Obstacles
On the path to AI
(or "how I learned to stop worrying and love unsupervised learning")

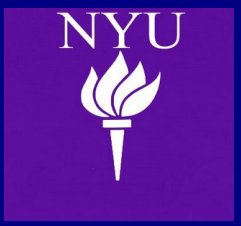
Yann LeCun

Facebook AI Research &

Center for Data Science, NYU

yann@cs.nyu.edu

http://yann.lecun.com



Or:
How do we incorporate Reasoning/Planning with Representation learning?

Yann LeCun
Facebook AI Research &
Center for Data Science, NYU
yann@cs.nyu.edu
http://yann.lecun.com

Or:
How do we incorporate Episodic Memory with Representation learning?

Yann LeCun
Facebook AI Research &
Center for Data Science, NYU
yann@cs.nyu.edu
http://yann.lecun.com

And How the Hell do we do Unsupervised Learning?

Yann LeCun

Facebook AI Research &

Center for Data Science, NYU

yann@cs.nyu.edu

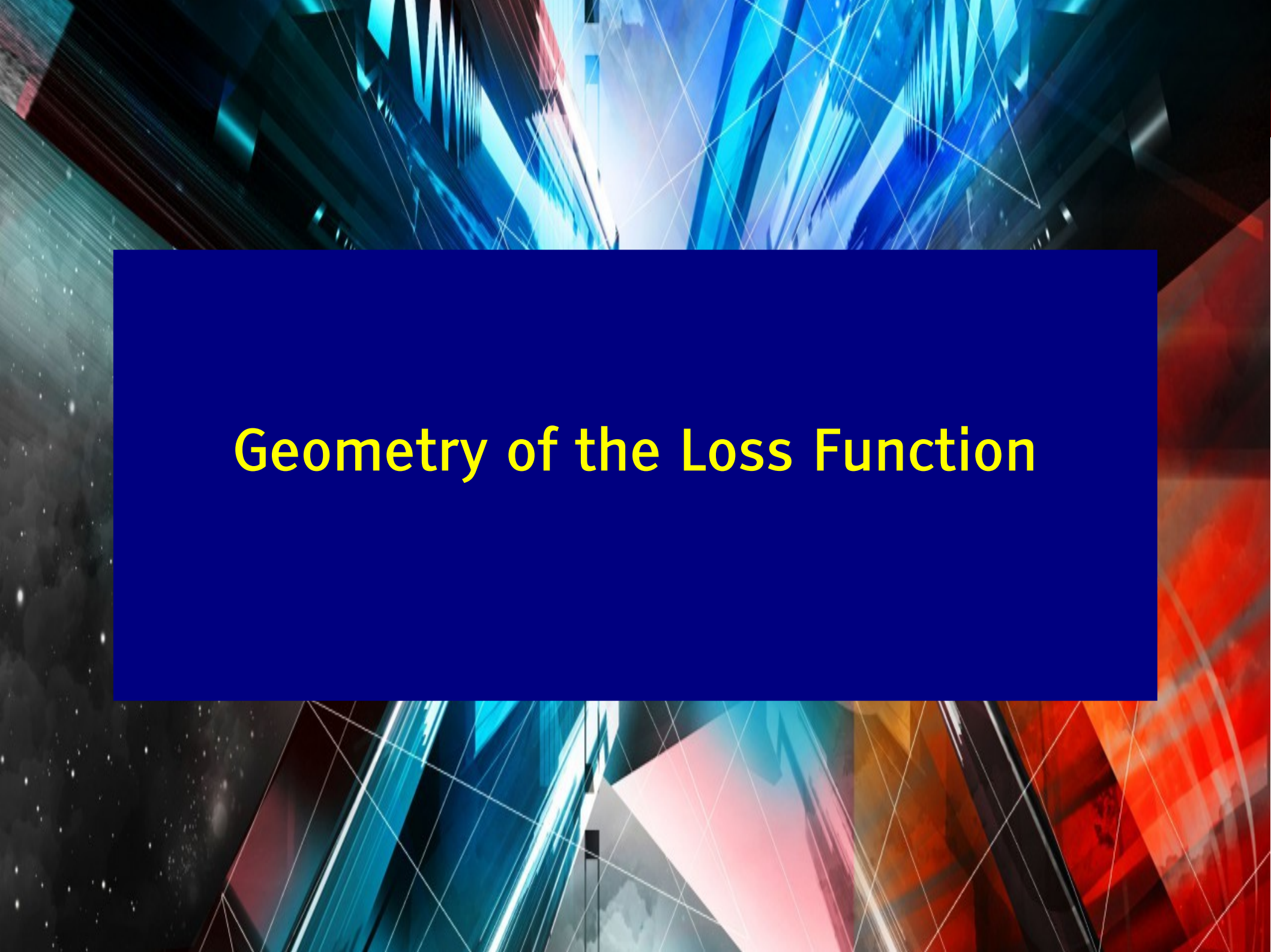
http://yann.lecun.com

Wait, what about reinforcement learning?

Y LeCun

- **Meh, it's the cherry on the cake.**

- This sounds like trolling.
- Ok, just a little bit.
- But there is no way in hell that you can learn billions of parameters with RL.
  - At least not within a reasonable amount of time.
  - One scalar reward per trial isn't going to cut it.

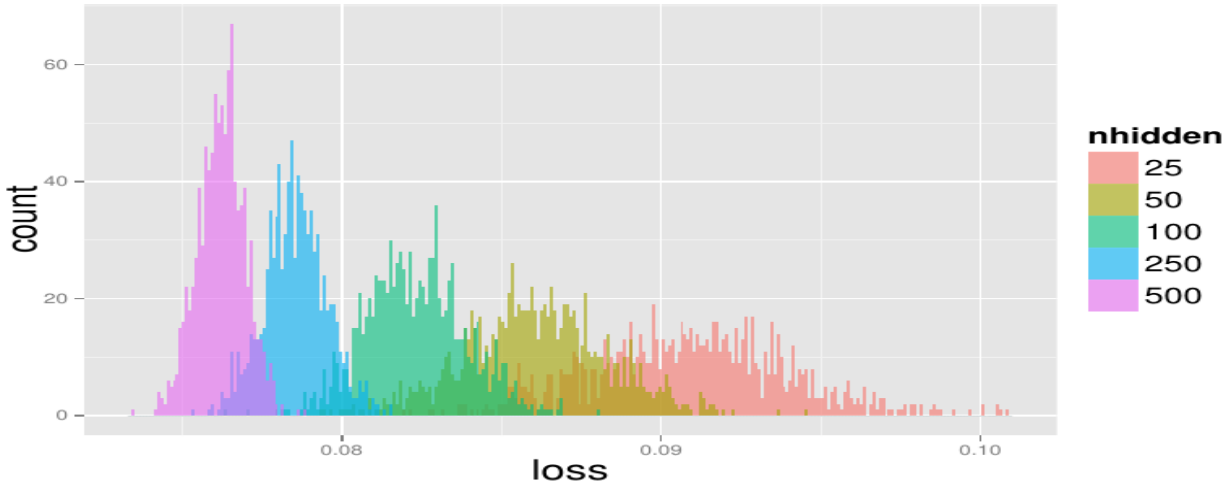

Geometry of the Loss Function

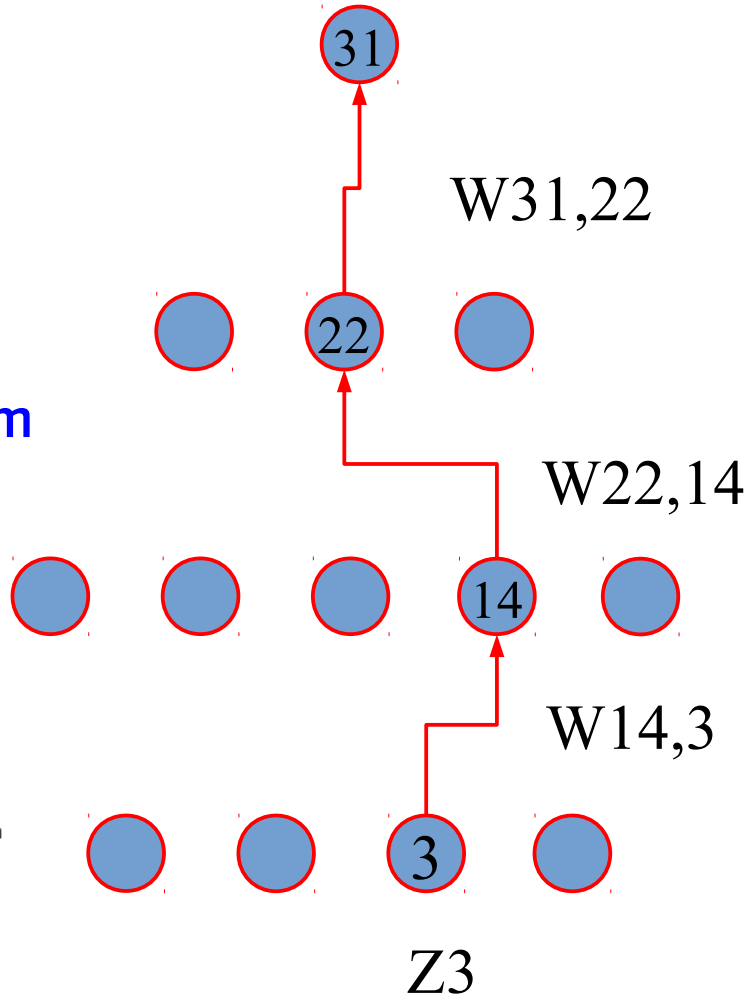
Deep Nets with ReLUs: Objective Function is Piecewise Polynomial

- If we use a hinge loss, delta now depends on label Yk:

$$L(W)=\sum_{P} C_p(X,Y,W)\left(\prod_{(ij)\in P} W_{ij}\right)$$

- Piecewise polynomial in W with random coefficients
- A lot is known about the distribution of critical points of polynomials on the sphere with random (Gaussian) coefficients [Ben Arous et al.]
  - High-order spherical spin glasses
  - Random matrix theory



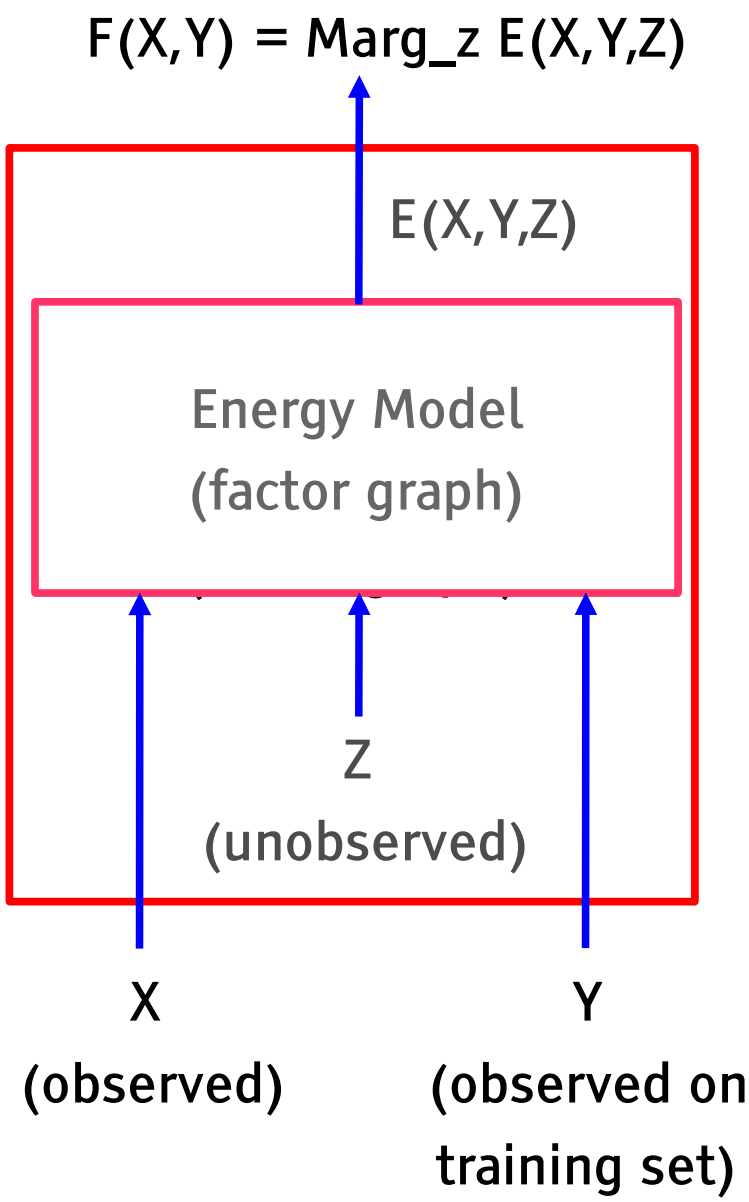



Reasoning
&
Representation Learning

Reasoning as Energy Minimization (structured prediction++)

Y LeCun

- Deep Learning systems can be assembled into energy models AKA factor graphs
  - Energy function is a sum of factors
  - Factors can embed whole deep learning systems
  - X: observed variables (inputs)
  - Z: never observed (latent variables)
  - Y: observed on training set (output variables)
- Inference is energy minimization (MAP) or free energy minimization (marginalization) over Z and Y given an X
  - F(X,Y) = MIN_z E(X,Y,Z)
  - F(X,Y) = -log SUM_z exp[-E(X,Y,Z)]

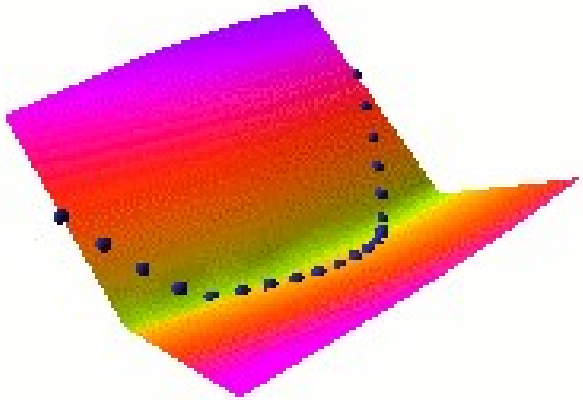


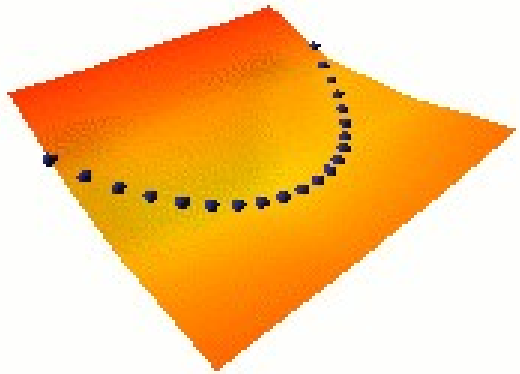
Energy-Based Learning [LeCun et al. 2006]

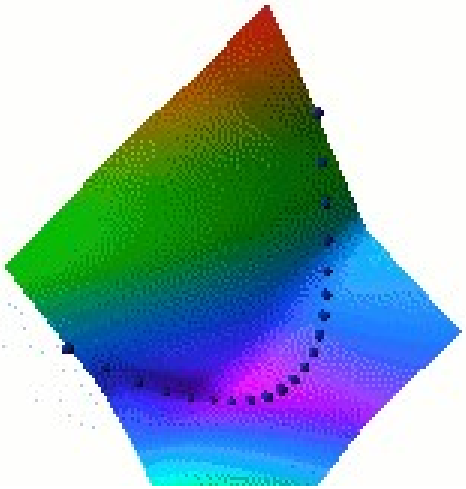
Push down on the energy of desired outputs

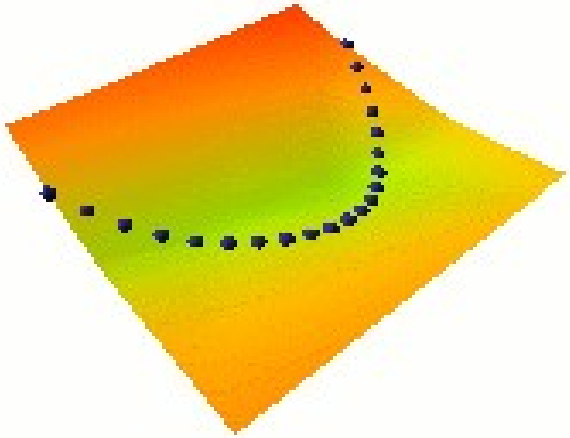
Push up on everything else

[LeCun et al 2006] "A tutorial on energy-based learning"










Stick a CRF on top of a ConvNet

Pose Estimation and Attribute Recovery with ConvNets

Y LeCun

Pose-Aligned Network for Deep Attribute Modeling

[Zhang et al. CVPR 2014] (Facebook AI Research)

(a) Highest scoring results for people wearing glasses.

(b) Highest scoring results for people wearing a hat.

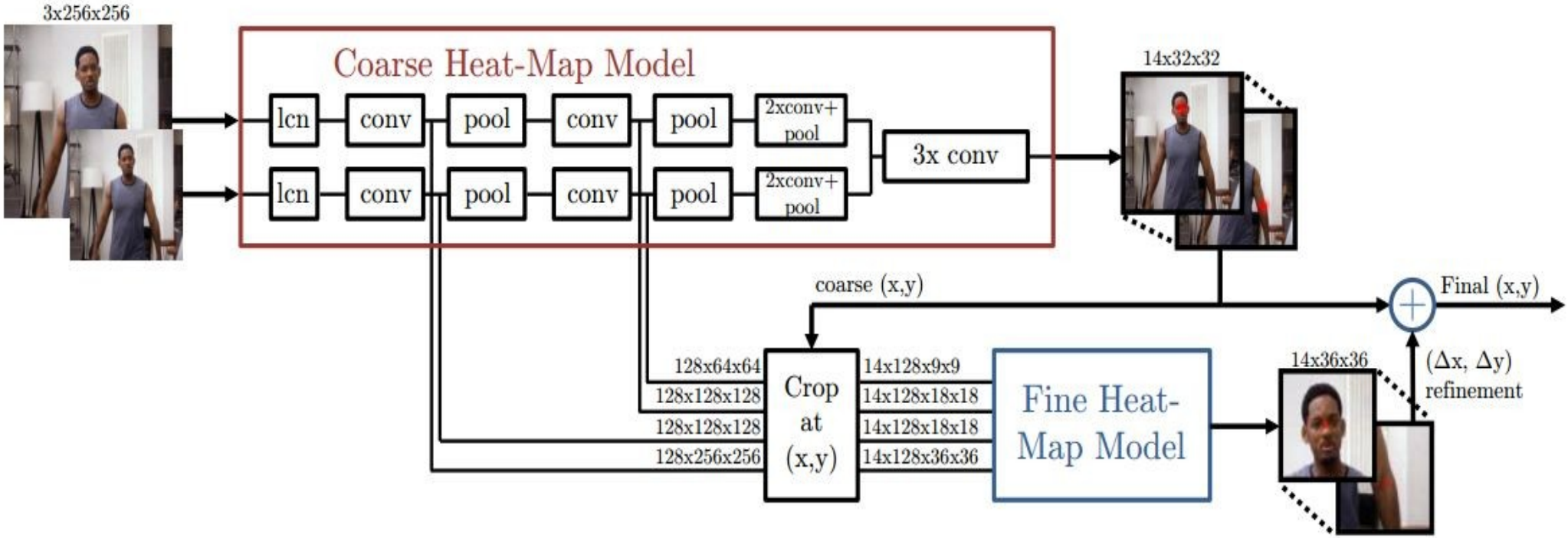
Real-time hand pose recovery

[Tompson et al. Trans. on Graphics 14]

Body pose estimation [Tompson et al. ICLR, 2014]

Person Detection and Pose Estimation

[Tompson, Goroshin, Jain, LeCun, Bregler CVPR 2015]



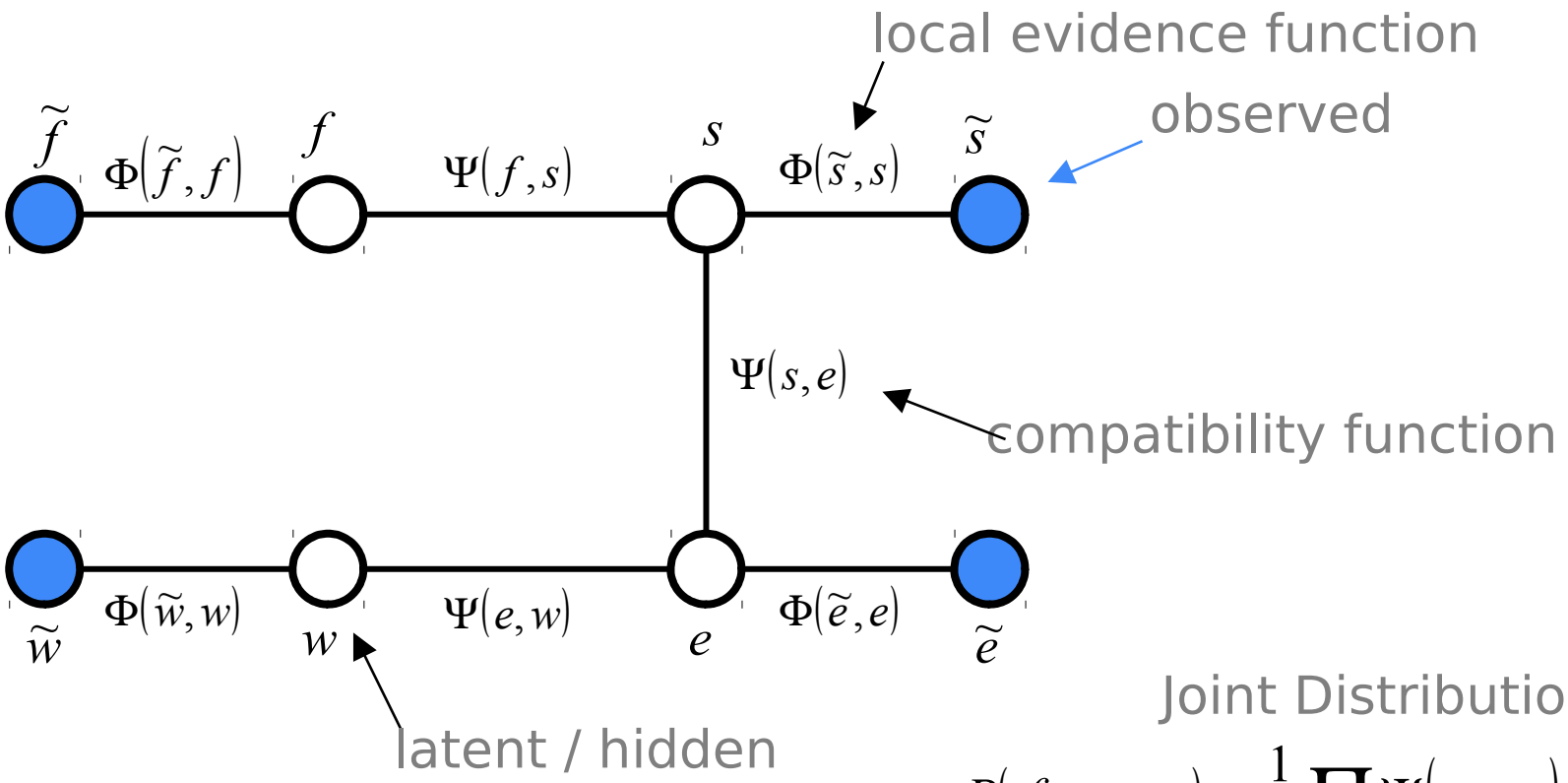
Person Detection and Pose Estimation

Tompson, Goroshin, Jain, LeCun, Bregler arXiv:1411.4280 (2014)

SPATIAL MODEL

Start with a tree graphical model

MRF over spatial locations



Joint Distribution:

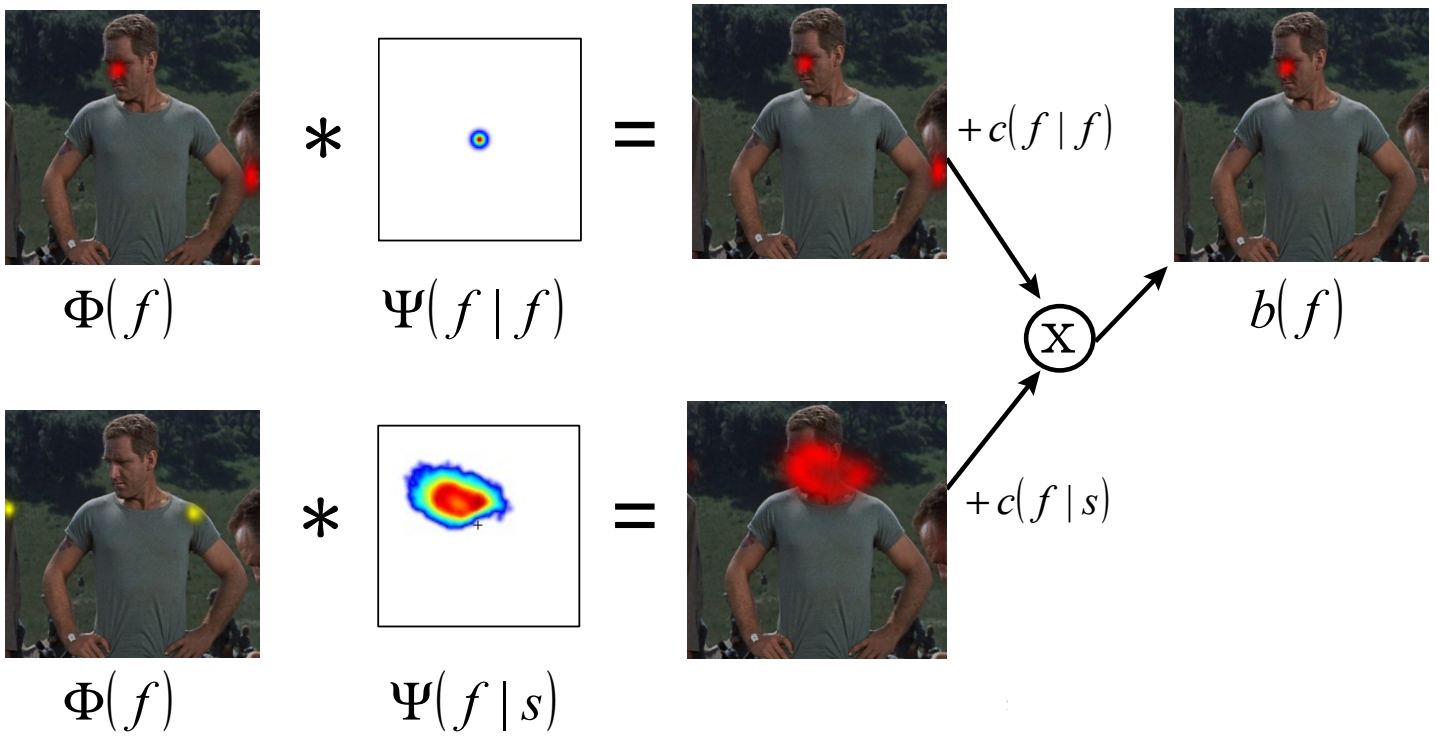
$$P(f,s,e,w)=\frac{1}{Z}\prod_{i,j}\Psi(x_i,x_j)\prod_{i}\Phi(x_i,\tilde{x}_i)$$

SPATIAL MODEL

Start with a tree graphical model

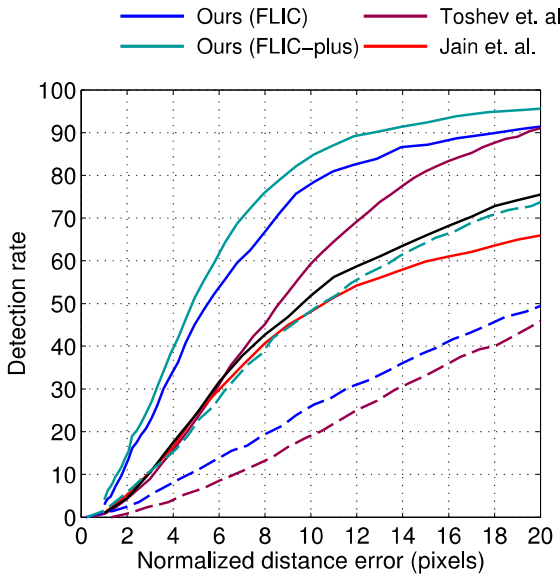
... And approximate it

$$b(f) = \Phi(f) \prod_i \left(\Phi(x_i) * \Psi(f \mid x_i) + c(f \mid x_i) \right)$$

