On the Difficulty of Training Deep Architectures

Yoshua Bengio, U. Montreal

Deep Learning Workshop @ Gatsby Unit, UCL, London, U.K.

July 9th, 2009

Thanks to: Aaron Courville, Pascal Vincent, Dumitru Erhan, Olivier Delalleau, Olivier Breuleux, Guillaume Desjardins, Pascal Lamblin, James Bergstra, Nicolas Le Roux, Myriam Côté, Jérôme Louradour, Ronan Collobert, Jason Weston

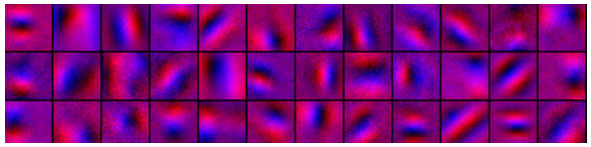
Deep Architectures Work Well

- Beating shallow neural networks on vision and NLP tasks
- Beating SVMs on visions tasks from pixels (and handling dataset sizes that SVMs cannot handle in NLP)
- Reaching state-of-the-art performance in NLP
- Beating deep neural nets without unsupervised component
- Learn visual features similar to V1 and V2 neurons

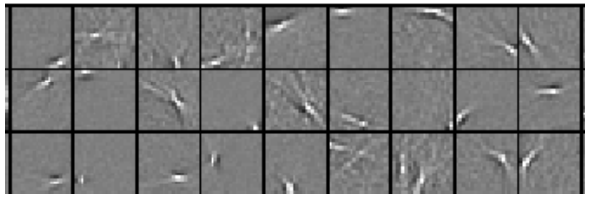
WHY?

