Convolutional Networks, Image Recognition

Yann LeCun

The Courant Institute of Mathematical Sciences

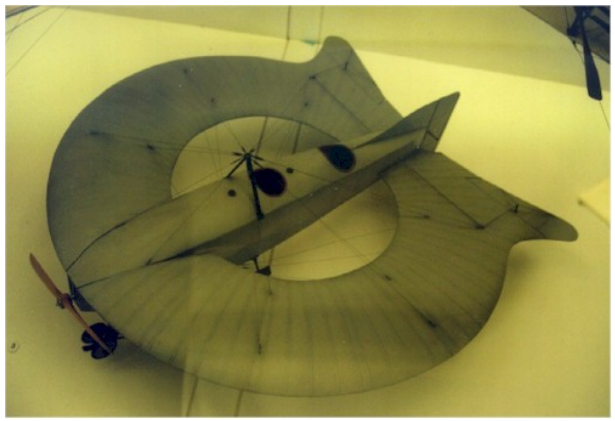
New York University

Collaborators:

Marc'Aurelio Ranzato, Sumit Chopra, Fu-Jie Huang, Y-Lan Boureau

The Challenges of Pattern Recognition, Computer Vision, and Visual Neuroscience

- **How do we learn "invariant representations"?**
  - From the image of an airplane, how do we extract a representation that is invariant to pose, illumination, background, clutter, object instance....
  - How can a human (or a machine) learn those representations by just looking at the world?
- **How can we learn visual categories from just a few examples?**
  - I don't need to see many airplanes before I can recognize every airplane (even really weird ones)

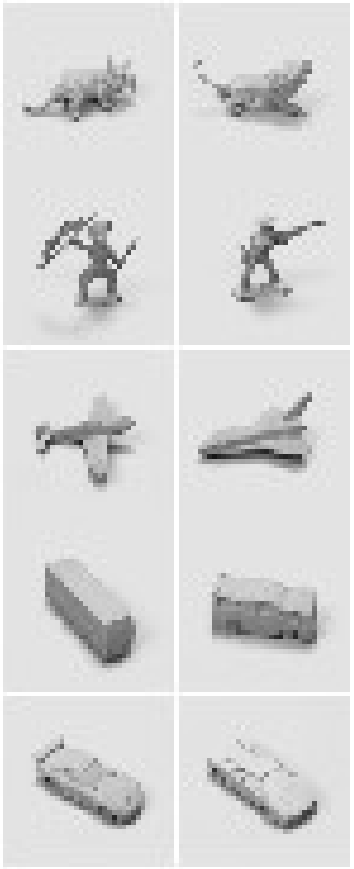




Invariance

- **The appearance of an object (in terms of pixels) changes considerably under changes of pose, illumination, clutter, and occlusions.**
- **Two instance of the same category may have widely differing shapes and appearances**
  - An airliner and a fighter plane, a person standing and another one kneeling,...
- **Template-based methods are doomed because the number of templates necessary to cover the space of variations grows exponentially with the number of dimensions of the variations.**

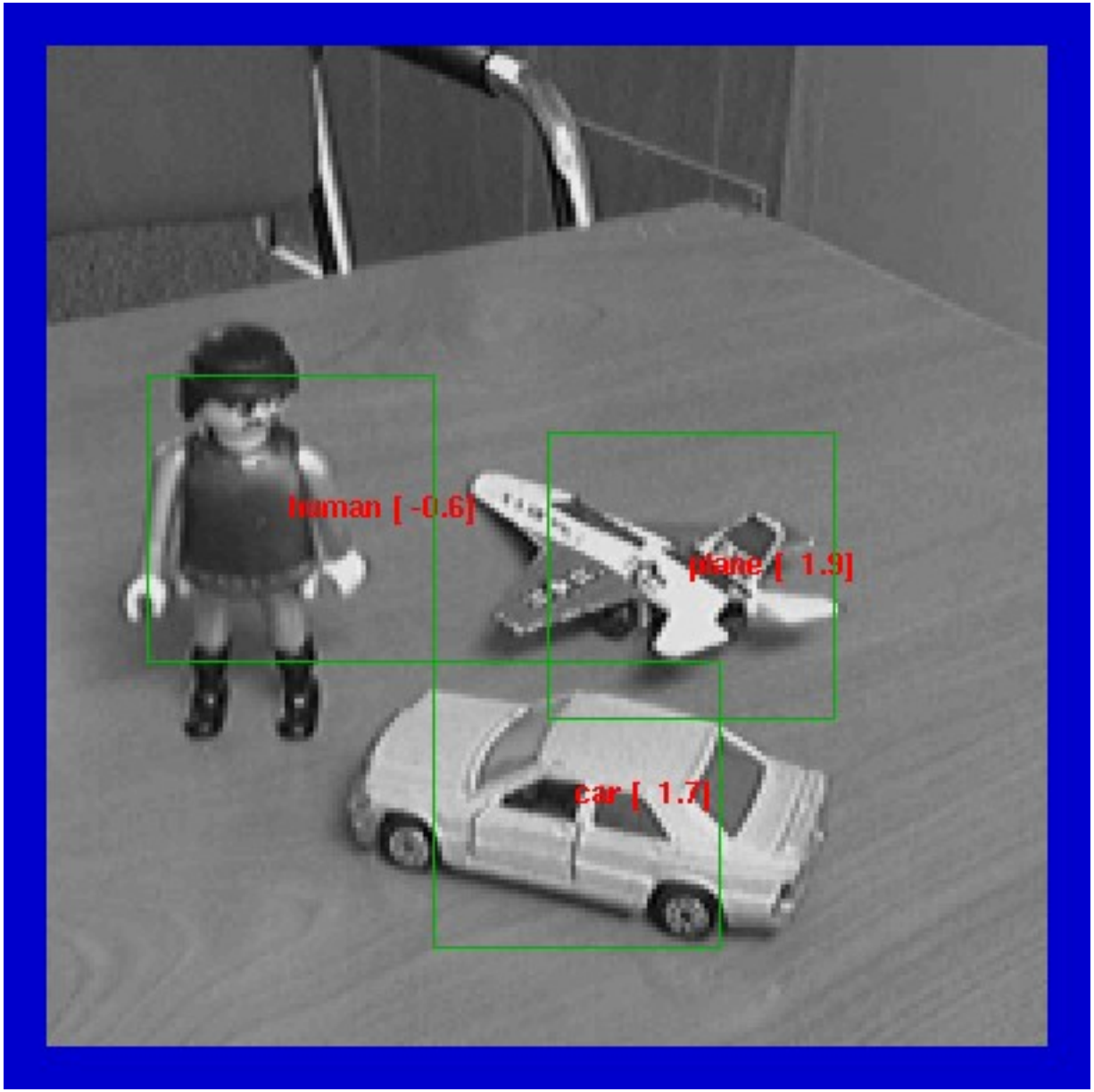
Generic Object Recognition

- **Generic Object Recognition** is the problem of detecting and classifying objects into generic categories such as "cars", "trucks", "airplanes", "animals", or "human figures"
- **Appearances are highly variable within a category** because of shape variation, position in the visual field, scale, viewpoint, illumination, albedo, texture, background clutter, and occlusions.
- **Learning invariant representations is key.**
- **Understanding the neural mechanism behind invariant recognition is one of the main goals of Visual Neuroscience.**



What we want to achieve

- **recognition from shape:** color, texture, and distinctive local features may be useful, but they merely allow us to sweep the real problems under the rug.
- **Full invariance to viewpoint, illumination, clutter, occlusions.**



The Traditional Architecture for Recognition

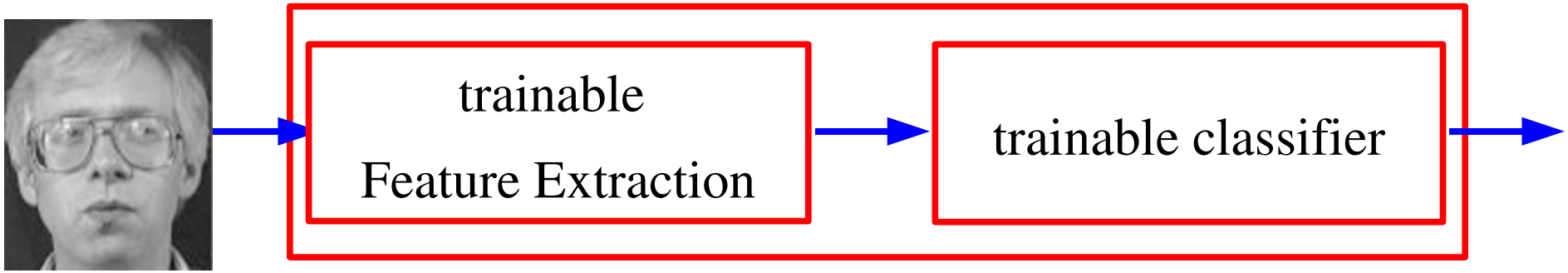
Pre-processing / Feature Extraction

Trainable Classifier

this part is mostly hand-crafted

- The raw input is pre-processed through a hand-crafted feature extractor
- The trainable classifier is often generic (task independent)

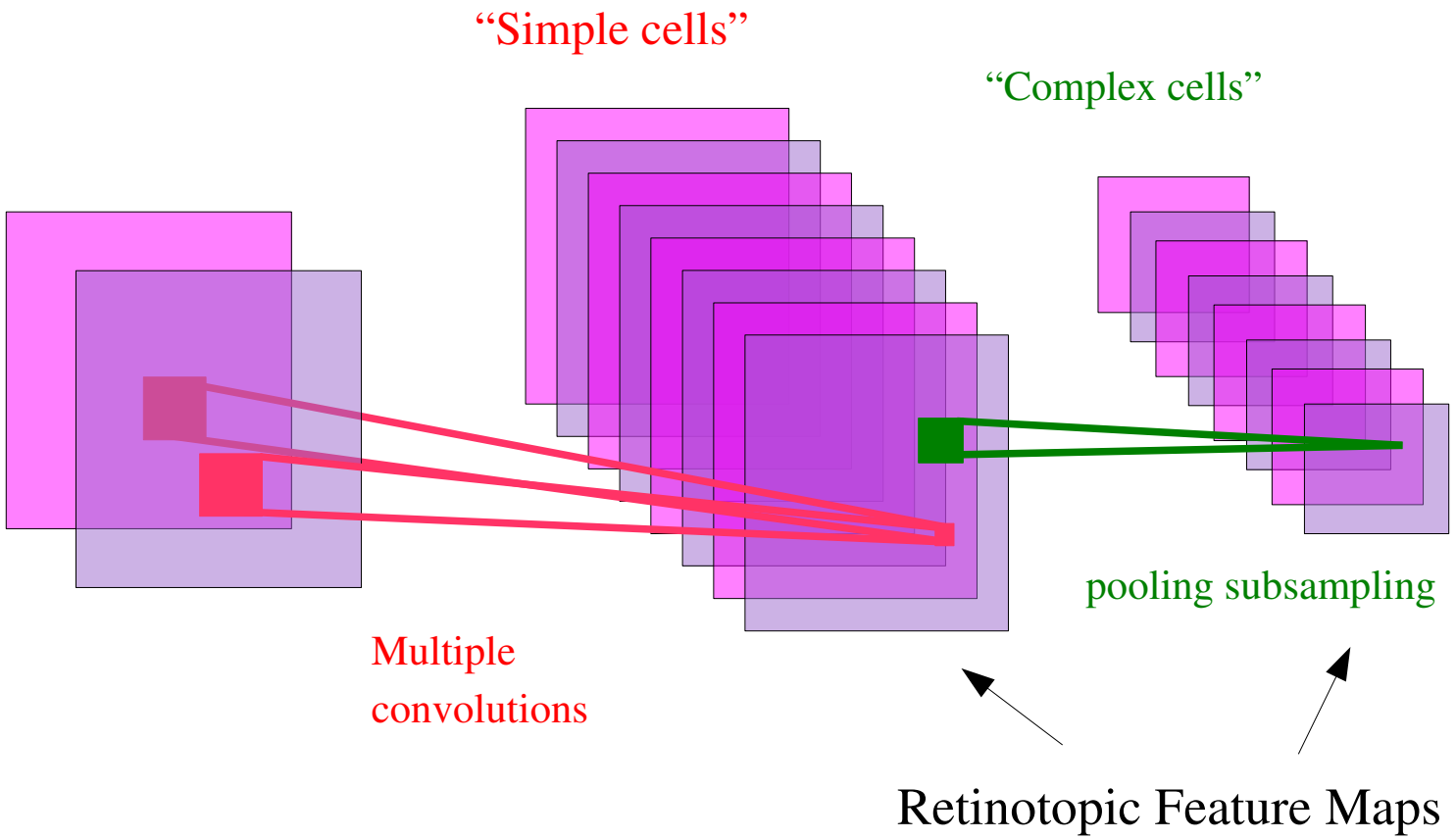
End-to-End Learning



- The entire system is integrated and trainable "end-to-end".
- In some of the models presented here, there will be no discernible difference between the feature extractor and the classifier.
- We can embed general prior knowledge about images into the architecture of the system.

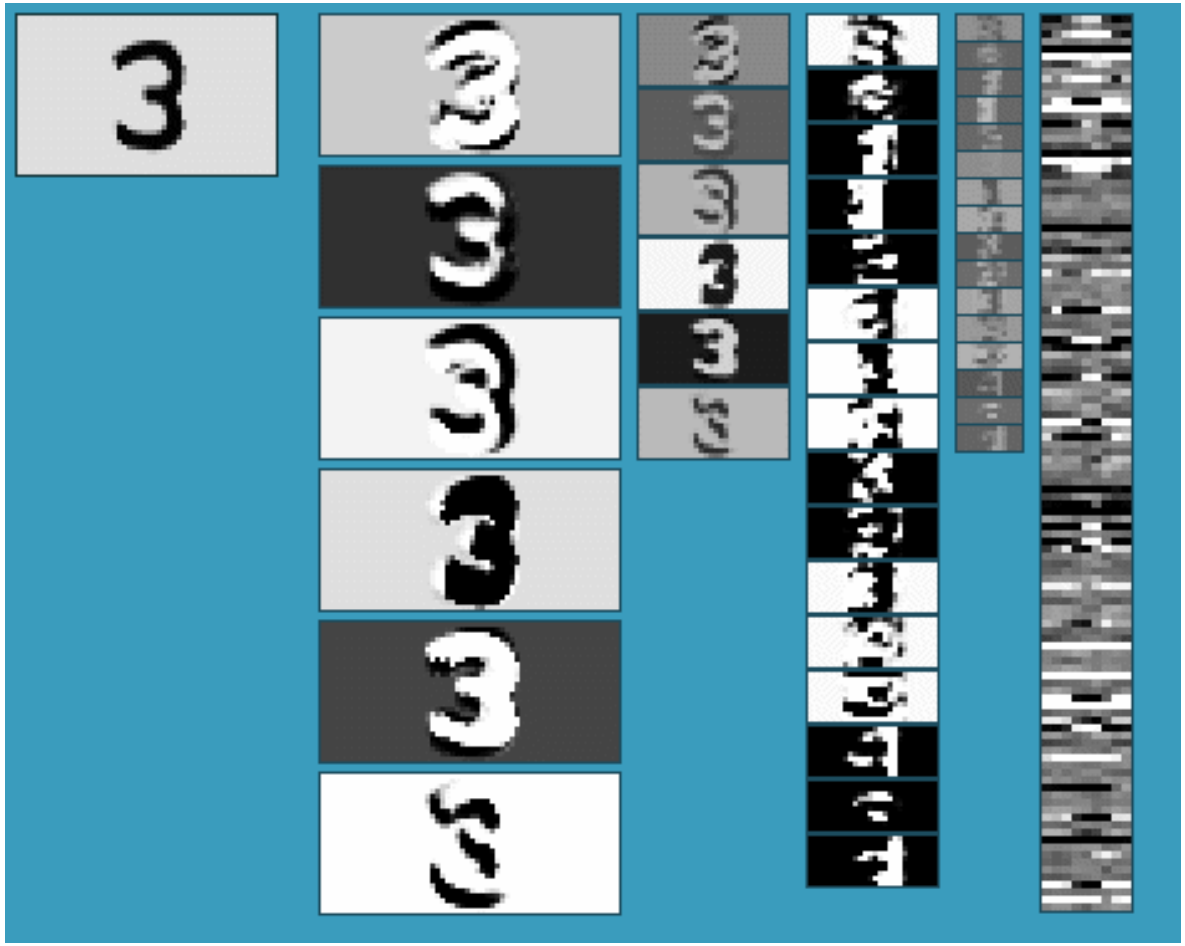
An Old Idea for Local Shift Invariance

- **[Hubel & Wiesel 1962]: architecture of the cat's visual cortex**
  - simple cells detect local features
  - complex cells "pool" the outputs of simple cells within a retinotopic neighborhood.



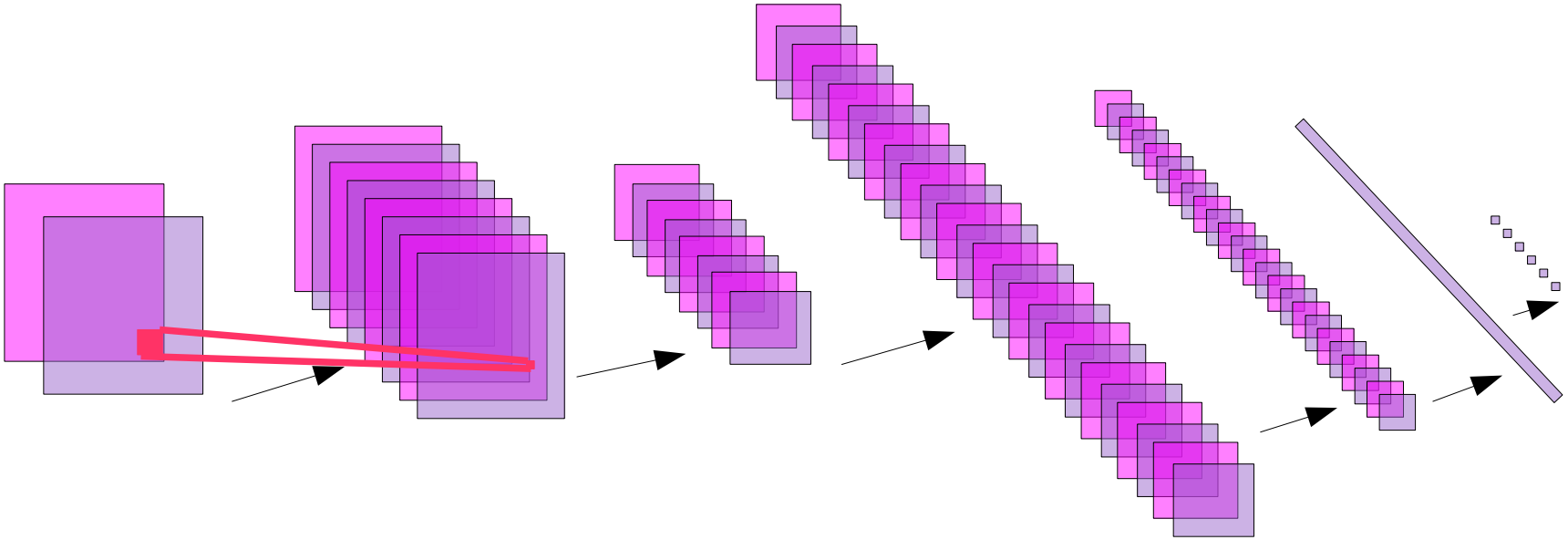
The Multistage Hubel-Wiesel Architecture

- **Building a complete artificial vision system:**
  - Stack multiple stages of simple cells / complex cells layers
  - Higher stages compute more global, more invariant features
  - Stick a classification layer on top
  - [Fukushima 1971-1982]
    - neocognitron
  - [LeCun 1988-2007]
    - convolutional net
  - [Poggio 2002-2006]
    - HMAX
  - [Ullman 2002-2006]
    - fragment hierarchy
  - [Lowe 2006]
    - HMAX



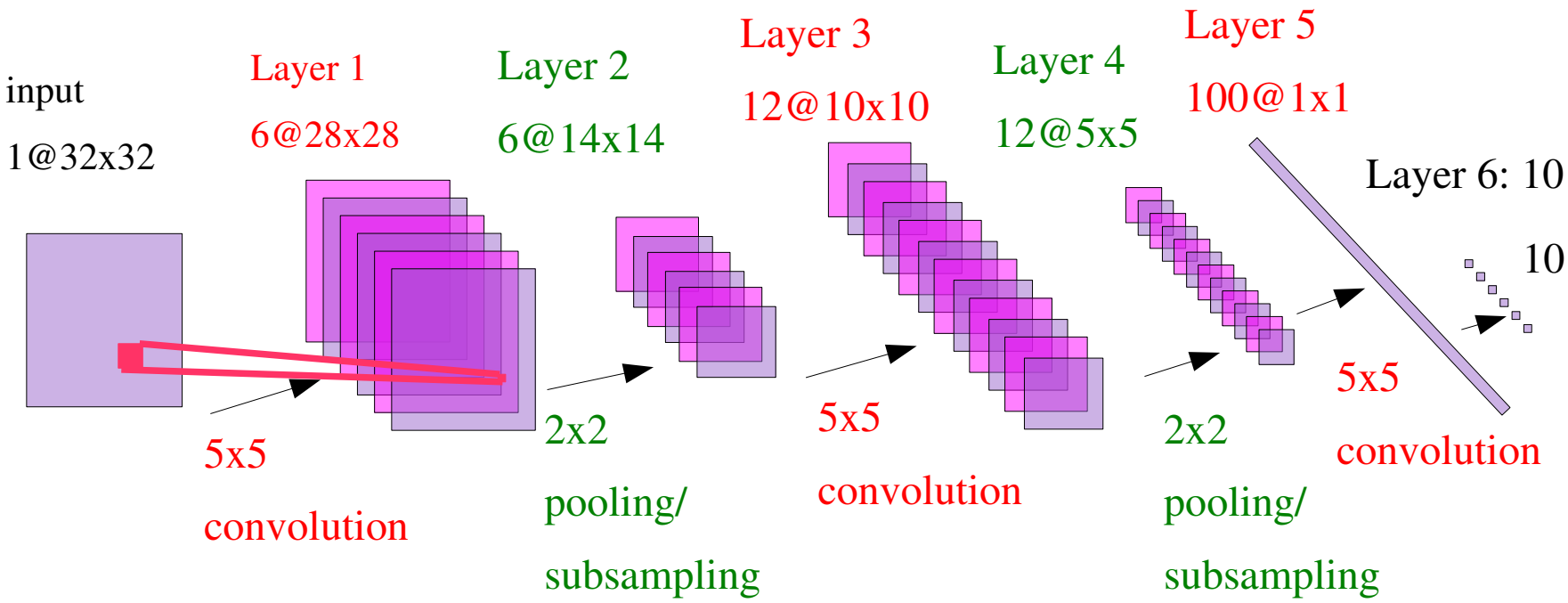
- **QUESTION: How do we find (or learn) the filters?**

Convolutional Network



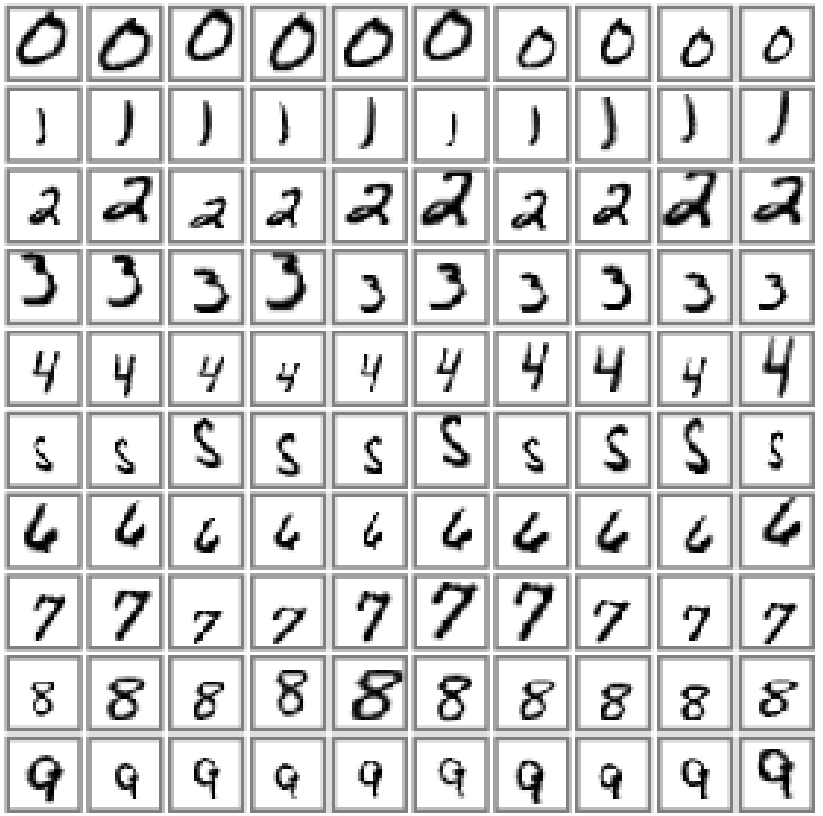
- **Hierarchical/multilayer:** features get progressively more global, invariant, and numerous
- **dense features:** features detectors applied everywhere (no interest point)
- **broadly tuned (possibly invariant) features:** sigmoid units are on half the time.
- **Global discriminative training:** The whole system is trained "end-to-end" with a gradient-based method to minimize a global loss function
- **Integrates segmentation, feature extraction, and invariant classification in one fell swoop.**

Convolutional Net Architecture



- **Convolutional net for handwriting recognition** (400,000 synapses)
- Convolutional layers (simple cells): all units in a feature plane share the same weights
- Pooling/subsampling layers (complex cells): for invariance to small distortions.
- **Supervised gradient-descent learning using back-propagation**
- **The entire network is trained end-to-end. All the layers are trained simultaneously**.

MNIST Handwritten Digit Dataset



- Handwritten Digit Dataset MNIST: 60,000 training samples, 10,000 test samples

Results on MNIST Handwritten Digits

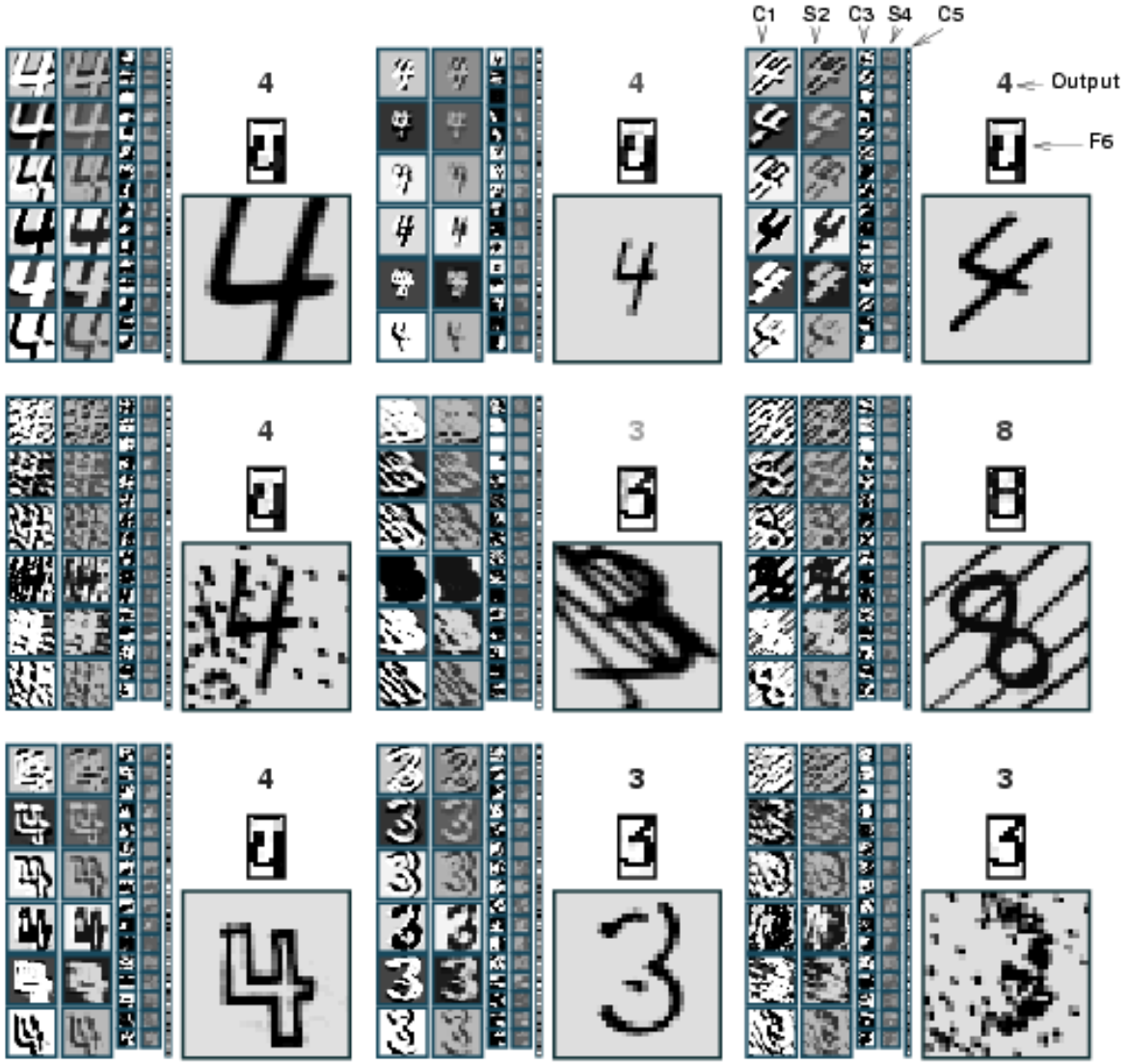
CLASSIFIER	DEFORMATION	PREPROCESSING	ERROR (%)	Reference
linear classifier (1-layer NN)		none	12.00	LeCun et al. 1998
linear classifier (1-layer NN)		deskewing	8.40	LeCun et al. 1998
pairwise linear classifier		deskewing	7.60	LeCun et al. 1998
K-nearest-neighbors, (L2)		none	3.09	Kenneth Wilder, U. Chicago
K-nearest-neighbors, (L2)		deskewing	2.40	LeCun et al. 1998
K-nearest-neighbors, (L2)		deskew, clean, blur	1.80	Kenneth Wilder, U. Chicago
K-NN L3, 2 pixel jitter		deskew, clean, blur	1.22	Kenneth Wilder, U. Chicago
K-NN, shape context matching		shape context feature	0.63	Belongie et al. IEEE PAMI 2002
40 PCA + quadratic classifier		none	3.30	LeCun et al. 1998
1000 RBF + linear classifier		none	3.60	LeCun et al. 1998
K-NN, Tangent Distance		subsamp 16x16 pixels	1.10	LeCun et al. 1998
SVM, Gaussian Kernel		none	1.40	
SVM deg 4 polynomial		deskewing	1.10	LeCun et al. 1998
Reduced Set SVM deg 5 poly		deskewing	1.00	LeCun et al. 1998
Virtual SVM deg-9 poly	Affine	none	0.80	LeCun et al. 1998
V-SVM, 2-pixel jittered		none	0.68	DeCoste and Scholkopf, MLJ 2002
V-SVM, 2-pixel jittered		deskewing	0.56	DeCoste and Scholkopf, MLJ 2002
2-layer NN, 300 HU, MSE		none	4.70	LeCun et al. 1998
2-layer NN, 300 HU, MSE,	Affine	none	3.60	LeCun et al. 1998
2-layer NN, 300 HU		deskewing	1.60	LeCun et al. 1998
3-layer NN, 500+150 HU		none	2.95	LeCun et al. 1998
3-layer NN, 500+150 HU	Affine	none	2.45	LeCun et al. 1998
3-layer NN, 500+300 HU, CE, reg		none	1.53	Hinton, unpublished, 2005
2-layer NN, 800 HU, CE		none	1.60	Simard et al., ICDAR 2003
2-layer NN, 800 HU, CE	Affine	none	1.10	Simard et al., ICDAR 2003
2-layer NN, 800 HU, MSE	Elastic	none	0.90	Simard et al., ICDAR 2003
2-layer NN, 800 HU, CE	Elastic	none	0.70	Simard et al., ICDAR 2003
Convolutional net LeNet-1		subsamp 16x16 pixels	1.70	LeCun et al. 1998
Convolutional net LeNet-4		none	1.10	LeCun et al. 1998
Convolutional net LeNet-5,		none	0.95	LeCun et al. 1998
Conv. net LeNet-5,	Affine	none	0.80	LeCun et al. 1998
Boosted LeNet-4	Affine	none	0.70	LeCun et al. 1998
Conv. net, CE	Affine	none	0.60	Simard et al., ICDAR 2003
Comv net, CE	Elastic	none	0.40	Simard et al., ICDAR 2003

Some Results on MNIST (from raw images: no preprocessing)

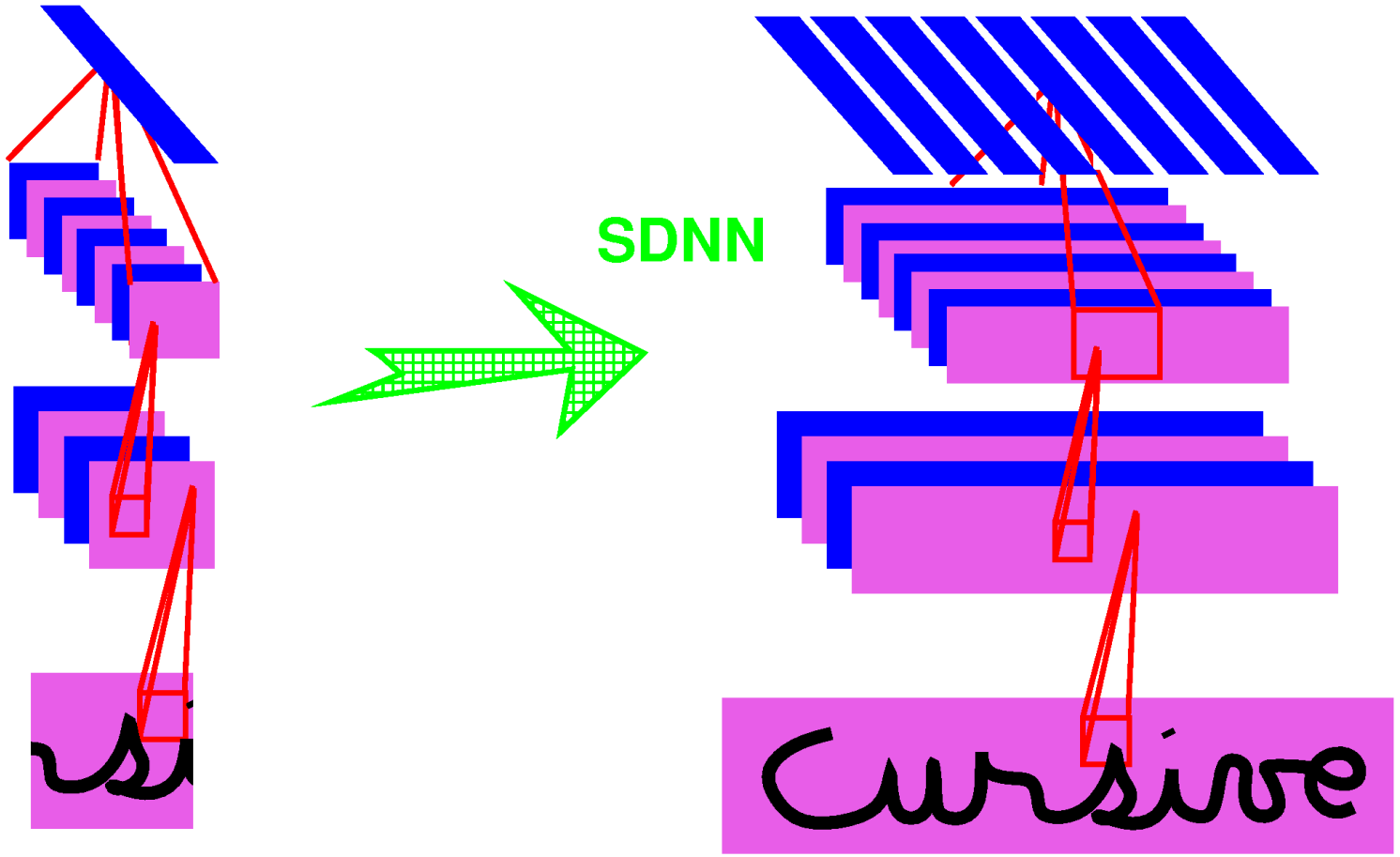
CLASSIFIER	DEFORMATION	ERROR	Reference
Knowledge-free methods (a fixed permutation of the pixels would make no difference)			
2-layer NN, 800 HU, CE		1.60	Simard et al., ICDAR 2003
3-layer NN, 500+300 HU, CE, reg		1.53	Hinton, in press, 2005
SVM, Gaussian Kernel		1.40	Cortes 92 + Many others
???		0.95	
Convolutional nets			
Convolutional net LeNet-5,		0.80	Ranzato et al. NIPS 2006
Convolutional net LeNet-6,		0.70	Ranzato et al. NIPS 2006
???		0.60	
Training set augmented with Affine Distortions			
2-layer NN, 800 HU, CE	Affine	1.10	Simard et al., ICDAR 2003
Virtual SVM deg-9 poly	Affine	0.80	Scholkopf
Convolutional net, CE	Affine	0.60	Simard et al., ICDAR 2003
Training et augmented with Elastic Distortions			
2-layer NN, 800 HU, CE	Elastic	0.70	Simard et al., ICDAR 2003
Convolutional net, CE	Elastic	0.40	Simard et al., ICDAR 2003
???		0.39	

Note: some groups have obtained good results with various amounts of preprocessing: [deCoste and Schoelkopf] get 0.56% with an SVM on deskewed images; [Belongie] get 0.63% with "shape context" features; [CENPARMI] get below 0.4% with features and SVM; [Liu] get 0.42% with features and SVM.