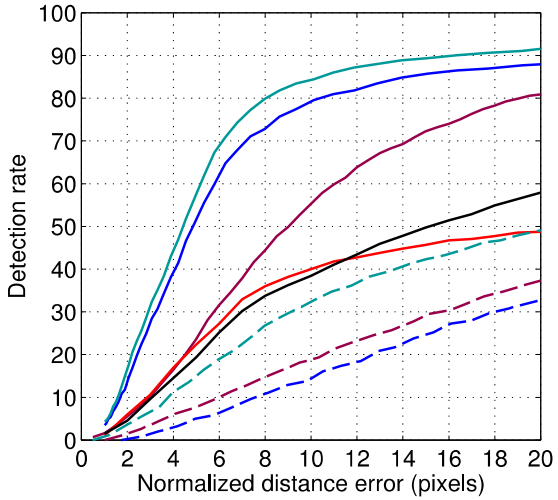


SPATIAL MODEL: RESULTS

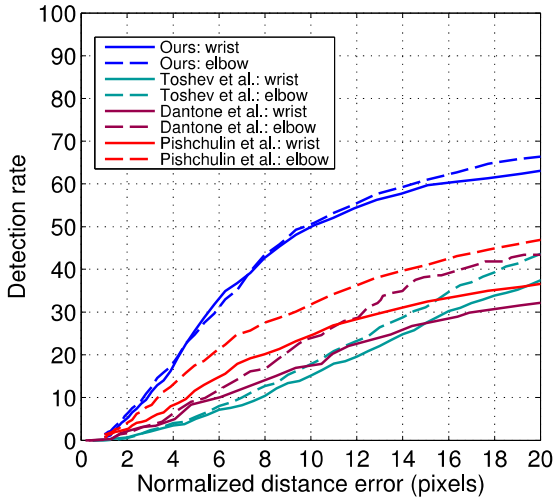
FLIC[1] Elbow



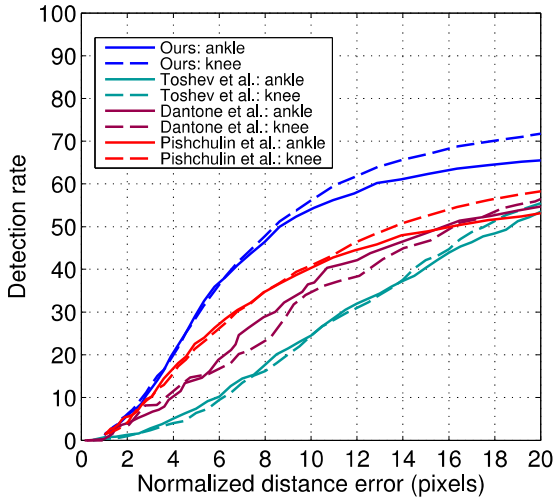
FLIC[1] Wrist



LSP[2] Arms



LSP[1] Legs

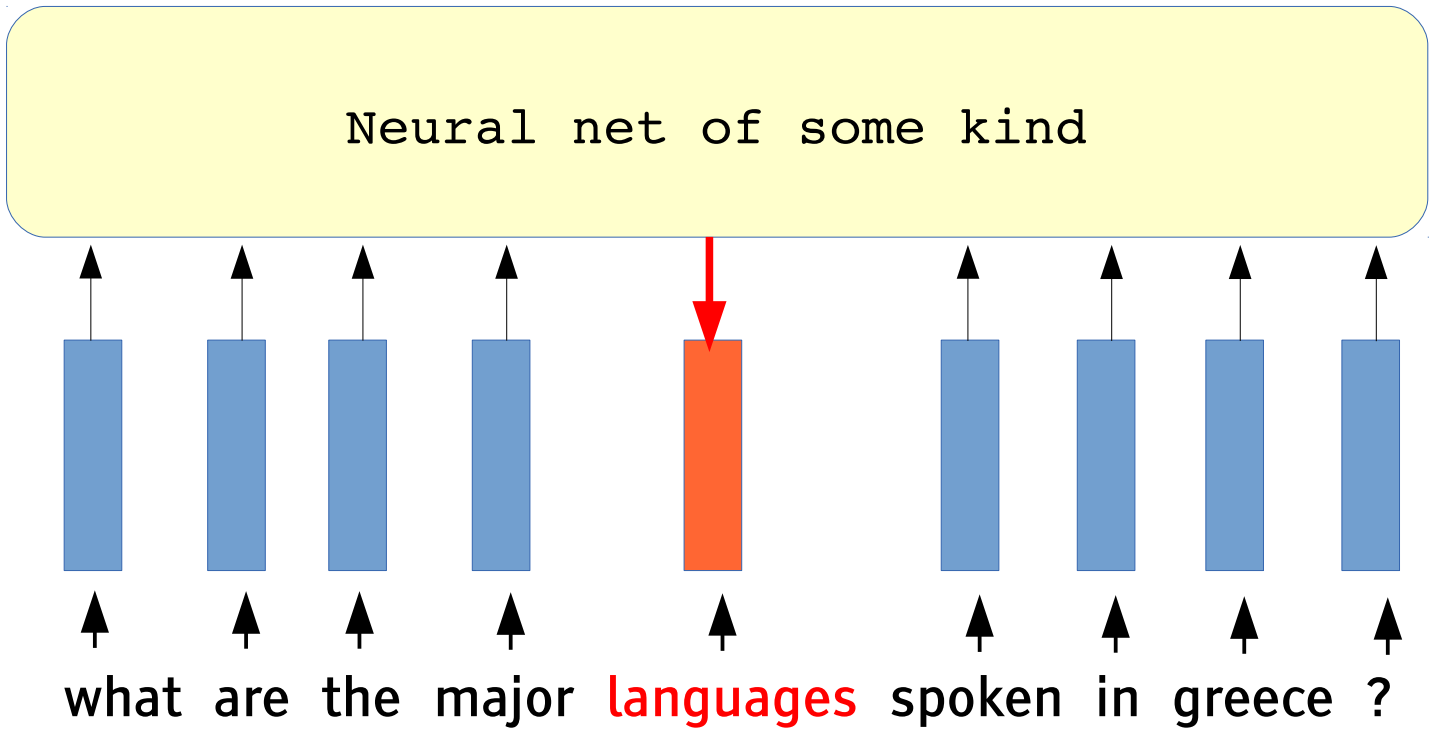


(1) B. Sapp and B. Taskar. MODEC: Multimodel decomposition models for human pose estimation. CVPR'13
(2) S. Johnson and M. Everingham. Learning Effective Human Pose Estimation for Inaccurate Annotation. CVPR'11

Episodic Memory

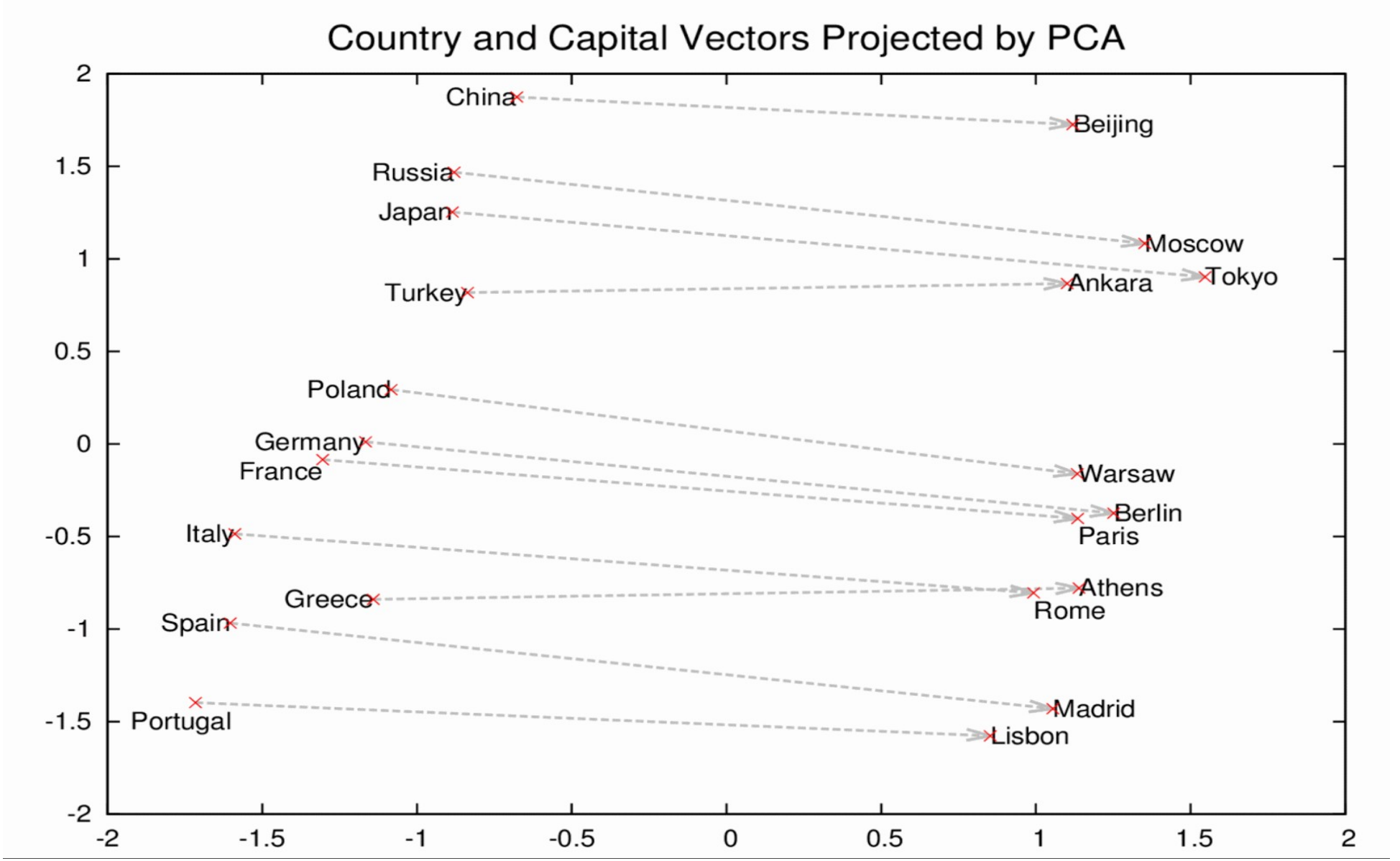
In Natural Language Processing: Word Embedding

- Word Embedding in continuous vector spaces
  - [Bengio 2003][Collobert & Weston 2010]
  - Word2Vec [Mikolov 2011]
  - Predict a word from previous words and/or following words



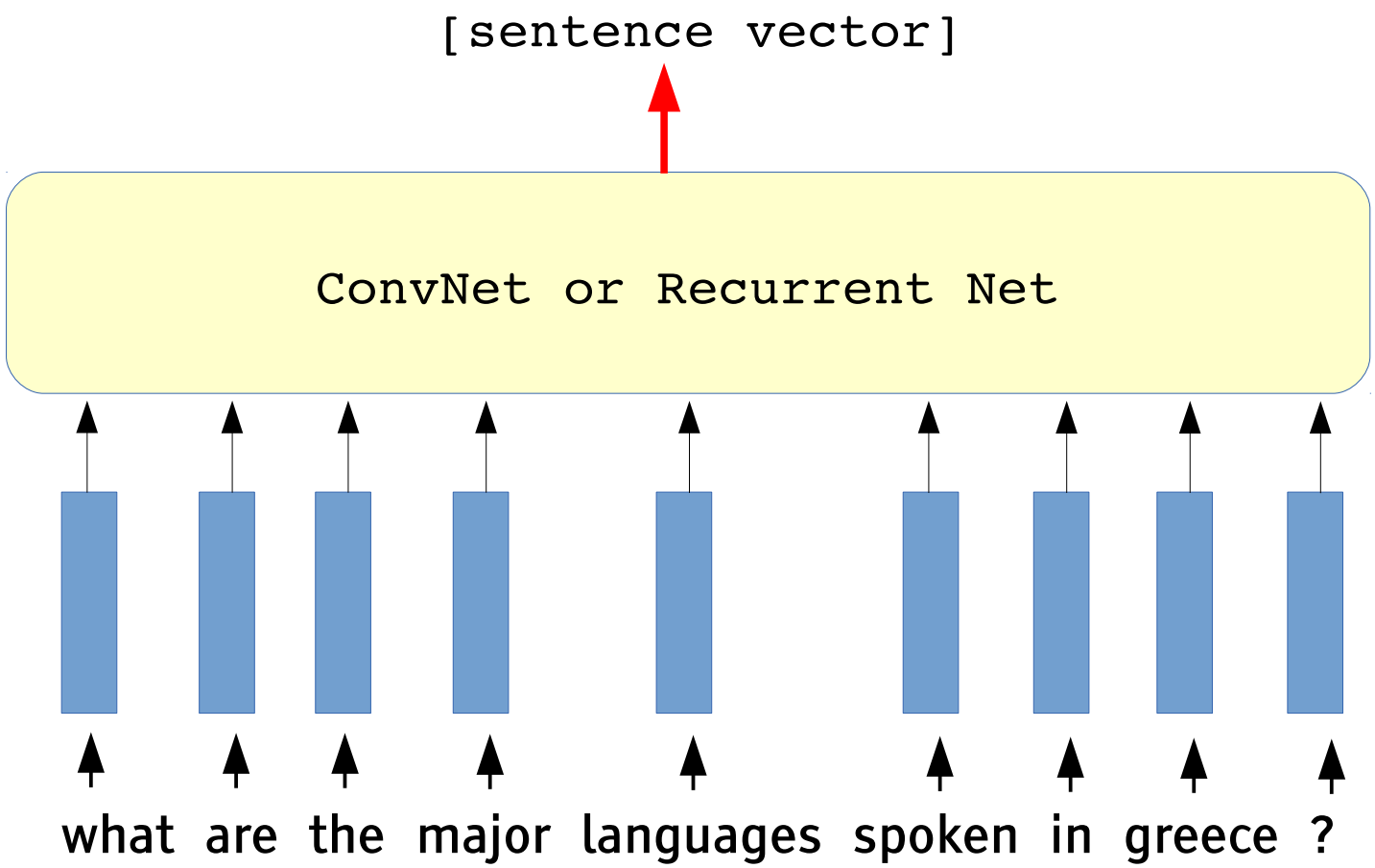
Compositional Semantic Property

Beijing – China + France = Paris

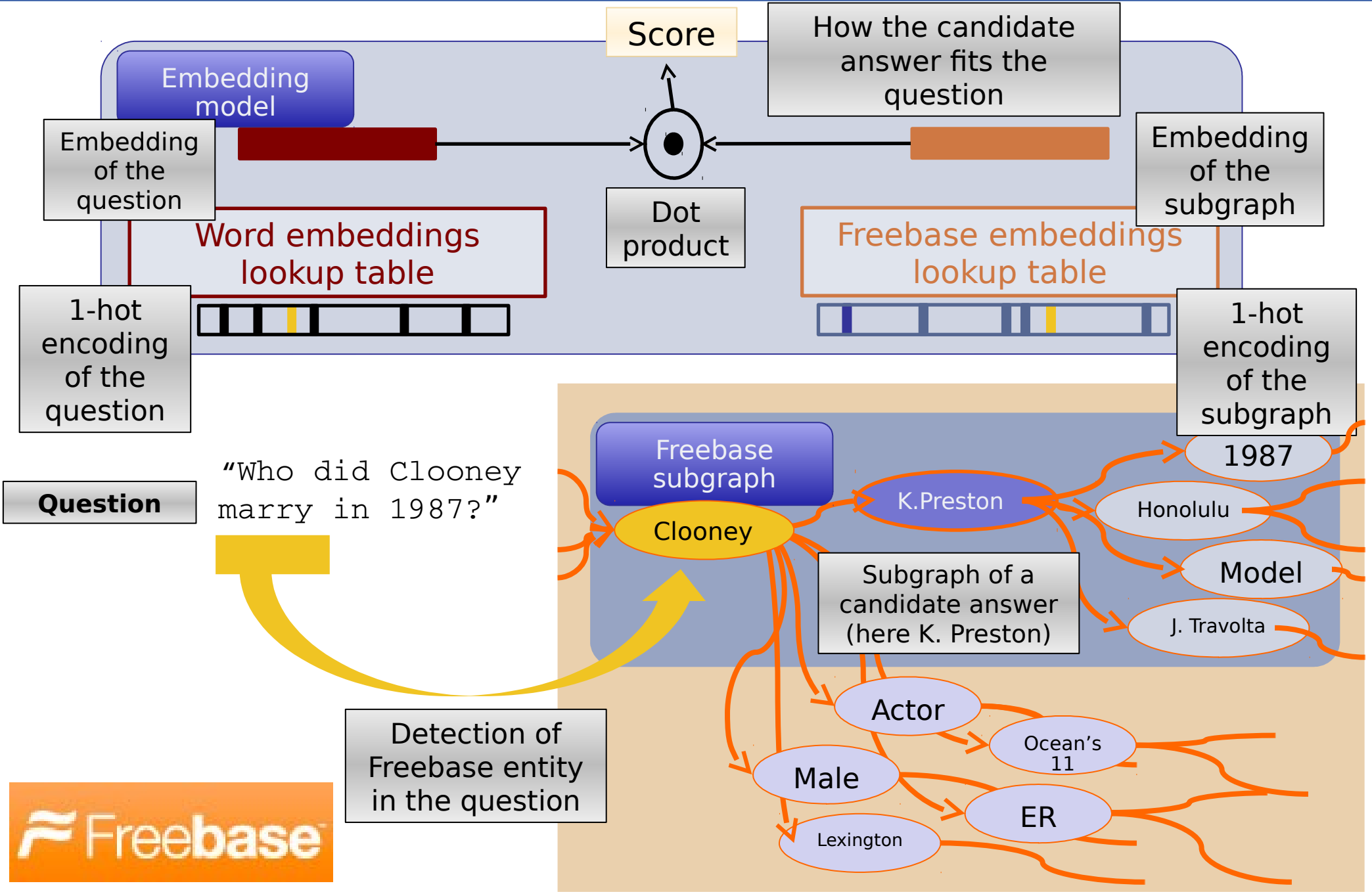


Embedding Text (with convolutional or recurrent nets)

- Embedding sentences into vector spaces
  - Using a convolutional net or a recurrent net.



Question-Answering System



Question-Answering System

what are bigos?
["stew"] ["stew"]
what are dallas cowboys colors?
["navy_blue", "royal_blue", "blue", "white", "silver"] ["blue", "navy_blue", "white", "royal_blue", "silver"]
how is egyptian money called?
["egyptian_pound"] ["egyptian_pound"]
what are fun things to do in sacramento ca?
["sacramento_zoo"] ["raging_waters_sacramento", "sutter_s_fort", "b_street_theatre", "sacramento_zoo", "california_state_capitol_museum", ...]
how are john terry's children called?
["georgie_john_terry", "summer_rose_terry"] ["georgie_john_terry", "summer_rose_terry"]
what are the major languages spoken in greece?
["greek_language", "albanian_language"] ["greek_language", "albanian_language"]
what was laura ingalls wilder famous for?
["writer", "author"] ["writer", "journalist", "teacher", "author"]

NLP: Question-Answering System

who plays sheldon cooper mother on the big bang theory?
["jim_parsons"] ["jim_parsons"]
who does peyton manning play football for?
["denver_broncos"] ["indianapolis_colts", "denver_broncos"]
who did vladimir lenin marry?
["nadezhda_krupskaya"] ["nadezhda_krupskaya"]
where was teddy roosevelt's house?
["new_york_city"] ["manhattan"]
who developed the tcp ip reference model?
["vint_cerf", "robert_e._kahn"] ["computer_scientist", "engineer”]

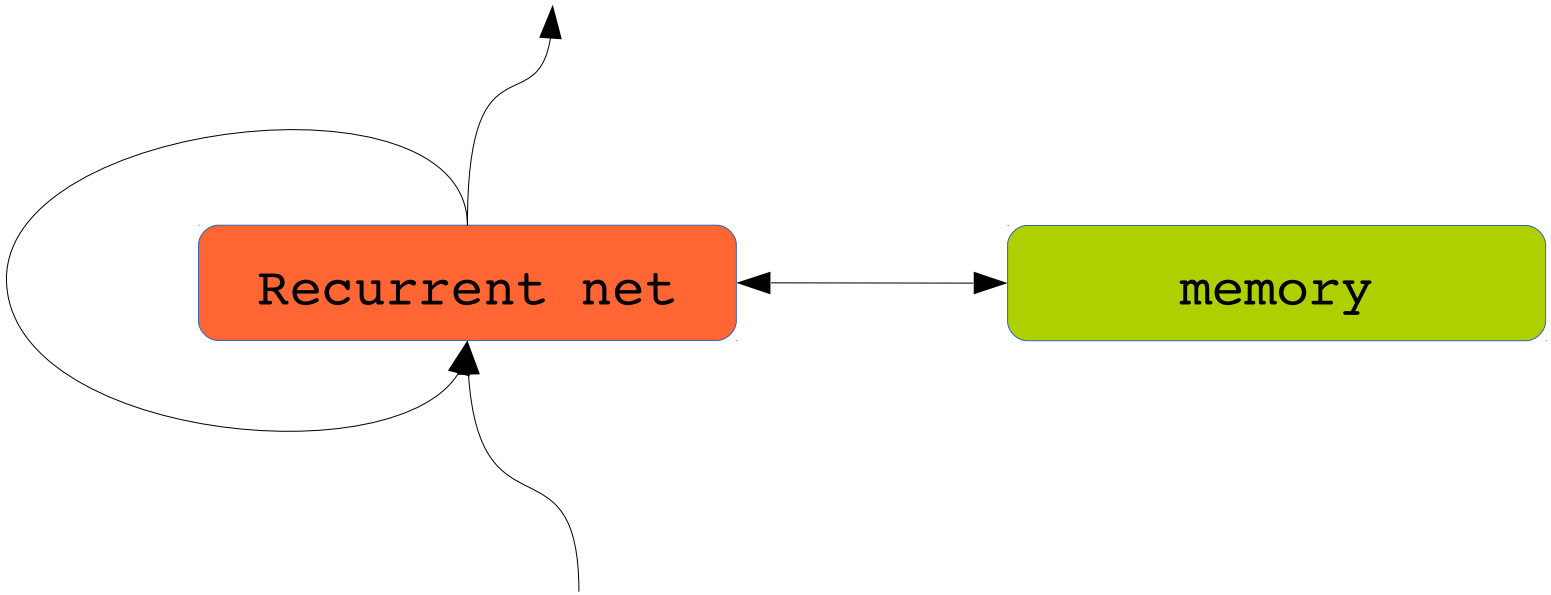
Representing the world with “thought vectors”

- **[Hinton 1831] [OK, OK 1987 ;-]**
- **Every object, concept or “thought” can be represented by a vector**
  - [-0.2, 0.3, -4.2, 5.1, …..] represent the concept “cat”
  - [-0.2, 0.4, -4.0, 5.1, …..] represent the concept “dog”
  - The vectors are similar because cats and dogs have many properties in common
- **Reasoning consists in manipulating thought vectors**
  - Comparing vectors for question answering, information retrieval, content filtering
  - Combining and transforming vectors for reasoning, planning, translating languages
- **Memory stores thought vectors**
  - MemNN (Memory Neural Network) is an example
- **At FAIR we want to “embed the world” in thought vectors**

We call this World2vec

But How can Neural Nets Remember Things?

- Recurrent networks cannot remember things for very long
  - The cortex only remember things for 20 seconds
- We need a "hippocampus" (a separate memory module)
  - LSTM [Hochreiter 1997], registers
  - **Memory networks** [Weston et 2014] (FAIR), associative memory
  - NTM [Graves et al. 2014], "tape".



Memory Network [Weston, Chopra, Bordes 2014]

Add a short-term memory to a network

http://arxiv.org/abs/1410.3916

I: (input feature map) – converts the incoming input to the internal feature representation.
G: (generalization) – updates old memories given the new input.
O: (output feature map) – produces a new output (in the feature representation space), given the new input and the current memory.
R: (response) – converts the output into the response format desired. For example, a textual response or an action.

Method	F1
(Fader et al., 2013) [4]	0.54
(Bordes et al., 2014) [3]	0.73
MemNN	0.71
MemNN (with BoW features)	0.79

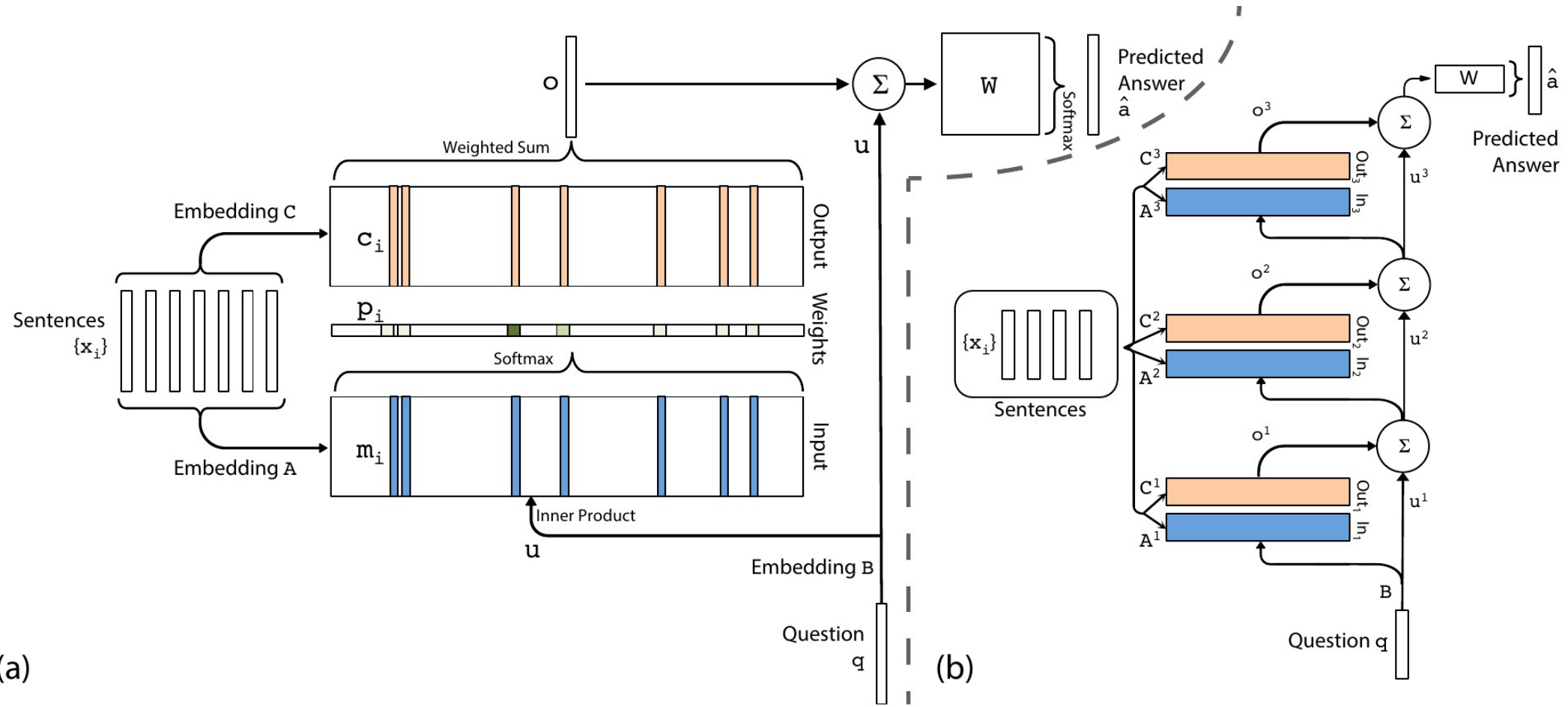
Results on Question Answering Task

Bilbo travelled to the cave.
Gollum dropped the ring there.
Bilbo took the ring.
Bilbo went back to the Shire.
Bilbo left the ring there.
Frodo got the ring.
Frodo journeyed to Mount-Doom.
Frodo dropped the ring there.
Sauron died.
Frodo went back to the Shire.
Bilbo travelled to the Grey-havens.
The End.
Where is the ring? A: Mount-Doom
Where is Bilbo now? A: Grey-havens
Where is Frodo now? A: Shire

Fig. 2. An example story with questions correctly answered by a MemNN. The MemNN was trained on the simulation described in Section 4.2 and had never seen many of these words before, e.g. Bilbo, Frodo and Gollum.

End-to-End Memory Network

- [Sukhbataar, Szlam, Weston, Fergus NIPS 2015, ArXiv:1503.08895]
- Weakly-supervised MemNN: no need to tell which memory location to use.



