NIPS 2010 Workshop on
Deep Learning and Unsupervised Feature Learning

Tutorial on Deep Learning and Applications

Honglak Lee
University of Michigan

Co-organizers: Yoshua Bengio, Geoff Hinton, Yann LeCun, Andrew Ng, and Marc'Aurelio Ranzato

* Includes slide material sourced from the co-organizers

Outline

- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

Outline

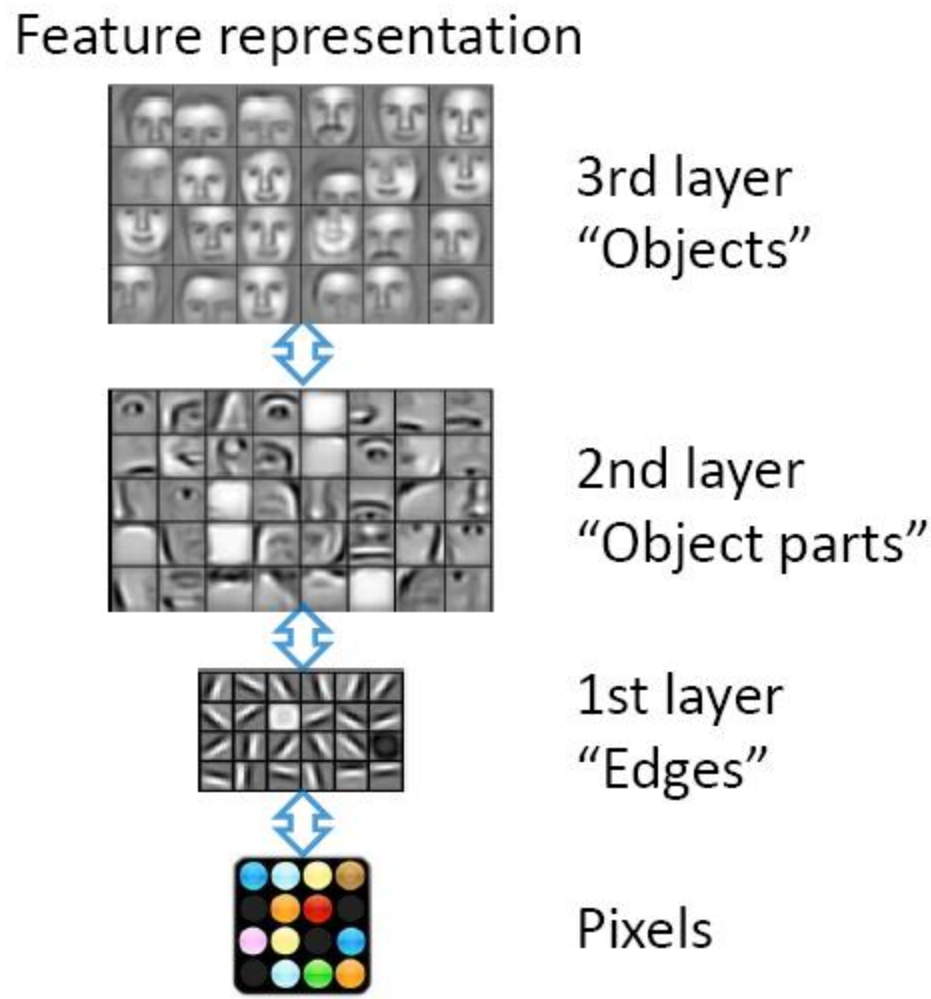
- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

Motivation: why go deep?

- Deep Architectures can be representationally efficient
  - Fewer computational units for same function
- Deep Representations might allow for a hierarchy or representation
  - Allows non-local generalization
  - Comprehensibility
- Multiple levels of latent variables allow combinatorial sharing of statistical strength
- Deep architectures work well (vision, audio, NLP, etc.)!

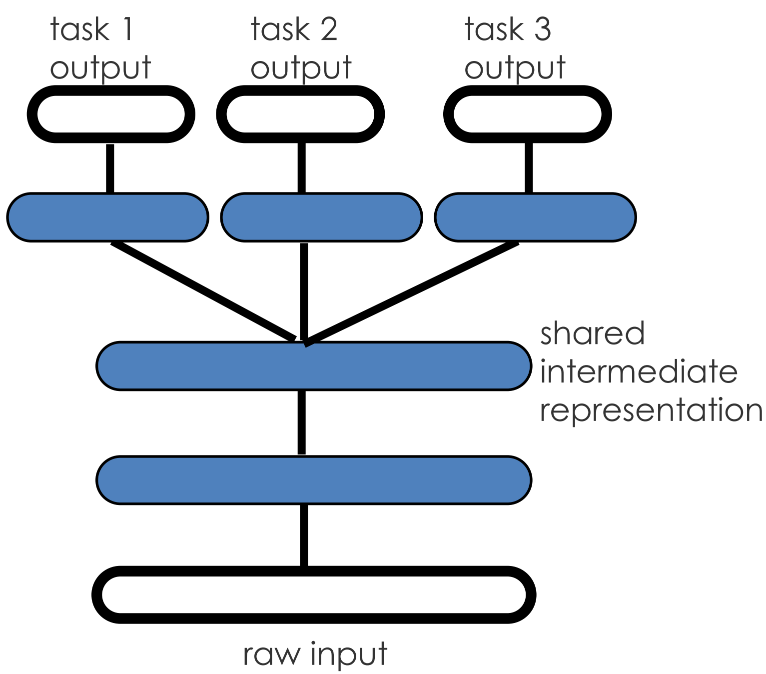
Different Levels of Abstraction

- Hierarchical Learning
  - Natural progression from low level to high level structure as seen in natural complexity
  - Easier to monitor what is being learnt and to guide the machine to better subspaces
  - A good lower level representation can be used for many distinct tasks

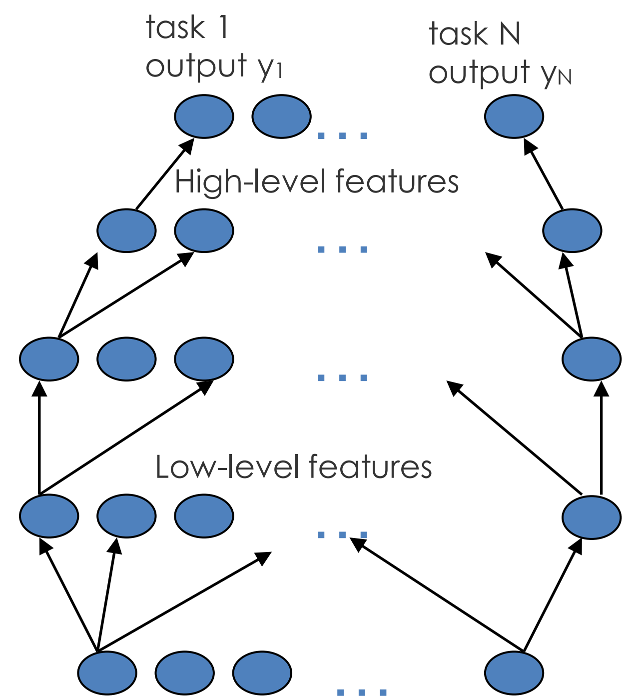


Generalizable Learning

- Shared Low Level Representations
  - Multi-Task Learning
  - Unsupervised Training

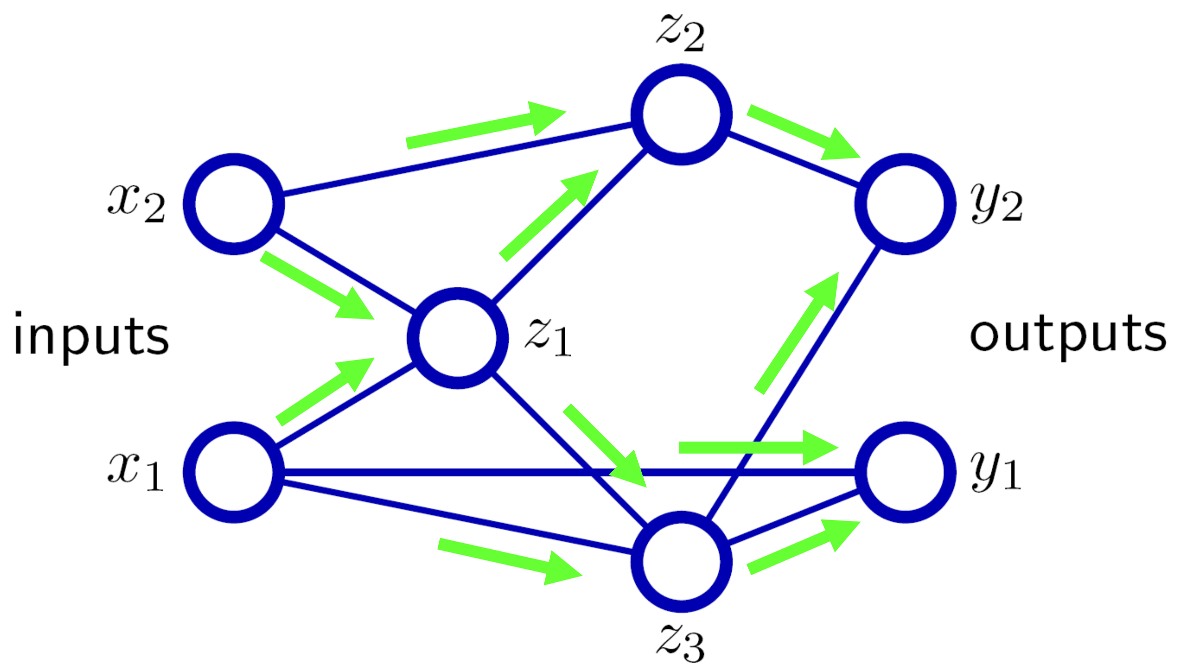


- Partial Feature Sharing
  - Mixed Mode Learning
  - Composition of Functions



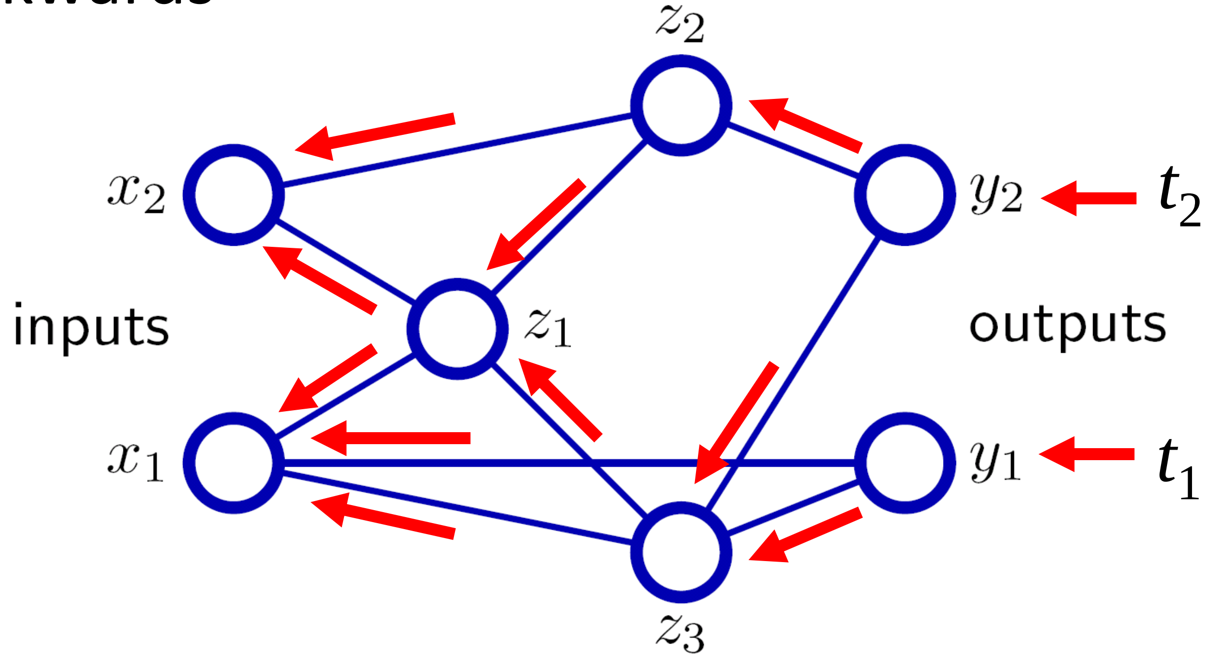
A Neural Network

- Forward Propagation :
  - Sum inputs, produce activation, feed-forward

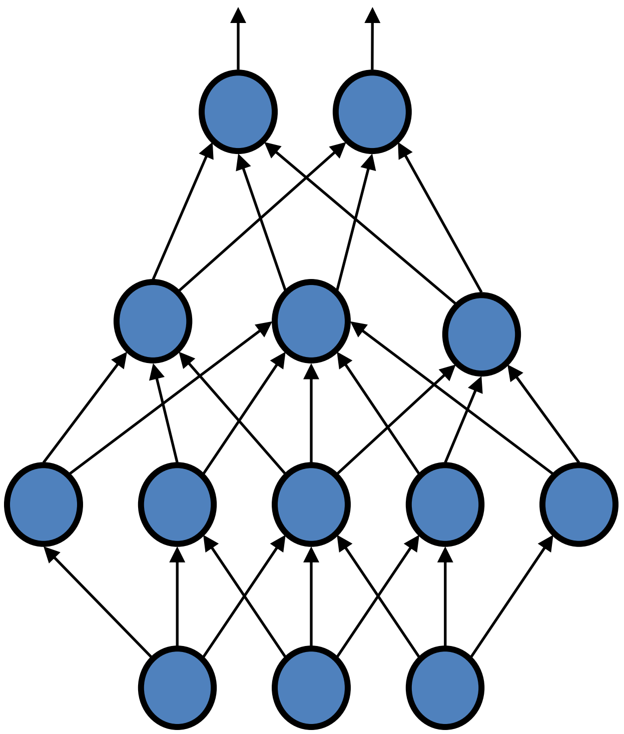


A Neural Network

- Training : Back Propagation of Error
  - Calculate total error at the top
  - Calculate contributions to error at each step going backwards



Deep Neural Networks



- Simple to construct
  - Sigmoid nonlinearity for hidden layers
  - Softmax for the output layer
- But, backpropagation does not work well (if randomly initialized)
  - Deep networks trained with backpropagation (without unsupervised pretraining) perform worse than shallow networks

	train.	valid.	test
DBN, unsupervised pre-training	0%	1.2%	1.2%
Deep net, auto-associator pre-training	0%	1.4%	1.4%
Deep net, supervised pre-training	0%	1.7%	2.0%
Deep net, no pre-training	.004%	2.1%	2.4%
Shallow net, no pre-training	.004%	1.8%	1.9%

(Bengio et al., NIPS 2007)

Problems with Back Propagation

- Gradient is progressively getting more dilute
  - Below top few layers, correction signal is minimal
- Gets stuck in local minima
  - Especially since they start out far from ‘good’ regions (i.e., random initialization)
- In usual settings, we can use only labeled data
  - Almost all data is unlabeled!
  - The brain can learn from unlabeled data

Deep Network Training (that actually works)

- Use unsupervised learning (greedy layer-wise training)
  - Allows abstraction to develop naturally from one layer to another
  - Help the network initialize with good parameters

- Perform supervised top-down training as final step
  - Refine the features (intermediate layers) so that they become more relevant for the task

