Deep learning of representations and its application to computer vision

Ian Goodfellow

Summary

- Deep learning background
- Four articles:
  - Spike-and-slab modeling
  - Multi-prediction deep Boltzmann machines
  - Maxout
  - Street number transcription

Machine learning

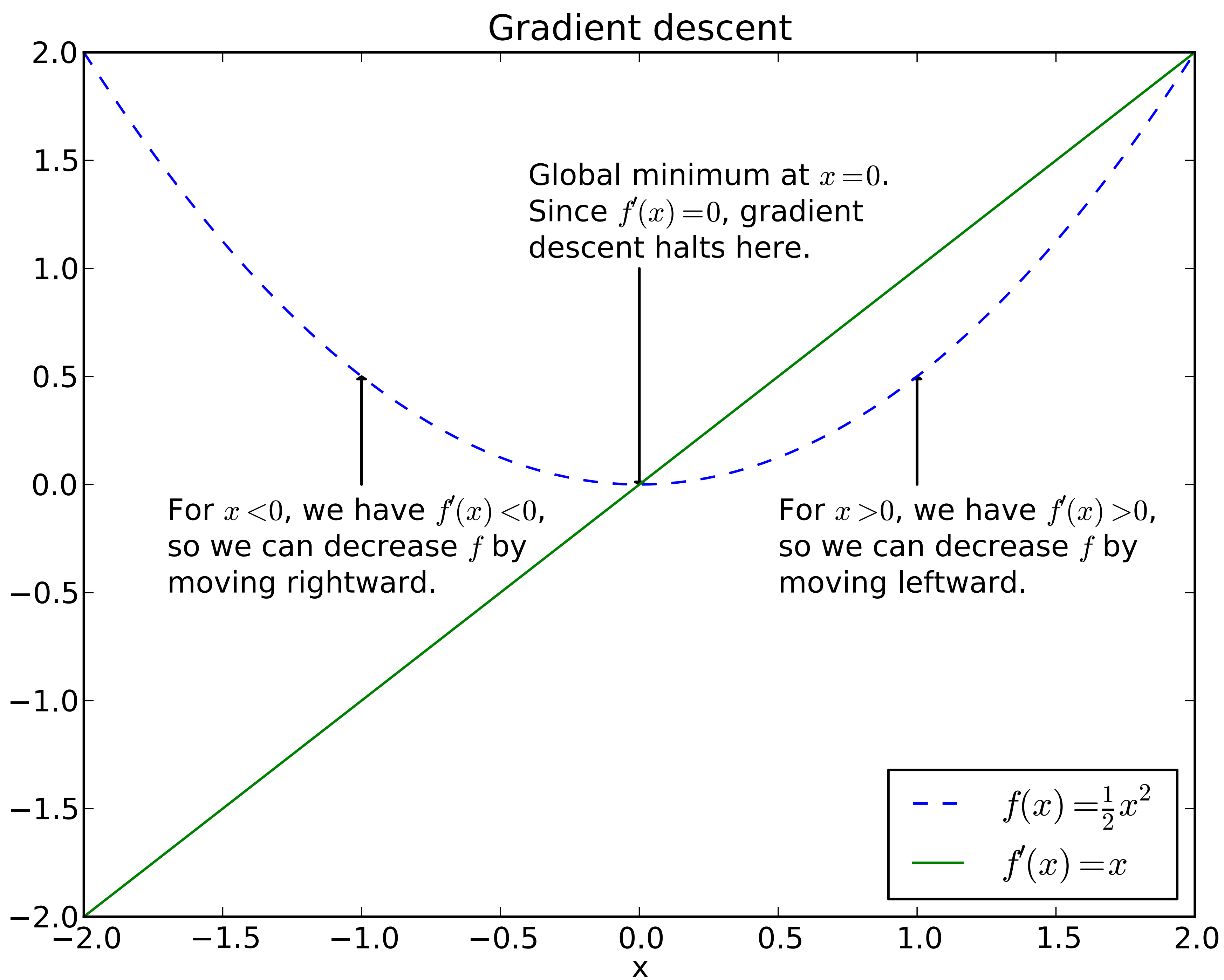
“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

-Tom Mitchell

Maximum likelihood estimation

- Pick parameters that maximize model's probability of generating the observed data
- Given enough data, recovers the true model

Gradient descent

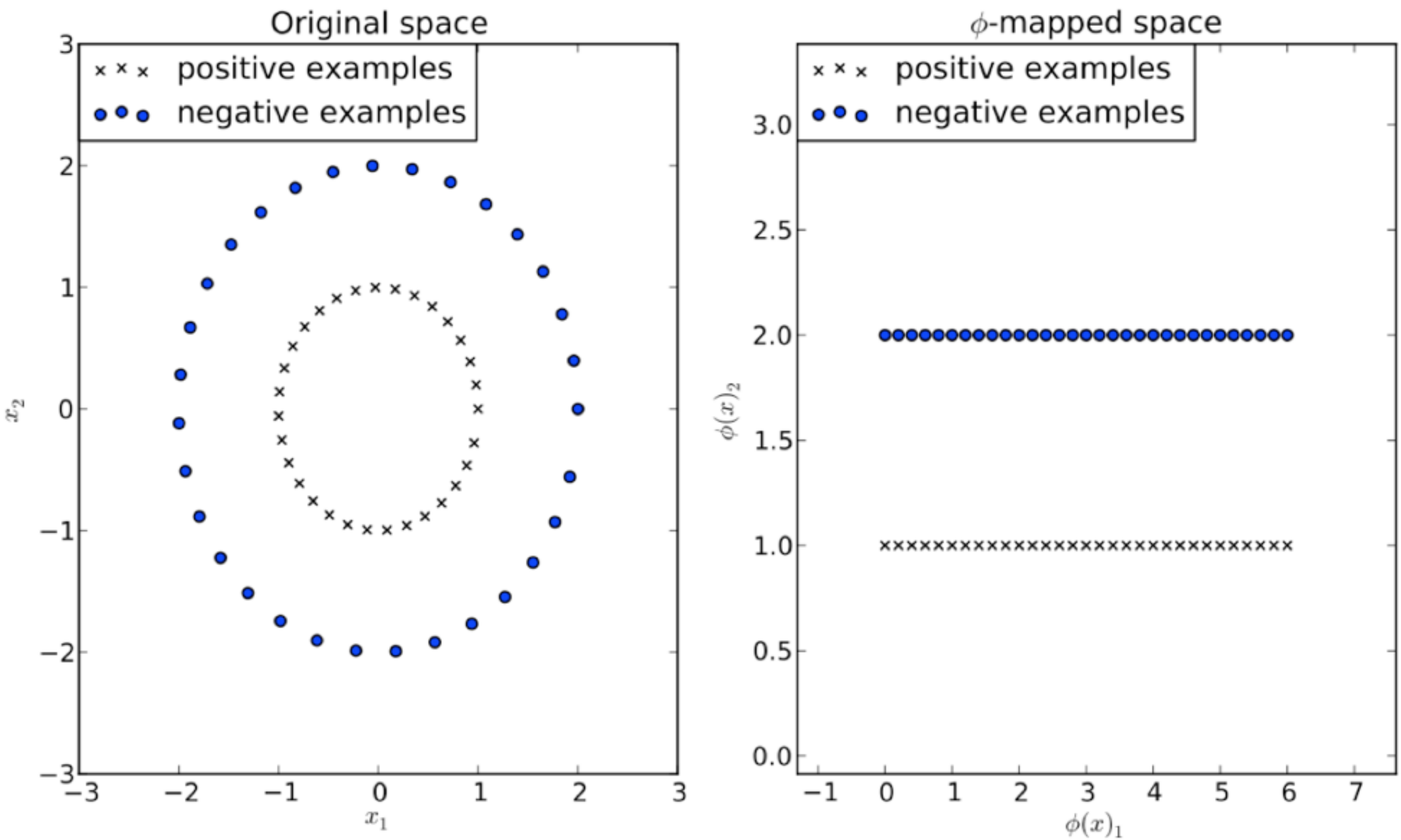


Supervised Learning

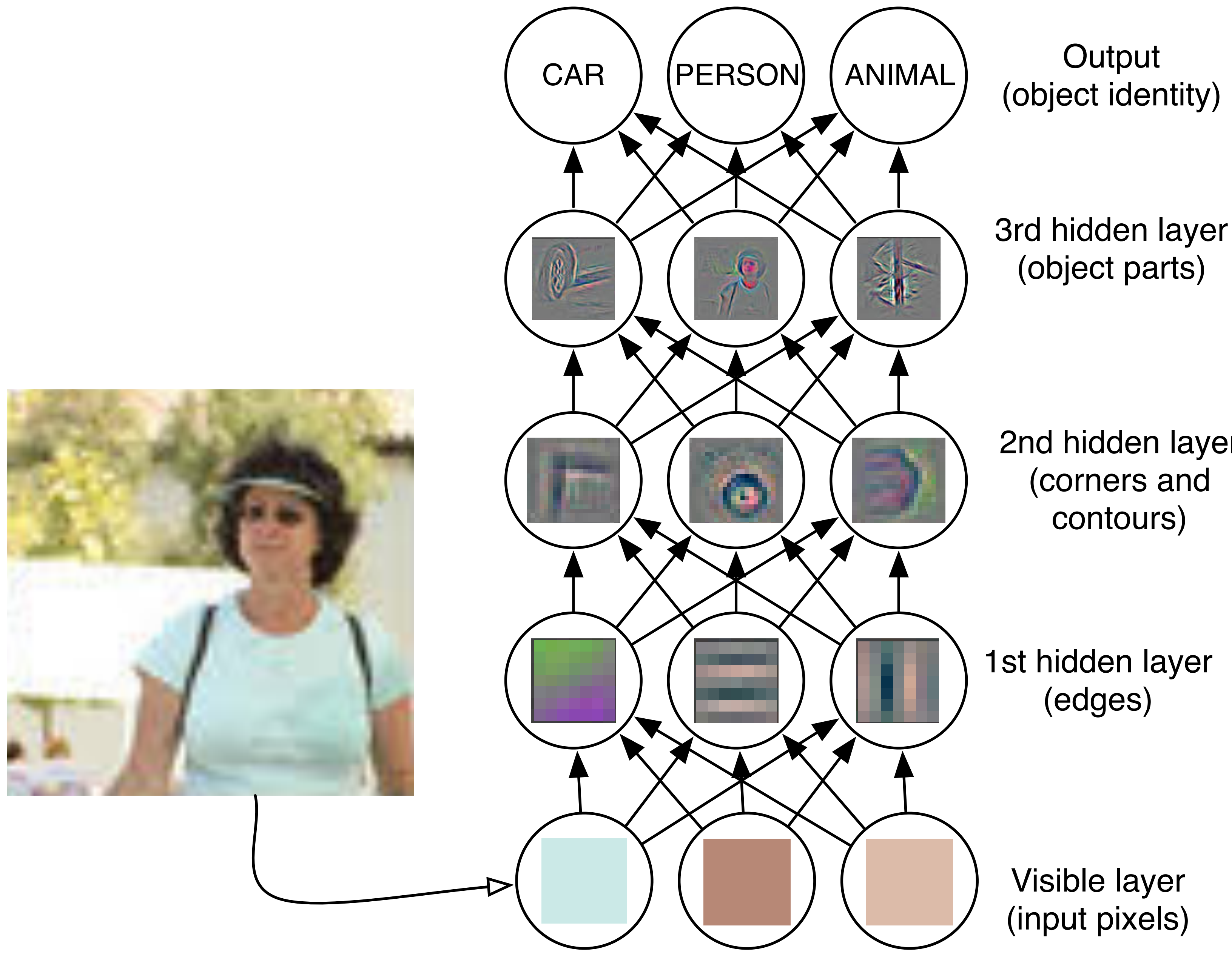
BIRD

- Data is features X and targets y
- Goal: learn to map x to y
- Classification: discrete y
- Regression: continuous y

Unsupervised learning for feature learning



Deep learning



(Images by Matt Zeiler)

Spike-and-Slab Sparse Coding

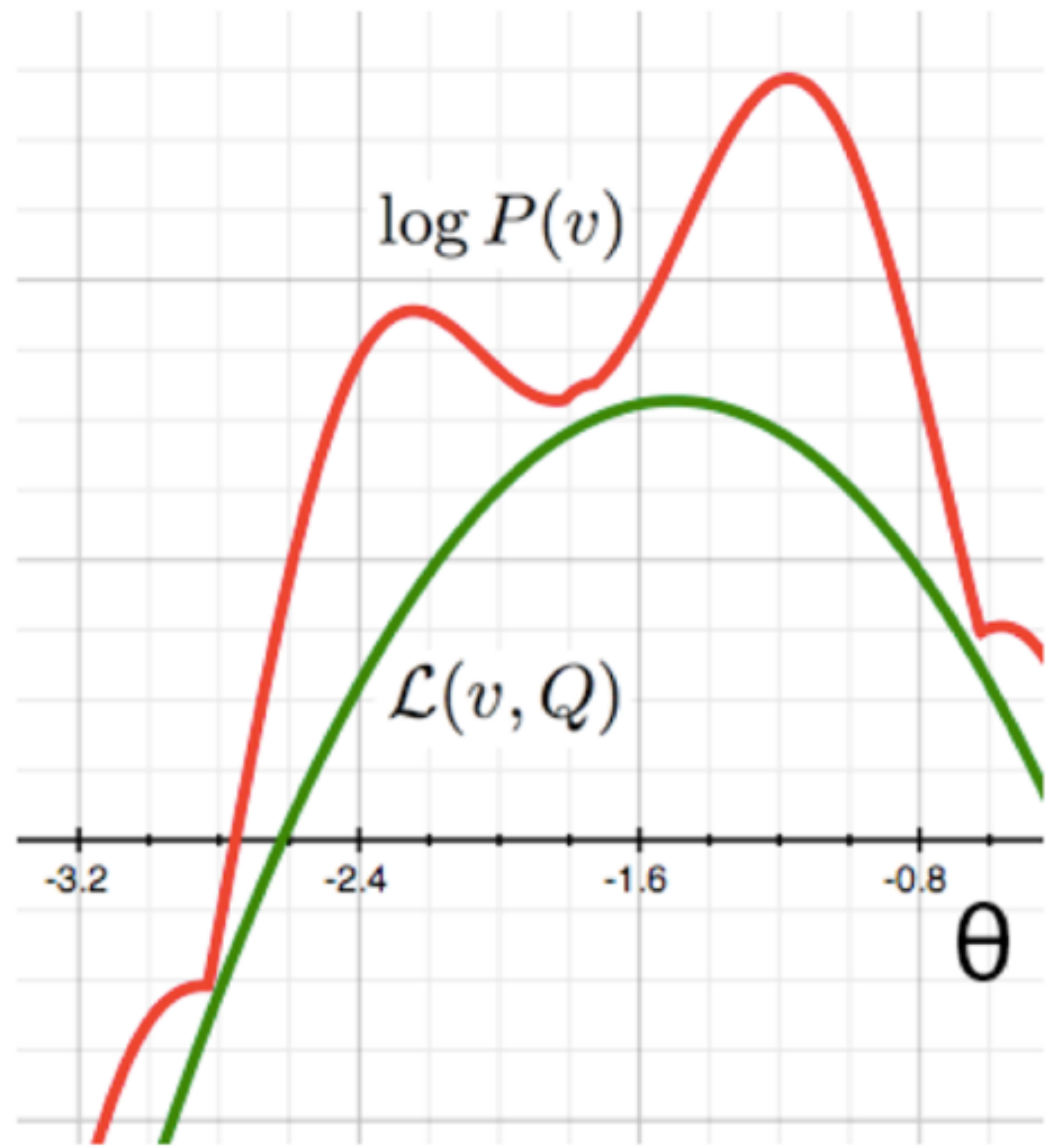
- Co-authors: Aaron Courville and Yoshua Bengio
- Motivated by Adam Coates' work on feature learning and feature extraction
- Faster form of variational inference
- Component of a deep model

