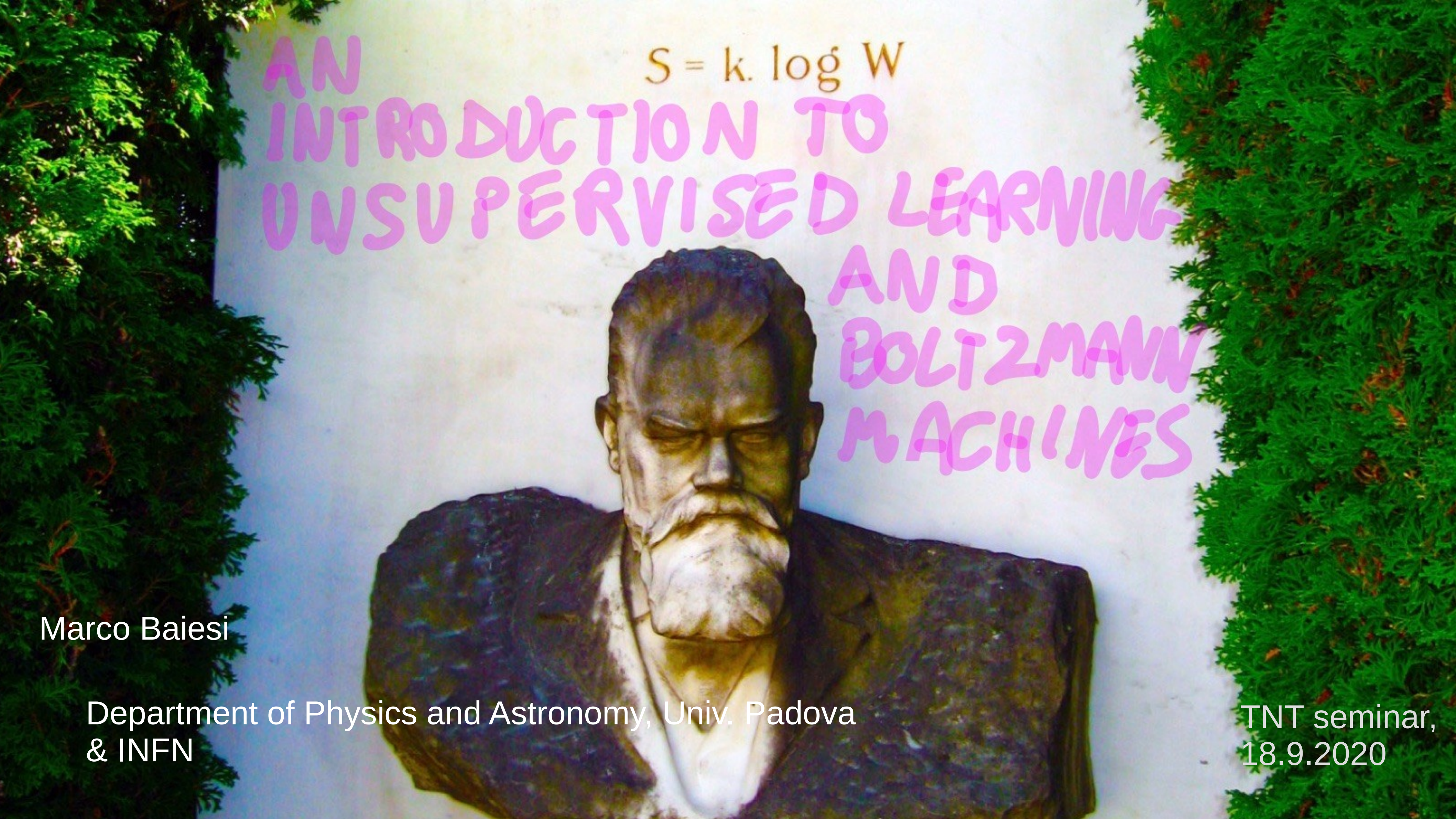
AN INTRODUCTION TO UNSUPERVISED LEARNING AND BOLTZMANN MACHINES



Marco Baiesi

Department of Physics and Astronomy, Univ. Padova
& INFN

TNT seminar,
18.9.2020

Physics of Data, UNIPD

- Now starting 3rd year of master
- Growing number of students
  - Complex systems
  - high energy
  - astrophysics

Laboratory of Computational Physics B

- 24 hours theory & exercises
- This seminar is extracted from there
- Review:

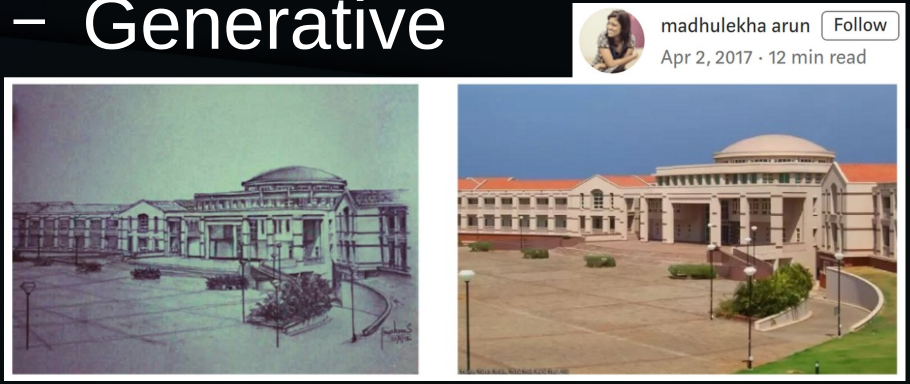
Mehta et al, “A high-bias, low-variance introduction to Machine Learning for physicists” Ph.Rep.810 (2019) 1–124

- 24 hours supervised projects in groups

Learning

- Supervised
  - Deep neural networks, regression,...
  - Labeled data
  - Optimization
  - Discriminative: obvious estimate of performaces

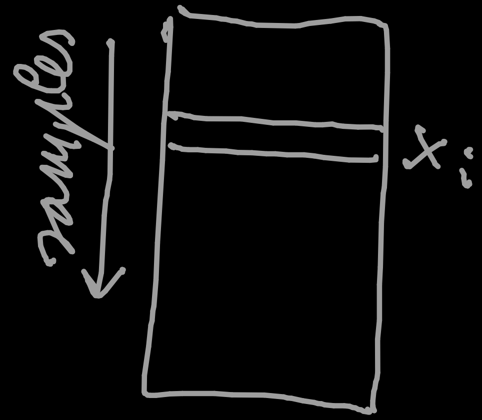
- Unsupervised
  - Boltzmann machines, variational autoencoders, generative adversarial nets
  - No labels
  - Generative



05-1

Unsupervised Learning (UL)

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



$\cancel{y}$ NO labels

$p(x)$ probability distribution function (PDF) of data, usually not 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?

- similar problems
  - variational free energy minimiz.
  - Jaynes MaxEnt
  - Disordered systems, spin glasses, ...

Why physics in UL ?

- similar problems

- useful setup

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

Why physics in UL?