V1 and V2-like Filters Learned

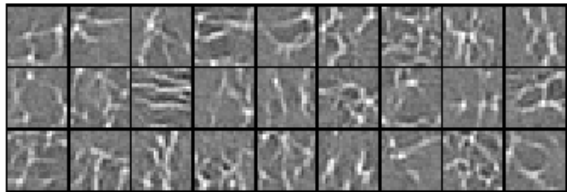
Slow features 1st layer



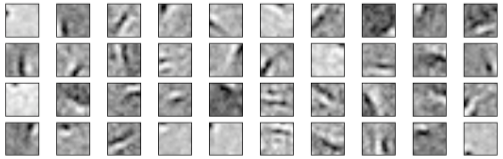
RBM 1st layer



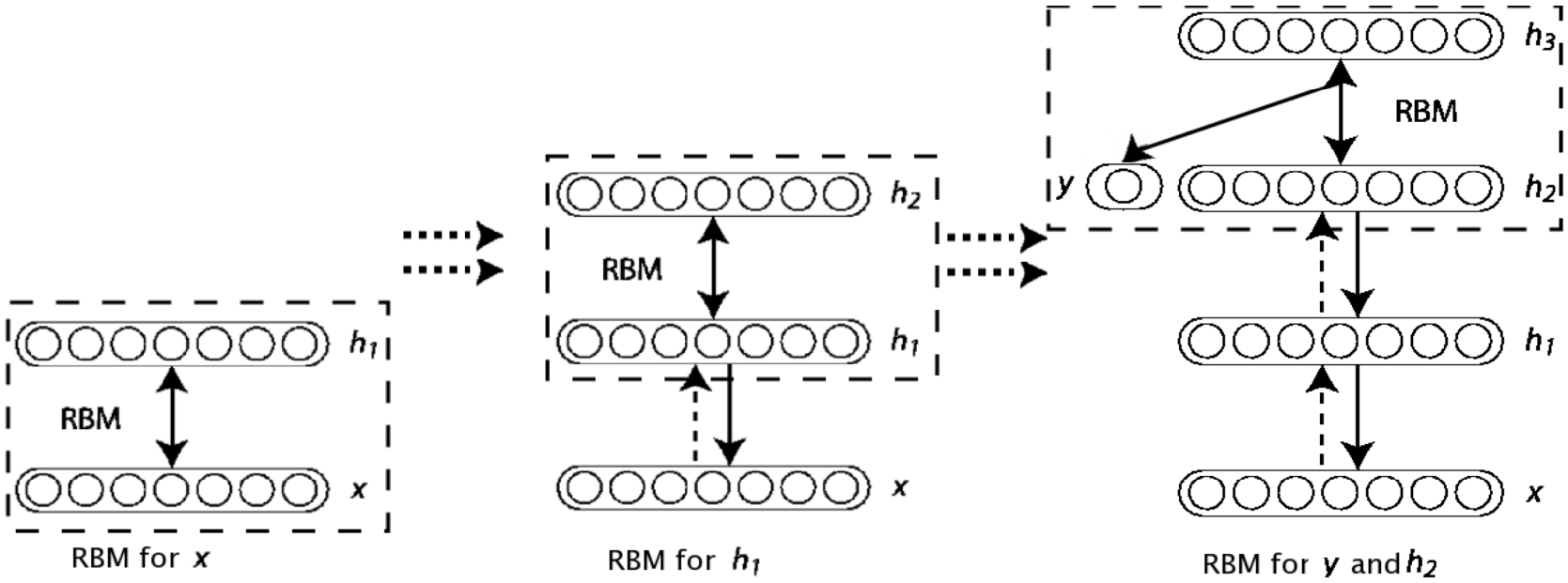
DBN 2nd layer



Denoising auto-encoder 1st layer



Greedy Layer-Wise Pre-Training

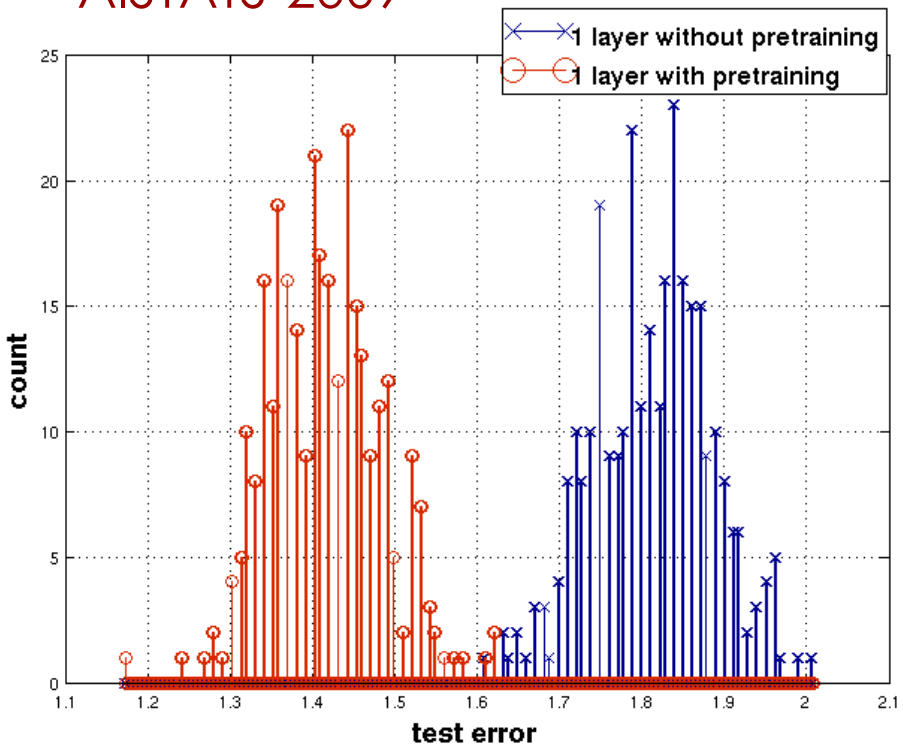


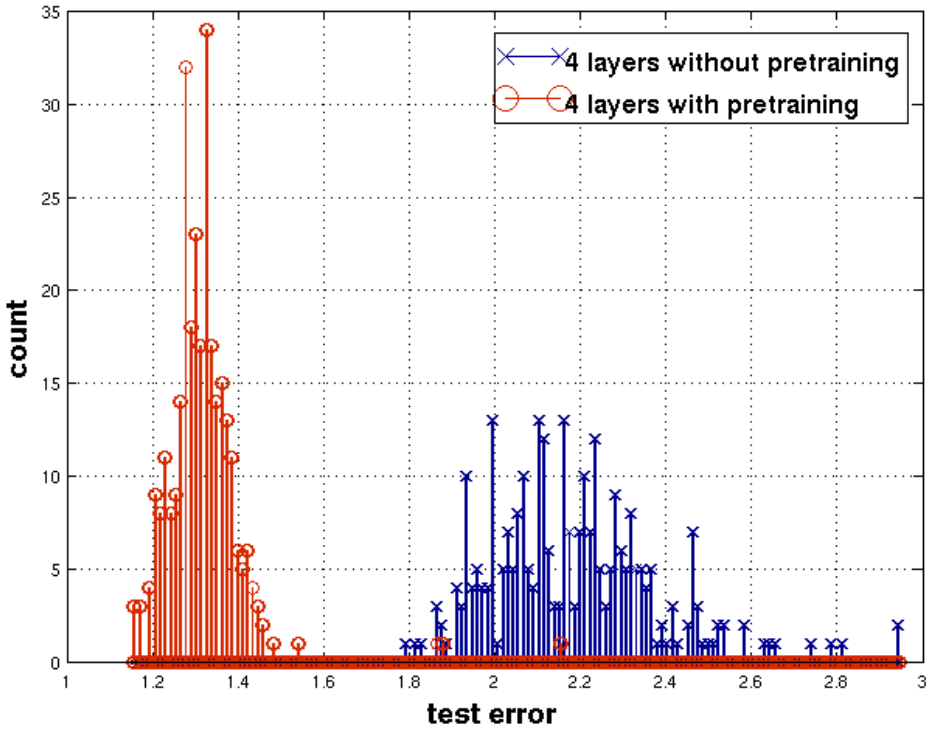
Stacking Restricted Boltzmann Machines (RBM) → Deep Belief Network (DBN)

→ Supervised deep neural network

Effect of Unsupervised Pre-training

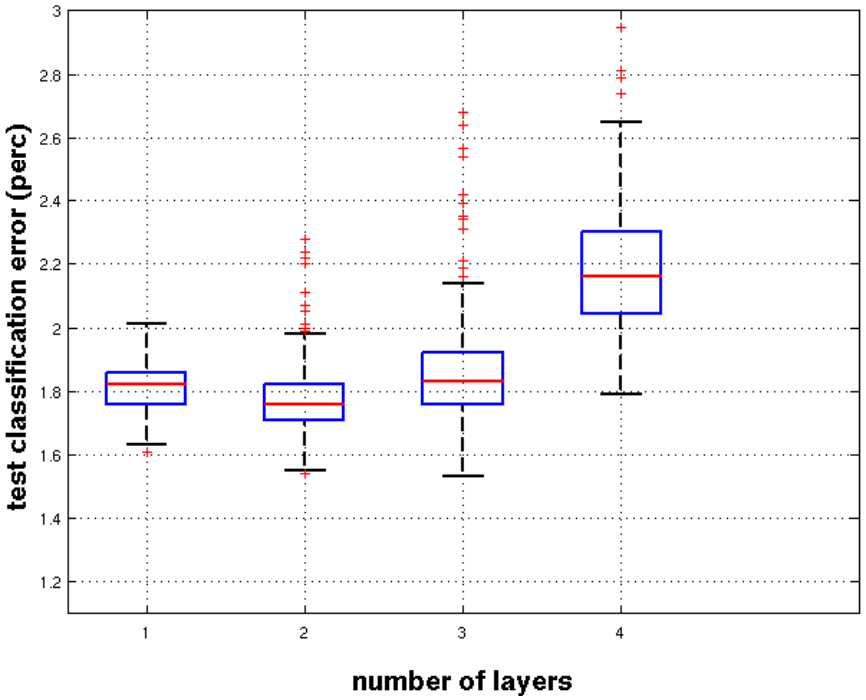
AISTATS'2009



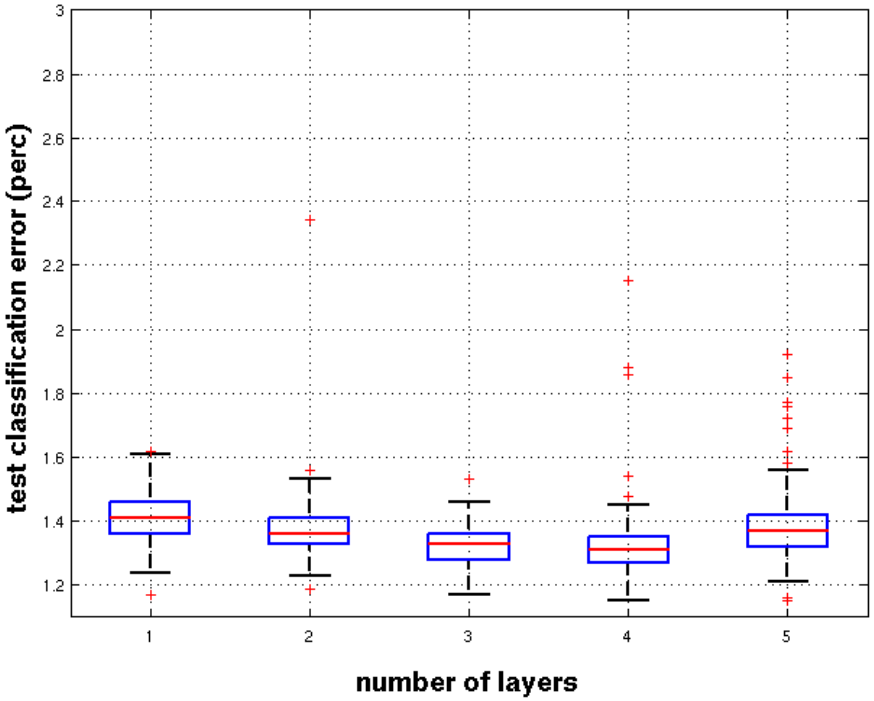


Effect of Depth

w/o pre-training



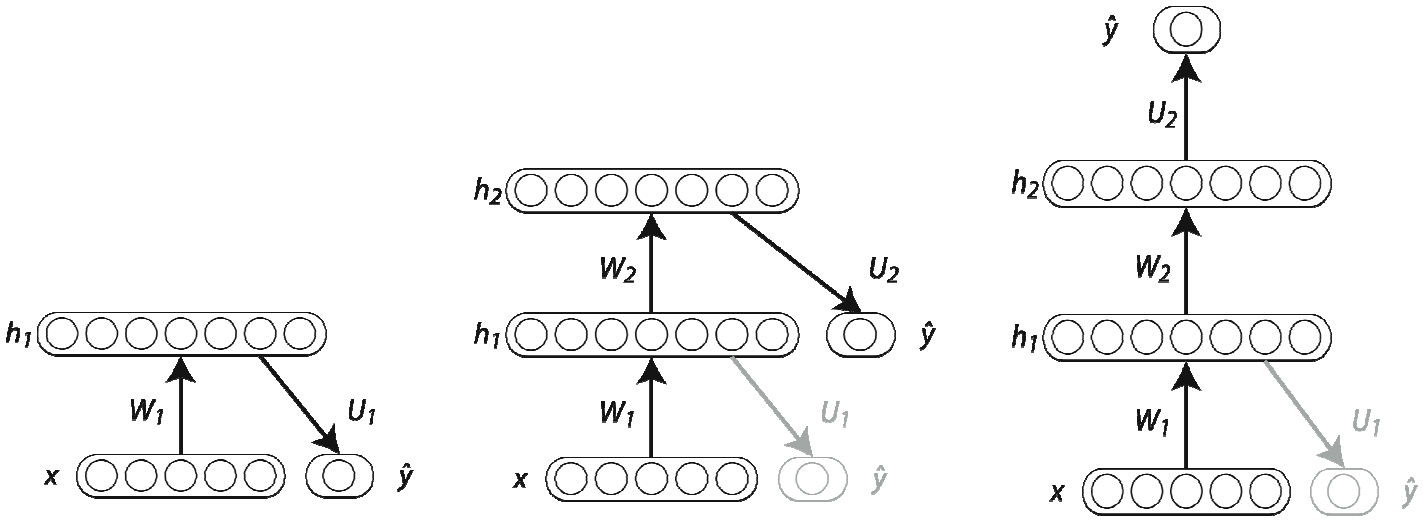
with pre-training



Why are Classifiers Obtained from Unsupervised Pre-Training Working so Well?

