# Deconvolutional Networks

Matthew D. Zeiler
Dilip Krishnan, Graham W. Taylor
Rob Fergus

Dept. of Computer Science, Courant Institute,
New York University

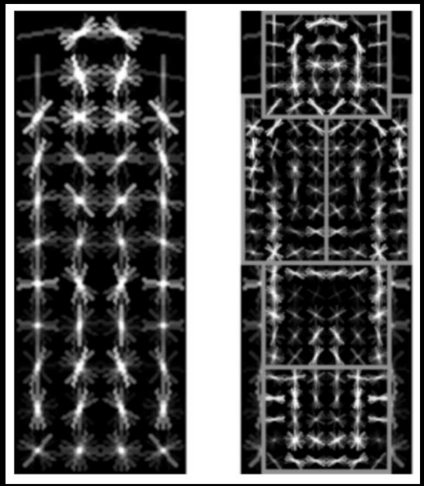
Matt Zeiler

# Overview

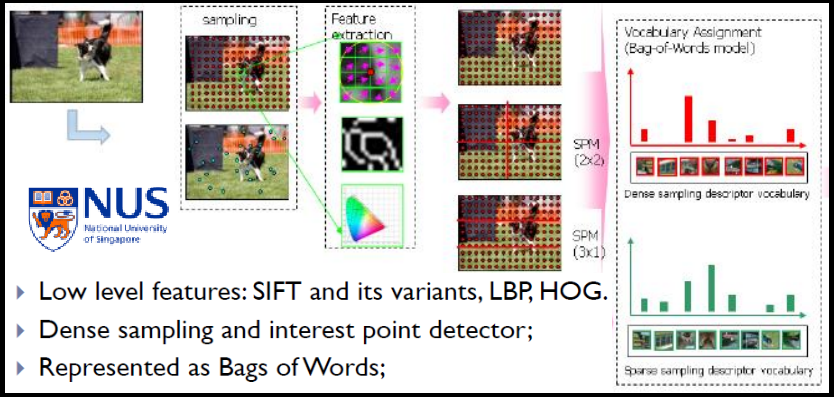
- Unsupervised learning of
  mid and high-level image representations
- Feature hierarchy built from alternating layers of:
  - Convolutional sparse coding (Deconvolution)
  - Max pooling
- Application to object recognition

# Motivation

- Good representations are key to many tasks in vision
- Edge-based representations are basis of many models
  - SIFT [Lowe'04], HOG [Dalal & Triggs '05] & others



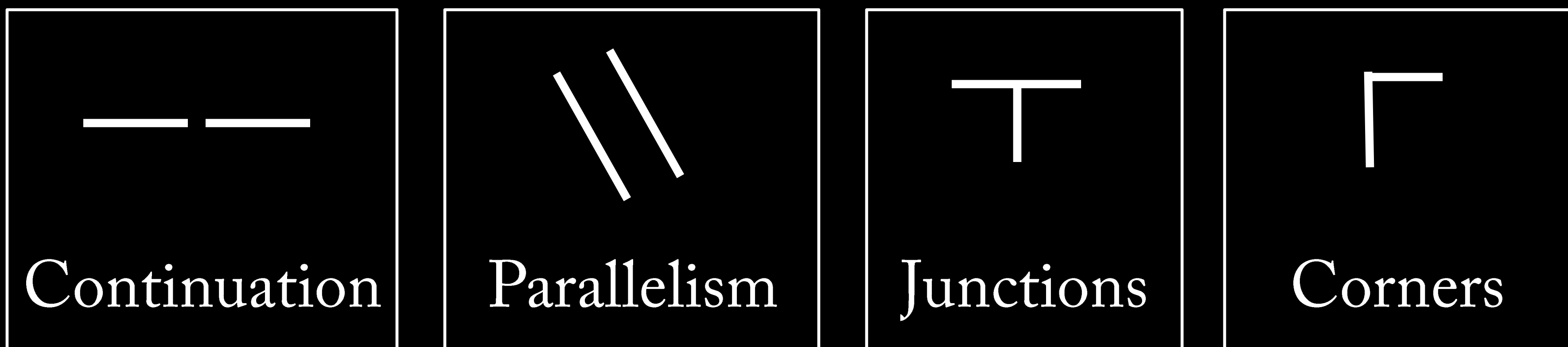
Felzenszwalb, Girshick, McAllester and Ramanan, PAMI 2007




Yan & Huang
(Winner of PASCAL 2010 classification competition)

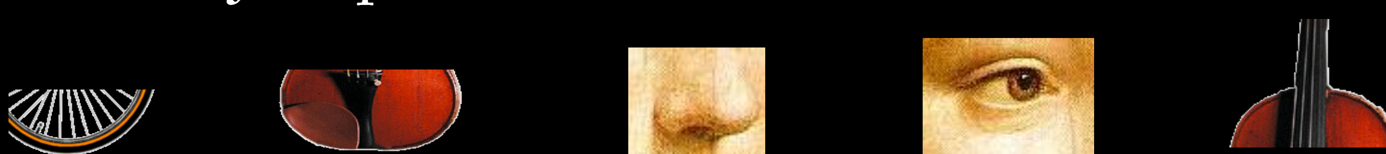
# Beyond Edges?

- Mid-level cues




“Tokens” from Vision by D.Marr

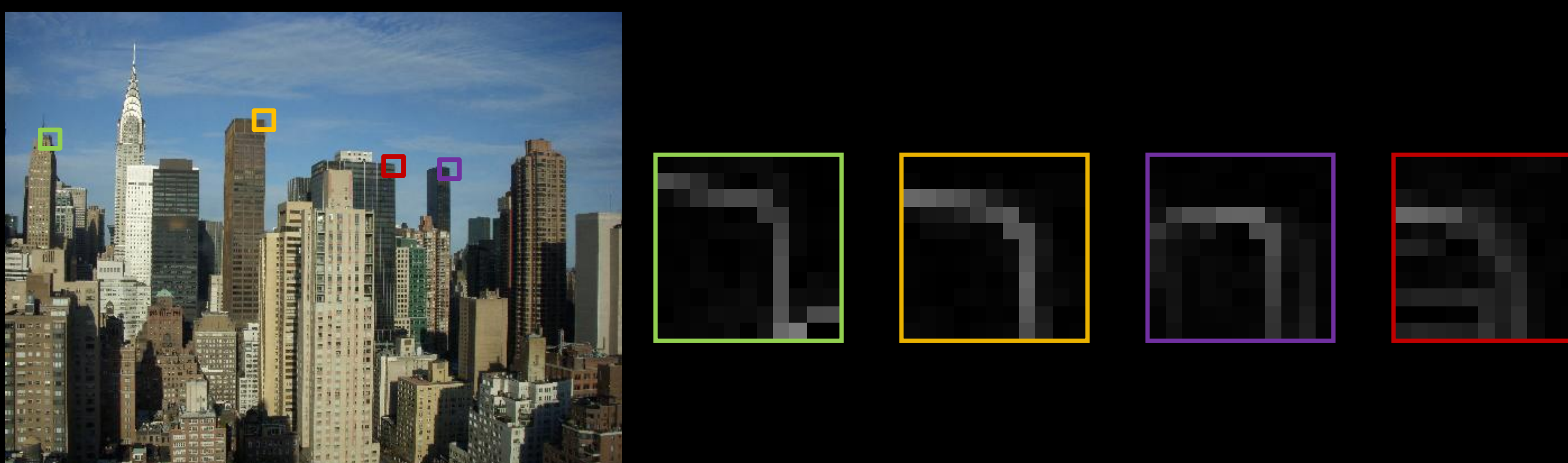
- High-level object parts:



# Two Challenges

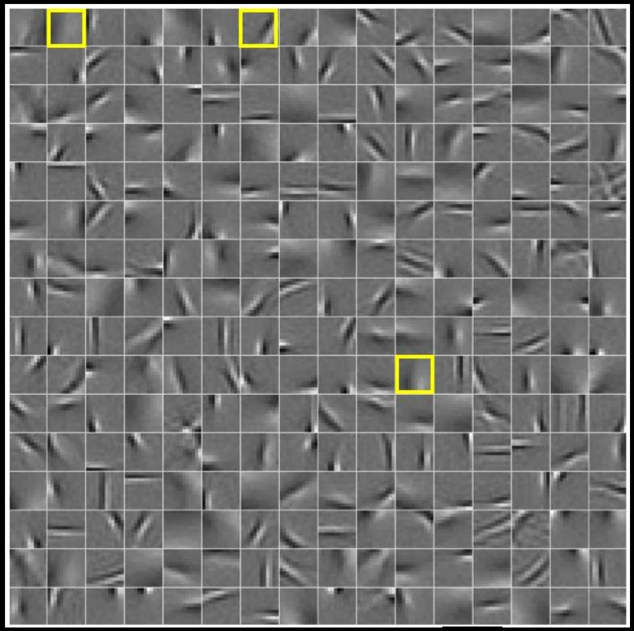
1. Grouping mechanism
   - Want edge structures to group into more complex forms
   - But hard to define explicit rules

2. Invariance to local distortions
   - Corners, T-junctions, parallel lines etc. can look quite different



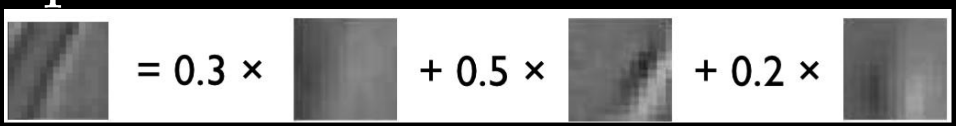
# Recap: Sparse Coding (Patch-based)

- Over-complete linear decomposition of input $y$ using dictionary $D$



Dictionary $D$

Input




$y$

$$C(y, D) = \underset{z}{\operatorname{argmin}} \ \frac{\lambda}{2}\|Dz - y\|_2^2 + |z|_1$$

- $\ell_1$ regularization yields solutions with few non-zero elements
- Output is sparse vector: $z = [0, 0.3, 0, \ldots, 0.5, \ldots, 0.2, \ldots, 0]$

