THE UNIVERSITY *of* EDINBURGH
informatics

Restricted Boltzmann Machines for Collaborative Filtering

Authors: Ruslan Salakhutdinov
Andriy Mnih
Geoffrey Hinton

Presentation by:

Benjamin Schwehn **Ioan Stanculescu**

Overview

- The Netflix prize problem
- Introduction to (Restricted) Boltzmann Machines
- Applying RBMs to the Netflix problem
  - Probabilistic model
  - Learning
  - The Conditional RBM
- Results

The Netflix Prize

- Automated Recommender System (ARS)
- Predict how a user would rate a movie
- Recommend movies likely to be rated high
- USD 1,000,000 prize for a 10% improvement

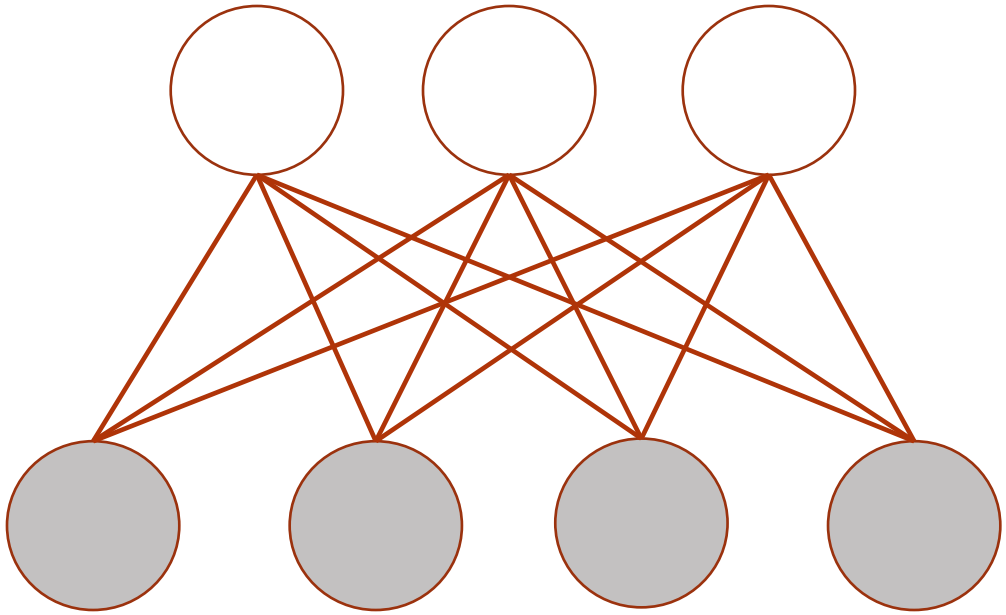
The Netflix Data Set

- *100,480,507* dated ratings by *N=480,189* users for *M=17,770* movies
- Qualifying data: *2,817,131* entries w/o rating
- Teams informed of the RMSE over a random half of qualifying data set to make optimizing on test set more difficult

User ID	Movie ID	Movie Name	Release	Rating (1-5)	Rating Date
1488844	1	Dinosaur Planet	2003	3	2005-09-06
822109	1	Dinosaur Planet	2003	5	2005-05-13
2578149	1904	Dodge City	1939	1	2003-09-02

Restricted Boltzmann Machine (RBM)

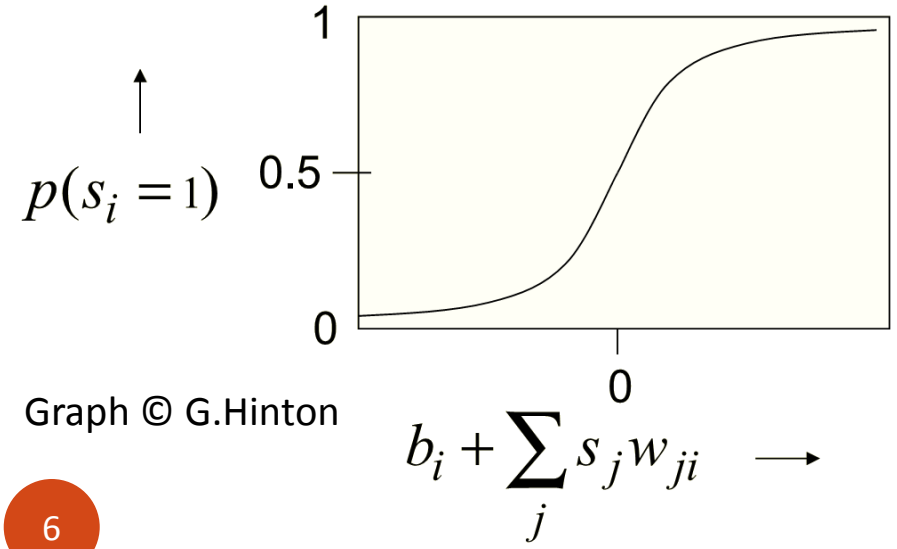
- Random Markov Field with no connections within both visible and hidden layers
- Layers are conditionally independent



