Introduction to unsupervised learning and generative models

From restricted Boltzmann machines to more advanced models

Felix Lübbe

Department of Physics and Astronomy
Heidelberg University

Quantum and Neural Networks 2019

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Overview

Supervized learning

given: data $\mathbf{x}$ and labels y
common task:

predict labels for unknown data
$\Rightarrow$ estimate $p(y|\,\mathbf{x})$

Unsupervized learning

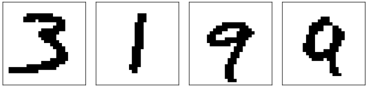
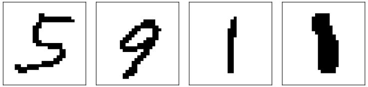
given: *unlabeled*, often high-dimensional data $\mathbf{x}$
possible tasks:

- Dimension reduction
- Clustering
- Sample generation $\Rightarrow$ estimate $p(\mathbf{x})$
  - (Restricted/ Deep) Boltzmann machines
  - Variational autoencoders
  - Generative adversarial networks

Outline

Energy-based models

- task: generate new samples similar to training data
  $\Rightarrow$ estimate $p(\mathbf{x})$ explicitly and draw samples from it

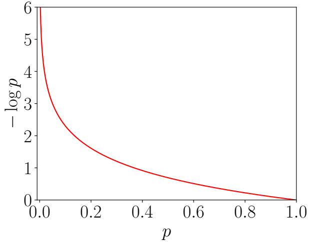
  e.g.  learn $\Rightarrow$ 

- parameterize probability distribution $p(\mathbf{x}; \boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$
  $\Rightarrow$ learn parameters $\boldsymbol{\theta}$
- parameterization of energy-based models:

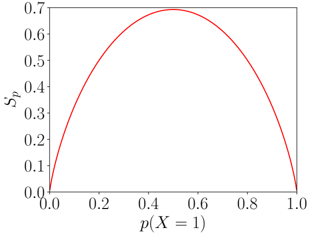
$$p(\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} e^{-E(\mathbf{x}; \boldsymbol{\theta})} \qquad Z(\boldsymbol{\theta}) = \int d\,\mathbf{x}\, e^{-E(\mathbf{x}; \boldsymbol{\theta})}$$

Energy-based models: The principle of maximum entropy

- quantification of uncertainty of an event: $-\log p$
- Shannon entropy: $S_p = -\operatorname{Tr} p(\mathbf{x}) \log p(\mathbf{x})$



- example coin toss:
  $S_p = -p \log p - (1-p) \log(1-p)$



- Principle of maximum entropy: Best choice of probability distribution is the one, that maximizes the entropy given the current knowledge

Energy-based models: The principle of maximum entropy

- keep averages of functions $f_i(\mathbf{x})$ fixed (e.g. averages $\langle x_i \rangle$, correlations $\langle x_i x_j \rangle$)):

$$\langle f_i \rangle_{\text{model}} = \int d\,\mathbf{x}\, f_i(\mathbf{x}) p(\mathbf{x}) = \langle f_i \rangle_{\text{data}}$$

- impose constraints on the entropy using Langrange multipliers:

$$\mathcal{L}[p] = -S_p + \sum_i \lambda_i \left(\langle f_i \rangle - \int dx\, f_i(\mathbf{x}) p(\mathbf{x}) \right) + \gamma \left(1 - \int d\,\mathbf{x}\, p(\mathbf{x}) \right)$$

$$0 = \frac{\delta \mathcal{L}}{\delta p} = (\log p(\mathbf{x}) + 1) - \sum_i f_i(\mathbf{x}) - \gamma \quad \Leftrightarrow \quad p(\mathbf{x}) = e^{\Sigma_i \lambda_i f_i(\mathbf{x}) + (\gamma - 1)}$$

Energy-based models

- the definition of the energy $E(\mathbf{x};\lambda)$ and partition function $Z(\lambda)$

$$E(\mathbf{x};\lambda) = -\sum_i \lambda_i f_i(\mathbf{x}) \qquad Z(\lambda) = \int d\,\mathbf{x}\, e^{-E(\mathbf{x};\lambda)}$$

leads to

$$p(\mathbf{x};\lambda) = e^{\gamma-1} e^{\Sigma_i \lambda_i f_i(\mathbf{x})} = \frac{1}{Z(\lambda)} e^{-E(\mathbf{x};\lambda)}$$