Motivating CIFAR-10 results

- Validation set accuracy (Coates and Ng 2011):
  - RBM features encoded with RBM: 74.1%
  - RBM features encoded with sparse coding: 76.7%
- Test set accuracy (Courville et al 2011):
  - ssRBM: 76.7%

Variational learning

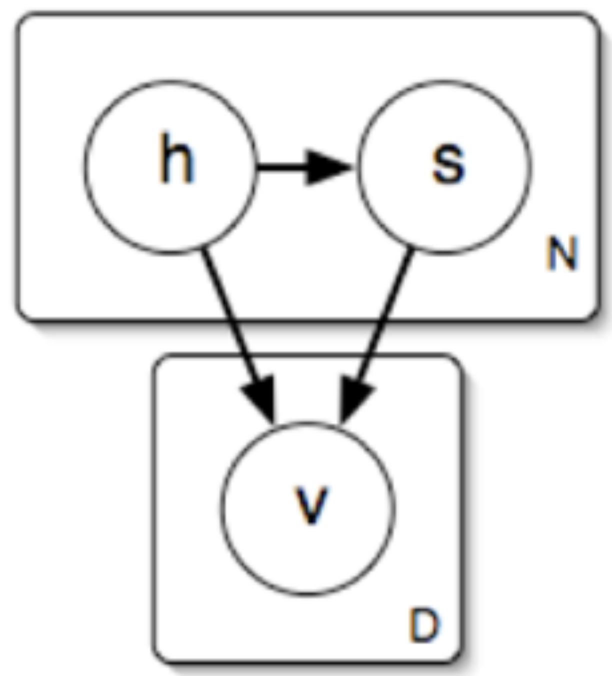
- Approximate intractable P(h|v) with tractable Q(h)
- Use Q to construct a lower bound on the log likelihood



Variational inference

- $\mathcal{L}(v, Q) = \mathbb{E}_{h \sim Q}[\log P(v, h)] + H_Q(h)$
- Maximizing this corresponds to minimizing
$$D_{KL}(Q(h) \| P(h \mid v))$$
- Often requires both analytical and iterative optimization

The Spike-and-Slab Sparse Coding (S3C) Generative Model

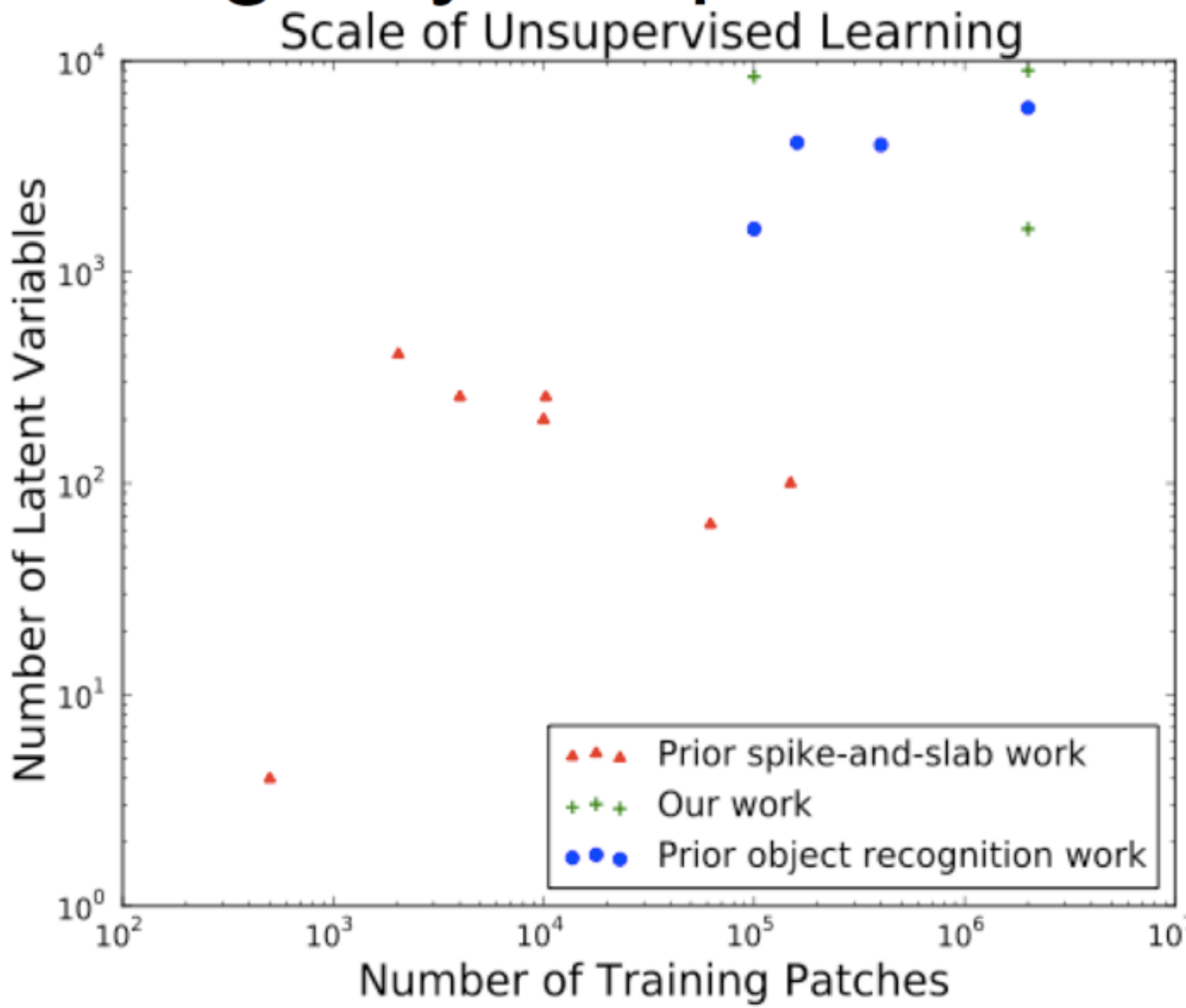


$$p(h_i = 1) = \sigma(b_i)$$

$$p(s_i \mid h_i) = \mathcal{N}(s_i \mid h_i \mu_i, \alpha_{ii}^{-1})$$

$$p(v_d \mid s, h) = \mathcal{N}(v_d \mid W_{d:}(h \circ s), \beta_{dd}^{-1})$$

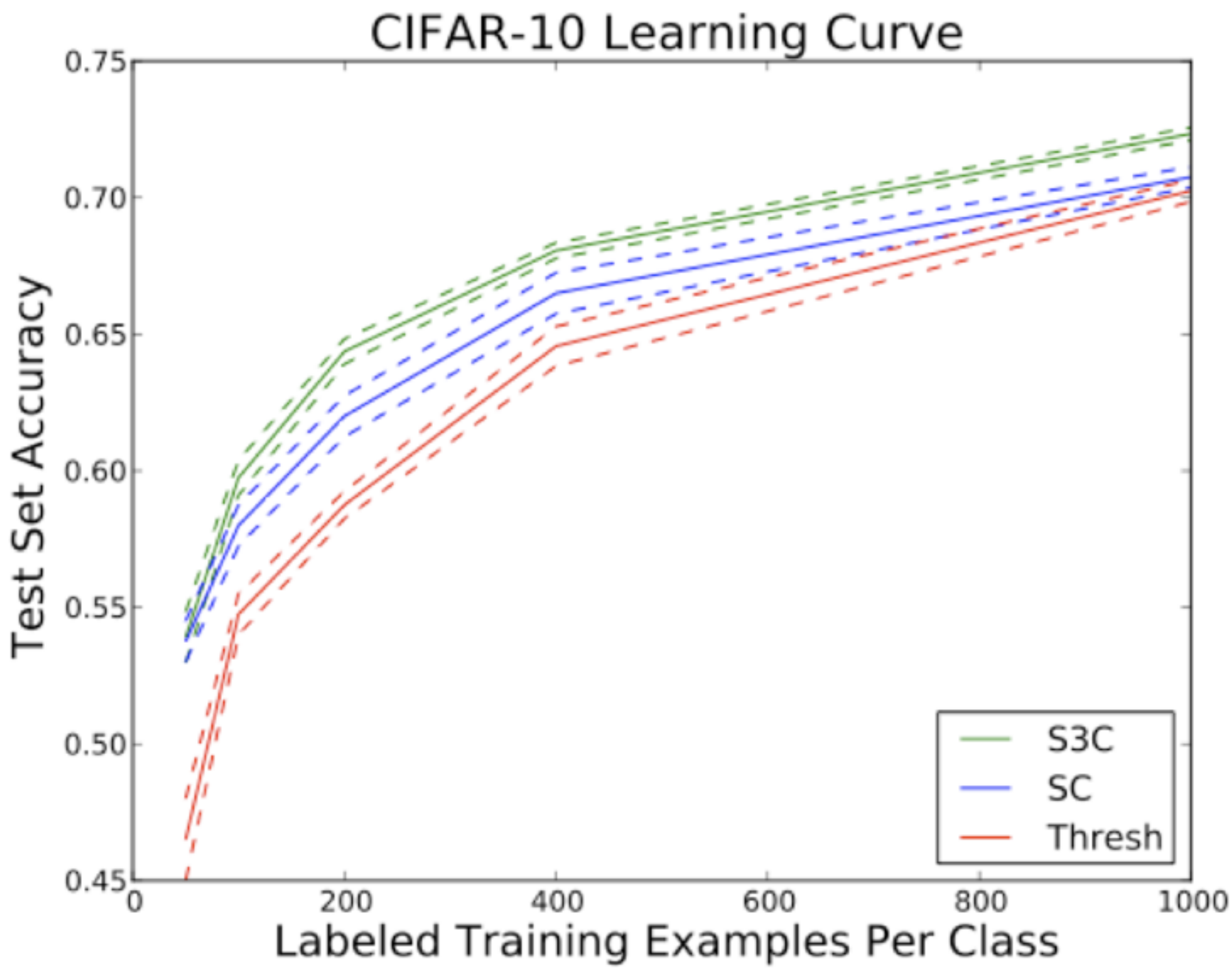
Scaling beyond previous work



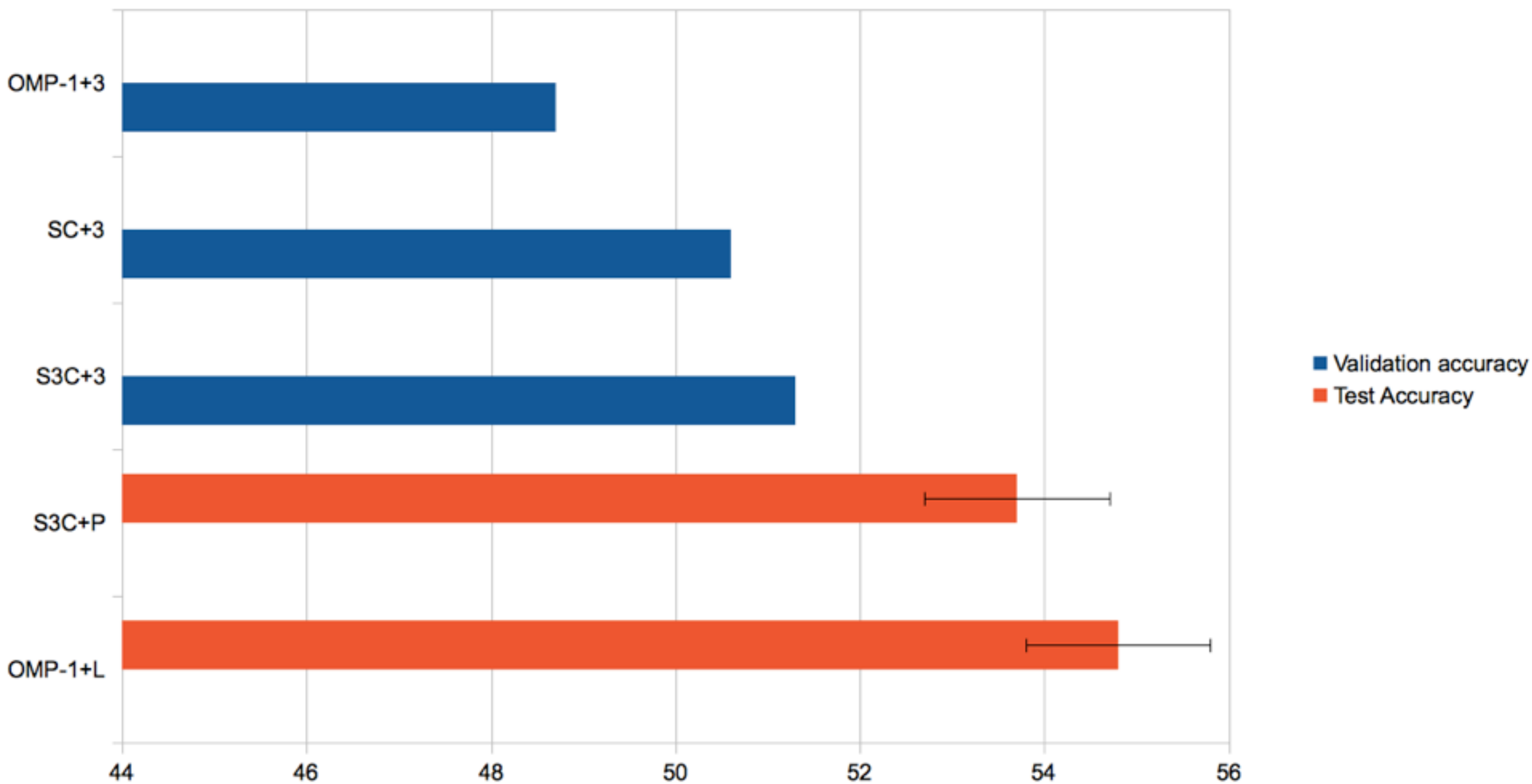
Spike-and-Slab work: Mohammed et al, 2011; Zhou et al., 2009; Garrigues and Olsahusen, 2008; Lücke and Sheik, 2011; Titsias and Lázaro-Gredilla, 2011