Binary Stochastic Neurons

- Nodes in an RBM have state in {0,1}
- $p(s_i = 1)$ given by a function of the input from neurons in "other" layer and bias:

$$p(s_i = 1) = \frac{1}{1 + \exp(-b_i - \sum_j s_j w_{ji})}$$



Graph © G.Hinton

bias

node in "other" layer

connection weight

Equilibrium

- If nodes are updated sequentially, eventually an equilibrium is reached
- Define energy of joint configuration as:

$$E(\boldsymbol{V}, \boldsymbol{h}) = -\sum_{ij} v_i h_j W_{ij} - \sum_{i} v_i b_i - \sum_{j} h_j b_j$$

connection between visible and hidden nodes

bias visible nodes

bias hidden nodes

Joint probability

$$E(\boldsymbol{V}, \boldsymbol{h}) = -\sum_{ij} v_i h_j W_{ij} - \sum_i v_i b_i - \sum_j h_j b_j$$

$$p(\boldsymbol{V}, \boldsymbol{h}) = \frac{\exp(-E(\boldsymbol{V}, \boldsymbol{h}))}{\sum_{\boldsymbol{V'}, \boldsymbol{h'}} \exp(-E(\boldsymbol{V'}, \boldsymbol{h'}))}$$

partition function

marginal distribution over visible nodes

$$p(\boldsymbol{V}) = \frac{\sum_h \exp(-E(\boldsymbol{V}, \boldsymbol{h}))}{\sum_{\boldsymbol{V'}, \boldsymbol{h'}} \exp(-E(\boldsymbol{V'}, \boldsymbol{h'}))}$$

Learning in RBMs (*PMR Week 9*)

$$\frac{\partial \log p(\mathbf{V})}{\partial \theta_l} = \sum_{\mathbf{h}} \phi_l(\mathbf{V}, \mathbf{h}) p(\mathbf{h} \mid \mathbf{V}) - \sum_{\mathbf{V},\mathbf{h}} \phi_l(\mathbf{V}, \mathbf{h}) p(\mathbf{V}, \mathbf{h})$$

$$\theta_l = W_{ij} \qquad \phi_l(\mathbf{V}, \mathbf{h}) = v_i h_j$$

$$\frac{\partial \log p(\mathbf{V})}{\partial W_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}$$

*expectation in equilibrium when **V** is clamped "clamped phase"*

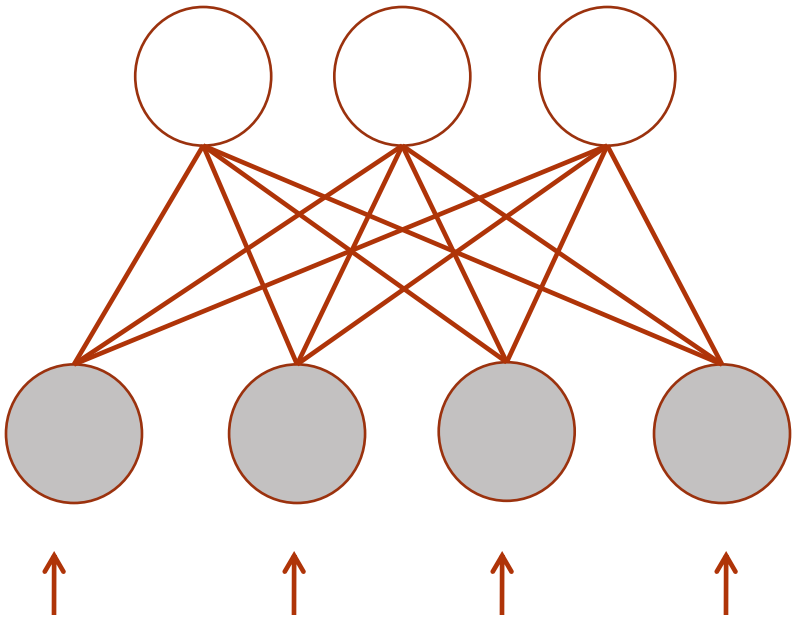
expectation in equilibrium without clamp "free running phase"

