10417/10617
Intermediate Deep Learning: Fall2019

Russ Salakhutdinov

Machine Learning Department
rsalakhu@cs.cmu.edu

https://deeplearning-cmu-10417.github.io/

Restricted Boltzmann Machines

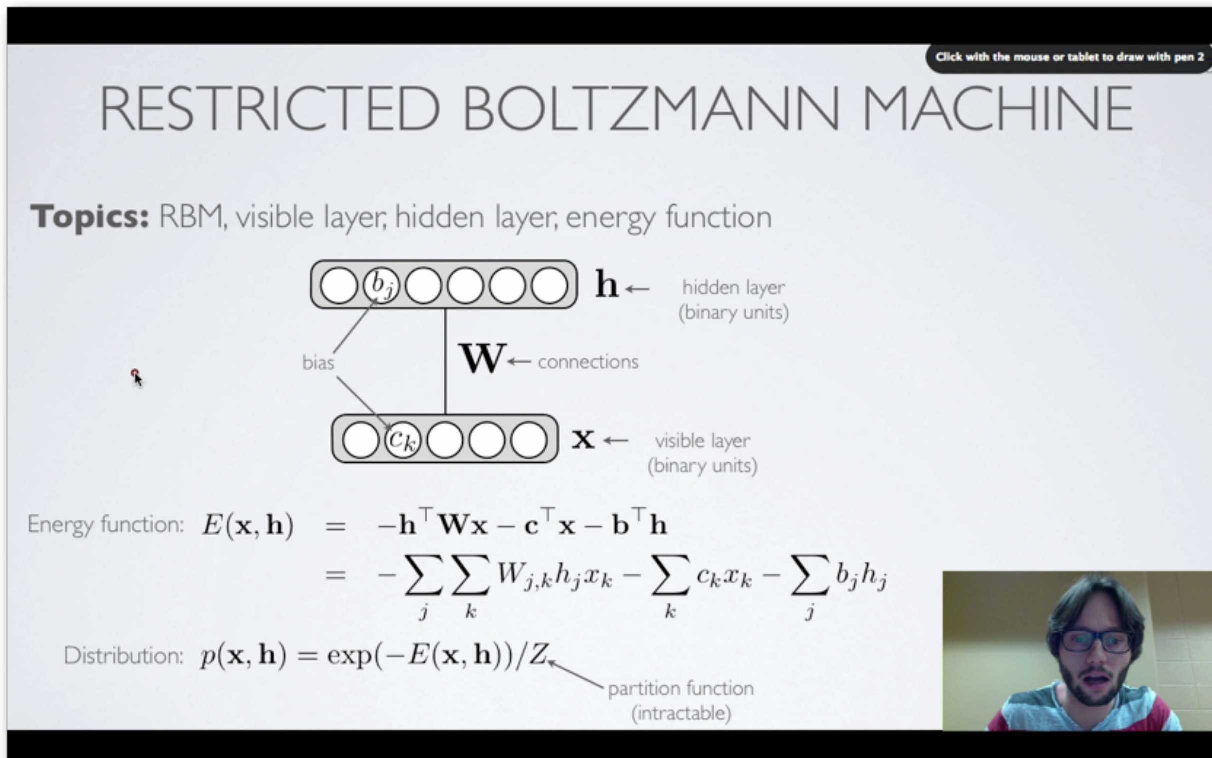
Neural Networks Online Course

- **Disclaimer**: Much of the material and slides for this lecture were borrowed from Hugo Larochelle's class on Neural Networks: https://sites.google.com/site/deeplearningsummerschool2016/

http://info.usherbrooke.ca/hlarochelle/neural_networks

- Hugo's class covers many other topics: convolutional networks, neural language model, Boltzmann machines, autoencoders, sparse coding, etc.

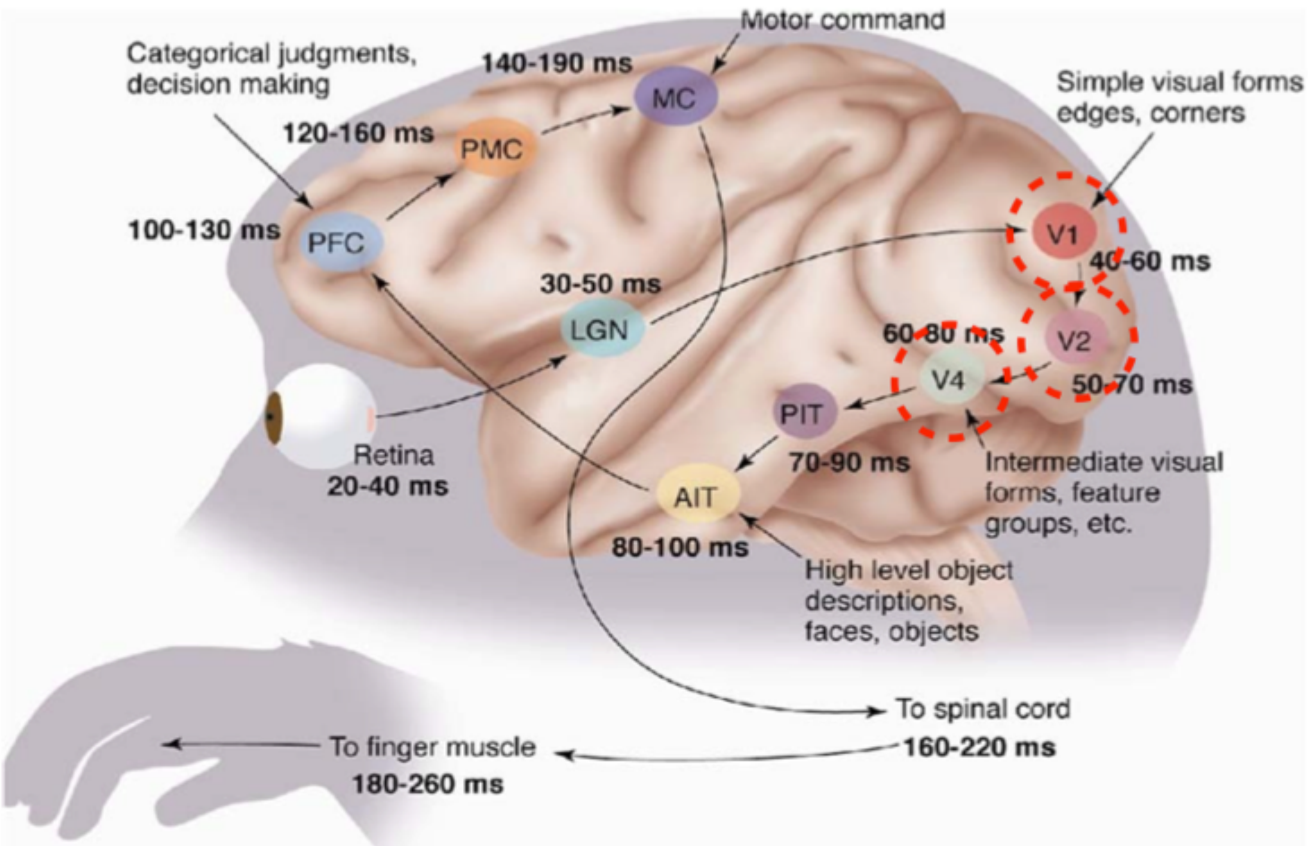
- We will use his material for some of the other lectures.



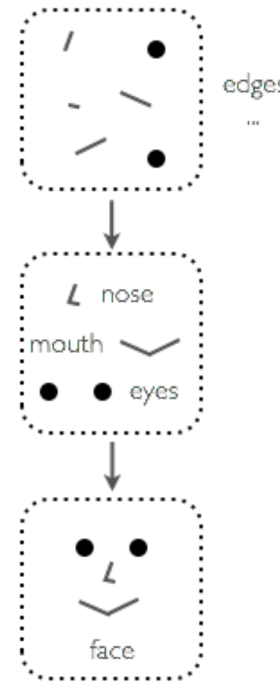
Learning Distributed Representations

- Deep learning is research on learning models with multilayer representations
  - ➢ multilayer (feed-forward) neural networks
  - ➢ multilayer graphical model (deep belief network, deep Boltzmann machine)
- Each layer learns "distributed representation"
  - ➢ Units in a layer are not mutually exclusive
    - each unit is a separate feature of the input
    - two units can be "active" at the same time
  - ➢ Units do not correspond to a partitioning (clustering) of the inputs
    - in clustering, an input can only belong to a single cluster

Inspiration from Visual Cortex

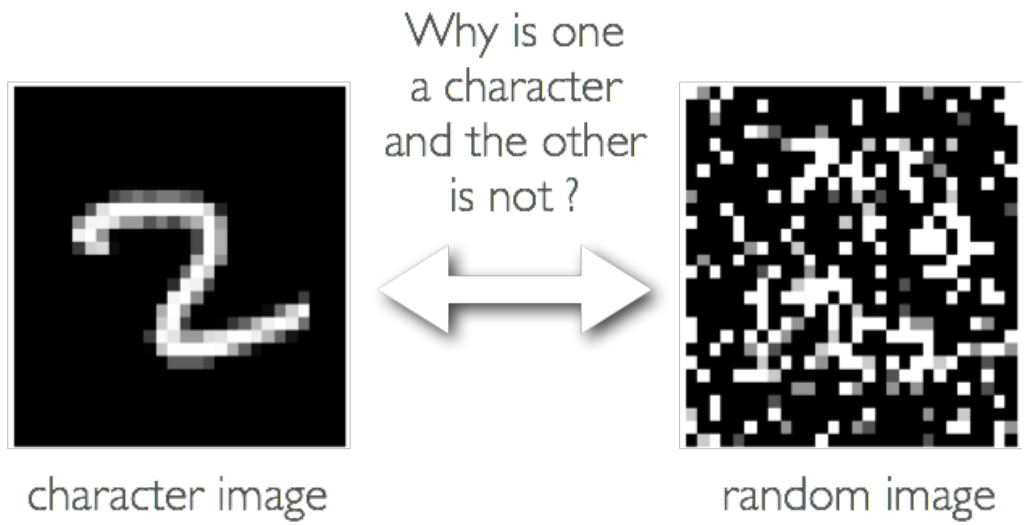


[picture from Simon Thorpe]



Unsupervised Pre-training

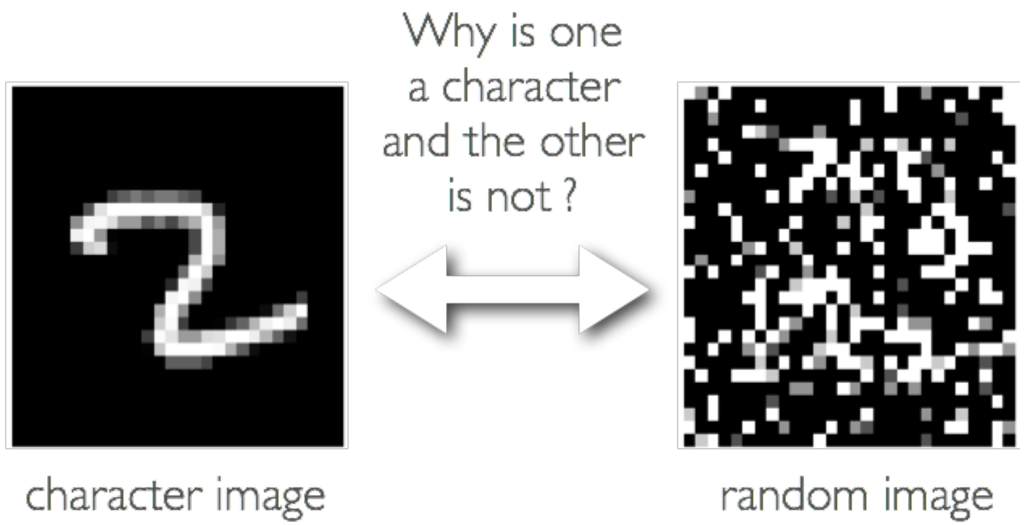
- Initialize hidden layers using unsupervised learning
  - Force network to represent latent structure of input distribution



