Introduction to generative AI

Panorama from the statistical learning point of view

Arnaud Breloy, 2025

le cnam

Machine learning

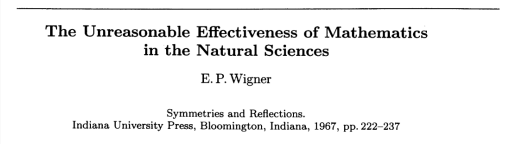
AI: field of computer science that studies or develops “intelligent” software

ML: develop **algorithms** to **solve problems** by **automatically processing data**
or “statistical learning”

A **broad field** that emerged from:

- **Informatics** computational science, data science
- **Applied mathematics** statistics, information theory, optimization
- **Applications** bio-informatics, signal processing, computer vision

A new relationship to data (1/3)



The Unreasonable Effectiveness of Mathematics in the Natural Sciences

E. P. Wigner

Symmetries and Reflections.
Indiana University Press, Bloomington, Indiana, 1967, pp. 222–237

Traditionally: **data** was made for **experts**

- **Scientific question** → **Experiments** → answer to an **hypothesis**

  The IPCC asking "Is global warming due to human activities?"

- **Mathematical models** → **Measures** → **Inversion**

  Meteorological data → yield forecasting

- Automated **classification** through **expert rules**

  Algorithmic transcription of "If # petals ≥ 5, then..."

Wigner (Nobel in φ), "*The unreasonable effectiveness of mathematics in the natural sciences*," Symmetries and Reflections, 1967.

A new relationship to data (2/3)

Current explosion of

- Available **data** sensors, measurements, experiments
- Data **dimension** pixels, monitored genes, sampling rate
- **Computing power**

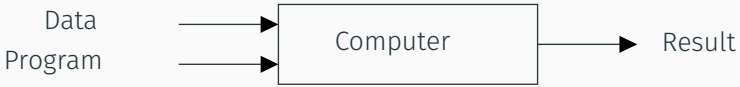
Paradigm shift

- **Learn models** directly from the **data**
- Gather **data first**, ask **questions later**

Halevy, Norvig, & Pereira (Google) "*The unreasonable effectiveness of data*," IEEE intelligent systems, 2009

A new relationship to data (3/3)

Expert system



Machine learning

Tools: **statistics**, **informatics**, **linear algebra**, **optimization**

Notation: supervised dataset

Supervised data appends a **labels** $\{\mathbf{y}_i\}_{i=1}^n \in (\mathcal{Y})^n$ to the **sample set** $\{\mathbf{x}_i\}_{i=1}^n \in (\mathbb{R}^d)^n$

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_i^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix} \leftarrow i^{\text{th}} \text{ sample} \quad \text{is paired with} \quad \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^\top \\ \mathbf{y}_2^\top \\ \vdots \\ \mathbf{y}_i^\top \\ \vdots \\ \mathbf{y}_n^\top \end{bmatrix} \leftarrow i^{\text{th}} \text{ label}$$

- **Classification**: $\mathcal{Y} = \{1, \ldots, m\}$ and $\mathbf{Y}$ stores the **class labels**
- **Regression**: $\mathcal{Y} = \mathbb{R}^d$ and $\mathbf{Y}$ stores the **latent variables** of a relationship $\mathbf{x} = g(\mathbf{y})$

Classification

Learn a **decision function** $f_\theta : \mathbb{R}^d \to \mathcal{Y}^1$ from **data** $\mathbf{X} \in \mathbb{R}^{n \times d}$ with **class labels** $\mathbf{Y} \in \mathcal{Y}^1$