Learning

$$\frac{\partial \log p(\mathbf{V})}{\partial W_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}$$

1. Clamped Phase

2. Free Phase



Training Data ***V***

$$p(h_j = 1|\boldsymbol{V}) = \frac{1}{1 + \exp(-b_j - \sum_i v_i W_{ij})}$$

$$\Rightarrow \langle v_i h_j \rangle_{data}$$

$$p(v_i = 1|\boldsymbol{h}) = \frac{1}{1 + \exp(-b_i - \sum_j h_j W_{ij})}$$

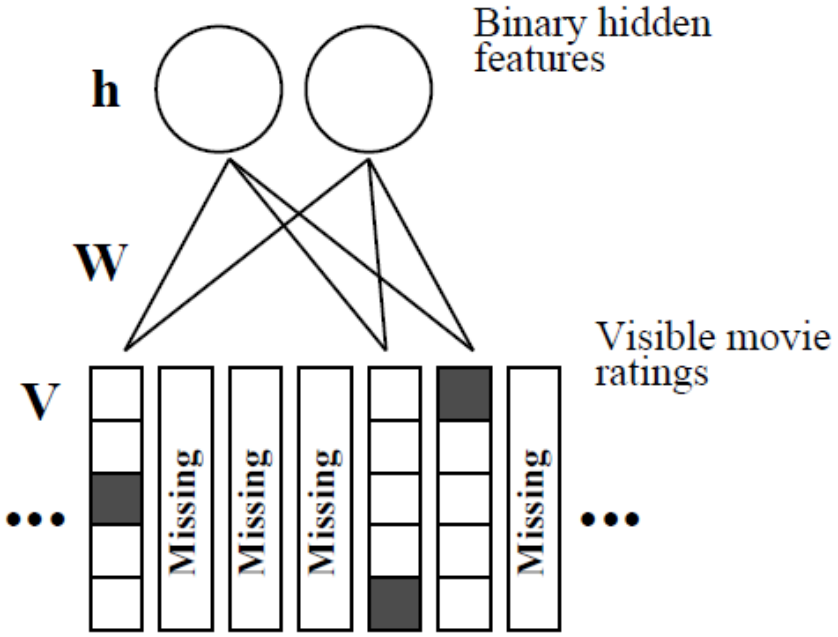
$$\Rightarrow \langle v_i h_j \rangle_{model}$$

RBMs for Collaborative filtering

FACT: The number of movies each user has rated is far less than the total number of movies M.

KEY IDEA #1: For each user build a different RBM . Every RBM has the same number of hidden nodes, but visible units only for the movies rated by that user.

KEY IDEA #2: If 2 users have rated the same movie, their 2 RBMs must use the same weights between the hidden layer and the movie's visible unit. To get the shared parameters just average their updates over all N users.