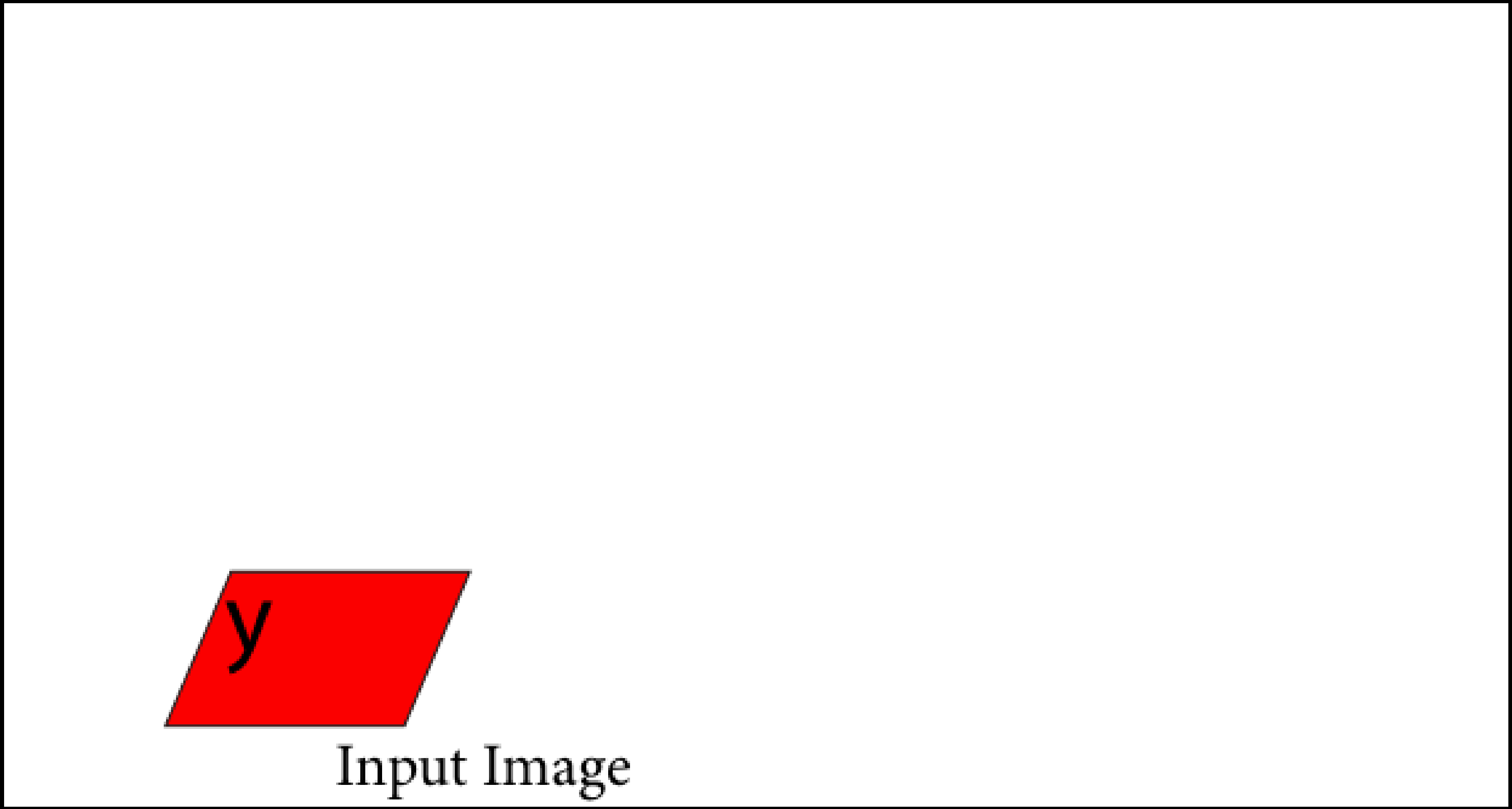
# Talk Overview

- Single layer
  - Convolutional Sparse Coding
  - Max Pooling
- Multiple layers
  - Multi-layer inference
  - Filter learning
- Comparison to related methods
- Experiments

# Talk Overview

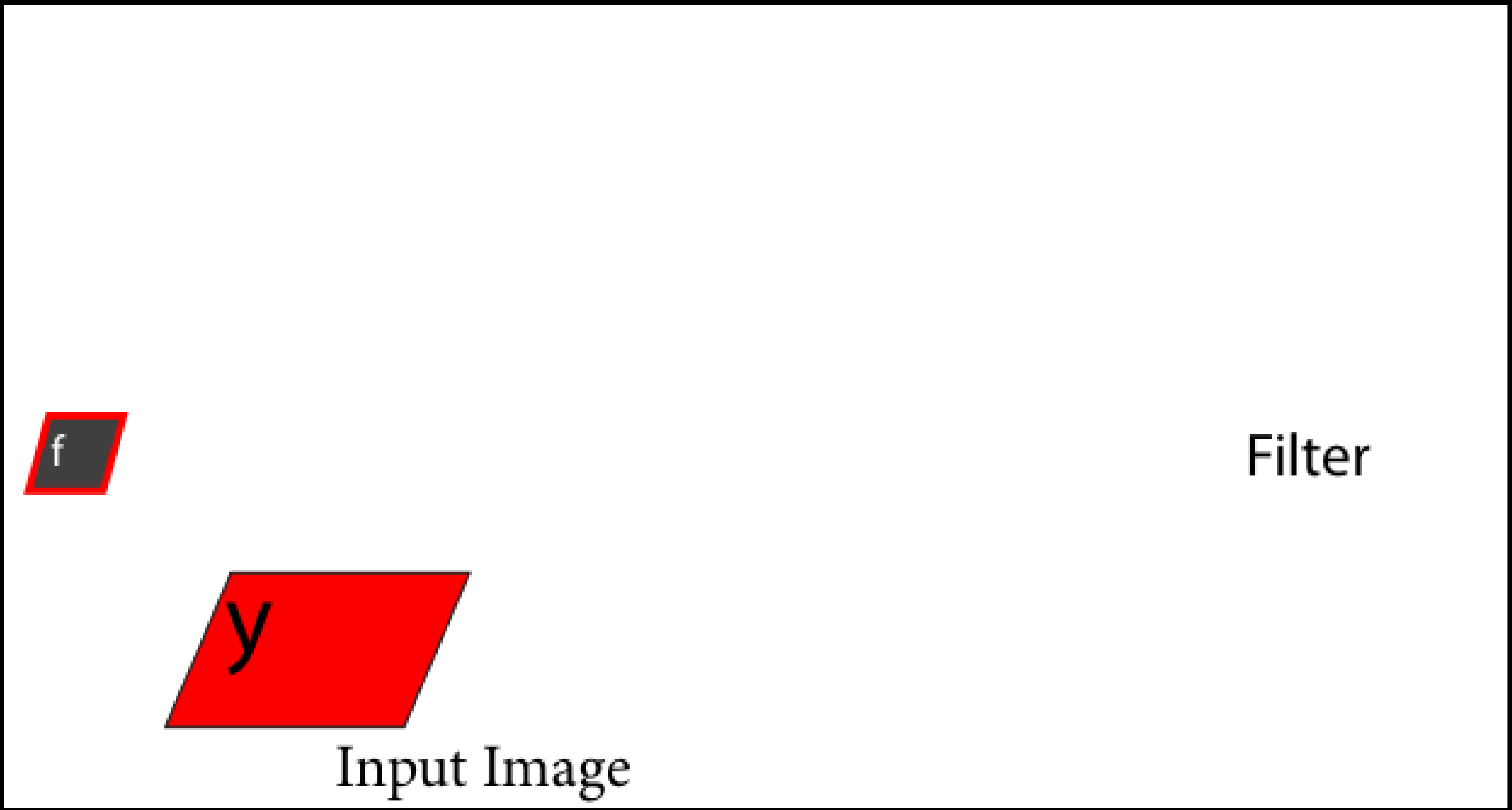
- Single layer
  - Convolutional Sparse Coding
  - Max Pooling
- Multiple layers
  - Multi-layer inference
  - Filter learning
- Comparison to related methods
- Experiments