- comparison to statistical physics:
  - canonical ensemble:

$$p(\mathbf{x}) = \frac{1}{Z} e^{-\beta E_{\mathsf{stat}}(\mathbf{x})}, \quad \beta = \frac{1}{k_B T}$$

  - grand canonical ensemble:

$$p(\mathbf{x}) = \frac{1}{Z} e^{-\beta(E_{\mathsf{stat}}(\mathbf{x}) - \mu N_{\mathsf{stat}}(\mathbf{x}))}$$

Energy-based models: loss function

- maximum likelihood loss

$$\mathcal{L}(\boldsymbol{\theta}) = \langle \log(p_{\boldsymbol{\theta}}(\mathbf{x})) \rangle_{\text{data}} = -\langle E(\mathbf{x}; \boldsymbol{\theta}) \rangle_{\text{data}} - \log Z(\boldsymbol{\theta})$$

- overfitting: learning training set specific details, that are not present in the *true* distribution (e.g. noise)
- usually a regularization term is added to prevent overfitting

$$E_{\text{reg}}(\boldsymbol{\theta}) = \Lambda \sum_i |\theta_i|^\alpha, \quad \alpha = 1, 2$$

Energy-based models: training procedure

$$-\mathcal{L}(\boldsymbol{\theta}) = -\langle \log(p_{\boldsymbol{\theta}}(\mathbf{x}))\rangle_{\mathsf{data}} = \langle E(\mathbf{x};\boldsymbol{\theta})\rangle_{\mathsf{data}} + \log Z(\boldsymbol{\theta})$$

$$Z(\boldsymbol{\theta}) = \int d\,\mathbf{x}\, e^{-E(\mathbf{x};\boldsymbol{\theta})}$$

- use a gradient descent-based method, e.g. stochastic gradient descent (SGD)

$\Rightarrow$ have to compute gradient:

$$\begin{aligned}
-\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_i} &= \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\theta})}{\partial \theta_i} \right\rangle_{\mathsf{data}} + \frac{\partial \log Z(\boldsymbol{\theta})}{\partial \theta_i} \\
&= \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\theta})}{\partial \theta_i} \right\rangle_{\mathsf{data}} + \frac{1}{Z(\boldsymbol{\theta})} \int d\,\mathbf{x}\, \frac{\partial}{\partial \theta_i} e^{-E(\mathbf{x};\boldsymbol{\theta})} \\
&= \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\theta})}{\partial \theta_i} \right\rangle_{\mathsf{data}} - \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\theta})}{\partial \theta_i} \right\rangle_{\mathsf{model}}
\end{aligned}$$

Energy-based models: sample generation

- drawing samples from model (fantasy particles) is necessary to compute gradients

$$p(\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} e^{-E(\mathbf{x};\boldsymbol{\theta})}$$

- computation of partition function is often intractable
- only have a function proportional to the probability: $p(\mathbf{x}; \boldsymbol{\theta}) \propto e^{-E(\mathbf{x};\boldsymbol{\theta})}$

$\Rightarrow$ use Markov chain Monte Carlo algorithms, e.g. Metropolis-Hastings algorithm

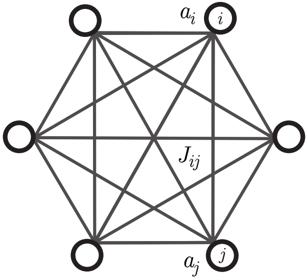
Energy-based models: sample generation

- Markov Chain of random variables: $X = \{X^{(k)} \mid k \in \mathbb{N}_0\}$, transition probability:

$$p_{ij}^{(k)} = \mathrm{Pr}(X^{(k+1)} = j \mid X^{(k)} = i)$$

- one can construct an update operator, such that for $k \to \infty$ the chain contains samples from the desired distribution
- cannot run the chain for an infinite amount of time $\Rightarrow$ approximation