  - Encourage hidden layers to encode that structure

Unsupervised Pre-training

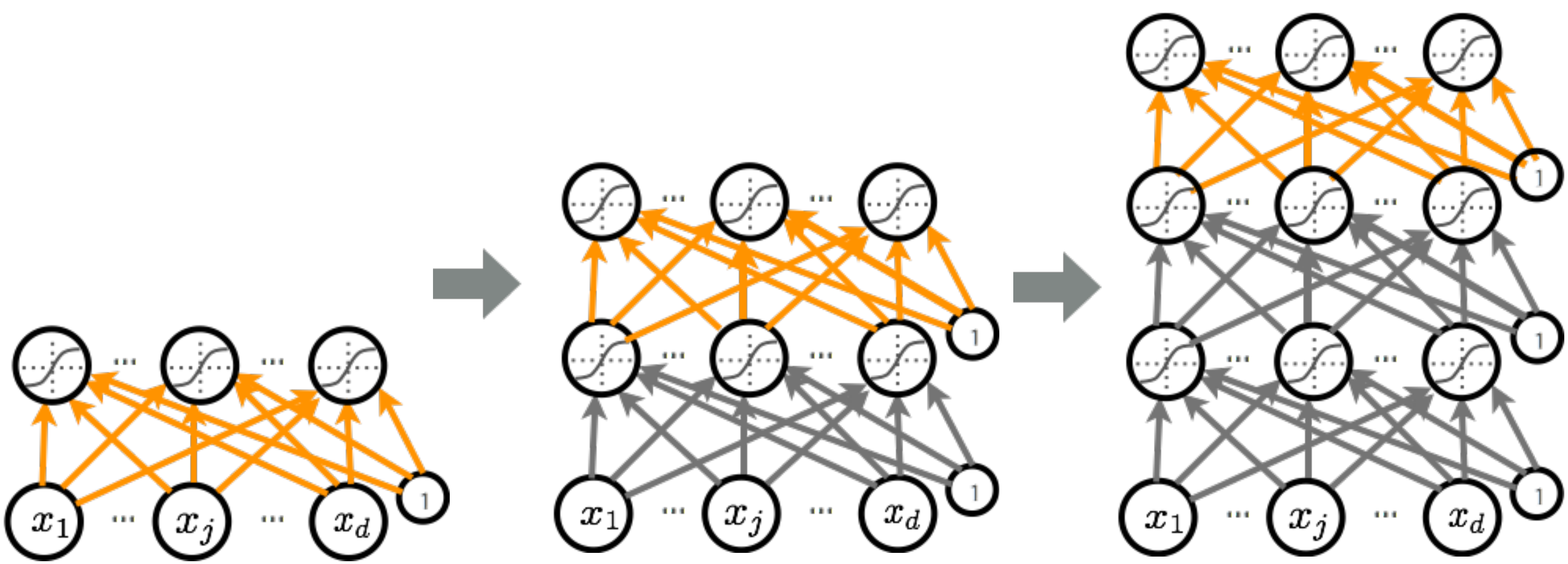
- Initialize hidden layers using unsupervised learning
  - This is a harder task than supervised learning (classification)



  - Hence we expect less overfitting

Pre-training

- We will use a greedy, layer-wise procedure
  - Train one layer at a time with unsupervised criterion
  - Fix the parameters of previous hidden layers
  - Previous layers viewed as feature extraction



Pre-training

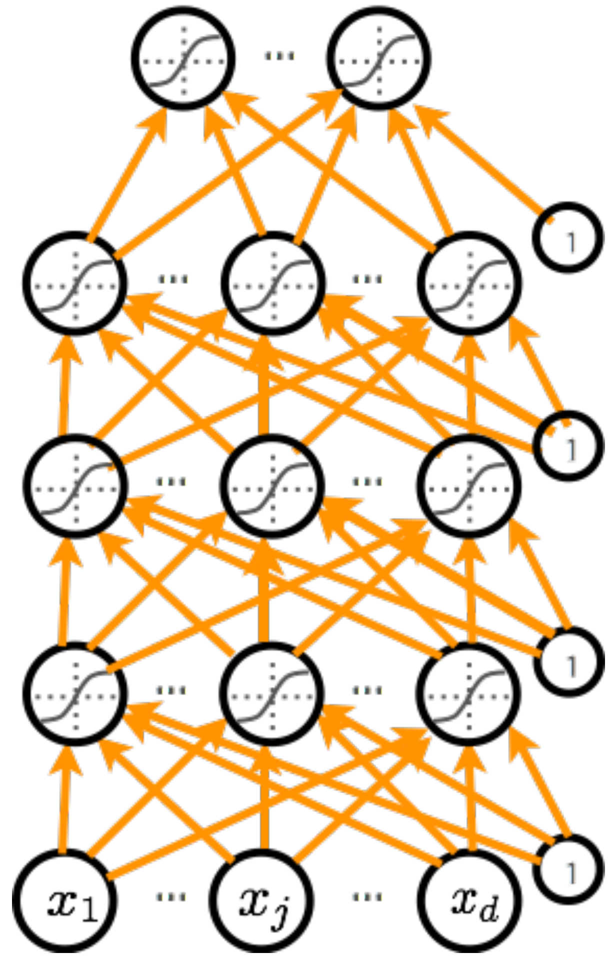
- Unsupervsed Pre-training

  - first layer: find hidden unit features that are more common in training inputs than in random inputs
  - second layer: find combinations of hidden unit features that are more common than random hidden unit features
  - third layer: find combinations of combinations of ...

- Pre-training initializes the parameters in a region such that the near local optima overfit less the data

Fine-tuning

- Once all layers are pre-trained
  - ➢ add output layer
  - ➢ train the whole network using supervised learning
- Supervised learning is performed as in a regular network
  - ➢ forward propagation, backpropagation and update
- We call this last phase fine-tuning
  - ➢ all parameters are "tuned" for the supervised task at hand
  - ➢ representation is adjusted to be more discriminative

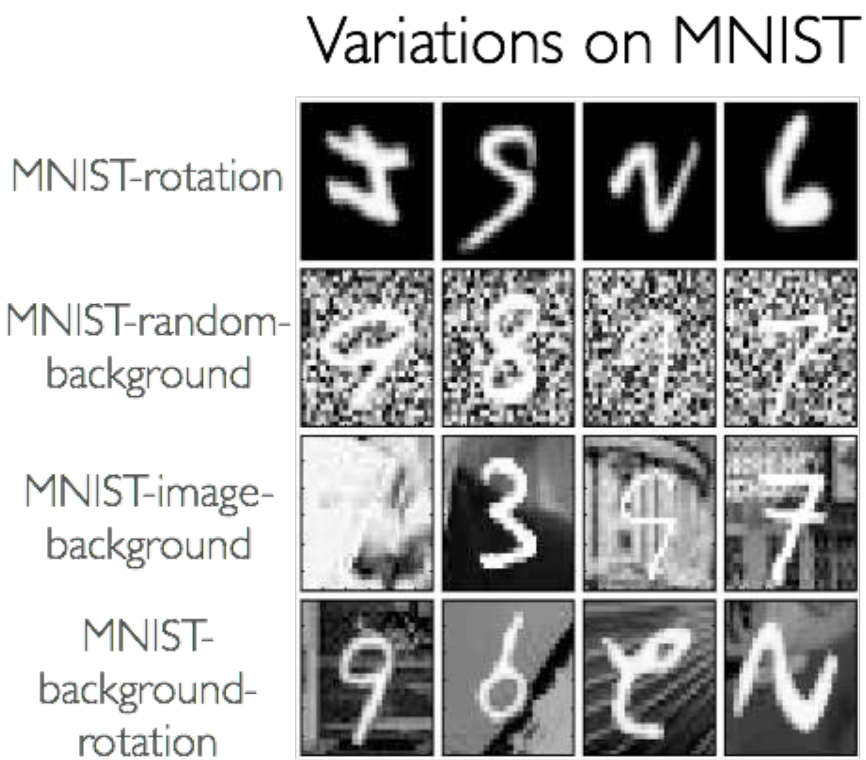


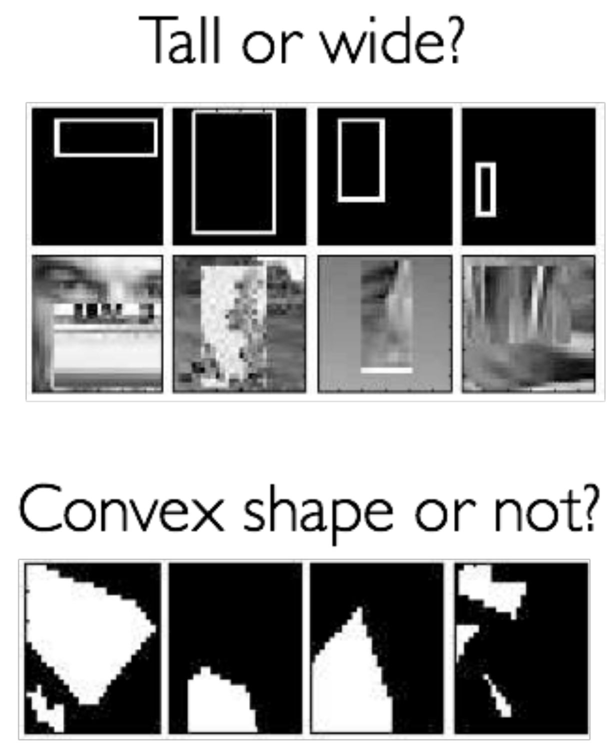
Stacking RBMs, Autoencoders

- Stacked Restricted Boltzmann Machines:
  - Hinton, Teh and Osindero suggested this procedure with RBMs,:
    A fast learning algorithm for deep belief nets.
  - To recognize shapes, first learn to generate images.
    Hinton, 2006.
- Stacked autoencoders, sparse-coding models, etc.
  - Bengio, Lamblin, Popovici and Larochelle (stacked autoencoders)
  - Ranzato, Poultney, Chopra and LeCun (stacked sparse coding models)
- Lots of others started stacking models together.

