Deep Learning

LxMLS 2015

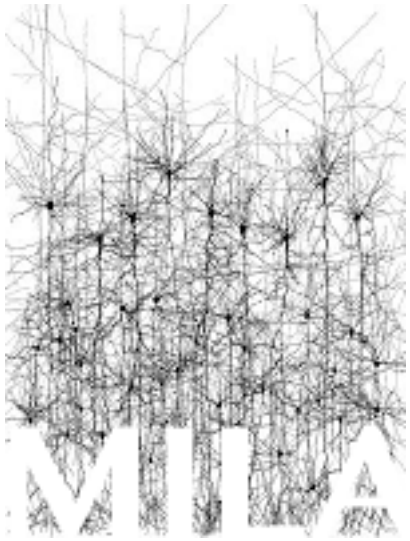
Lisbon Machine Learning Summer School

Lisbon, Portugal

Yoshua Bengio

July 23, 2015

Université de Montréal



Outline of the Tutorial

1. Representation Learning, motivations
2. Why does deep learning work so well?
3. Algorithms: backprop, convnets, RNNs
4. Unsupervised & generative learning
5. Attention mechanisms

Upcoming MIT Press book: "Deep Learning" http://www.iro.umontreal.ca/~bengioy/dlbook/
for draft chapters (in preparation)

Breakthrough

- **Deep Learning: machine learning algorithms based on learning multiple levels of representation / abstraction.**

Amazing improvements in error rate in object recognition, object detection, speech recognition, and more recently, in natural language processing / understanding

Ongoing Progress: Combining Vision and Natural Language Understanding

- Recurrent nets generating credible sentences, even better if conditionally:
  - Machine translation
  - Image 2 text

Xu et al, ICML'2015

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

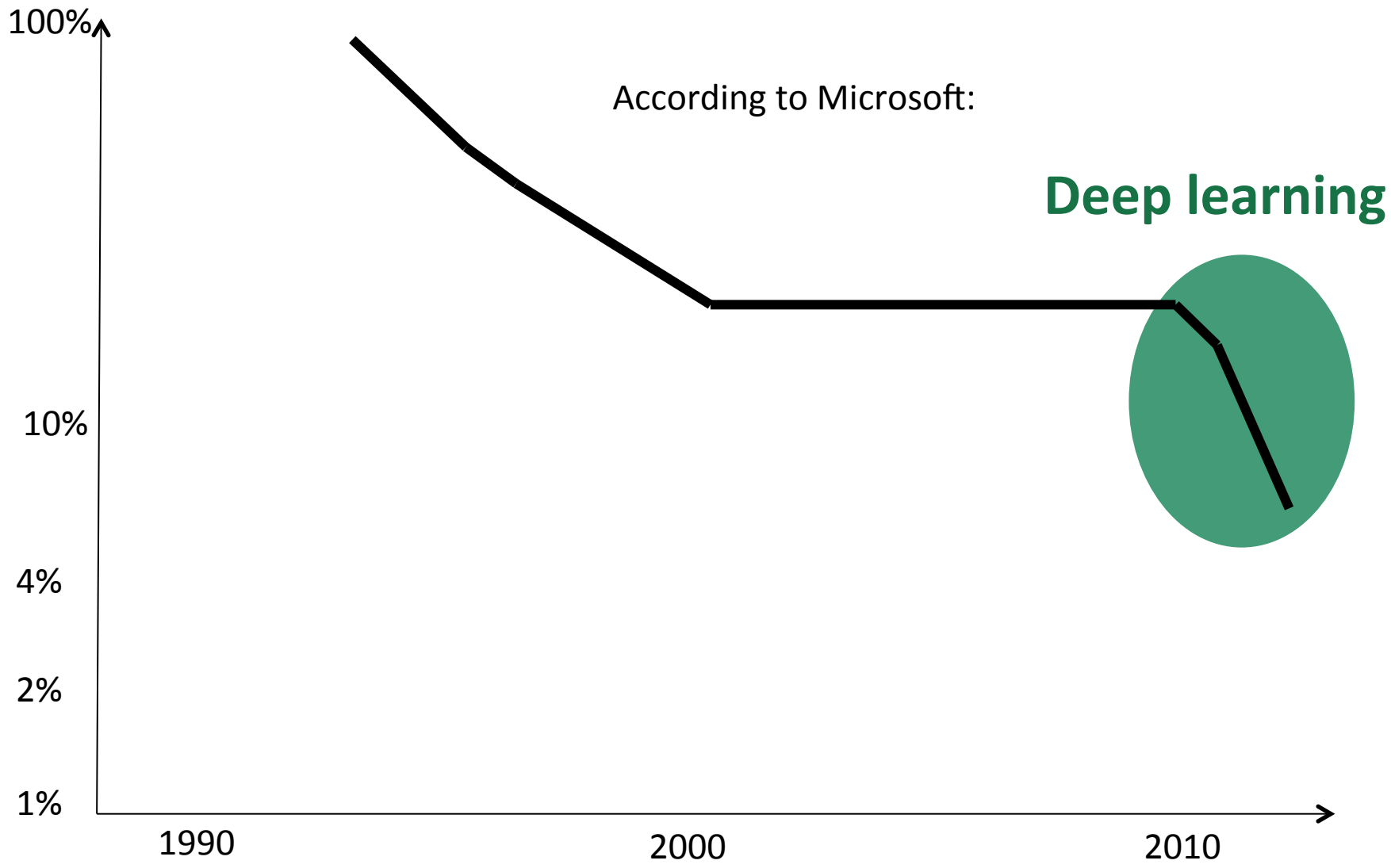
Initial Breakthrough in 2006

Canadian initiative: CIFAR

- Ability to train deep architectures by using layer-wise unsupervised learning, whereas previous purely supervised attempts had failed

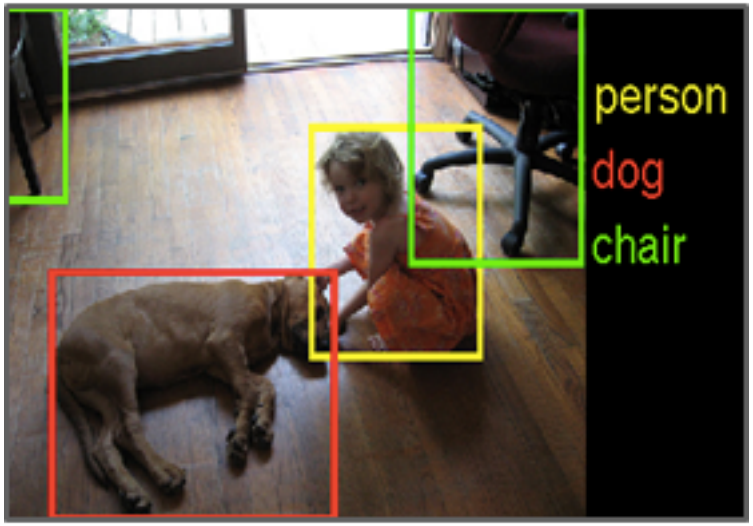
- Unsupervised feature learners:
  - RBMs
  - Auto-encoder variants
  - Sparse coding variants

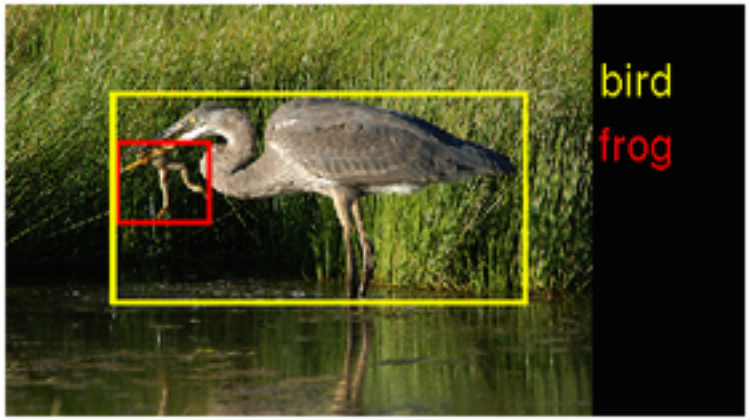
2010-2012: Breakthrough in speech recognition → in Androids by 2012



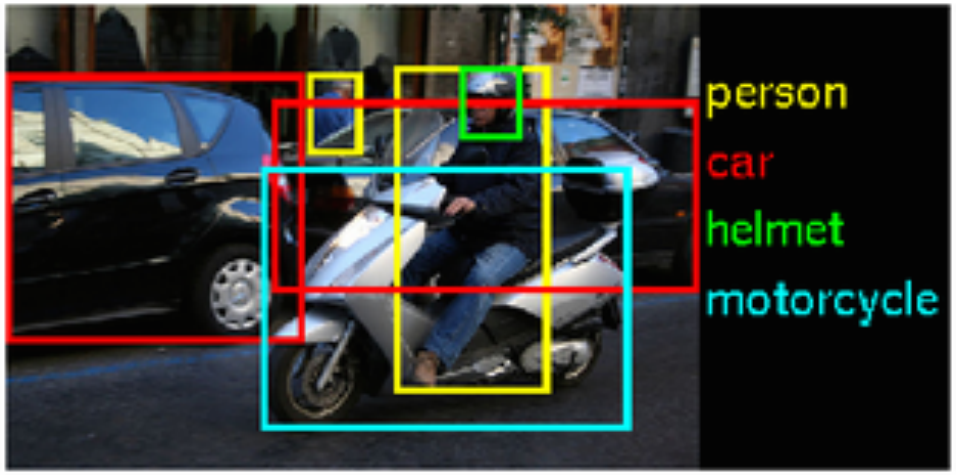
Breakthrough in computer vision: 2012-2015

- GPUs + 10x more data





- 1000 object categories,
- Facebook: millions of faces
- 2015: human-level performance



Deep Learning in the News

The New York Tim

Monday, June 25, 2012 Last Update: 11:50 PM ET

EXCLUSIVE

Facebook, Google in 'Deep Learning' Arms Race

Yann LeCun, an NYU artificial intelligence researcher who now works for Facebook. *Photo: Josh Valcarcel/WIRED*

WIRED

NEWS BULLETIN

Google Beat Facebook for DeepMind

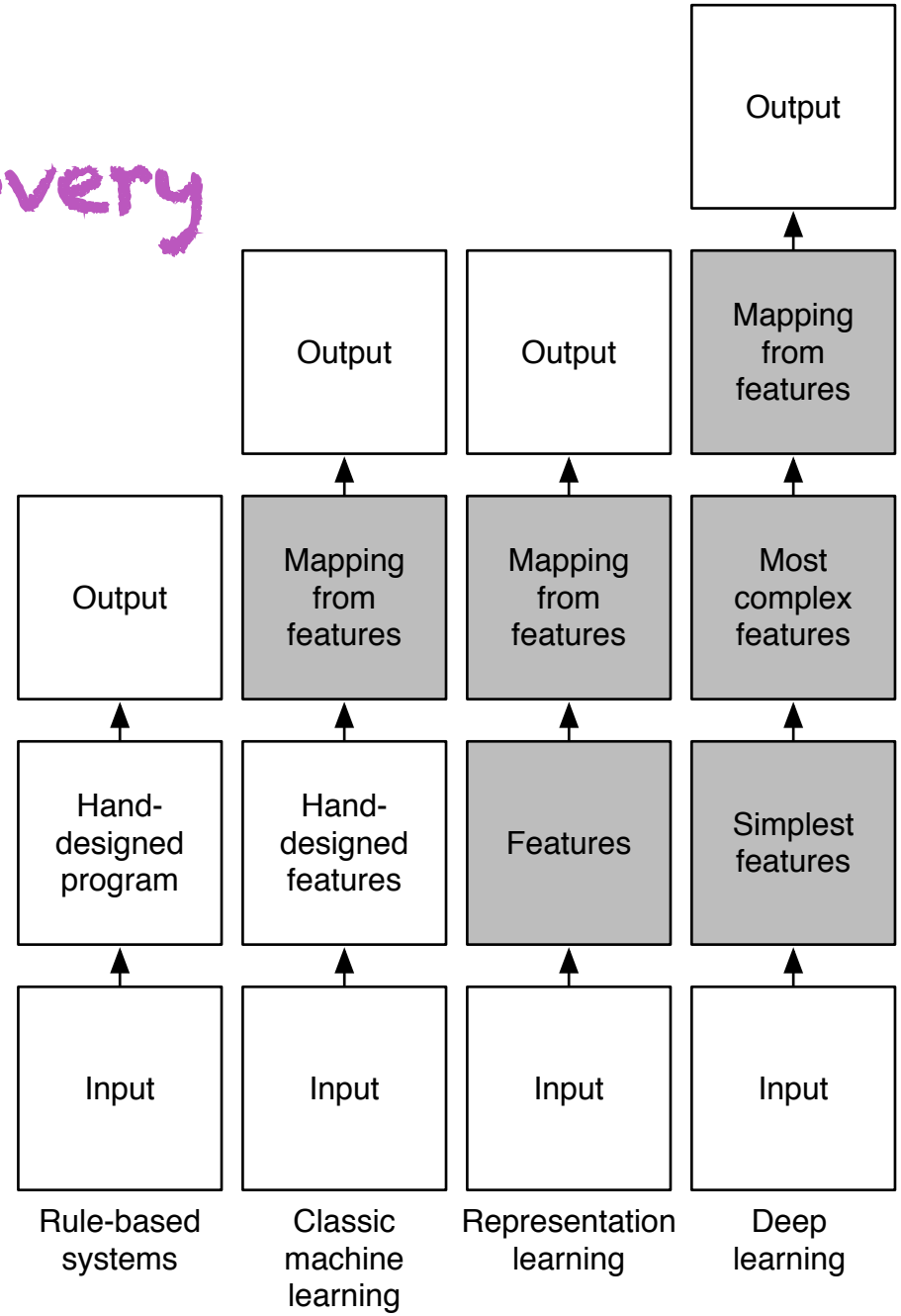
Google Acquires Artificial Intelligence Startup DeepMind For More Than $500M

Posted Jan 26, 2014 by Catherine Shu (@catherineshu)

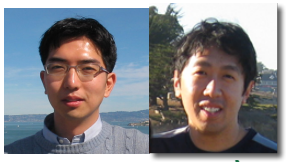
IT Companies are Racing into Deep Learning

Why is deep learning working so well?

Automating Feature Discovery

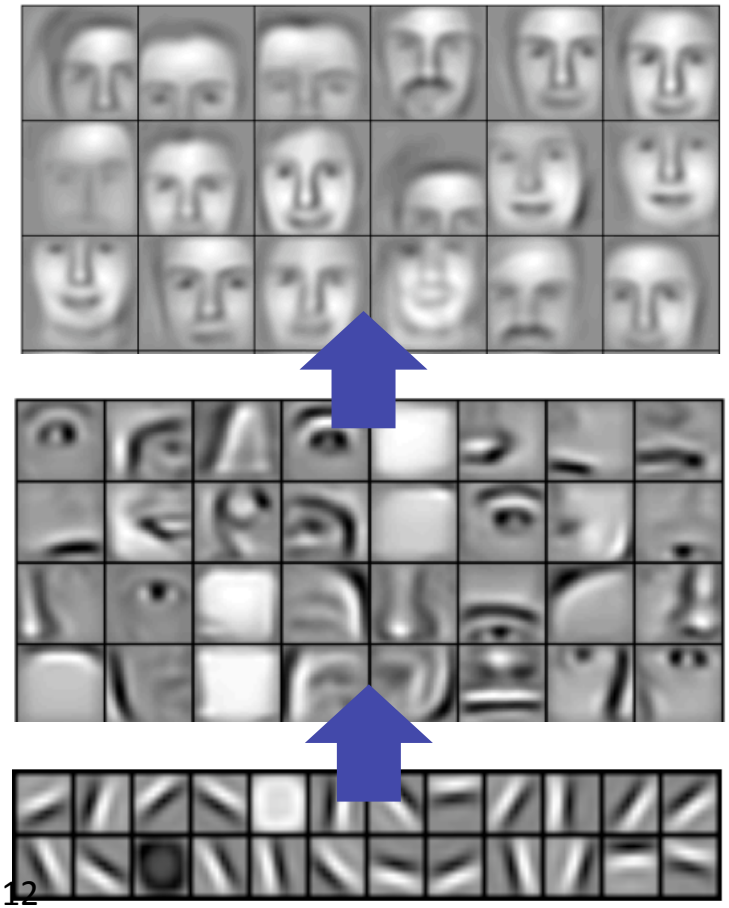


Learning multiple levels of representation



(Lee, Largman, Pham & Ng, NIPS 2009)
(Lee, Grosse, Ranganath & Ng, ICML 2009)

Successive model layers learn deeper intermediate representations

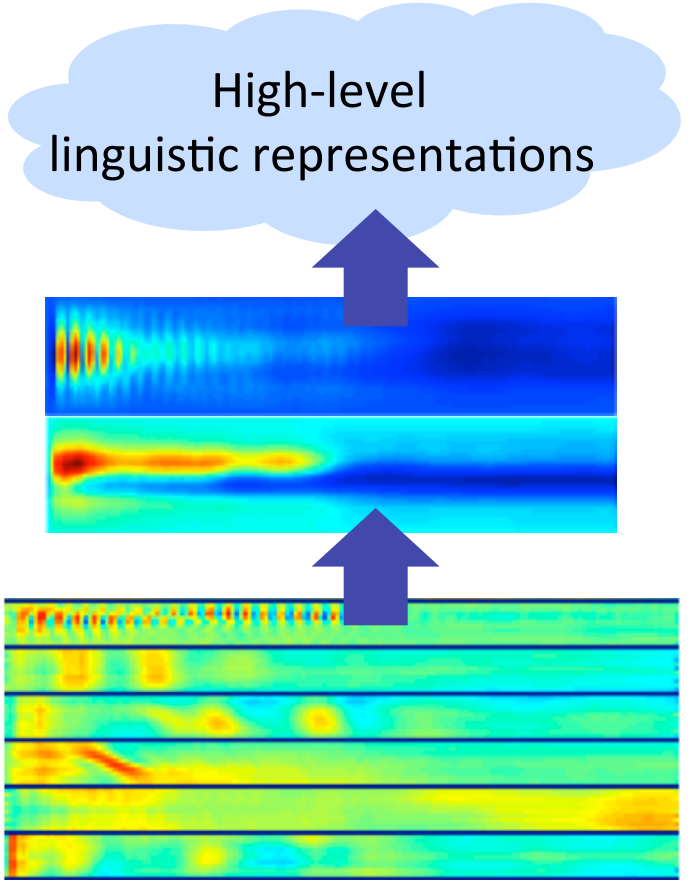


Layer 3

Parts combine to form objects

Layer 2

Layer 1



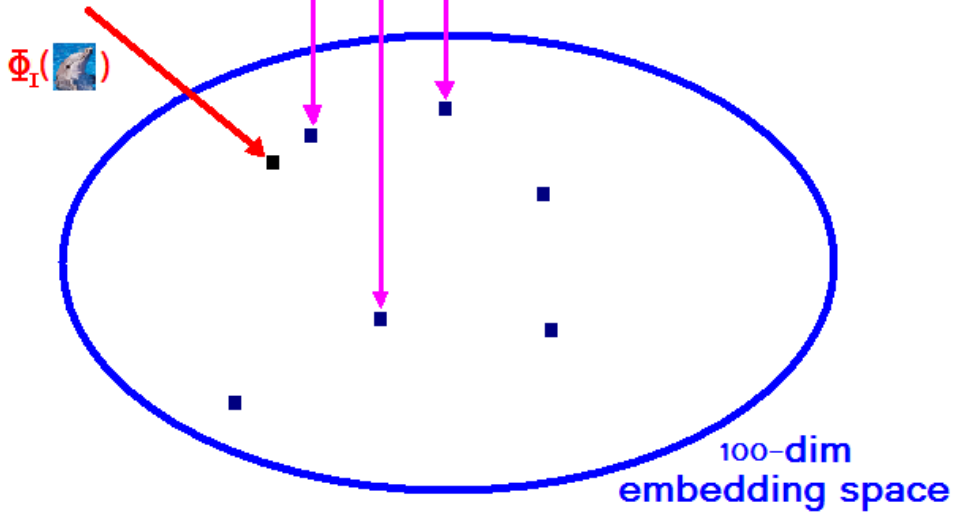
Prior: underlying factors & concepts compactly expressed w/ multiple levels of abstraction

Google Image Search:
Different object types represented in the same space

Google:

S. Bengio, J. Weston & N. Usunier

(IJCAI 2011, NIPS'2010, JMLR 2010, MLJ 2010)



Learn $\Phi_I(\cdot)$ and $\Phi_W(\cdot)$ to optimize precision@k.

Machine Learning, AI & No Free Lunch

- Three key ingredients for ML towards AI

  1. Lots & lots of data

  2. Very flexible models

  3. Powerful priors that can defeat the curse of dimensionality

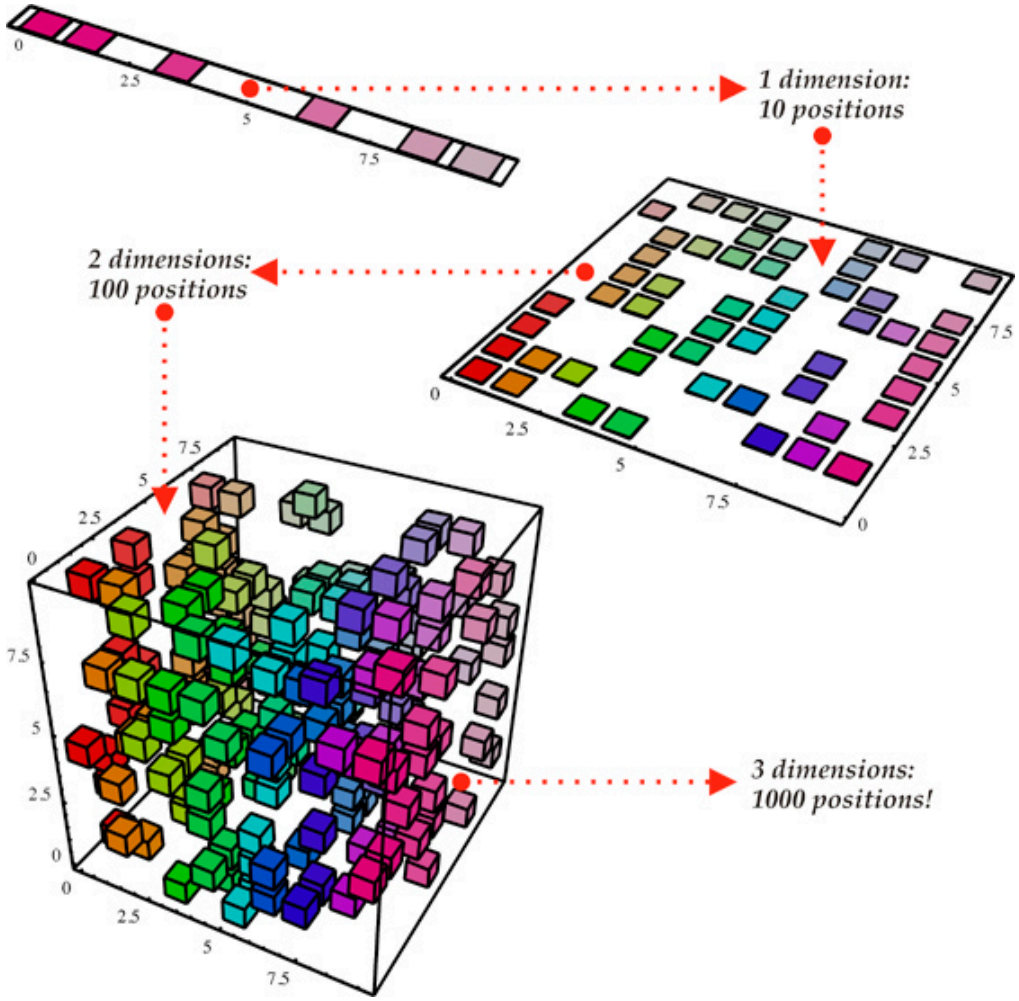
Ultimate Goals

- **AI**
- Needs **knowledge**
- Needs **learning**
  (involves priors + ***optimization****/search*)
- Needs **generalization**
  (guessing where probability mass concentrates)
- Needs ways to fight the curse of dimensionality
  (exponentially many configurations of the variables to consider)
- Needs disentangling the underlying explanatory factors
  (making sense of the data)

ML 101. What We Are Fighting Against: The Curse of Dimensionality

To generalize locally, need representative examples for all relevant variations!

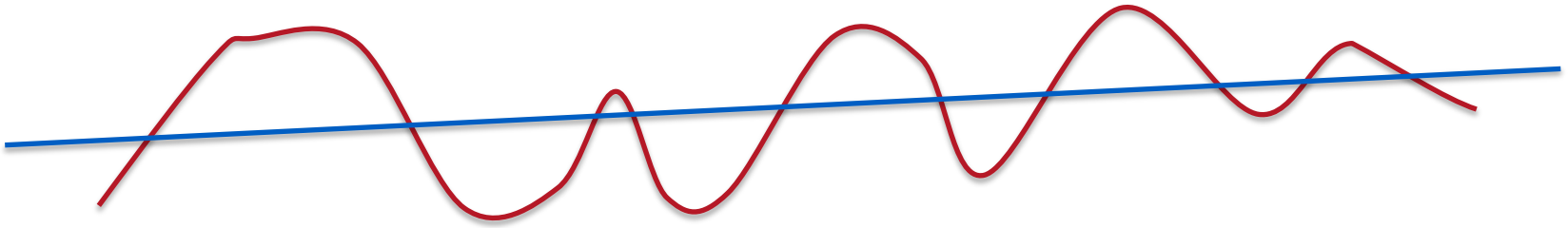
Classical solution: hope for a smooth enough target function, or make it smooth by handcrafting good features / kernel



Not Dimensionality so much as Number of Variations

(Bengio, Dellalleau & Le Roux 2007)

- Theorem: Gaussian kernel machines need at least *k* examples to learn a function that has *2k* zero-crossings along some line



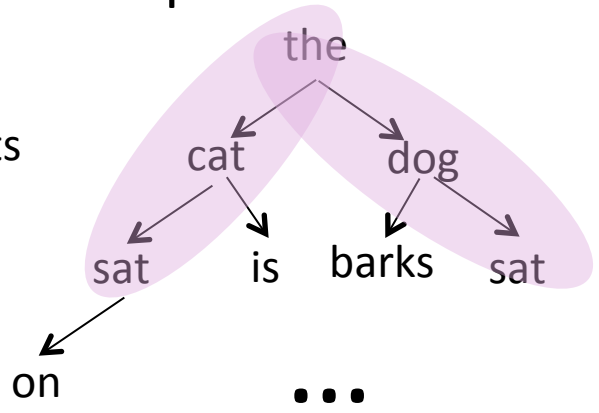
- Theorem: For a Gaussian kernel machine to learn some maximally varying functions over *d* inputs requires $O(2^d)$ examples

Why N-grams have poor generalization

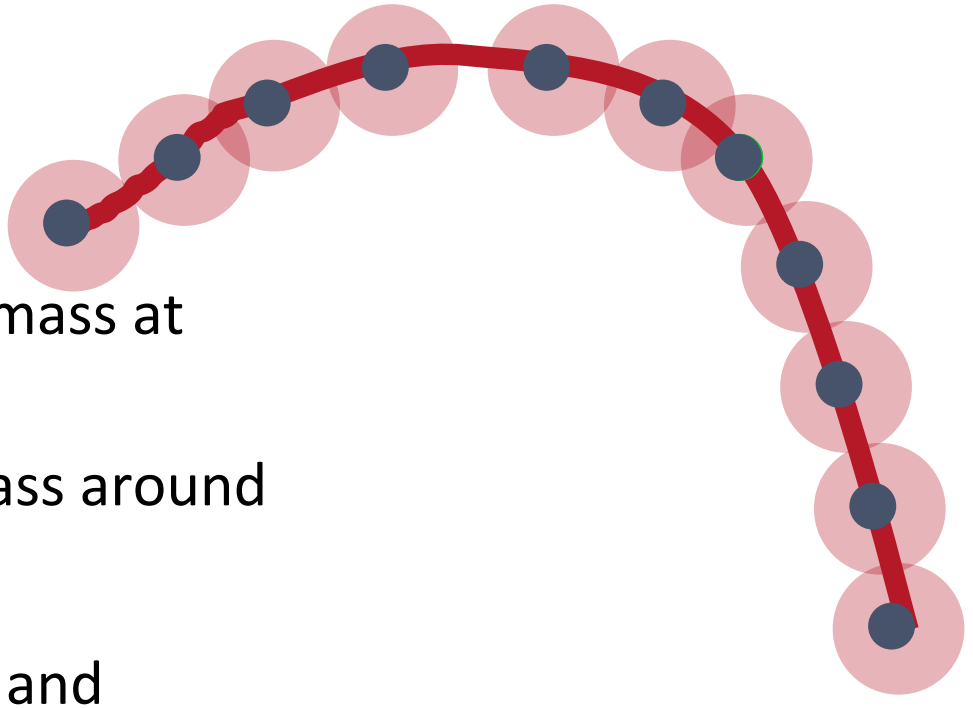
- For fixed N, the function *P(next word | last N-1 words)* is learned purely from the instances of the specific N-tuples associated with each possible (N-1)-word context. No generalization to other sequences of N words and no cross-generalization between different N-tuples!

- With back-off / smoothing models, there is some (limited) generalization arising from shorter n-grams, for which there is more data, at the price of less specific predictions.

No sharing, where lots would be possible



Putting Probability Mass where Structure is Plausible



- Empirical distribution: mass at training examples
- Smoothness: spread mass around
- Insufficient
- Guess some 'structure' and generalize accordingly

Bypassing the curse of dimensionality

We need to build compositionality into our ML models

Just as human languages exploit compositionality to give representations and meanings to complex ideas

Exploiting compositionality gives an exponential gain in representational power

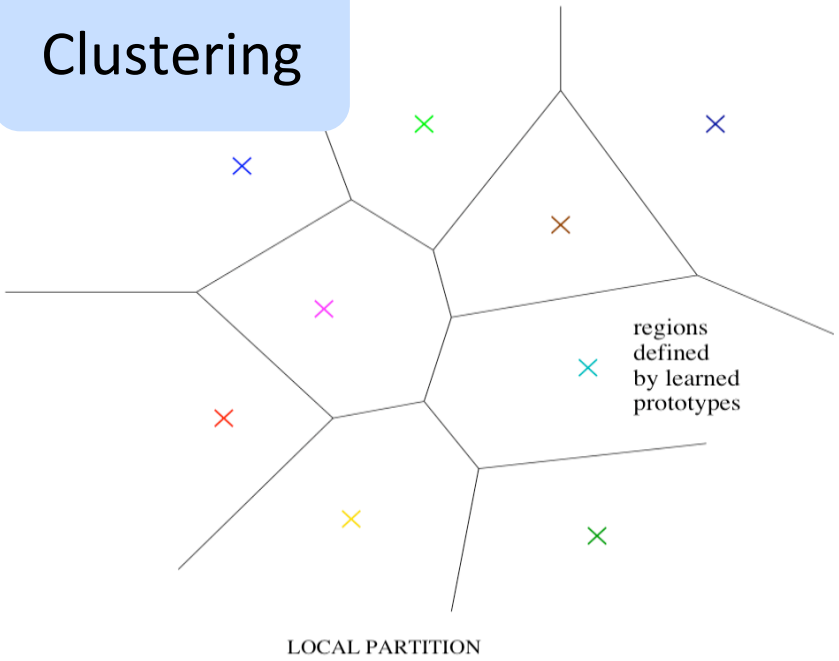
Distributed representations / embeddings: feature learning

Deep architecture: multiple levels of feature learning

Prior: compositionality is useful to describe the world around us efficiently

Non-distributed representations

Clustering



- Clustering, n-grams, Nearest-Neighbors, RBF SVMs, local non-parametric density estimation & prediction, decision trees, etc.

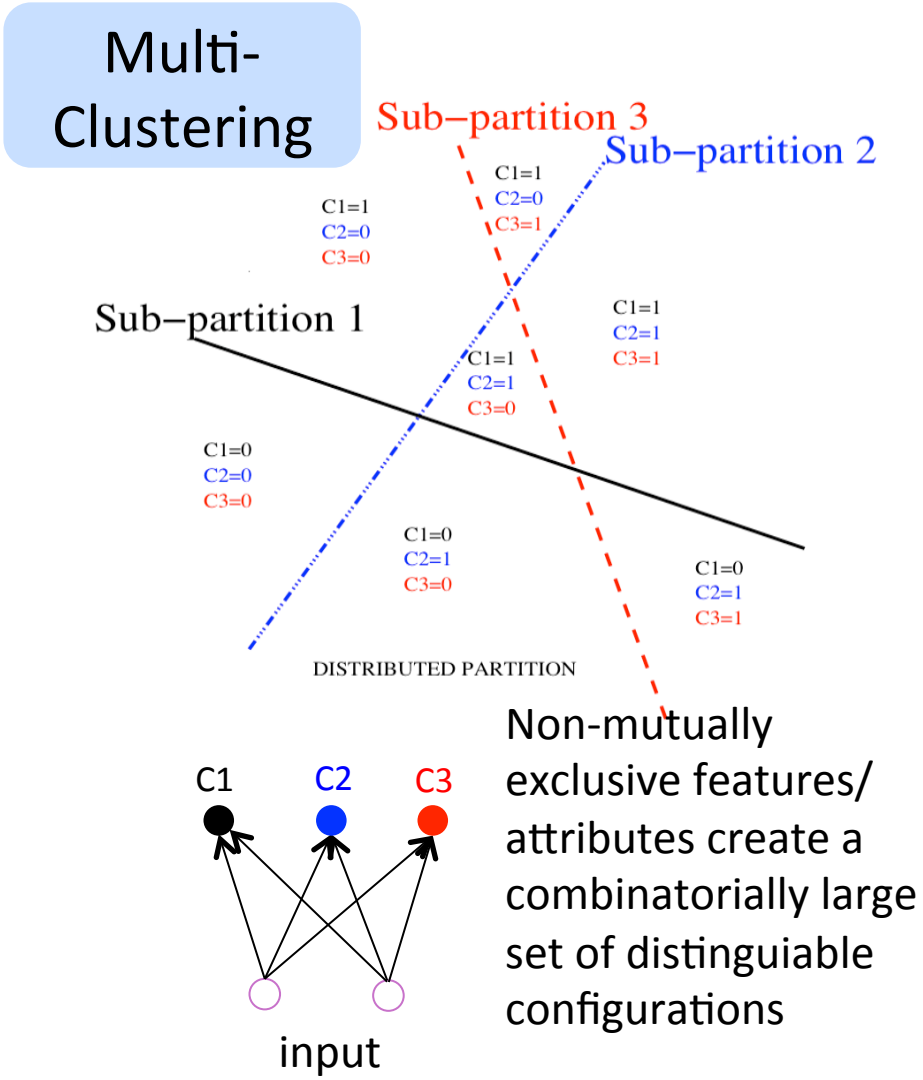
- Parameters for each distinguishable region
- **# of distinguishable regions is linear in # of parameters**

→ No non-trivial generalization to regions without examples

The need for distributed representations

- Factor models, PCA, RBMs, Neural Nets, Sparse Coding, Deep Learning, etc.

