

Deep Learning in Image Processing and Vision

Tutorial

CIFAR
CANADIAN
INSTITUTE
FOR
ADVANCED
RESEARCH

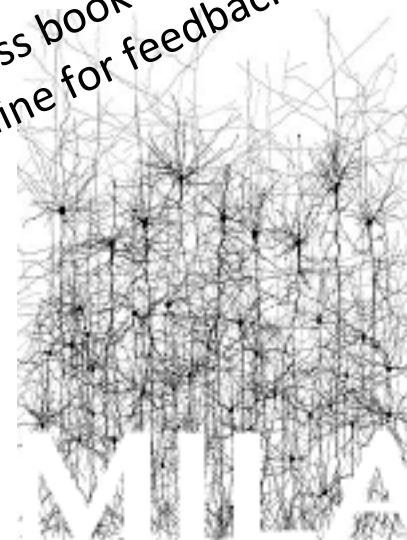
Yoshua Bengio & Roland Memisevic

September 27, 2015

ICIP'2015, Quebec City

Université 
de Montréal

PLUG: **Deep Learning**, MIT Press book in preparation, draft chapters online for feedback



Ultimate Goal

- **Understand the principles giving rise to intelligence**

Focus

- **Learning:** mathematical and computational principles allowing one to learn from examples in order to acquire knowledge

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

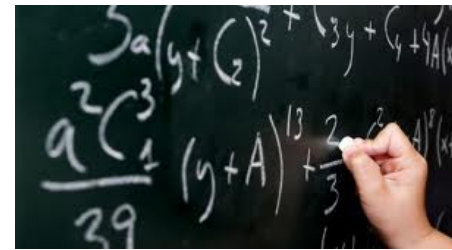
Potential Outcome: AI

- **Computers that can**
 - **see and hear**
 - **understand natural language**
 - **understand human behavior**
- **Better understanding of human & animal intelligence**
- **Personal assistants, self-driving cars...**

What is Machine Learning?

Mathematical principles and computer algorithms exploiting data

- for extracting what is **GENERAL**



- so as to be able to say something meaningful **about new cases**
- to identify which **configurations** of variables are **plausible**
- to **generate** new plausible configurations or **choose best ones**
- to learn to **predict, classify, take decisions**

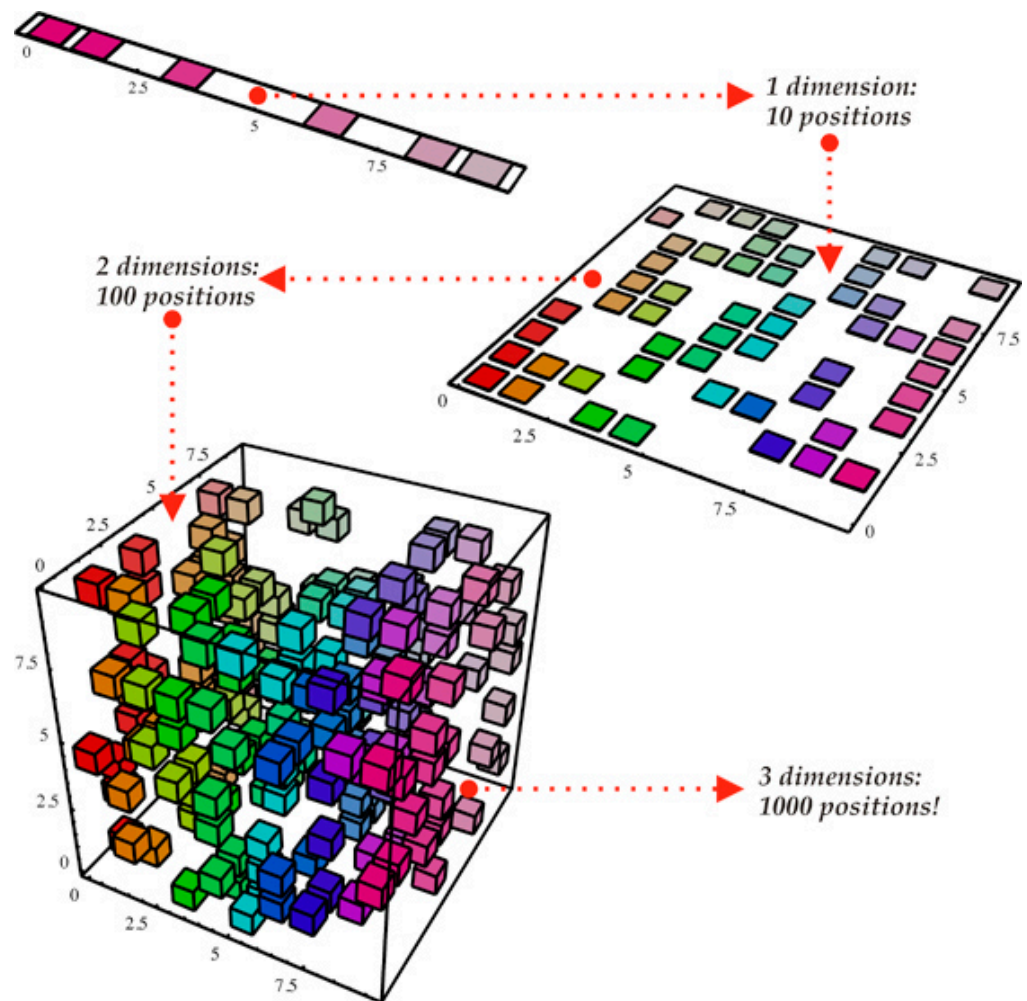
Intermediate Goals

- **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

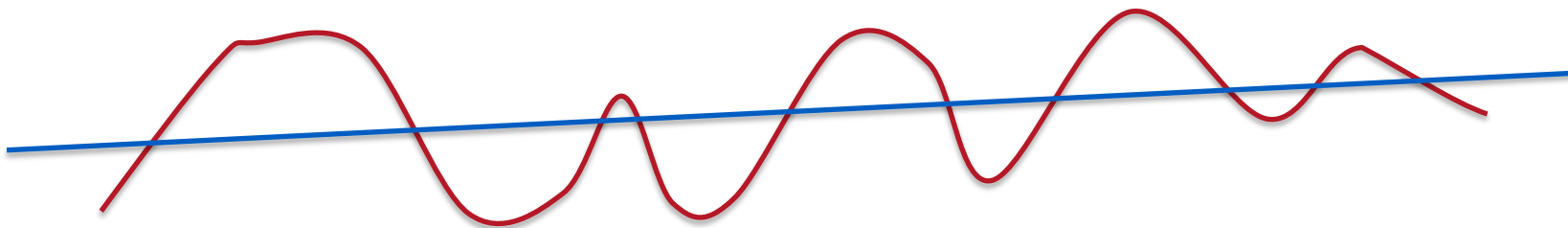


Not Dimensionality so much as Number of Variations



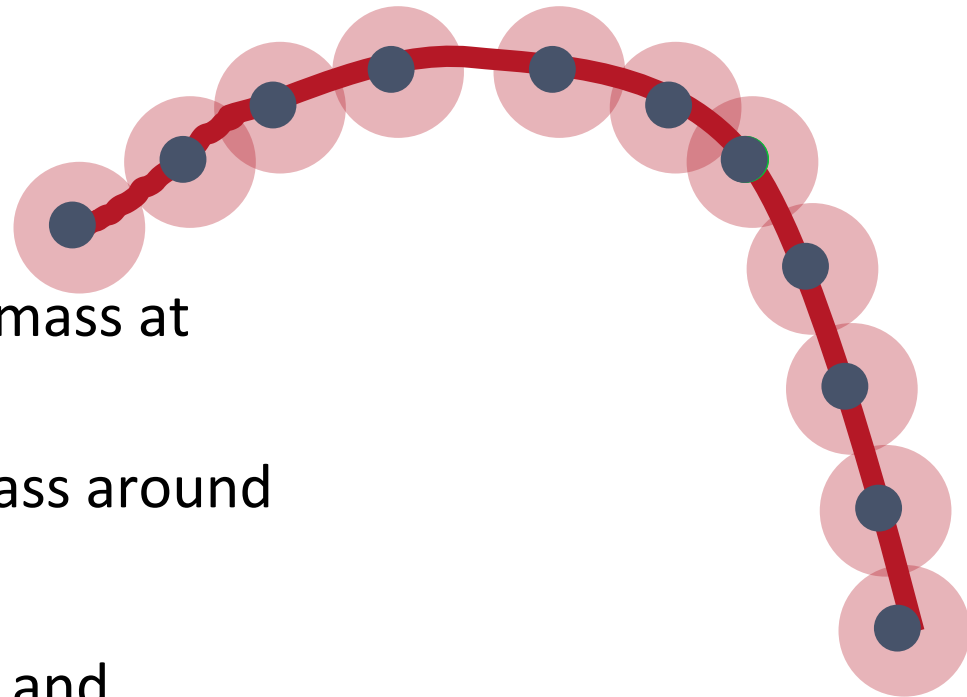
(Bengio, Dellalleau & Le Roux 2007)

- **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

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**

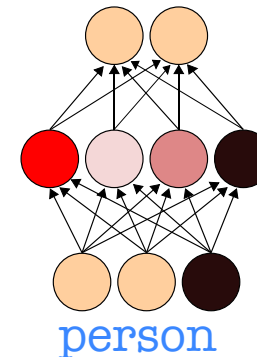
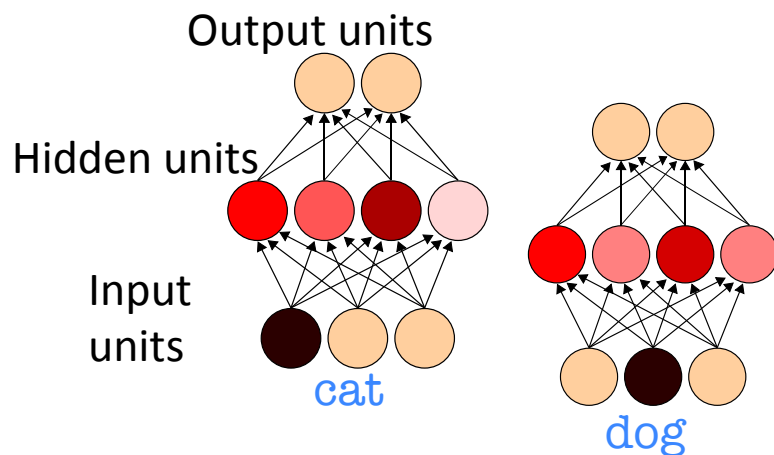
Deep architecture: **multiple levels of feature learning**

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

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)



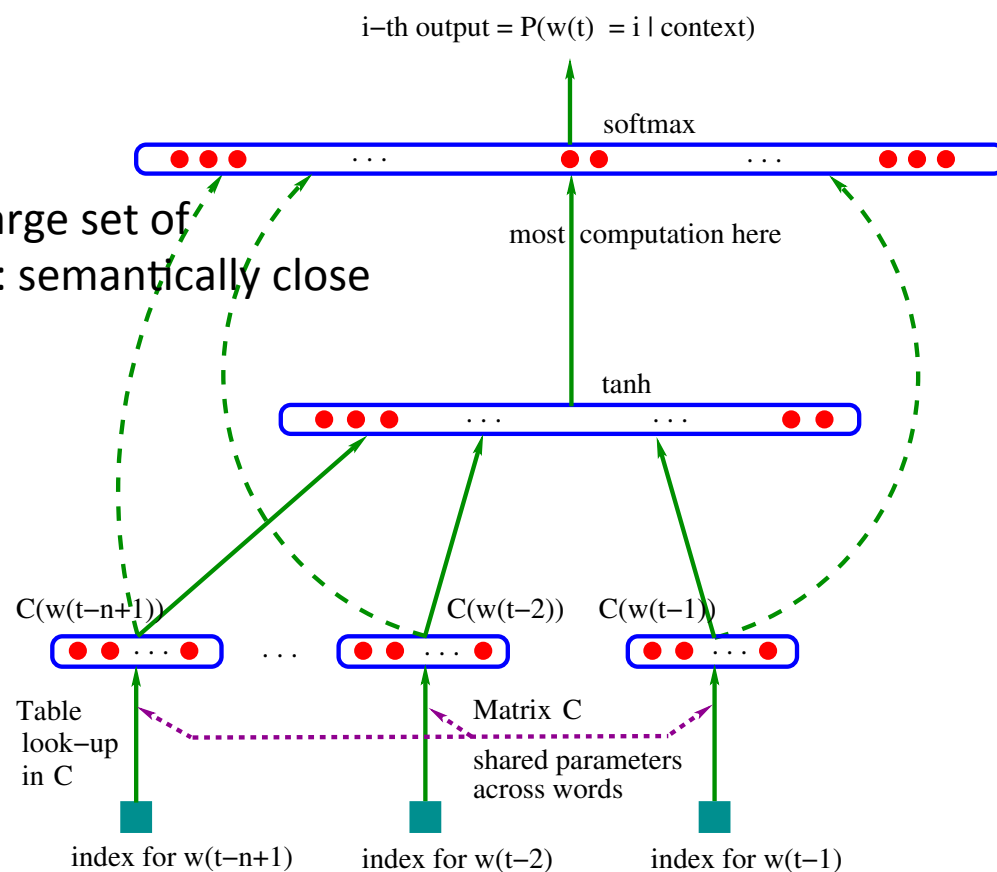
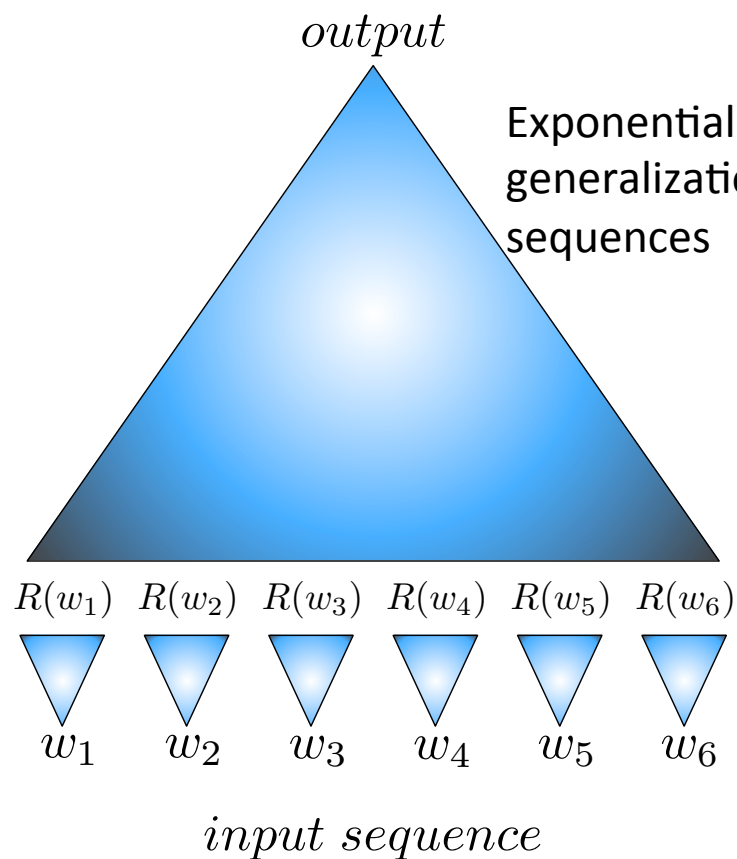
Geoffrey Hinton



David Rumelhart

Neural Language Models: fighting one exponential by another one!

- (Bengio et al NIPS'2000)



Exponentially large set of possible contexts

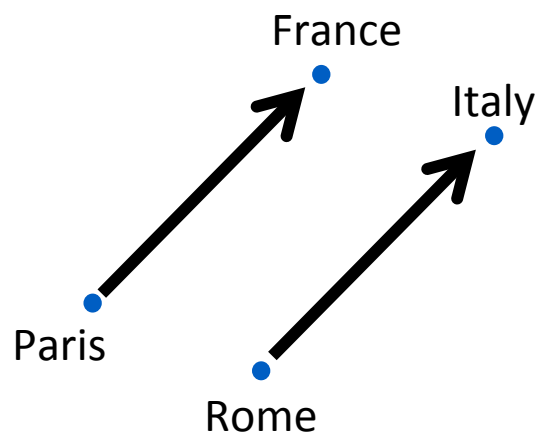
Neural word embeddings: visualization

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 $\approx$ Man – Woman
- Paris – France + Italy $\approx$ Rome

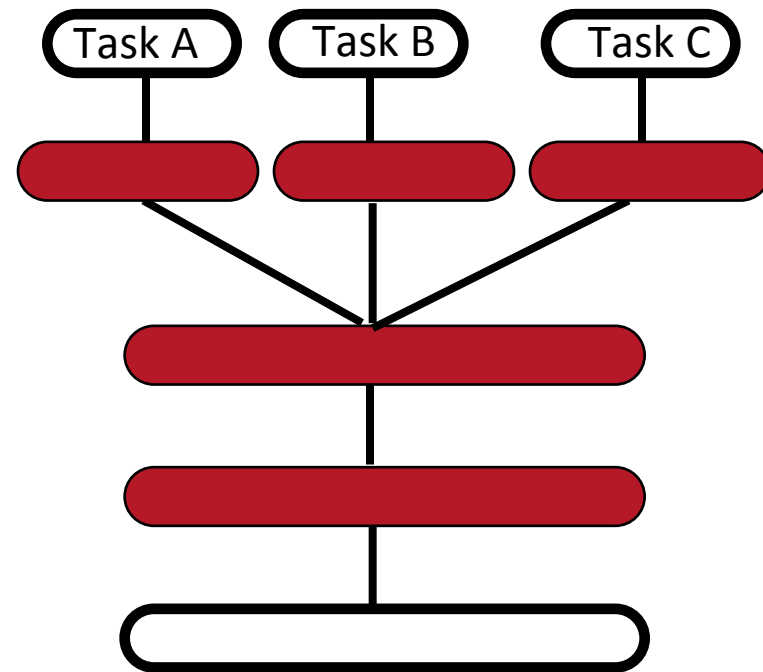


How do humans generalize from very few examples?