End-to-End Memory Network on bAbI tasks [Weston 2015]

Y LeCun

```
Sam walks into the kitchen.
Sam picks up an apple.
Sam walks into the bedroom.
Sam drops the apple.
Q: Where is the apple?
A. Bedroom
```

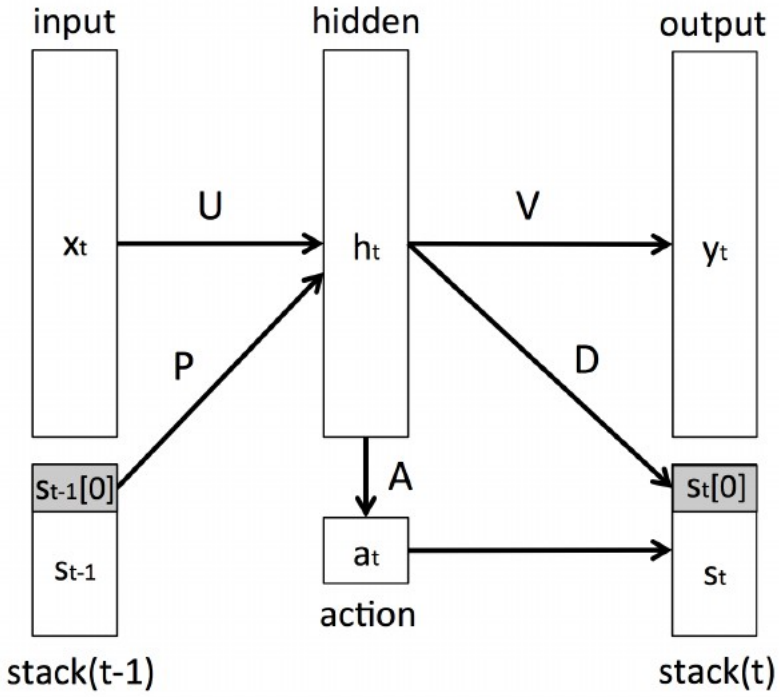
```
Brian is a lion.
Julius is a lion.
Julius is white.
Bernhard is green.
Q: What color is Brian?
A. White
```

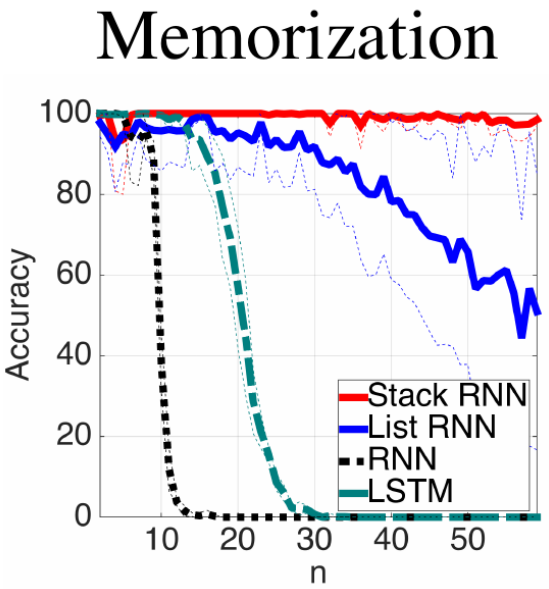
```
Mary journeyed to the den.
Mary went back to the kitchen.
John journeyed to the bedroom.
Mary discarded the milk.
Q: Where was the milk before the den?
A. Hallway
```

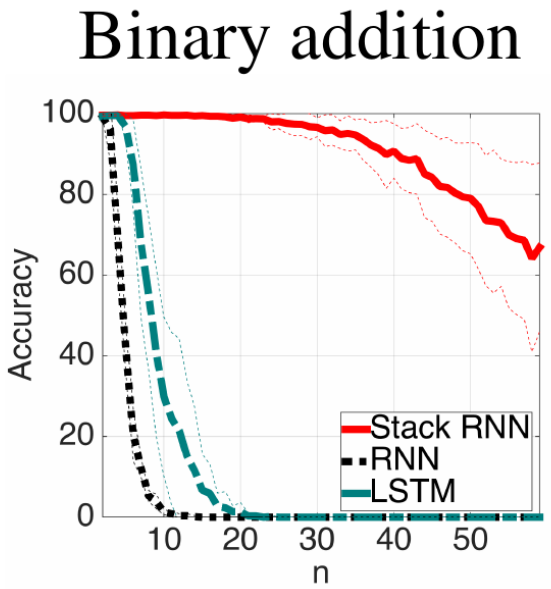
	Baseline			MemN2N								
Task	Strongly Supervised MemNN [21]	LSTM [21]	MemNN WSH	BoW	PE	PE LS	PE LS RN	1 hop PE LS joint	2 hops PE LS joint	3 hops PE LS joint	PE LS RN joint	PE LS LW joint
1: 1 supporting fact	0.0	50.0	0.1	0.6	0.1	0.2	0.0	0.8	0.0	0.1	0.0	0.1
2: 2 supporting facts	0.0	80.0	42.8	17.6	21.6	12.8	8.3	62.0	15.6	14.0	11.4	18.8
3: 3 supporting facts	0.0	80.0	76.4	71.0	64.2	58.8	40.3	76.9	31.6	33.1	21.9	31.7
4: 2 argument relations	0.0	39.0	40.3	32.0	3.8	11.6	2.8	22.8	2.2	5.7	13.4	17.5
5: 3 argument relations	2.0	30.0	16.3	18.3	14.1	15.7	13.1	11.0	13.4	14.8	14.4	12.9
6: yes/no questions	0.0	52.0	51.0	8.7	7.9	8.7	7.6	7.2	2.3	3.3	2.8	2.0
7: counting	15.0	51.0	36.1	23.5	21.6	20.3	17.3	15.9	25.4	17.9	18.3	10.1
8: lists/sets	9.0	55.0	37.8	11.4	12.6	12.7	10.0	13.2	11.7	10.1	9.3	6.1
9: simple negation	0.0	36.0	35.9	21.1	23.3	17.0	13.2	5.1	2.0	3.1	1.9	1.5
10: indefinite knowledge	2.0	56.0	68.7	22.8	17.4	18.6	15.1	10.6	5.0	6.6	6.5	2.6
11: basic coreference	0.0	38.0	30.0	4.1	4.3	0.0	0.9	8.4	1.2	0.9	0.3	3.3
12: conjunction	0.0	26.0	10.1	0.3	0.3	0.1	0.2	0.4	0.0	0.3	0.1	0.0
13: compound coreference	0.0	6.0	19.7	10.5	9.9	0.3	0.4	6.3	0.2	1.4	0.2	0.5
14: time reasoning	1.0	73.0	18.3	1.3	1.8	2.0	1.7	36.9	8.1	8.2	6.9	2.0
15: basic deduction	0.0	79.0	64.8	24.3	0.0	0.0	0.0	46.4	0.5	0.0	0.0	1.8
16: basic induction	0.0	77.0	50.5	52.0	52.1	1.6	1.3	47.4	51.3	3.5	2.7	51.0
17: positional reasoning	35.0	49.0	50.9	45.4	50.1	49.0	51.0	44.4	41.2	44.5	40.4	42.6
18: size reasoning	5.0	48.0	51.3	48.1	13.6	10.1	11.1	9.6	10.3	9.2	9.4	9.2
19: path finding	64.0	92.0	100.0	89.7	87.4	85.6	82.8	90.7	89.9	90.2	88.0	90.6
20: agent's motivation	0.0	9.0	3.6	0.1	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.2
Mean error (%)	6.7	51.3	40.2	25.1	20.3	16.3	13.9	25.8	15.6	13.3	12.4	15.2
Failed tasks (err. $>$ 5%)	4	20	18	15	13	12	11	17	11	11	11	10
On 10k training data												
Mean error (%)	3.2	36.4	39.2	15.4	9.4	7.2	6.6	24.5	10.9	7.9	7.5	11.0
Failed tasks (err. $>$ 5%)	2	16	17	9	6	4	4	16	7	6	6	6

Stack-Augmented RNN: learning "algorithmic" sequences

- [Joulin & Mikolov, ArXiv:1503.01007]







method	$a^n b^n$	$a^n b^n c^n$	$a^n b^n c^n d^n$	$a^n b^{2n}$	$a^n b^m c^{n+m}$
RNN	25%	23.3%	13.3%	23.3%	33.3%
LSTM	100%	100%	68.3%	75%	100%
List RNN 40+5	100%	33.3%	100%	100%	100%
Stack RNN 40+10	100%	100%	100%	100%	43.3%
Stack RNN 40+10 + rounding	100%	100%	100%	100%	100%


Unsupervised Learning

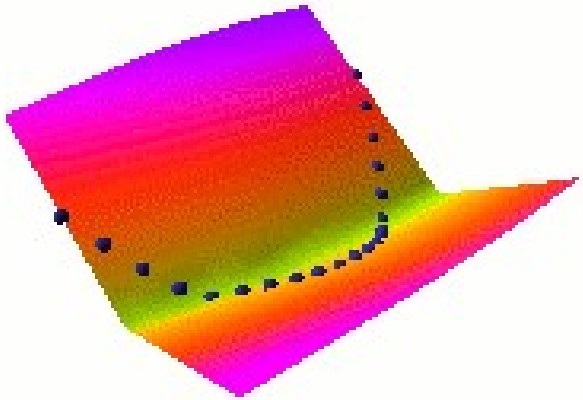
How do we do unspervised learning?

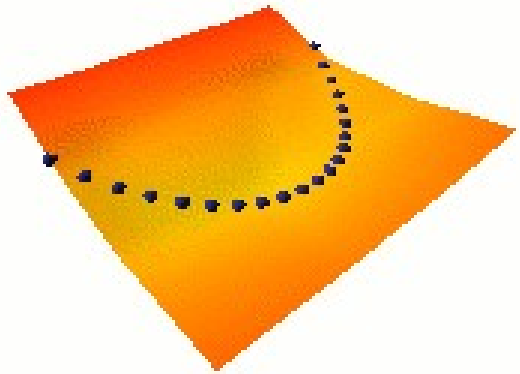
- **Unsupervised learning can be based on reconstruction or prediction**
  - Reconstruction is just prediction with delay 0
  - It seems qualitatively different from prediction, but it's not
  - Because a good way to predict the future is to copy the present
- **How do we deal with the fact that the world is only partially predictable?**
  - Model the future as a **distribution**: **hahaha, good luck with that!**
  - Modeling a distribution works when the world is discrete (e.g. words)
  - It doesn't work when the world is continuous or compositional.
- **Even if the world were noiseless and quasi deterministic, our current methods would fail**
  - Imagine the world follows a trajectory on a low-dimensional manifold with a few discrete choices at certain places.
  - Our current methods would still fail.
  - Our failure to do unsupervised learning has nothing to do with our inability to model **probability distributions** in high-dim spaces.
  - **It has to do with our inability to capture complex regularities.**
  - **I'm allergic to sampling anyway.**

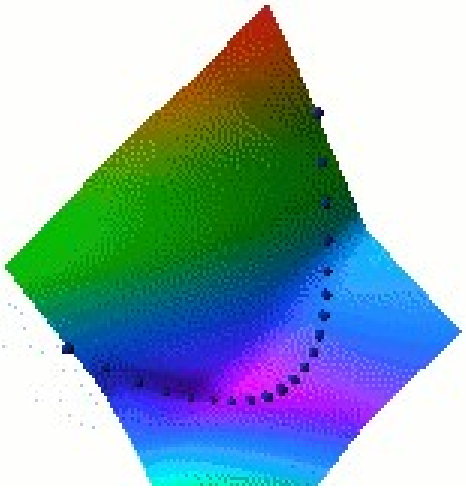
Energy-Based Unsupervised Learning

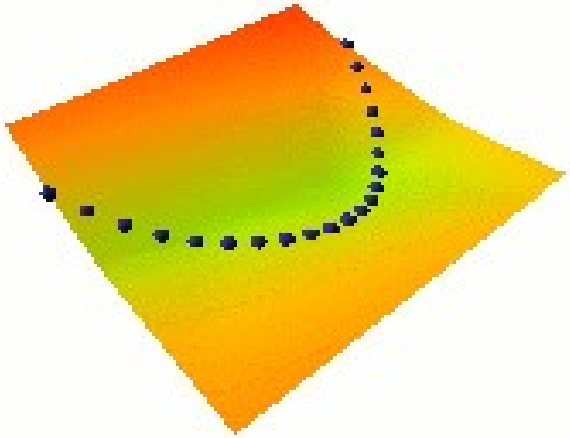
Push down on the energy of desired outputs

Push up on everything else









Eight Strategies to Shape the Energy Function

- 1. build the machine so that the volume of low energy stuff is constant
  - PCA, K-means, GMM, square ICA
- 2. push down of the energy of data points, push up everywhere else
  - Max likelihood (needs tractable partition function)
- 3. push down of the energy of data points, push up on chosen locations
  - contrastive divergence, Ratio Matching, Noise Contrastive Estimation, Minimum Probability Flow
- 4. minimize the gradient and maximize the curvature around data points
  - score matching
- 5. train a dynamical system so that the dynamics goes to the manifold
  - denoising auto-encoder
- 6. use a regularizer that limits the volume of space that has low energy
  - Sparse coding, sparse auto-encoder, PSD
- 7. if E(Y) = ||Y - G(Y)||^2, make G(Y) as "constant" as possible.
  - Contracting auto-encoder, saturating auto-encoder
- 8. Adversarial training: generator tries to fool real/synthetic classifier.

#1: constant volume of low energy
Energy surface for PCA and K-means

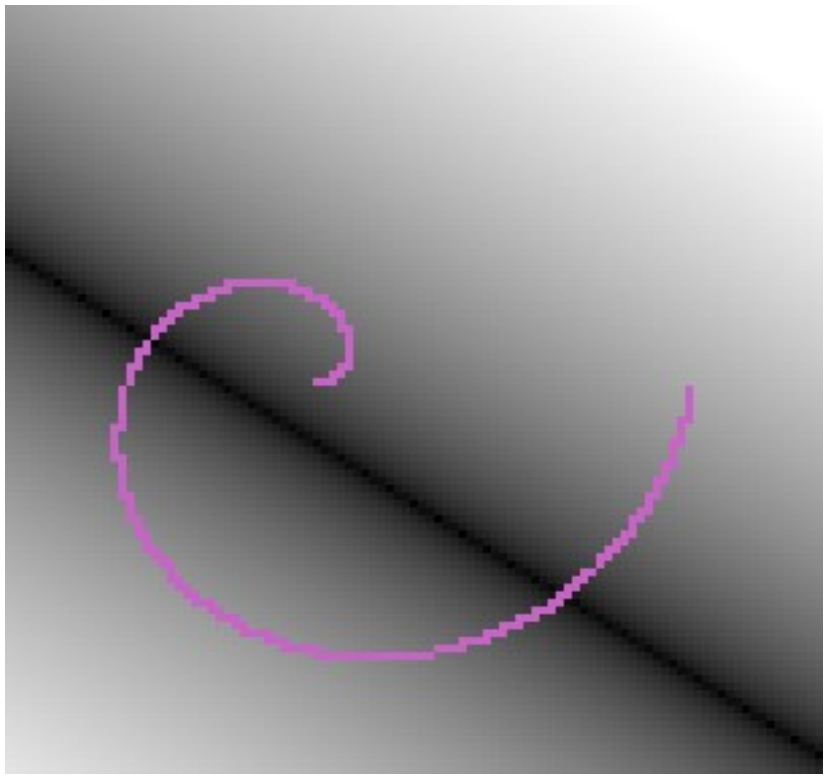
- 1. build the machine so that the volume of low energy stuff is constant
  - PCA, K-means, GMM, square ICA...

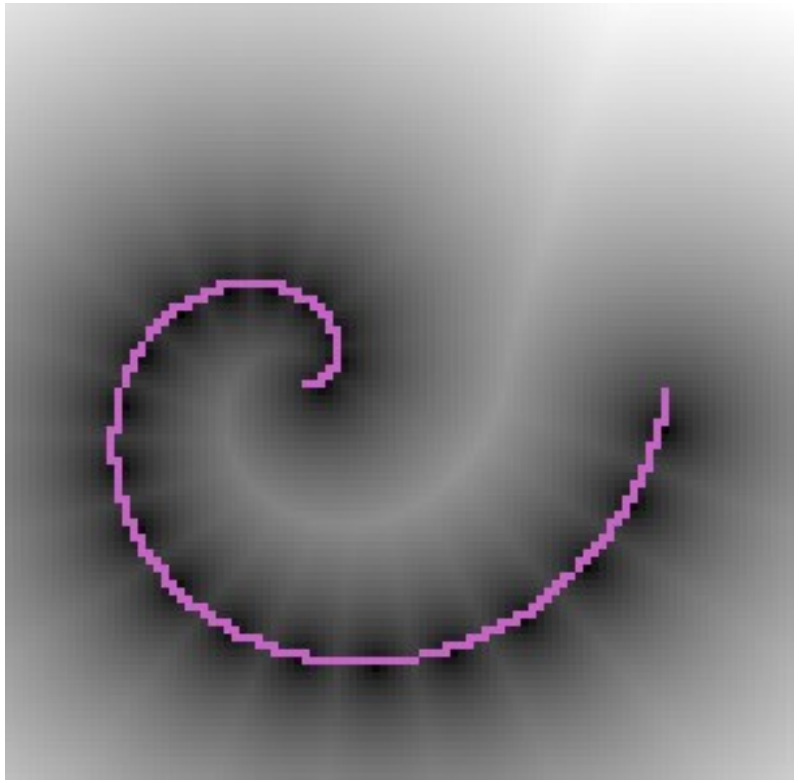
PCA

$$E(Y) = \|W^T WY - Y\|^2$$

K-Means,
Z constrained to 1-of-K code

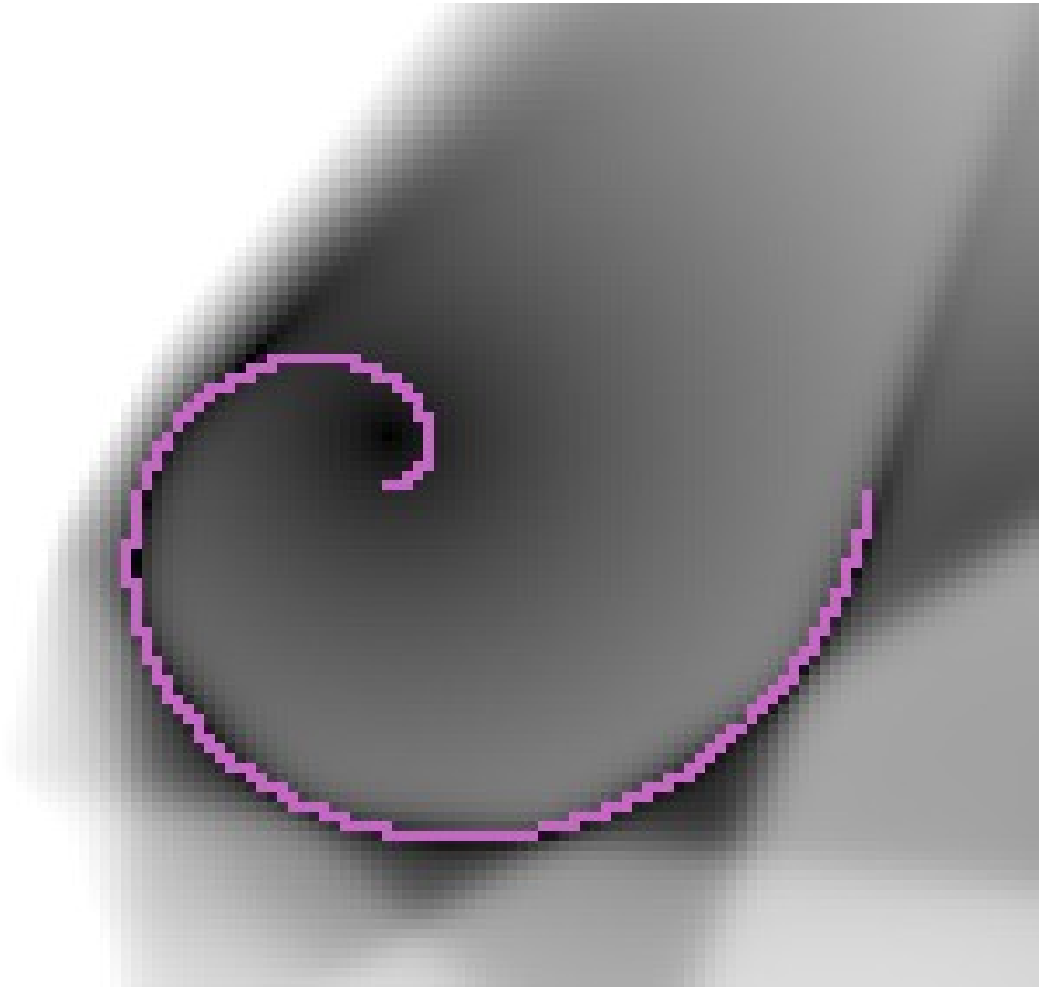
$$E(Y) = min_z \sum_i \|Y - W_i Z_i\|^2$$



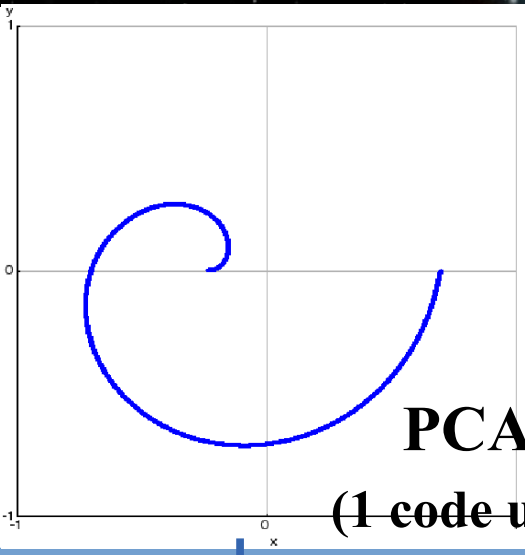


#6. use a regularizer that limits the volume of space that has low energy

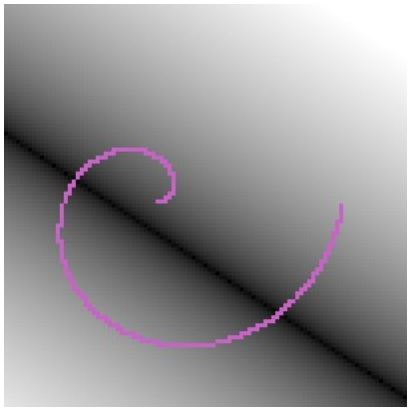
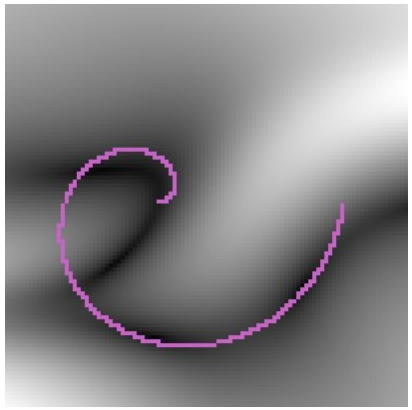
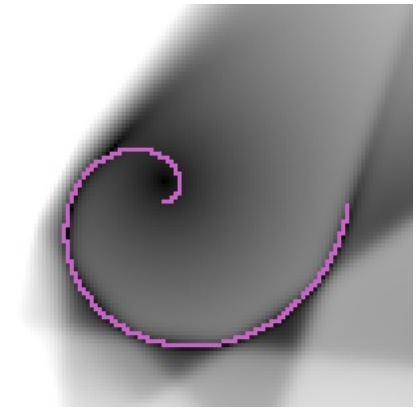
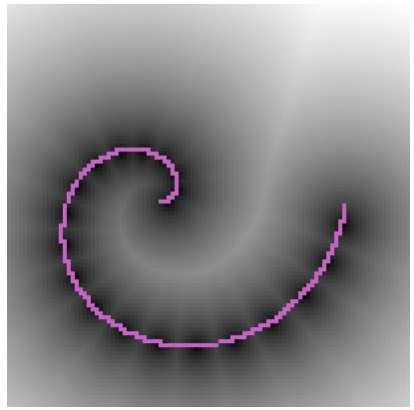
- Sparse coding, sparse auto-encoder, Predictive Sparse Decomposition



Energy Functions of Various Methods



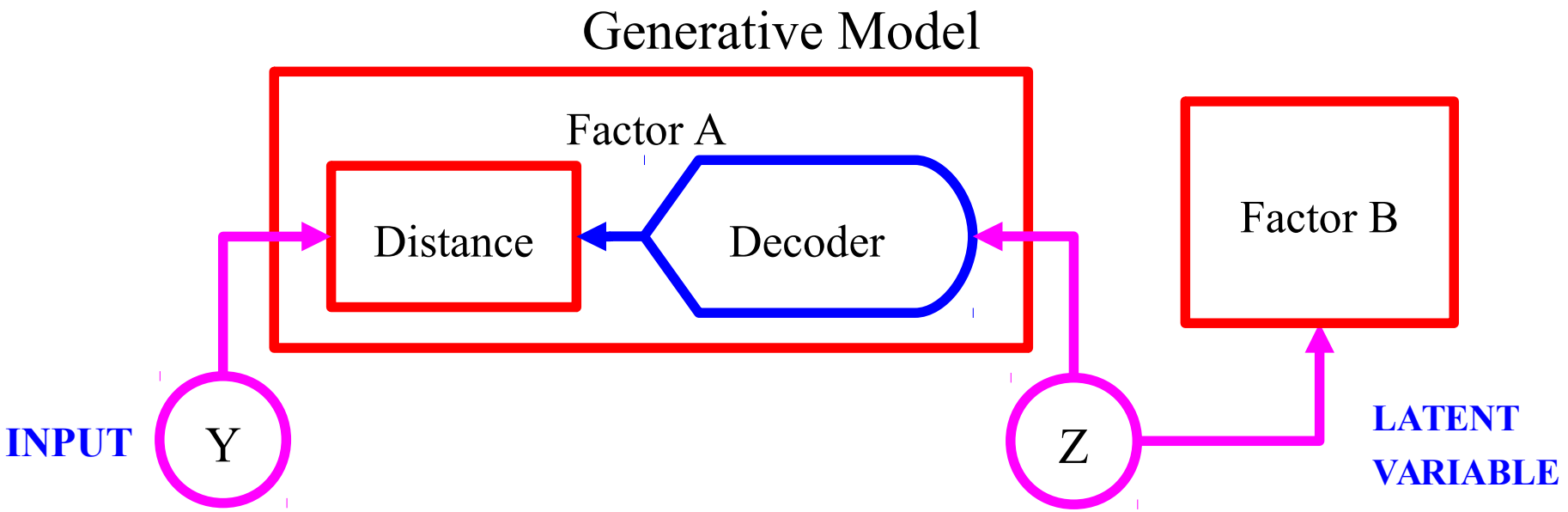
- 2 dimensional toy dataset: spiral
- Visualizing energy surface
  - (black = low, white = high)

	PCA (1 code unit)	autoencoder (1 code unit)	sparse coding (20 code units)	K-Means (20 code units)
encoder	$W'Y$	$\sigma(W_e Y)$	$\sigma(W_e Z)$	–
decoder	WZ	$W_d Z$	$W_d Z$	WZ
energy	$\|\|Y-WZ\|\|^2$	$\|\|Y-WZ\|\|^2$	$\|\|Y-WZ\|\|^2$	$\|\|Y-WZ\|\|^2$
loss	$F(Y)$	$F(Y)$	$F(Y)$	$F(Y)$
pull-up	dimens.	dimens.	sparsity	1-of-N code
				


Learning to Infer

How to Speed Up Inference in a Generative Model?