Attribute classes to **new samples** $\hat{\mathbf{y}} = f(\mathbf{x})$

"**Learning the model**" is finding θ so that f_θ produces the right boundaries

“Models” in ML

A **model** is a **function**

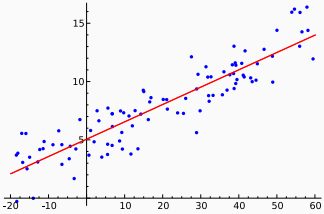
$$f_\theta : \begin{array}{ccc} \mathbb{R}^d & \to & \mathbb{R}^{d'} \\ \mathbf{x} & \mapsto & \hat{\mathbf{y}} = f_\theta(\mathbf{x}) \end{array}$$

with tunable set of **parameters** θ

Example: 1D linear function

$$\hat{y} = ax + b$$

with parameters $\theta = \{a, b\}$

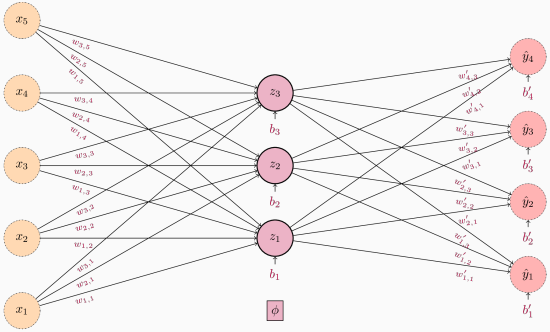


How to chose θ?

Neural networks are models

Multi layer preceptron (MLP):

$$f_{\boldsymbol{\theta}}(\mathbf{x}) = \phi_{\text{out}}(\mathbf{W}^{\text{out}}\phi_{\text{in}}(\mathbf{W}^{\text{in}}\mathbf{x} + \mathbf{b}^{\text{in}}) + \mathbf{b}^{\text{out}}) \qquad \text{with} \qquad \boldsymbol{\theta} = \{\mathbf{W}^{\text{in}}, \mathbf{b}^{\text{in}}, \mathbf{W}^{\text{out}}, \mathbf{b}^{\text{out}}\}$$



ϕ: **activation function**

e.g., ReLU

Generalizes to **more layers**

$$\mathbf{h}^{(k+1)} = \phi^{(k)}(\mathbf{W}^{(k)}\mathbf{h}^{(k)} + \mathbf{b}^{(k)})$$

Other neural networks are models suited to some specific data

Signals and **Images**

- Convolutional neural networks (**CNN**)
- Vision transformers (**ViT**)

Data on graphs

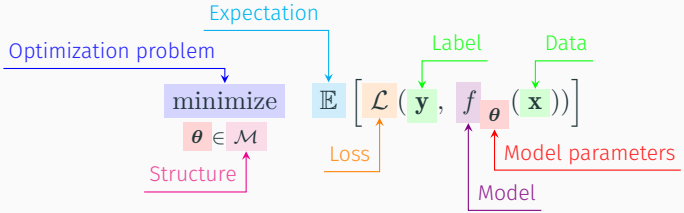
- Graph neural networks (**GNNs**)

Time series

- Residual neural networks (**RNN**)
- **LSTM**, **GRU**
- **Transformers**

Still, how to chose θ?

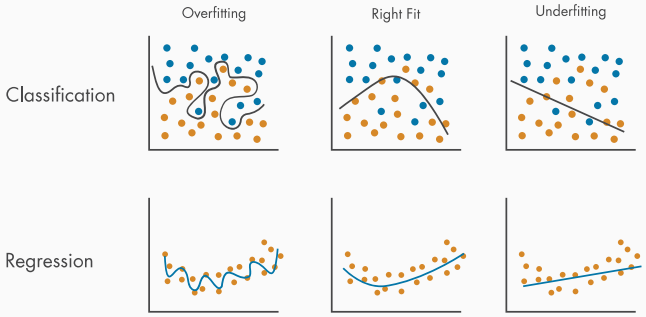
Supervized ML optimization philosophy



- **Design** prediction model $\hat{\mathbf{y}} = f_{\boldsymbol{\theta}}(\mathbf{x})$ and loss $\mathcal{L}$
- **Learn** the model parameters θ approx. $\mathbb{E}$ with data at hand + solve the optimization problem
- **Apply** model to new data

Capacity, generalization, over-fitting, ...