Outline

- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

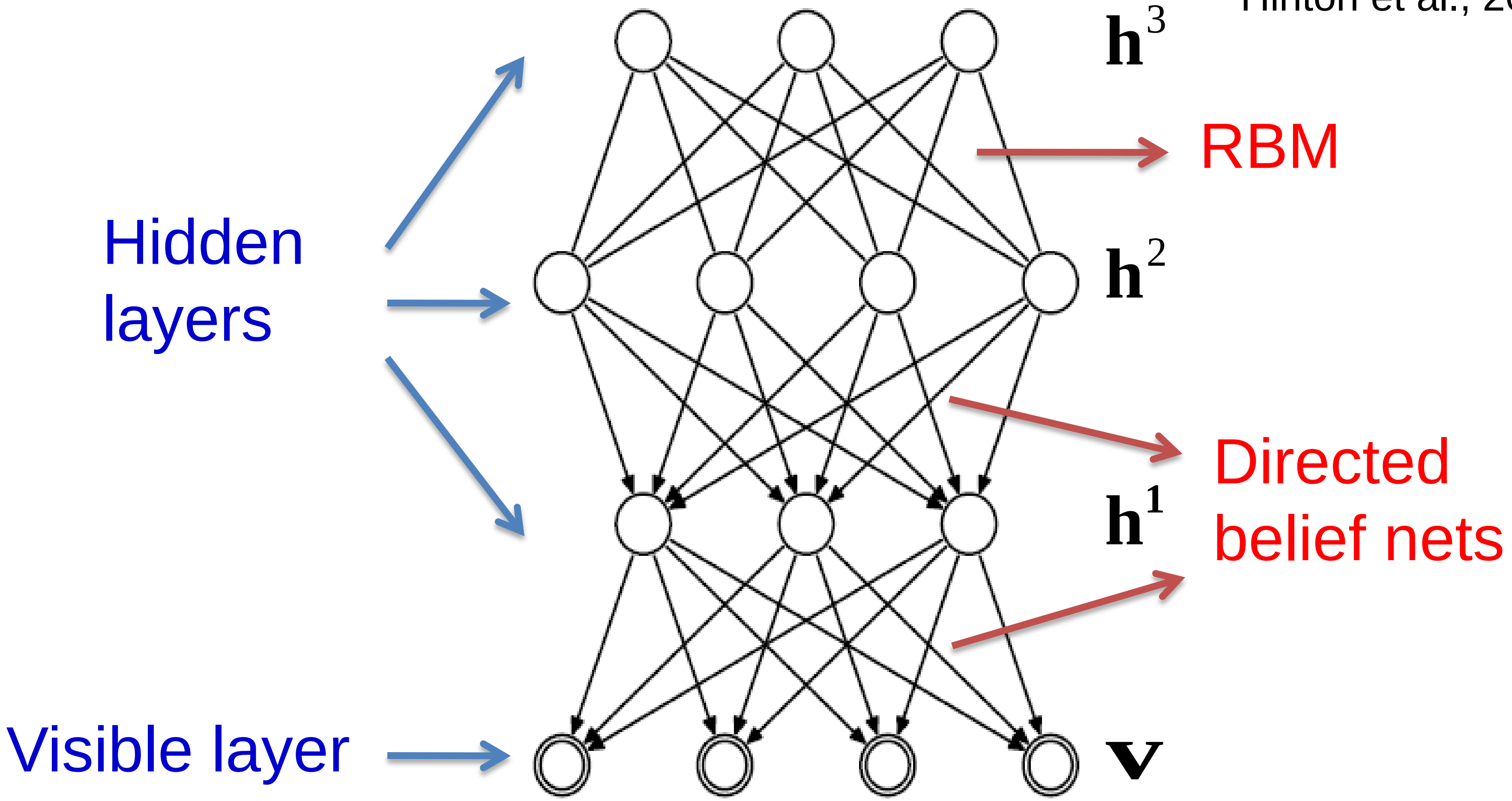
Deep Belief Networks(DBNs)

Hinton et al., 2006

- Probabilistic generative model
- Deep architecture – multiple layers
- Unsupervised pre-learning provides a good initialization of the network
  - maximizing the lower-bound of the log-likelihood of the data
- Supervised fine-tuning
  - Generative: Up-down algorithm
  - Discriminative: backpropagation

DBN structure

Hinton et al., 2006

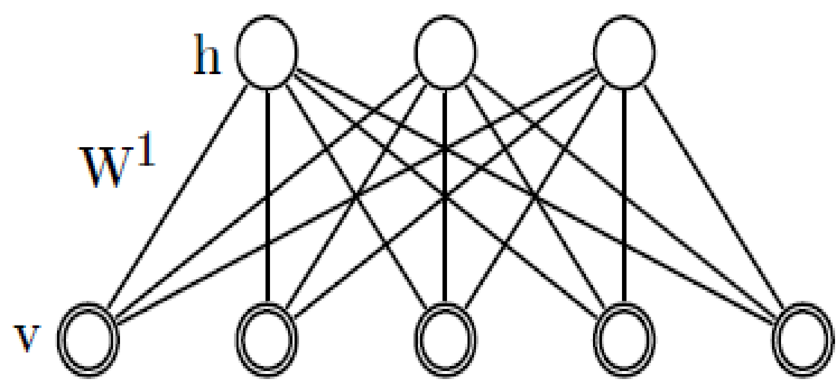


$$P(\mathbf{v},\mathbf{h}^1,\mathbf{h}^2,\ldots,\mathbf{h}^l) = P(\mathbf{v}\,|\,\mathbf{h}^1)P(\mathbf{h}^1\,|\,\mathbf{h}^2)\ldots P(\mathbf{h}^{l-2}\,|\,\mathbf{h}^{l-1})P(\mathbf{h}^{l-1},\mathbf{h}^l)$$

DBN Greedy training

Hinton et al., 2006

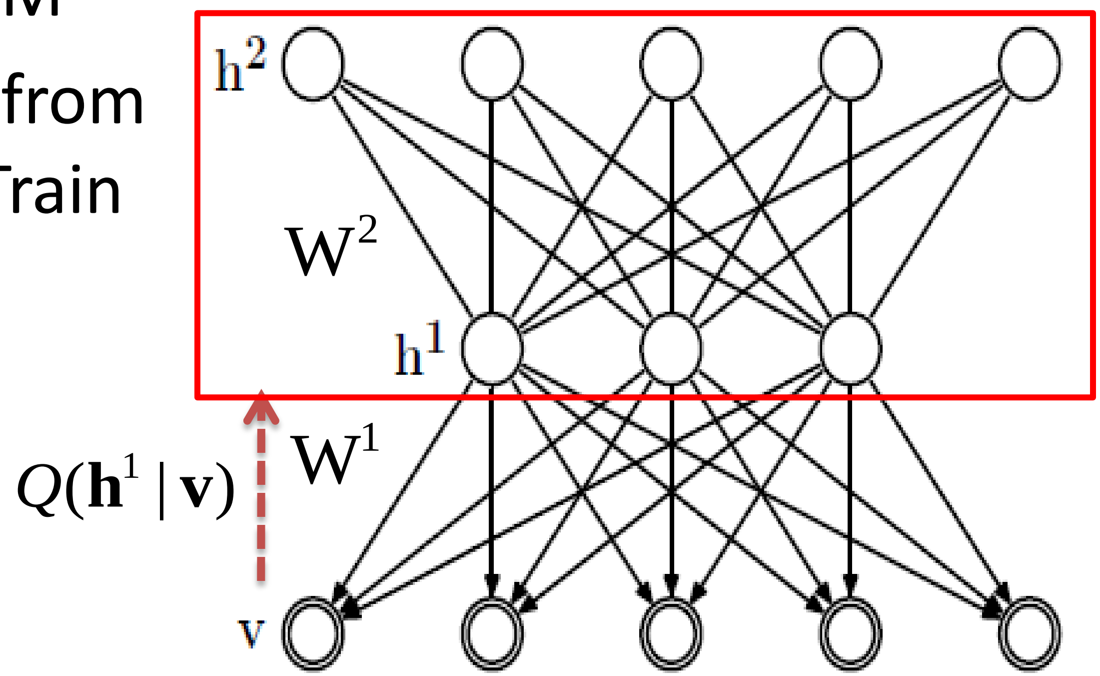
- First step:
  - Construct an RBM with an input layer **v** and a hidden layer **h**
  - Train the RBM



DBN Greedy training

Hinton et al., 2006

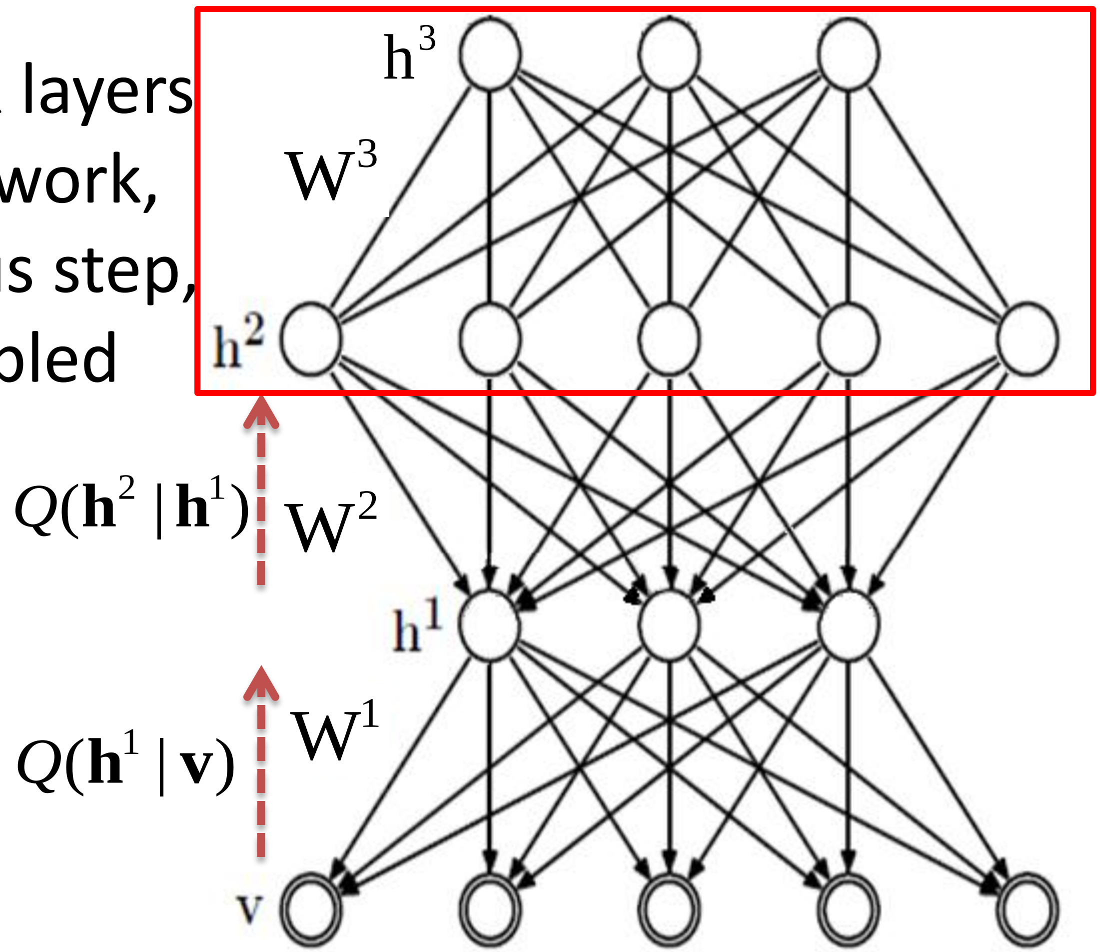
- Second step:
  - Stack another hidden layer on top of the RBM to form a new RBM
  - Fix W^1, sample $\mathbf{h}^1$ from $Q(\mathbf{h}^1 \mid \mathbf{v})$ as input. Train W^2 as RBM.



DBN Greedy training

Hinton et al., 2006

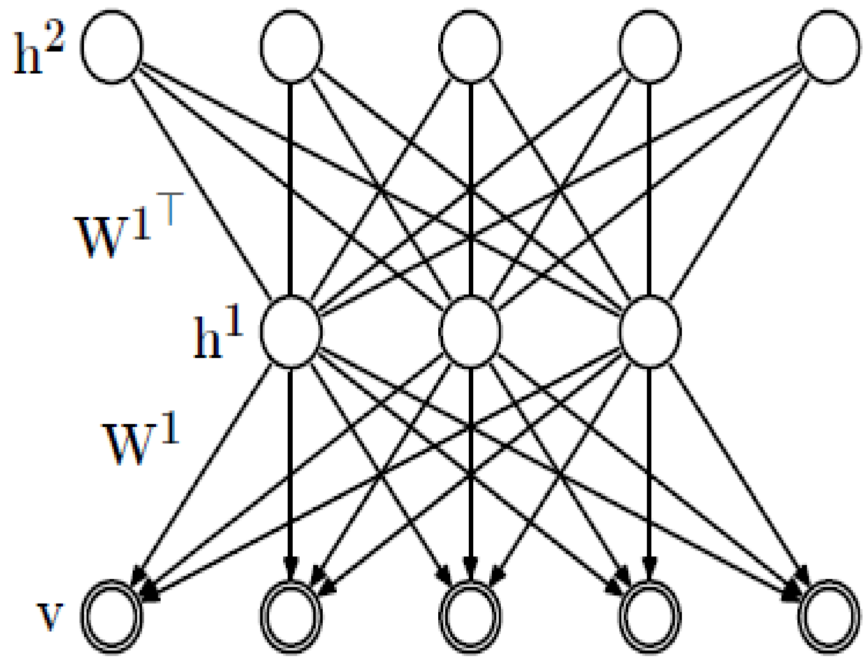
- Third step:
  - Continue to stack layers on top of the network, train it as previous step, with sample sampled from $Q(\mathbf{h}^2 \mid \mathbf{h}^1)$
- And so on…



Why greedy training works?

Hinton et al., 2006

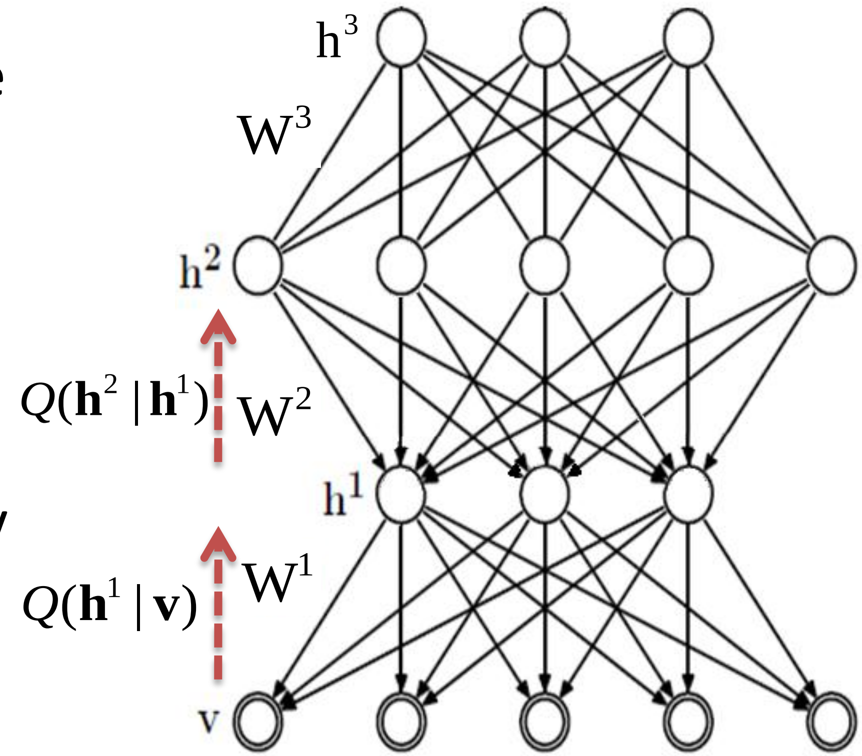
- RBM specifies P(v,h) from P(v|h) and P(h|v)
  - Implicitly defines P(v) and P(h)
- Key idea of stacking
  - Keep P(v|h) from 1st RBM
  - Replace P(h) by the distribution generated by 2nd level RBM



Why greedy training works?

