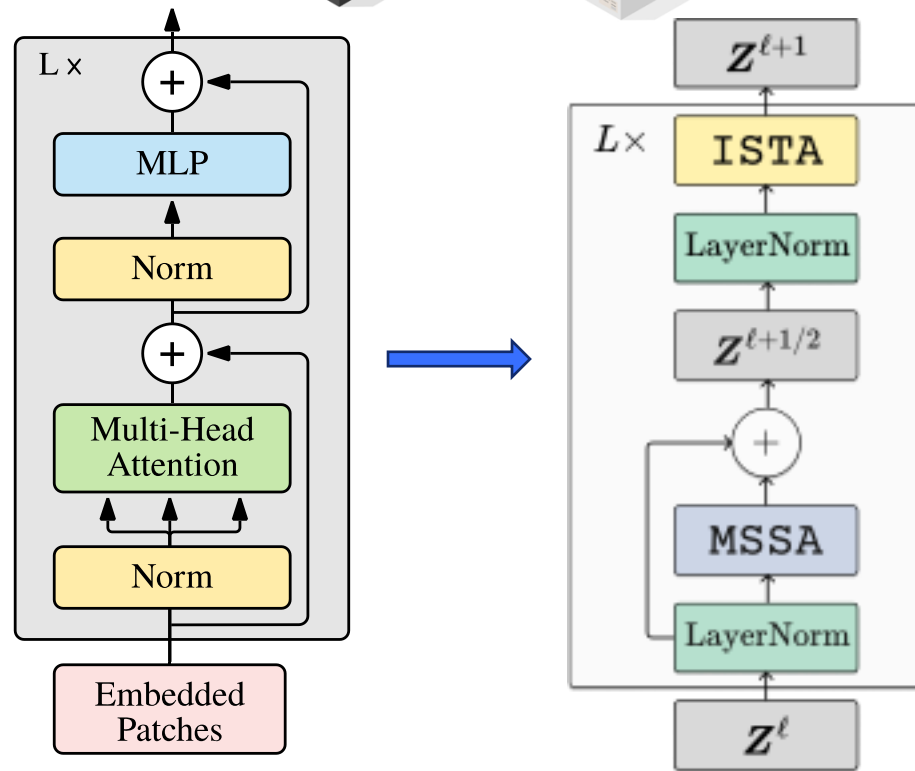


How to Learn: Better Semantic Interpretability

Not only mathematically fully explainable, but also semantically more interpretable!



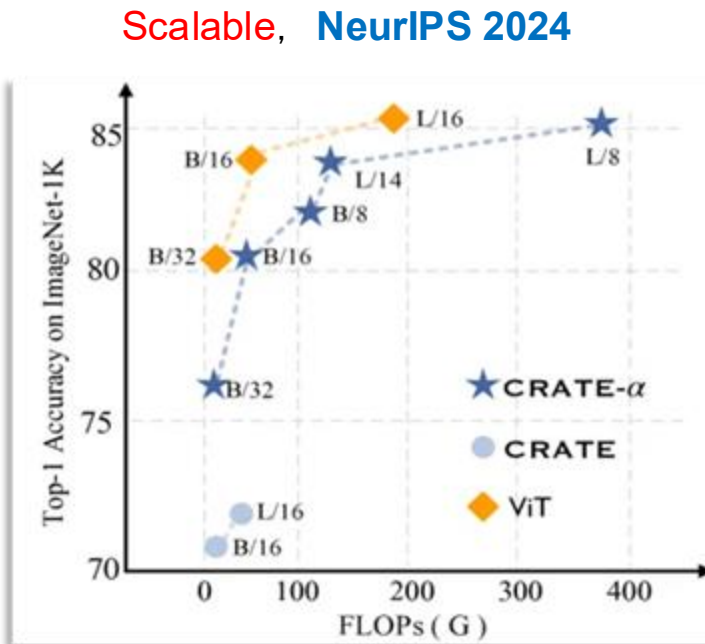
For real data (ImageNet), CRATE learns semantically meaningful structures.