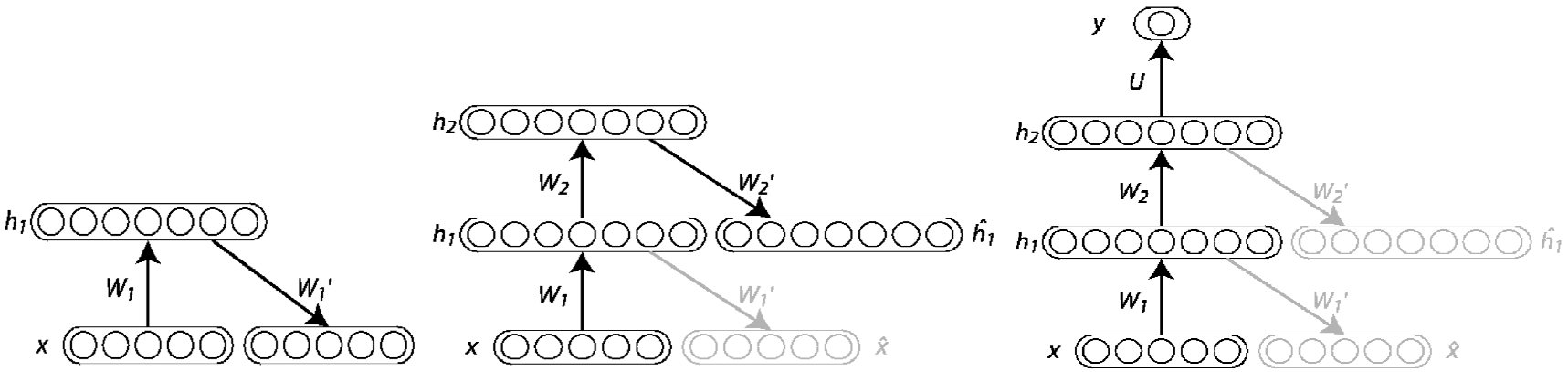
- General principles?
- Would these principles work for other single-level algorithms?
- Explanatory hypotheses?

Greedy Layerwise Supervised Training



Generally worse than unsupervised pre-training but better than ordinary training of a deep neural network (Bengio et al. 2007).

Stacking Auto-Encoders



Auto-Encoders and CD

RBM log-likelihood gradient written as converging expansion:

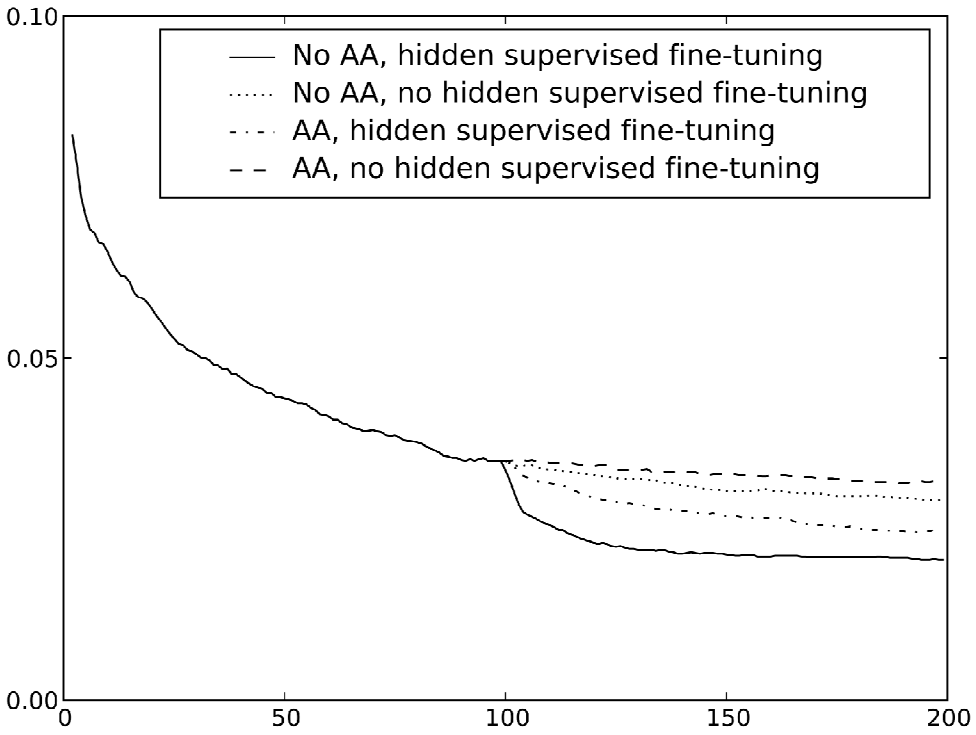
- CD-k = 2 k terms
- reconstruction error ~ 1 term

$$\frac{\partial \log P(x_1)}{\partial \theta} = \sum_{s=1}^{t-1} \left(E\left[\frac{\partial \log P(x_s|h_s)}{\partial \theta} \middle| x_1 \right] + E\left[\frac{\partial \log P(h_s|x_{s+1})}{\partial \theta} \middle| x_1 \right] \right) + E\left[\frac{\partial \log P(x_t)}{\partial \theta} \middle| x_1 \right]$$

(Bengio & Delalleau 2009)

Supervised Fine-Tuning is Important

- Greedy layer-wise unsupervised pre-training phase with RBMs or auto-encoders on MNIST
- Supervised phase with or without unsupervised updates, with or without fine-tuning of hidden layers
- Can train all RBMs at the same time, same results



Two phases?
Pre-training + Fine-tuning

- Currently best results generally obtained when doing purely supervised fine-tuning after unsupervised pre-training
- Kind of disappointing
- Can we avoid the fine-tuning alltogether?
- Can we fold both phases together? (would be very useful for online learning on huge datasets)
- Can we avoid layer-wise initialization?

Sparse Auto-Encoders

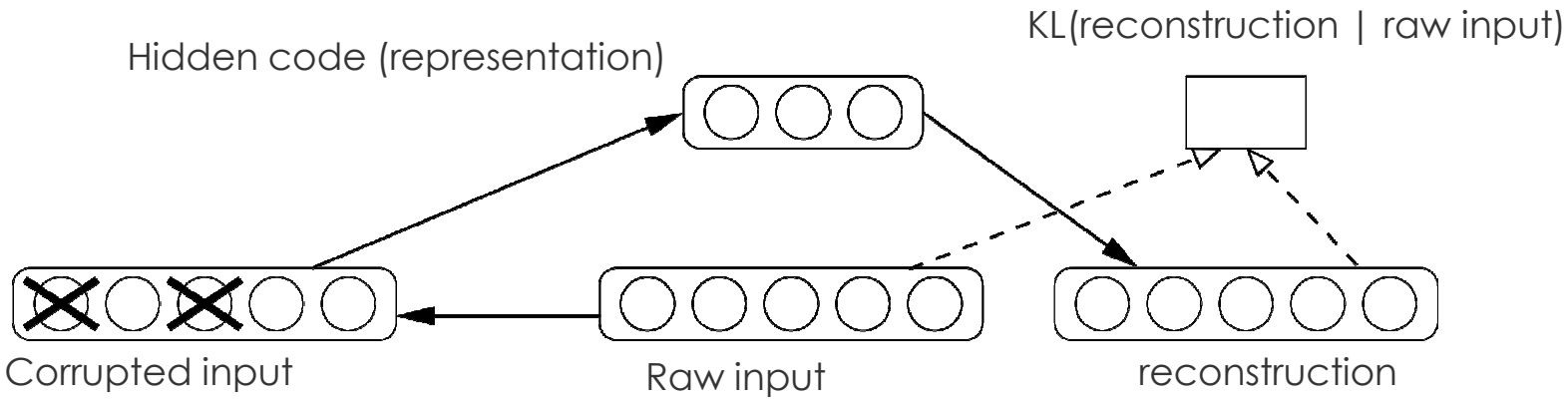
(Ranzato et al, 2007; Ranzato et al 2008)

- Sparsity penalty on the intermediate codes
- Like sparse coding but with efficient run-time encoder
- Sparsity penalty pushes up the free energy of all configurations (proxy for minimizing the partition function)
- Impressive results in object classification (convolutional nets):
  - MNIST 0.5% error = record-breaking
  - Caltech-101 65% correct = state-of-the-art (Jarrett et al, ICCV 2009)
- Similar results with a sparse convolutional DBN (Lee et al, ICML'2009)