Hinton et al., 2006

- Easy approximate inference
  - $P(h_{k+1}|h_k)$ approximated from the associated RBM
  - Approximation because $P(h_{k+1})$ differs between RBM and DBN
- Training:
  - Variational bound justifies greedy layerwise training of RBMs



$$\log P(\mathbf{x}) \geq H_{Q(\mathbf{h}|\mathbf{x})} + \sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{x}) \left(\underline{\log P(\mathbf{h})} + \log P(\mathbf{x}|\mathbf{h})\right)$$

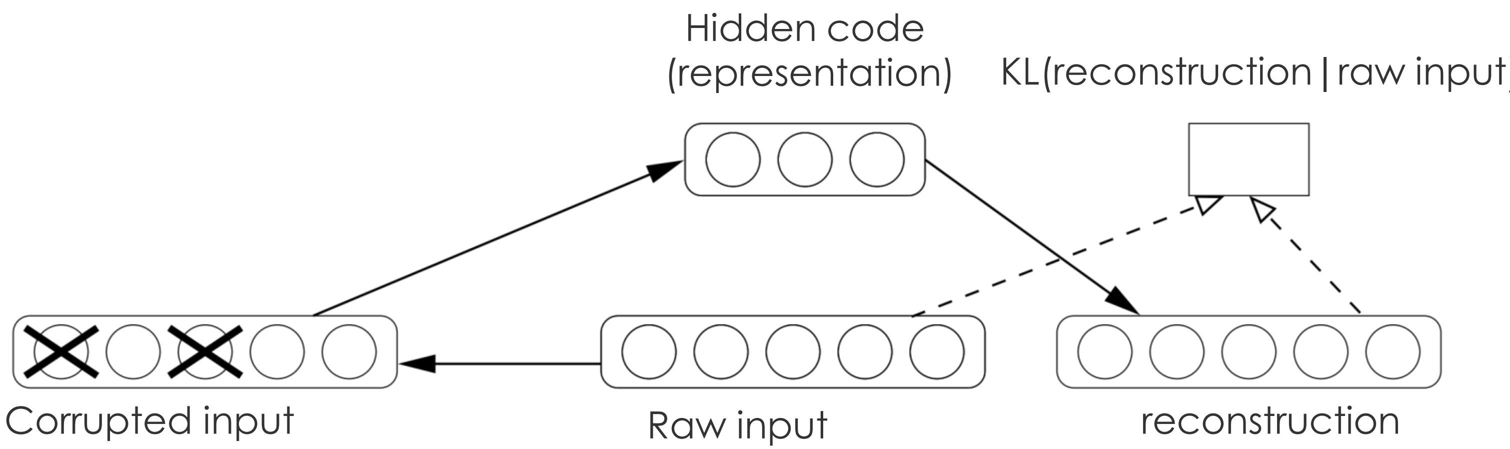
Trained by the second layer RBM

Outline

- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

Denoising Auto-Encoder

(Vincent et al, 2008)

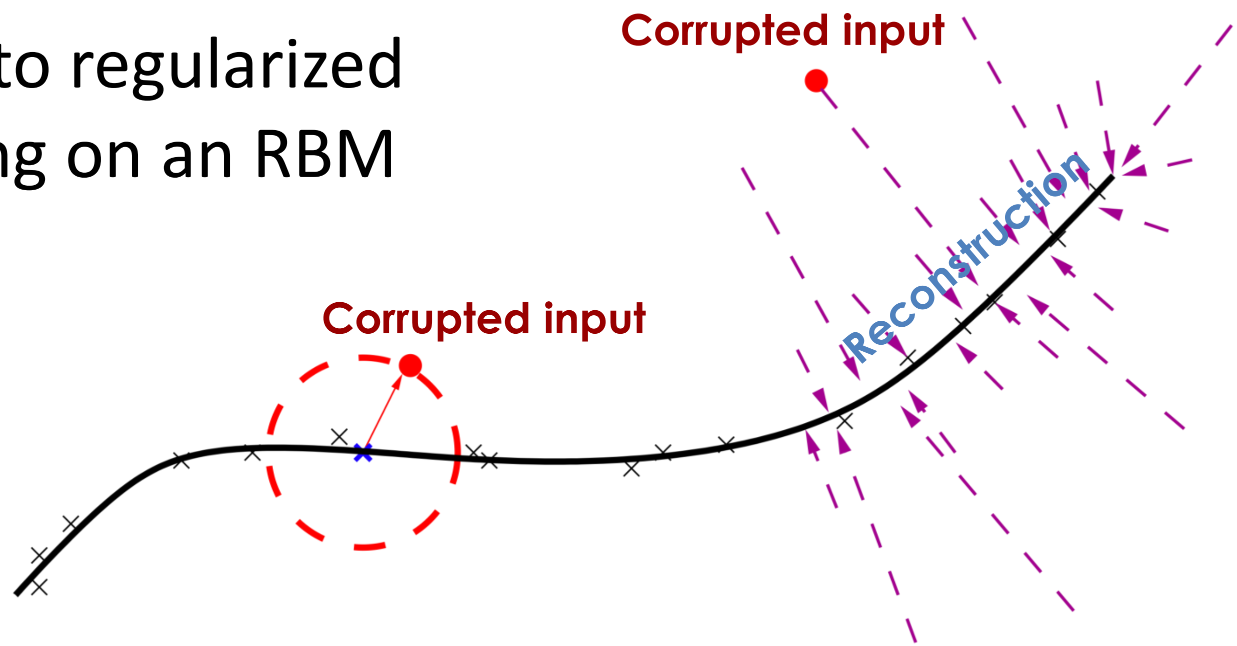


- Corrupt the input (e.g. set 25% of inputs to 0)
- Reconstruct the uncorrupted input
- Use uncorrupted encoding as input to next level

Denoising Auto-Encoder

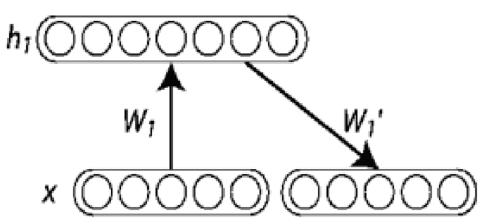
(Vincent et al, 2008)

- Learns a vector field towards higher probability regions
- Minimizes variational lower bound on a generative model
- Corresponds to regularized score matching on an RBM



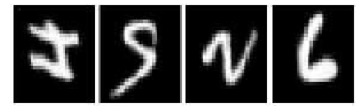
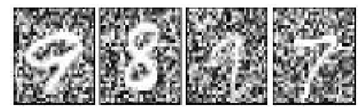
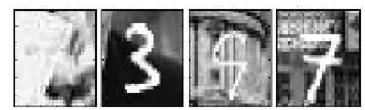
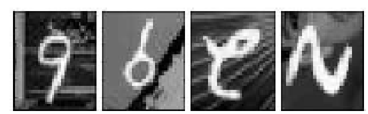
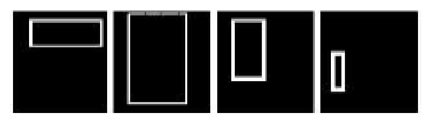
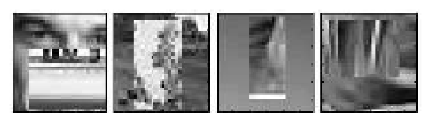
Stacked (Denoising) Auto-Encoders

- Greedy Layer wise learning
  - Start with the lowest level and stack upwards
  - Train each layer of auto-encoder on the intermediate code (features) from the layer below
  - Top layer can have a different output (e.g., softmax non-linearity) to provide an output for classification



Denoising Auto-Encoders: Benchmarks

Larochelle et al., 2009

basic: subset of MNIST digits.	(10 000 training samples)
rot: applied random rotation (angle between 0 and 2π radians)	
bg-rand: background made of random pixels (value in $0 \ldots 255$)	
bg-img: background is random patch from one of 20 images	
rot-bg-img: combination of rotation and background image	
rect: discriminate between tall and wide rectangles.	
rect-img: same but rectangles are random image patches	
convex: discriminate between convex and non-convex shapes.	

Denoising Auto-Encoders: Results

- Test errors on the benchmarks

Larochelle et al., 2009

Problem	SVM_{rbf}	DBN-1	DBN-3	SAA-3	SdA-3 (ν)	$SVM_{rbf}(\nu)$
basic	3.03 ± 0.15	3.94 ± 0.17	3.11 ± 0.15	3.46 ± 0.16	2.80 ± 0.14 (10%)	3.07 (10%)
rot	11.11 ± 0.28	14.69 ± 0.31	10.30 ± 0.27	10.30 ± 0.27	10.29 ± 0.27 (10%)	11.62 (10%)
bg-rand	14.58 ± 0.31	9.80 ± 0.26	6.73 ± 0.22	11.28 ± 0.28	10.38 ± 0.27 (40%)	15.63 (25%)
bg-img	22.61 ± 0.37	16.15 ± 0.32	16.31 ± 0.32	23.00 ± 0.37	16.68 ± 0.33 (25%)	23.15 (25%)
rot-bg-img	55.18 ± 0.44	52.21 ± 0.44	47.39 ± 0.44	51.93 ± 0.44	44.49 ± 0.44 (25%)	54.16 (10%)
rect	2.15 ± 0.13	4.71 ± 0.19	2.60 ± 0.14	2.41 ± 0.13	1.99 ± 0.12 (10%)	2.45 (25%)
rect-img	24.04 ± 0.37	23.69 ± 0.37	22.50 ± 0.37	24.05 ± 0.37	21.59 ± 0.36 (25%)	23.00 (10%)
convex	19.13 ± 0.34	19.92 ± 0.35	18.63 ± 0.34	18.41 ± 0.34	19.06 ± 0.34 (10%)	24.20 (10%)

Outline

- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

Predictive Sparse Coding

- Recall the objective function for sparse coding:

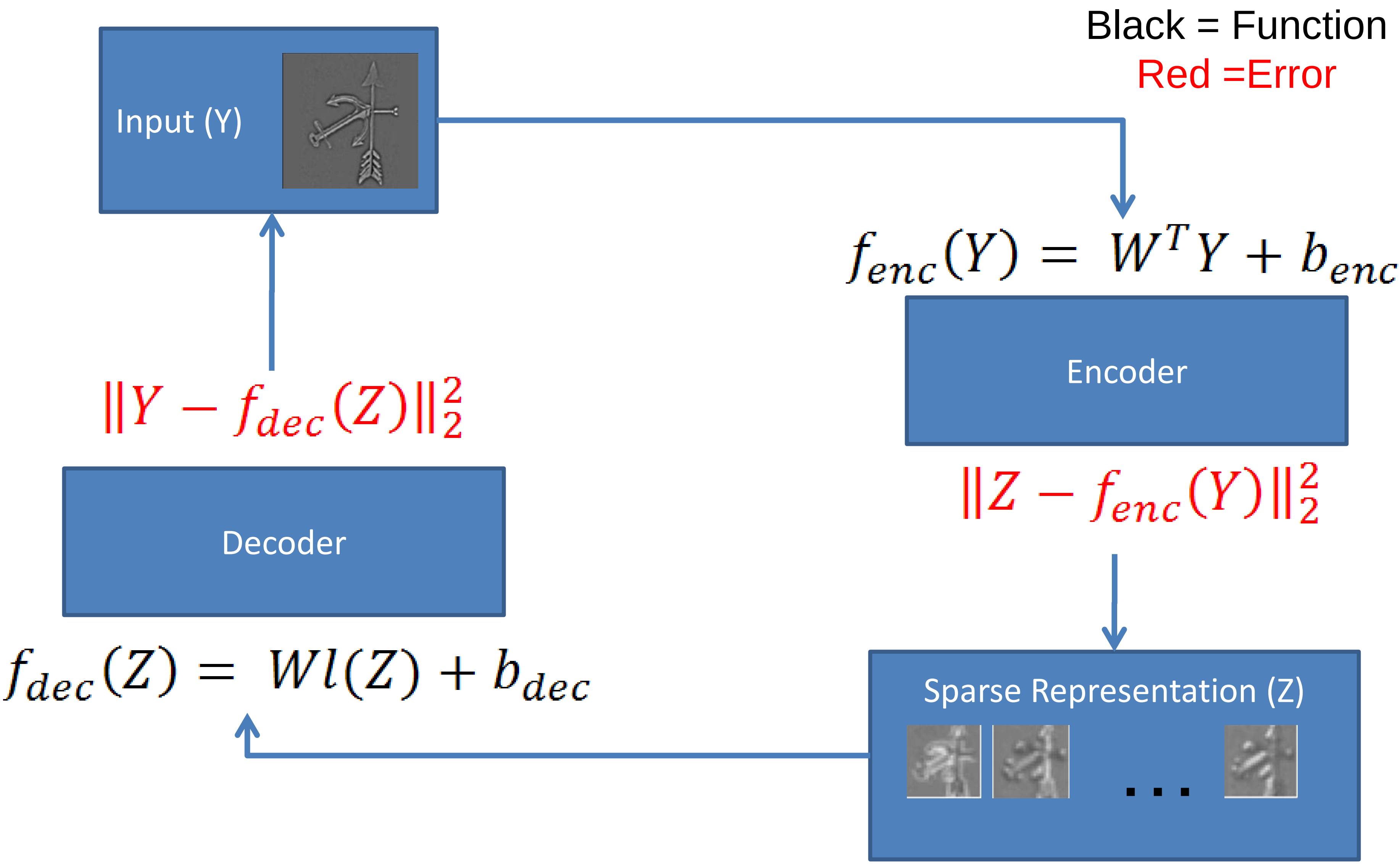
$$\min \frac{1}{2} \|Y - BZ\|_2^2 + \lambda \|Z\|_1$$

- Modify by adding a penalty for prediction error:
  - Approximate the sparse code with an encoder

$$\min \frac{1}{2} \|Y - BZ\|_2^2 + \lambda \|Z\|_1 + \alpha \left\| \boldsymbol{Z} - \boldsymbol{F}(\boldsymbol{Y}; \boldsymbol{P_f}) \right\|_2^2$$

$$F(Y; G, W, D) = G \tanh(WY + D)$$

Encoder Decoder Model

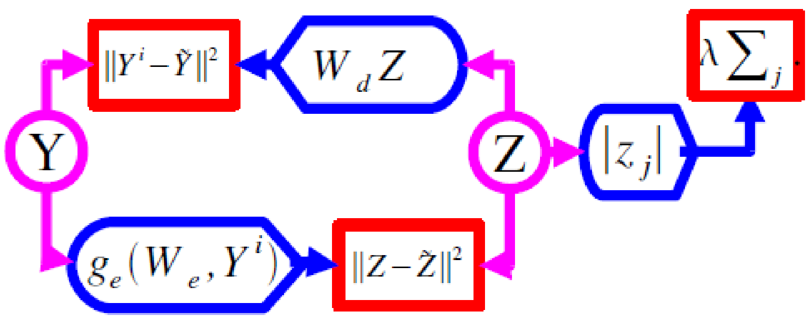


Using PSD to Train a Hierarchy of Features

- Phase 1: train first layer using PSD

$$E(Y^i, Z) = \|Y^i - W_d Z\|^2 + \|Z - g_e(W_e, Y^i)\|^2 + \lambda \sum_j |z_j|$$