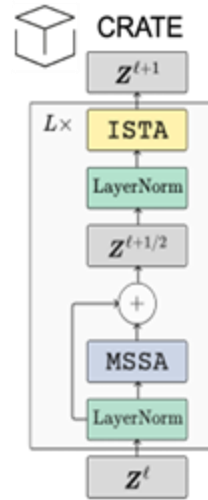
How to Learn: Better Networks from First Principles

No more trial and error to design better network architectures:

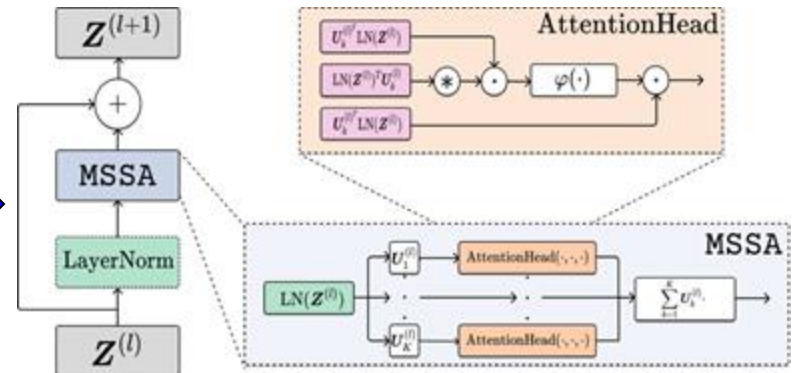
- Explainable [NeurIPS 2023]
- Scalable [NeurIPS 2024]
- More efficient [ICLR 2025]
- More compact [ICML 2025]



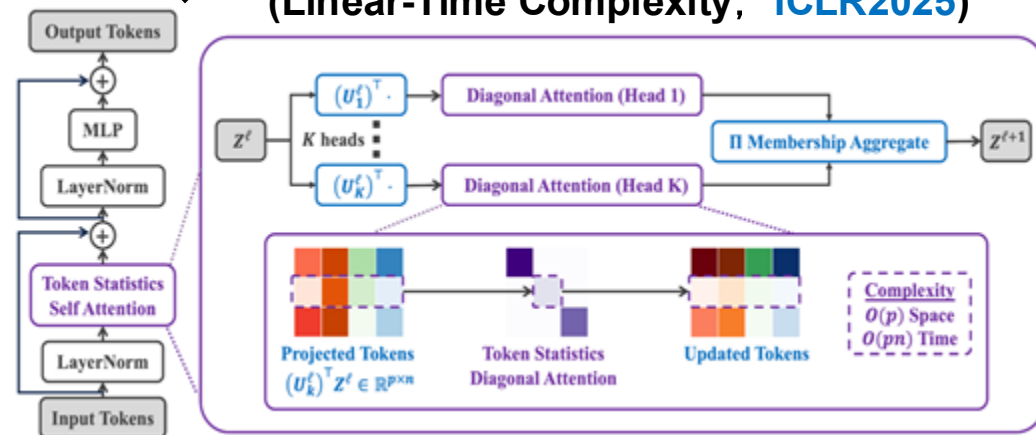
NeurIPS 2023



More Compact
(Attention Only, ICML2025)