- similar problems

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

- training by physics methods, Monte Carlo (MC) → (RBM)
  NOT via backpropagation, ~~Keras~~

however:

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

$\downarrow$

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

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

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

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

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

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)

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

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

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

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

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

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

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

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

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, intractable 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 $f(x)$

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

log-derivative trick

to compute gradient of any $f(x)$

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

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

log-derivative trick

to compute gradient of any $f(x)$

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

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

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

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

05-2

Latent variables and Restricted Boltzmann Machines

Latent variables

enhance expressive power of generative models by encoding complex correlations between data

Latent variables

enhance expressive power of generative models
by encoding complex correlations between data

$z \longrightarrow h$ for "hidden" in this case

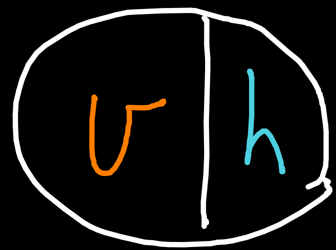
$x \longrightarrow v$ for "visible"

Latent variables

enhance expressive power of generative models by encoding complex correlations between data

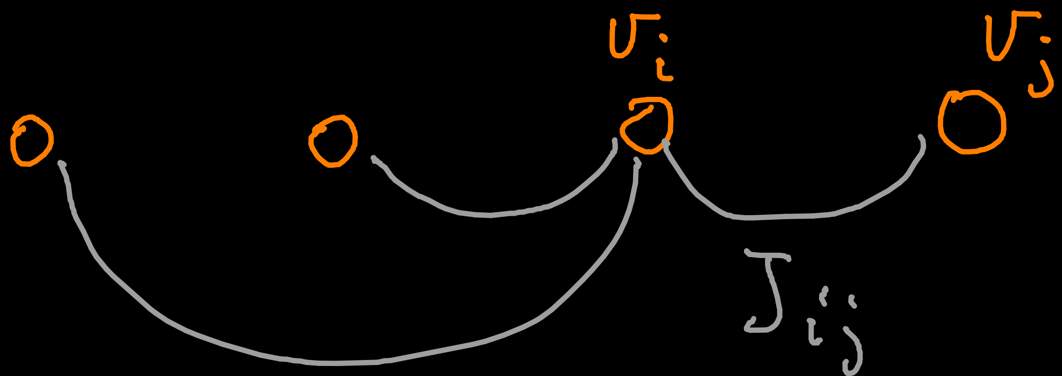
$z \to \underline{h}$ for "hidden" in this case

$x \to \underline{v}$ for "visible"



$v \cup h$ system

- spin systems (physics again relevant for UL ...)

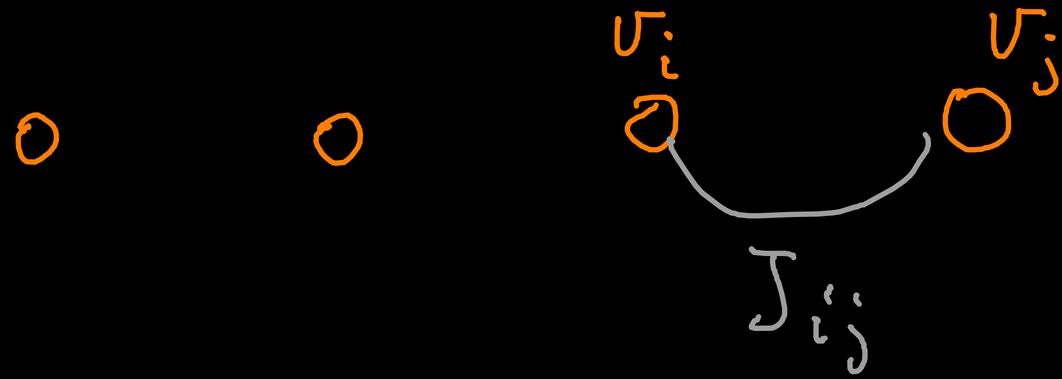


i = index of the spin (j ..)

mean field : all couplings $J_{ij} \neq 0$

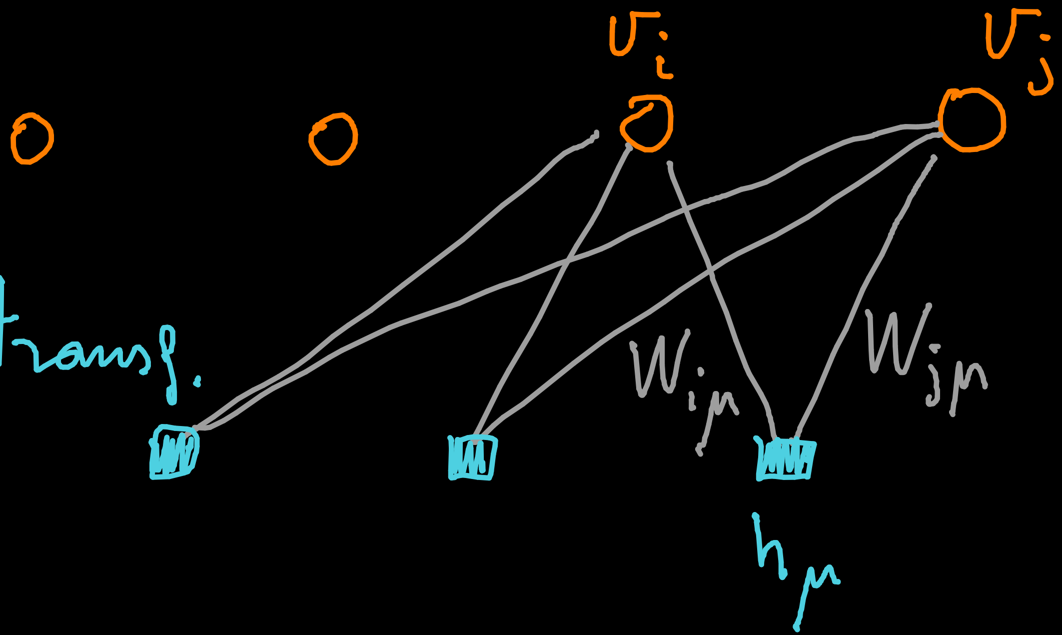
energy $\quad E(\sigma) = -\sum_i a_i \sigma_i - \frac{1}{2} \sum_{ij} J_{ij} \sigma_i \sigma_j$

$$J_{ij} = \sum_{\mu} W_{i\mu} W_{\mu j}$$



$$E(v) = -\sum_i a_i v_i - \frac{1}{2} \sum_{ij} J_{ij} v_i v_j$$

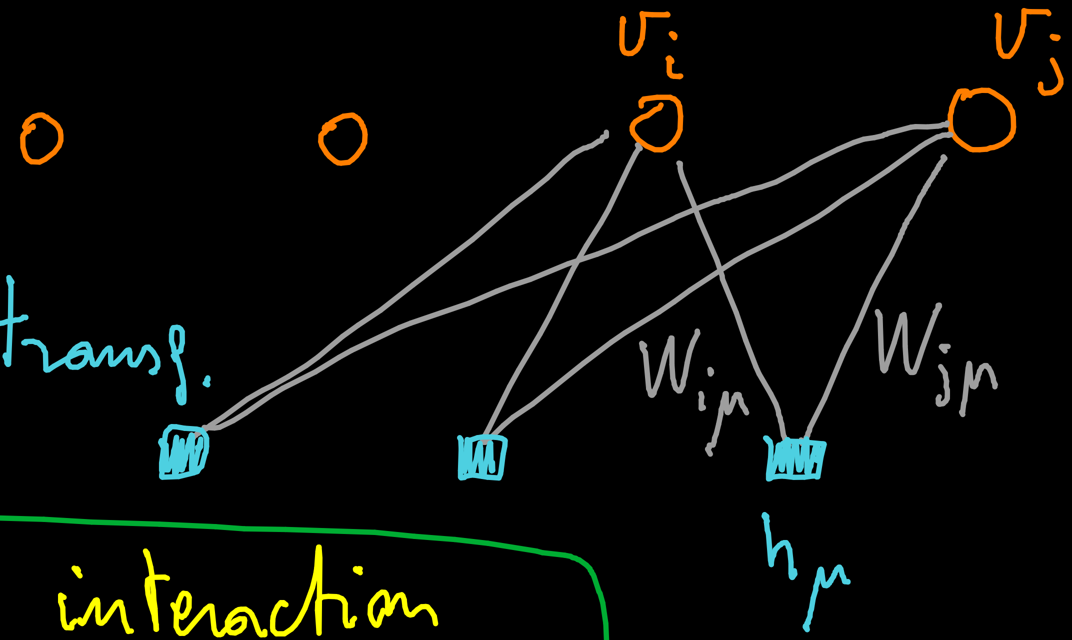
$$J_{ij} = \sum_{\mu} W_{i\mu} W_{j\mu}$$