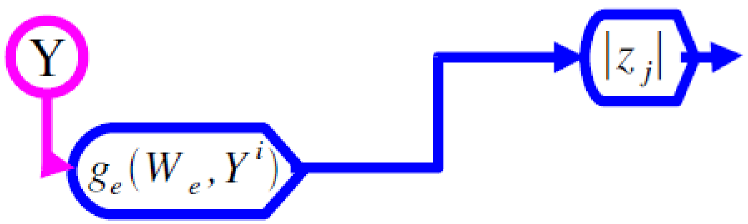
$$g_e(W_e, Y^i) = D \tanh(W_e Y)$$



FEATURES

Using PSD to Train a Hierarchy of Features

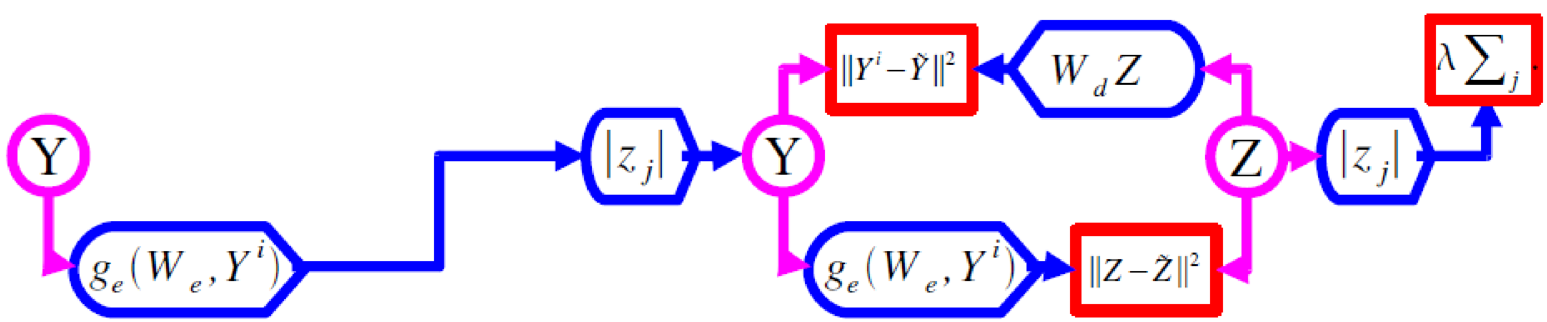
- Phase 1: train first layer using PSD
- Phase 2: use encoder+absolute value as feature extractor



FEATURES

Using PSD to Train a Hierarchy of Features

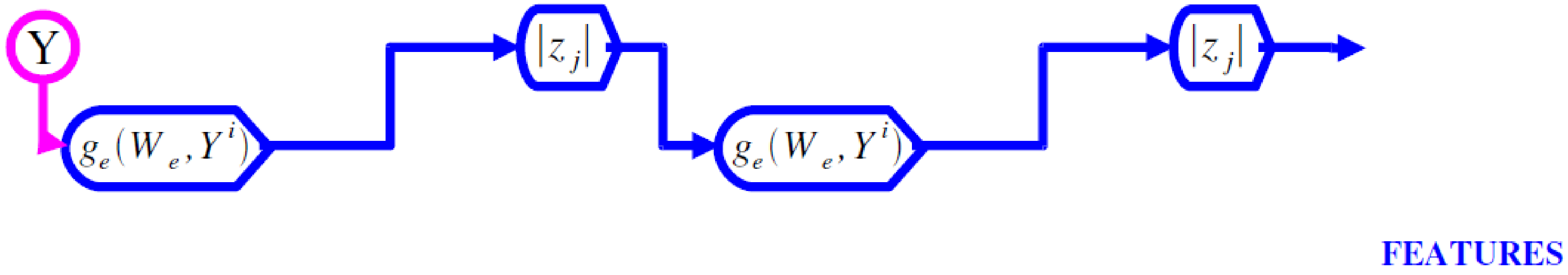
- Phase 1: train first layer using PSD
- Phase 2: use encoder+absolute value as feature extractor
- Phase 3: train the second layer using PSD



FEATURES

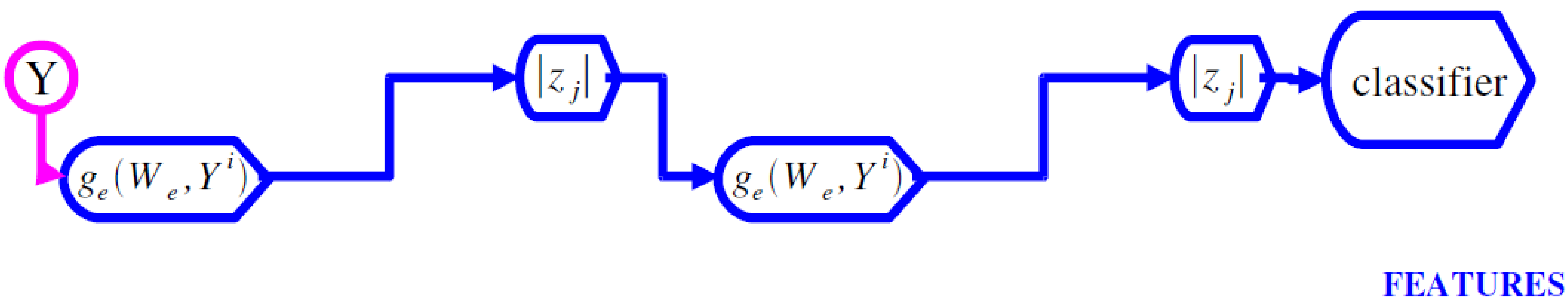
Using PSD to Train a Hierarchy of Features

- Phase 1: train first layer using PSD
- Phase 2: use encoder+absolute value as feature extractor
- Phase 3: train the second layer using PSD
- Phase 4: use encoder + absolute value as 2nd feature extractor



Using PSD to Train a Hierarchy of Features

- Phase 1: train first layer using PSD
- Phase 2: use encoder+absolute value as feature extractor
- Phase 3: train the second layer using PSD
- Phase 4: use encoder + absolute value as 2nd feature extractor
- Phase 5: train a supervised classifier on top
- Phase 6: (optional): train the entire system with supervised back-propagation



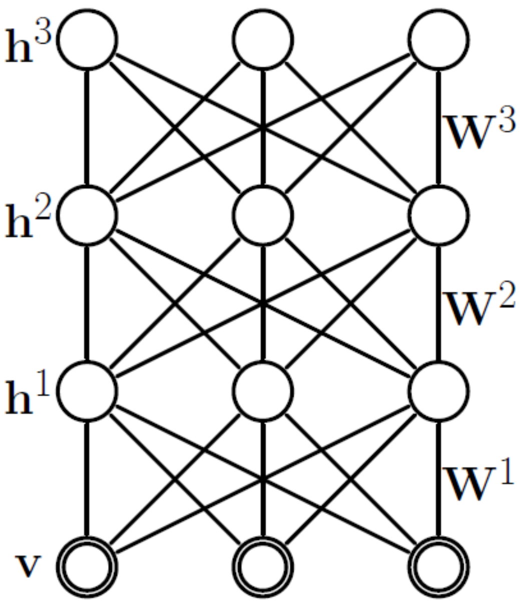
Outline

- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

Deep Boltzmann Machines

Salakhutdinov & Hinton, 2009

$$P(\mathbf{v}) = \sum_{\mathbf{h}^1,\mathbf{h}^2,\mathbf{h}^3} \frac{1}{\mathcal{Z}} \exp\left[\mathbf{v}^\top W^1 \mathbf{h} + \mathbf{h}^{1\top} W^2 \mathbf{h}^2 + \mathbf{h}^{2\top} W^3 \mathbf{h}^3\right].$$



Undirected connections between all layers
(no connections between the nodes in the same layer)

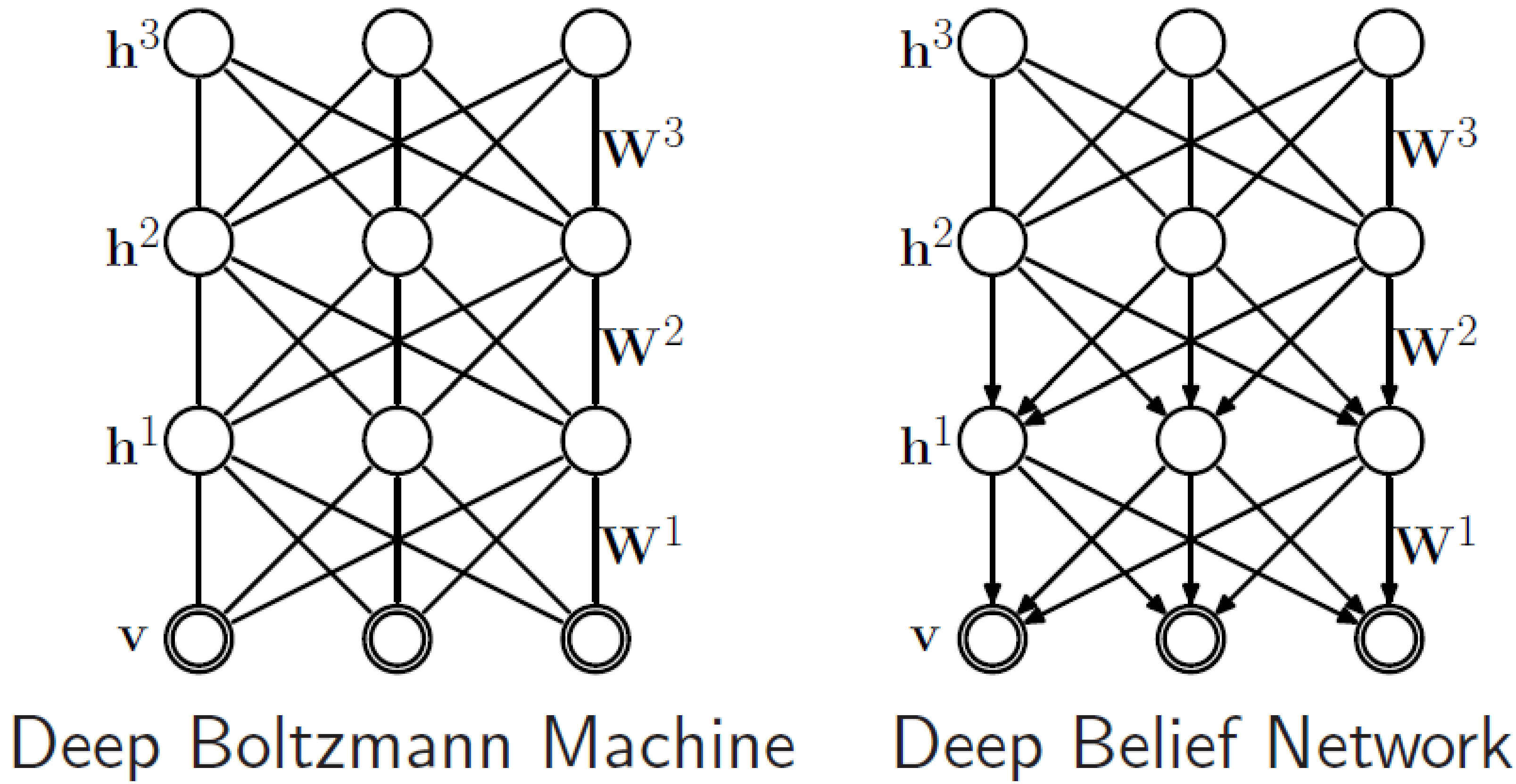
High-level representations are built from unlabeled inputs.

Labeled data is used to only slightly fine-tune the model.

Slide credit: R. Salskhutdinov

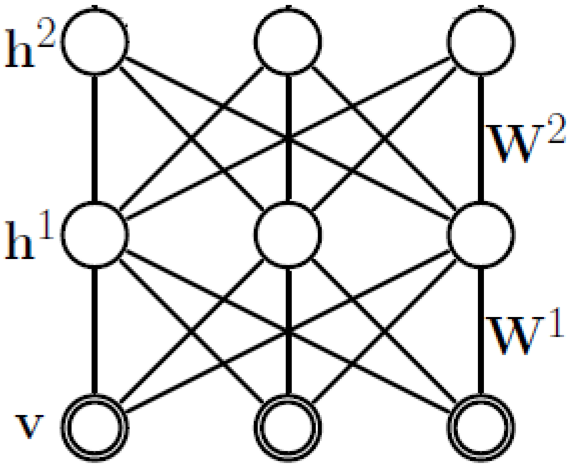
DBMs vs. DBNs

Salakhutdinov & Hinton, 2009



- In multiple layer model, the undirected connection between the layers make complete Boltzmann machine.

Two layer DBM example



$$E(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2; \theta) = -\mathbf{v}^\top \mathbf{W}^1 \mathbf{h}^1 - \mathbf{h}^{1\top} \mathbf{W}^2 \mathbf{h}^2$$

$$\theta = \{\mathbf{W}^1, \mathbf{W}^2\}$$

$$p(\mathbf{v}; \theta) = \frac{1}{Z(\theta)} \sum_{\mathbf{h}^1, \mathbf{h}^2} \exp\left(-E(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2; \theta)\right)$$

***Assume no within layer connection.**

$$p(h_j^1 = 1 | \mathbf{v}, \mathbf{h}^2) = \sigma\Big(\sum_i W_{ij}^1 v_i + \sum_m W_{jm}^2 h_j^2\Big)$$

