Deep Learning – Autoencoder Models

Davide Bacciu

Dipartimento di Informatica
Università di Pisa

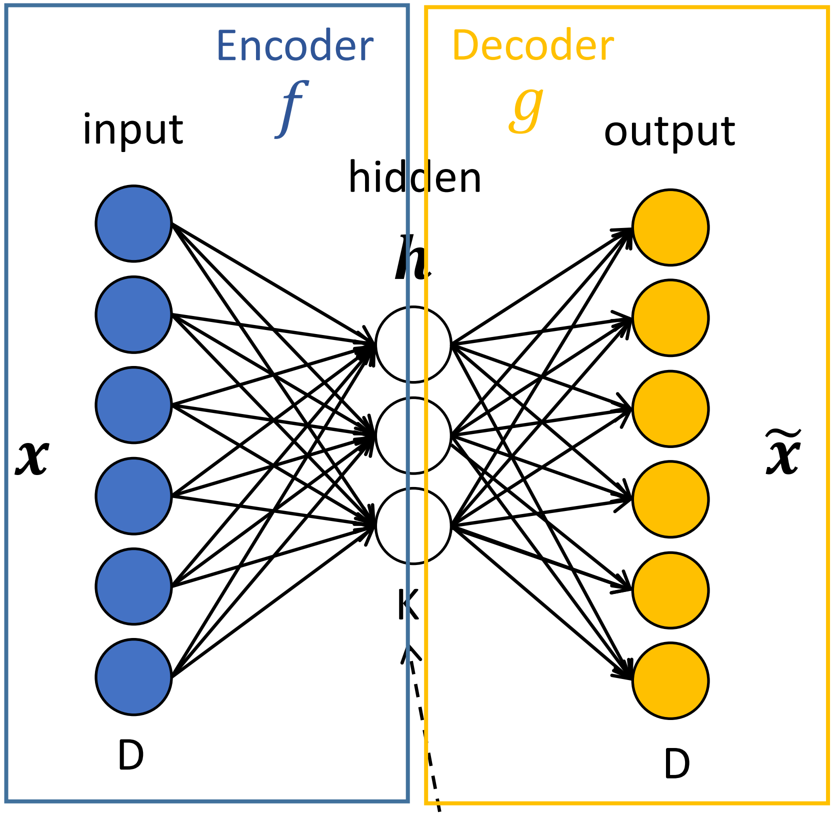
Intelligent Systems for Pattern Recognition (ISPR)

Lecture Outline

Autoencoders a.k.a. The first and the latest deep learning model

- Autoencoders and dimensionality reduction
- Deep neural autoencoders
  - Sparse
  - Denoising
  - Contractive
- Deep generative-based autoencoders
  - Deep Belief Networks
  - Deep Boltzmann Machines
- Application Examples

Basic Autoencoder (AE)

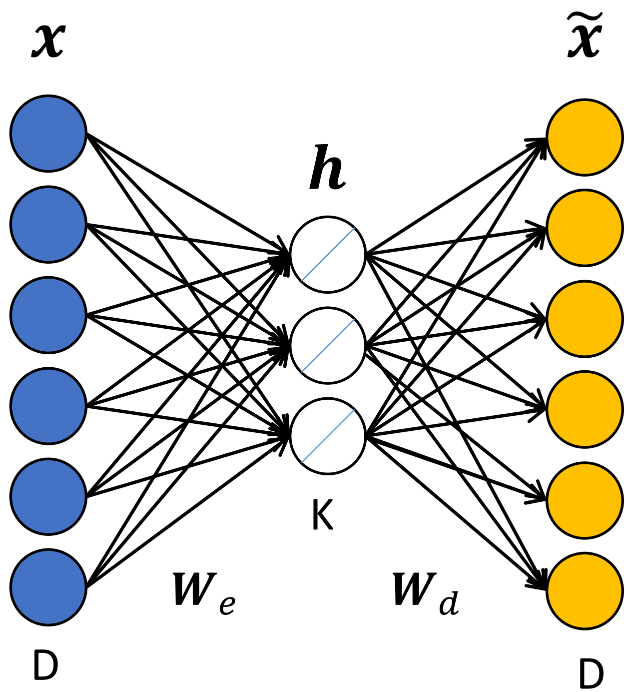


Latent space projection (again)

- Train a model to reconstruct the input
- Passing through some form of information bottleneck
  - K << D, or?
  - **h** sparsely active
- Train by loss minimization

$$L(\boldsymbol{x}, \widetilde{\boldsymbol{x}}) = L(\boldsymbol{x}, g(f(\boldsymbol{x})))$$

A Very Well Known Autoencoder



Encoding-Decoding

$$\boldsymbol{h} = f(\boldsymbol{x}) = \boldsymbol{W}_e\boldsymbol{x}$$

$$\tilde{\boldsymbol{x}} = g(\boldsymbol{h}) = \boldsymbol{W}_d\boldsymbol{W}_e\boldsymbol{x}$$

Tied weights (often, not always)

$$\boldsymbol{W}_d = \boldsymbol{W}_e^T = \boldsymbol{W}^T$$

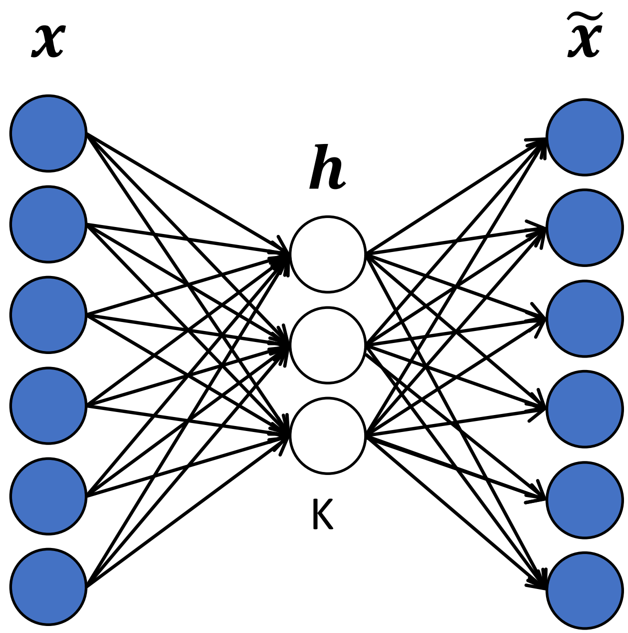
Euclidean Loss

$$L(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \|\boldsymbol{x} - \boldsymbol{W}^T\boldsymbol{W}\boldsymbol{x}\|_2^2$$

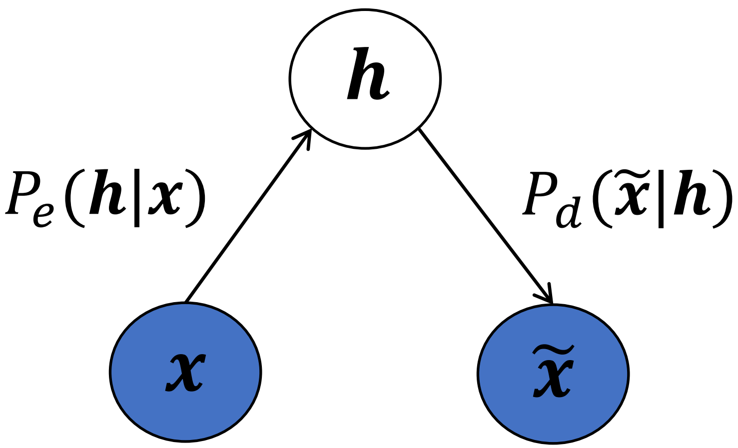
What if we take f and g linear and K<<D?

Learns the same subspace of PCA

A Probabilistic View



Stochastic Autoencoder



- A unifying view of neural and generative AE
- Paves the way to Variational Autoencoders

Neural Autoencoders

Generally, we would like to train nonlinear AEs, with possibly K>D, that do not learn trivial identity

- Regularized autoencoders
  - Sparse AE
  - Denoising AE
  - Contractive AE
- Autoencoders with dropout layers

Sparse Autoencoder

Add a term to the cost function to penalize **h** (want the number of active units to be small)

$$J_{SAE}(\theta) = \sum_{x \in S}(L(\boldsymbol{x}, \widetilde{\boldsymbol{x}}) + \lambda\Omega(\boldsymbol{h}))$$

Typically

$$\Omega(\boldsymbol{h}) = \Omega(f(\boldsymbol{x})) = \sum_j |h_j(\boldsymbol{x}))|$$

Probabilistic Interpretation (Oh No, Again!)