Invariance and Robustness to Noise

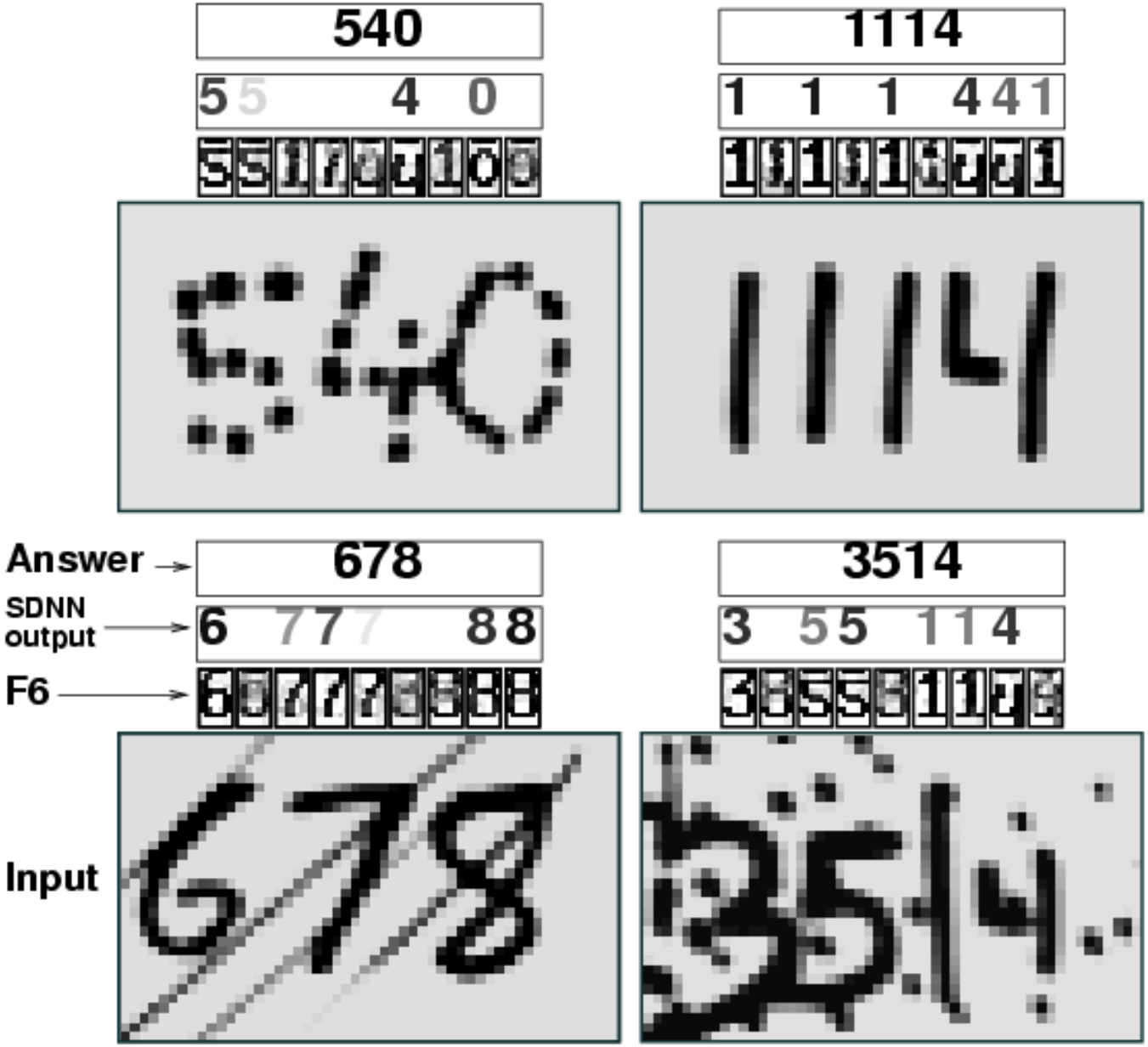


Recognizing Multiple Characters with Replicated Nets



Recognizing Multiple Characters with Replicated Nets

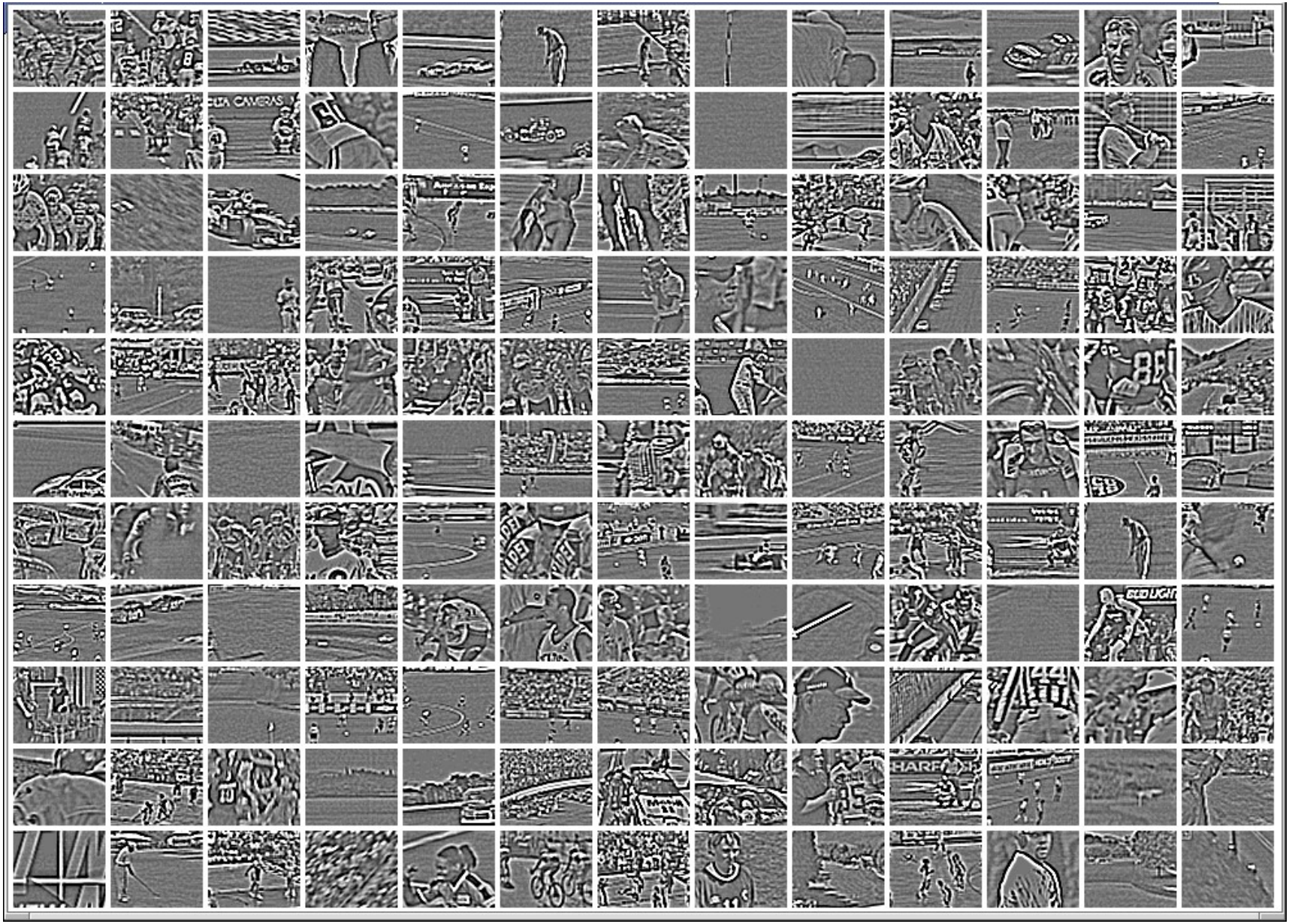
Handwriting Recognition



TV sport categorization (with Alex Niculescu, Cornell)

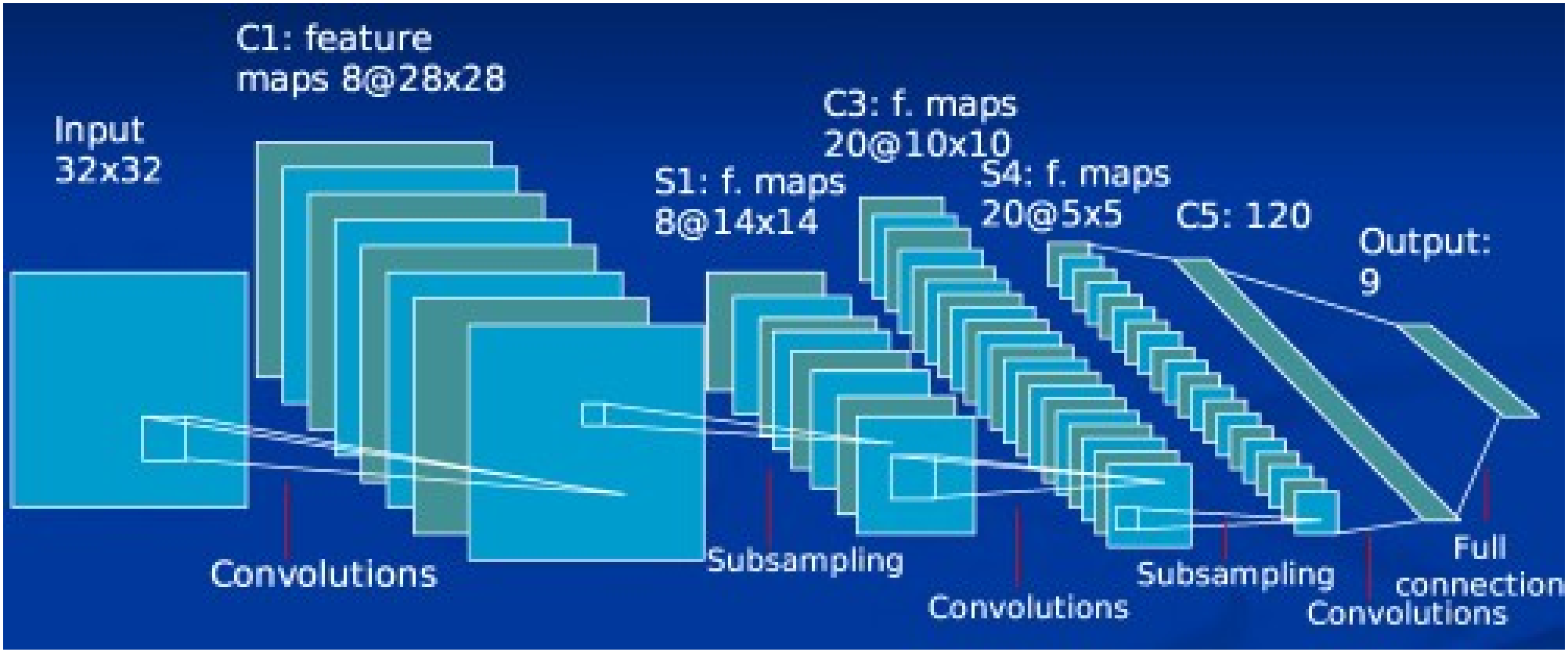
- **Classifying TV sports snapshots into 7 categories: auto racing, baseball, basketball, bicycle, golf, soccer, football.**
- **123,900 training images (300 sequence with 59 frames for each sport)**
- **82,600 test images (200 sequences with 59 frames for each sport)**
- **Preprocessing: convert to YUV, high-pass filter the Y component, crop, subsample to 72x60 pixels**
- **Results:**
  - frame-level accuracy: 61% correct
  - Sequence-level accuracy 68% correct (simple voting scheme).

TV sport categorization (with Alex Niculescu, Cornell)



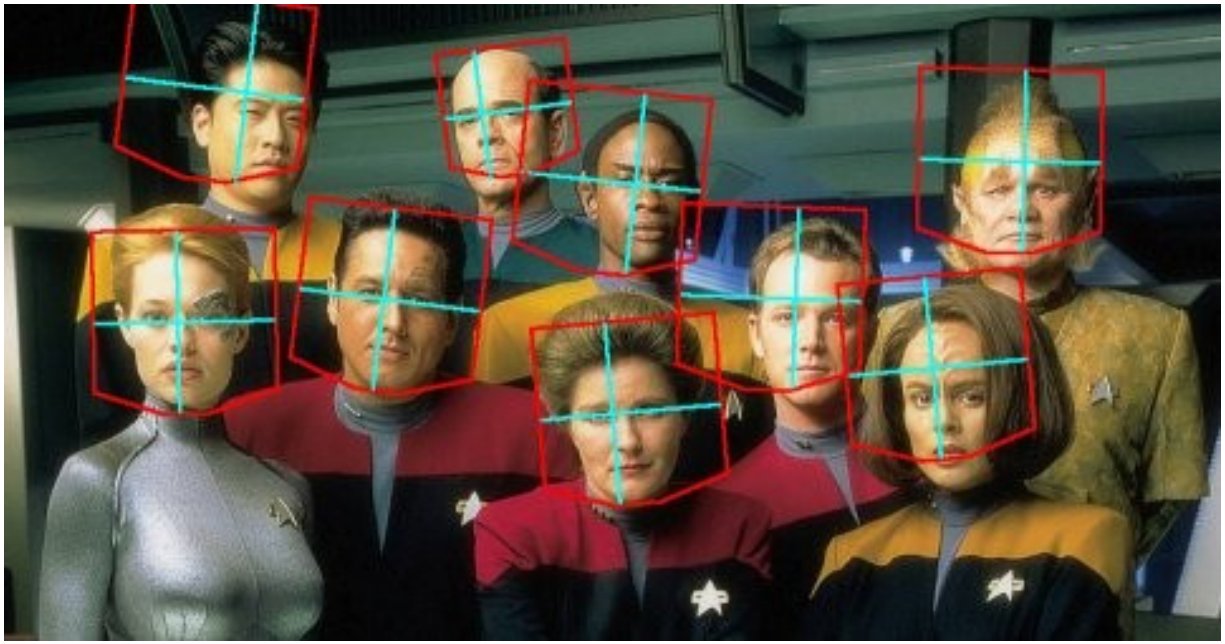
Face Detection and Pose Estimation with Convolutional Nets

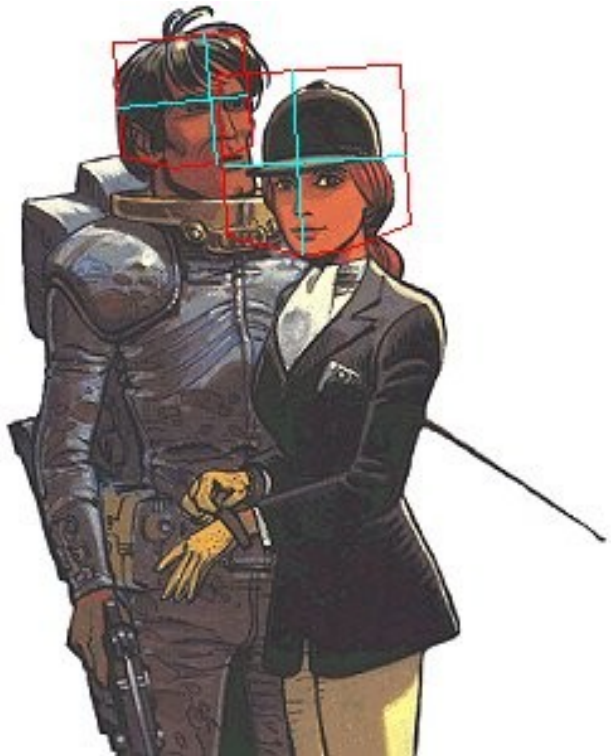
- **Training:** 52,850, 32x32 grey-level images of faces, 52,850 non-faces.
- **Each sample:** used 5 times with random variation in scale, in-plane rotation, brightness and contrast.
- **2nd phase:** half of the initial negative set was replaced by false positives of the initial version of the detector .



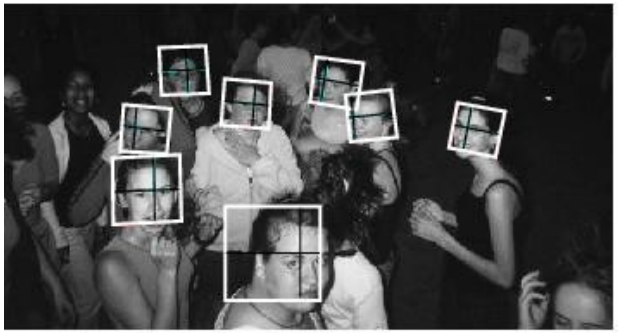
Face Detection: Results

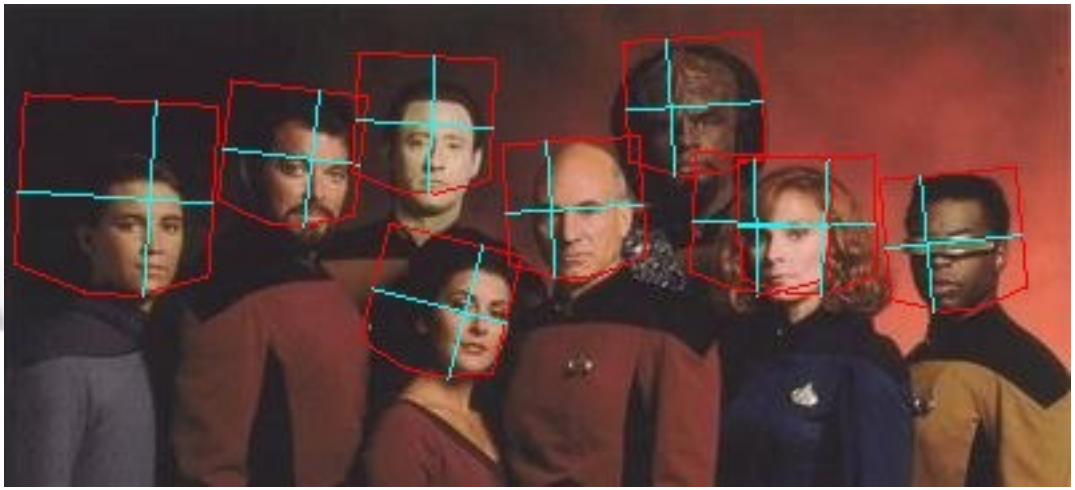
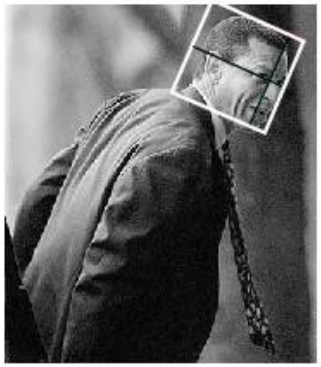
Data Set->	TILTED		PROFILE		MIT+CMU	
False positives per image->	4.42	26.9	0.47	3.36	0.5	1.28
Our Detector	90%	97%	67%	83%	83%	88%
Jones & Viola (tilted)	90%	95%	x		x	
Jones & Viola (profile)	x		70%	83%	x	



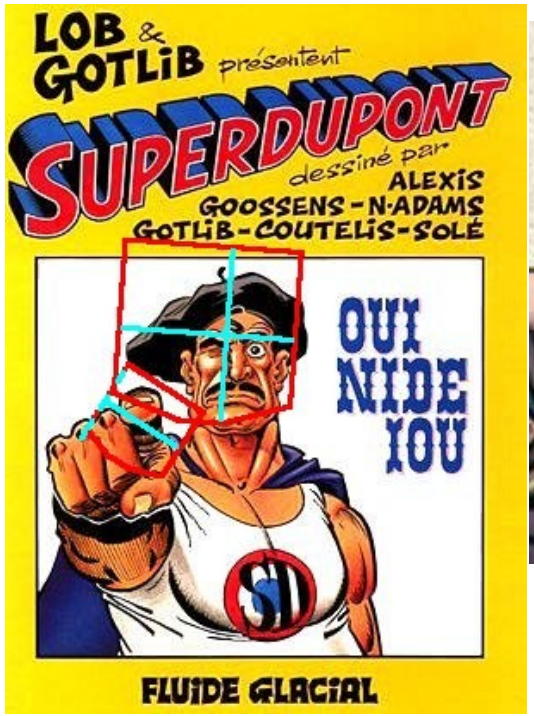


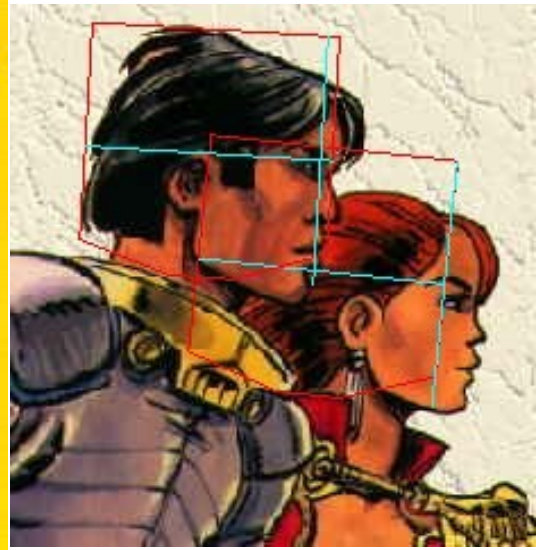
Face Detection and Pose Estimation: Results



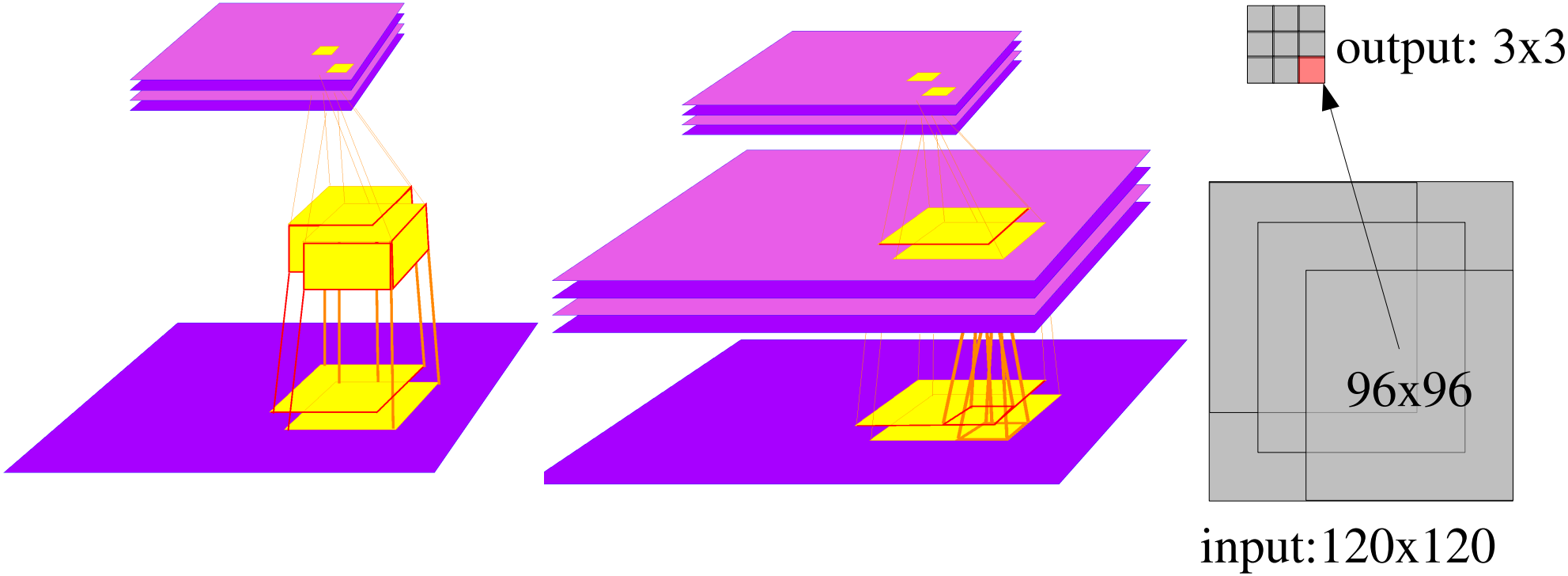






Face Detection with a Convolutional Net

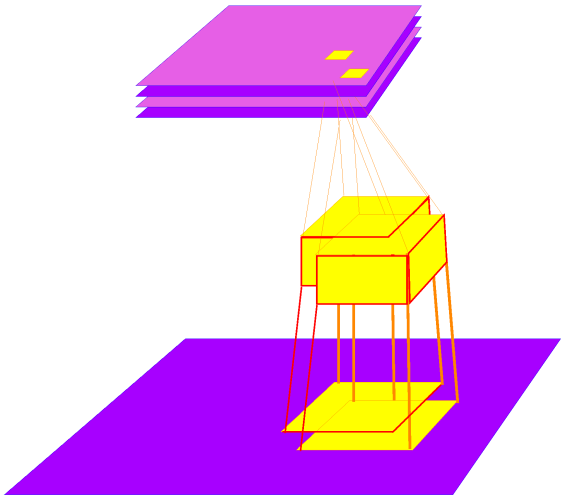
Applying a ConvNet on Sliding Windows is Very Cheap!

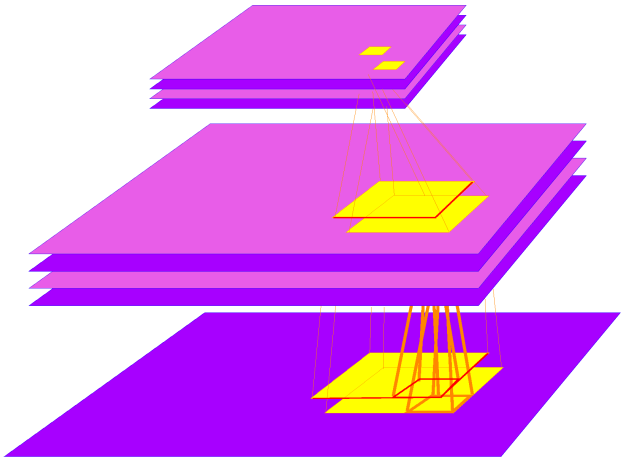


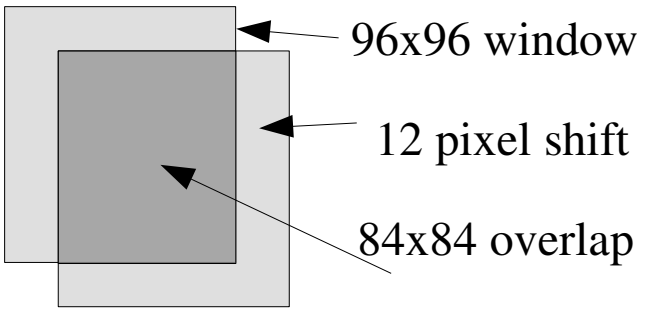
- Traditional Detectors/Classifiers must be applied to every location on a large input image, at multiple scales.
- Convolutional nets can replicated over large images very cheaply.
- The network is applied to multiple scales spaced by 1.5.

