Machine learning

Boltzmann Machines

Dima Kochkov[1]

[1]Department of Physics
University of Illinois at Urbana-Champaign

Algorithm interest meeting, 2016

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Motivation for machine learning

Machine learning is a problem solving approach that comes in handy when algorithmic solution is hard to obtain.
Pros:

- requires minimum prior knowledge
- solution can adapt to a new environment

Cons:

- inefficient use of hardware
- weaker guarantees of correctness
- requires big datasets for complex problems

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Examples of successful applications

Just to name a few:

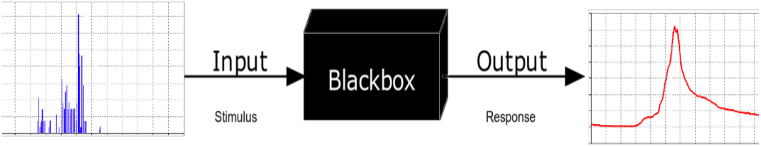
- Speech recognition - Siri, ok Google
- Image recognition - ImageNet
- Fraud detection
- Recommendation systems - Netflix competition
- Games - AlphaGo
- Funky stuff like self driving cars, robotics etc

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Optimizational point of view

- Solution to the problem is given in a variational form of a black box with a gazillion of knobs.
- Algorithm tunes those knobs to have a better solution. Algorithm is data driven.



- supervised learning (SVM, BackProp, Decision Trees etc)
- **unsupervised learning** (Kmeans, EM, **Boltzmann Machines**)

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Artificial Neural Networks

Artificial Neural Networks represent a class of models that constitute a set of connected units (neurons).
Most of the time one can define following properties of a neuron:

- input values, x_i
  - vector $< bool >$
  - vector$< double >$
- output value, f(input, links), usually f($w_i x_i$)
  - f $= tanh(w_i x_i)$
  - f $= \frac{1}{1+e^{-(w_i x_i)}}$
  - f $= max(0, w_i x_i)$

Activity pattern evolves according to a specific rule of the network.
I will use s_i to represent i^{th} neuron, or v_i and h_i.

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. **Energy based models**
   - **Hopfield Nets**
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Memory storage

One of the simplest energy based models - Hopfield Net:

- Binary units $s_i \in 0, 1$
- Symmetric weights $w_{i,j} = w_{j,i}$
- Features a global energy function E
- Energy minimas correspond to memories

$$E = -\sum_i b_i s_i - \sum_{i,j} w_{i,j} s_i s_j \quad (1)$$

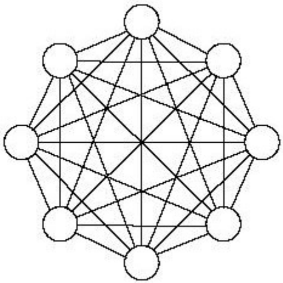


Figure: Hopfield Net

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Basic Boltzmann machine

Ingredients for the Boltzmann machine:

- Hopfield net + hidden units
- Gibbs probability distribution $P = \frac{e^{\frac{-E}{T}}}{Z}$

$$E = -\sum_i b_i s_i - \sum_{i,j} w_{i,j} s_i s_j \quad (2)$$

s can be either visible (**v**) or hidden (**h**) units of the model

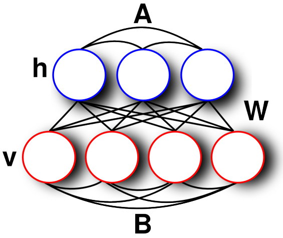


Figure: Boltzmann machine

A generative model with a potential for data interpretation.

How does BM "interpret" the data

- States of the hidden units correspond to interpretations of the data.
- Low energy states of the hidden units given visible units correspond to "good" interpretations
- Common structure allows low energy interpretations

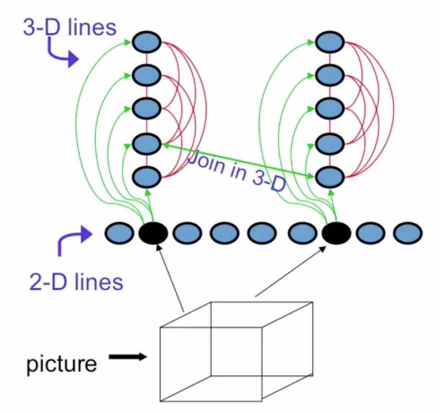


Figure: Data interpretation

How do we learn?