(From ML Course) Training with regularization is MAP inference

$$\max \log P(\boldsymbol{h}, \boldsymbol{x}) = \max \left(\log P(\boldsymbol{x}|\boldsymbol{h}) + \log P(\boldsymbol{h})\right)$$

Likelihood

Prior

$$P(\boldsymbol{h}) = \frac{\lambda}{2}\exp\left(-\frac{\lambda}{2}|\boldsymbol{h}|_1\right) \Rightarrow \Omega(\boldsymbol{h}) = \lambda|\boldsymbol{h}|_1$$

Laplace

Denoising Autoencoder (DAE)

Train the AE to minimize the function

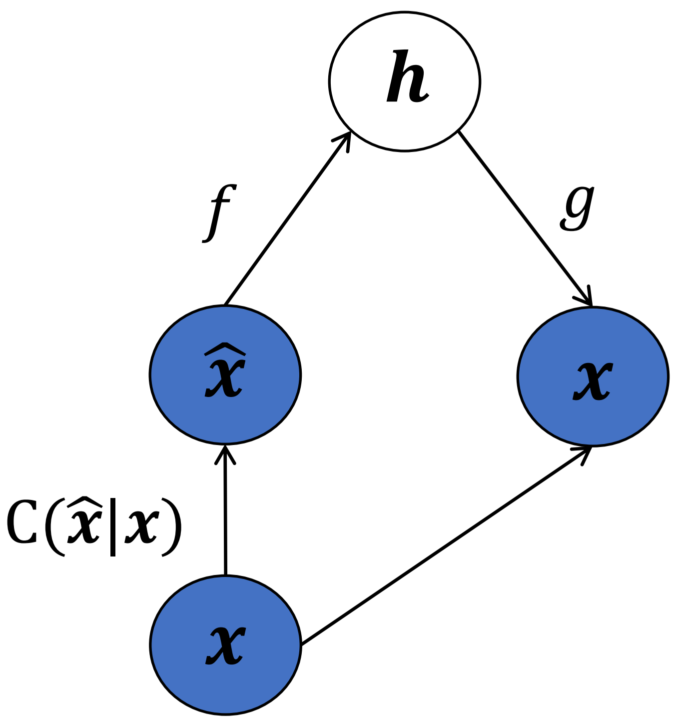
$$L(\boldsymbol{x}, g(f(\widehat{\boldsymbol{x}})))$$

where $\widehat{\boldsymbol{x}}$ is a version of original input $\boldsymbol{x}$ corrupted by some noise process $\mathrm{C}(\widehat{\boldsymbol{x}}|\boldsymbol{x})$

Key Intuition - Learned representations should be robust to partial destruction of the input

Another Interpretation...

...yes, exactly the one you are thinking of



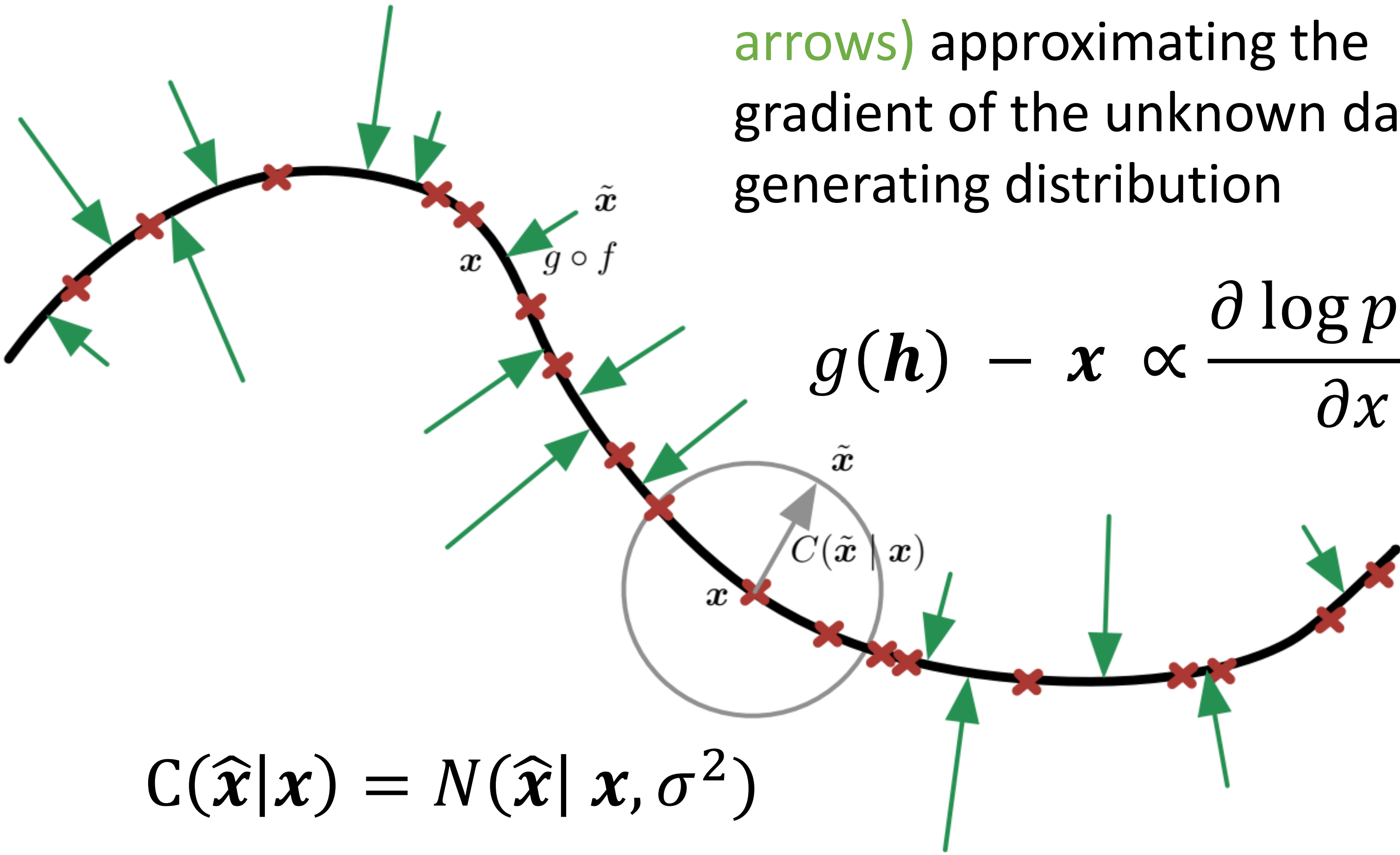
Learns the denoising distribution

$$P(\boldsymbol{x}|\tilde{\boldsymbol{x}})$$

By minimizing

$$-\log P_d(\boldsymbol{x}|\boldsymbol{h} = f(\tilde{\boldsymbol{x}}))$$

DAE as Manifold Learning

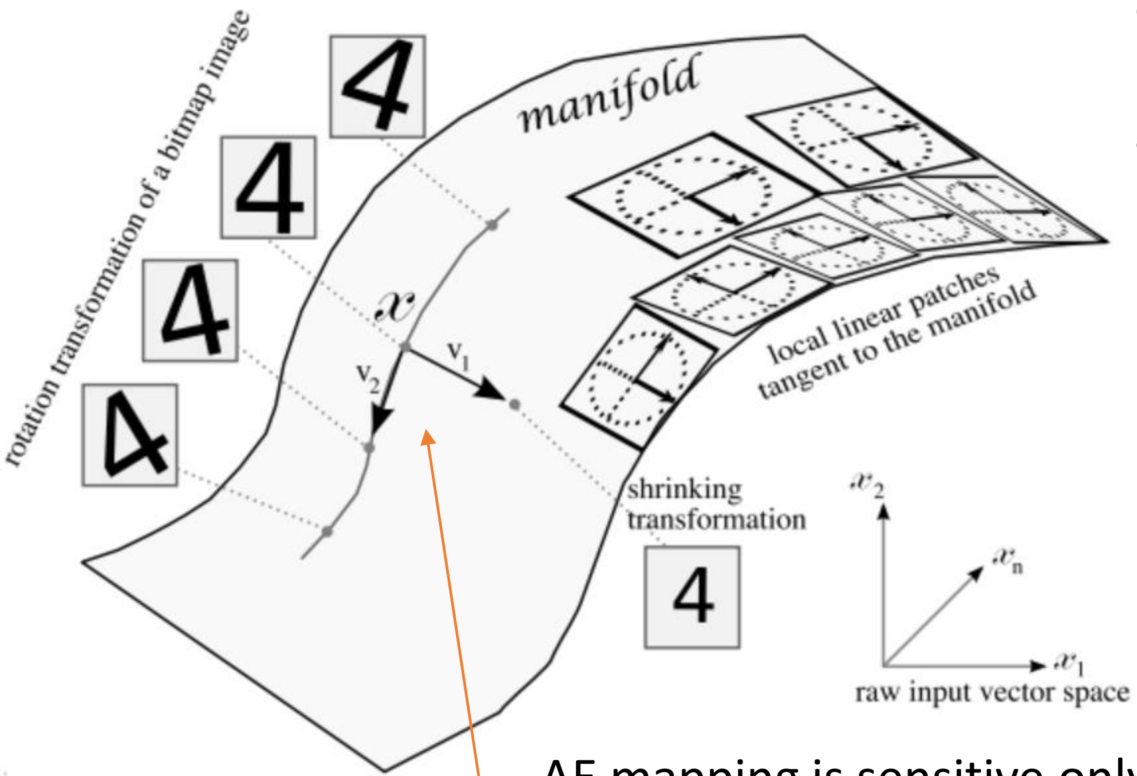


Learning a vector field (green arrows) approximating the gradient of the unknown data generating distribution

$$g(\boldsymbol{h}) - \boldsymbol{x} \propto \frac{\partial \log p(\boldsymbol{x})}{\partial x}$$

$$\mathrm{C}(\widehat{\boldsymbol{x}}|\boldsymbol{x}) = N(\widehat{\boldsymbol{x}}|\, \boldsymbol{x}, \sigma^2)$$

The Manifold Assumption



Assume data lies on a lower dimensional non-linear manifold since variables in data are typically dependent

Regularized AE can afford to represent only variations that are needed to reconstruct training examples

AE mapping is sensitive only to changes in manifold direction

Yoshua Bengio, Learning Deep Architectures for AI, Foundations and Trends in Machine Learning, 2009.