- Factor Graph with an asymmetric factor
- Inference Z → Y is easy
  - Run Z through deterministic decoder, and sample Y
- Inference Y → Z is hard, particularly if Decoder function is many-to-one
  - MAP: minimize sum of two factors with respect to Z
  - Z* = argmin_z Distance[Decoder(Z), Y] + FactorB(Z)
- Examples: K-Means (1of K), Sparse Coding (sparse), Factor Analysis

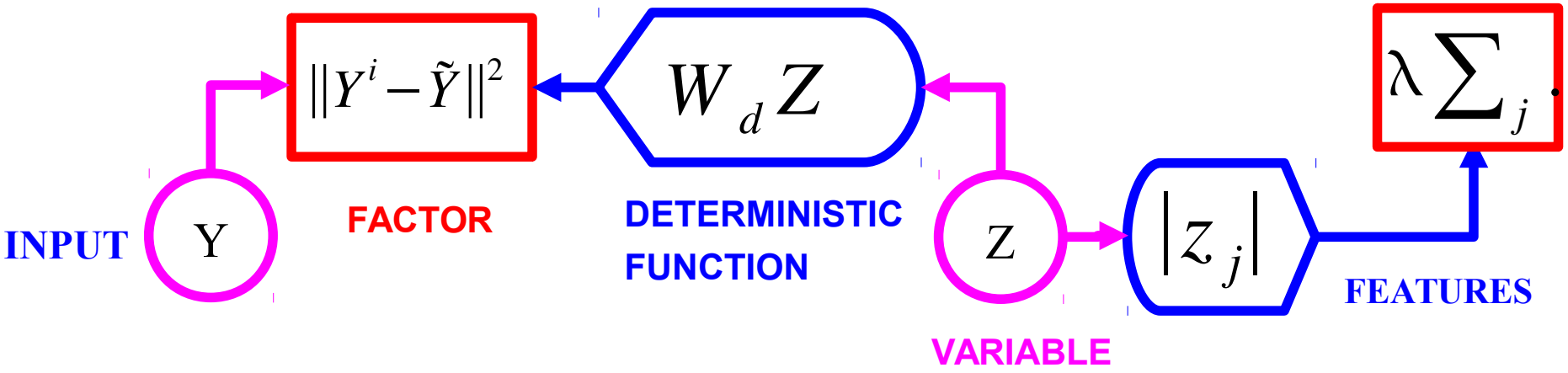


Sparse Modeling: Sparse Coding + Dictionary Learning

[Olshausen & Field 1997]

- Sparse linear reconstruction
- Energy = reconstruction_error + code_prediction_error + code_sparsity

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

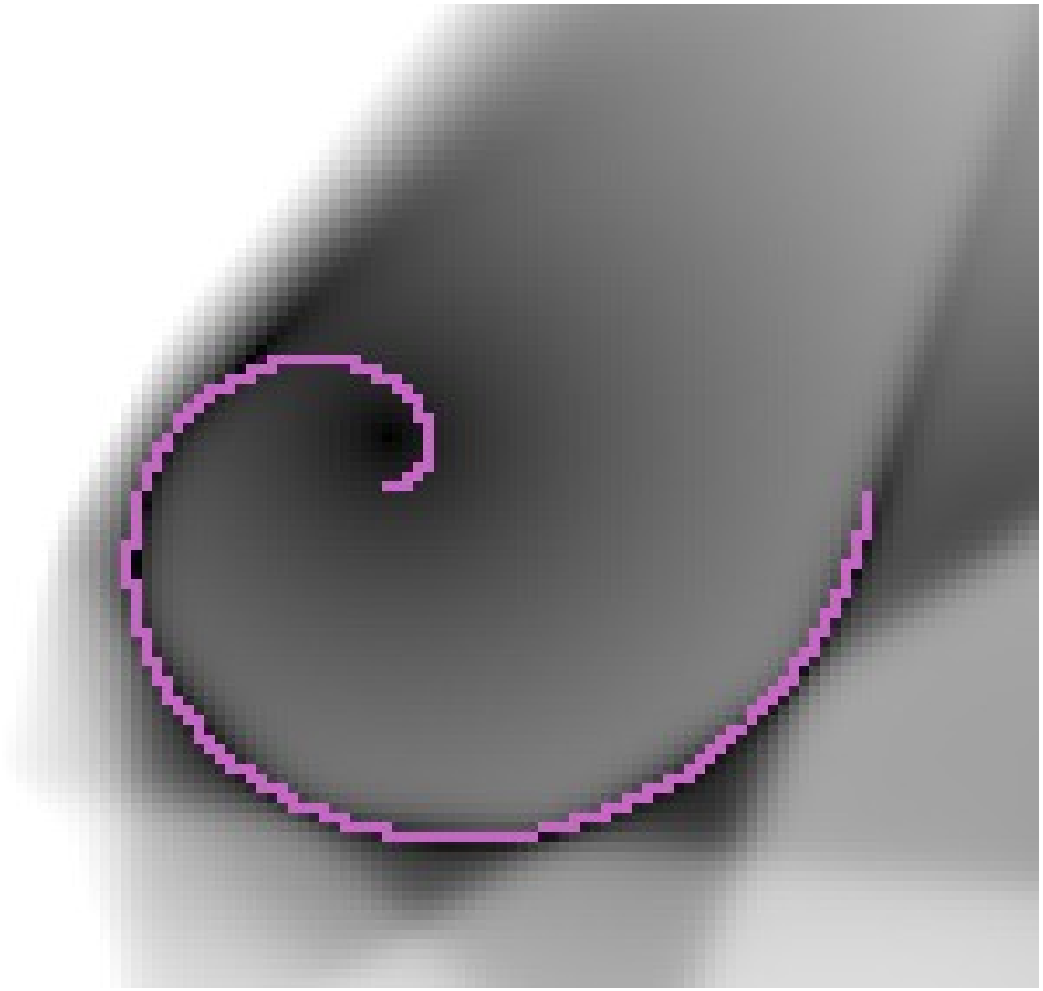


- Inference is expensive: ISTA/FISTA, CGIHT, coordinate descent....

$$Y \rightarrow \hat{Z} = argmin_Z E(Y, Z)$$

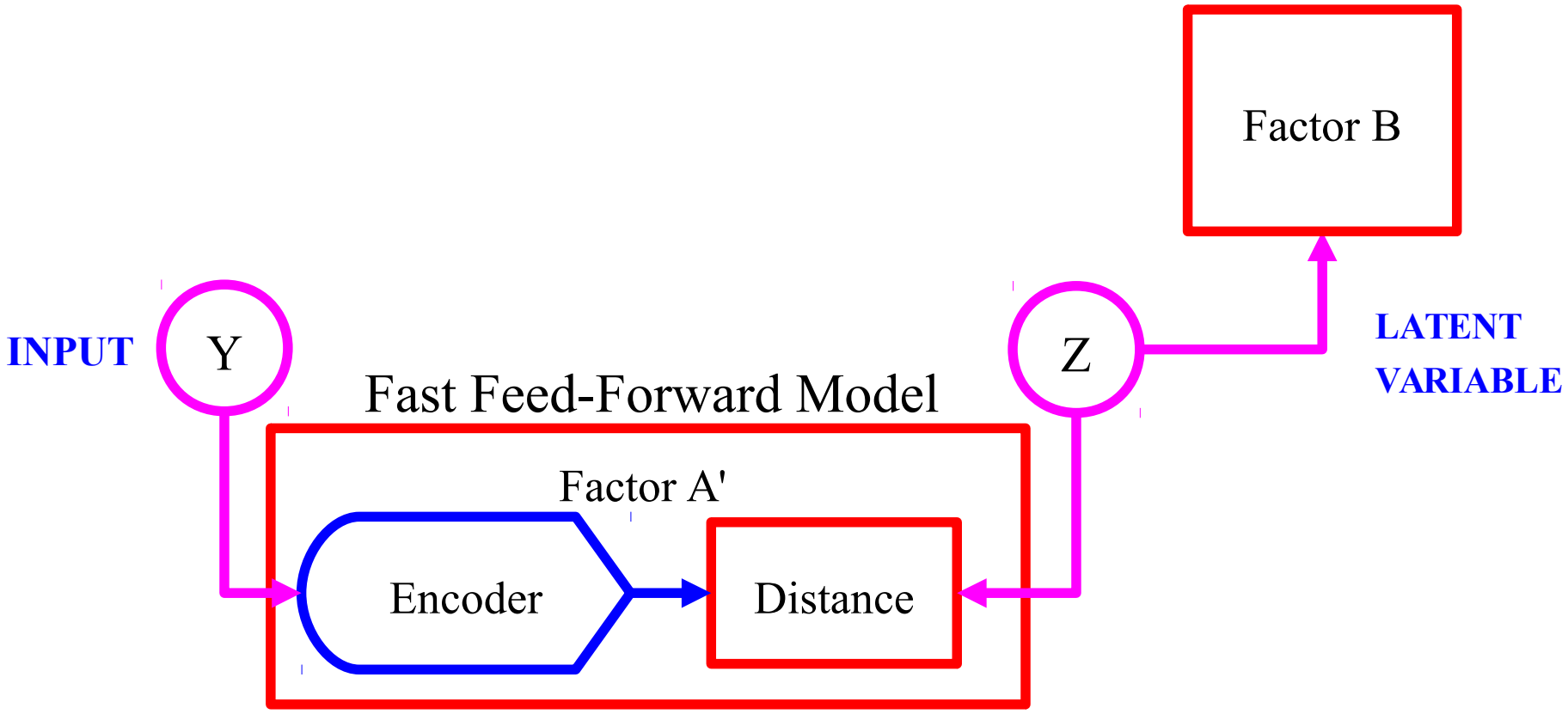
#6. use a regularizer that limits the volume of space that has low energy

- Sparse coding, sparse auto-encoder, Predictive Saprse Decomposition



Encoder Architecture

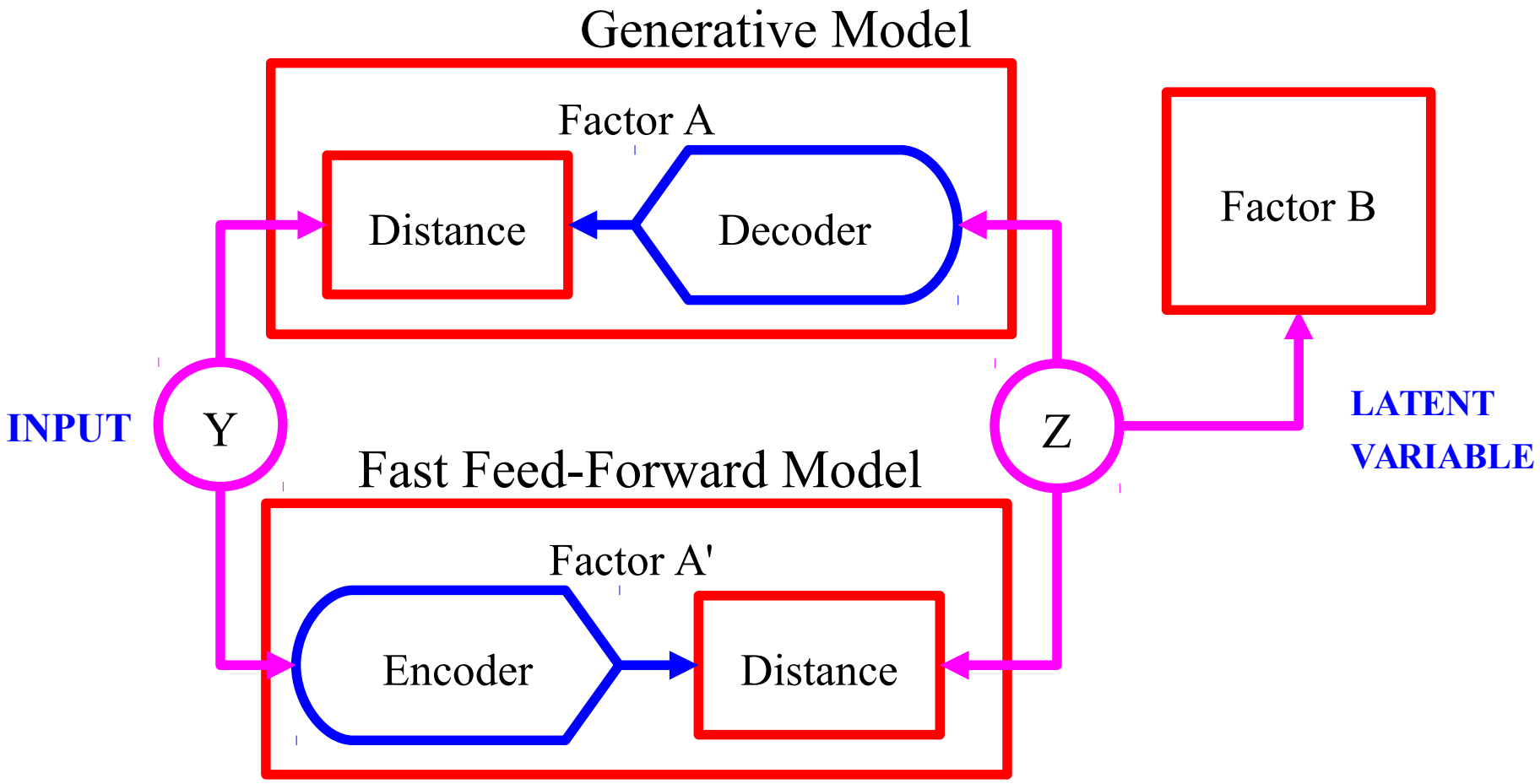
- Examples: most ICA models, Product of Experts



Encoder-Decoder Architecture

[Kavukcuoglu, Ranzato, LeCun, rejected by every conference, 2008-2009]

- Train a "simple" feed-forward function to predict the result of a complex optimization on the data points of interest



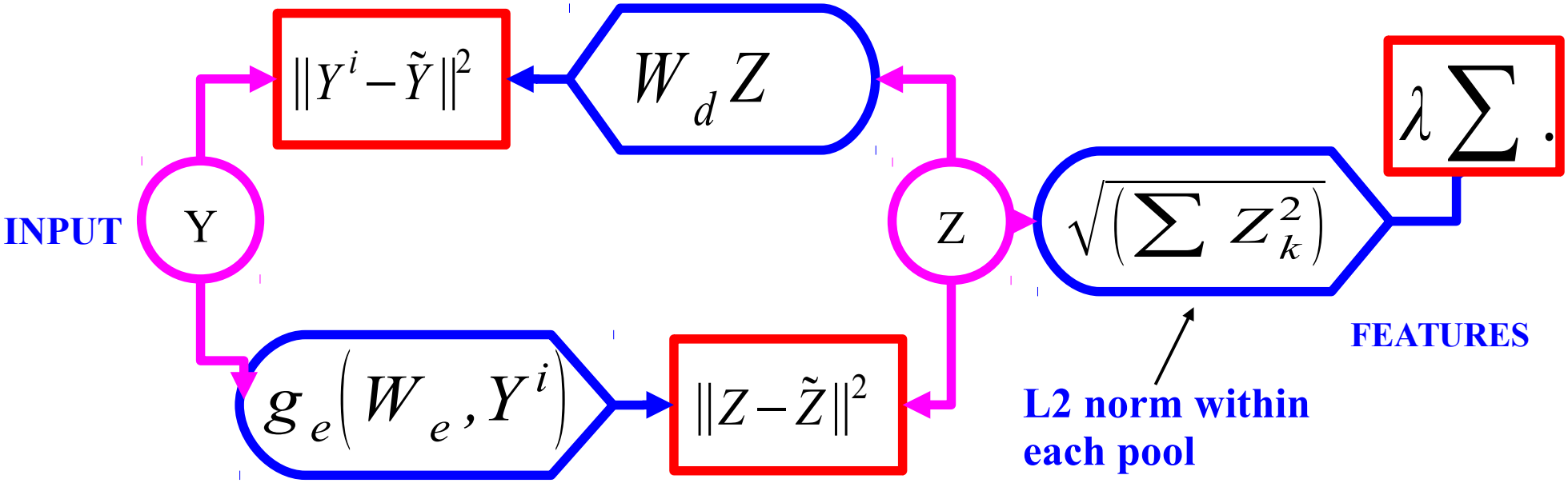
- 1. Find optimal Zi for all Yi; 2. Train Encoder to predict Zi from Yi


Unsupervised Learning:
Invariant Features

Learning Invariant Features with L2 Group Sparsity

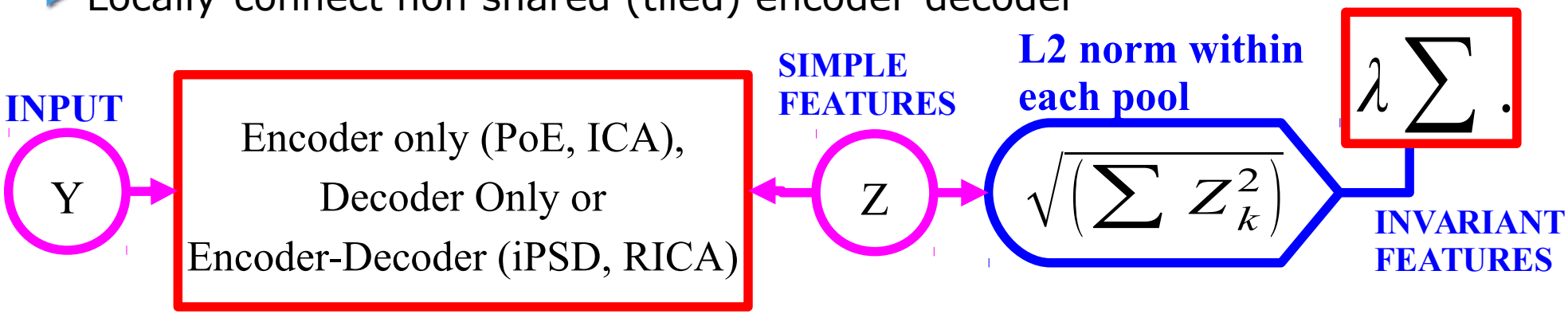
- Unsupervised PSD ignores the spatial pooling step.
- Could we devise a similar method that learns the pooling layer as well?
- Idea [Hyvarinen & Hoyer 2001]: group sparsity on pools of features
  - Minimum number of pools must be non-zero
  - Number of features that are on within a pool doesn't matter
  - Pools tend to regroup similar features

$$E(Y,Z) = \|Y - W_d Z\|^2 + \|Z - g_e(W_e, Y)\|^2 + \sum_j \sqrt{\sum_{k \in P_j} Z_k^2}$$



Learning Invariant Features with L2 Group Sparsity

- Idea: features are pooled in group.
  - Sparsity: sum over groups of L2 norm of activity in group.
- [Hyvärinen Hoyer 2001]: "subspace ICA"
  - decoder only, square
- [Welling, Hinton, Osindero NIPS 2002]: pooled product of experts
  - encoder only, overcomplete, log student-T penalty on L2 pooling
- [Kavukcuoglu, Ranzato, Fergus LeCun, CVPR 2010]: Invariant PSD
  - encoder-decoder (like PSD), overcomplete, L2 pooling
- [Le et al. NIPS 2011]: Reconstruction ICA
  - Same as [Kavukcuoglu 2010] with linear encoder and tied decoder
- [Gregor & LeCun arXiv:1006:0448, 2010] [Le et al. ICML 2012]
  - Locally-connect non shared (tiled) encoder-decoder



Groups are local in a 2D Topographic Map

- The filters arrange themselves spontaneously so that similar filters enter the same pool.
- The pooling units can be seen as complex cells
- Outputs of pooling units are invariant to local transformations of the input
  - For some it's translations, for others rotations, or other transformations.

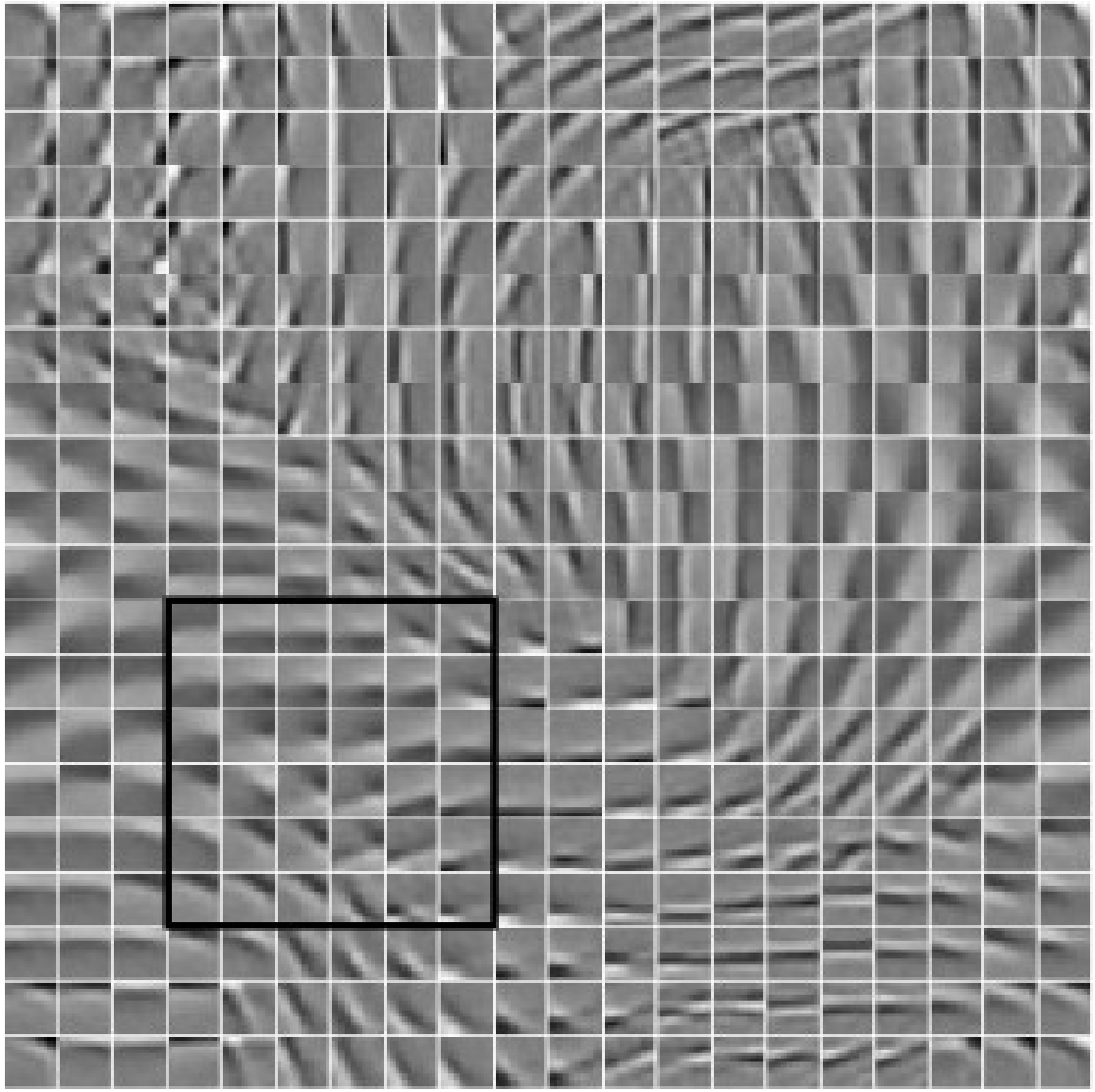


Image-level training, local filters but no weight sharing

- Training on 115x115 images. Kernels are 15x15 (not shared across space!)
  - [Gregor & LeCun 2010]
  - Local receptive fields
  - No shared weights
  - 4x overcomplete
  - L2 pooling
  - Group sparsity over pools

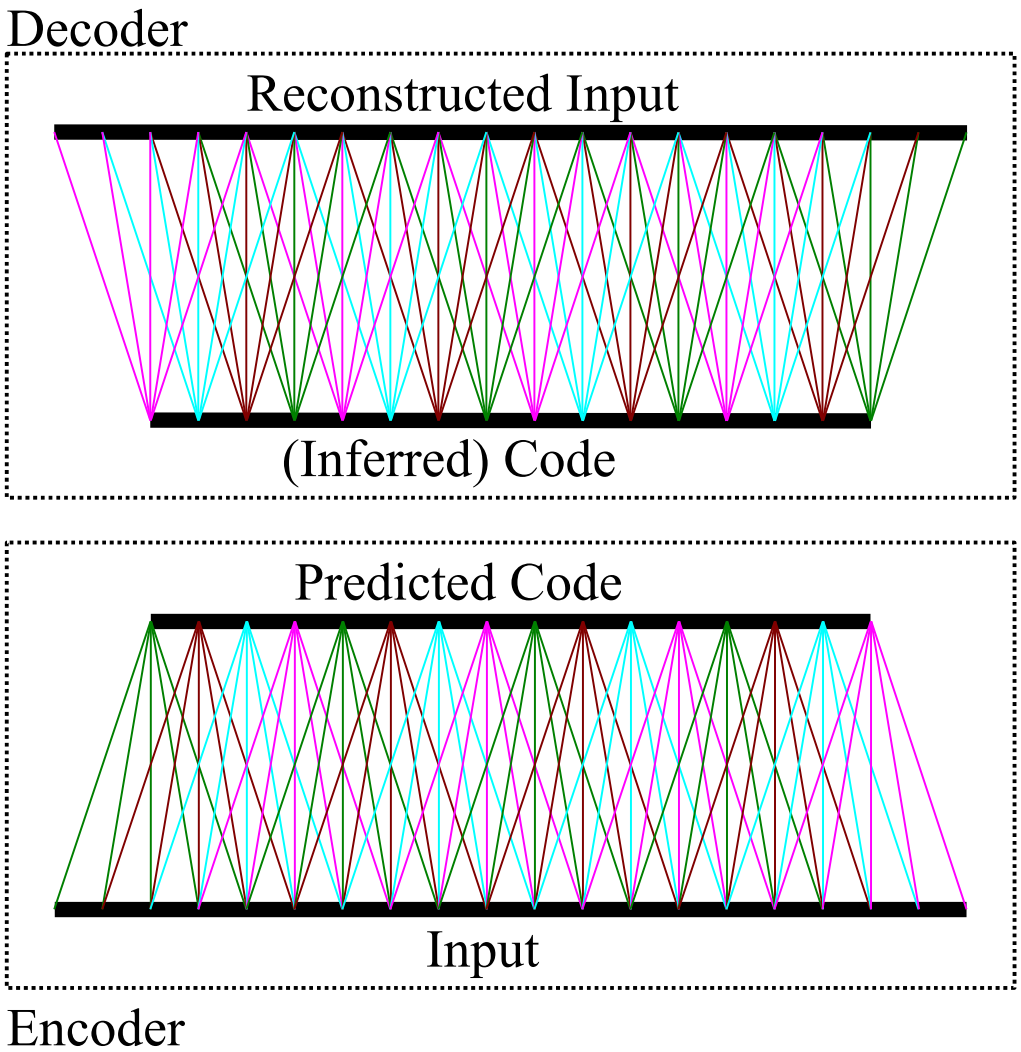
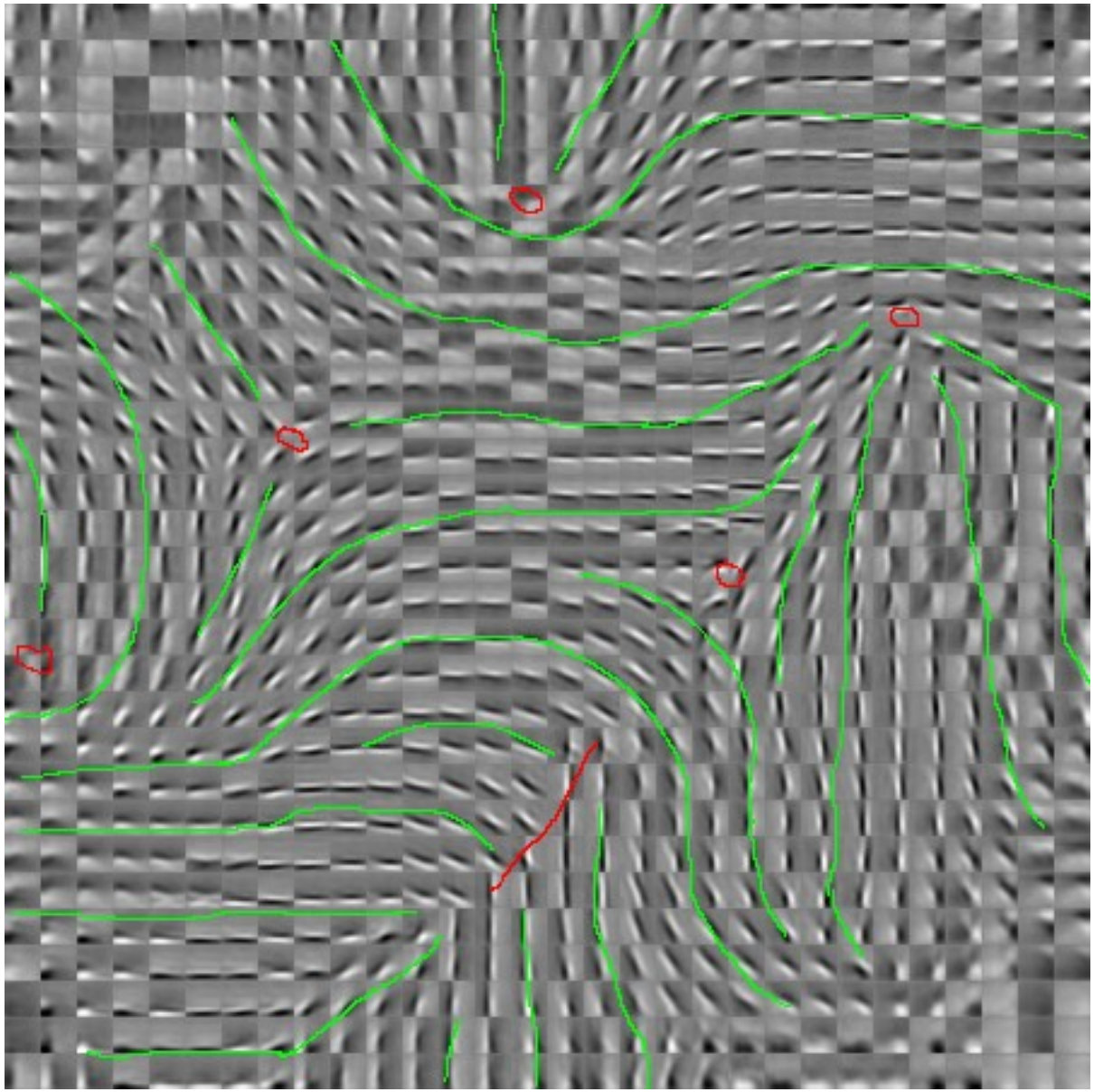


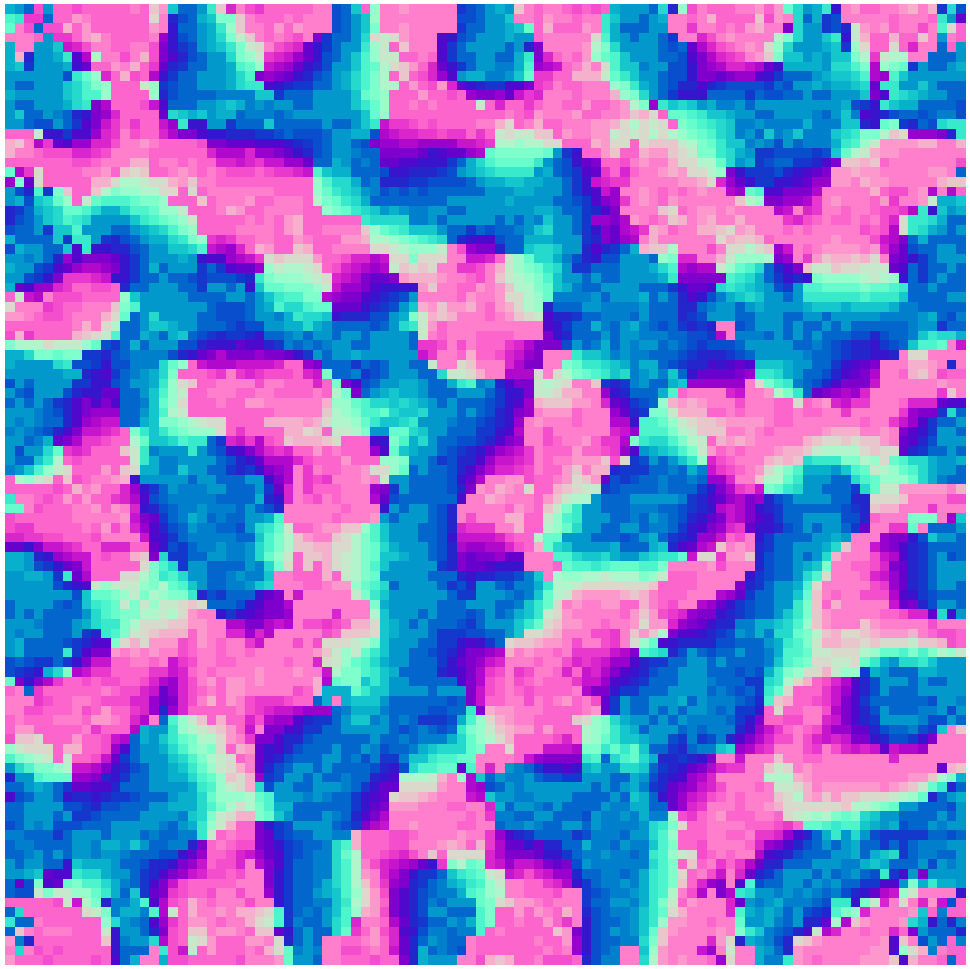
Image-level training, local filters but no weight sharing

- Training on 115x115 images. Kernels are 15x15 (not shared across space!)