Scaling to more classes and fewer labeled examples per class

- Clic



CIFAR-100 Results



OMP-1+L: Jia and Huang 2011

Transfer Learning Challenge

Labeled training set:

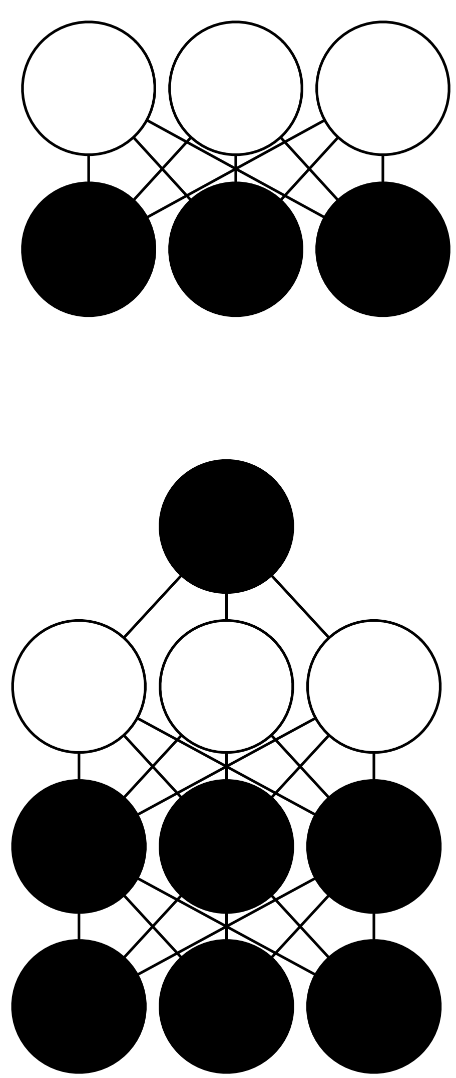
Self-taught learning with S3C won the challenge with a test set accuracy of 48.26%

Multi-Prediction deep Boltzmann machines

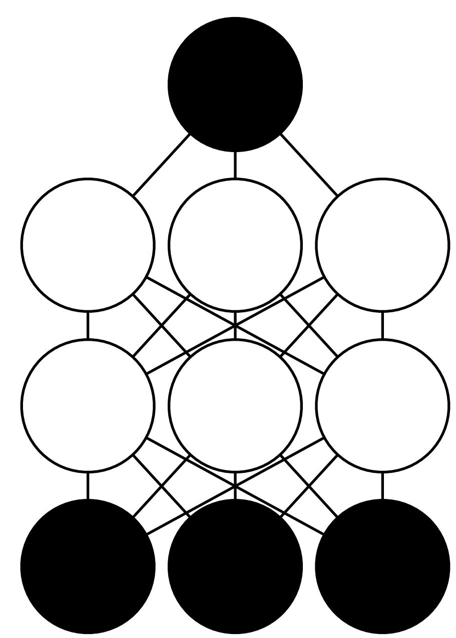
- Co-authors: Mehdi Mirza, Aaron Courville, Yoshua Bengio
- Simplified training procedure for deep Boltzmann machines
- Improved accuracy of approximate inference

Typical DBM Training

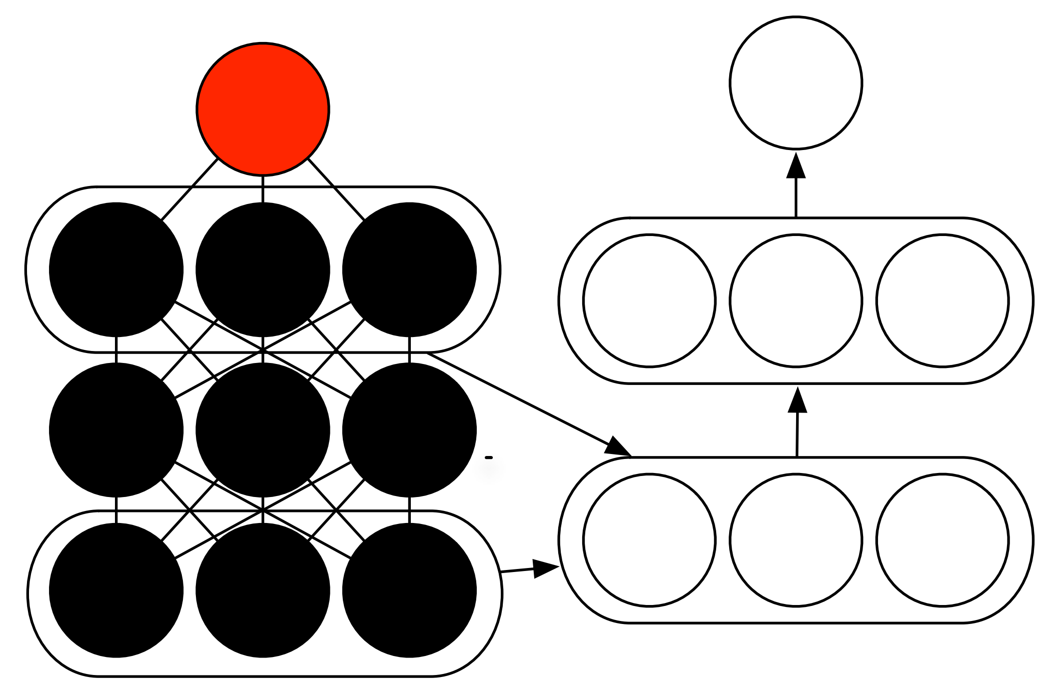
1. Greedy layerwise pretraining



2. Joint generative training



3. Discriminative fine-tuning



Salakhutdinov and Hinton, 2009

Sampling-based approximations

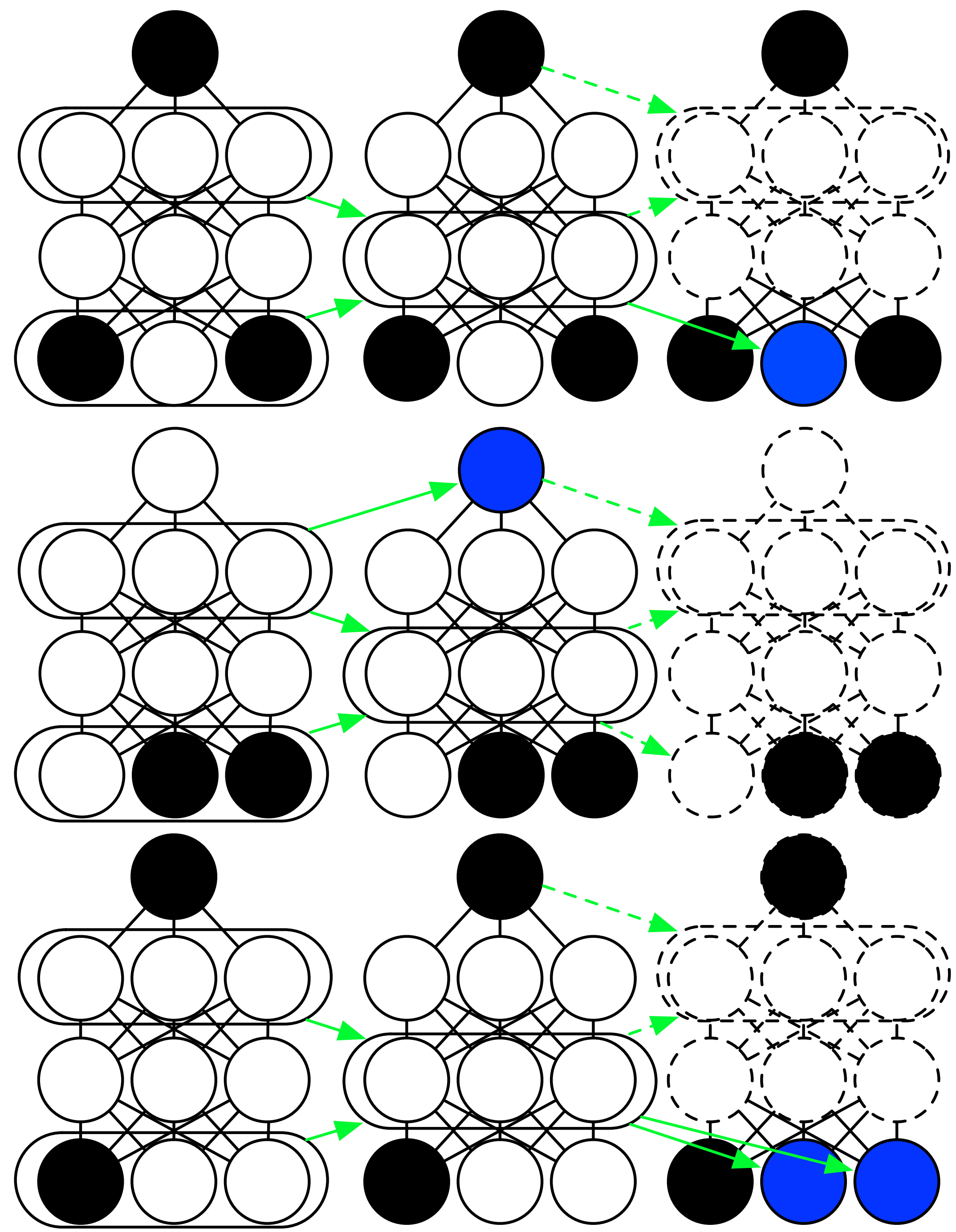
- p(x;θ)=exp(-E(x;θ))/Z(θ)
- What if Z(θ) is intractable?
- $\frac{\partial}{\partial \theta_i} \log Z = -\mathbb{E}_x \left[\frac{\partial}{\partial \theta_i} E(x;\theta) \right]$
- Approximate expectations via sampling
- CD-k: sample k steps from data points
- SML/PCD: sample continuously, use low learning rate

Simplify, simplify, simplify

	Classic approach	Goal
# models	#layers+2	1
# criteria	#layers+2	1
Classifier	Extra classifier model	Same unified probabilistic model

Multi-Prediction Training

Randomly sample different inference problems



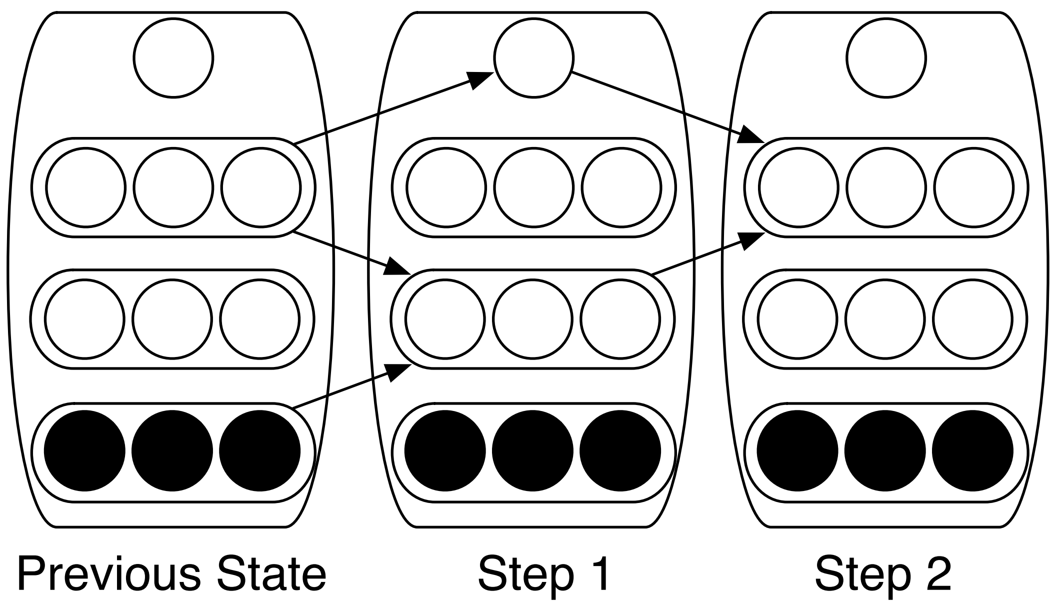
Backprop through the mean field inference graph