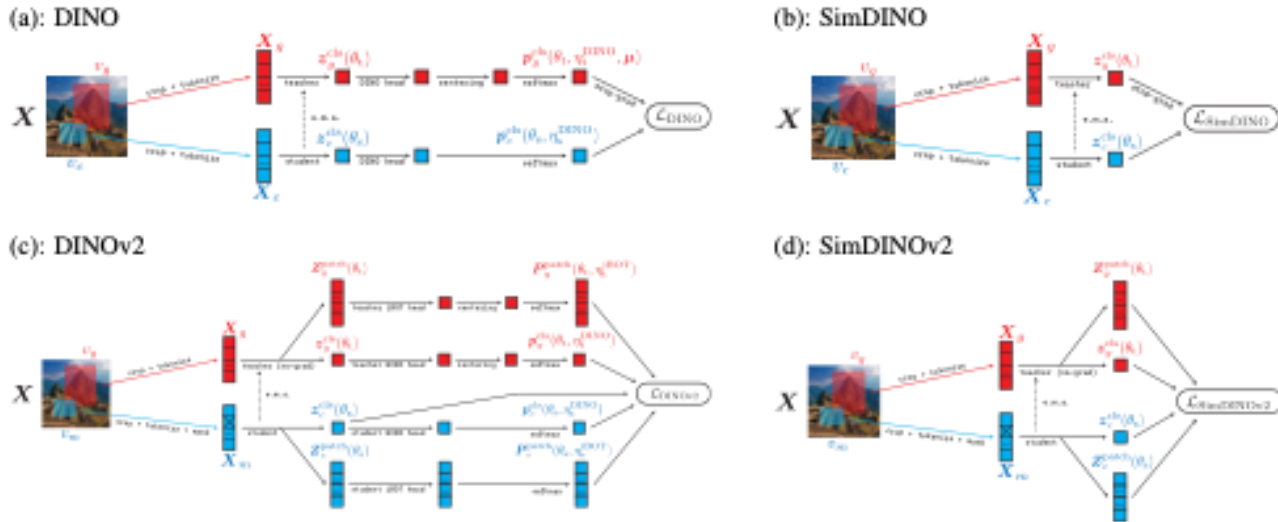
More Efficient
(Linear-Time Complexity, ICLR2025)



How to Learn: Better Networks from First Principles

No more trial and error to design better network architectures:

SimDINO: Simplifying DINO via Coding Rate Regularization [ICML2025]



Classification (ImageNet)

Method	Model	Epochs	k-NN	Linear
DINO	ViT-B	100	72.9	76.3
SimDINO	ViT-B	100	74.9	77.3
DINO	ViT-L	100	—	—
SimDINO	ViT-L	100	75.6	77.4
DINOv2	ViT-B	100	76.0	77.2
SimDINOv2	ViT-B	100	78.1	79.7
DINOv2	ViT-L	100	80.8	82.0
SimDINOv2	ViT-L	100	81.1	82.4
SwAV	ViT-S	800	66.3	73.5
MoCov3	ViT-B	300	—	76.7

Segmentation (Microsoft COCO)

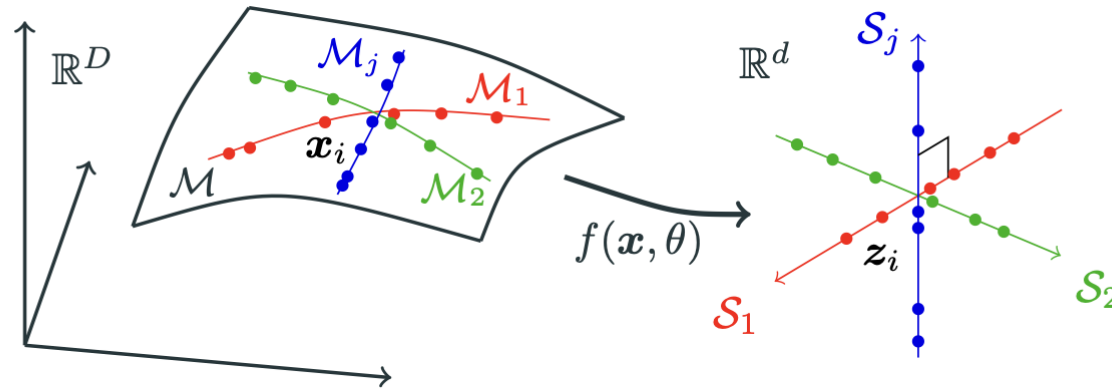
Method	Model	Detection ↑			Segmentation ↑		
		AP ₅₀	AP ₇₅	AP	AP ₅₀	AP ₇₅	AP
SimDINO	ViT-L/16	5.4	1.9	2.4	4.5	1.4	1.9
SimDINO	ViT-B/16	5.2	2.0	2.5	4.7	1.5	2.0
DINO	ViT-B/16	3.9	1.5	1.8	3.1	1.0	1.4
DINO	ViT-B/8	5.1	2.3	2.5	4.1	1.3	1.8