Denoising Auto-Encoder

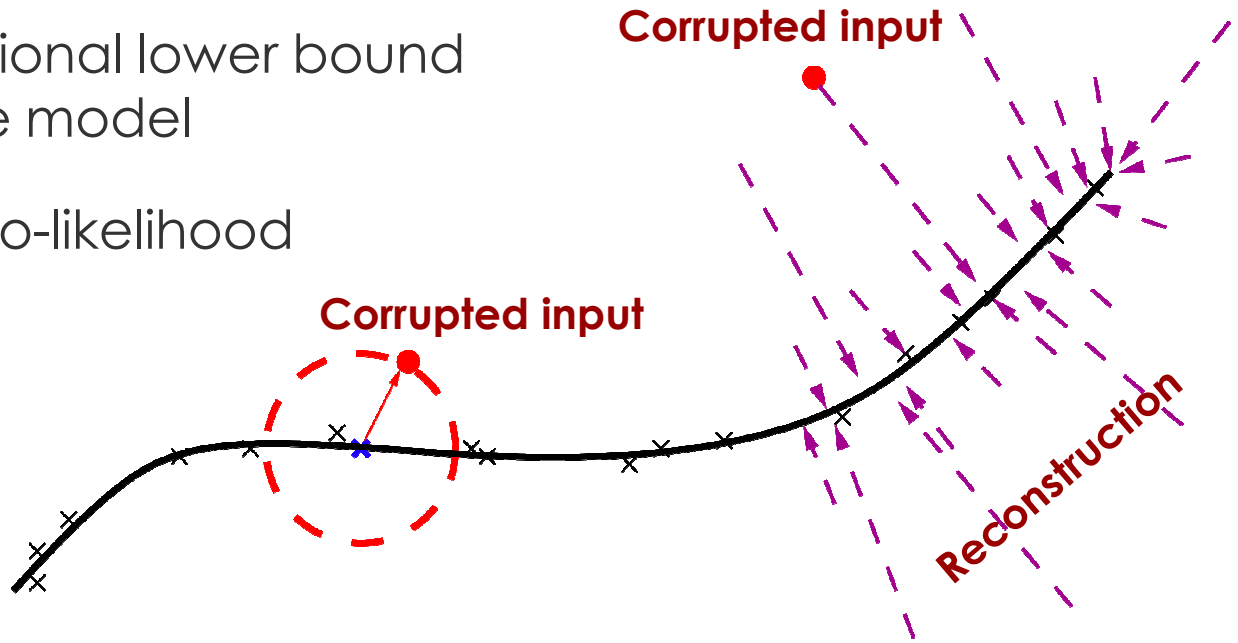
(Vincent et al, ICML 2008)

- Corrupt the input
- Reconstruct the uncorrupted input



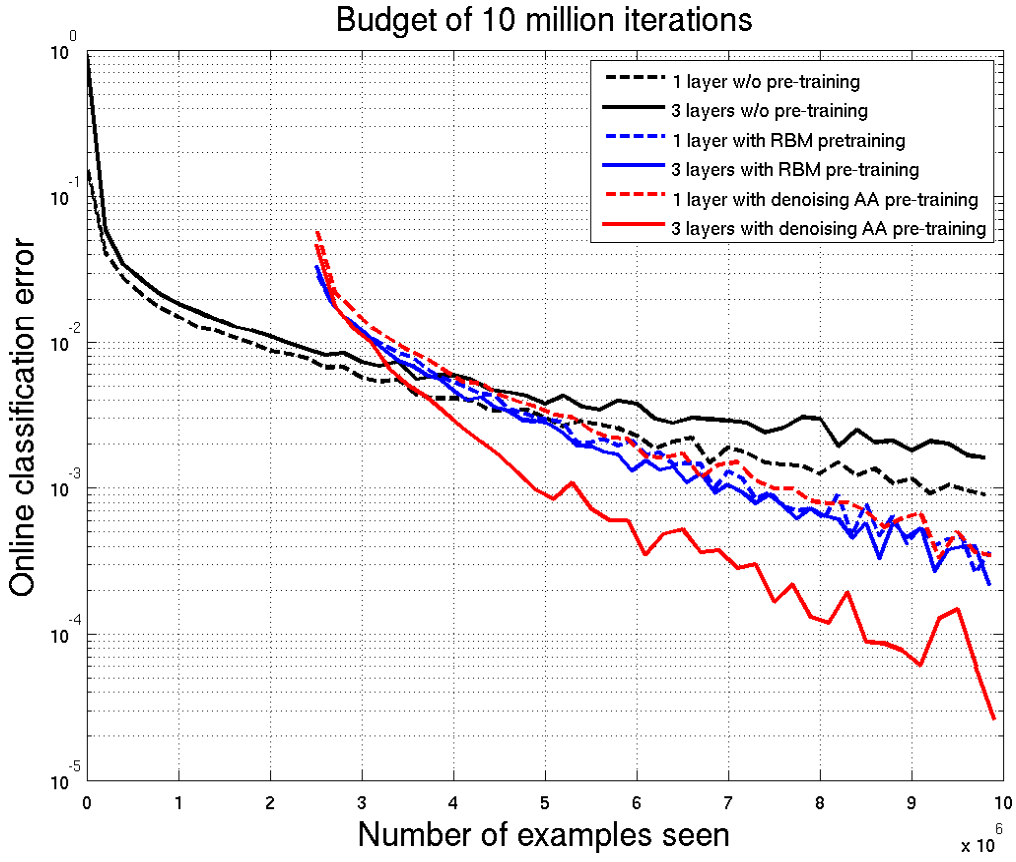
Denoising Auto-Encoder

- Learns a vector field towards higher probability regions
- Minimizes variational lower bound on a generative model
- Similar to pseudo-likelihood



Stacked Denoising Auto-Encoders

- No partition function, can measure training criterion
- Encoder & decoder: any parametrization
- Performs as well or better than stacking RBMs for usupervised pre-training



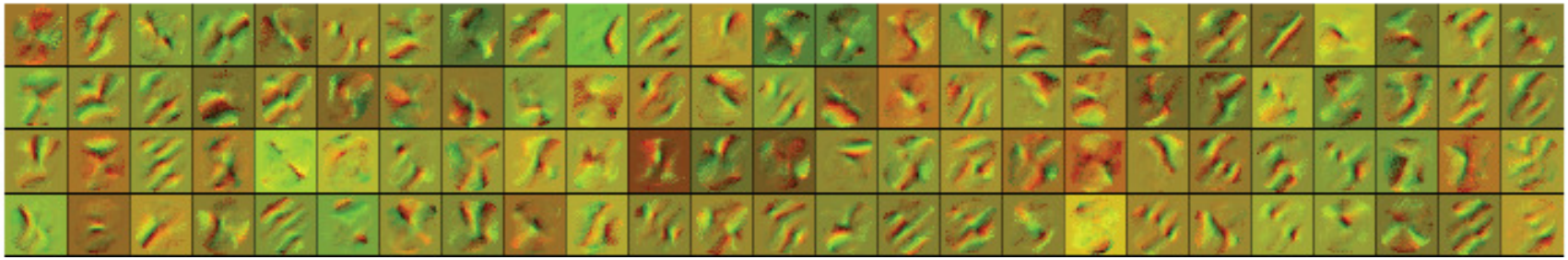
Infinite MNIST

Learning Layer-Local Embeddings

- (Weston & Collobert, ICML 2008) similar/dissimilar examples provide layer-local unsupervised criterion (Hadsell et al, CVPR 2006)
- Margin hinge loss: learned representations of similar examples should be more similar than of non-similar pairs
- Global supervised + layer-local unsupervised gradients
- Successfully tested in semi-supervised setting
- Trained up to 15-layer deep networks!
- No comparison yet with RBMs and auto-encoder variants

Slow Features & Temporal Constancy

- Similar pairs = successive inputs in a sequence (e.g. video)
- Try to make code covariance ~ **I**, can be done in O(nfeat)



Pre-training Dataset	Number of pretraining iterations ($\times 10^4$)					
	0	1	2	3	4	5
MIXED-movies	1.56	1.32	1.42	1.34	1.35	1.28
MNIST-movies	1.56	1.32	1.33	1.32	1.27	1.30