Example

- Datasets generated with varying number of factors of variations



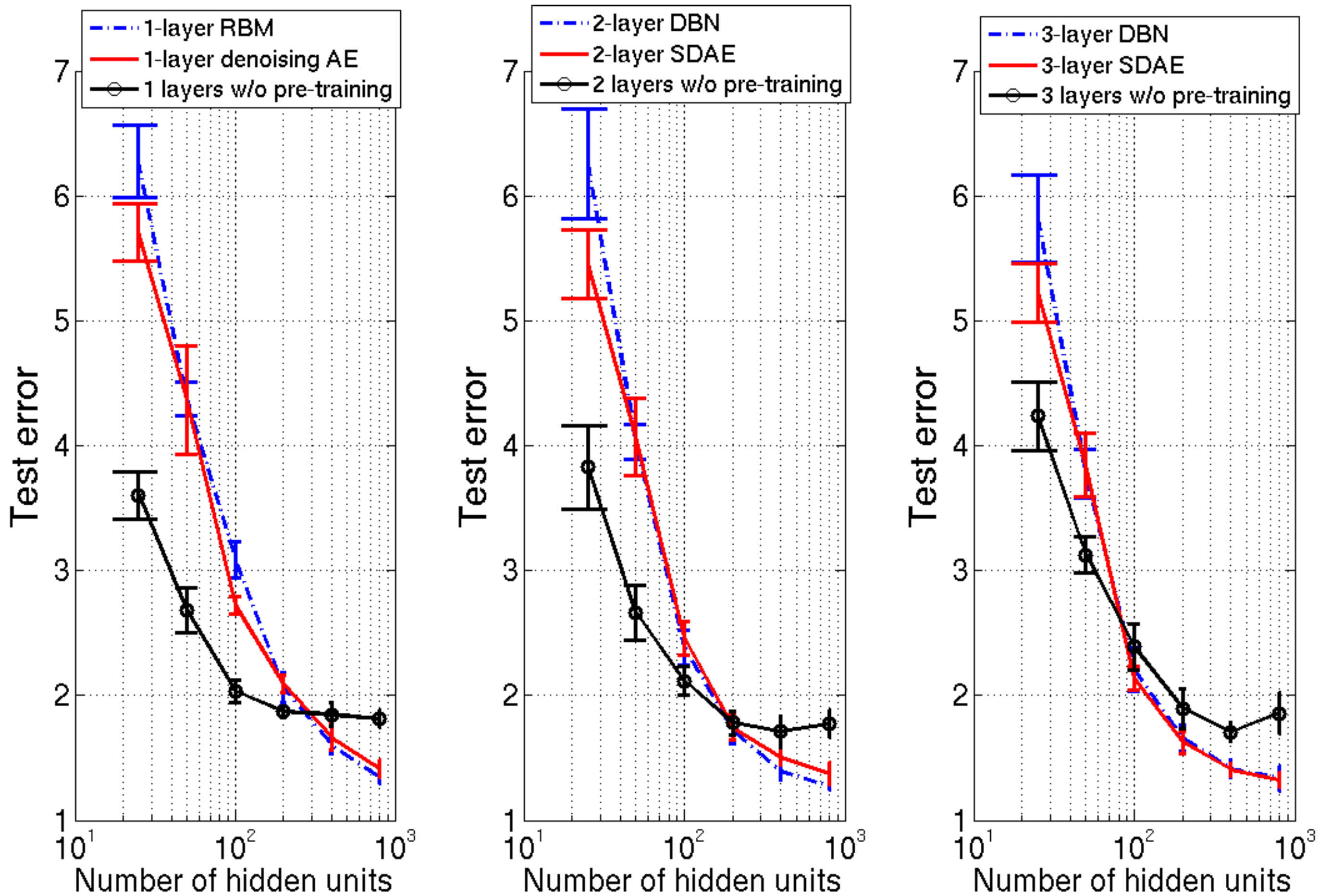


An Empirical Evaluation of Deep Architectures on Problems with Many Factors of Variation, Larochelle, Erhan, Courville, Bergstra and Bengio, 2007

Impact of Initialization

Network Type	Depth	MNIST-small classif. test error	MNIST-rotation classif. test error
Deep net	1	**4.14** % ± 0.17	15.22 % ± 0.31
	2	**4.03** % ± 0.17	**10.63** % ± 0.27
	3	**4.24** % ± 0.18	11.98 % ± 0.28
	4	4.47 % ± 0.18	11.73 % ± 0.29
Deep net + autoencoder	1	3.87 % ± 0.17	11.43% ± 0.28
	2	**3.38** % ± 0.16	9.88 % ± 0.26
	3	**3.37** % ± 0.16	**9.22** % ± 0.25
	4	**3.39** % ± 0.16	**9.20** % ± 0.25
Deep net + RBM	1	3.17 % ± 0.15	10.47 % ± 0.27
	2	**2.74** % ± 0.14	9.54 % ± 0.26
	3	**2.71** % ± 0.14	**8.80** % ± 0.25
	4	**2.72** % ± 0.14	**8.83** % ± 0.24

Impact of Pretraining



Acts as a regularizer: overfits less with large capacity, underfits with small capacity

Performance on Different Datasets

Stacked Autoencoders	Stacked RBMS	Stacked Denoising Autoencoders
SAA-3	**DBN-3**	**SdA-3** (ν)
3.46±0.16	3.11±0.15	**2.80±0.14** (10%)
10.30±0.27	**10.30±0.27**	**10.29±0.27** (10%)
11.28±0.28	**6.73±0.22**	10.38±0.27 (40%)
23.00±0.37	**16.31±0.32**	**16.68±0.33** (25%)
51.93±0.44	47.39±0.44	**44.49±0.44** (25%)
2.41±0.13	2.60±0.14	**1.99±0.12** (10%)
24.05±0.37	22.50±0.37	**21.59±0.36** (25%)
18.41±0.34	**18.63±0.34**	**19.06±0.34** (10%)

Extracting and Composing Robust Features with Denoising Autoencoders,
Vincent, Larochelle, Bengio and Manzagol, 2008.

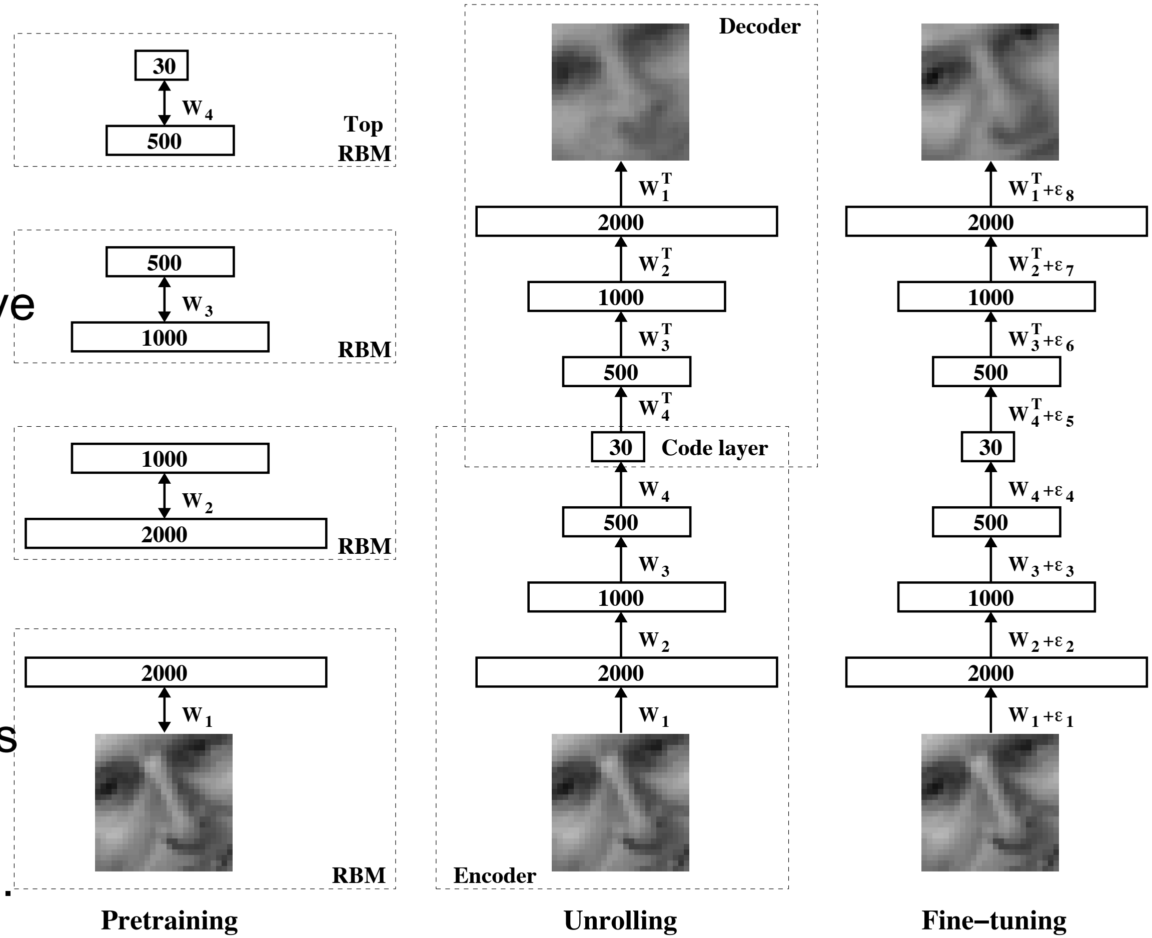
Deep Autoencoder

- Pre-training can be used to initialize a deep autoencoder

➢ Pre-training initializes the optimization problem in a region with better local optima of the training objective

➢ Each RBM used to initialize parameters both in encoder and decoder ("unrolling")

➢ Better optimization algorithms can also help: Deep learning via Hessian-free optimization. Martens, 2010

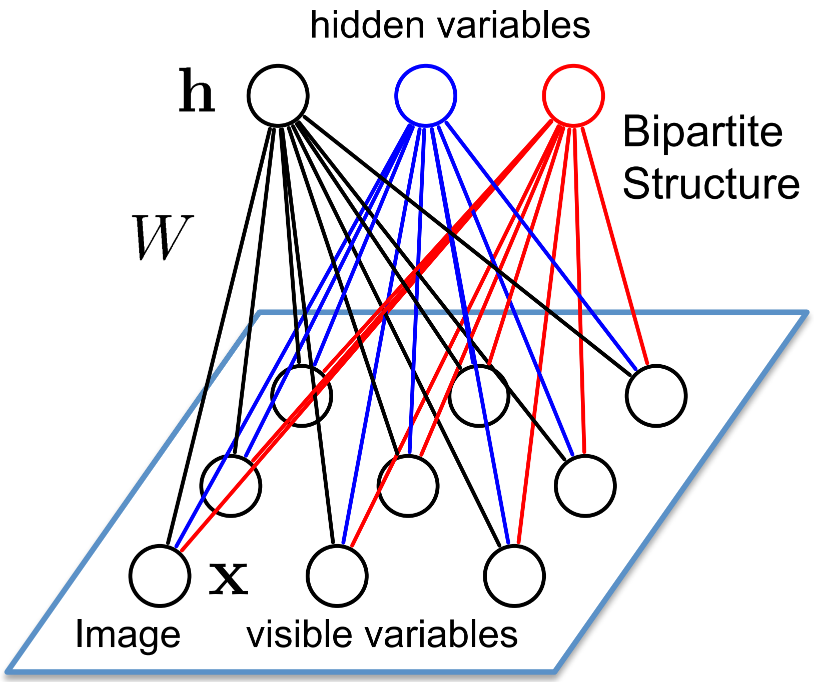


Unsupervised Learning

- Unsupervised learning: we only use the inputs $\mathbf{x}^{(t)}$ for learning
  - automatically extract meaningful features for your data
  - leverage the availability of unlabeled data
  - add a data-dependent regularizer to training ($-\log p(\mathbf{x}^{(t)})$)

- We will consider 3 models for unsupervised learning that will form the basic building blocks for deeper models:
  - Restricted Boltzmann Machines
  - Autoencoders
  - Sparse coding models

Restricted Boltzmann Machines



- Undirected bipartite graphical model
- Stochastic binary visible variables:

$$\mathbf{x} \in \{\mathbf{0}, \mathbf{1}\}^{\mathbf{D}}$$

- Stochastic binary hidden variables:

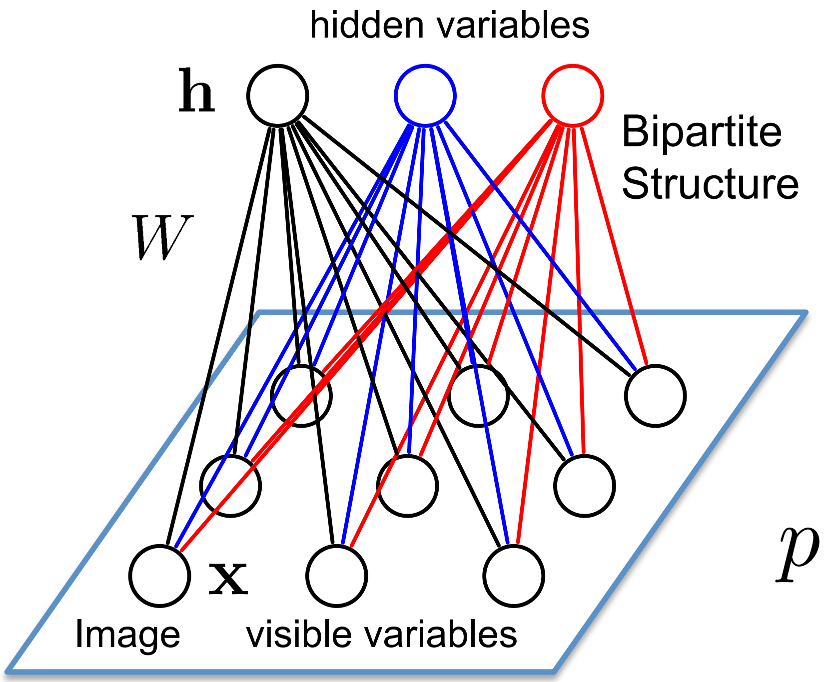
$$\mathbf{h} \in \{0, 1\}^{F}$$

- The energy of the joint configuration:

$$\begin{aligned} E(\mathbf{x}, \mathbf{h}) &= -\mathbf{h}^\top \mathbf{W}\mathbf{x} - \mathbf{c}^\top \mathbf{x} - \mathbf{b}^\top \mathbf{h} \\ &= -\sum_j \sum_k W_{j,k} h_j x_k - \sum_k c_k x_k - \sum_j b_j h_j \end{aligned}$$

Markov random fields, Boltzmann machines, log-linear models.

Restricted Boltzmann Machines



• Probability of the joint configuration is given by the Boltzmann distribution:

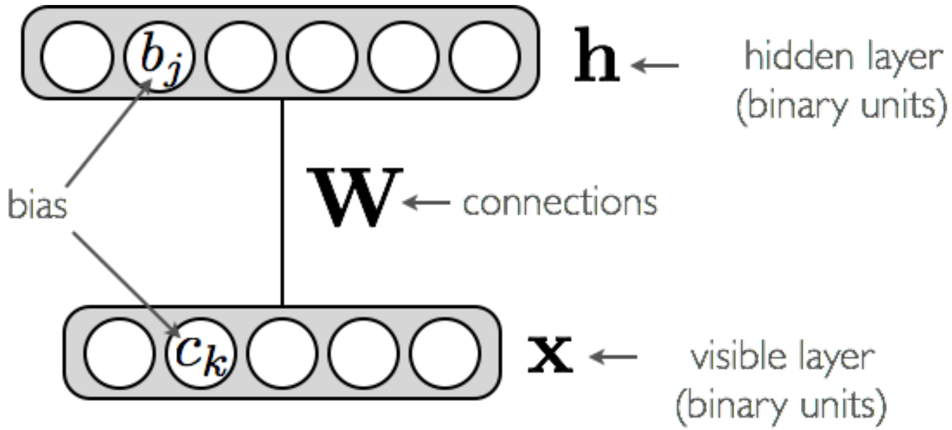
$$p(\mathbf{x}, \mathbf{h}) = \exp(-E(\mathbf{x}, \mathbf{h}))/Z$$

Partition function (intractable)

$$Z = \sum_{\mathbf{x},\mathbf{h}} \exp\big(- E(\mathbf{x}, \mathbf{h})\big)$$

Markov random fields, Boltzmann machines, log-linear models.

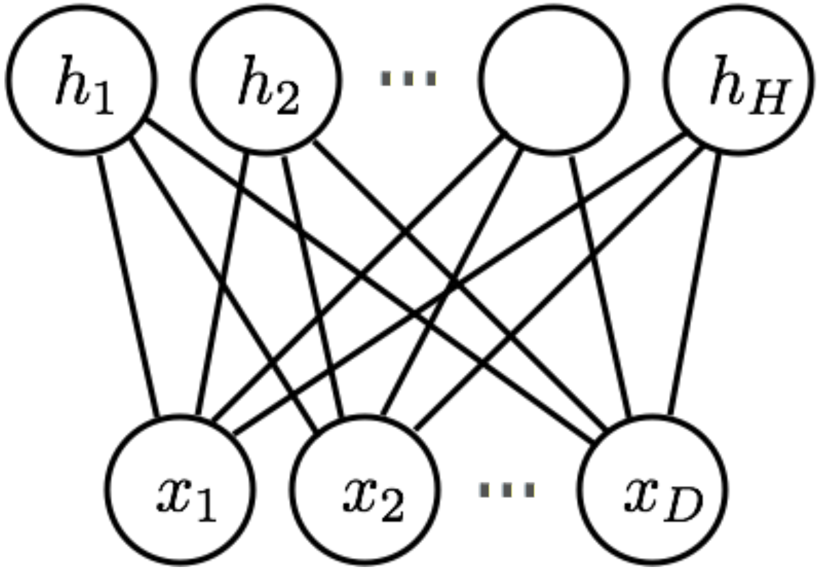
Restricted Boltzmann Machines



$$
\begin{aligned}
p(\mathbf{x},\mathbf{h}) &= \exp(-E(\mathbf{x},\mathbf{h}))/Z \\
&= \exp(\mathbf{h}^\top \mathbf{W}\mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h})/Z \\
&= \underbrace{\exp(\mathbf{h}^\top \mathbf{W}\mathbf{x})\exp(\mathbf{c}^\top \mathbf{x})\exp(\mathbf{b}^\top \mathbf{h})}_{\text{Factors}}/Z
\end{aligned}
$$

- The notation based on an energy function is simply an alternative to the representation as the product of factors

Restricted Boltzmann Machines



Pair-wise factors

$$p(\mathbf{x}, \mathbf{h}) = \frac{1}{Z} \prod_j \prod_k \exp(W_{j,k} h_j x_k)$$

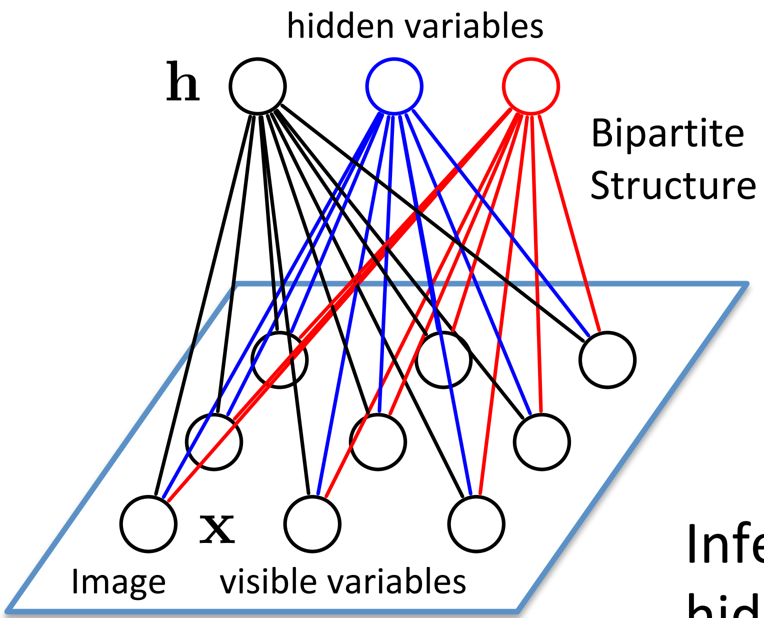
$$\prod_k \exp(c_k x_k)$$

$$\prod_j \exp(b_j h_j)$$

Unary factors

• The scalar visualization is more informative of the structure within the vectors.

Inference



Restricted: No interaction between hidden variables

Inferring the distribution over the hidden variables is easy:

$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

Factorizes: Easy to compute

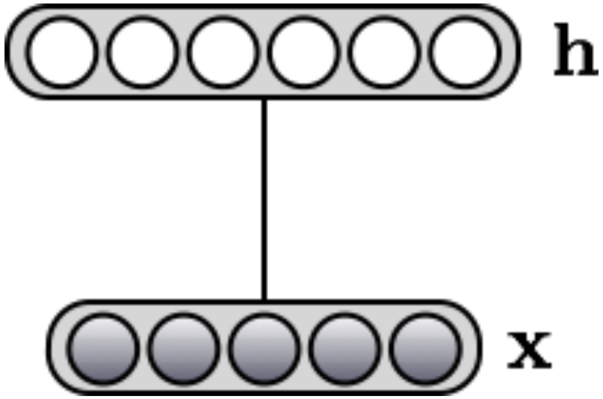
Similarly:

$$p(\mathbf{x}|\mathbf{h}) = \prod_k p(x_k|\mathbf{h})$$

Markov random fields, Boltzmann machines, log-linear models.

Inference

- Conditional Distributions:

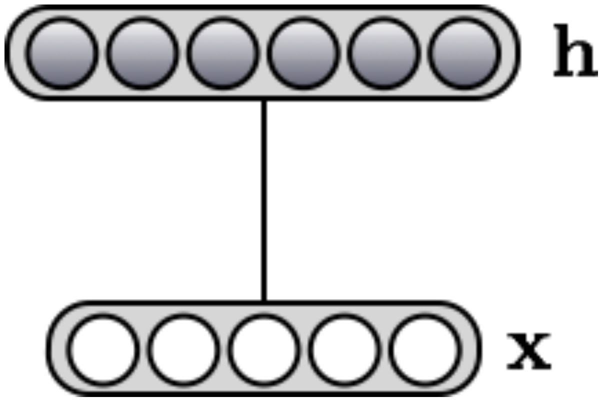


$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

$$p(h_j = 1|\mathbf{x}) = \frac{1}{1 + \exp(-(b_j + \mathbf{W}_{j}.\mathbf{x}))}$$

$$= \mathrm{sigm}(b_j + \mathbf{W}_{j}.\mathbf{x})$$

j[th] row of W



$$p(\mathbf{x}|\mathbf{h}) = \prod_k p(x_k|\mathbf{h})$$

$$p(x_k = 1|\mathbf{h}) = \frac{1}{1 + \exp(-(c_k + \mathbf{h}^\top \mathbf{W}_{\cdot k}))}$$

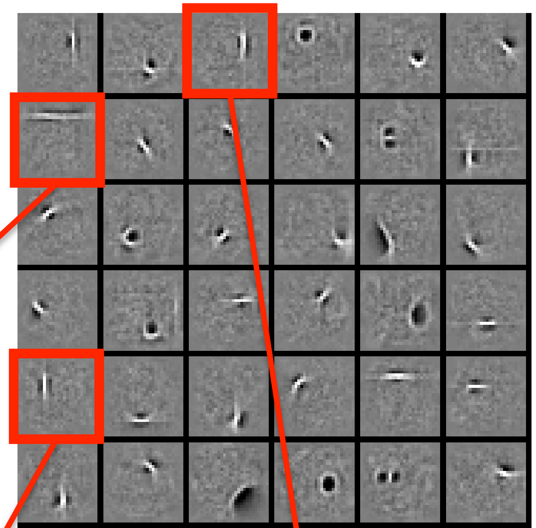
$$= \mathrm{sigm}(c_k + \mathbf{h}^\top \mathbf{W}_{\cdot k})$$

k[th] column of W

Learning Features

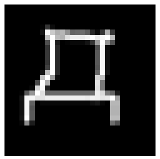
Observed Data
Subset of 25,000 characters