Hubbard–Stratonovich transf.
(h_μ's with Gaussian stat.)

$$E(v) = -\sum_i a_i v_i - \frac{1}{2} \sum_{ij} J_{ij} v_i v_j$$

$$J_{ij} = \sum_\mu W_{i\mu} W_{j\mu}$$



Hubbard-Stratonovich transf.

$\boxed{J_{ij} \text{ removed: no direct interaction between "spins" } v_i \text{ \& } v_j}$

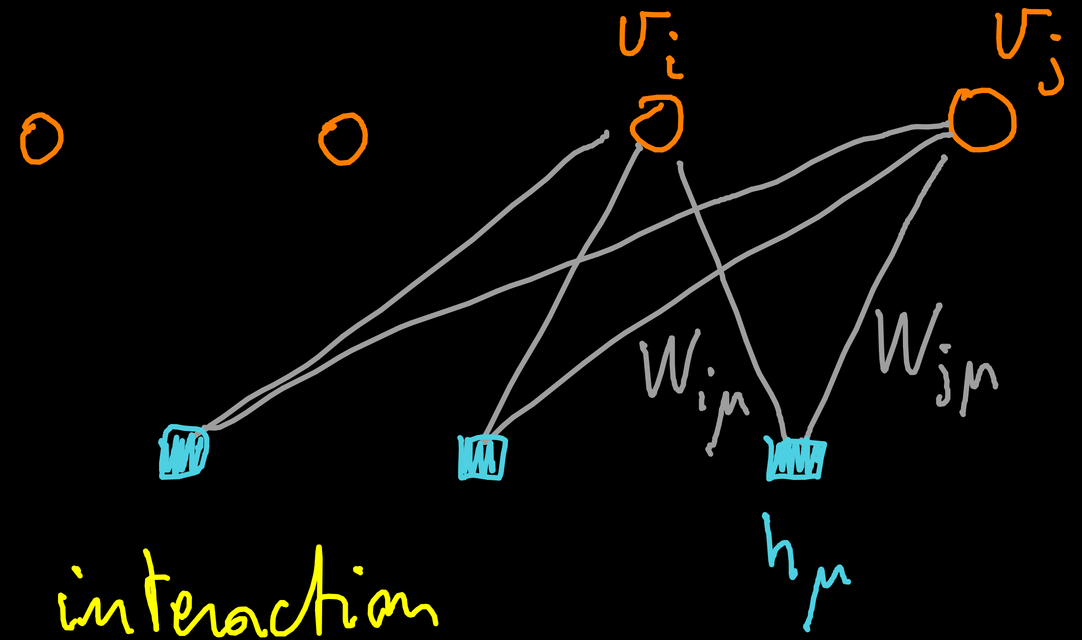
$$E(v) = -\sum_i a_i v_i - \frac{1}{2}\sum_{ij} J_{ij} v_i v_j$$

$$\Downarrow$$

$$E(v,h) = -\sum_i a_i v_i + \frac{1}{2}\sum_\mu h_\mu^2 - \sum_{i\mu} v_i W_{i\mu} h_\mu$$

visible layer

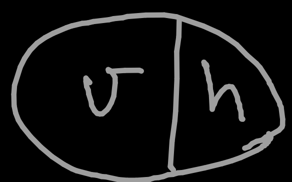
hidden layer



J_{ij} removed : no direct interaction between "spins" v_i & v_j

"Restricted"

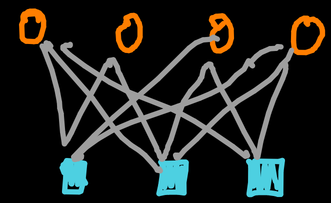
(also NO $h_\mu h_\nu$ interaction)



bipartite system

$$E(v,h) = -\sum_i a_i v_i + \frac{1}{2}\sum_\mu h_\mu^2 - \sum_{i\mu} v_i W_{i\mu} h_\mu$$

Restricted Boltzmann Machines



inspired by previous considerations,

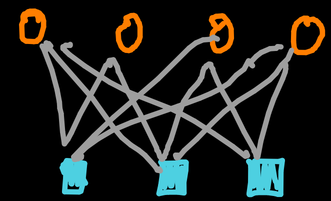
energy $$\boxed{E(v,h) = -\sum_i a_i(v_i) - \sum_\mu b_\mu(h_\mu) - \sum_{i\mu} v_i W_{i\mu} h_\mu}$$

functions

$a_i(\cdot)$

$b_\mu(\cdot)$

Restricted Boltzmann Machines



inspired by previous considerations,

energy $$\boxed{E(v,h) = -\sum_i a_i(v_i) - \sum_\mu b_\mu(h_\mu) - \sum_{i\mu} v_i W_{i\mu} h_\mu}$$

functions

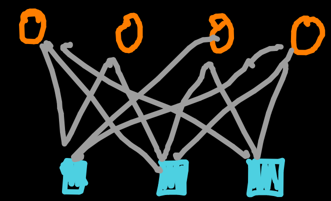
$a_i(\cdot)$

$b_\mu(\cdot)$

	Bernoulli Layers binary $v_i \in \{0,1\}$	Gaussian $v_i \in \mathbb{R}$
$a_i(v_i)$	$a_i v_i$	$\frac{v_i^2}{2\sigma_i^2}$
$b_\mu(h_\mu)$	$b_\mu h_\mu$	$\frac{h_\mu^2}{2\sigma_\mu^2}$

(also other versions, see Mehta et al.)

Restricted Boltzmann Machines



inspired by previous considerations,

energy $$\boxed{E(v,h) = -\sum_i a_i(v_i) - \sum_\mu b_\mu(h_\mu) - \sum_{i\mu} v_i W_{i\mu} h_\mu}$$

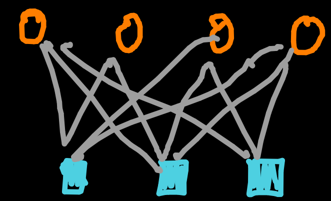
functions

$a_i(\cdot)$

$b_\mu(\cdot)$

	Bernoulli Layers: binary $v_i \in \{0,1\}$	Gaussian: $v_i \in \mathbb{R}$
$a_i(v_i)$	$a_i v_i$	$\frac{v_i^2}{2\sigma_i^2}$
$b_\mu(h_\mu)$	$b_\mu h_\mu$	$\frac{h_\mu^2}{2\sigma_\mu^2}$

Restricted Boltzmann Machines



energy $\boxed{E(v,h) = -\sum_i a_i v_i - \sum_\mu b_\mu h_\mu - \sum_i v_i W_{i\mu} h_\mu}$