- Each parameter influences many regions, not just local neighbors
- **# of distinguishable regions grows almost exponentially with # of parameters**
- **GENERALIZE NON-LOCALLY TO NEVER-SEEN REGIONS**



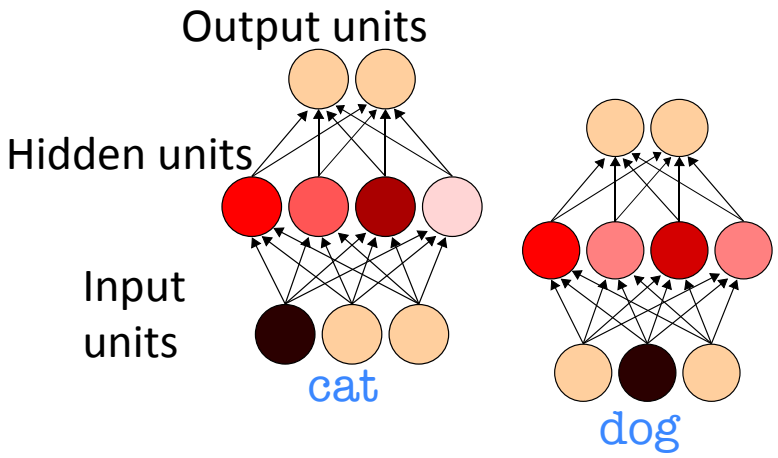
Classical Symbolic AI vs Representation Learning

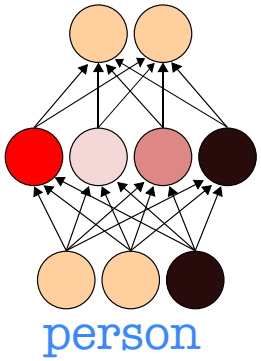
- Two symbols are equally far from each other
- Concepts are not represented by symbols in our brain, but by patterns of activation

  (Connectionism, 1980's)


Geoffrey Hinton

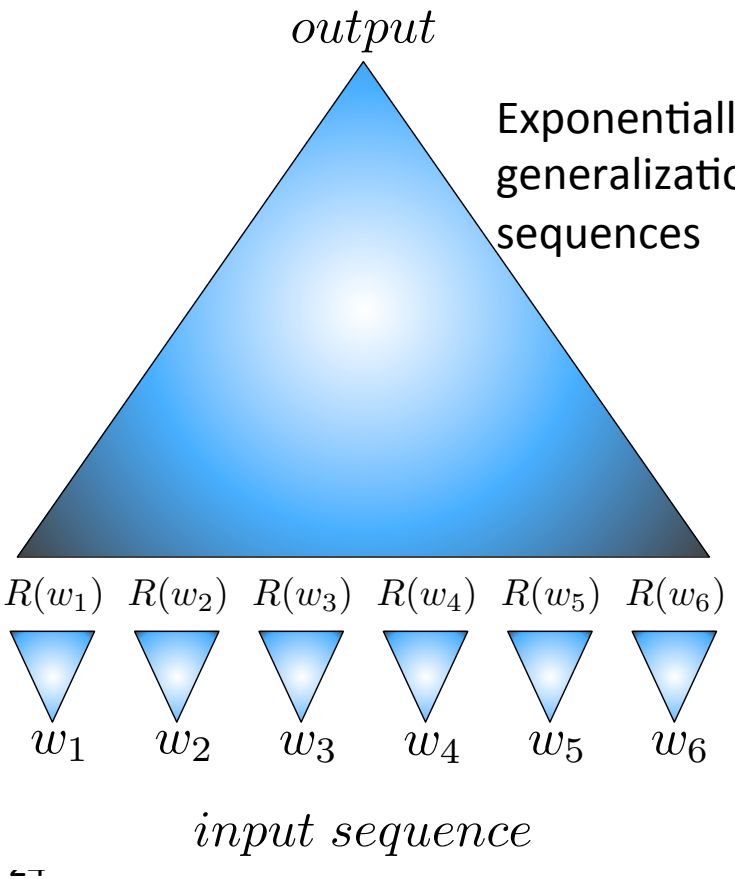

David Rumelhart



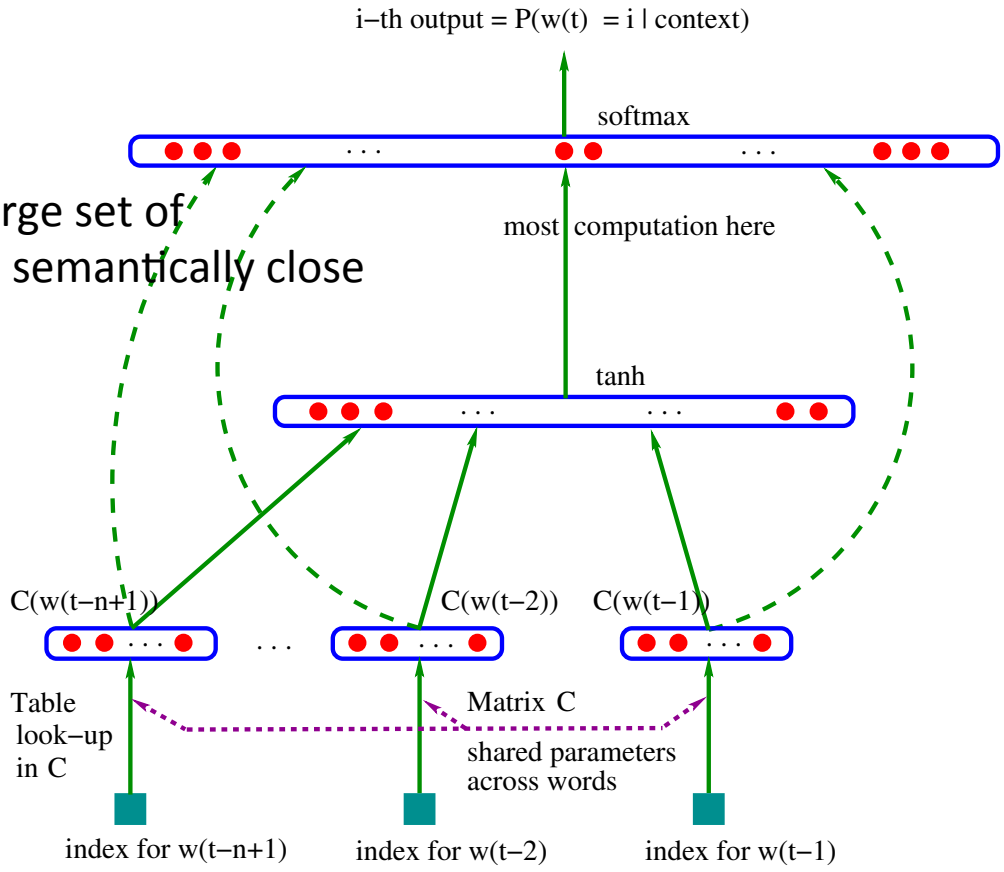


Neural Language Models: fighting one exponential by another one!

- (Bengio et al NIPS'2000)

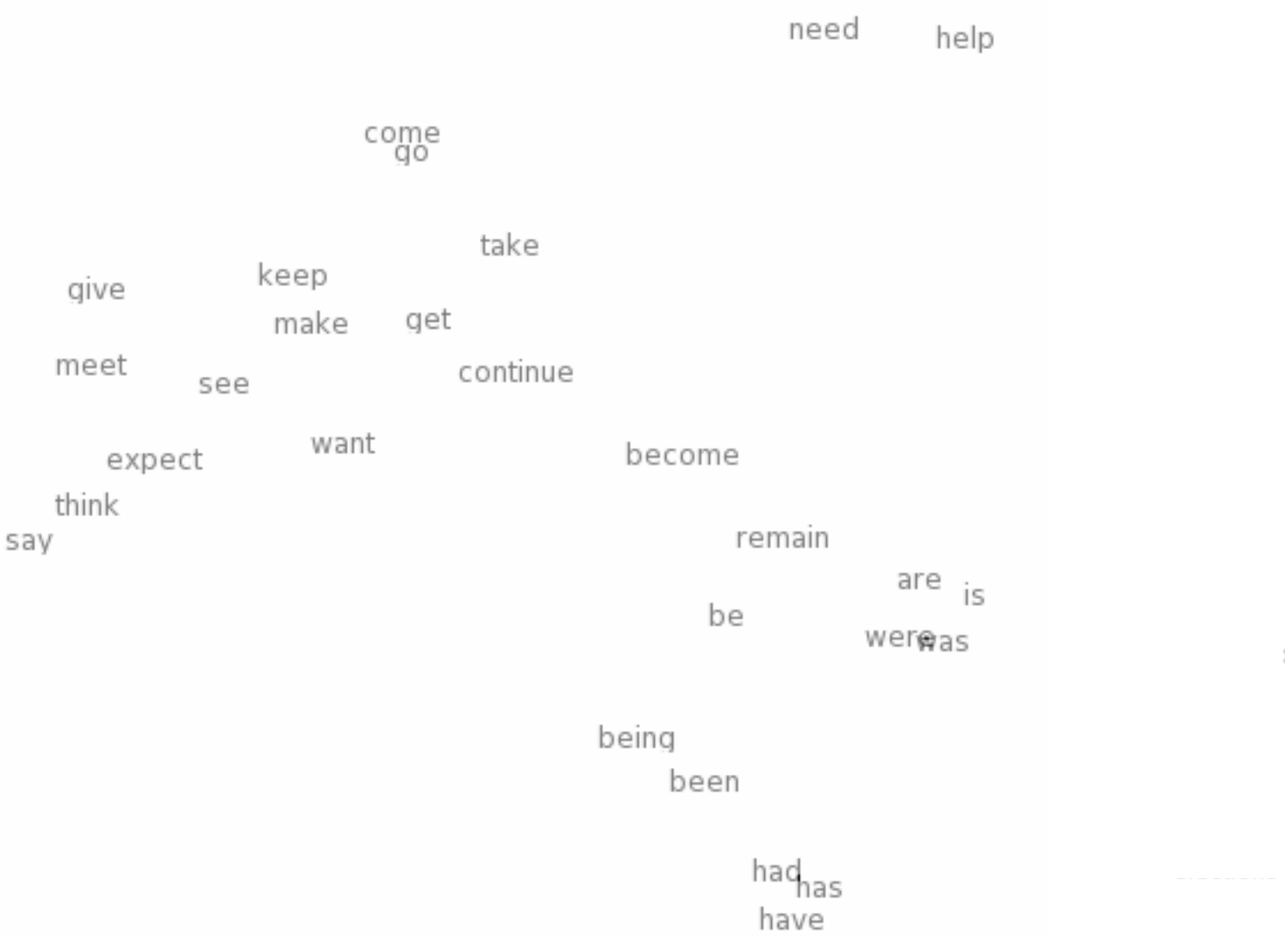


Exponentially large set of generalizations: semantically close sequences



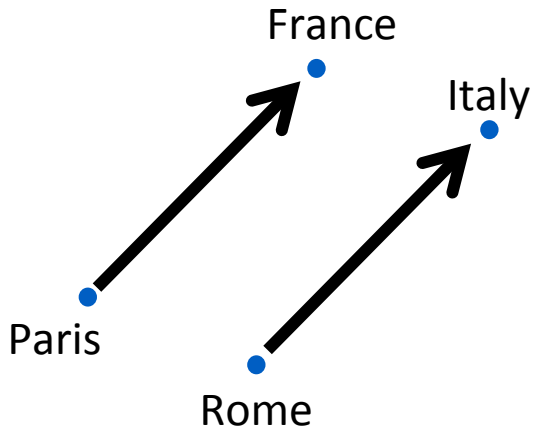
Exponentially large set of possible contexts

Neural word embeddings: visualization directions = Learned Attributes



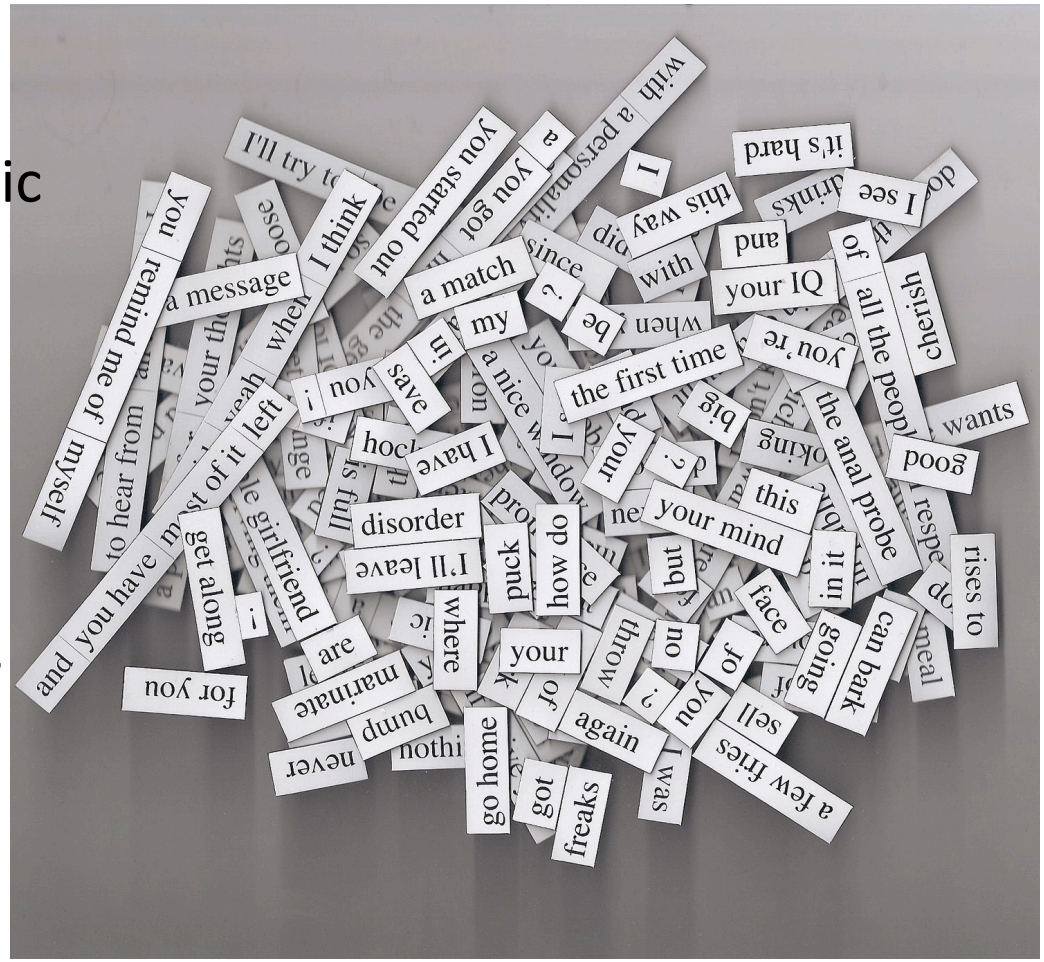
Analogical Representations for Free (Mikolov et al, ICLR 2013)

- Semantic relations appear as linear relationships in the space of learned representations
- King – Queen ≈ Man – Woman
- Paris – France + Italy ≈ Rome



The Next Challenge: Rich Semantic Representations for Word Sequences

- Impressive progress in capturing word semantics
  Easier learning: non-parametric (table look-up)
- Optimization challenge for mapping sequences to rich & complete representations
- Good test case: machine translation with auto-encoder framework



A Semantic Challenge: End-to-End Machine Translation

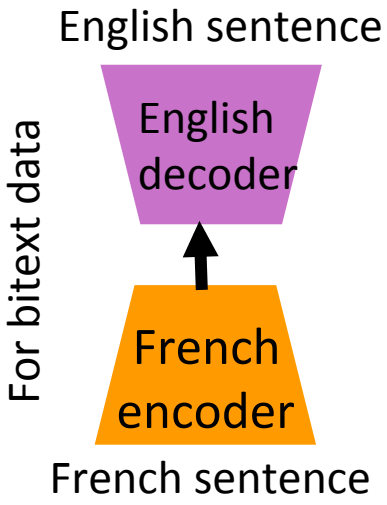
- Classical Machine Translation: several models separately trained by max. likelihood, brought together with logistic regression on top, based on n-grams

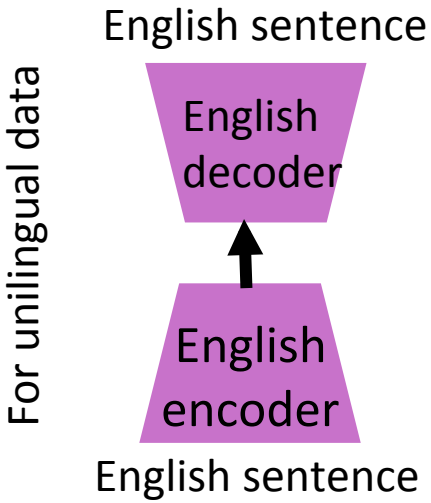
- Neural language models already shown to outperform n-gram models in terms of generalization power

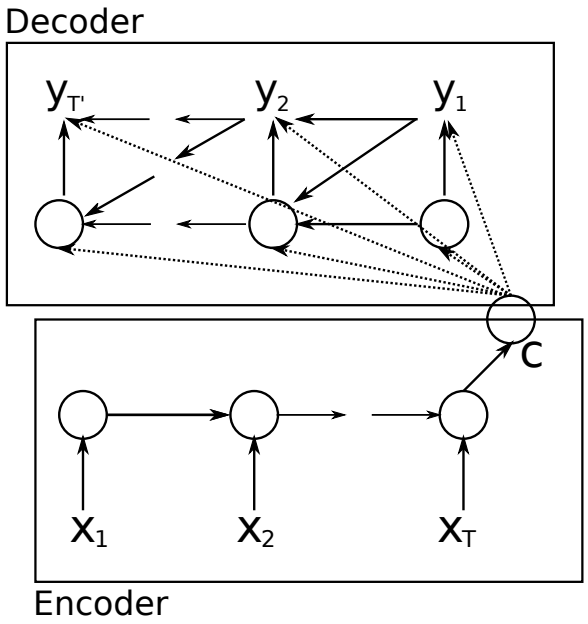
- Why not train a neural translation model end-to-end to estimate P(target sentence | source sentence)?

Encoder-Decoder Framework

- Intermediate representation of meaning
  = 'universal representation'
- Encoder: from word sequence to sentence representation
- Decoder: from representation to word sequence distribution







The Depth Prior can be Exponentially Advantageous

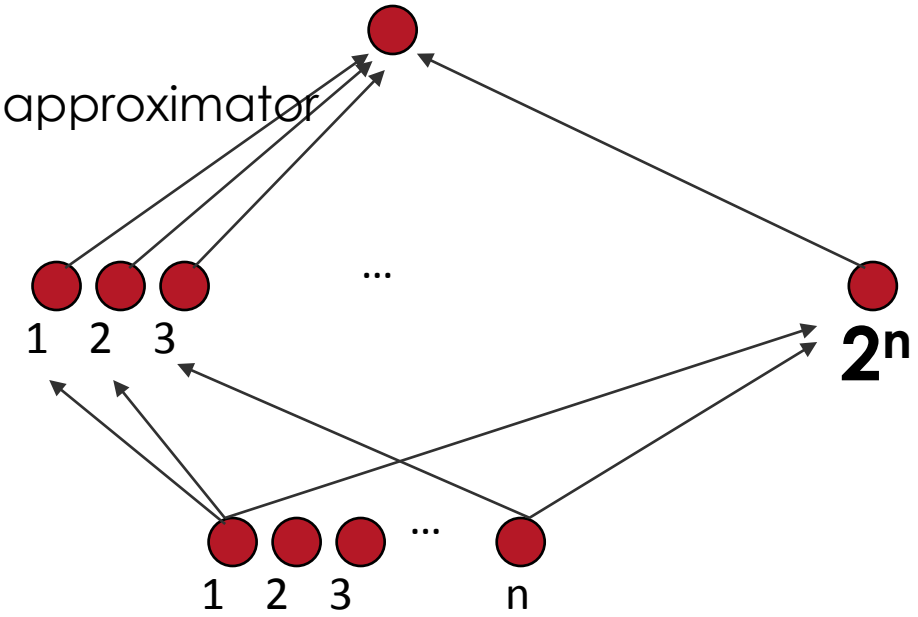
Theoretical arguments:

2 layers of { Logic gates / Formal neurons / RBF units } = universal approximator

RBMs & auto-encoders = universal approximator

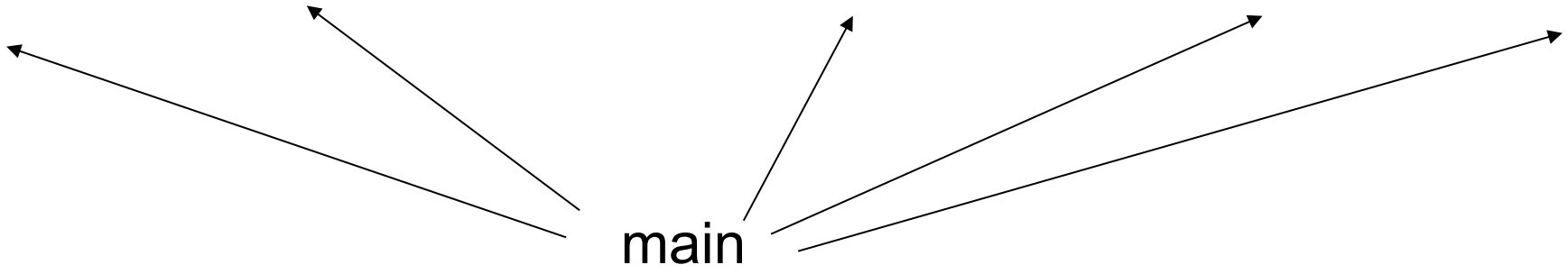
Theorems on advantage of depth:
(Hastad et al 86 & 91, Bengio et al 2007, Bengio & Delalleau 2011, Braverman 2011, Pascanu et al 2014, Montufar et al **NIPS 2014**)

Some functions compactly represented with k layers may require exponential size with 2 layers

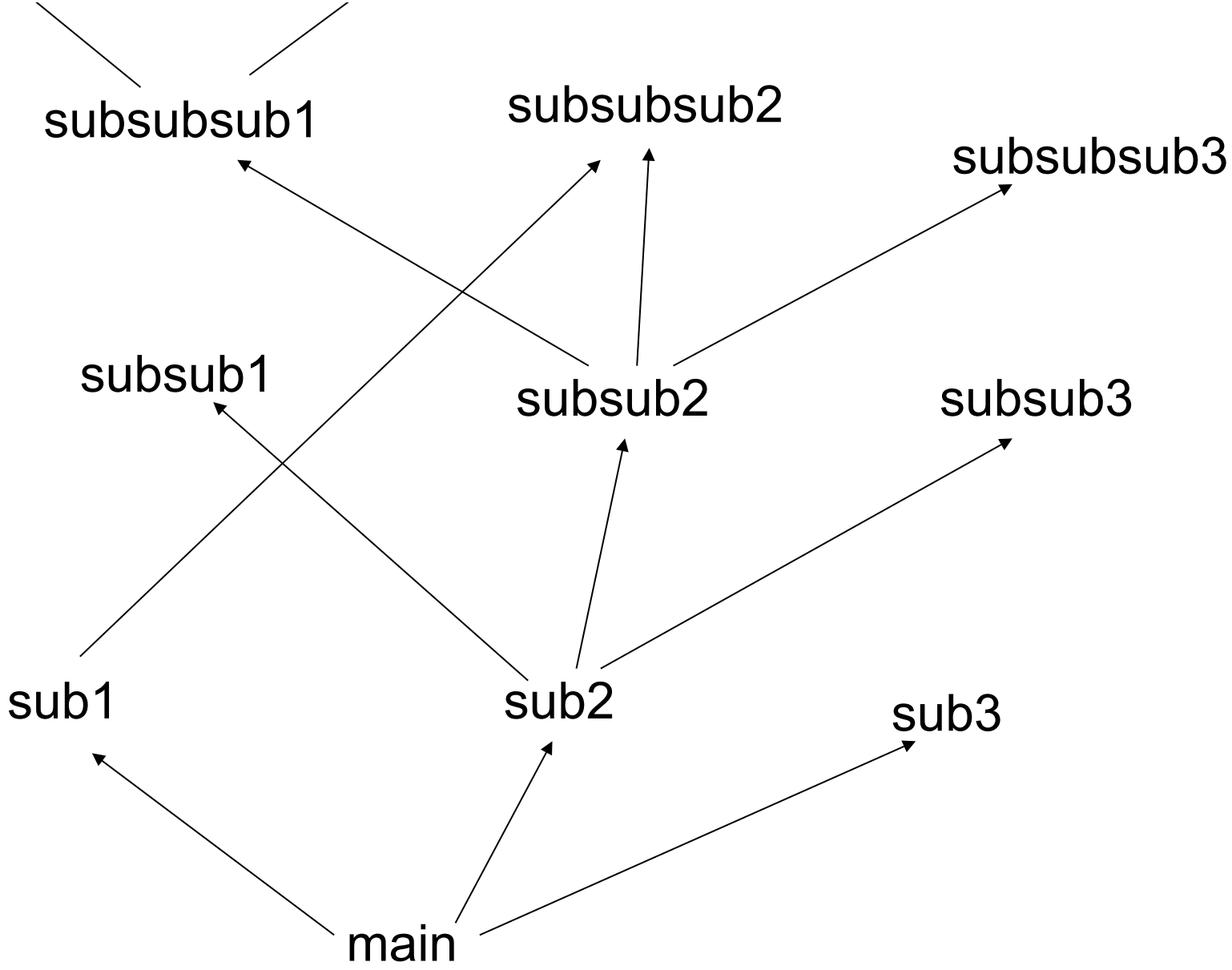


subroutine1 includes
subsub1 code and
subsub2 code and
subsubsub1 code

subroutine2 includes
subsub2 code and
subsub3 code and
subsubsub3 code and …



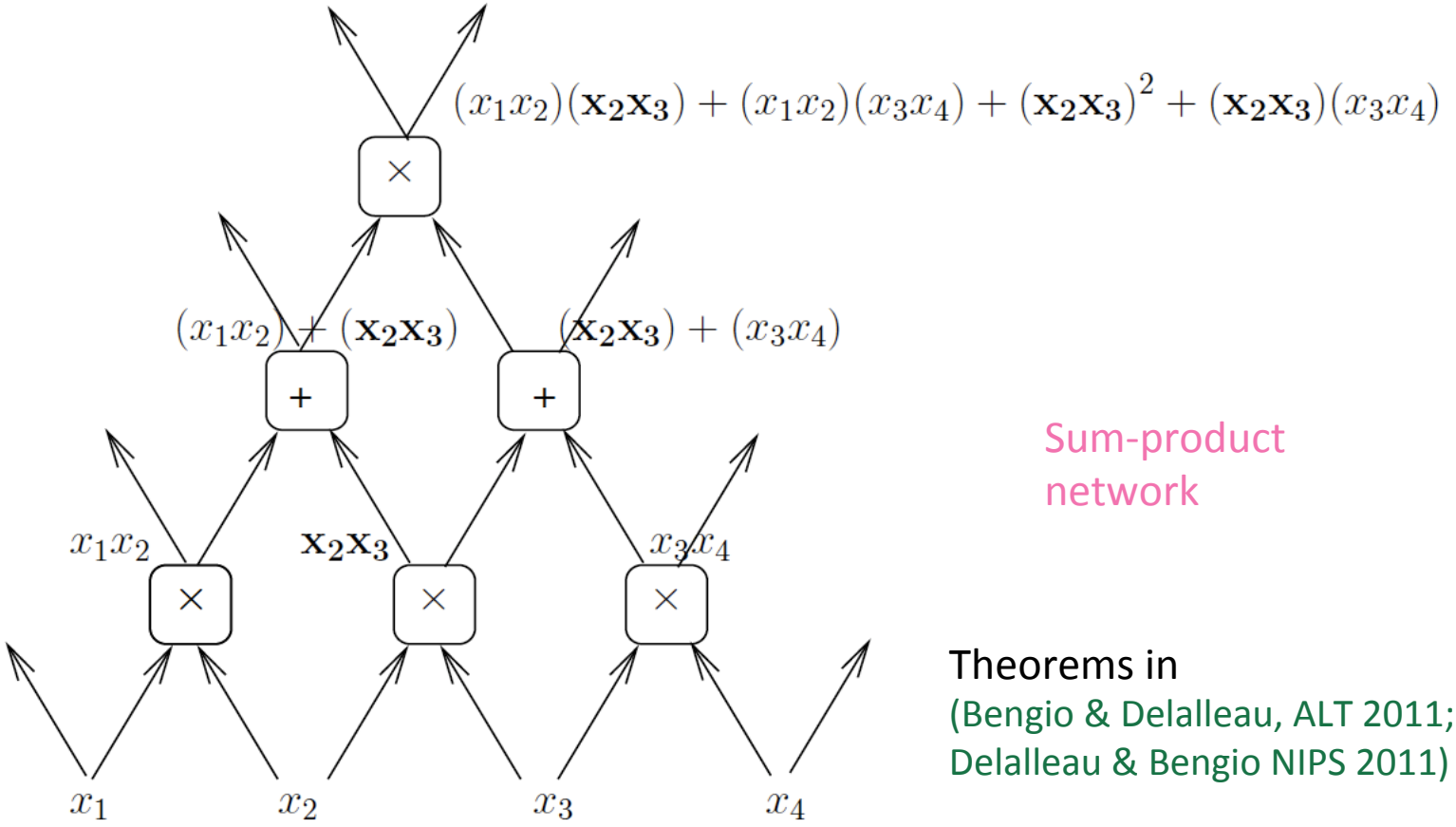
“Shallow” computer program



“Deep” computer program

Sharing Components in a Deep Architecture

Polynomial expressed with shared components: advantage of depth may grow exponentially



Sum-product network

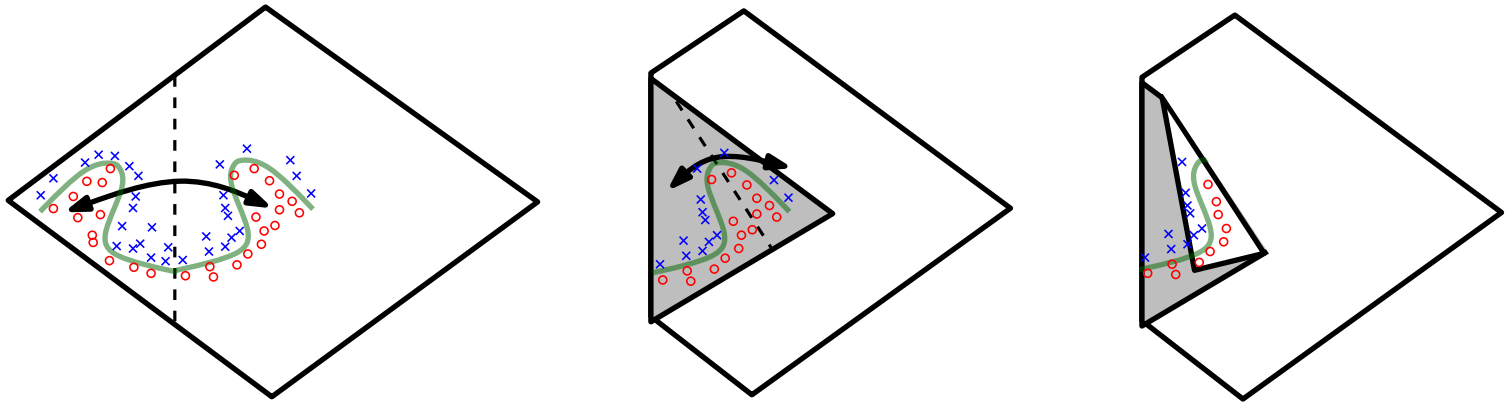
Theorems in
(Bengio & Delalleau, ALT 2011;
Delalleau & Bengio NIPS 2011)

New theoretical result: Expressiveness of deep nets with piecewise-linear activation fns

(Pascanu, Montufar, Cho & Bengio; ICLR 2014)

(Montufar, Pascanu, Cho & Bengio; NIPS 2014)

Deeper nets with rectifier/maxout units are exponentially more expressive than shallow ones (1 hidden layer) because they can split the input space in many more (not-independent) linear regions, with constraints, e.g., with abs units, each unit creates mirror responses, folding the input space:

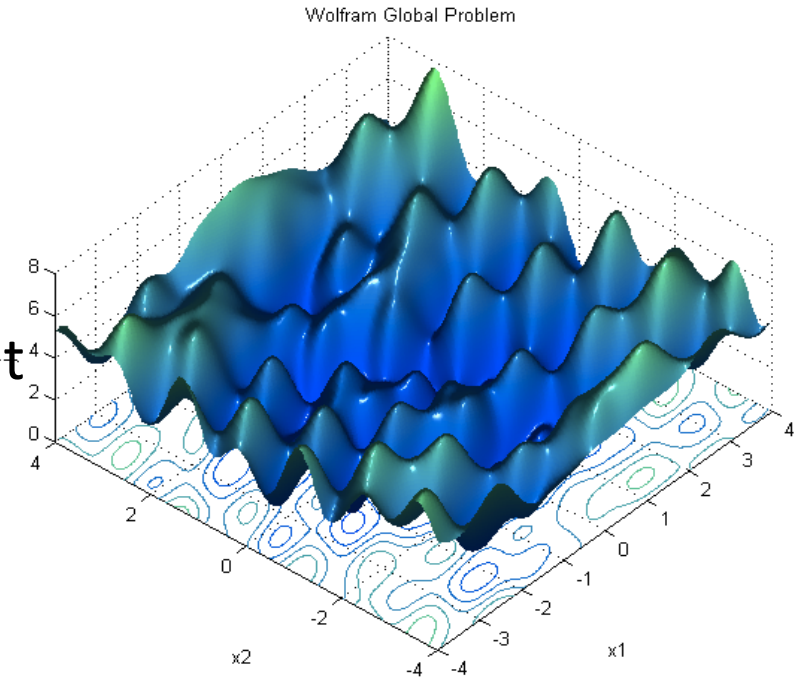


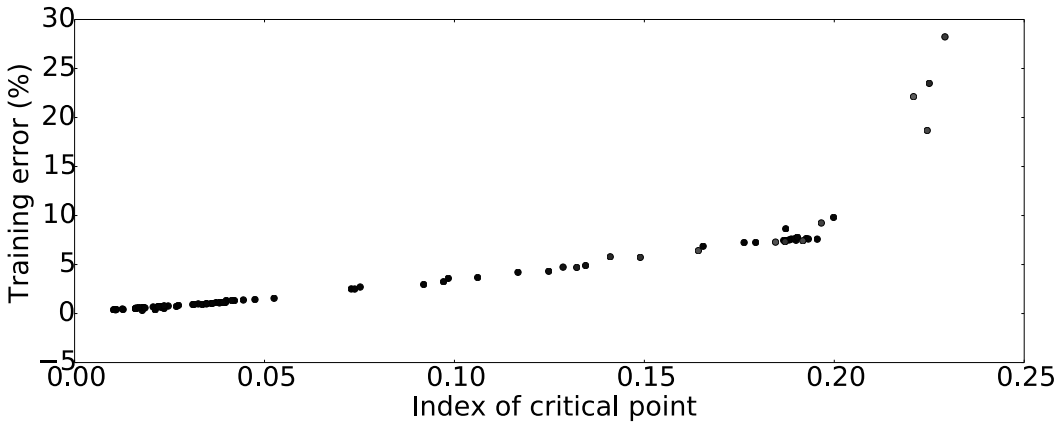
A Myth is Being Debunked: Local Minima in Neural Nets → Convexity is not needed

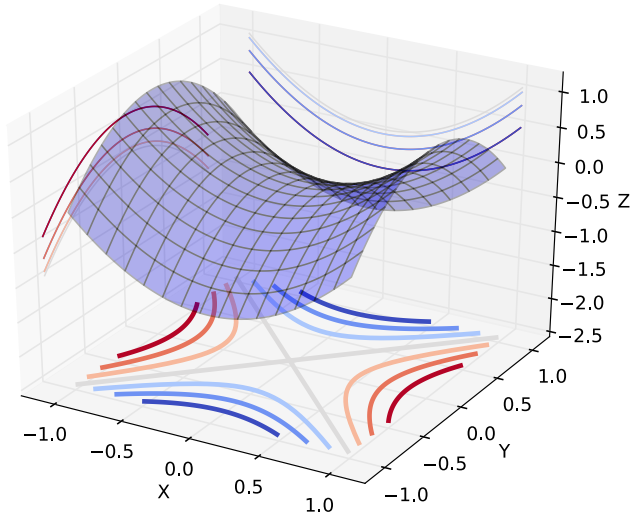
- (Pascanu, Dauphin, Ganguli, Bengio, arXiv May 2014): *On the saddle point problem for non-convex optimization*
- (Dauphin, Pascanu, Gulcehre, Cho, Ganguli, Bengio, NIPS' 2014): *Identifying and attacking the saddle point problem in high-dimensional non-convex optimization*
- (Choromanska, Henaff, Mathieu, Ben Arous & LeCun 2014): *The Loss Surface of Multilayer Nets*

Saddle Points

- Local minima dominate in low-D, but saddle points dominate in high-D
- Most local minima are close to the bottom (global minimum error)

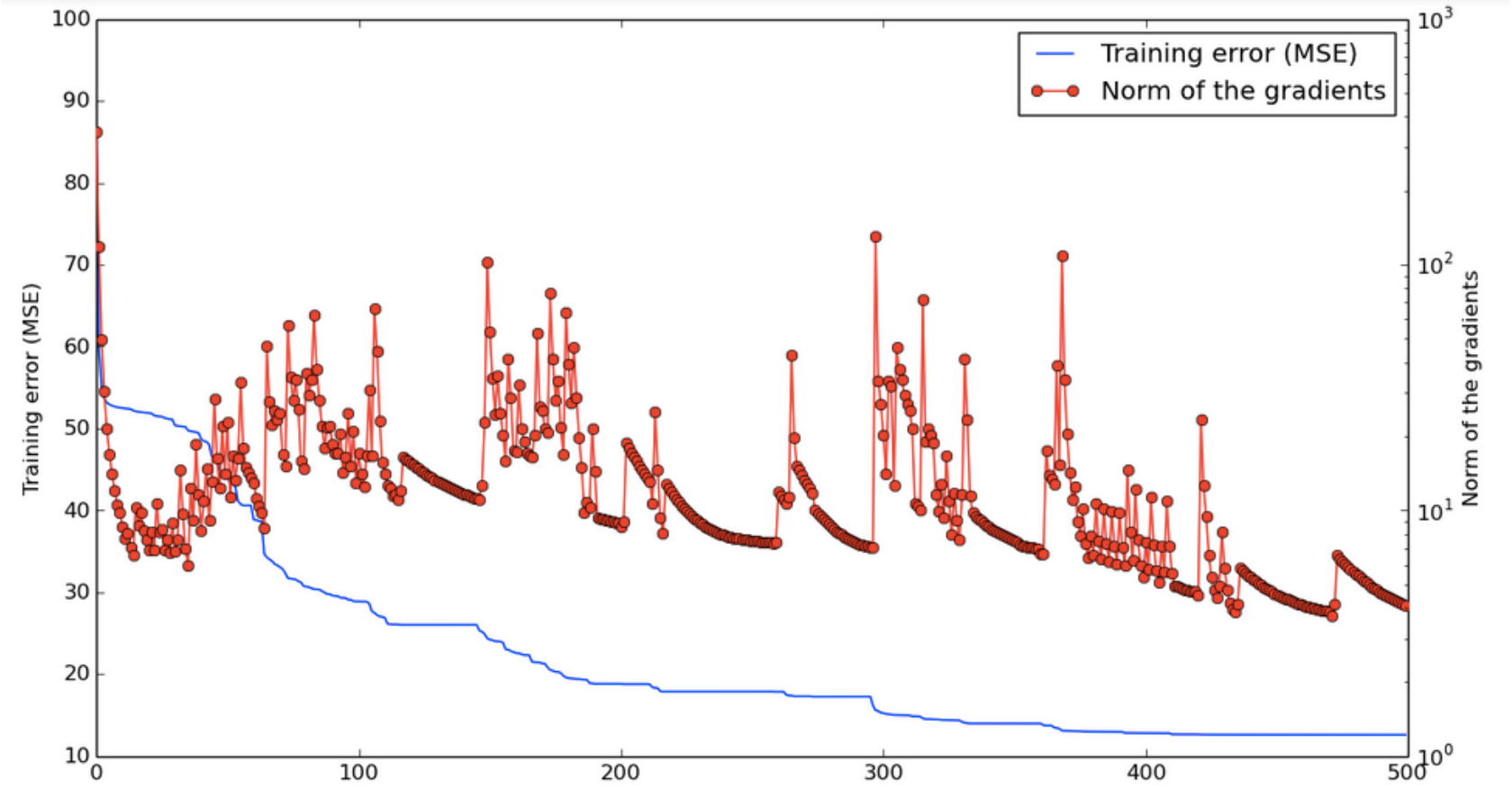






Saddle Points During Training

- Oscillating between two behaviors:
  - Slowly approaching a saddle point
  - Escaping it

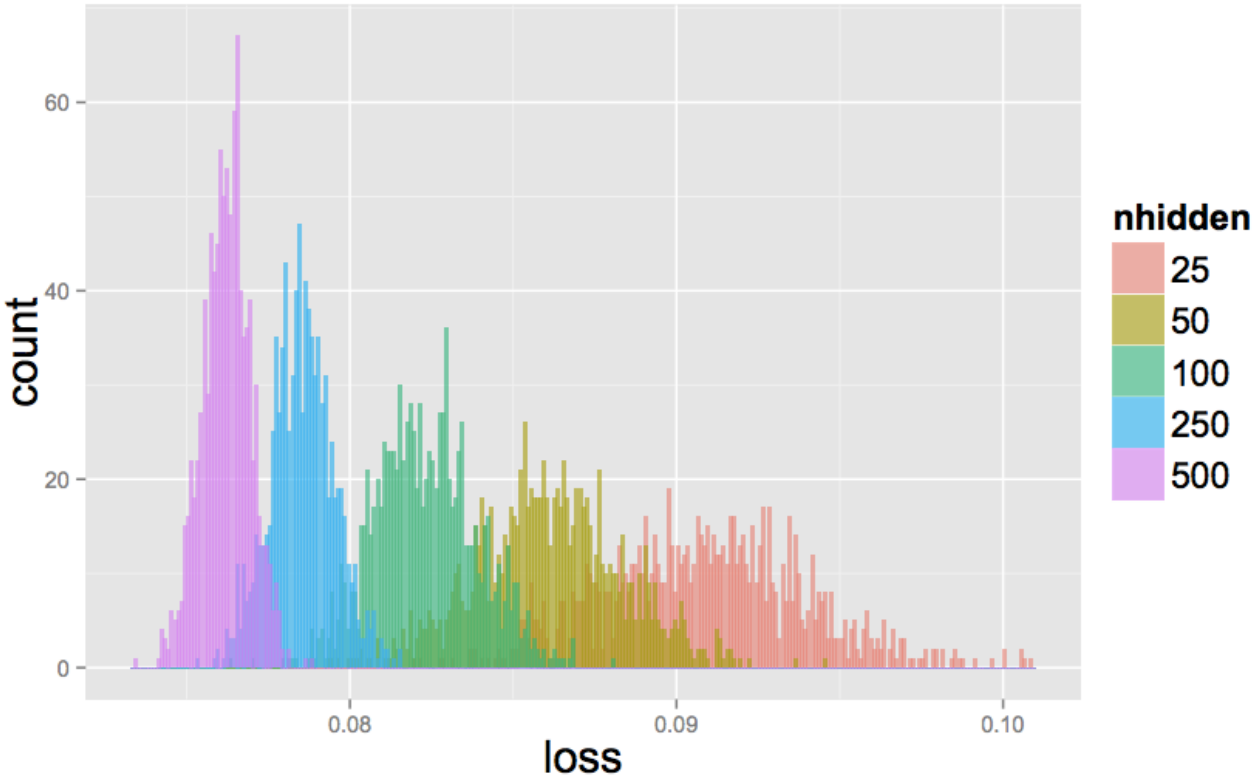


Low Index Critical Points

Choromanska et al & LeCun 2014, 'The Loss Surface of Multilayer Nets'

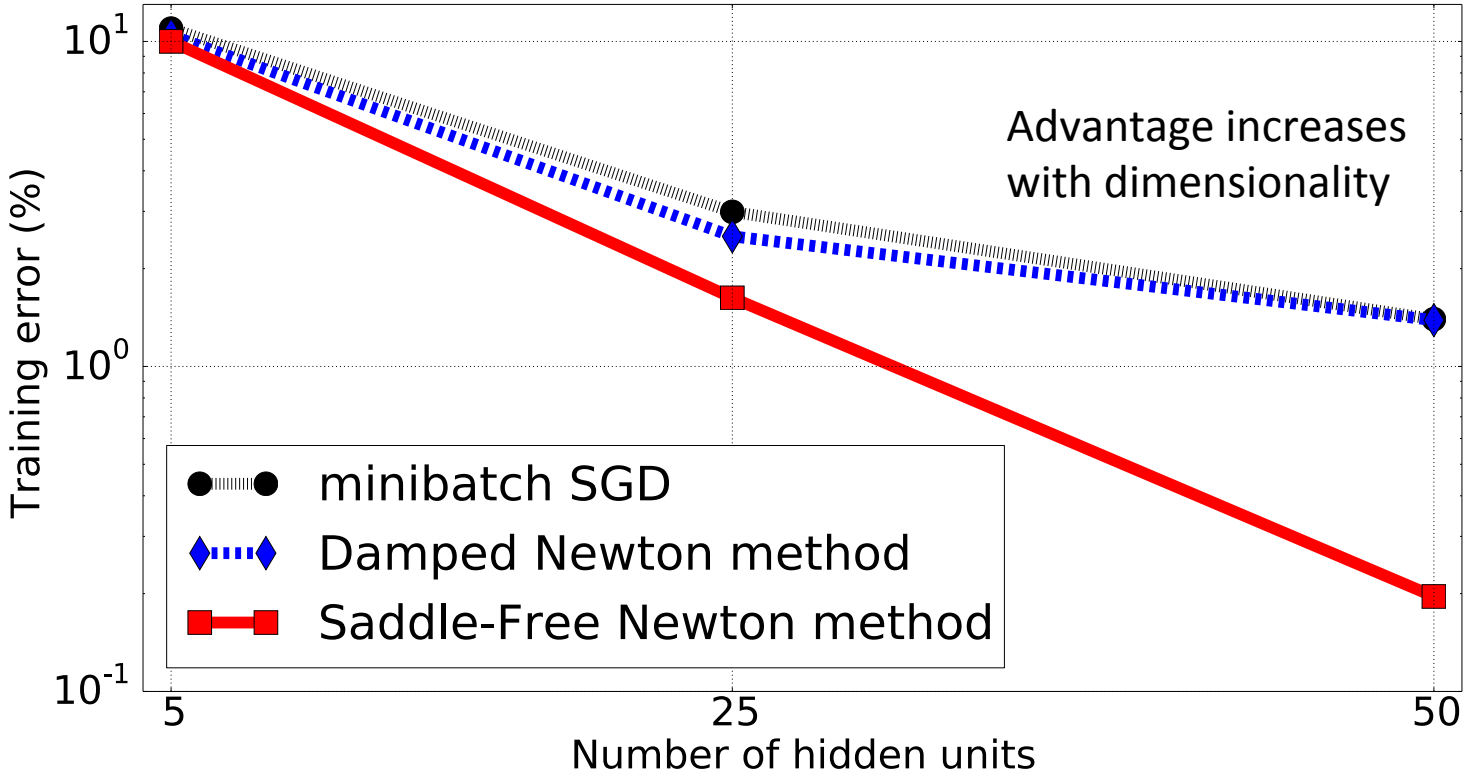
Shows that deep rectifier nets are analogous to spherical spin-glass models

The low-index critical points of large models concentrate in a band just above the global minimum



Saddle-Free Optimization (Pascanu, Dauphin, Ganguli, Bengio 2014)

- Saddle points are ATTRACTIVE for Newton's method
- Replace eigenvalues λ of Hessian by |λ|
- Justified as a particular trust region method



How do humans generalize from very few examples?

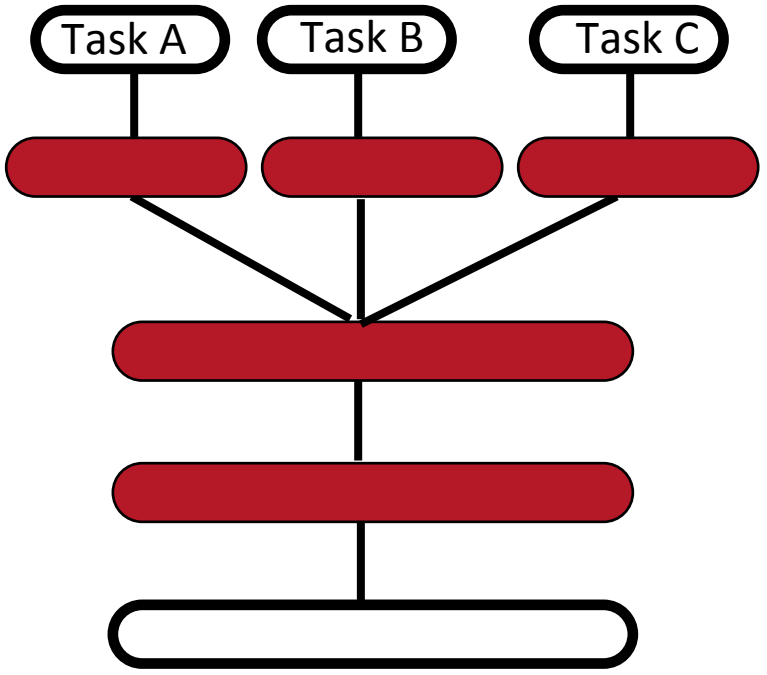
- They **transfer** knowledge from previous learning:
  - Representations
  - Explanatory factors
- Previous learning from: unlabeled data

  + labels for other tasks
- **Prior: shared underlying explanatory factors, in particular between P(x) and P(Y|x)**

Multi-Task Learning

- Generalizing better to new tasks (tens of thousands!) is crucial to approach AI
- Deep architectures learn good intermediate representations that can be shared across tasks

  (Collobert & Weston ICML 2008, Bengio et al AISTATS 2011)
- Good representations that disentangle underlying factors of variation make sense for many tasks because **each task concerns a subset of the factors**

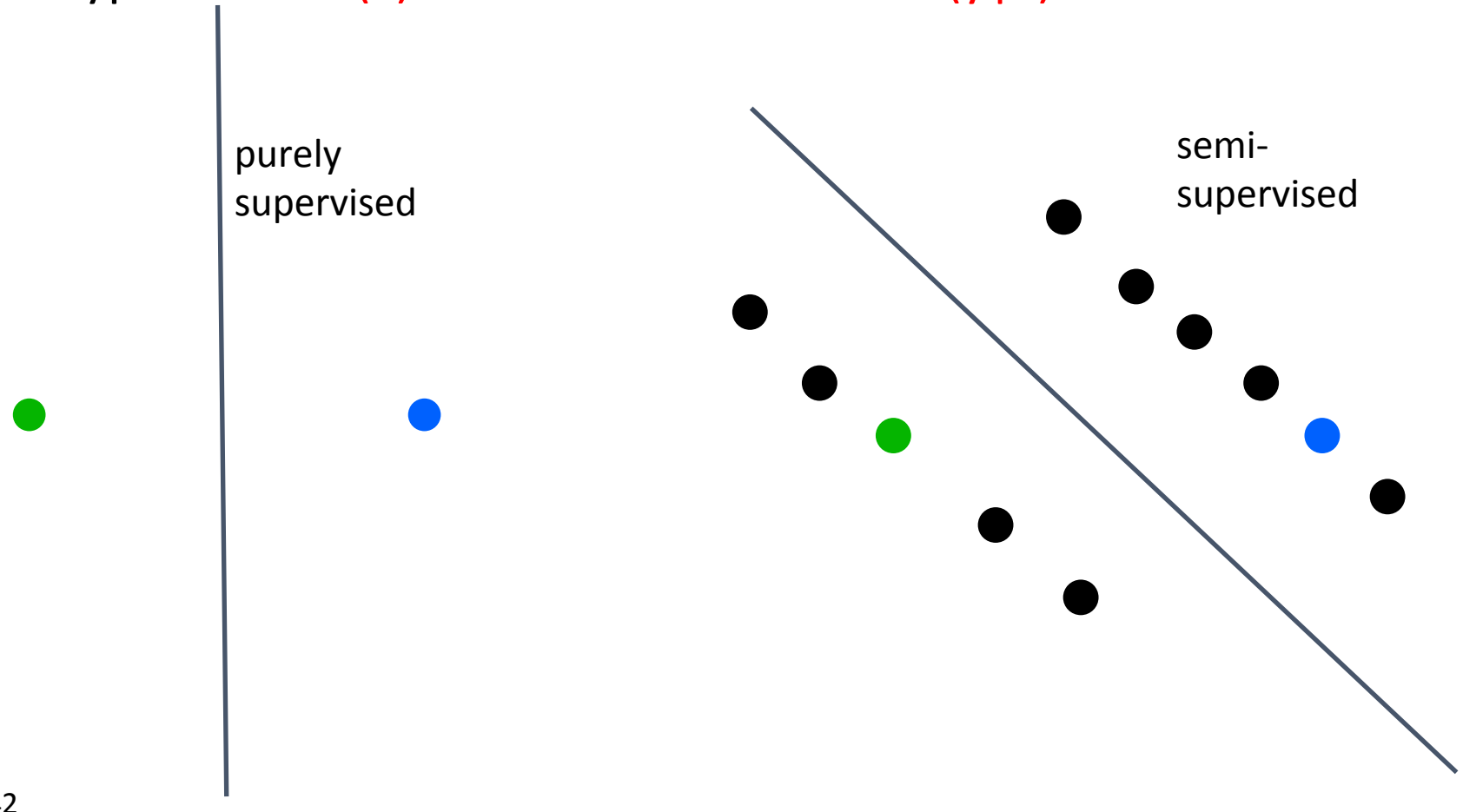


E.g. dictionary, with intermediate concepts re-used across many definitions

Prior: shared underlying explanatory factors between tasks

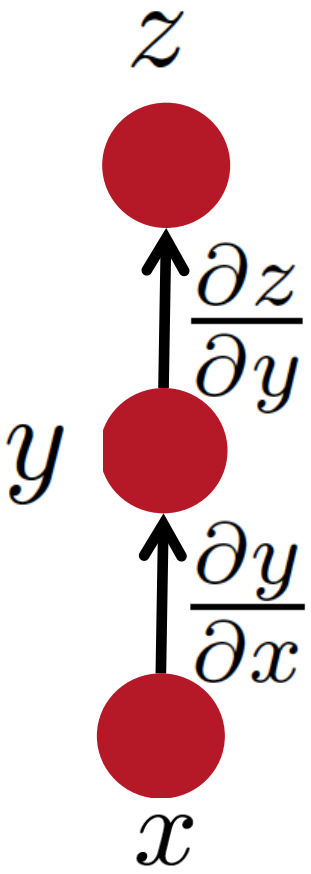
Sharing Statistical Strength by Semi-Supervised Learning

- Hypothesis: P(x) shares structure with P(y|x)



Algorithms

Simple Chain Rule



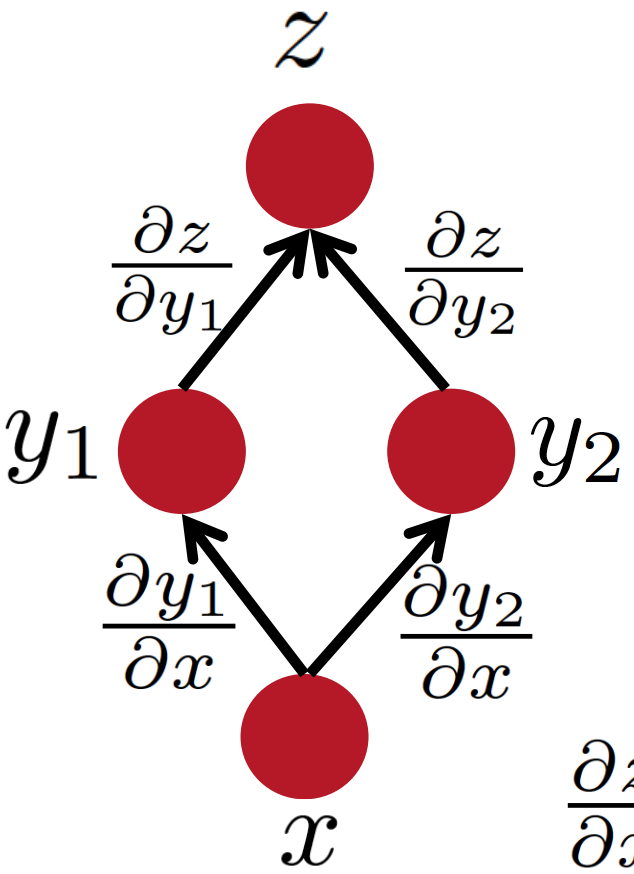
$$\Delta z = \frac{\partial z}{\partial y}\Delta y$$

$$\Delta y = \frac{\partial y}{\partial x}\Delta x$$

$$\Delta z = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}\Delta x$$

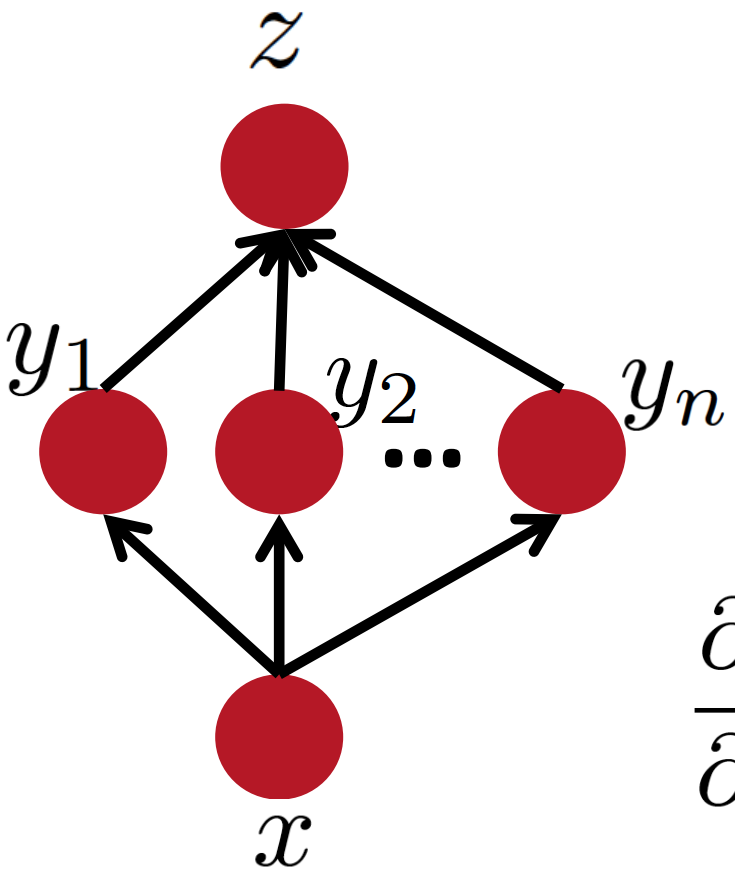
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}$$

Multiple Paths Chain Rule



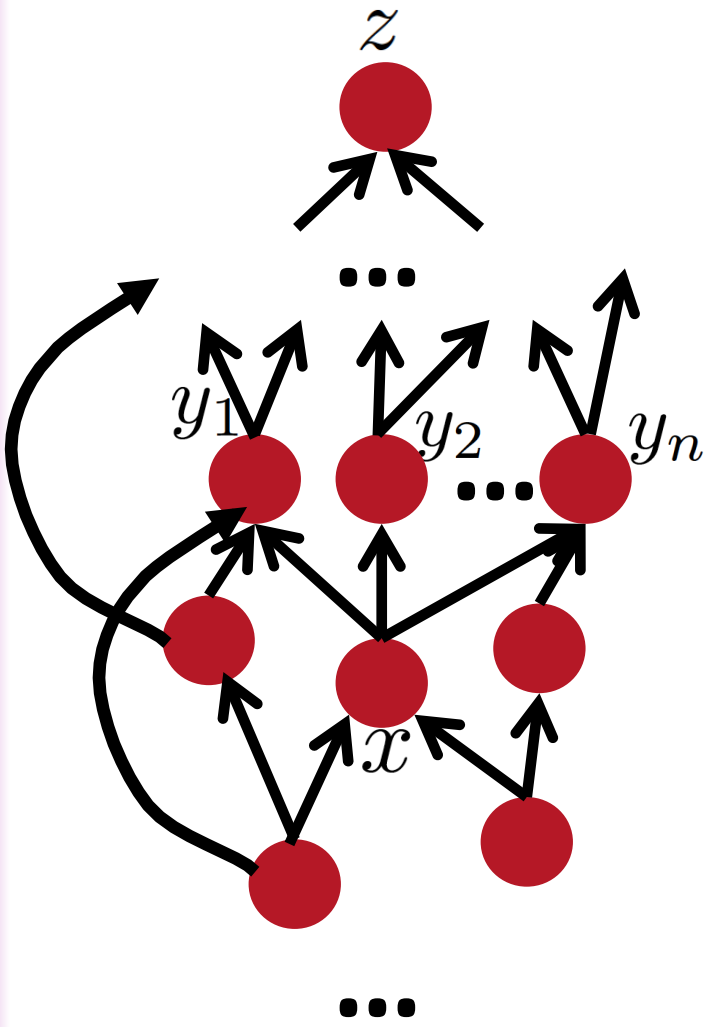
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y_1}\frac{\partial y_1}{\partial x} + \frac{\partial z}{\partial y_2}\frac{\partial y_2}{\partial x}$$

Multiple Paths Chain Rule - General



$$\frac{\partial z}{\partial x} = \sum_{i=1}^{n} \frac{\partial z}{\partial y_i} \frac{\partial y_i}{\partial x}$$

Chain Rule in Flow Graph

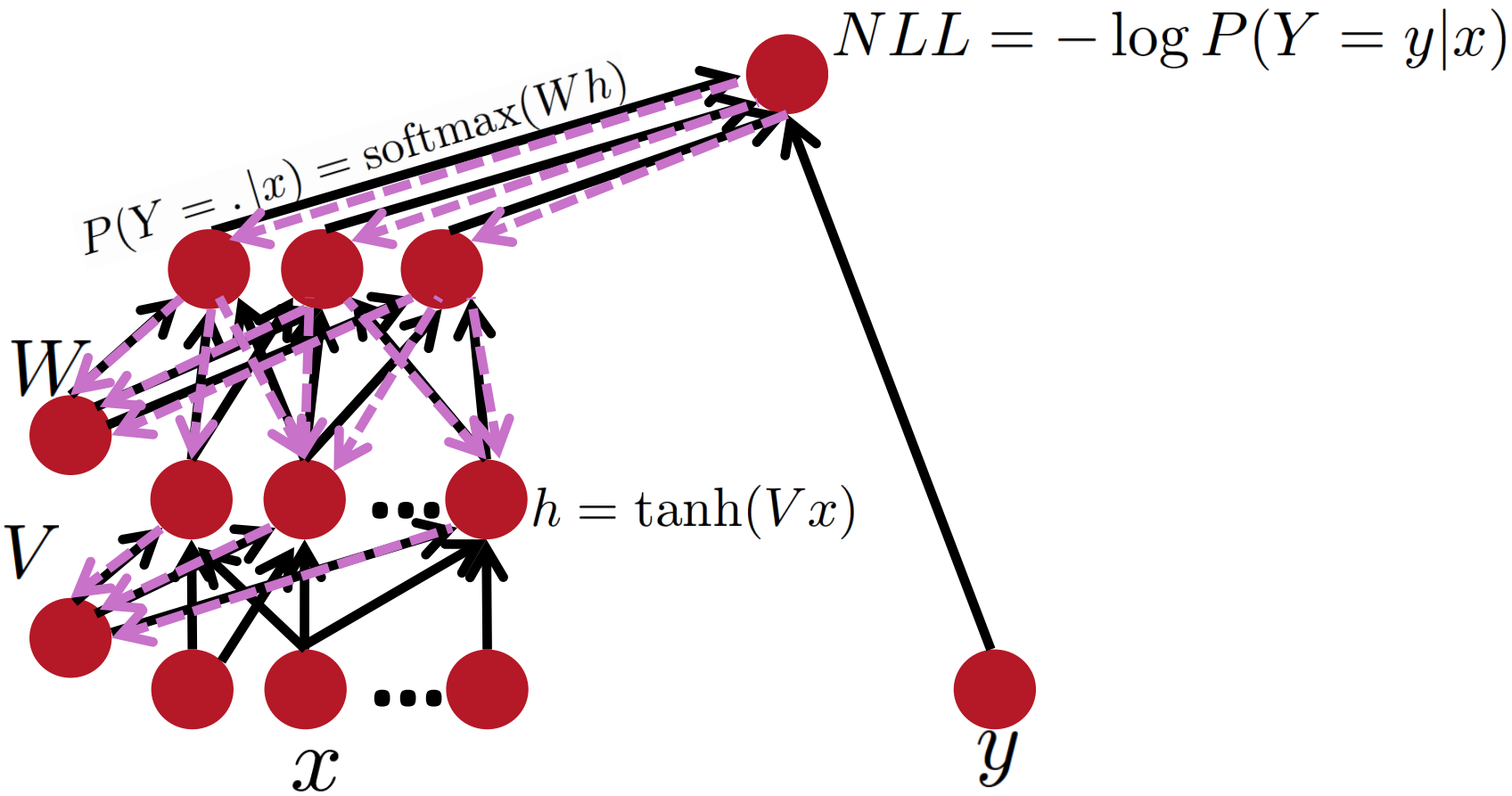


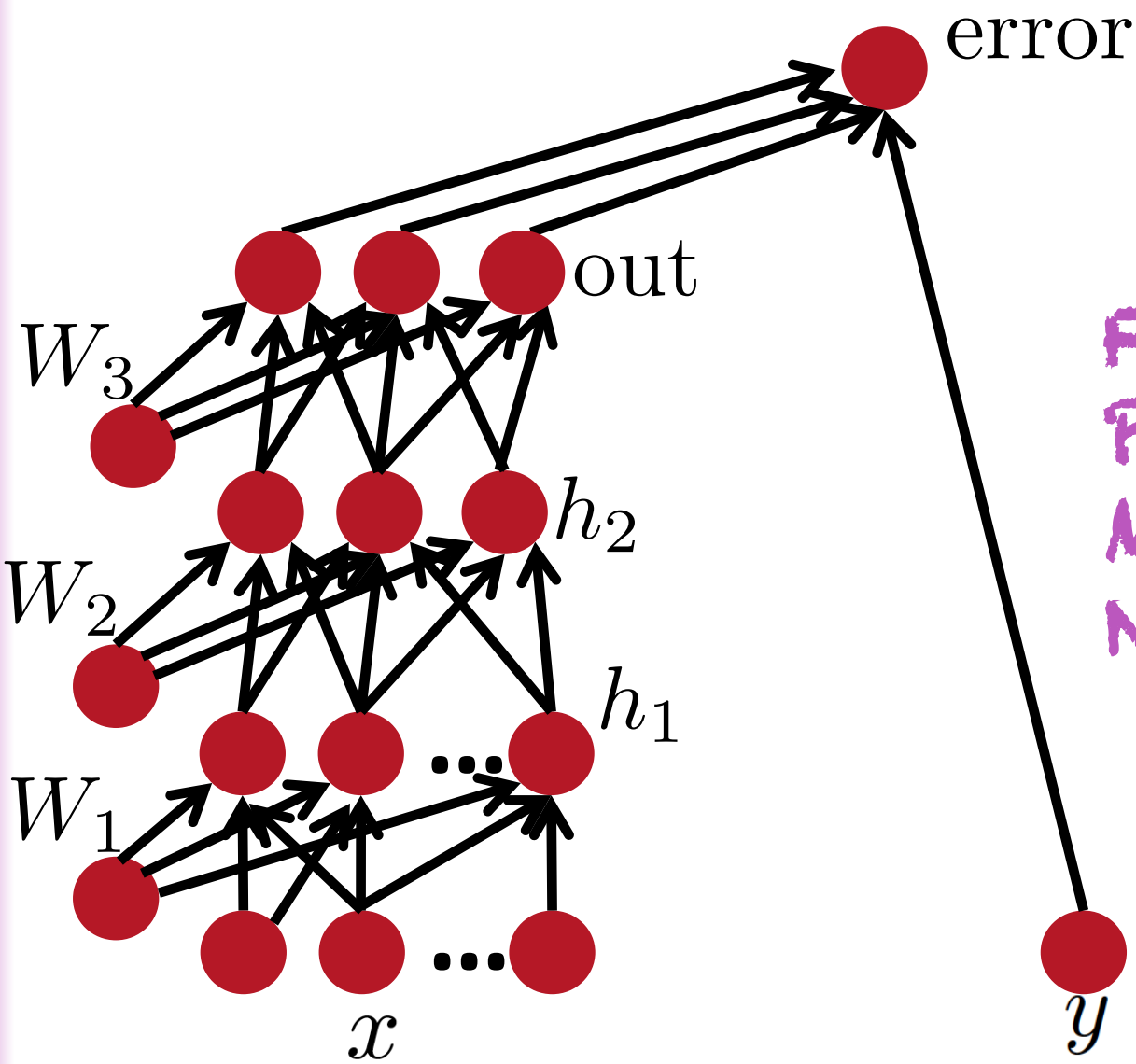
Flow graph: any directed acyclic graph
node = computation result
arc = computation dependency