# Single Deconvolutional Layer

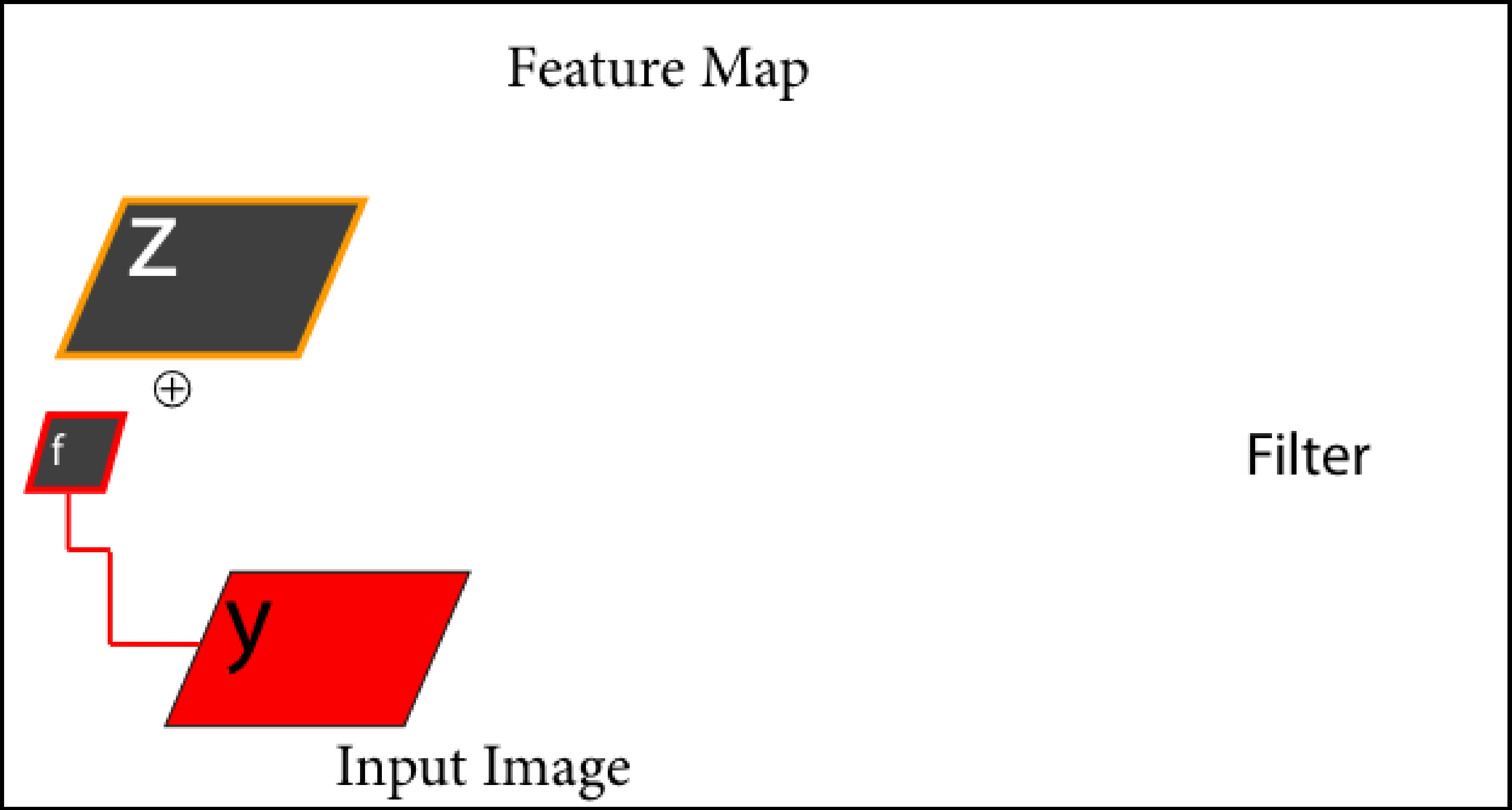



- Convolutional form of sparse coding

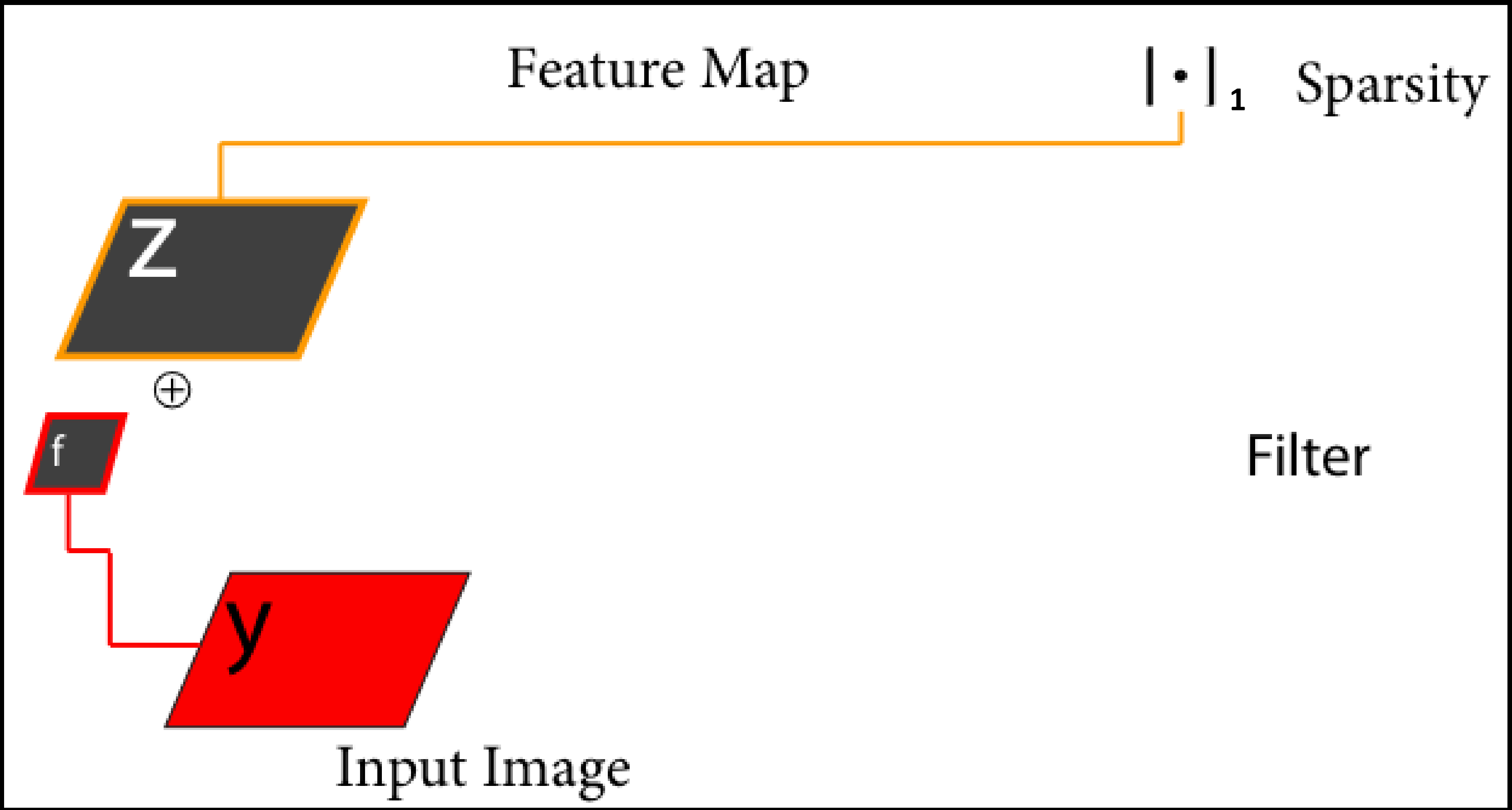
# Single Deconvolutional Layer



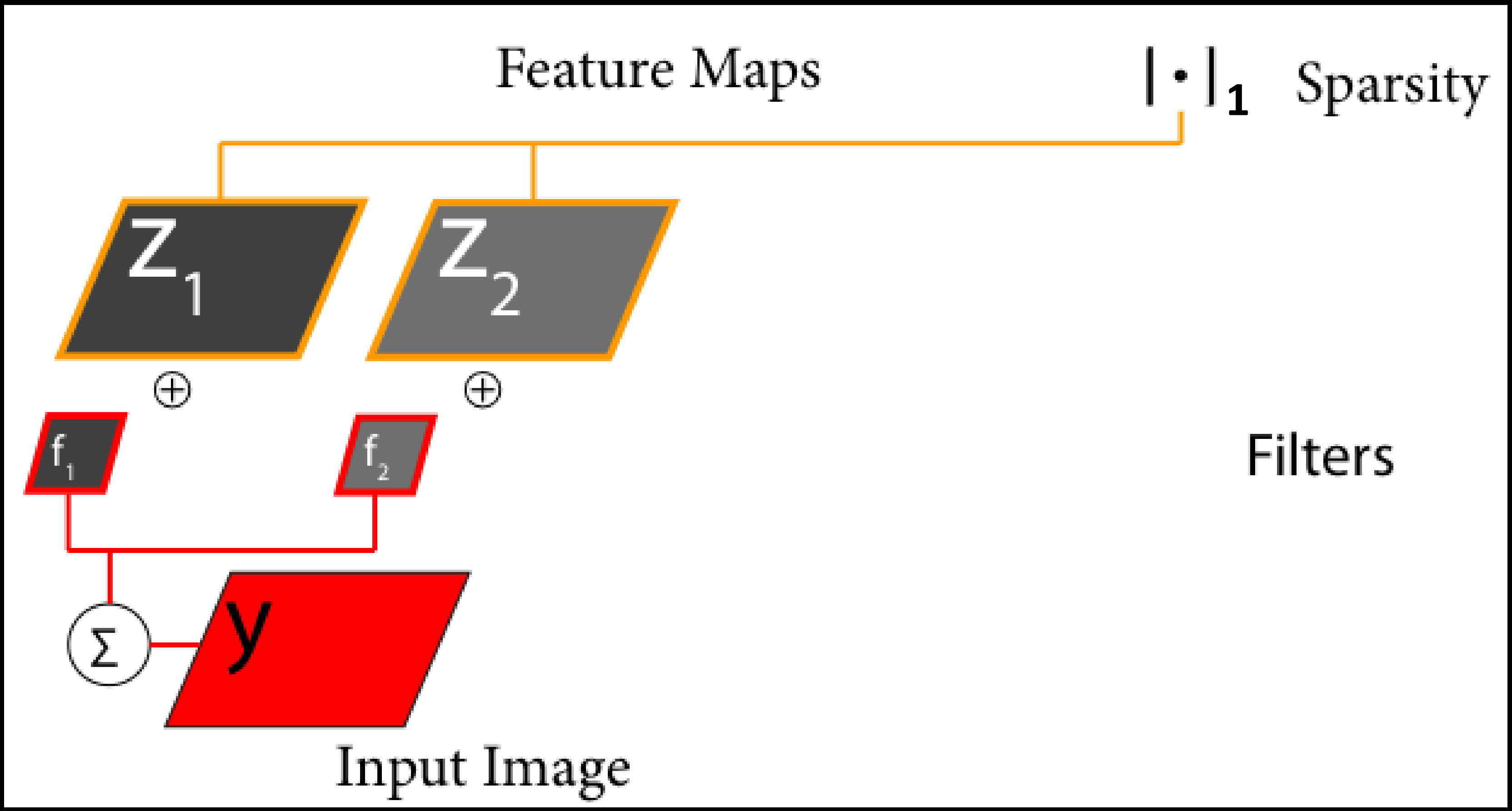
# Single Deconvolutional Layer



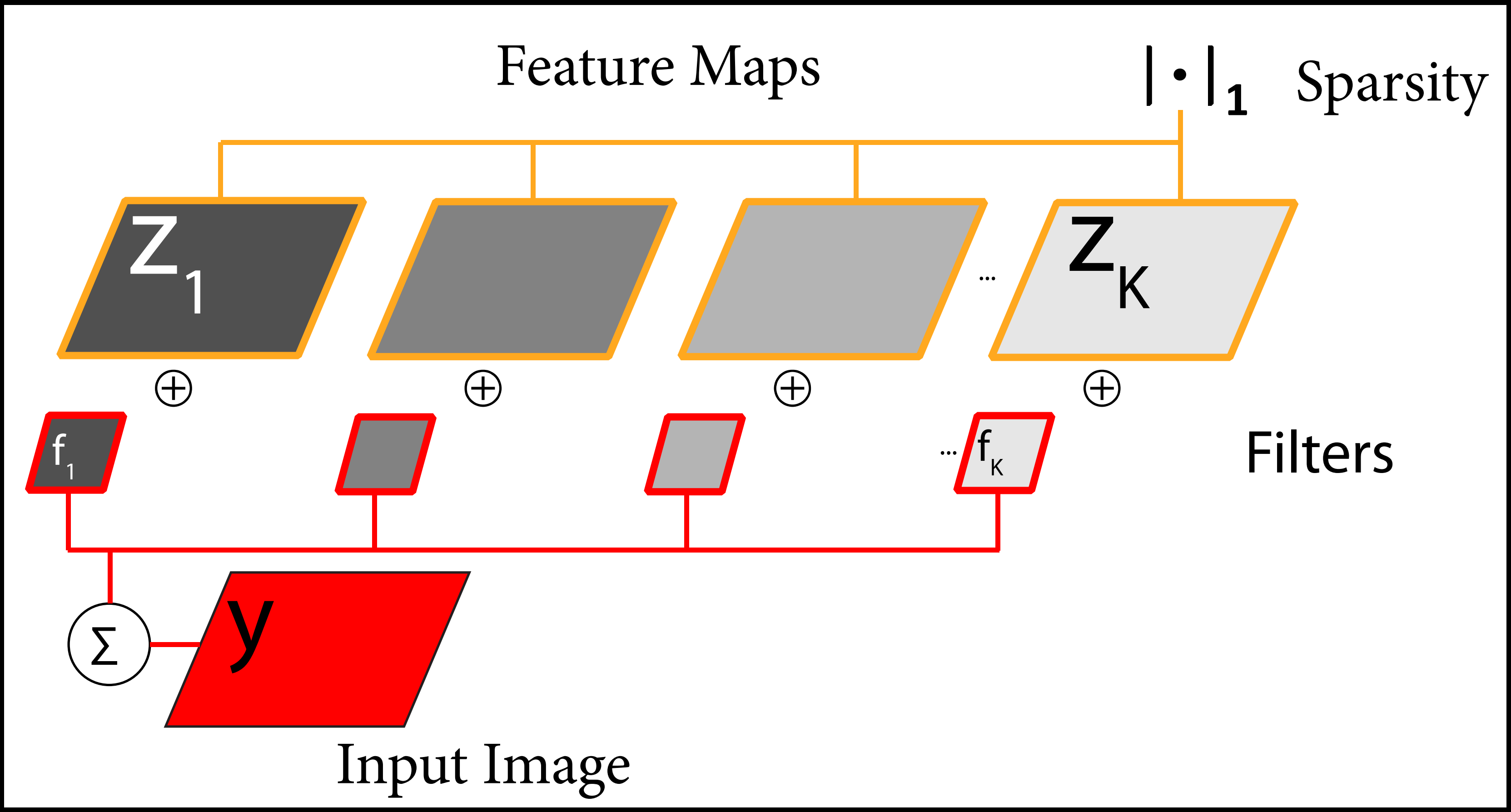
# Single Deconvolutional Layer



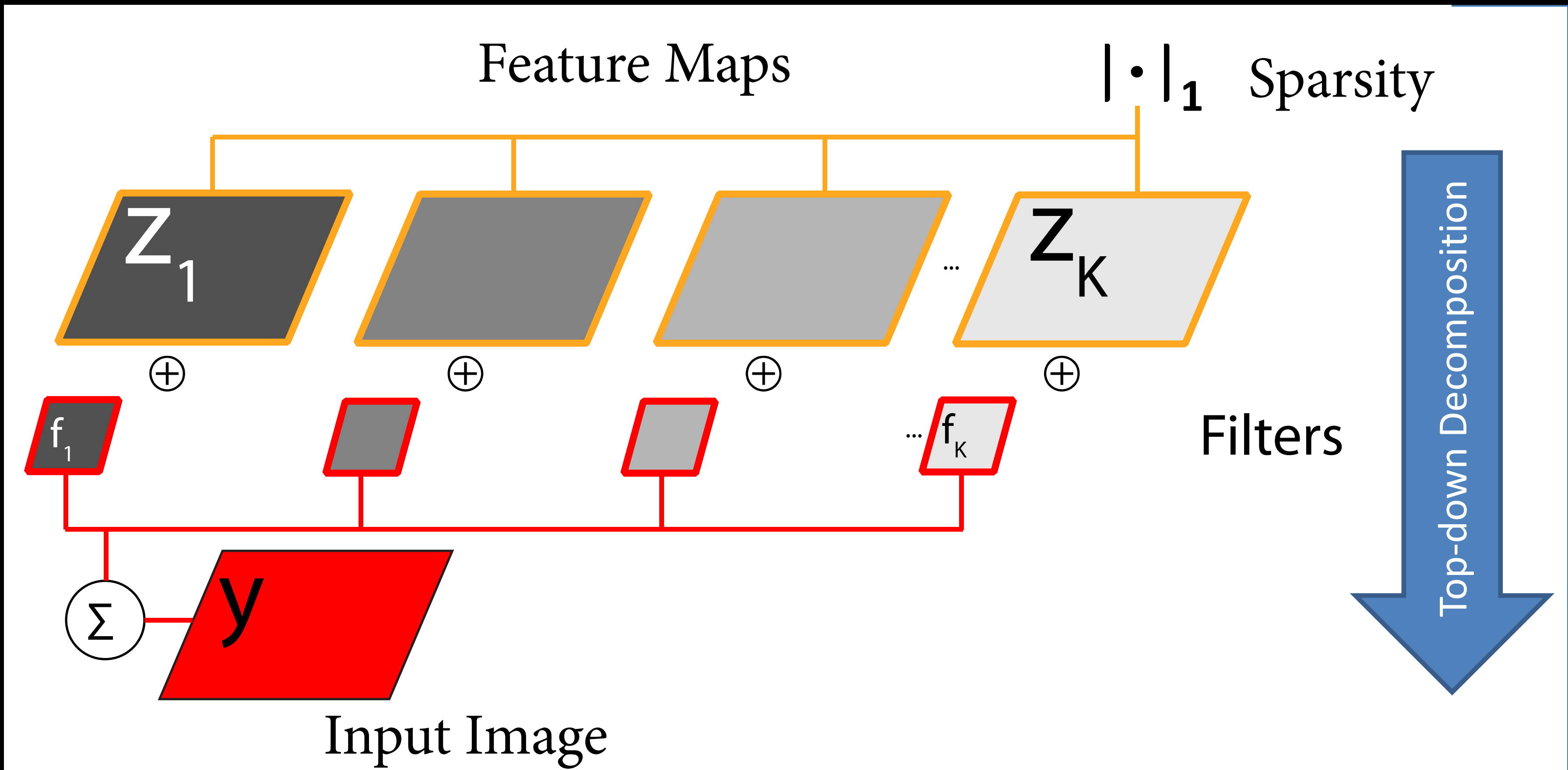