Building a Detector/Recognizer: Replicated Convolutional Nets

- Computational cost for replicated convolutional net:
  - 96x96 -> 4.6 million multiply-accumulate operations
  - 120x120 -> 8.3 million multiply-accumulate operations
  - 240x240 -> 47.5 million multiply-accumulate operations
  - 480x480 -> 232 million multiply-accumulate operations
- Computational cost for a non-convolutional detector of the same size, applied every 12 pixels:
  - 96x96 -> 4.6 million multiply-accumulate operations
  - 120x120 -> 42.0 million multiply-accumulate operations
  - 240x240 -> 788.0 million multiply-accumulate operations
  - 480x480 -> 5,083 million multiply-accumulate operations



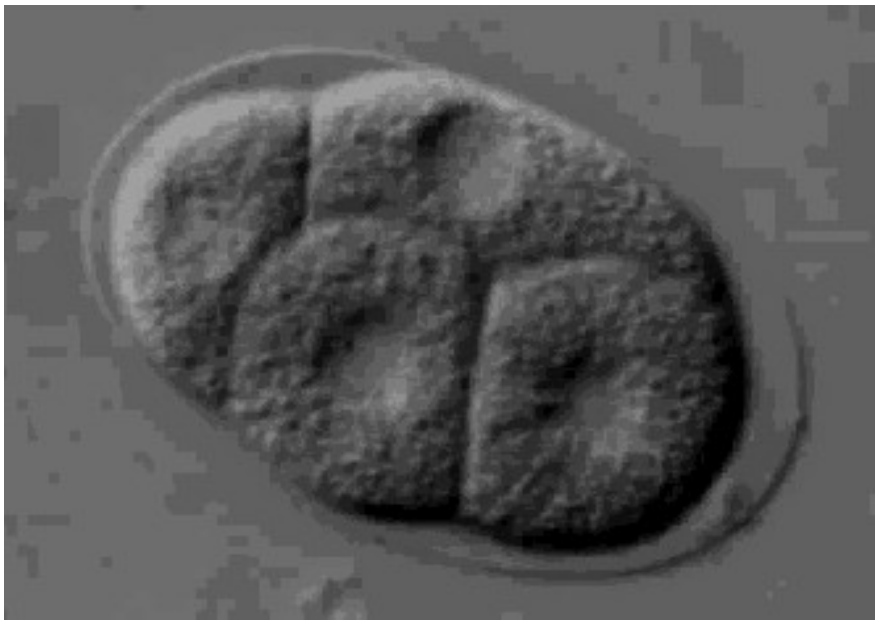


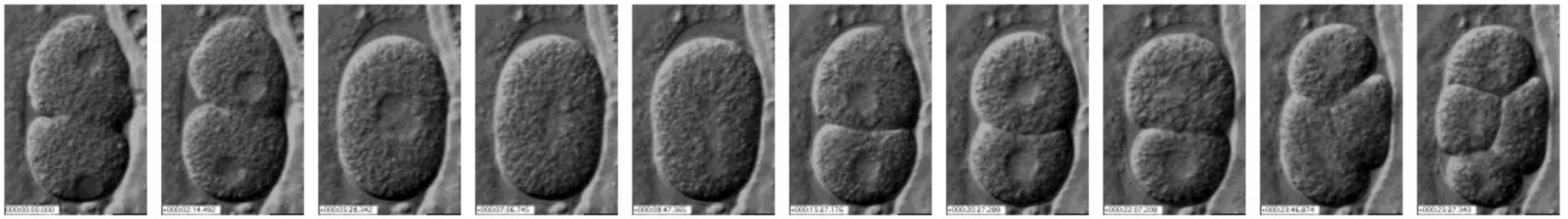


C. Elegans Embryo Phenotyping

[Ning et al. IEEE Trans. Image Processing, Nov 2005]

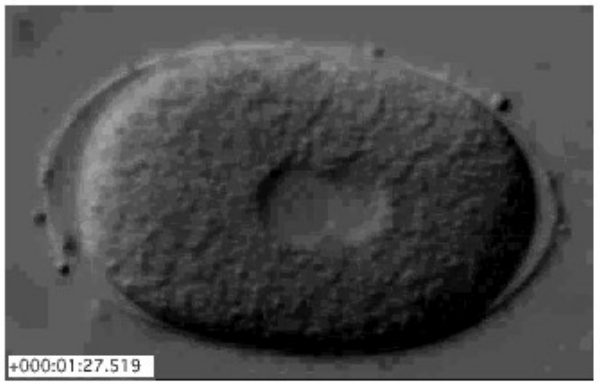
- **Analyzing results for Gene Knock-Out Experiments**

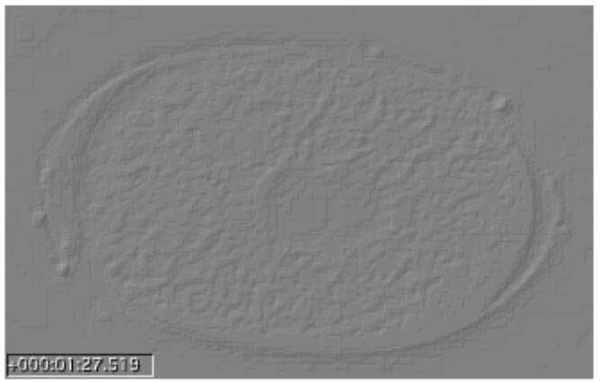


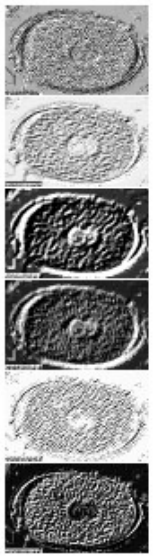


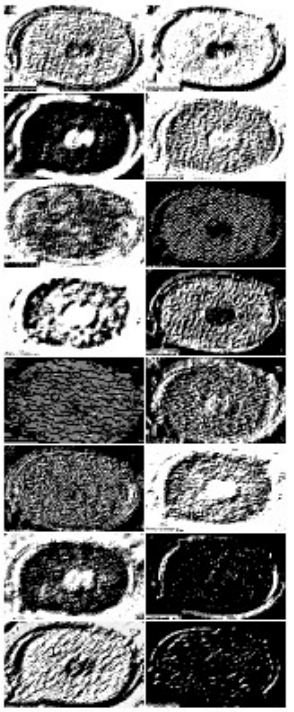
C. Elegans Embryo Phenotyping

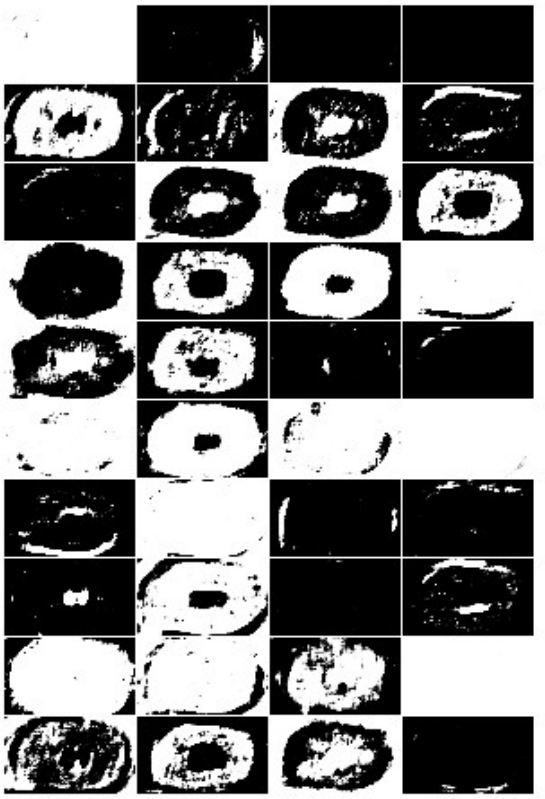
Analyzing results for Gene Knock-Out Experiments

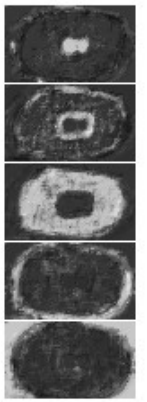




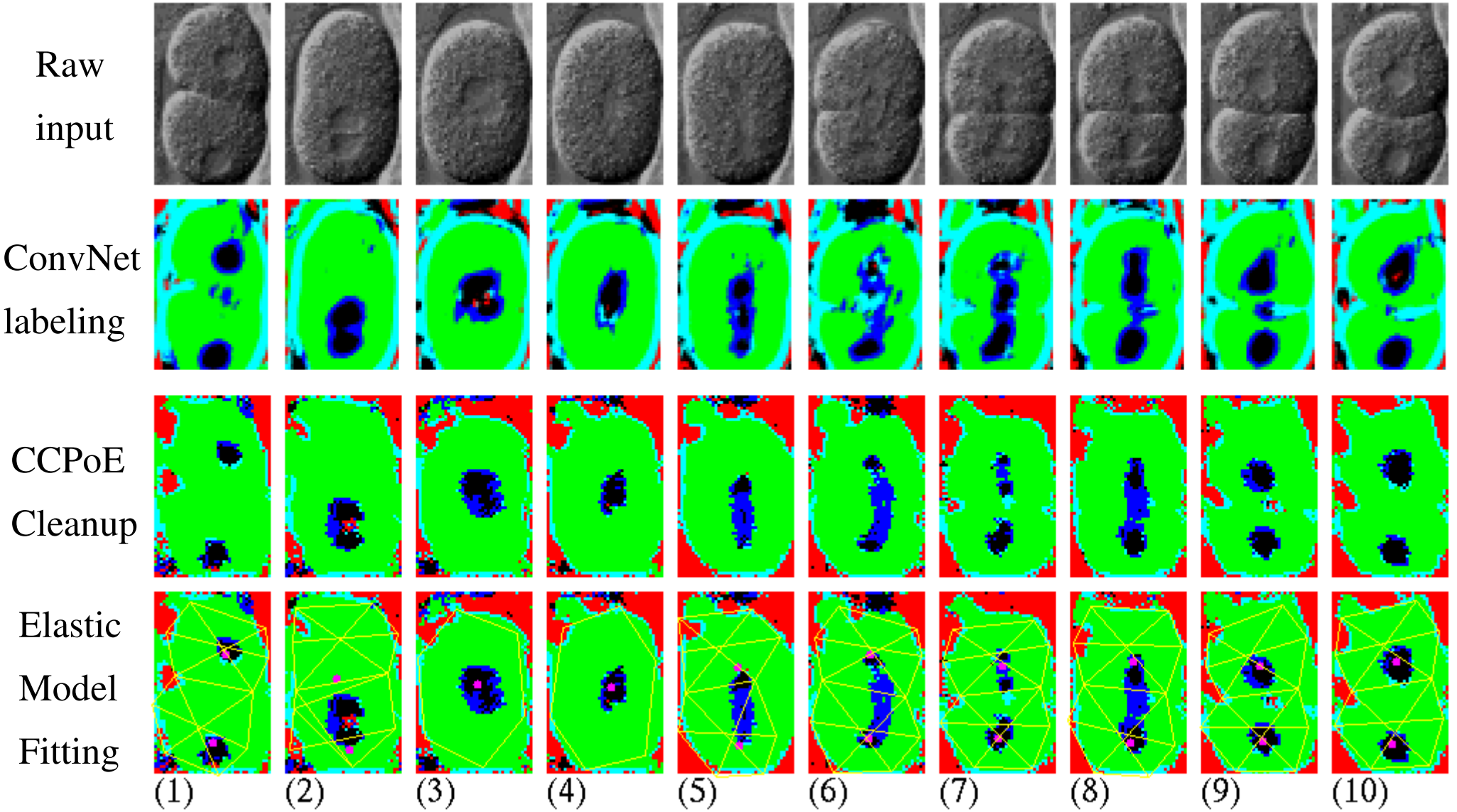








C. Elegans Embryo Phenotyping

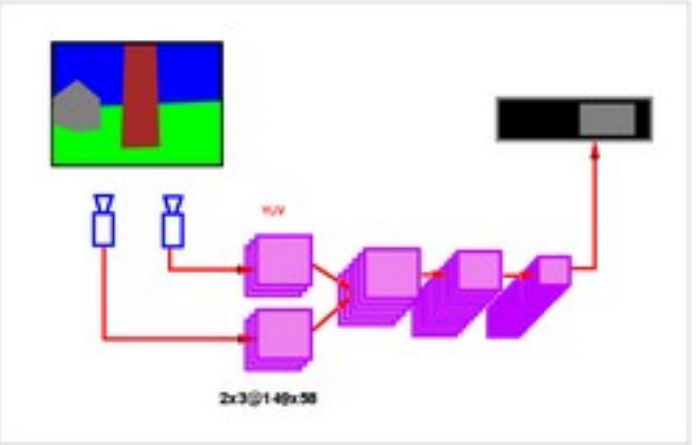


CCPoE = Convolutional Conditional Product of Experts [Ning et al, IEEE TIP 2005]

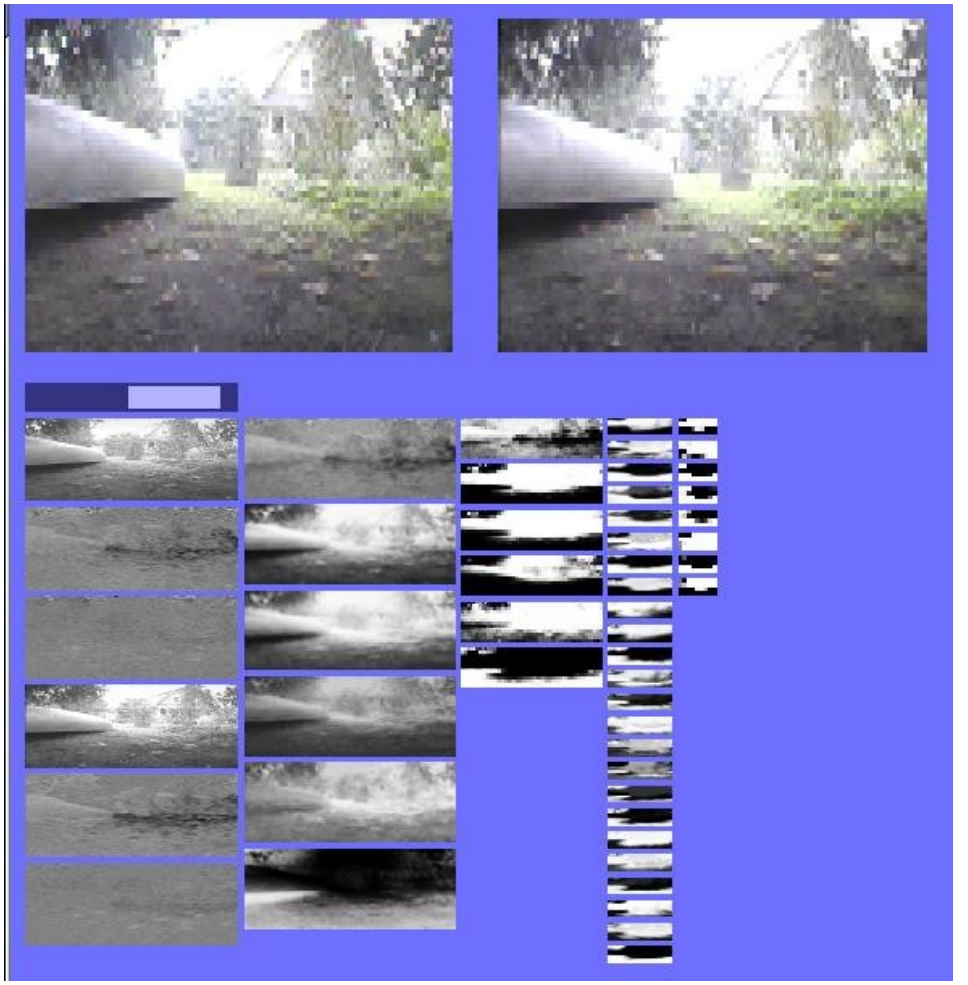
(similar to Field of Experts [Roth & Black, CVPR 2005])

Visual Navigation for a Mobile Robot

[LeCun et al. NIPS 2005]

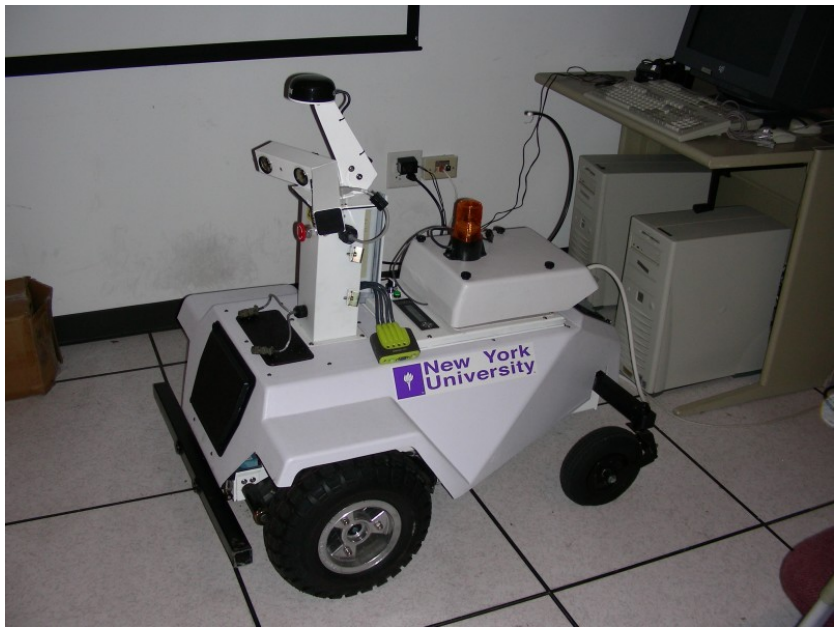


- Mobile robot with two cameras
- The convolutional net is trained to emulate a human driver from recorded sequences of video + human-provided steering angles.
- The network maps stereo images to steering angles for obstacle avoidance



LAGR: Learning Applied to Ground Robotics

- Getting a robot to drive autonomously in unknown terrain solely from vision (camera input).
- Our team (NYU/Net-Scale Technologies Inc.) is one of 8 participants funded by DARPA
- All teams received identical robots and can only modify the software (not the hardware)
- The robot is given the GPS coordinates of a goal, and must drive to the goal as fast as possible. The terrain is unknown in advance. The robot is run 3 times through the same course.



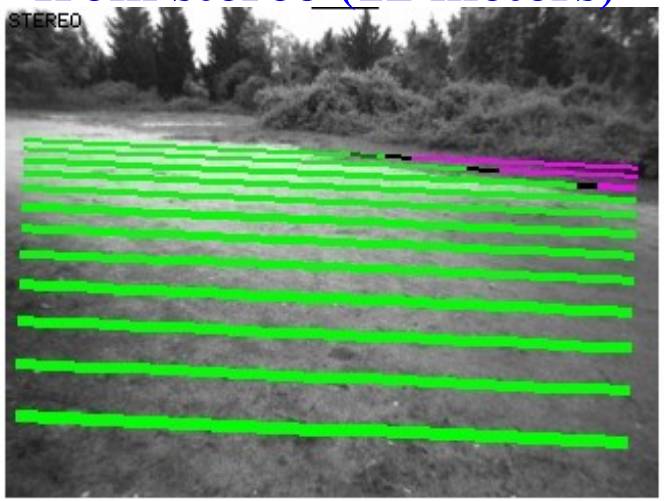
Training a ConvNet On-line to detect obstacles

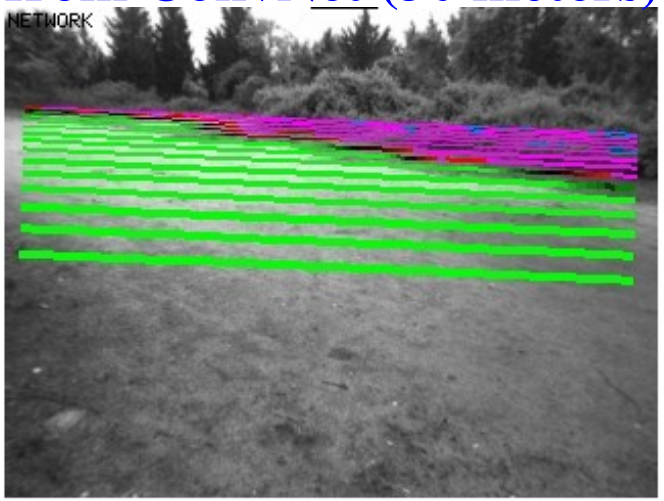
[Hadsell et al. Robotics Science and Systems 2007]

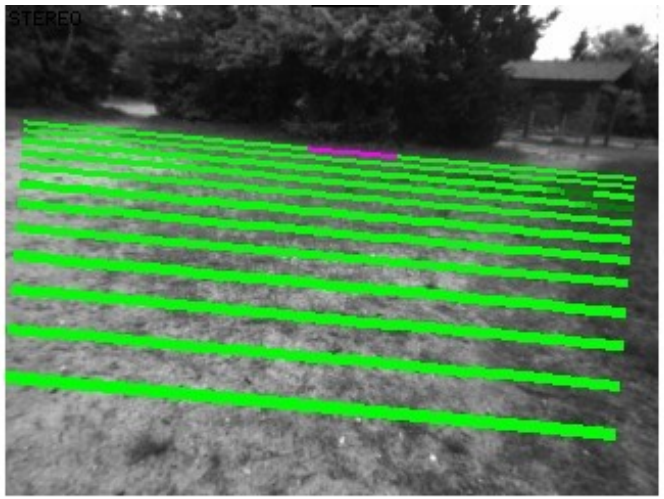
Raw image

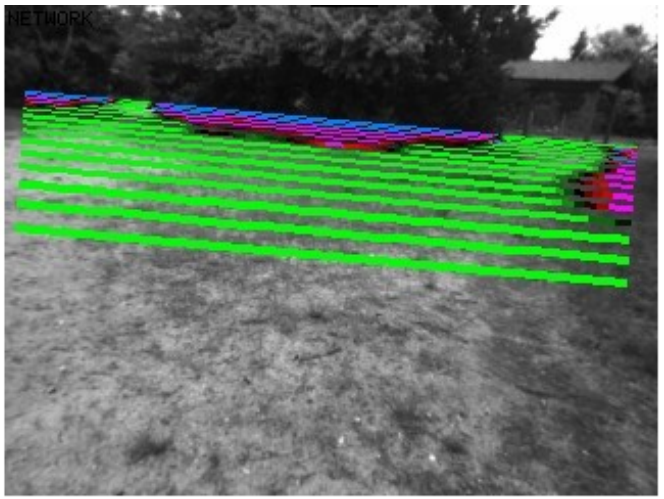
Traversability labels from stereo (12 meters)

Traversability labels from ConvNet (30 meters)





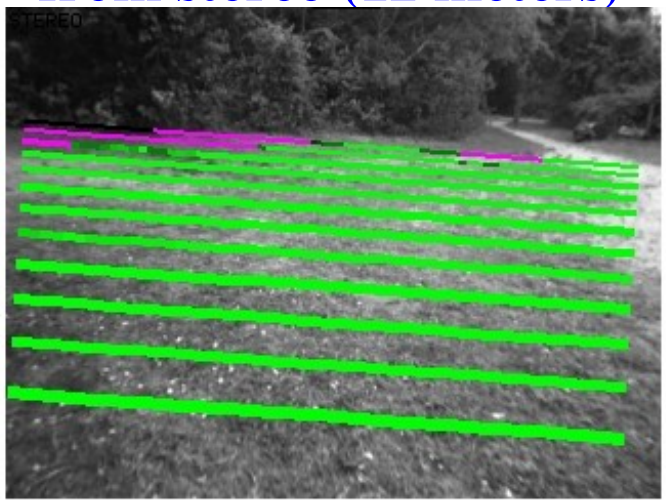
Training a ConvNet On-line to detect obstacles

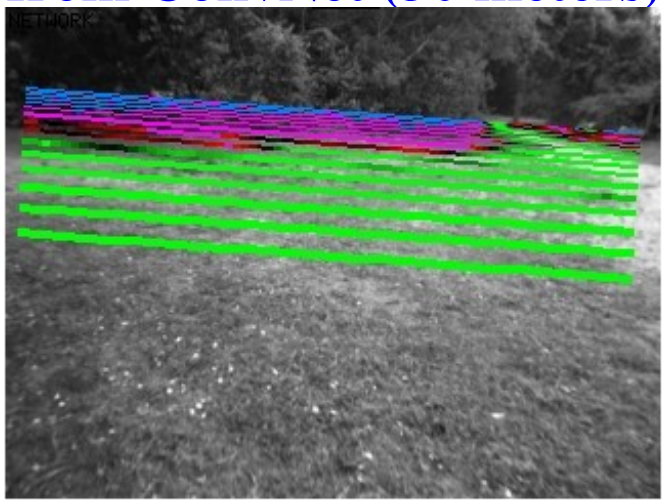
[Hadsell et al. Robotics Science and Systems 2007]

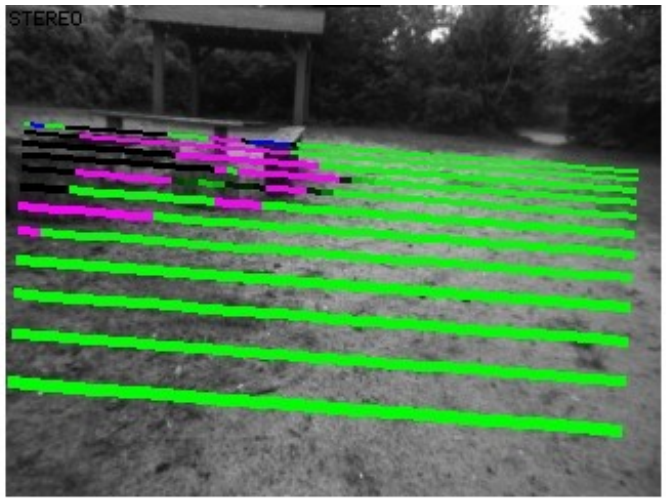
Raw image

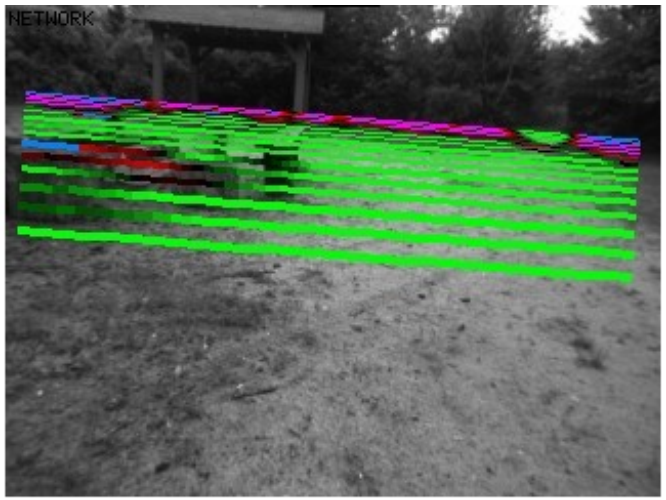
Traversability labels from stereo (12 meters)

Traversability labels from ConvNet (30 meters)

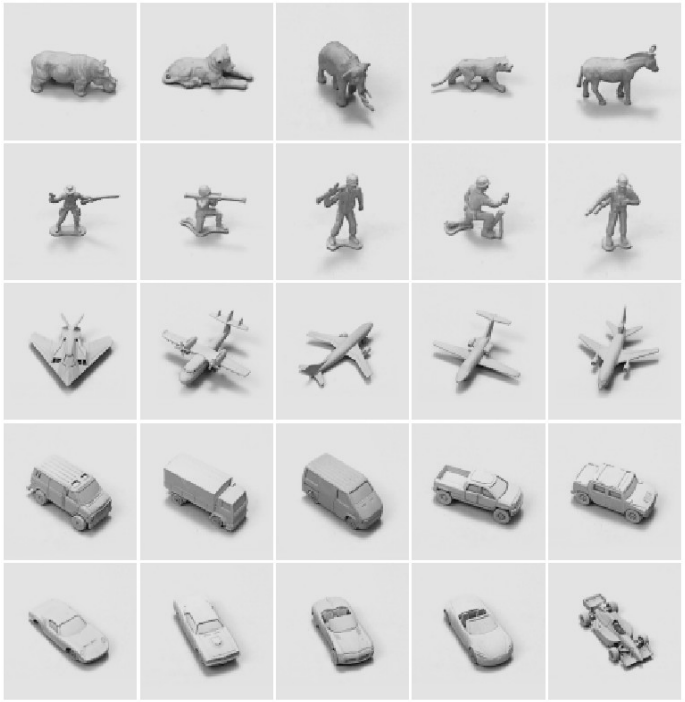




Generic Object Detection and Recognition with Invariance to Pose and Illumination

- **50** toys belonging to 5 categories: **animal, human figure, airplane, truck, car**
- **10** instance per category: 5 instances used for training, 5 instances for testing
- **Raw dataset: 972** stereo pair of each object instance. **48,600** image pairs total.
- **For each instance:**
  - **18 azimuths**
    - 0 to 350 degrees every 20 degrees
  - **9 elevations**
    - 30 to 70 degrees from horizontal every 5 degrees
  - **6 illuminations**
    - on/off combinations of 4 lights
  - **2 cameras (stereo)**
    - 7.5 cm apart
    - 40 cm from the object

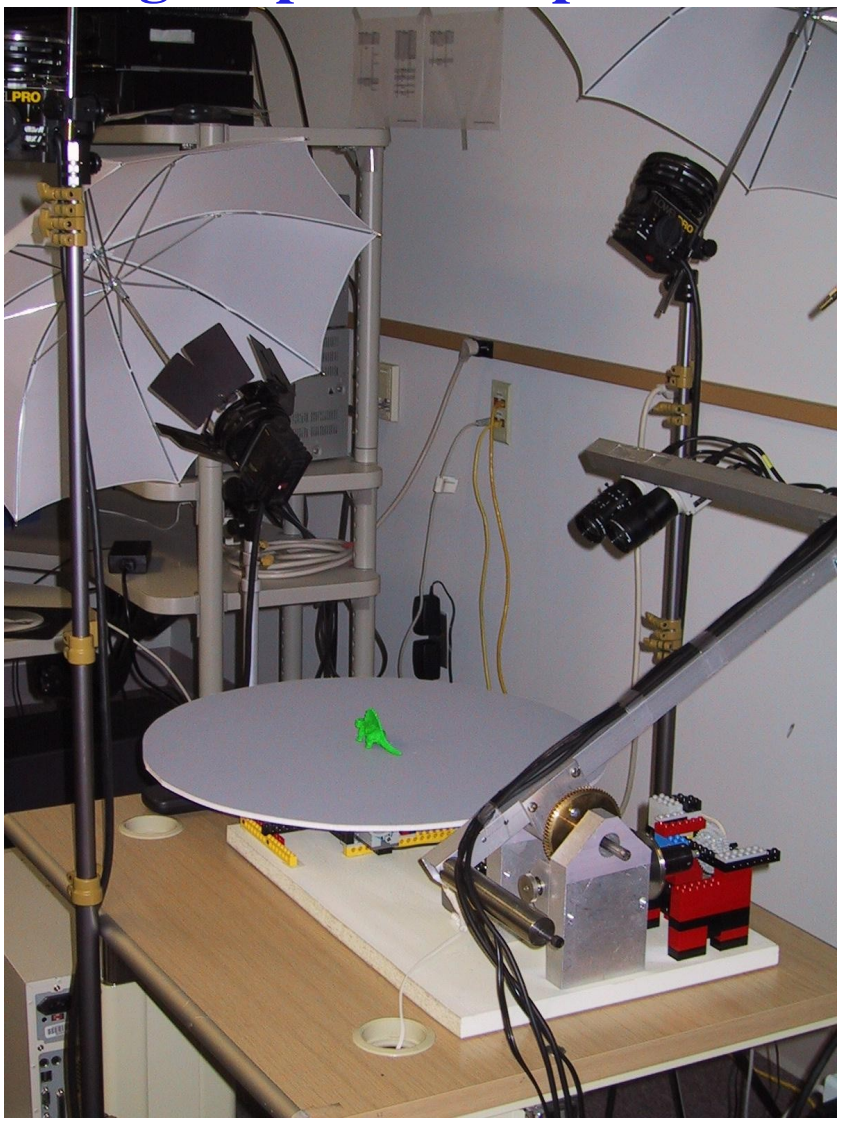


Training instances

Test instances

Data Collection, Sample Generation

Image capture setup

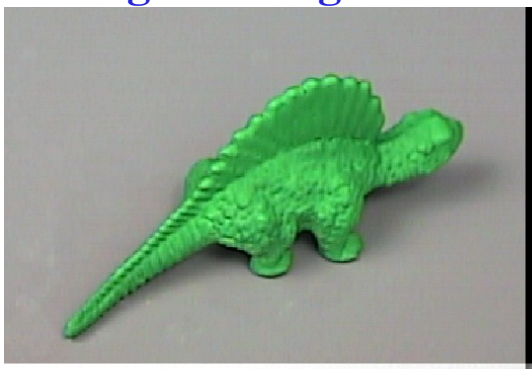


Objects are painted green so that:

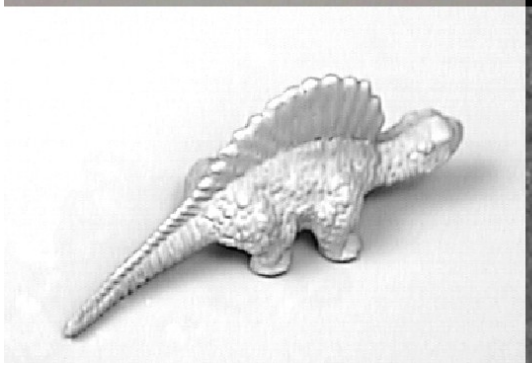
- all features other than shape are removed
- objects can be segmented, transformed,

  and composited onto various backgrounds

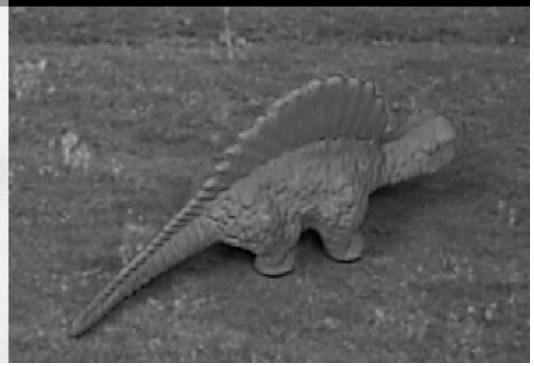
Original image



Object mask

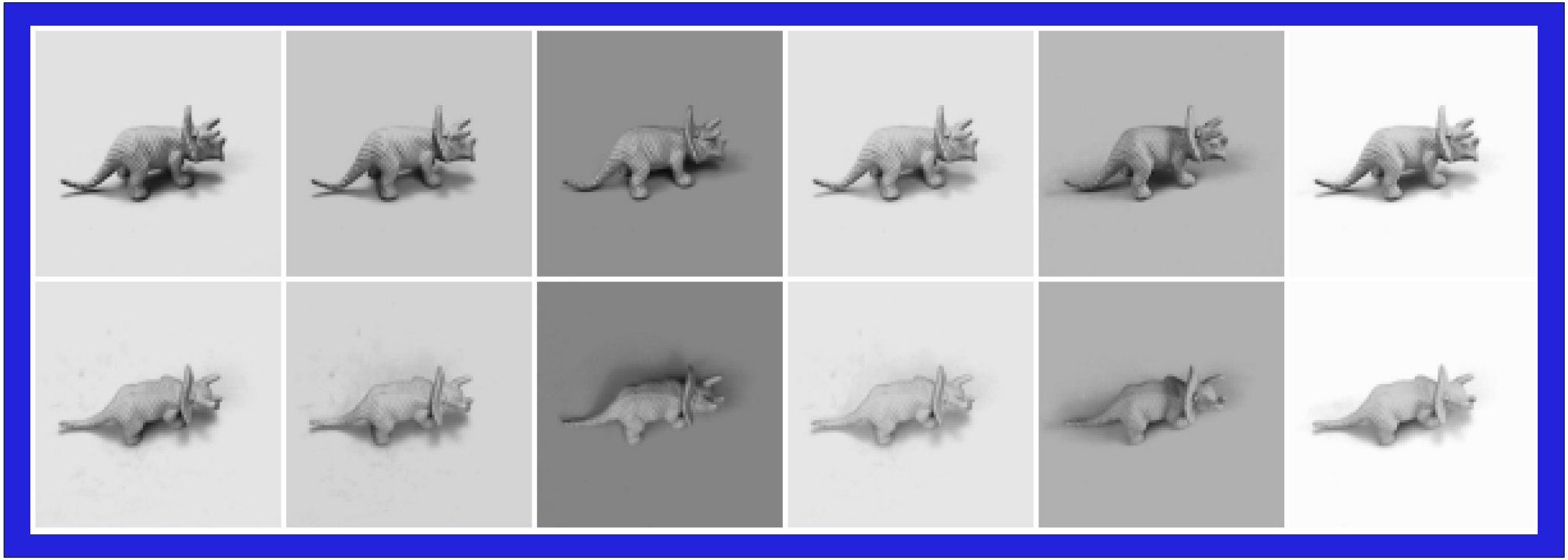


Shadow factor



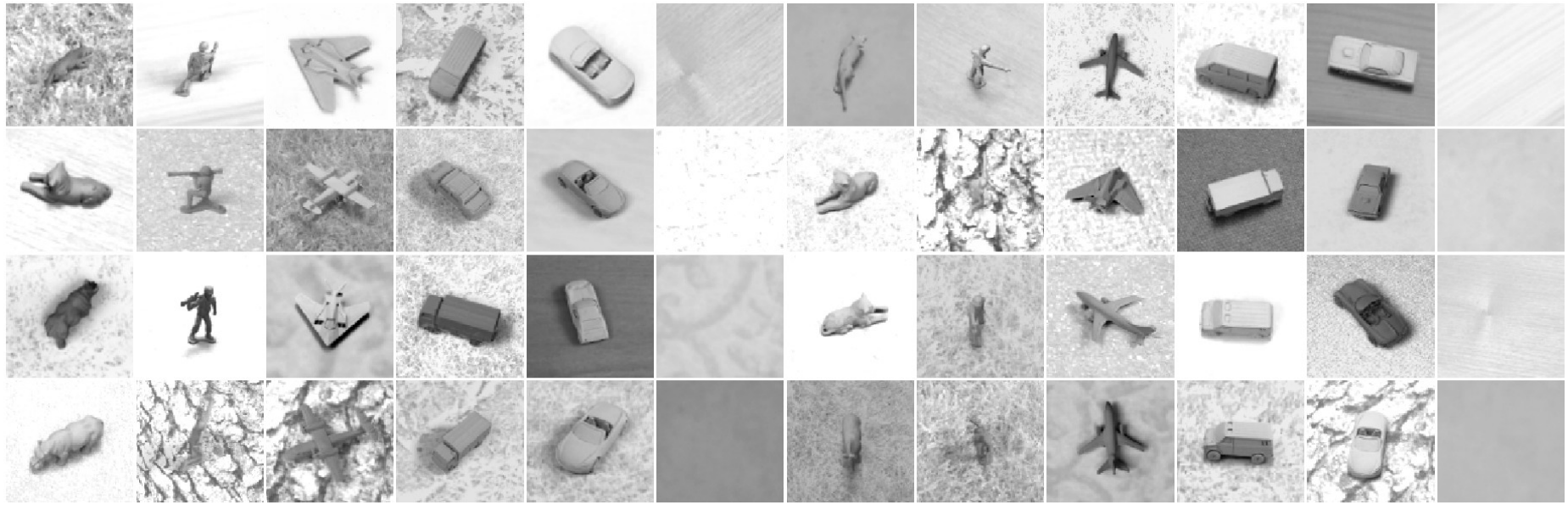
Composite image

Data Collection, Sample Generation



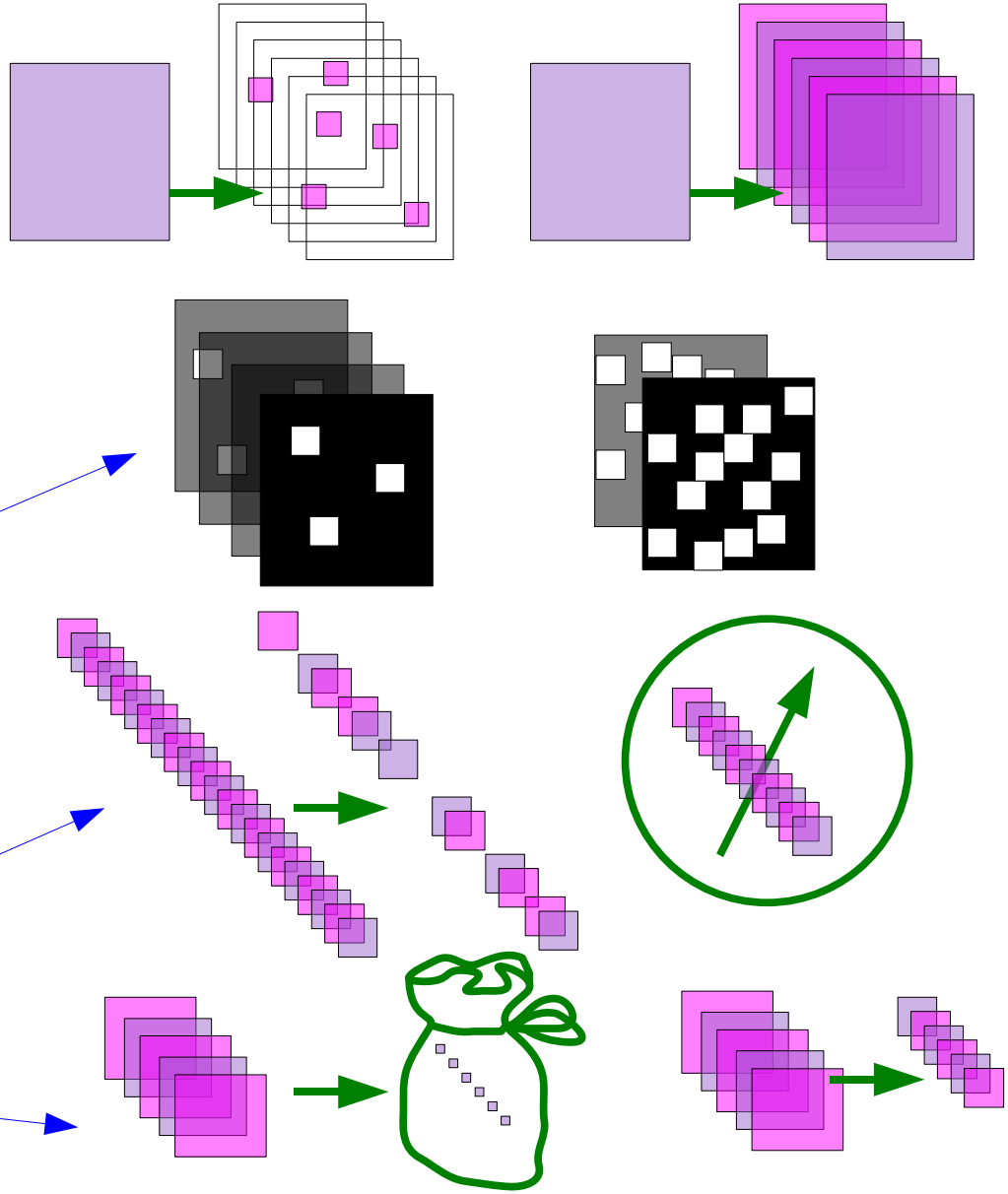
Samples showing the 6 different illuminations for 2 different elevations

Textured and Cluttered Datasets



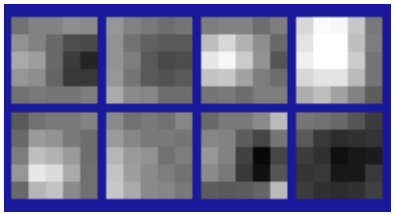
Computational Models of Object Recognition

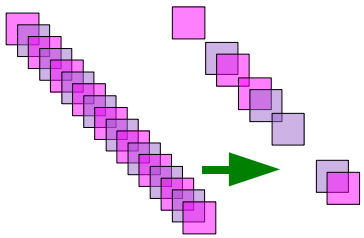
- Detecting features at interest points (Schmid, Perona, Ponce, Lowe) versus detecting them everywhere (LeCun, Ullman).
- Fixed features (Gabor, SIFT, Shape Context...), versus learned features
- Many sparse/selective features (Ullman's fragments) versus few dense/broad features (features that are "on" half the time).
- Selection from lots of simple features (Viola/Jones), vs tuning/optimization of a small number of features.
- Bag of features vs spatial relationships

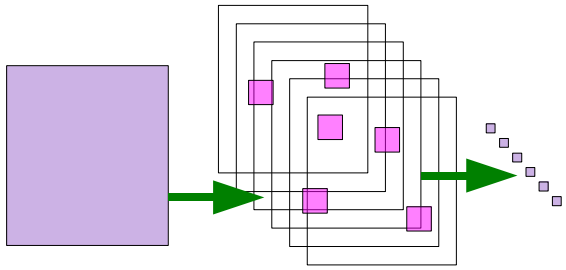


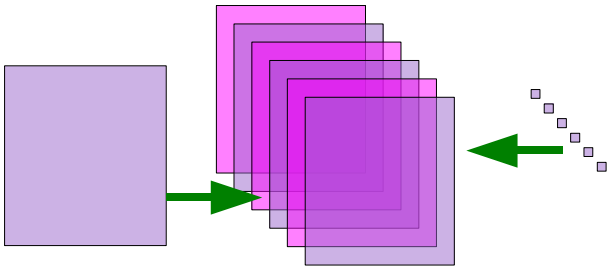
What Architecture, what training?

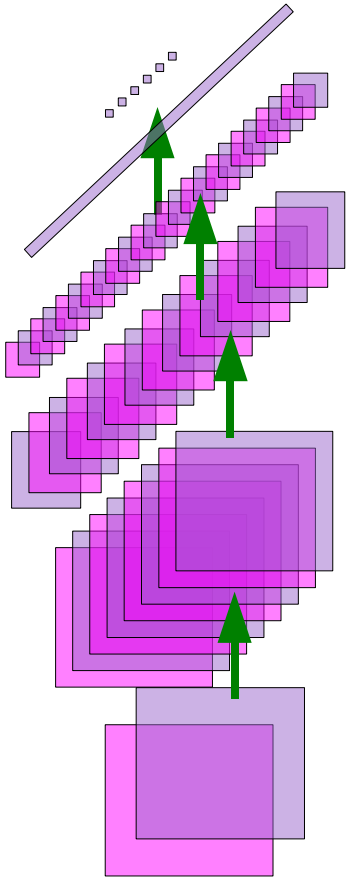
- Selection of "patch" features (Schmid, Ullman, Ponce, Perona,.....), versus optimization of non-template features.
- "heuristic" feature selection (e.g. Using mutual information) versus learning the features by optimizing a global performance measure.
- Piecemeal training of feature and model, versus global training of the whole system
- 2-layer feature+model (almost everyone), versus hierarchical/multi-level (LeCun, Riesenhuber, Geman, Ullman)
- Generative (Perona, Al[illegible]reeman), versus discriminative ([illegible]un, Viola)

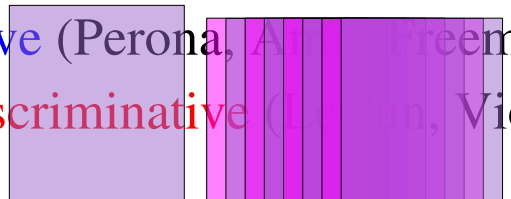








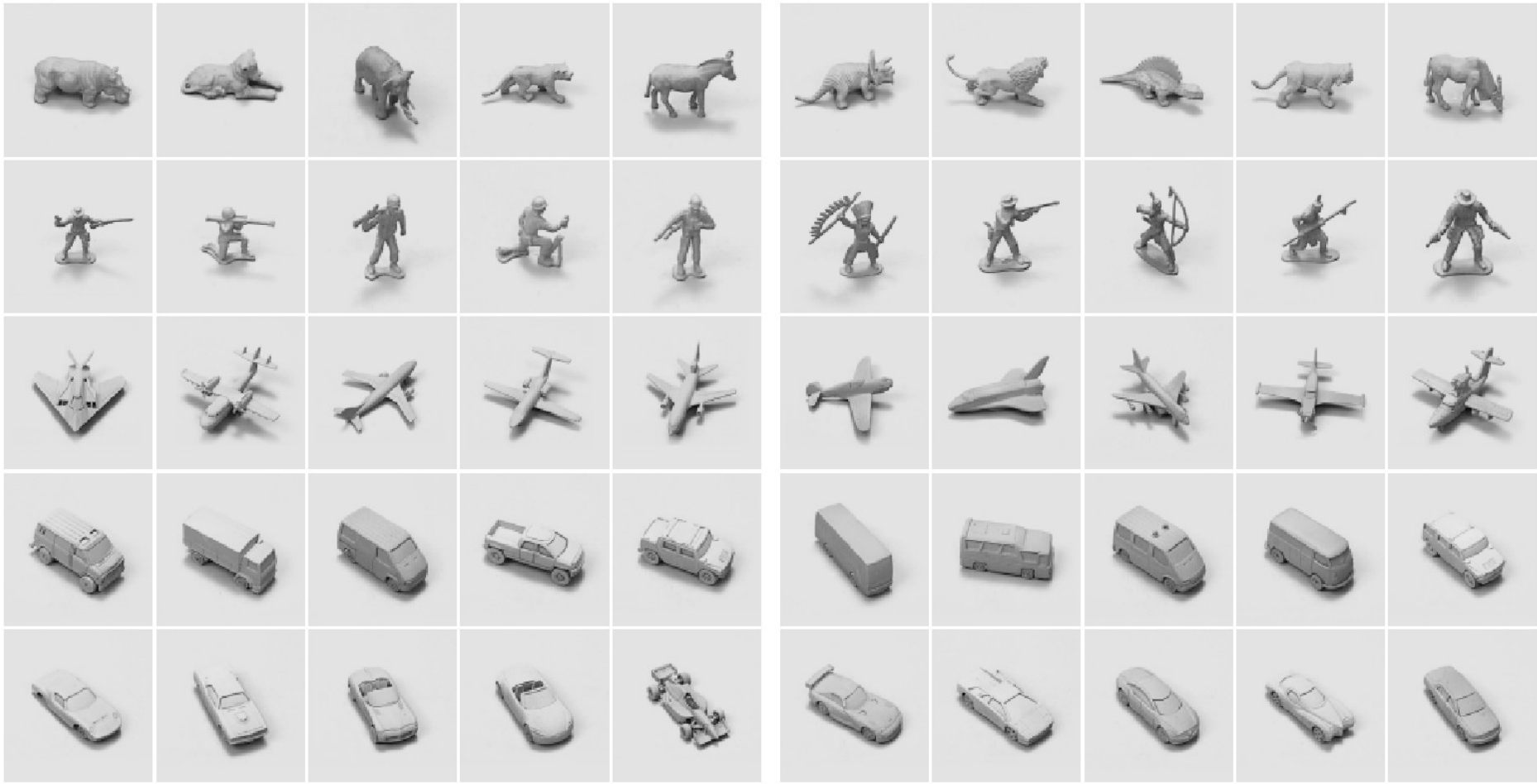




Experiment 1: Normalized-Uniform Dataset

- **Normalized-Uniform Dataset: 972 stereo pair of each object instance** (18 azimuths X 9 elevations X 6 illuminations).
- **5 categories. 5 instances/category for training, 5 instances/category for testing**
- **24,300** stereo pairs for training, **24,300** for testing
- Objects are centered and size-normalized so all the views of each object instance fits in an 80x80 pixel window.
- Objects are placed on uniform backgrounds (one for each of the 6 illuminations) of size 96x96 pixels
- Each sample is composed of two 96x96 images

Experiment 1: Normalized-Uniform Dataset

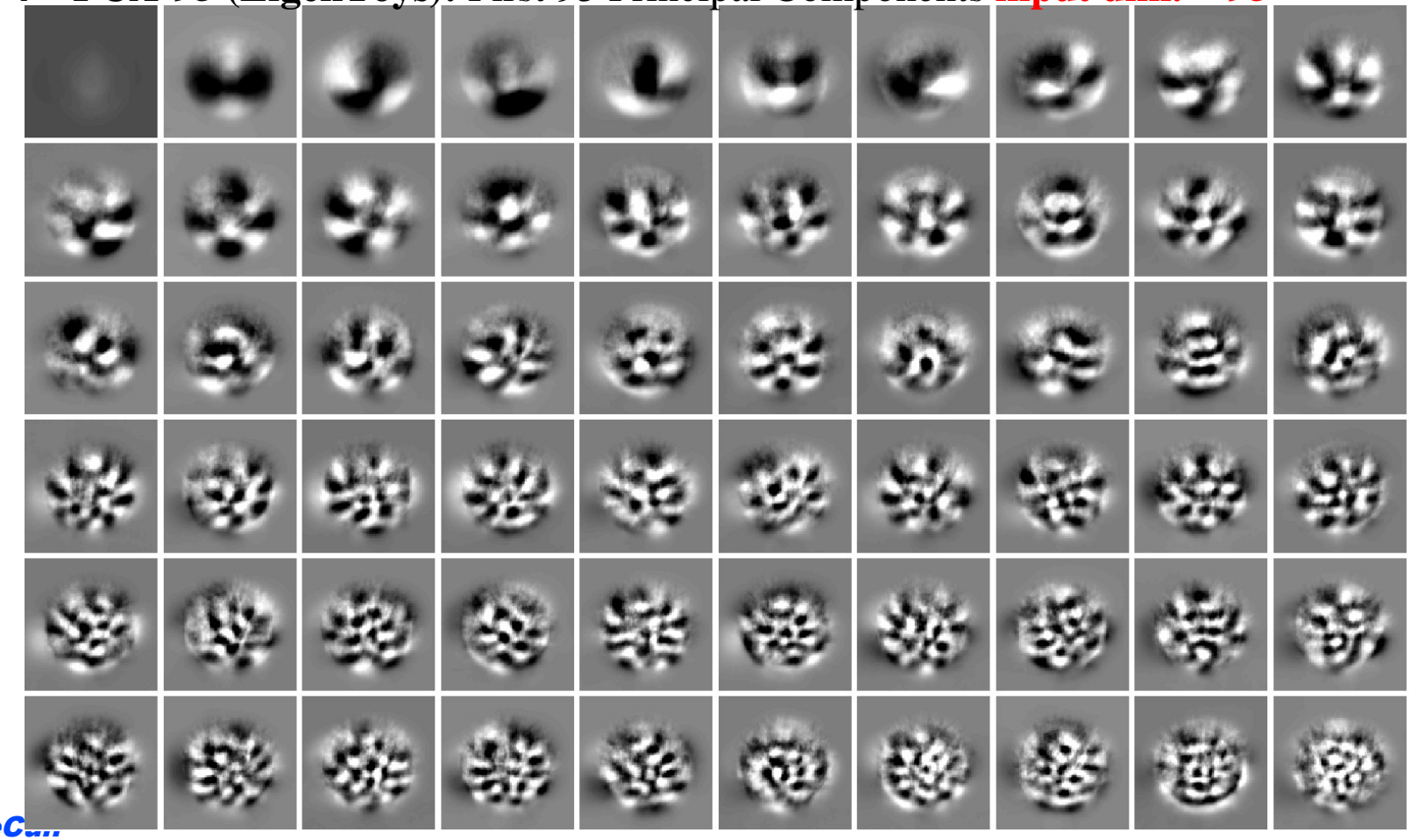


Training instances

Test instances

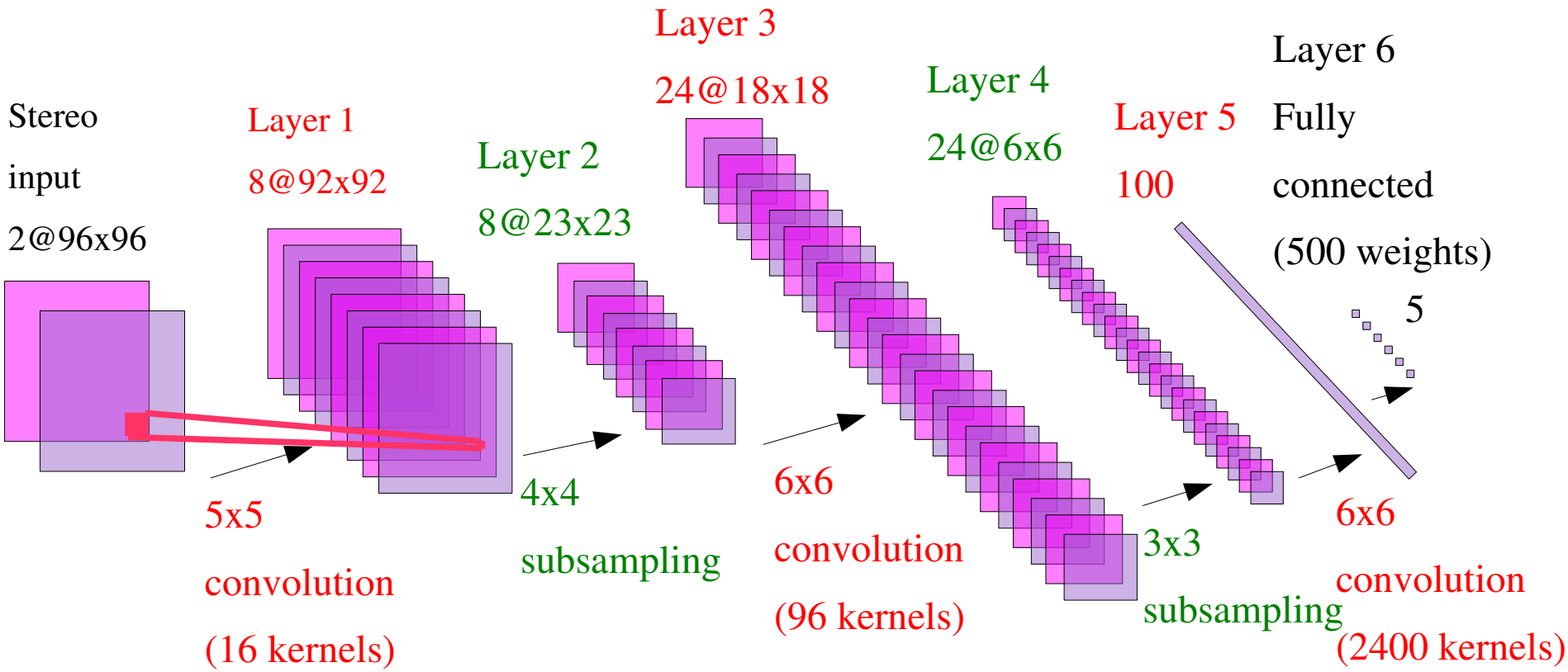
Experiment 1: Normalized-Uniform Set: Representations

- **1 - Raw Stereo Input:** 2 images 96x96 pixels **input dim. = 18432**
- **2 - Raw Monocular Input:** 1 image, 96x96 pixels **input dim. = 9216**
- **3 – Subsampled Mono Input:** 1 image, 32x32 pixels **input dim = 1024**
- **4 – PCA-95 (EigenToys):** First 95 Principal Components **input dim. = 95**



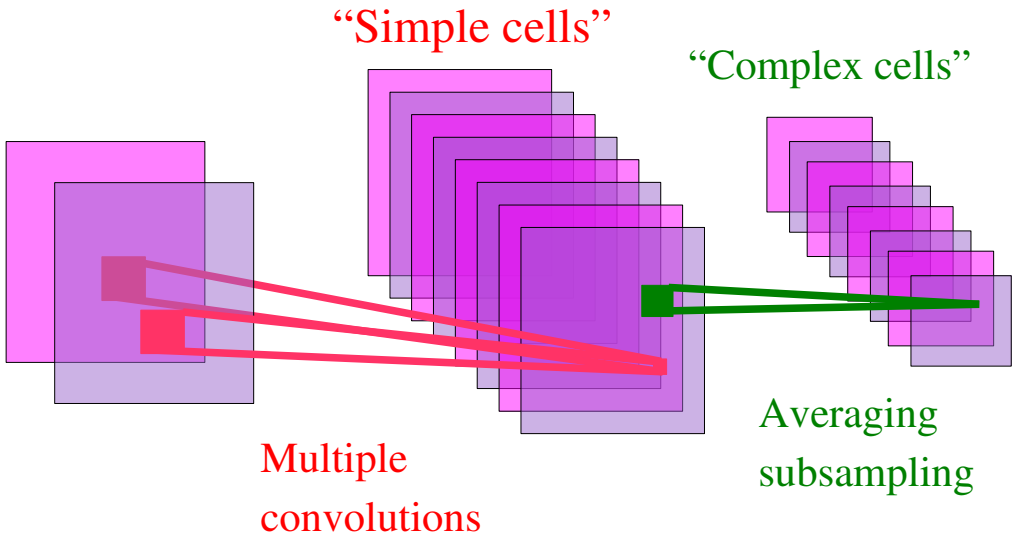
First 60 eigenvectors (EigenToys)

Convolutional Network

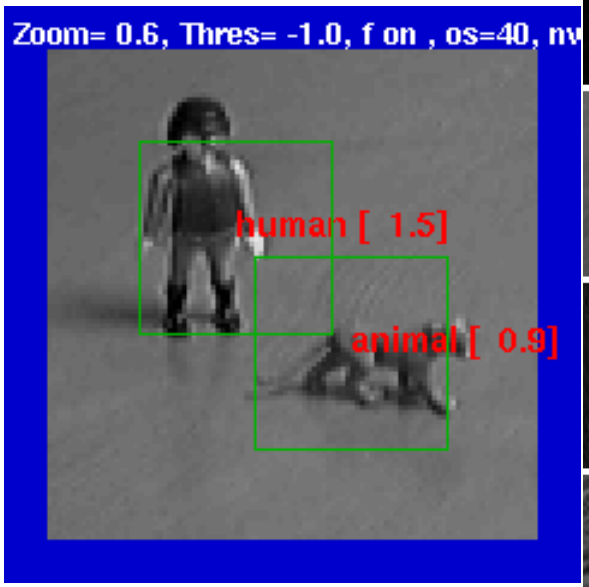


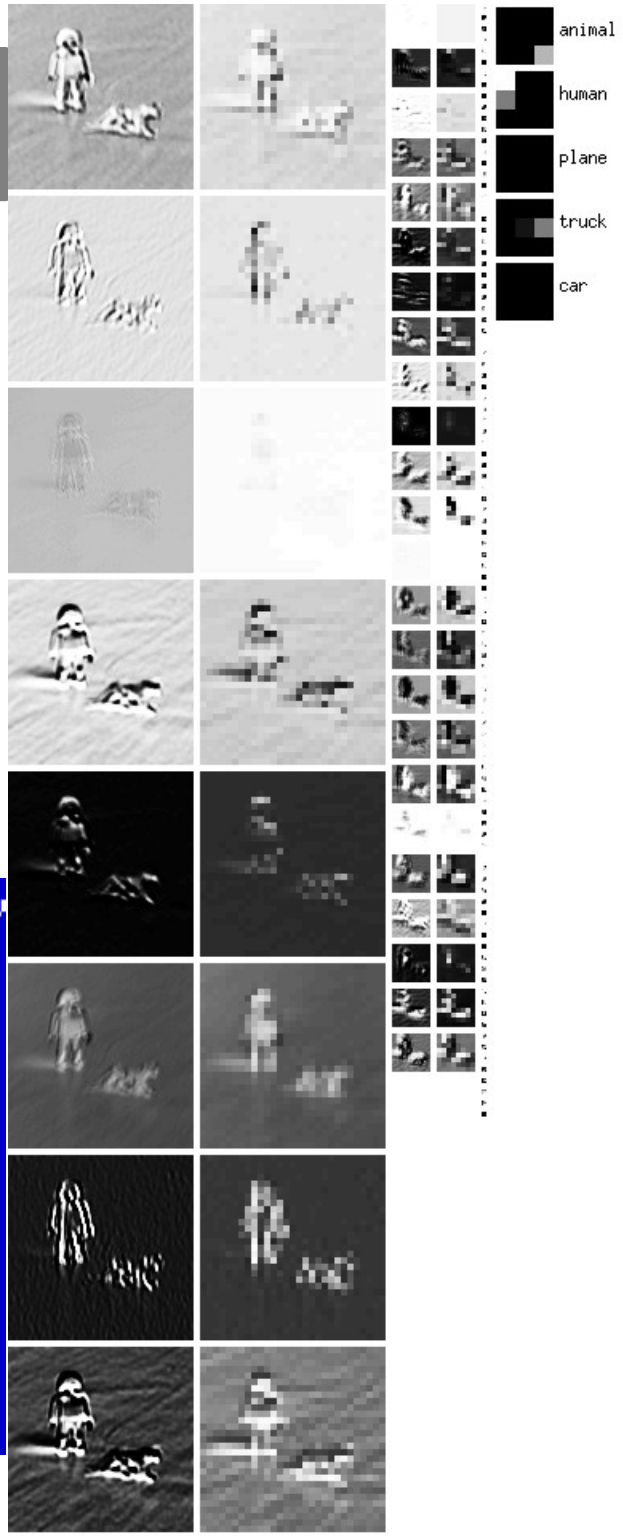
- **90,857 free parameters, 3,901,162 connections.**
- The architecture alternates convolutional layers (feature detectors) and subsampling layers (local feature pooling for invariance to small distortions).
- **The entire network is trained end-to-end** (all the layers are trained simultaneously).
- A gradient-based algorithm is used to minimize a supervised loss function.

Alternated Convolutions and Subsampling



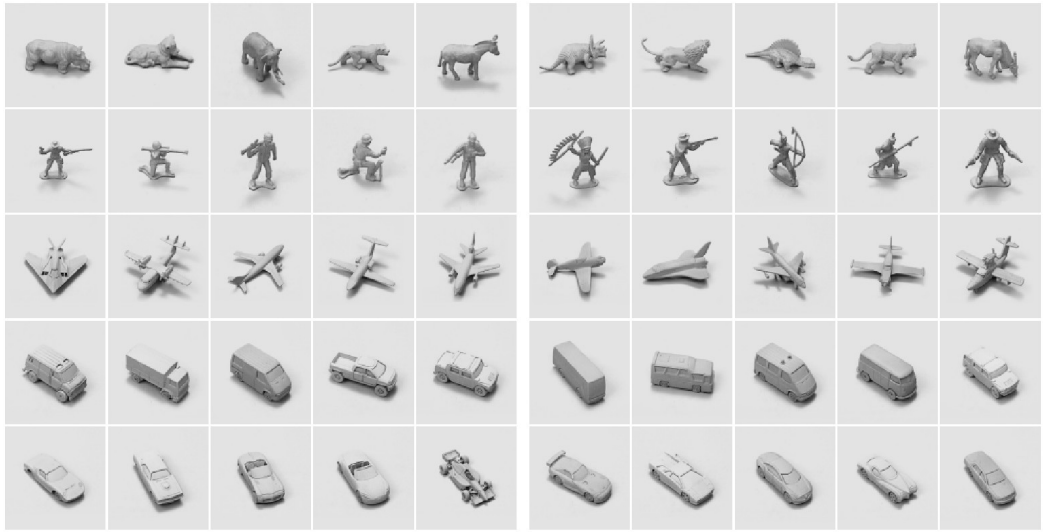
- Local features are extracted everywhere.
- averaging/subsampling layer builds robustness to variations in feature locations.
- Hubel/Wiesel'62, Fukushima'71, LeCun'89, Riesenhuber & Poggio'02, Ullman'02,....