Benefits of Multi-Prediction Training

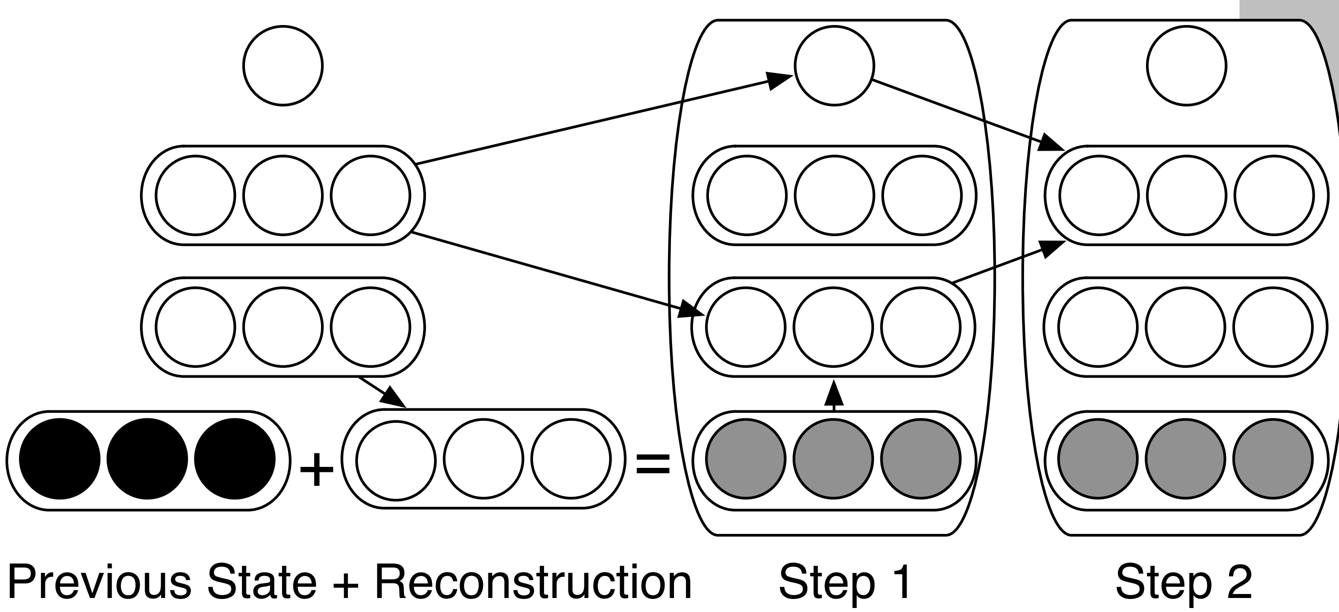
- Learning rate doesn't affect approximation accuracy
- Training compensates for approximate inference
  - Similar to Stoyanov et al 2011

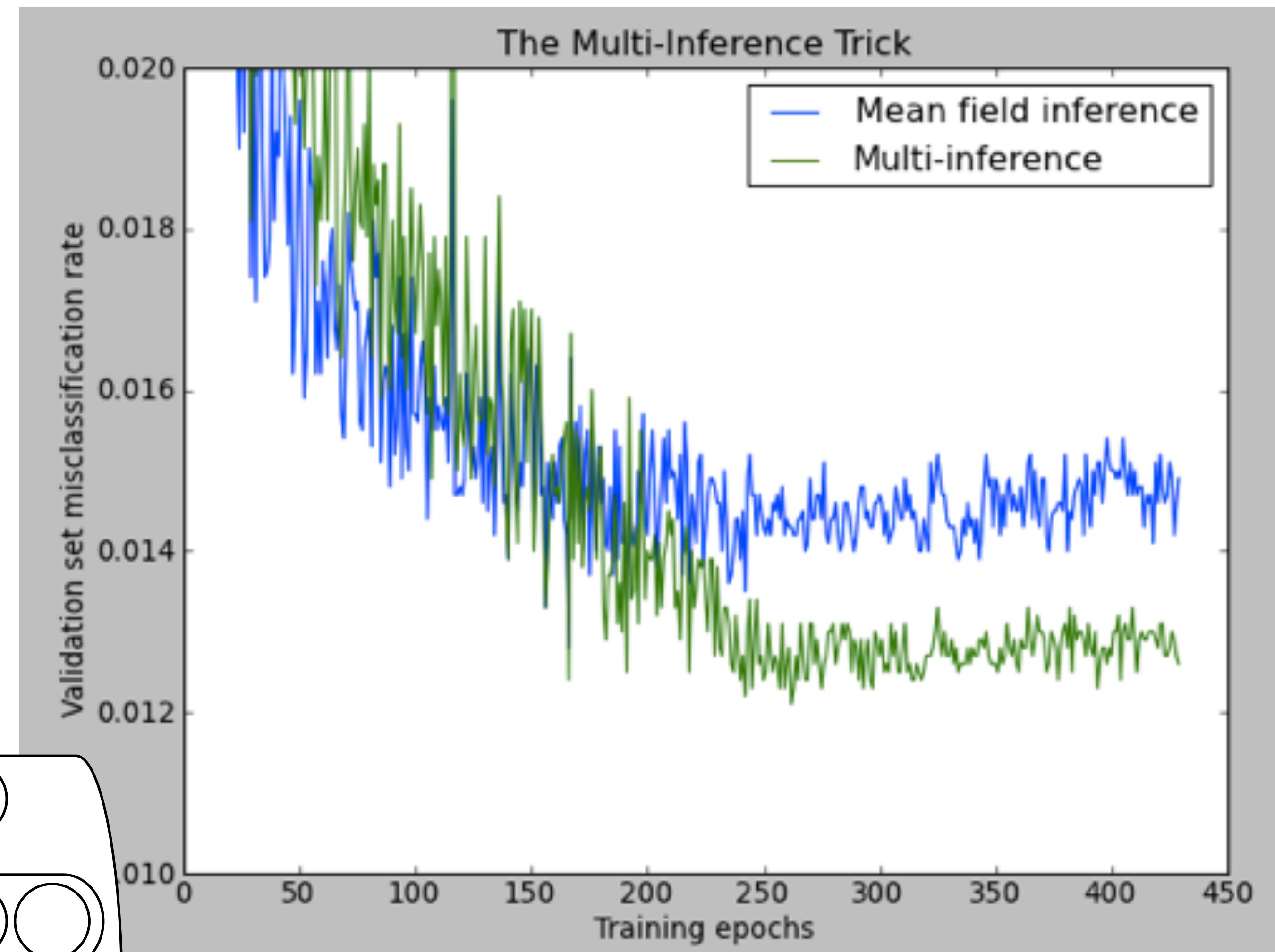
Multi-Inference Trick

Mean Field Iteration

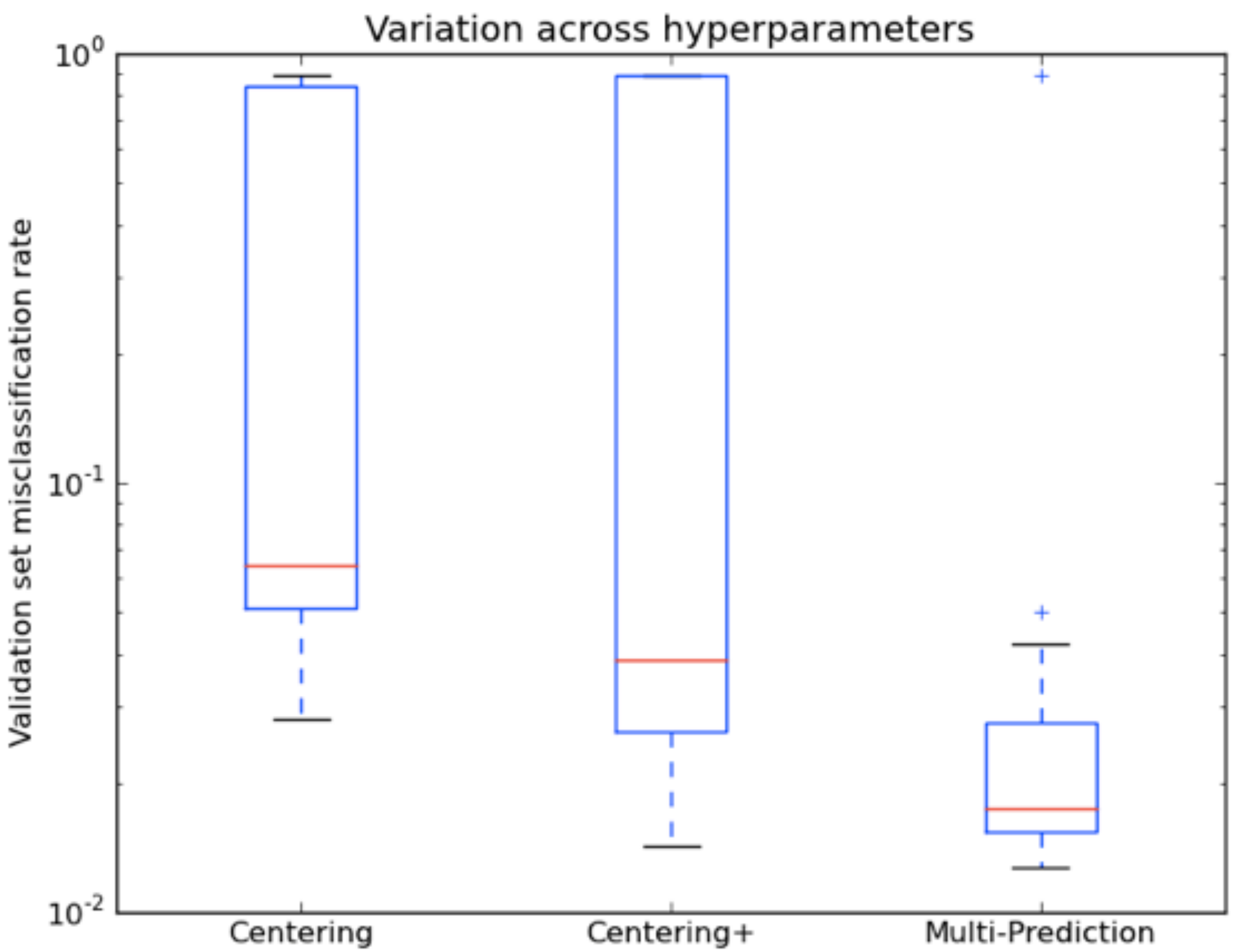


Multi-Inference Iteration





Results



Model	Test error with fine-tuning
S&H 2009*	0.95
Centered DBM	1.22
MP-DBM	0.99

Centering: Montavon and Müller, 2012

Multi-prediction, 2X hidden units, no fine-tuning*: 0.91

*Retrained using validation set.

Mission Accomplished

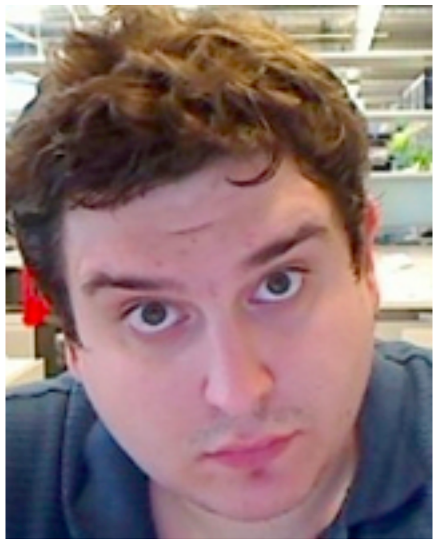
	Classic approach	Goal
# models	#layers+2	1 ✓
# criteria	#layers+2	1 ✓
Classifier	Extra classifier model	Same unified probabilistic model ✓