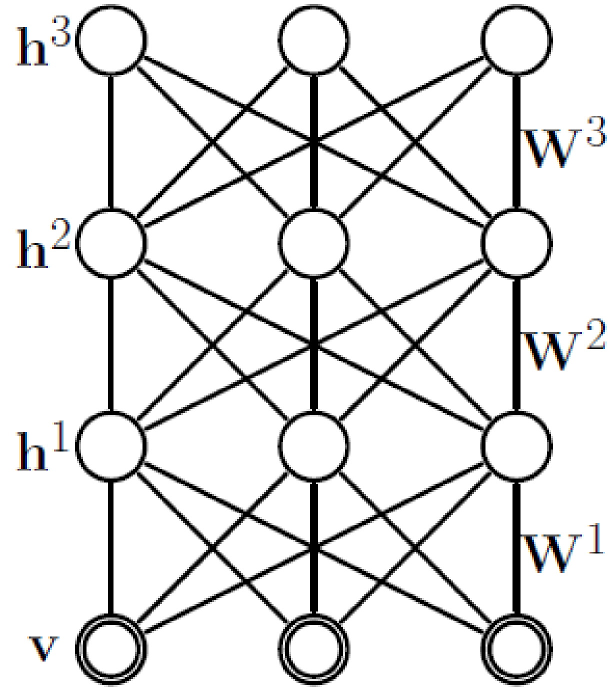
$$p(h_m^2 = 1 | \mathbf{h}^1) = \sigma\Big(\sum_j W_{im}^2 h_i^1\Big),$$

$$p(v_i = 1 | \mathbf{h}^1) = \sigma\Big(\sum_j W_{ij}^1 h_j\Big).$$

Deep Boltzman Machines

Salakhutdinov & Hinton, 2009

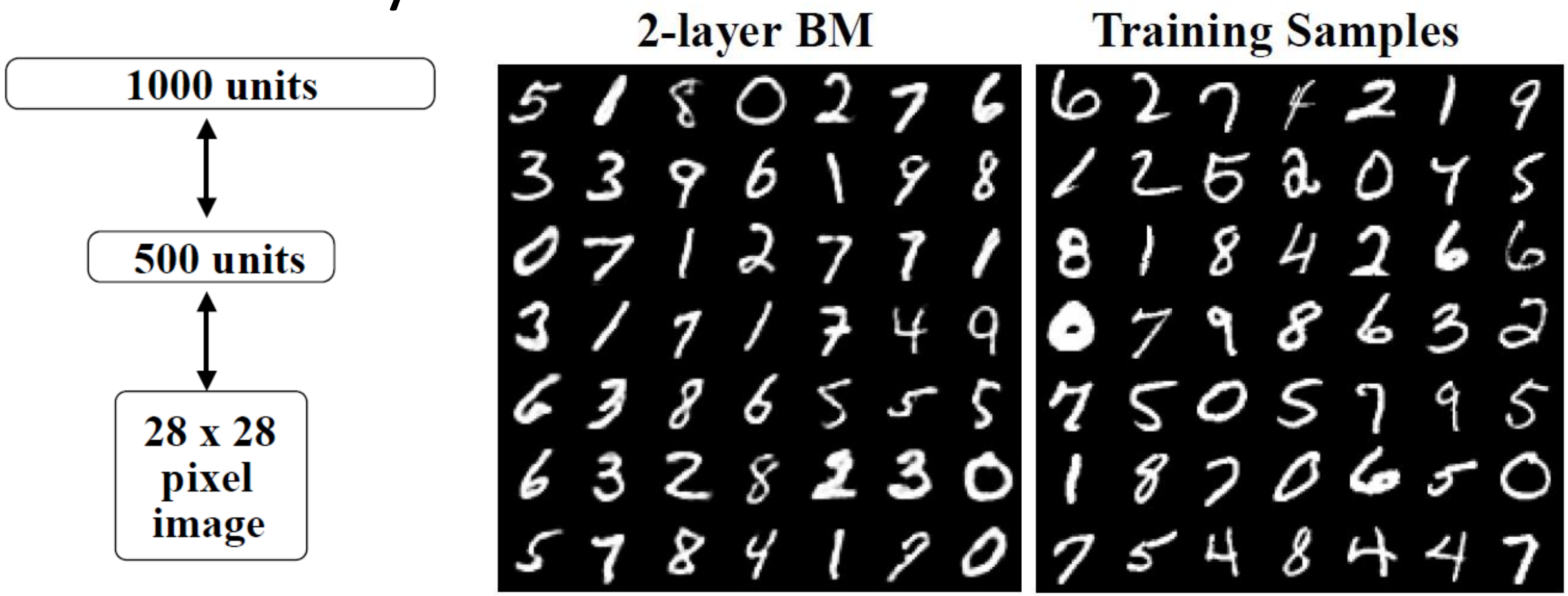
- Pre-training:
  - Can (must) initialize from stacked RBMs
- Generative fine-tuning:
  - Positive phase: variational approximation (mean-field)
  - Negative phase: persistent chain (stochastic approxiamtion)
- Discriminative fine-tuning:
  - backpropagation



Deep Boltzmann Machine

Experiments

- MNIST: 2-layer BM

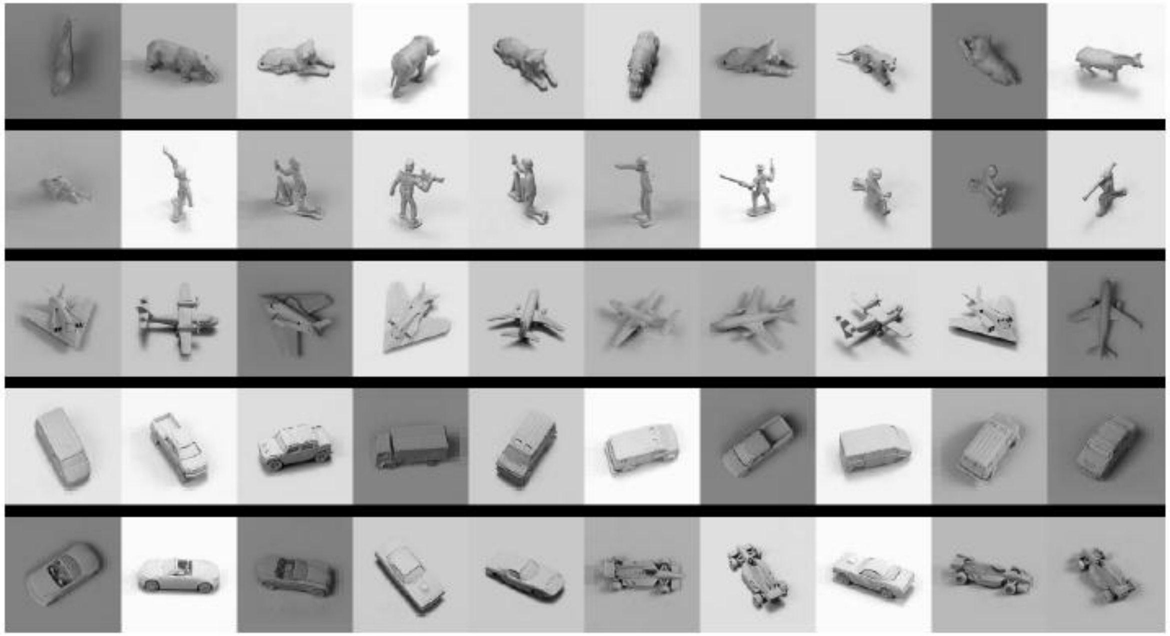


60,000 training and 10,000 testing examples
0.9 million parameters
Gibbs sampler for 100,000 steps

After discriminative fine-tuning: 0.95% error rate
Compare with DBN 1.2%, SVM 1.4%

Experiments

- NORB dataset

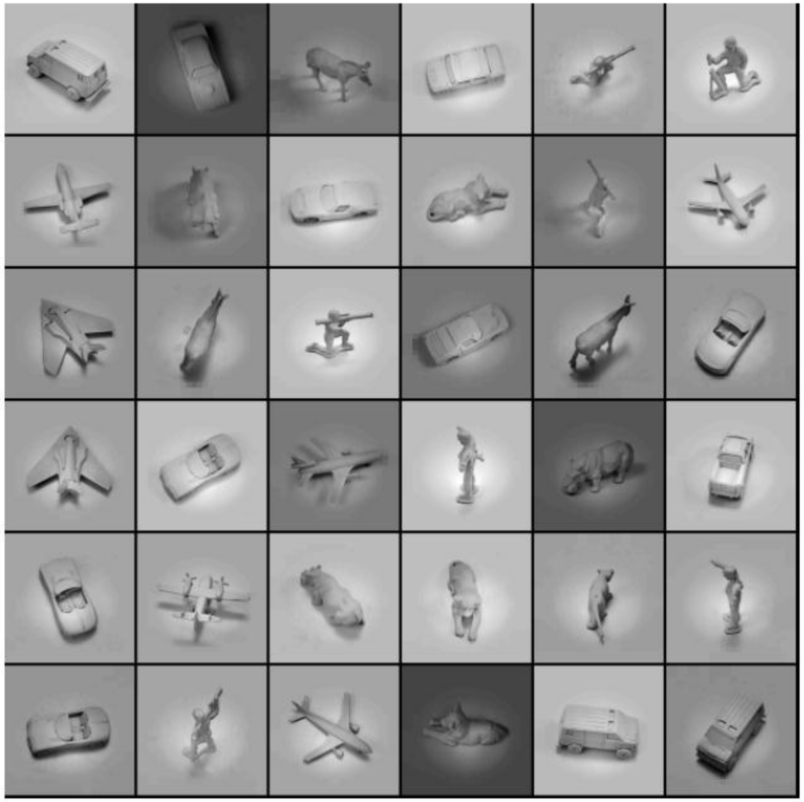


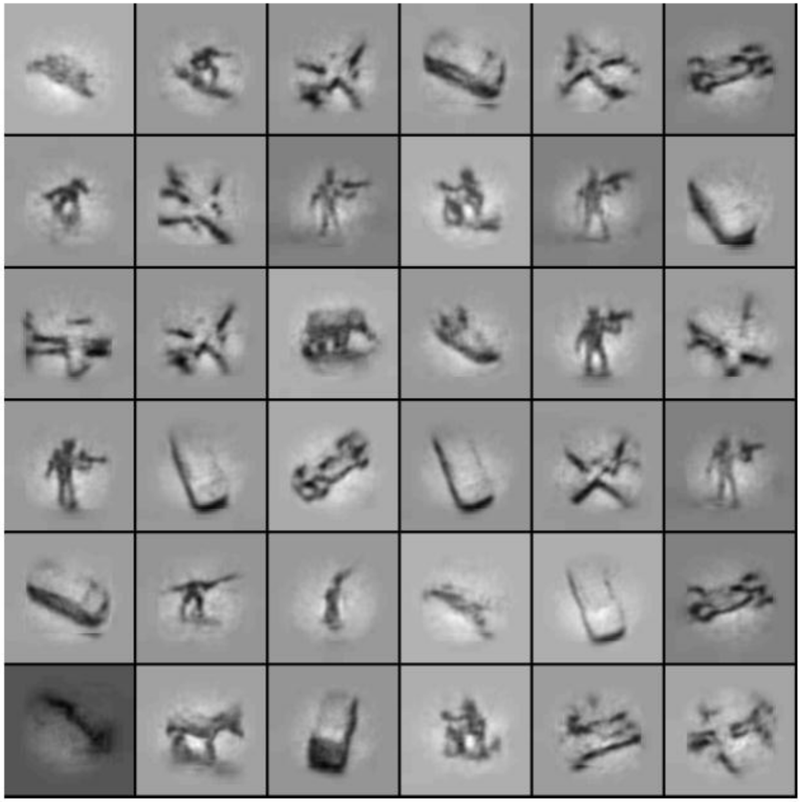
5 object categories, 5 different objects within each category, 6 lightning conditions, 9 elevations, 18 azimuth.

24,300 training and 24,300 test cases.

Slide credit: R. Salskhutdinov

Experiments





Test error of 7.2%.

SVM's get 11.6%, logistic regression gets 22.5%.

Slide credit: R. Salskhutdinov

Why Greedy Layer Wise Training Works

(Bengio 2009, Erhan et al. 2009)

- Regularization Hypothesis
  - Pre-training is “constraining” parameters in a region relevant to unsupervised dataset
  - Better generalization

    (Representations that better describe unlabeled data are more discriminative for labeled data)

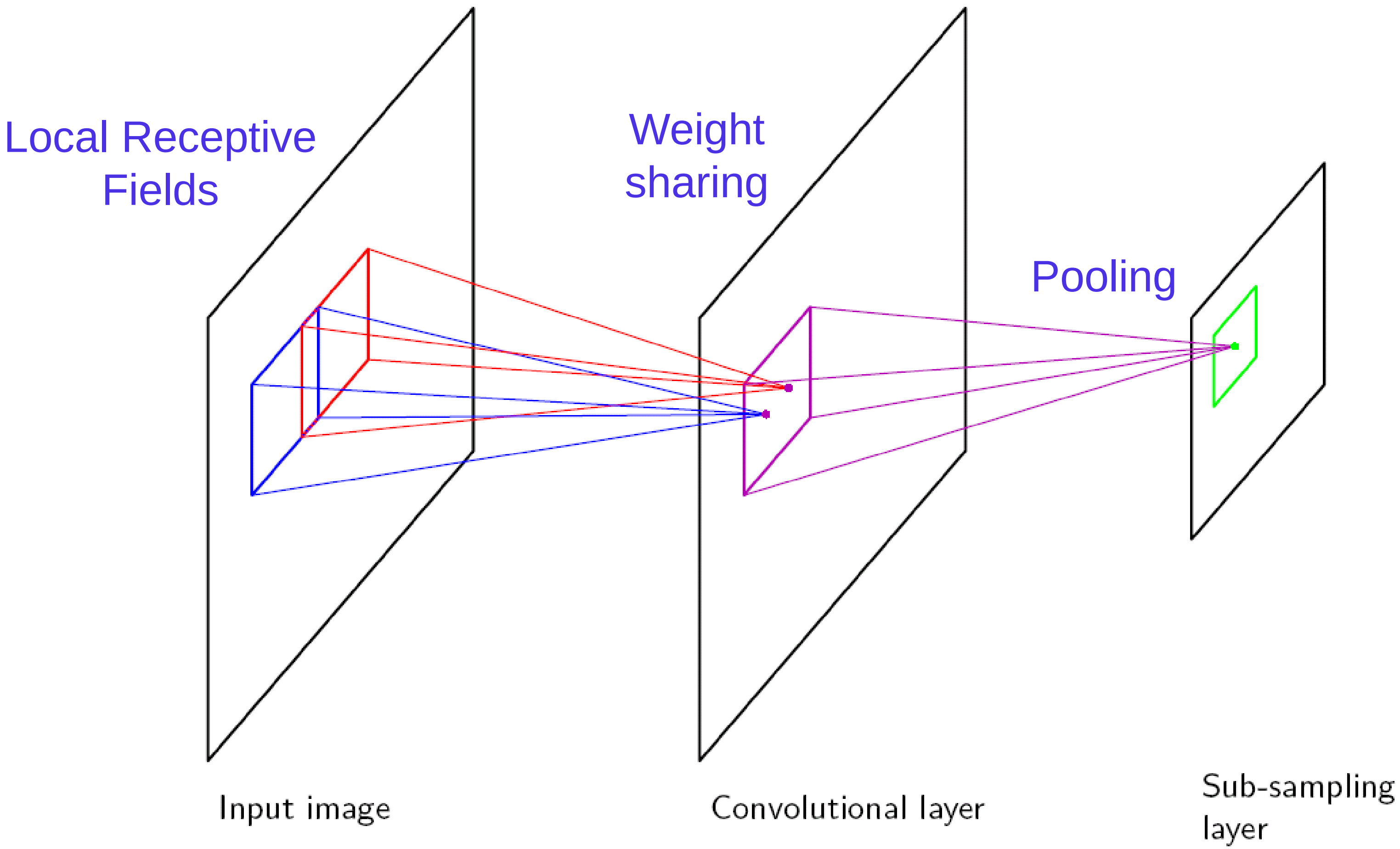
- Optimization Hypothesis
  - Unsupervised training initializes lower level parameters near localities of better minima than random initialization can