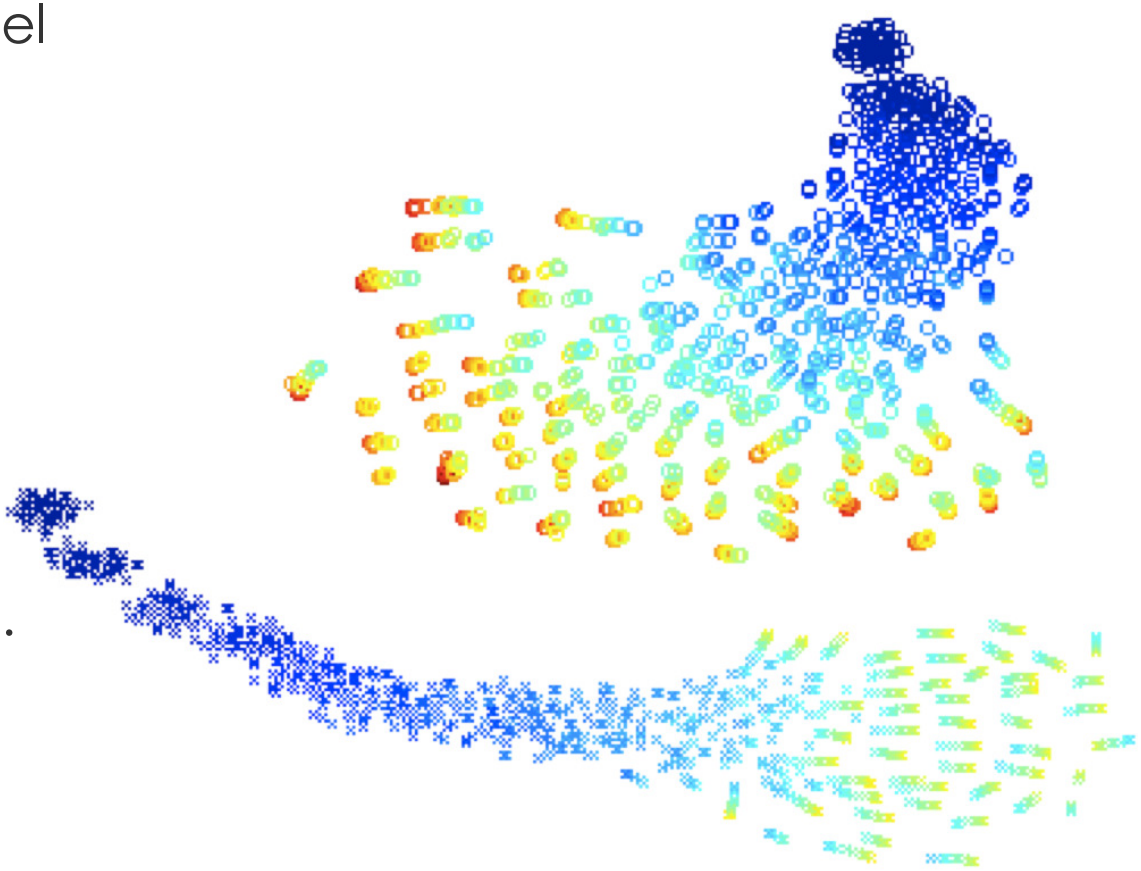
Why is Unsupervised Pre-Training Working So Well? Hypotheses:

- Regularization hypothesis:
  - Unsupervised component forces model close to P(x)
  - Representations good for P(x) are good for P(y | x)

- Optimization hypothesis:
  - Unsupervised initialization near better local minimum of P(y | x)
  - Can reach lower local minimum otherwise not achievable by random initialization

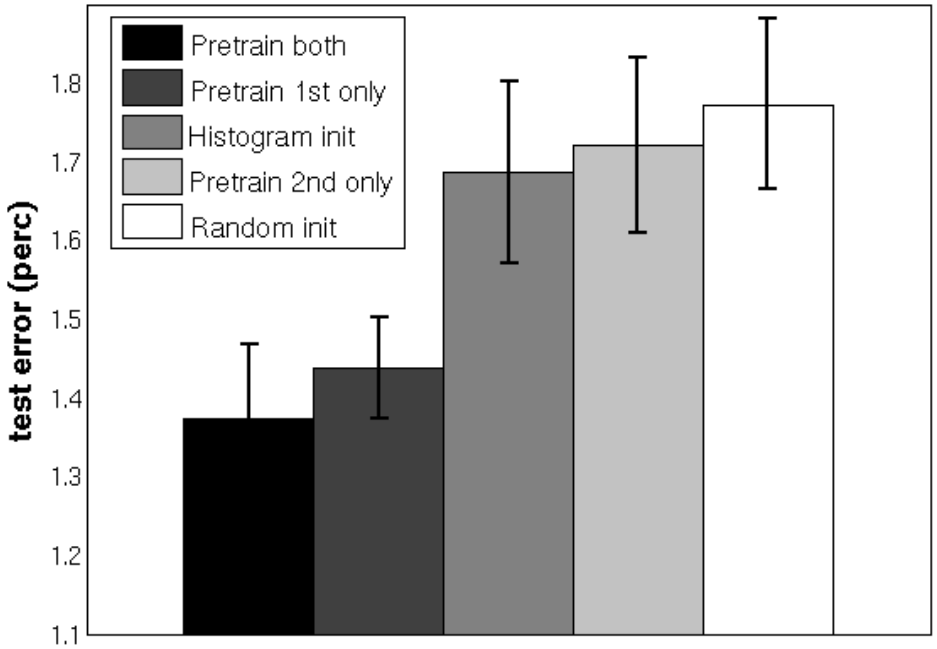
Learning Trajectories in Function Space

- Each point a model in function space
- Color = epoch
- Top: trajectories w/o pre-training
- Each trajectory converges in different local min.
- No overlap of regions with and w/o pre-training



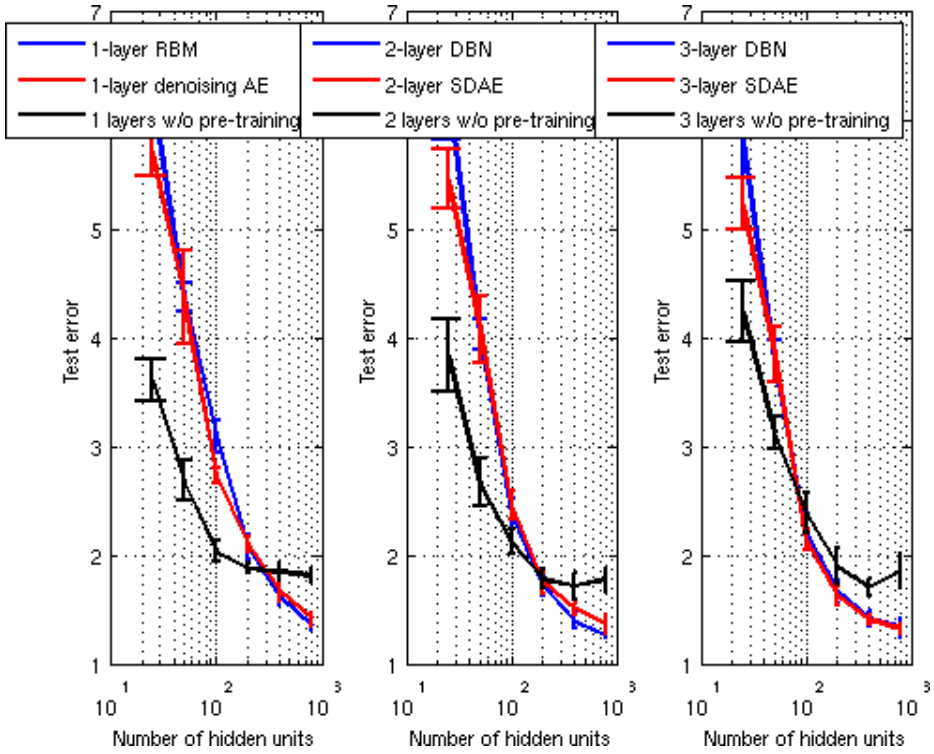
Pre-Training Lower Layers More Critical

What matters is not just the marginal distribution over initial weight values (Histogram init.)



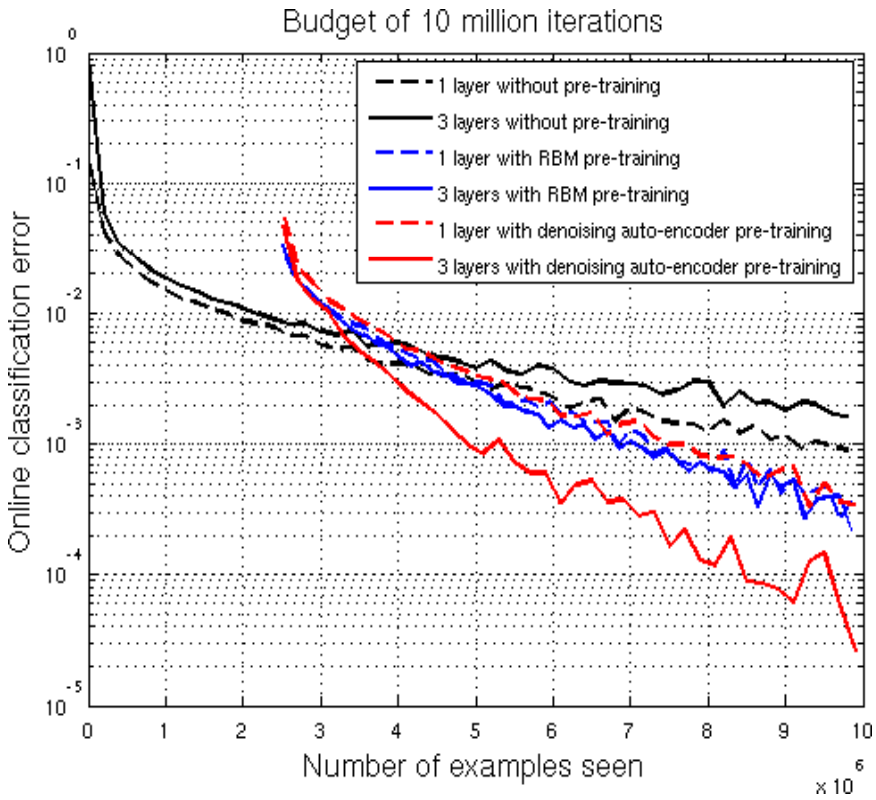
Unsupervised Learning as Regularizer

- Adding extra regularization (reducing # hidden units) hurts more the pre-trained models
- Pre-trained models have less variance wrt training sample
- Regularizer = infinite penalty outside of region compatible with unsupervised pre-training