$\{y_1, y_2, \ldots y_n\}$ = successors of x

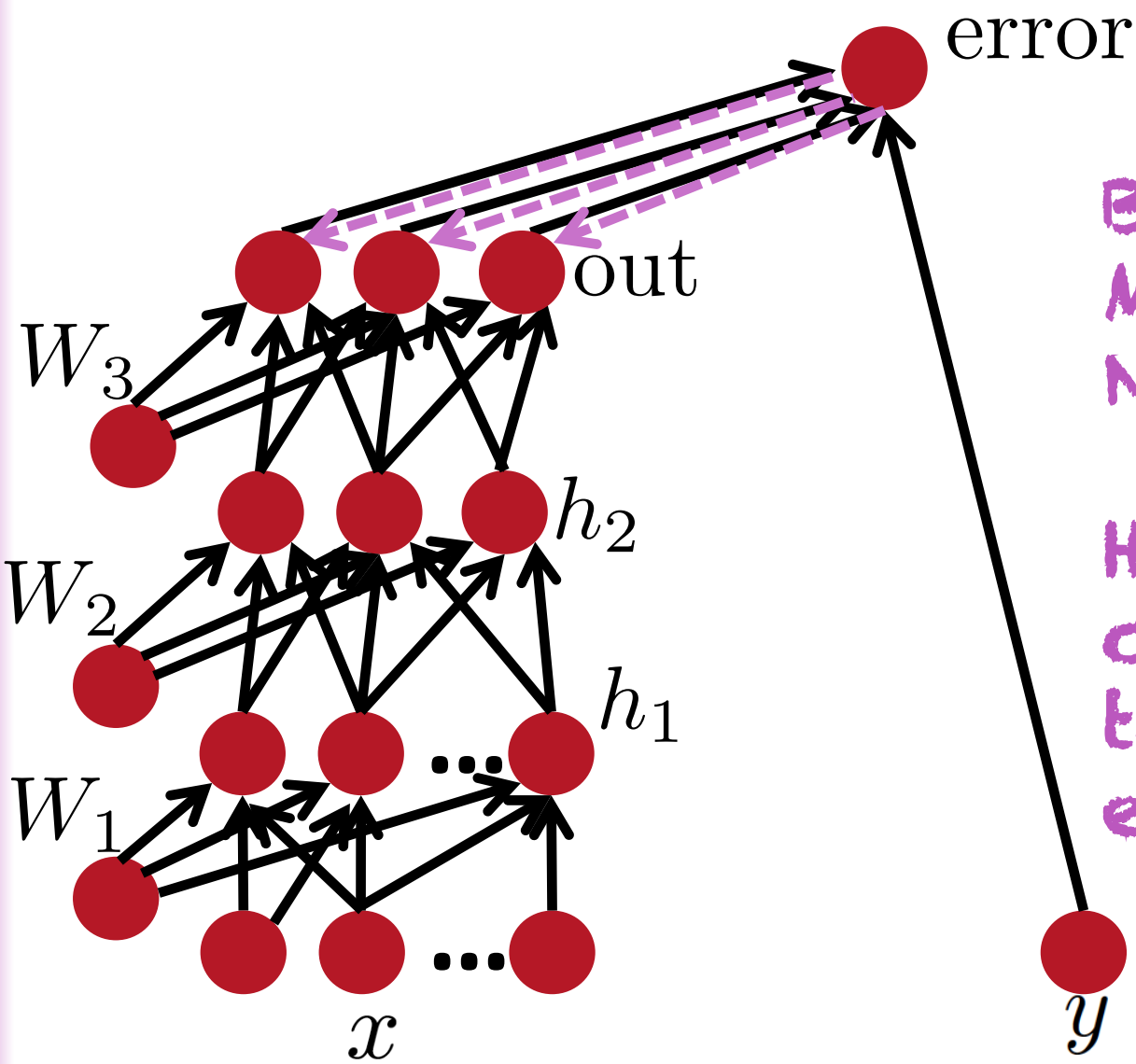
$$\frac{\partial z}{\partial x} = \sum_{i=1}^{n} \frac{\partial z}{\partial y_i} \frac{\partial y_i}{\partial x}$$

Back-Prop in Multi-Layer Net



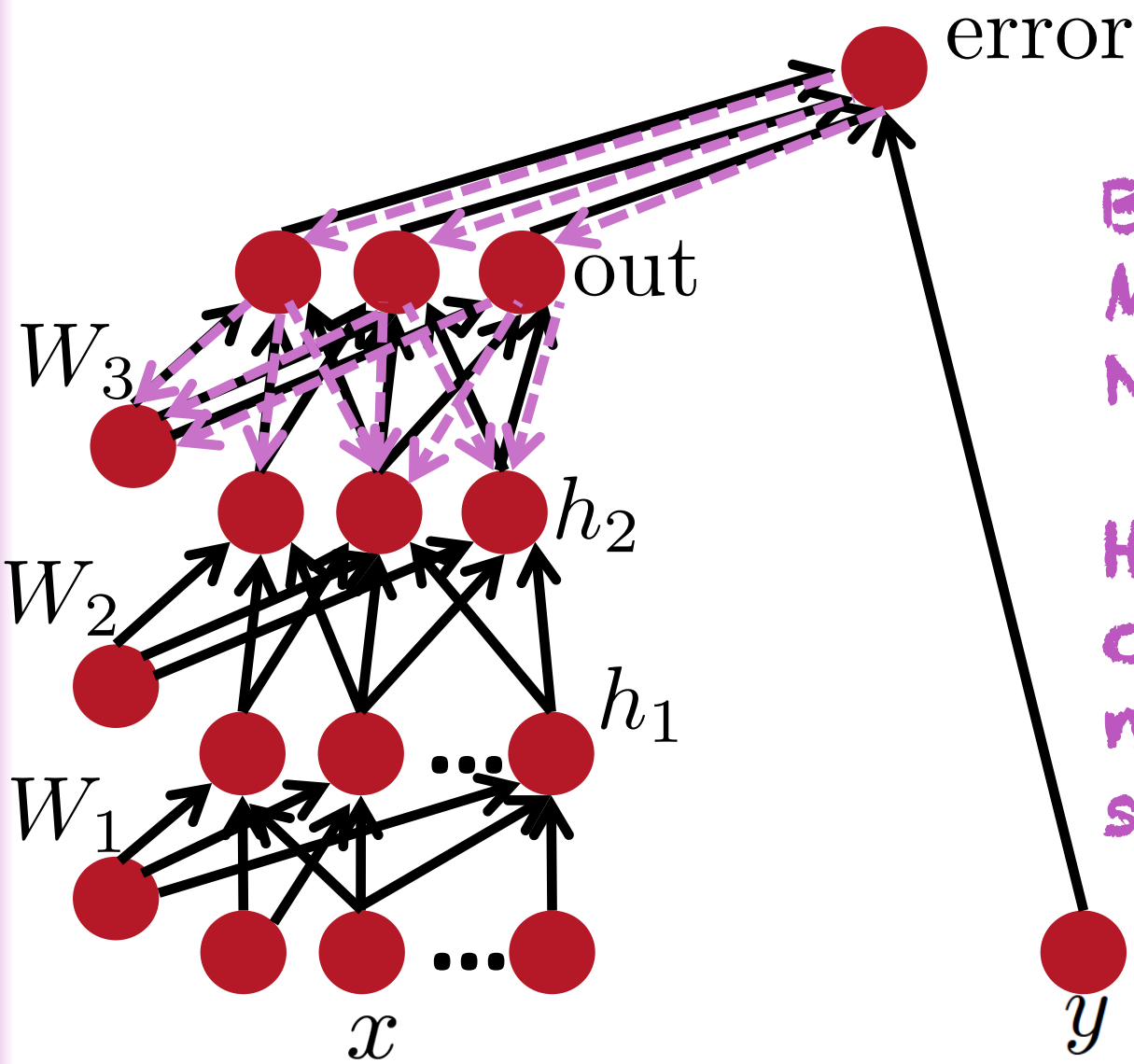


Forward-Prop in Multi-Layer Net



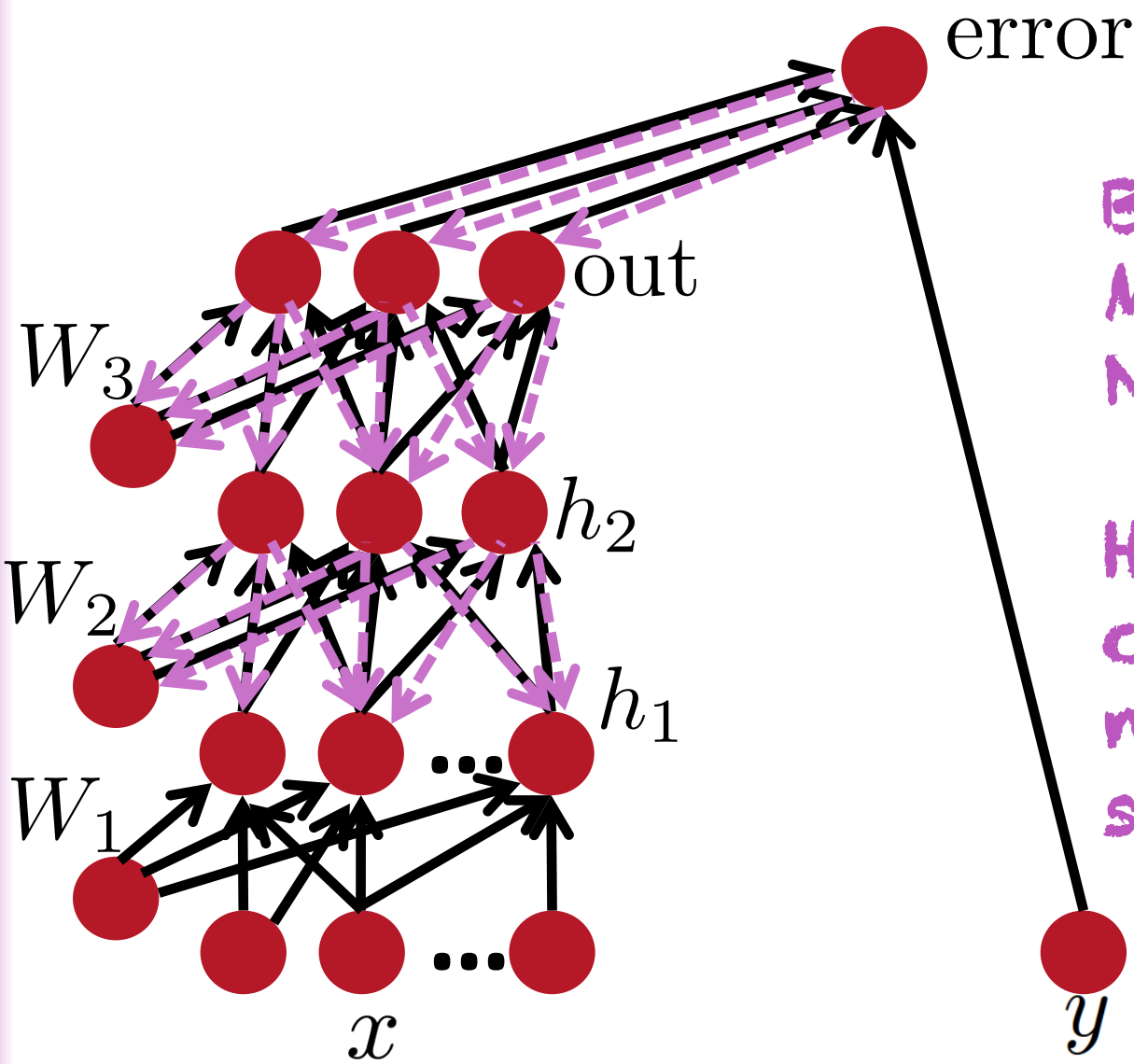
Backprop in Multi-Layer Net:

How outputs could change to make error smaller



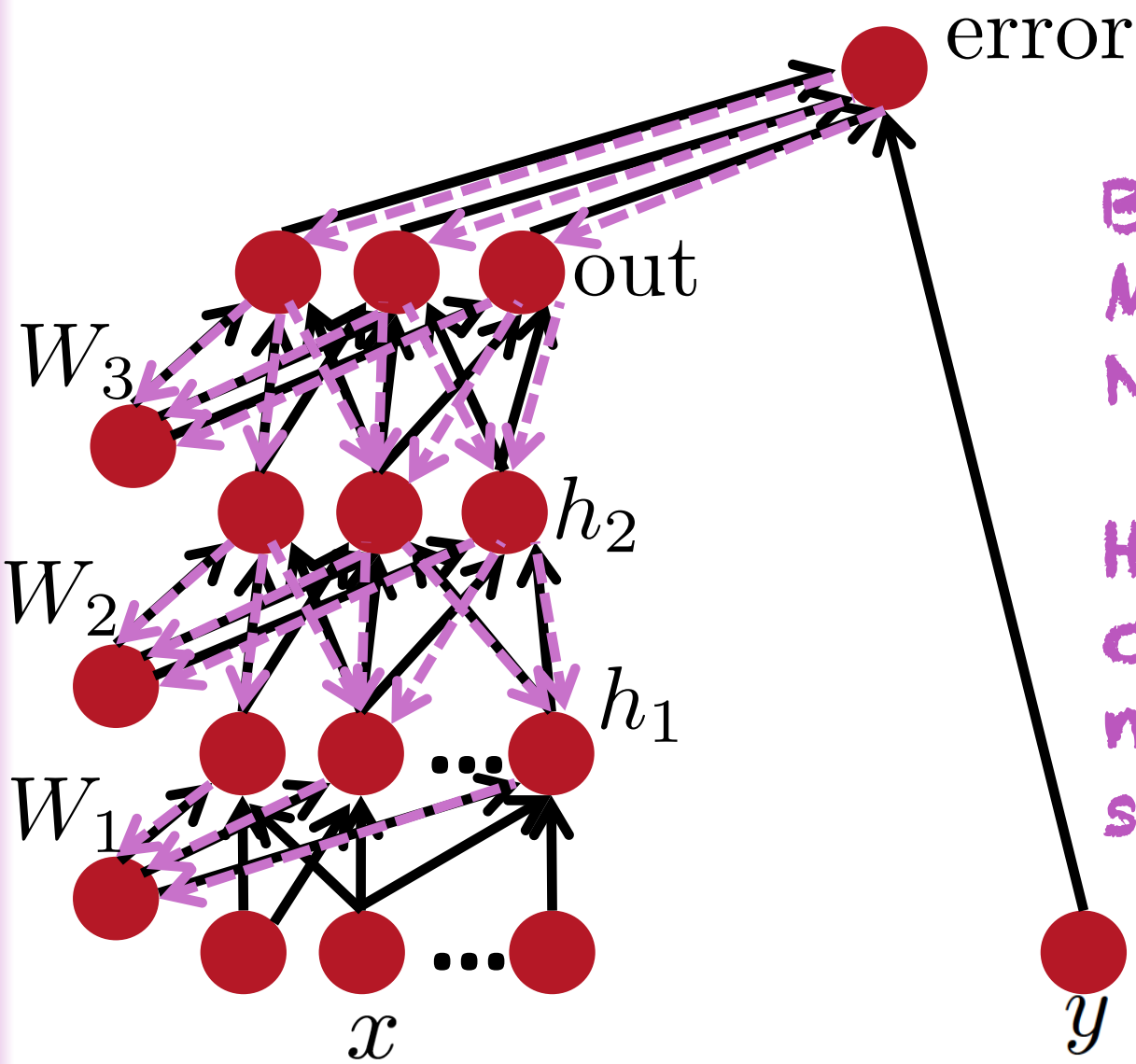
Backprop in Multi-Layer Net:

How h_2 could change to make error smaller



Backprop in Multi-Layer Net:

How h_1 could change to make error smaller

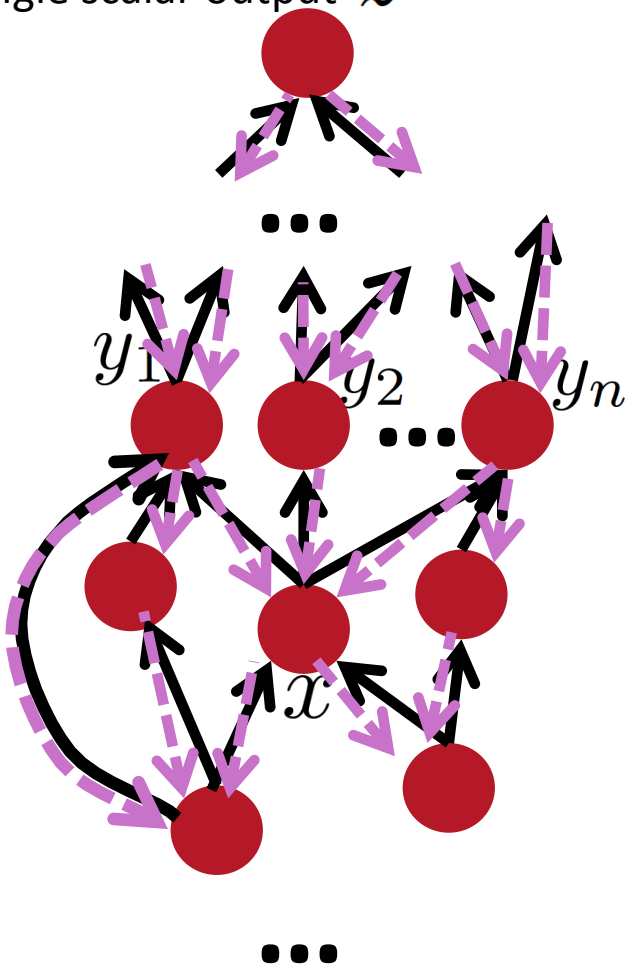


Backprop in Multi-Layer Net:

How W_1 could change to make error smaller

Back-Prop in General Flow Graph

Single scalar output z



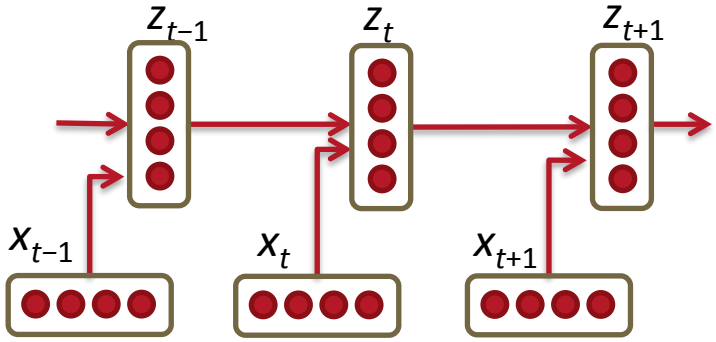
1. Fprop: visit nodes in topo-sort order
   - Compute value of node given predecessors
2. Bprop:
   - initialize output gradient = 1
   - visit nodes in reverse order:
     Compute gradient wrt each node using gradient wrt successors

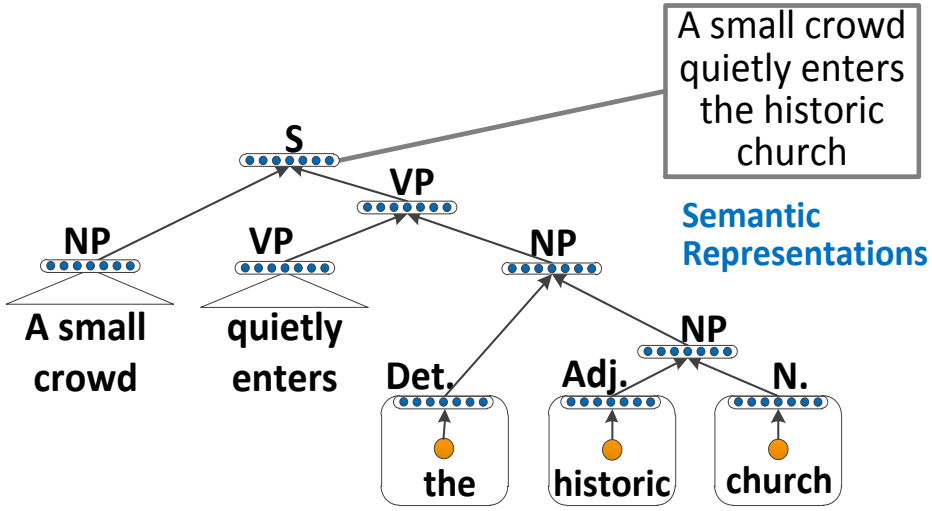
$\{y_1, y_2, \ldots y_n\}$ = successors of x

$$\frac{\partial z}{\partial x} = \sum_{i=1}^{n} \frac{\partial z}{\partial y_i} \frac{\partial y_i}{\partial x}$$

Back-Prop in Recurrent & Recursive Nets

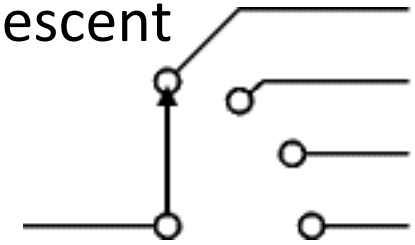
- Replicate a parameterized function over different time steps or nodes of a DAG
- Output state at one time-step / node is used as input for another time-step / node



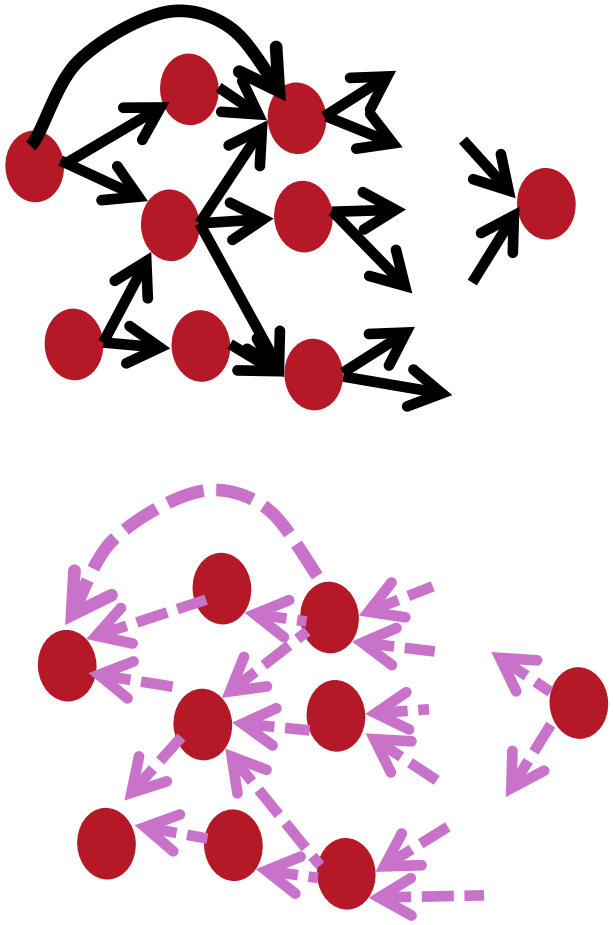


Backpropagation Through Structure

- Inference → discrete choices
  - (e.g., shortest path in HMM, best output configuration in CRF)
- E.g. Max over configurations or sum weighted by posterior
- The loss to be optimized depends on these choices
- The inference operations are flow graph nodes
- If continuous, can perform stochastic gradient descent
  - Max(a,b) is continuous.



Automatic Differentiation



- The gradient computation can be automatically inferred from the symbolic expression of the fprop.
- Each node type needs to know how to compute its output and how to compute the gradient wrt its inputs given the gradient wrt its output.
- Easy and fast prototyping

theano

Machine Learning 101

- Family of functions f_θ
- Tunable parameters θ
- Examples *Z* ~ unknown data generating distribution *P(Z)*
- Loss *L* maps *Z* and f_θ to a scalar
- Regularizer *R* (typically on depends on θ but possibly also on *Z*)
- Training criterion:
$$C(\theta) = \text{average}_{Z \sim \text{dataset}} L(f_\theta, Z) + R(\theta, Z)$$
- Approximate minimization algorithm to search for good θ
- Supervised learning:
  - *Z=(X,Y)* and $L = L(f_\theta(X), Y)$

Log-Likelihood for Neural Nets

- Estimating a conditional probability $P(Y|X)$
- Parametrize it by $P(Y|X) = P(Y|\omega = f_\theta(X))$
- Loss = $-\log P(Y|X)$
- E.g. Gaussian *Y*, $\omega = (\mu, \sigma)$

  typically only μ is the network output, depends on *X*

  Equivalent to MSE criterion:

  Loss = $-\log P(Y|X) = \log \sigma + ||f_\theta(X) - Y||^2/\sigma^2$
- E.g. Multinoulli *Y* for classification,

  $\omega_i = P(Y = i|x) = f_{\theta,i}(X) = \mathrm{softmax}_i(a(X))$

  Loss = $-\log \omega_Y = -\log f_{\theta,Y}(X)$

Multiple Output Variables

- If they are conditionally independent (given X), the individual prediction losses add up:

$$-\log P(Y|X) = -\log P(Y_1, \ldots Y_k|X) = -\log \prod_i P(Y_i|X) = -\sum_i \log P(Y_i|X)$$

- Likelihood if some Y_i's are missing: just ignore those losses

- If not conditionally independent, need to capture the conditional joint distribution
  - Example: output = image, sentence, tree, etc. $P(Y_1, \ldots Y_k|X)$
  - Similar to unsupervised learning problem of capturing joint
  - Exact likelihood may similarly be intractable, depending on model

Deep Supervised Neural Nets, Rectifiers

- Now can train them even witho unsupervised pre-training:

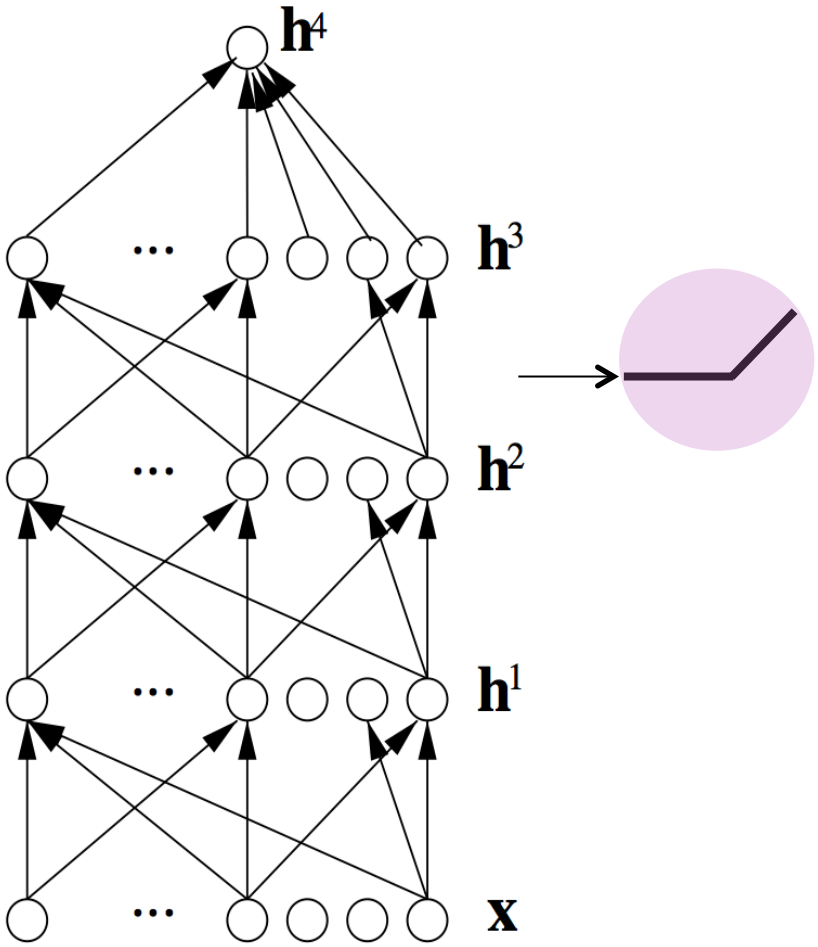
  better initialization and non-linearities (rectifiers, maxout),

  (Glorot & Bengio AISTATS 2011; Goodfellow et al ICML 2013)

  generalize well with large labe sets and regularizers (dropout)

- **Unsupervised pre-training:**

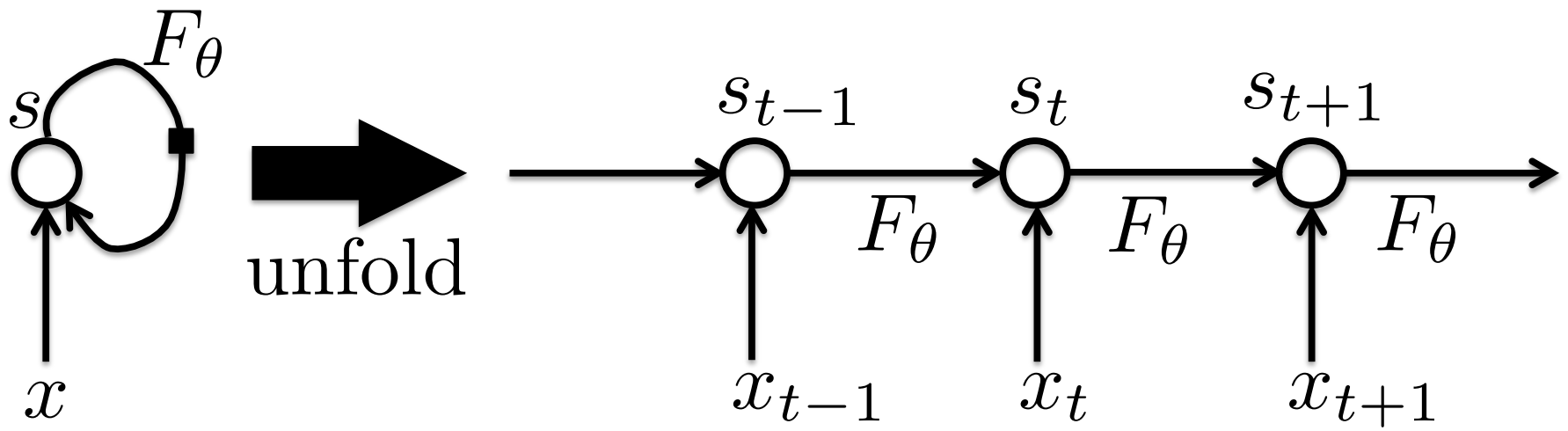
  rare classes, transfer, smaller labeled sets, or as extra regularizer.



