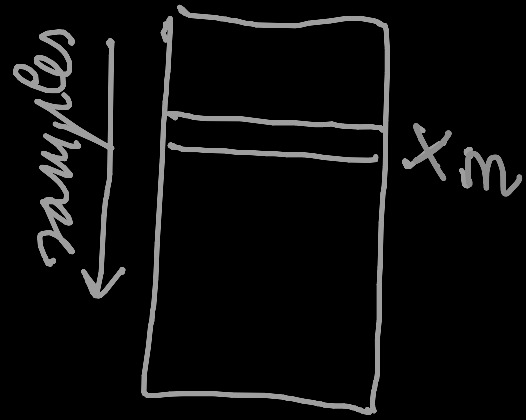
Unsupervised Learning (UL)

$\mathbf{X} = \{x_1, x_2, \dots\}$ data



$\cancel{y}$ No labels

$p(\mathbf{x})$ probability distribution function (PDF) of data, usually <u>not</u> known

$\mathbf{Z} = \{z_1, z_2, \dots\}$ hidden, latent variables

$\boldsymbol{\theta} = \{\theta_1, \theta_2, \dots\}$ parameters of the model

$$\boxed{p_\theta(x)}$$

goal of UL:

$$\boxed{\text{represent “true” } p(x) \text{ of data by approximate } p_\theta(x)}$$

generative models

- generating "fantasy" data x
- denoising
- filling missing data
- discrimination

Key quantities: (information theory)

$$S_p = -\sum_i p(x_i) \log p(x_i)$$ Shannon entropy

Key quantities:

$$S_p = -\sum_i p(x_i) \log p(x_i)$$ Shannon entropy

$$D_{KL}(p \| p') = \sum_i p(x_i) \log \frac{p(x_i)}{p'(x_i)}$$ Kullback Leibler divergence

Key quantities:

$$S_p = -\sum_i p(x_i) \log p(x_i)$$ Shannon entropy

$$D_{KL}(p \| p') = \sum_i p(x_i) \log \frac{p(x_i)}{p'(x_i)}$$ Kullback Leibler divergence (relative entropy)

- $D_{KL} \geq 0$ (using $\log \frac{1}{r} \geq 1 - r$)
- $D_{KL}(p \| p') \neq D_{KL}(p' \| p)$

Why physics in UL ?

- <u>similar</u> <u>problems</u>
  - variational free energy minimiz.
  - Jaynes MaxEnt
  - Disordered systems, spin glasses, ...

Why physics in UL ?

- similar problems

- useful setup

$$\boxed{\begin{aligned} p(x) &= \frac{1}{Z} e^{-\beta E(x)} \\ \log p &= -\beta E(x) - \ln Z \end{aligned}}$$

Why physics in UL ?

- similar problems

- useful setup $\qquad p(x) = \frac{1}{Z} e^{-\beta E(x)}$

- training by physics methods, Monte Carlo (MC)

  NOT via backpropagation, ~~Keras~~ (RBM) however:

physics $\langle f \rangle_{obs}$ conceptual average with respect to model

$\downarrow$

UL $\langle f \rangle_{data}$ empirical average from data

$\langle f \rangle_{model}$ will be compared to $\langle f \rangle_{data}$

however:

physics $\langle f \rangle_{obs}$ conceptual average with respect to model

$\downarrow$

UL $\langle f \rangle_{data}$ empirical average from data

- overfitting
- heterogeneity in precision

log-likelihood maximization

by varying pars θ

review (189)

$$\mathcal{L}(\theta) = \langle \log p_\theta(x) \rangle_{data}$$

$$= -\langle E_\theta(x) \rangle_{data} - \log Z_\theta$$

$\boxed{\beta = 1}$

NO data in the partition function!