- 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**

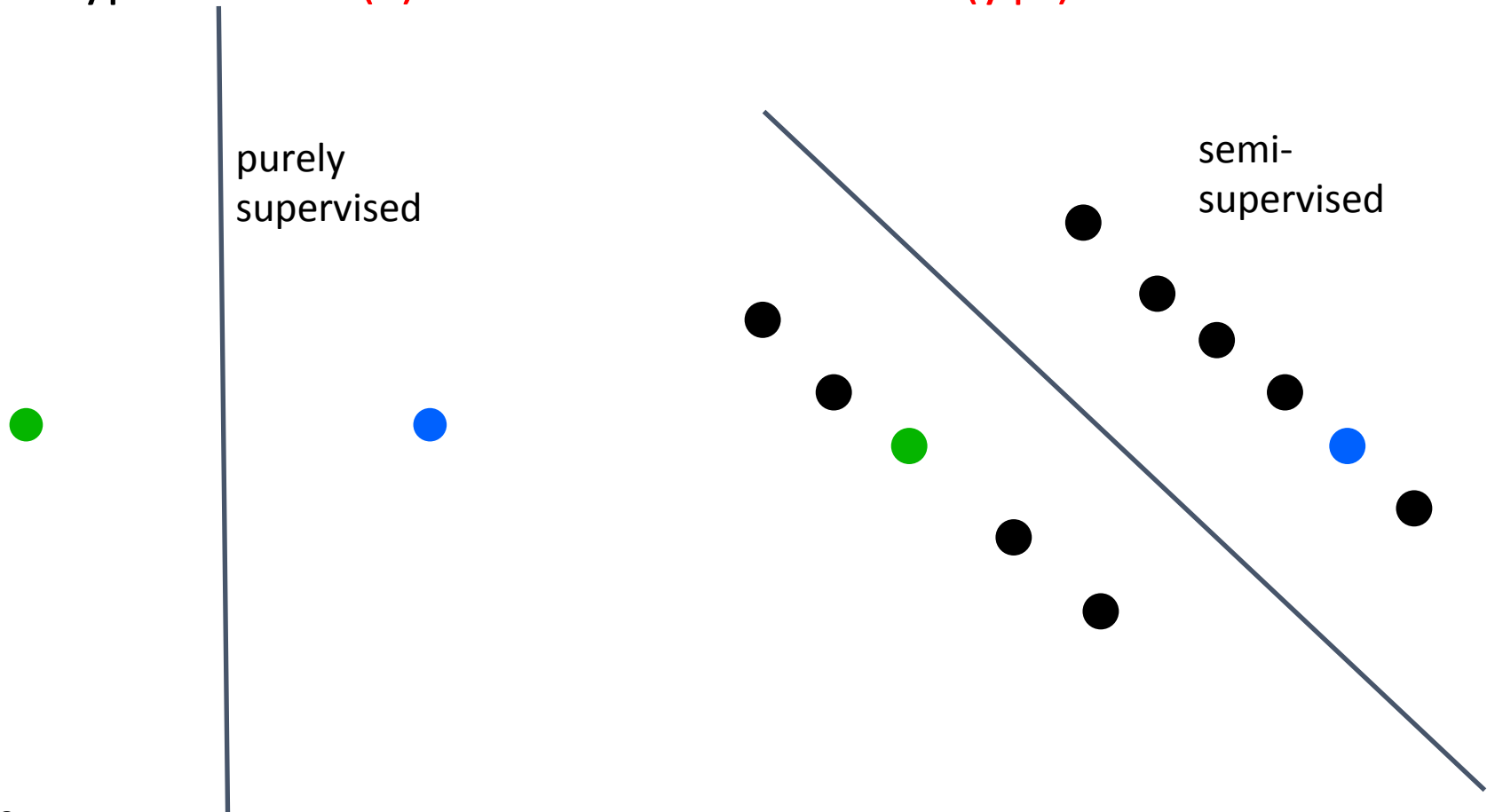


E.g. dictionary, with intermediate concepts re-used across many definitions

Prior: shared underlying explanatory factors between tasks

Sharing Statistical Strength by Semi-Supervised Learning

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



Algorithms

Machine Learning 101

- Family of functions f_θ
- Tunable parameters θ
- Examples $Z \sim$ 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:

$$\text{Loss} = -\log P(Y|X) = \log \sigma + ||f_\theta(X) - Y||^2 / \sigma^2$$

- E.g. Multinoulli Y for classification,

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

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

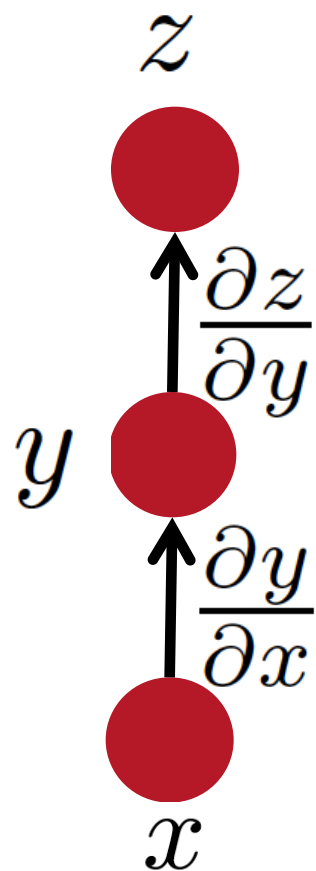
Multiple Output Variables

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

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

- 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. $P(Y_1, \dots, Y_k|X)$
 - Similar to unsupervised learning problem of capturing joint
 - Exact likelihood may similarly be intractable, depending on model

Simple Chain Rule



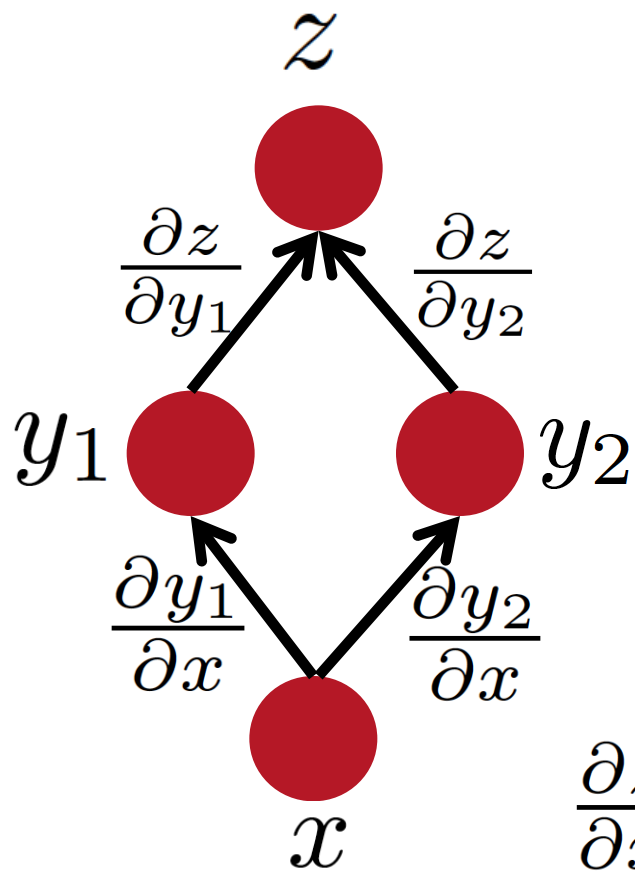
$$\Delta z = \frac{\partial z}{\partial y} \Delta y$$

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

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

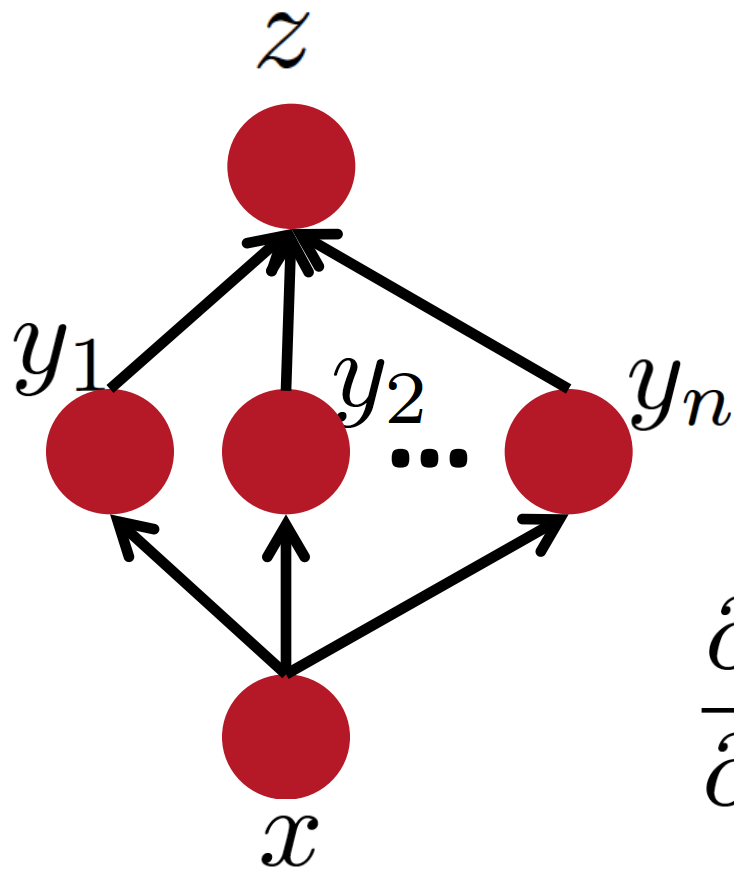
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \frac{\partial y}{\partial x}$$

Multiple Paths Chain Rule



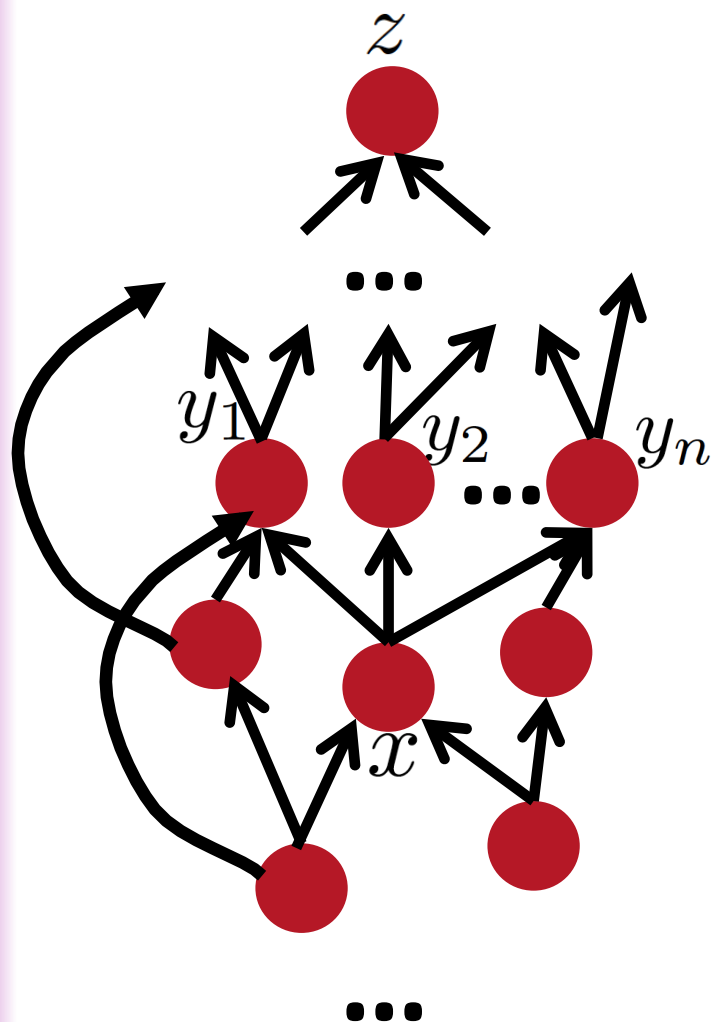
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y_1} \frac{\partial y_1}{\partial x} + \frac{\partial z}{\partial y_2} \frac{\partial y_2}{\partial x}$$

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

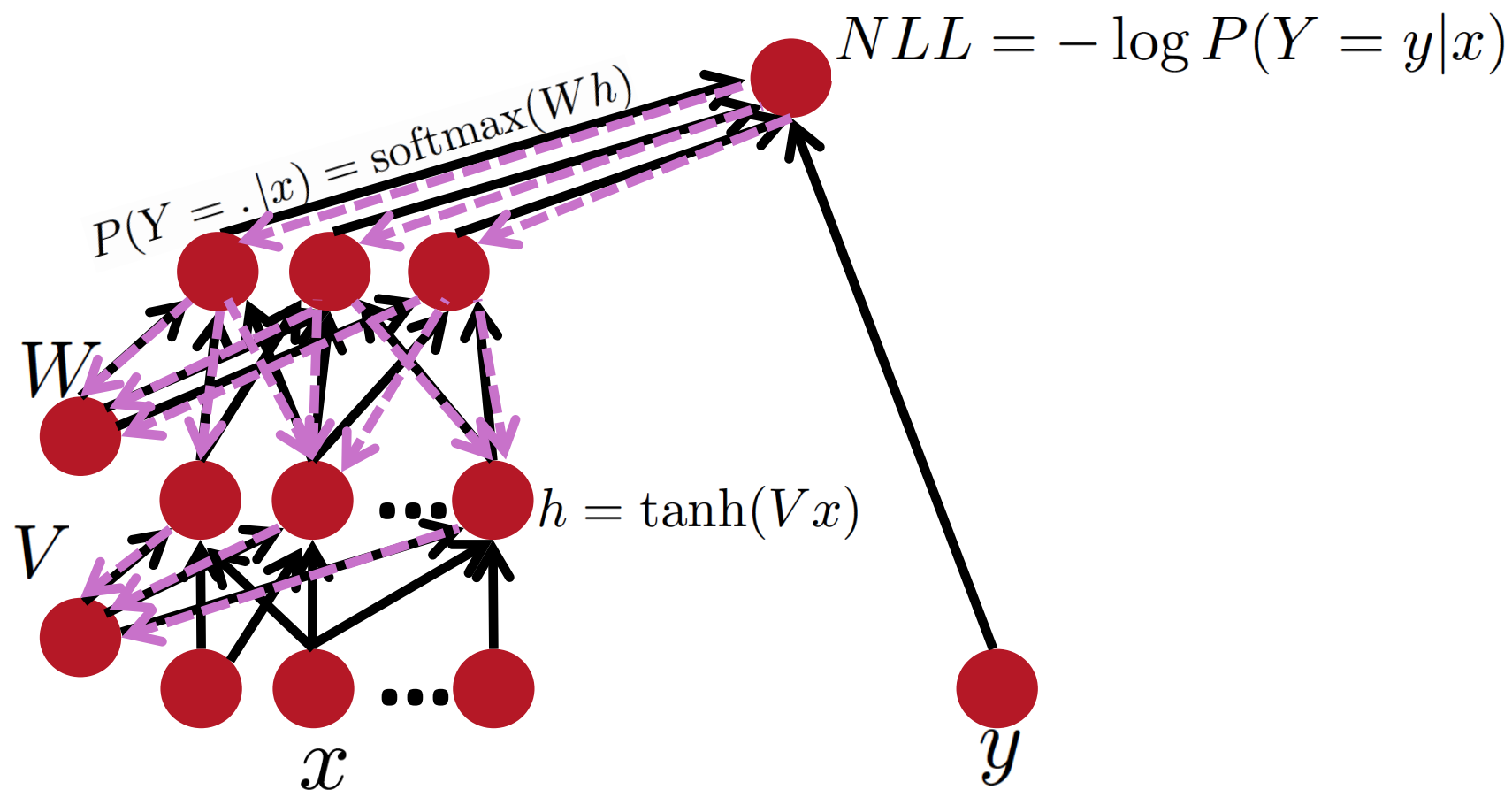


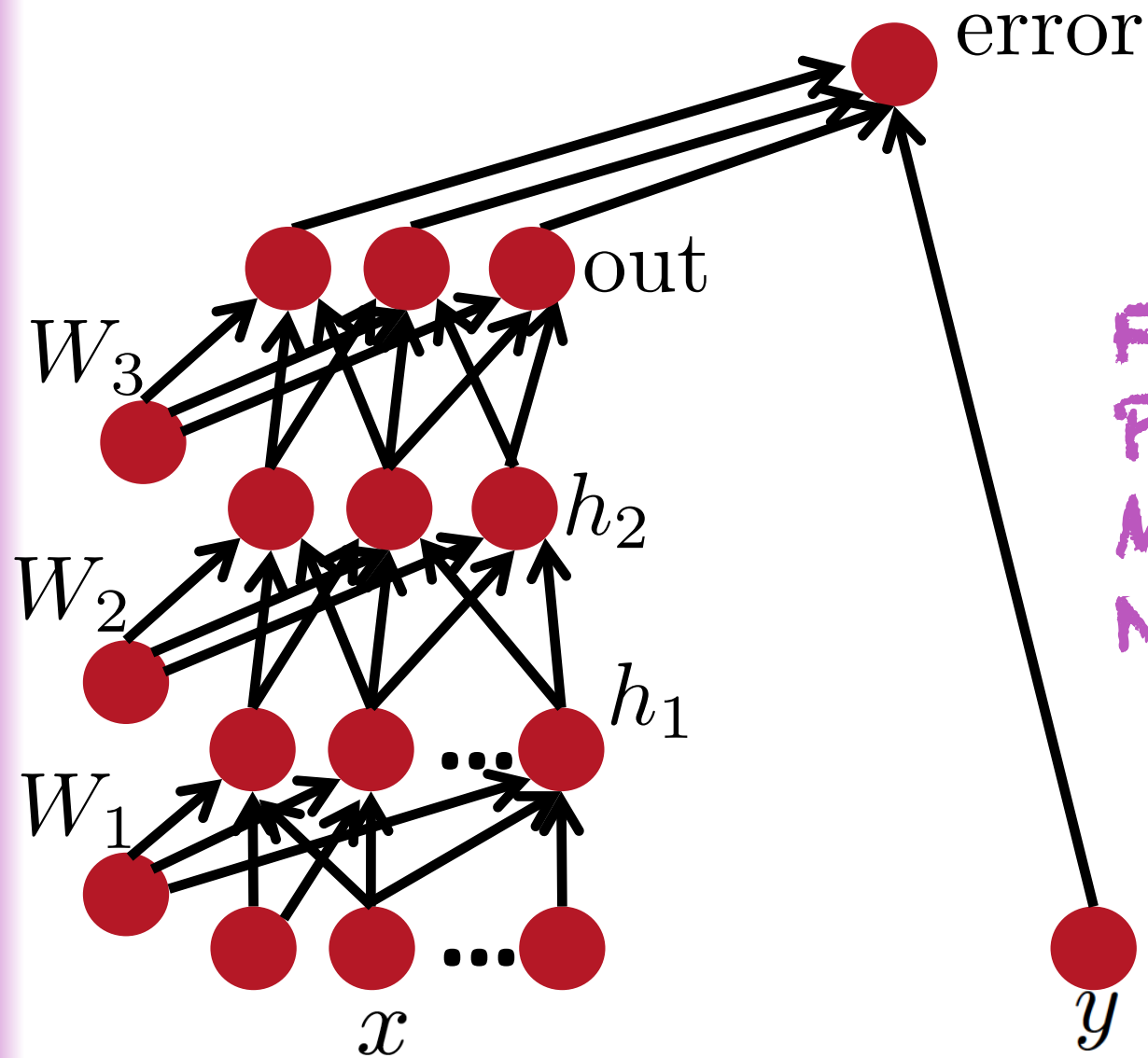
Flow graph: any directed acyclic graph
 node = computation result
 arc = computation dependency