Boltzmann machines



- Energy function:

$$E(x) = -\sum_i a_i x_i - \sum_{i,j} J_{ij} x_i x_j$$

$\Rightarrow$ fixes first and second order moment

- Type of variables: discrete or continuous?
  - discrete states:
    - often two state units (e.g. $\{0, 1\}$, Bernoulli units)

    $\Rightarrow$ Boltzmann machine
  - continuous states:
    - probability distribution is multi-dimensional Gaussian

    $\Rightarrow$ can solve partition function analytically

Boltzmann machines: hidden units

- energy function for Bernoulli units:

$$E(\mathbf{v},\mathbf{h}) = -\sum_i a_i v_i - \sum_\mu b_\mu h_\mu - \sum_{i,j} J_{ij} v_i v_j - \sum_{\mu,\nu} K_{\mu\nu} h_\mu h_\nu - \sum_{i,\mu} W_{i\mu} v_i h_\mu$$

- marginalized distribution:

$$p(\mathbf{v}) = \int d\,\mathbf{h}\, p(\mathbf{v},\mathbf{h}) = \int d\,\mathbf{h}\, \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z}$$

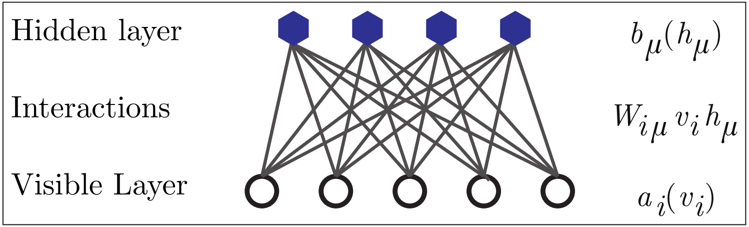
$\Rightarrow$ higher order interactions between visible units in the marginalized distribution

- problem: Boltzmann machines scale poorly with dimension of system

Restricted Boltzmann machines

- for Bernoulli units :

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



- general form:

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

Restricted Boltzmann machines

- can capture high order interactions between visible units
- variable number of hidden units
- sufficiently large RBM can take on any probability distribution
- bipartite structure leads to efficient training algorithm

Restricted Boltzmann machines: sample generation and training

- interactions: visible $\leftrightarrow$ hidden

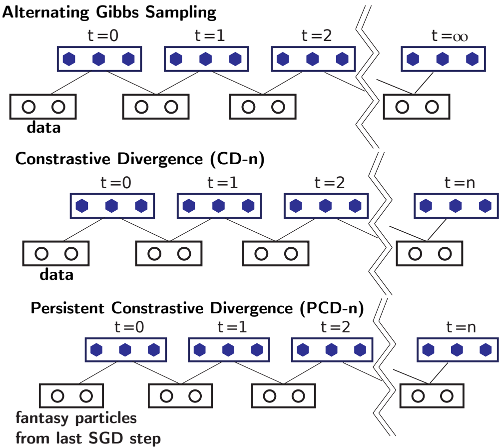
$$p(\mathbf{v}\,|\,\mathbf{h}) = \prod_i p(v_i|\,\mathbf{h})$$

$$p(\mathbf{h}\,|\,\mathbf{v}) = \prod_i p(h_i|\,\mathbf{v})$$

- probability for a single unit:

$$p(v_i = 1|\,\mathbf{h}) = \sigma(a_i + \sum_\mu W_{i\mu} h_\mu)$$

$$p(h_\mu = 1|\,\mathbf{v}) = \sigma(b_\mu + \sum_i W_{i\mu} v_i)$$

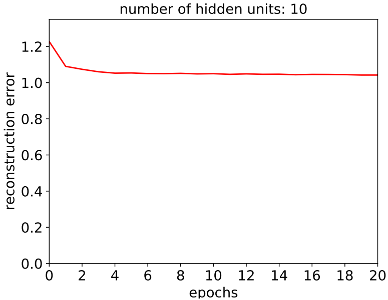


Restricted Boltzmann machines: MNIST with the Paysage package

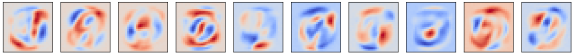
- MNIST dataset: 70000 handwritten digits (28px $\times$ 28px, black and white)
- Paysage package: built to train unsupervized generative models
- SGD with ADAM optimizer and minibatches of size 100
- L^2 regularization with $\Lambda = 10^{-3}$
- Persistent Constrastive Divergence with 1 Gibbs step per SGD step
- sample generation after training with 100000 Gibbs steps