Maxout Networks

by Ian Goodfellow

Joint work with

David Warde-Farley　Mehdi Mirza　Aaron Courville　Yoshua Bengio

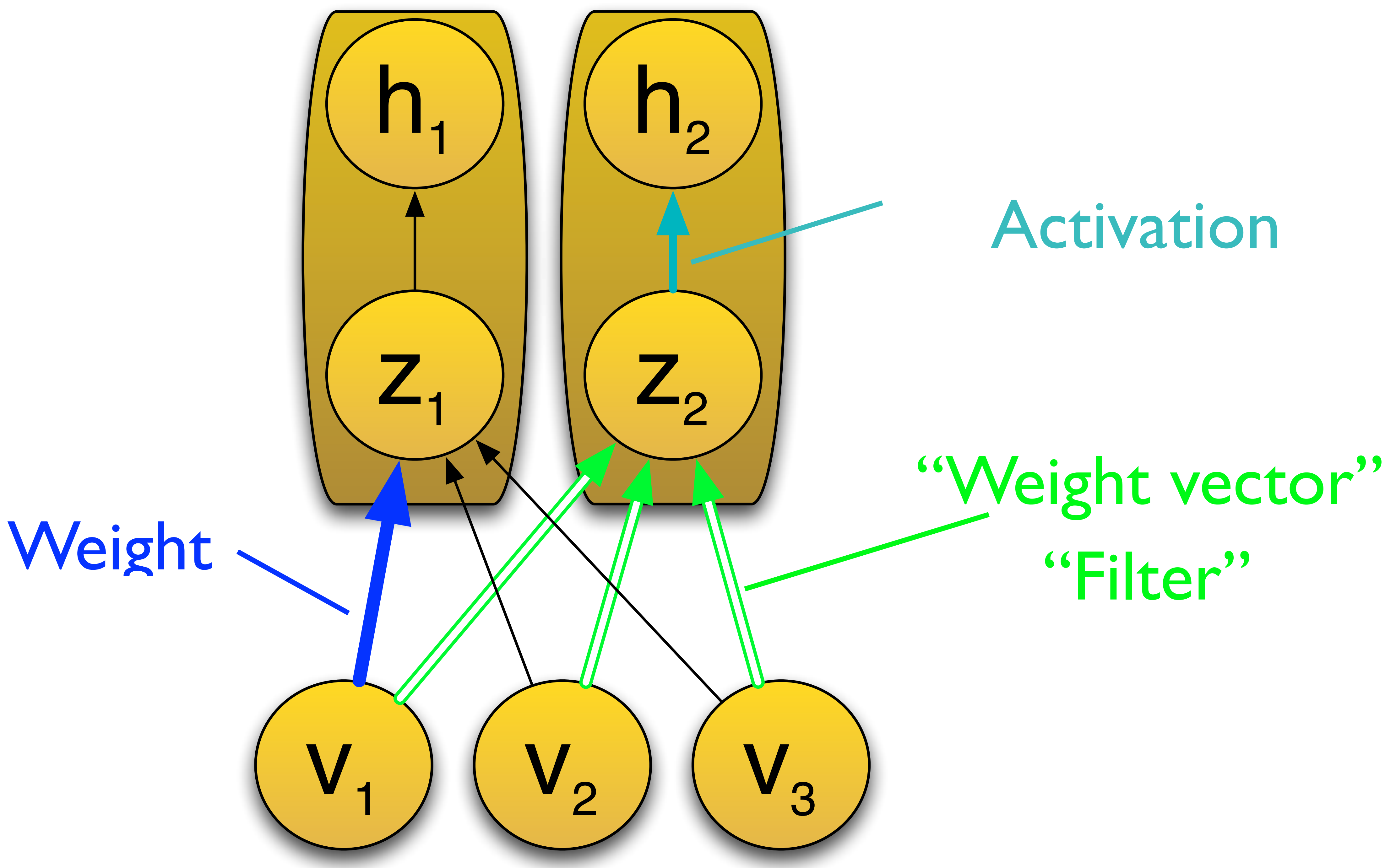


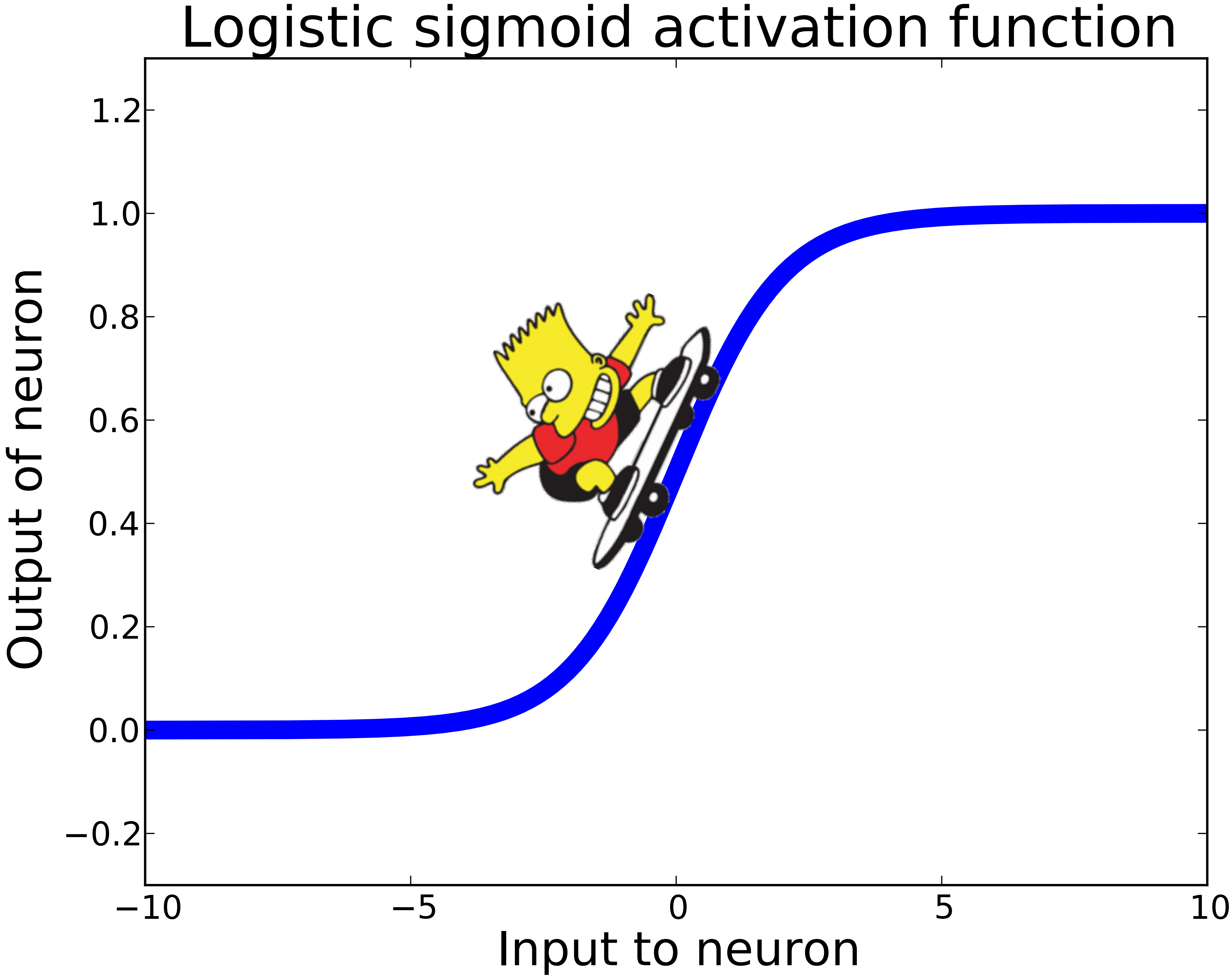

with acknowledgments to



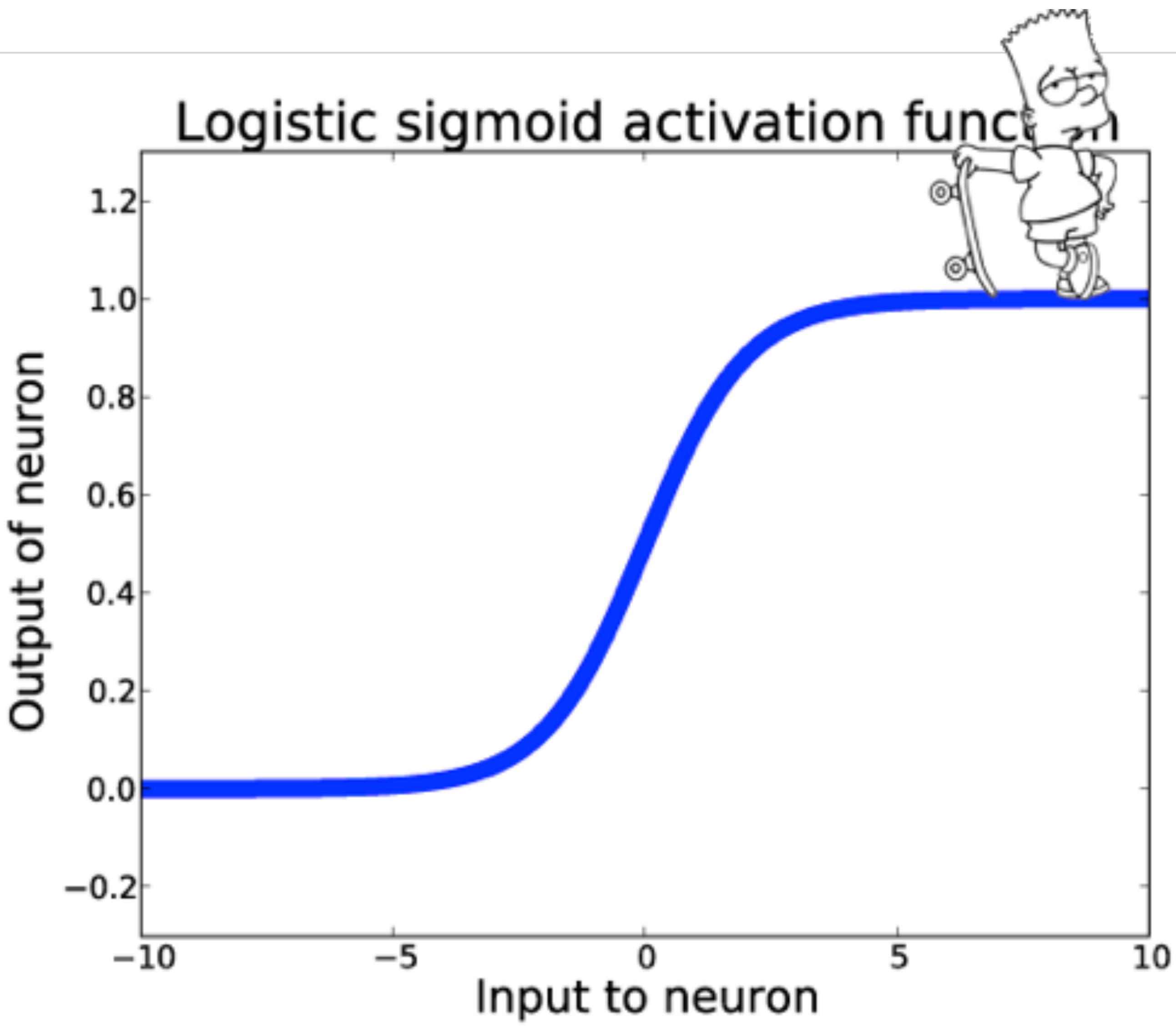
Frédéric Bastien　Yann Dauphin　Pascal Lamblin

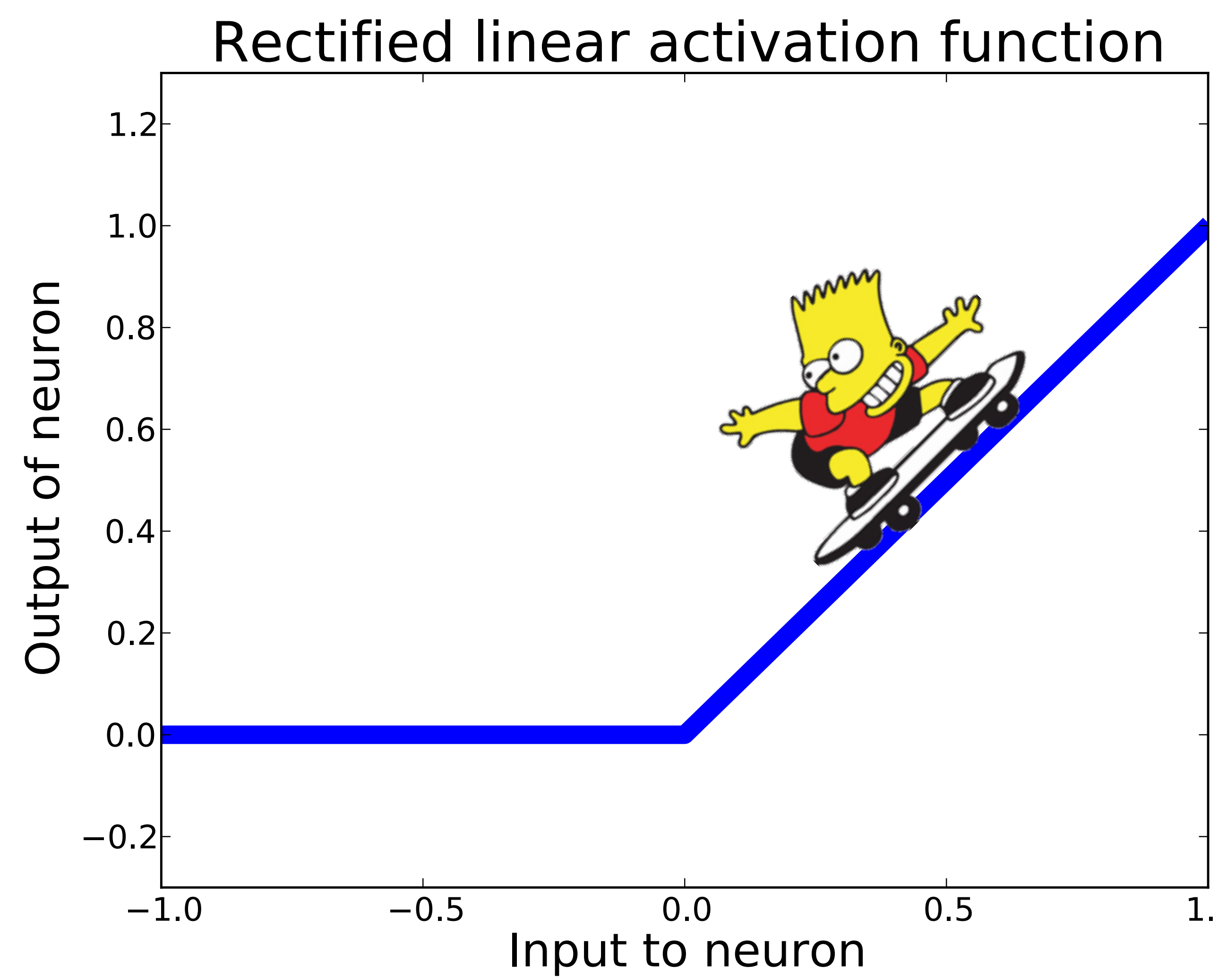
Traditional activation functions



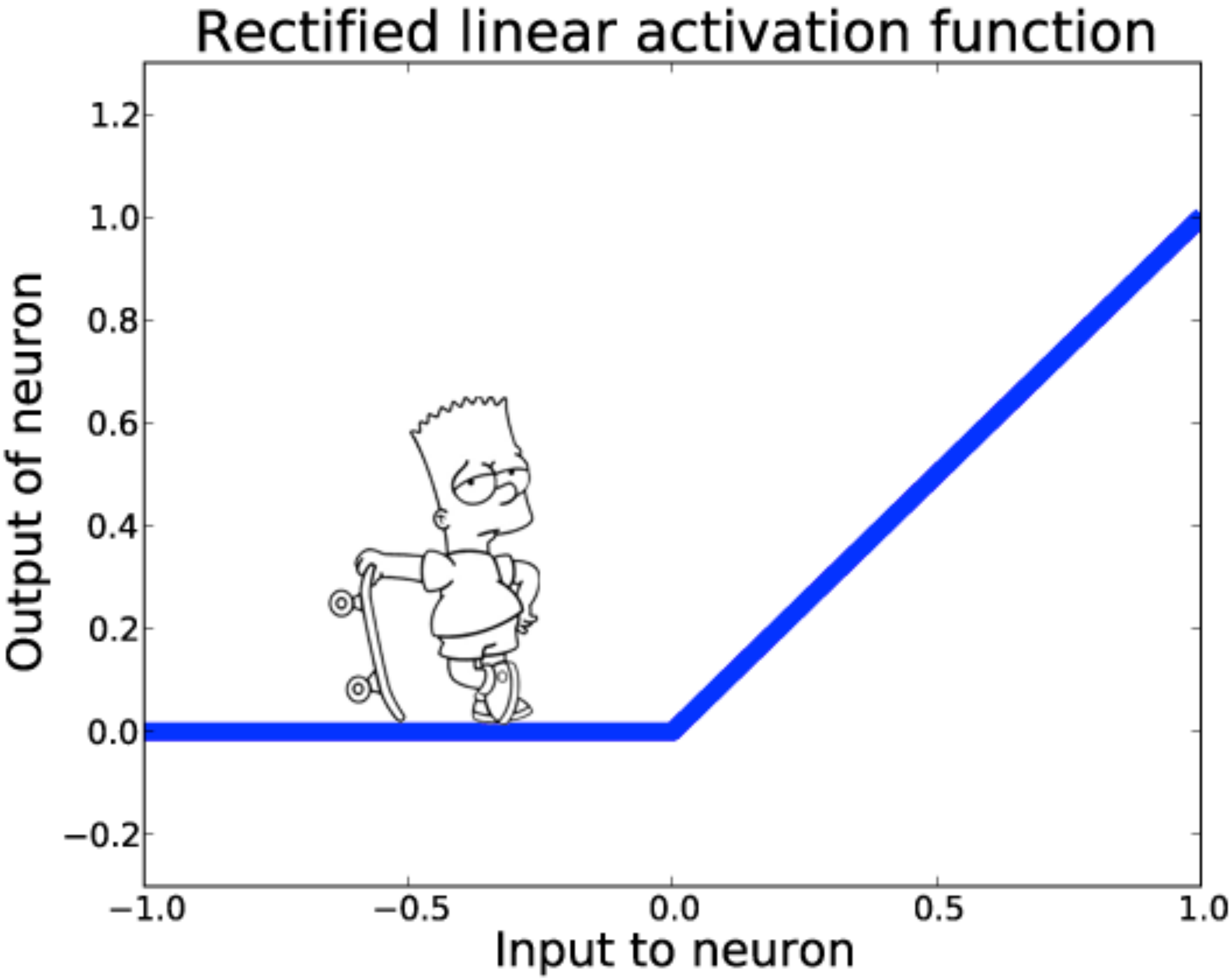

Logistic sigmoid activation function
Output of neuron
Input to neuron
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−10
−5
0
5
10