Contractive Autoencoder

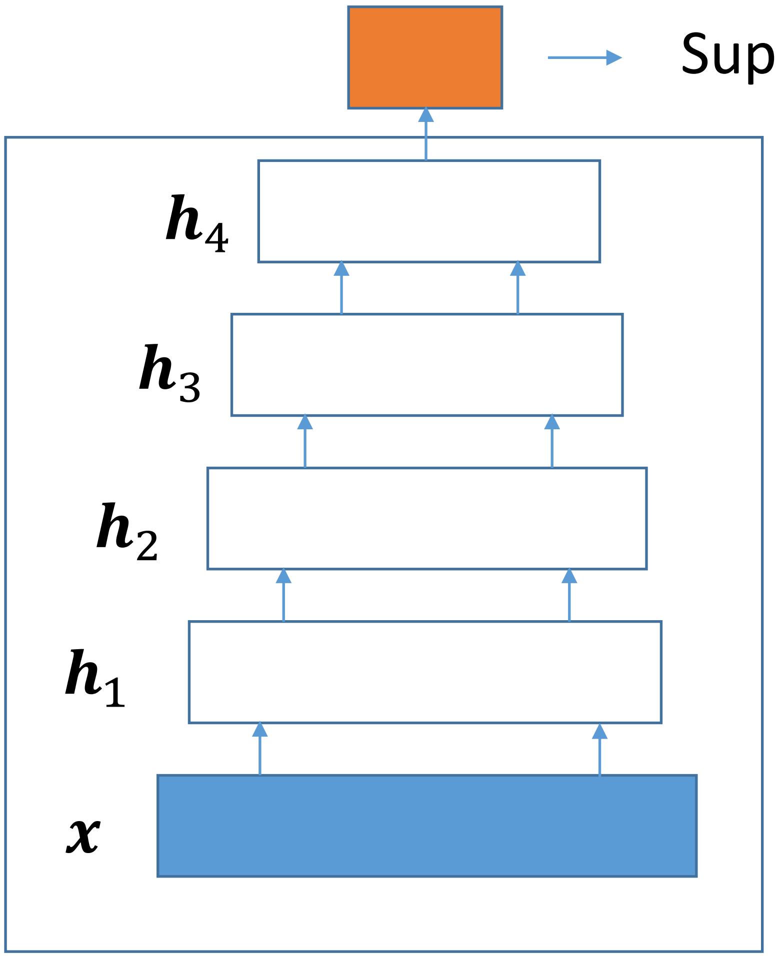
Penalize **encoding function** for input sensitivity

$$J_{CAE}(\theta) = \sum_{x \in S} (L(\boldsymbol{x}, \widetilde{\boldsymbol{x}}) + \lambda \Omega(\boldsymbol{h}))$$

$$\Omega(\boldsymbol{h}) = \Omega(f(\boldsymbol{x})) = \left\| \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} \right\|_F^2$$

You can as well penalize on higher order derivatives

Deep Autoencoder



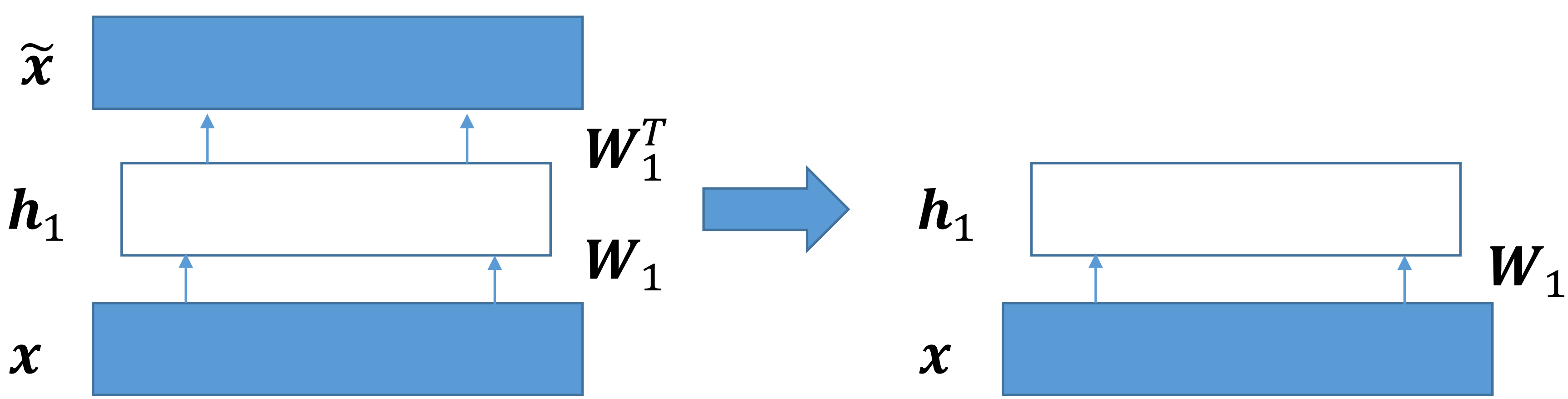
→ Supervised learning

- Unsupervised training
- Hierarchical autoencoder
- Extracts a representation of inputs that facilitates
  - Data visualization, exploration, indexing,…
  - Realization of a supervised task

Unsupervised Layerwise Pretraining

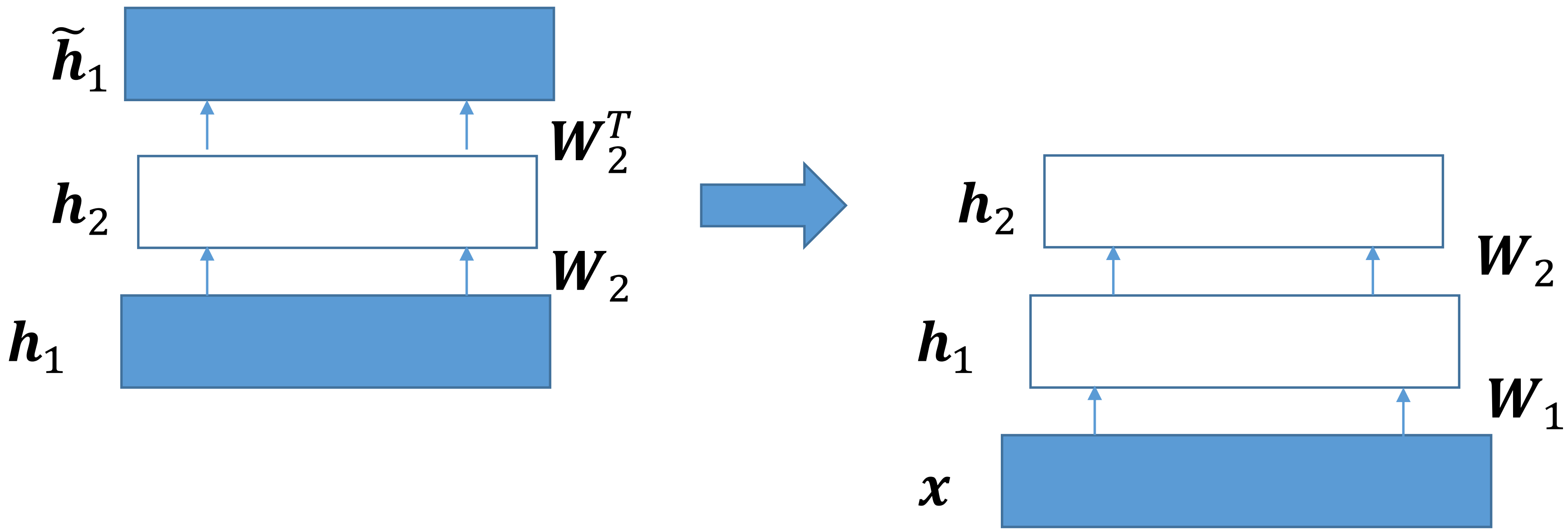
Incremental unsupervised construction of the Deep AE