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

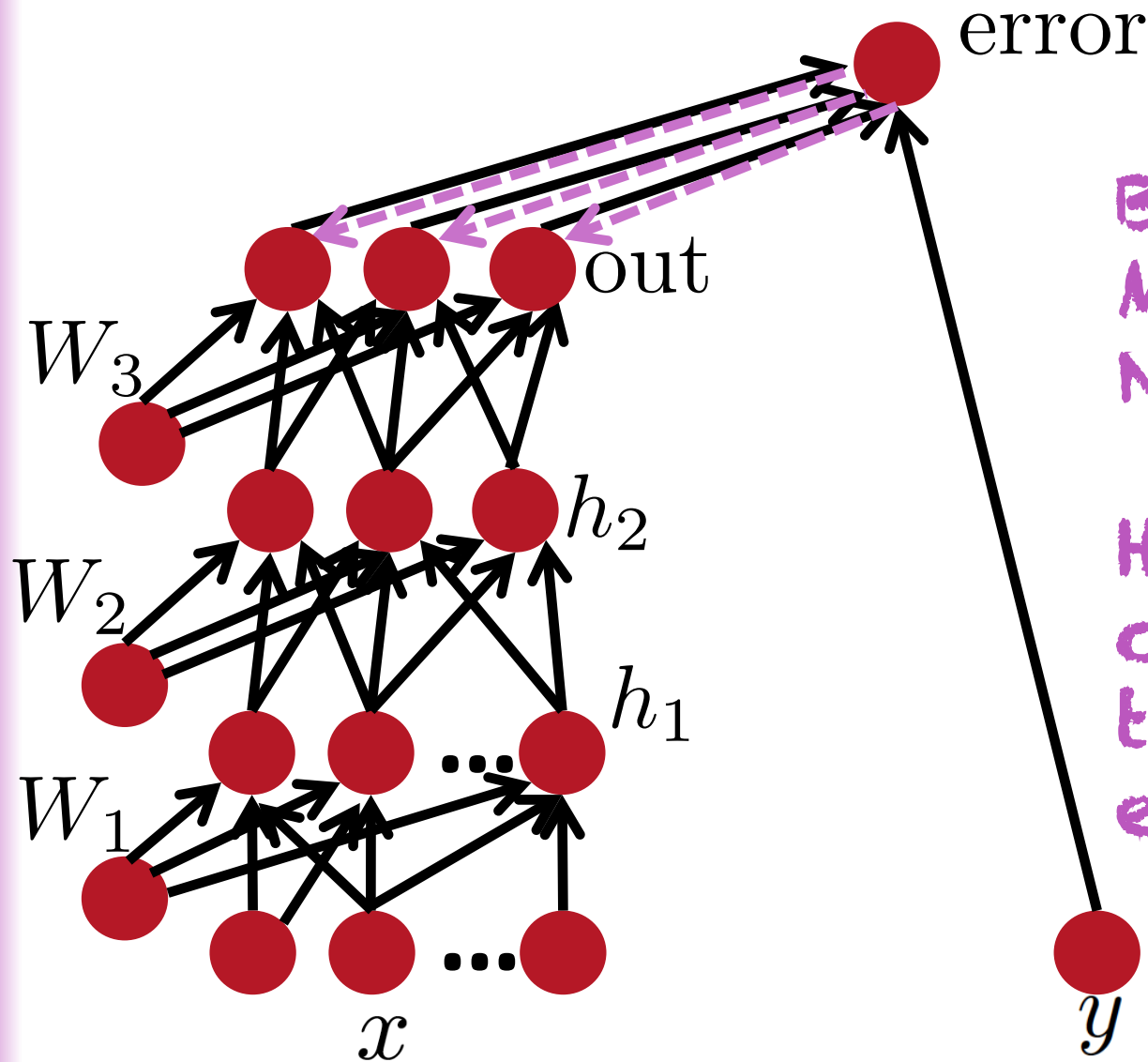
$$\frac{\partial z}{\partial x} = \sum_{i=1}^n \frac{\partial z}{\partial y_i} \frac{\partial y_i}{\partial x}$$

Back-Prop in Multi-Layer Net



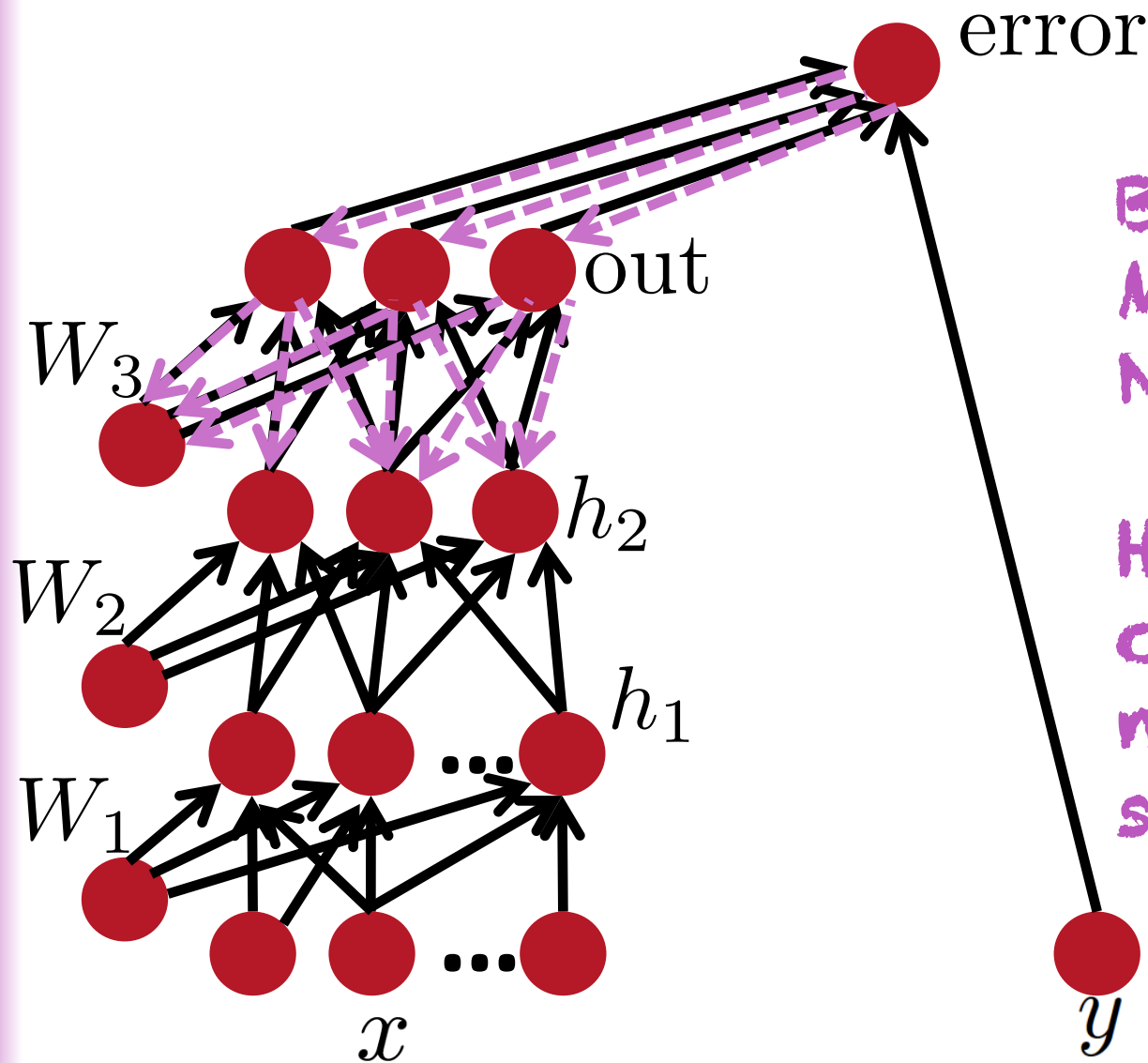


Forward-
Prop in
Multi-Layer
Net



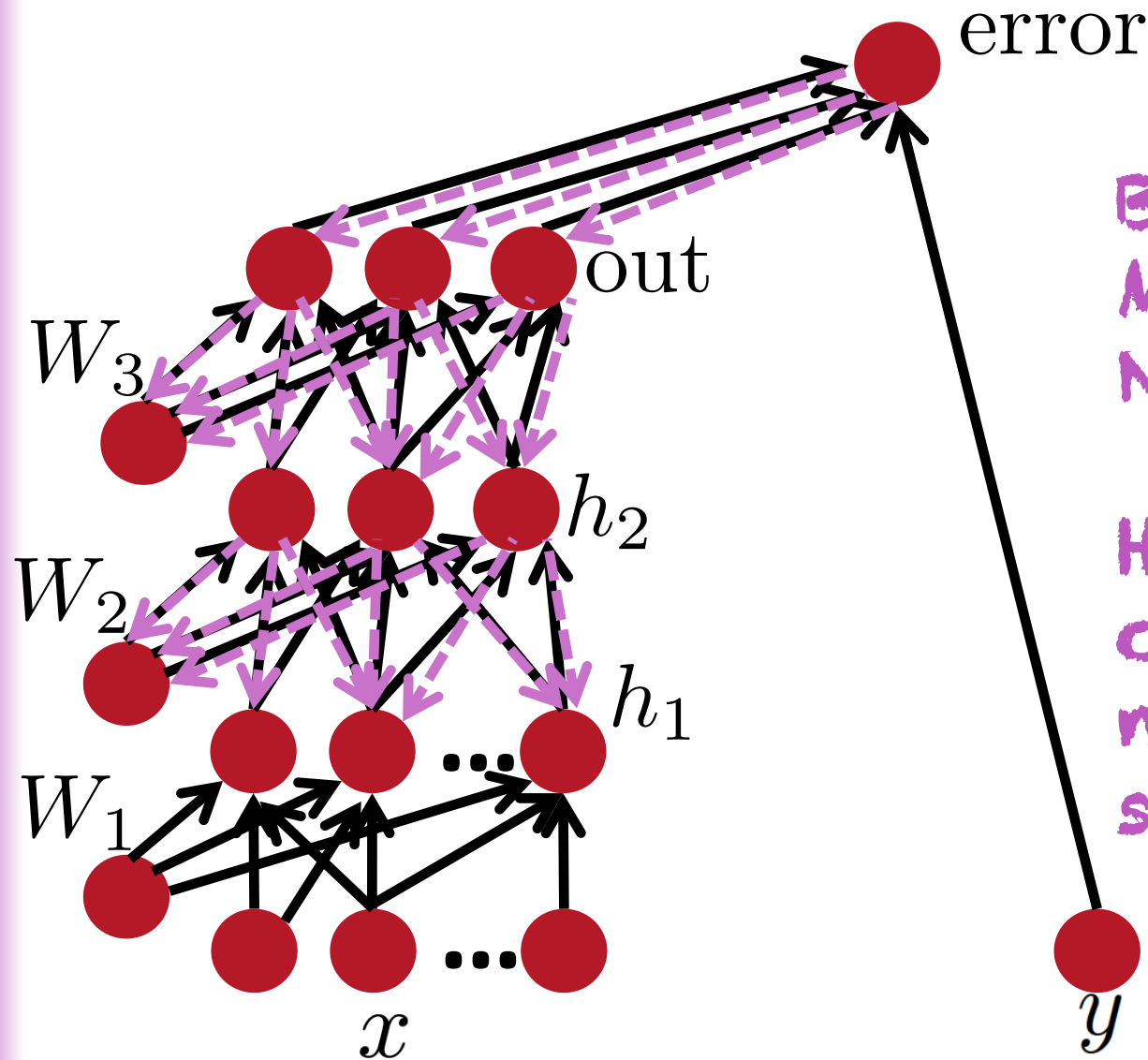
Backprop in
Multi-Layer
Net:

How outputs
could change
to make
error smaller



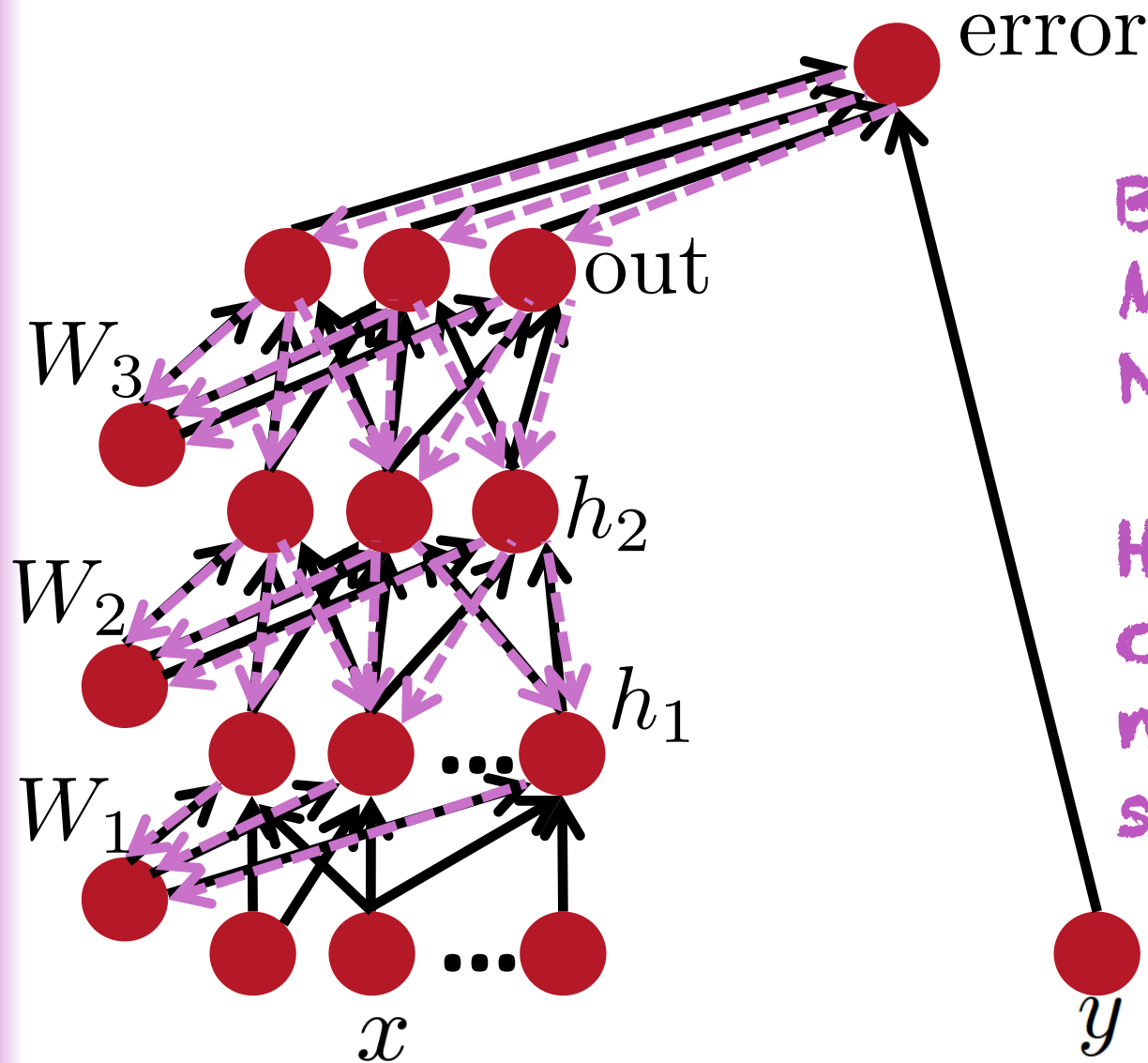
Backprop in
Multi-Layer
Net:

How h_2 could
change to
make error
smaller



Backprop in
Multi-Layer
Net:

How h_1 could
change to
make error
smaller

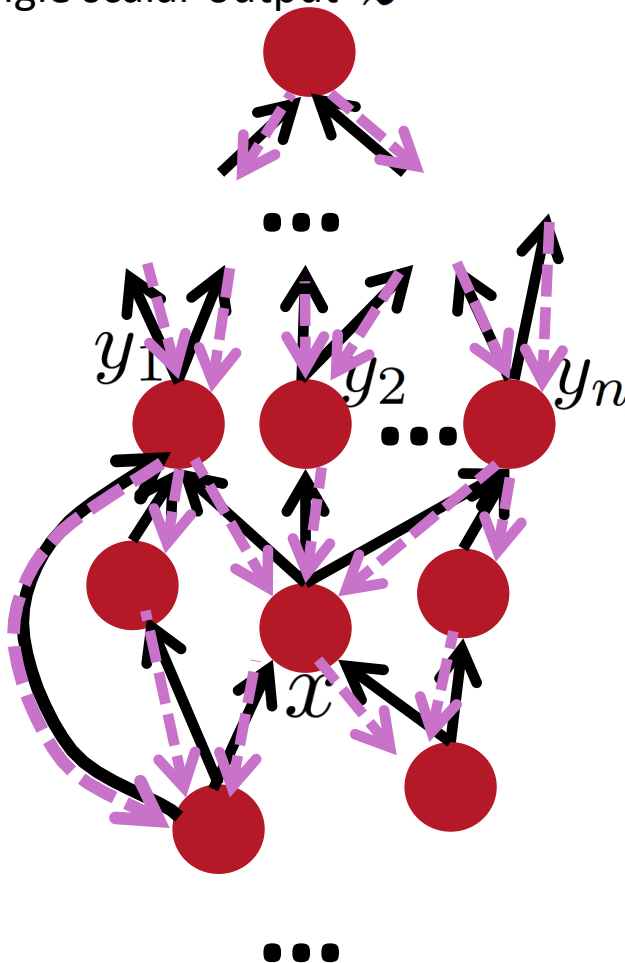


Backprop in
Multi-Layer
Net:

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:

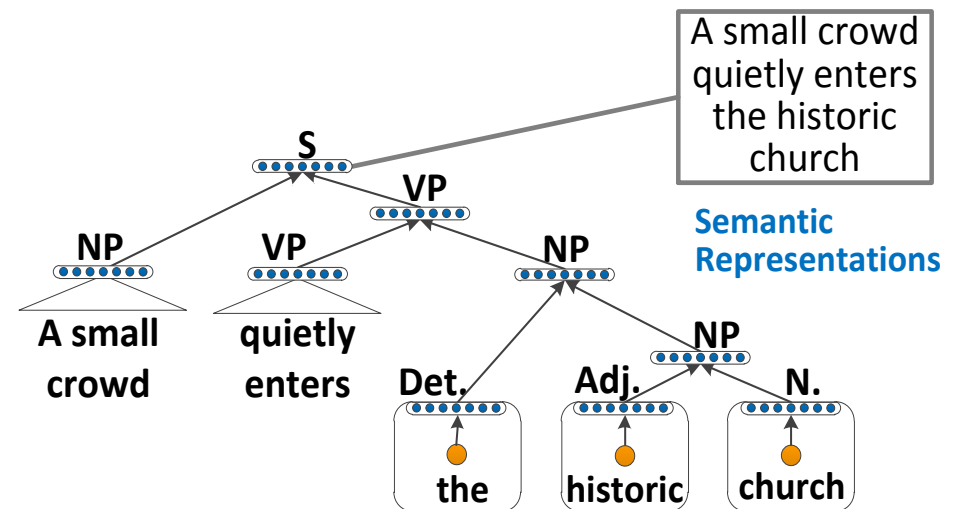
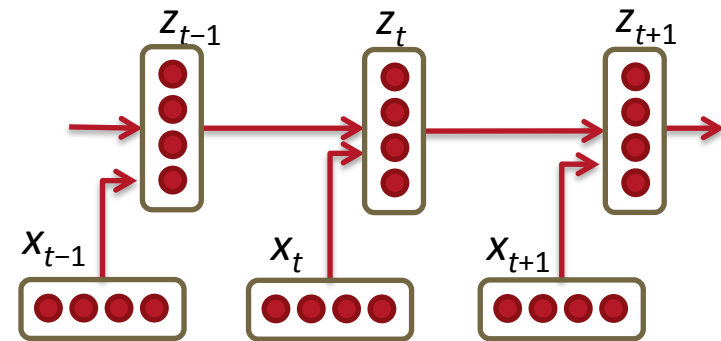
Compute gradient wrt each node using gradient wrt successors

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

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

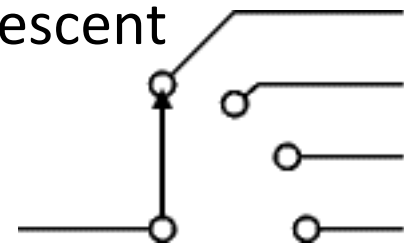
Back-Prop in Recurrent & Recursive Nets

- 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

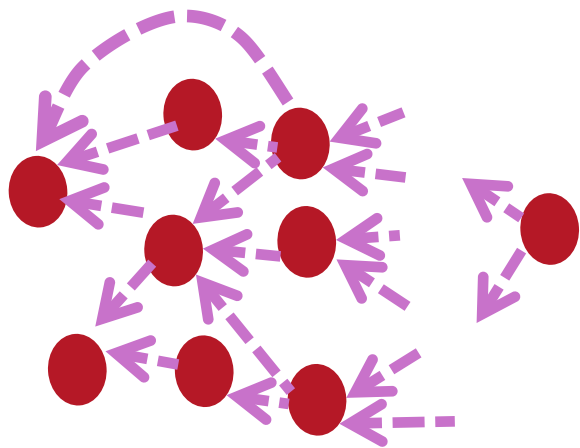
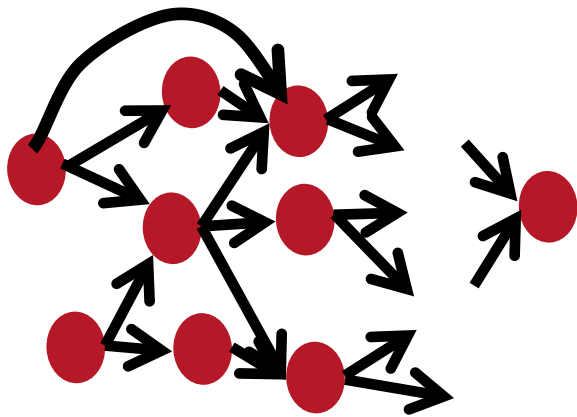


Backpropagation Through Structure

- Inference $\rightarrow$ 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
 - $\text{Max}(a,b)$ is continuous.