Normalized-Uniform Set: Error Rates

- **Linear Classifier on raw stereo images:** **30.2% error.**
- **K-Nearest-Neighbors on raw stereo images:** **18.4% error.**
- **K-Nearest-Neighbors on PCA-95:** **16.6% error.**
- **Pairwise SVM on 96x96 stereo images:** **11.6% error**
- **Pairwise SVM on 95 Principal Components:** **13.3% error.**
- **Convolutional Net on 96x96 stereo images:** **5.8% error.**



Training instances Test instances

Normalized-Uniform Set: Learning Times

	SVM	Conv Net				SVM/Conv
test error	11.6%	10.4%	6.2%	5.8%	6.2%	5.9%
train time (min*GHz)	480	64	384	640	3,200	50+
test time per sample (sec*GHz)	0.95	0.03				0.04+
#SV	28%					28%
parameters	σ=2,000 C=40					dim=80 σ=5 C=0.01

SVM: using a parallel implementation by Graf, Durdanovic, and Cosatto (NEC Labs)

Chop off the last layer of the convolutional net and train an SVM on it

Jittered-Cluttered Dataset

- **Jittered-Cluttered Dataset:**
- **291,600** tereo pairs for training, **58,320** for testing
- Objects are jittered: position, scale, in-plane rotation, contrast, brightness, backgrounds, distractor objects,...
- Input dimension: 98x98x2 (approx 18,000)

Experiment 2: Jittered-Cluttered Dataset

- 291,600 training samples, 58,320 test samples
- SVM with Gaussian kernel — 43.3% error
- Convolutional Net with binocular input: — 7.8% error
- Convolutional Net + SVM on top: — 5.9% error
- Convolutional Net with monocular input: — 20.8% error
- Smaller mono net (DEMO): — 26.0% error
- Dataset available from http://www.cs.nyu.edu/~yann

Jittered-Cluttered Dataset

	SVM	Conv Net			SVM/Conv
test error	43.3%	16.38%	7.5%	7.2%	5.9%
train time (min*GHz)	10,944	420	2,100	5,880	330+
test time per sample (sec*GHz)	2.2	0.04			0.06+
#SV	5%				2%
parameters	σ=10^4 C=40				dim=100 σ=5 C=1

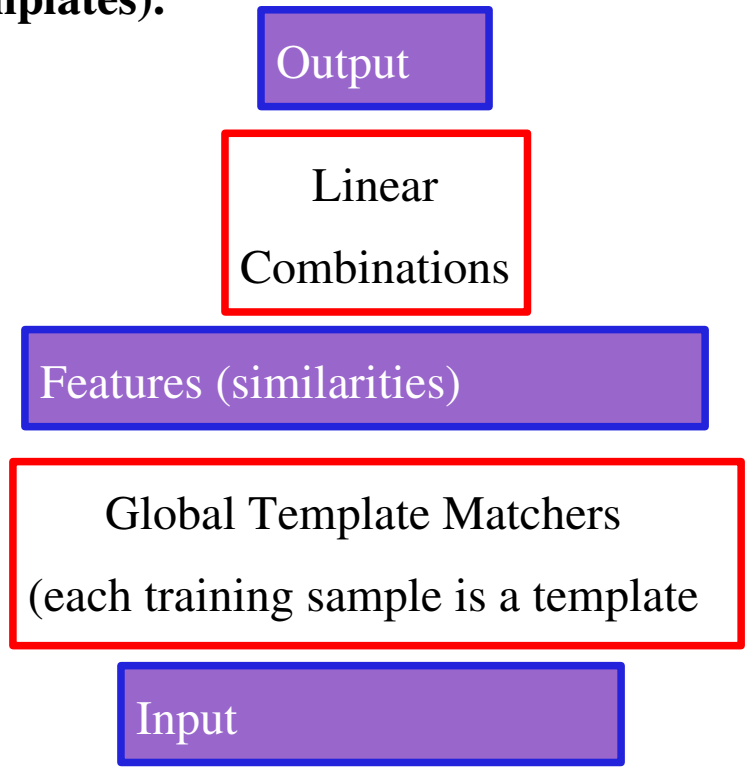
OUCH!

The convex loss, VC bounds and representers theorems don't seem to help

Chop off the last layer, and train an SVM on it it works!

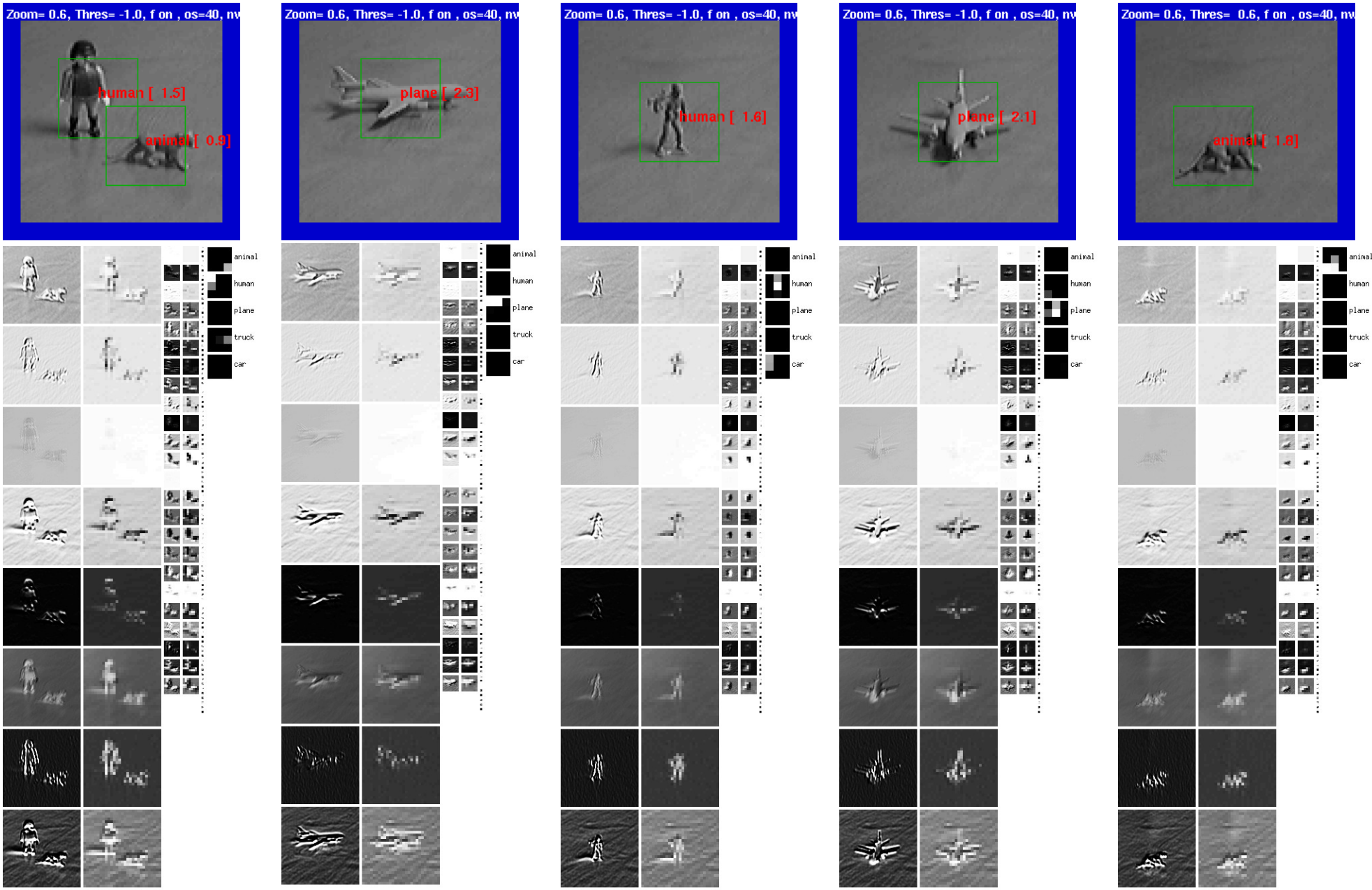
What's wrong with SVMs? they are shallow!

- SVM with Gaussian kernels is based on matching global templates
- It is a "shallow" architectures
- There is now way to learn invariant recognition tasks with such naïve architectures (unless we use an impractically large number of templates).
  - The number of necessary templates grows exponentially with the number of dimensions of variations.
  - Global templates are in trouble when the variations include: category, instance shape, configuration (for articulated object), position, azimuth, elevation, scale, illumination, texture, albedo, in-plane rotation, background luminance, background texture, background clutter, ...



SVM is glorified template matching

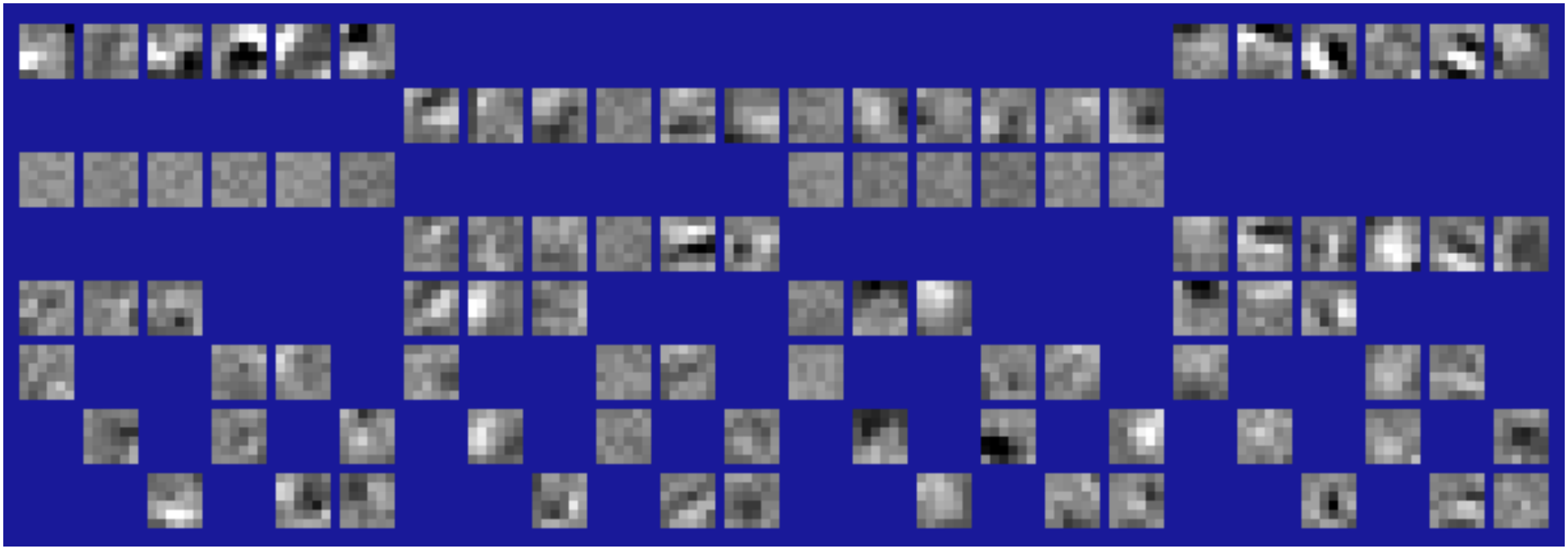
Examples (Monocular Mode)



Learned Features

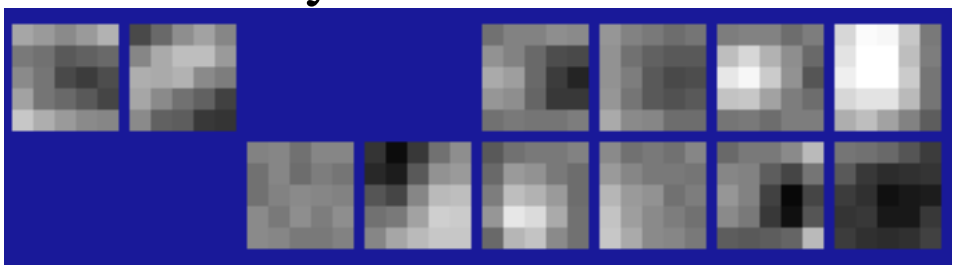
Layer 3

Layer 2

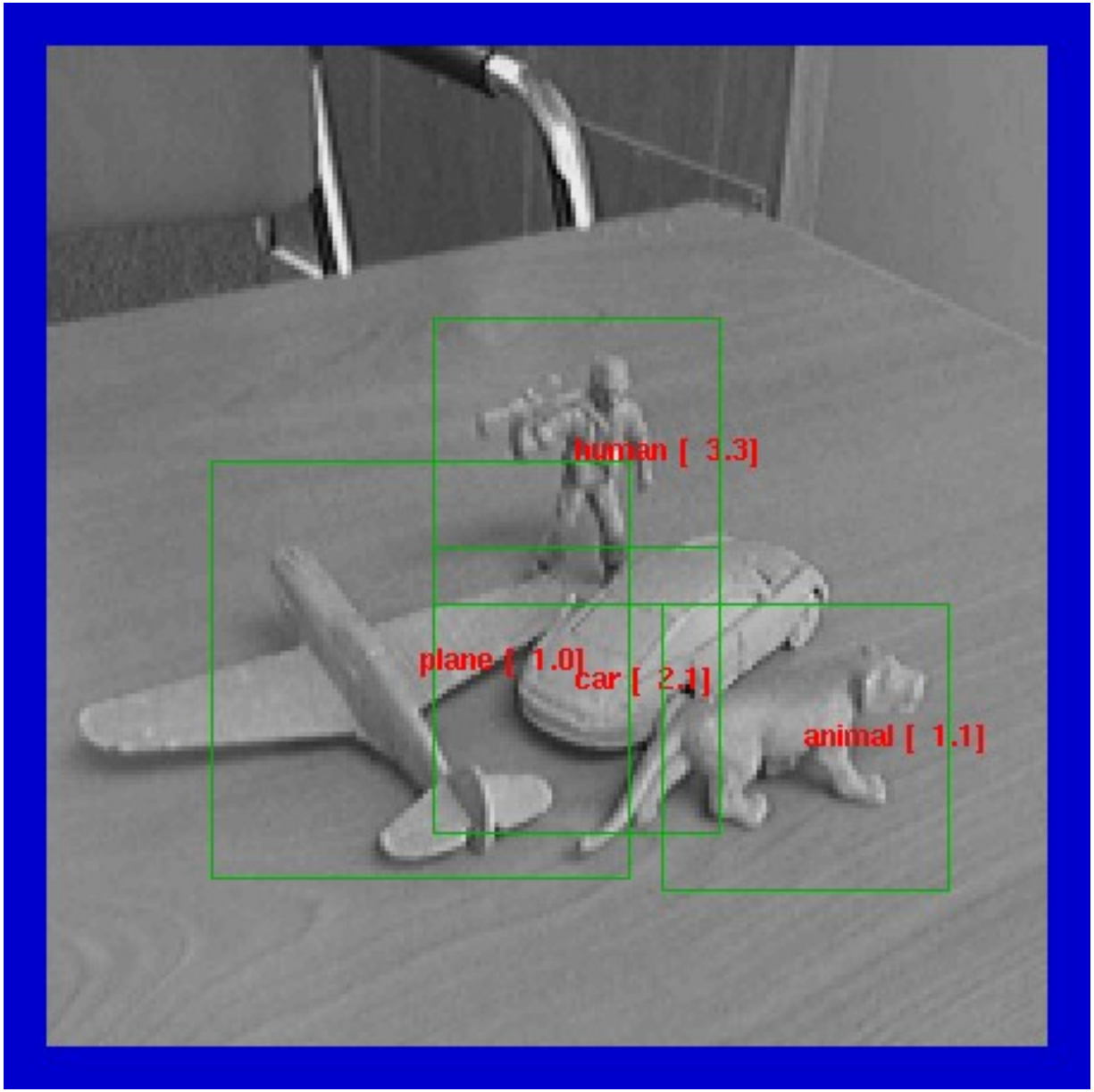


Layer 1

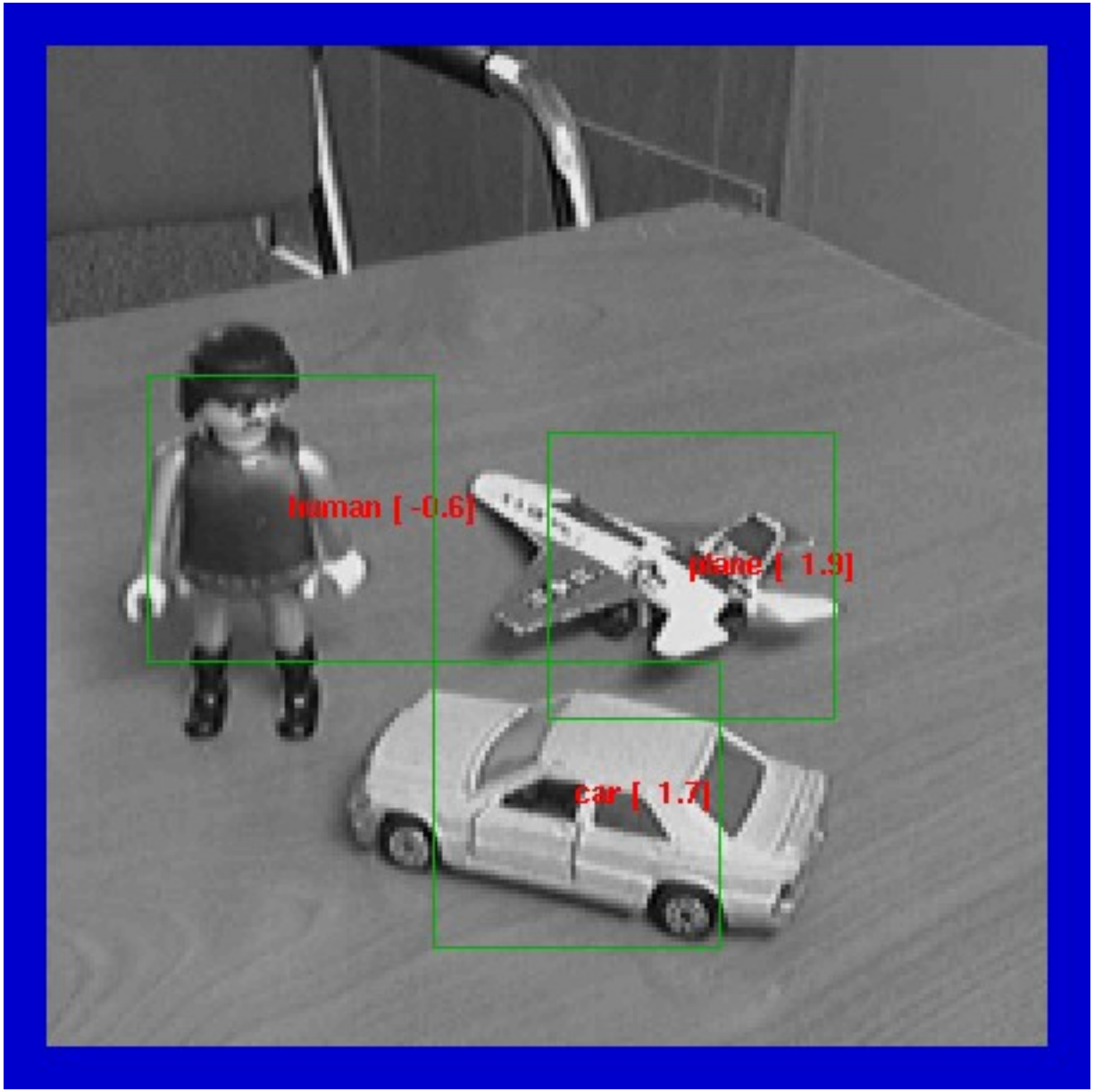
Input



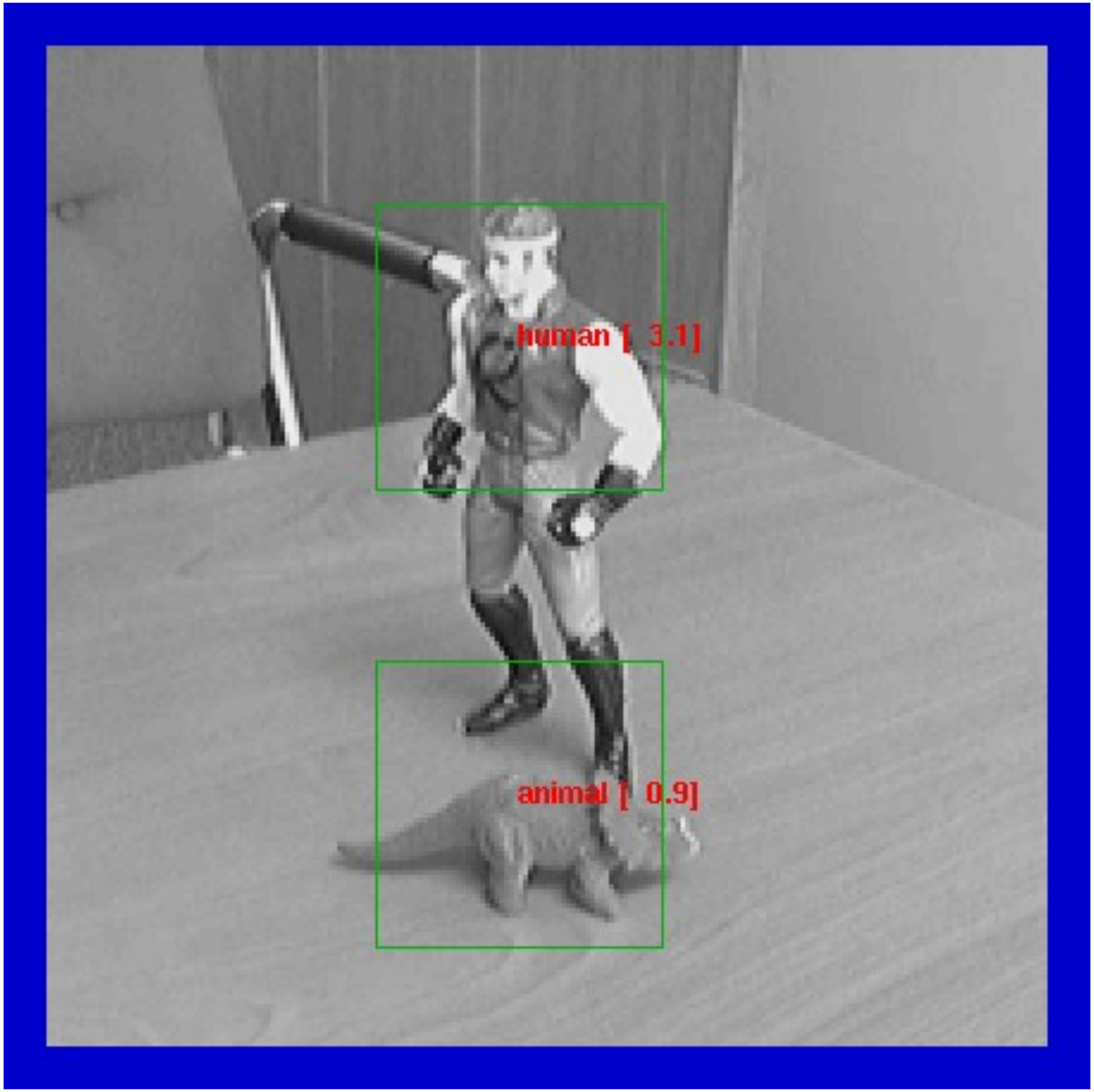
Examples (Monocular Mode)



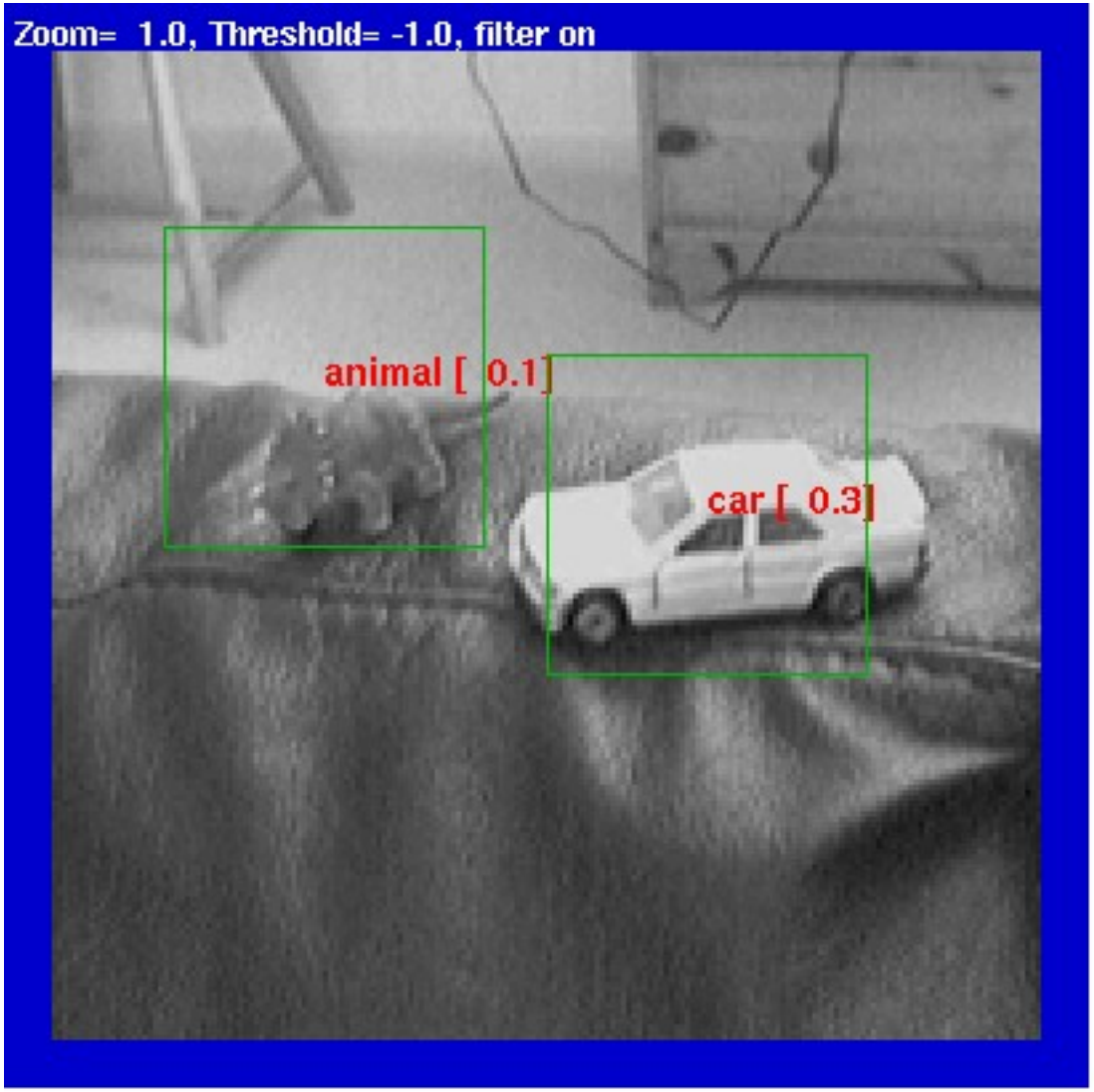
Examples (Monocular Mode)



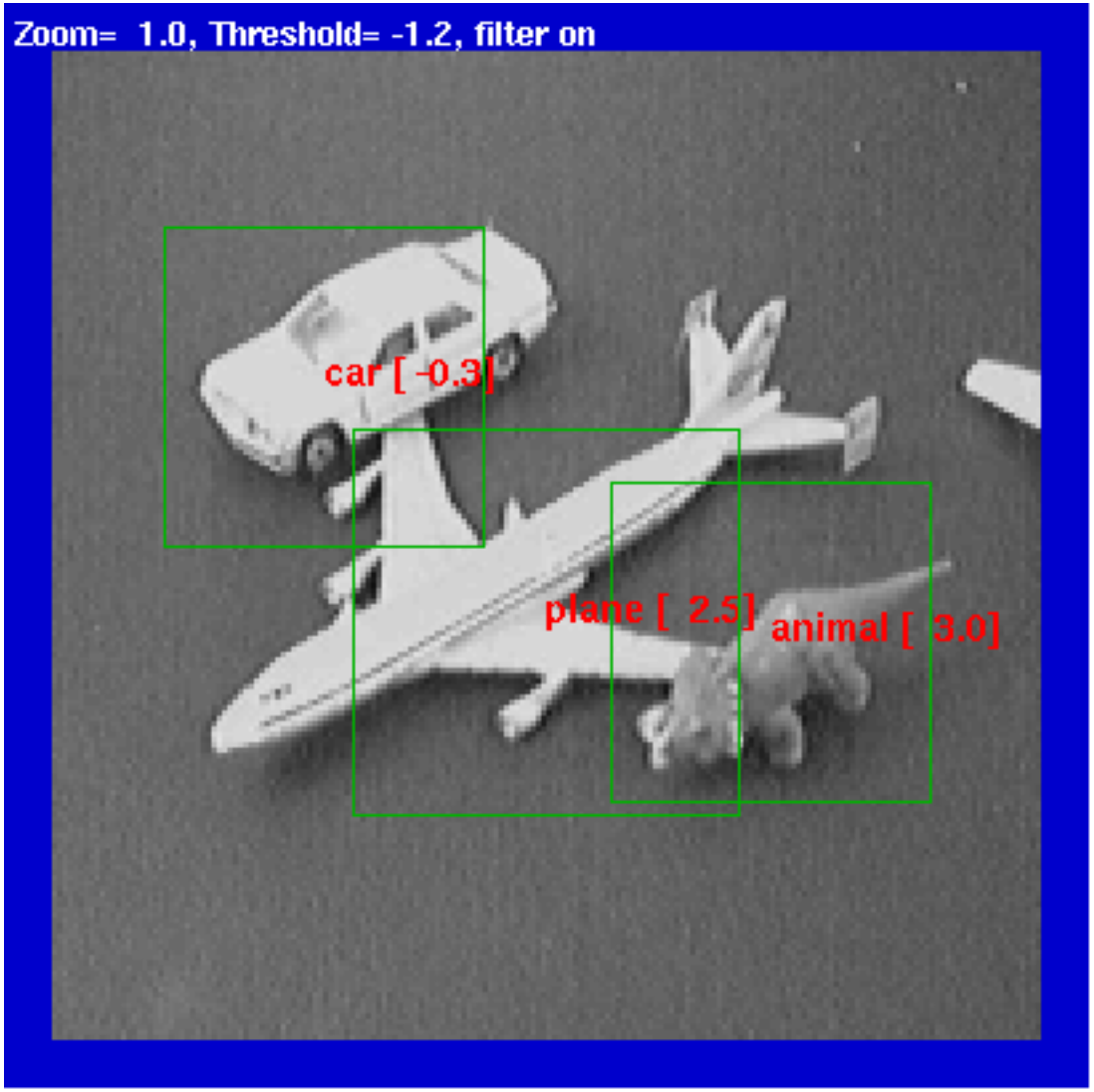
Examples (Monocular Mode)



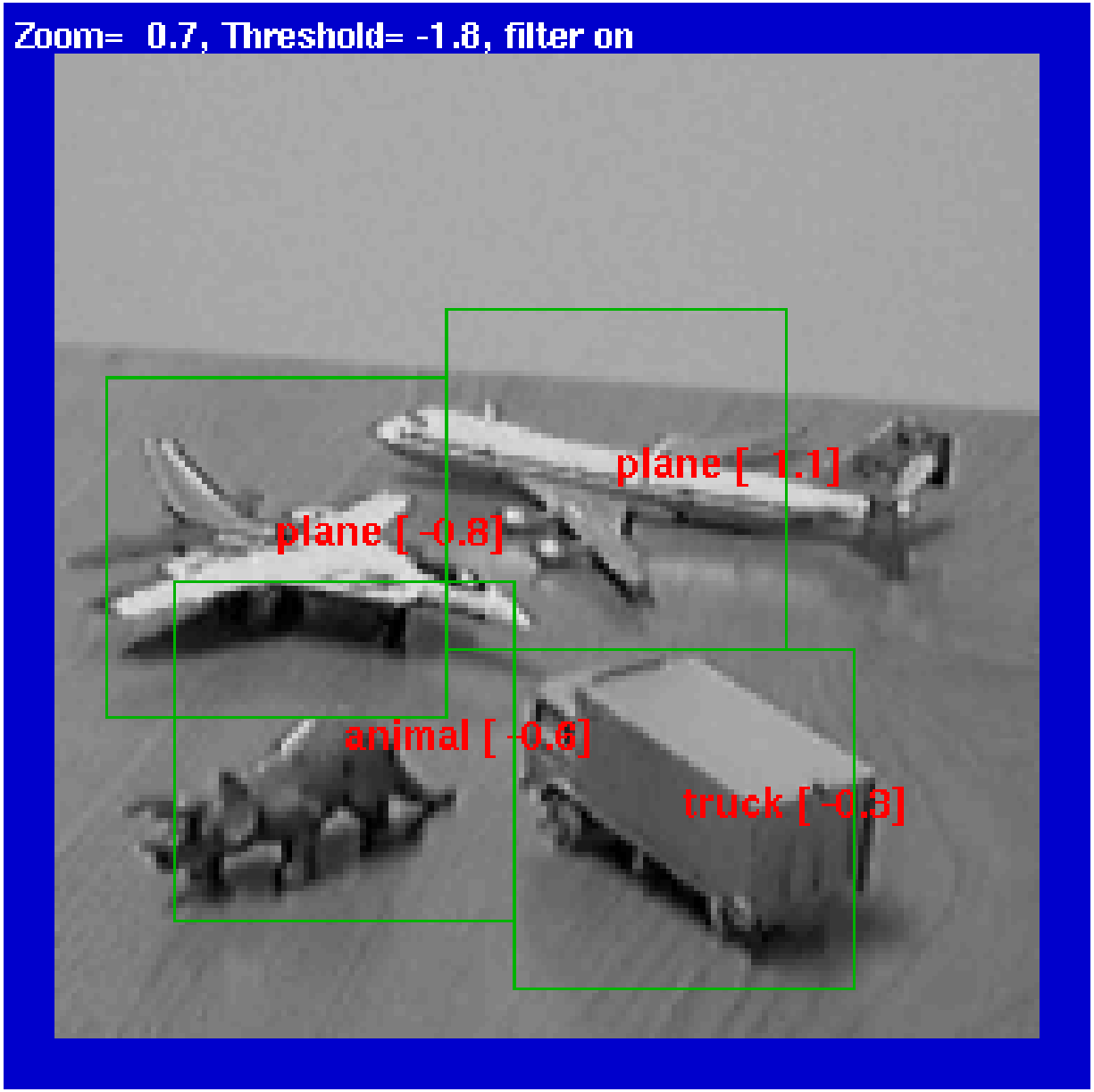
Examples (Monocular Mode)