Hyperparameter		SimDINOv2	DINOv2	SimDINO	DINO
Model	Patch size	16			
	Register tokens	4		0	
	Pos-embedding anti-alias	True		False	
	Init layer scale	0.1	1e-5	-	
	Drop path rate	0.3		0.1	
	Weight normalize last layer	removed	True	removed	True
	Output prototypes K	removed	65536	removed	65536
Pipeline	Init EMA momentum	0.9	0.992	0.996	
	Centering temperature	removed	0.07	removed	0.07
	Warm-up temperature	removed	0.04	removed	0.04
	Warm-up temperature epochs	removed	30	removed	30
	iBOT sample prob.	0.5		-	
	iBOT mask ratio	0.1-0.5		-	
	iBOT head tying	False		-	
	Koleo loss weight	removed	0.1	-	
Data	Global crops scale	0.4 - 1			
	Local crops scale	0.05 - 0.4			
	Local crops number	10			
	Global crops size	224			
	Local crops size	96			
Optim.	Batch size	128x8		64x8	
	Epochs	100			
	Warm-up epochs	10			
	Freeze last layer epochs	removed	1	removed	1
	Learning rate	0.004		0.002	
	Layerwise lr decay	0.9		-	
	Weight decay	0.04			
	Weight decay end	0.4			
	Gradient clip	3.0		0.3	

How to Learn: Summary with a Comparison

No more trial and error to design better network architectures:

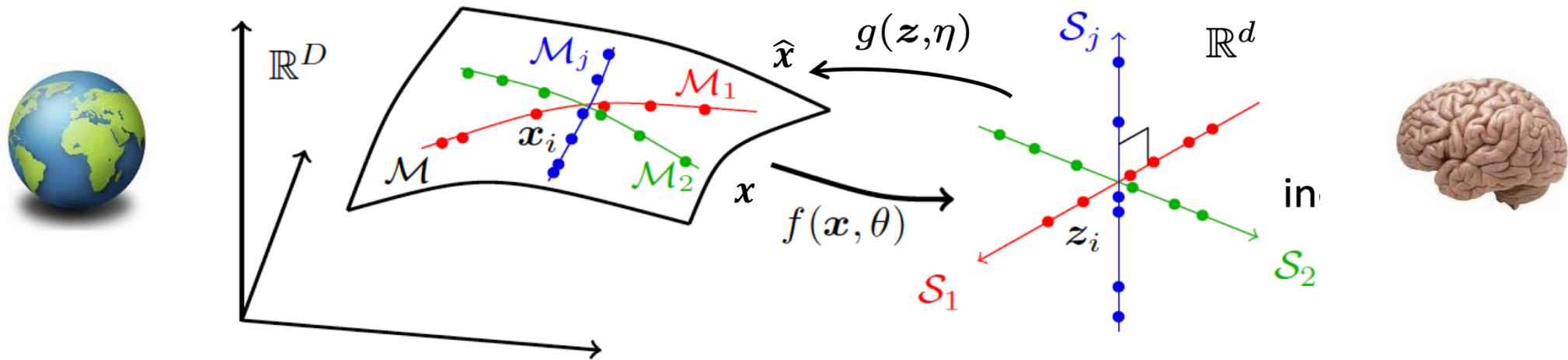
Deep
Representation
Learning



	 Black-box	 White-box
Objective	Input/label fitting	Information gain
Deep architecture	Empirical design	Iterative optimization
Representation	Hidden/latent	Incoherent subspaces/ dictionaries

Why Correct? (Consistency)