Automatic Differentiation



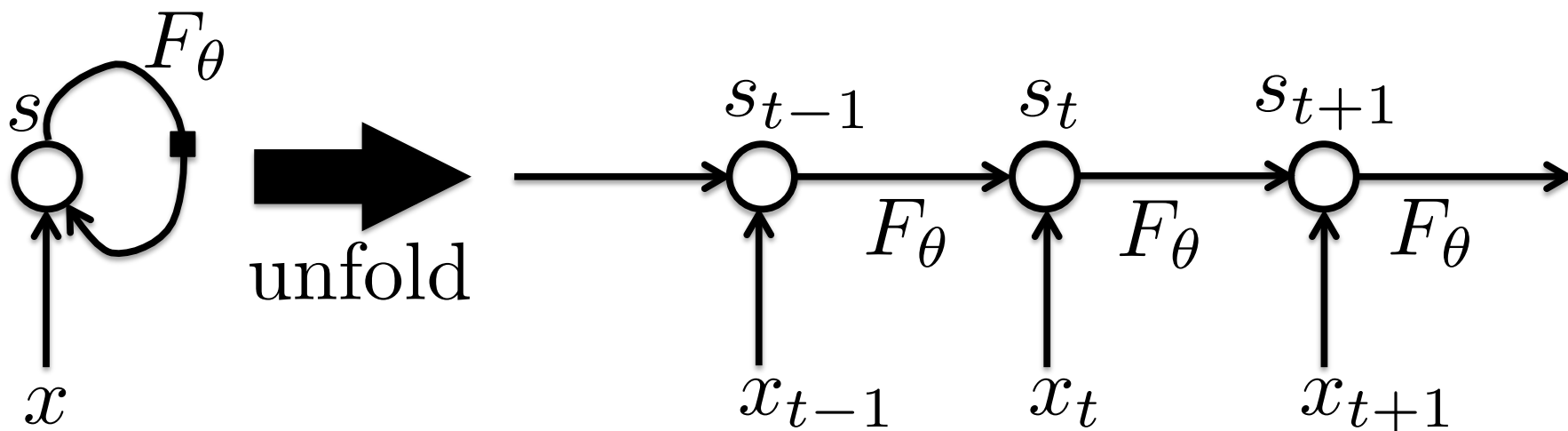
- The gradient computation can be automatically inferred from the symbolic expression of the fprop.
- Each node type needs to know how to compute its output and how to compute the gradient wrt its inputs given the gradient wrt its output.
- Easy and fast prototyping

theano

Recurrent Neural Networks

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

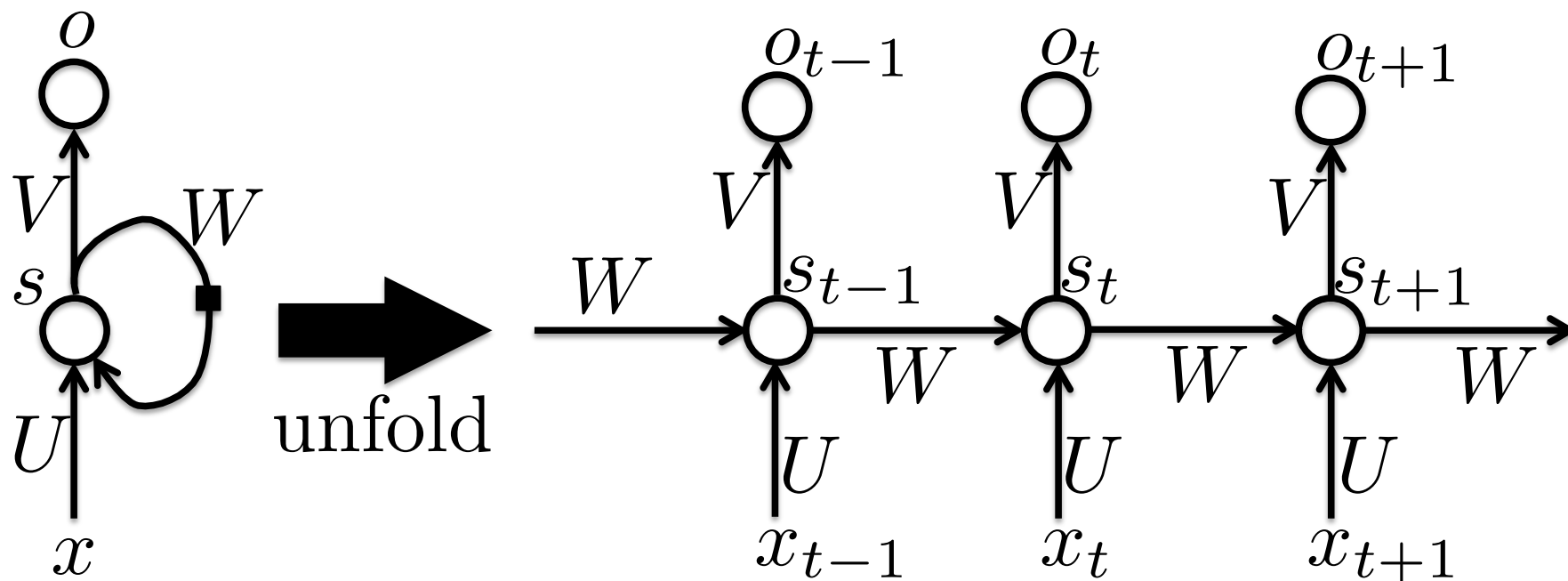
$$s_t = F_\theta(s_{t-1}, x_t)$$



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

Recurrent Neural Networks

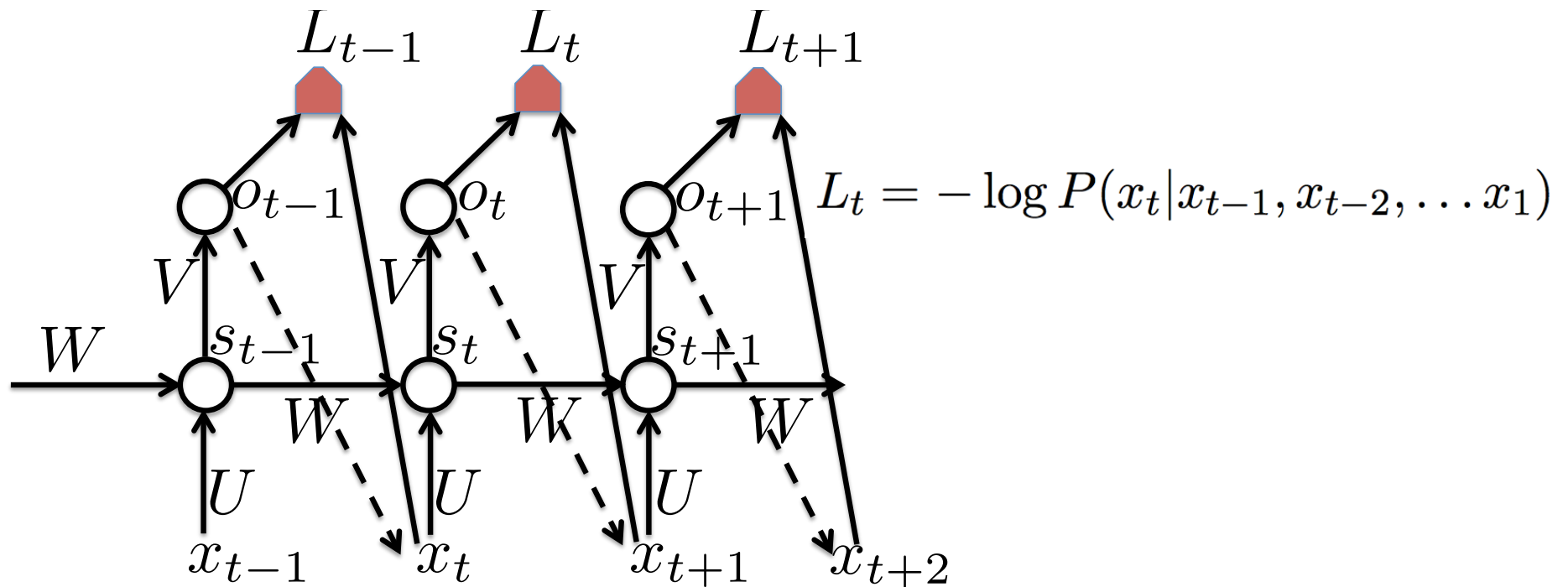
- Can produce an output at each time step: unfolding the graph tells us how to back-prop through time.



Generative RNNs

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

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

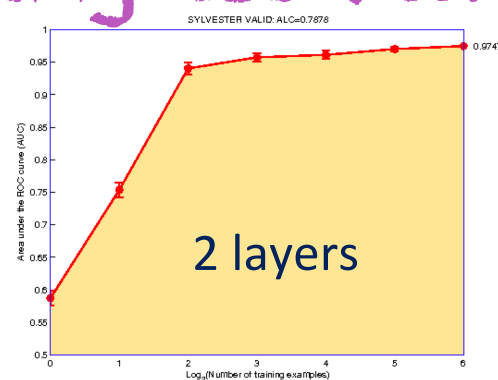
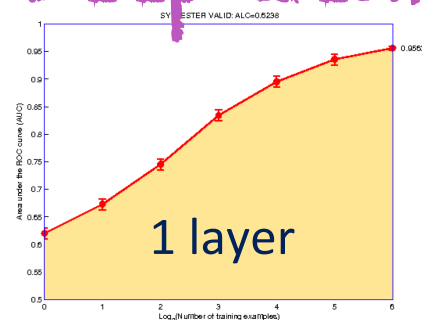
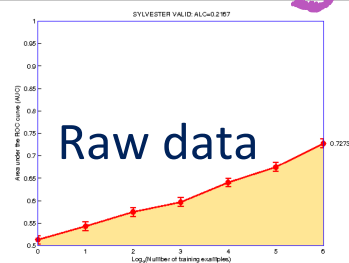


Unsupervised or Semi-Supervised Deep Learning & Generative Deep Learning

The Next Challenge: Unsupervised Learning

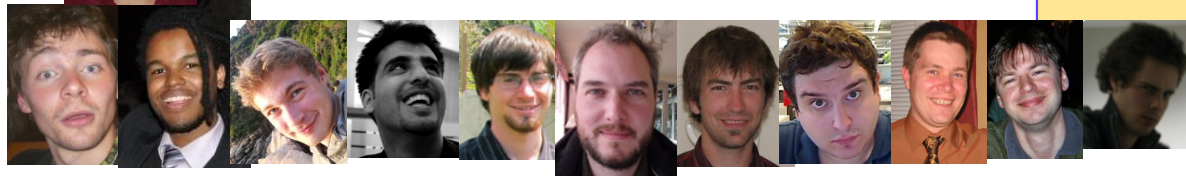
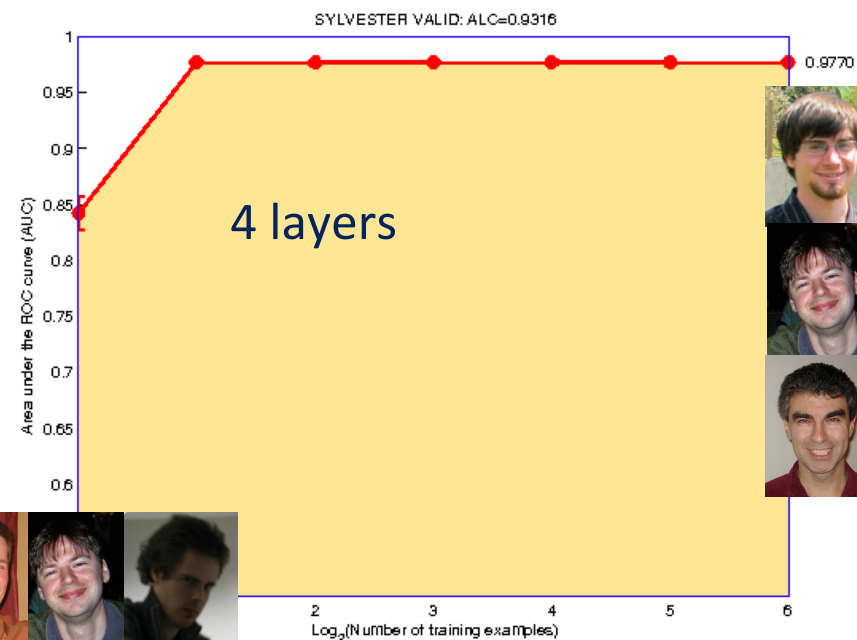
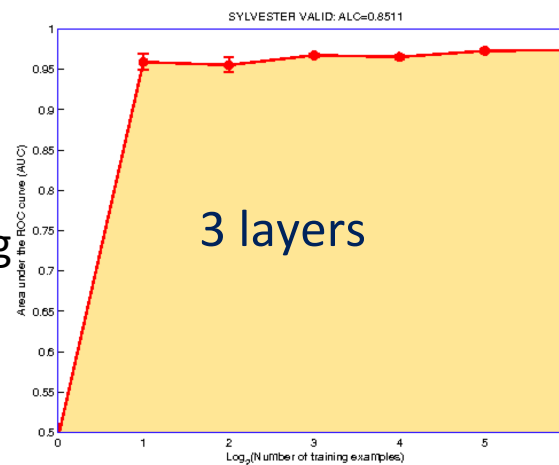
- Recent progress mostly in supervised DL
- Real technical challenges for unsupervised DL
- Potential benefits:
 - Exploit tons of unlabeled data
 - Answer new questions about the variables observed
 - Regularizer – transfer learning – domain adaptation
 - Easier optimization (local training signal)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

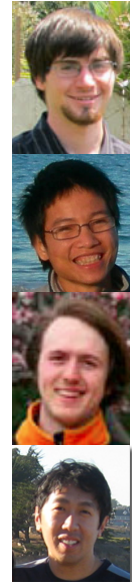
Invariance and Disentangling

- Invariant features
- Which invariances?
- Alternative: learning to disentangle factors
- Good disentangling →
avoid the curse of dimensionality



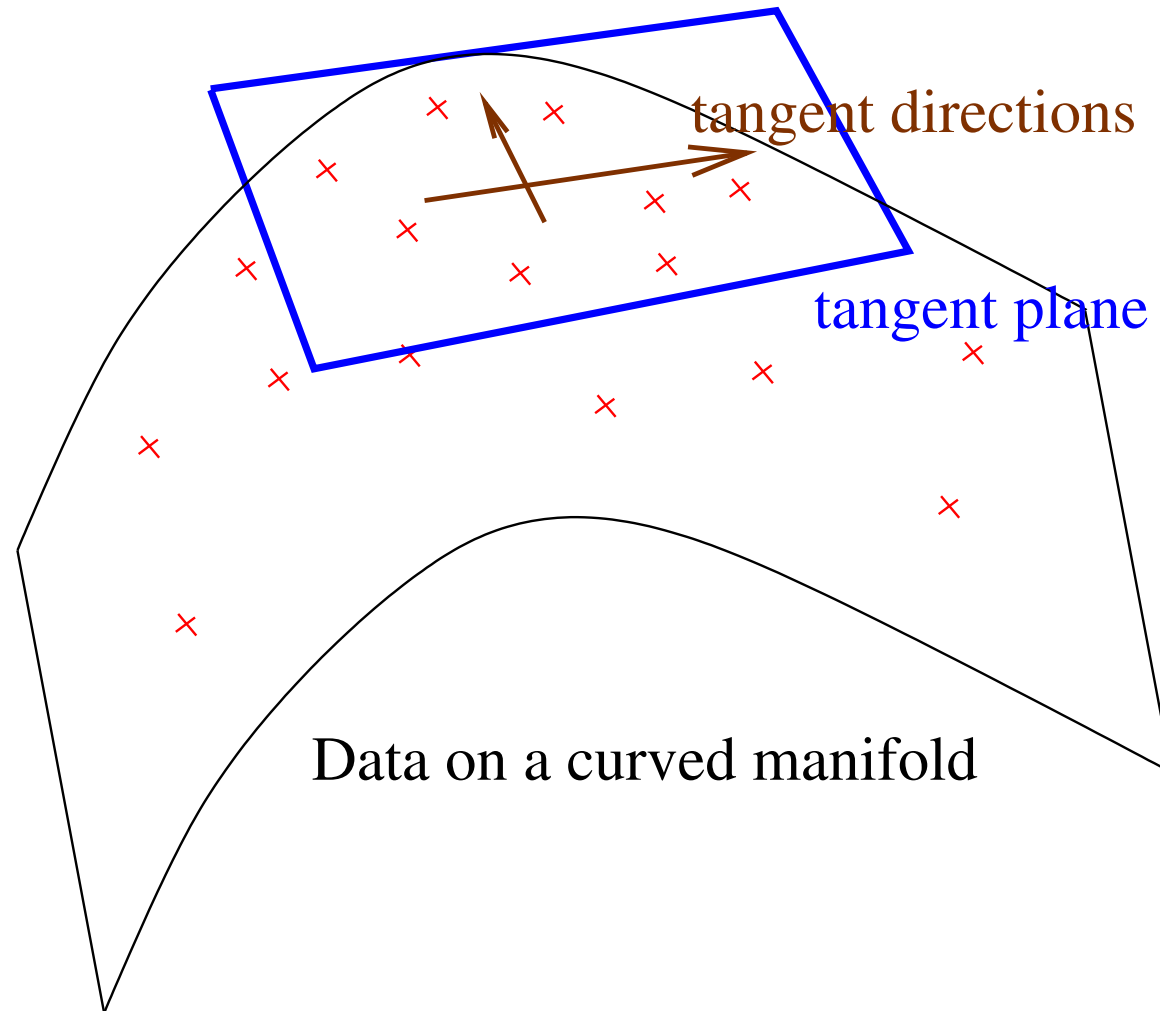
Emergence of Disentangling

- (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)