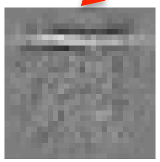
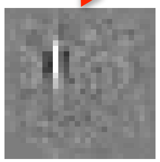
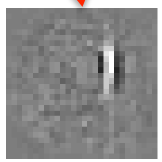
Learned W: "edges"
Subset of 1000 features



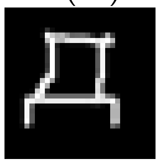
Most hidden variables are off

New Image: $p(h_7 = 1|v)$ $p(h_{29} = 1|v)$

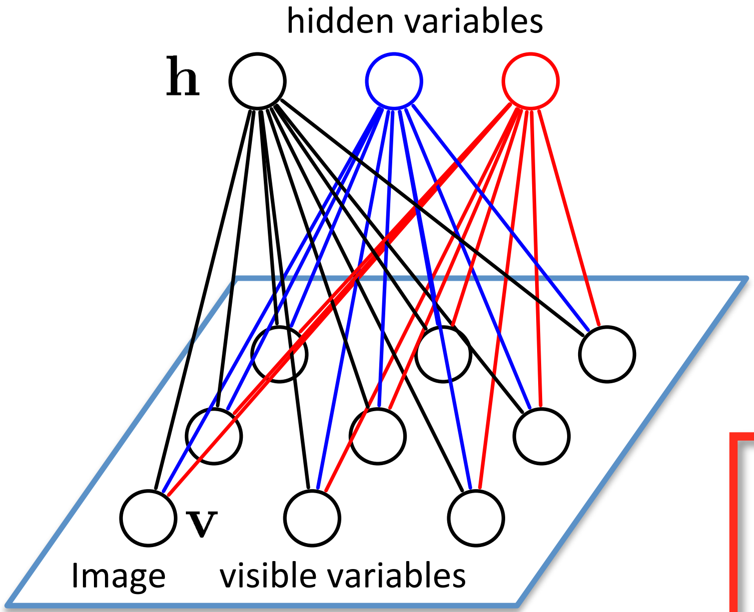


$= \sigma\Big(0.99 \times$  $+ \; 0.97 \times$  $+ \; 0.82 \times$  $\dots\Big)$

$\sigma(x) = \frac{1}{1+\exp(-x)}$ Logistic Function: Suitable for modeling binary images

Represent:  as $P(\mathbf{h}|\mathbf{v}) = [0,\ 0,\ 0.82,\ 0,\ 0,\ 0.99,\ 0,\ 0\ \dots\ ]$

Model Learning



- Given a set of *i.i.d.* training examples we want to minimize the average negative log-likelihood (NLL):

$$\frac{1}{T}\sum_t l(f(\mathbf{x}^{(t)})) = \frac{1}{T}\sum_t -\log p(\mathbf{x}^{(t)})$$

Remember:

$$p(\mathbf{x},\mathbf{h}) \quad = \quad \exp(-E(\mathbf{x},\mathbf{h}))/Z$$

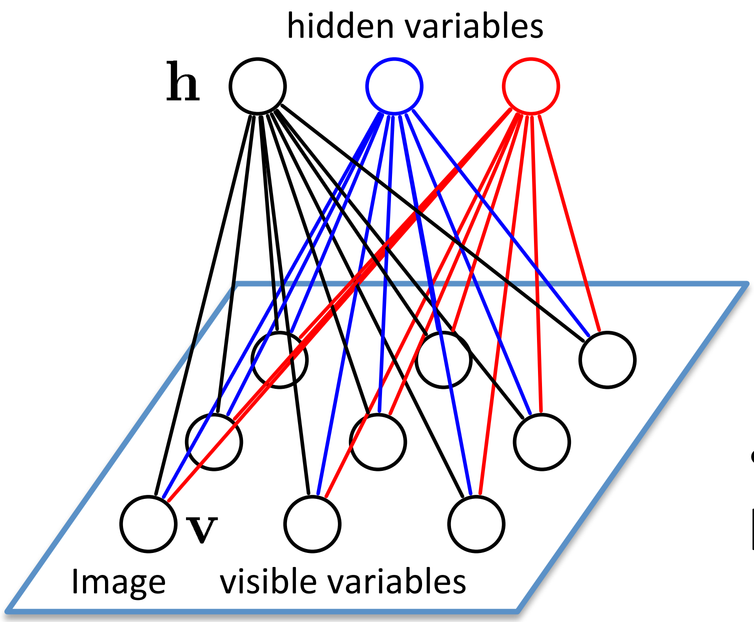
- Derivative of the negative log-likelihood objective:

$$\frac{\partial -\log p(\mathbf{x}^{(t)})}{\partial \theta} = \mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)},\mathbf{h})}{\partial \theta}\,\middle|\,\mathbf{x}^{(t)}\right] - \mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial \theta}\right]$$

Positive Phase

Negative Phase
Hard to compute

Model Learning



$$p(\mathbf{x}, \mathbf{h}) = \exp(-E(\mathbf{x}, \mathbf{h}))/Z$$

$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

- Derivative of the negative log-likelihood objective:

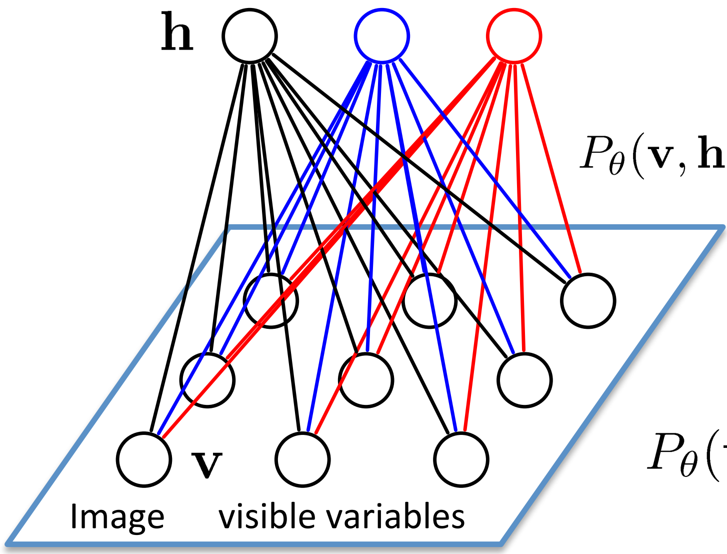
$$\frac{\partial - \log p(\mathbf{x}^{(t)})}{\partial \theta} = \mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)}, \mathbf{h})}{\partial \theta} \Big| \mathbf{x}^{(t)}\right] - \mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \theta}\right]$$

Data-Dependent Expectations w.r.t P(h|x)

Model: Expectation w.r.t joint P(x,h)

- Second term: intractable due to exponential number of configurations.

Gaussian Bernoulli RBMs



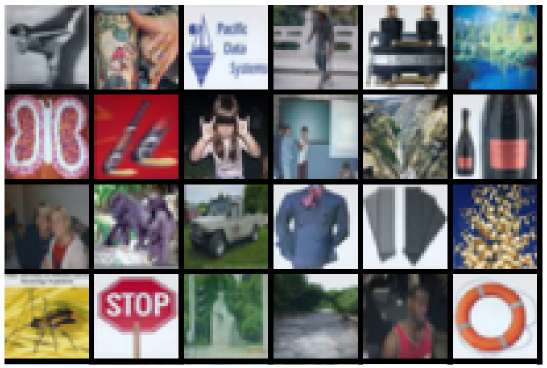
Pair-wise **Unary**

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{i=1}^{D}\sum_{j=1}^{F} W_{ij} h_j \frac{v_i}{\sigma_i} + \sum_{i=1}^{D} \frac{(v_i - b_i)^2}{2\sigma_i^2} + \sum_{j=1}^{F} a_j h_j\right)$$

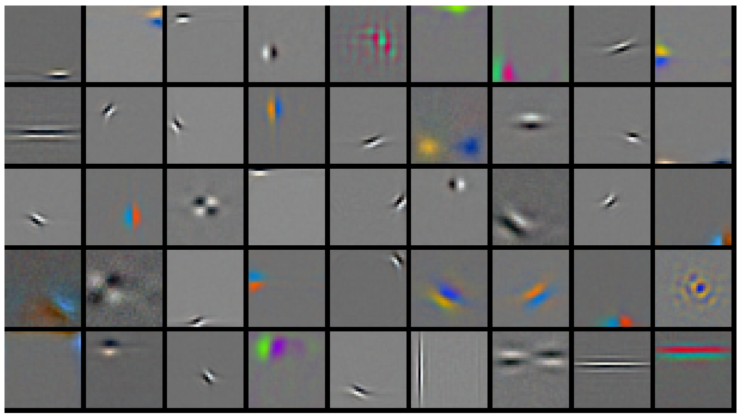
$$\theta = \{W, a, b\}$$

$$P_\theta(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} P_\theta(v_i|\mathbf{h}) = \prod_{i=1}^{D} \mathcal{N}\left(b_i + \sum_{j=1}^{F} W_{ij} h_j, \sigma_i^2\right)$$

4 million **unlabelled** images



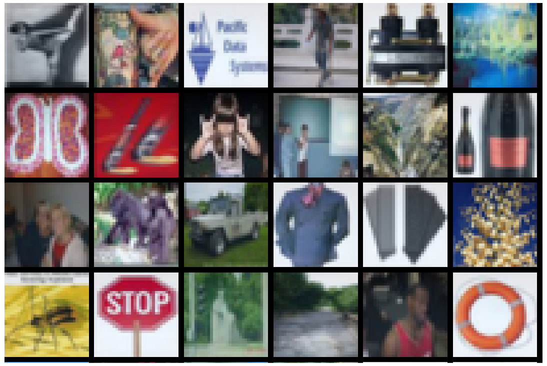
Learned features (out of 10,000)



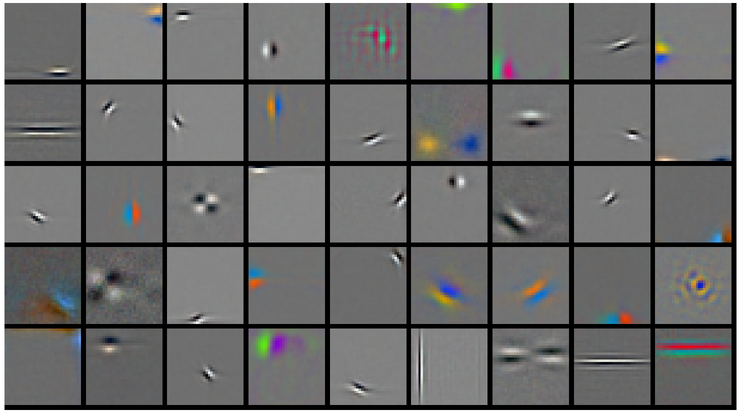
(Notation: vector x is replaced with v).