Recurrent Neural Networks

- Selectively summarize an input sequence in a fixed-size state vector via a recursive update

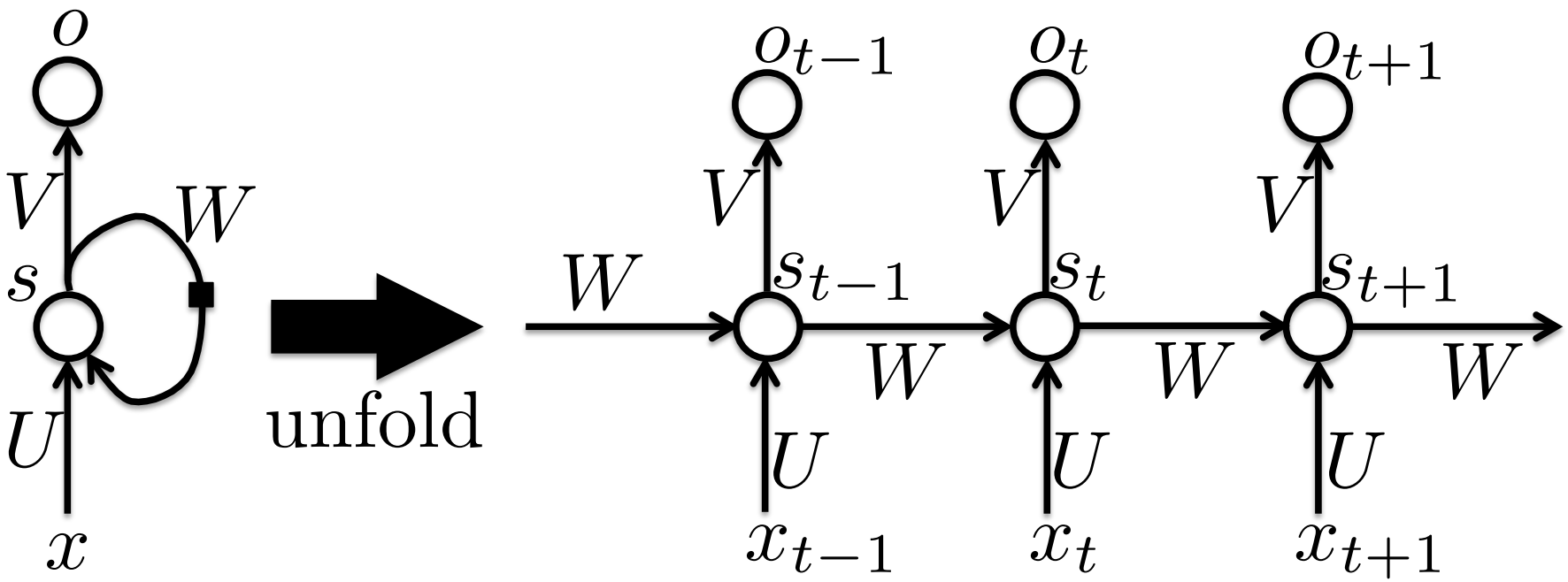
$$s_t = F_\theta(s_{t-1}, x_t)$$



$$s_t = G_t(x_t, x_{t-1}, x_{t-2}, \ldots, x_2, x_1)$$

Recurrent Neural Networks

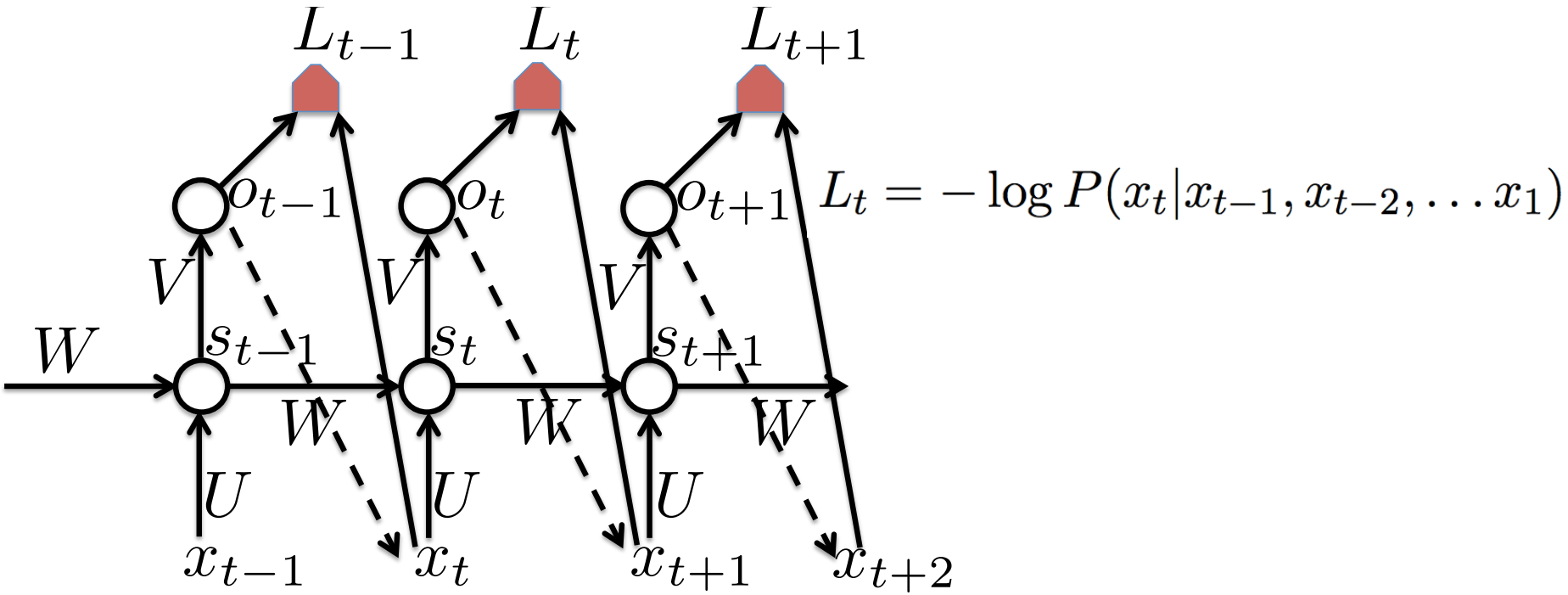
- Can produce an output at each time step: unfolding the graph tells us how to back-prop through time.



Generative RNNs

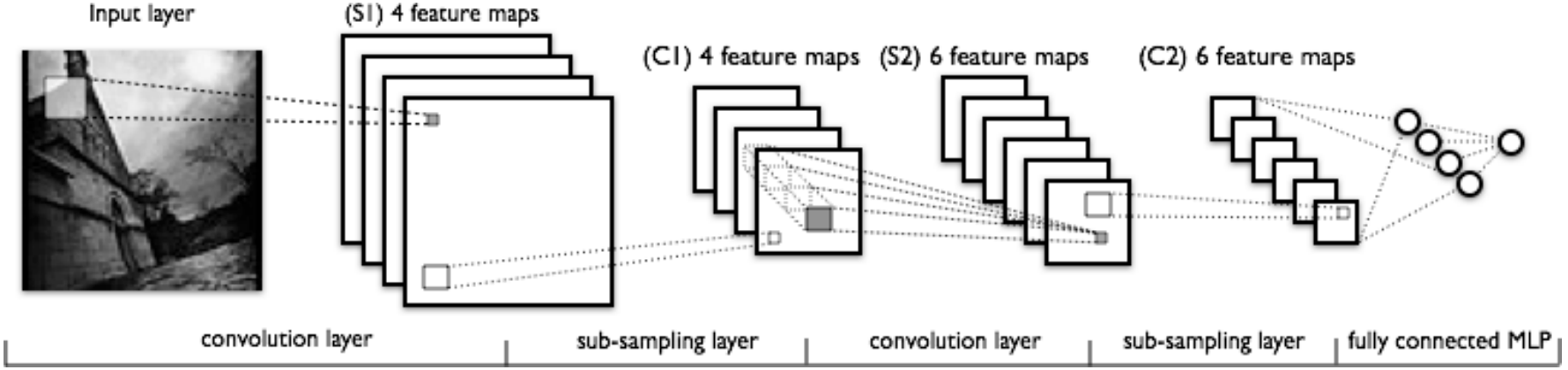
- An RNN can represent a fully-connected directed generative model: every variable predicted from all previous ones.

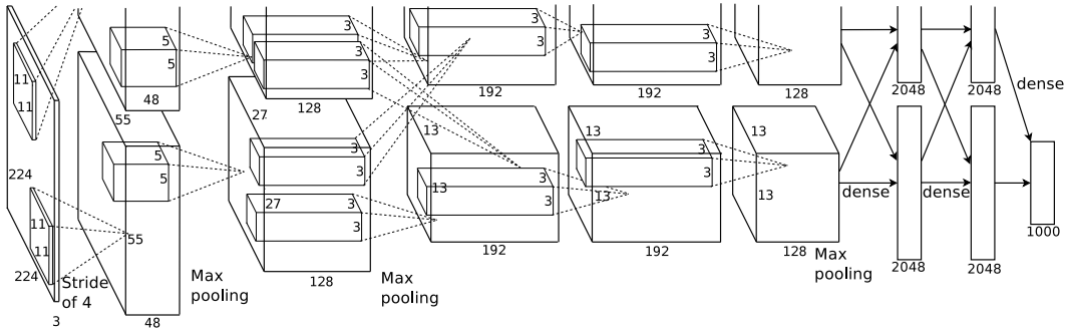
$$P(\mathbf{x}) = P(x_1, \ldots x_T) = \prod_{t=1}^{T} P(x_t | x_{t-1}, x_{t-2}, \ldots x_1)$$

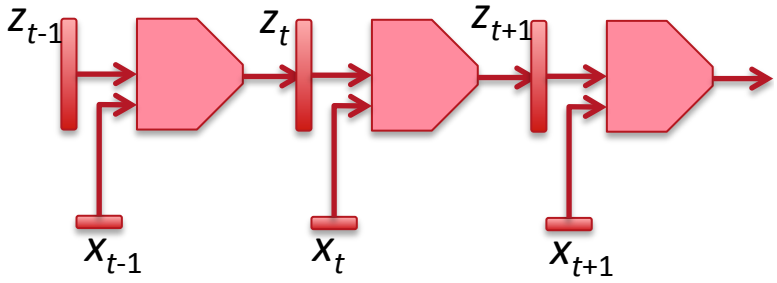


Temporal & Spatial Inputs: Convolutional & Recurrent Nets

- Local connectivity across time/space
- Sharing weights across time/space (translation equivariance)
- Pooling (translation invariance, cross-channel pooling for learned invariances)





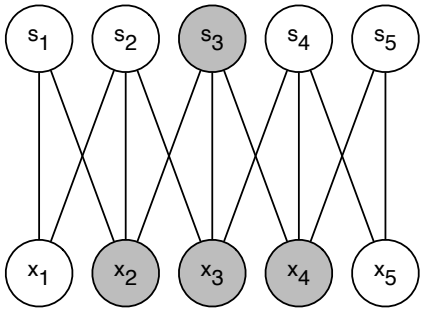


Recurrent nets (RNNs) can summarize information from the past

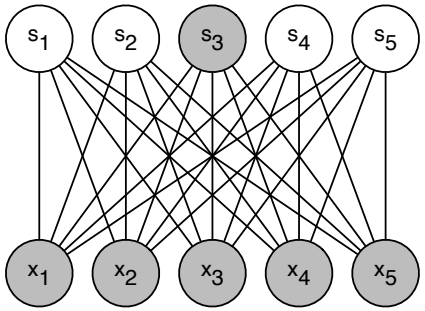
Bidirectional RNNs also summarize information from the future

Convolution = sparse connectivity + parameter sharing

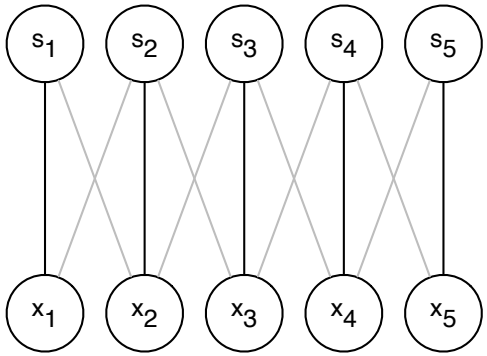
$$s[t] = (x * w)(t) = \sum_{a=-\infty}^{\infty} x[a]w[t-a]$$



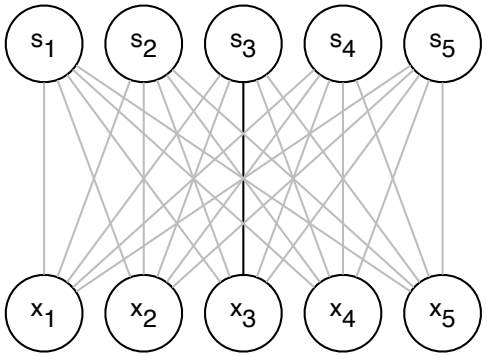
sparse



dense



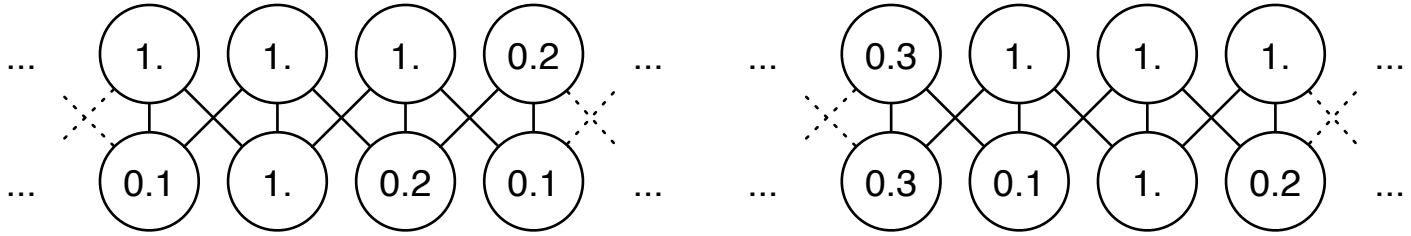
shared



not shared

Pooling Layers

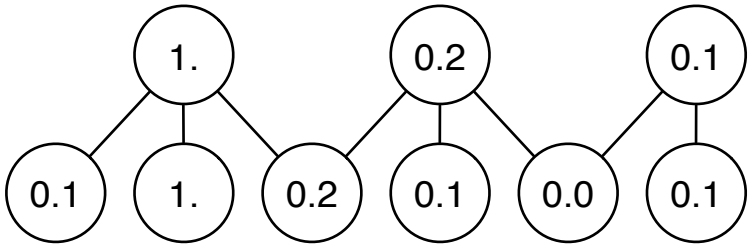
- Aggregate to achieve local invariance



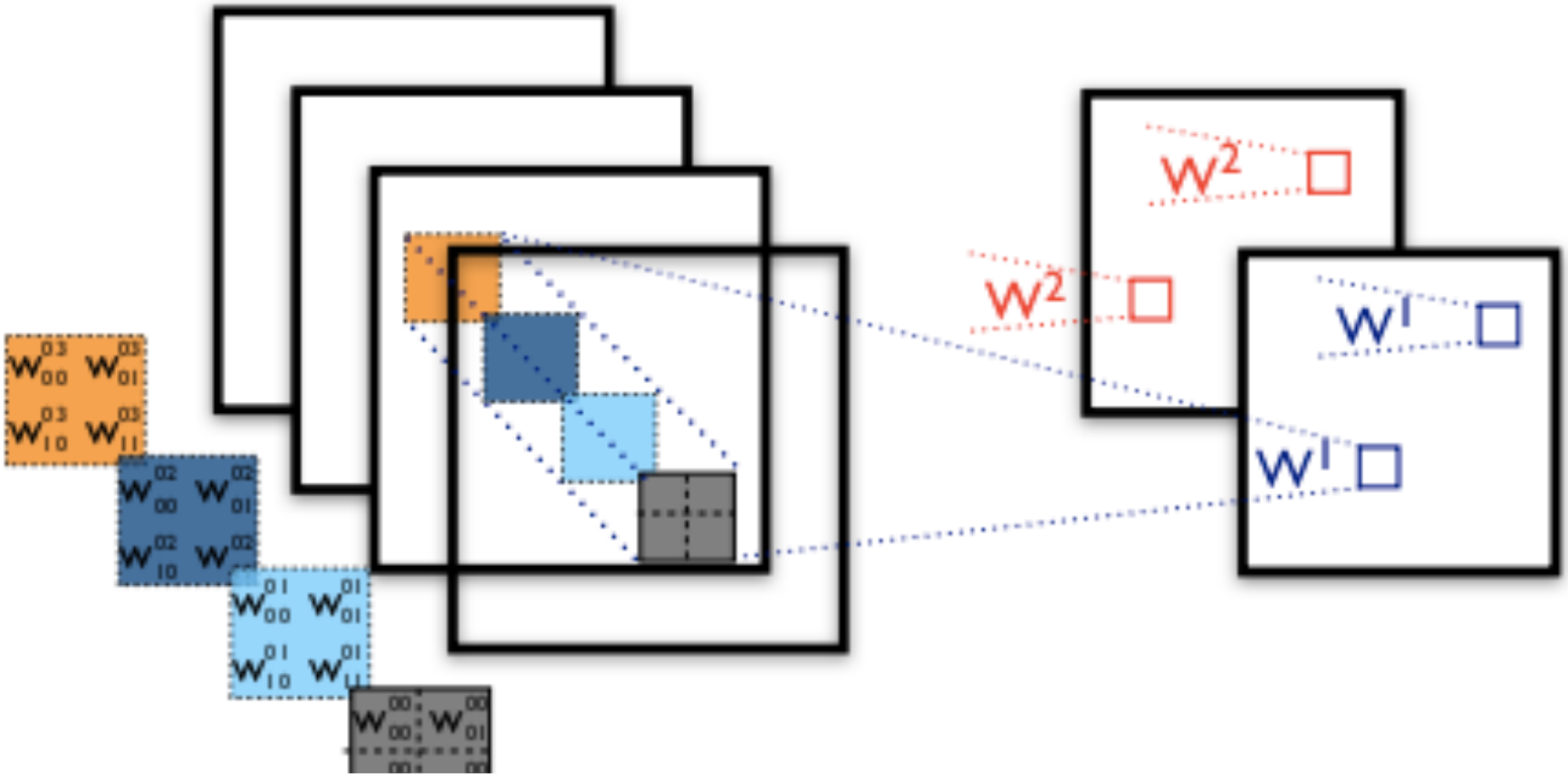
Max-pooling

Effect of translation

- Subsampling to reduce temporal/spatial scale and computation

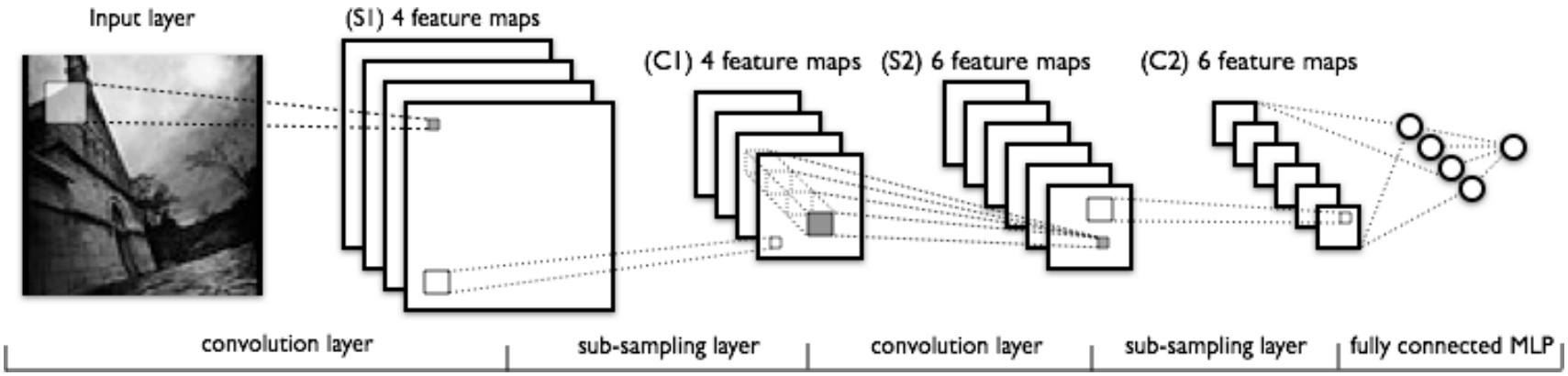


Multiple Convolutions: Feature Maps



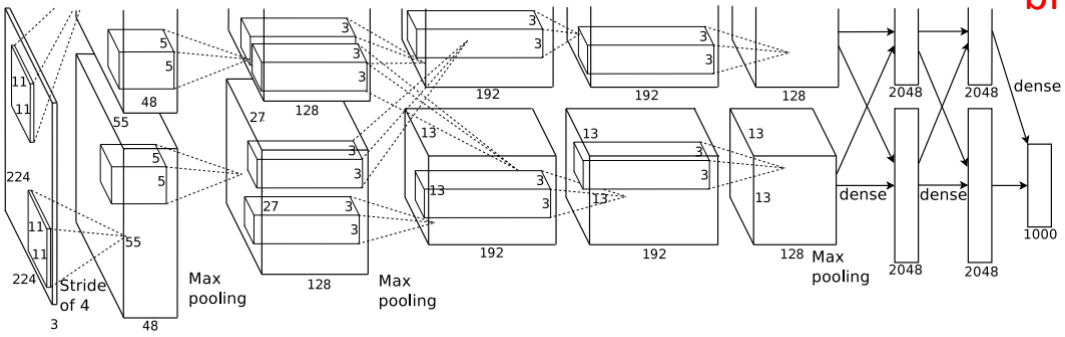
Alternating convolutions & pooling

- Inspired by visual cortex, idea from Fukushima's Neocognitron, combined with back-prop and developped by **LeCun** since 1989

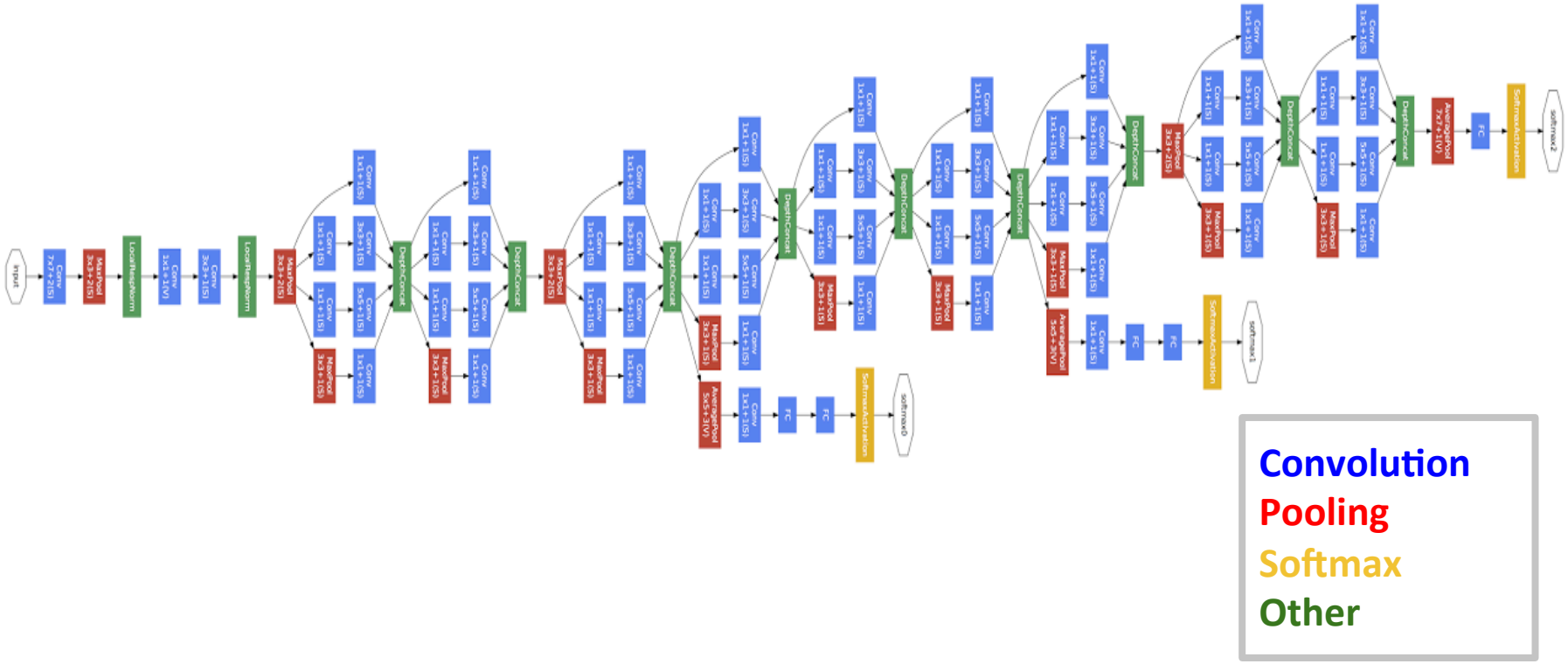


- Increasing number of features, decreasing spatial resolution
- Top layers are fully connected

Krizhevsky, Sutskever & Hinton 2012
breakthrough in object recognition



GoogLeNet: 22 layers, intermediate targets

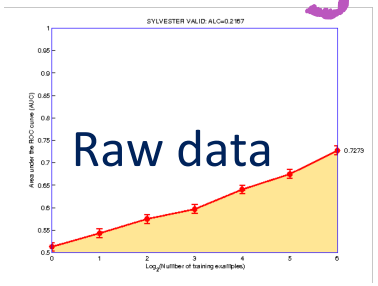


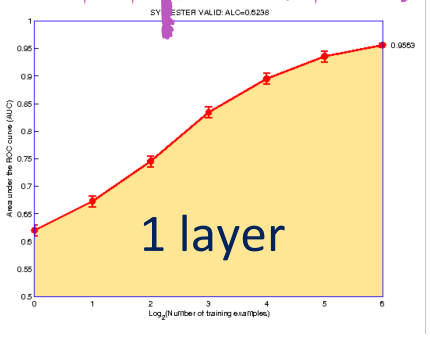
Unsupervised or Semi-Supervised Deep Learning & Generative Deep Learning

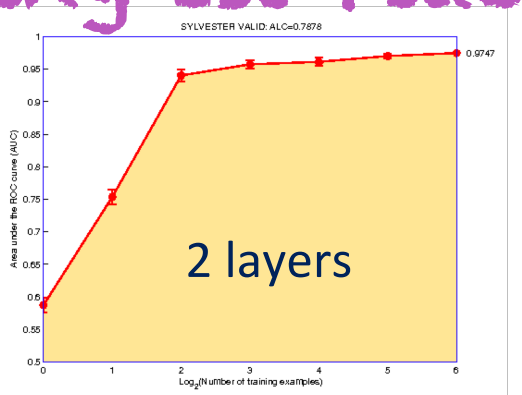
The Next Challenge: Unsupervised Learning

- Recent progress mostly in supervised DL
- Real technical challenges for unsupervised DL
- Potential benefits:
  - Exploit tons of unlabeled data
  - Answer new questions about the variables observed
  - Regularizer – transfer learning – domain adaptation
  - Easier optimization (local training signal)
  - Structured outputs

Unsupervised and Transfer Learning Challenge + Transfer Learning Challenge: Deep Learning 1st Place

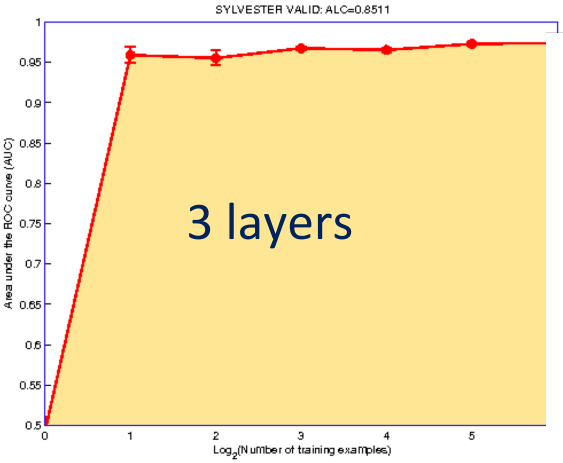






NIPS'2011 Transfer Learning Challenge Paper: ICML'2012

ICML'2011 workshop on Unsup. & Transfer Learning



Why Latent Factors & Unsupervised Representation Learning? Because of Causality.

- If Ys of interest are among the causal factors of X, then

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

is tied to P(X) and P(X|Y), and P(X) is defined in terms of P(X|Y), i.e.

- The best possible model of X (unsupervised learning) MUST involve Y as a latent factor, implicitly or explicitly.
- Representation learning SEEKS the latent variables H that explain the variations of X, making it likely to also uncover Y.

Invariance and Disentangling

- Invariant features

- Which invariances?
- Alternative: learning to disentangle factors
- Good disentangling →

  avoid the curse of dimensionality

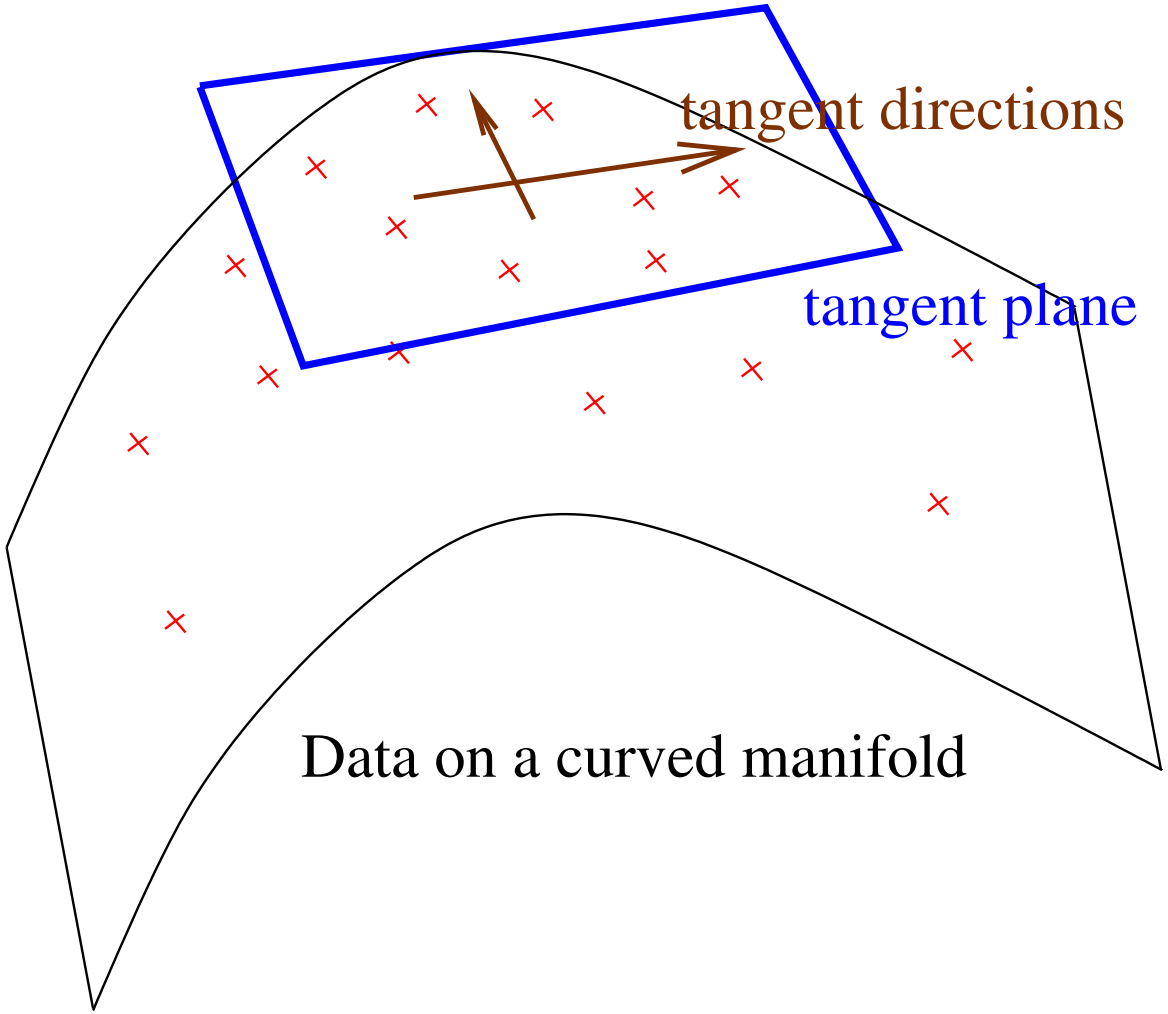
Emergence of Disentangling

- (Goodfellow et al. 2009): sparse auto-encoders trained on images
  - some higher-level features more invariant to geometric factors of variation

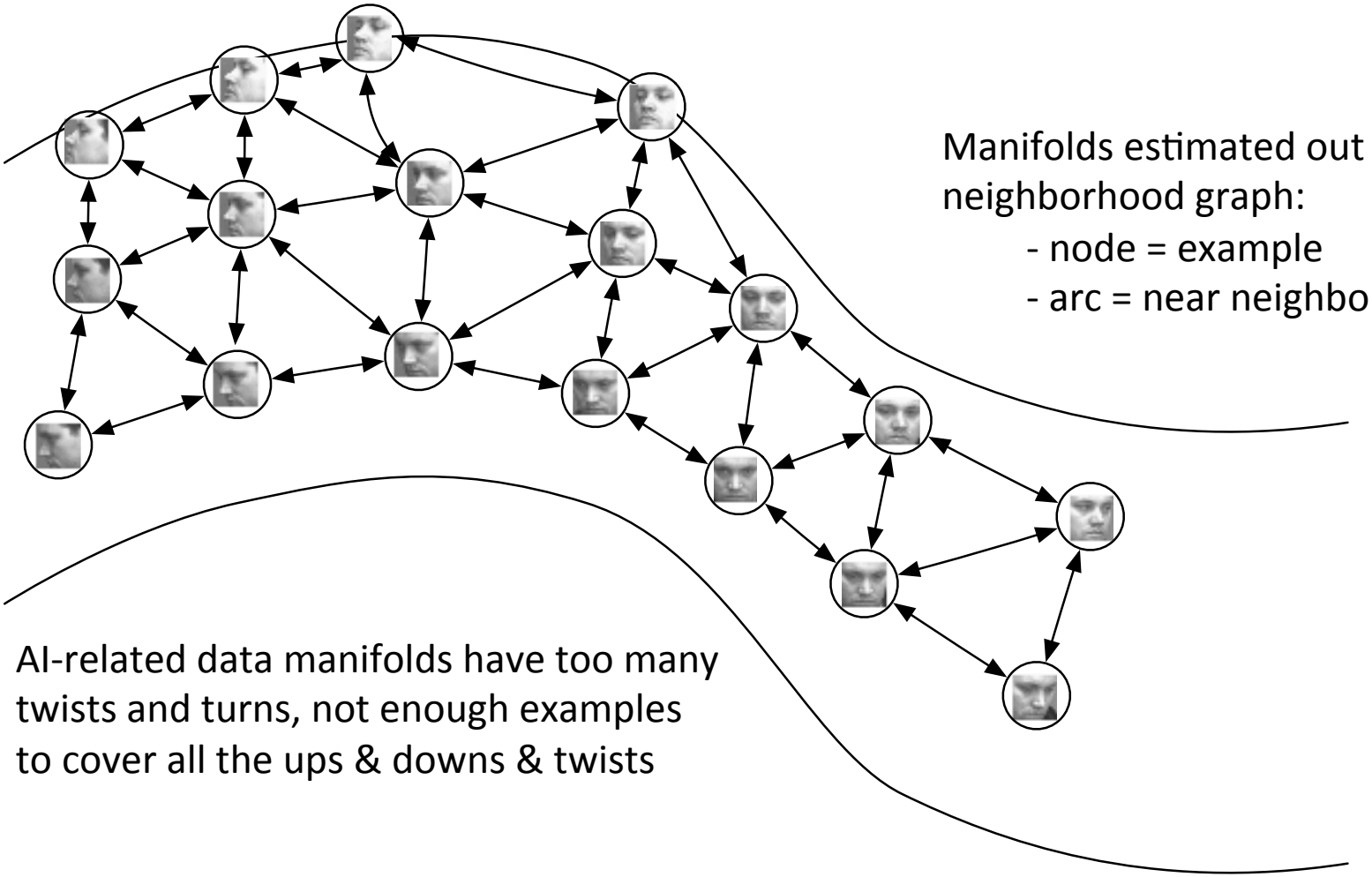
- (Glorot et al. 2011): sparse rectified denoising auto-encoders trained on bags of words for sentiment analysis
  - different features specialize on different aspects (domain, sentiment)

WHY?