Any form of AE, e.g. those shown in previous slides



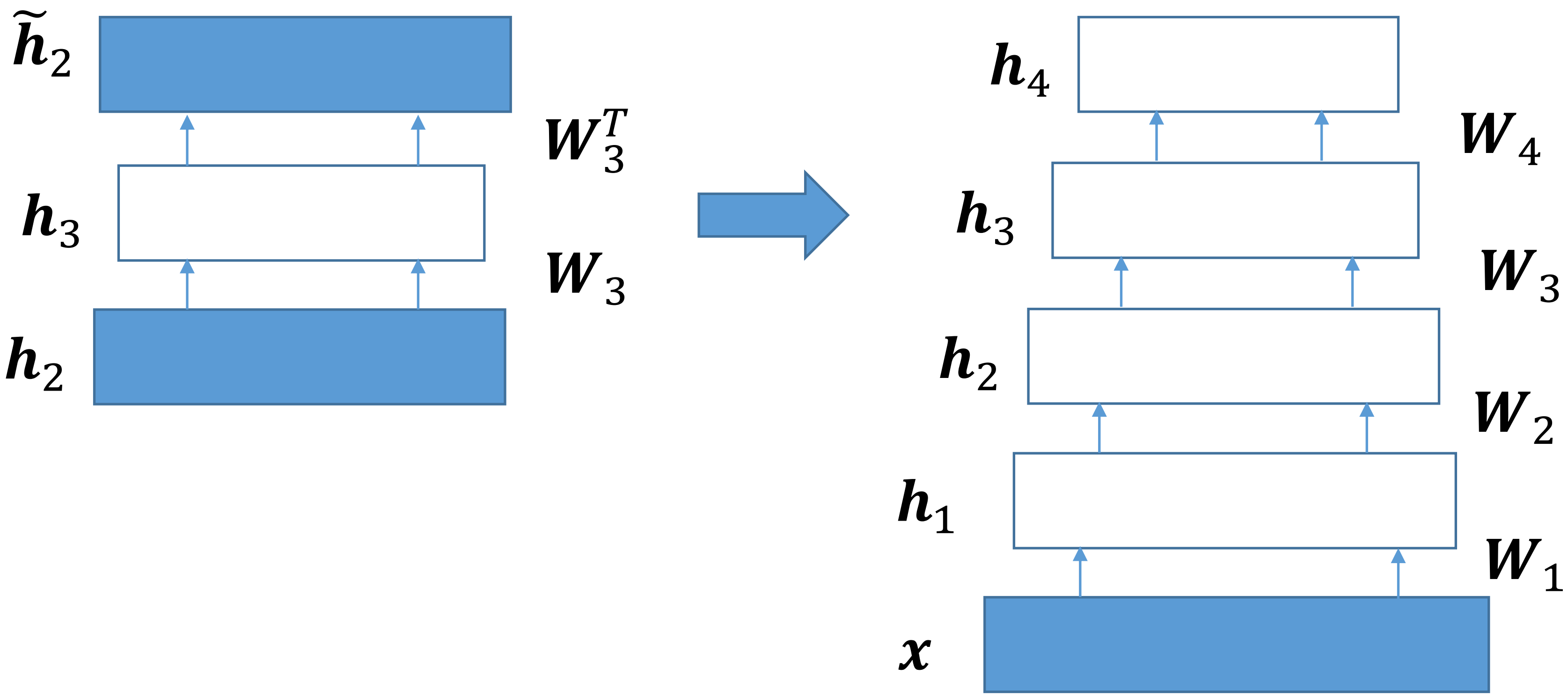
Unsupervised Layerwise Pretraining

Incremental unsupervised construction of the Deep AE



Unsupervised Layerwise Pretraining

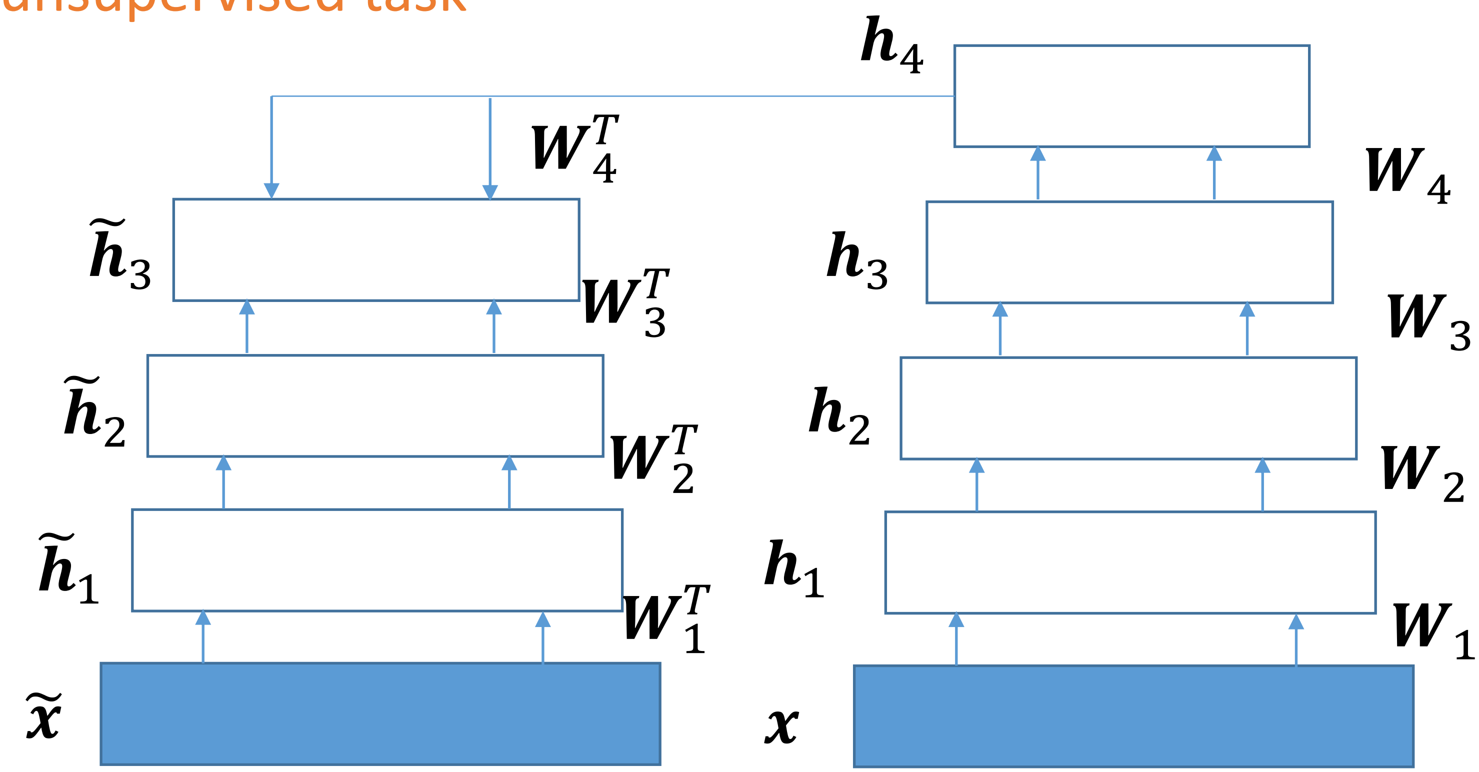
Incremental unsupervised construction of the Deep AE



Optional Fine Tuning

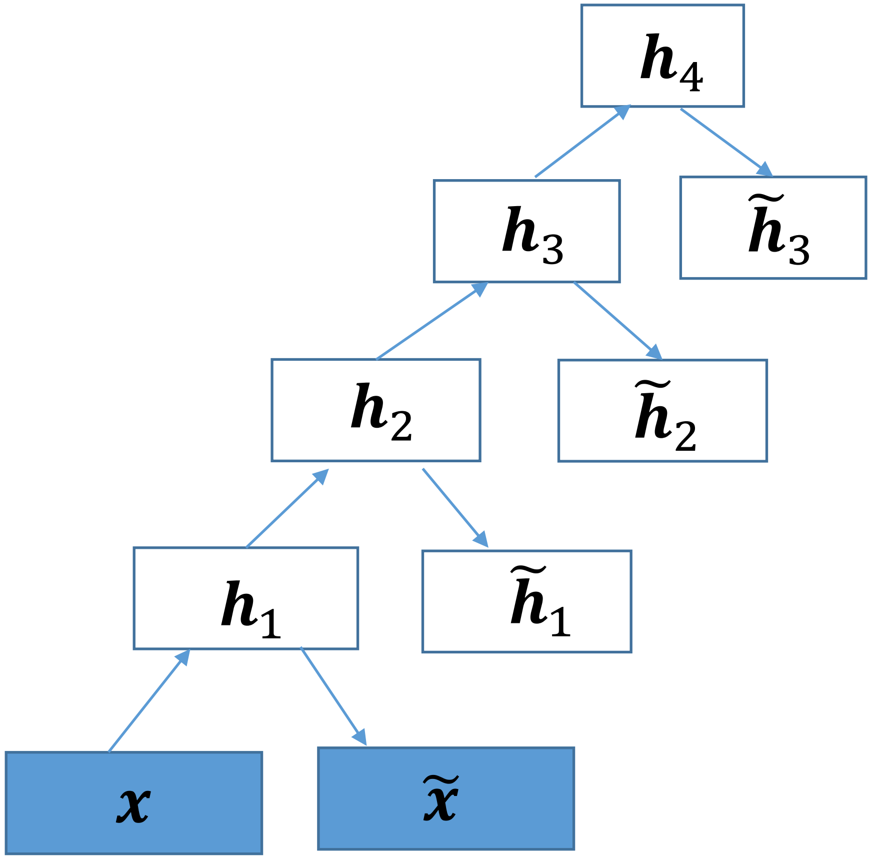
Fine tune the whole autoencoder to optimize input reconstruction

You can use backpropagation, but it remains an unsupervised task



Rearranging the Graphics

Does it look like something familiar?