# Single Deconvolutional Layer



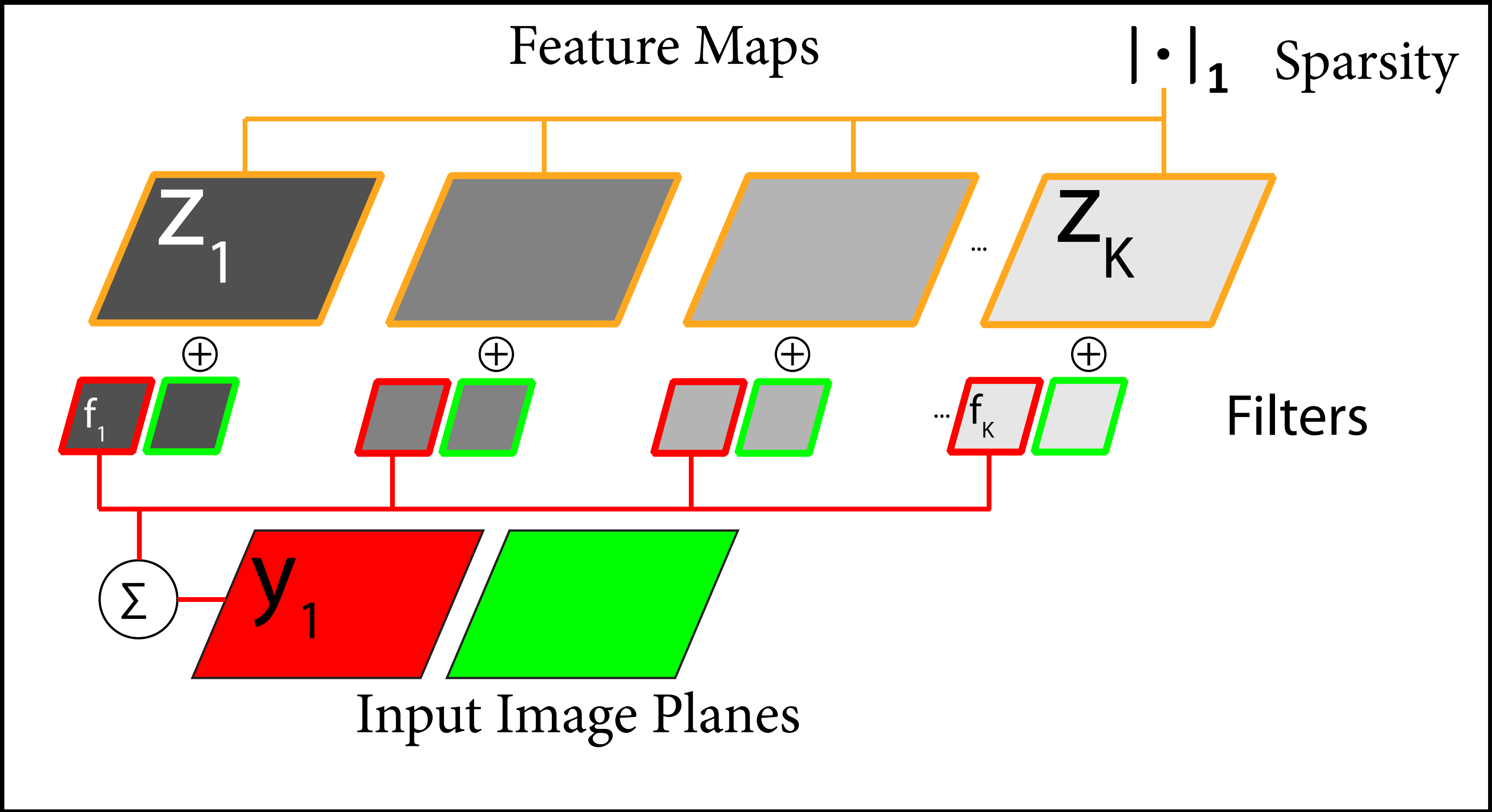
# Single Deconvolutional Layer



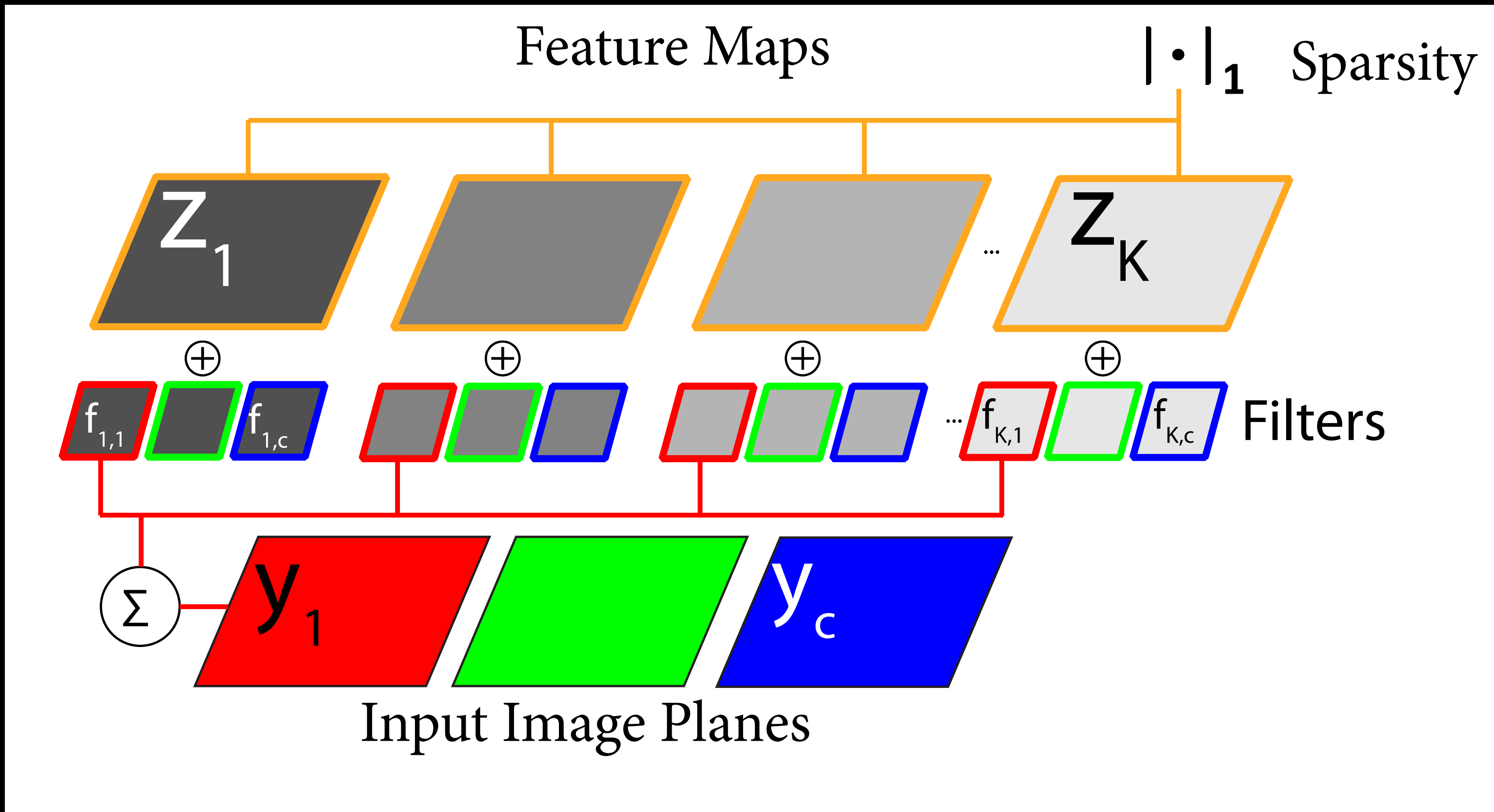
# Single Deconvolutional Layer



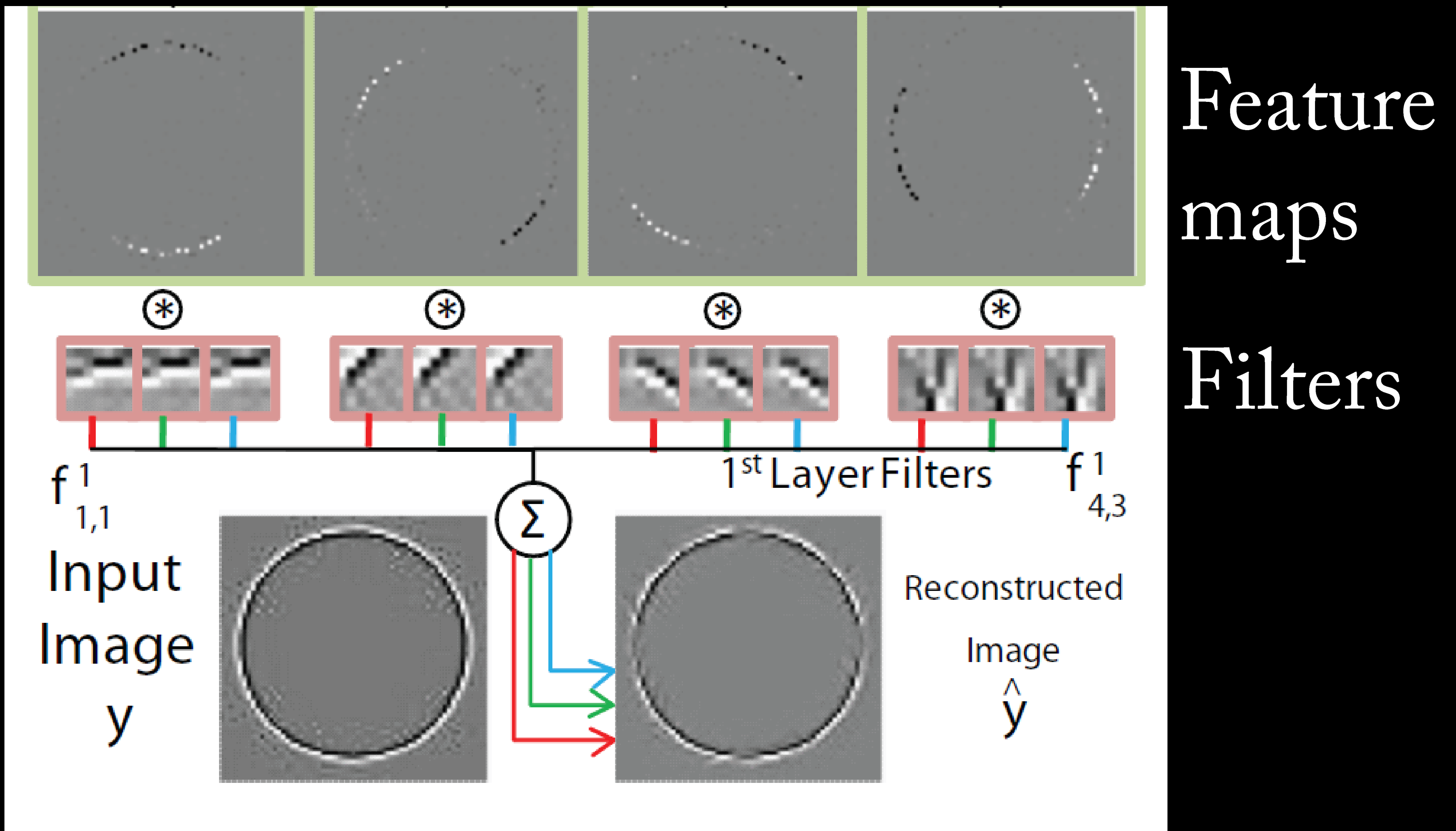
# Single Deconvolutional Layer



# Single Deconvolutional Layer



# Toy Example


Feature maps
Filters
1st Layer Filters
f 1 1,1
f 1 4,3
Σ
Input Image y
Reconstructed Image
ŷ