Outline

- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

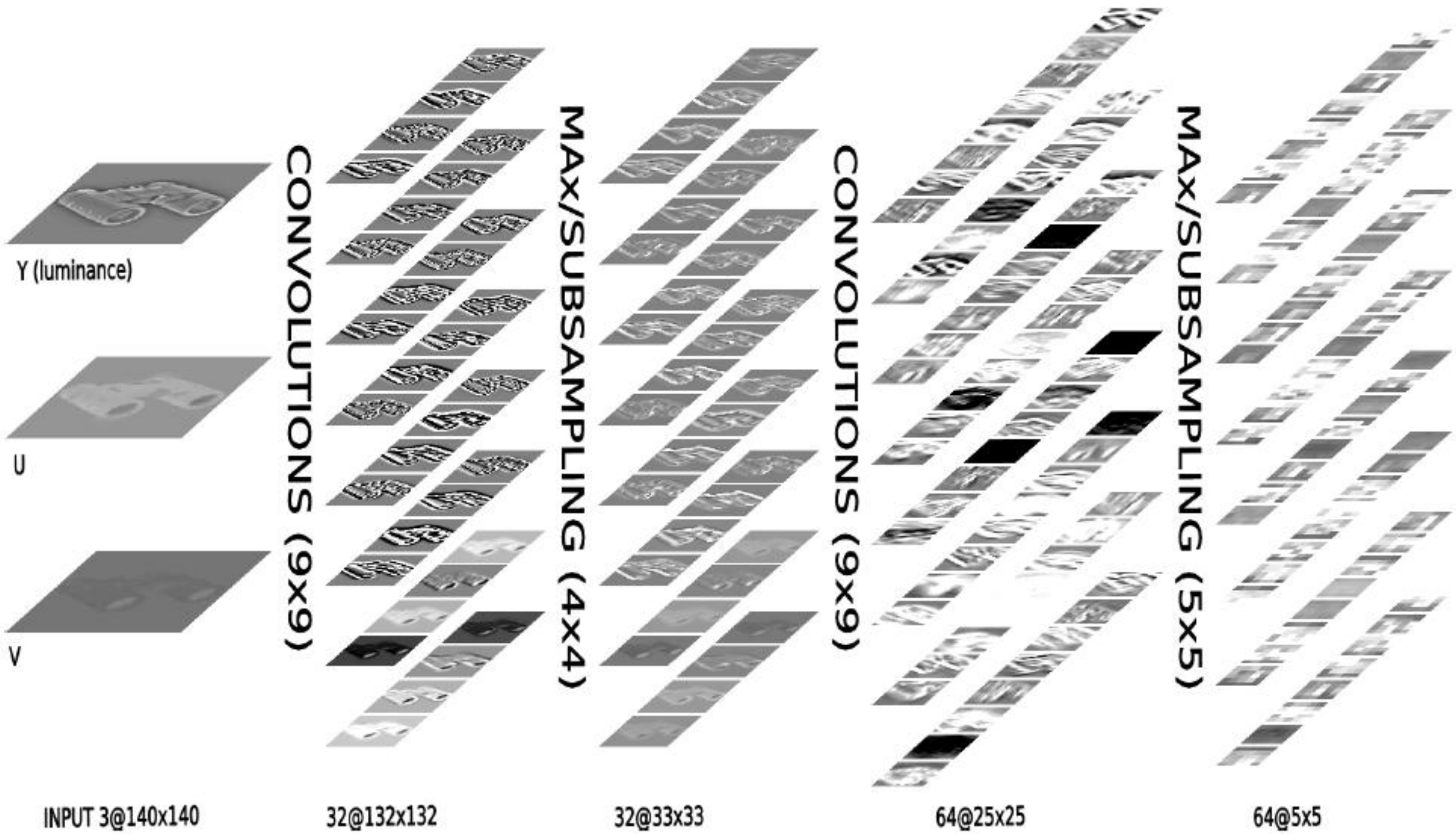
Convolutional Neural Networks

(LeCun et al., 1989)



Deep Convolutional Architectures

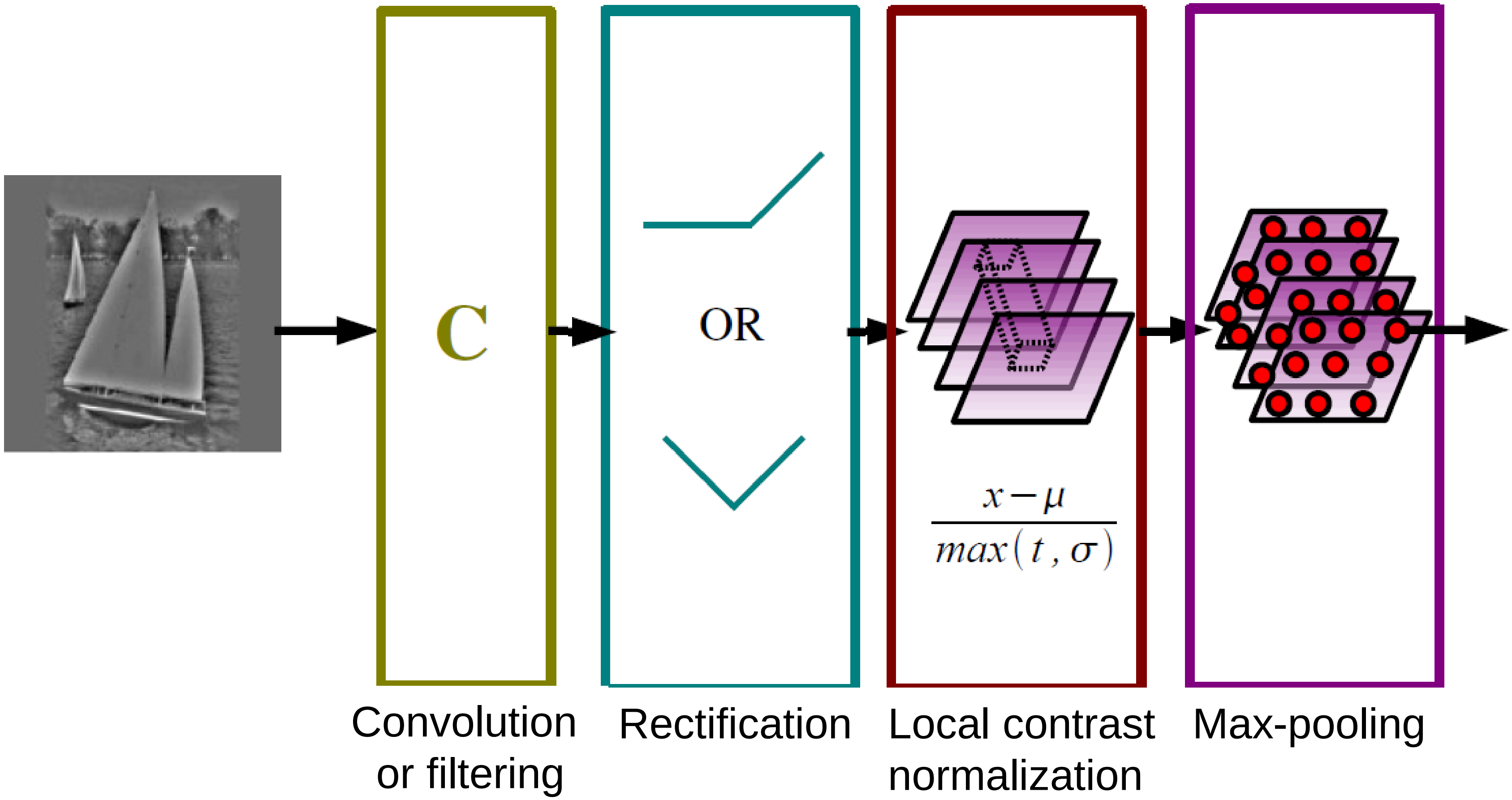
State-of-the-art on MNIST digits, Caltech-101 objects, etc.



Nonlinearities and pooling

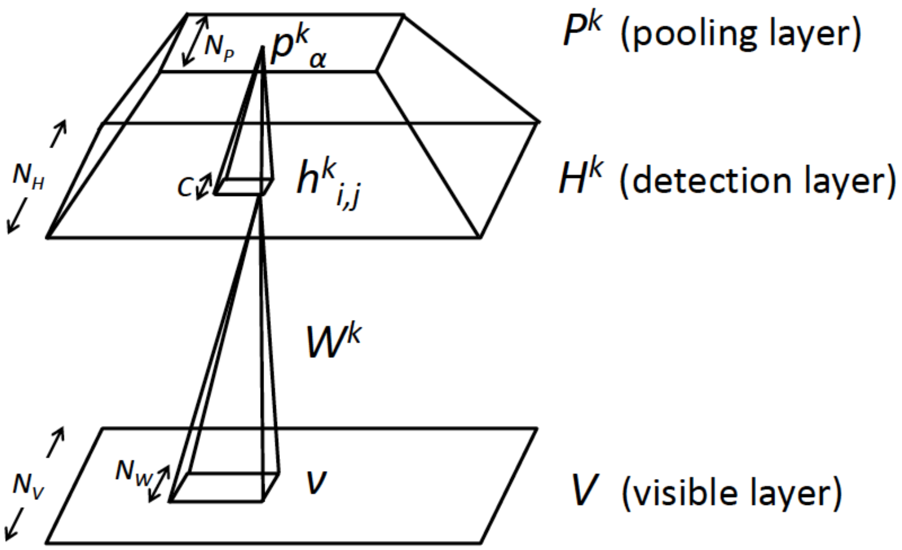
(Jarret et al., 2009)

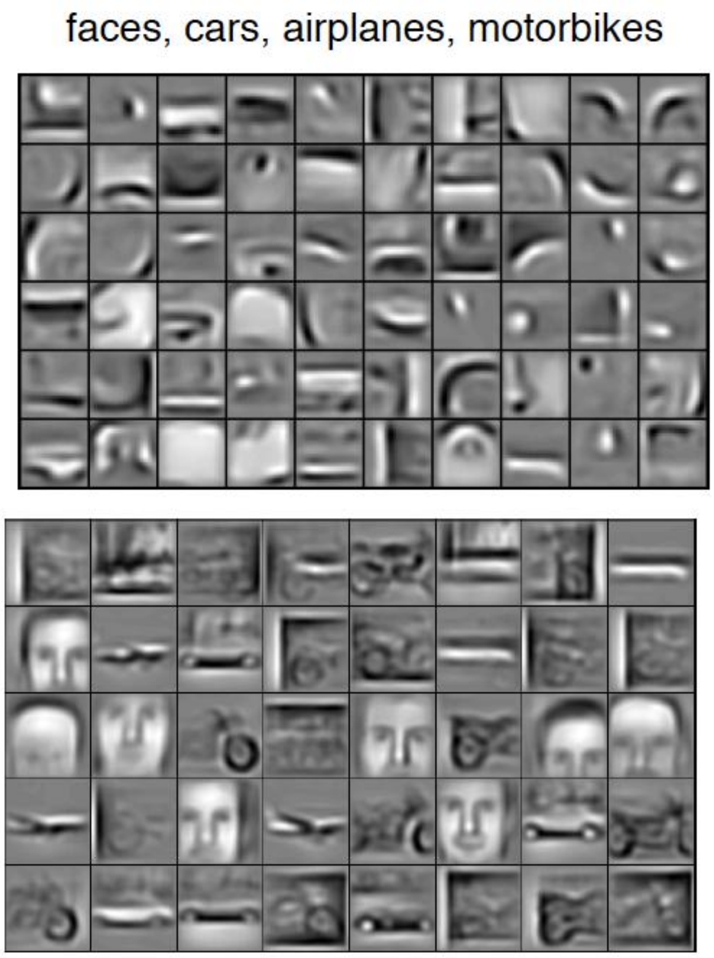
- Details of feature processing stage for PSD



Convolutional DBNs

Convolutional RBM: Generative training of convolutional structures (with probabilistic max-pooling)



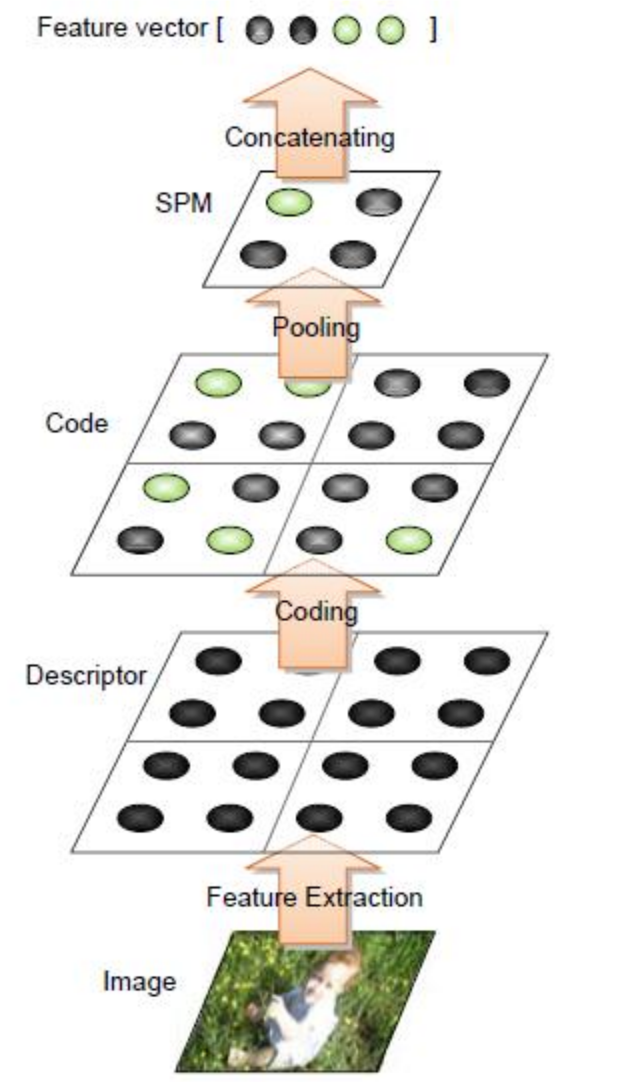


(Lee et al, 2009; Desjardins and Bengio, 2008; Norouzi et al., 2009)

Spatial Pyramid Structure

(Yang et al., 2009)

- **Descriptor Layer:** detect and locate features, extract corresponding descriptors (e.g. SIFT)
- **Code Layer:** code the descriptors
  - Vector Quantization (VQ): each code has only one non-zero element
  - Soft-VQ: small group of elements can be non-zero
- **SPM layer:** pool codes across subregions and average/normalize into a histogram

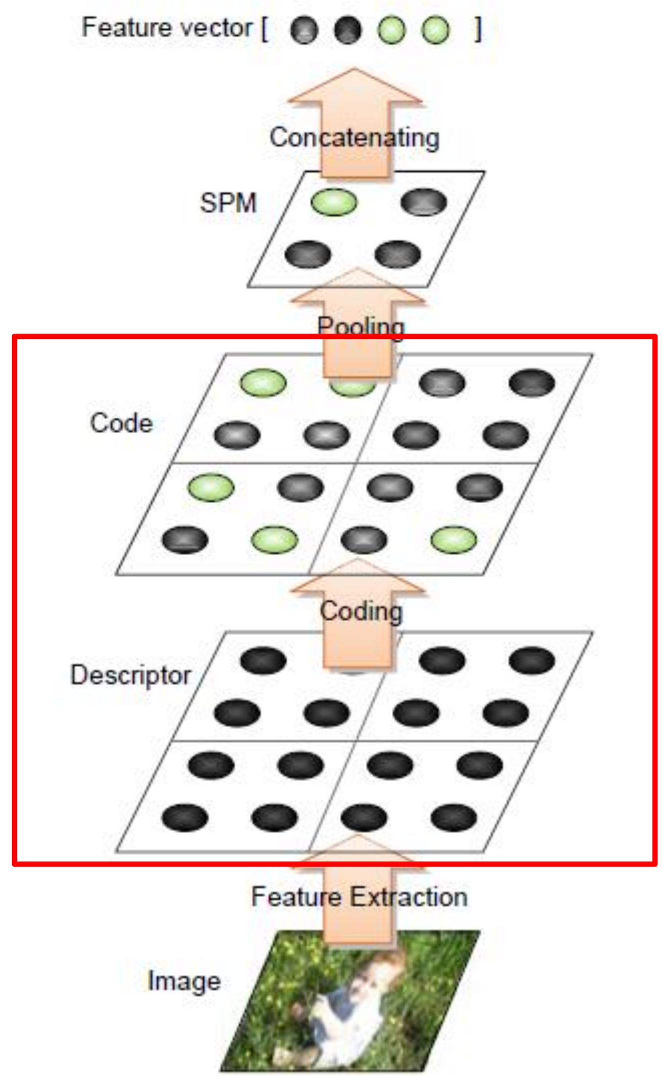


Improving the coding step

(Yang et al., 2009)