Expected performance is evaluated on a training set, does it work on **new unseen data** ?



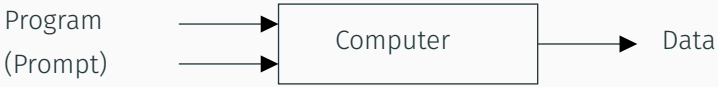
Possibility of **over-fitting**: trade off between the model **capacity** and **generalization**

ML literature provides **methodologies** to **properly control this**

Generative models: an even weirder relationship to data

Learning step

Sampling step



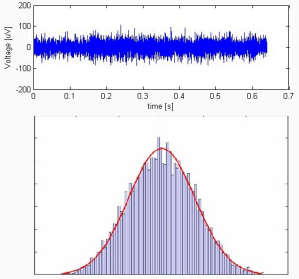
Tools: all techniques adopt a **statistical** point of view $\mathbf{x} \sim p(\mathbf{x})$

Simple examples of generative modeling

Gaussian distribution $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

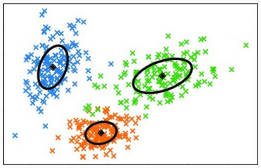
$$p(\mathbf{x}) = (2\pi)^{-d/2} \det(\boldsymbol{\Sigma})^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

- Learning: **estimate** parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$
- Generating: **sample** from $\mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$



Gaussian mixture models (**GMM**)

$$\mathbf{x}|c_k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \text{ with } p(c_k) = \pi_k \text{ and } \sum_{k=1}^{K} \pi_k = 1 \quad (1)$$



Estimating and sampling from this is easy, but limiting

Generative modeling with neural networks

We want to approximate the **data density** (p.d.f.) p_{data} by a network p_θ

p_θ should be **learned from samples** $\{\mathbf{x}_i\}_{i=1}^n$

This is **very difficult**

- Assume we learn $f_\theta : \mathbb{R}^d \to \mathbb{R}^+$, a p.d.f. can be obtrained by **normalizing** it

$$p_\theta(\mathbf{x}) = \frac{e^{f_\theta(\mathbf{x})}}{Z(\theta)} \quad \text{with} \quad Z(\theta) = \int e^{f_\theta(x)} \mathrm{d}\mathbf{x}$$

  we cannot reasonably access $Z(\theta)$

- Even if provided with p_θ, how to sample from it efficiently?

All methods are **proposals to circumvent this**

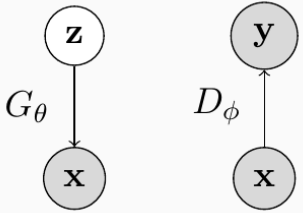
Some landmarks of modern generative AI

- Statistical models (GMM, PPCA), Bayesian sampling methods, etc.
- **2012**: Neural networks are back on the spotlight
- **2014**: GAN
- **2015**: VAE
- **2017**: Transformers
- **2019**: Score-based diffusion
- **2020**: Diffusion models
- **2021**: CLIP
- **2022**: Dall-E, Stable Diffusion, ChatGPT
- **2023**: Flow-matching
- And things are a bit crazy atm...

Some landmarks of modern generative AI

- Statistical models (GMM, PPCA), Bayesian sampling methods, etc.
- **2012**: Neural networks are back on the spotlight
- **2014**: **Generative adversarial networks**
- **2015**: VAE
- **2017**: Transformers
- **2019**: Score-based diffusion
- **2020**: Diffusion models
- **2021**: CLIP
- **2022**: Dall-E, Stable Diffusion, ChatGPT
- **2023**: Flow-matching
- And things are a bit crazy atm...