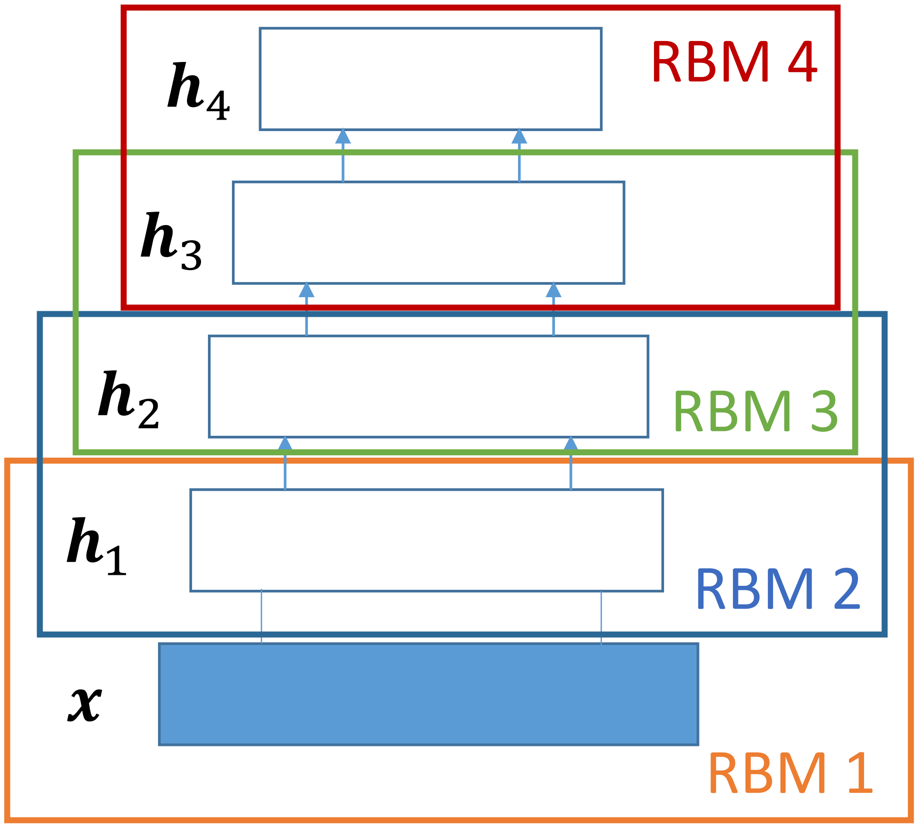


A layered Restricted Boltzmann Machine

Can use RBM to perform layerwise pretraining and learn the matrices $\boldsymbol{W}_i$

Deep Belief Network (DBN)

A stack of pairwise RBM



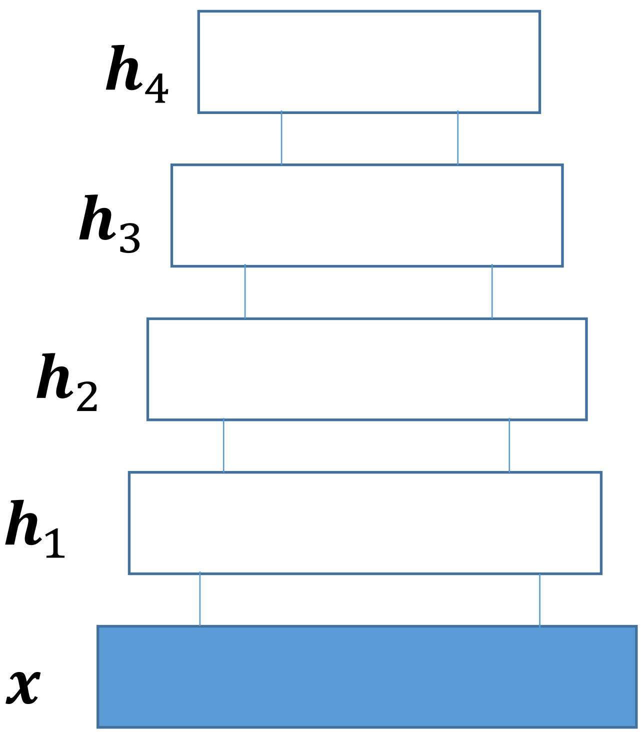
IMPORTANT NOTE

A DBM is a deep autoencoder but it is NOT a deep RBM

It is (mostly) directed!

Deep Boltzmann Machine (DBM)

How do we get this?



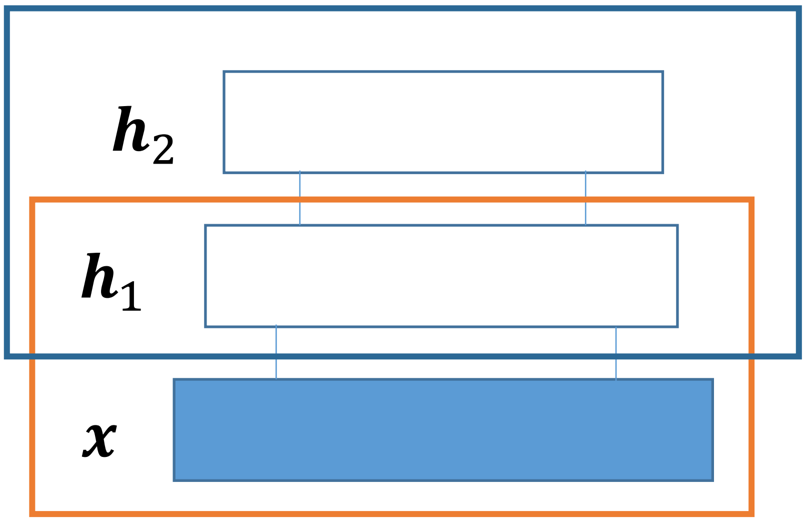
Training requires some attention because of the recurrent interactions from higher layers to the bottom

$$P(h_j^1 | \boldsymbol{x}, \boldsymbol{h}^2) = \sigma\left(\sum_i W_{ij}^1 \, x_i + \sum_m W_{jm}^2 \, h_m^2\right)$$

$$P(x_i | \boldsymbol{h}^1) = \sigma\left(\sum_j W_{ij}^1 \, h_j^1\right)$$

Pretraining DBM

How do we get this?



2) (Pre)training the second layer changes $\boldsymbol{h}^1$prior by

$$P(\boldsymbol{h}^1|\boldsymbol{W}^2) = \sum_{\boldsymbol{h}^2} P(\boldsymbol{h}^1, \boldsymbol{h}^2|\boldsymbol{W}^2)$$

When putting things together, we need to average between the two

$$P(\boldsymbol{h}^1|\boldsymbol{W}^1) = \sum_{\boldsymbol{x}} P(\boldsymbol{h}^1, \boldsymbol{x}|\boldsymbol{W}^1)$$