Topographic Maps

K Obermayer and GG Blasdel, Journal of Neuroscience, Vol 13, 4114-4129 (**Monkey**)

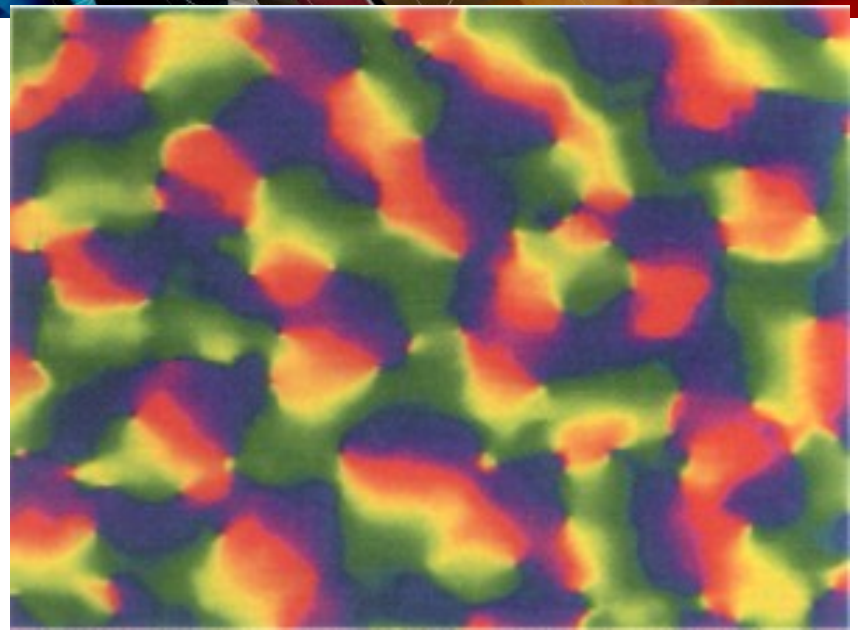


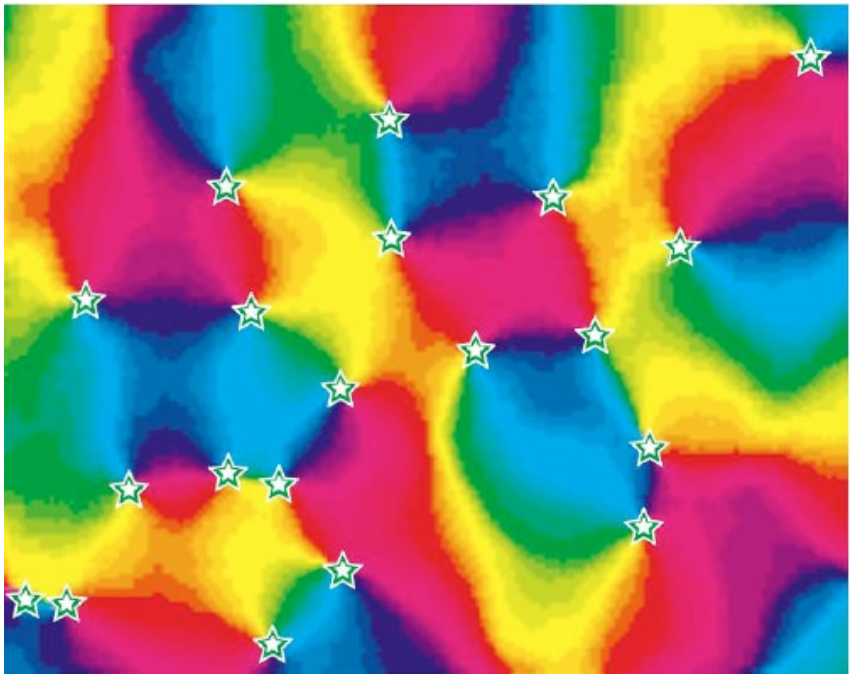
119x119 Image Input

100x100 Code

20x20 Receptive field size

sigma=5

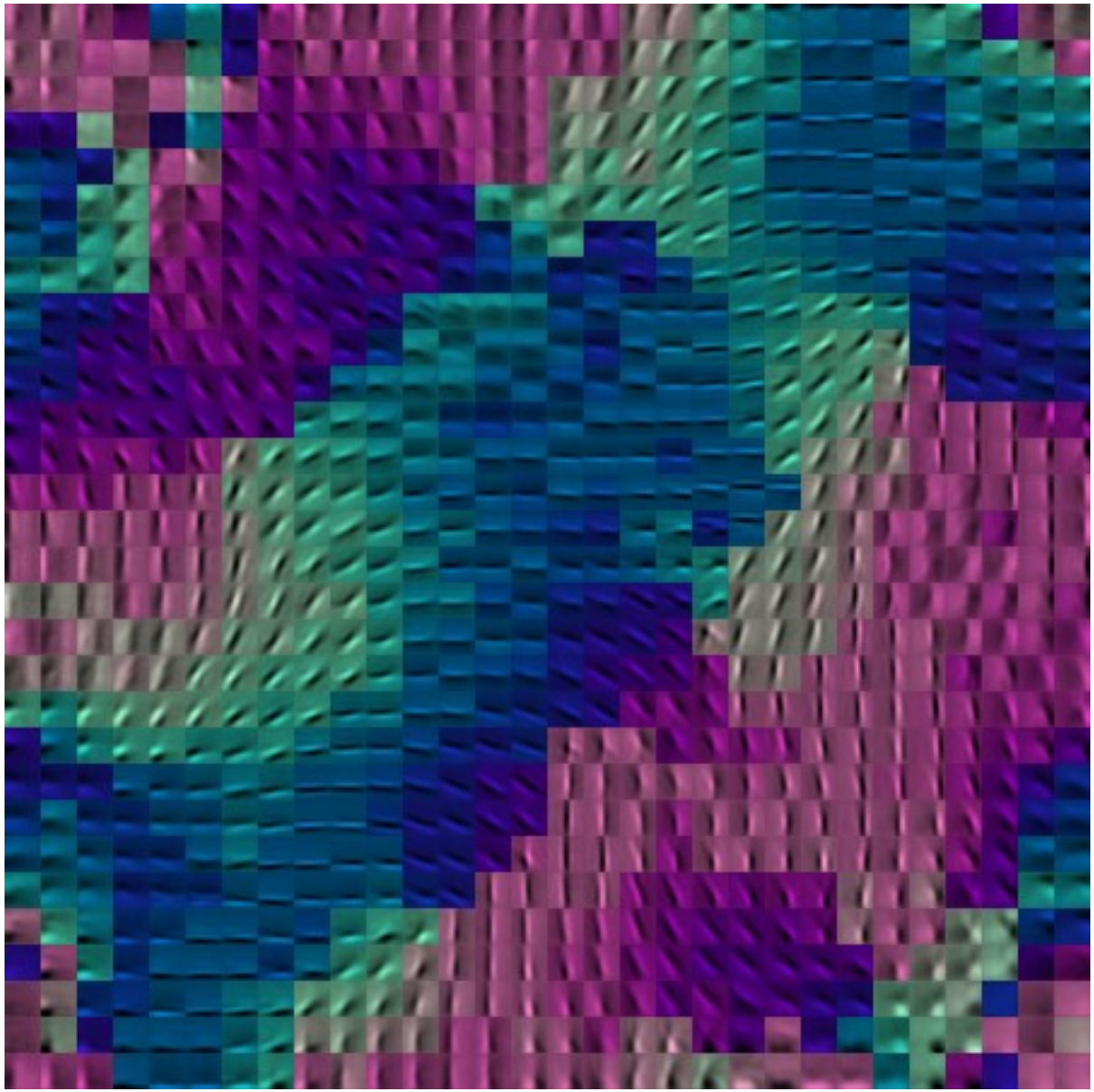




Michael C. Crair, et. al. The Journal of Neurophysiology Vol. 77 No. 6 June 1997, pp. 3381-3385 (**Cat**)

Image-level training, local filters but no weight sharing

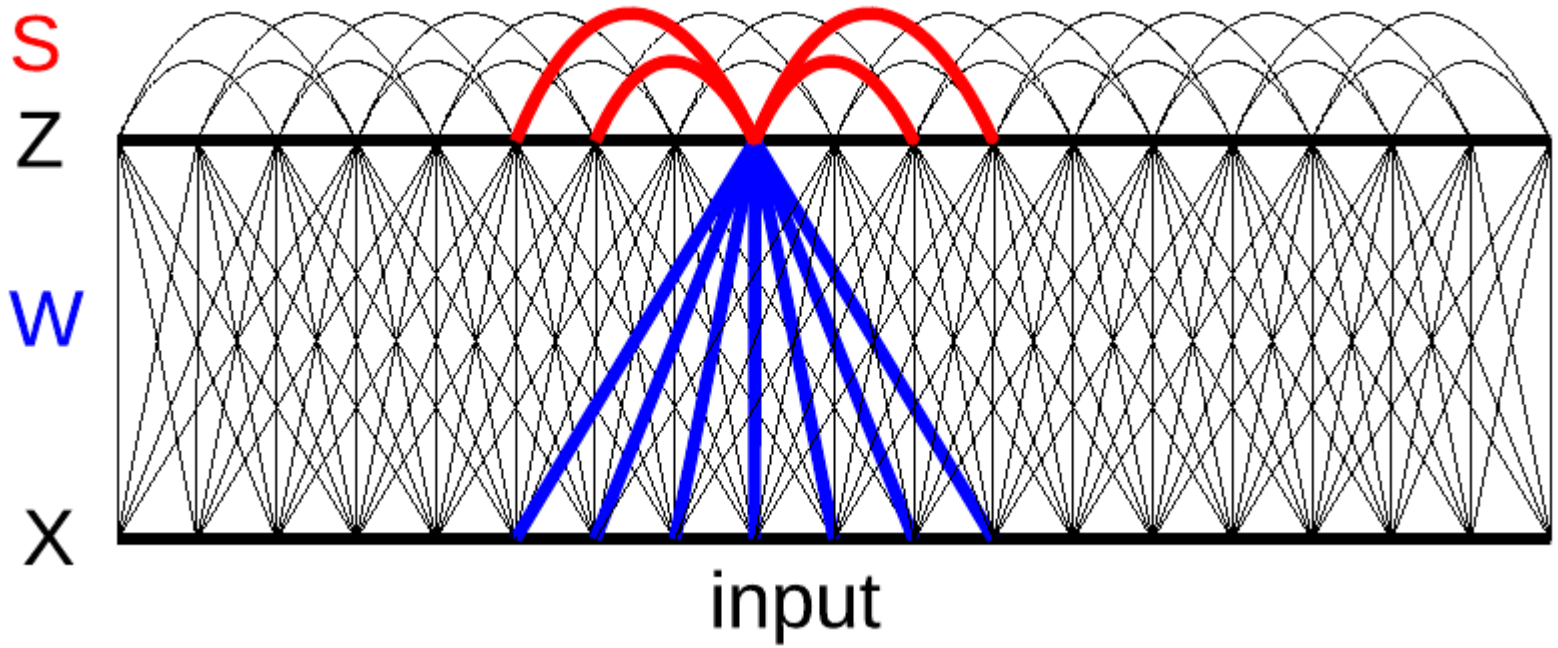
- Color indicates orientation (by fitting Gabors)



Invariant Features Lateral Inhibition

- Replace the L1 sparsity term by a lateral inhibition matrix
- Easy way to impose some structure on the sparsity

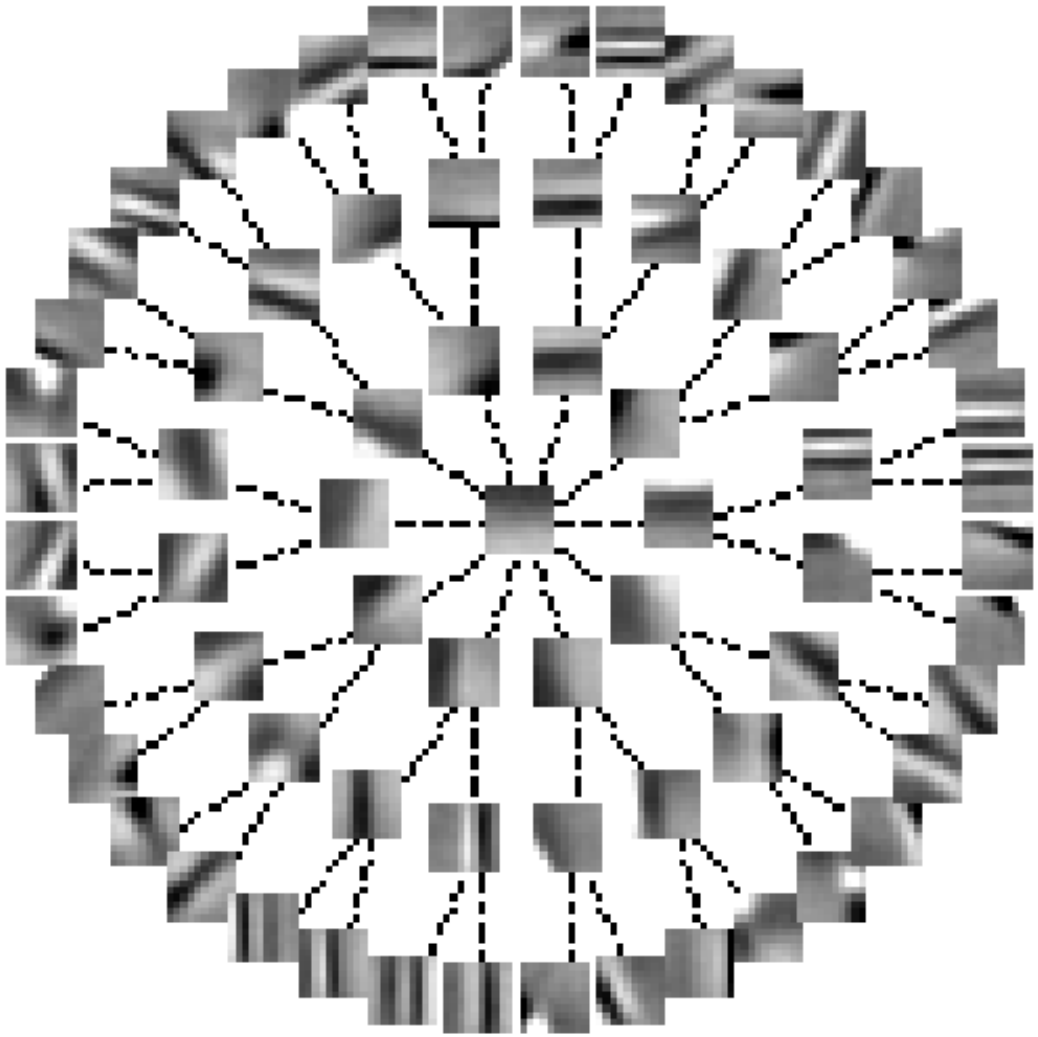
$$\min_{W,Z} \sum_{x \in X} ||Wz - x||^2 + |z|^T S|z|$$



[Gregor, Szlam, LeCun NIPS 2011]

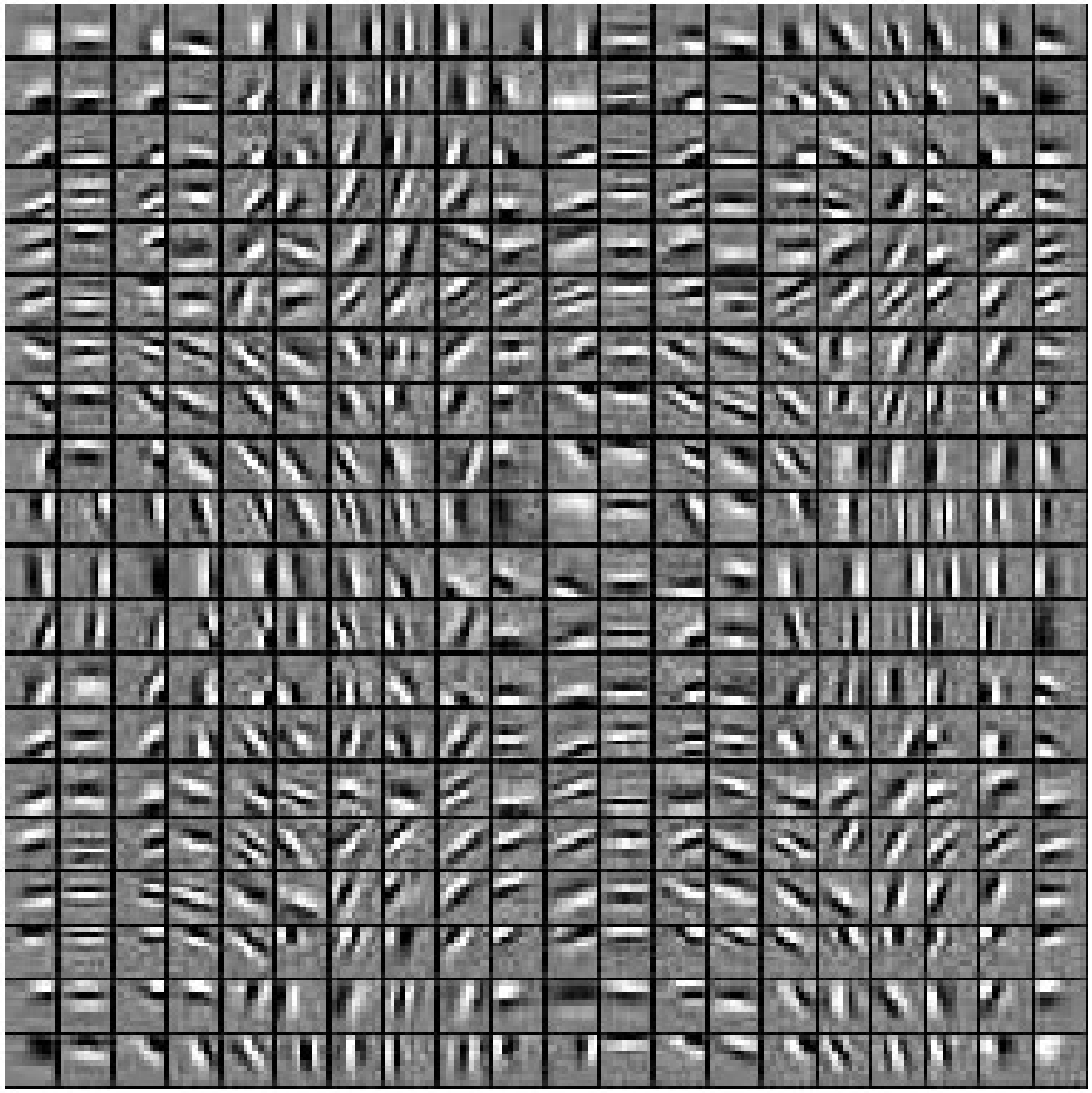
Invariant Features via Lateral Inhibition: Structured Sparsity

- Each edge in the tree indicates a zero in the S matrix (no mutual inhibition)
- Sij is larger if two neurons are far away in the tree



Invariant Features via Lateral Inhibition: Topographic Maps

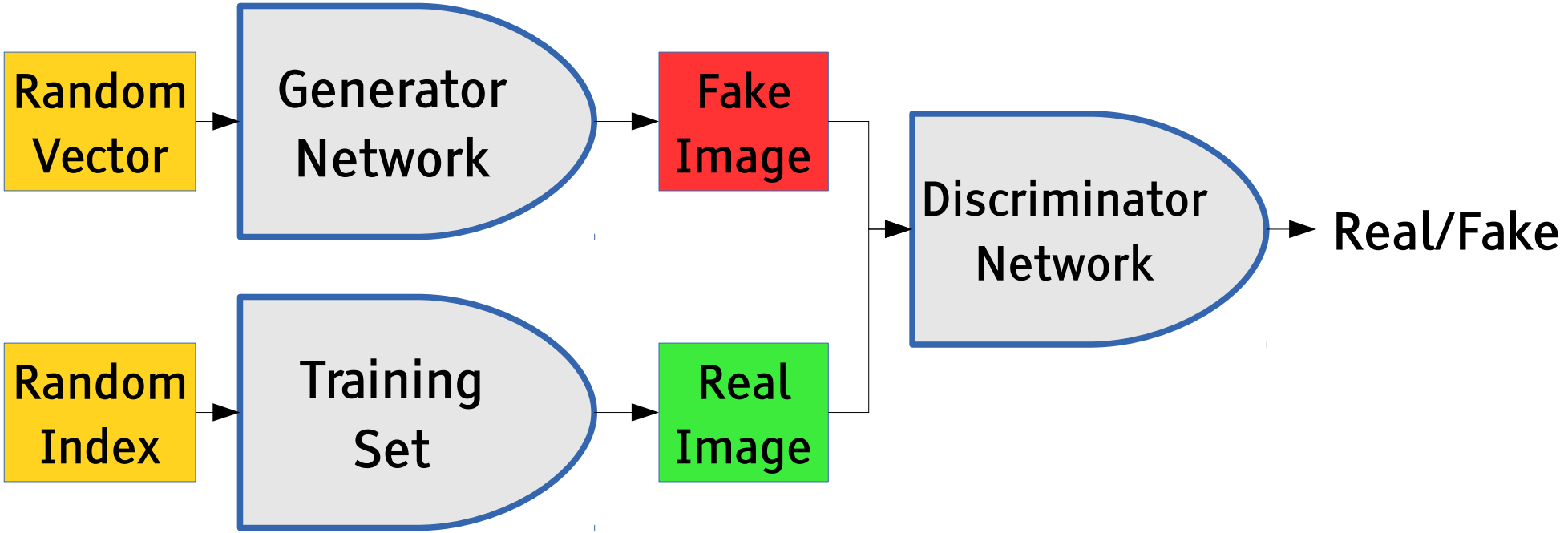
- **Non-zero values in S form a ring in a 2D topology**
  - Input patches are high-pass filtered




Laplacian Generative
Adverserial Network
(LAGPAN)
Also known as EyeScream

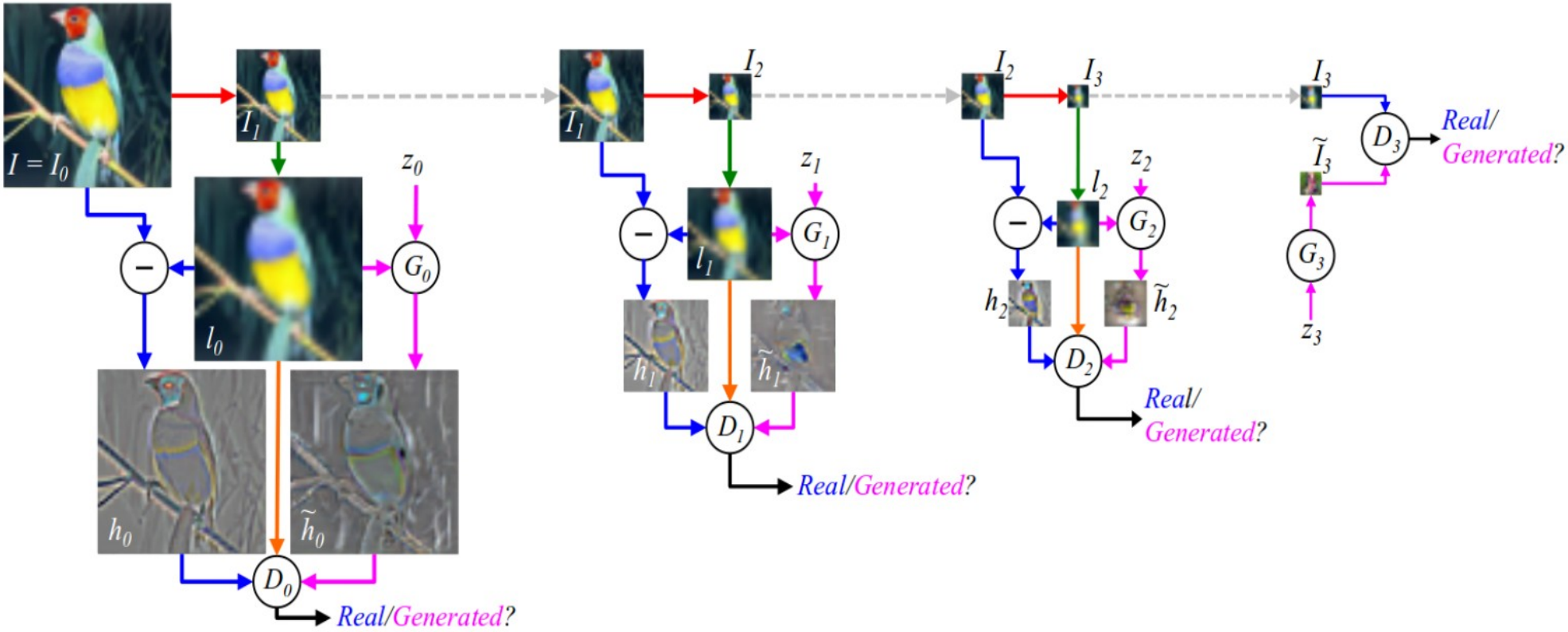
Generative Adversarial Networks

- [Goodfellow et al. NIPS 2014]
- Generator net maps random numbers to image
- Discriminator learns to tell real from fake images.
- Generator can cheat: it knows the gradient of the output of the discriminator with respect to its input



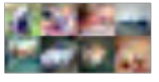
Laplacian GAN: LAPGAN (aka EyeScream)

- Learns to generate images [Denton et al. NIPS 2015]
- Generator net produces coefficients of a Laplacian Pyramid representation of the image
- Discriminator learns to tell real from fake Laplacian images.