The vanishing gradient problem

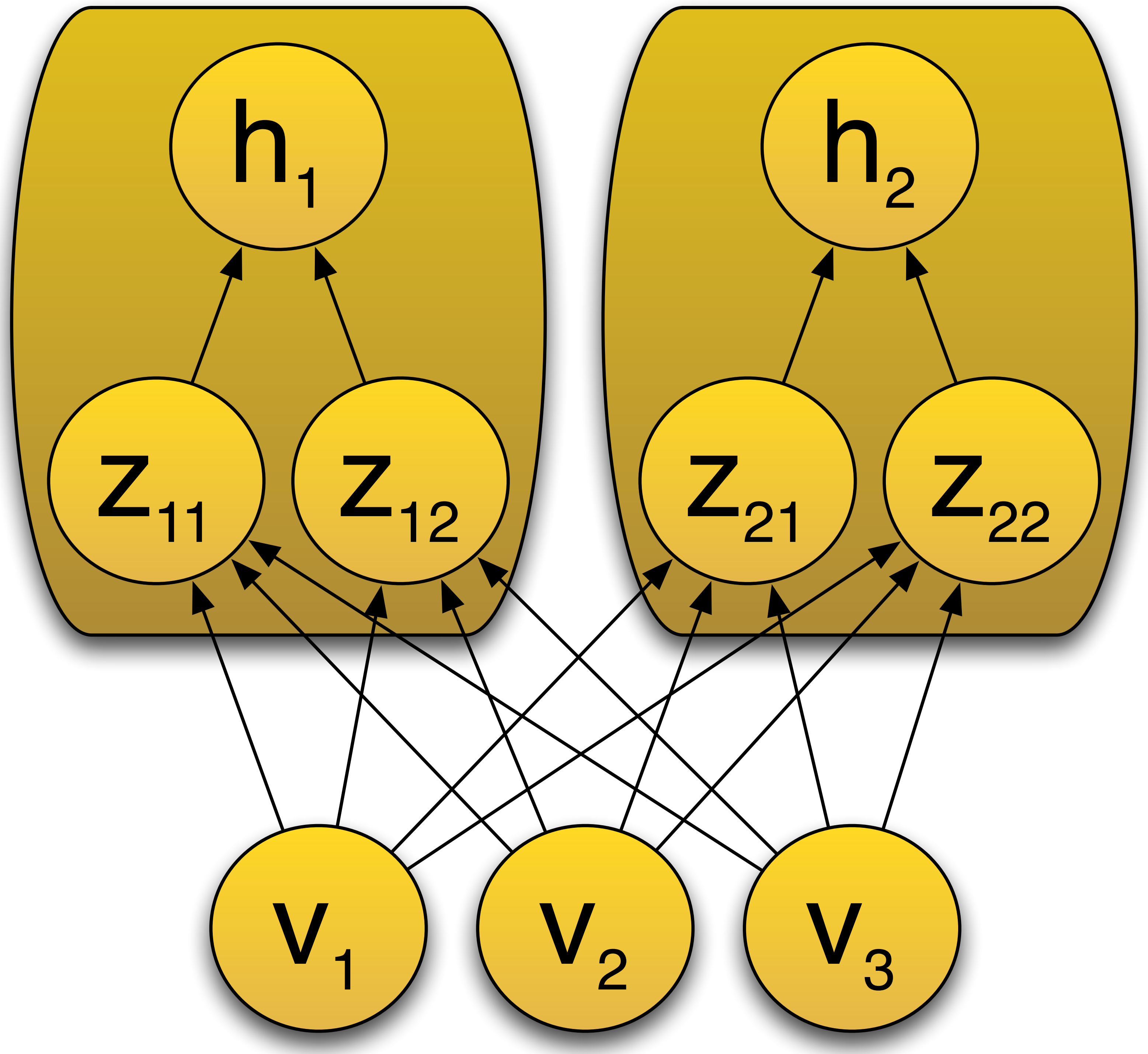




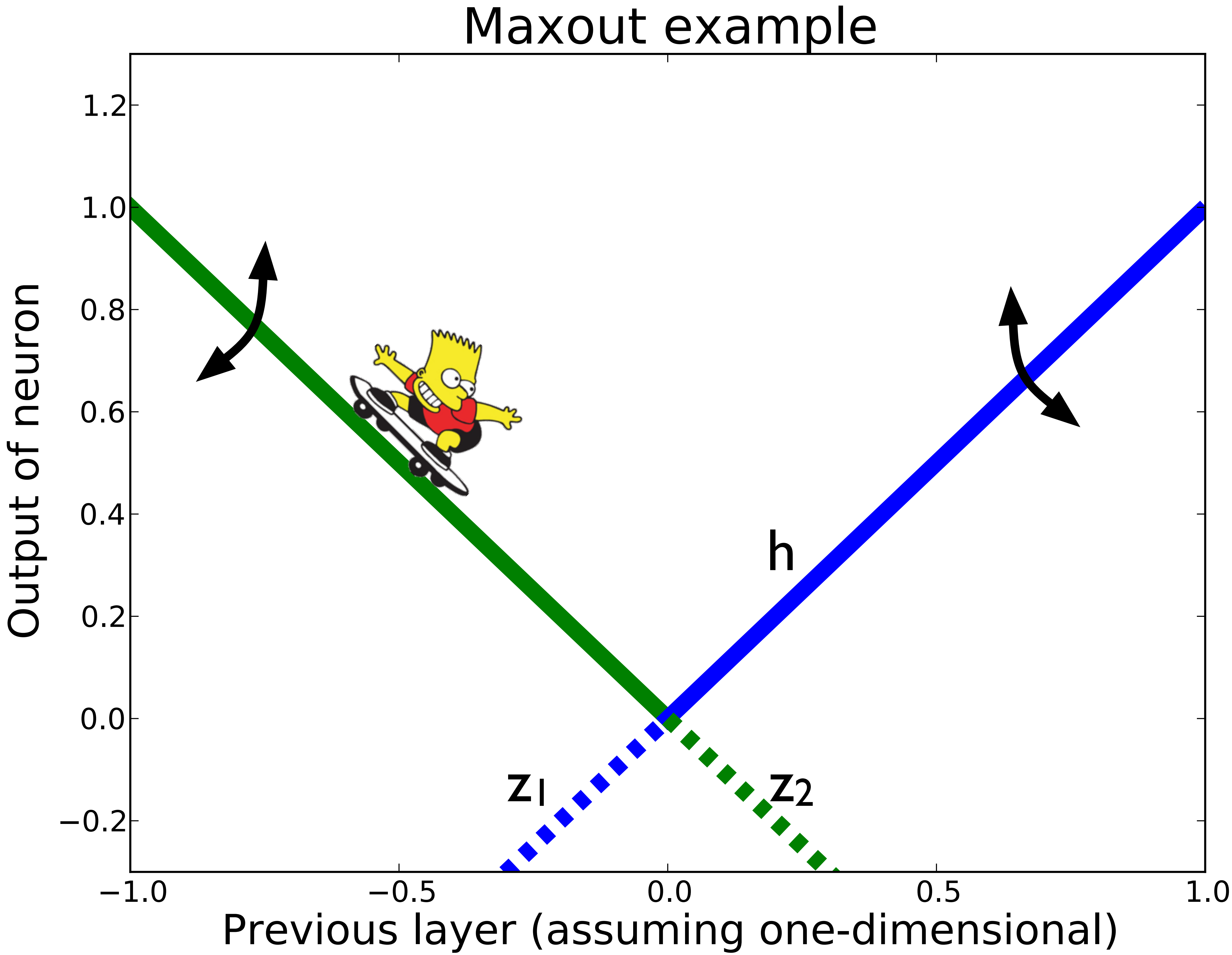
Uh-oh


Rectified linear activation function
Output of neuron
Input to neuron
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−1.0
−0.5
0.0
0.5
1.0

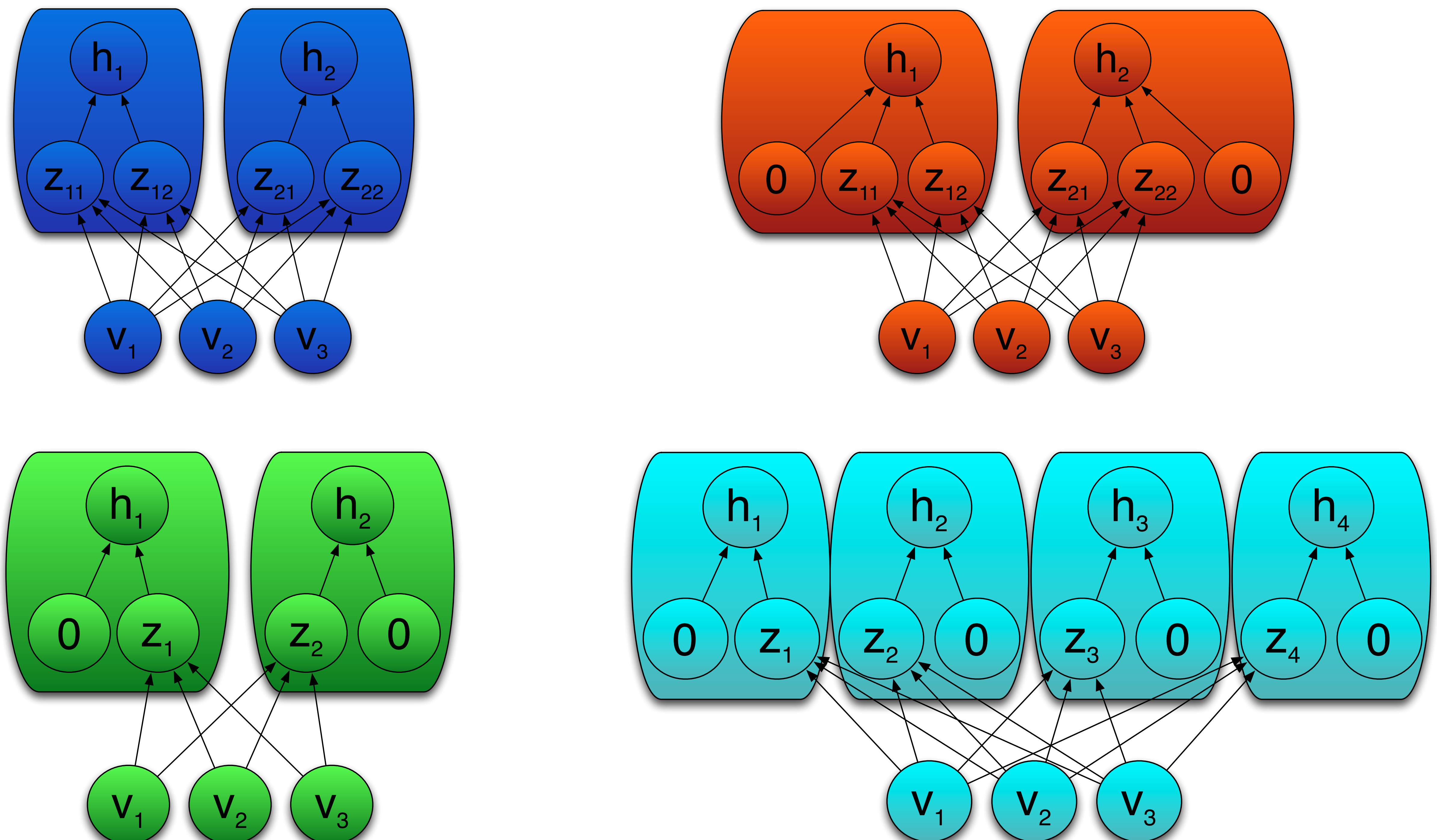
Maxout



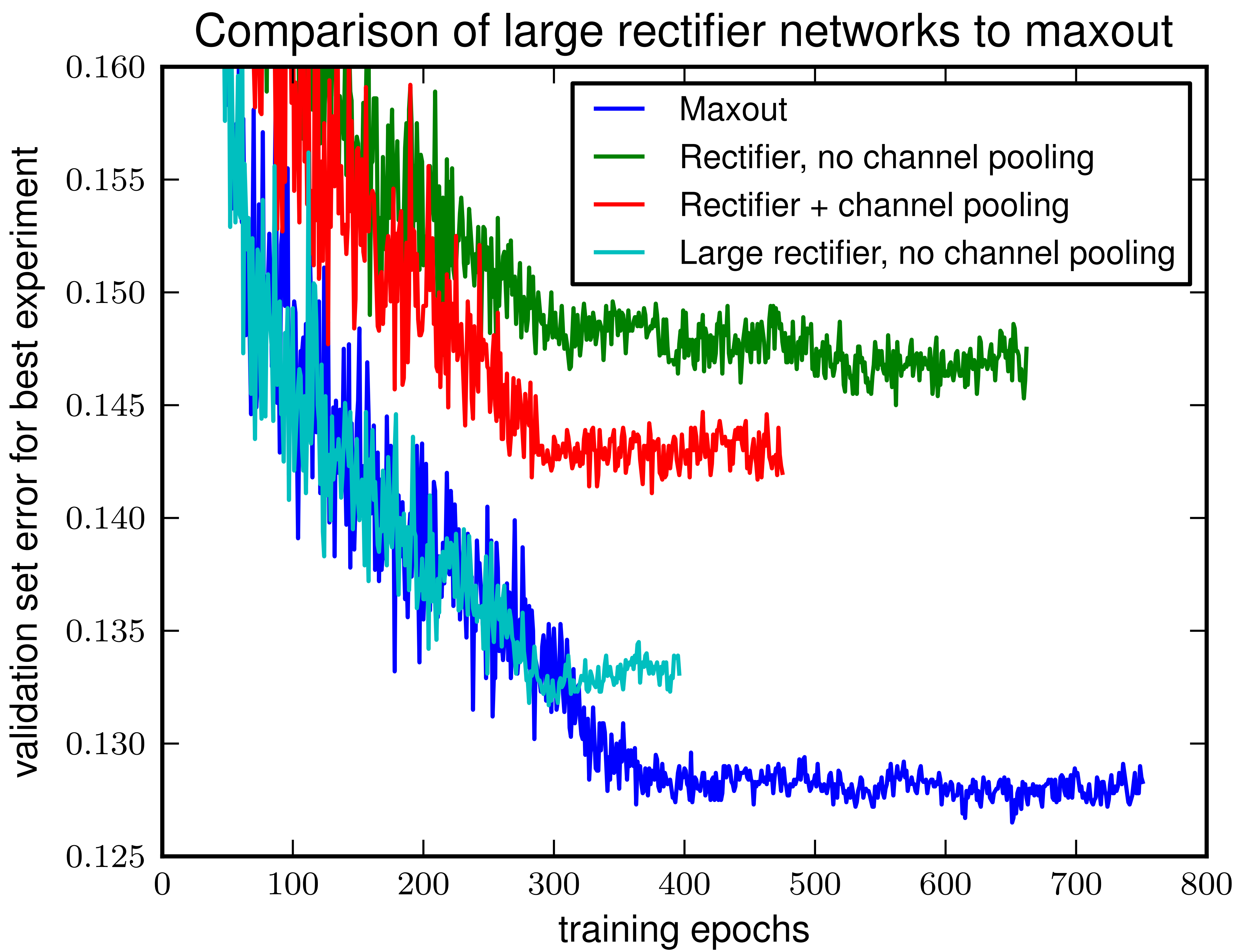
$$h_i = \max_j z_{ij}$$


Maxout example
Output of neuron
Previous layer (assuming one-dimensional)
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−1.0
−0.5
0.0
0.5
1.0
h
z1
z2