Learning objective

We want minimize the "distance" between the probability distribution our Boltzmann machine generates and the distribution from which the data was drawn.

$$P = \log(\prod_{i \in data} P(v = d_i)) = \sum_{i \in data} \log(P(v = d_i) \tag{3}$$

$$\frac{\partial P}{\partial w_{\alpha,\beta}} = \sum_i \frac{\partial}{\partial w_{\alpha,\beta}} (\sum_{h'} -E(v = d_i, h = h^{'}) - \sum_{v^{'},h'} -E(v = v^{'}, h = h^{'})) \tag{4}$$

$$\frac{\partial P}{\partial w_{\alpha,\beta}} = \sum_i (\sum_{h'} s_\alpha s_\beta)|_{v=d_i} - (\sum_{v^{'},h'} s_\alpha s_\beta) \tag{5}$$

$$\frac{\partial P}{\partial w_{\alpha,\beta}} = < s_\alpha s_\beta >_{data} - < s_\alpha s_\beta > \tag{6}$$

Algorithm

To train a Boltzmann machine on a given dataset we:

- fix visible units to the values of the data instance
- compute $< s_\alpha s_\beta >$ (positive phase)
- set visible units free and again compute $< s_\alpha s_\beta >$ (negative phase)
- after processing a batch of data, update parameters

Potential issues

- Can't efficiently compute $< s_\alpha s_\beta >$

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

MCMC

Markov Chain Monte Carlo:

$$< s_\alpha s_\beta >= \sum_s s_\alpha s_\beta p(s) = \sum_s s_\alpha s_\beta \frac{e^{-E}}{Z} \tag{7}$$

- clamp the data on the visible neurons
- sample $s_\alpha s_\beta$ from the Markov chain (positive phase)
- set visible units free and again sample $s_\alpha s_\beta$ (negative phase)
- repeat for the dataset, update weights

Potential issues

- Markov chain might take a very long time to equilibrate
- How do we know if we have a good estimate?

better MCMC

We can use a clever trick to have a warm start. We keep a set of equilibrated Markov chains with fixed and free visible units.
Equilibrated chains with clamped units ("Particles") are used to evaluate $< s_\alpha s_\beta >_{data}$
Equilibrated chains with free visible units ("Fantasy particles") are used to evaluate $< s_\alpha s_\beta >$
Status

- Still to slow for most applications
- In theory should work well only for a full batch learning
- Much better than previously described method

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Mean Field

If every input state has only one "good" interpretation, then every neurons interact with averages of others.

Modifications to the positive phase:

- Promote all units to real valued units $in[0,1)$ (p(1))
- Stochastically update the values based on values of others:

$$p_i^{t+1} = \frac{1}{1 + e^{b_i + \sum_j p_j^t w_{i,j}}} \tag{8}$$

This is not correct, but works quite well. To kill oscillations one can use dumped mean field

$$p_i^{t+1} = \lambda p_i^t + (1 - \lambda) \frac{1}{1 + e^{b_i + \sum_j p_j^t w_{i,j}}} \tag{9}$$

Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

RBM

Restricted Boltzmann machines are models with only one hidden layer and no connections within hidden units.

Profit:

- Correlations in positive phase can be computed exactly in one go.

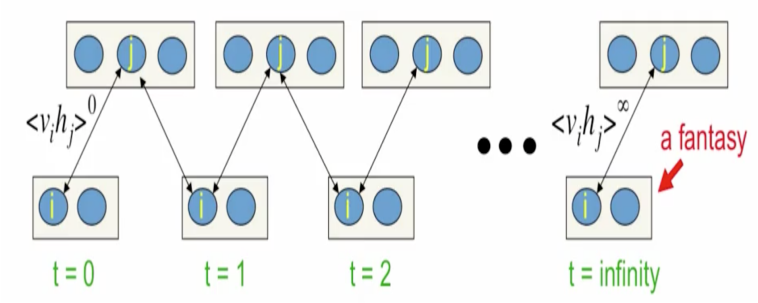
$$< v_i h_j >_{data} = v_i^d p(h_j = 1|v) = \frac{\sum_{h|h_j=1} e^{-E(v^d,h)}}{\sum v', he^{-E(v',h}} \qquad (10)$$