- Classifiers using these features need nonlinear kernels
  - Increases computational complexity

- Modify the Coding step to produce feature representations that linear classifiers can use effectively
  - Sparse coding
  - Local Coordinate coding



Experimental results

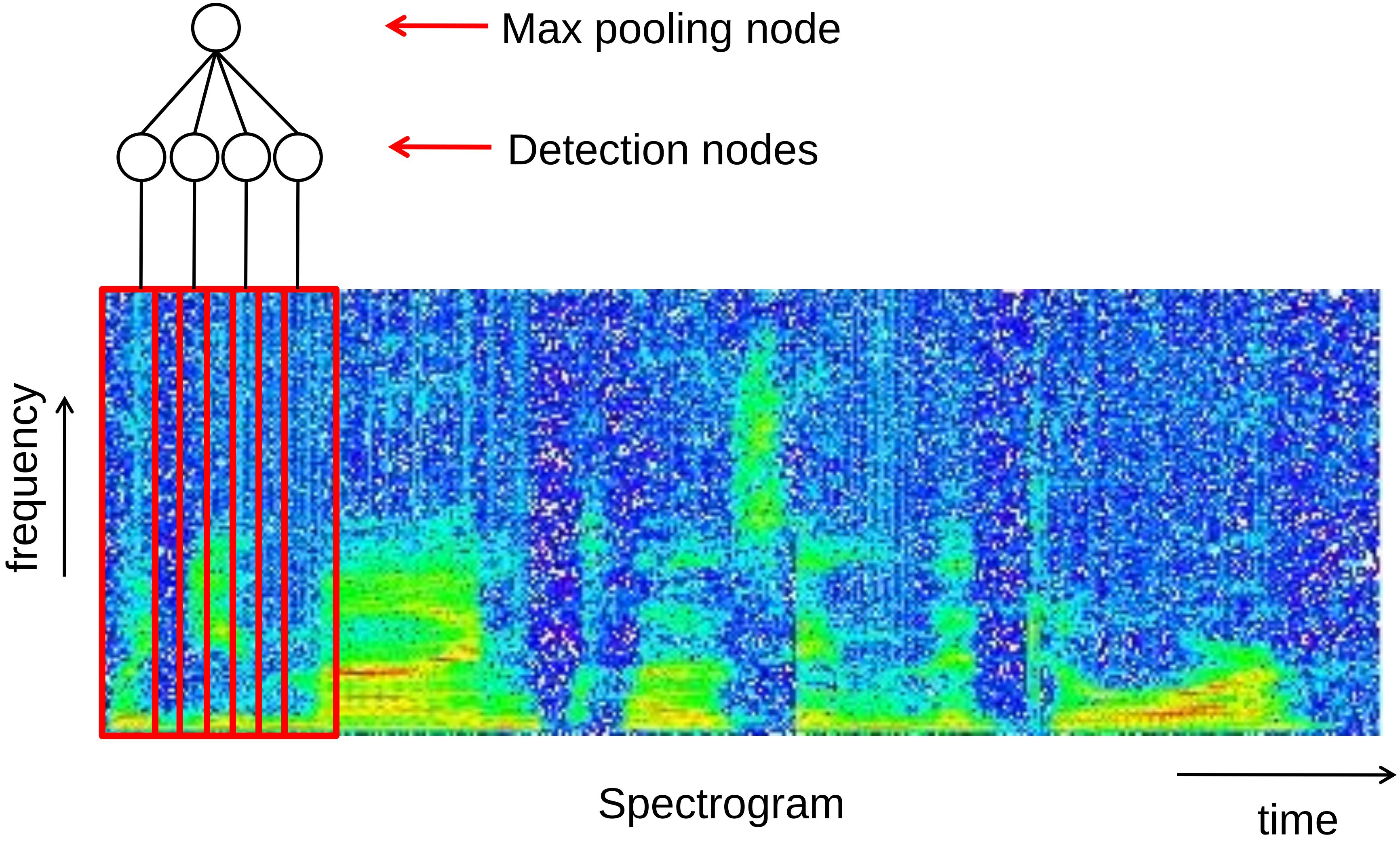
- Competitive performance to other state-of-the-art methods using a single type of features on object recognition benchmarks
- E.g.: Caltech 101 (30 examples per class)
  – Using pixel representation: ~65% accuracy (Jarret et al., 2009; Lee et al., 2009; and many others)
  – Using SIFT representation: 73~75% accuracy (Yang et al., 2009; Jarret et al., 2009, Boureau et al., 2010, and many others)

Outline

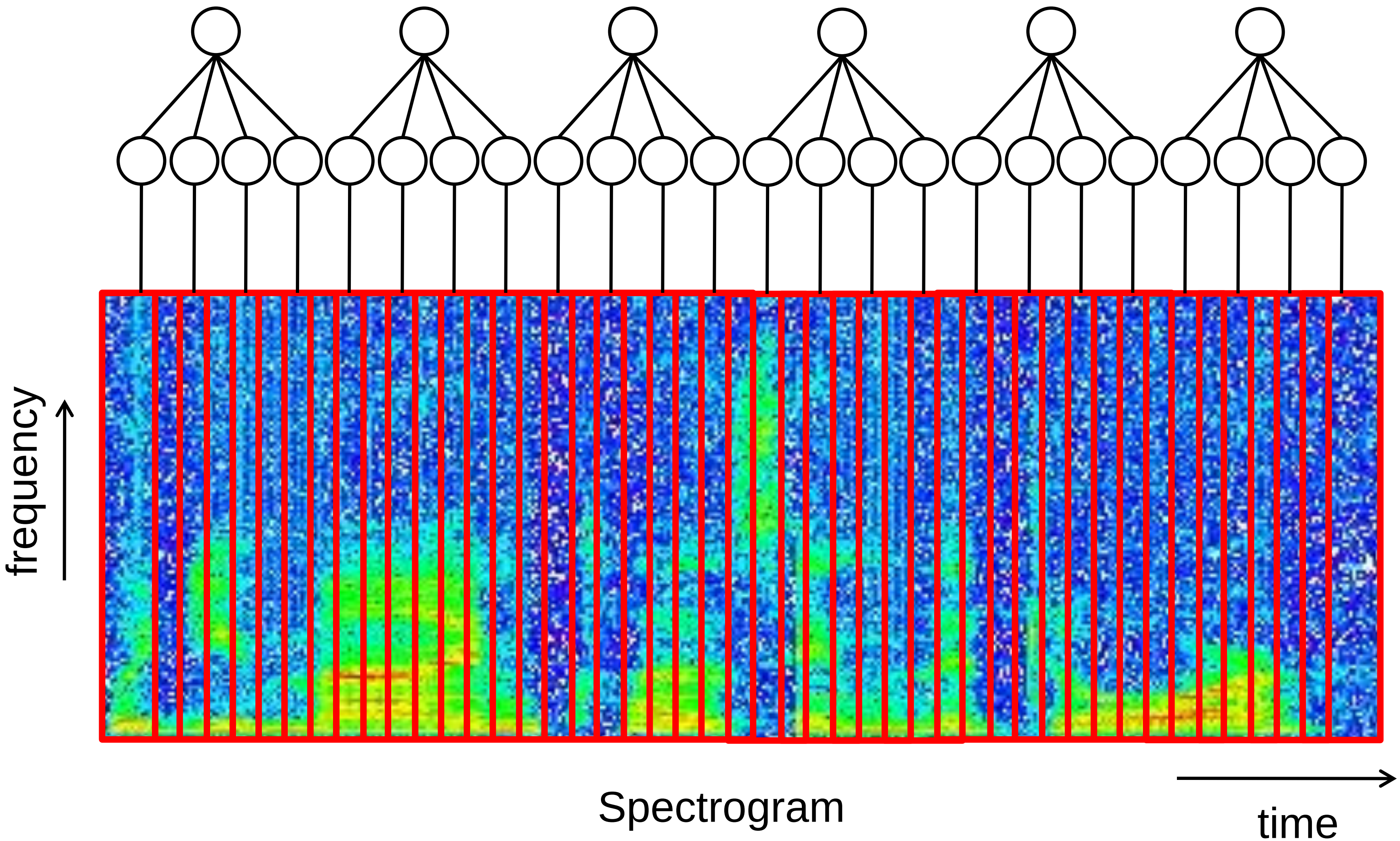
- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

Convolutional DBN for audio

(Lee et al., 2009)

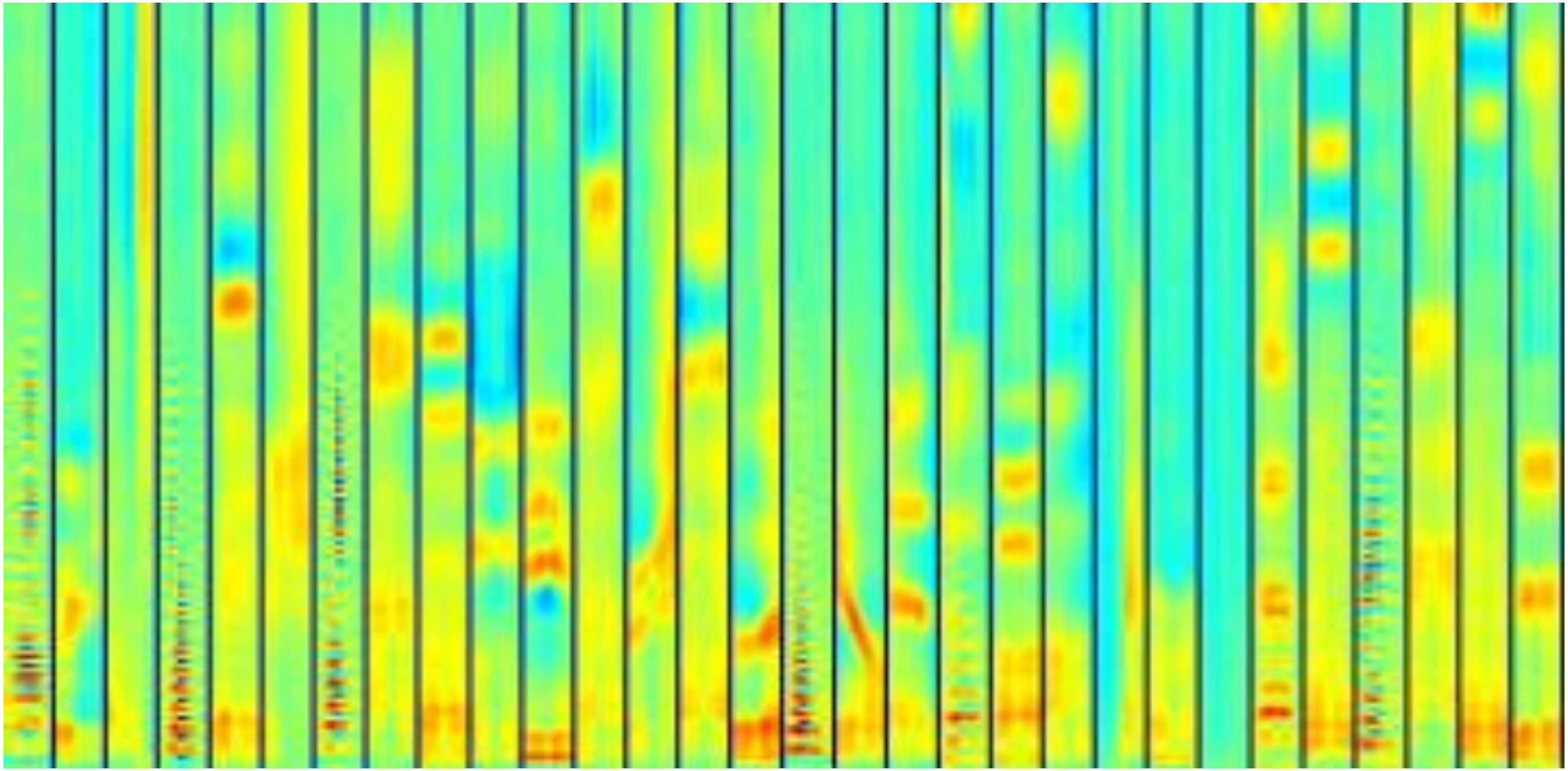


Convolutional DBN for audio



CDBNs for speech

Trained on unlabeled TIMIT corpus



Learned first-layer bases

Experimental Results

(Lee et al., 2009)

- Speaker identification

TIMIT Speaker identification	Accuracy
Prior art (Reynolds, 1995)	99.7%
Convolutional DBN	**100.0%**

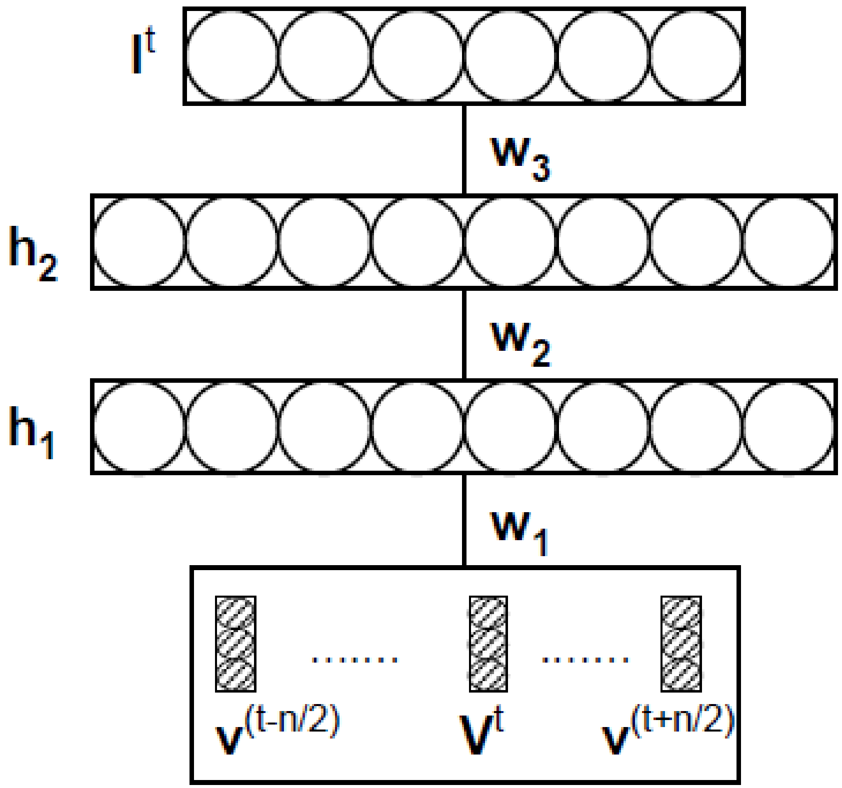
- Phone classification

TIMIT Phone classification	Accuracy
Clarkson et al. (1999)	77.6%
Gunawardana et al. (2005)	78.3%
Sung et al. (2007)	78.5%
Petrov et al. (2007)	78.6%
Sha & Saul (2006)	78.9%
Yu et al. (2009)	79.2%
Convolutional DBN	**80.3%**

Phone recognition using DBNs

(Dahl et al., 2010)