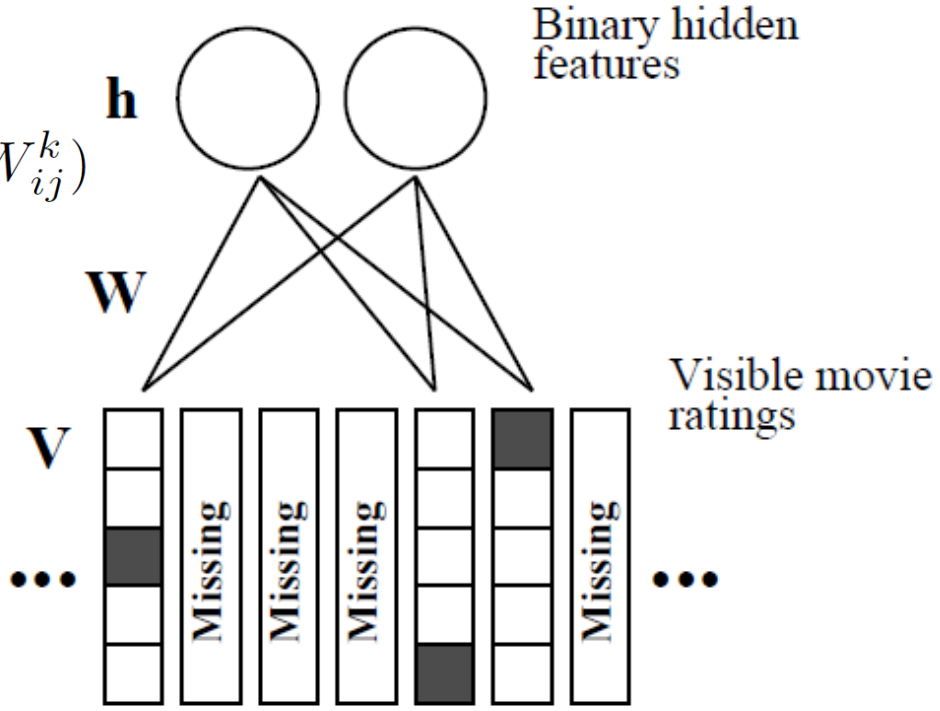


A single user-specific RBM

The user has rated *m* movies. Ratings are modeled as *1* to *K* vectors. Consequently, **V** is a *K*m* matrix. *F* is the number of hidden units.

$$p(v_i^k = 1|\mathbf{h}) = \frac{\exp\left(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k\right)}{\sum_{l=1}^{K} \exp\left(b_i^l + \sum_{j=1}^{F} h_j W_{ij}^l\right)}$$

$$p(h_j = 1|\mathbf{V}) = \sigma\left(b_j + \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k W_{ij}^k\right)$$



Learning – Netflix RBM (1)

The goal is to maximize the likelihood of the visible units with respect to the weights, *W*, and biases, *b*.

$$p(\mathbf{V}) = \sum_{\mathbf{h}} \frac{\exp\left(-E(\mathbf{V}, \mathbf{h})\right)}{\sum_{\mathbf{V}', \mathbf{h}'} \exp\left(-E(\mathbf{V}', \mathbf{h}')\right)}$$

The "energy" term is given by:

$$E(\mathbf{V}, \mathbf{h}) = -\sum_{i=1}^{m}\sum_{j=1}^{F}\sum_{k=1}^{K} W_{ij}^{k} h_j v_i^k + \sum_{i=1}^{m} \log Z_i - \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k b_i^k - \sum_{j=1}^{F} h_j b_j$$

$$Z_i = \sum_{l=1}^{K} \exp\left(b_i^l + \sum_j h_j W_{ij}^l\right)$$

In the following we only discuss learning the weights.
Biases are trained in the same fashion.

Learning – Netflix RBM (2)

Gradient descent update rule:

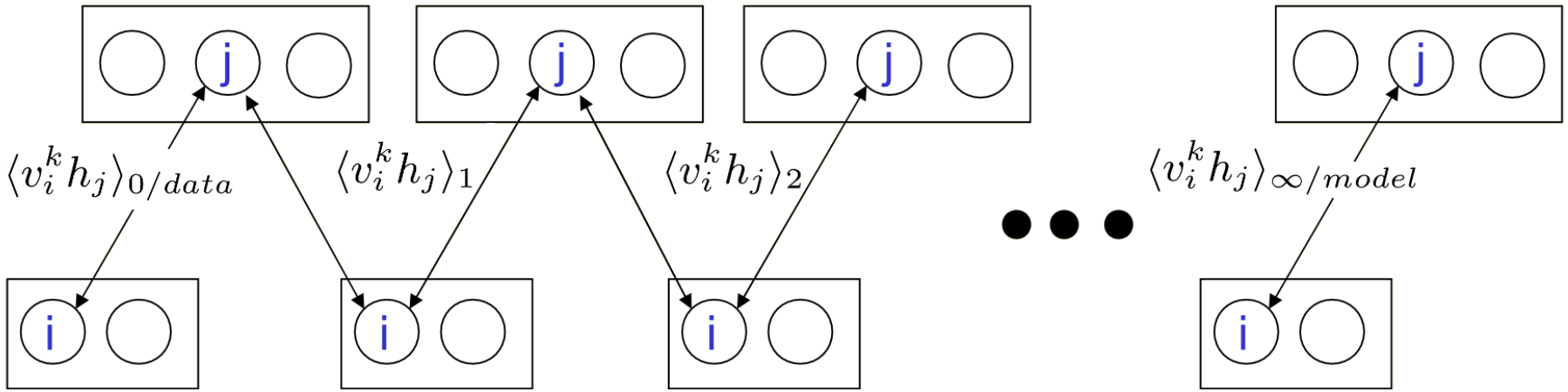
Easy to compute. To be read as "*frequency with which movie **i** with rating **k** and feature **j** are on together when the features are being driven by the observed user-rating data*"