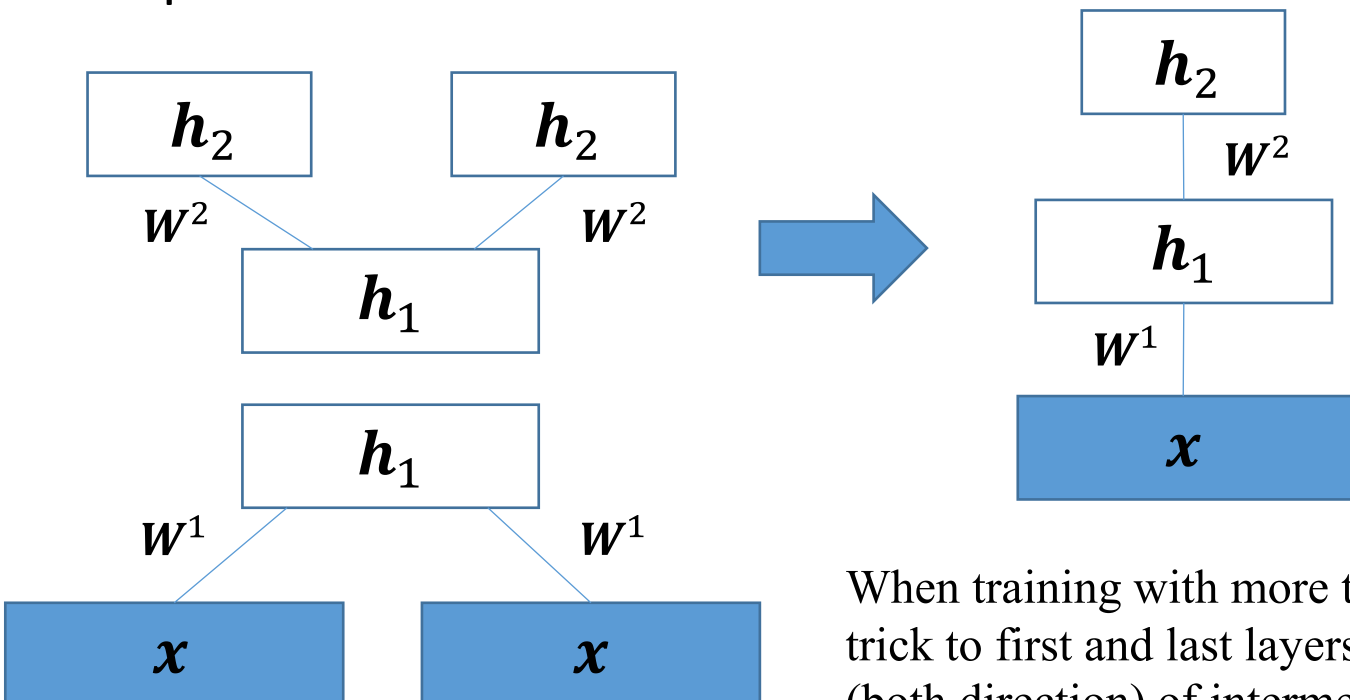
1) (Pre)training the first layer entails fitting this model

$$P(\boldsymbol{x}|\theta) = \sum_{\boldsymbol{h}^1} P(\boldsymbol{h}^1|\boldsymbol{W}^1) P(\boldsymbol{x}|\boldsymbol{h}^1, \boldsymbol{W}^1)$$

Pretraining DBM - Trick

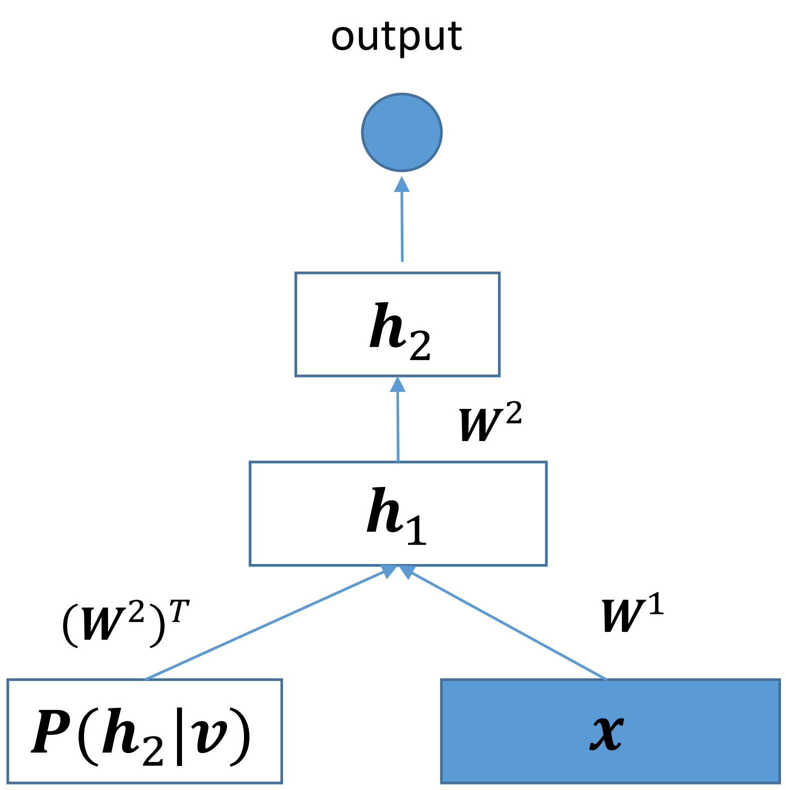
Averaging the two models of $\boldsymbol{h}^1$ can be approximated by taking half contribution from $\boldsymbol{W}^1$ and half from $\boldsymbol{W}^2$

- Using full $\boldsymbol{W}^1$ and $\boldsymbol{W}^2$ would double count $\boldsymbol{x}$ contribution as $\boldsymbol{h}^2$ depends on $\boldsymbol{x}$



When training with more than two RBMs apply trick to first and last layers and halve weights (both direction) of intermediate RBM

DBM – Discriminative Fine Tuning



The pretrained DBM matrices can be used to initialize a deep autoencoder

- Add input from $\boldsymbol{h}^2$ to the first hidden layer
- Add output layer
- Fine tuning of the RBM matrices by backpropagation

Software - Deep Neural Autoencoders

- All deep learning frameworks offer facilities to build (deep) AEs
- Check out classic Theano-based tutorials for denoising autoencoders and their stacked version
- A variety of deep AE in Keras and their counterpart in Torch (plus a selection in Pytorch)
- Stacked autoencoders built with official Matlab toolbox functions

Matlab - Deep Generative Models

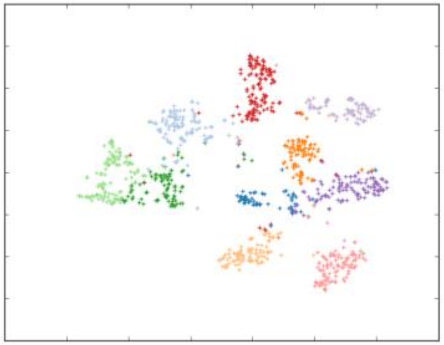
- Matlab code for the DBN paper with a demo on MNIST data
- Matlab code for Deep Boltzmann Machines with a demo on MNIST data
- Deepmat – Matlab library for deep generative models
- DeeBNet – Matlab/Octave toolbox for deep generative models with GPU support

Python - Deep Generative Models

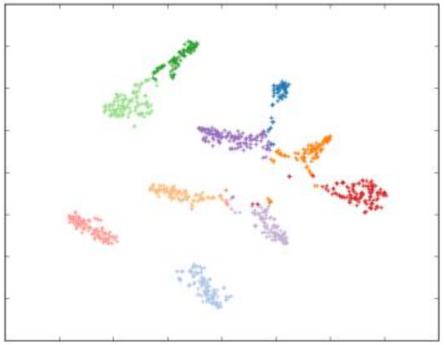
- DBN and DBM implementations exist for all major deep learning libraries
- Deep Boltzmann machine implementation (Tensorflow-based) with image processing application, pre-trained networks and notebooks
- Deepnet – A Toronto based implementation of deep autoencoders (neural and generative)
- Check out classic Theano-based tutorials for deep belief networks and RBM

AE Applications - Visualization

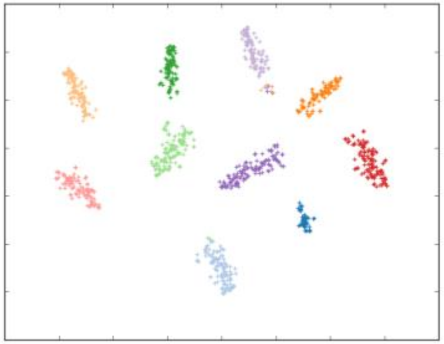
Visualizing complex data in learned latent space