# Objective for Single Layer

$$\text{min z:} \quad C = \frac{\lambda}{2}\| \sum_{k=1}^{K} z_k \oplus f_k - y \|_2^2 + \sum_{k=1}^{K} |z_k|_1$$

$y$ = Input, $z$ = Feature maps, $f$ = Filters

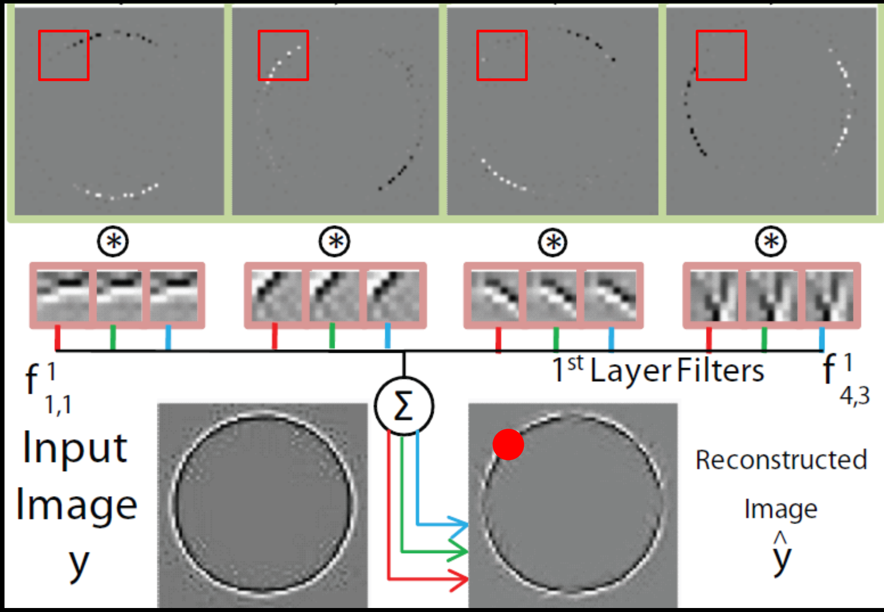
# Inference for Single Layer

Objective: $$C = \frac{\lambda}{2}\|Fz - y\|_2^2 + |z|_1$$

Known: $y$ = Input, $F$ = Filter weights. $\boxed{\text{Solve for : } z = \text{Feature maps}}$

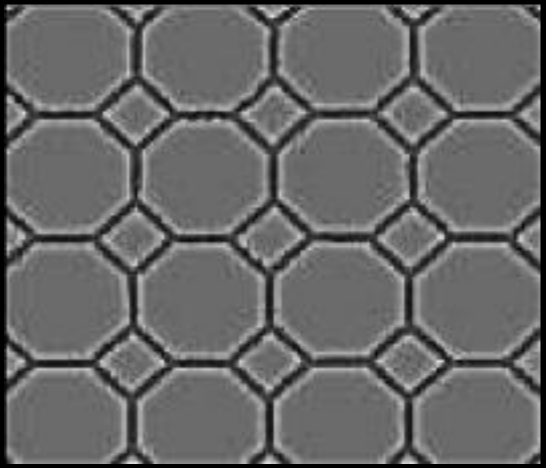
# Effect of Sparsity

- Introduces local competition in feature maps
  - Explaining away

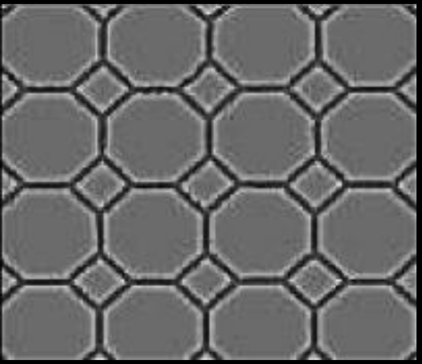


# Local Inhibition/Explaining Away

- How many different line segments (filters) are needed to represent this image?



# Local Inhibition/Explaining Away



Image

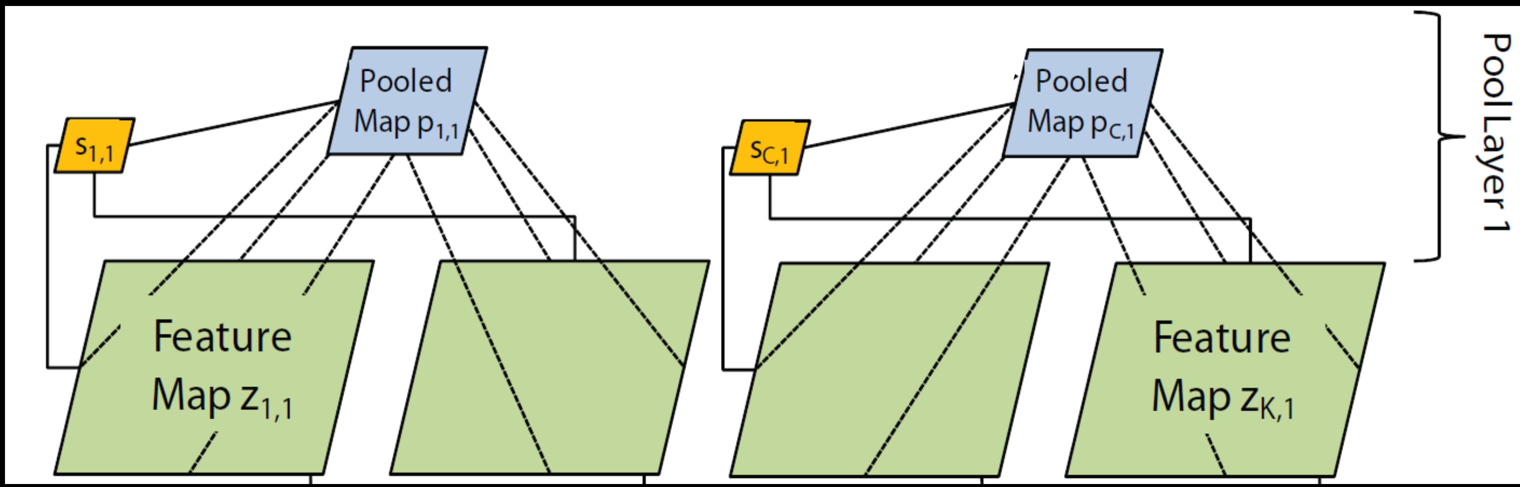
Filters

# Talk Overview

- Single layer
  - Convolutional Sparse Coding
  - Max Pooling
- Multiple layers
  - Multi-layer inference
  - Filter learning
- Comparison to related methods
- Experiments

# 3D Max Pooling

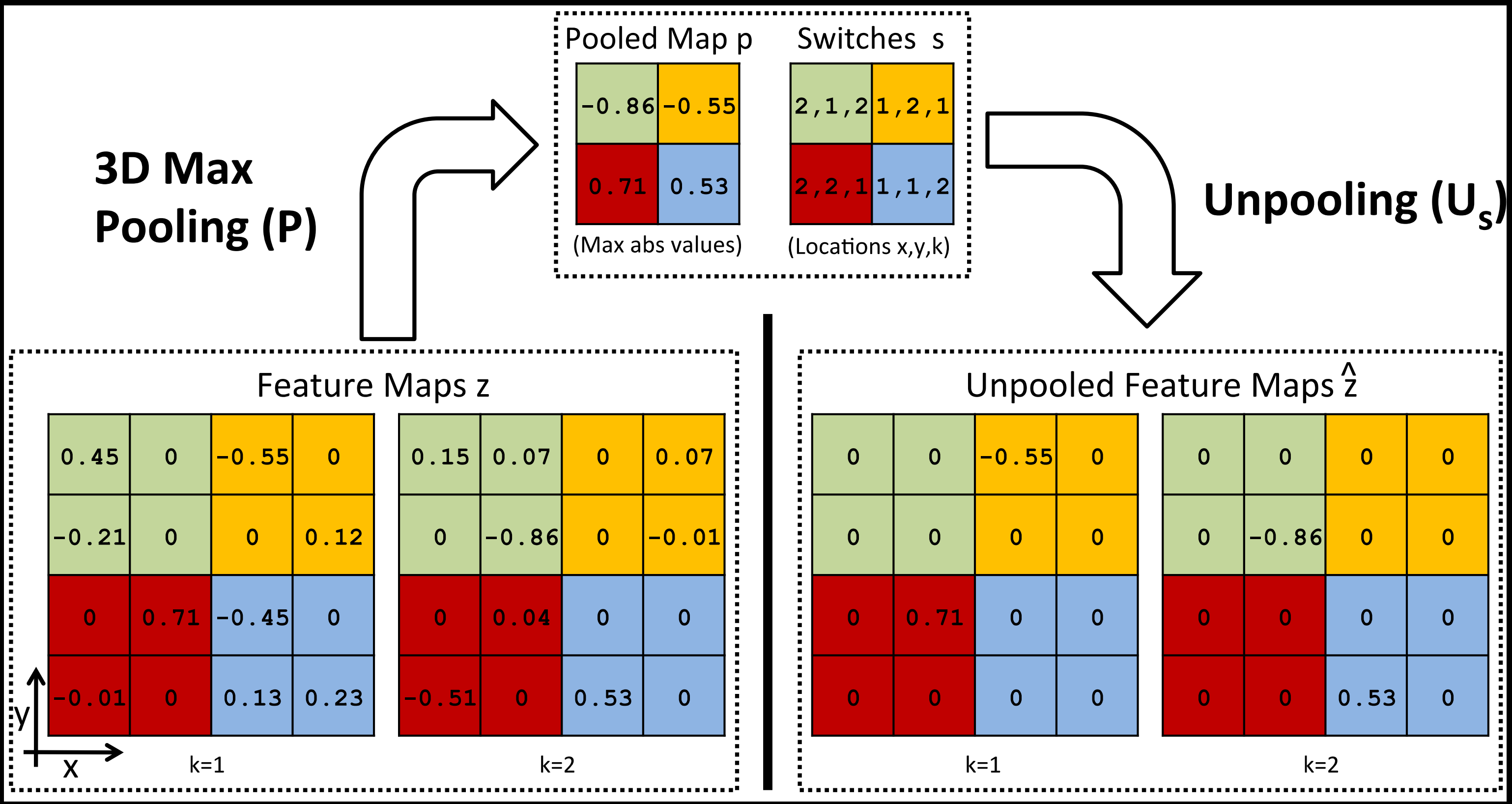
- Pool within & between feature maps




- Take absolute max value (& preserve sign)
- Record locations of max in switches