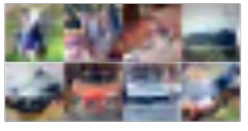


“EyeScream”

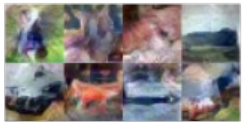
- **http://soumith.ch/eyescream/**



CIFAR-8

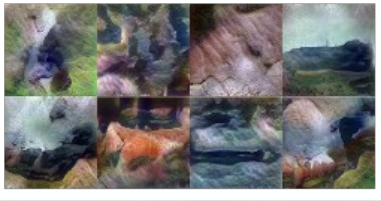


CIFAR-16

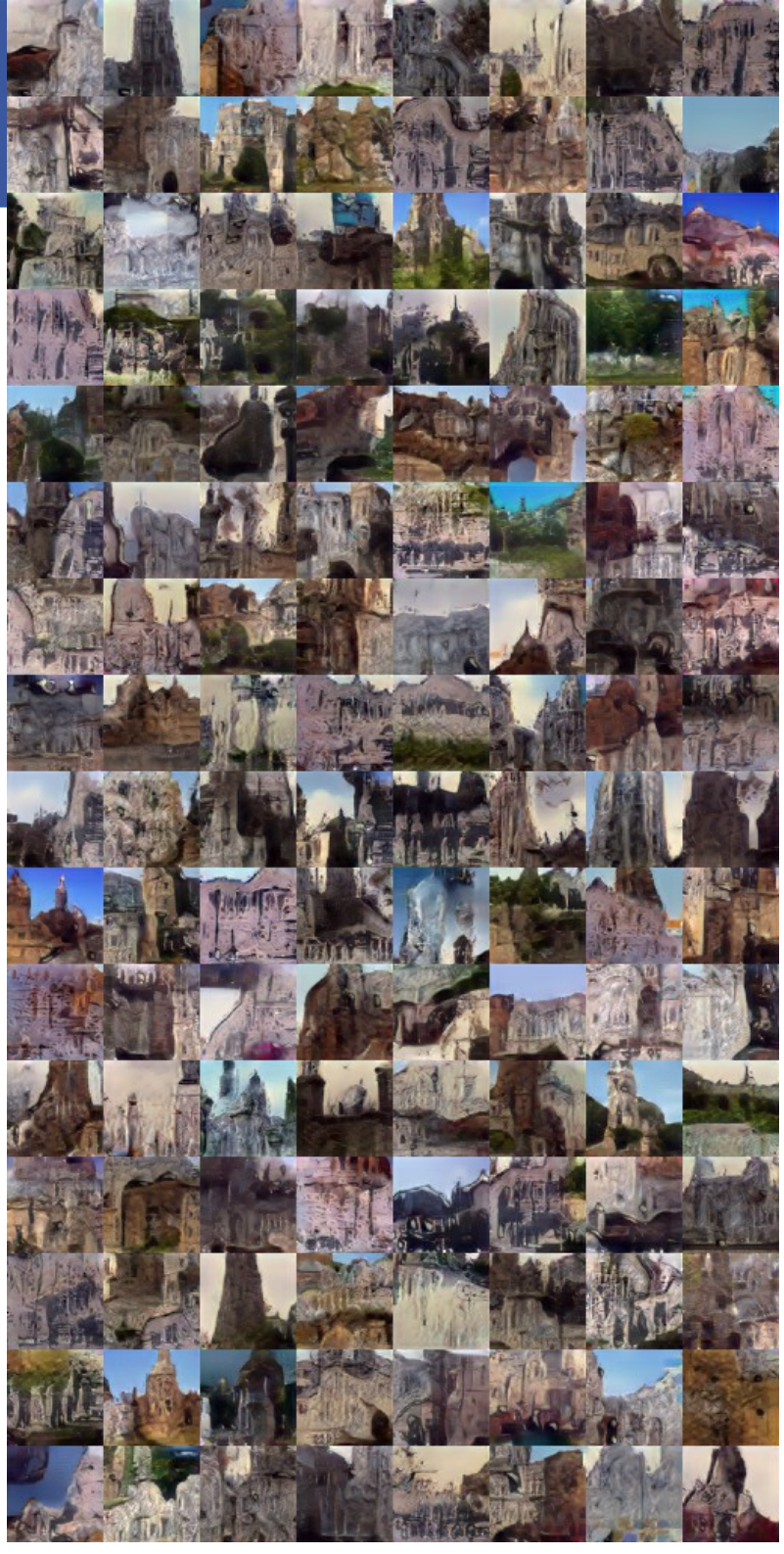


Imagenet-32

Imagenet-32
(recursive)

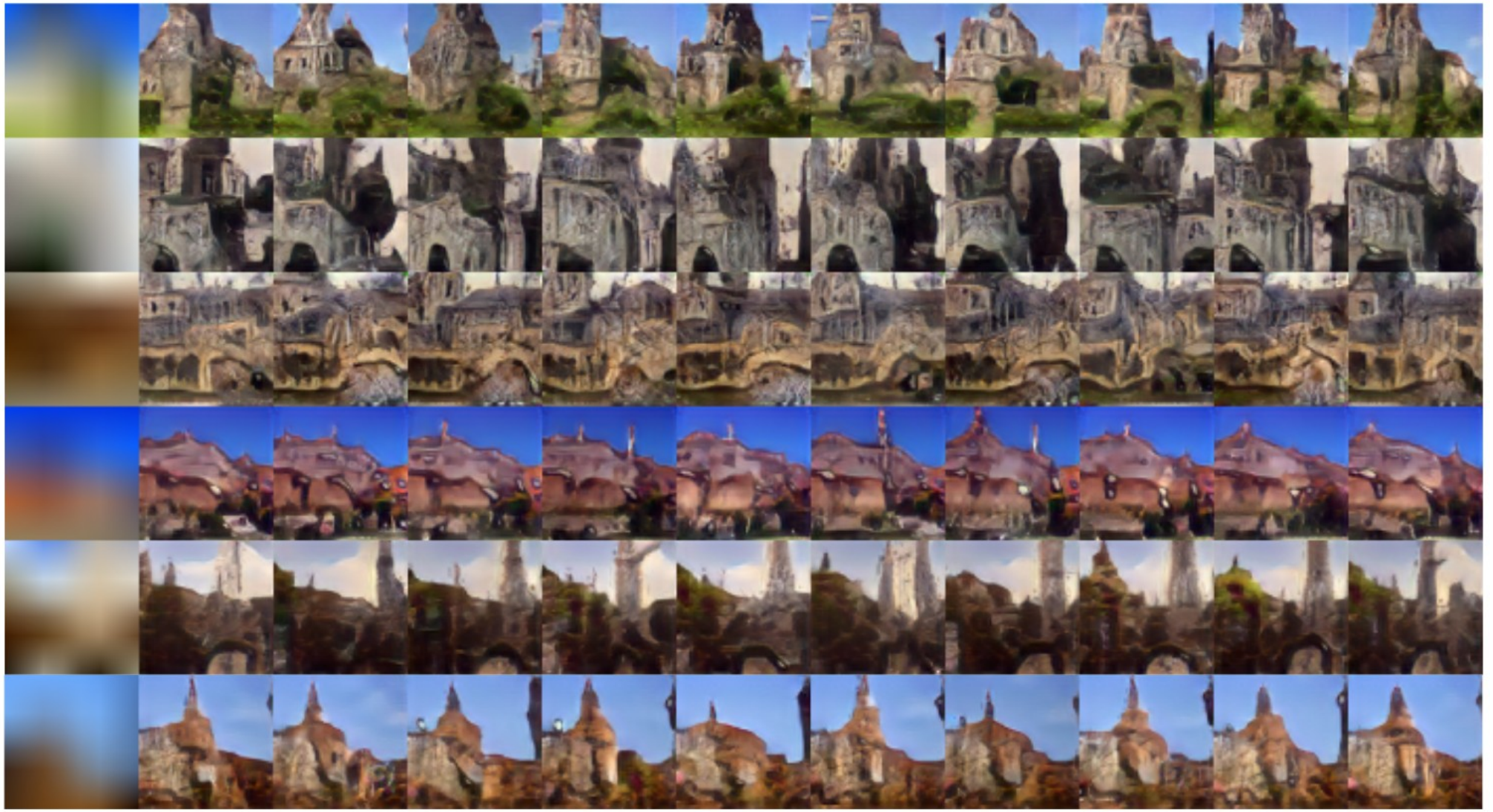


Imagenet-32
(recursive)



"EyeScream"

- **http://soumith.ch/eyescream/**

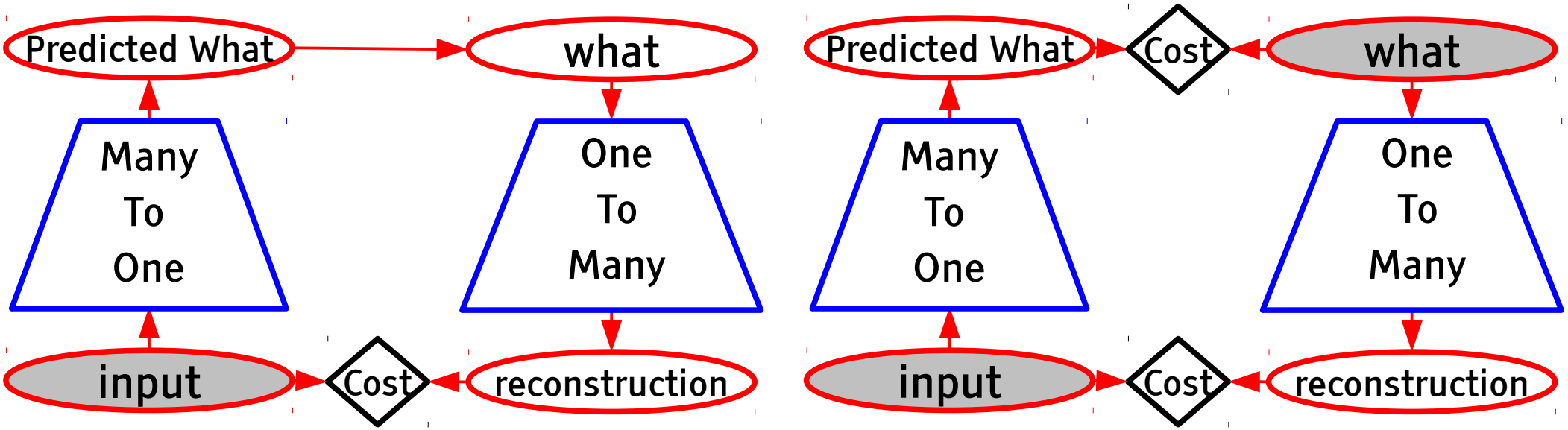



Integrating
Supervised & Unsupervised
Learning

Supervised and Unsupervised in One Learning Rule?

- **Boltzmann Machines have all the right properties [Hinton 1831] [OK, OK 1983 ;-]**
  - Sup & unsup, generative & discriminative in one simple/local learning rule
  - Feedback circuit reconstructs and propagates virtual hidden targets
  - But they don't really work (or at least they don't scale).
- **Problem: the feedforward path eliminates information**
- **If the feedforward path is invariant, then**

  the reconstruction path is a one-to-many mapping
  - Usual solution: sampling. But I'm allergic.



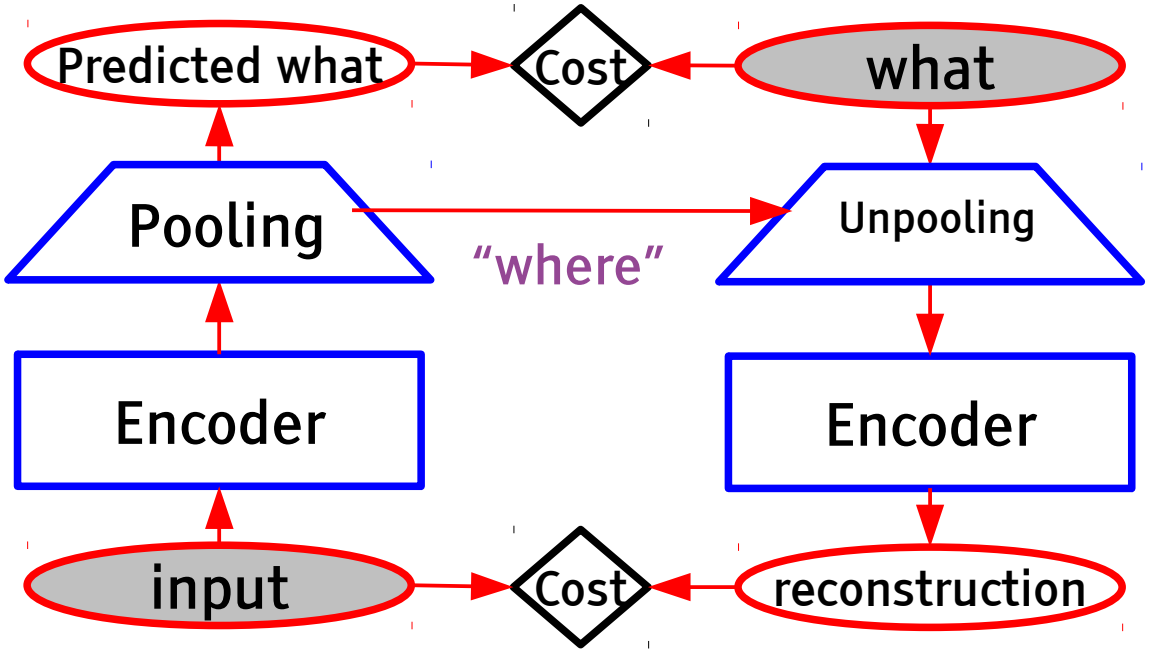
Supervised and Unsupervised in One Learning Rule?

Idea: keep the complementary information around

- So that the generative path is a function

What-Where Auto-Encoder

- [Ranzato 2007], [Gregor 2011], Hinton's Capsules
- Very old ideas by Hinton and others about separating identity from instantiation parameters.

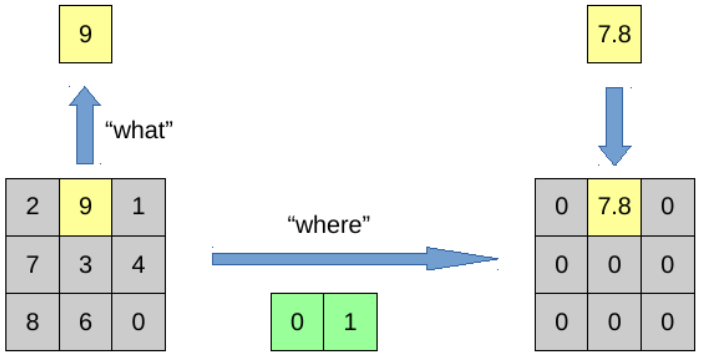


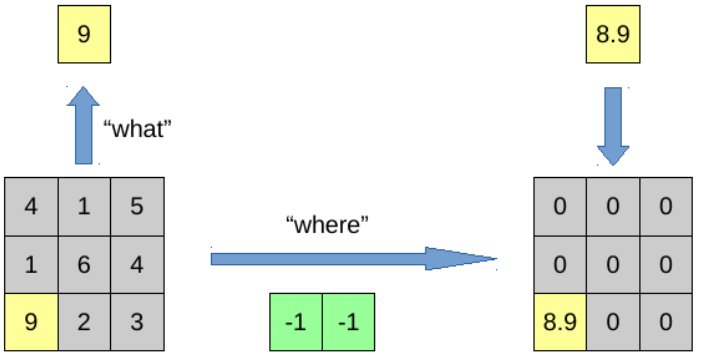
Computing the "where": Phase Pooling

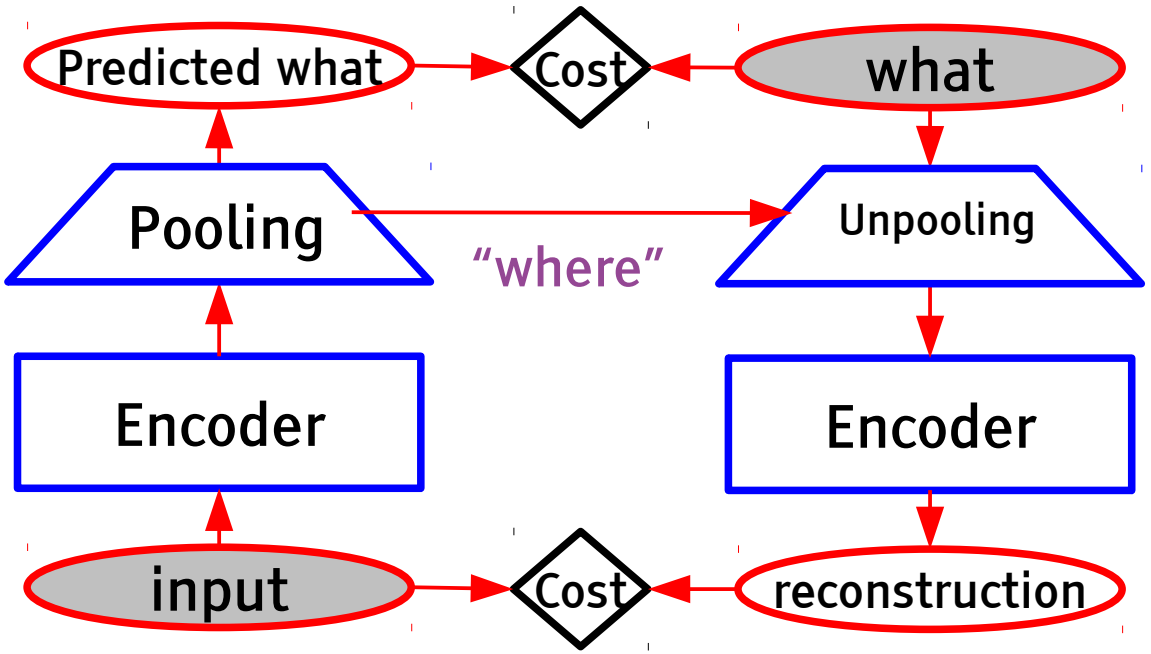
A funny kind of pooling/unpooling

$$m_k = \sum_{N_k} z(f,x,y) \frac{e^{\beta z(f,x,y)}}{\sum_{N_k} e^{\beta z(f',x',y')}} \approx \max_{N_k} z(f,x,y)$$

$$\mathbf{p}_k = \sum_{N_k} \begin{bmatrix} f \\ x \\ y \end{bmatrix} \frac{e^{\beta z(f,x,y)}}{\sum_{N_k} e^{\beta z(f',x',y')}} \approx \arg\max_{N_k} z(f,x,y)$$



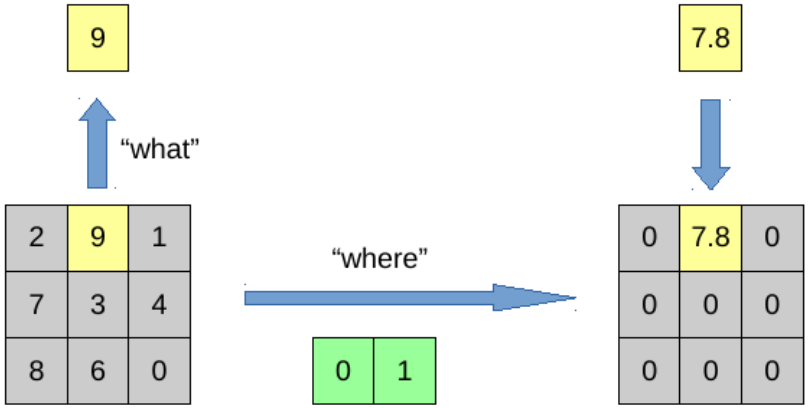




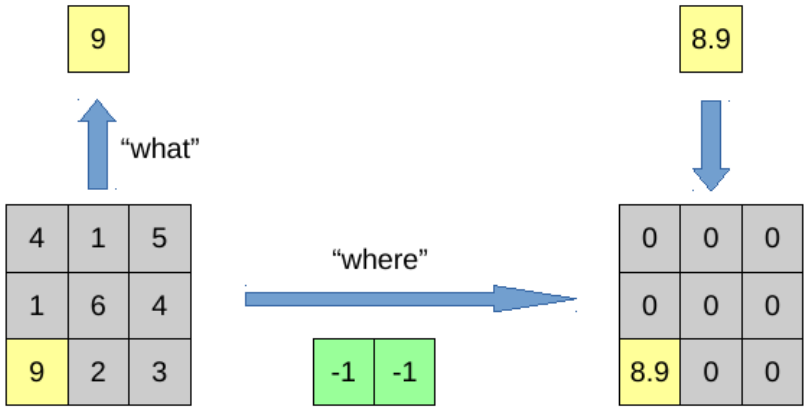
Stacked What-Where Auto-Encoder (SWWAE)

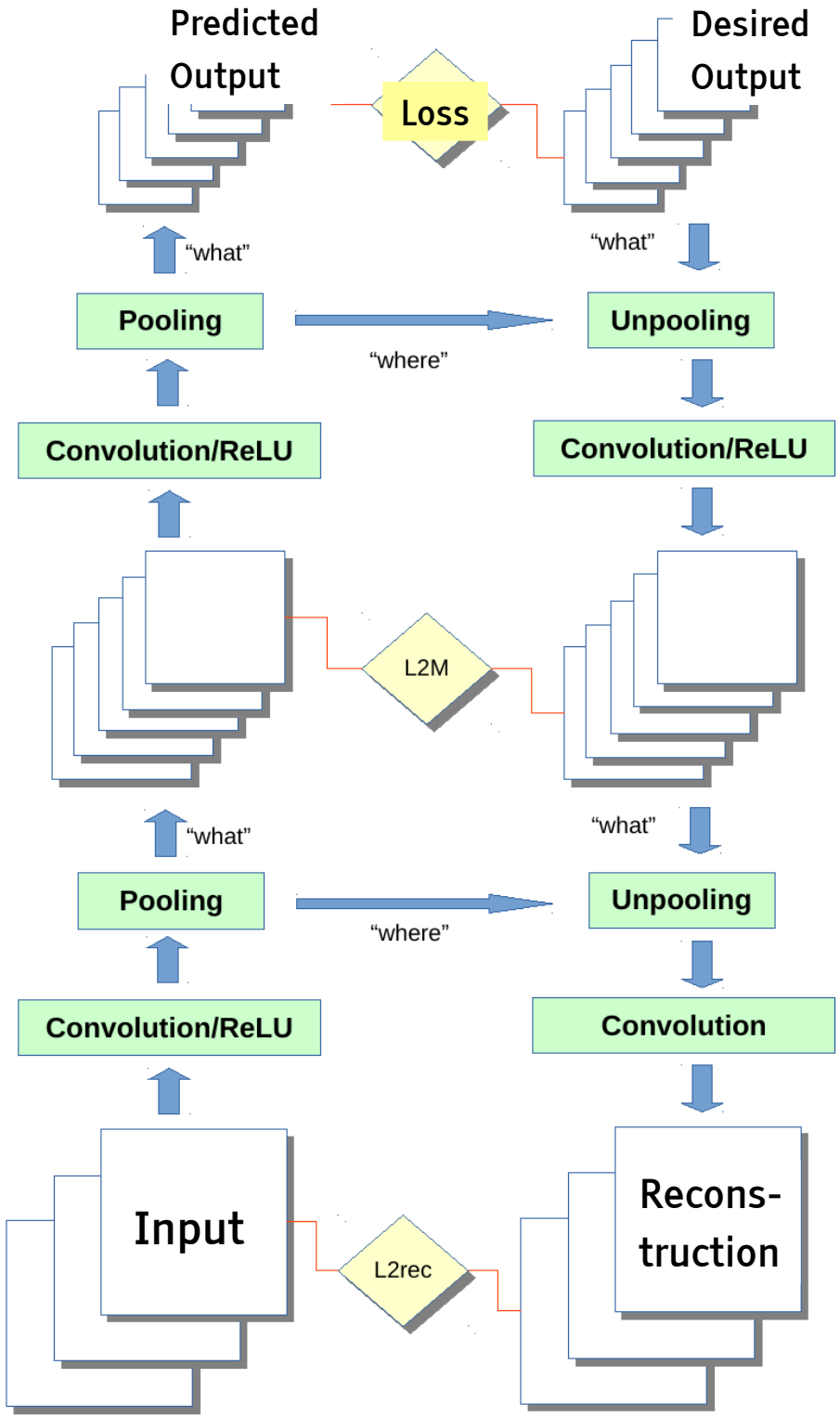
[Zhao, Mathieu, LeCun arXiv:1506.02351]

Stacked What-Where Auto-Encoder



A bit like a ConvNet paired with a DeConvNet

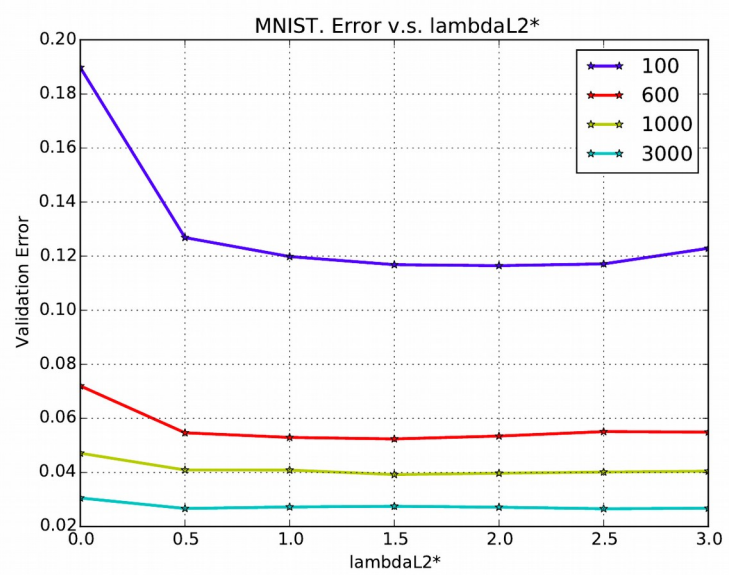




MNIST: Recognition & Generation

model / N	100	600	1000	3000
SWWAE	[1]11.65 ± 0.20	[1]5.24 ± 0.06	[1]3.92 ± 0.09	[2]2.66 ± 0.07
dp	21.11 ± 0.65	7.11 ± 0.27	5.34 ± 0.39	3.23 ± 0.30
dp-fc	16.09 ± 0.96	6.14 ± 0.36	4.368 ± 0.50	2.98 ± 0.08
unsup-sfx	17.81 ± 0.06	8.41 ± 0.08	6.40 ± 0.06	4.76 ± 0.03
unsup-pretr	-	9.80 ± 0.06	6.135 ± 0.03	4.41 ± 3.11
noL2M	[2]13.48 ± 2.35	[2]5.69 ± 0.33	[2]3.97 ± 0.37	[1]2.52 ± 0.06

model / N	100	600	1000	3000
NN	25.81	11.44	10.7	6.04
Convnet [14]	22.98	7.86	6.45	3.35
TSVM [28] [28]	16.81	6.16	5.38	3.45
CAE [26]	13.47	6.3	4.77	3.22
MTC [25]	12.03	5.13	3.64	2.57
PL-DAE [15]	10.49	5.03	3.46	2.69
M1+M2 [11]	[1]3.33 ± 0.14	[1]2.59 ± 0.05	[1]2.40 ± 0.02	[1]2.18 ± 0.04
SWWAE	[2]9.17 ± 0.11	[2]4.16 ± 0.11	[2]3.39 ± 0.01	[2]2.50 ± 0.01

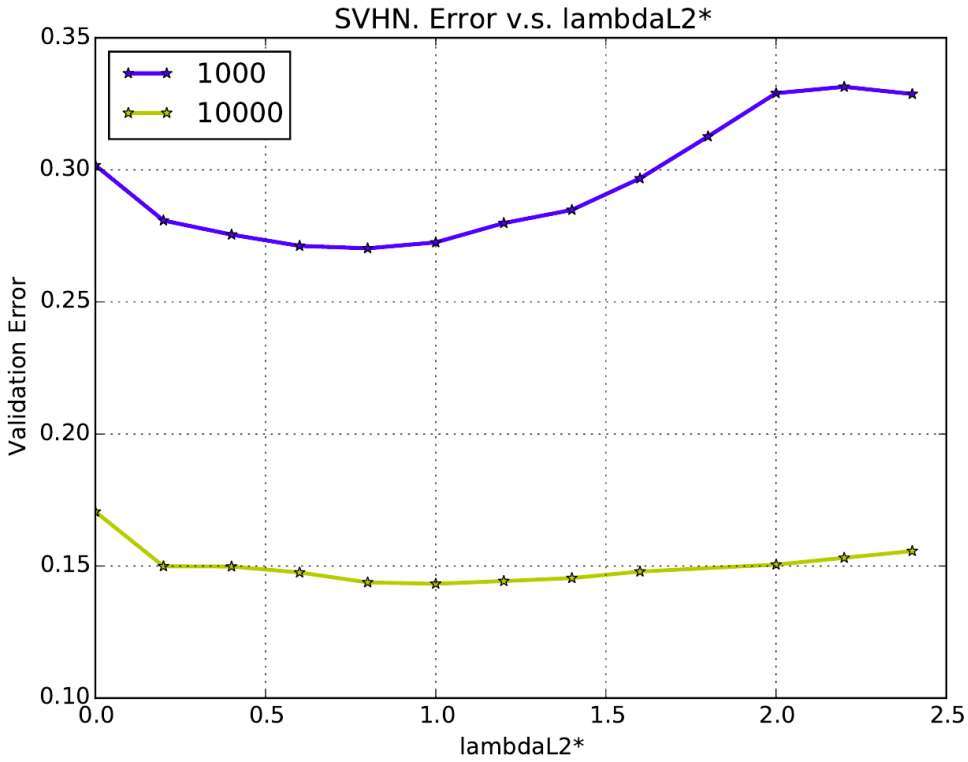


SVHN: Recognition

- **House Numbers**
- **Network:**

(128)5c-2p-(128)3c-(256)3c-2p-(256)3c-2p

- **Error rate on full dataset**
  - No reconstruction: 5.89%
  - With reconstruction: 4.94%