$v_i \, , \, h_\mu = -1, 1$

OR

$= 0, 1$

Correlations induced by latent variables $\longrightarrow$ see the review

training

parameters $\theta = \{ W_{i\mu}, \; a_i, \; b_\mu \}$

$$O_j = \partial_{\theta_j} E_\theta(v,h) \qquad O_j(x) = O_j(v,h)$$

$$\partial_{\theta_j}\left(-\mathcal{L}(\theta)\right) = \langle O_j \rangle_{\text{data}} - \langle O_j \rangle_{\text{model}} \qquad (195)$$

for example $\partial_{W_{i\mu}} E = -v_i h_\mu$

thanks to the simple linear appearance of term $v_i W_{i\mu} h_\mu$

hence training via (195) follows these gradient components of $-\mathcal{L}(\theta)$ to minimize it:

$$-\partial_{W_{i\mu}} \mathcal{L} = \langle -v_i h_\mu \rangle_{data} - \langle -v_i h_\mu \rangle_{model}$$

$$-\partial_{a_i} \mathcal{L} = \langle -v_i \rangle_{data} - \langle -v_i \rangle_{model}$$

$$-\partial_{b_\mu} \mathcal{L} = \langle -h_\mu \rangle_{data} - \langle -h_\mu \rangle_{model}$$

hence training via (195) follows these gradient components of $\mathcal{L}(\theta)$ to maximize

$$\partial_{W_{i\mu}} \mathcal{L} = \langle v_i h_\mu \rangle_{data} - \langle v_i h_\mu \rangle_{model}$$

$$\partial_{a_i} \mathcal{L} = \langle v_i \rangle_{data} - \langle v_i \rangle_{model}$$

$$\partial_{b_\mu} \mathcal{L} = \langle h_\mu \rangle_{data} - \langle h_\mu \rangle_{model}$$

maximize log-likelihood

$$\partial_{W_{i\mu}} \mathcal{L} = \langle v_i h_\mu \rangle_{\text{data}} - \langle v_i h_\mu \rangle_{\text{model}}$$

$$\partial_{a_i} \mathcal{L} = \langle v_i \rangle_{\text{data}} - \langle v_i \rangle_{\text{model}}$$

$$\partial_{b_\mu} \mathcal{L} = \langle h_\mu \rangle_{\text{data}} - \langle h_\mu \rangle_{\text{model}}$$

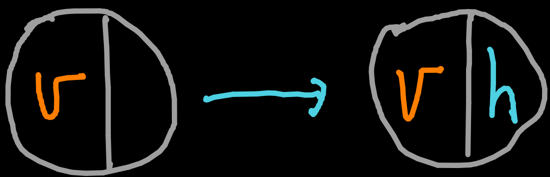
same interpretation: optimum where predictions of model match the averages from data

maximize log-likelihood

$$\partial_{W_{i\mu}} \mathcal{L} = \langle v_i h_\mu \rangle_{data} - \langle v_i h_\mu \rangle_{model}$$

$$\partial_{a_i} \mathcal{L} = \langle v_i \rangle_{data} - \langle v_i \rangle_{model}$$

$$\partial_{b_\mu} \mathcal{L} = \underbrace{\langle h_\mu \rangle_{data}}_{\substack{\text{from "data"} \\ v \,\cup\, h}} - \langle h_\mu \rangle_{model}$$



maximize log-likelihood

$$\partial_{W_{i\mu}} \mathcal{L} = \langle v_i h_\mu \rangle_{\text{data}} - \langle v_i h_\mu \rangle_{\text{model}}$$

$$\partial_{a_i} \mathcal{L} = \langle v_i \rangle_{\text{data}} - \langle v_i \rangle_{\text{model}}$$

$$\partial_{b_\mu} \mathcal{L} = \langle h_\mu \rangle_{\text{data}} - \underbrace{\langle h_\mu \rangle_{\text{model}}}_{\text{run MC to generate } v' \,\&\, h'}$$

Gibbs sampling

much simplified by bipartite structure of restricted B.M. (no interaction between v's and between h's)

$\Rightarrow$ conditionally independent variables

Gibbs sampling

much simplified by bipartite structure of restricted B.M. (no interaction between v's and between h's)

$\Rightarrow$ conditionally independent variables

$$p(v|h) = \prod_i p(v_i|h)$$

$$p(h|v) = \prod_\mu p(h_\mu|v)$$

$v = \{v_i\}$

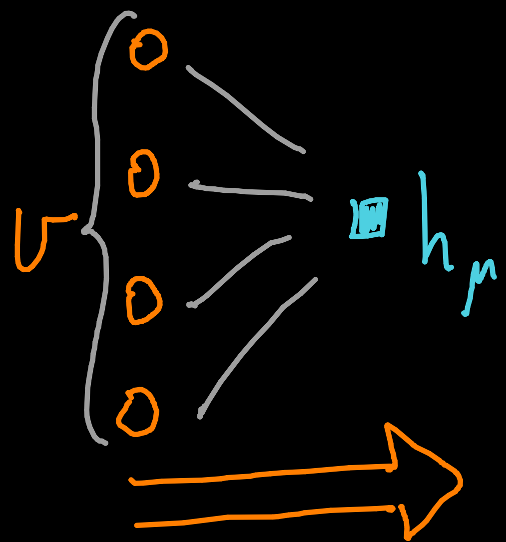
$h = \{h_\mu\}$

(212)

probabilities are factorized

We can draw each h_μ independently from the others ("restricted"!) according to its

$$p(h_\mu | v)$$

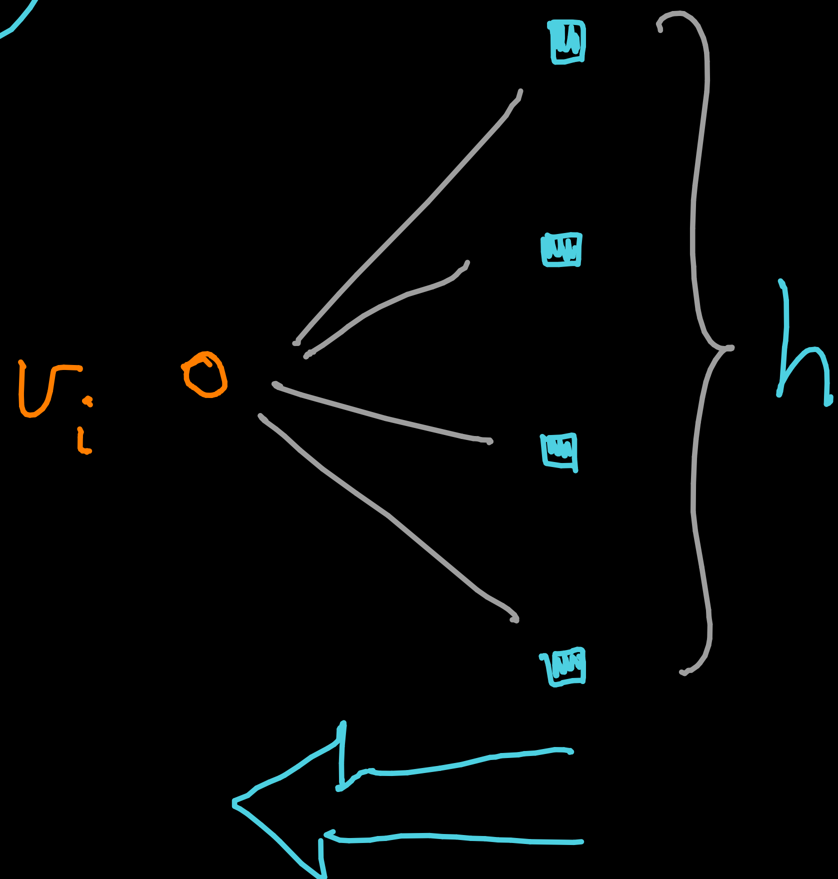


$$p(h|v) = \prod_\mu p(h_\mu | v)$$

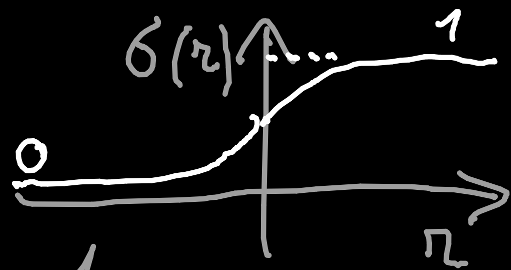
probabilities are factorized

We can draw each v_i independently from the others ("restricted"!) according to its