$$\Delta W_{ij}^k = \epsilon \frac{\partial \log p(V)}{\partial W_{ij}^k} = \epsilon \left(\langle v_i^k h_j \rangle_{data} - \langle v_i^k h_j \rangle_{model} \right)$$

Impossible to compute in less than exponential time.

Contrastive Divergence

KEY IDEA: Performing T steps of Gibbs sampling is enough to get a good approximation of $\langle v_i^k h_j \rangle_{model}$



Graph adapted from © Hinton

Remember that the gradient updates are obtained by averaging the gradient updates of each of the N user-specific RBMs!

Making predictions

There are 2 ways of making predictions in the trained model.

1. By directly conditioning on the observed matrix V:

$$p(v_q^k = 1|\mathbf{V}) \propto \sum_{h_1,\ldots,h_p} \exp(-E(v_q^k, \mathbf{V}, \mathbf{h}))$$

2. By performing one iteration of the mean field updates to get a probability distribution over ratings of a movie:

$$\hat{p}_j = p(h_j = 1|\mathbf{V}) = \sigma(b_j + \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k W_{ij}^k)$$

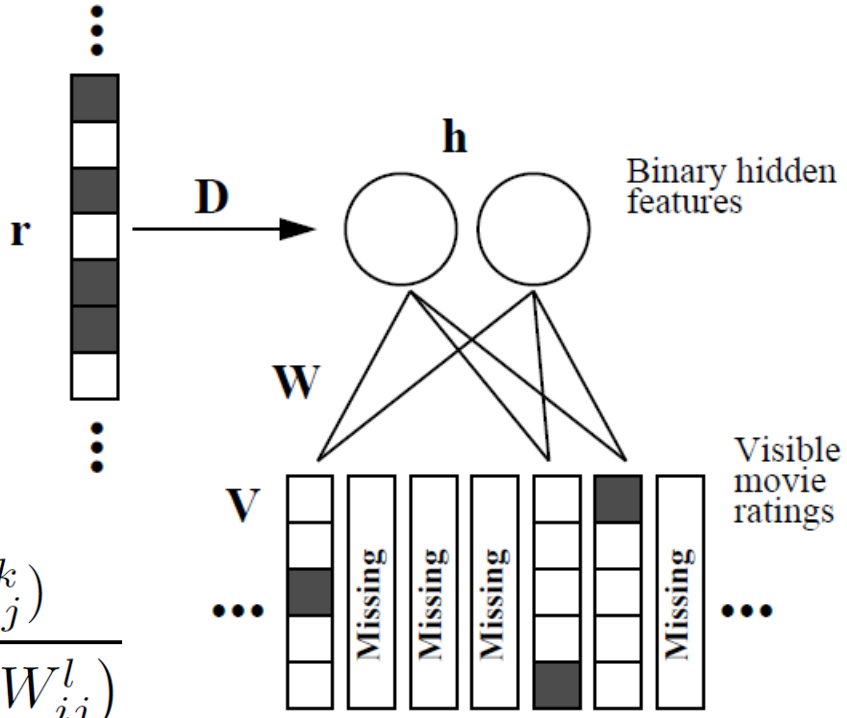
$$p(v_q^k = 1|\hat{\mathbf{p}}) = \frac{\exp\left(b_q^k + \sum_{j=1}^{F} \hat{p}_j W_{qj}^k\right)}{\sum_{l=1}^{K} \exp\left(b_q^l + \sum_{j=1}^{F} \hat{p}_j W_{qj}^l\right)}$$

Even if the first approach is more accurate, the second one is preferred in the experiments because of its higher speed.

Conditional RBMs

FACT: The user/movie pairs in the test set are known as well.