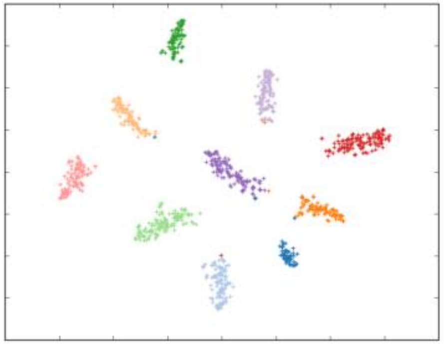
(a) Epoch 0



(b) Epoch 3

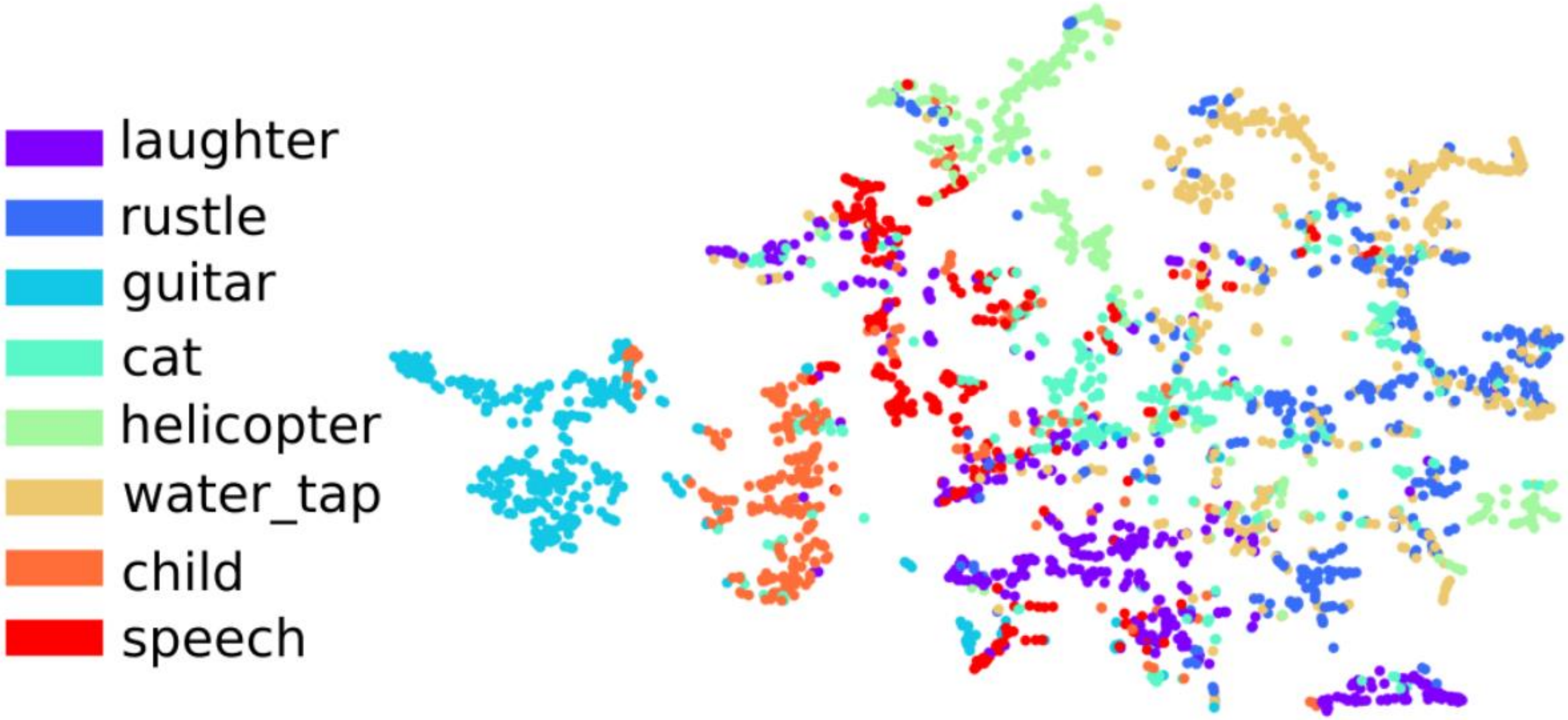


(d) Epoch 9



(e) Epoch 12

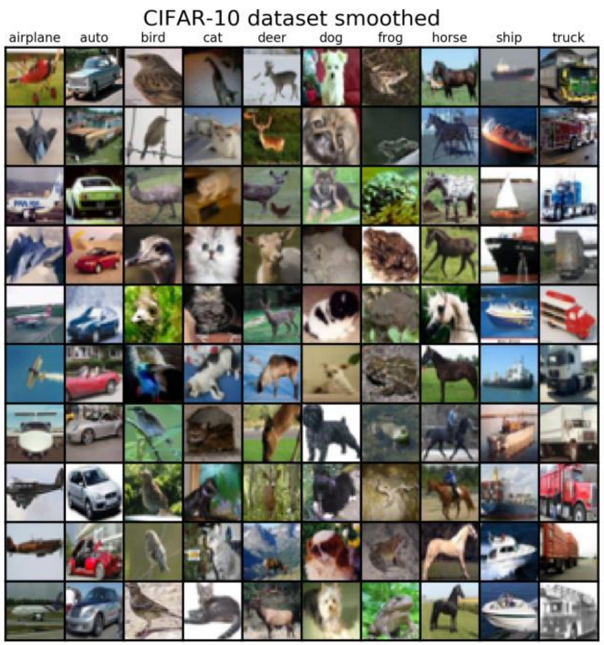
Visualizing Sound



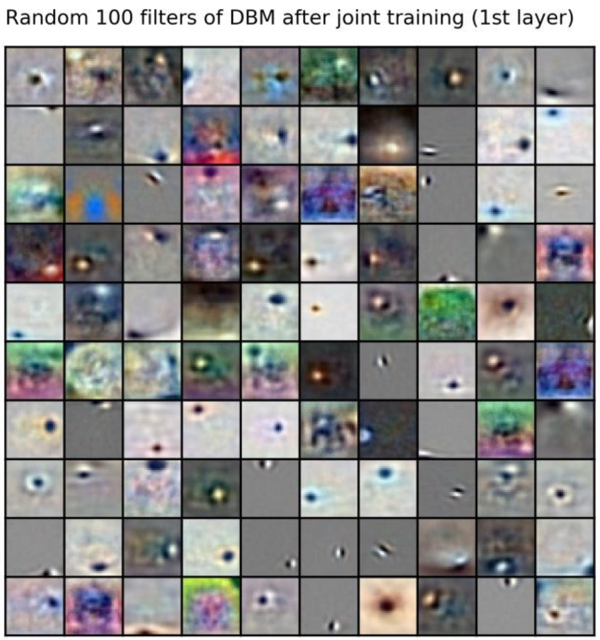
AE Applications – Image Restoration/Colorization

Apply autoencoder construction with advanced building blocks (e.g. CNN layers)