$$p(v_i \mid h)$$



for Bernoulli layers



(re-)defining sigmoid $\sigma(x) = \dfrac{1}{1+e^{-x}}$

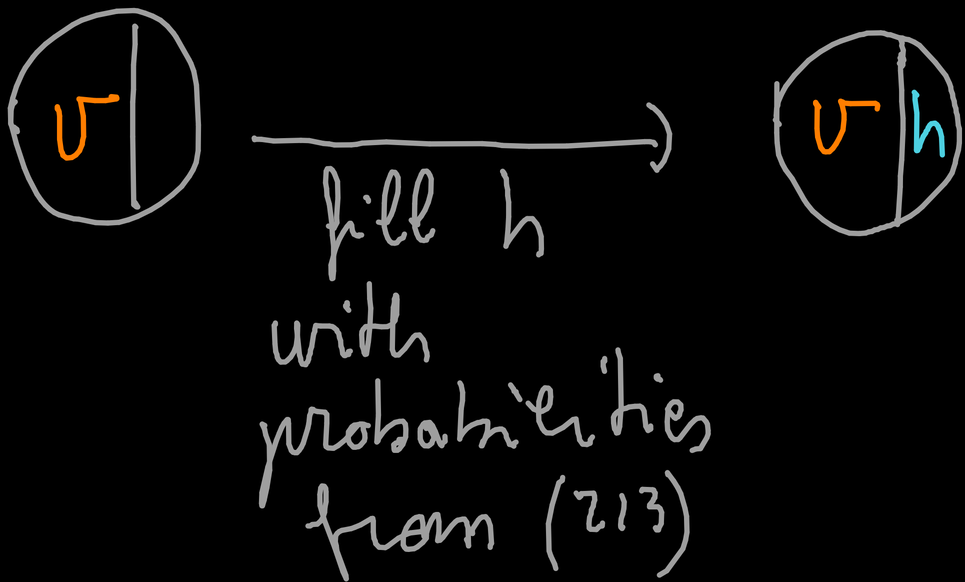
if $v_i = 0, 1$

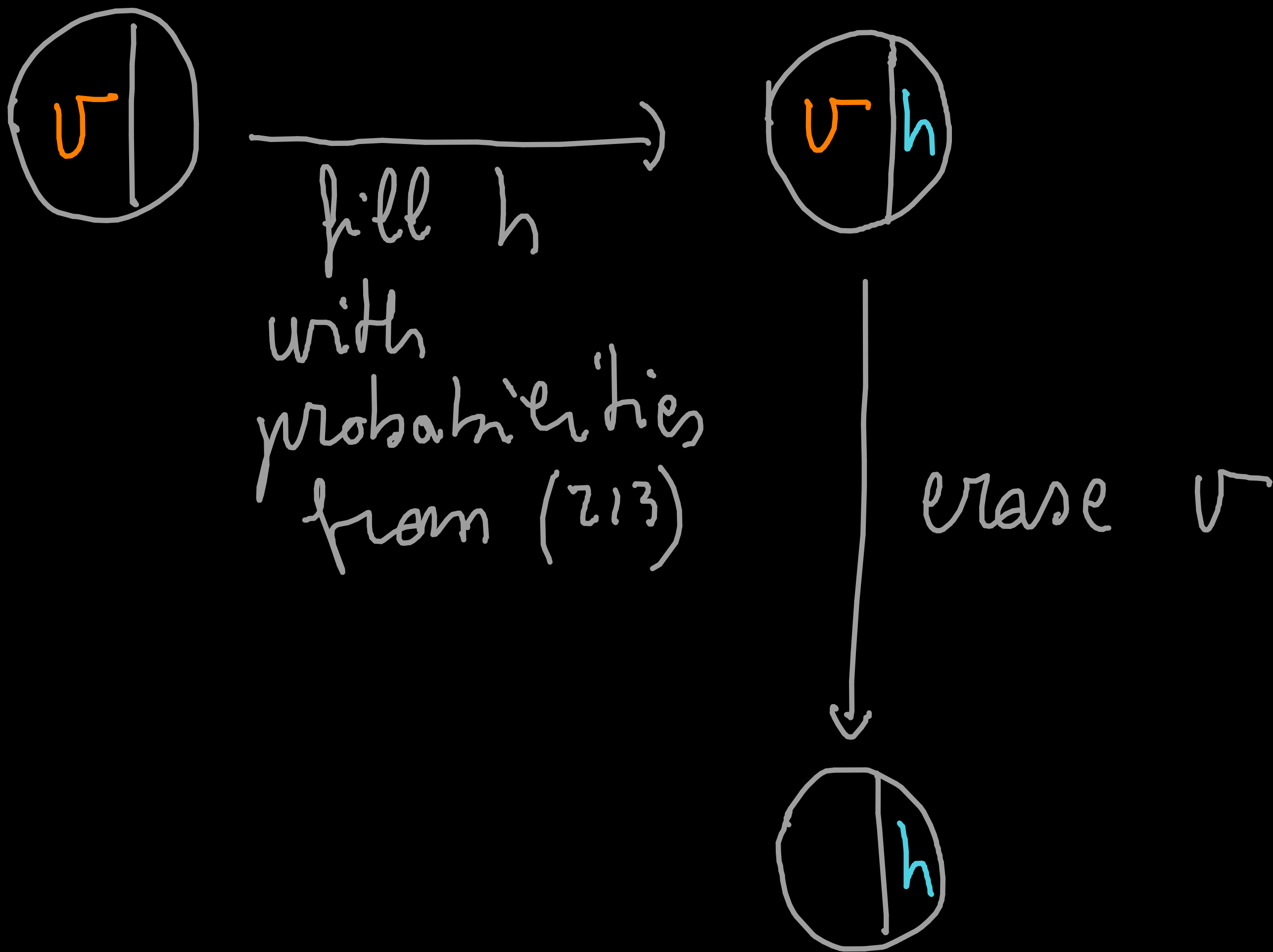
$$p(v_i = 1 \mid h) = \sigma\Big(a_i + \sum_\mu W_{i\mu} h_\mu\Big)$$