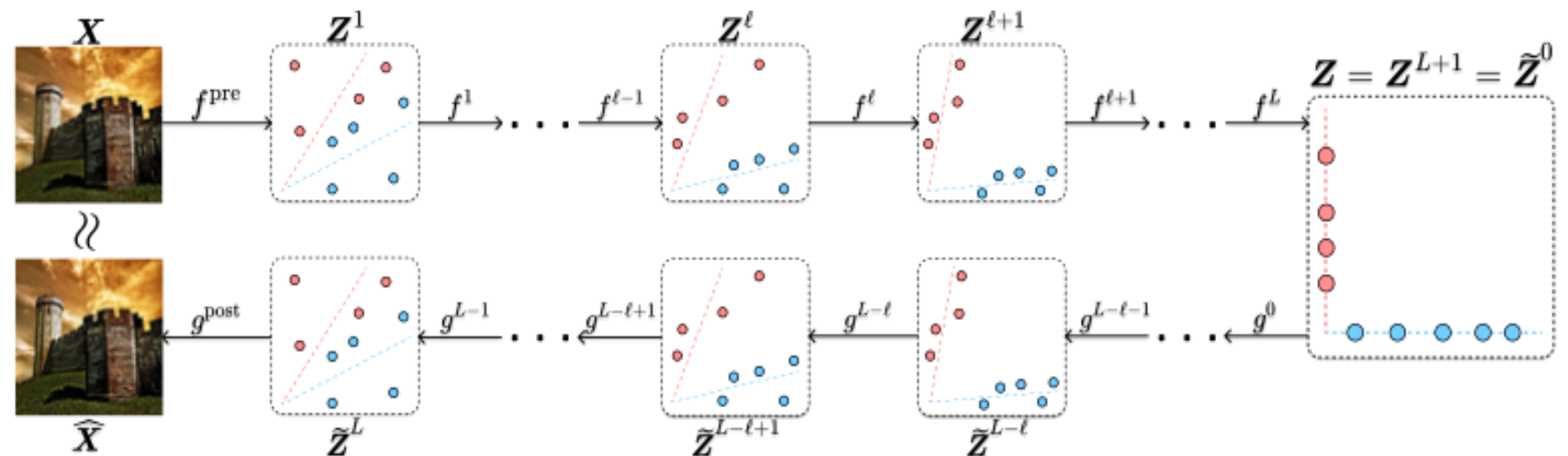
Bi-directional encoding and decoding (e.g., compression and generation)



$p(\hat{x}) \approx p(x)$? or $\hat{X} \approx X$?