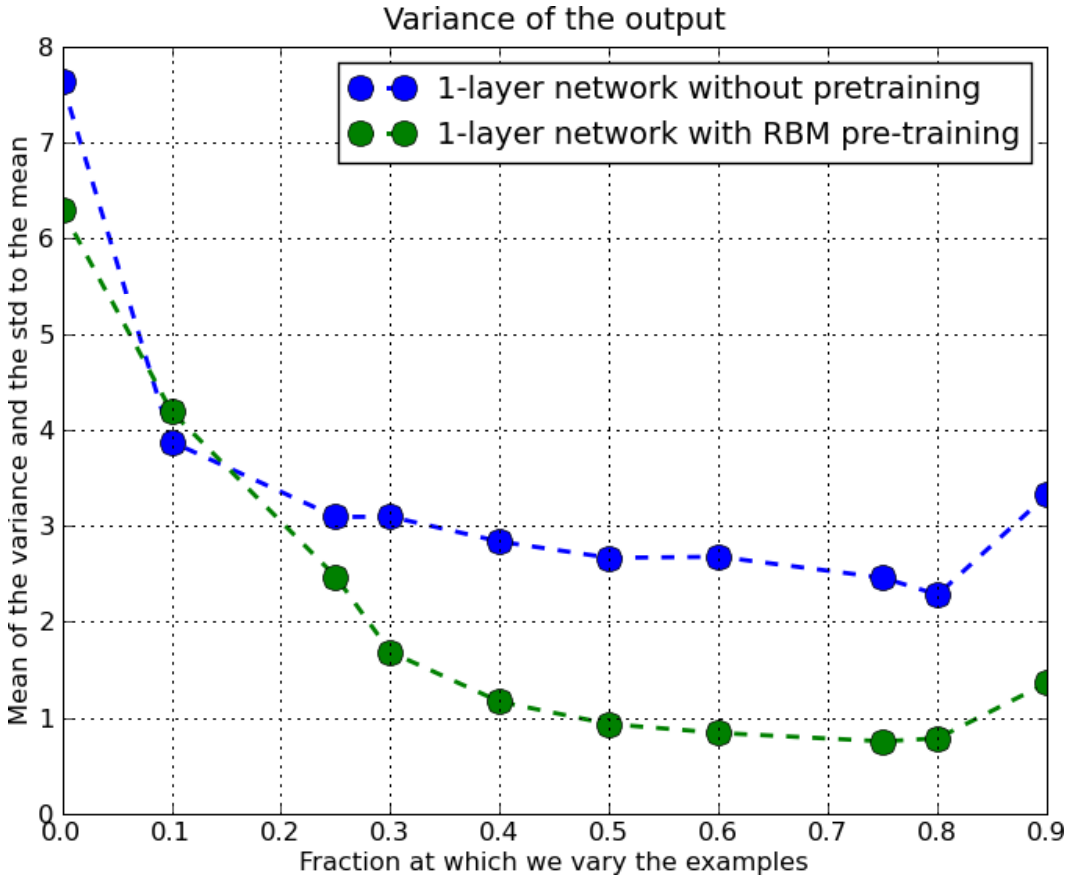


Better Optimization of Online Error

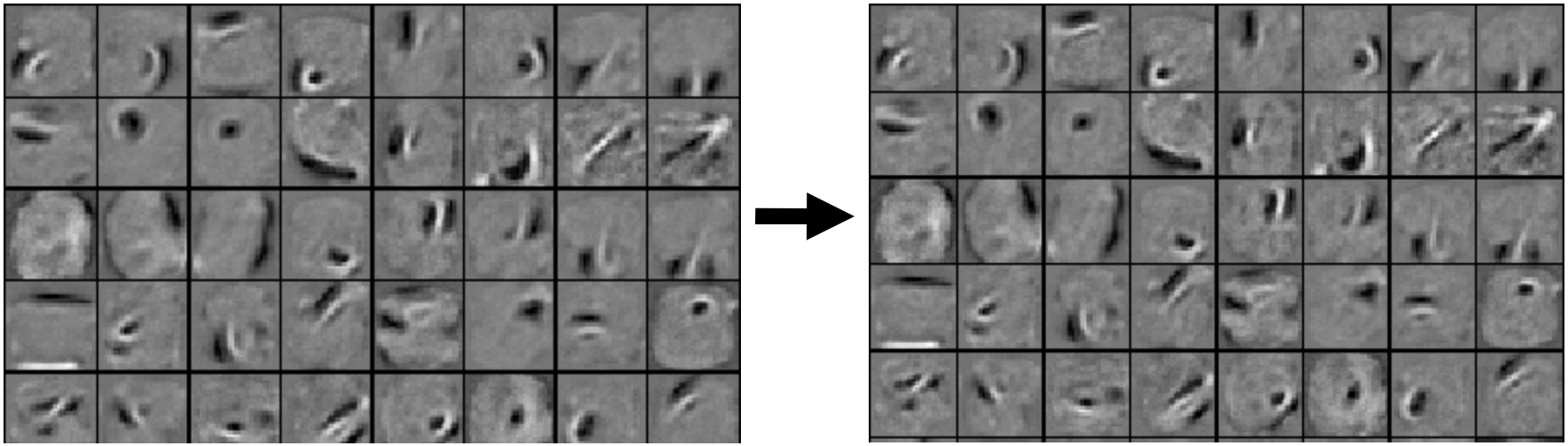
- Both training and online error are smaller with unsupervised pre-training
- As # samples → ∞
  training err. = online err. = generalization err.
- Without unsup. pre-training: can't exploit capacity to capture complexity in target function from training data



Critical Impact of Early Updates



Learning Dynamics of Deep Nets

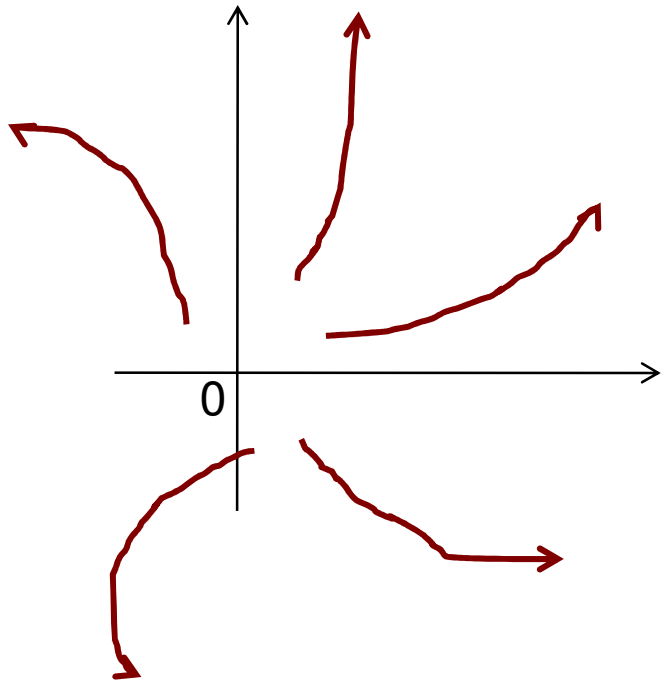


Before fine-tuning

After fine-tuning

Learning Dynamics of Deep Nets

- As weights become larger, get trapped in basin of attraction ("quadrant" does not change)
- Initial updates have a crucial influence ("critical period"), explain more of the variance
- Unsupervised pre-training initializes in basin of attraction with good generalization properties