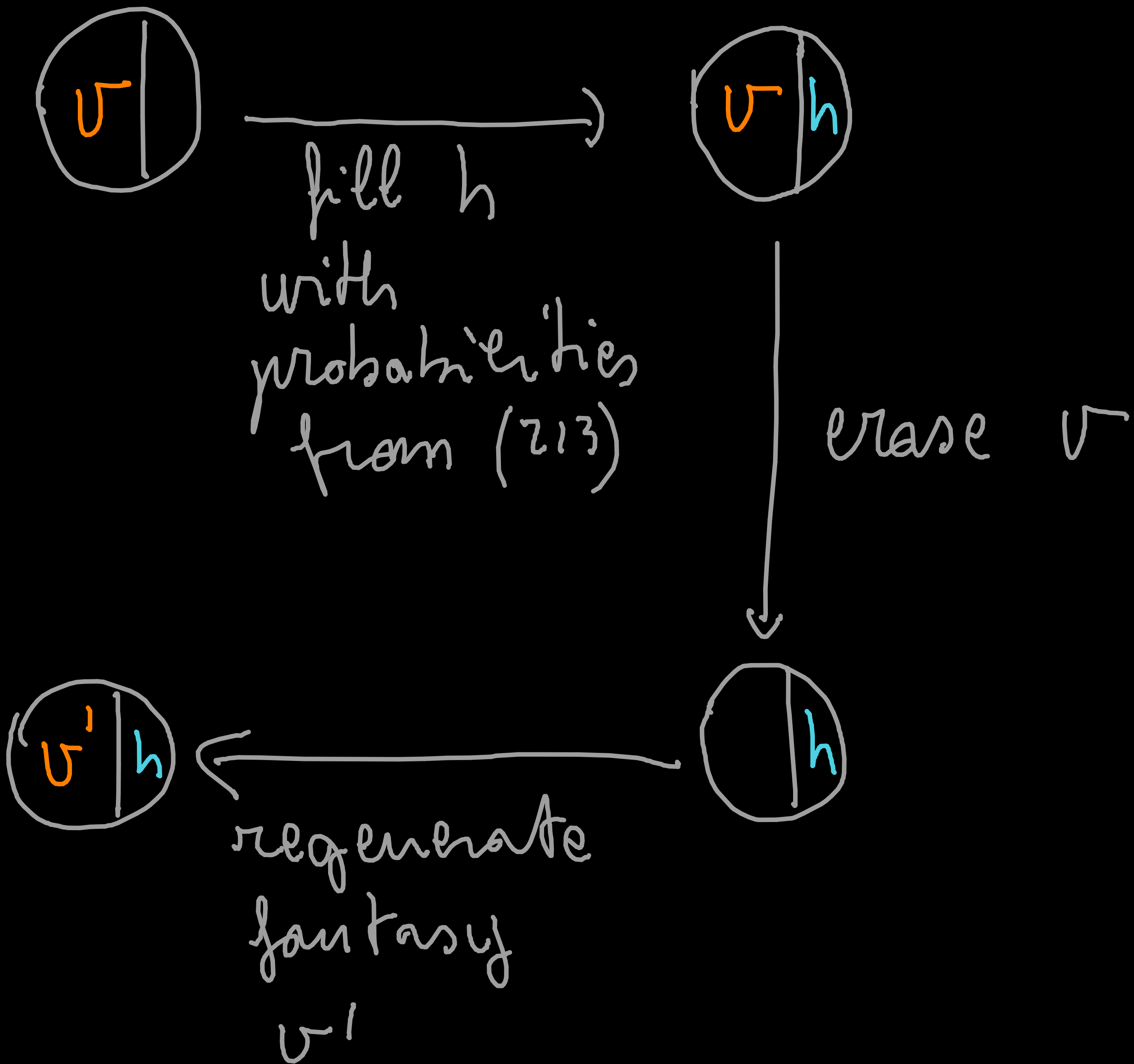
$$p(h_\mu = 1 \mid v) = \sigma\Big(b_\mu + \sum_i W_{i\mu} v_i\Big) \tag{213}$$