Generative adversarial networks



- **Generator** G_θ: outputs $\mathbf{x}$ from input noise $\mathbf{z}$
- **Discriminator** D_ϕ: detects if sample is real or generated
- Learning by **min-max game**

$$\min_\theta \max_\phi \quad V(G_\theta, D_\phi) = \underbrace{\mathbb{E}_{\mathbf{x}\sim\mathbf{p}_{\text{data}}}[\log D_\phi(\mathbf{x})]}_{\nearrow \text{ score real data}} + \underbrace{\mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D_\phi(G_\theta(\mathbf{z})))]}_{\searrow, \nearrow \text{ score generated data}}$$

can be theoretically linked to a distance between p_{G_θ} and p_{data}!

Generative adversarial networks

Sampling

- Draw random noise $\mathbf{z}$
- Apply model $\hat{\mathbf{x}} = G_\theta(\mathbf{z})$
- Repeat for new samples

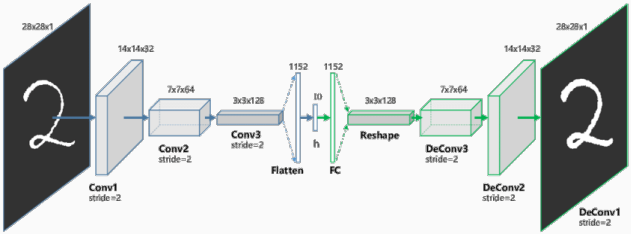
In practice

- Very unstable and hard to train
- Improvement brought over the year (WGAN)
- Quite realistic results

Some landmarks of modern generative AI

- Statistical models (GMM, PPCA), Bayesian sampling methods, etc.
- **2012**: Neural networks are back on the spotlight
- **2014**: GAN
- **2015**: **Variational auto-encoders**
- **2017**: Transformers
- **2019**: Score-based diffusion
- **2020**: Diffusion models
- **2021**: CLIP
- **2022**: Dall-E, Stable Diffusion, ChatGPT
- **2023**: Flow-matching
- And things are a bit crazy atm...

Auto-encoders

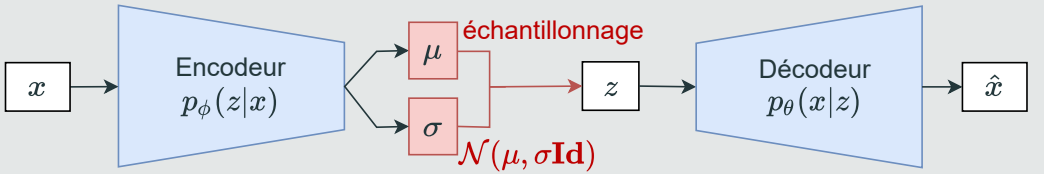


- Encoder $\mathcal{E} : \mathbb{R}^d \to \mathbb{R}^{d'}$
- Decoder $\mathcal{D} : \mathbb{R}^{d'} \to \mathbb{R}^d$
- Learning to compress-decompress

$$\theta^* = \text{argmin}_\theta \quad \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} \left[\| \mathcal{D}(\mathcal{E}(x)) - x \| \right]$$

Variational auto encoder

Gaussian VAE



$$\mathcal{L}(\theta, \phi; x) = \underbrace{\mathbb{E}_{q_\phi(z|x)} \|\hat{x} - x\|}_{\text{reconstruction loss}} + \beta \underbrace{\mathrm{KL}\left(q_\phi(z|x) \,||\, p_\theta(z)\right)}_{\text{regularizing latent space}}$$

Can be theoretically linked to maximum likelihood estimation!