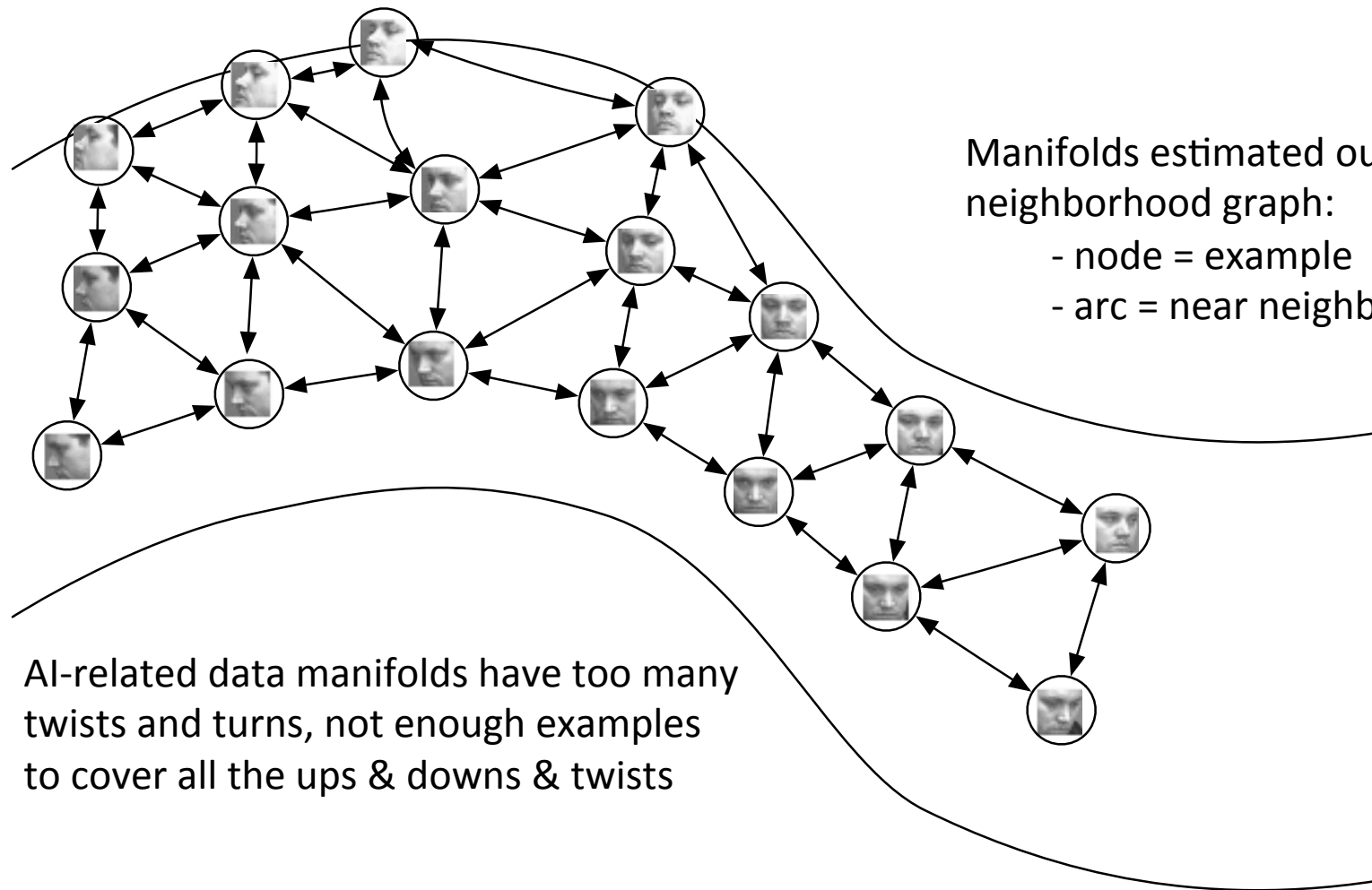


WHY?

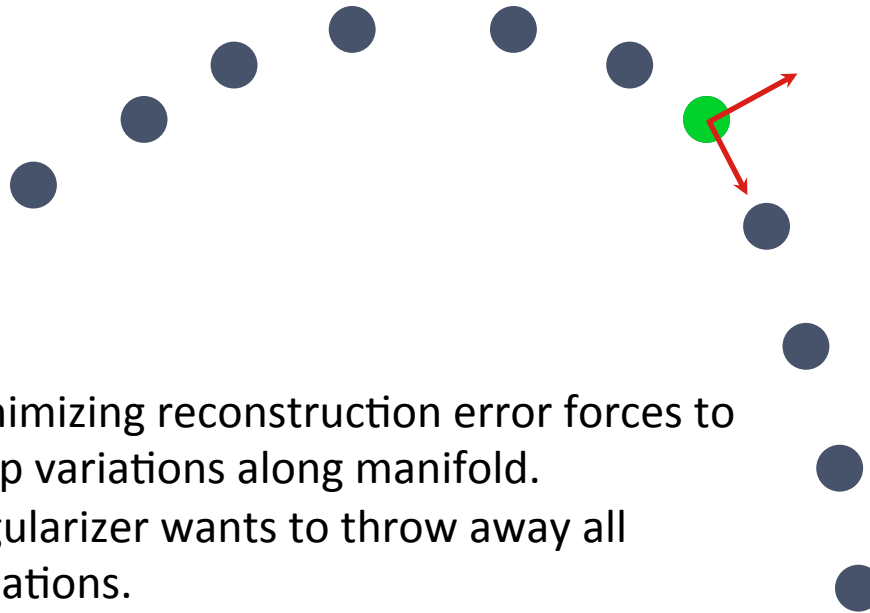
Manifold Learning = Representation Learning



Non-Parametric Manifold Learning: hopeless without powerful enough priors



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

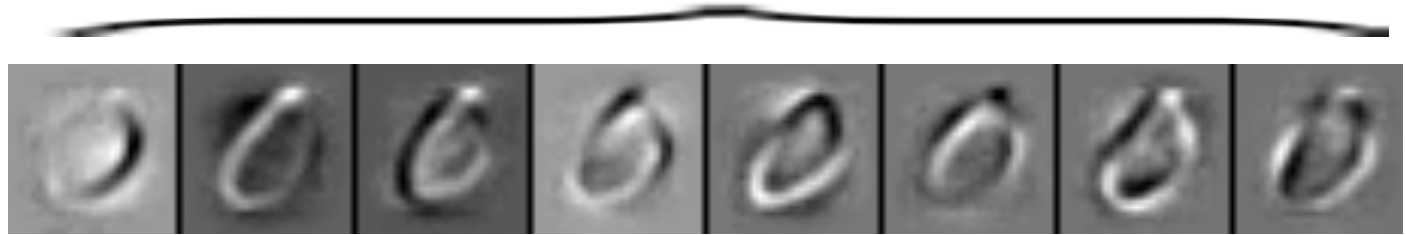


- Minimizing reconstruction error forces to keep variations along manifold.
- Regularizer wants to throw away all variations.
- With both: keep ONLY sensitivity to variations ON the manifold.

Input Point



Tangents



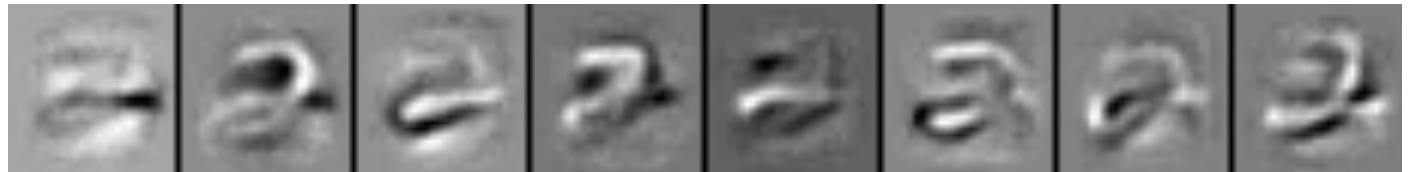
$$\text{Input Point} + 0.5 \times \text{Tangent} = \text{Result}$$

MNIST

Input Point



Tangents

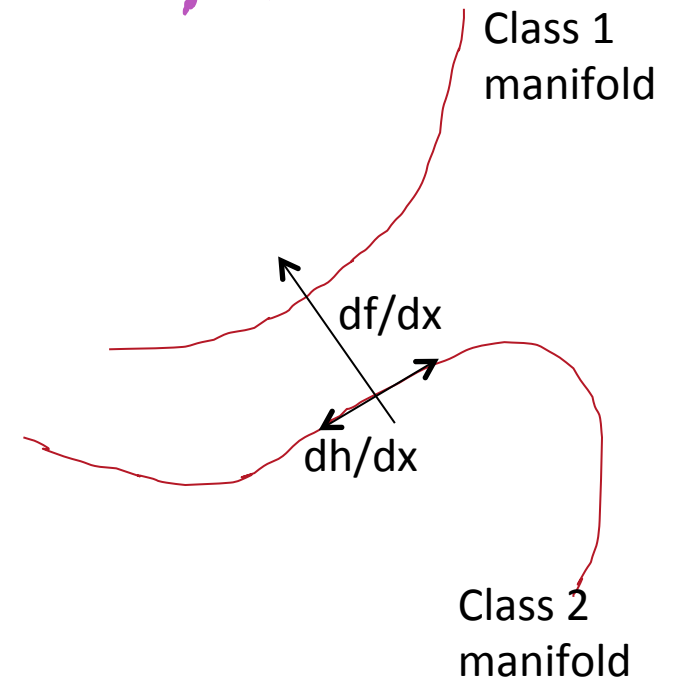


MNIST Tangents

Learned Tangent Prop: the Manifold Tangent Classifier

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

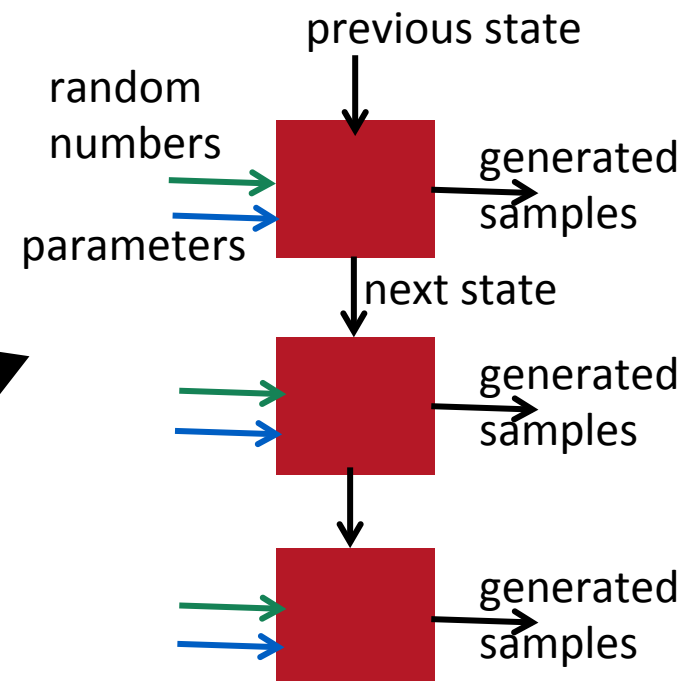
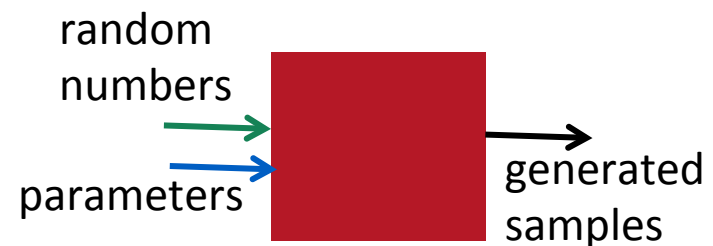
Tangent plane characterized by $dh(x)/dx$



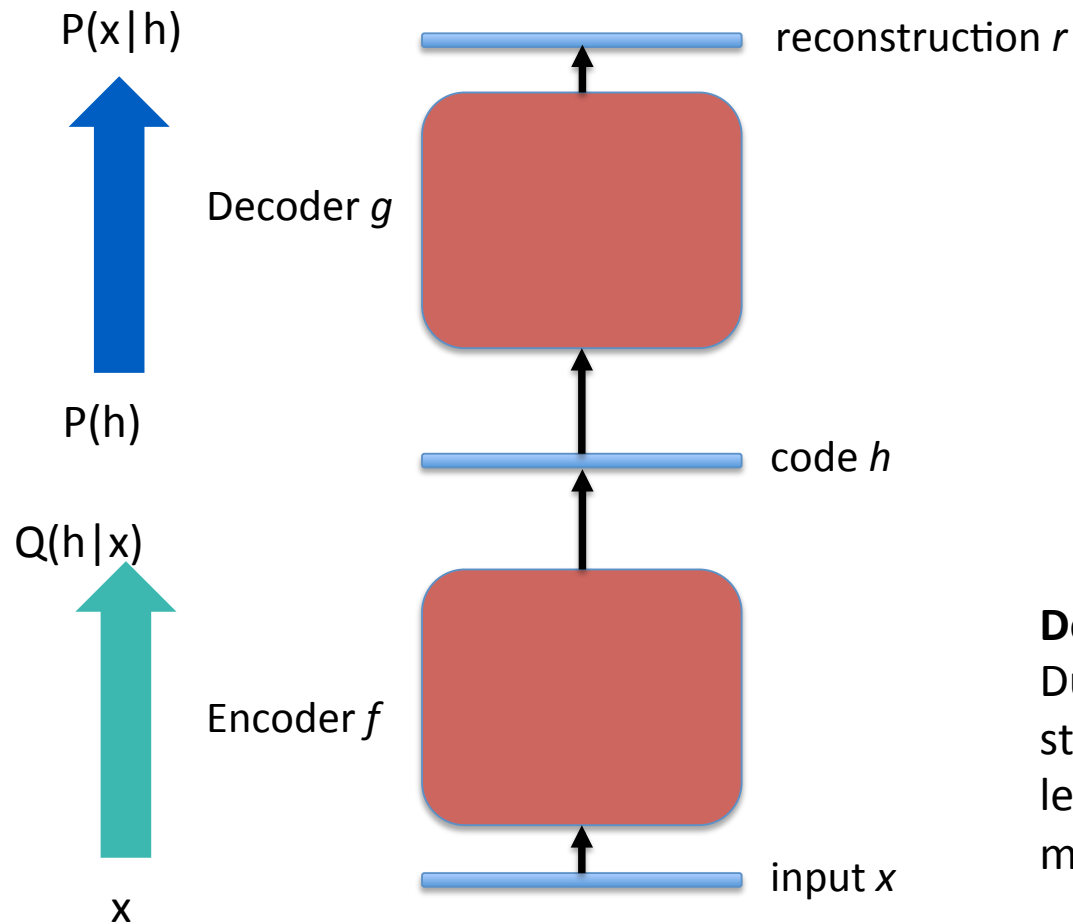
(Rifai et al NIPS'2012)

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)



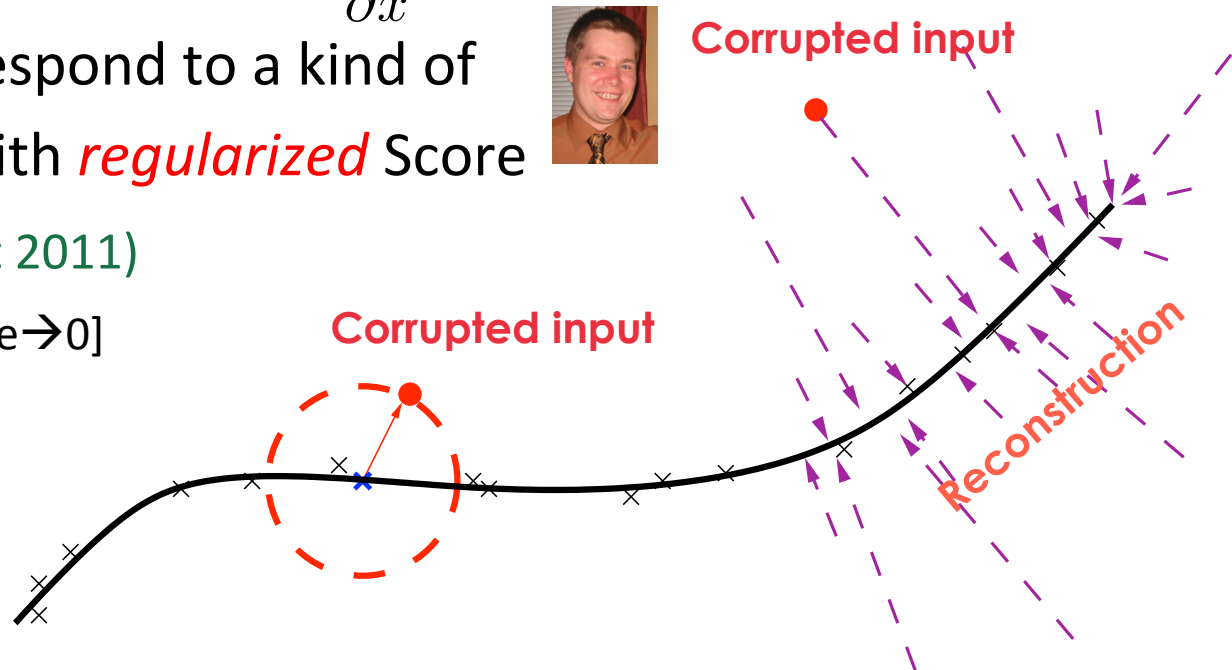
$$\text{reconstruction}(x) - x \rightarrow \sigma^2 \frac{\partial \log p(x)}{\partial x}$$