# 3D Max Pooling

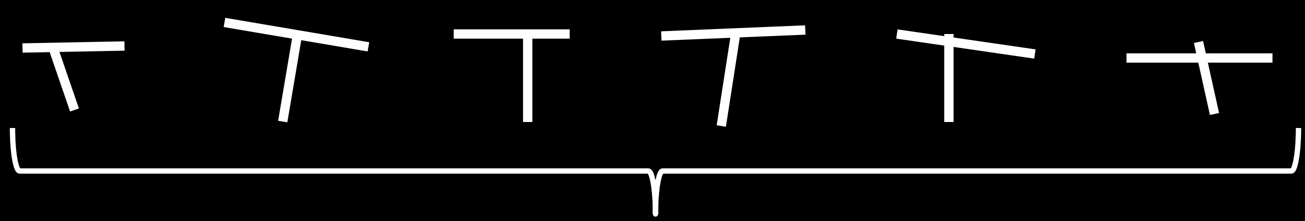
- Pool within & between feature maps:




- Pooling/unpooling is linear, given max locations:
  - Pooling: $[p, s] = P(z)$ Unpooling: $\hat{z} = U_s p$

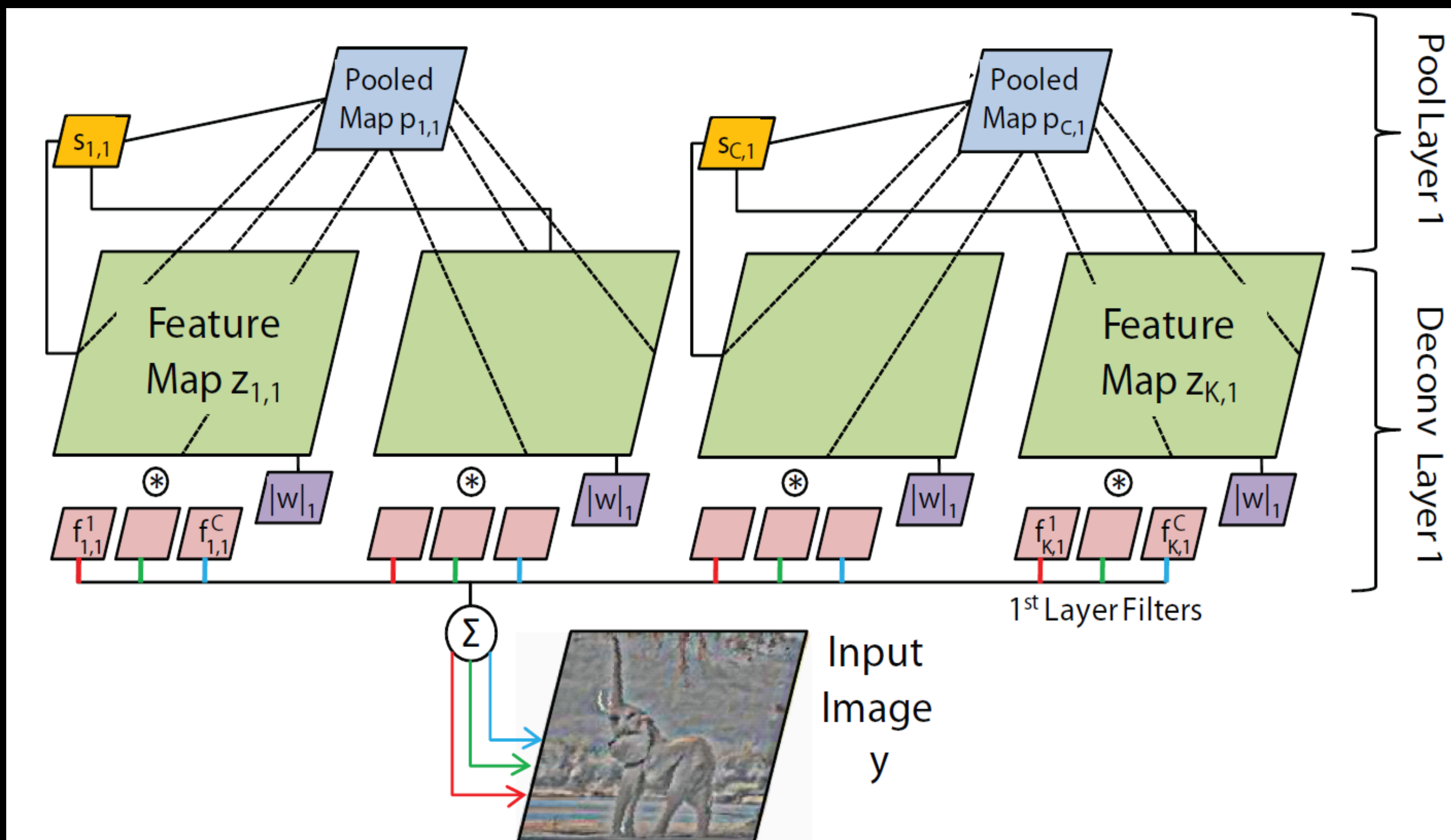
# Role of Switches

- Permit reconstruction path back to input
  - Record position of local max
  - Important for multi-layer inference
- Set during inference of each layer
  - Held fixed for subsequent layers' inference
- Provide invariance:

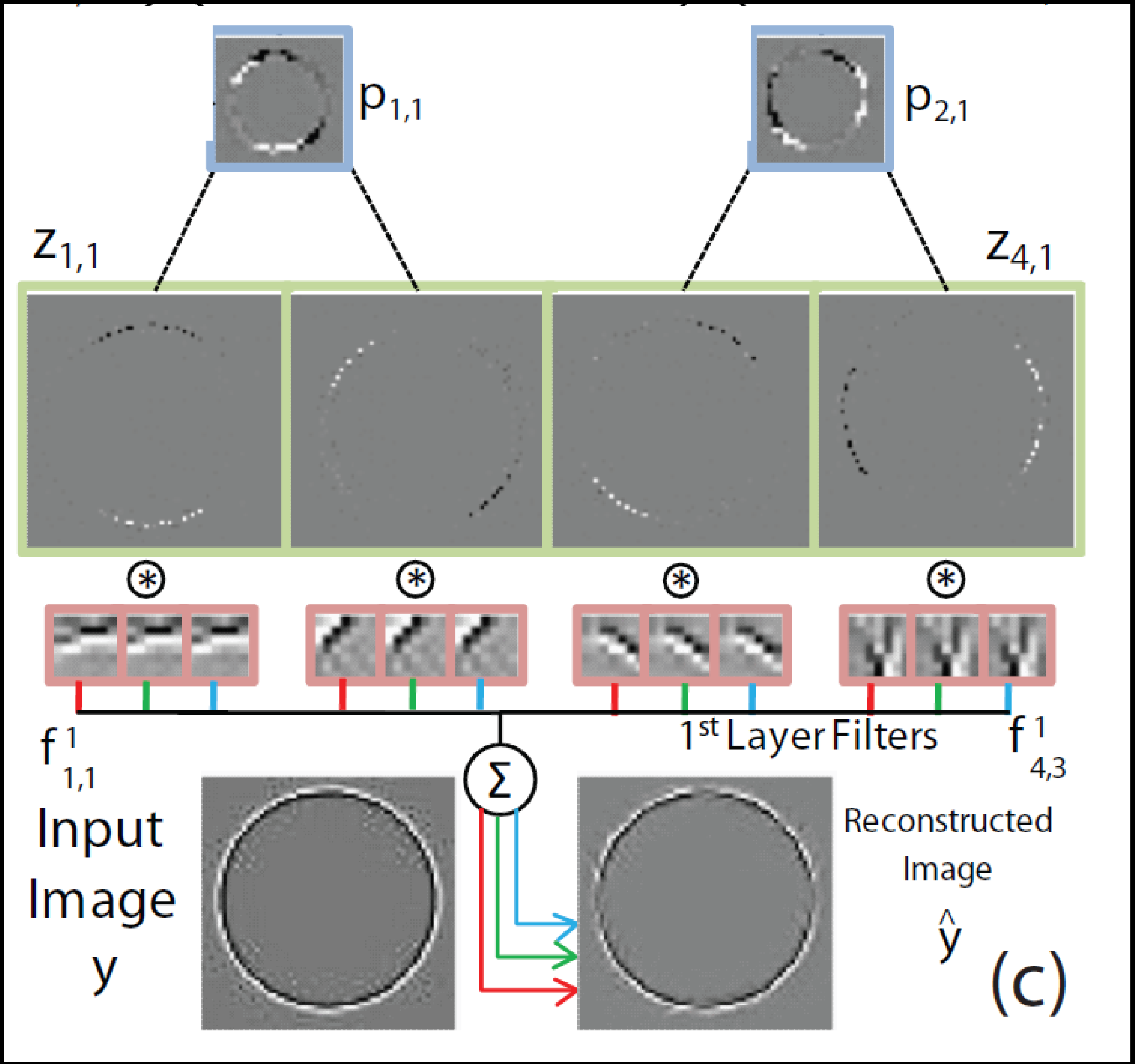


Single feature map

# Overall Architecture (1 layer)



# Toy Example




Pooled maps

Feature maps

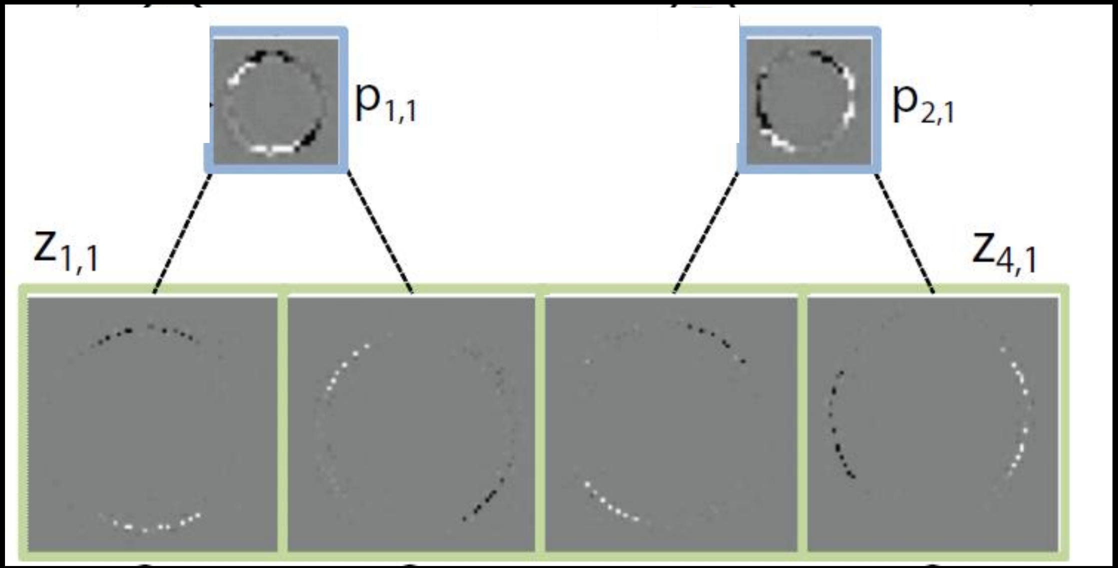
Filters

# Effect of Pooling

- Reduces size of feature maps
  - So we can have more of them in layers above

- Pooled maps are dense
  - Ready to be decomposed by sparse coding of layer above

- Additional competition
  - For 3D pooling



# Talk Overview

- Single layer
  - Convolutional Sparse Coding
  - Max Pooling
- Multiple layers
  - Multi-layer inference
  - Filter learning
- Comparison to related methods
- Experiments