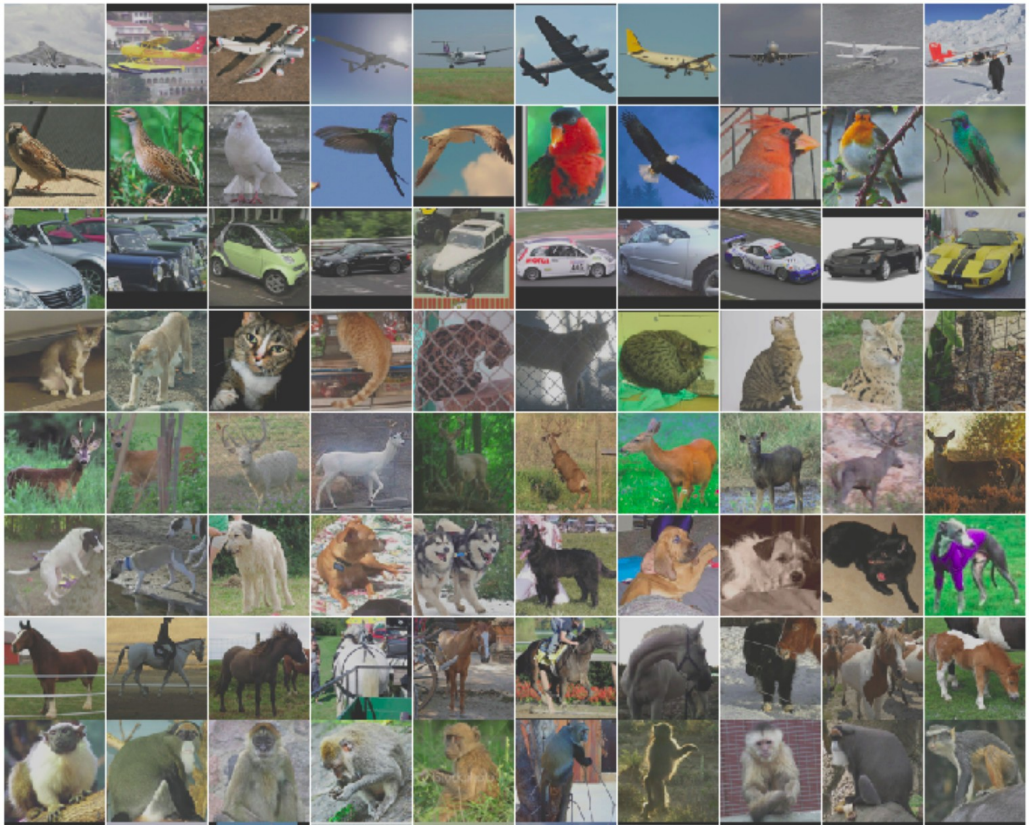
model / N	1000	10000
SWWAE	[1]27.03	[1]14.33
dp	32.25	15.72
dp-fc	[2]27.81	[2]14.39
L1	30.22	14.57
unsup-pretr	33.03	17.31
noL2M	29.12	16.51

model / N	1000	10000
KNN	77.93	NA
TSVM [28]	66.55	NA
M1+KNN [11]	65.63	NA
M1+TSVM [11]	54.33	NA
M1+M2 [11]	[2]36.02	NA
SWWAE	[1]27.83	[1]14.22

STL-10: Recognition

- 10 classes: airplane, bird, cat, car, deer, dog, horse, monkey, ship, truck
- 96x96 color images (from ImageNet)
- 10 predefined folds with 1000 training samples each (100 per class)
- 5000 total training samples
- 100K unlabeled samples (other categories)
- 8000 test samples (800 per class)

[Coates et al. 2011]



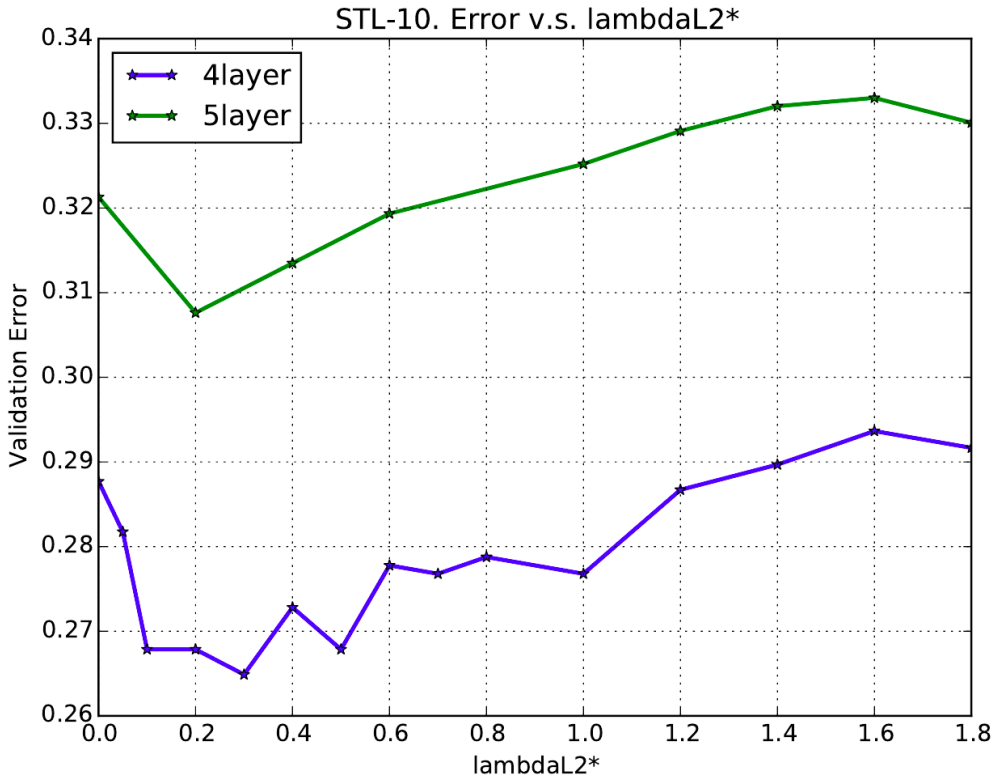
model	accuracy(val)
SWWAE-4layer	[1]73.51%
SWWAE-5layer	69.24%
noL2M-5layer	68.26%
noL2M-4layer	70.93%
dp-4layer	70.54%
dp-fc-4layer	71.43%

model	accuracy
Convolutional Kernel Networks [17]	62.32%
HMP [1]	64.5%
NOMP [16]	67.9%
Multi-task Bayesian Optimization [27]	70.1%
Zero-bias Convnets + ADCU [20]	70.2%
Exemplar Convnets [2]	72.8%
SWWAE-4layer	[1]74.80%

STL-10: Recognition

- 10 classes: airplane, bird, cat, car, deer, dog, horse, monkey, ship, truck
- 96x96 color images (from ImageNet)
- 10 predefined folds with 1000 training samples each (100 per class)
- 5000 total training samples
- 100K unlabeled samples (other categories)
- 8000 test samples (800 per class)

[Coates et al. 2011]



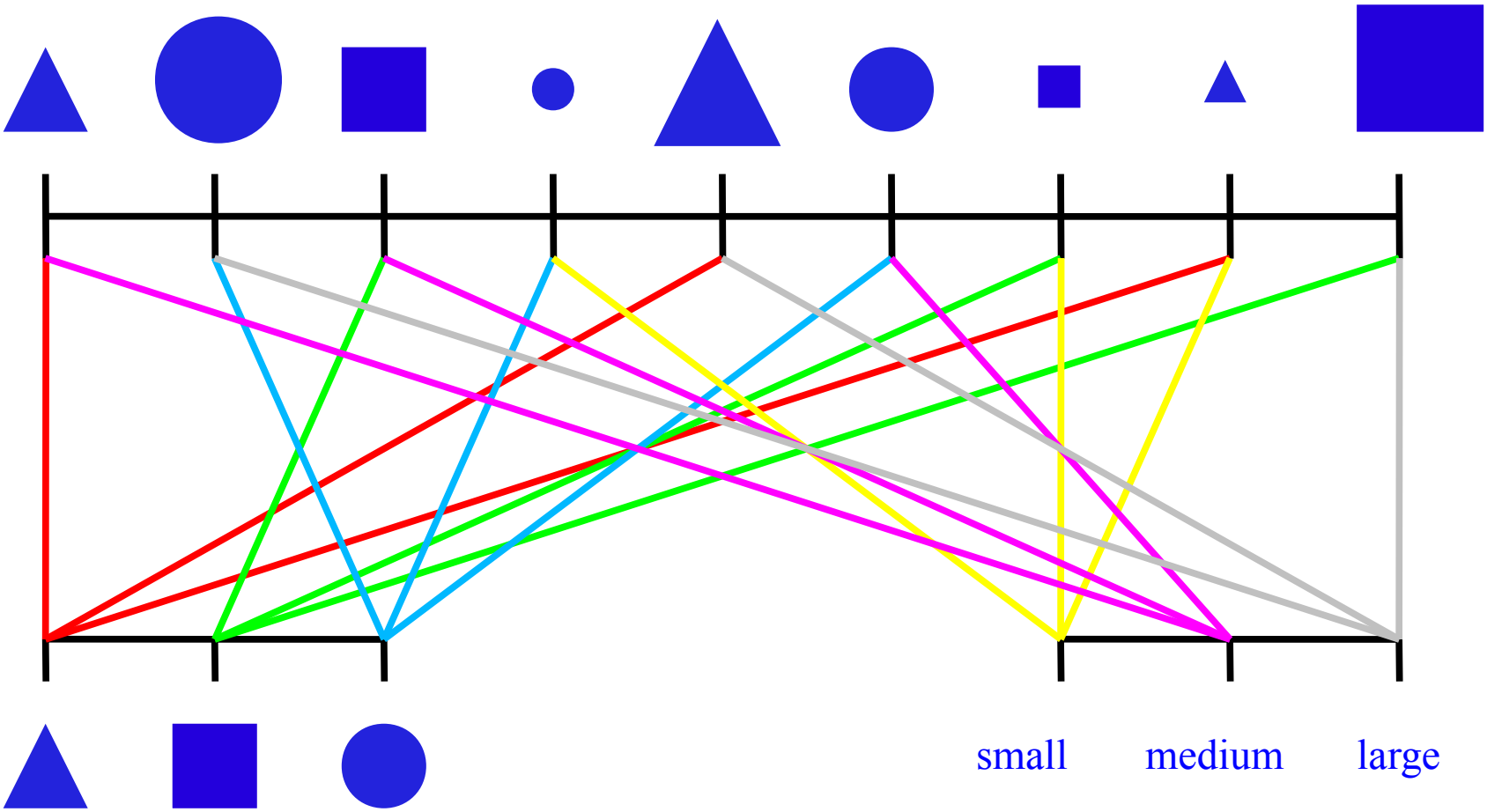
model	accuracy(val)
SWWAE-4layer	[1]73.51%
SWWAE-5layer	69.24%
noL2M-5layer	68.26%
noL2M-4layer	70.93%
dp-4layer	70.54%
dp-fc-4layer	71.43%

model	accuracy
Convolutional Kernel Networks [17]	62.32%
HMP [1]	64.5%
NOMP [16]	67.9%
Multi-task Bayesian Optimization [27]	70.1%
Zero-bias Convnets + ADCU [20]	70.2%
Exemplar Convnets [2]	72.8%
SWWAE-4layer	[1]74.80%


Unsupervised Learning
Through Temporal Prediction
& Temporal Consistency

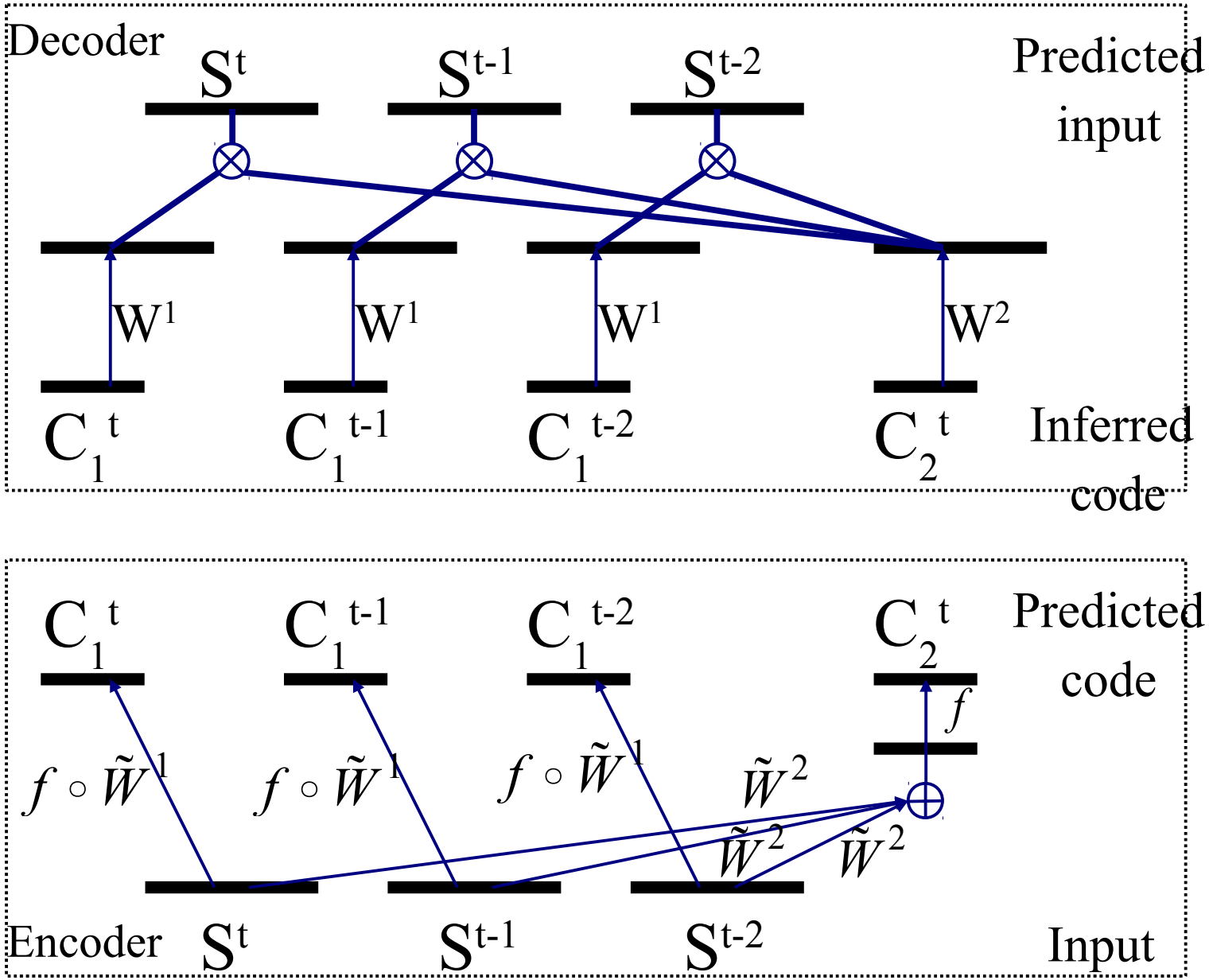
Invariant Features through Temporal Constancy

- Object is cross-product of object type and instantiation parameters
  - Mapping units [Hinton 1981], capsules [Hinton 2011]



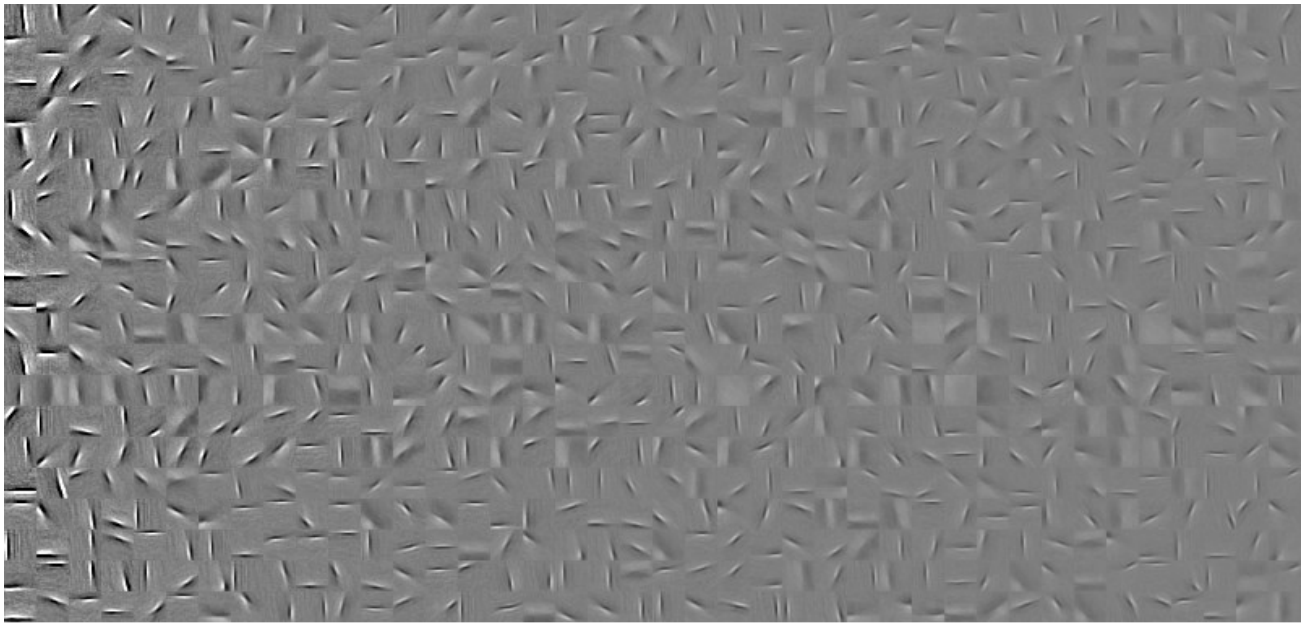
[Karol Gregor et al.]

What-Where Auto-Encoder Architecture

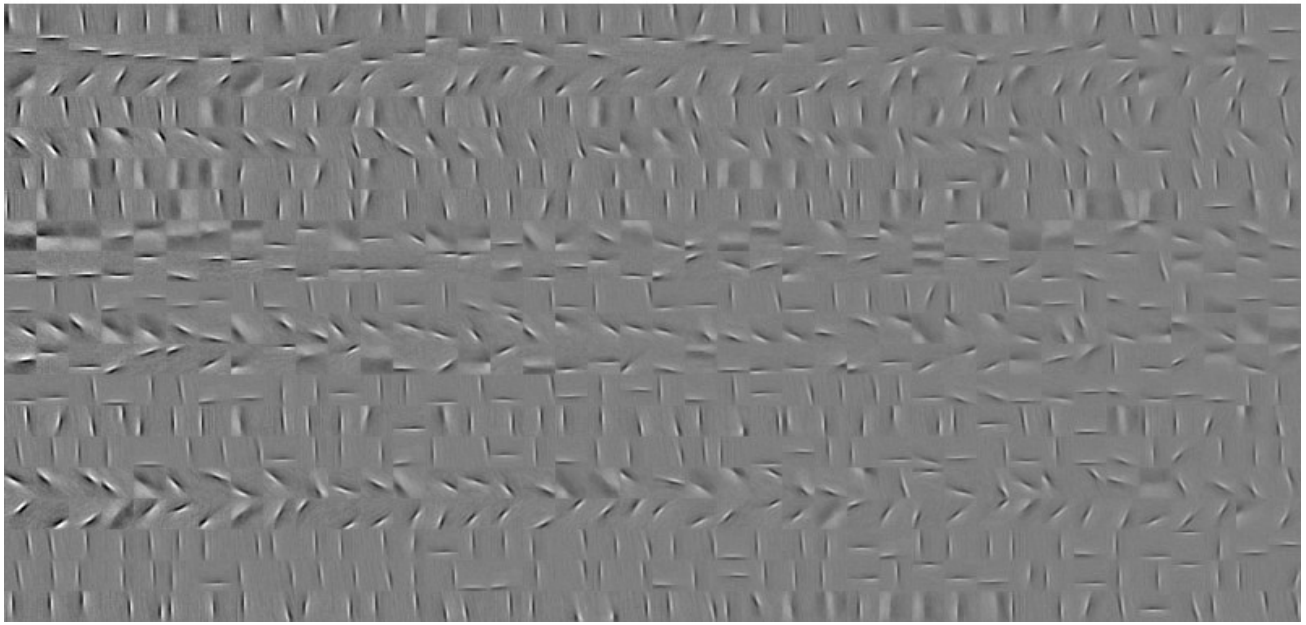


Low-Level Filters Connected to Each Complex Cell

C1
(where)

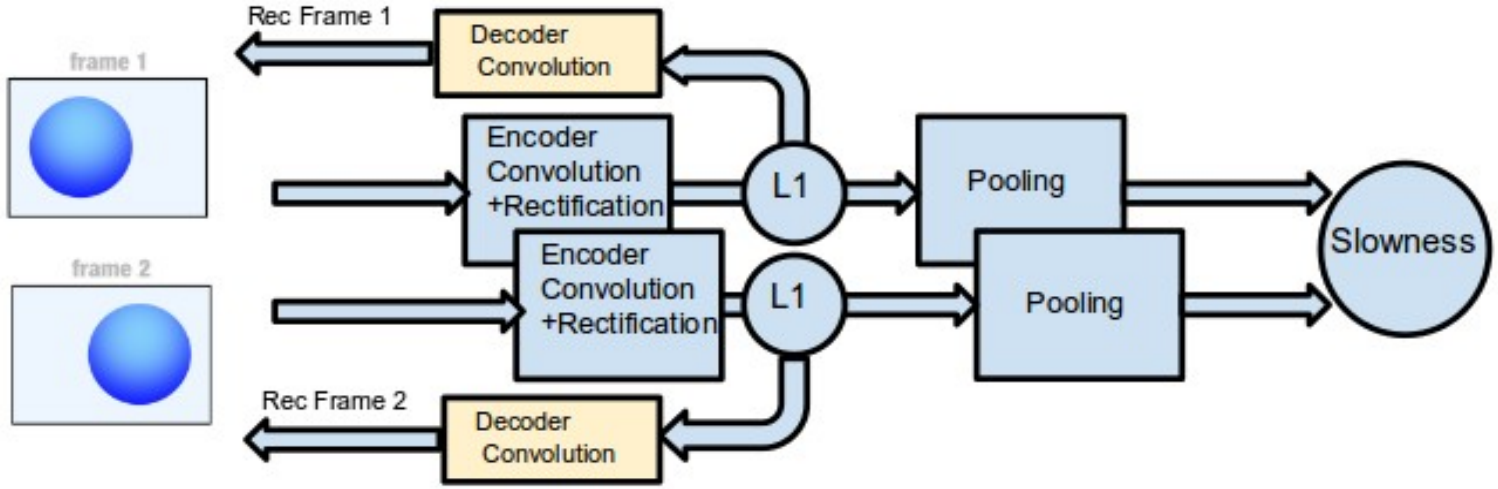


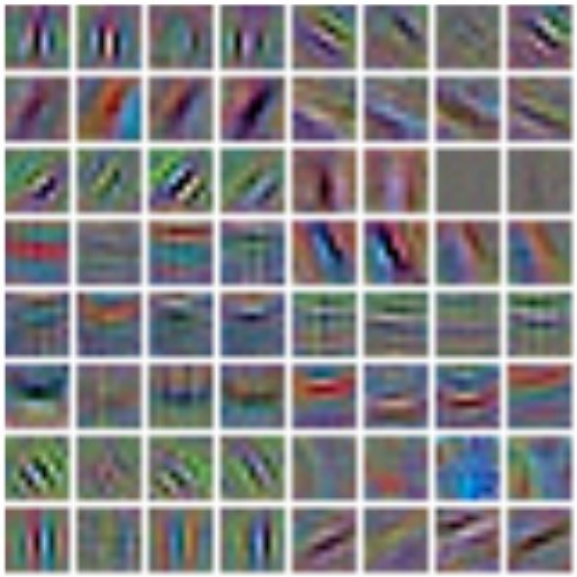
C2
(what)



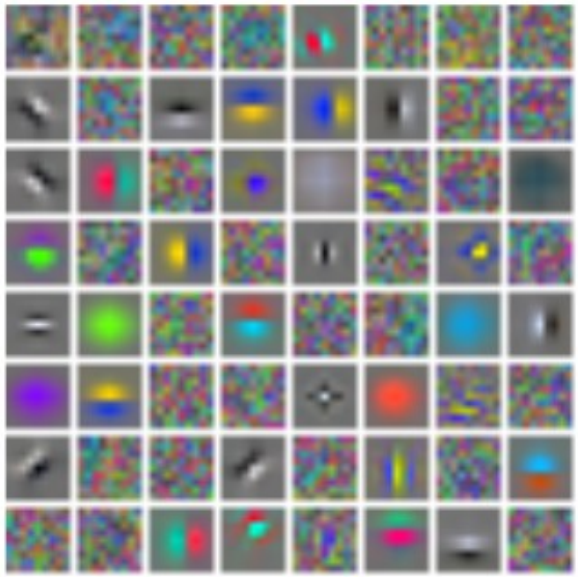
Sparse Auto-Encoder with "Slow Feature" Penalty

Y LeCun

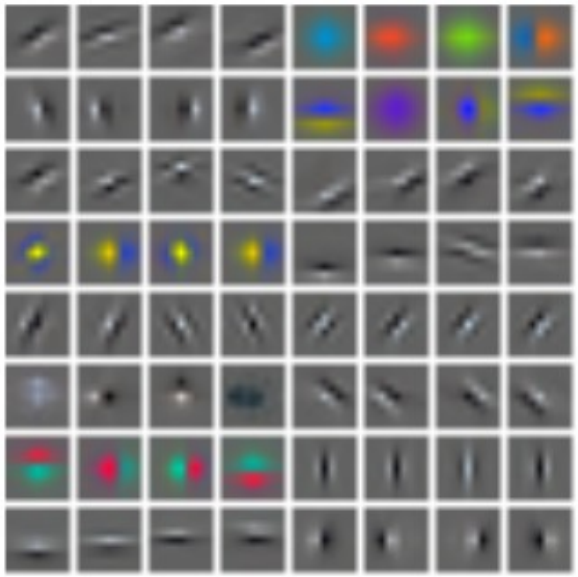




Supervised filters CIFAR10



sparse conv. auto-encoder



slow & sparse conv. auto-encoder

Trained on YouTube videos

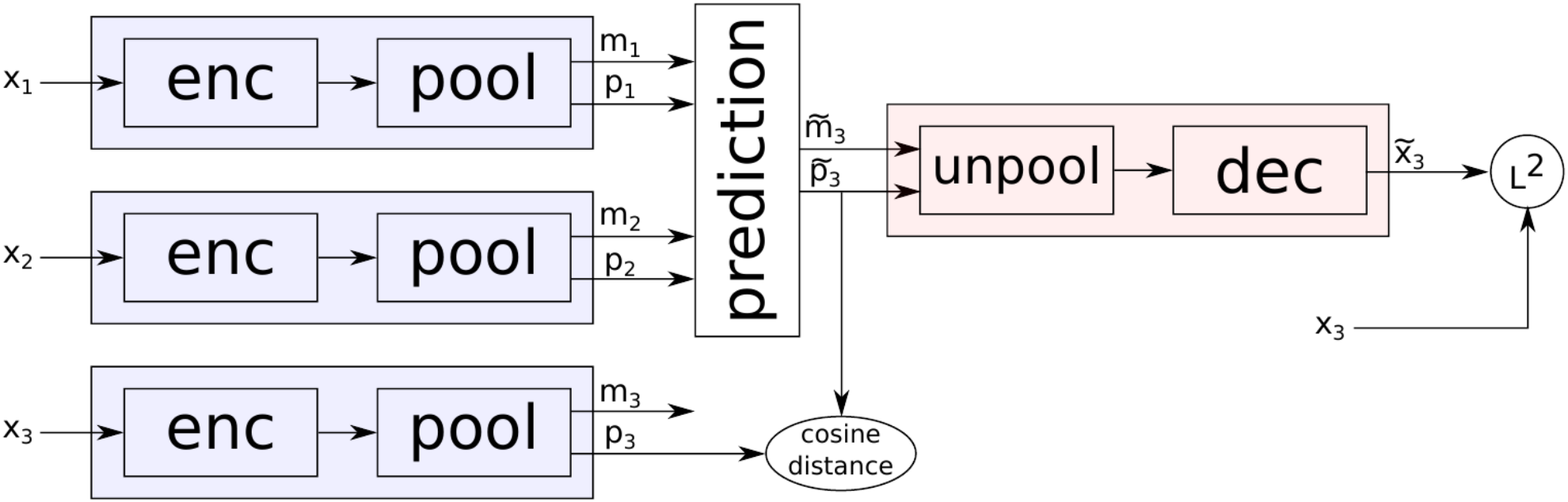
[Goroshin et al. ICCV 2015, Arxiv:1412.6056]

Unsupervised Learning by Prediction and Linearization

[Goroshin, Mathieu, LeCun NIPS 2015, arXiv:1506.03011]

- Trained on 3 successive frames of video
- Maximize the colinearity between successive changes of feature vectors
- So that "natural" changes stay in linear manifold in feature space.

$$L = \frac{1}{2}\|G_W(\mathbf{a}\begin{bmatrix} z^t & z^{t-1}\end{bmatrix}^T) - x^{t+1}\|_2^2 - \lambda \frac{(z^t - z^{t-1})^T(z^{t+1} - z^t)}{\|z^t - z^{t-1}\|\|z^{t+1} - z^t\|}$$