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



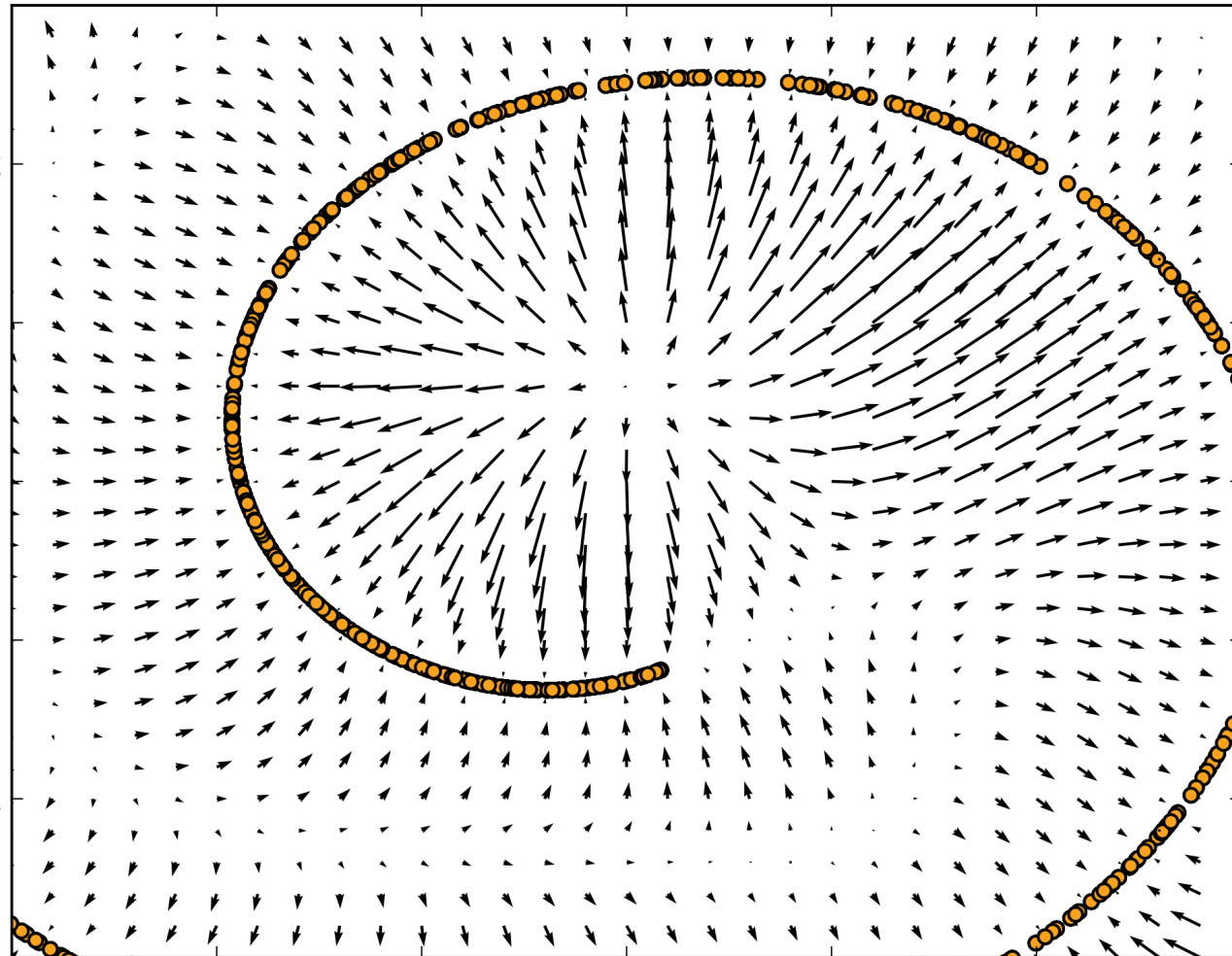
[equivalent when noise $\rightarrow 0$]

prior: examples concentrate near a lower dimensional "manifold"

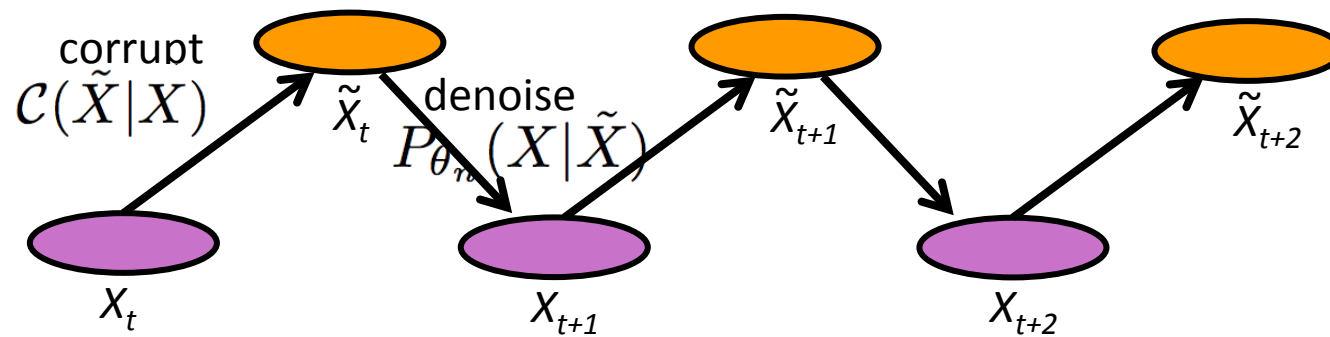


Regularized Auto-Encoders Learn a Vector Field that Estimates a Gradient Field

(Alain & Bengio ICLR 2013)

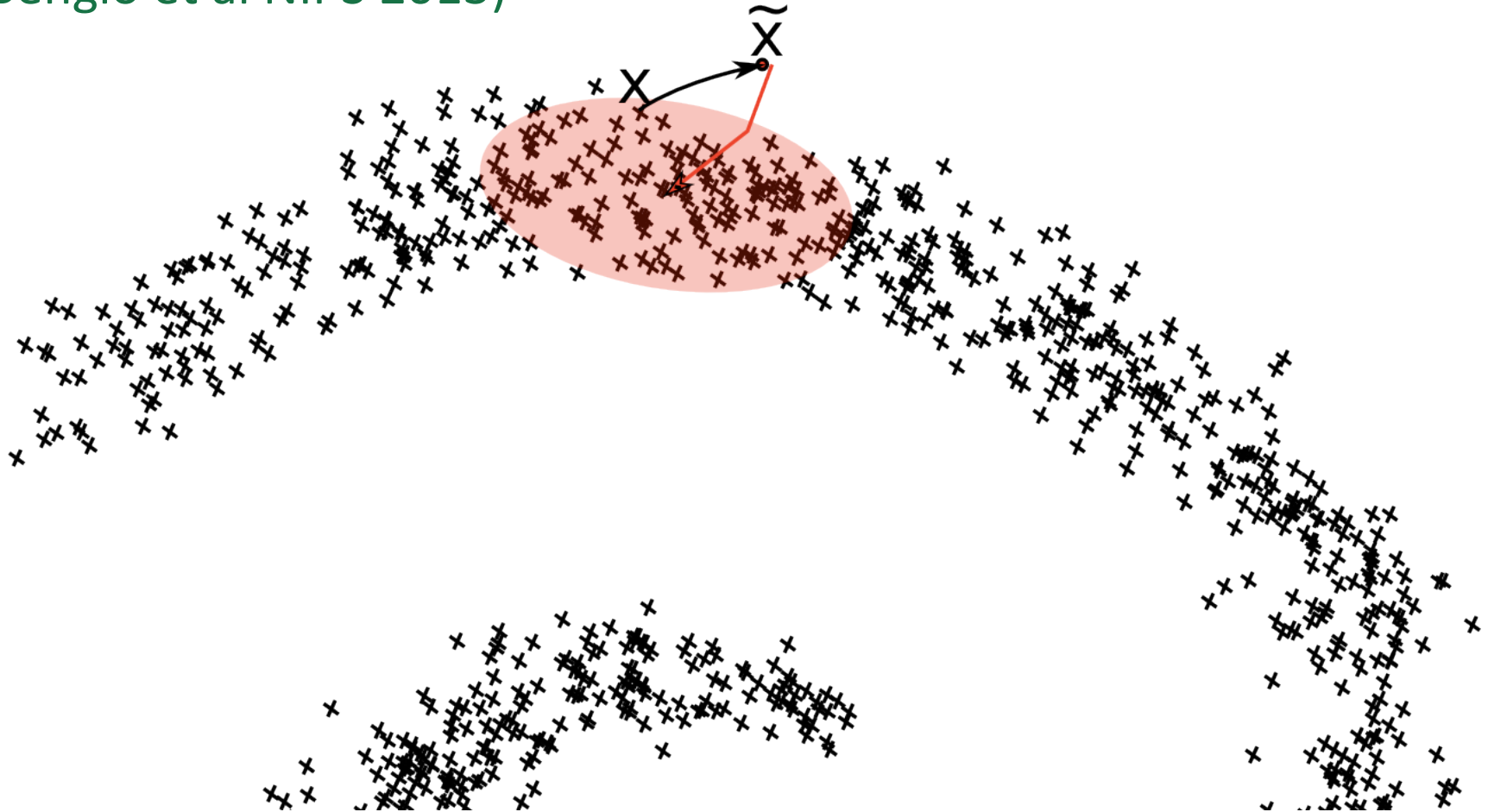


Denoising Auto-Encoder Markov Chain



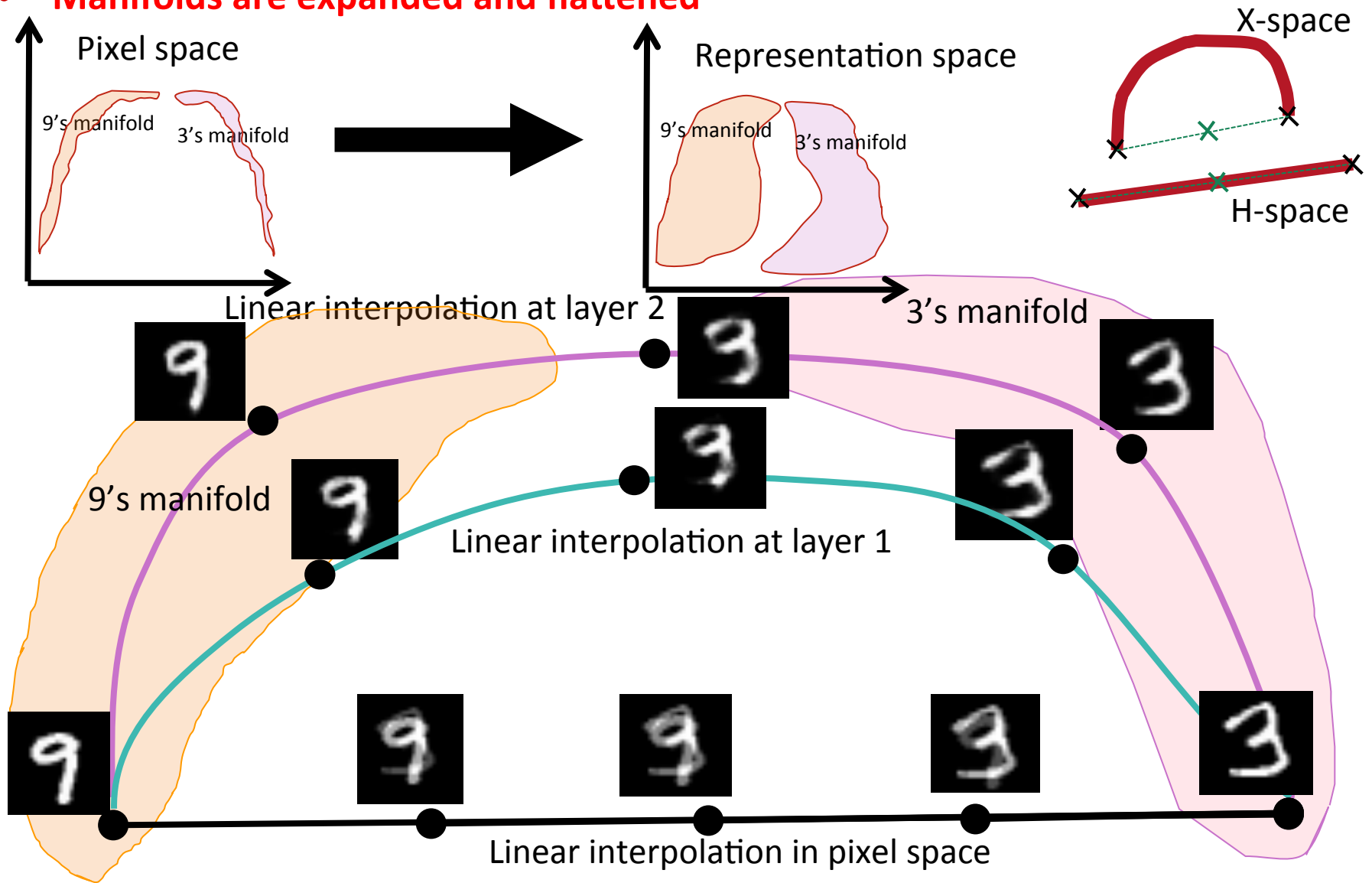
Denoising Auto-Encoders Learn a Markov Chain Transition Distribution

(Bengio et al NIPS 2013)



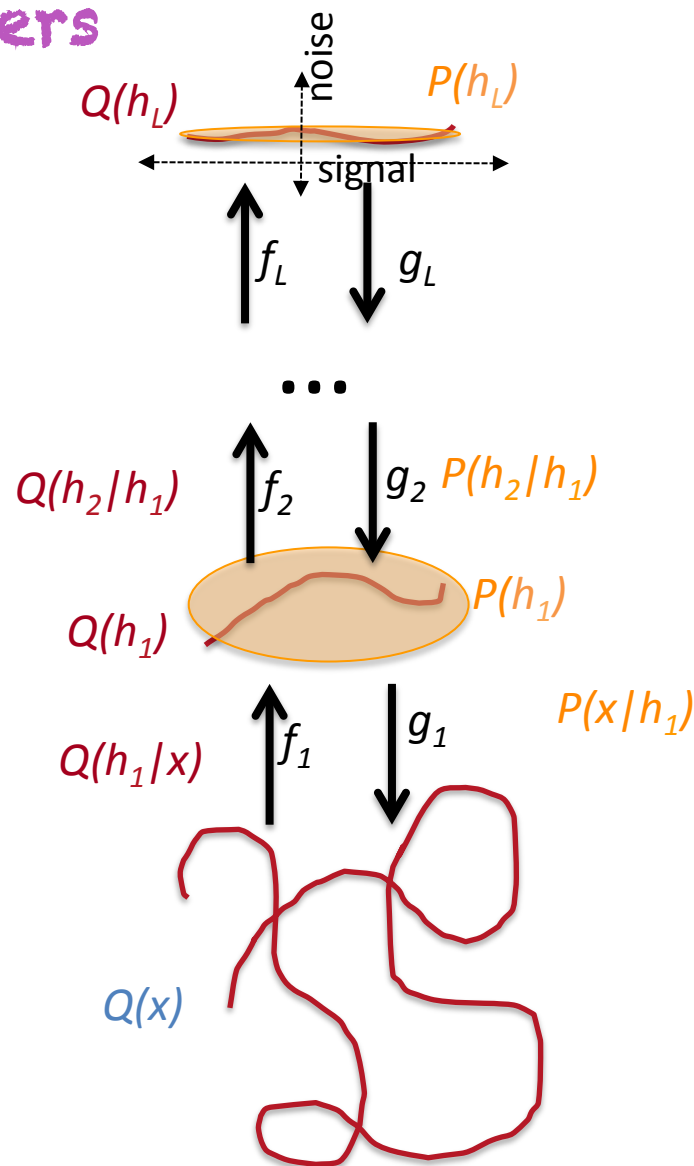
Space-Filling in Representation-Space

- Deeper representations $\rightarrow$ abstractions $\rightarrow$ disentangling
- Manifolds are expanded and flattened



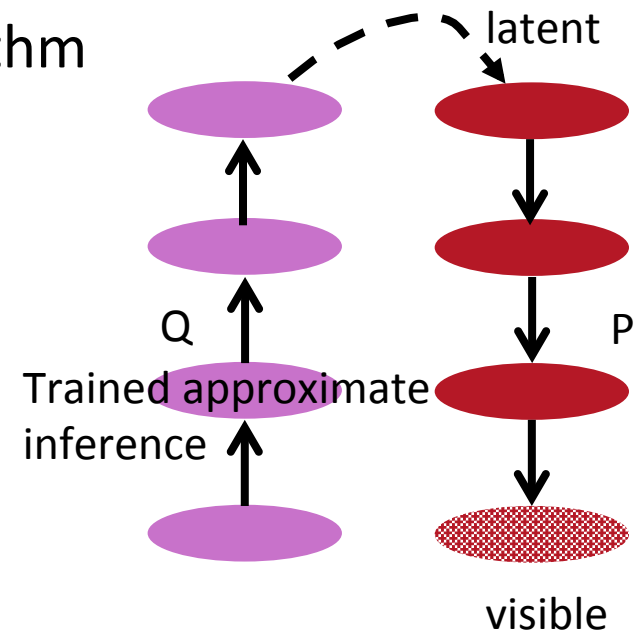
Extracting Structure By Gradual Disentangling and Manifold Unfolding & Variational Auto-Encoders

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.