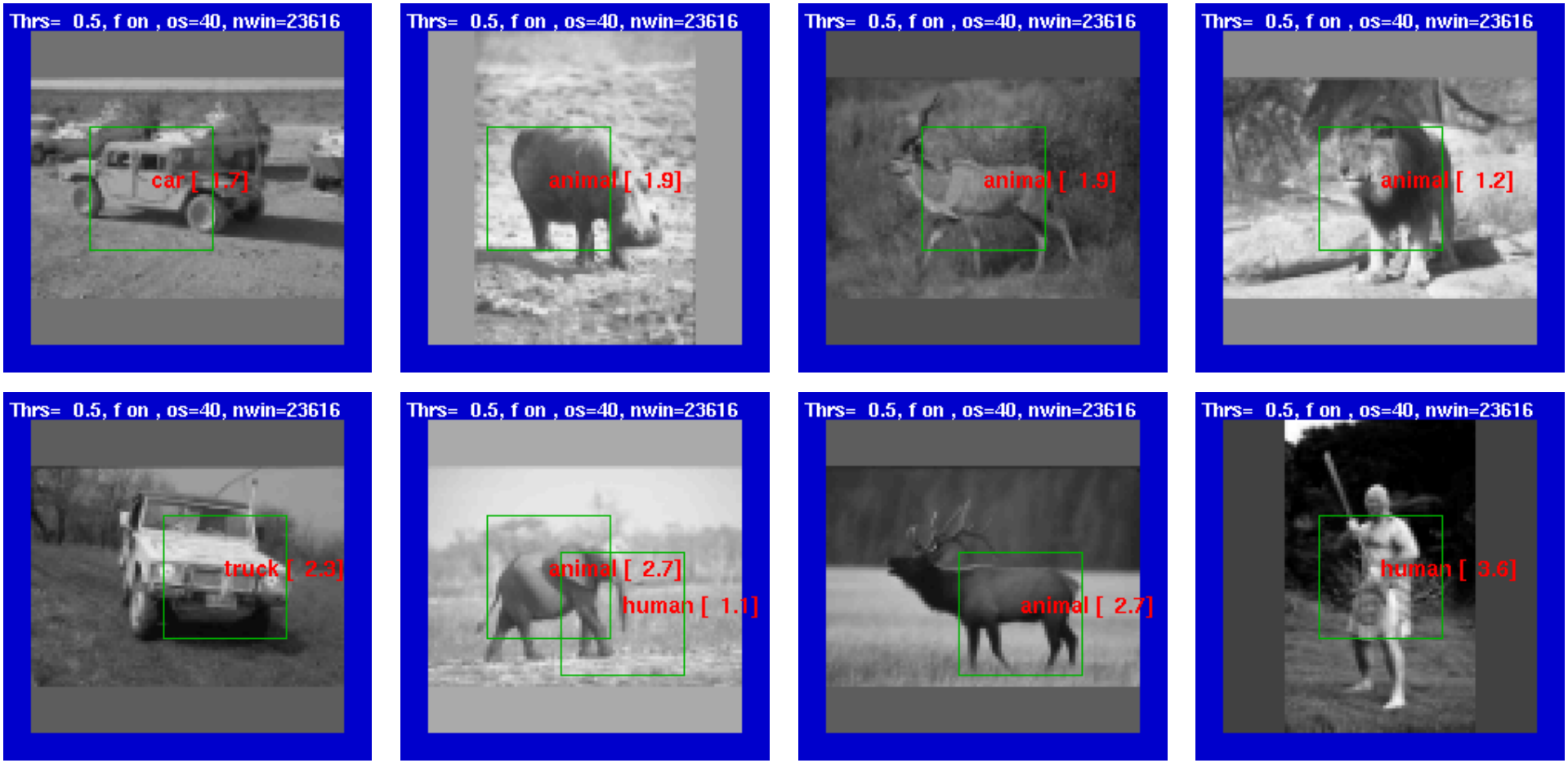
Examples (Monocular Mode)



Examples (Monocular Mode)



Natural Images (Monocular Mode)



Commercially Deployed applications of Convolutional Nets

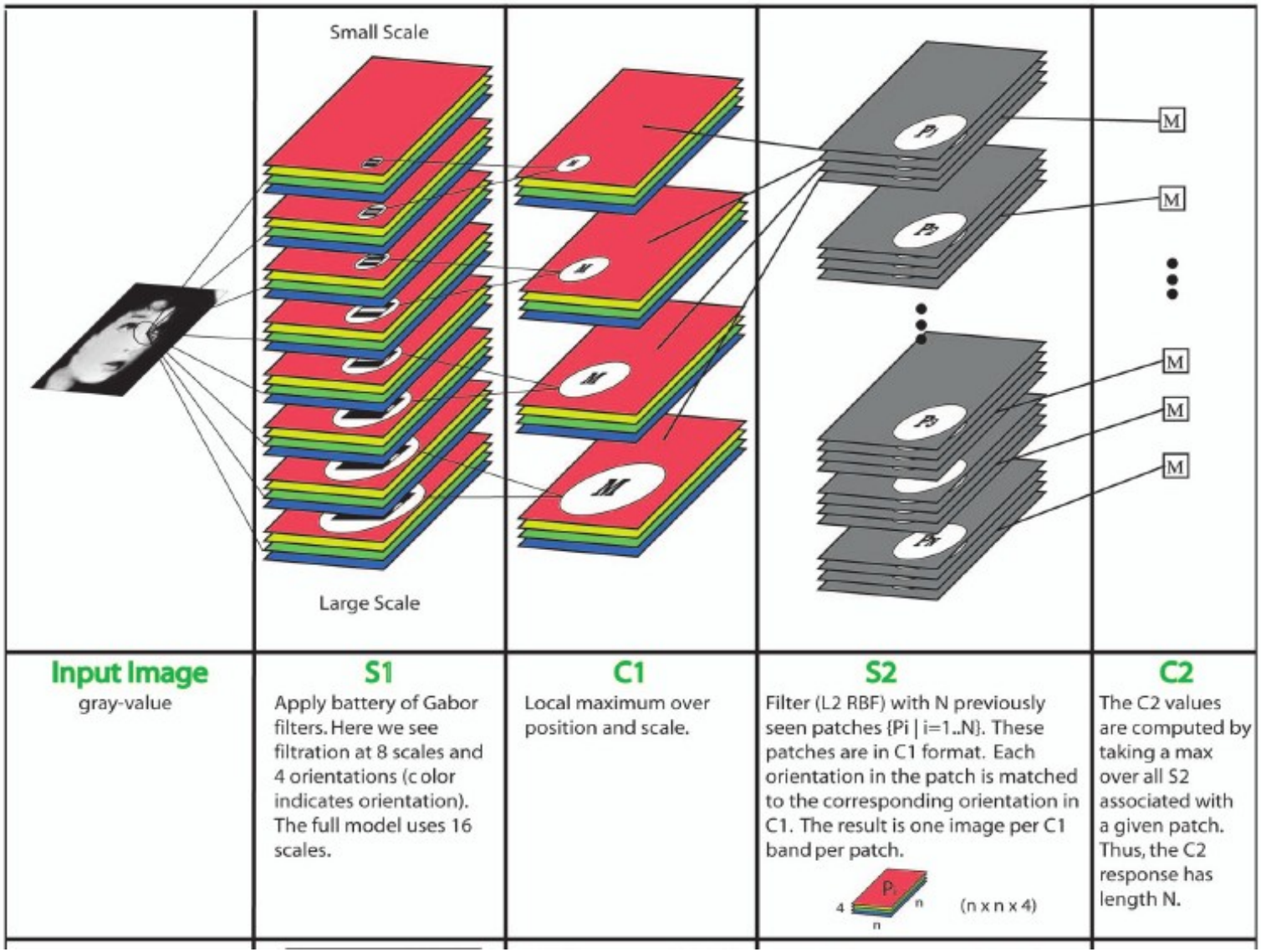
- **Faxed form reader**
  - Developed at AT&T Bell Labs in the early 90's
  - Commercially deployed in 1994
- **Check Reading system:**
  - Developed at AT&T Bell Labs in the mid 90's
  - Commercially deployed by NCR in 1996
  - First practical system for reading handwritten checks
  - Read 10 to 20% of all the checks in the US in the late 90's
- **Face detector / Person detector / Intrusion detector**
  - Developed at NEC Research Institute in 2002/2003
  - Commercially deployed in 2004 by Vident Technologies
  - Used at San Francisco Airport (among others).

Supervised Convolutional Nets: Pros and Cons

- **Convolutional nets can be trained to perform a wide variety of visual tasks.**
  - Global supervised gradient descent can produce parsimonious architectures
- **BUT: they require lots of labeled training samples**
  - 60,000 samples for handwriting
  - 120,000 samples for face detection
  - 25,000 to 350,000 for object recognition
- **Since low-level features tend to be non task specific, we should be able to learn them unsupervised.**
- **Hinton has shown that layer-by-layer unsupervised "pre-training" can be used to initialize "deep" architectures**
  - [Hinton & Shalakhutdinov, Science 2006]
- **Can we use this idea to reduce the number of necessary labeled examples.**

Models Similar to ConvNets

- **HMAX**
  - [Poggio & Riesenhuber 2003]
  - [Serre et al. 2007]
  - [Mutch and Low CVPR 2006]
- **Difference?**
  - the features are not learned
- **HMAX is very similar to Fukushima's Neocognitron**



[from Serre et al. 2007]

Part 3:
Unsupervised Training of "Deep" Energy-Based Models, Learning Invariant Feature Hierarchies

- **Why do we need Deep Learning?**
  - "scaling learning algorithms towards AI" [Bengio and LeCun 2007]
- **Deep Belief Networks, Deep Learning**
  - Stacked RBM [Hinton, Osindero, and Teh, Neural Comp 2006]
  - Stacked autoencoders [Bengio et al. NIPS 2006]
  - Stacked sparse features [Ranzato & al., NIPS 2006]
  - Improved stacked RBM [Salakhutdinov & Hinton, AI-Stats 07]
- **Unsupervised Learning of Invariant Feature Hierarchies**
  - learning features for Caltech-101 [Ranzato et al. CVPR 2006]
  - learning features hierarchies for hand-writing [Ranzato et al ICDAR'07]

[See Mar'cAurelio Ranzato's poster on Wednesday]

Why do we need "Deep" Architectures?

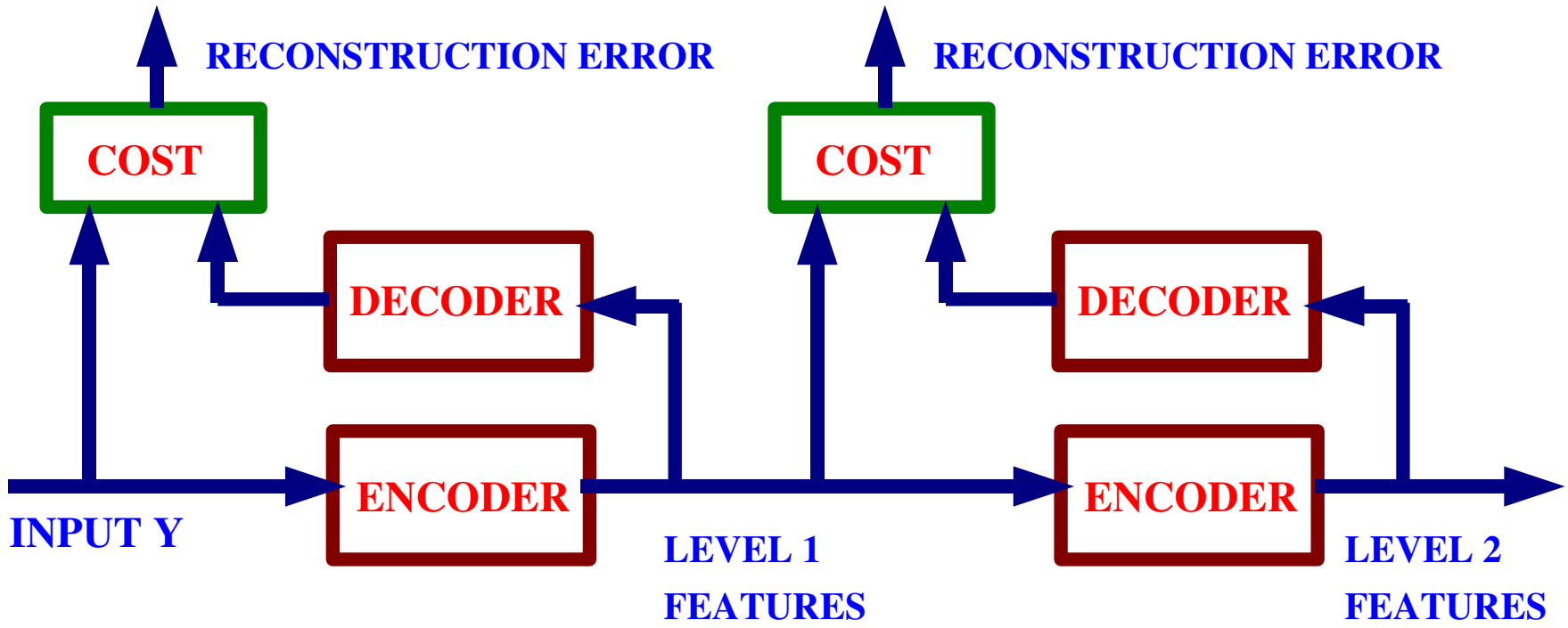
[Bengio & LeCun 2007]

- **Conjecture: we won't solve the perception problem without solving the problem of learning in deep architectures [Hinton]**
  - Neural nets with lots of layers
  - Deep belief networks
  - Factor graphs with a "Markov" structure
- **We will not solve the perception problem with kernel machines**
  - Kernel machines are glorified template matchers
  - You can't handle complicated invariances with templates (you would need too many templates)
- **Many interesting functions are "deep"**
  - Any function can be approximated with 2 layers (linear combination of non-linear functions)
  - But many interesting functions a more efficiently represented with multiple layers
  - Stupid examples: binary addition

The Basic Idea of Deep Learning

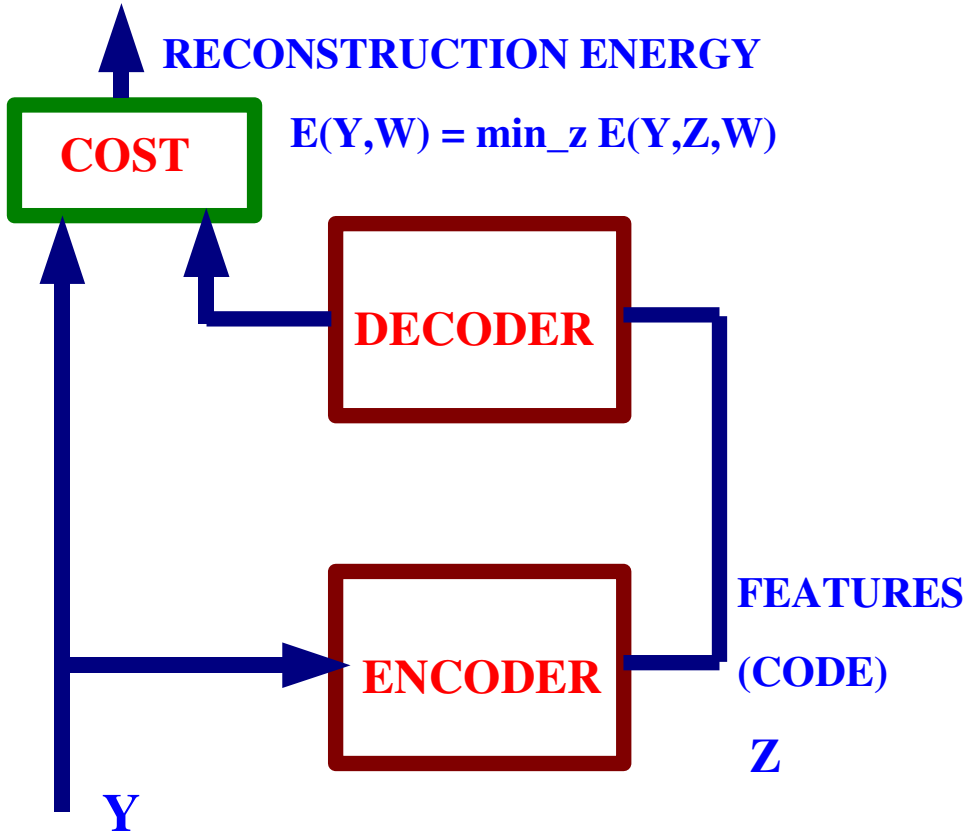
[Hinton et al. 2005 - 2007]

- **Unsupervised Training of Feature Hierarchy [Hinton et al. 2005 – 2007]**
  - Each layer is designed to extract higher-level features from lower-level ones
  - Each layer is trained unsupervised with a reconstruction criterion
  - The layers are trained one after the other, in sequence.



Encoder-Decoder Architecture for Unsupervised Learning

- **A principle on which unsupervised algorithms can be built is reconstruction of the input from a code (feature vector)**
  - reconstruction from compact feature vectors (e.g. PCA).
  - reconstruction from sparse overcomplete feature vectors [Olshausen & Field 1997], [Ranzato et al NIPS 06].
  - approximation of data likelihood: Restricted Boltzmann Machine [Hinton 2005-...]



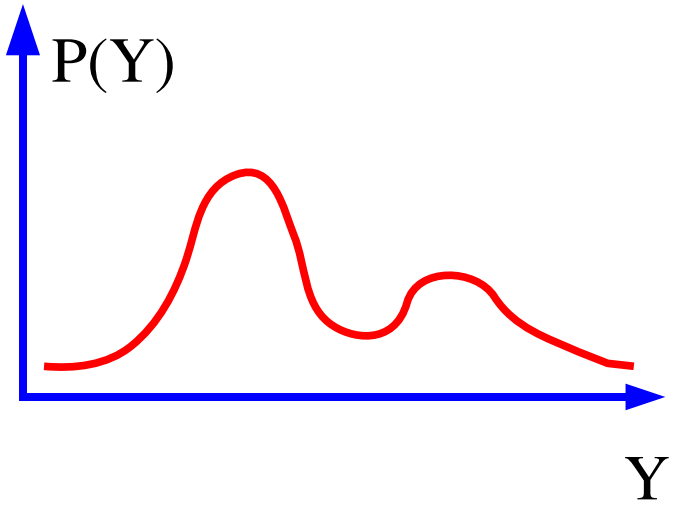
$$E(Y,W) = min_Z E(Y,Z,W)$$

$$\bar{Z}_Y = argmin_Z E(Y,Z,W)$$

What is Energy-Based Unsupervised Learning?

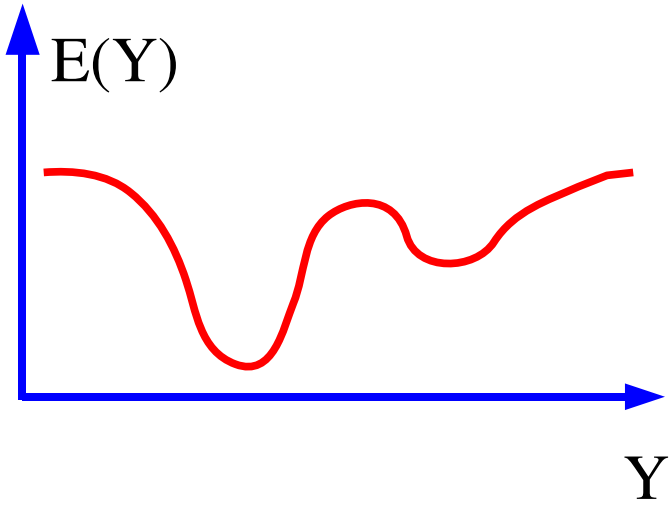
Probabilistic View:

- Produce a probability density function that:
- has high value in regions of high sample density
- has low value everywhere else (integral=1)
- **Training:** maximize the data likelihood **(intractable)**
- $$P(Y,W) = \frac{e^{-\beta E(Y,W)}}{\int_y e^{-\beta E(y,W)}}$$



Energy-Based View:

- produce an energy function E(Y) that:
- has low value in regions of high sample density
- has high(er) value everywhere else



Unsupervised Training of Energy-Based Models

- **Basic Idea:**
  - push down on the energy of training samples
  - pull up on the energy of everything else
  - **but this is often intractable**
- **Approximation #1: Contrastive Divergence [Hinton et al 2005]**
  - Push down on the energy of the training samples
  - Pull up on the energies of configuration that have low energy **near the training samples** (to create local minima of the energy surface)
- **Approximation #2: Minimizing the information content of the code [Ranzato et al. AI-Stats 2007]**
  - Reduce the information content of the code **by making it sparse**
  - This has the effect of increasing the reconstruction error for non-training samples.

Deep Learning for Non-Linear Dimensionality Reduction

- **Restricted Boltzmann Machine.**
  - simple energy function

$$E(Y,Z,W)=\sum_{ij}-Y_i W_{ij} Z_j$$

  - code units are binary stochastic
  - training with contrastive divergence

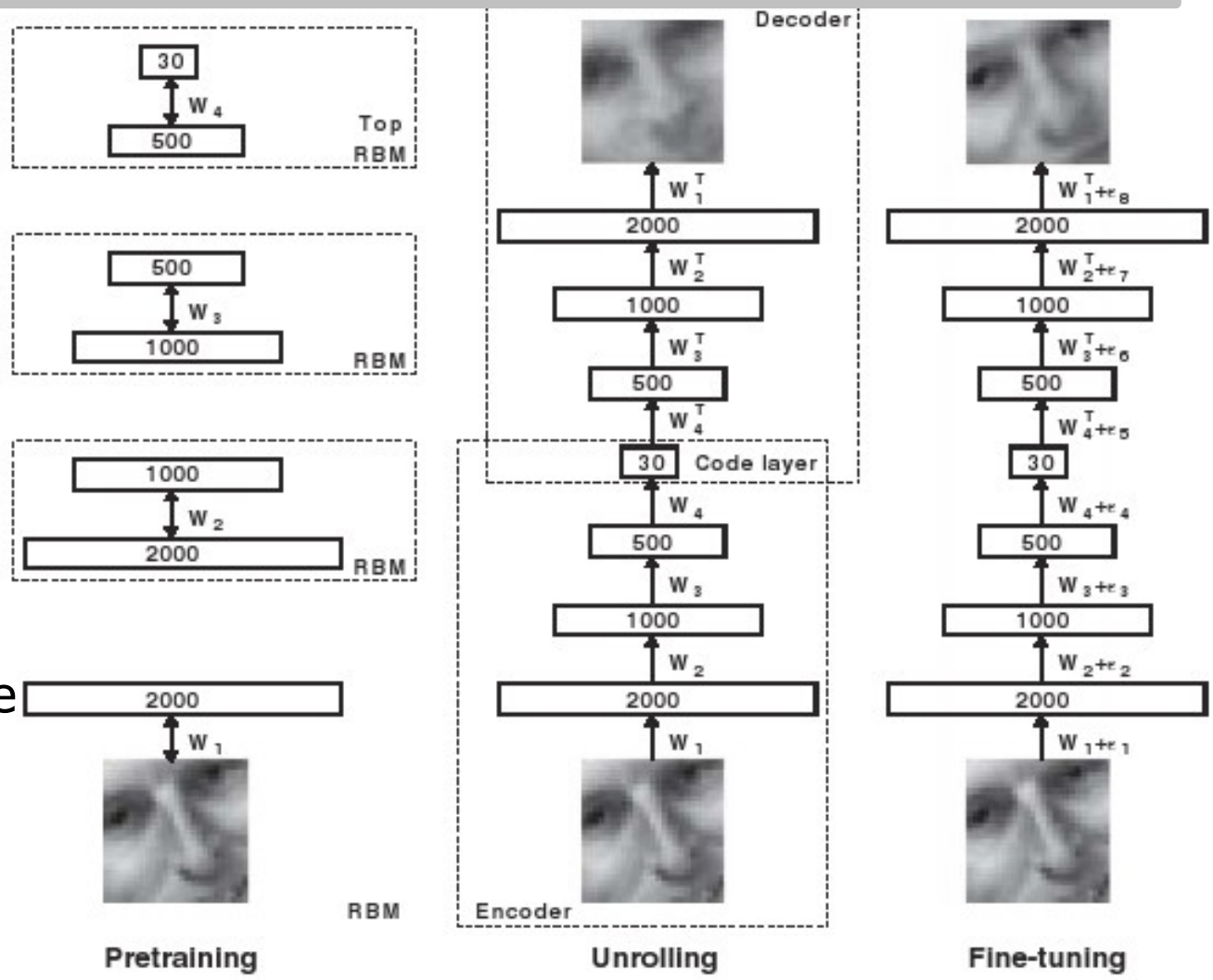
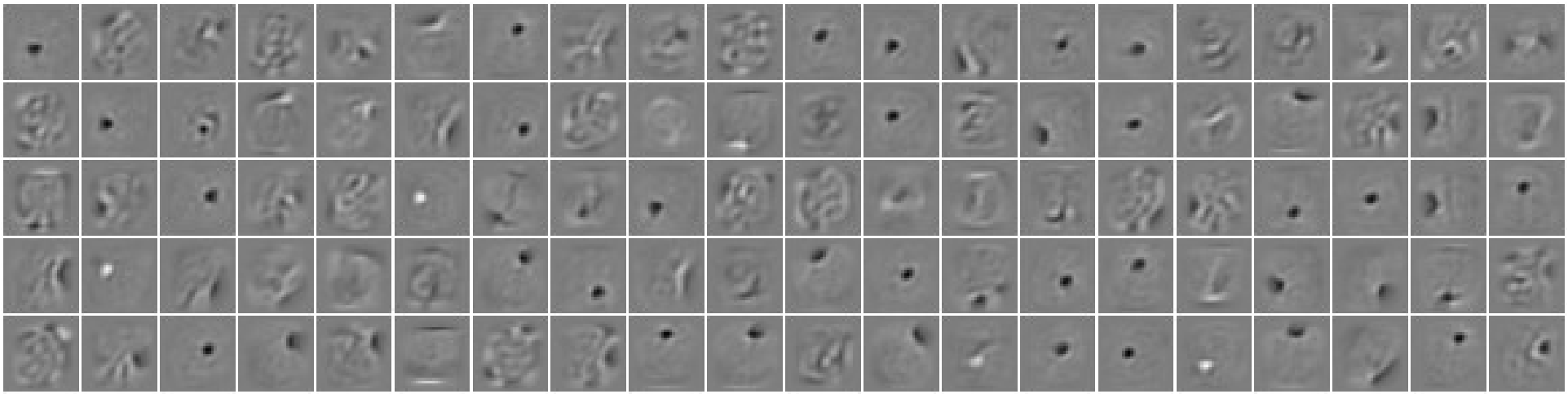


Fig. 1. Pretraining consists of learning a stack of restricted Boltzmann machines (RBMs), each having only one layer of feature detectors. The learned feature activations of one RBM are used as the "data" for training the next RBM in the stack. After the pretraining, the RBMs are "unrolled" to create a deep autoencoder, which is then fine-tuned using backpropagation of error derivatives.

- **From: [Hinton and Salakhutdinov, Science 2006]**

RBM: filters trained on MNIST

- "bubble" detectors



Non-Linear Dimensionality Reduction: MNIST

- [Hinton and Salakhutdinov, Science 2006]

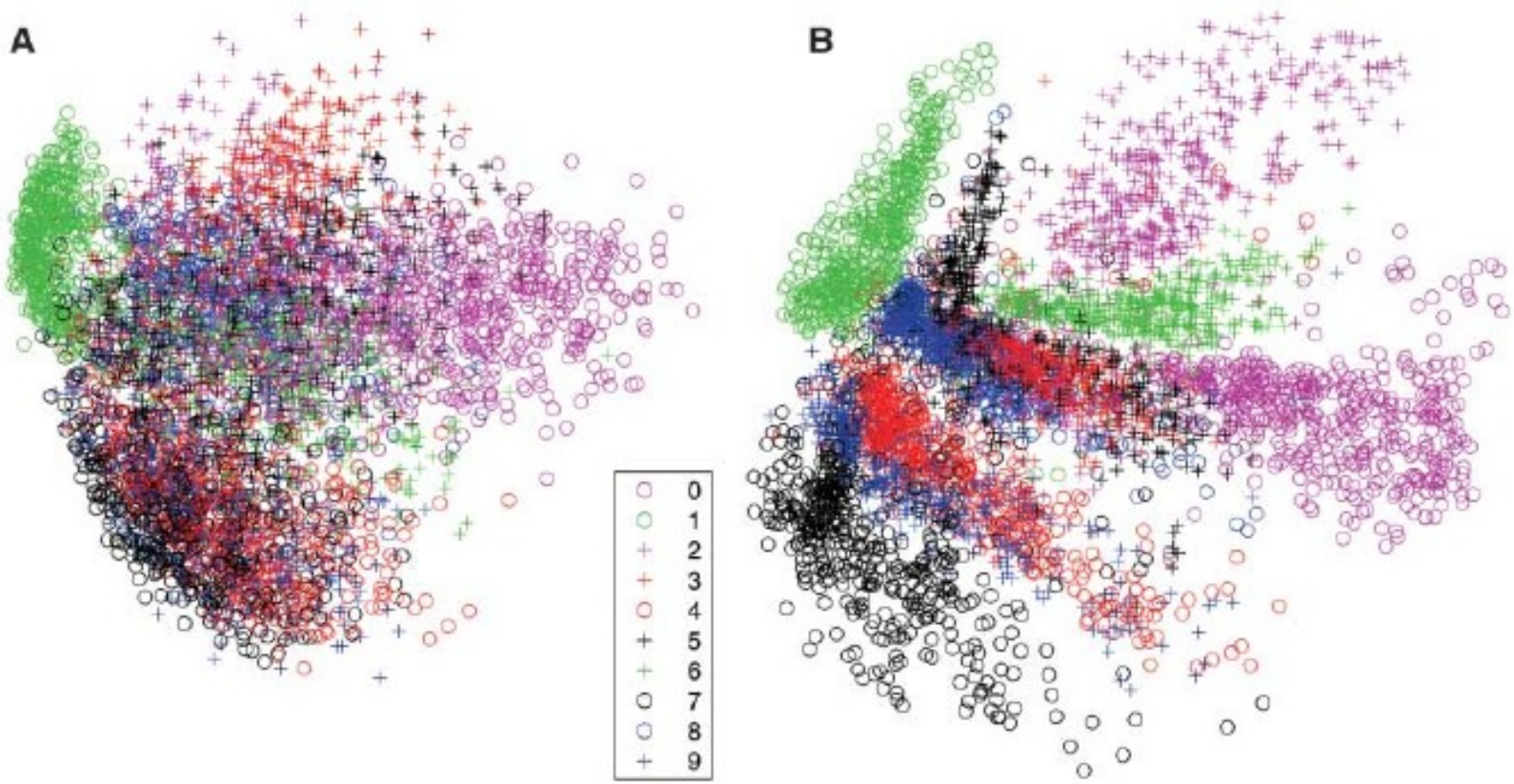


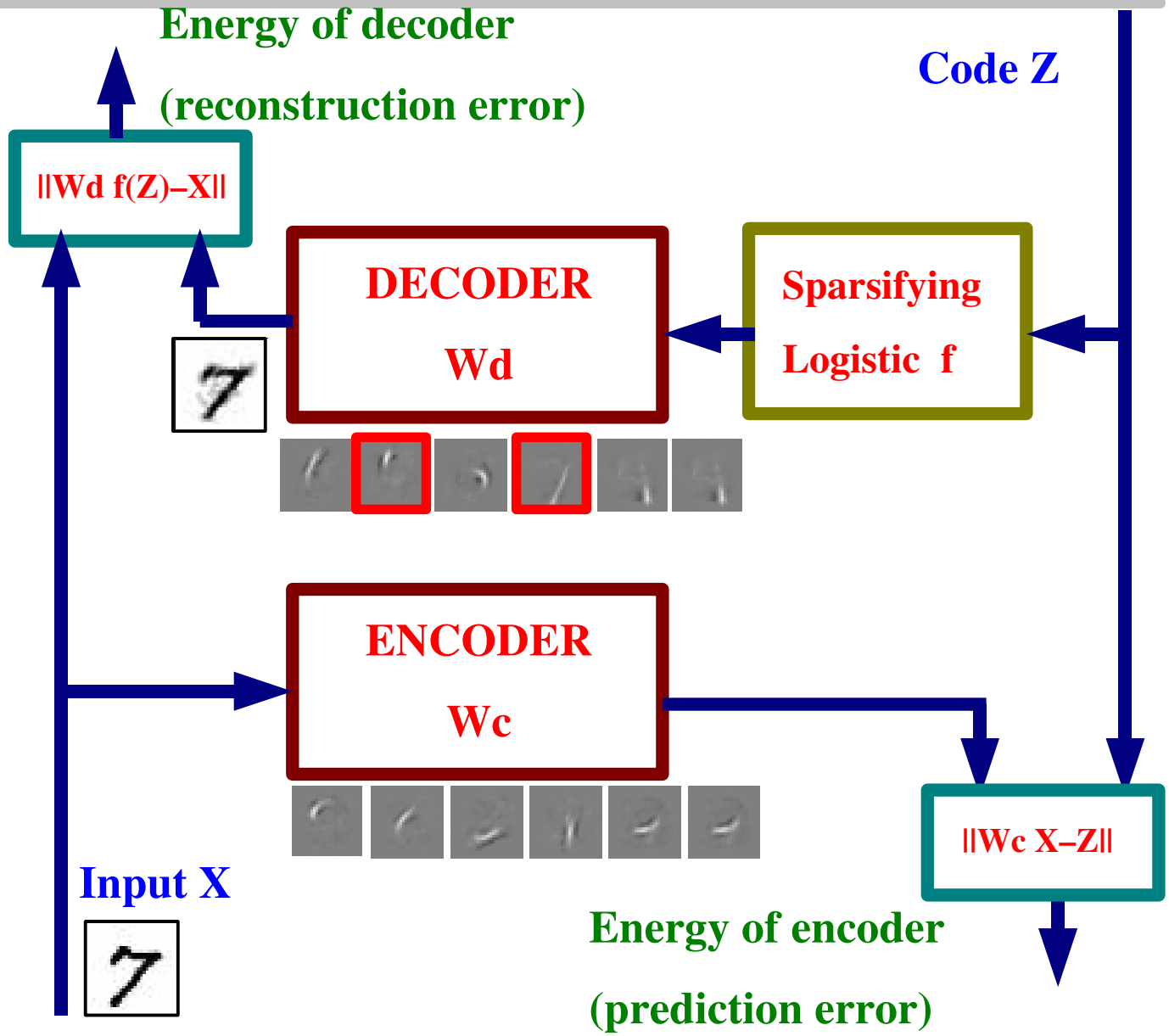
Fig. 3. (**A**) The two-dimensional codes for 500 digits of each class produced by taking the first two principal components of all 60,000 training images. (**B**) The two-dimensional codes found by a 784-1000-500-250-2 autoencoder. For an alternative visualization, see (*8*).