(only parameters θ and possible x's)

$$Z_\theta = \sum_x p_\theta(x)$$

minus

log-likelihood ~~maximization~~ minimization

$$-\mathcal{L}(\theta) = -\langle \log p_\theta(x) \rangle_{data}$$

$$= +\langle E_\theta(x) \rangle_{data} + \log Z_\theta$$

energy is minimized

NO data in the partition function!

minus

log-likelihood ~~maximization~~ minimization

$$-\mathcal{L}(\theta) = -\langle \log p_\theta(x) \rangle_{data} \tag{191}$$

$$= +\langle E_\theta(x) \rangle_{data} + \log Z_\theta + E_\theta^{reg}$$

eventually, also regularization term

$E_\theta^{reg} = \Lambda \sum_i |\theta_i|$ (LASSO) / $E_\theta^{reg} = \Lambda \sum_i |\theta_i|^2$ (ridge)

Computing gradients

to minimize $-\mathcal{L}(\theta)$ via e.g. stochastic gradient descent

define "operators"

$$O_j = \partial_{\theta_j} E_\theta(x)$$

role of minus force

(193)

Computing gradients

to minimize $-\mathcal{L}(\theta)$ via e.g. stochastic gradient descent

define "operators"

$$O_j = \partial_{\theta_j} E_\theta(x)$$

role of minus force

(193)

$$\partial_{\theta_j}\left(-\mathcal{L}(\theta)\right) = \left\langle \partial_{\theta_j} E_\theta(x) \right\rangle_{\text{data}} + \partial_{\theta_j} \log Z_\theta$$

Computing gradients

to minimize $-\mathcal{L}(\theta)$ via e.g. stochastic gradient descent

define "operators"

$$O_j = \partial_{\theta_j} E_\theta(x)$$

role of minus force

(193)

$$\partial_{\theta_j}\left(-\mathcal{L}(\theta)\right) = \left\langle \partial_{\theta_j} E_\theta(x) \right\rangle_{\text{data}} + \partial_{\theta_j} \log Z_\theta$$

(*)

(195)

$$= \left\langle O_j(x) \right\rangle_{\text{data}} - \left\langle O_j(x) \right\rangle_{\text{model}}$$

GIVEN

$$Z_\theta = \sum_x p_\theta(x) = \sum_x e^{-E_\theta(x)}$$

(*)

$$\partial_{\theta_j} \log Z_\theta = \frac{1}{Z_\theta} \sum_x \left(-\partial_{\theta_j} E_\theta(x)\right) e^{-E_\theta(x)}$$

$$= -\left\langle \partial_{\theta_j} E_\theta(x) \right\rangle_{model}$$

$$= -\left\langle O_j \right\rangle_{model}$$

Computing gradients

$$\partial_{\theta_j}\left(-\mathcal{L}(\theta)\right) = \left\langle \partial_{\theta_j} E_\theta(x) \right\rangle_{\text{data}} + \partial_{\theta_j} \log Z_\theta$$

$$= \underbrace{\left\langle O_j(x) \right\rangle_{\text{data}}}_{\text{positive phase of the gradient}} - \underbrace{\left\langle O_j(x) \right\rangle_{\text{model}}}_{\text{negative phase} \ldots}$$

(contains all info on data) (only model)

Computing gradients

$$\partial_{\theta_j}\left(-\mathcal{L}(\theta)\right) = \left\langle \partial_{\theta_j} E_\theta(x) \right\rangle_{\text{data}} + \partial_{\theta_j} \log Z_\theta$$

$$= \left\langle O_j(x) \right\rangle_{\text{data}} - \left\langle O_j(x) \right\rangle_{\text{model}}$$

Nice physical interpretation:

optimum when zero "force", i.e.
when expectation from model
equals " " data

Computing gradients

$$\partial_{\theta_j}\left(-\mathcal{L}(\theta)\right) = \left\langle \partial_{\theta_j} E_\theta(x) \right\rangle_{data} + \partial_{\theta_j} \log Z_\theta$$

$$= \left\langle O_j(x) \right\rangle_{data} - \left\langle O_j(x) \right\rangle_{model}$$