Manifold Learning = Representation Learning



Non-Parametric Manifold Learning: hopeless without powerful enough priors

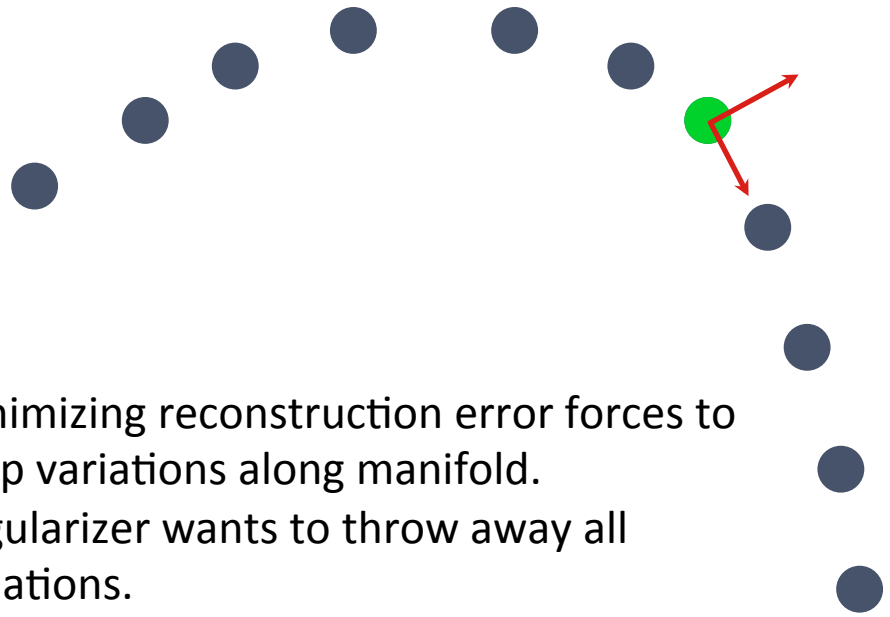


Manifolds estimated out of the neighborhood graph:

- node = example
- arc = near neighbor

AI-related data manifolds have too many twists and turns, not enough examples to cover all the ups & downs & twists

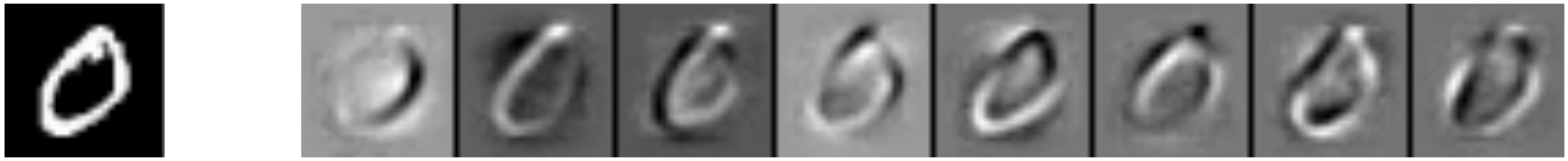
Auto-Encoders Learn Salient Variations, like a non-linear PCA

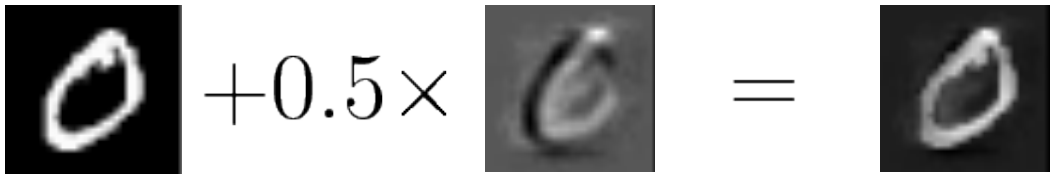


- Minimizing reconstruction error forces to keep variations along manifold.
- Regularizer wants to throw away all variations.
- With both: keep ONLY sensitivity to variations ON the manifold.

Input Point

Tangents

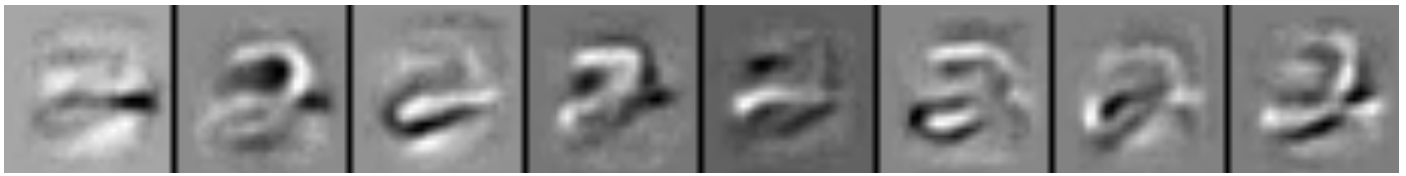




MNIST

Input Point

Tangents

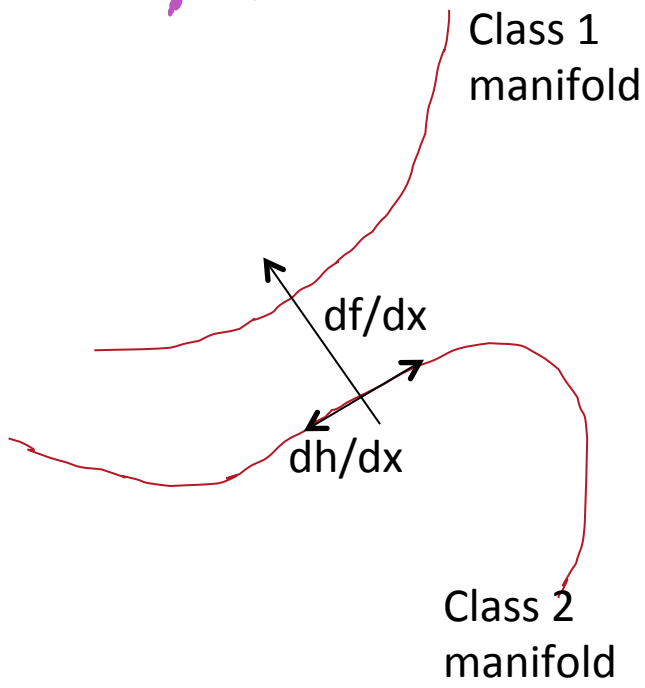


MNIST Tangents

Learned Tangent Prop: the Manifold Tangent Classifier

Makes classifier f(x) insensitive to variations on manifold at x

Tangent plane characterized by dh(x)/dx



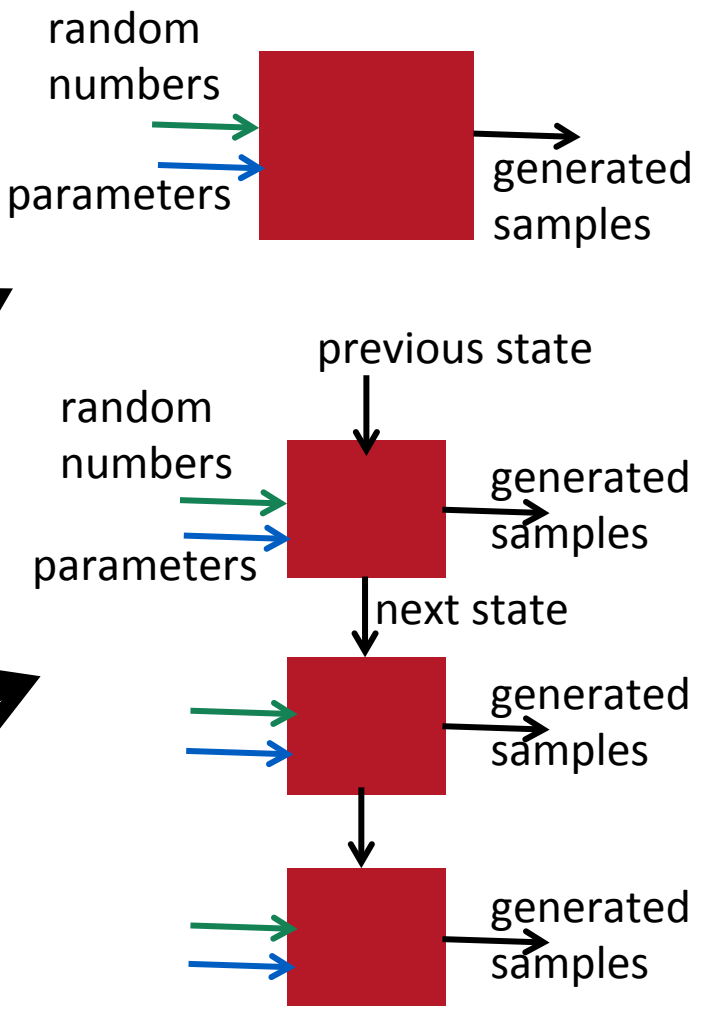
(Rifai et al NIPS'2012)

Bypassing Normalization Constants with Generative Black Boxes

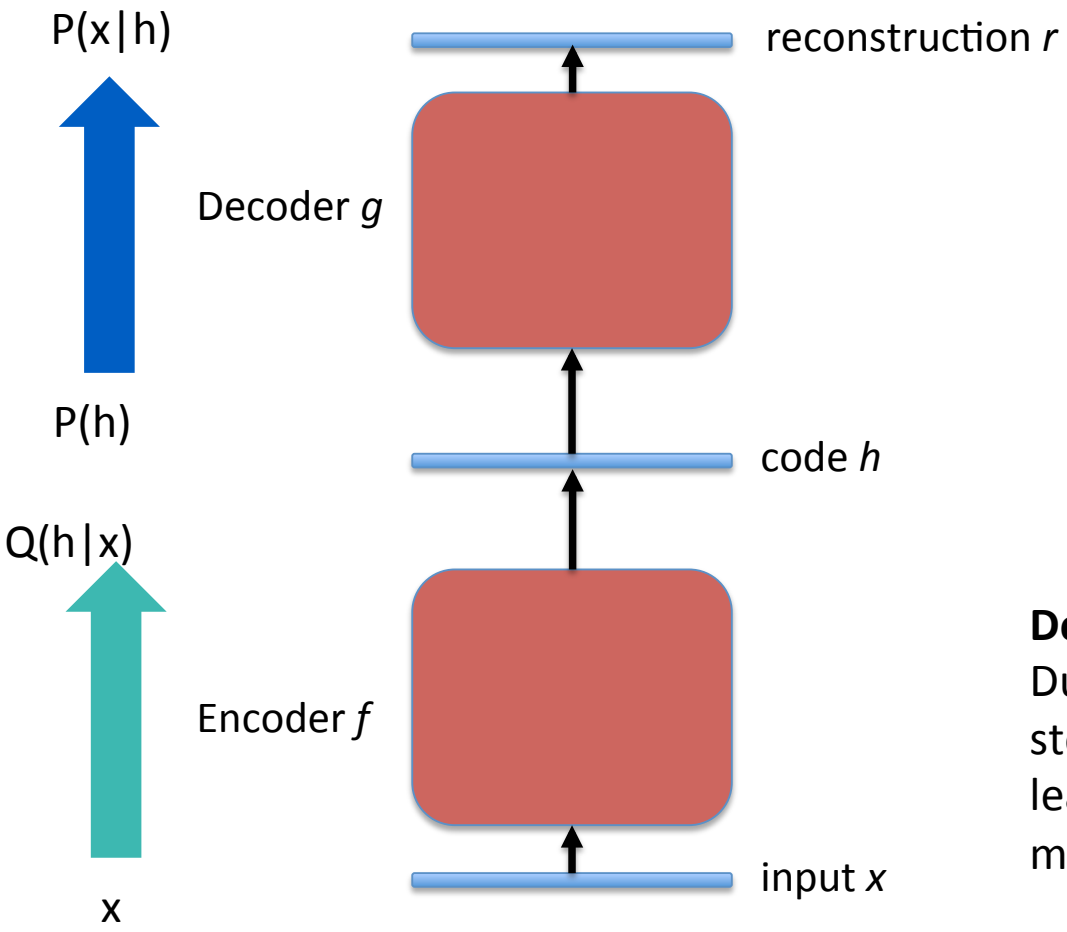
- **Instead of parametrizing p(x), parametrize a machine which generates samples**

- (Goodfellow et al, NIPS 2014, Generative adversarial nets) for the case of ancestral sampling in a deep generative net. Variational auto-encoders are closely related.

- (Bengio et al, ICML 2014, Generative Stochastic Networks), learning the transition operator of a Markov chain that generates the data.



Auto-Encoders



Probabilistic criterion:

Reconstruction log-likelihood =

- log P(x | h)

Denoising auto-encoder:
During training, input is corrupted stochastically, and auto-encoder must learn to guess the distribution of the missing information.

Denoising Auto-Encoder

- Learns a vector field pointing towards higher probability direction (Alain & Bengio 2013)

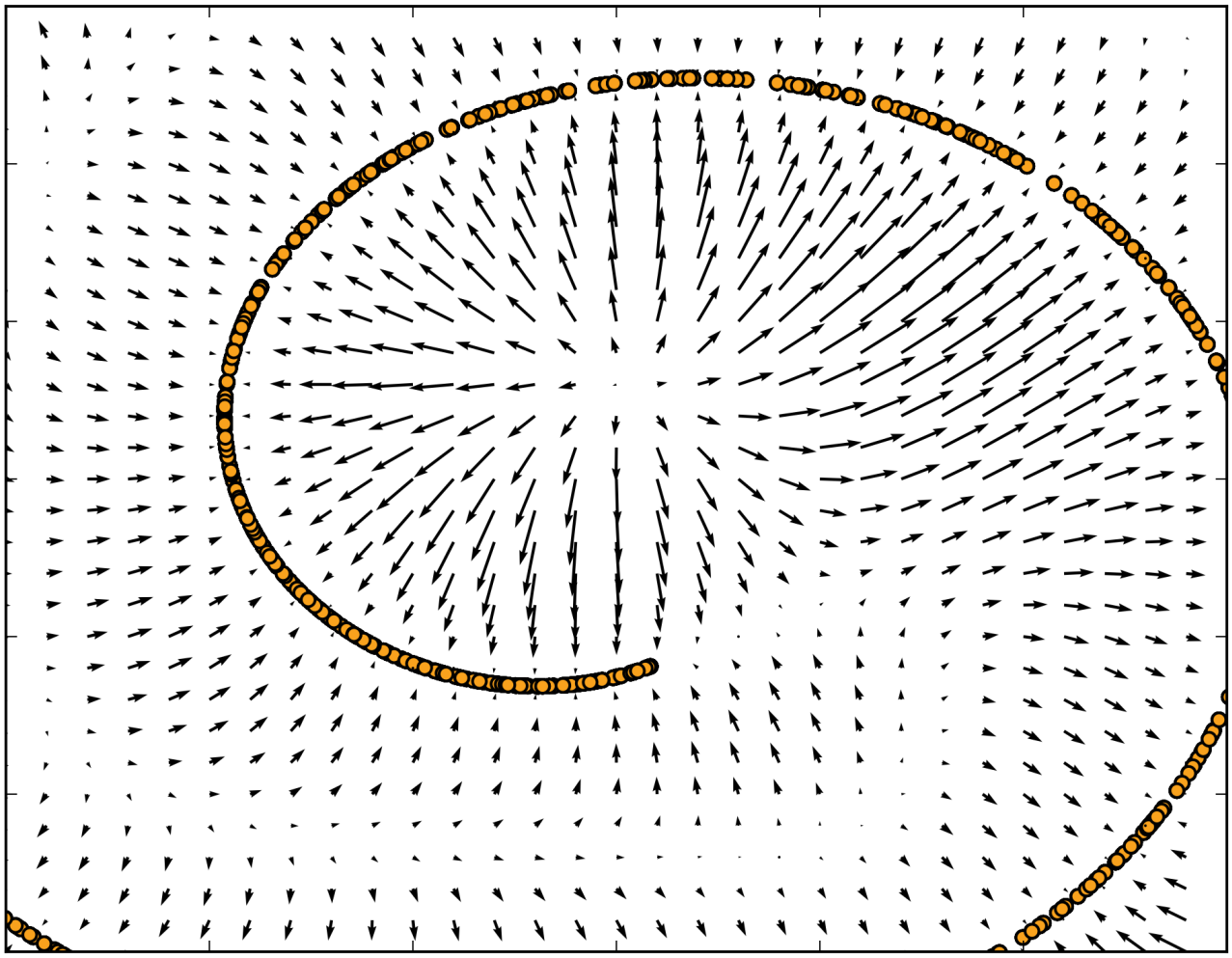
$$\text{reconstruction}(x) - x \quad \rightarrow \quad \sigma^2 \frac{\partial \log p(x)}{\partial x}$$

- Some DAEs correspond to a kind of Gaussian RBM with *regularized* Score Matching (Vincent 2011)

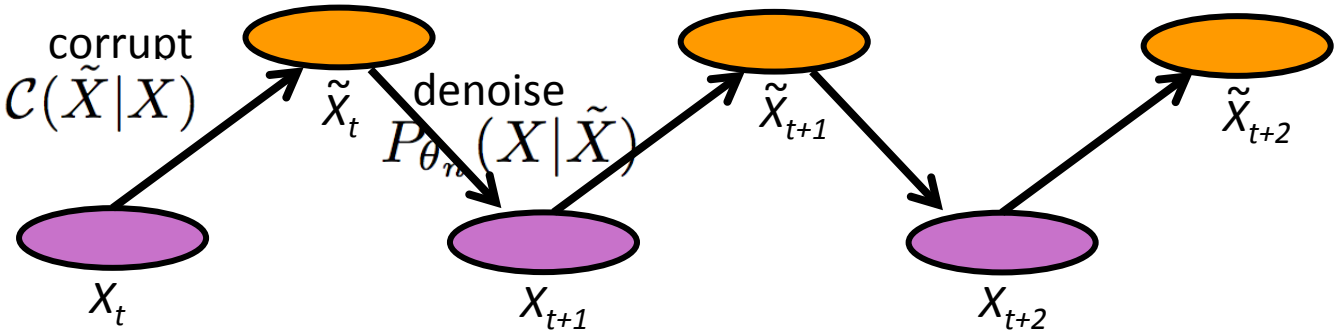
  [equivalent when noise→0]

prior: examples concentrate near a lower dimensional "manifold"

Regularized Auto-Encoders Learn a Vector Field that Estimates a Gradient Field (Alain & Bengio ICLR 2013)

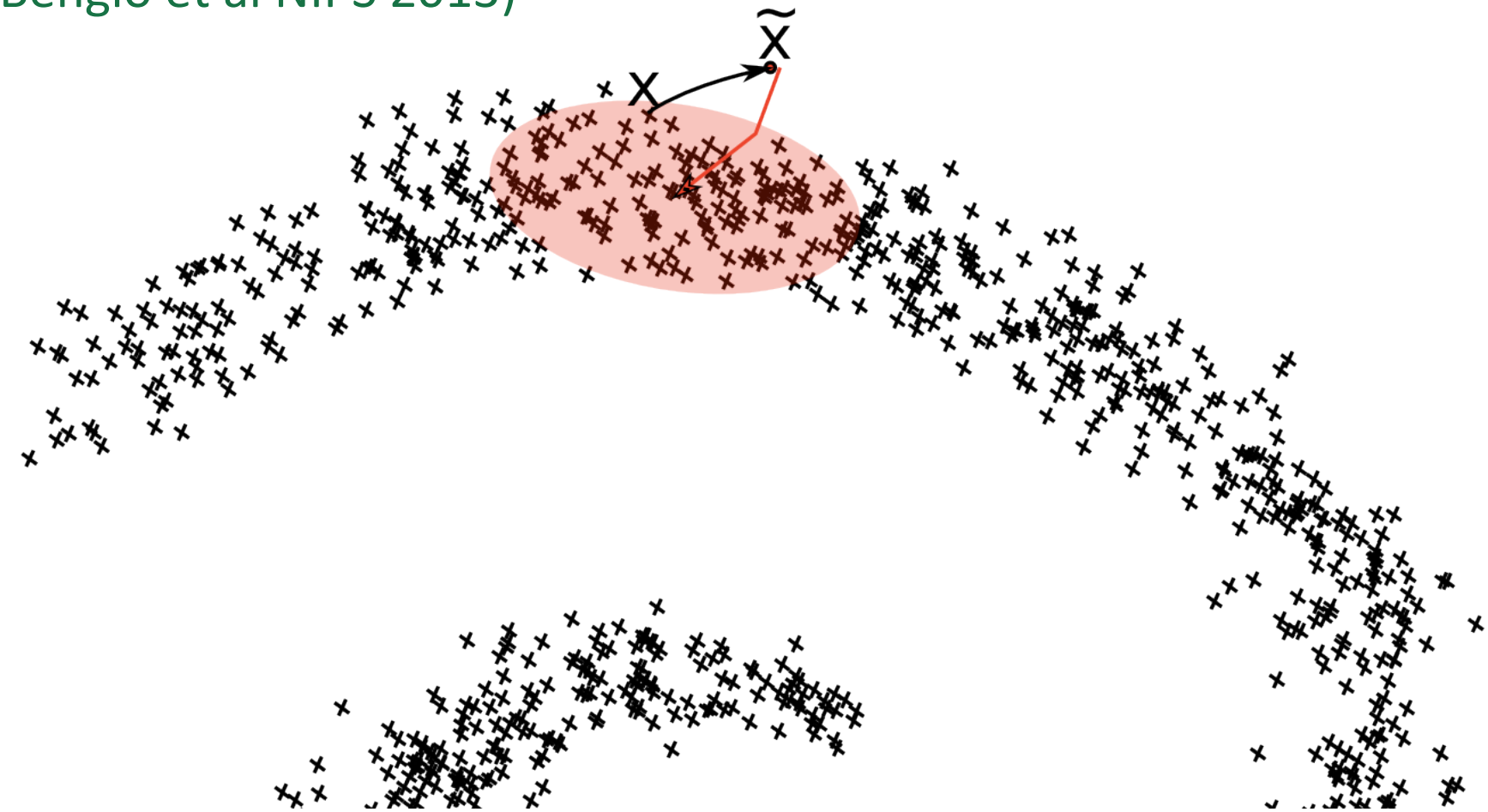


Denoising Auto-Encoder Markov Chain



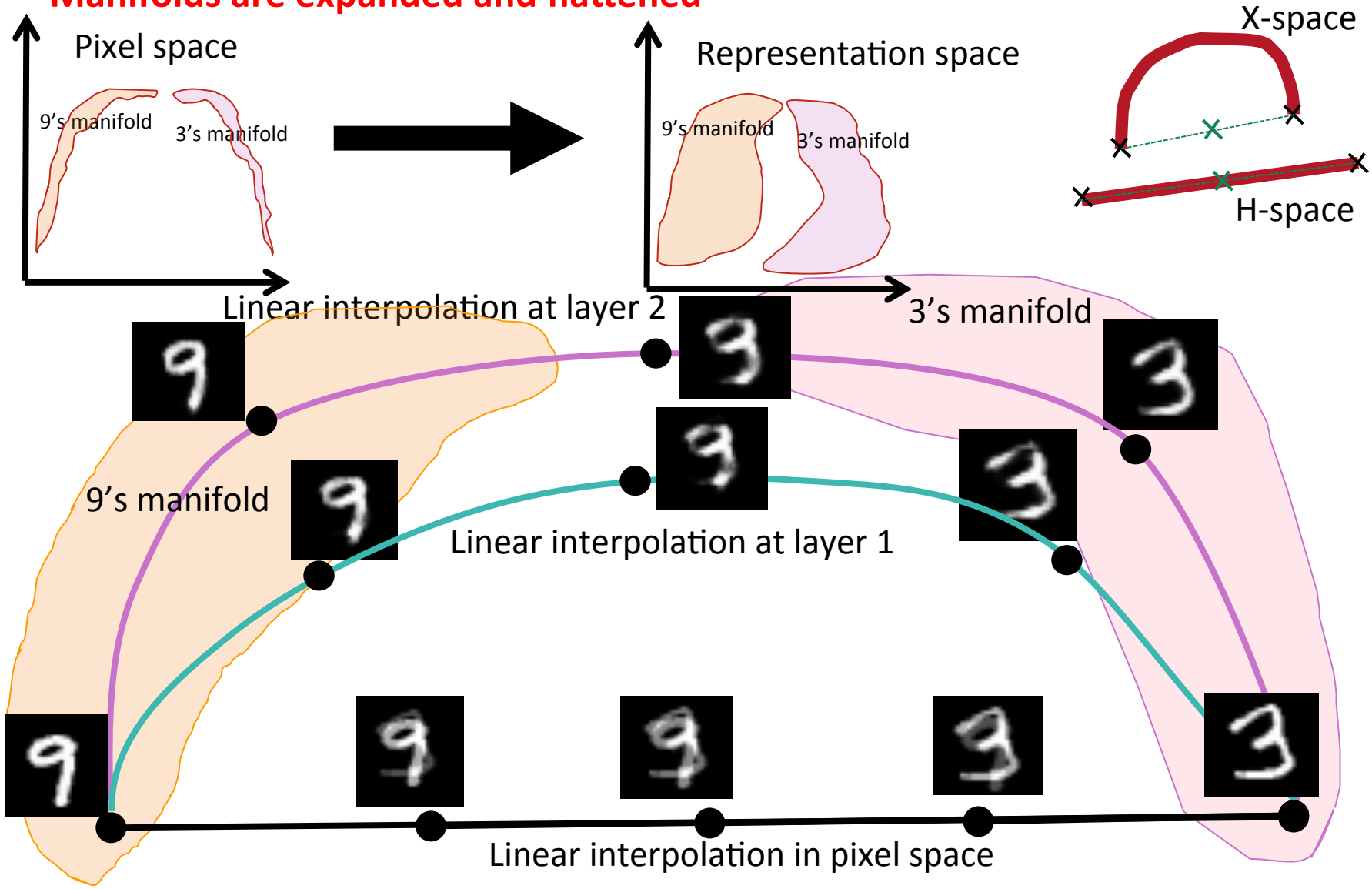
Denoising Auto-Encoders Learn a Markov Chain Transition Distribution

(Bengio et al NIPS 2013)



Space-Filling in Representation-Space

- **Deeper representations → abstractions → disentangling**
- **Manifolds are expanded and flattened**

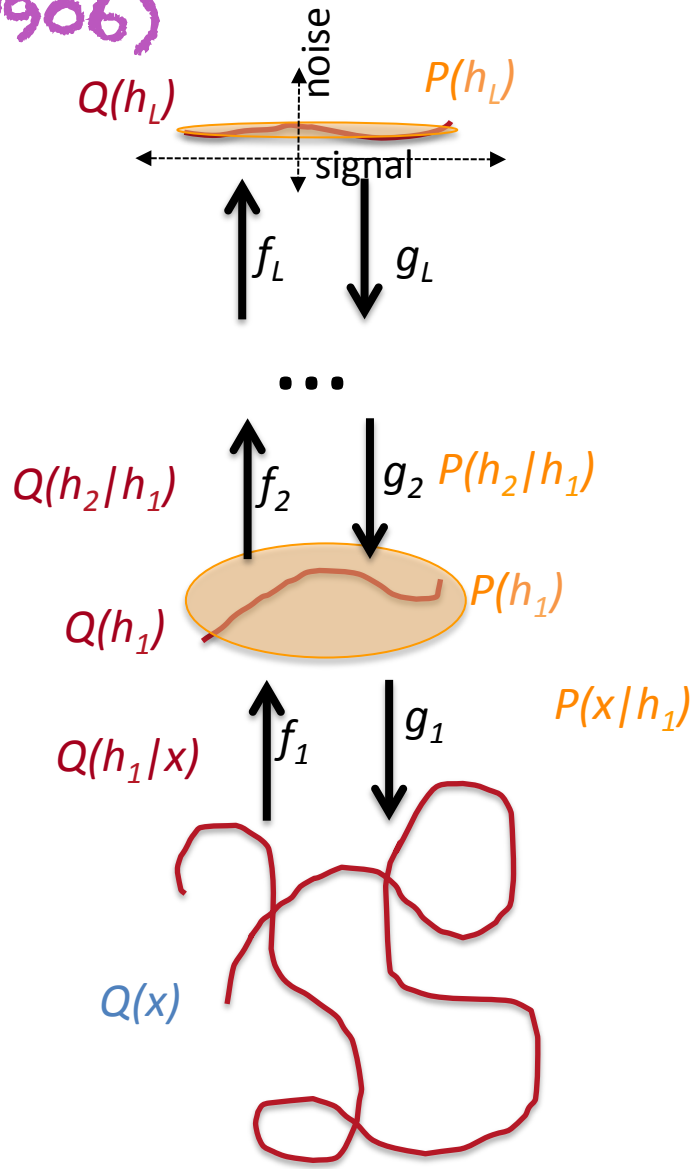


Extracting Structure By Gradual Disentangling and Manifold Unfolding (Bengio 2014, arXiv 1407.7906)

Each level transforms the data into a representation in which it is easier to model, unfolding it more, contracting the noise dimensions and mapping the signal dimensions to a factorized (uniform-like) distribution.

$$\min KL(Q(x,h)||P(x,h))$$

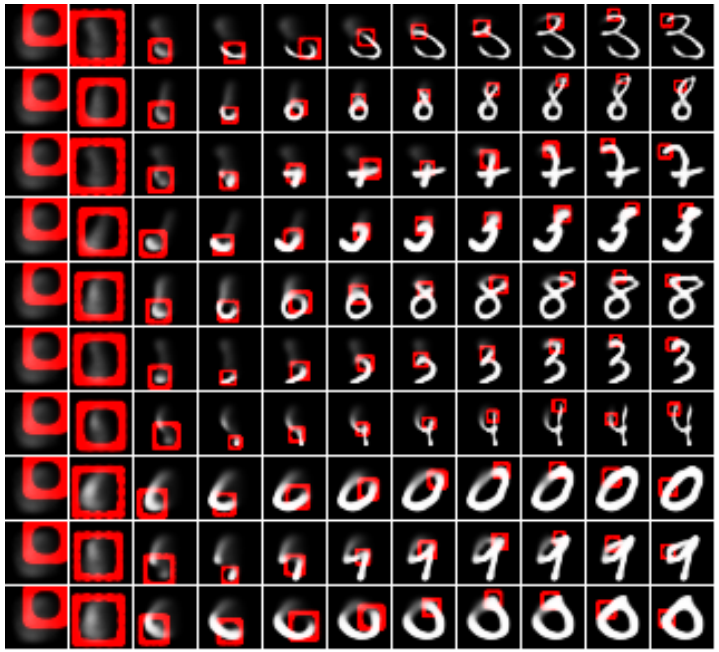
for each intermediate level h

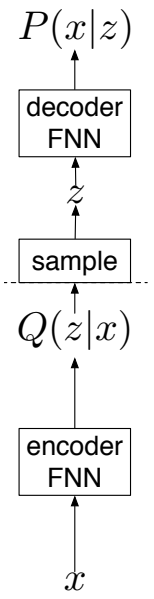


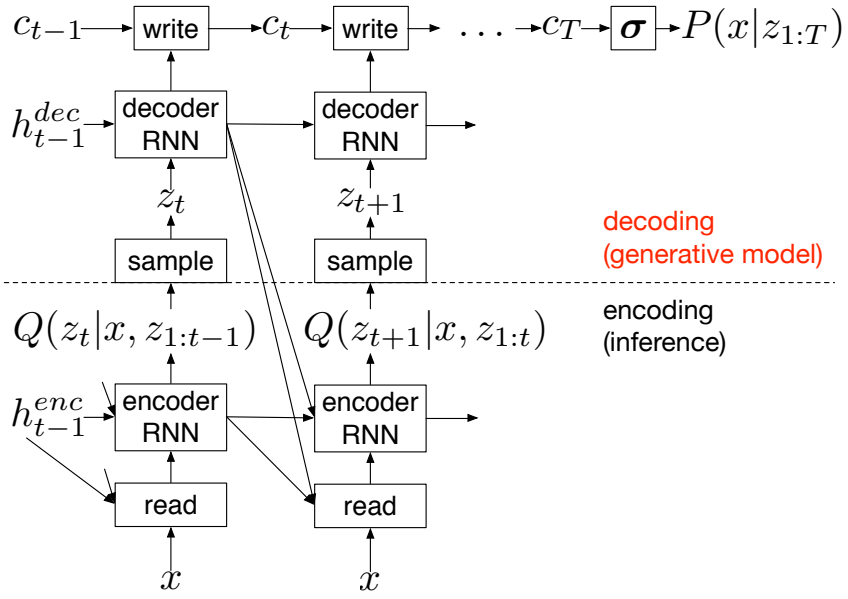
DRAW: the latest variant of Variational Auto-Encoder

(Gregor et al of Google DeepMind, arXiv 1502.04623, 2015)

- Even for a static input, the encoder and decoder are now **recurrent nets**, which gradually add elements to the answer, and use an attention mechanism to choose where to do so.