Comparing maxout to rectifiers



Effectiveness of pooling



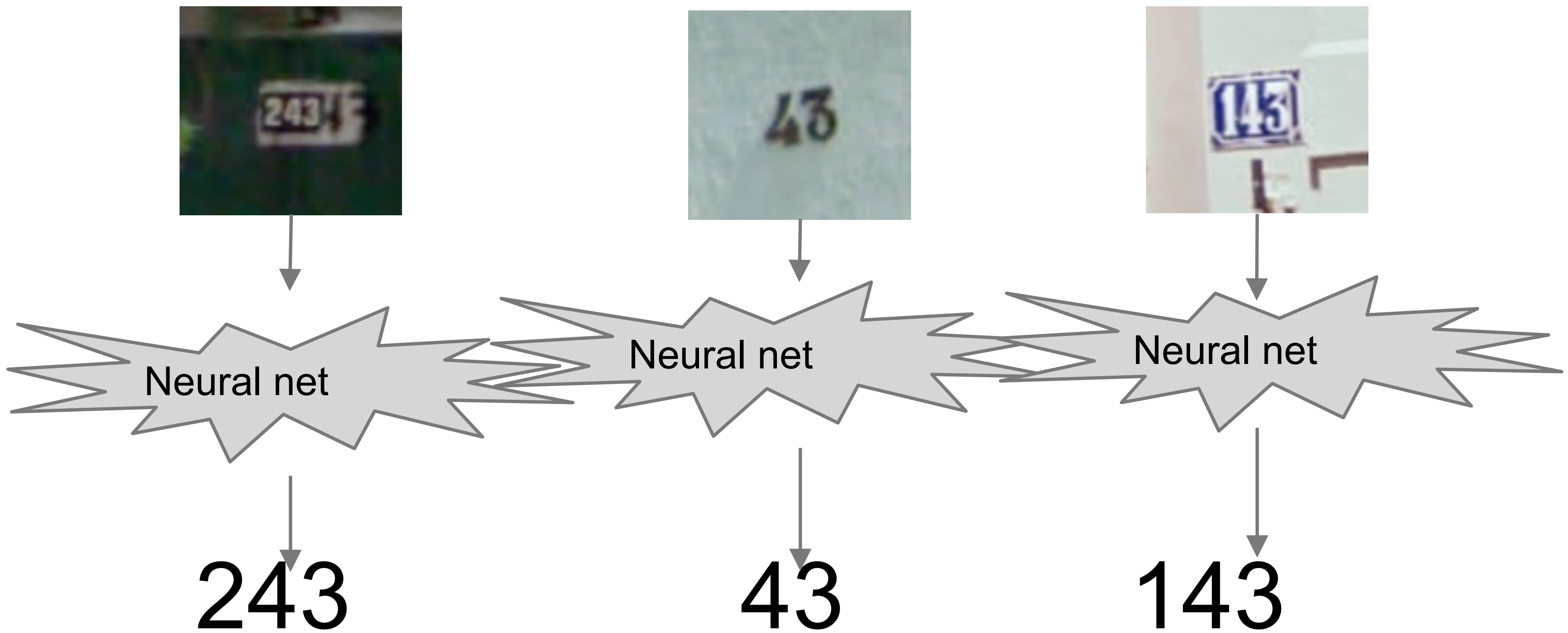
Applications of maxout

- Speech: Miao et al 2013, Cai et al 2013, Zhang et al 2014, Swietojanski et al 2014
- Multiplayer game matchmaking: Laufer et al, 2013
- Text detection: Jaderberg et al 2014
- Text transcription: Alsharif and Pineau, 2013
- Simplifying optimization: Gulcehre 2013
- Recurrent networks: Pascanu 2014
- Whale call detection: Smirnov 2013
- Black-box classification: Xie et al, 2013

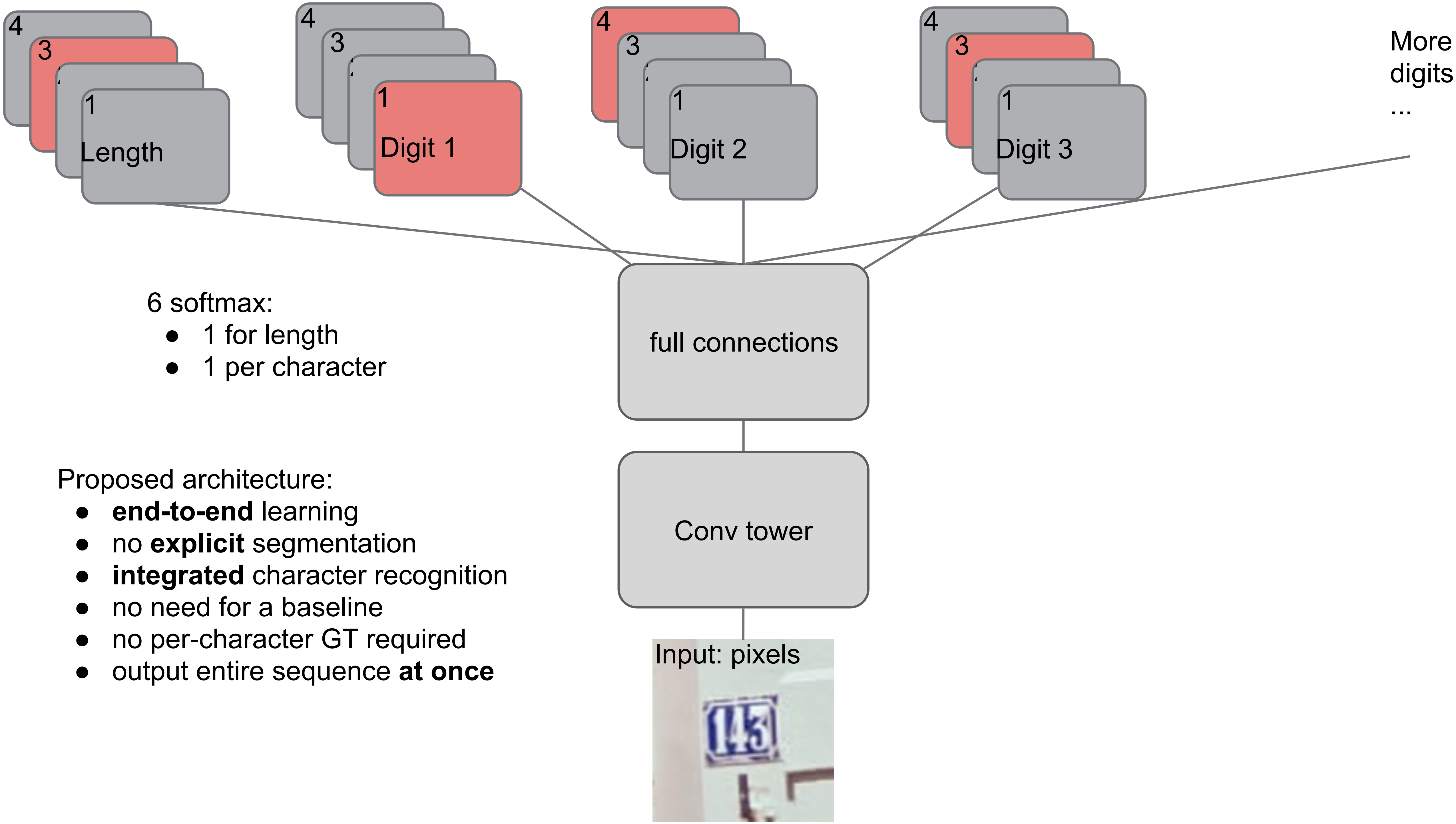
Street number transcription

- Co-authors: Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, Vinay Shet
- Use convolutional networks to read address numbers from Street View Images
- Automated transcription of over 100 million real address numbers

Want



Architecture



6 softmax:

- 1 for length
- 1 per character

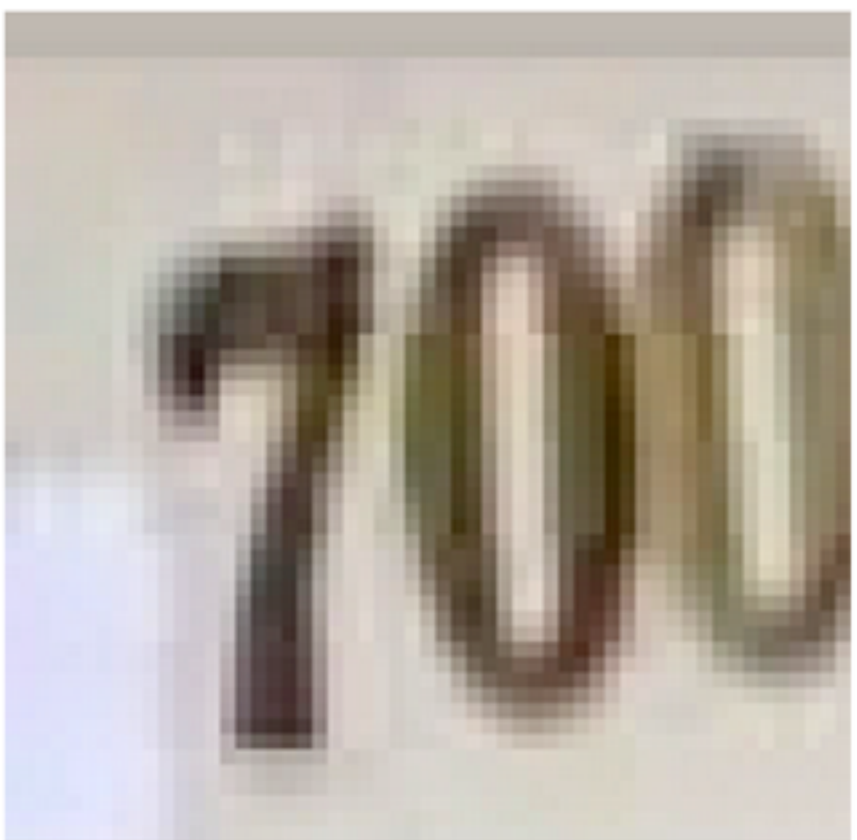
Proposed architecture:

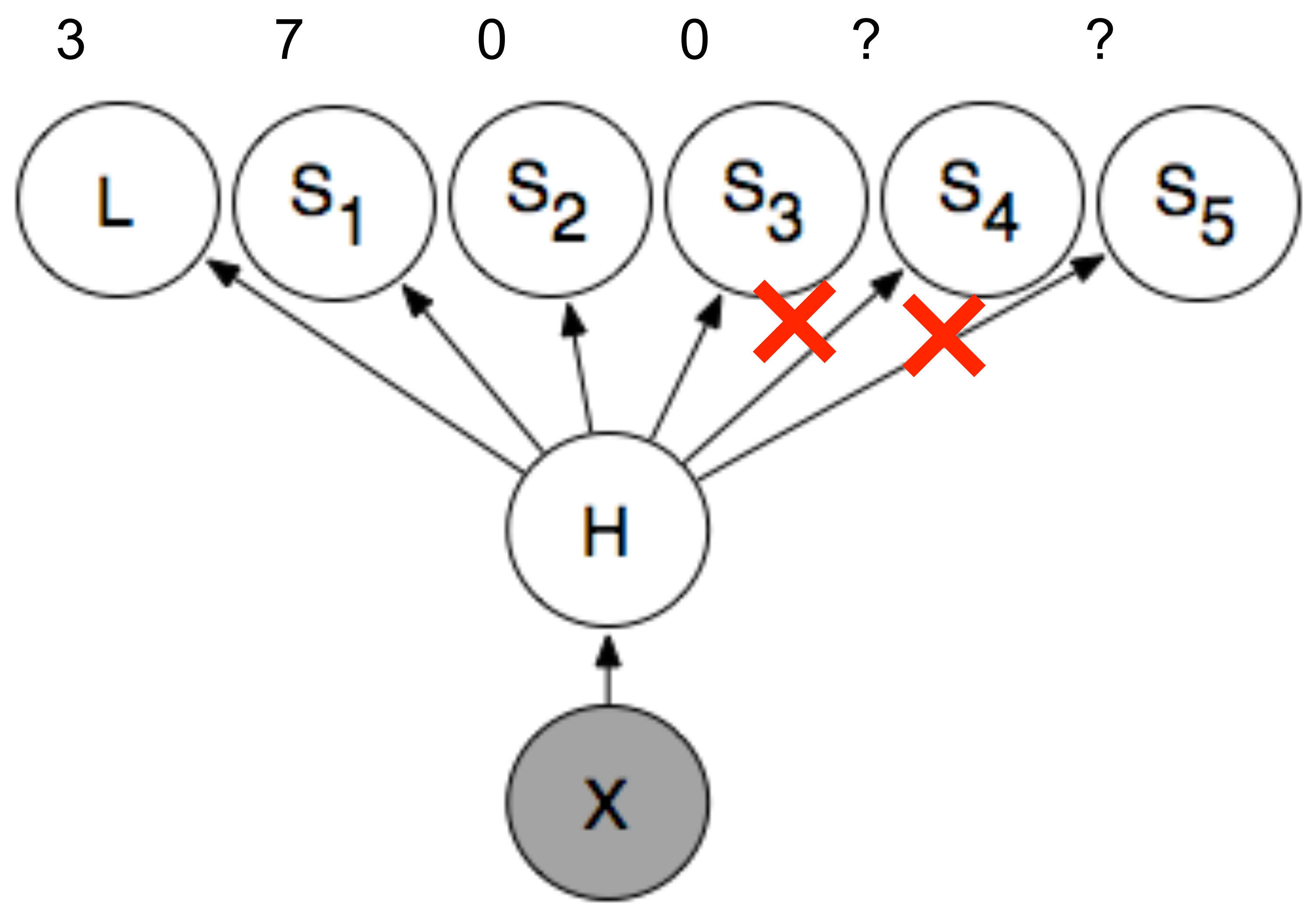
- **end-to-end** learning
- no **explicit** segmentation
- **integrated** character recognition
- no need for a baseline
- no per-character GT required
- output entire sequence **at once**