- only in some Gaussian cases we have analytic solutions
- in general, <u>intractable</u> likelihood

to evaluate

$$\langle f(x) \rangle_{model} = \sum_{x} p_\theta(x) \, f(x)$$

to evaluate

$$\langle f(x) \rangle_{model} = \sum_{x} p_\theta(x)\, f(x) \approx \frac{\sum_{x'_i} f(x'_i)}{\sum_{x'_i} 1}$$

draw samples x'_i from the model according to p_θ, following Monte Carlo procedure

to evaluate

$$\langle f(x) \rangle_{model} = \sum_x p_\theta(x)\, f(x) \approx \frac{\sum_{x'_i} f(x'_i)}{\sum_{x'_i} 1}$$

draw samples x'_i
from the model
according to p_θ,
following Monte Carlo procedure

x'_i , "fantasy particle"

to evaluate

$$\langle f(x) \rangle_{model} = \sum_x p_\theta(x)\, f(x) \approx \frac{\sum_{x'_i} f(x'_i)}{\sum_{x'_i} 1}$$

draw samples x'_i from the model according to p_θ, following Monte Carlo procedure

Normalization

x'_i , "fantasy particle"

log-derivative trick

to compute gradient of any $f(x)$

$$\partial_{\theta_j} \langle f(x) \rangle_{\text{model}} = \sum_i \partial_{\theta_j} p_\theta(x_i)\, f(x_i) \qquad \left(\partial_{\theta_j} = \frac{\partial}{\partial \theta_j} \right)$$

log-derivative trick

to compute gradient of any $f(x)$

$$\partial_{\theta_j} \langle f(x) \rangle_{\text{model}} = \sum_i \partial_{\theta_j} p_\theta(x_i)\, f(x_i)$$

$$= \left\langle \partial_{\theta_j} \log p_\theta(x)\ f(x) \right\rangle_{\text{model}}$$

log-derivative trick

to compute gradient of any $f(x)$

$$
\begin{aligned}
\partial_{\theta_j} \langle f(x) \rangle_{\text{model}} &= \sum_i \partial_{\theta_j} p_\theta(x_i)\, f(x_i) \\
&= \langle \underbrace{\partial_{\theta_j} \log p_\theta(x)}\; f(x) \rangle_{\text{model}} \\
&= \langle O_j(x)\, f(x) \rangle_{\text{model}}
\end{aligned}
$$

log-derivative trick

to compute gradient of any $f(x)$

$$\partial_{\theta_j} \langle f(x) \rangle_{\text{model}} = \sum_i \partial_{\theta_j} p_\theta(x_i)\, f(x_i)$$

$$= \left\langle \partial_{\theta_j} \log p_\theta(x)\; f(x) \right\rangle_{\text{model}}$$

$$= \langle O_j(x)\, f(x) \rangle_{\text{model}}$$

$$\approx \frac{\sum_{x'_i} O_j(x'_i)\, f(x'_i)}{\sum_{x'_i} 1} \qquad (198)$$

Summary of training procedure

goal: fit $\{\theta\}$ of model $p_\theta(x) = \frac{1}{Z_\theta} e^{-E_\theta(x)}$

train:
1) read minibatch B of data, $\{x\}_B$
2) generate fantasy particles $\{x'\}_B \sim p_\theta(x)$ with MC
3) compute gradients (195) (for B)
4) update $\mathcal{L}$ with gradient descent

Summary of training procedure

goal: fit $\{\theta\}$ of model $p_\theta(x) = \frac{1}{Z_\theta} e^{-E_\theta(x)}$

train:
1) read minibatch B of data, $\{x\}_B$
2) generate fantasy particles $\{x'\}_B \sim p_\theta(x)$ with MC
3) $\boxed{\text{compute gradients}}$ (195) (for B)
4) update $\mathcal{L}$ with gradient descent

NOT ANALYTICALLY

(as in backpropagation)