DRAW Samples of SVHN Images: the drawing process

DRAW Samples of SVHN Images: generated samples vs training nearest neighbor

Nearest training example for last column of samples

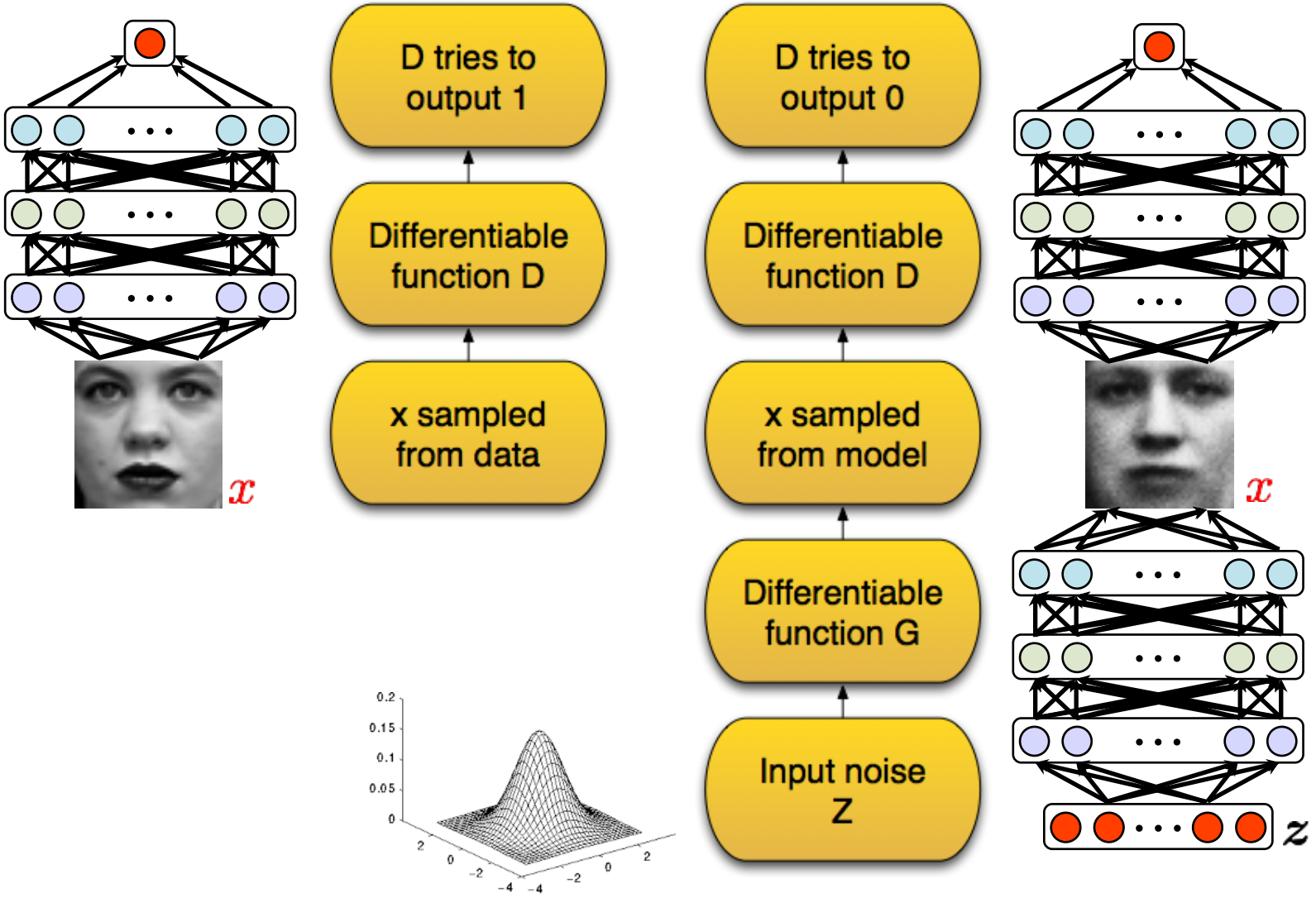
Generative adversarial networks

- Don't write a formula for $p(x)$, just learn to sample directly.
- No Markov Chain
- No variational bound
- How? By playing a game.

Adversarial nets framework

- A game between two players:
  1. Discriminator D
  2. Generator G
- D tries to discriminate between:
  - A sample from the data distribution.
  - And a sample from the generator G.
- G tries to "trick" D by generating samples that are hard for D to distinguish from data.

Adversarial nets framework



Zero-sum game

- Minimax value function:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Discriminator pushes up

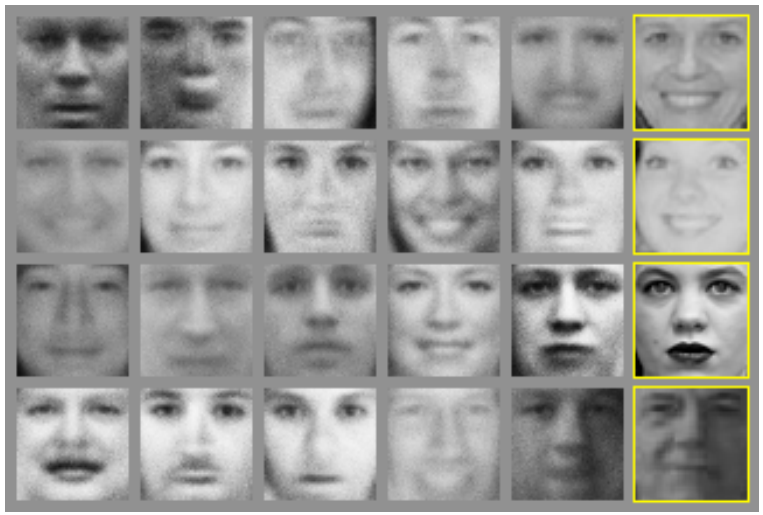
Discriminator's ability to recognize data as being real

Discriminator's ability to recognize generator samples as being fake

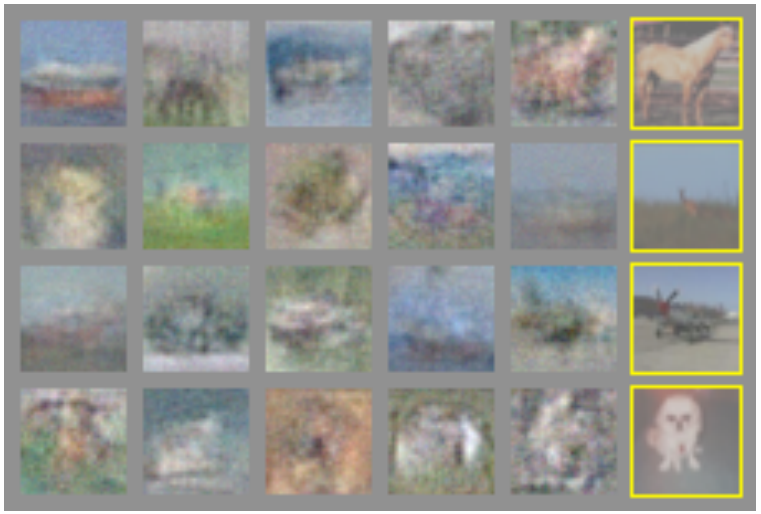
Generator pushes down

Visualization of model samples

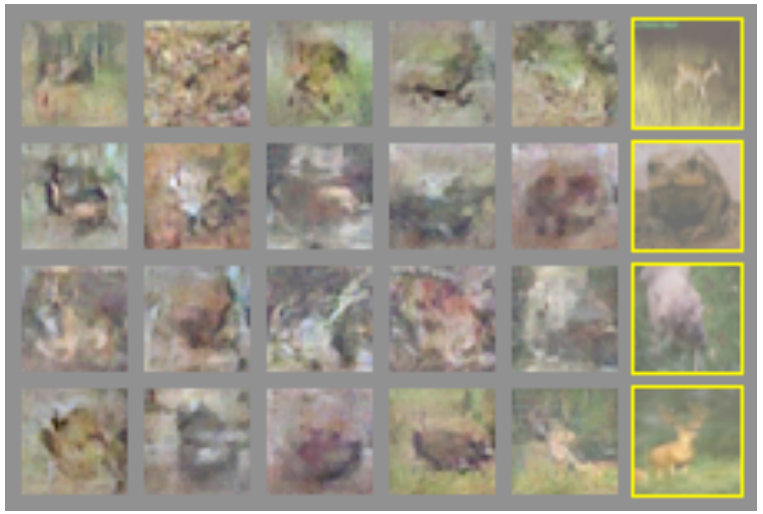
MNIST



TFD

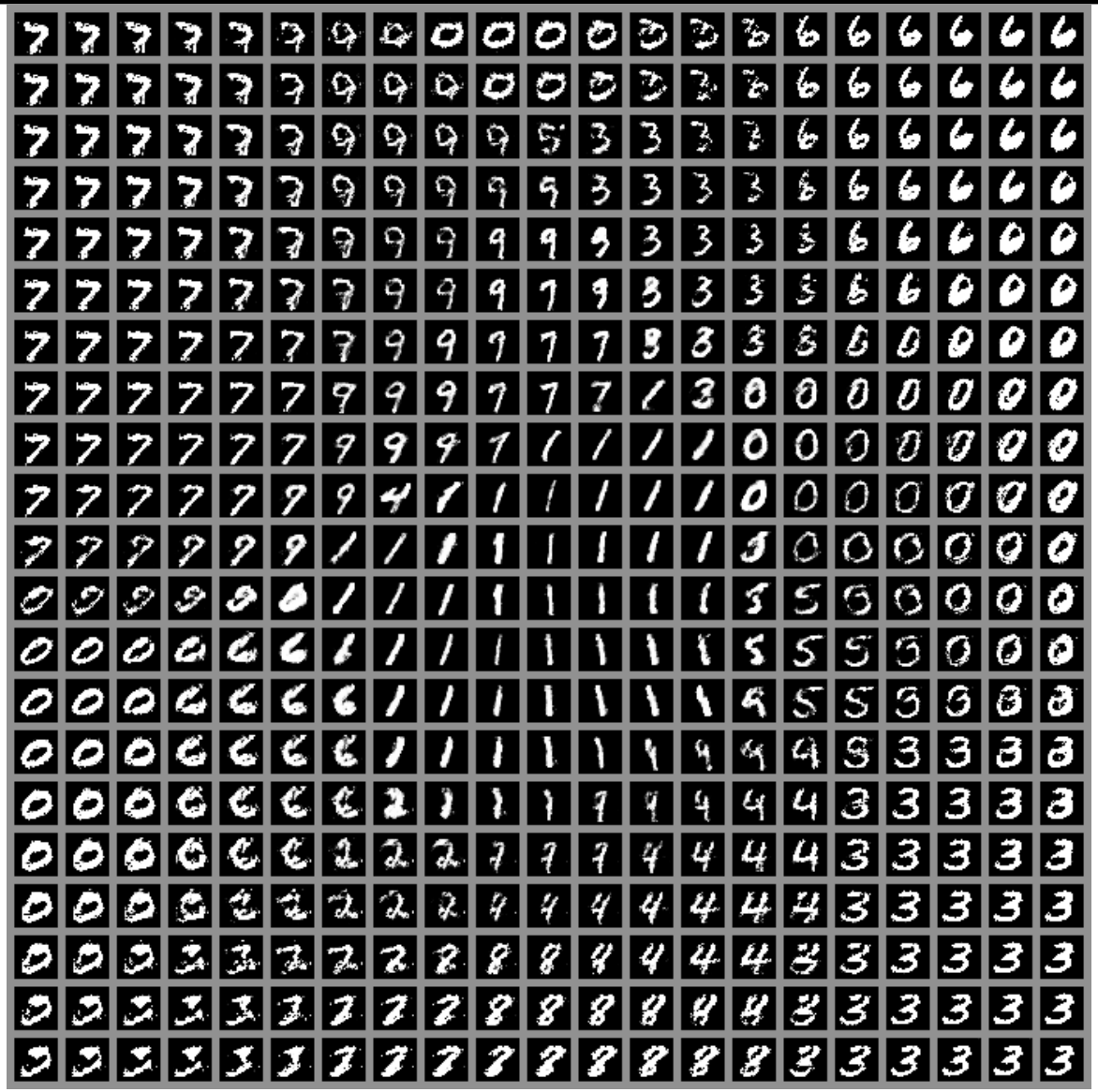


CIFAR-10 (fully connected)

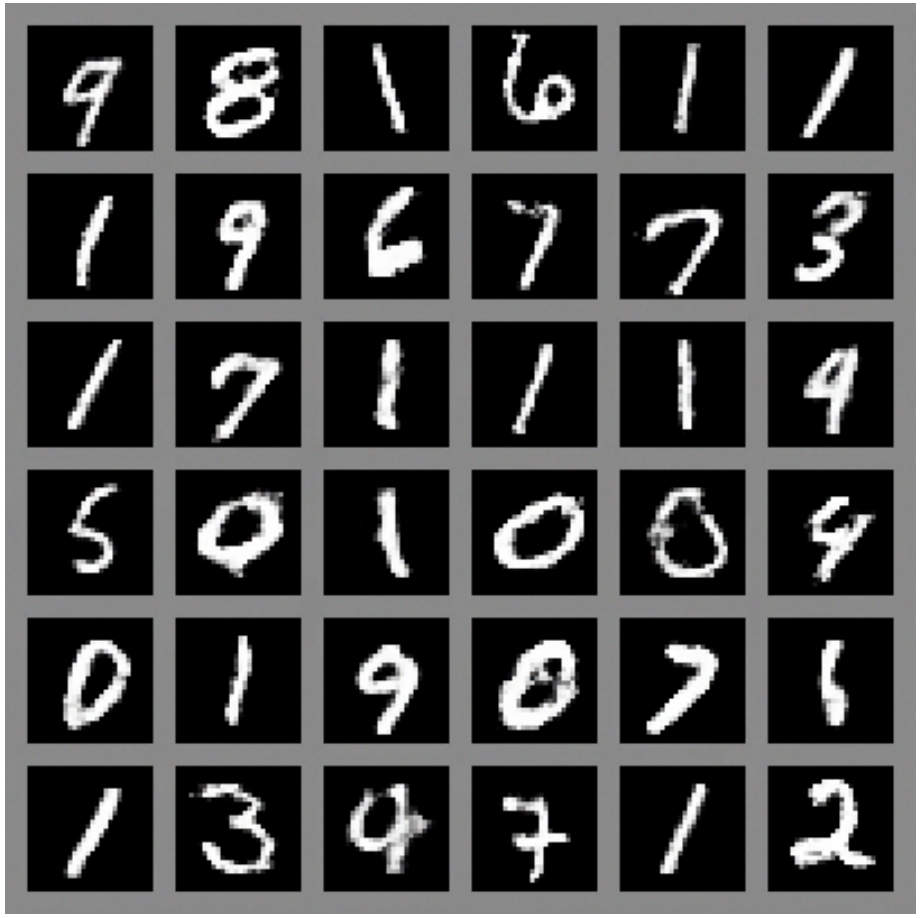


CIFAR-10 (convolutional)

Learned 2-D manifold of MNIST



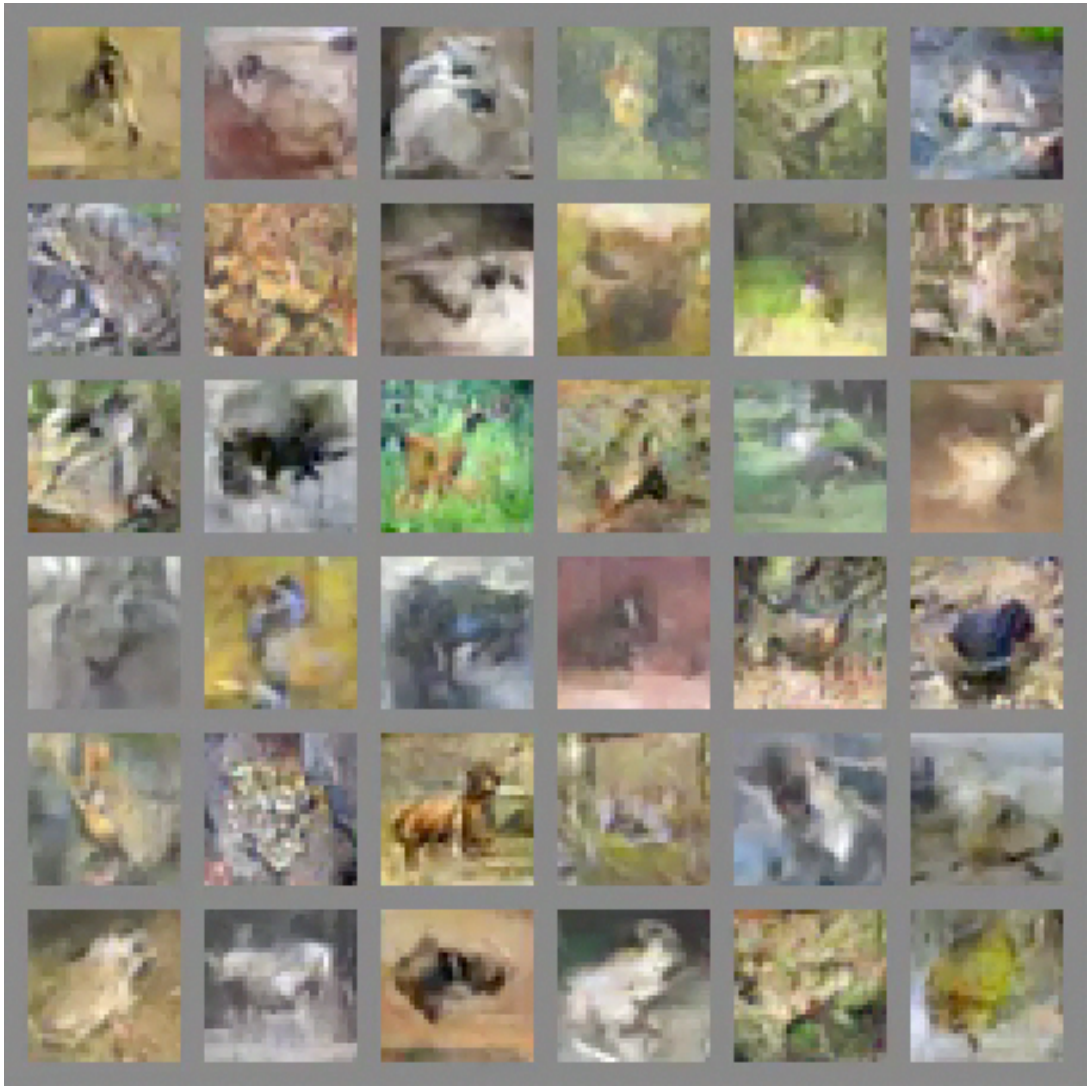
Visualization of model trajectories



MNIST digit dataset

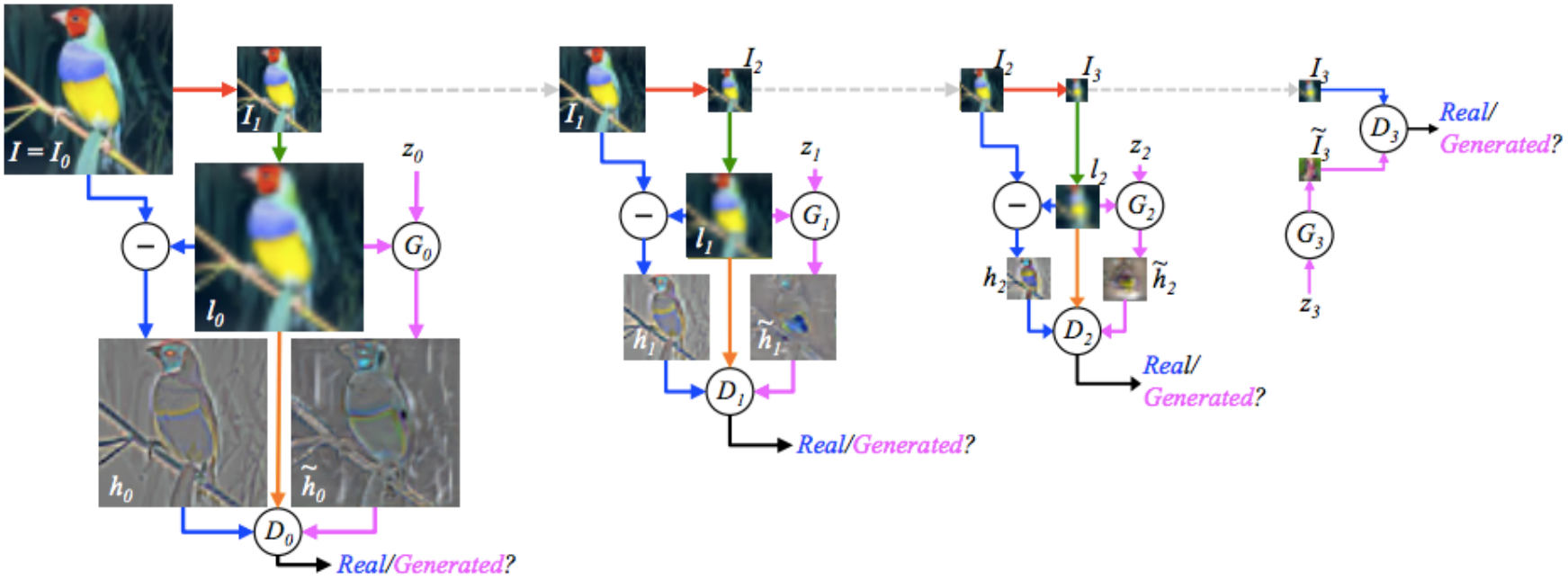
Toronto Face Dataset (TFD)

Visualization of model trajectories



CIFAR-10
(convolutional)

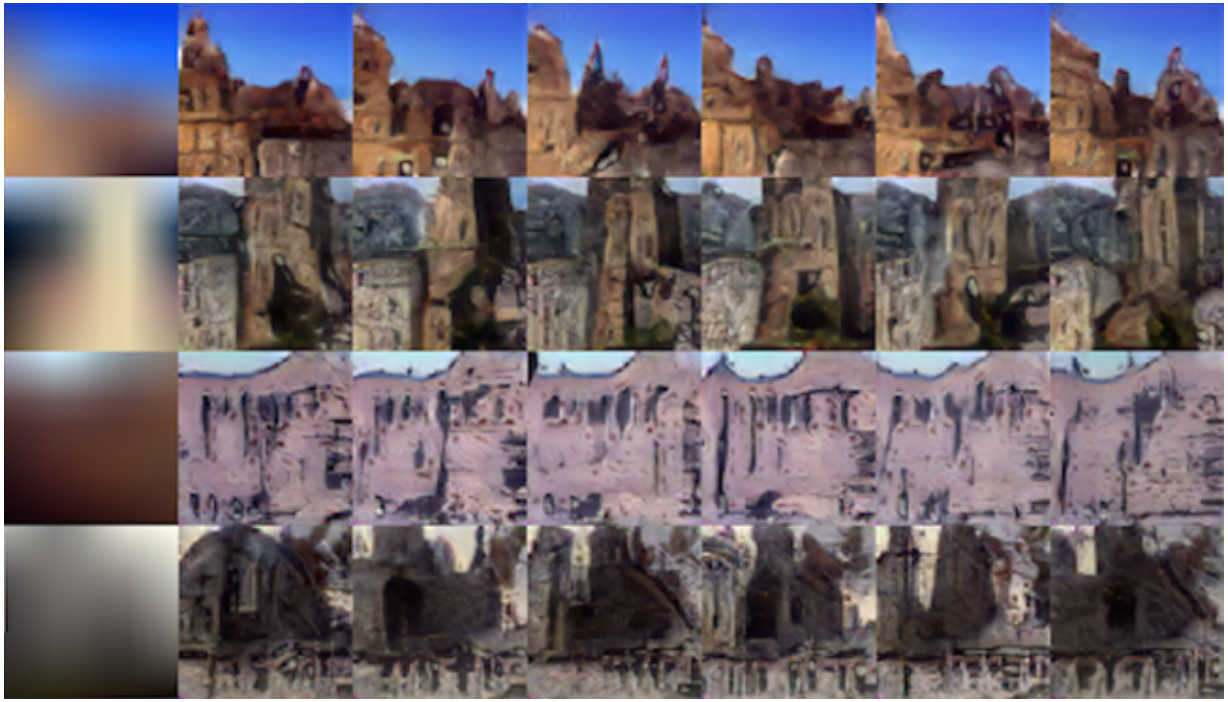
Laplacian Pyramid



(Denton + Chintala, et al 2015)

LAPGAN results

- 40% of samples mistaken *by humans* for real photos

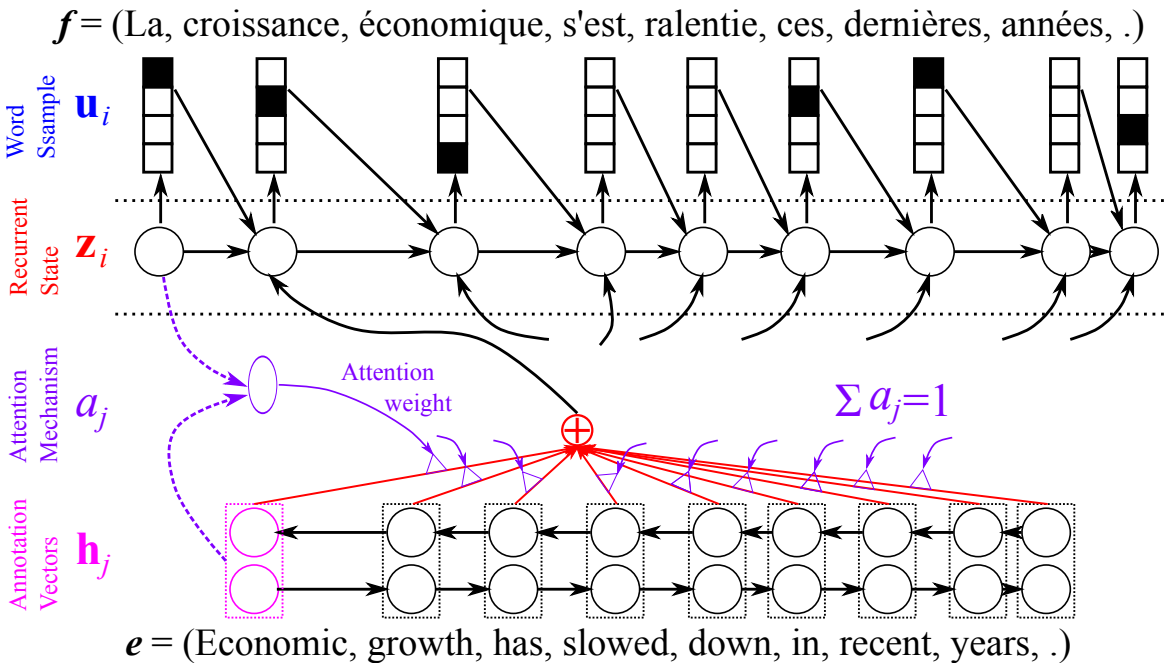


(Denton + Chintala, et al 2015)

Attention-Based Neural Machine Translation

Related to earlier Graves 2013 for generating handwriting

- (Bahdanau, Cho & Bengio, arXiv sept. 2014)
- (Jean, Cho, Memisevic & Bengio, arXiv dec. 2014)

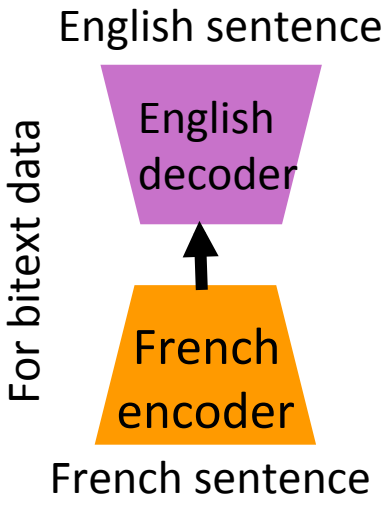


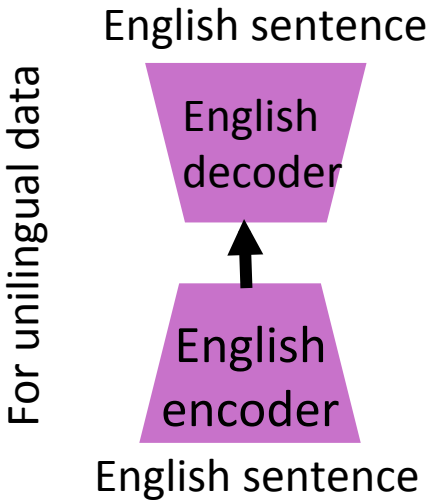
Applying an attention mechanism to

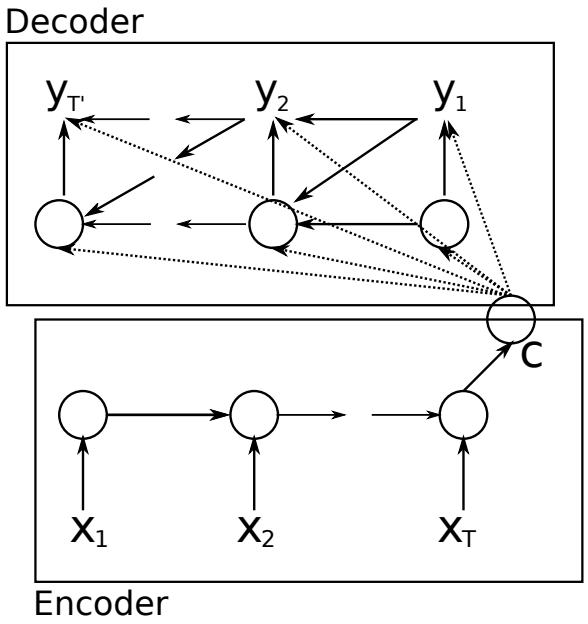
- Translation
- Speech
- Images
- Video
- Memory

Encoder-Decoder Framework

- Intermediate representation of meaning
  = 'universal representation'
- Encoder: from word sequence to sentence representation
- Decoder: from representation to word sequence distribution

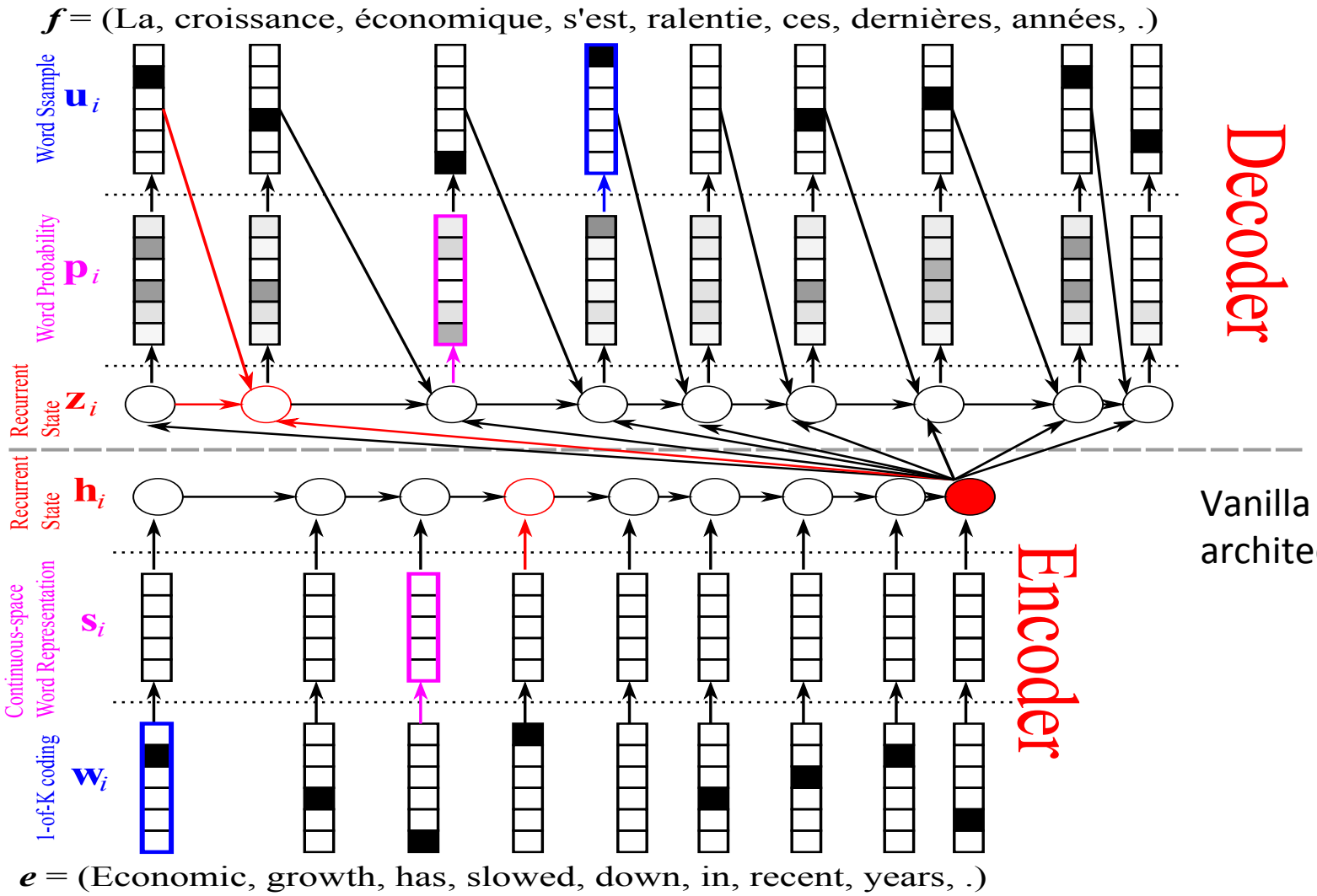






Encoder & Decoder RNN

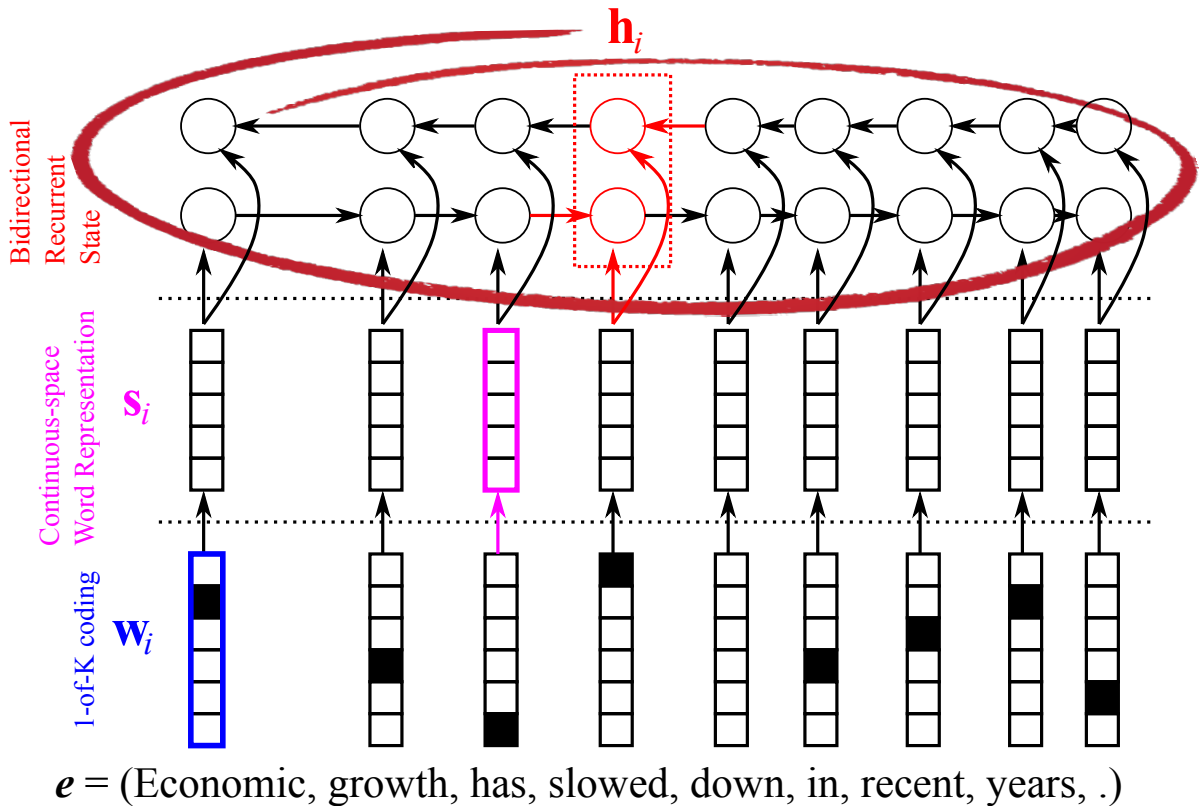
- Need to use gated RNN such as LSTM or GRU