- [Salakhutdinov and Hinton, AI-Stats 2007]:
  - < 1.00% error on MNIST using K-NN on 30 dimensions:
  - **BEST ERROR RATE OF ANY KNOWEDGE-FREE METHODS!!!**

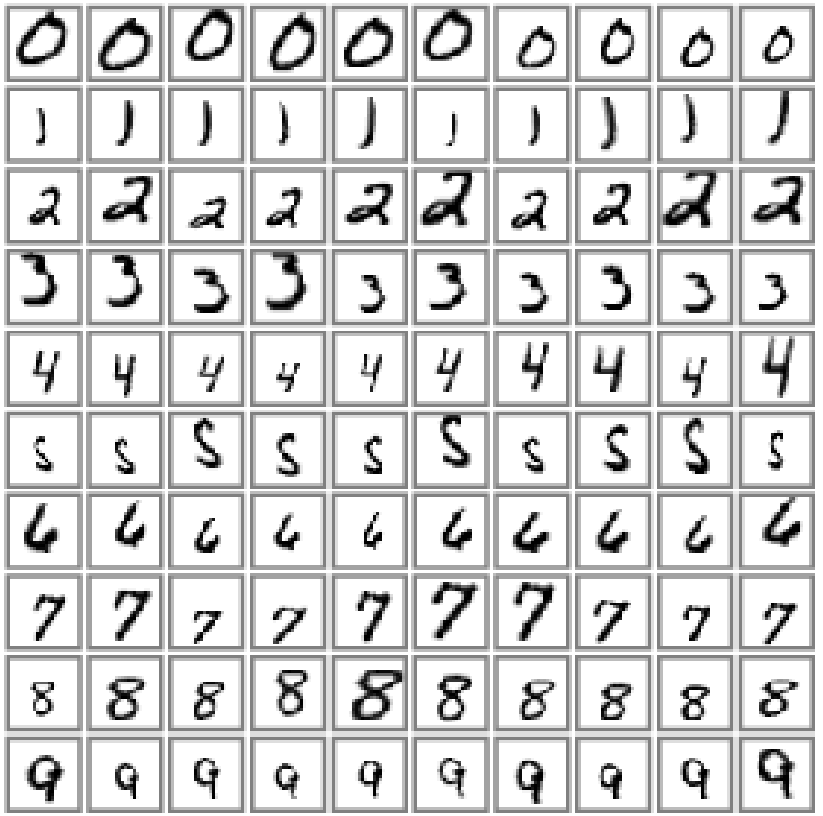
Encoder/Decoder Architecture for learning Sparse Feature Representations

Algorithm:

- 1. find the code Z that minimizes the reconstruction error AND is close to the encoder output
- 2. Update the weights of the decoder to decrease the reconstruction error
- 3. Update the weights of the encoder to decrease the prediction error

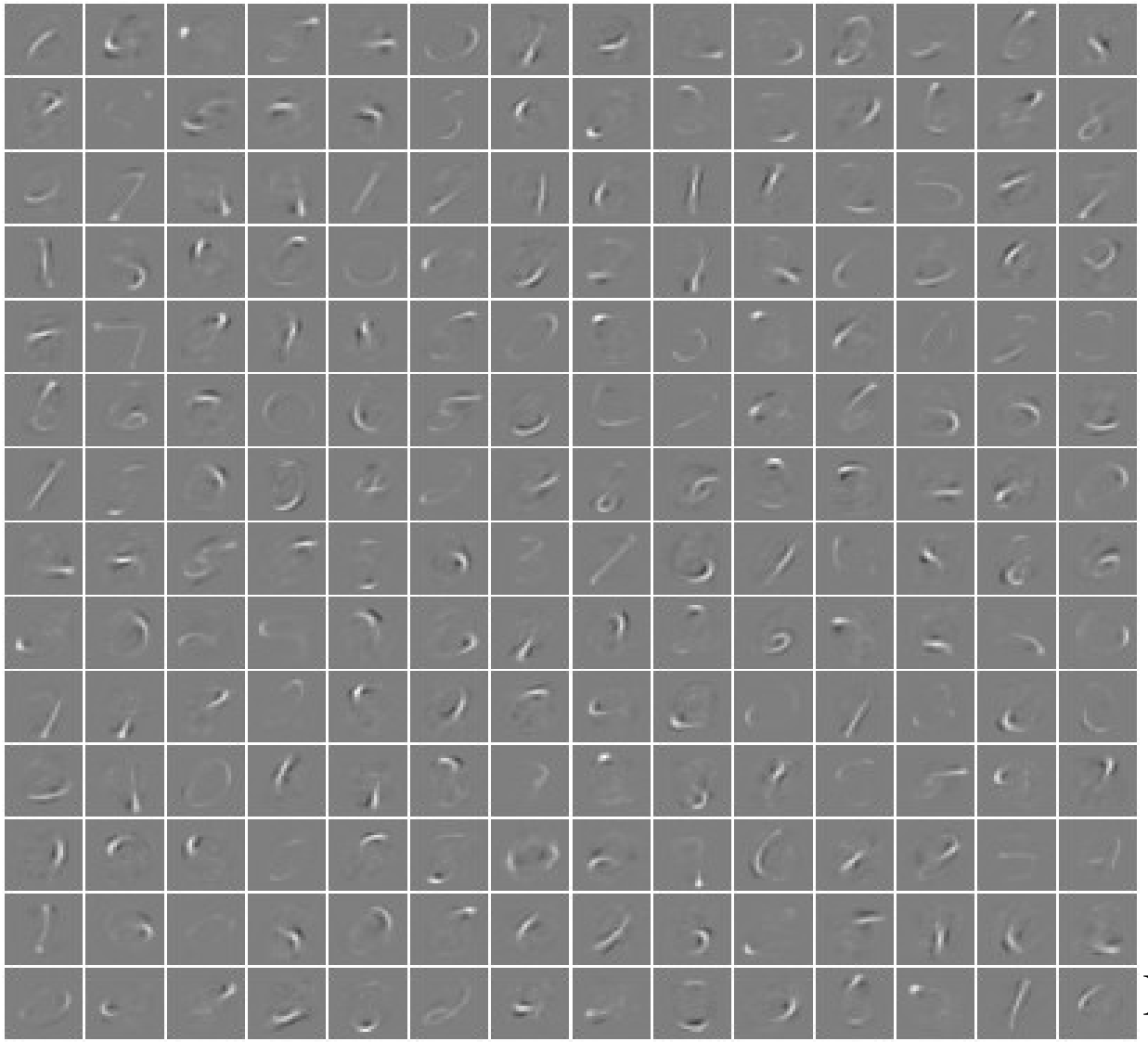


MNIST Dataset



- Handwritten Digit Dataset MNIST: 60,000 training samples, 10,000 test samples

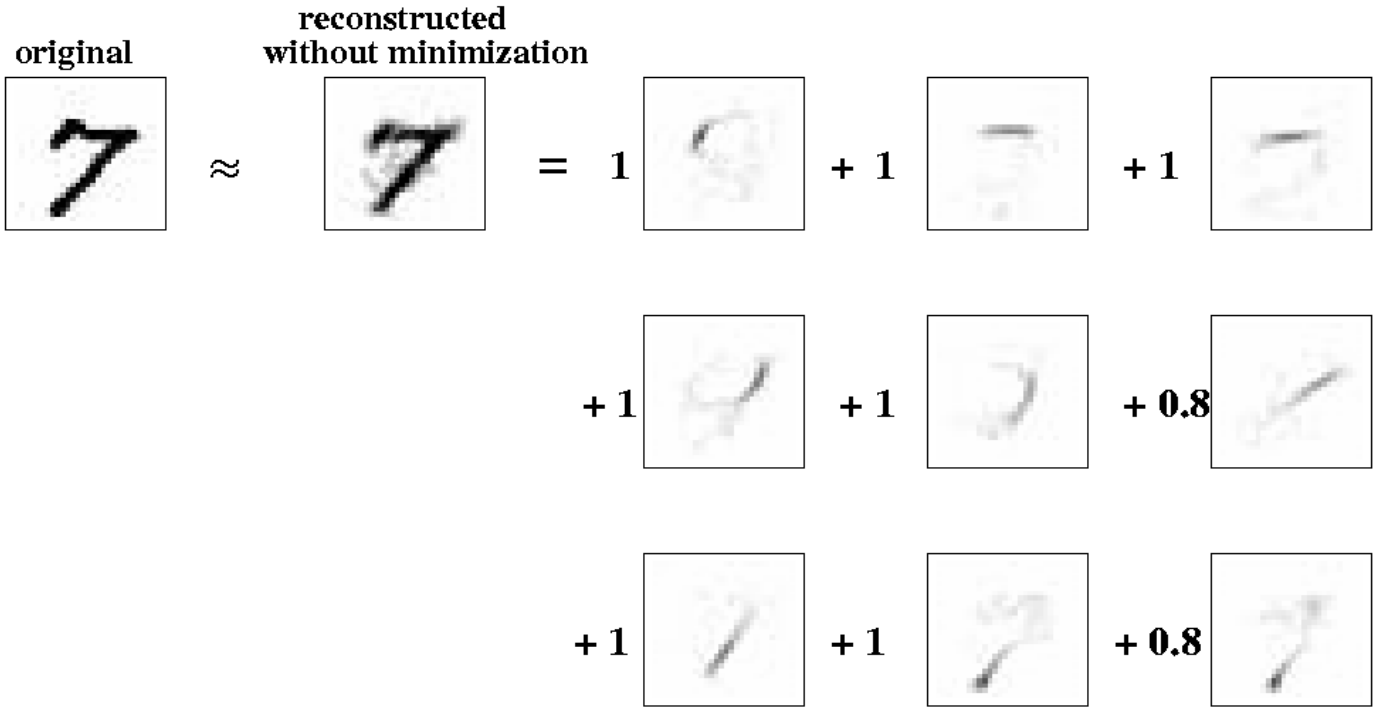
Training on handwritten digits

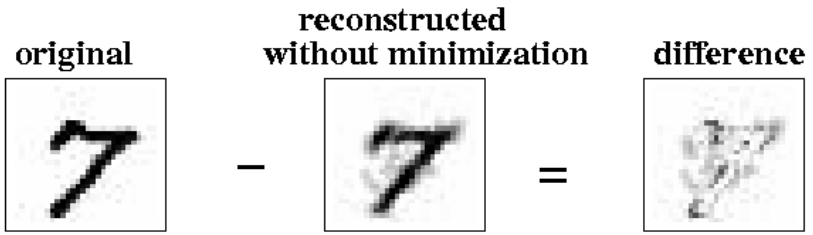


- 60,000 28x28 images
- 196 units in the code
- η 0.01
- β 1
- learning rate 0.001
- L1, L2 regularizer 0.005

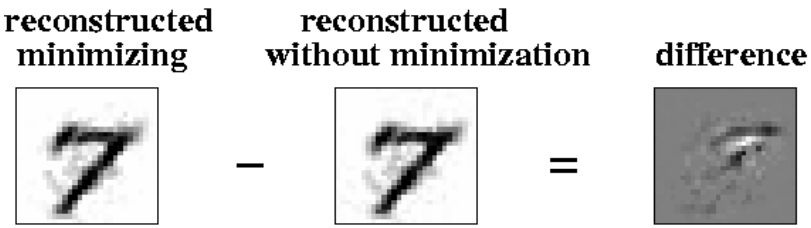
Encoder *direct* filters

Handwritten digits - MNIST





forward propagation through encoder and decoder



after training there is no need to minimize in code space

Training The Layers of a Convolutional Net Unsupervised

- Extract windows from the MNIST images
- Train the sparse encoder/decoder on those windows
- Use the resulting encoder weights as the convolution kernels of a convolution network
- Repeat the process for the second layer
- Train the resulting network supervised.

Unsupervised Training of Convolutional Filters

CLASSIFICATION EXPERIMENTS

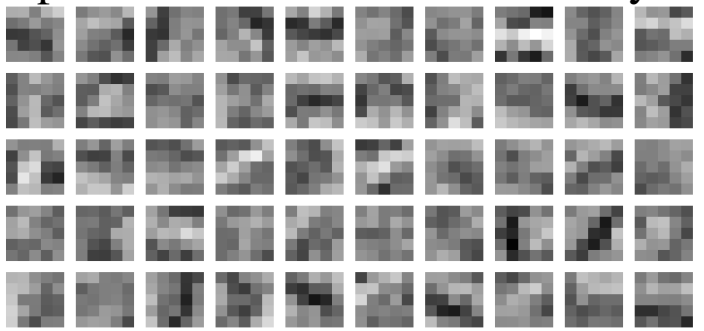
IDEA: improving supervised learning by pre-training with the unsupervised method (*)

sparse representations & *lenet6* (1->50->50->200->10)

- The baseline: *lenet6* initialized randomly

Test error rate: **0.70%**. Training error rate: 0.01%.

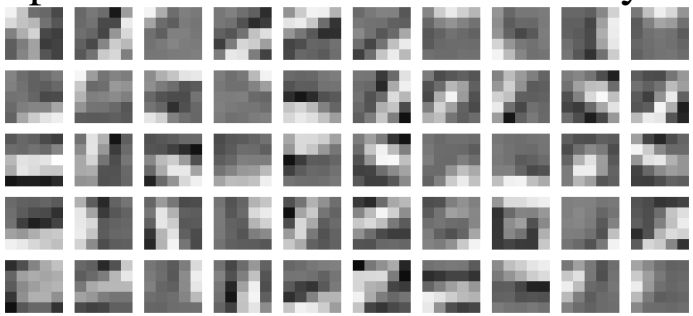
supervised filters in first conv. layer



- *Experiment 1*
  - Train on 5x5 patches to find 50 features
  - Use the scaled filters in the encoder to initialize the kernels in the first convolutional layer

Test error rate: 0.60%. Training error rate: 0.00%.

unsupervised filters in first conv. layer



- *Experiment 2*
  - Same as experiment 1, but training set augmented by elastically distorted digits (random initialization gives test error rate equal to **0.49%**).

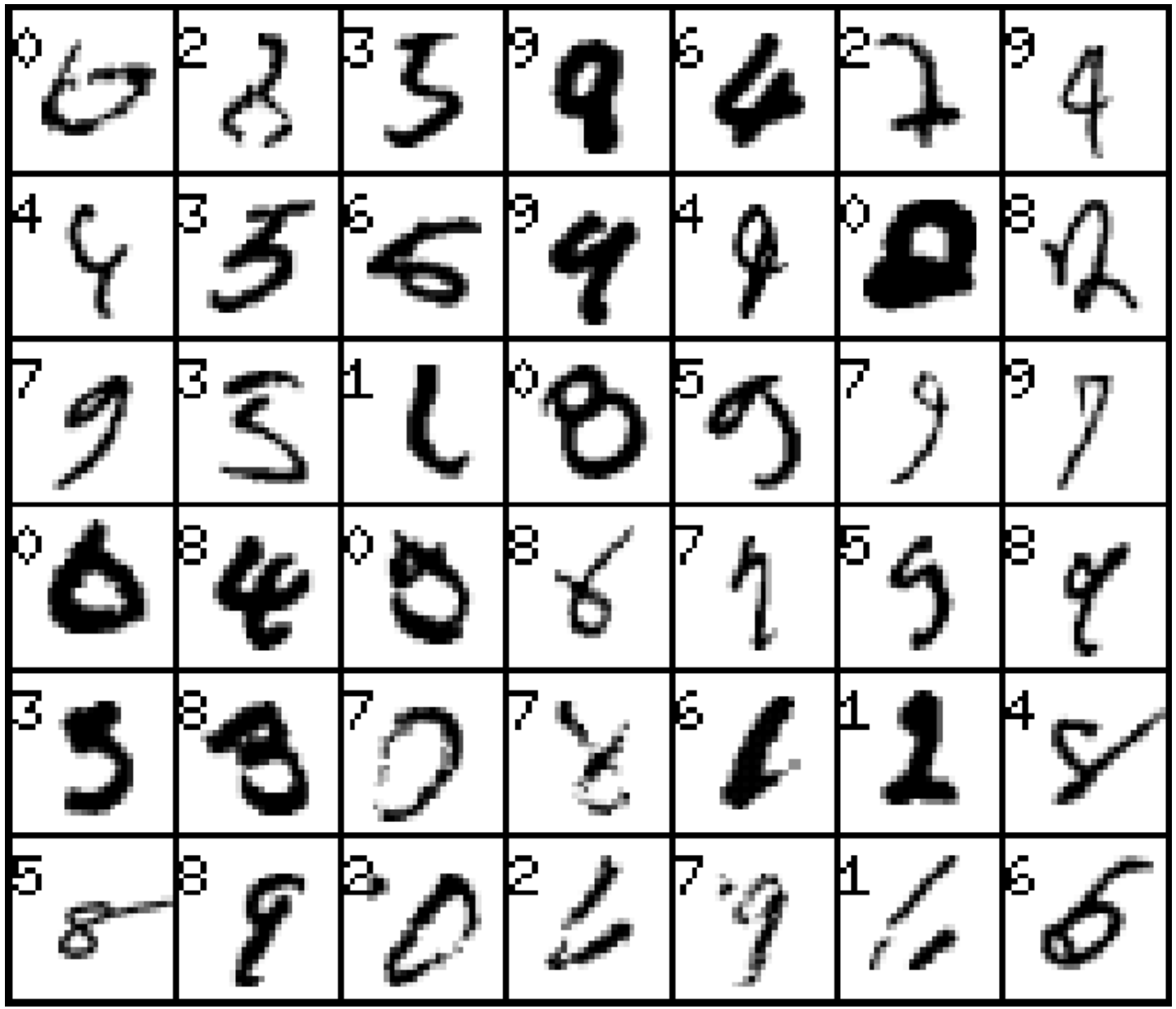
Test error rate: 0.39%. Training error rate: 0.23%.

(*)[Hinton, Osindero, Teh "*A fast learning algorithm for deep belief nets*" Neural Computaton 2006]

Best Results on MNIST (from raw images: no preprocessing)

CLASSIFIER	DEFORMATION	ERROR	Reference
Knowledge-free methods			
2-layer NN, 800 HU, CE		1.60	Simard et al., ICDAR 2003
3-layer NN, 500+300 HU, CE, reg		1.53	Hinton, in press, 2005
SVM, Gaussian Kernel		1.40	Cortes 92 + Many others
Unsupervised Stacked RBM + backprop		0.95	Hinton, Neur Comp 2006
Convolutional nets			
Convolutional net LeNet-5,		0.80	Ranzato et al. NIPS 2006
Convolutional net LeNet-6,		0.70	Ranzato et al. NIPS 2006
Conv. net LeNet-6- + unsup learning		0.60	Ranzato et al. NIPS 2006
Training set augmented with Affine Distortions			
2-layer NN, 800 HU, CE	Affine	1.10	Simard et al., ICDAR 2003
Virtual SVM deg-9 poly	Affine	0.80	Scholkopf
Convolutional net, CE	Affine	0.60	Simard et al., ICDAR 2003
Training et augmented with Elastic Distortions			
2-layer NN, 800 HU, CE	Elastic	0.70	Simard et al., ICDAR 2003
Convolutional net, CE	Elastic	0.40	Simard et al., ICDAR 2003
Conv. net LeNet-6- + unsup learning	Elastic	0.39	Ranzato et al. NIPS 2006

MNIST Errors (0.42% error)

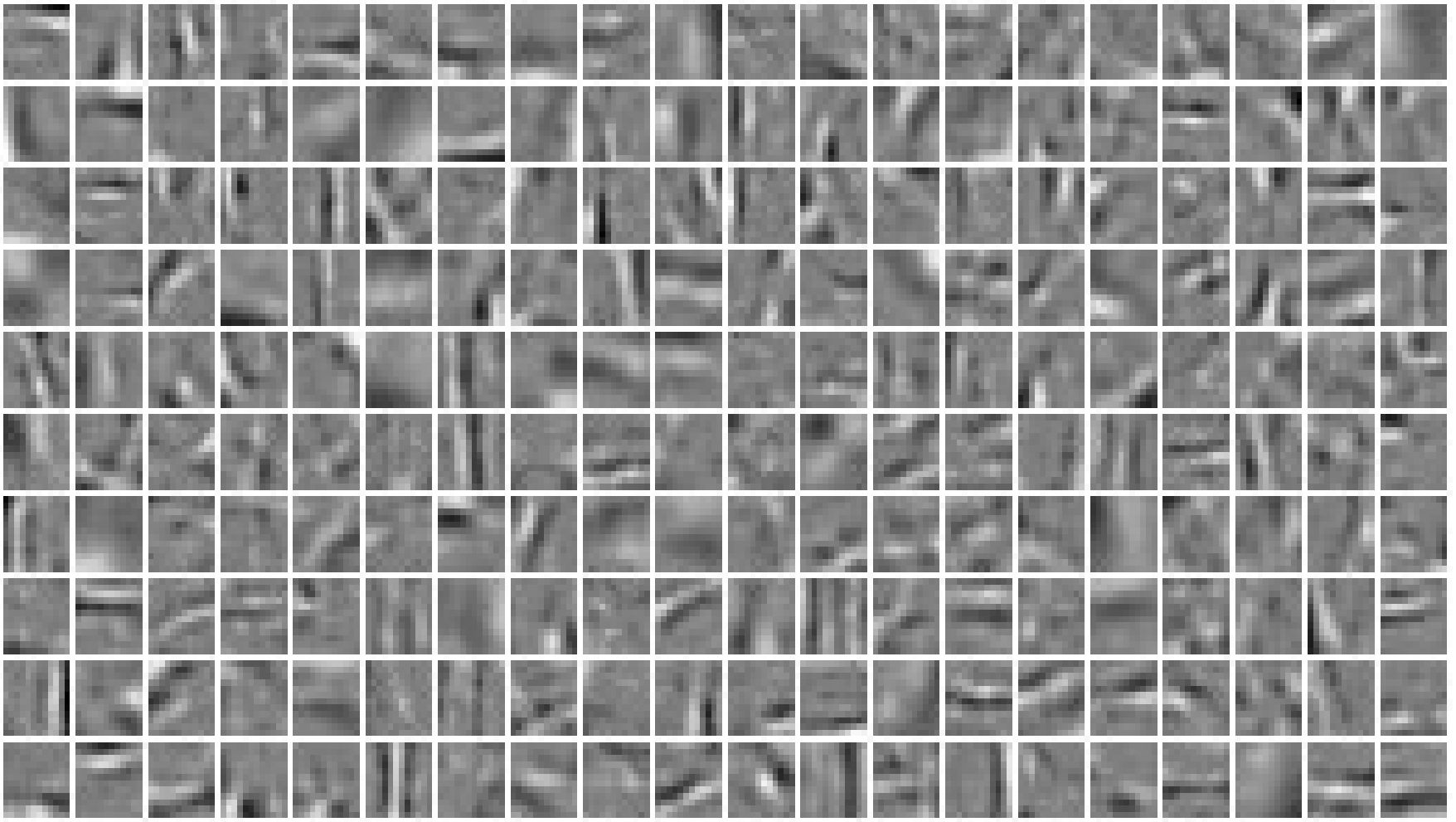


Training on natural image patches

Berkeley data set

- 100,000 12x12 patches
- 200 units in the code
- η
- β 0.02
- 1
- learning rate 0.001
- L1 regularizer 0.001
- fast convergence: < 30min.

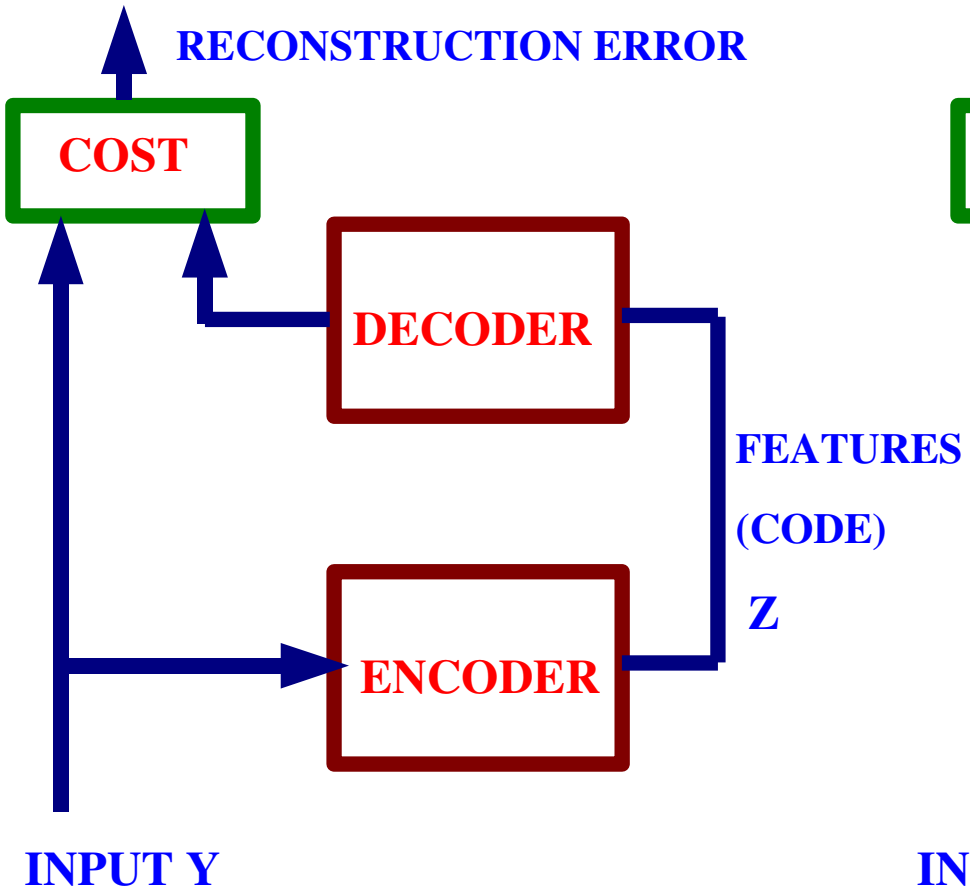
Natural image patches: Filters



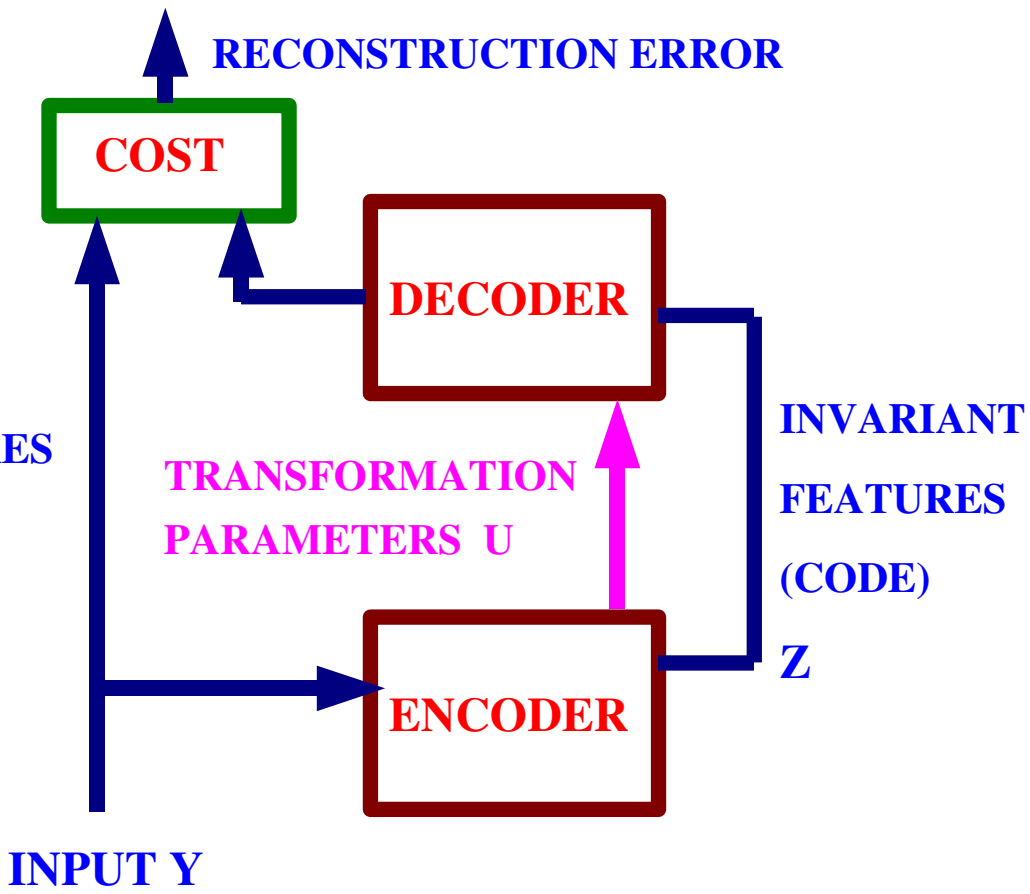
200 decoder filters (reshaped columns of matrix $\mathbf{W_d}$)