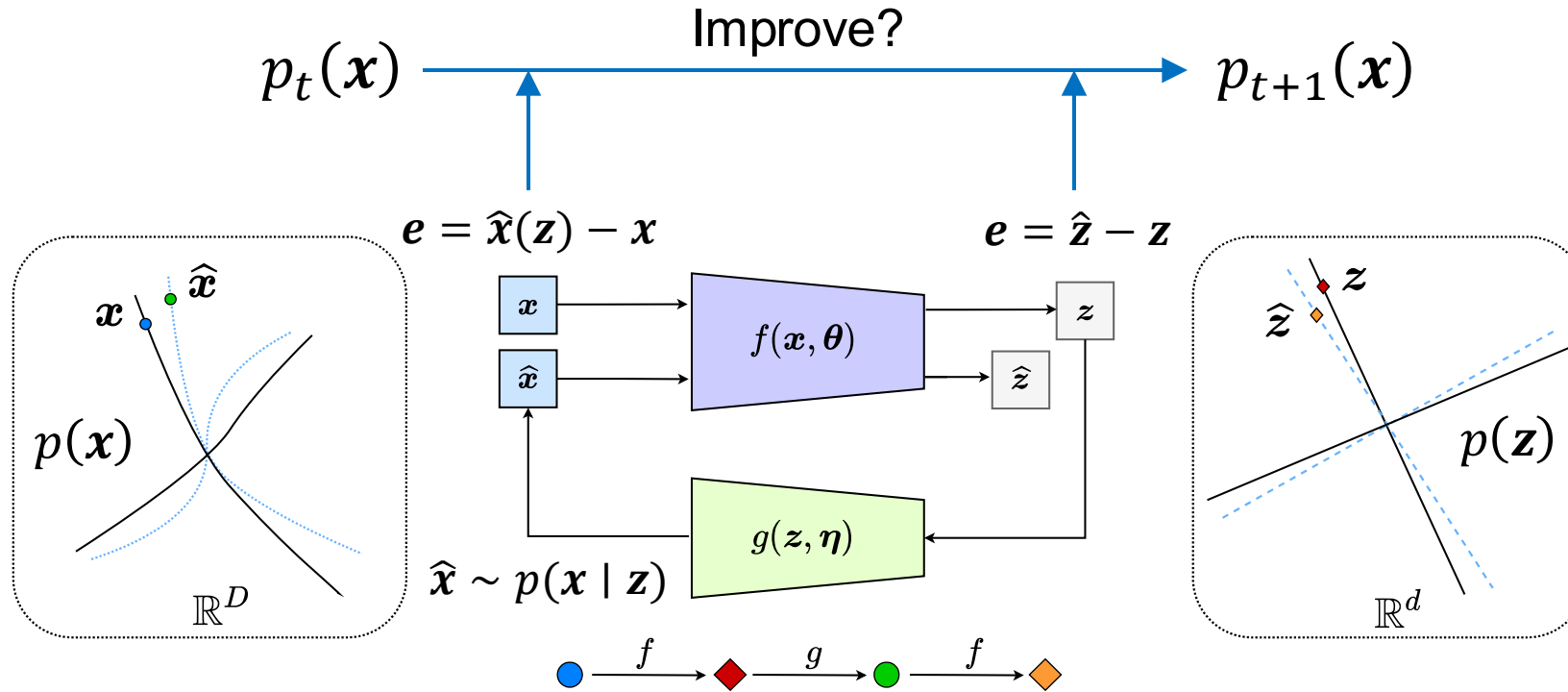
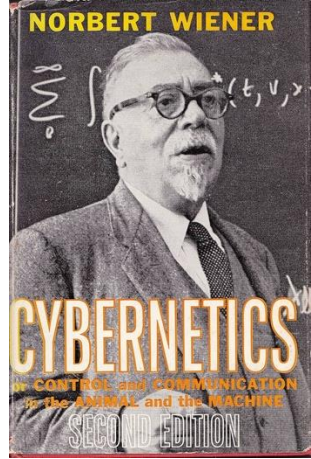
Autoencoding

(2024 Nobel Prize Physics)



Towards Autonomous Intelligence (AI 2.0)

How to self-learn a more consistent representation, continuously?



In nature, **all intelligent systems learn from closed-loop feedback!** (Cybernetics)

What is Intelligence?

Definition [Intelligence]: an intelligent system is one that has the mechanisms for **self-correcting** and **self-improving** its existing knowledge (or information).

$$\begin{aligned}\text{Knowledge} &= \int_0^t \text{Intelligence}, \\ \text{Intelligence} &= \frac{d}{dt} \text{Knowledge}.\end{aligned}$$

Any system without such mechanisms, however large, does not have any intelligence!