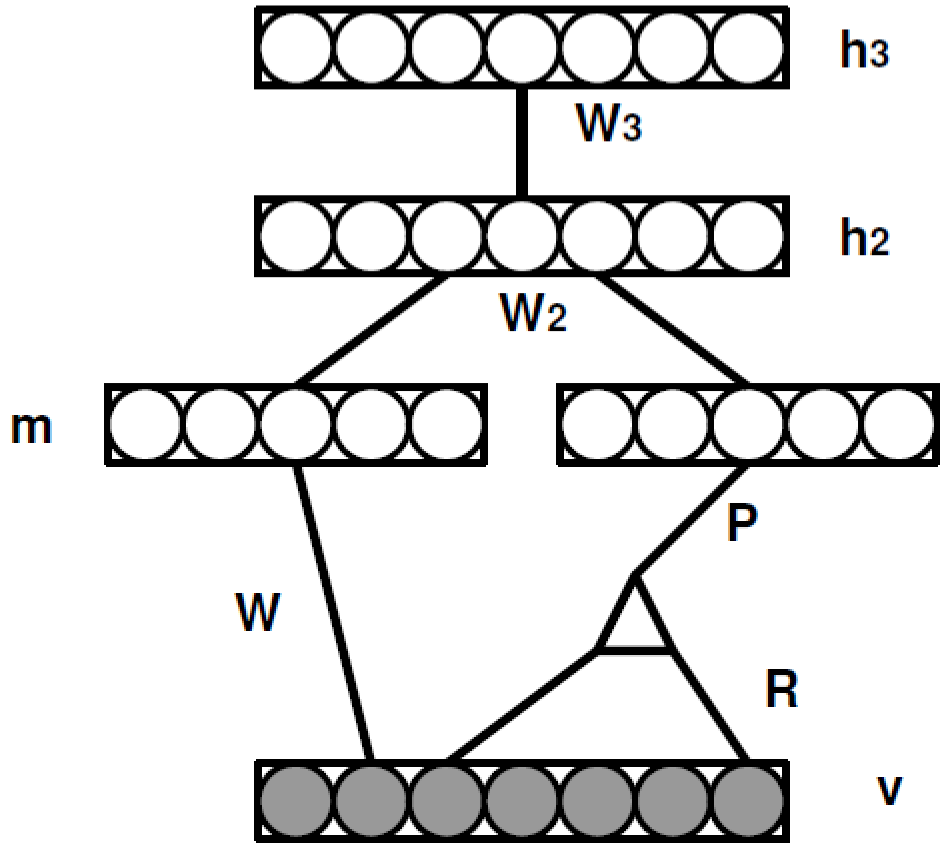
- Pre-training RBMs followed by fine-tuning with back propagation



Phone recognition using mcRBM

(Dahl et al., 2010)

- Mean-covariance RBM + DBN



Mean-covariance RBM

$$E(\mathbf{v},\mathbf{h}) =$$

$$-\mathbf{d}^{\mathrm{T}}\mathbf{h} - (\mathbf{v}^{\mathrm{T}}\mathbf{R})^2\mathbf{P}\mathbf{h},$$

Phone recognition results

(Dahl et al., 2010)

Method	PER
Stochastic Segmental Models	36.0%
Conditional Random Field	34.8%
Large-Margin GMM	33.0%
CD-HMM	27.3%
Augmented conditional Random Fields	26.6%
Recurrent Neural Nets	26.1%
Bayesian Triphone HMM	25.6%
Monophone HTMs	24.8%
Heterogeneous Classifiers	24.4%
Deep Belief Networks(DBNs)	**23.0%**
Triphone HMMs discriminatively trained w/ BMMI	22.7%
Deep Belief Networks with mcRBM feature extraction	**20.5%**

Outline

- Deep learning
  - Greedy layer-wise training (for supervised learning)
  - Deep belief nets
  - Stacked denoising auto-encoders
  - Stacked predictive sparse coding
  - Deep Boltzmann machines
- Applications
  - Vision
  - Audio
  - Language

Language modeling

- Language Models
  - Estimating the probability of the next word w given a sequence of words
- Baseline approach in NLP
  - N-gram models (with smoothing):

$$P(w_i|w_{i-(n-1)},\ldots,w_{i-1}) = \frac{count(w_{i-(n-1)},\ldots,w_{i-1},w_i)}{count(w_{i-(n-1)},\ldots,w_{i-1})}$$

- Deep Learning approach
  - Bengio et al. (2000, 2003): via Neural network
  - Mnih and Hinton (2007): via RBMs

Other NLP tasks

(Collobert and Weston, 2009)

- Part-Of-Speech Tagging (POS)
  - mark up the words in a text (corpus) as corresponding to a particular tag
    - E.g. Noun, adverb, ...
- Chunking
  - Also called shallow parsing
  - In the view of phrase: Labeling phrase to syntactic constituents
    - E.g. NP (noun phrase), VP (verb phrase), …
  - In the view of word: Labeling word to syntactic role in a phrase
    - E.g. B-NP (beginning of NP), I-VP (inside VP), …

Other NLP tasks

(Collobert and Weston, 2009)

- Named Entity Recognition (NER)
  - In the view of thought group: Given a stream of text, determine which items in the text map to proper names
  - E.g., labeling "atomic elements" into "PERSON", "COMPANY", "LOCATION"
- Semantic Role Labeling (SRL)
  - In the view of sentence: giving a semantic role to a syntactic constituent of a sentence
  - E.g. [John]$_{ARG0}$ [ate]$_{REL}$ [the apple]$_{ARG1}$ (Proposition Bank)
    - An Annotated Corpus of Semantic Roles (Palmer et al.)

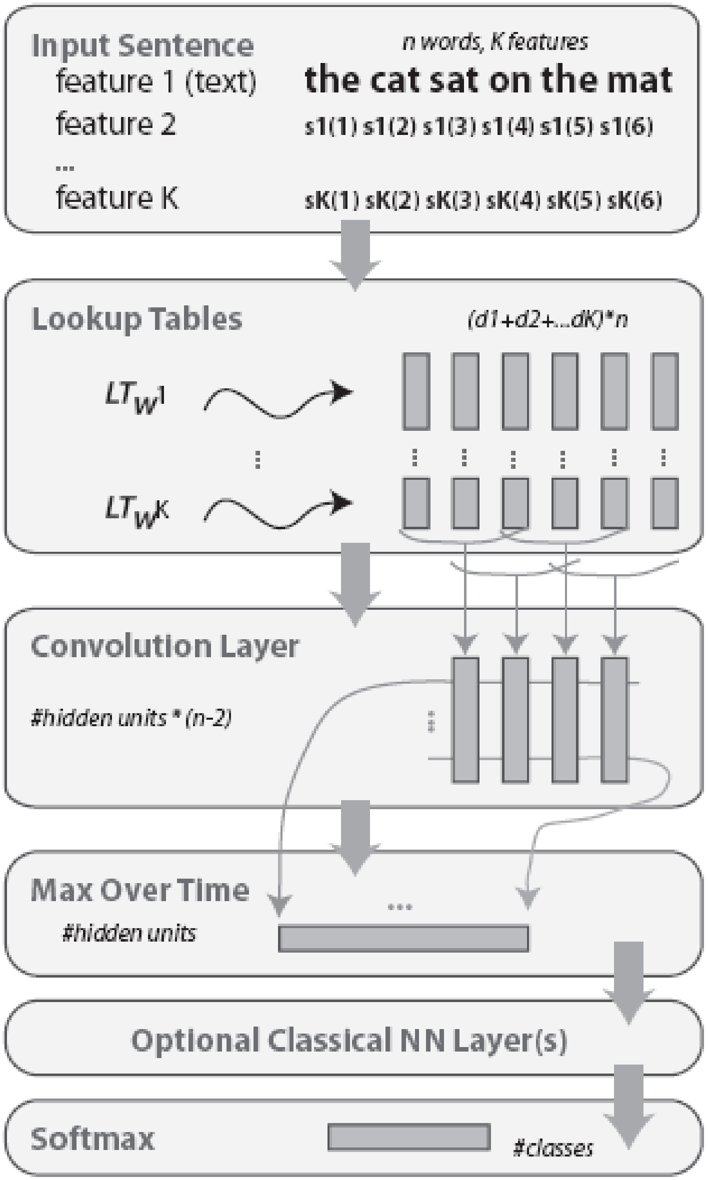
A unified architecture for NLP

(Collobert and Weston, 2009)

- Main idea: a unified architecture for NLP
  - Deep Neural Network
  - Trained jointly with different tasks (feature sharing and multi-task learning)
  - Language model is trained in an unsupervised fashion
- Show the generality of the architecture
- Improve SRL performance

General Deep Architecture for NLP

(Collobert and Weston, 2009)



Basic features (e.g., word, capitalization, relative position)

Embedding by lookup table

Convolution (i.e., how each word is relevant to its context?)

Max pooling

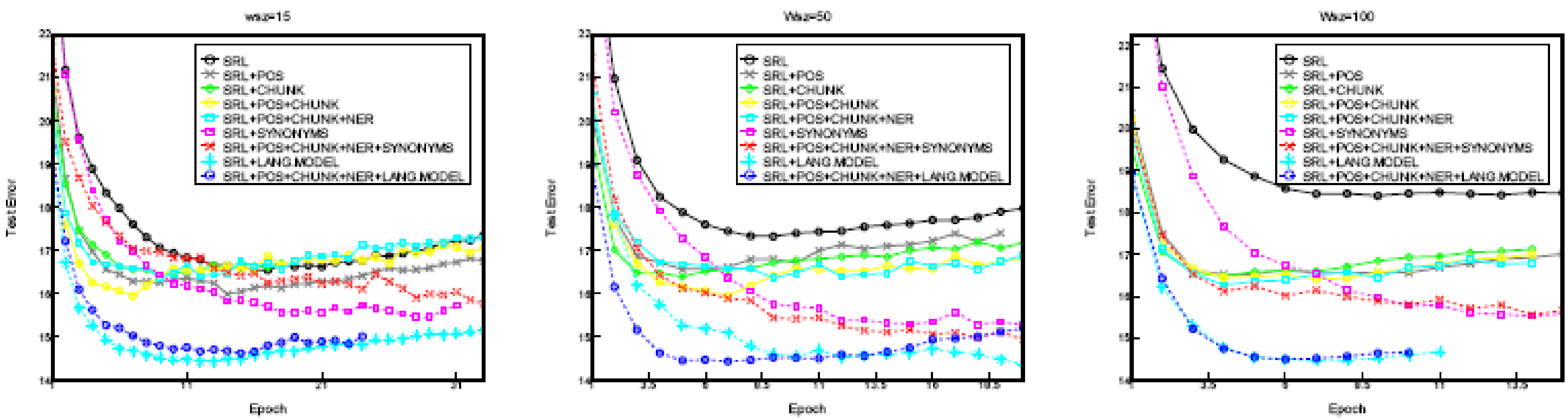
Supervised learning

Results

(Collobert and Weston, 2009)

- MTL improves SRL's performance

	wsz=15	wsz=50	wsz=100
SRL	16.54	17.33	18.40
SRL + POS	15.99	16.57	16.53
SRL + Chunking	16.42	16.39	16.48
SRL + NER	16.67	17.29	17.21
SRL + Synonyms	15.46	15.17	15.17
SRL + Language model	14.42	14.30	14.46
SRL + POS + Chunking	16.46	15.95	16.41
SRL + POS + NER	16.45	16.89	16.29
SRL + POS + Chunking + NER	16.33	16.36	16.27
SRL + POS + Chunking + NER + Synonyms	15.71	14.76	15.48
SRL + POS + Chunking + NER + Language model	14.63	14.44	14.50



Summary

- Training deep architectures
  – Unsupervised pre-training helps training deep networks
  – Deep belief nets, Stacked denoising auto-encoders, Stacked predictive sparse coding, Deep Boltzmann machines
- Deep learning algorithms and unsupervised feature learning algorithms show promising results in many applications
  – vision, audio, natural language processing, etc.

Thank you!

References

- B. Olshausen, D. Field. Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images. Nature, 1996.
- H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. NIPS, 2007.
- R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: Transfer learning from unlabeled data. ICML, 2007.
- H. Lee, R. Raina, A. Teichman, and A. Y. Ng. Exponential Family Sparse Coding with Application to Self-taught Learning. IJCAI, 2009.
- J. Yang, K. Yu, Y. Gong, and T. Huang. Linear Spatial Pyramid Matching Using Sparse Coding for Image Classification. CVPR, 2009.
- Y. Bengio. Learning Deep Architectures for AI, Foundations and Trends in Machine Learning, 2009.
- Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. NIPS, 2007.
- P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol. Extracting and composing robust features with denoising autoencoders. ICML, 2008.

References

- H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area V2. NIPS, 2008.
- Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1:541–551, 1989.
- H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. ICML, 2009.
- H. Lee, Y. Largman, P. Pham, and A. Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. NIPS, 2009.
- A. R. Mohamed, G. Dahl, and G. E. Hinton, Deep belief networks for phone recognition. NIPS 2009 workshop on deep learning for speech recognition.
- G. Dahl, M. Ranzato, A. Mohamed, G. Hinton, Phone Recognition with the Mean-Covariance Restricted Boltzmann Machine, NIPS 2010
- M. Ranzato, A. Krizhevsky, G. E. Hinton, Factored 3-Way Restricted Boltzmann Machines for Modeling Natural Images. AISTATS, 2010.

References

- M. Ranzato, G. E. Hinton. Modeling Pixel Means and Covariances Using Factorized Third-Order Boltzmann Machines. CVPR, 2010.
- G. Taylor, G. E. Hinton, and S. Roweis. Modeling Human Motion Using Binary Latent Variables. NIPS, 2007.
- G. Taylor and G. E. Hinton. Factored Conditional Restricted Boltzmann Machines for Modeling Motion Style. ICML, 2009.
- G. Taylor, R. Fergus, Y. LeCun and C. Bregler. Convolutional Learning of Spatio-temporal Features. ECCV, 2010.
- K. Kavukcuoglu, M. Ranzato, R. Fergus, and Y. LeCun, Learning Invariant Features through Topographic Filter Maps. CVPR, 2009.
- K. Kavukcuoglu, M. Ranzato, and Y. LeCun, Fast Inference in Sparse Coding Algorithms with Applications to Object Recognition. CBLL-TR-2008-12-01, 2008.
- K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, What is the Best Multi-Stage Architecture for Object Recognition? ICML, 2009.
- R. Salakhutdinov and I. Murray. On the Quantitative Analysis of Deep Belief Networks. ICML, 2008.

References

- R. Salakhutdinov and G. E. Hinton. Deep Boltzmann machines. AISTATS, 2009.
- K. Yu, T. Zhang, and Y. Gong. Nonlinear Learning using Local Coordinate Coding, NIPS, 2009.
- J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Learning Locality-constrained Linear Coding for Image Classification. CVPR, 2010.
- H. Larochelle, Y. Bengio, J. Louradour and P. Lamblin, Exploring Strategies for Training Deep Neural Networks, JMLR, 2009.
- D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent and S. Bengio, Why Does Unsupervised Pre-training Help Deep Learning? JMLR, 2010.
- J. Yang, K. Yu, and T. Huang. Supervised Translation-Invariant Sparse Coding. CVPR, 2010.
- Y. Boureau, F. Bach, Y. LeCun and J. Ponce: Learning Mid-Level Features for Recognition. CVPR, 2010.
- I. J. Goodfellow, Q. V. Le, A. M. Saxe, H. Lee, and A. Y. Ng. Measuring invariances in deep networks. NIPS, 2009.

References

- Y. Boureau, J. Ponce, Y. LeCun, A theoretical analysis of feature pooling in vision algorithms. ICML, 2010.
- R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. ICML, 2009.

References

(1) links to code:

- sparse coding
  - http://www.eecs.umich.edu/~honglak/softwares/nips06-sparsecoding.htm
- DBN
  - http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html
- DBM
  - http://web.mit.edu/~rsalakhu/www/DBM.html
- convnet
  - http://cs.nyu.edu/~koray/publis/code/eblearn.tar.gz
  - http://cs.nyu.edu/~koray/publis/code/randomc101.tar.gz

(2) link to general website on deeplearning:

  - http://deeplearning.net