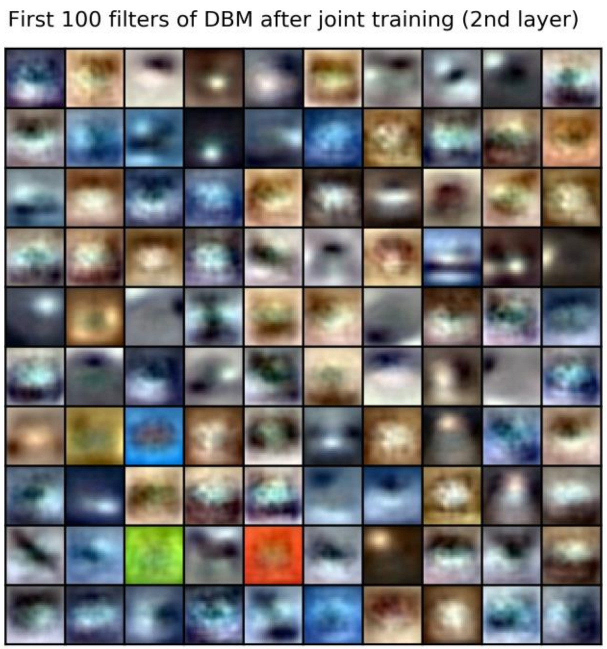
DBM – Learning Image Features



CIFAR-10 Images



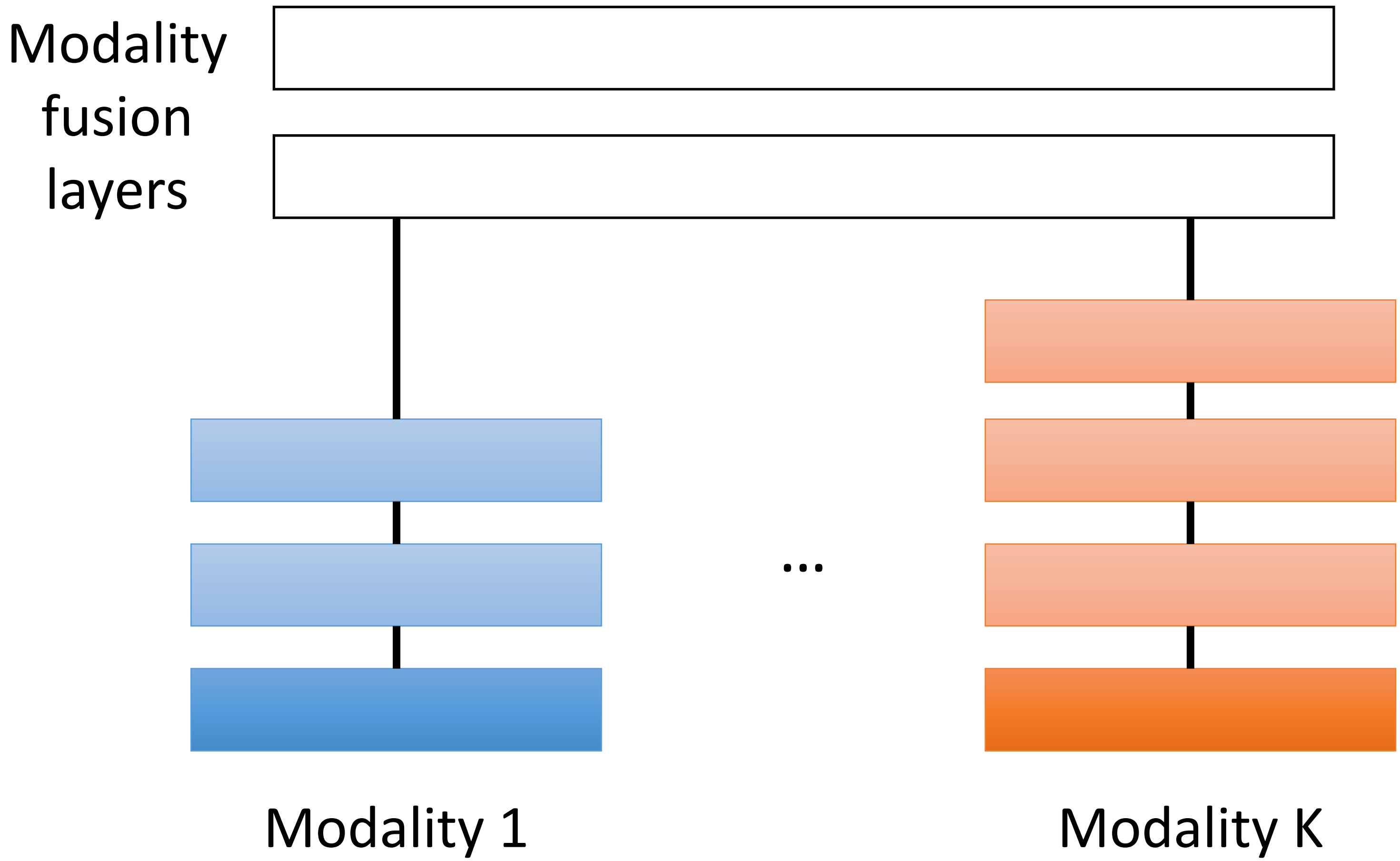
Level 1 Filters



Level 2 Filters

https://github.com/monsta-hd/boltzmann-machines

Multimodal DBM



N. Srivastava, R. Salakhutdinov, Multimodal Learning with Deep Boltzmann Machines, JMLR 2014

Multimodal DBM – Image and Text

$P(txt|img)$

Image	Given Tags	Generated Tags	Input Tags	Nearest neighbors to generated image features
	pentax, k10d, kangarooisland, southaustralia, sa, 300mm, australia, australiansealion	beach, sea, surf, strand, shore, wave, seascape, sand, ocean, waves	nature, hill, scenery, green, clouds	 
	< no text >	night, lights, christmas, nightshot, nacht, nuit, notte, longexposure, noche, nocturna	flower, nature, green, flowers, petal, petals, bud	 
	aheram, 0505, sarahc, moo	portrait, bw, balckandwhite, people, faces, girl, blackwhite, person, man	blue, red, art, artwork, painted, paint, artistic, surreal, gallery, bleu	 
	unseulpixel, naturey crap	fall, autumn, trees, leaves, foliage, forest, woods, branches, path	bw, blackandwhite, noiretblanc, bianconero, blancoynegro	 

$P(img|txt)$

N. Srivastava, R. Salakhutdinov, Multimodal Learning with Deep Boltzmann Machines, JMLR 2014

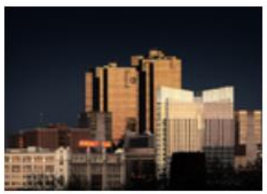
Multimodal DBM – Sampling

Step 50	Step 100	Step 150	Step 200	Step 250
travel	beach	sea	water	italy
trip	ocean	beach	canada	water
vacation	waves	island	bc	sea
africa	sea	vacation	britishcolumbia	boat
earthasia	sand	travel	reflection	italia
asia	nikon	ocean	alberta	mare
men	surf	caribbean	lake	venizia
2007	rocks	tropical	quebec	acqua
india	coast	resort	ontario	ocean
tourism	shore	trip	ice	venice

Input tags	Step 50	Step 100	Step 150	Step 200	Step 250
purple, flowers					
car, automobile					

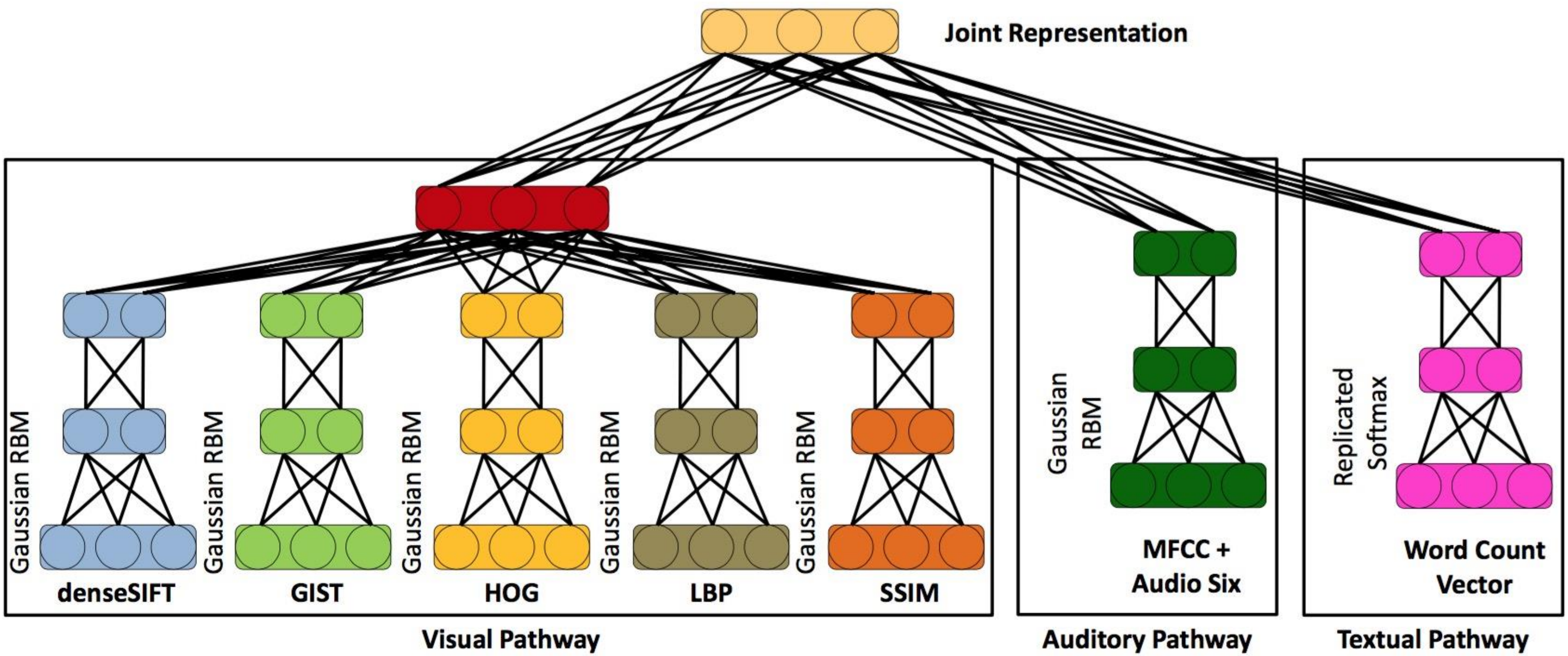
N. Srivastava, R. Salakhutdinov, Multimodal Learning with Deep Boltzmann Machines, JMLR 2014

Multimodal DBM – Multimodal Quering

Multimodal Query	**Top 4 retrieved results**			
 hongkong, causewaybay, shoppingcentre, building, mall	 howell, bridge, genesee, river, rochester, downtown, building	 london, uk, night, skyline, river, thames, lights, bridge	 edinburgh, scotland, dusk, bank	 arcoiris, fincadehierro, lluvia, sannicolas, valencia
 me, myself, eyes, blue, hair	 urban, me, abigfave, fiveflickrfavs,	 trisha, mynewcamera, lake, field, girl	 me, ofme, self, selfportrait	 pink, prettyinpink, explored

N. Srivastava, R. Salakhutdinov, Multimodal Learning with Deep Boltzmann Machines, JMLR 2014

Multimodal DBM for Multimedia



Pang et al. Deep Multimodal Learning for Affective Analysis and Retrieval. IEEE Trans. on Multimedia, 2015

Take Home Messages

- Regularized autoencoder
  - Optimize reconstruction quality
  - Constrain stored information
- Autoencoder training is manifold learning
  - Learn a latent space manifold where input data resides
  - Store only variations that are useful to represent training data
- Autoencoders learn a (conditional) distribution of input data $P(\hat{\boldsymbol{x}}| \ldots)$
- Deep AE: pretraining, fine tuning, supervised optimization
- Use AE for finding new/useful data representations
  - Or to learn its distribution

Next Lecture

Gated Recurrent Networks

- Learning with sequential data
- Gradient issues
- Gated RNN
  - Long-Short Term Memories (LSTM)
  - Gated Recurrent Units (GRU)
- Advanced topics
  - Understanding and exploiting memory encoding
  - Applications