Unsupervised Learning by Prediction and Linearization

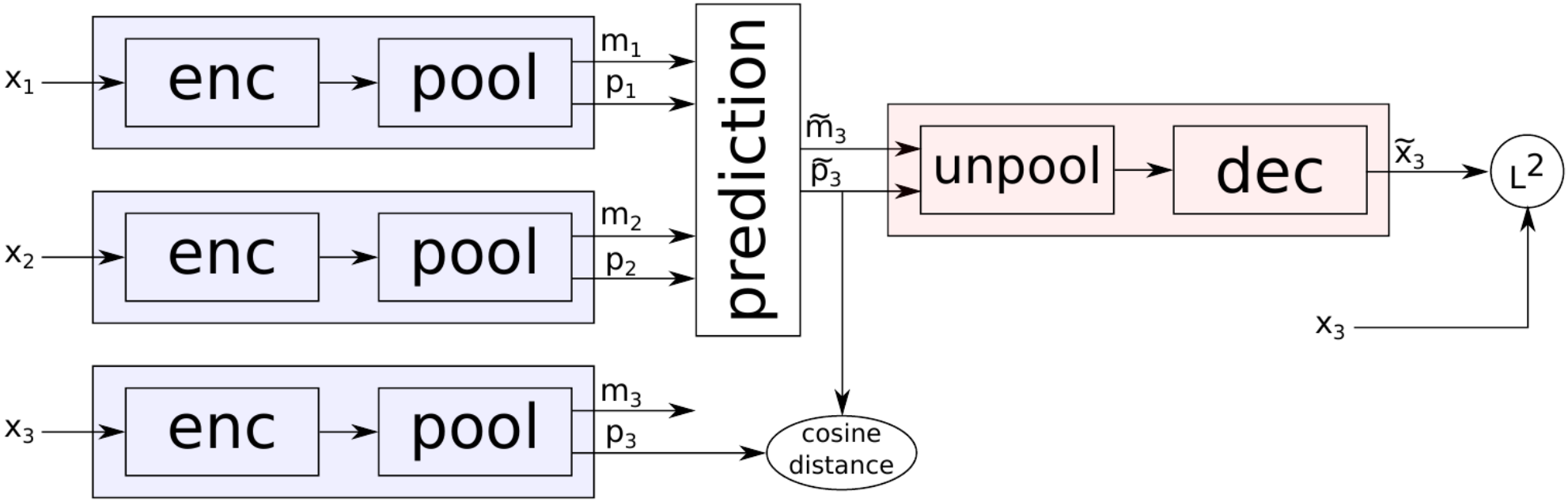
$$L = \frac{1}{2}\|G_W(\mathbf{a}\begin{bmatrix} z^t & z^{t-1}\end{bmatrix}^T) - x^{t+1}\|_2^2 - \lambda \frac{(z^t - z^{t-1})^T(z^{t+1} - z^t)}{\|z^t - z^{t-1}\|\|z^{t+1} - z^t\|}$$

- **Magnitude**
  - (soft max)

$$m_k = \sum_{N_k} z(f,x,y) \frac{e^{\beta z(f,x,y)}}{\sum_{N_k} e^{\beta z(f\prime,x\prime,y\prime)}} \approx \max_{N_k} z(f,x,y)$$

- **Phase**
  - (soft argmax)

$$\mathbf{p}_k = \sum_{N_k} \begin{bmatrix} f \\ x \\ y \end{bmatrix} \frac{e^{\beta z(f,x,y)}}{\sum_{N_k} e^{\beta z(f\prime,x\prime,y\prime)}} \approx \arg\max_{N_k} z(f,x,y)$$



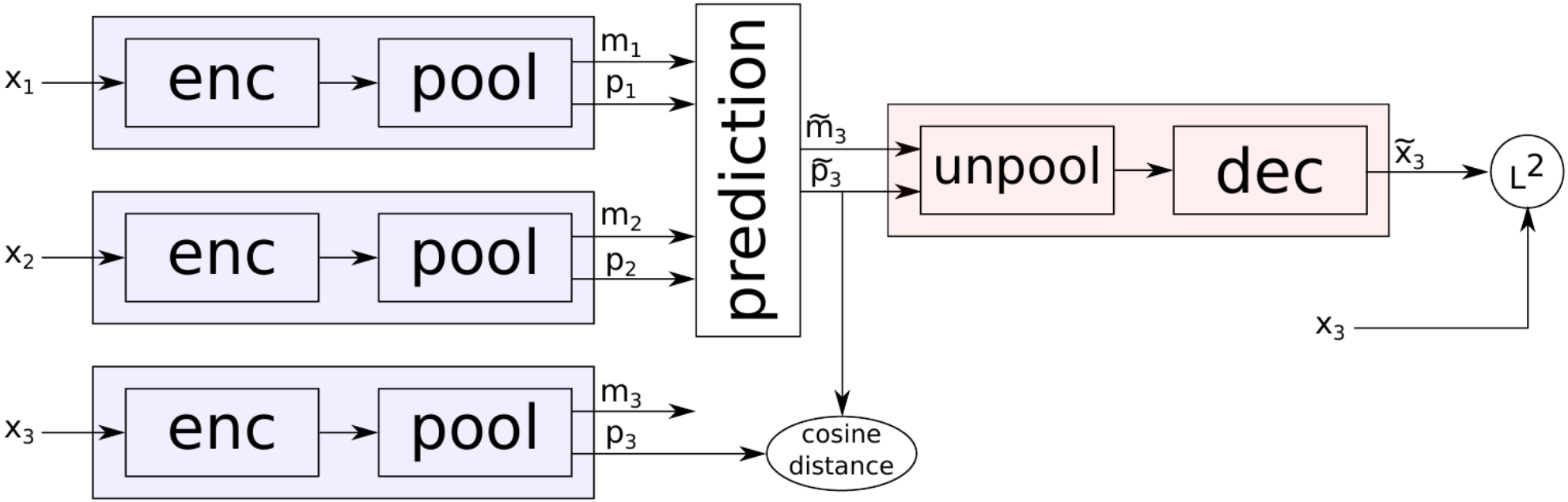
Unsupervised Learning by Prediction and Linearization

[Goroshin, Mathieu, LeCun NIPS 2015, arXiv:1506.03011]

- **Version 2:**
  - Make sure the "what"/magnitude changes slowly
  - Make sure the "where"/phase changes linearly

$$m^{t+1} = \frac{m^t + m^{t-1}}{2}$$

$$\mathbf{p}^{t+1} = 2\mathbf{p}^t - \mathbf{p}^{t-1}$$

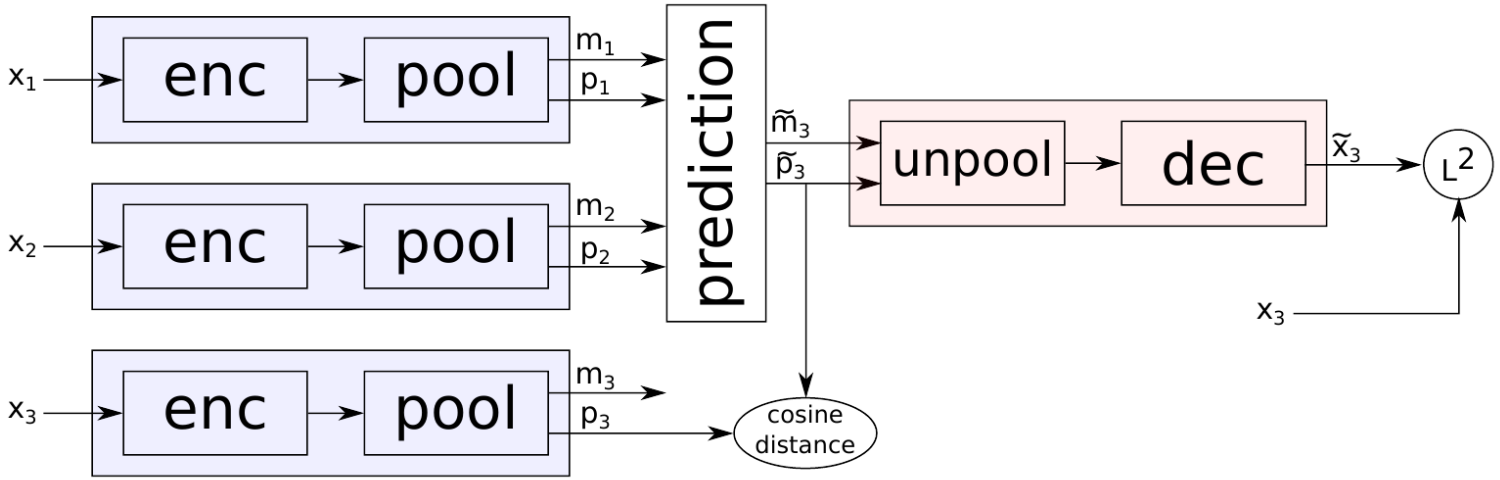


[Goroshin, Mathieu, LeCun arXiv:1506.03011]

- **Version 3: the world is unpredictable**
  - Add latent variable to compensate for that.
  - Minimize over them during training

$$\hat{z}_\delta^{t+1} = z^t + (W_1\delta) \odot \mathbf{a}\begin{bmatrix} z^t & z^{t-1} \end{bmatrix}^T$$

$$L = \min_\delta \|G_W(\hat{z}_\delta^{t+1}) - x^{t+1}\|_2^2 - \lambda \frac{(z^t - z^{t-1})^T(z^{t+1} - z^t)}{\|z^t - z^{t-1}\|\|z^{t+1} - z^t\|}$$



Algorithm (with latent variables)

Algorithm 1 Minibatch stochastic gradient descent training for prediction with uncertainty. The number of δ-gradient descent steps (k) is treated as a hyper-parameter.

for number of training epochs **do**
  Sample a mini-batch of temporal triplets $\{x^{t-1}, x^t, x^{t+1}\}$
  Set $\delta_0 = 0$
  Forward propagate x^{t-1}, x^t through the network and obtain the codes z^{t-1}, z^t and the prediction $\hat{x}_0^{t+1}$
  for i =1 to k **do**
    Compute the L^2 prediction error
    Back propagate the error through the decoder to compute the gradient $\frac{\partial L}{\partial \delta^{i-1}}$
    Update $\delta_i = \delta_{i-1} - \alpha \frac{\partial L}{\partial \delta^{i-1}}$
    Compute $\hat{z}_{\delta_i}^{t+1} = z^t + (W_1 \delta_i) \odot \mathbf{a}\left[z^t \quad z^{t-1}\right]^T$
    Compute $\hat{x}_i^{t+1} = G_W(z_{\delta^i}^{t+1})$
  end for
  Back propagate the full encoder/predictor loss from Equation 7 using δ_k, and update the weight matrices W and W_1
end for

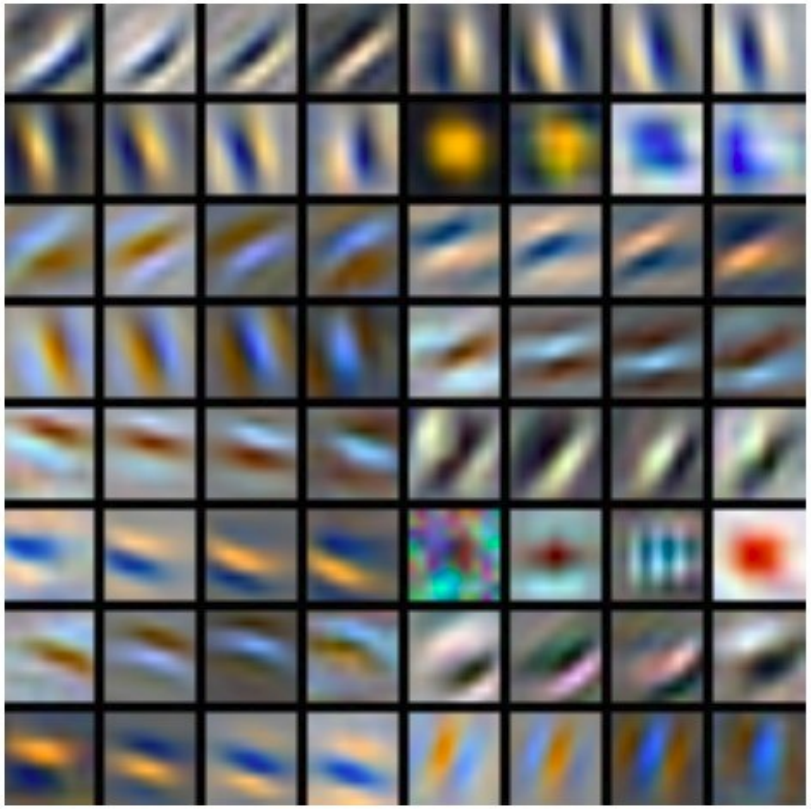
$$\hat{z}_\delta^{t+1} = z^t + (W_1 \delta) \odot \mathbf{a}\left[z^t \quad z^{t-1}\right]^T$$

$$L = \min_\delta \|G_W(\hat{z}_\delta^{t+1}) - x^{t+1}\|_2^2 - \lambda \frac{(z^t - z^{t-1})^T (z^{t+1} - z^t)}{\|z^t - z^{t-1}\| \|z^{t+1} - z^t\|}$$

Architectures

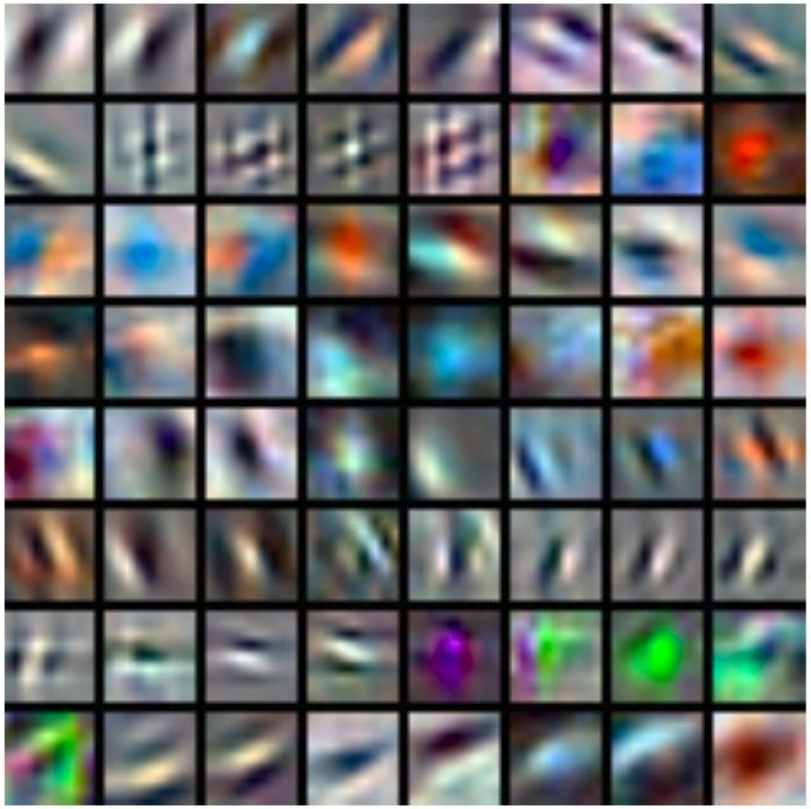
	Encoder	Prediction	Decoder
Shallow Architecture 1	Conv+ReLU $64 \times 9 \times 9$ Phase Pool 4	Average Mag. Linear Extrap. Phase	Conv $64 \times 9 \times 9$
Shallow Architecture 2	Conv+ReLU $64 \times 9 \times 9$ Phase Pool 4 stride 2	Average Mag. Linear Extrap. Phase	Conv $64 \times 9 \times 9$
Deep Architecture 1	Conv+ReLU $16 \times 9 \times 9$ Conv+ReLU $32 \times 9 \times 9$ FC+ReLU 8192×4096	None	FC+ReLU $\mathbf{8192} \times 8192$ Reshape $32 \times 16 \times 16$ SpatialPadding 8×8 Conv+ReLU $16 \times 9 \times 9$ SpatialPadding 8×8 Conv $1 \times 9 \times 9$
Deep Architecture 2	Conv+ReLU $16 \times 9 \times 9$ Conv+ReLU $32 \times 9 \times 9$ FC+ReLU 8192×4096	Linear Extrapolation	FC+ReLU 4096×8192 Reshape $32 \times 16 \times 16$ SpatialPadding 8×8 Conv+ReLU $16 \times 9 \times 9$ SpatialPadding 8×8 Conv $1 \times 9 \times 9$
Deep Architecture 3	Conv+ReLU $16 \times 9 \times 9$ Conv+ReLU $32 \times 9 \times 9$ FC+ReLU 8192×4096 Reshape $64 \times 8 \times 8$ Phase Pool 8×8	Average Mag. Linear Extrap. Phase	Unpool 8×8 FC+ReLU 4096×8192 Reshape $32 \times 16 \times 16$ SpatialPadding 8×8 Conv+ReLU $16 \times 9 \times 9$ SpatialPadding 8×8 Conv $1 \times 9 \times 9$

Filters: pooling over groups of 4 filters.



(a) Shallow Architecture 1

Non-overlapping pooling
4x4x4 (feat,x,y)



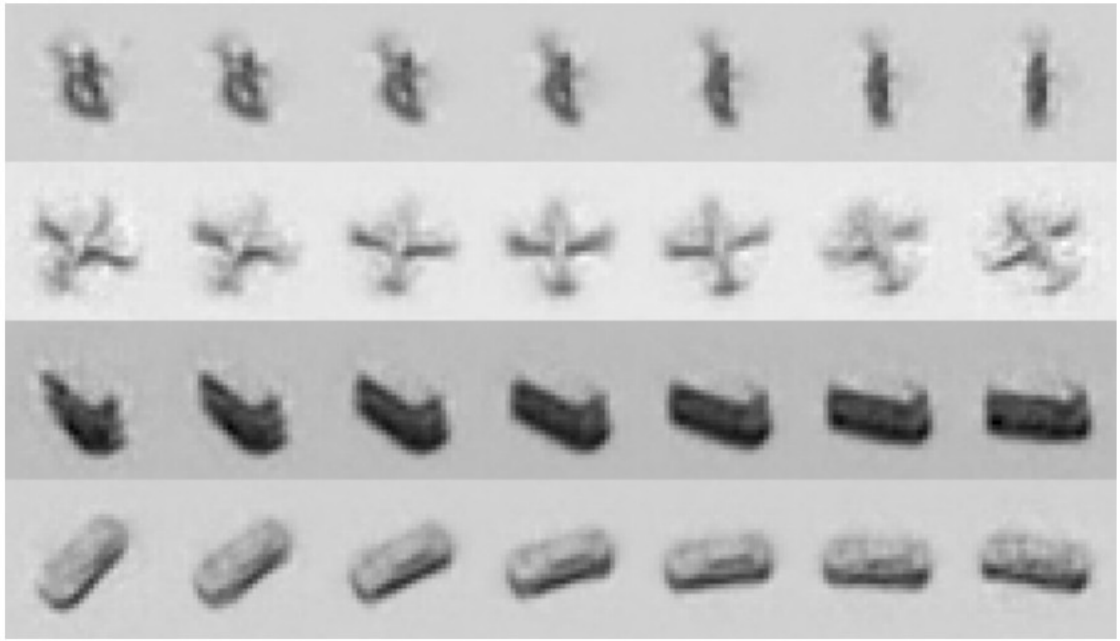
(b) Shallow Architecture 2

Overlapping pooling
4x4x4 (feat,x,y)
Stride 2 over features

Image interpolation in feature space: deterministic world

Y LeCun

- No phase pooling
- No curvature regularization



- With phase pooling
- With curvature regularization

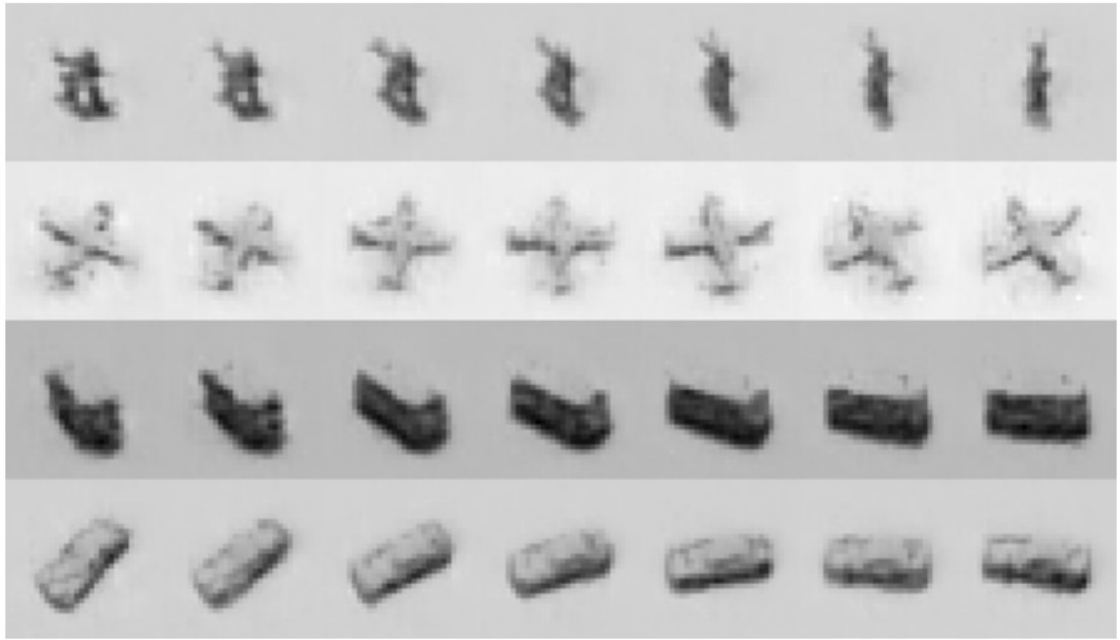


Image interpolation in feature space: deterministic world

- Image interpolation with the basic model (siamese net)

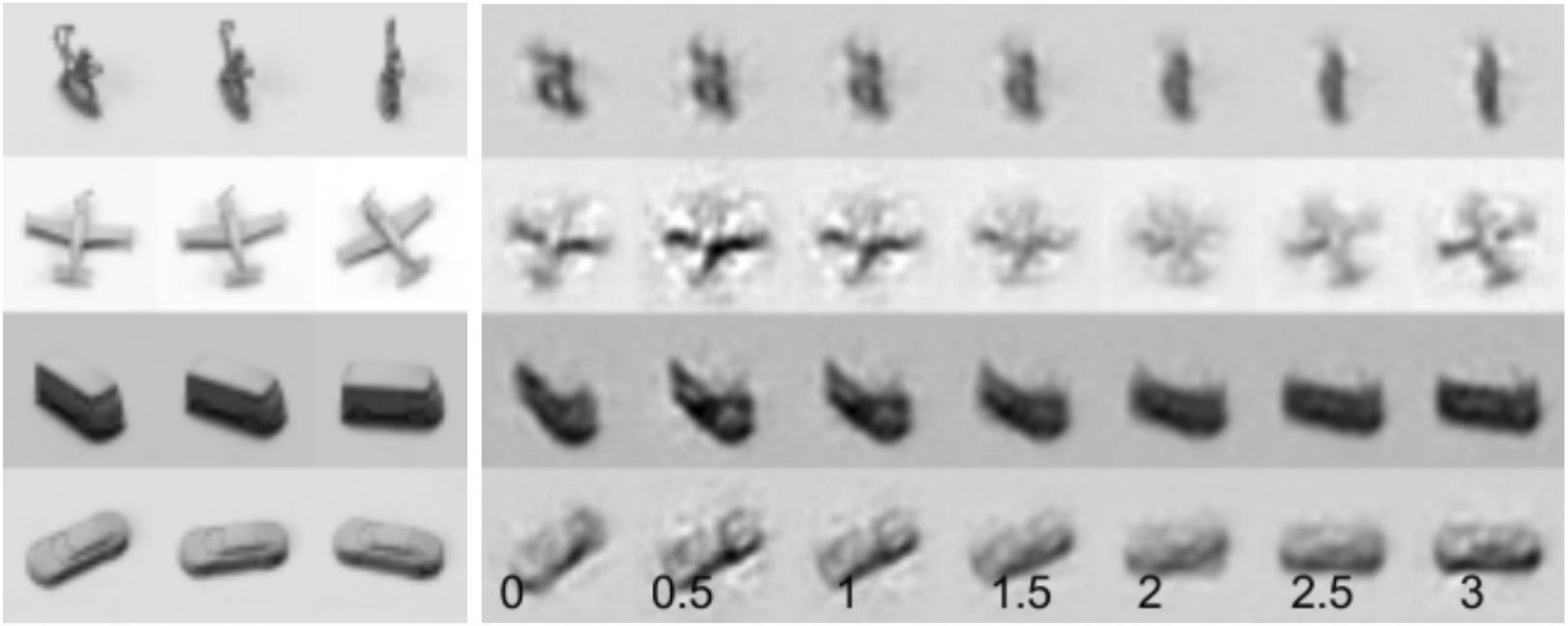
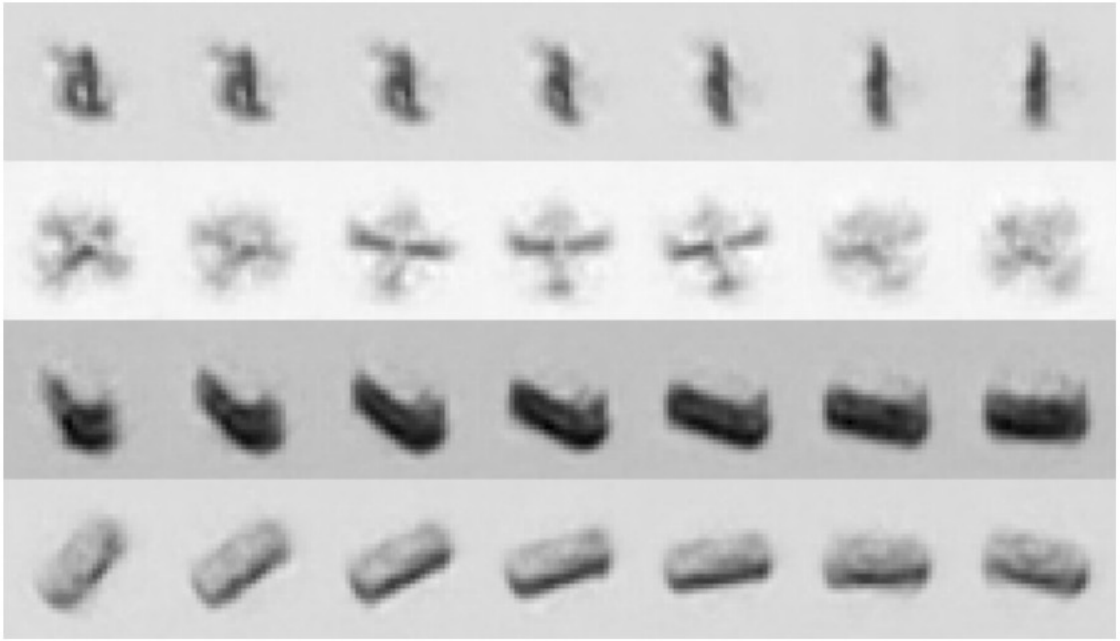


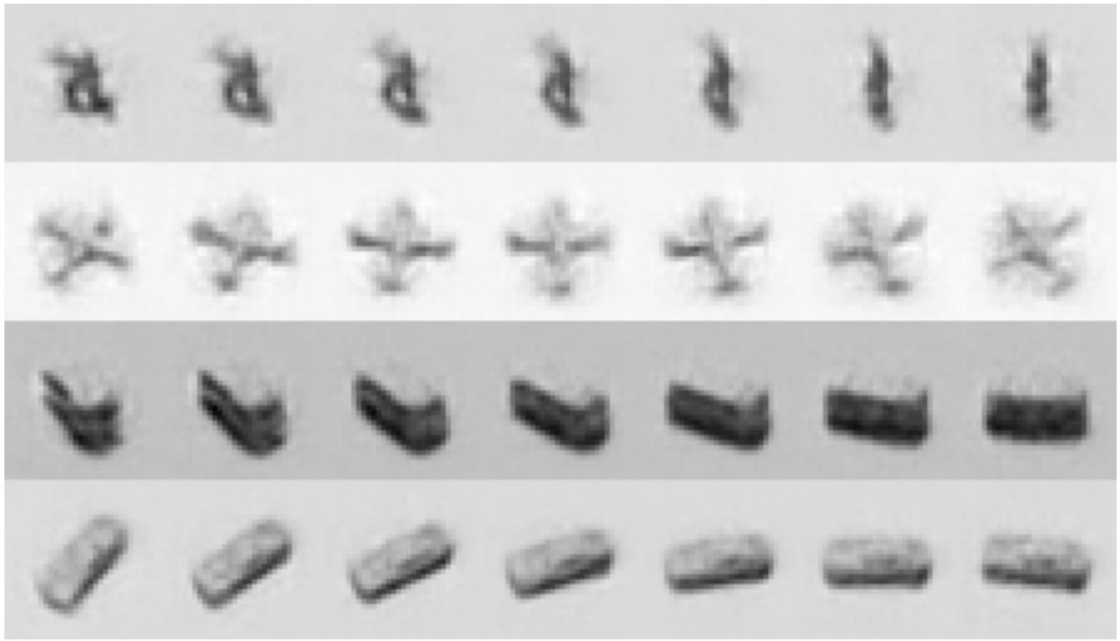
Image interpolation in feature space: unpredictable world

Y LeCun

- Phase interpolation
- No latent variable



- Phase interpolation
- With latent variable




The End