Ancestral Sampling with Learned Approximate Inference: Replace Intractable $P(h|x)$ by Learned $Q(h|x)$

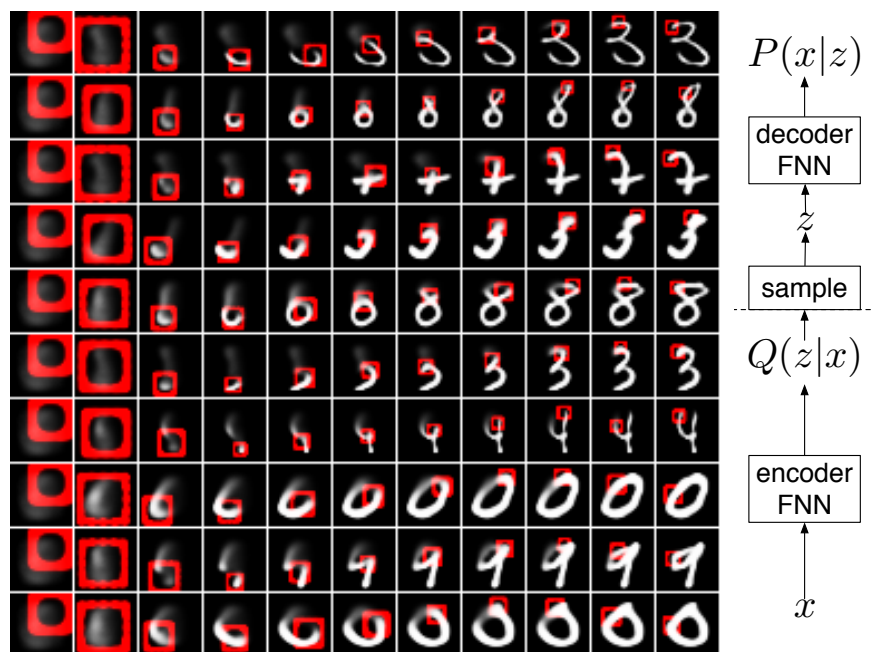
- Helmholtz machine & Wake-Sleep algorithm
 - (Hinton, Dayan, Frey, Neal, 1995; Dayan, Hinton, Neal, Zemel 1995)
- Variational Auto-Encoders
 - (Kingma & Welling 2013, ICLR 2014)
 - (Gregor et al ICML 2014)
 - (Rezende et al ICML 2014)
 - (Mnih & Gregor ICML 2014)
- Reweighted Wake-Sleep (Bornschein & Bengio 2014, ICML2015)
- Target Propagation (Bengio 2014)
- Deep Directed Generative Auto-Encoders (Ozair & Bengio 2014)
- NICE (Dinh et al 2014)



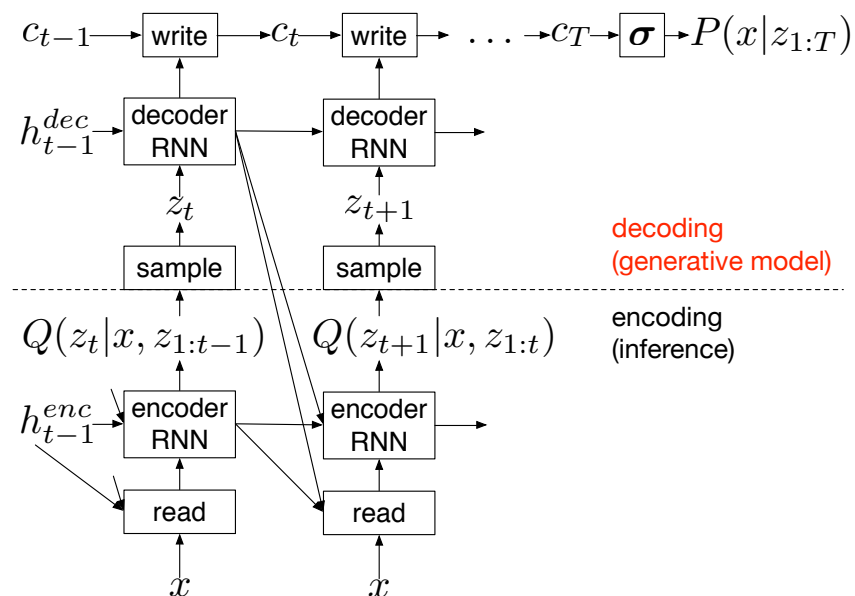
DRAW: the latest variant of Variational Auto-Encoder

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

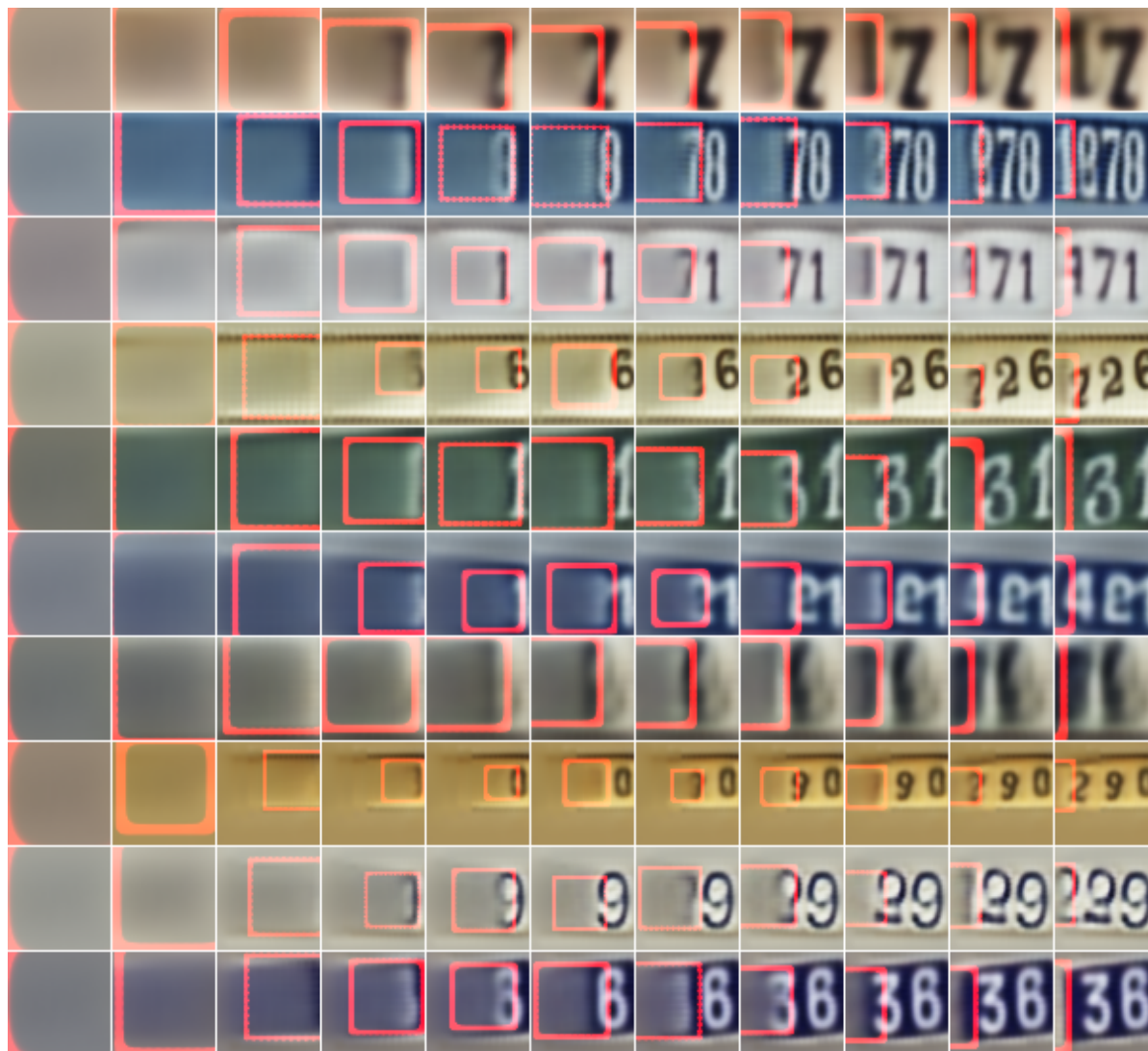
- Even for a static input, the encoder and decoder are now **recurrent nets**, which gradually add elements to the answer, and use an attention mechanism to choose where to do so.



Time $\longrightarrow$



DRAW Samples of SVHN Images: the drawing process



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

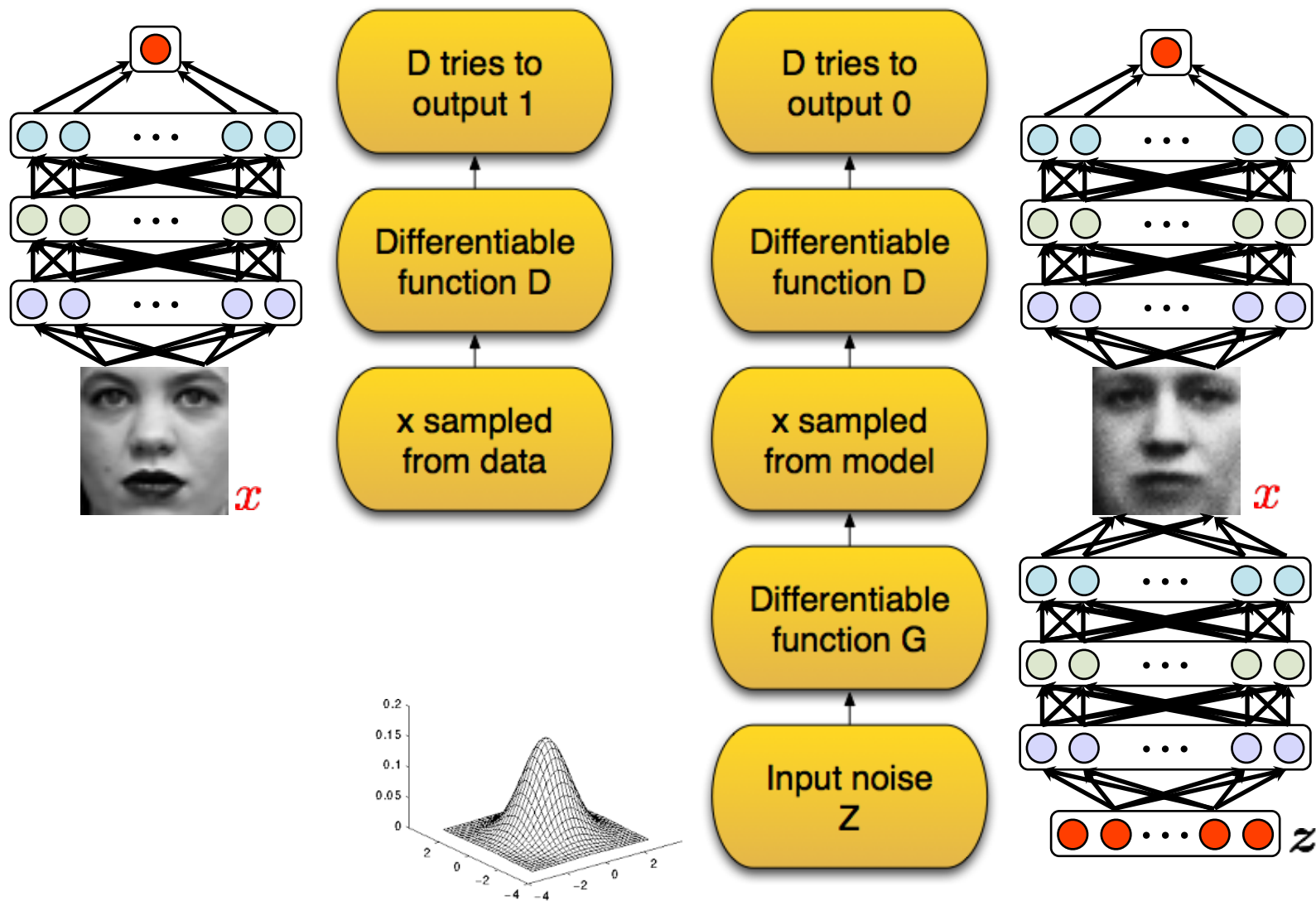


Nearest training
example for last
column of samples

Generative adversarial networks

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

Adversarial nets framework



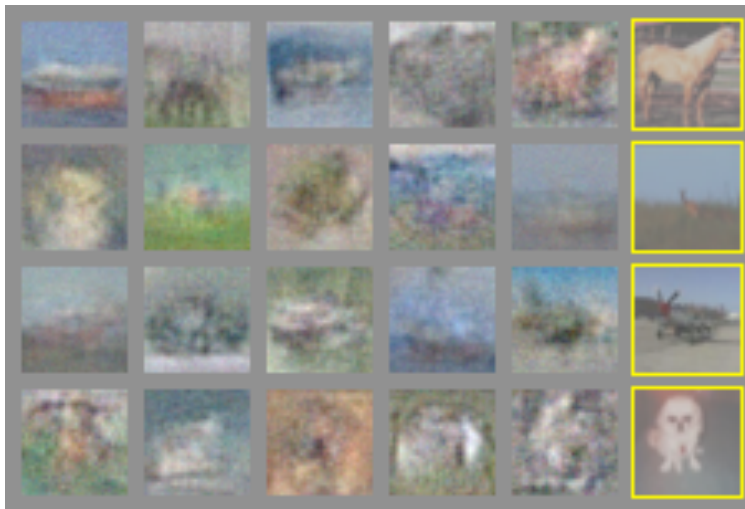
Visualization of model samples



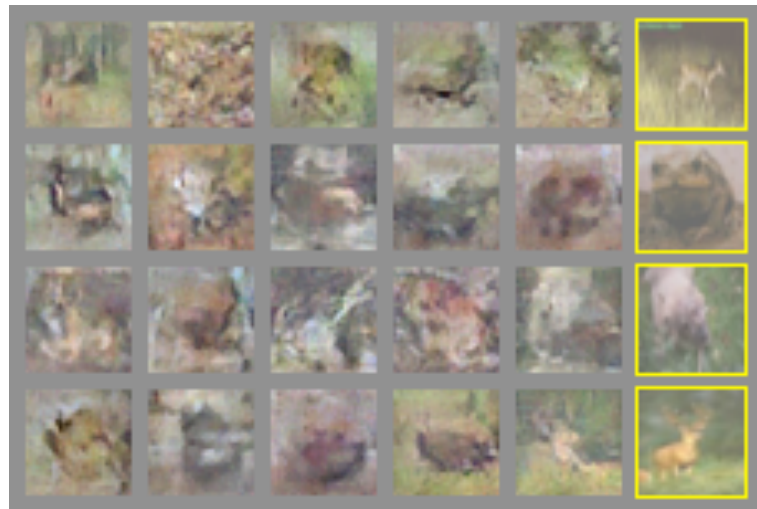
MNIST



TFD

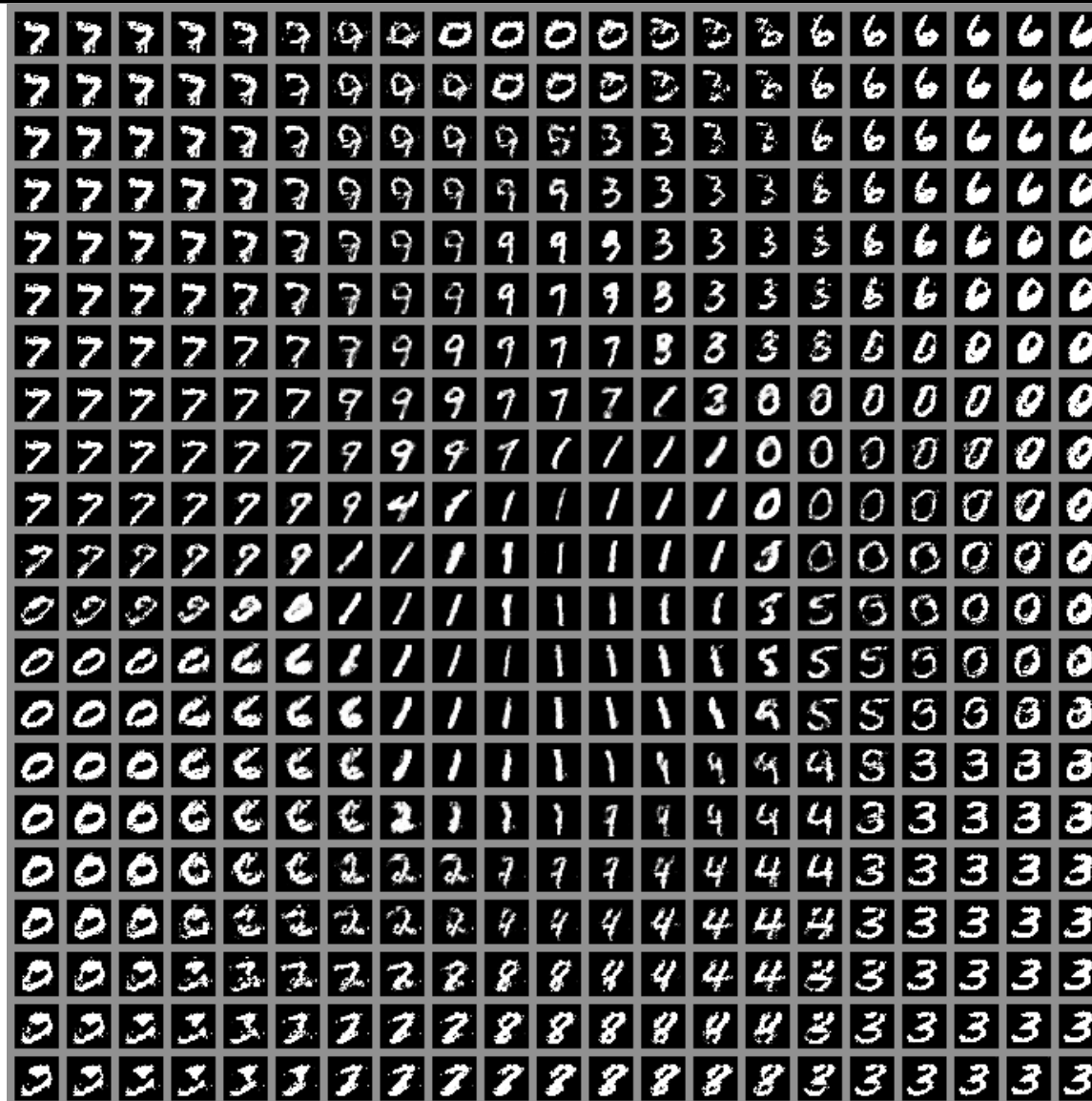


CIFAR-10 (fully connected)

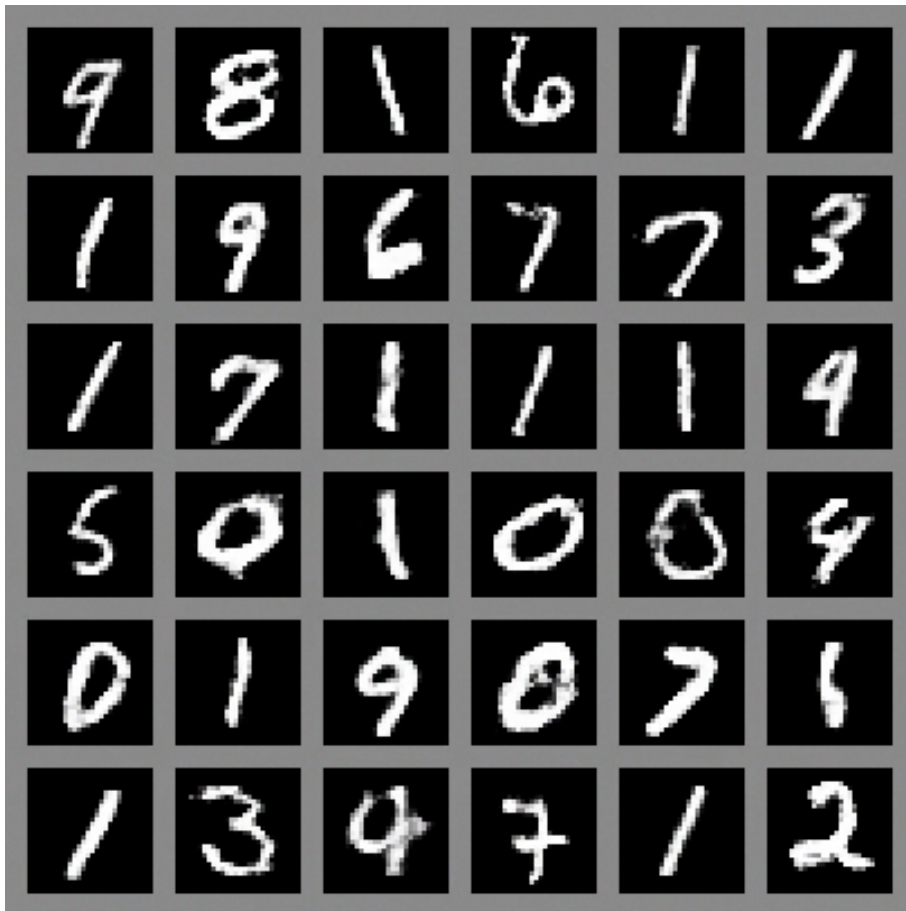


CIFAR-10 (convolutional)