$\Rightarrow$ vary number of hidden units and epochs

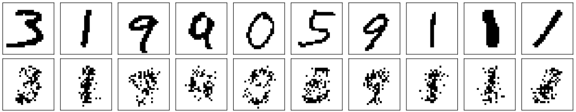
Restricted Boltzmann machines: MNIST with 10 hidden units



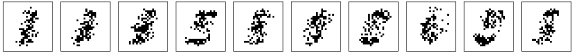
Weights of the hidden units:



Reconstructed samples:

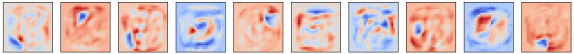


Fantasy particles:

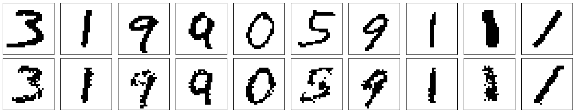


Restricted Boltzmann machines: MNIST with 100 hidden units

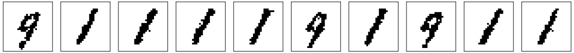
Weights of the hidden units:



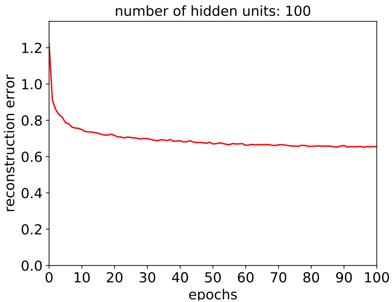
Reconstructed samples:



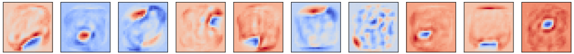
Fantasy particles:



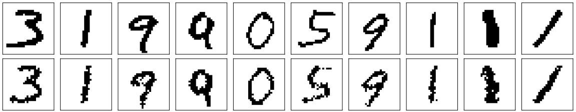
Restricted Boltzmann machines: MNIST with 100 hidden units



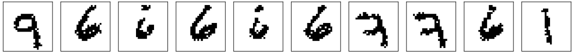
Weights of the hidden units:



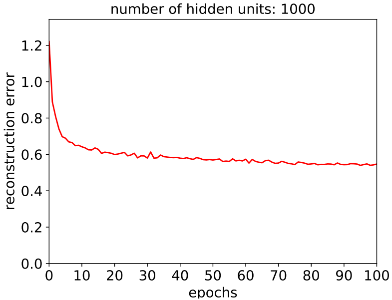
Reconstructed samples:



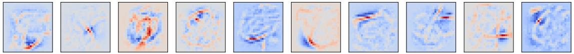
Fantasy particles:



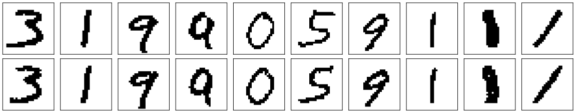
Restricted Boltzmann machines: MNIST with 1000 hidden units



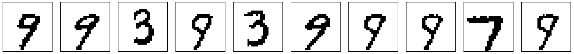
Weights of the hidden units:



Reconstructed samples:

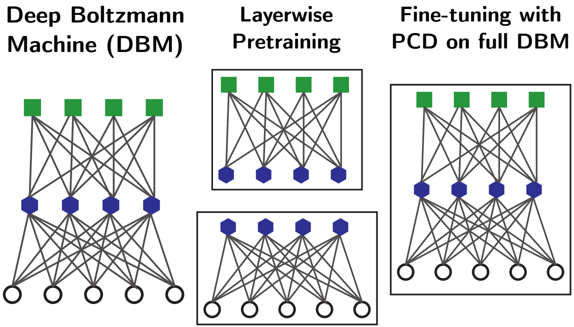


Fantasy particles:

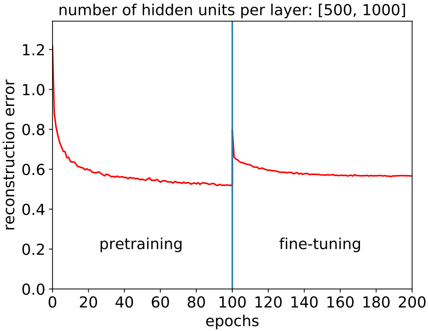


Deep Boltzmann machines

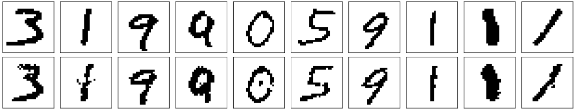
- capture complex interaction between hidden units
- not to be confused with deep belief networks



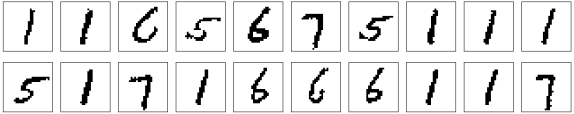
Deep Boltzmann machines: attempt on MNIST with two layers



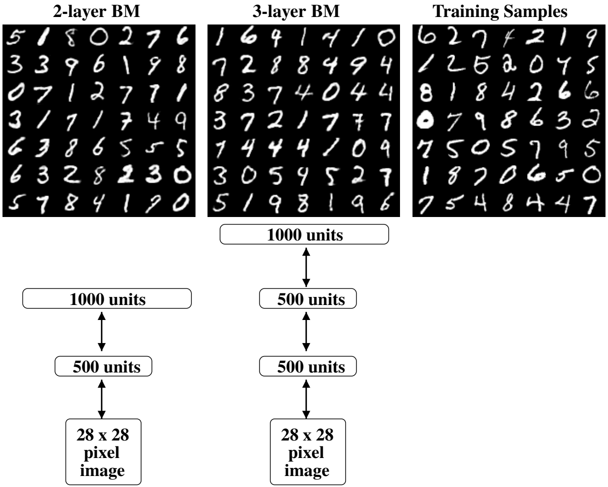
Reconstructed samples:



Fantasy particles:

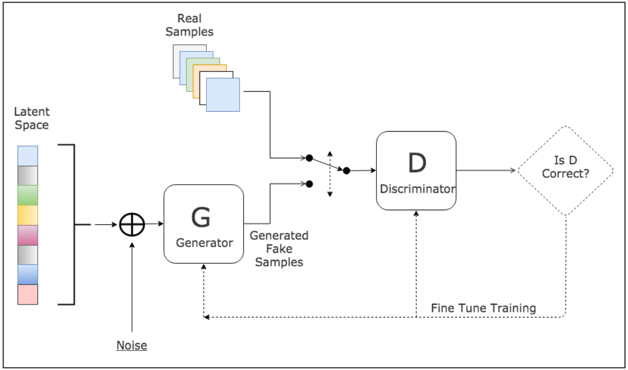


Deep Boltzmann machines: MNIST



Salakhutdinov, Hinton: Deep Boltzmann Machines

Generative adversarial networks



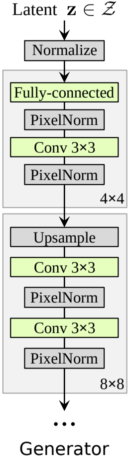


image sources: GAN, Generator: [arXiv:1812.04948]

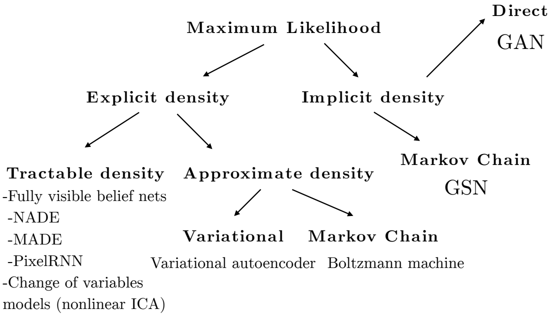
Generative adversarial networks: Style-based GANs

thispersondoesnotexist.com

[arXiv:1812.04948] A Style-based Generator Architecture for Generative Adversarial Networks

Summary

- Energy-based models
- Boltzmann machines
- Restricted Boltzmann machines
- Deep Boltzmann machines
- Generative adversarial networks



[arXiv:1701.00160] NIPS 2016 Tutorial: Generative Adversarial Networks