Gaussian Bernoulli RBMs

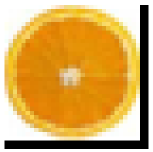
4 million **unlabelled** images



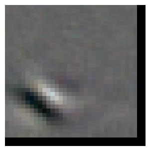
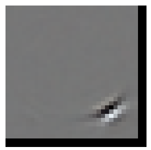
Learned features (out of 10,000)



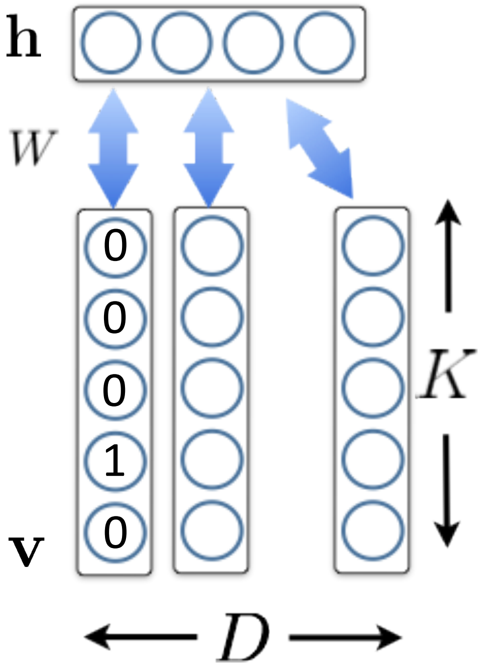
$p(h_7 = 1|v)$ $\quad$ $p(h_{29} = 1|v)$



New Image

$= 0.9 *$  $+ \ 0.8 *$  $+ 0.6 *$  $\ldots$

RBMs for Word Counts



$$p(\mathbf{x}, \mathbf{h}) = \exp(-E(\mathbf{x}, \mathbf{h}))/Z$$

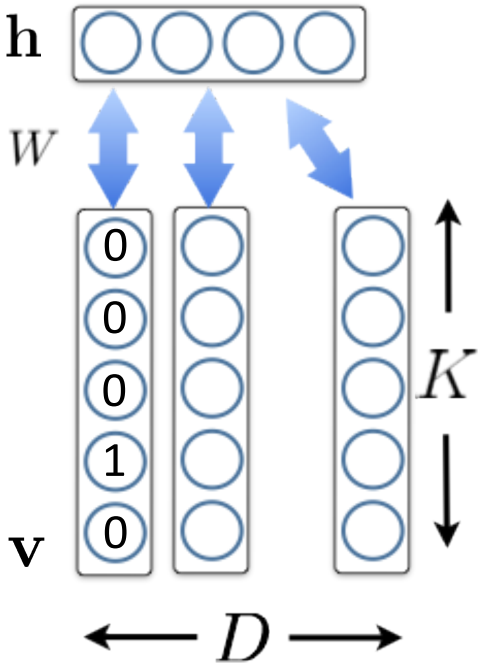
$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

Replicated Softmax Model: undirected topic model:

- Stochastic 1-of-K visible variables.
- Stochastic binary hidden variables $\mathbf{h} \in \{0, 1\}^F$.
- Bipartite connections.

(Salakhutdinov & Hinton, NIPS 2010, Srivastava & Salakhutdinov, NIPS 2012)

RBMs for Word Counts



$$p(\mathbf{x}, \mathbf{h}) = \exp(-E(\mathbf{x}, \mathbf{h}))/Z$$

$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

Reuters dataset:
804,414 **unlabeled**
newswire stories
Bag-of-Words

Learned features: ``topics''

russian	clinton	computer	trade	stock
russia	house	system	country	wall
moscow	president	product	import	street
yeltsin	bill	software	world	point
soviet	congress	develop	economy	dow

RBMs for Word Counts

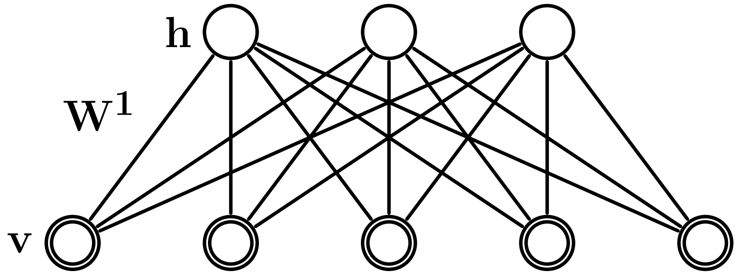
One-step reconstruction from the Replicated Softmax model.

Input	**Reconstruction**
chocolate, cake	cake, chocolate, sweets, dessert, cupcake, food, sugar, cream, birthday
nyc	nyc, newyork, brooklyn, queens, gothamist, manhattan, subway, streetart
dog	dog, puppy, perro, dogs, pet, filmshots, tongue, pets, nose, animal
flower, high, 花	flower, 花, high, japan, sakura, 日本, blossom, tokyo, lily, cherry
girl, rain, station, norway	norway, station, rain, girl, oslo, train, umbrella, wet, railway, weather
fun, life, children	children, fun, life, kids, child, playing, boys, kid, play, love
forest, blur	forest, blur, woods, motion, trees, movement, path, trail, green, focus
españa, agua, granada	españa, agua, spain, granada, water, andalucía, naturaleza, galicia, nieve

Collaborative Filtering

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{ijk} W_{ij}^k v_i^k h_j + \sum_{ik} b_i^k v_i^k + \sum_j a_j h_j \right)$$

Binary hidden: user preferences



Multinomial visible: user ratings

Netflix dataset:
480,189 users
17,770 movies
Over 100 million ratings

Learned features: ``genre''

Fahrenheit 9/11
Bowling for Columbine
The People vs. Larry Flynt
Canadian Bacon
La Dolce Vita

Independence Day
The Day After Tomorrow
Con Air
Men in Black II
Men in Black

Friday the 13th
The Texas Chainsaw Massacre
Children of the Corn
Child's Play
The Return of Michael Myers

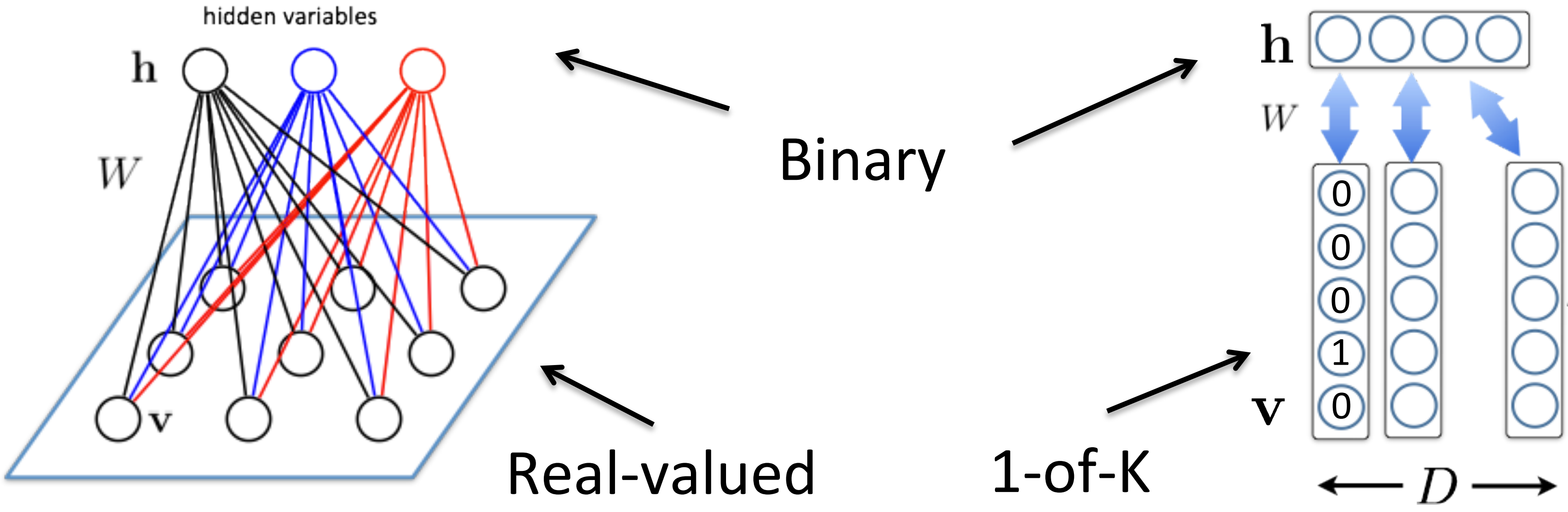
Scary Movie
Naked Gun
Hot Shots!
American Pie
Police Academy

State-of-the-art performance on the Netflix dataset.

(Salakhutdinov, Mnih, Hinton, ICML 2007)

Different Data Modalities

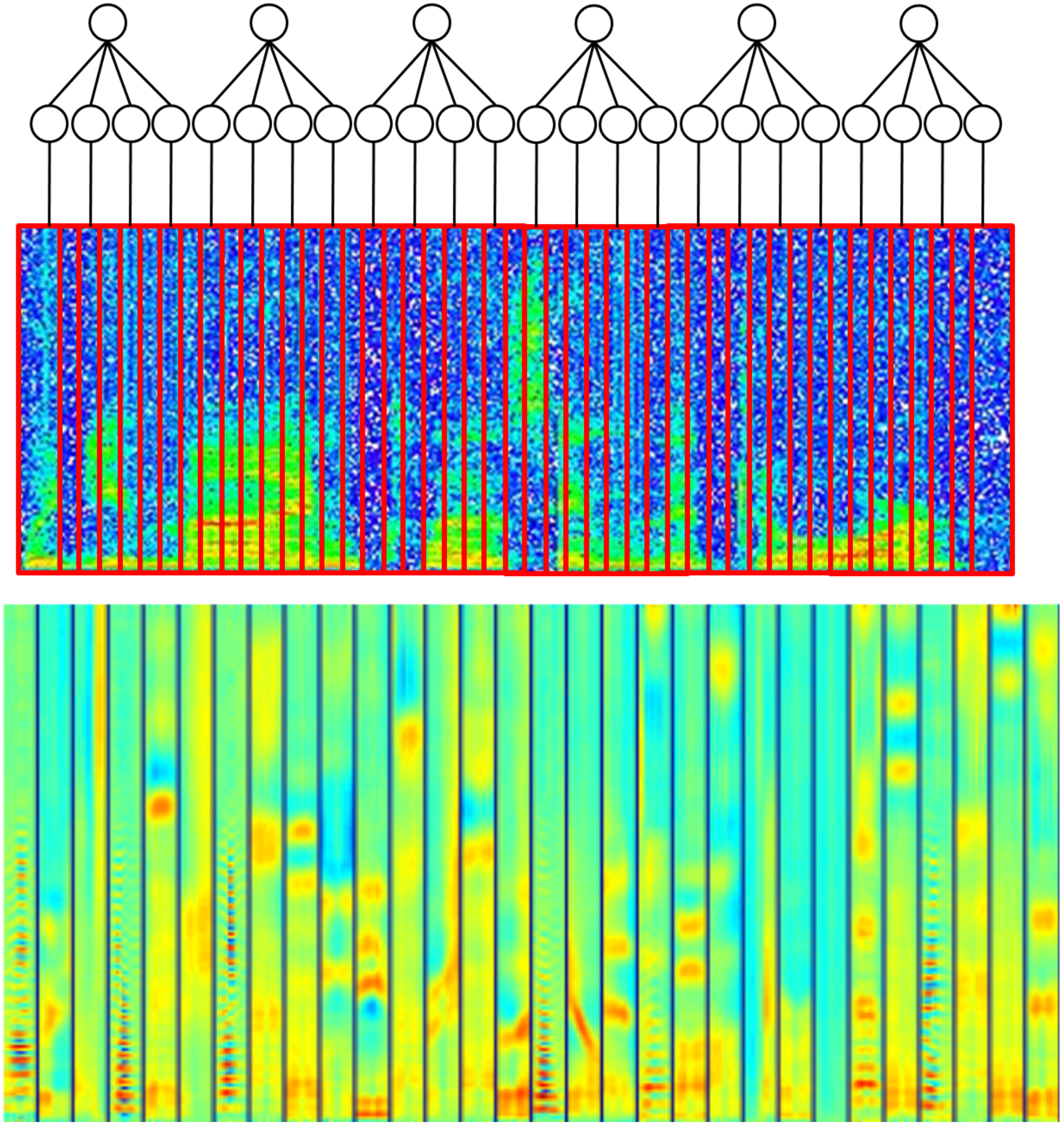
- Binary/Gaussian/Softmax RBMs: All have binary hidden variables but use them to model different kinds of data.