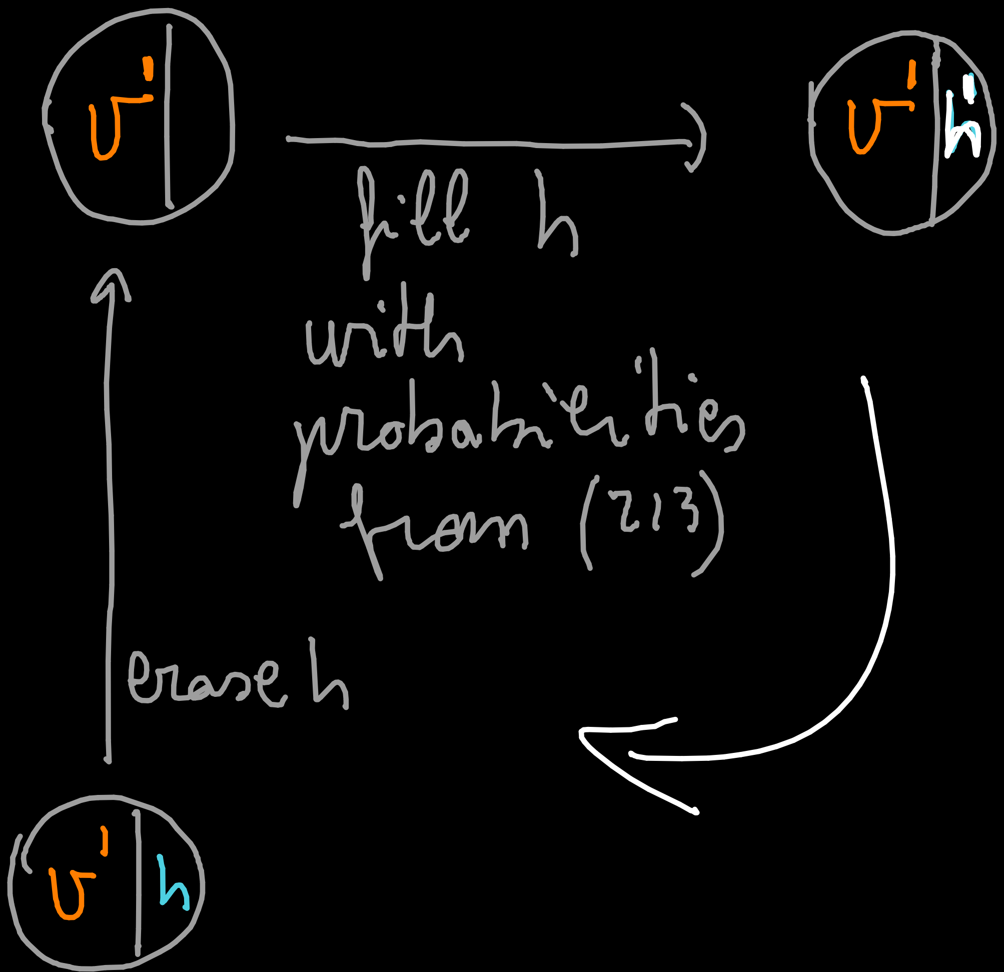
$v| \longrightarrow$

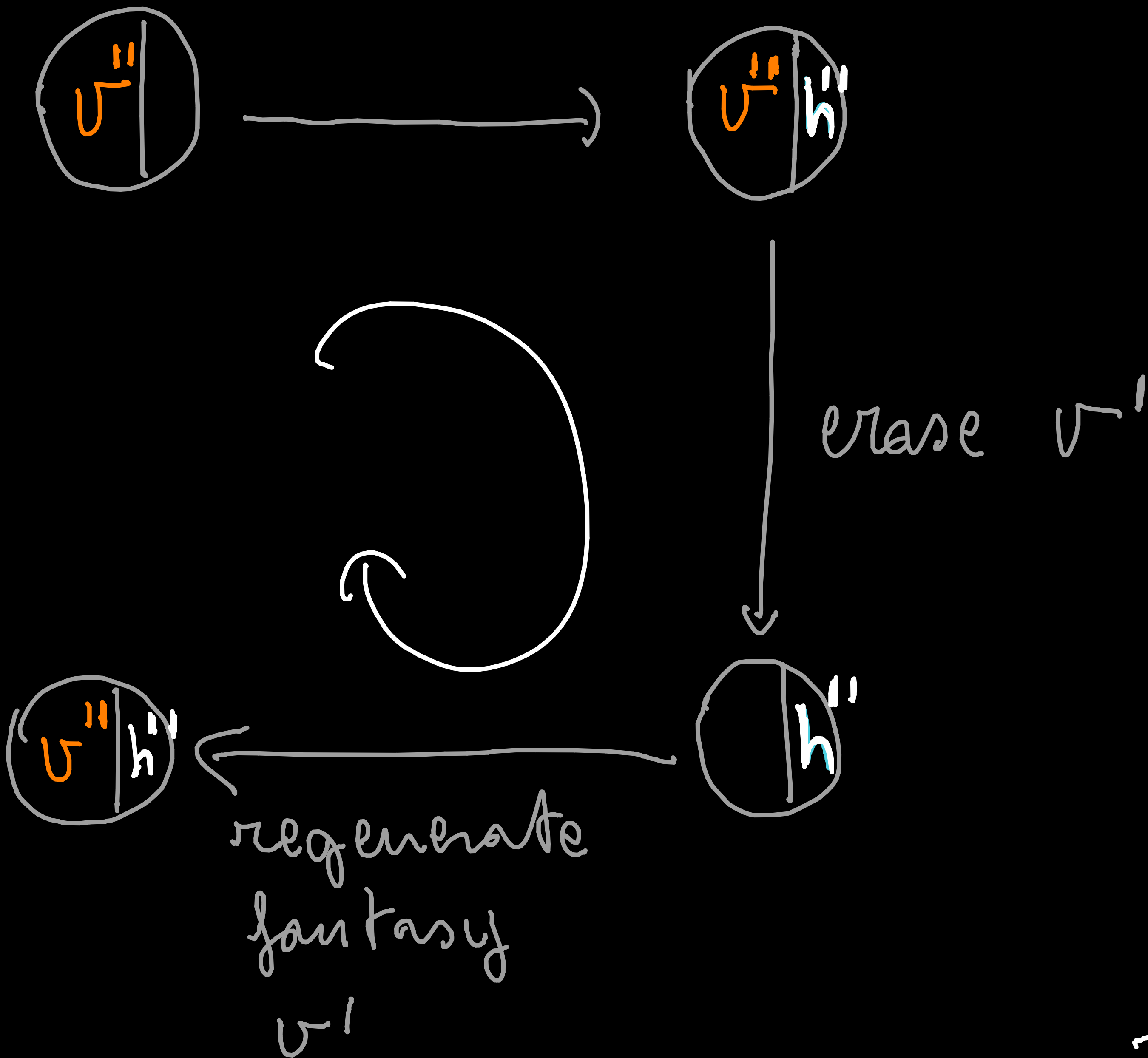
fill h
with
probabilities
from (213)


v
v
h
fill h
with
probabilities
from (213)


v
v
h
fill h
with
probabilities
from (213)
erase v
h


v
v h
fill h with probabilities from (213)
erase v
h
v' h
regenerate fantasy v'


v'
v' h'
v' h
fill h
with
probabilities
from (213)
erase h



repeat the process to sample the model averages

Alternating Gibbs sampling

$$v(0) \to h(0) \to v(1) \to h(1) \to v(2) \to h(2) \to \cdots\cdots$$

Alternating Gibbs sampling

$$v(0) \rightarrow h(0) \rightarrow v(1) \rightarrow h(1) \rightarrow v(2) \rightarrow h(2) \rightarrow \cdots\cdots$$

positive phase

negative phase

Alternating Gibbs sampling

$$\begin{tikzcd} v(0) \arrow[dr] & & v(1) \arrow[dr] & & v(2) \arrow[dr] & & \cdots \\ & h(0) \arrow[ur] & & h(1) \arrow[ur] & & h(2) \arrow[ur] & \end{tikzcd}$$

"data"

at $t = 0$

$$\downarrow$$

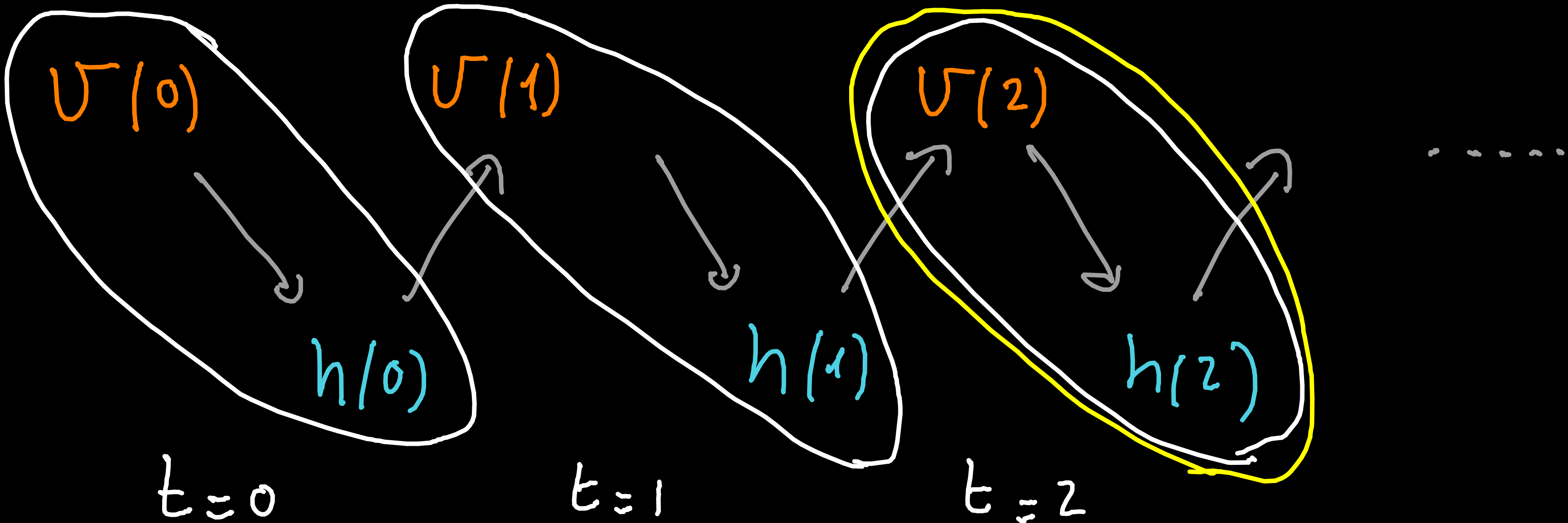
$$\langle \ldots\ldots \rangle_{data}$$

Alternating Gibbs sampling

$$v(0) \rightarrow h(0) \rightarrow v(1) \rightarrow h(1) \rightarrow v(2) \rightarrow h(2) \rightarrow \cdots\cdots$$

"model"
at $t \rightarrow \infty$

Contrastive Divergence (CD-n)