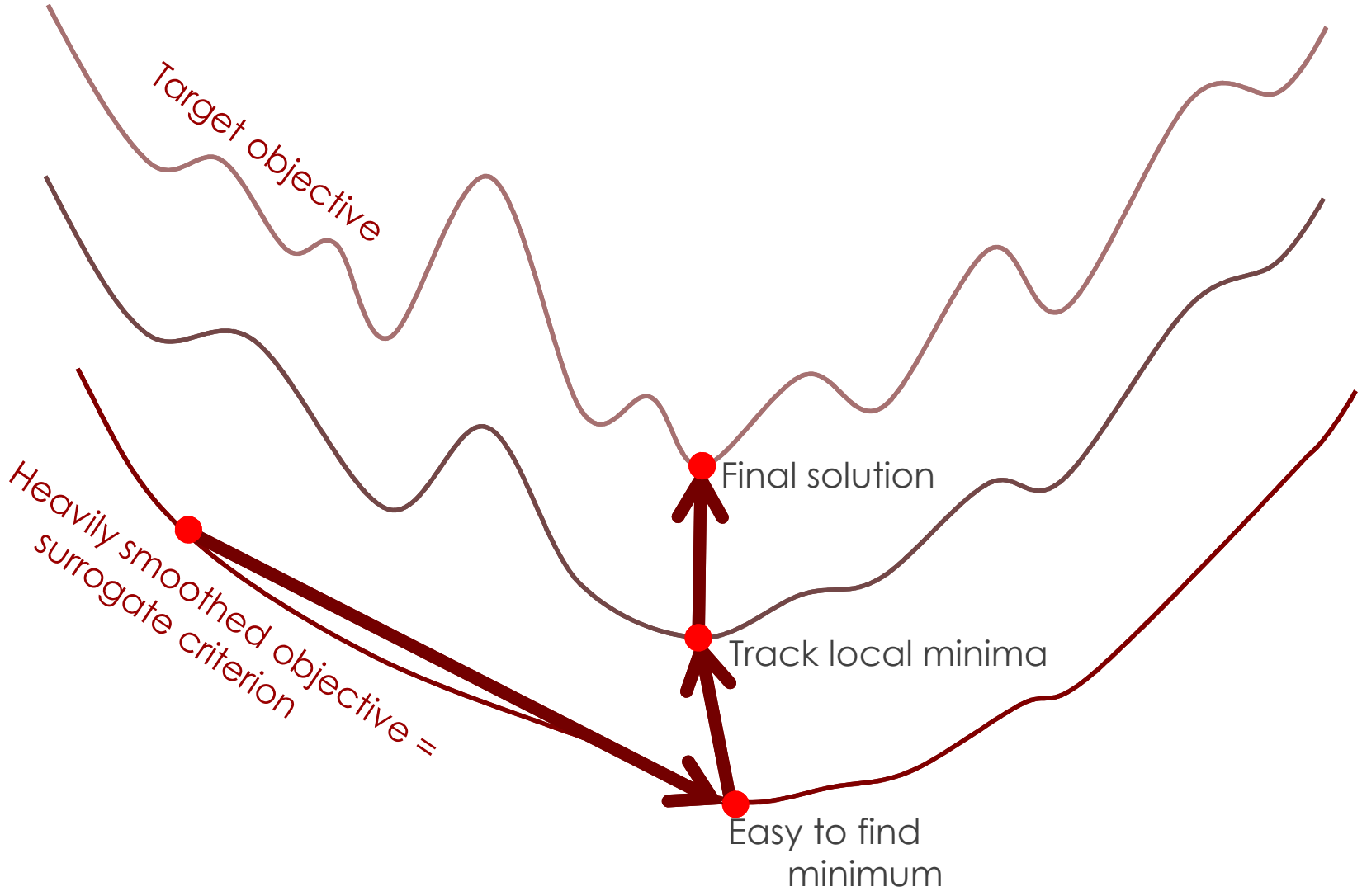
What Optimiztion Tricks?

- Humans somehow find a good solution to an intractable non-convex optimization problem

  How?

- Guiding the optimization near good solutions
- Guiding / giving hints to intermediate layers

Continuation Methods



The Credit Assignment Problem

- Even with the correct gradient, lower layers (far from the prediction, close to input) are the most difficult to train
- Lower layers benefit most from unsupervised pre-training:
  - Local unsupervised signal = extract / disentangle factors
  - Temporal constancy
  - Mutual information between multiple modalities
- Credit assignment / error information not flowing easily?
- Related to difficulty of credit assignment through time?

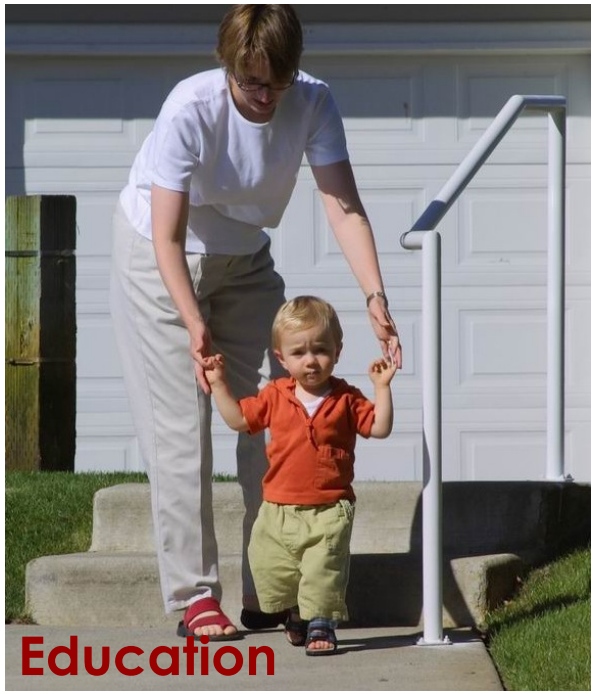
Guiding the Stochastic Optimization of Representations

- Train lower levels first (DBNs)
- Start with more noise / larger learning rate (babies vs adults)
- Slow features / multiple time scales
- Cross-modal mutual information
- Curriculum / shaping

Curriculum Learning

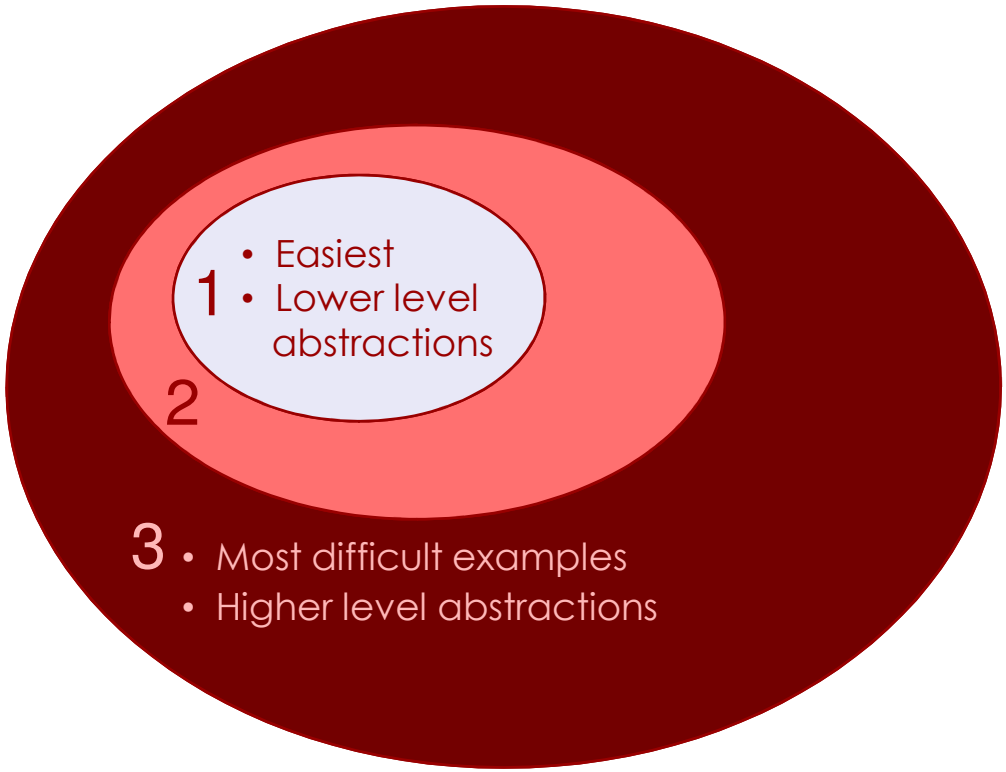
ICML'2009

- Guided learning helps training humans and animals



- Start from simpler examples / easier tasks (Piaget 1952, Skinner 1958)
- Cognition Journal: (Elman 1993) vs (Rohde & Plaut 1999), (Krueger & Dayan 2009)