- It is easy to infer the states of the hidden variables:

$$P_\theta(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{F} P_\theta(h_j|\mathbf{v}) = \prod_{j=1}^{F} \frac{1}{1 + \exp(-a_j - \sum_{i=1}^{D} W_{ij} v_i)}$$

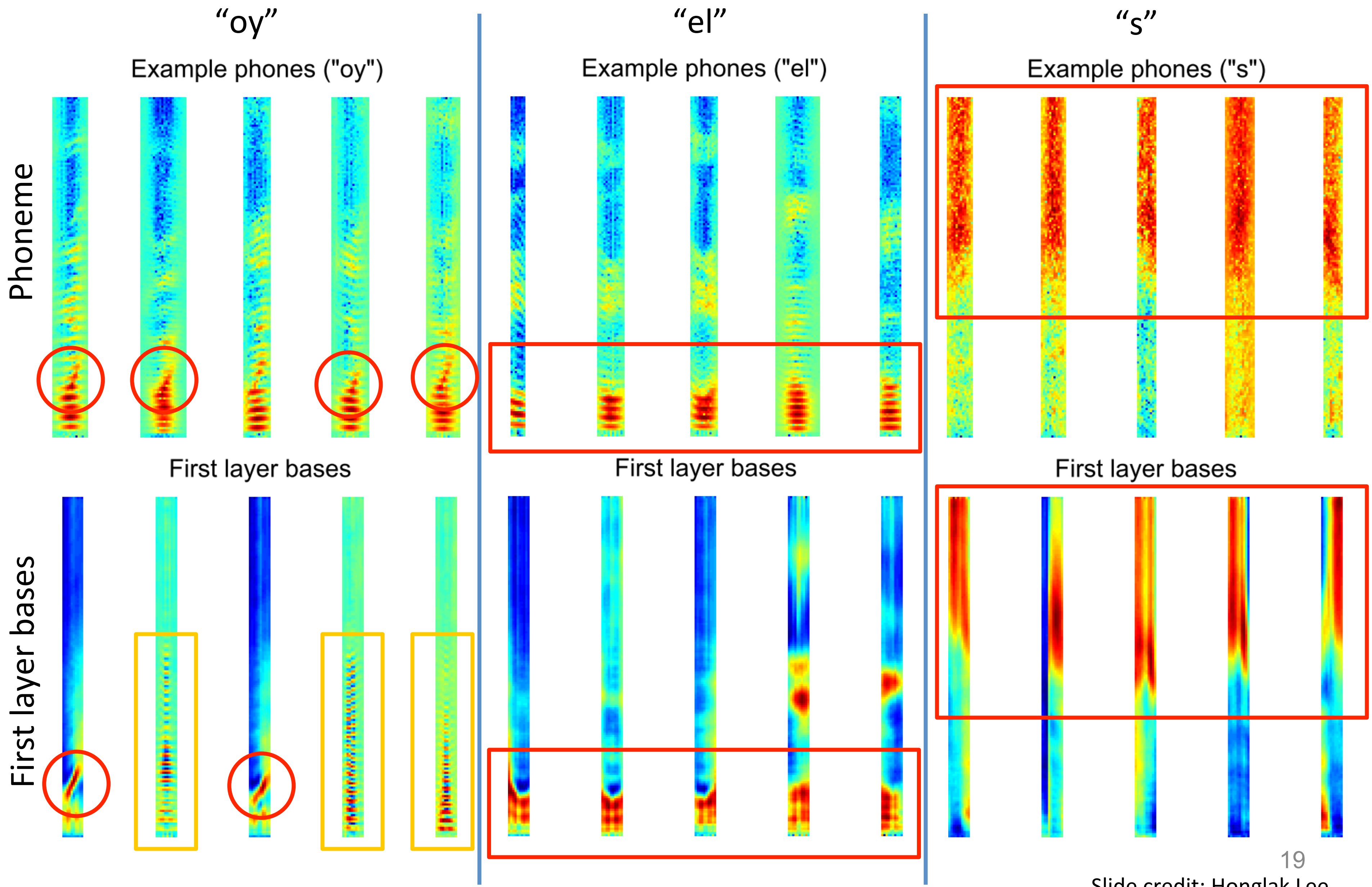
Speech



Learned first-layer bases

Lee et.al., NIPS 2009

Comparison of bases to phonemes



Slide credit: Honglak Lee

Product of Experts

The joint distribution is given by:

$$P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\Big(\sum_{ij} W_{ij} v_i h_j + \sum_i b_i v_i + \sum_j a_j h_j\Big)$$

Marginalizing over hidden variables:

Product of Experts

$$P_\theta(\mathbf{v}) = \sum_{\mathbf{h}} P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \prod_i \exp(b_i v_i) \prod_j \Big(1 + \exp(a_j + \sum_i W_{ij} v_i)\Big)$$

government	clinton	bribery	mafia	stock	…
authority	house	corruption	business	wall	
power	president	dishonesty	gang	street	
empire	bill	corrupt	mob	point	
federation	congress	fraud	insider	dow	

Silvio Berlusconi

Topics "government", "corruption" and "mafia" can combine to give very high probability to a word "Silvio Berlusconi".

Product of Experts

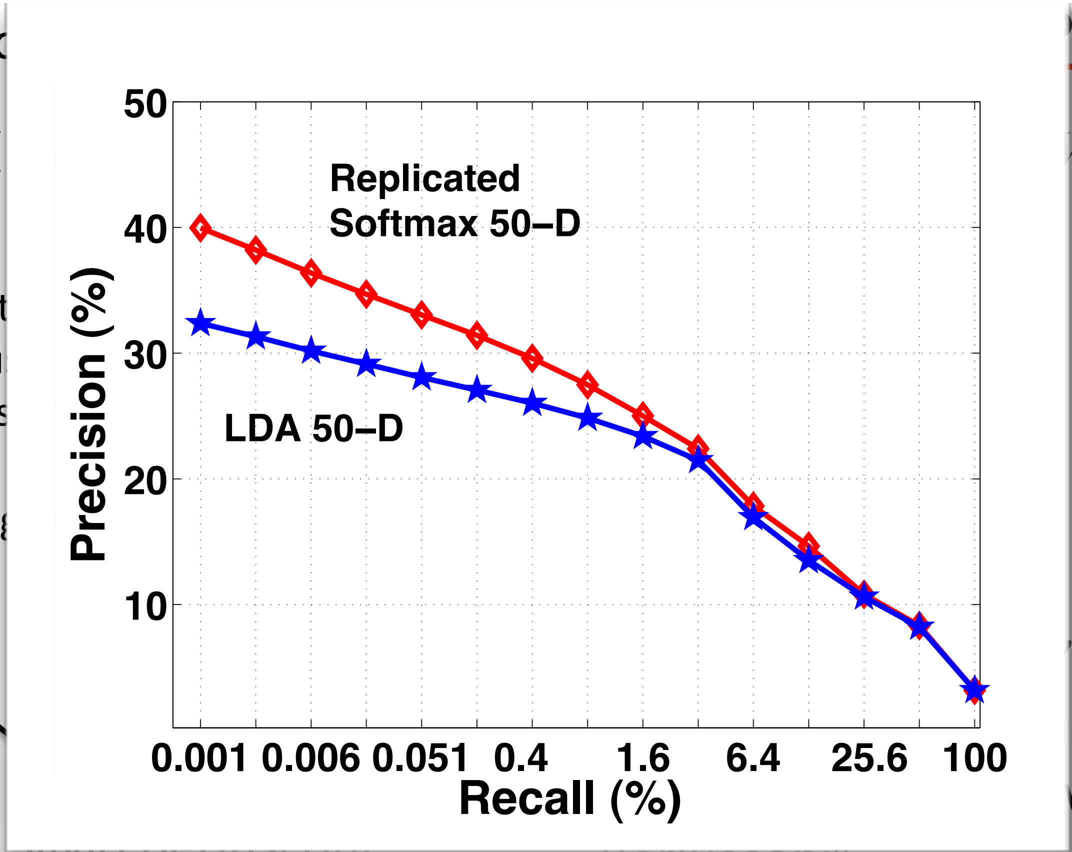
The joint distribution is given by:

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)}\exp\Big(\sum_{ij}W_{ij}v_ih_j + \sum_i b_iv_i + \sum_j a_jh_j\Big)$$

Marginalizing c

$P_\theta(\mathbf{v}) = \sum_{\mathbf{h}}$ … $W_{ij}v_i)\Big)$

duct of Experts



government
authority
power
empire
federation

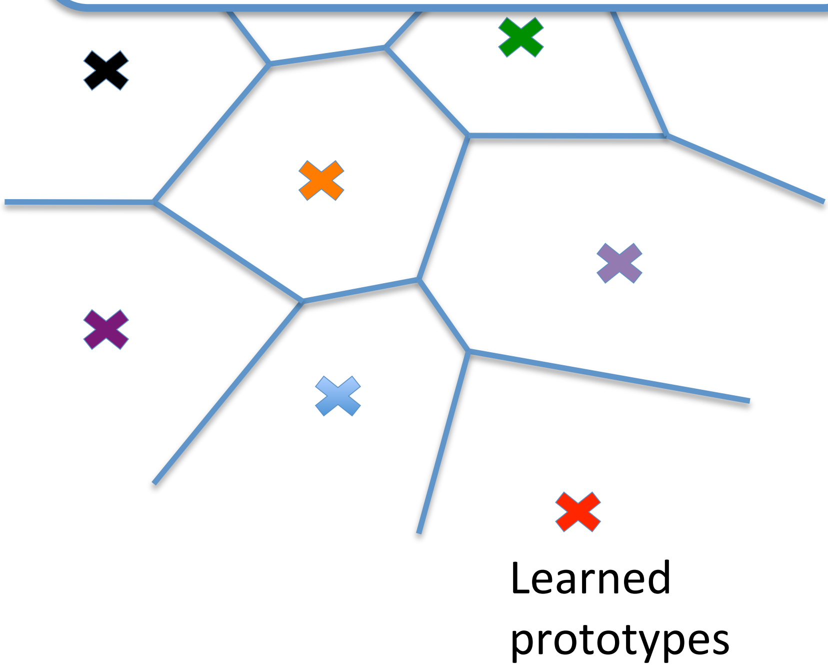
clint
hou
pres
bill
con

", "corruption"
bine to give very
word "Silvio
Berlusconi".

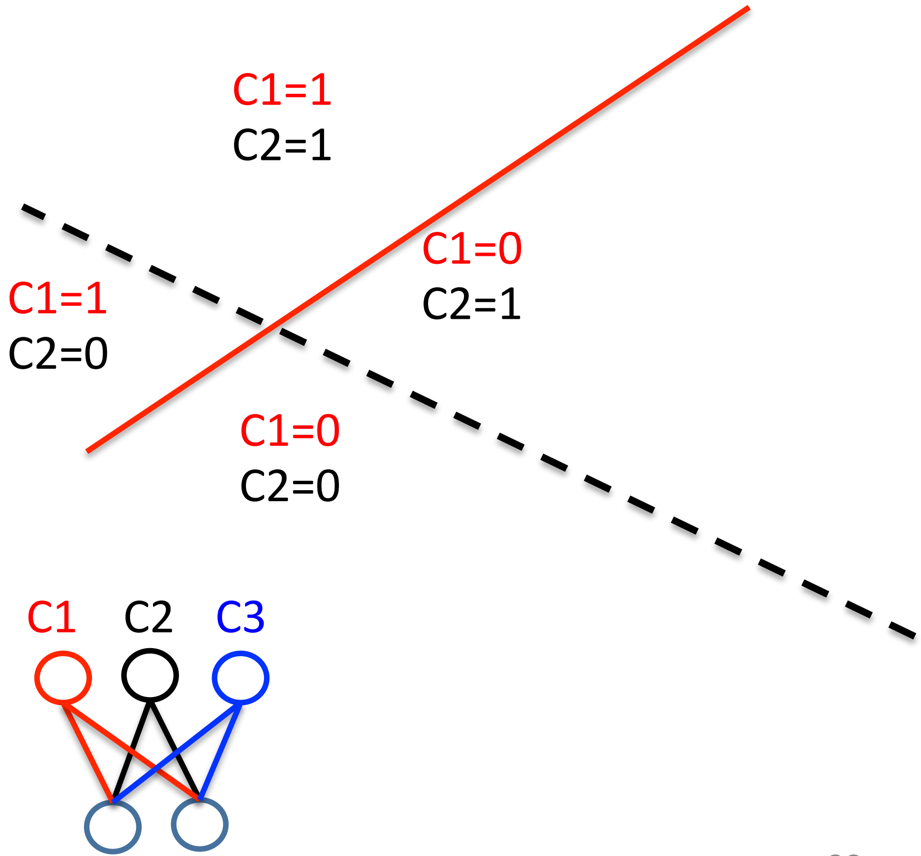
Local vs. Distributed Representations

- Clustering, Nearest Neighbors, RBF SVM, local density estimators

- Parameters for each region.
- # of regions is linear with # of parameters.