$$= \frac{e^{-E(h_j=1)} \sum_{\bar{h}} e^{-E(v^d,\bar{h})}}{e^{-E(h_j=1)} \sum_{\bar{h}} e^{-E(v^d,\bar{h})} + e^{-E(h_j=0)} \sum_{\bar{h}} e^{-E(v^d,\bar{h})}} \qquad (11)$$

$$= \frac{1}{1 + e^{\frac{E(h_j=1)}{E(h_j=0)}}} \qquad (12)$$

Contrastive divergence

Thermal equilibrium is independent of initial conditions



We can cheat and use $< v_ih_j >^n$ instead of $< v_ih_j >^{\text{inf}}$. It gives us an incorrect gradient, but it works quite well.

We look at the direction in which the "particle" tries to move and increase the energy of states in that direction, while lowering the energy of the data.

Stacking

We can stack Restricted Boltzmann Machines by treating the state of the hidden layer as an input to a second RBM.
This approach is used for training Deep Boltzmann Machines and abstract feature detection and can be used in:

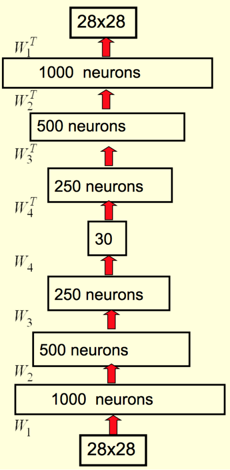
- Deep neural nets (pretraining)
- Autoencoders, hashing

Stacking

Autoencoders compress the data into a feature vector based on high level features

That enables smart addressing and hashing for complicated data with a lot of structure (images, etc) It was shown, that this procedure can be exactly mapped to a Kadanoff RG procedure

arxiv

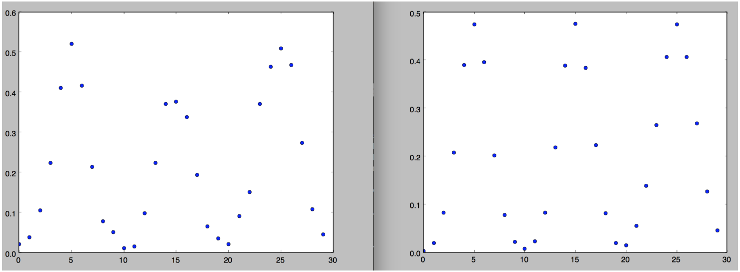


Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Locally grown

A basic code was able to faithfully learn the probability distribution of binary vectors from the data.
Works better if the energy landscape is smooth and doesn't feature deep disjoint minimas.

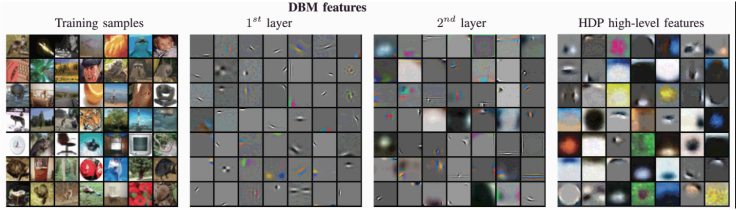


Outline

1. Machine learning
   - Motivation
   - Hall of fame
2. Machine learning basics
   - General framework
   - Neural networks
3. Energy based models
   - Hopfield Nets
   - Boltzmann machines
4. Algorithmic improvements
   - Markov Chain Monte Carlo
   - Mean Field approximation
   - Restricted Boltzmann Machines
5. Examples
   - Matching probability distribution
   - Image features extraction

Imported

A more sophisticated model trained on a set of images is capable of recovering a number of interesting features



Summary

- Boltzmann machine adjusts its weights to reproduce "correlations" within the data
- Even unlabeled data can be used for learning
- Some physics models can be successfully used in CS
- Outlook
  - Can we have more/less expressive models? How do we learn them?
  - Can some known RG methods be useful for learning?