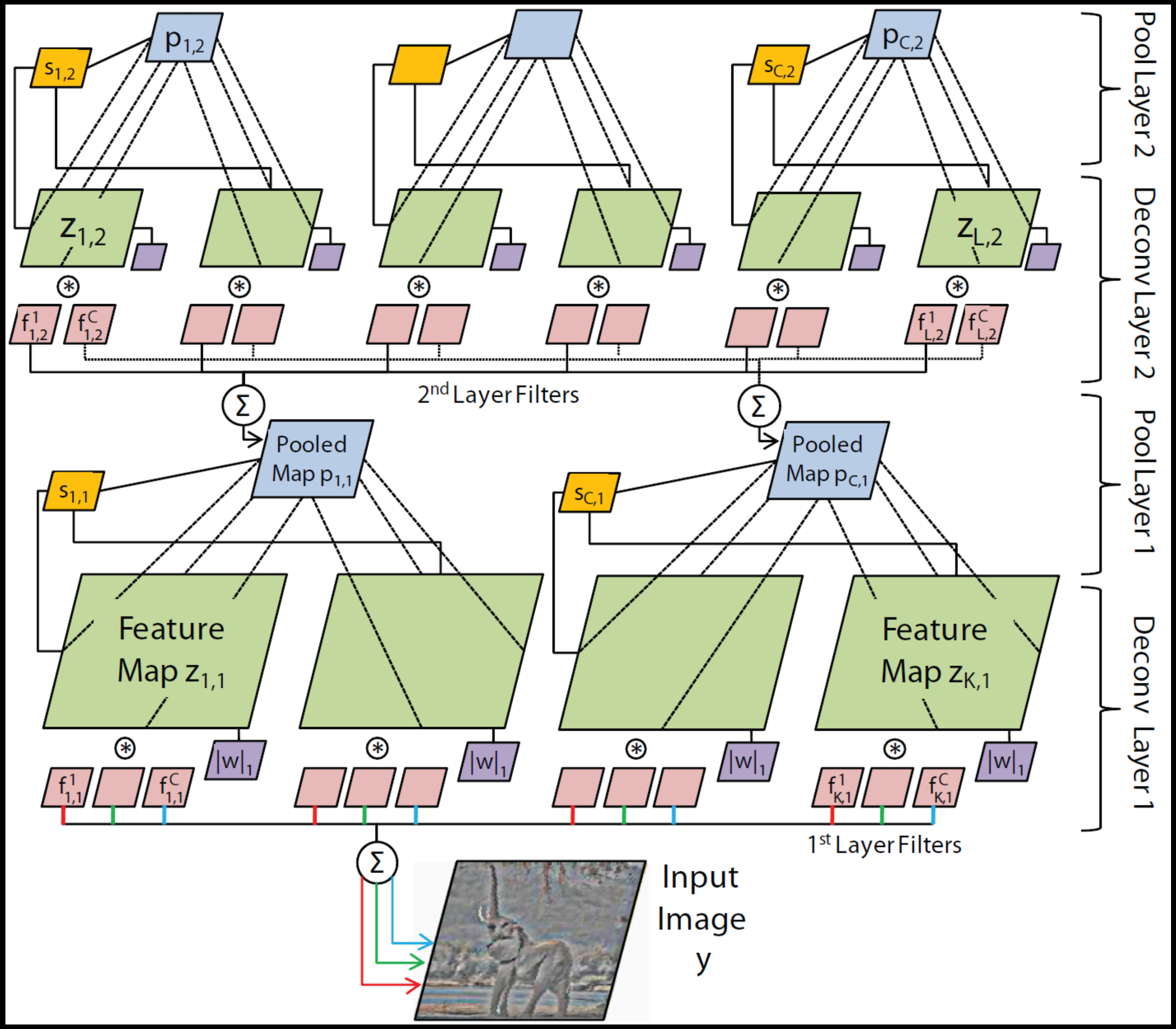
# Stacking the Layers

- Take pooled maps as input to next deconvolution/pooling layer
- Learning & inference is layer-by-layer

- Objective is reconstruction error
  - Key point: with respect to input image
  - Constraint of using filters in layers below

- Sparsity & pooling make model non-linear
  - No sigmoid-type non-linearities

# Overall Architecture (2 layers)



# Multi-layer Inference

- Consider layer 2 inference:
  - Want to minimize reconstruction error of input image $||\hat{y} - y||_2^2$, subject to sparsity.
  - Don't care about reconstructing layers below
- ISTA:
  - Update $z_l$ :

# Filter Learning

Objective: $$C = \frac{\lambda}{2}\|Fz - y\|_2^2 + |z|_1$$

Known: $y$ = Input, $z$ = Feature maps. $\boxed{\text{Solve for : } F = \text{Filter weights}}$

# Overall Algorithm

- For Layer 1 to L: % Train each layer in turn
  - For Epoch 1 to E: % Loops through dataset
    - For Image 1 to N: % Loop over images
      - For ISTA_step 1 to T: % ISTA iterations
        - Reconstruct $\hat{y}_l$ % Gradient
        - Compute error $e_l = (\hat{y}_l - y)$ % Gradient
        - Propagate error $g_l = R_l^T e_l$ % Gradient
        - Gradient step $z_l = z_l - \lambda_l \beta_l g_l$ % Gradient
        - Skrink $z_l = sh(z_l)$ % Shrinkage
        - Pool/Update Switches $[p_l, s_l]$ % Update Switches
      - Update filters % Learning, via linear CG system

2nd layer pooled maps

2nd layer feature maps

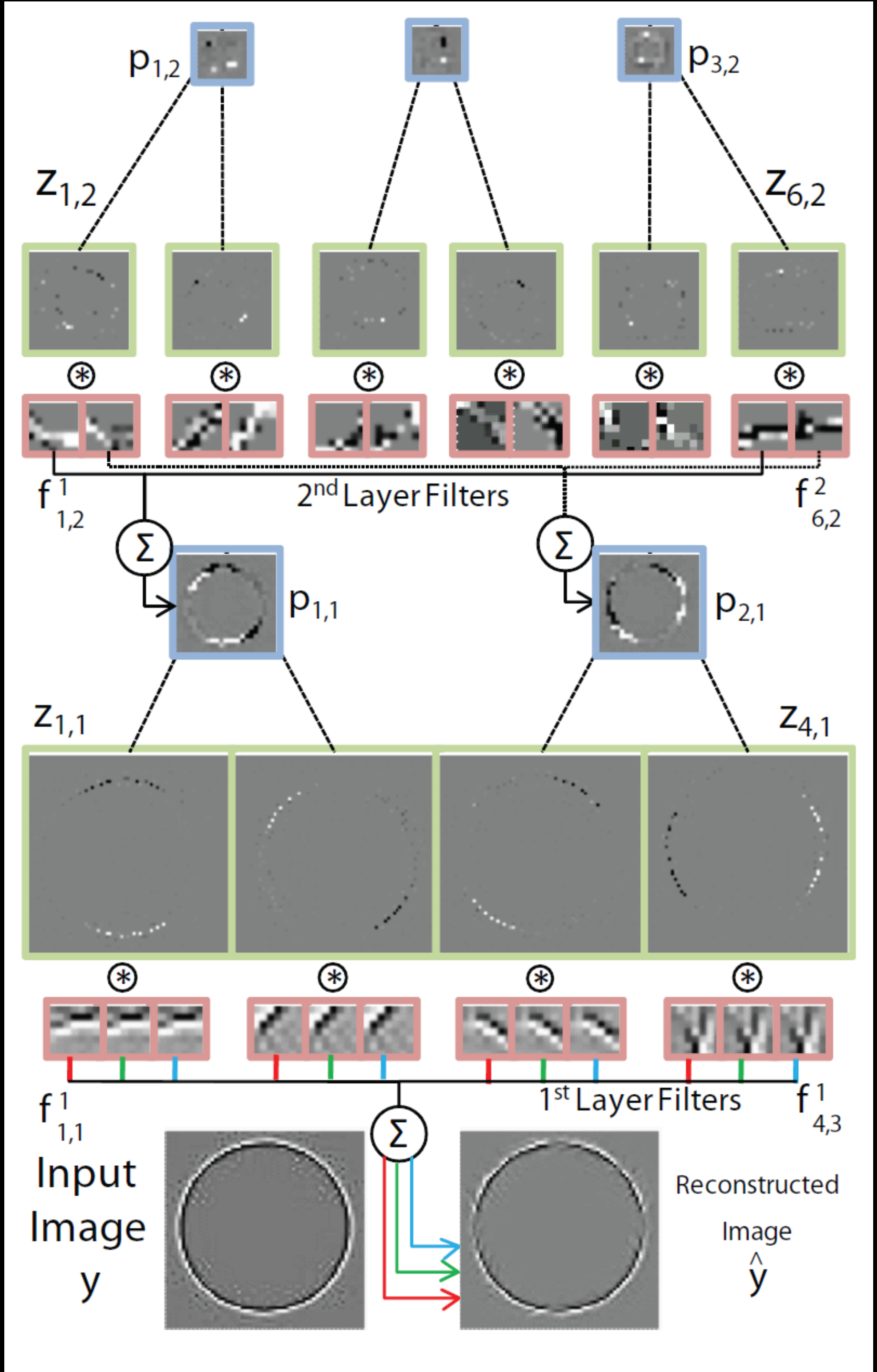
2nd layer filters

1st layer pooled maps

1st layer feature maps

1st layer filters

# Toy Input



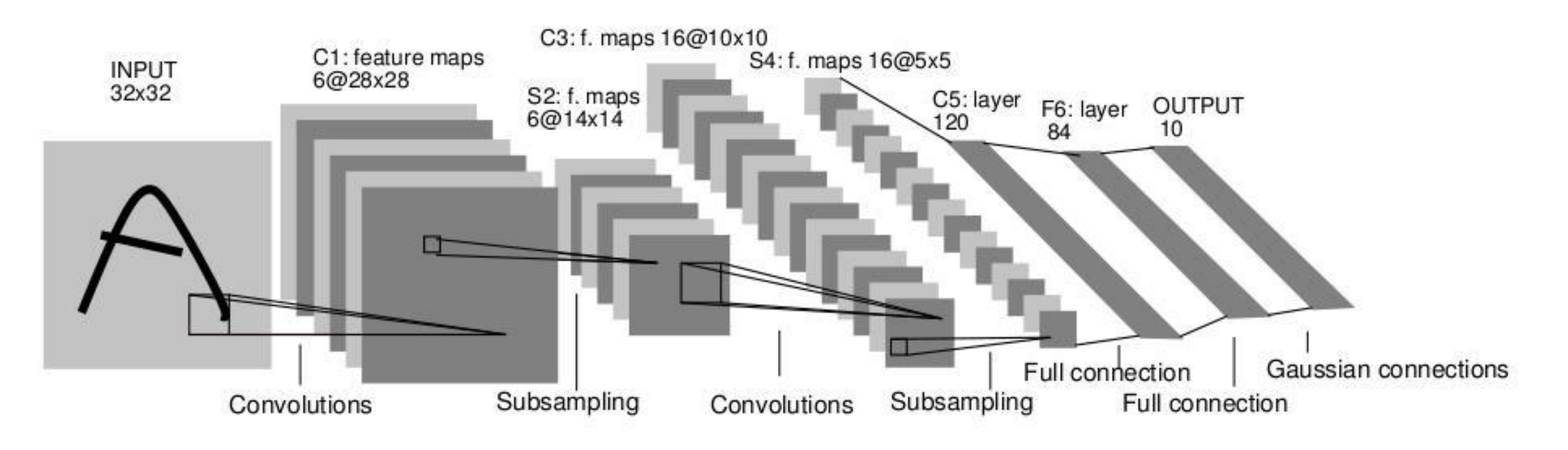
# Talk Overview

- Single layer
  - Convolutional Sparse Coding
  - Max Pooling
- Multiple layers
  - Multi-layer inference
  - Filter learning
- Comparison to related methods
- Experiments