- RBMs, Factor models, PCA, Sparse Coding, Deep models

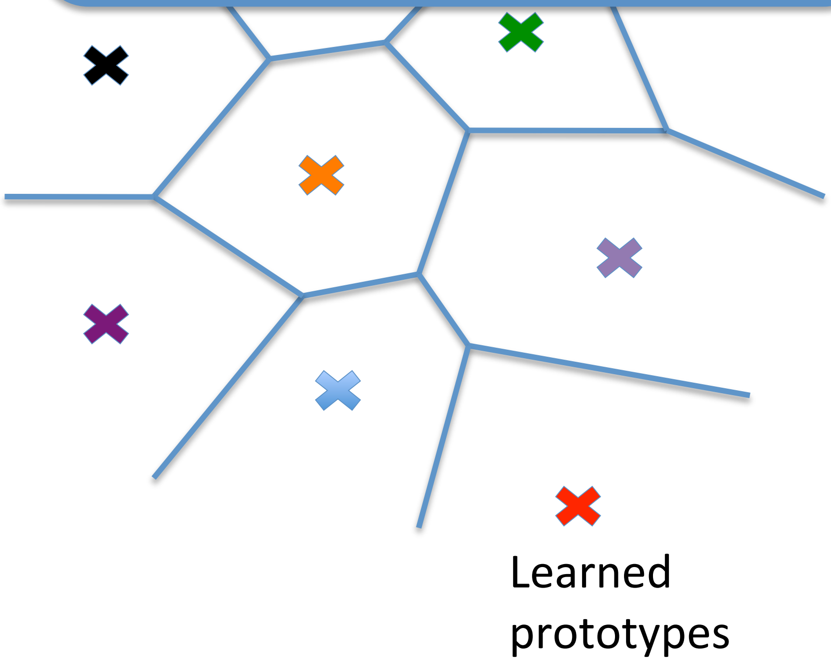


Bengio, 2009, Foundations and Trends in Machine Learning

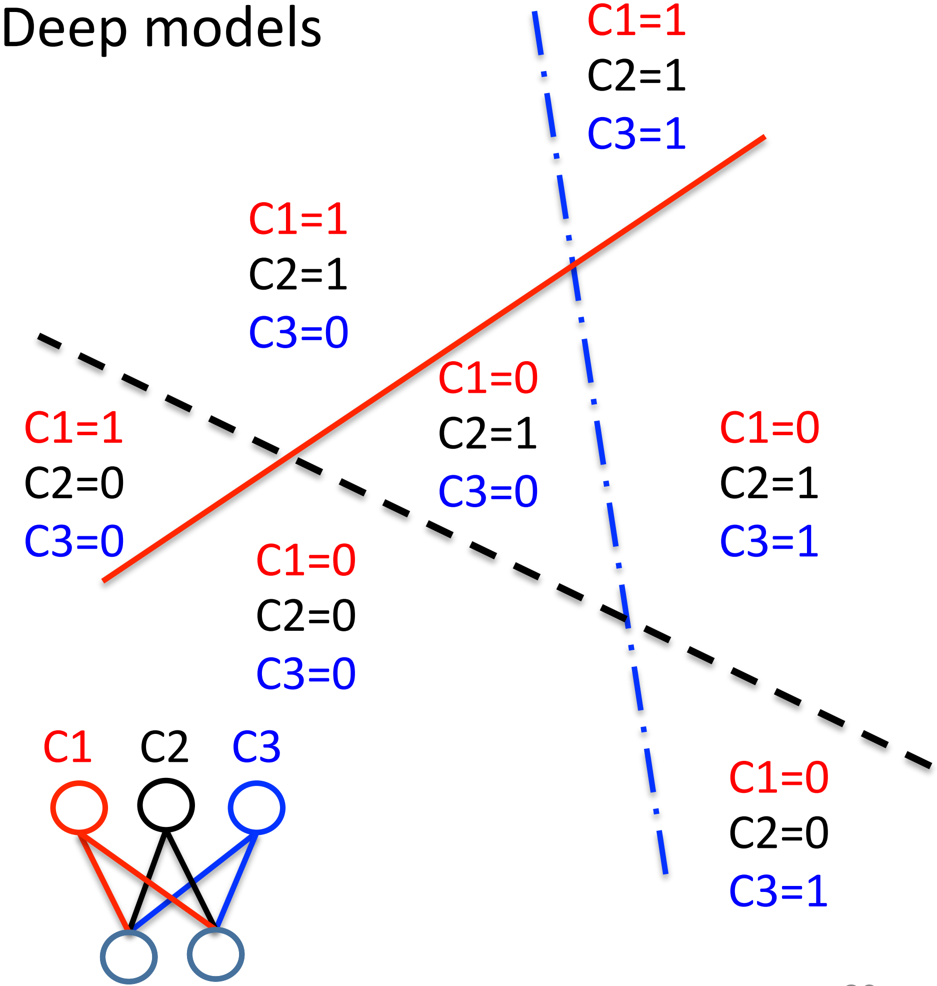
Local vs. Distributed Representations

- Clustering, Nearest Neighbors, RBF SVM, local density estimators

- Parameters for each region.
- # of regions is linear with # of parameters.



- RBMs, Factor models, PCA, Sparse Coding, Deep models

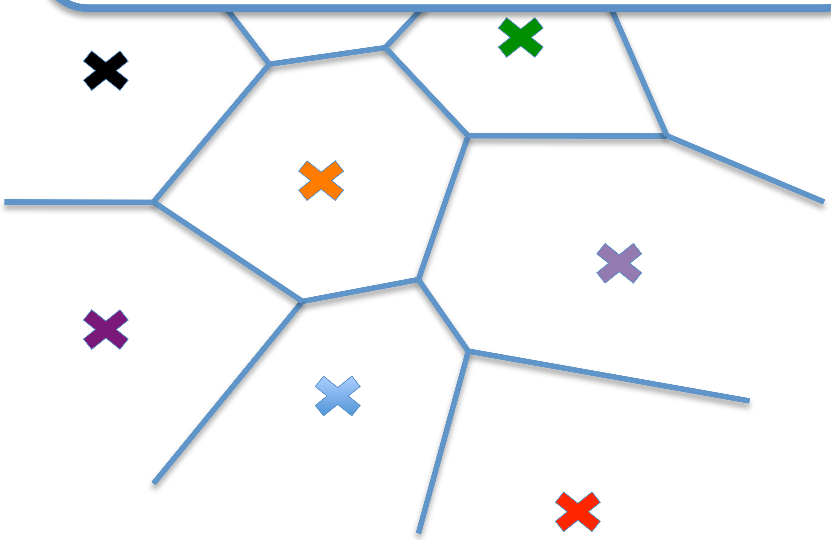


Bengio, 2009, Foundations and Trends in Machine Learning

Local vs. Distributed Representations

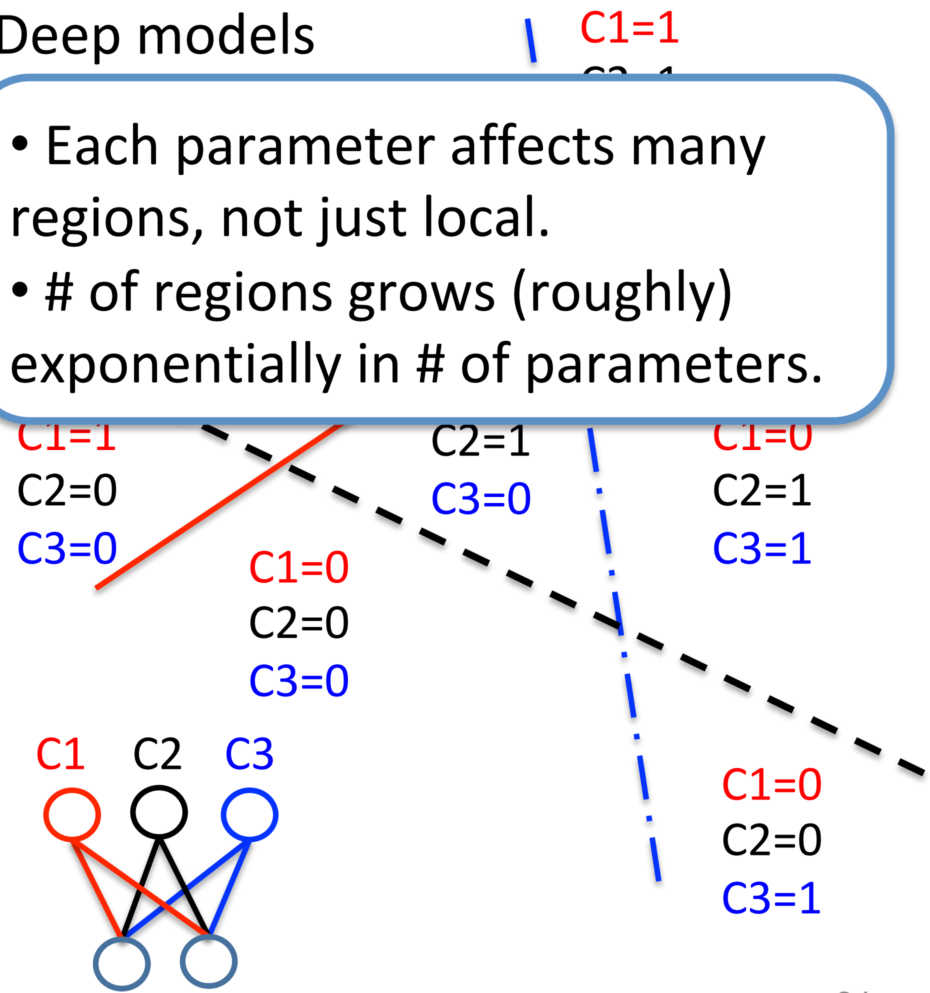
- Clustering, Nearest Neighbors, RBF SVM, local density estimators

- Parameters for each region.
- # of regions is linear with # of parameters.



- RBMs, Factor models, PCA, Sparse Coding, Deep models

- Each parameter affects many regions, not just local.
- # of regions grows (roughly) exponentially in # of parameters.



Bengio, 2009, Foundations and Trends in Machine Learning

Multiple Application Domains

- Natural Images
- Text/Documents
- Collaborative Filtering / Matrix Factorization
- Video (Langford, et al. ICML 2009)
- Motion Capture (Taylor et.al. NIPS 2007)
- Speech Perception (Dahl et. al. NIPS 2010, Lee et.al. NIPS 2010)

Same learning algorithm --
multiple input domains.

Limitations on the types of structure that can be represented by a single layer of low-level features!