Sampling from VAE

Sampling

- Draw $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
- Decode $\hat{\mathbf{x}} = \mathcal{D}(\mathbf{z})$
- Repeat for new samples

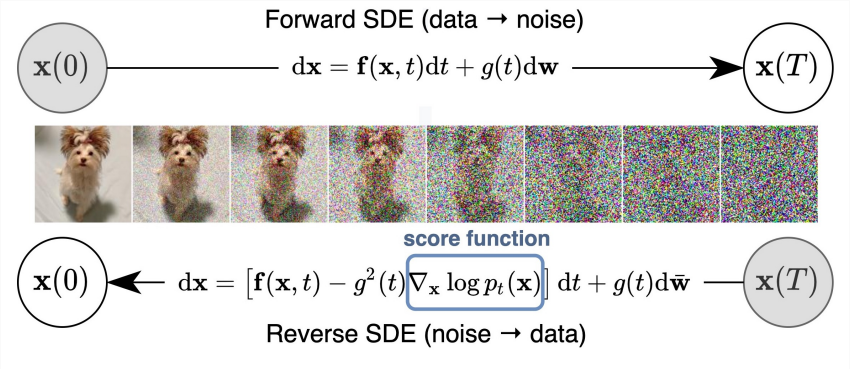
In practice

- Much more stable than GANs
- Less impressive sometimes (blurry)
- GANs were still “best” sota in 2019

Some landmarks of modern generative AI

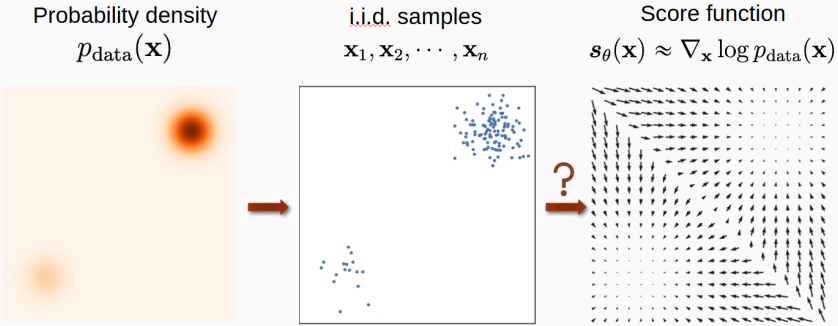
- Statistical models (GMM, PPCA), Bayesian sampling methods, etc.
- **2012**: Neural networks are back on the spotlight
- **2014**: GAN
- **2015**: VAE
- **2017**: Transformers
- **2019**: Score-based diffusion
- **2020**: **Diffusion models**
- **2021**: CLIP
- **2022**: Dall-E, Stable Diffusion, ChatGPT
- **2023**: Flow-matching
- And things are a bit crazy atm...

Diffusion models: “a long sequence of small denoisers”



Theoretical links with **score estimation** and **Langevin diffusion**

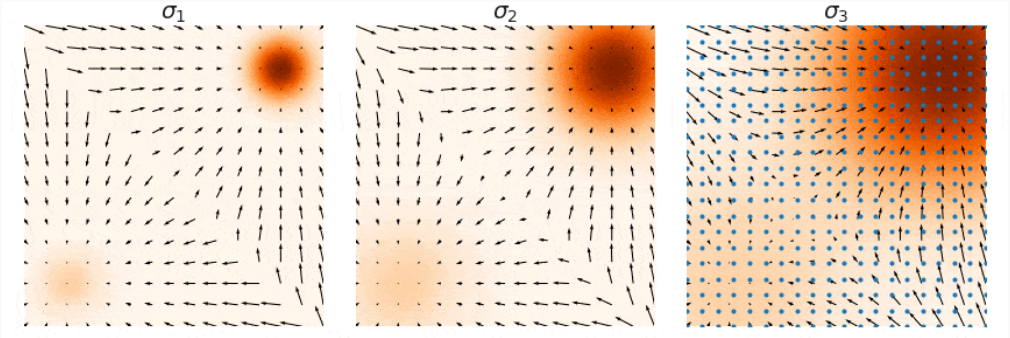
Score estimation



Done by **learning to denoise** at different noise levels $\{\sigma_\ell\}_{\ell=1}^{L}$

$$\mathcal{L}(\theta) \quad = \quad \frac{1}{L}\sum_{\ell=1}^{L} \lambda(\sigma_\ell)\mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[||\boldsymbol{s}_\theta(\mathbf{x}+\sigma_\ell\mathbf{z},\sigma_\ell)-\mathbf{z}/\sigma_\ell||_2^2\right]$$