Training

Log likelihood:

$$\log P(L = l \mid X) + \sum_{i=1}^{L} \log P(S_i = s_i \mid X)$$





MAP sequence inference

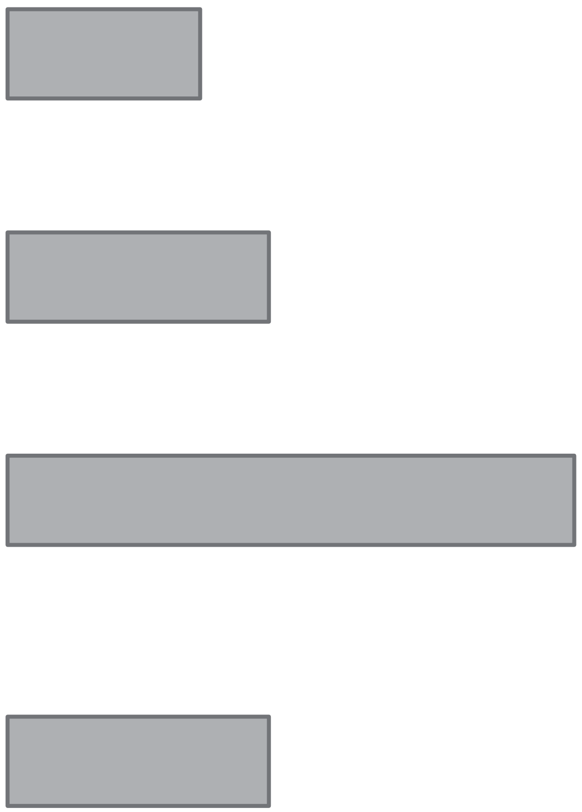
$$\operatorname{argmax}_{L, S_1, \ldots, S_L} \log P(S \mid X)$$

$\log P(L=1) + \log P(S_1 = \text{“1”})$

$\log P(L=2) + \log P(S_{1:2} = \text{“17”})$

$\log P(L=3) + \log P(S_{1:3} = \text{“175”})$

$\log P(L=4) + \log P(S_{1:4} = \text{“1751”})$



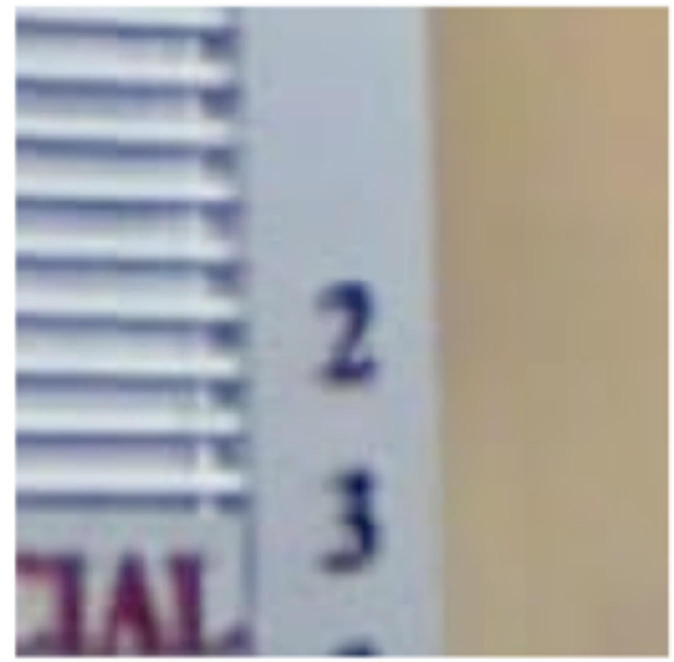
Accuracy

	Coverage@ human accuracy (98%)	Accuracy	Per Character Accuracy	Per Character Accuracy (Prev. state of the Art)
Public SVHN	95.6%	96%	97.8%	97.7%
Private Dataset	89%	91%		

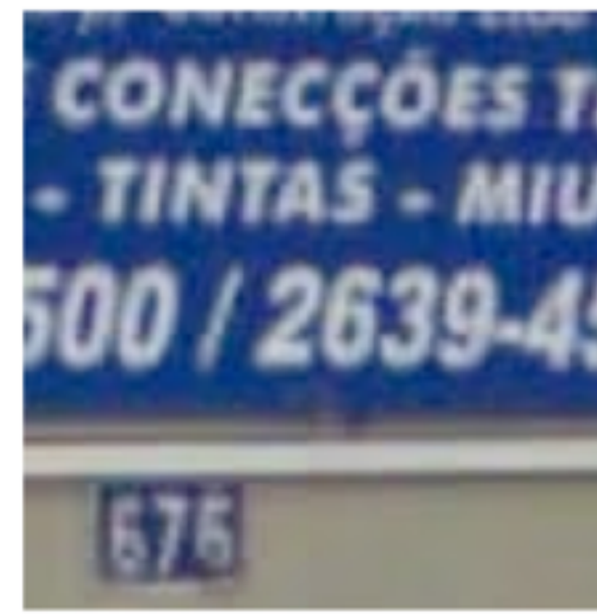
Example failures

1180 vs. 1780

1844 vs. 184

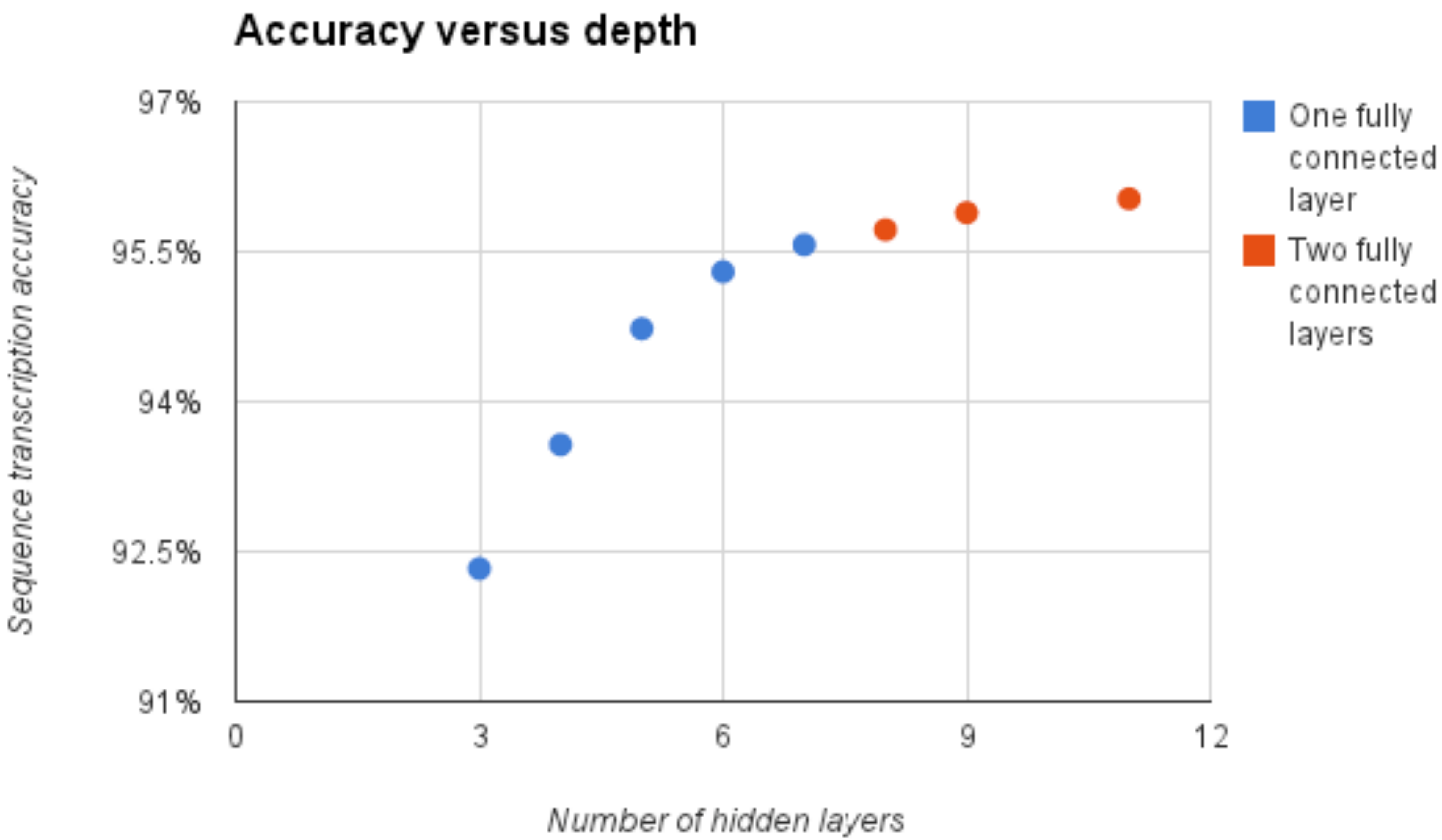


2 vs 239

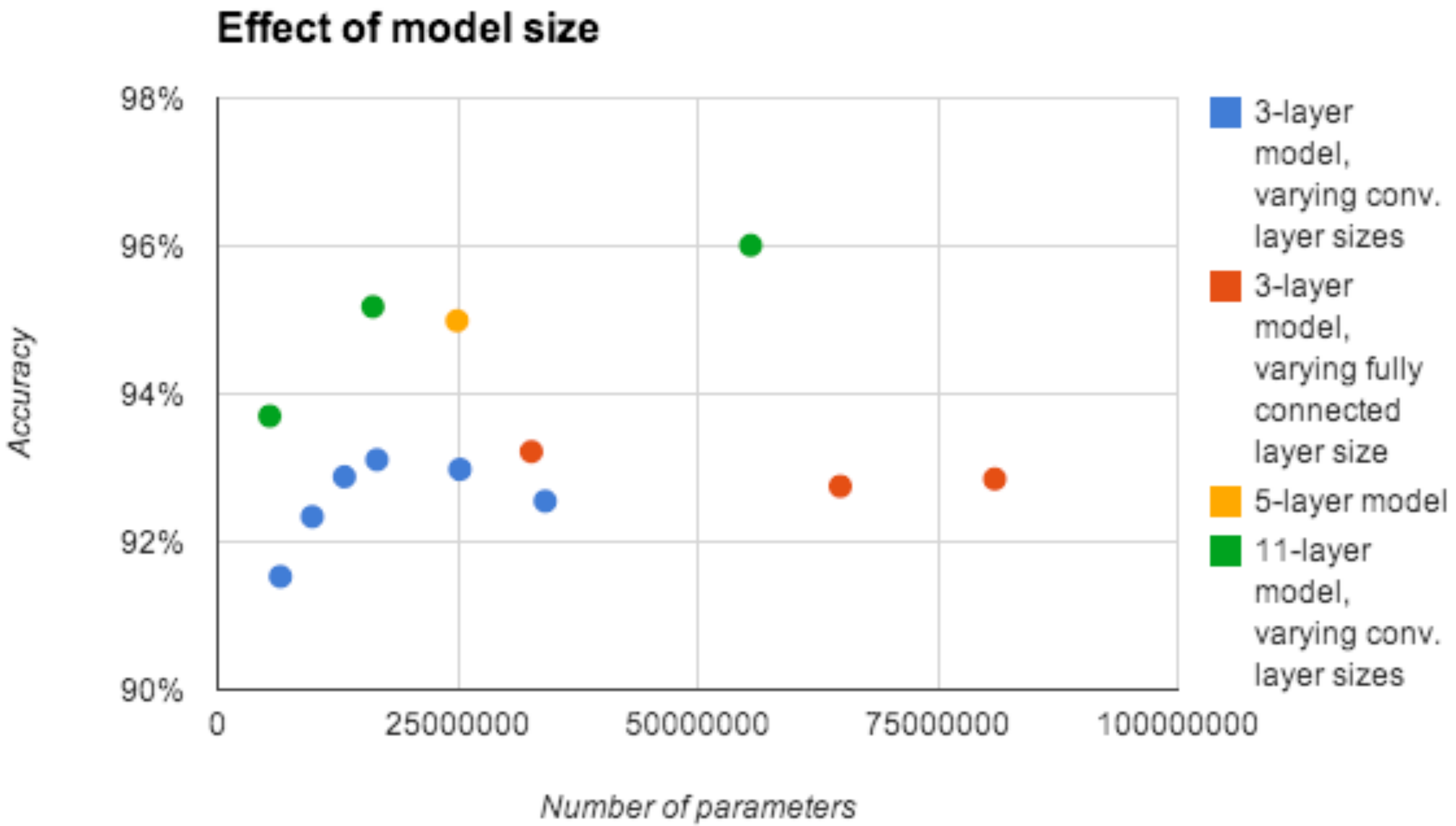


100 vs. 676

Effect of depth



Effect of # of parameters



Conclusion

- Unsupervised learning useful when very little labeled data available
- Generative models useful for missing value problems
- Implicit ensembles and/or lots of data are much more effective