KEY IDEA: Build a binary vector **r** indicating all the movies a user has rated (known + unknown ratings). This vector will affect the states of the hidden units.



$$p(v_i^k = 1|\mathbf{h}) = \frac{\exp\left(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k\right)}{\sum_{l=1}^{K} \exp\left(b_i^l + \sum_{j=1}^{F} h_j W_{ij}^l\right)}$$

$$p(h_j = 1|\mathbf{V}, \mathbf{r}) = \sigma\left(b_j + \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k W_{ij}^k + \sum_{i=1}^{M} r_i D_{ij}\right)$$

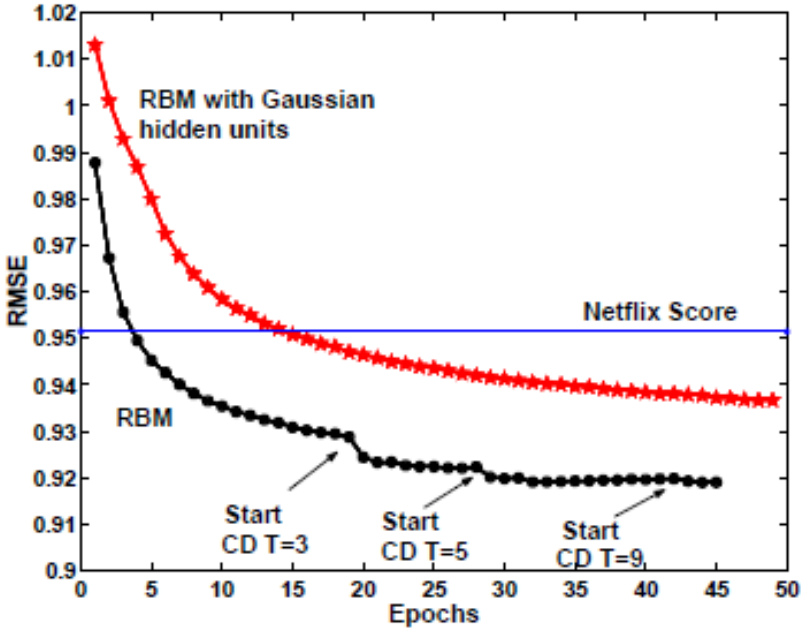
Experiments

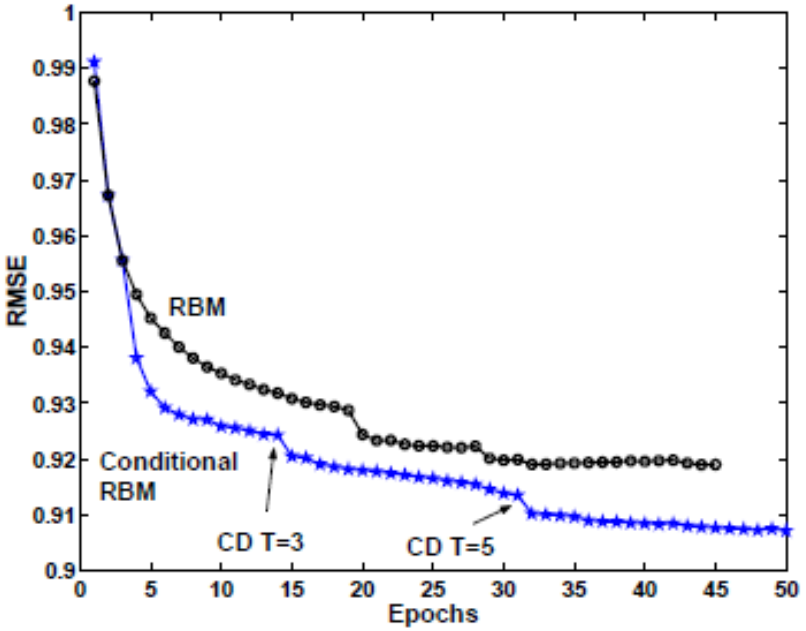
- RBM with *F=100* hidden units
- RBM with *F=100* Gaussian hidden units
  - the only difference is that the conditional Bernoulli distribution over hidden variables is replaced by a Gaussian:

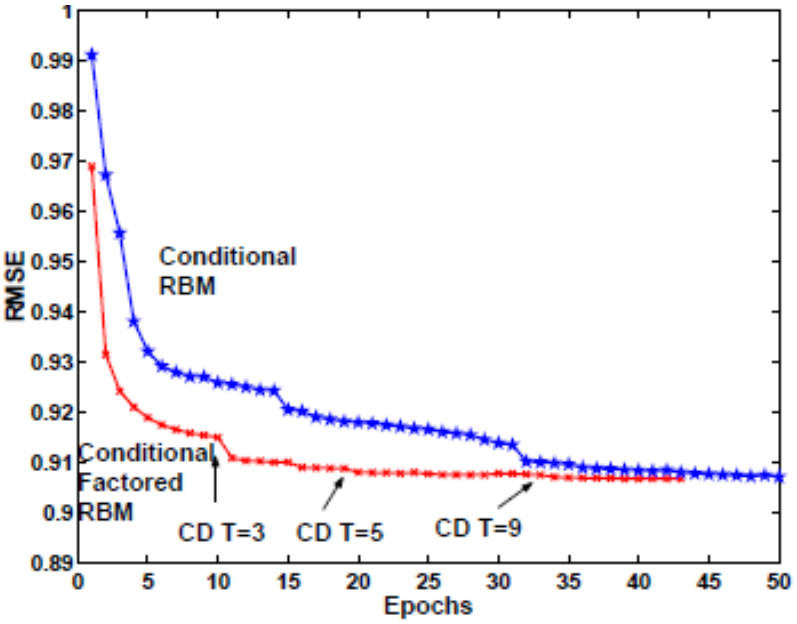
$$p(h_j = h|\mathbf{V}) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(- \frac{\left(h - b_j - \sigma_j \sum_{ik} v_i^k W_{ij}^k\right)^2}{2\sigma_j^2} \right)$$

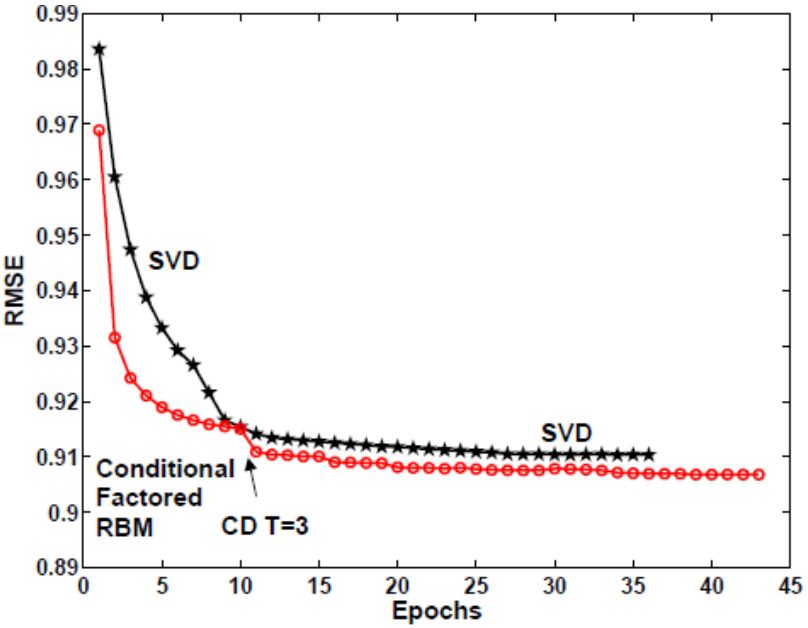
- Conditional RBM with *F=500* hidden units
- Conditional Factored RBM with *F=500* hidden units and *C=30*
  - Each of the *K* weight matrices $\boldsymbol{W}^k$ (size *M*F)* factorizes into product of low rank matrices $\boldsymbol{A}^k$ (size *M*C*) and $\boldsymbol{B}^k$ (size *C*F*).
- SVD with *C=40*

Results


RBM with Gaussian hidden units
Netflix Score
RBM
Start CD T=3
Start CD T=5
Start CD T=9
RMSE
Epochs


RBM
Conditional RBM
CD T=3
CD T=5
RMSE
Epochs


Conditional RBM
Conditional Factored RBM
CD T=3
CD T=5
CD T=9
RMSE
Epochs


SVD
SVD
Conditional Factored RBM
CD T=3
RMSE
Epochs

Conclusions

- ✓ Gaussian hidden units perform worse.
- ✓ Conditional RBMs outperform the simple RBMs, but also make use of information in the test data.
- ✓ Factorization speeds up convergence at early stages. This might also help avoid over-fitting.
- ✓ SVDs and RBMs make different errors → combine them linearly. 6% improvement over Netflix baseline score.

Questions?