Sampling: annealed Langevin dynamics $\mathbf{x}_{t-1} \leftarrow \mathbf{x}_t + \frac{\epsilon}{2} s_\theta(\mathbf{x}, \sigma_t) + \epsilon \mathbf{z}_t$



Diffusion models

Sampling

- Draw $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
- Apply the sequence of denoisers
- Repeat for new samples

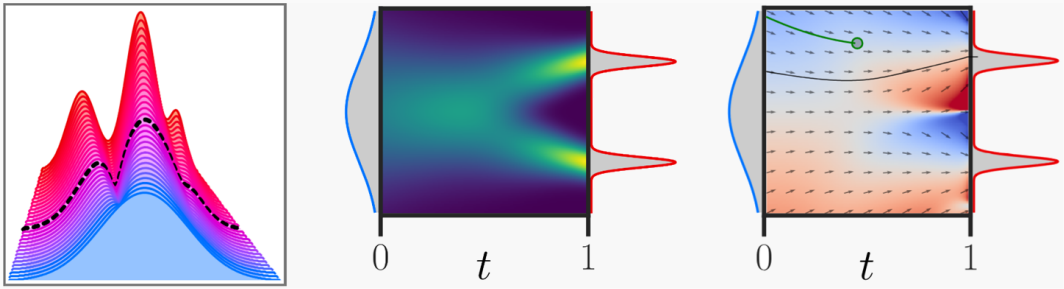
In practice

- Much more stable than GANs, more detailed than VAE
- State of the art between 2019-2023
- Deeper links with likelihood $\rightarrow$ useful for other applications
- Still of use atm

Some landmarks of modern generative AI

- Statistical models (GMM, PPCA), Bayesian sampling methods, etc.
- **2012**: Neural networks are back on the spotlight
- **2014**: GAN
- **2015**: VAE
- **2017**: Transformers
- **2019**: Score-based diffusion
- **2020**: Diffusion models
- **2021**: CLIP
- **2022**: Dall-E, Stable Diffusion, ChatGPT
- **2023**: **Flow-matching**
- And things are a bit crazy atm...

Flow matching - goal



We seek a velocity field u_t such that

$$\frac{d}{dt}\varphi_t(x) = u_t(\varphi_t(x)), \quad \forall t \in [0,1] \tag{2}$$

for which $\varphi_0 = \pi$ and $\varphi_0 = p_{data}$

Diffusion models

Sampling

- Draw $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
- Perform

$$\hat{x}_1 = \varphi_{t=1}(x_0) = \text{EDO}^{v_\theta}(x_0, 0 \to 1)$$

where $\text{EDO}^{v_\theta}(\cdot, t_0 \to t_1)$ solves (2) from t_0 to t_1 with $v_\theta(t, \cdot)$ instead of u_t

- Repeat for new samples

In practice

- Recently outperformed the rest
- Links with optimal transport $\to$ useful for other applications

A quick word on text generation

Architectures dedicated to **time-series**

- From statistics: **auto-regressive** models, **Markov-chains**, etc.
- **RNN**, **LSTM**, **GRU**, **Transformers**

Texts are time-series!

- Text embedding allow to map text to vectors (word2vec 2013)
- Train models to predict next vectors from input (self-supervized)
- Large corpus to train from (digital library, web scrapping)
- Basis of recent commercial LLMs

Example: GPT means “generative pre-trained transformer”

About text generation and LLMs

Many recent advances in natural language processing (**NLP**)

Pushed to **large scales** + **commercial release** of large language models (**LLMs**) 2022

Adopted quickly, and re-sold ad nauseam everywhere...

Hopefully, this introduction showed that we should not restrict “AI” to LLMs

Generative modeling is **more than generation**

- Parameter estimation in complex systems
- Data mining, outlier detection
- Signal and images denoising, restauration