Learned 2-D manifold of MNIST



Visualization of model trajectories



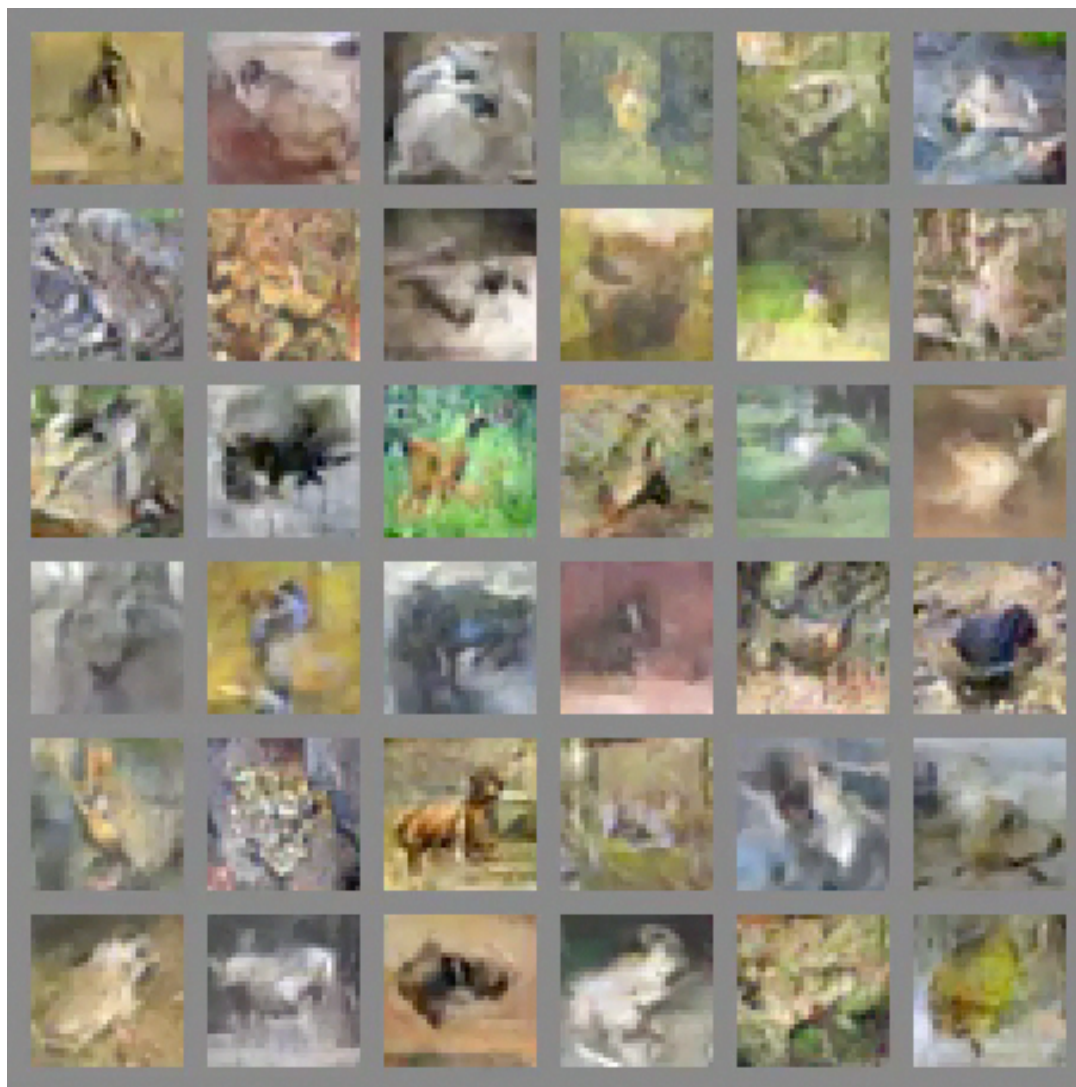
MNIST digit dataset



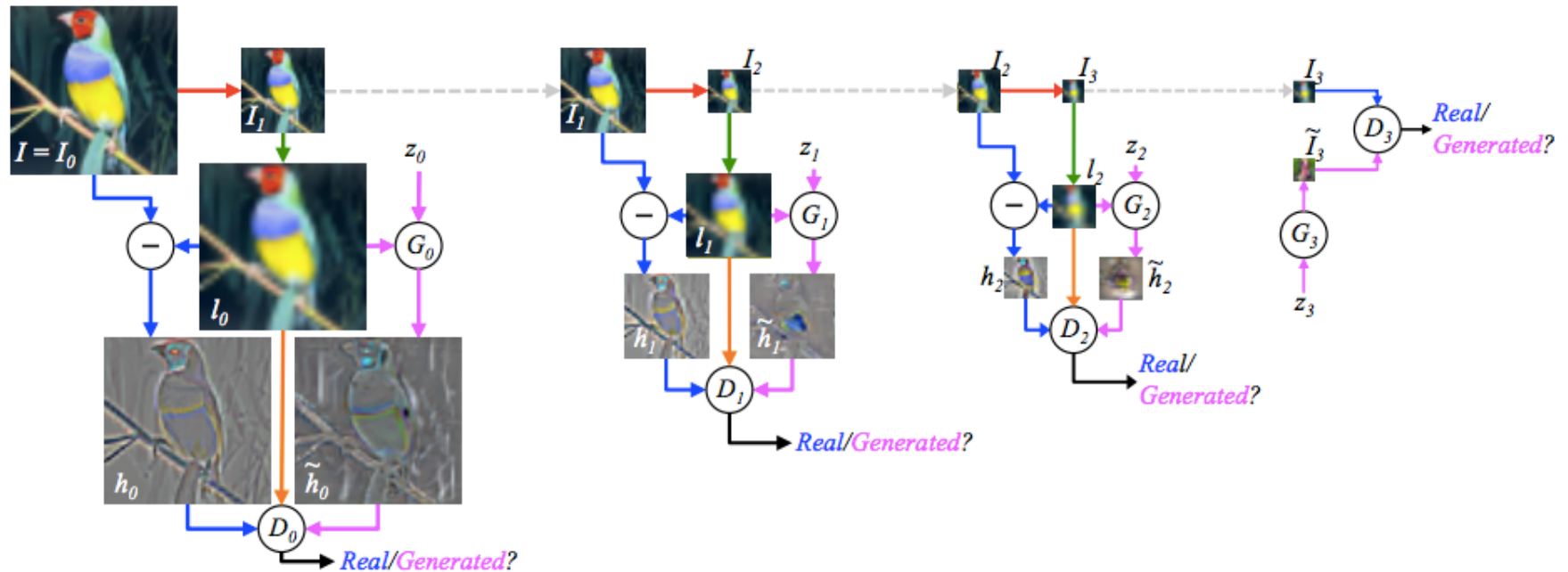
Toronto Face Dataset
(TFD)

Visualization of model trajectories

CIFAR-10
(convolutional)



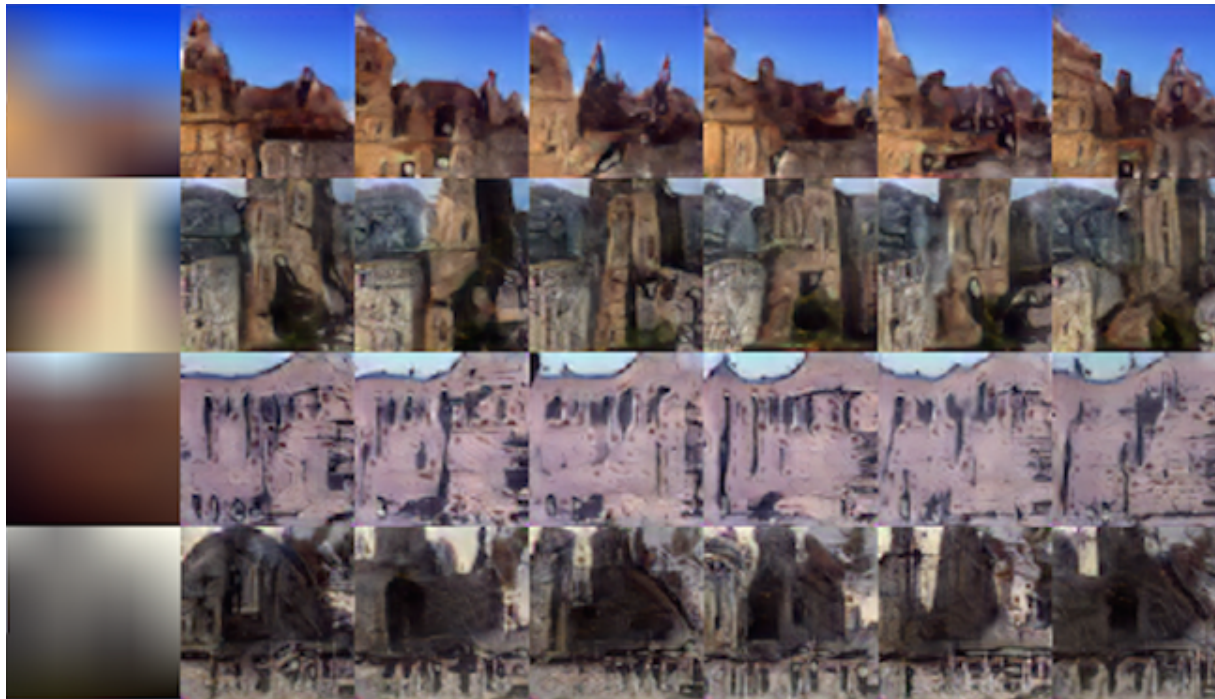
Laplacian Pyramid



(Denton + Chintala, et al 2015)

LAPGAN results

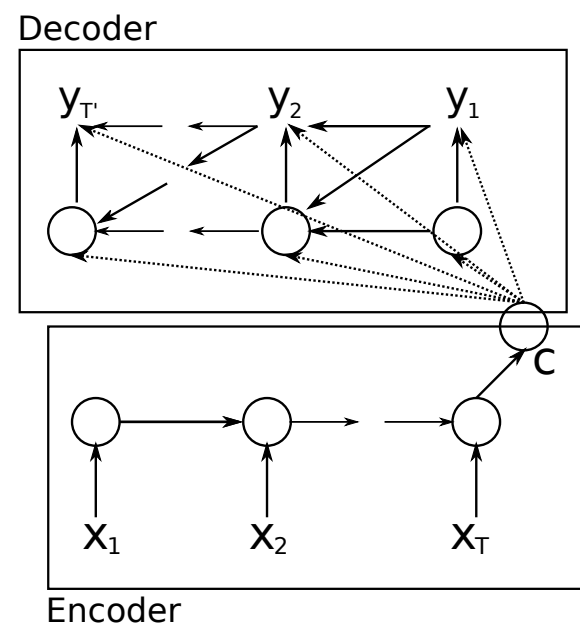
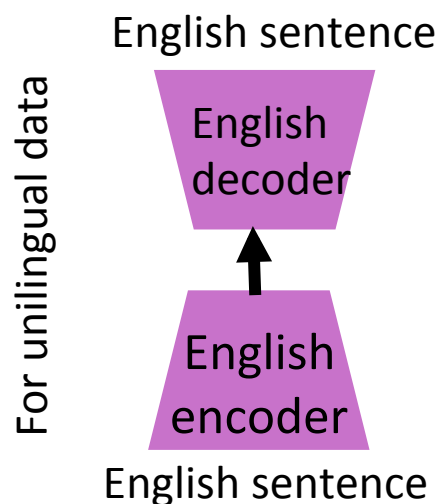
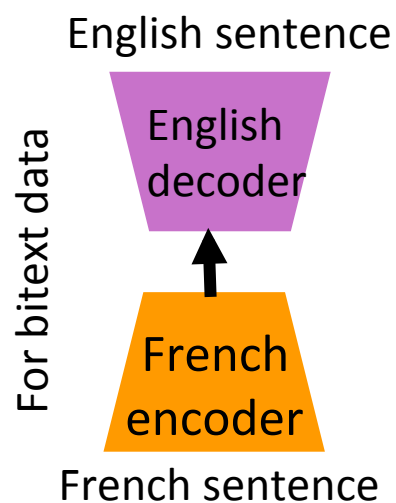
- 40% of samples mistaken *by humans* for real photos



(Denton + Chintala, et al 2015)

Encoder-Decoder Framework

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



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)

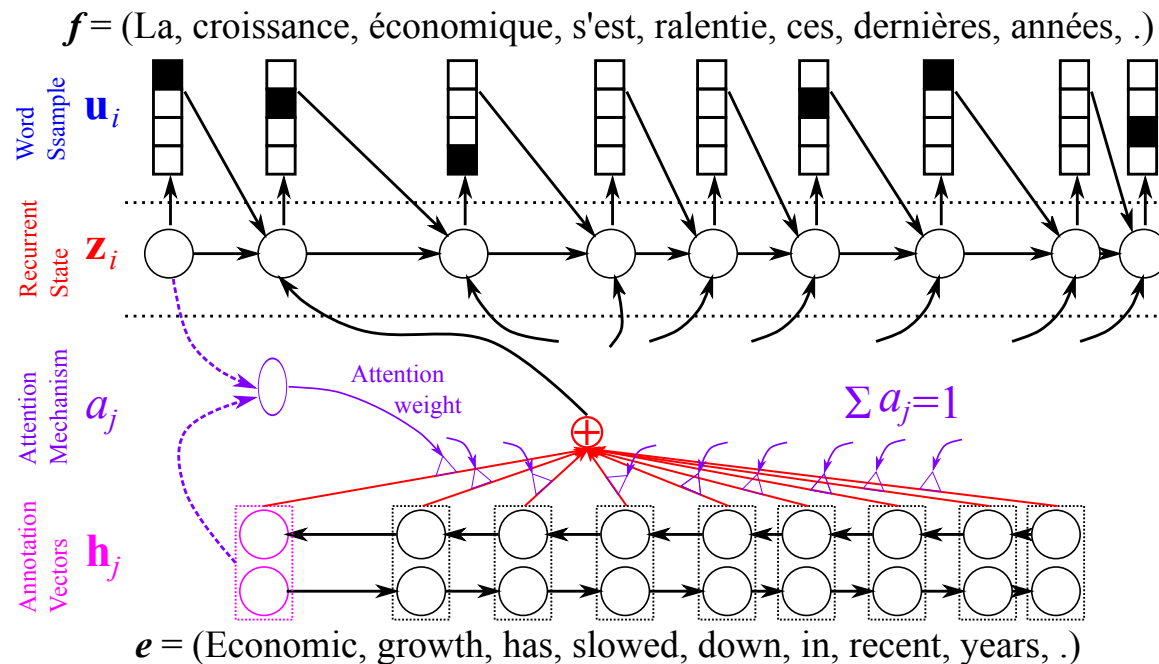
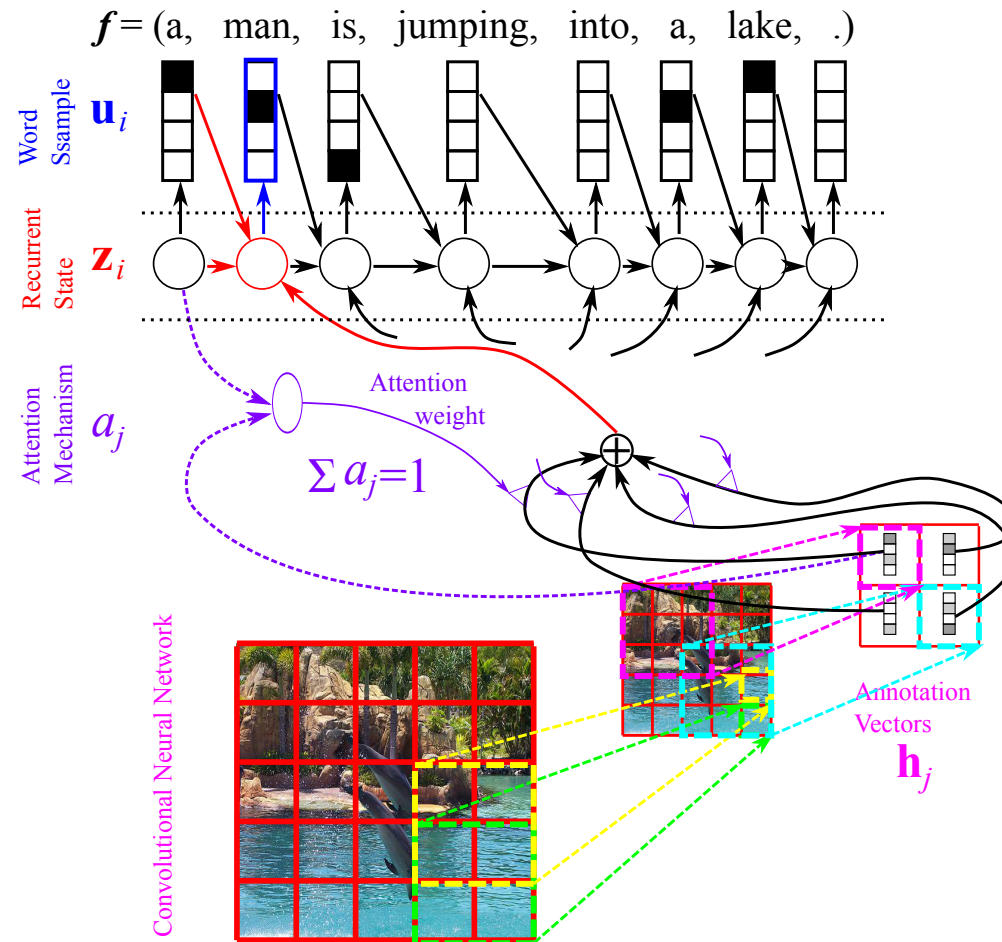


Image-to-Text: Caption Generation



(Xu et al., 2015), (Yao et al., 2015)

Navigation icons: back, forward, search, etc.

The Good



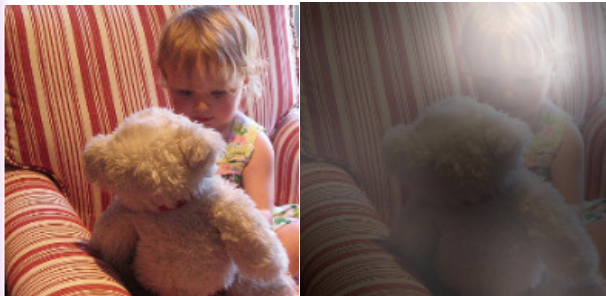
A woman is throwing a frisbee in a park.



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.

Conclusions

- **Deep learning works**
 - flexible family of functions and distributions
 - Simple but powerful optimization by SGD, gradient from backprop
 - Lack of convexity but local minima not
 - handles fixed-size vectors, or variable size sequences & images
 - Recurrent nets & convolutional nets
 - large labeled datasets or the combination of large unlabeled and small labeled datasets or auxiliary tasks (transfer learning)
 - upcoming challenges in unsupervised learning, generative models of images

MILA: Montreal Institute for Learning Algorithms