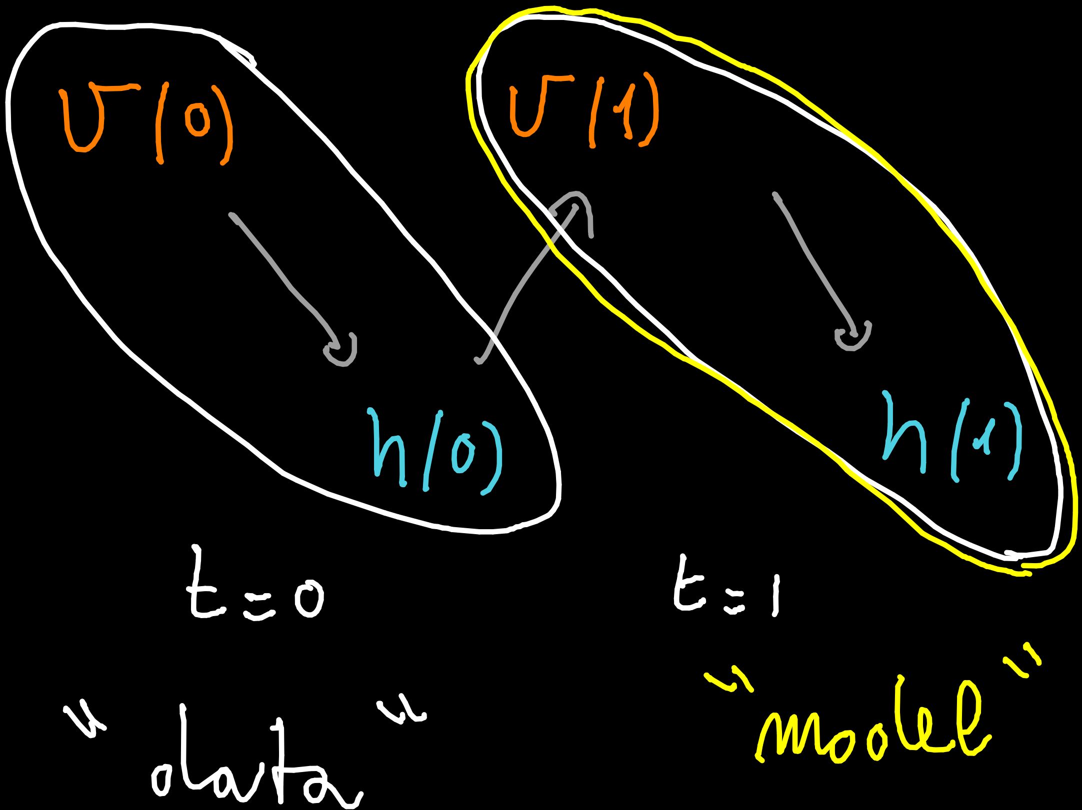
"data"

$n = 2$ (for example)

$\downarrow$

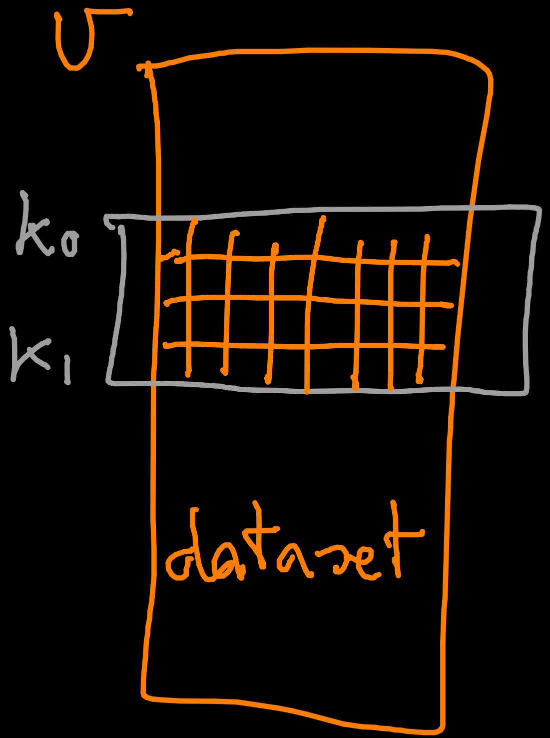
"model" evaluated at $t = n$ rather than $t \to \infty$

Contrastive Divergence (CD-1)



- most extreme example of CD
- fastest
- it works ...

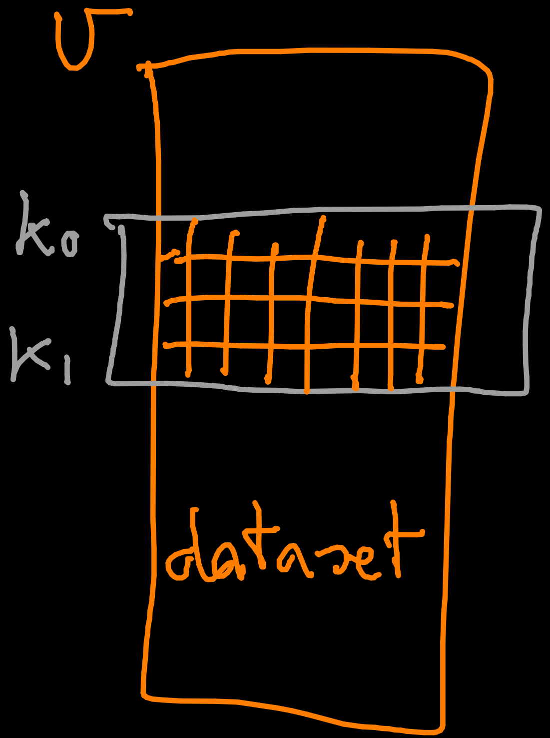
Mini batches



$$B = \{ v^{(k_0)}, \dots, v^{(k_1)} \}$$

$$v^{(k)} = \left(v_{i=1}, v_{i=2}, \dots v_{i=L} \right)^{(k)}$$

Mini batches



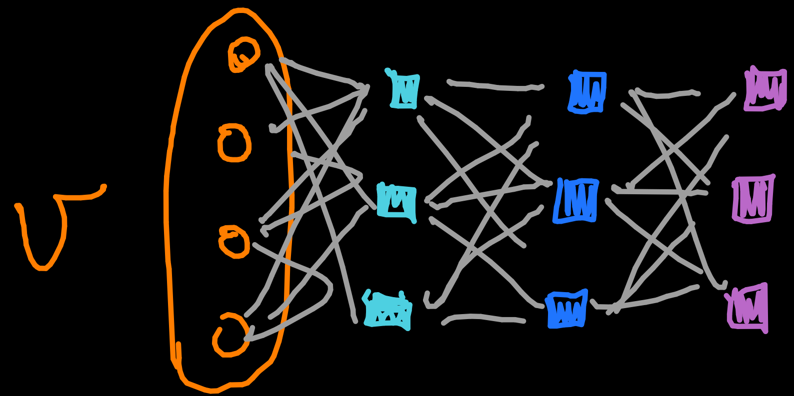
$$B = \{ v^{(k_0)}, \dots, v^{(k_1)} \}$$

$$v^{(k)} = \left(v_{i=1},\ v_{i=2},\ \dots\ v_{i=L} \right)^{(k)}$$

$$\left.\begin{array}{l} \langle \dots \rangle_{\text{data}} \longrightarrow \langle \dots \rangle^{B}_{\text{data}} \\[2ex] \langle \dots \rangle_{\text{model}} \longrightarrow \langle \dots \rangle^{B}_{\text{model}} \end{array}\right\} \text{gradient} \Rightarrow \text{SGD method}$$

More reading in the review:

- initialization
- regularization
- learning rates
- persistent contrastive divergence
- deep Boltzmann machines (many hidden layers)



Summary: after training

- RBM has hidden layer that responds to data and can send back "fantasy data" with similar features

- generative

- denoising

- ...

Final example (Tubiana, Cocco, Monasson)