Learning Invariant Feature Hierarchies

Learning Shift Invariant Features



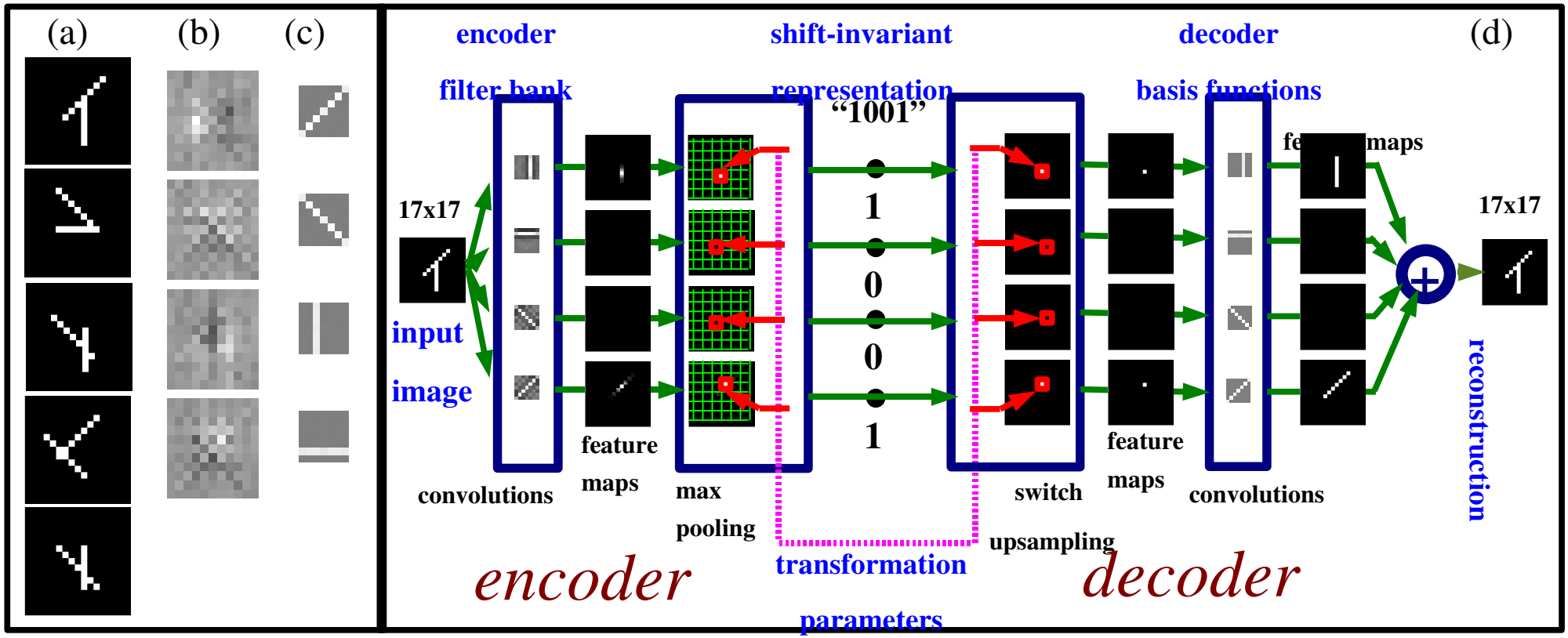
Standard Feature Extractor



Invariant Feature Extractor

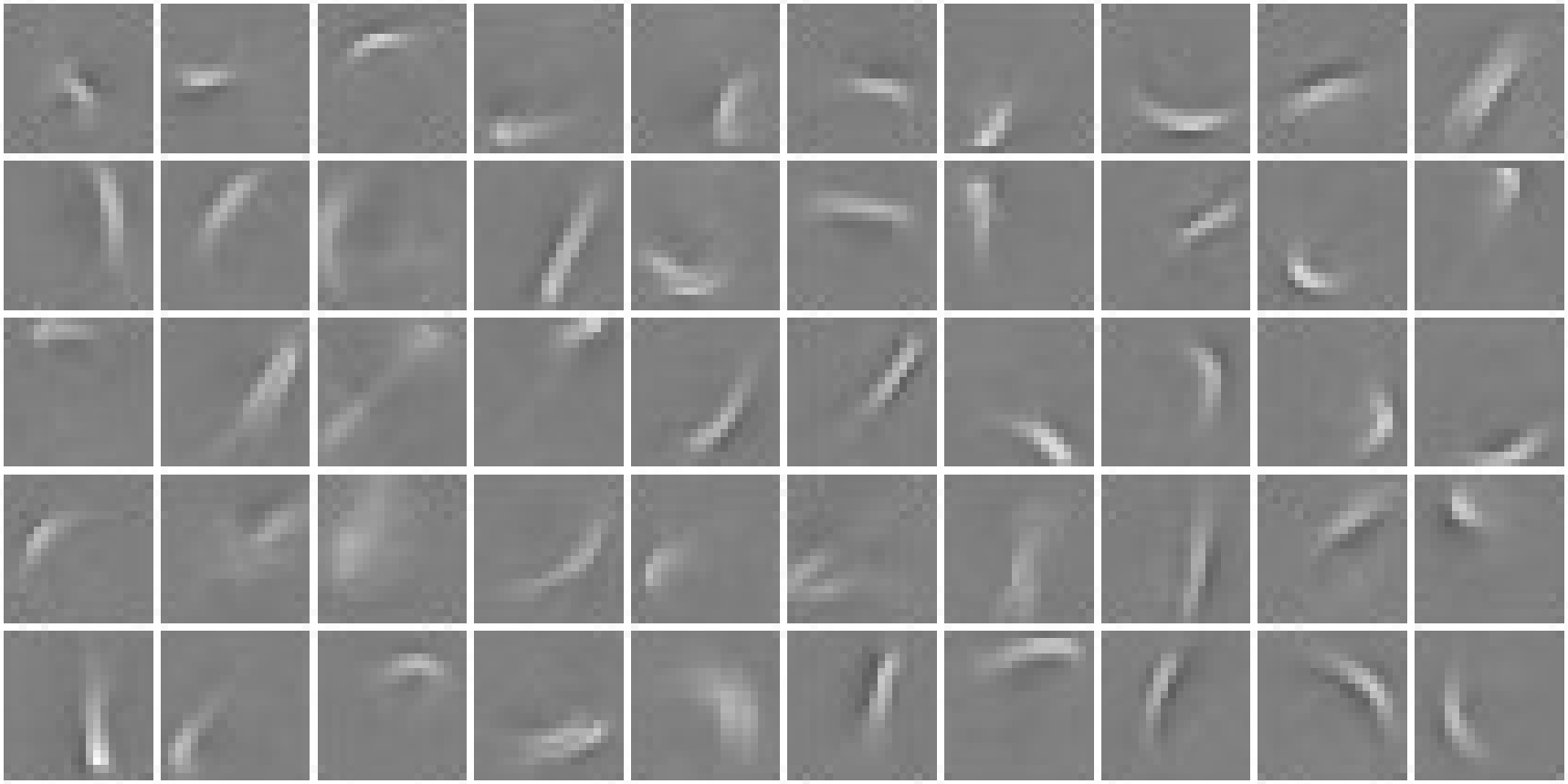
Learning Invariant Feature Hierarchies

Learning Shift Invariant Features



Shift Invariant Global Features on MNIST

- **Learning 50 Shift Invariant Global Features on MNIST:**
  - 50 filters of size 20x20 movable in a 28x28 frame (81 positions)
  - movable strokes!

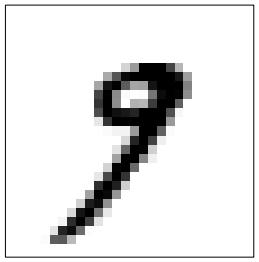
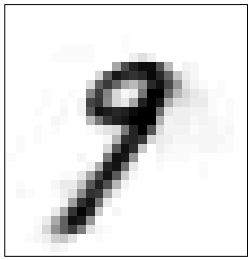


Example of Reconstruction

- Any character can be reconstructed as a linear combination of a small number of basis functions.

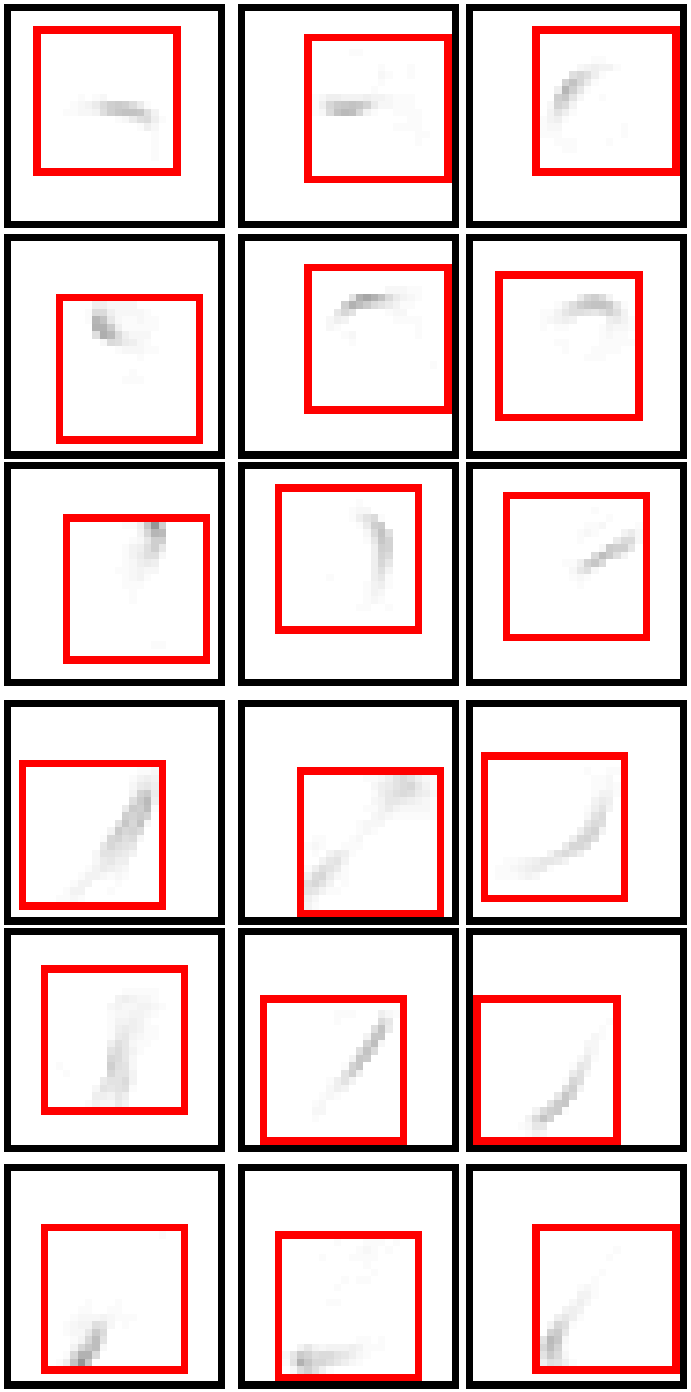
ORIGINAL DIGIT

RECONS-TRUCTION

$\sum$  $\cong$  $=$

ACTIVATED DECODER BASIS FUNCTIONS

(in feed-back layer)



red squares: decoder bases

Learning Invariant Filters in a Convolutional Net

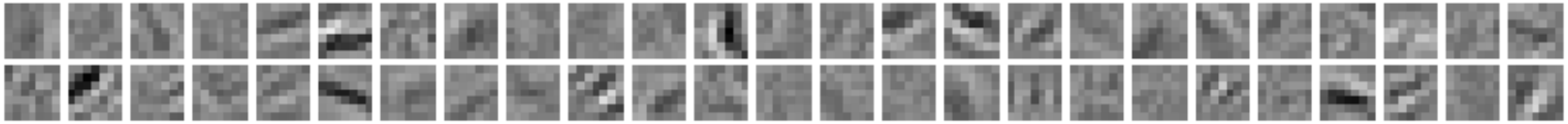


Figure 1: 50 7x7 filters in the first convolutional layer that were learned by the network trained supervised from *random* initial conditions with 600K digits.

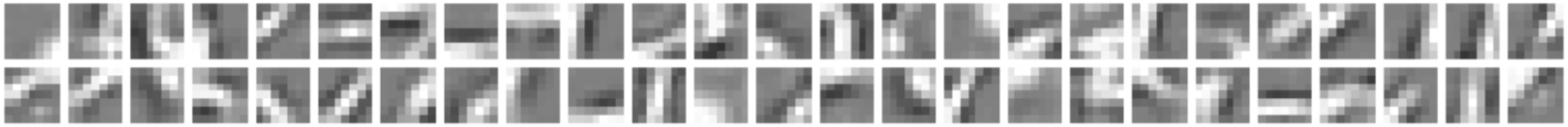


Figure 2: 50 7x7 filters that were learned by the unsupervised method (on 60K digits), and that are used to initialize the first convoltional layer of the network.

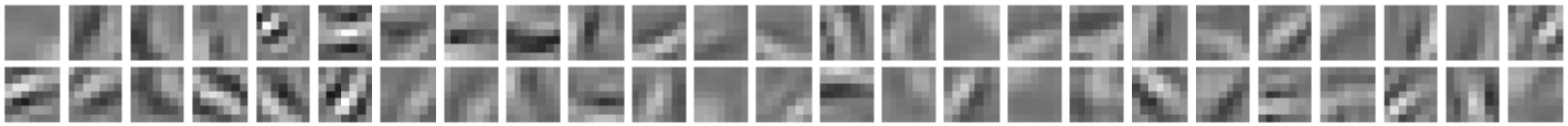
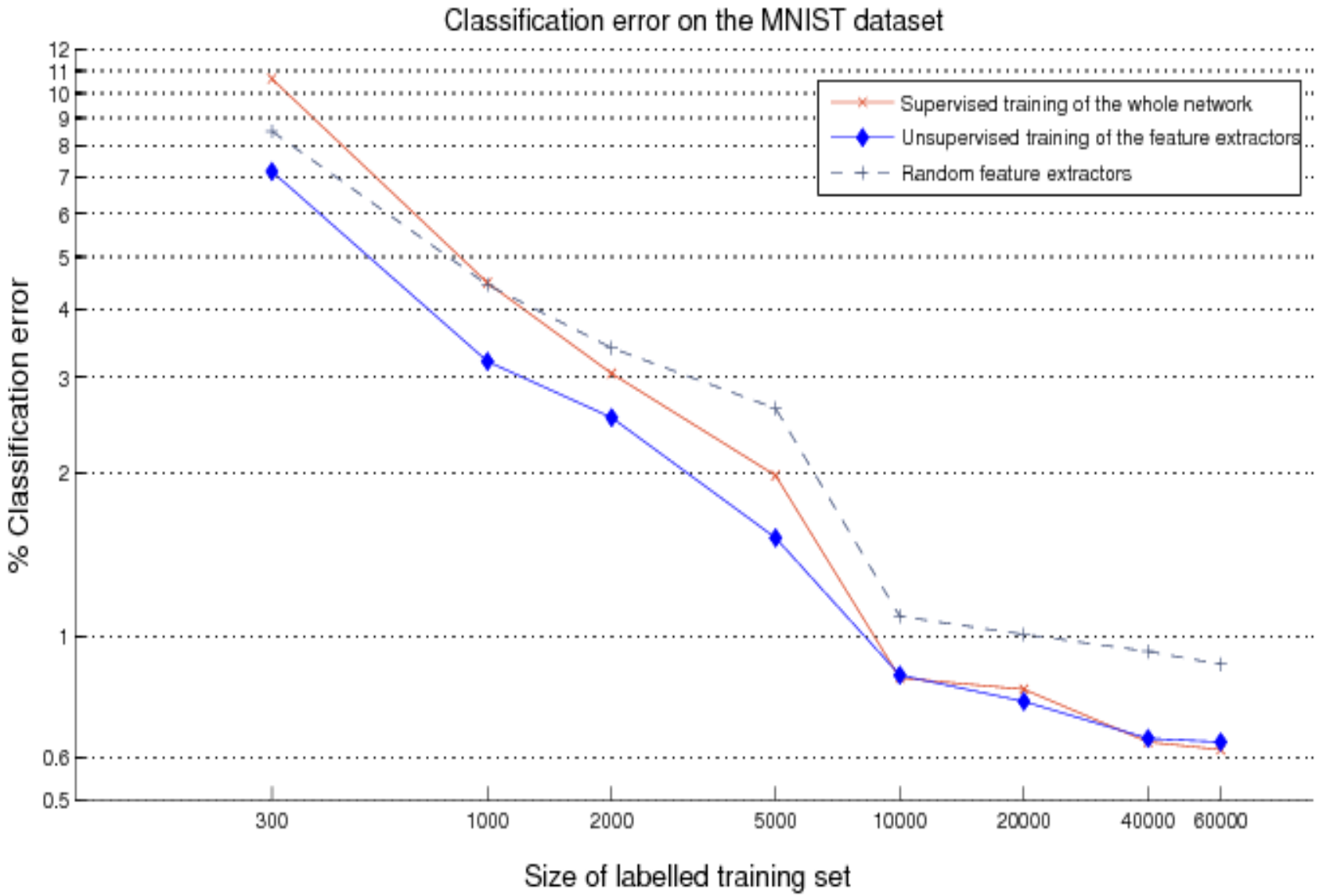


Figure 3: 50 7x7 filters in the first convolutional layer that were learned by the network trained supervised from the initial conditions given by the *unsupervised method* (see fig.2) with 600K digits.

Influence of Number of Training Samples

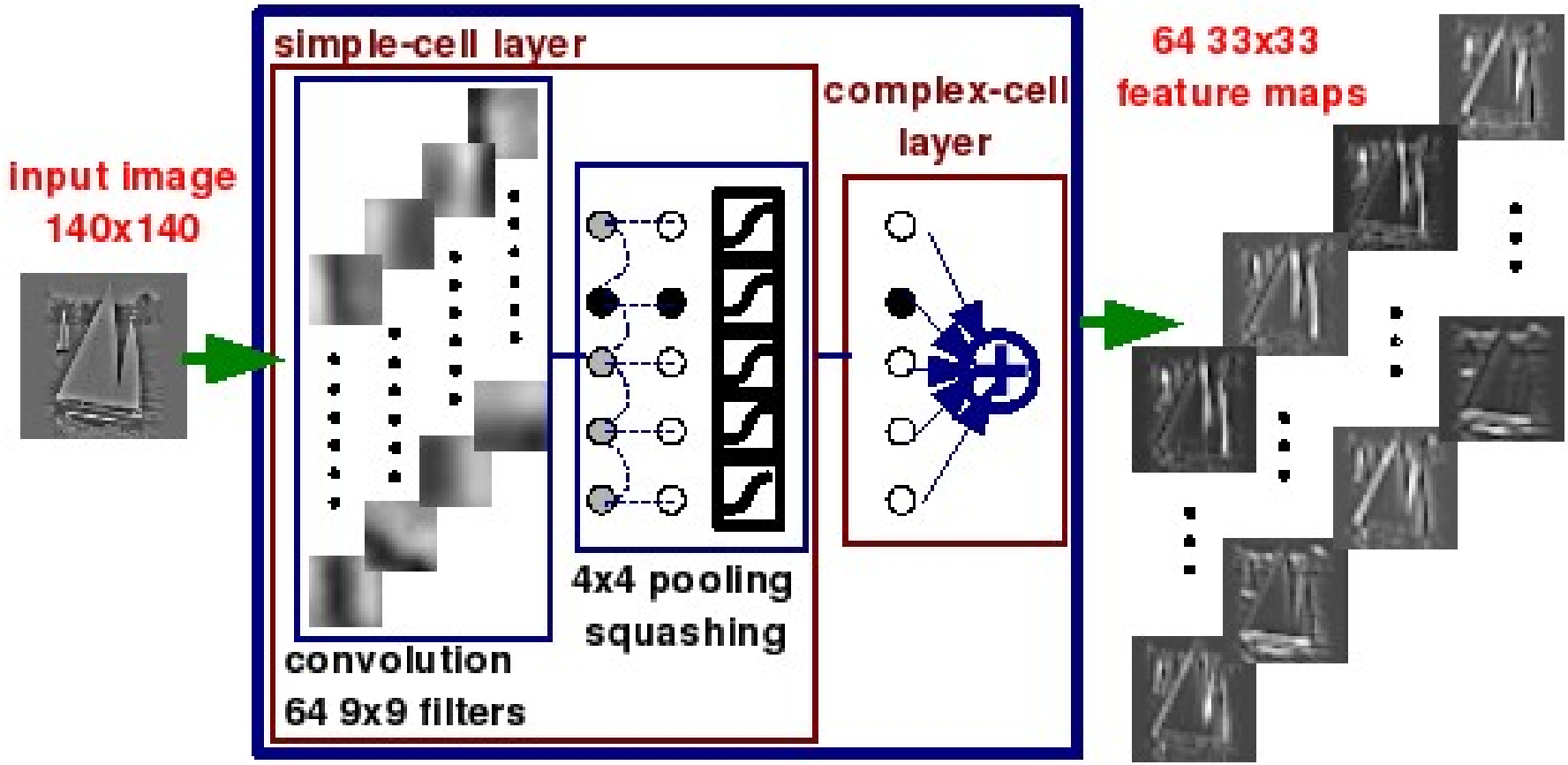


Generic Object Recognition: 101 categories + background

- **Caltech-101 dataset: 101 categories**
  - accordion airplanes anchor ant barrel bass beaver binocular bonsai brain brontosaurus buddha butterfly camera cannon car_side ceiling_fan cellphone chair chandelier cougar_body cougar_face crab crayfish crocodile crocodile_head cup dalmatian dollar_bill dolphin dragonfly electric_guitar elephant emu euphonium ewer Faces Faces_easy ferry flamingo flamingo_head garfield gerenuk gramophone grand_piano hawksbill headphone hedgehog helicopter ibis inline_skate joshua_tree kangaroo ketch lamp laptop Leopards llama lobster lotus mandolin mayfly menorah metronome minaret Motorbikes nautilus octopus okapi pagoda panda pigeon pizza platypus pyramid revolver rhino rooster saxophone schooner scissors scorpion sea_horse snoopy soccer_ball stapler starfish stegosaurus stop_sign strawberry sunflower tick trilobite umbrella watch water_lilly wheelchair wild_cat windsor_chair wrench yin_yang
- **Only 30 training examples per category!**
- **A convolutional net trained with backprop (supervised) gets 20% correct recognition.**
- **Training the filters with the sparse invariant unsupervised method**

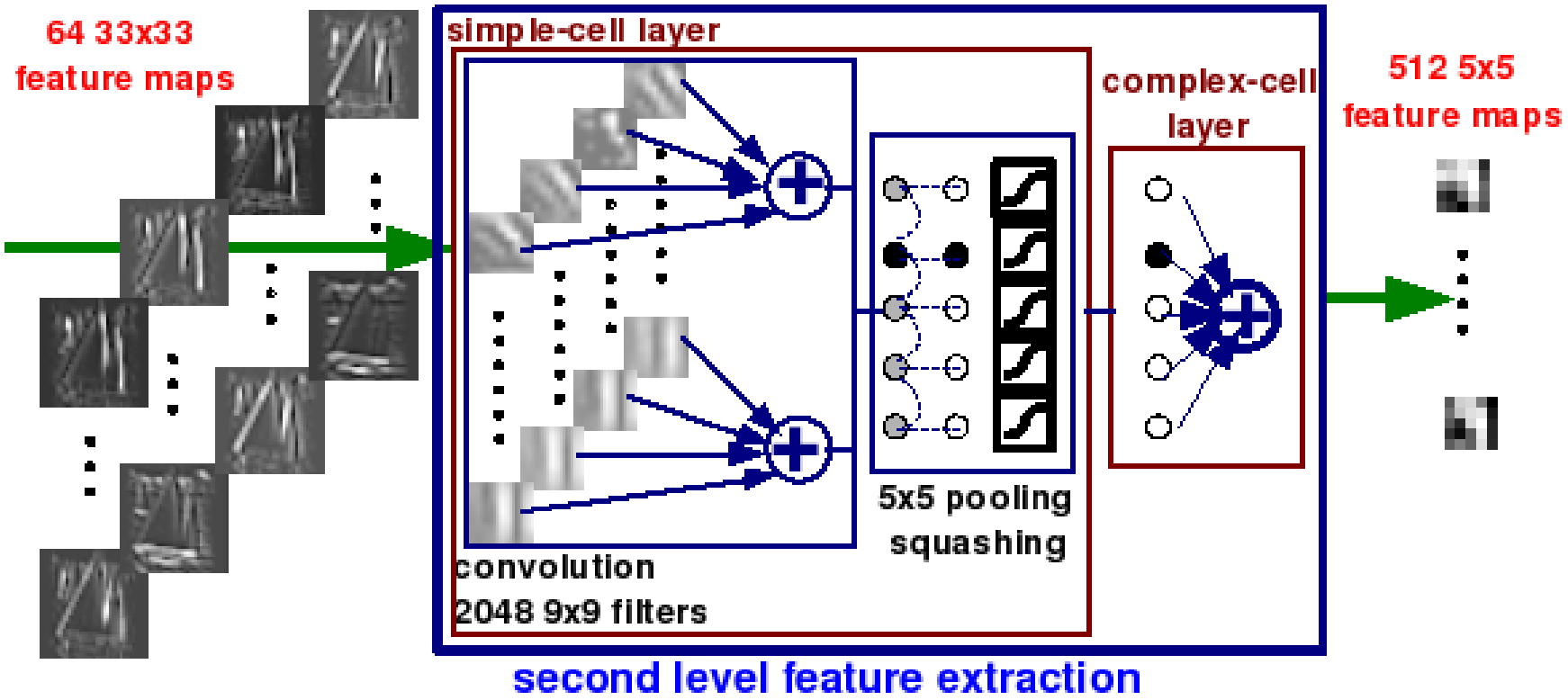
Training the 1st stage filters

- 12x12 input windows (complex cell receptive fields)
- 9x9 filters (simple cell receptive fields)
- 4x4 pooling



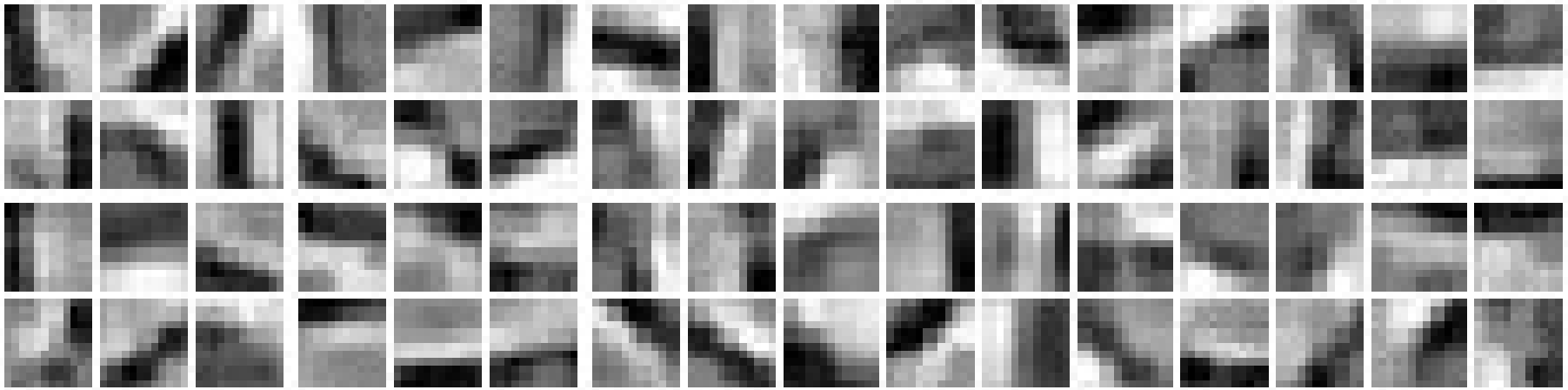
Training the 2nd stage filters

- 13x13 input windows (complex cell receptive fields on 1st features)
- 9x9 filters (simple cell receptive fields)
- Each output feature map combines 4 input feature maps
- 5x5 pooling

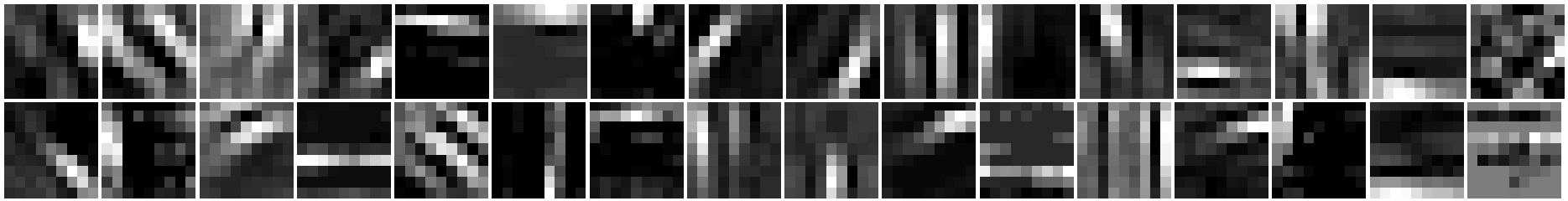


Generic Object Recognition: 101 categories + background

- 9x9 filters at the first level

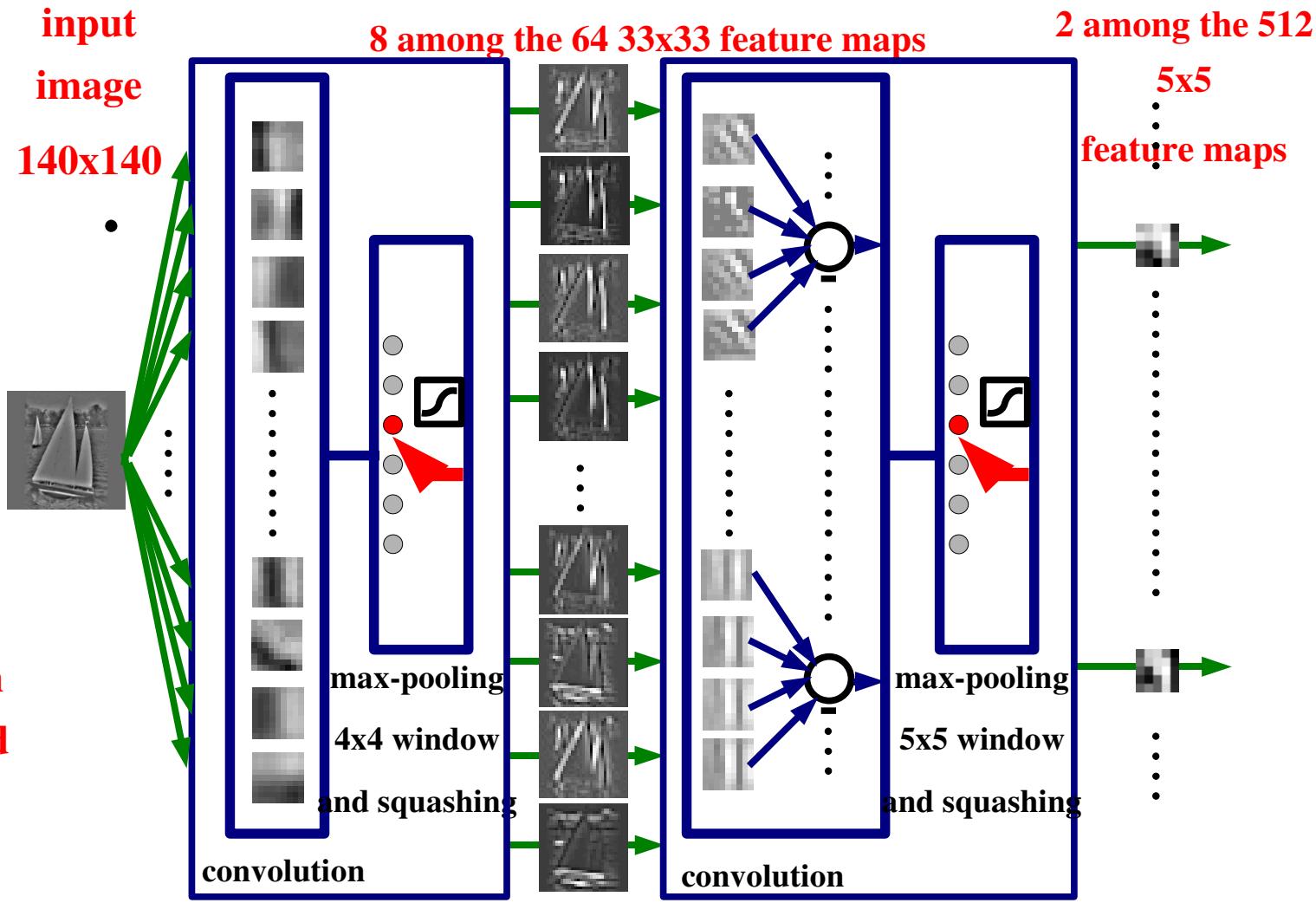


- 9x9 filters at the second level



Shift-Invariant Feature Hierarchies on Caltech-101

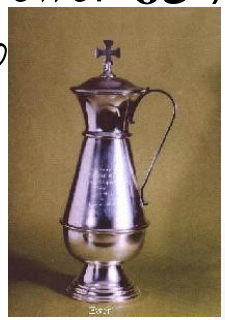
- 2 layers of filters trained unsupervised
- supervised classifier on top.
- 54% correct on Caltech-101 with 30 examples per class
- 20% correct with purely supervised backprop



Recognition Rate on Caltech 101


dollar 100%


w. chair 92%

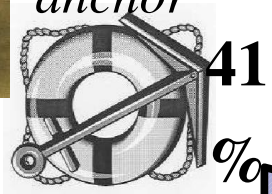

ewer 65%


bonsai 34%


beaver 13%

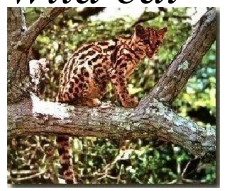

skate 100%


face 84%


anchor 41%


lotus 22%


cougar body 12%


wild cat 1%

BEST

WORST


okapi 100%


minaret 91%


cellphone 83%


joshua t. 47%


sea horse 37%


ant 16%

metronome 100%

background 3%

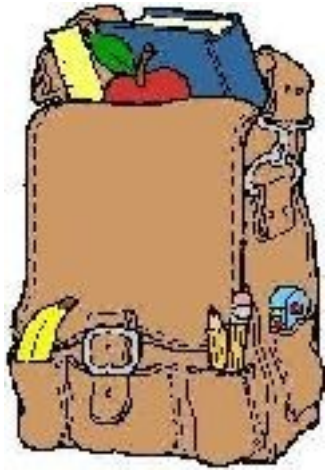
Caltech 256


















Conclusion

- **Energy-Based Models is a general framework for probabilistic and non-probabilistic learning**
  - Make the energy of training samples low, make the energy of everything else high (e.g. Discriminant HMM, Graph Transformer Networks, Conditional Random Fields, Max Margin Markov Nets,...)
- **Invariant vision tasks require deep learning**
  - shallow models such as SVM can't learn complicated invariances.
- **Deep Supervised Learning works well with lots of samples**
  - Convolutional nets have record accuracy on handwriting recognition and face detection, and can be applied to many tasks.
- **Unsupervised Learning can reduce the need for labeled samples**
  - Stacks of sequentially-trained RBMs or sparse encoder-decoder layers learn good feature without requiring labeled samples
- **Learning invariant feature hierarchies**
  - yields excellent accuracy for shape recognition

Thank You