Curriculum Learning as Continuation



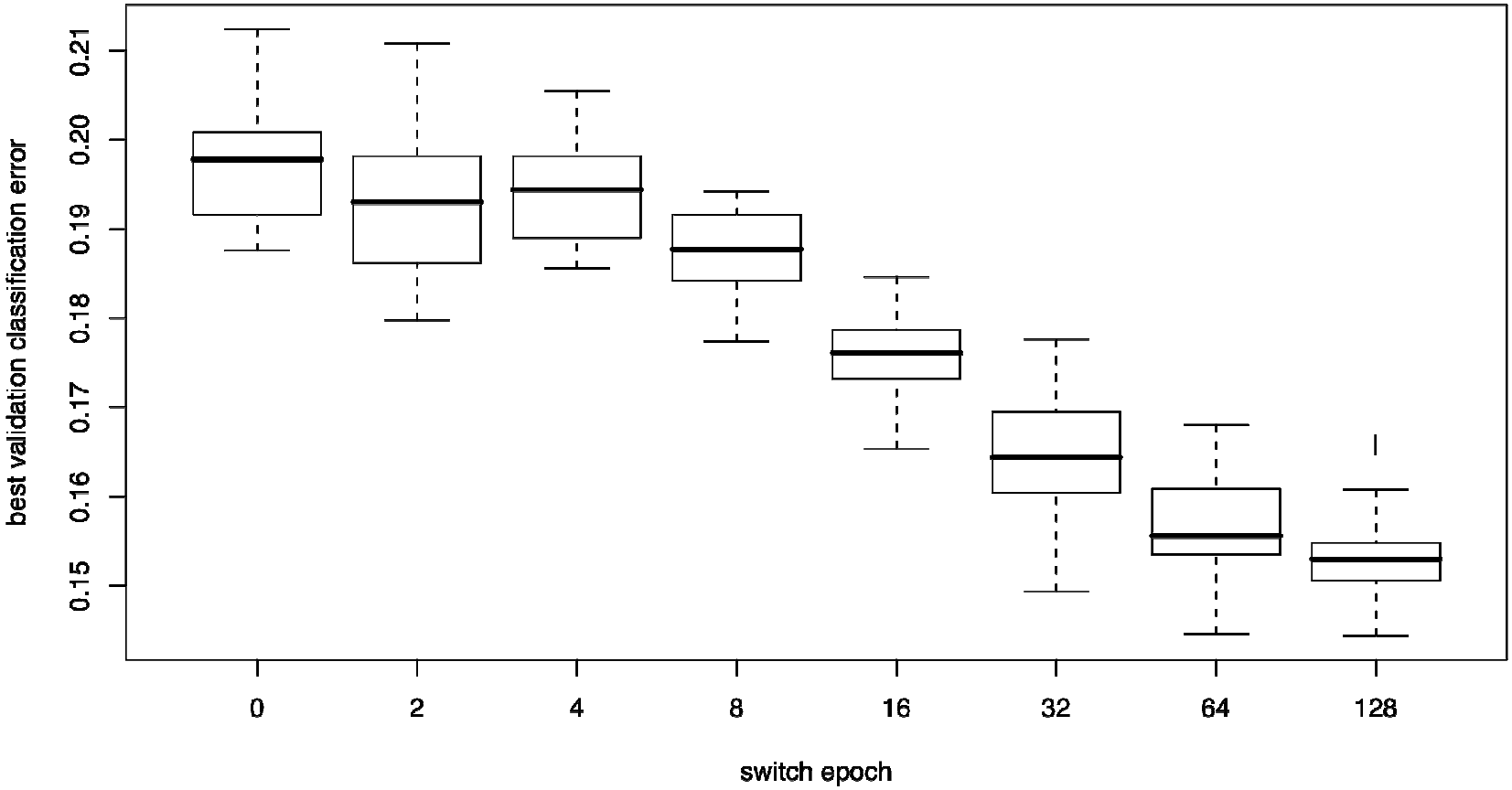
- Sequence of training distributions
- Initially peaking on easier / simpler ones
- Gradually give more weight to more difficult ones until reach target distribution

Shape Recognition

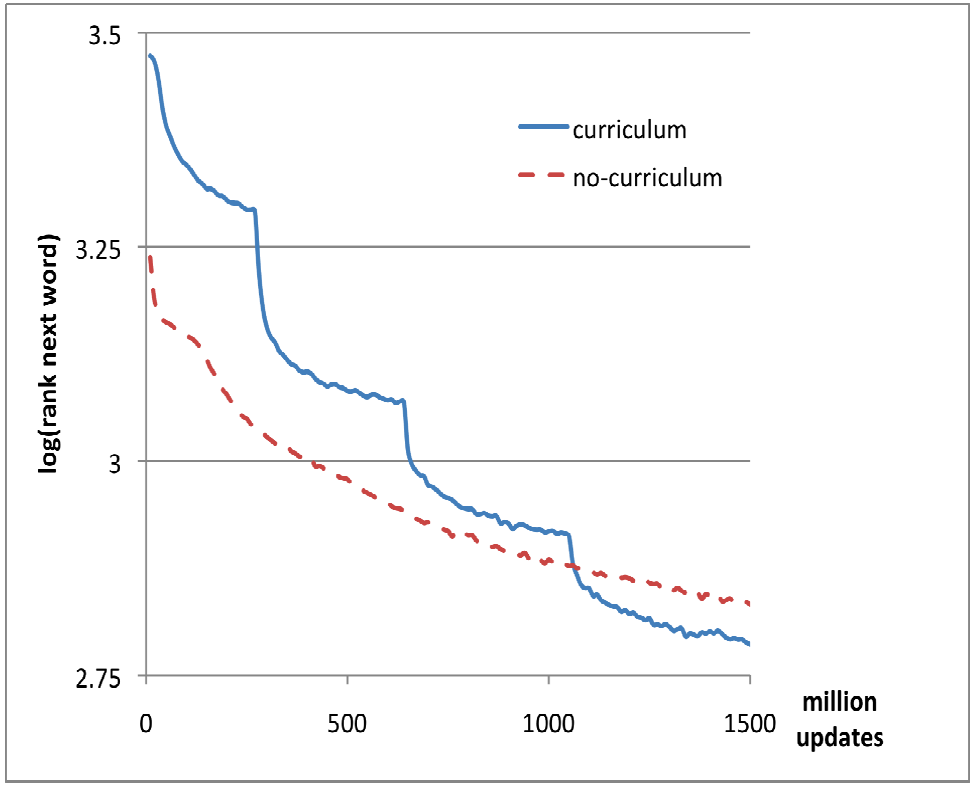
First: easier, basic shapes

Second = target: more varied geometric shapes

Shape Recognition Results



Language Modeling Results



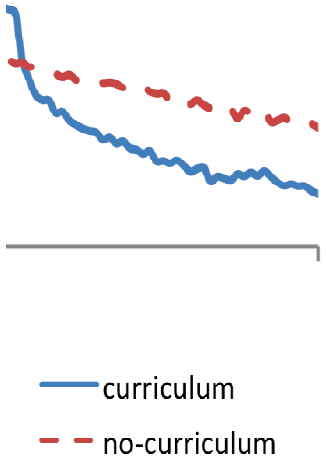
- Gradually increase the vocabulary size (dips)
- Train on Wikipedia with sentences containing only words in vocabulary

Order & Selection of Examples Matters

- Curriculum learning

(Bengio et al, ICML'2009; Krueger & Dayan 2009)

- Start with easier examples
- Faster convergence to a better local minimum in deep architectures
- Also acts like a regularizer with optimization effect?
- Influencing learning dynamics can make a big difference



Level-Local Learning is Important?

- Initializing each layer of an unsupervised Deep Boltzmann Machine helps a lot
- Initializing each layer of a supervised neural network as an RBM helps a lot
- Helps most the layers further away from the target
- Not just an effect of unsupervised prior
- Jointly training all the levels of a deep architecture is difficult
- Initializing using a level-local learning algorithm (RBM, auto-encoders, etc.) is a useful trick

Take-Home Messages

- Unsupervised pre-training greatly helps deep architectures
- Unsupervised pre-training of classifiers acts like a strange *regularizer* with improved *optimization* of online error
- Inference approximations and *learning dynamics* at least as important as the model
- Early examples have greater influence: critical period?
- Guiding learning dynamics seems important:
  - Local hints to each layer
  - Curriculum / shaping = continuation?

Some Open Problems

- Why is it difficult to train deep architectures?
- What is important in the learning dynamics?
- How to improve joint training and sampling of all layers?
- Other ways to guide training of intermediate representations?
- How to design curricula / select examples?
- More complex models to handle spatial structure of images, occlusion, temporal structure, stereo, multiple modalities, etc.

Thank you for your attention!

- Questions?
- Comments?