Vanilla architecture

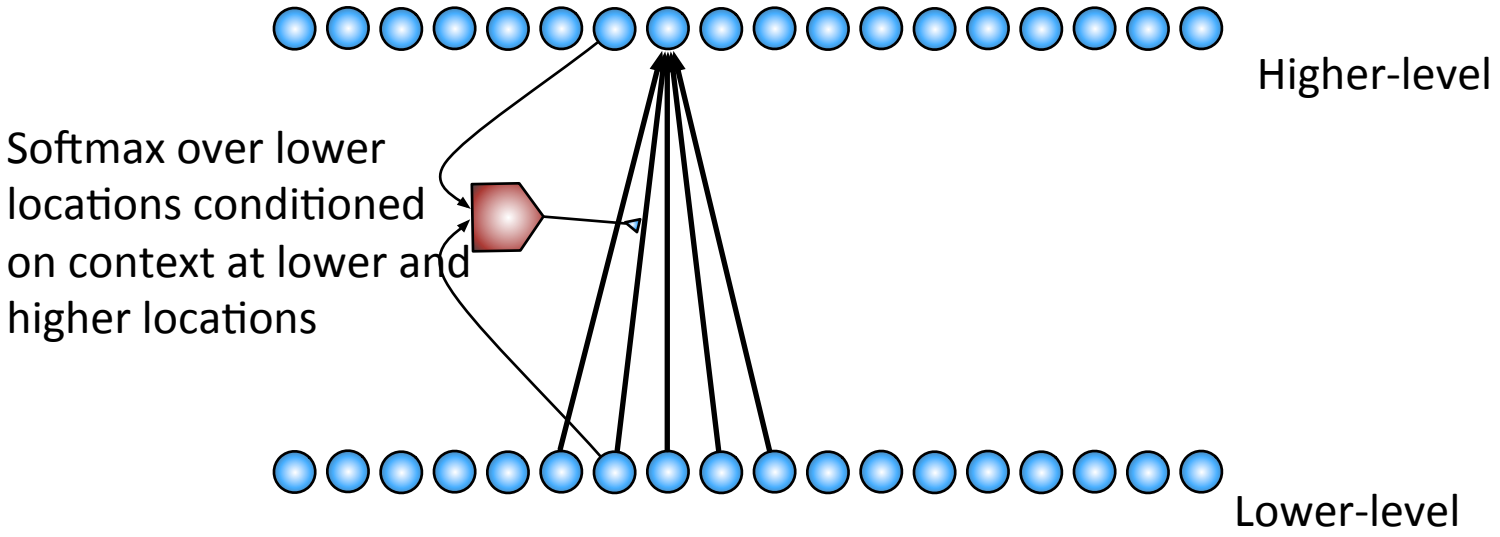
Bidirectional RNN for Input Side

- Following Alex Graves' work on handwriting

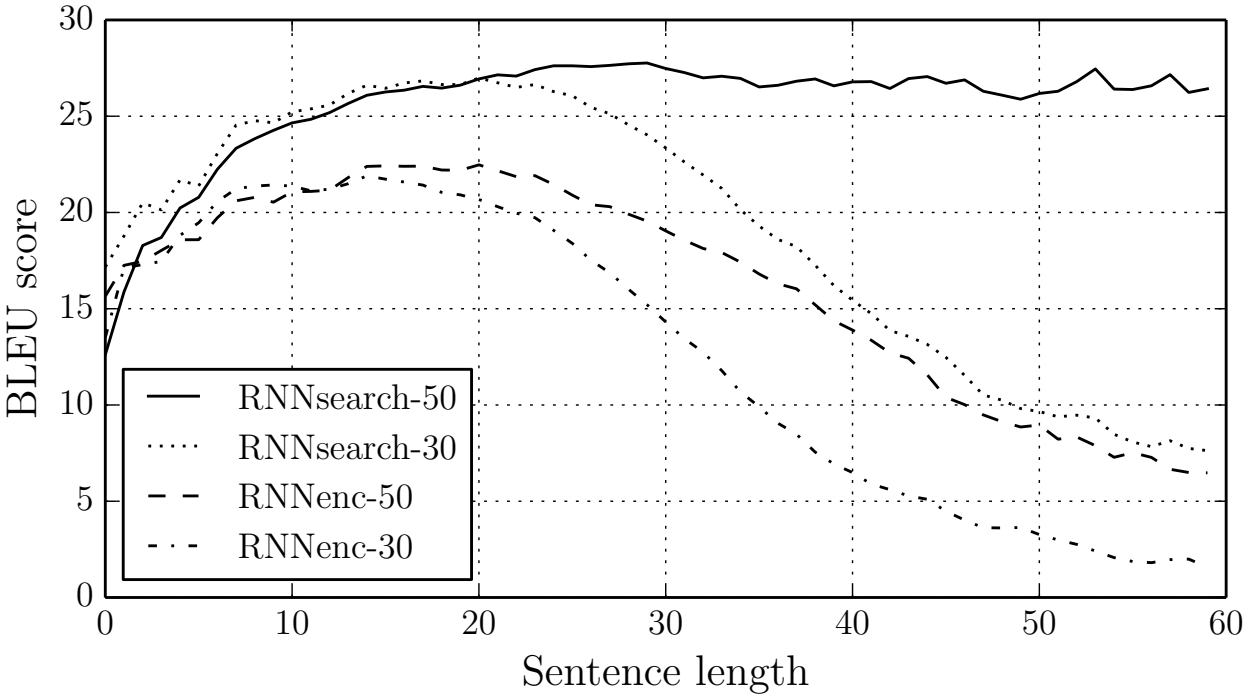


Attention Mechanism for Deep Learning

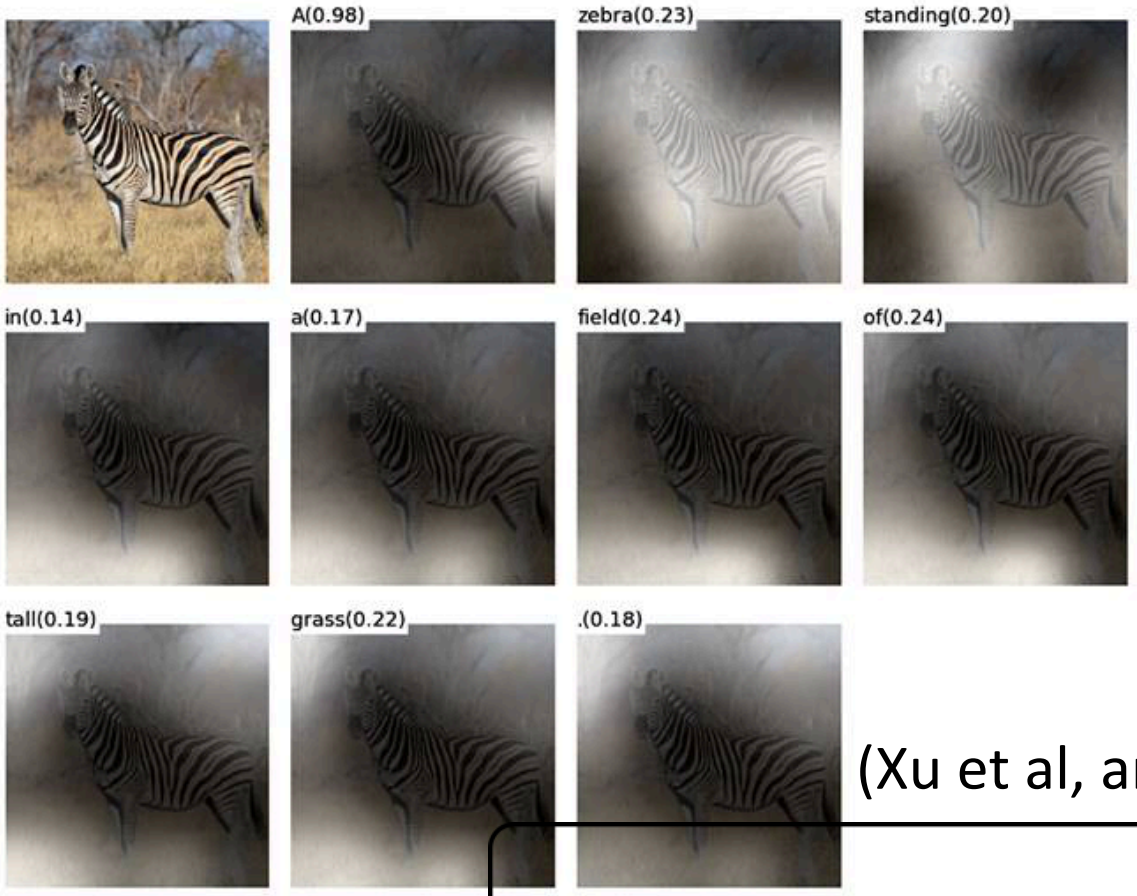
- Consider an input (or intermediate) sequence or image
- Consider an upper level representation, which can choose « where to look », by assigning a weight or probability to each input position, as produced by an MLP, applied at each position



Improvements over Pure AE Model

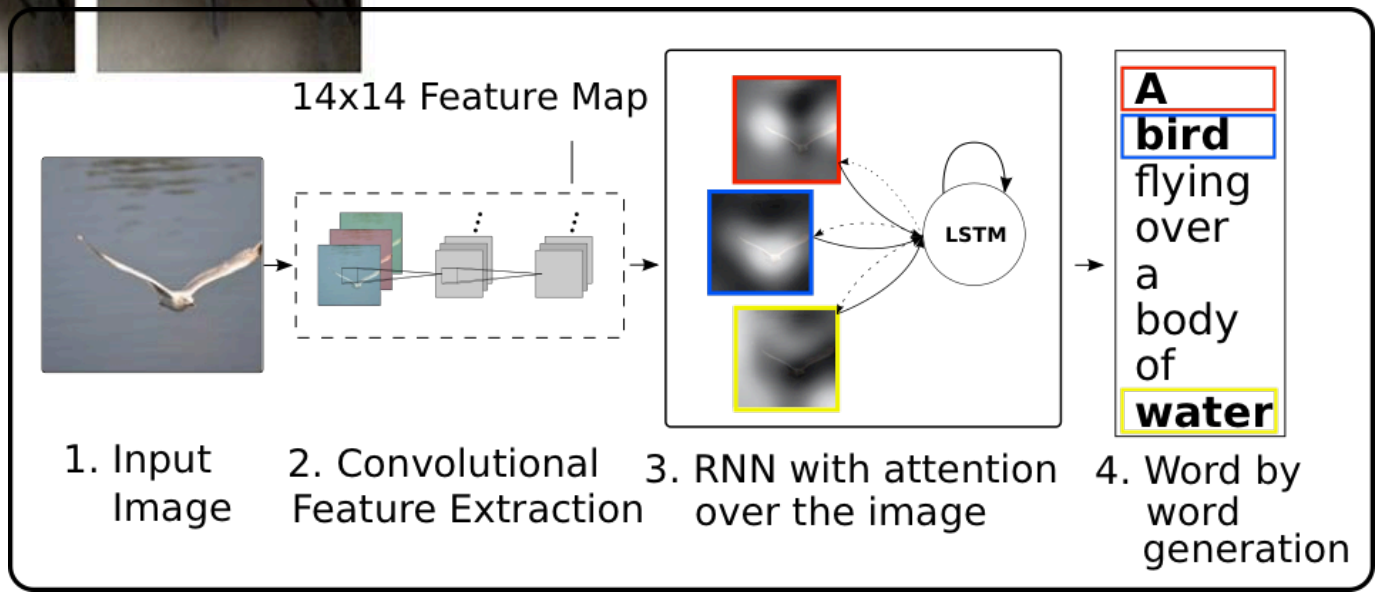


- RNNenc: encode whole sentence
- RNNsearch: predict alignment
- BLEU score on full test set (including UNK)

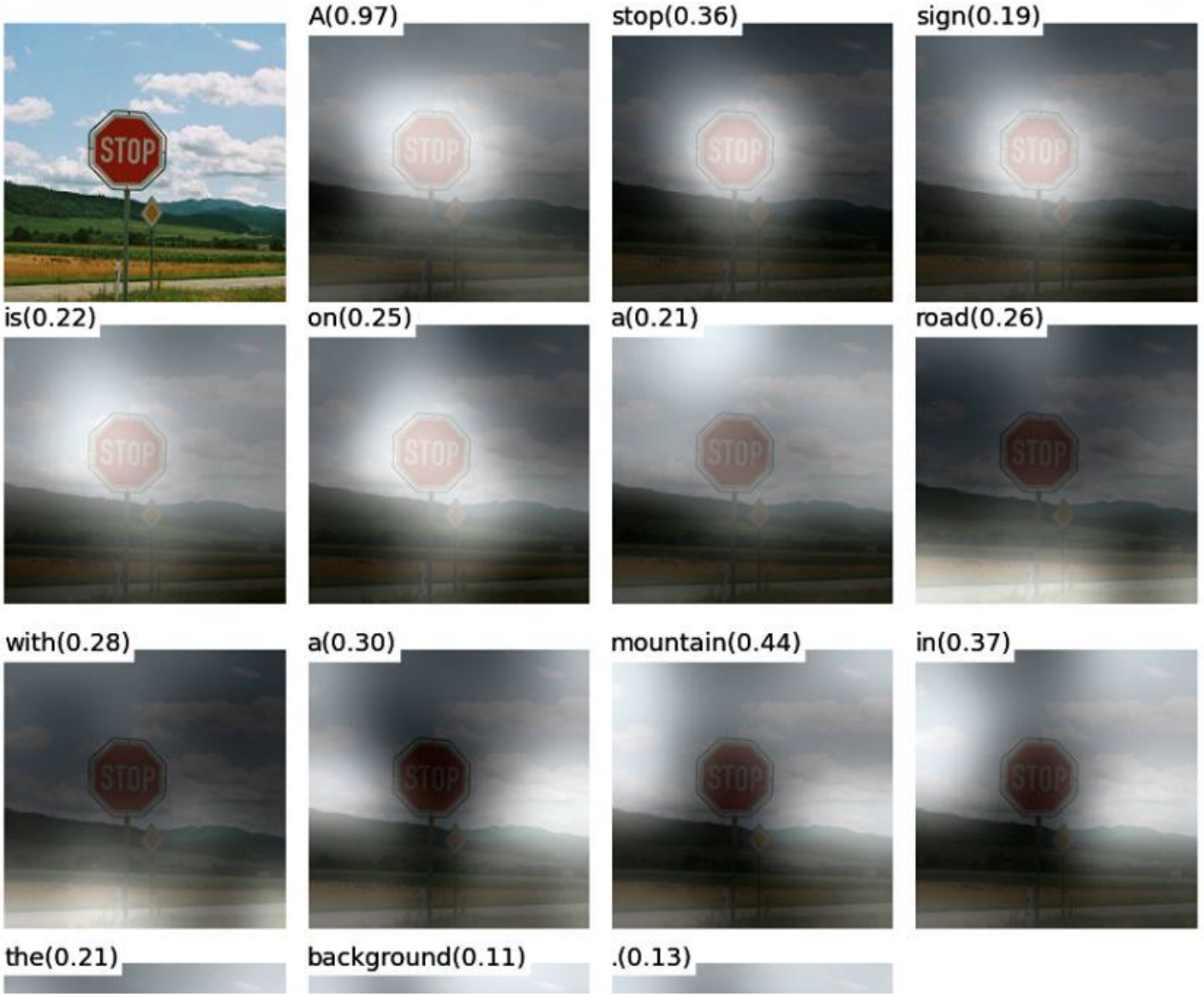


Paying Attention to Selected Parts of the Image While Uttering Words

(Xu et al, arXiv Jan. 2015, ICML 2015)



Speaking about what one sees



Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

Results from (Xu et al, arXiv Jan. 2015, ICML 2015)

Table 1. BLEU-1,2,3,4/METEOR metrics compared to other methods, † indicates a different split, (—) indicates an unknown metric, ∘ indicates the authors kindly provided missing metrics by personal communication, Σ indicates an ensemble, a indicates using AlexNet

		BLEU				
Dataset	Model	B-1	B-2	B-3	B-4	METEOR
Flickr8k	Google NIC(Vinyals et al., 2014)$^{\dagger\Sigma}$	63	41	27	—	—
	Log Bilinear (Kiros et al., 2014a)$^{\circ}$	65.6	42.4	27.7	17.7	17.31
	Soft-Attention	**67**	44.8	29.9	19.5	18.93
	Hard-Attention	**67**	**45.7**	**31.4**	**21.3**	**20.30**
Flickr30k	Google NIC$^{\dagger\circ\Sigma}$	66.3	42.3	27.7	18.3	—
	Log Bilinear	60.0	38	25.4	17.1	16.88
	Soft-Attention	66.7	43.4	28.8	19.1	**18.49**
	Hard-Attention	**66.9**	**43.9**	**29.6**	**19.9**	18.46
COCO	CMU/MS Research (Chen & Zitnick, 2014)^{a}	—	—	—	—	20.41
	MS Research (Fang et al., 2014)$^{\dagger a}$	—	—	—	—	20.71
	BRNN (Karpathy & Li, 2014)$^{\circ}$	64.2	45.1	30.4	20.3	—
	Google NIC$^{\dagger\circ\Sigma}$	66.6	46.1	32.9	24.6	—
	Log Bilinear$^{\circ}$	70.8	48.9	34.4	24.3	20.03
	Soft-Attention	70.7	49.2	34.4	24.3	**23.90**
	Hard-Attention	**71.8**	**50.4**	**35.7**	**25.0**	23.04

The Good

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

And the Bad

A large white bird standing in a forest.

A woman holding a clock in her hand.

A man wearing a hat and
a hat on a skateboard.

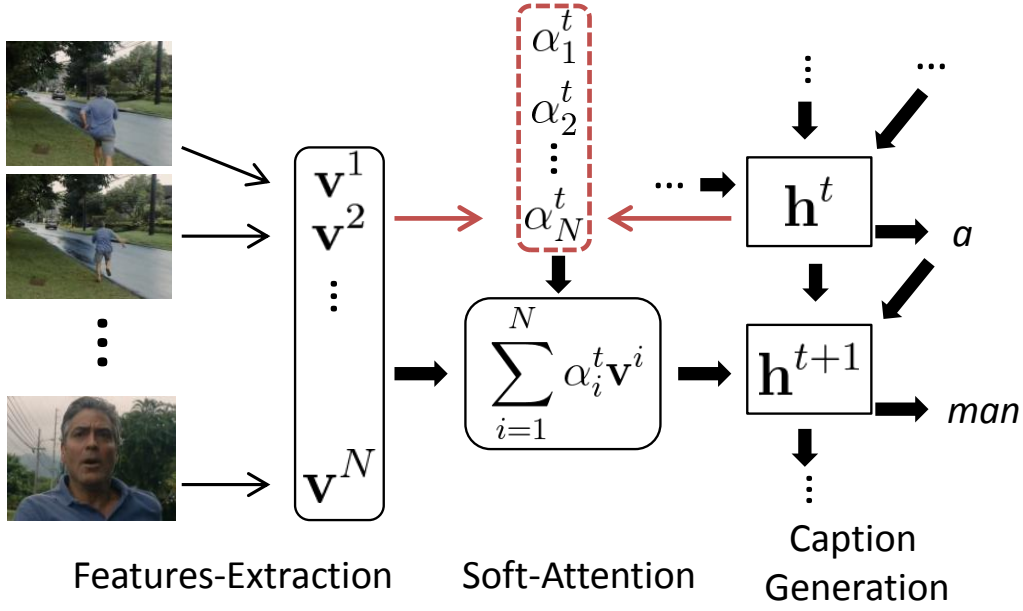
A person is standing on a beach
with a surfboard.

A woman is sitting at a table
with a large pizza.

A man is talking on his cell phone
while another man watches.

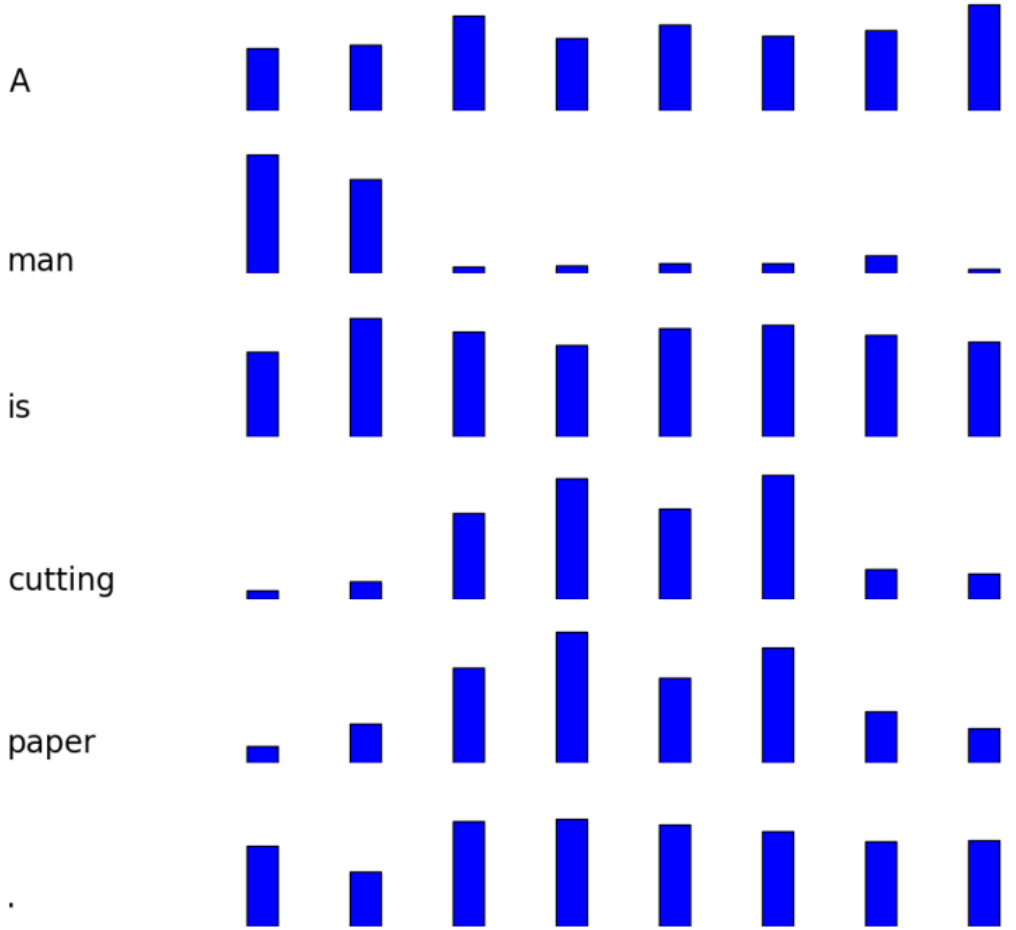
Attention through time for video caption generation

- (Yao et al arXiv 1502.08029, 2015) *Video Description Generation Incorporating Spatio-Temporal Features and a Soft-Attention Mechanism*
- Attention can be focused temporally, i.e., selecting input frames



Attention through time for video caption generation (Yao et al 2015)

- Attention is focused at appropriate frames depending on which word is generated.



Attention through time for video caption generation (Yao et al 2015)

- Soft-attention worked best in this setting

Model	Feature	Bleu					Meteor	Perplexity
		1	2	3	4	mb		
non-attention	GNet	32.0	9.2	3.4	1.2	0.3	4.43	88.28
	$\text{GNet+3DConv}_{\text{non-att}}$	33.6	10.4	4.3	1.8	0.7	5.73	84.41
soft-attention	GNet	31.0	7.7	3.0	1.2	0.3	4.05	66.63
	$\text{GNet+3DConv}_{\text{att}}$	28.2	8.2	3.1	1.3	0.7	5.6	**65.44**

Corpus:
She rushes out.
Test_sample:
The woman turns away.

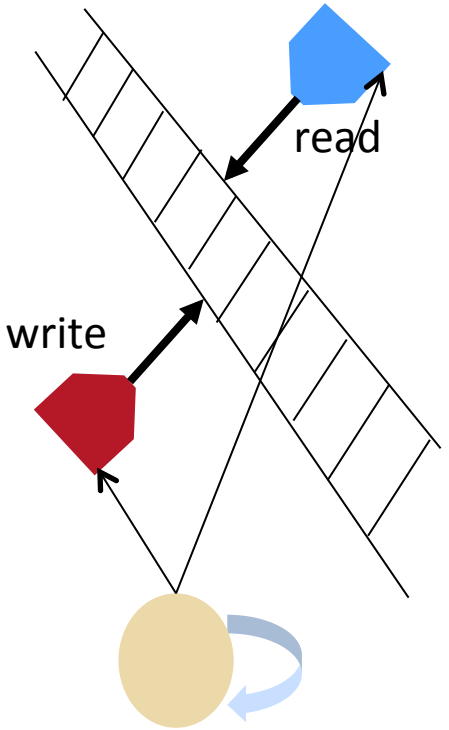
Corpus:
SOMEONE sits with his arm around SOMEONE.
He nuzzles her cheek, then kisses tenderly.
Test_sample:
SOMEONE sits beside SOMEONE.

Corpus:
SOMEONE shuts the door.
Test_sample:
as he turns on his way to the door , SOMEONE turns away.

Generated captions

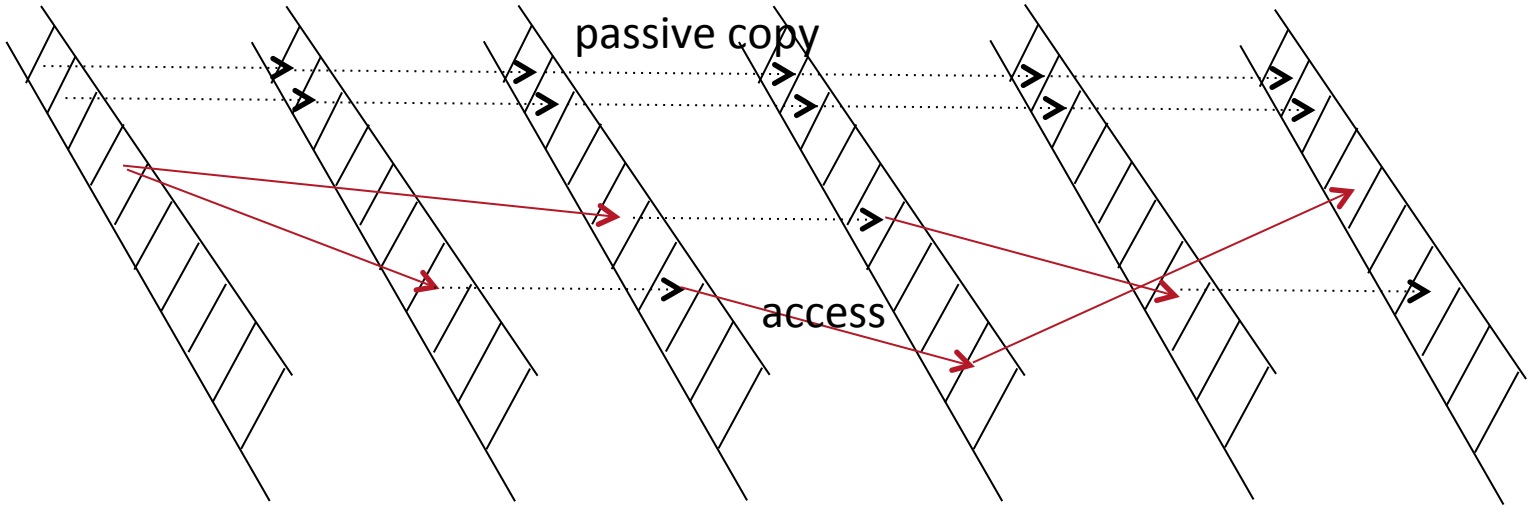
Attention Mechanisms for Memory Access

- Neural Turing Machines (Graves et al 2014)
- and Memory Networks (Weston et al 2014)
- Use a form of attention mechanism to control the read and write access into a memory
- The attention mechanism outputs a softmax over memory locations
- For efficiency, the softmax should be sparse (mostly 0's), e.g. maybe using a hash-table formulation.



Sparse Access Memory for Long-Term Dependencies

- Whereas LSTM memories always decay exponentially (even if slowly), a mental state stored in an external memory can stay for arbitrarily long durations, until evoked for read or write.
- Need to replace the soft gater or softmax attention by hard one that is 0 most of the time, and yet for which training works (again, may use noisy decisions and/or REINFORCE).
- Different « threads » can run in parallel if we view the memory as an associative one.

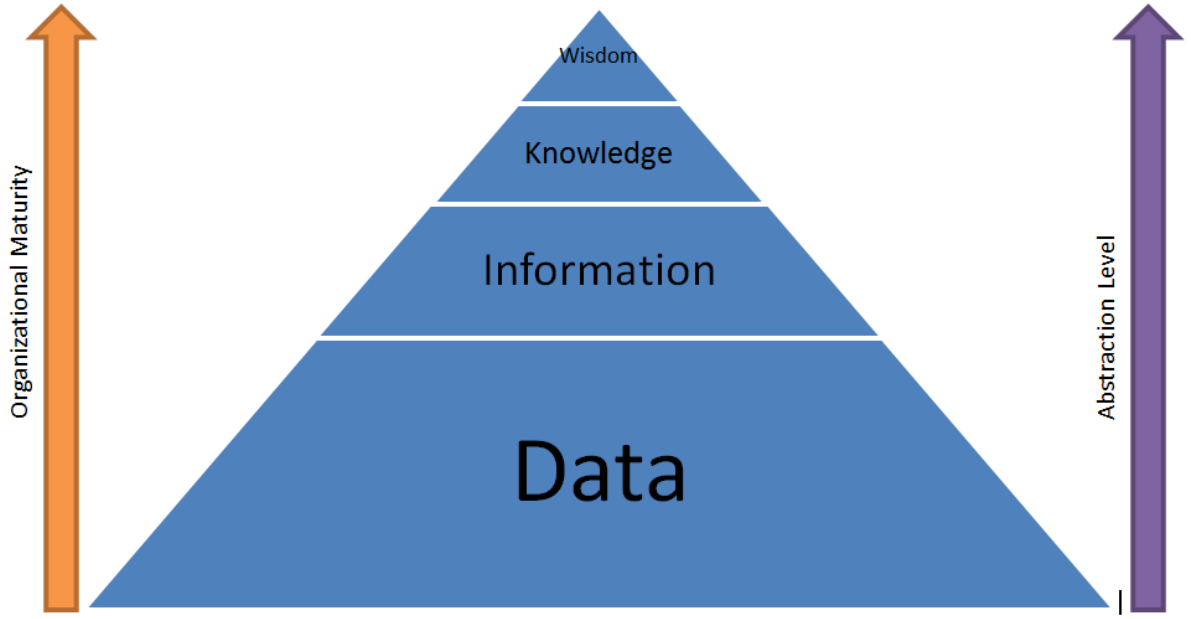


Deep Learning Challenges (Bengio, arxiv 1305.0445 Deep learning of representations: looking forward)

- Computational Scaling
- Optimization & Underfitting
- Intractable Marginalization, Approximate Inference & Sampling
- Disentangling Factors of Variation
- Reasoning & One-Shot Learning of Facts

Learning Multiple Levels of Abstraction

- The big payoff of deep learning is to allow learning higher levels of abstraction
- Higher-level abstractions disentangle the factors of variation, which allows much easier generalization and transfer



Conclusions

- **Distributed representations**:
  - prior that can buy exponential gain in generalization
- **Deep composition of non-linearities**:
  - prior that can buy exponential gain in generalization
- Both yield **non-local generalization**
- Strong evidence that **local minima are not an issue, saddle points**
- **Auto-encoders capture the data generating distribution**
  - Gradient of the energy
  - Markov chain generating an estimator of the dgd
  - Can be generalized to deep generative models

MILA: Montreal Institute for Learning Algorithms