vs

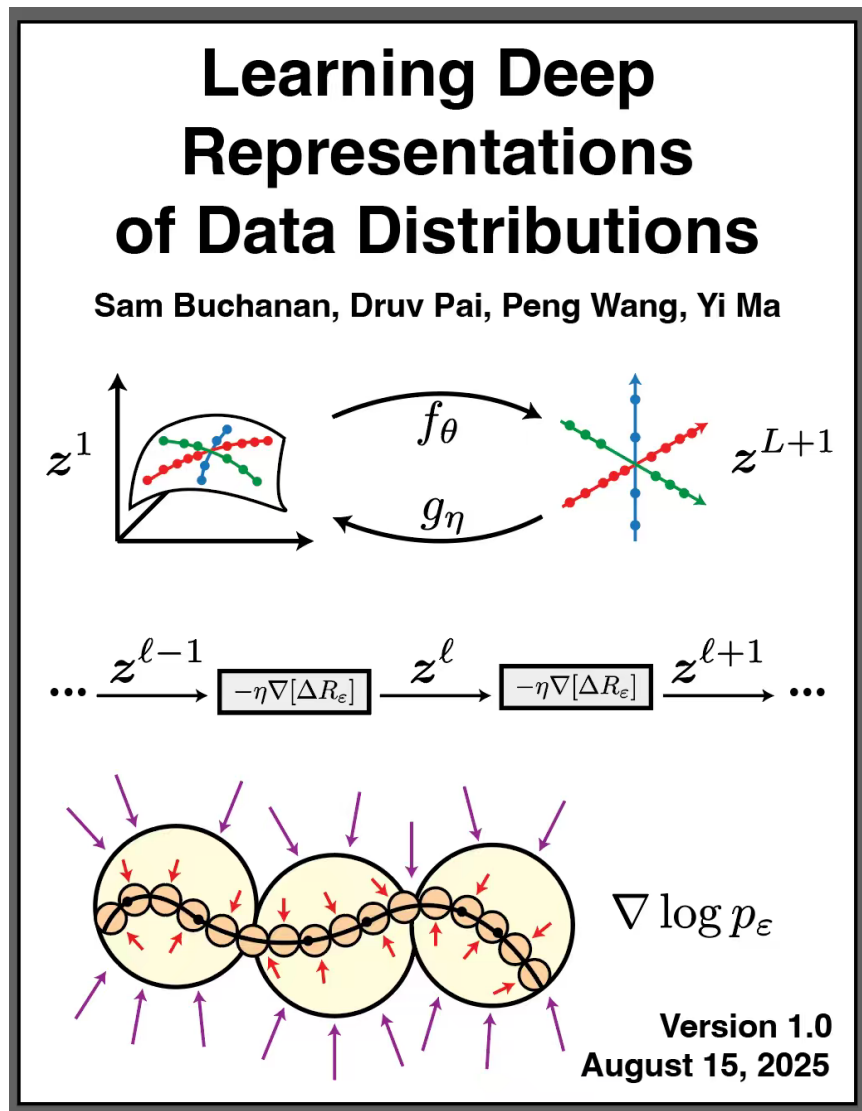


**Who has intelligence,
who has knowledge?**

Epilogue

A new open-source online textbook!

<https://ma-lab-berkeley.github.io/deep-representation-learning-book/>



Epilogue

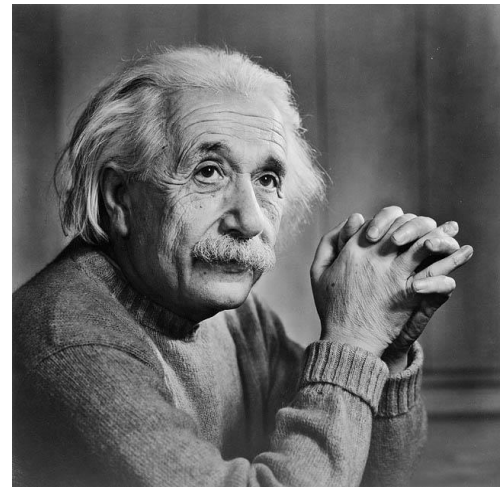
Seek a scientific and theoretical foundation for **Intelligence**:

- **what to learn?** *parsimony*
- **how to learn?** *compression*
- **why correct?** *consistency*

大道至简

*“Everything should be made as simple as possible,
but not any simpler.”*

-- Albert Einstein



Acknowledgement

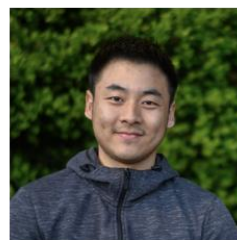
A Truly Multi-University and Multi-Disciplinary Effort from Academia!



Druv Pai
UC Berkeley



Sam Buchanan
TTI Chicago



Yaodong Yu
OpenAI, U. Maryland



Ziyang Wu
UC Berkeley



Shengbang Tong
New York University



Tianzhe Chu
Hong Kong University



Benjamin Haeffele
Johns Hopkins



Peng Wang
UMichigan



Simon Zhai
UC Berkeley, DeepMind



Jack Bai
UIUC



Ryan Chan
UPenn



HKU Musketeers Foundation
Institute of Data Science
香港大學同心基金數據科學研究院



Pursuing the Nature of Intelligence

Thanks



SCHOOL OF
**COMPUTING &
DATA SCIENCE**
The University of Hong Kong