# Related Work

- Convolutional Sparse Coding
  - Zeiler, Krishnan, Taylor & Fergus [CVPR '10]
  - Kavukcuoglu, Sermanet, Boureau, Gregor, Mathieu & LeCun [NIPS '10]
  - Chen, Spario, Dunson & Carin [JMLR submitted]
    - Only 2 layer models
- Deep Learning
  - Hinton & Salakhutdinov [Science '06]
  - Ranzato, Poultney, Chopra & LeCun [NIPS '06]
  - Bengio, Lamblin, Popovici & Larochelle [NIPS '05]
  - Vincent, Larochelle, Bengio & Manzagol [ICML '08]
  - Lee, Grosse, Ranganth & Ng [ICML '09]
  - Jarrett, Kavukcuoglu, Ranzato & LeCun [ICCV '09]
  - Ranzato, Mnih, Hinton [CVPR'11]
    - Reconstruct layer below, not input

# Comparison: Convolutional Nets




LeCun *et al.* 1989

## Convolutional Networks

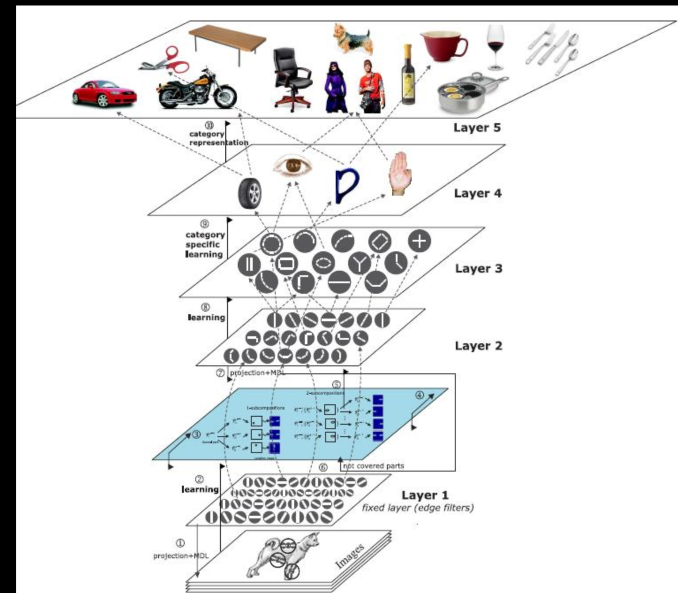
- Bottom-up filtering with convolutions in image space.
- Trained supervised requiring labeled data.

## Deconvolutional Networks

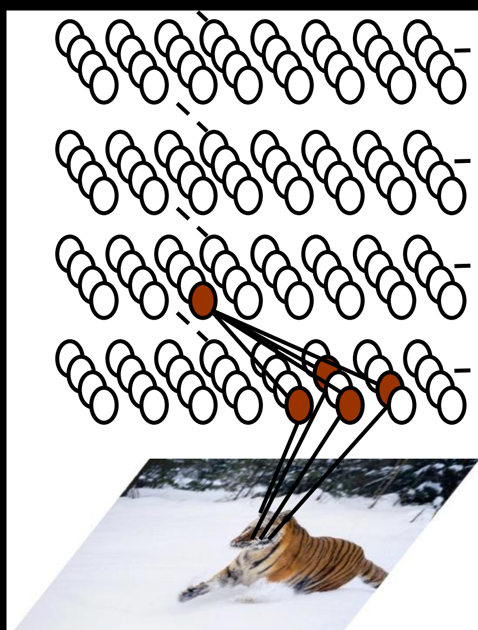
- Top-down decomposition with convolutions in feature space.
- Non-trivial unsupervised optimization procedure involving sparsity.

# Related Work

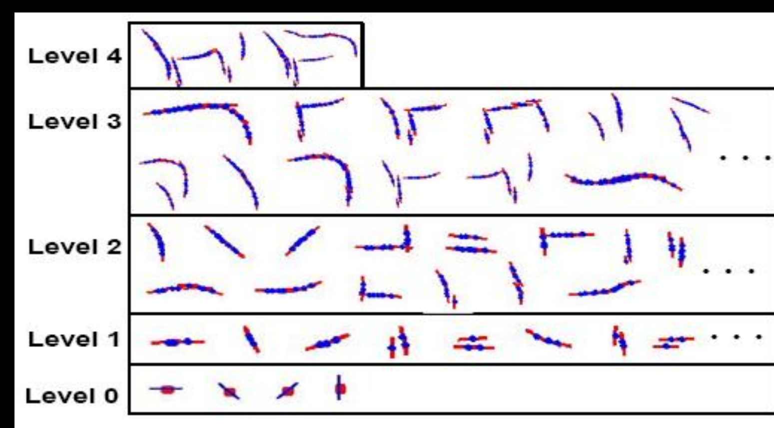
- Hierarchical vision models
  - Zhu & Mumford [F&T '06]
  - Tu & Zhu [IJCV '06]
  - Serre, Wolf & Poggio [CVPR '05]




Fidler & Leonardis [CVPR '07]



Jin & Geman [CVPR '06]




Zhu & Yuille [NIPS '07]

# Talk Overview

- Single layer
  - Convolutional Sparse Coding
  - Max Pooling
- Multiple layers
  - Multi-layer inference
  - Filter learning
- Comparison to related methods
- Experiments

# Training Details

- 3060 training images from Caltech 101
  - 30 images/class, 102 classes (Caltech 101 training set)

- Resized/padded to 150x150 grayscale
- Subtractive & divisive contrast normalization

- Unsupervised

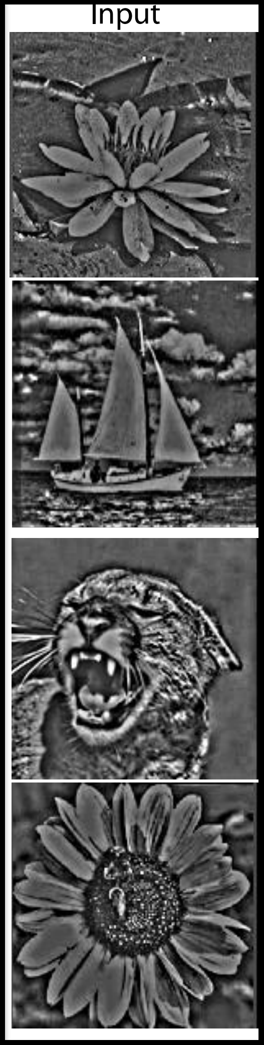
- 6 hrs total training time (Matlab, 6 core CPU)

# Model Parameters/Statistics

| Property | Layer 1 | Layer 2 | Layer 3 | Layer 4 |
|---|---|---|---|---|
| # Feature maps $K_l$ | 15 | 50 | 100 | 150 |
| Pooling size | 3x3x3 | 3x3x2 | 3x3x2 | 3x3x2 |
| $\lambda_l$ | 2 | 0.1 | 0.005 | 0.001 |

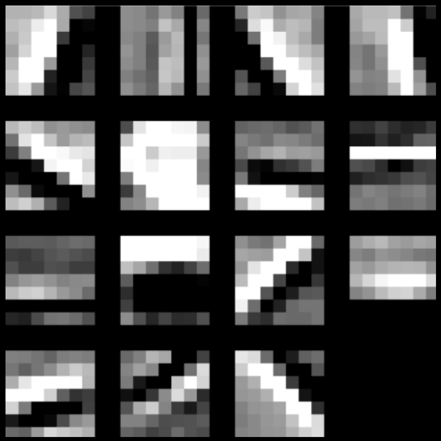
- 7x7 filters at all layers

# Model Reconstructions



# Layer 1 Filters

- 15 filters/feature maps, showing max for each map

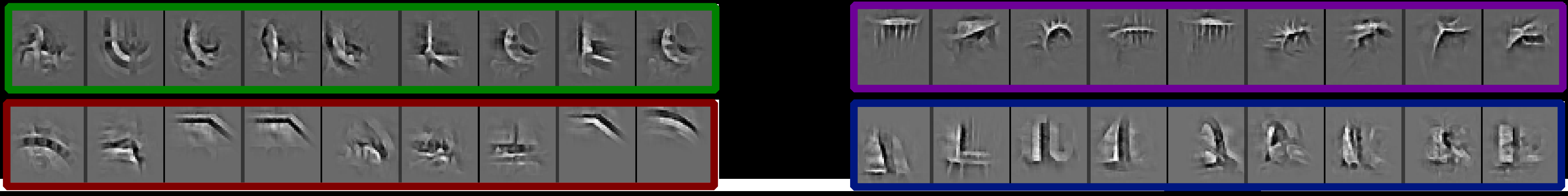


# Layer 2 Filters

- 50 filters/feature maps, showing max for each map projected down to image

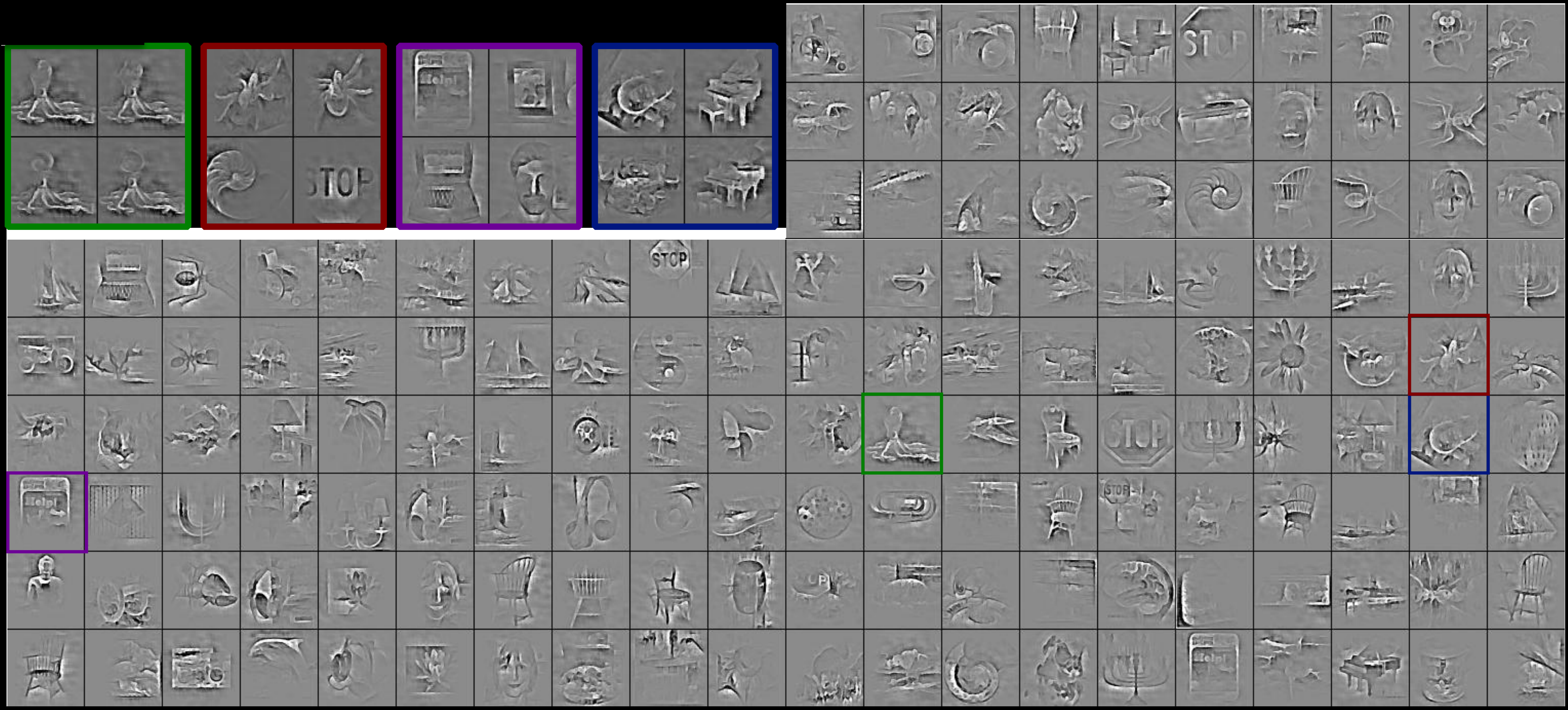
# Layer 3 filters

- 100 filters/feature maps, showing max for each map



# Layer 4 filters

- 150 in total; receptive field is entire image

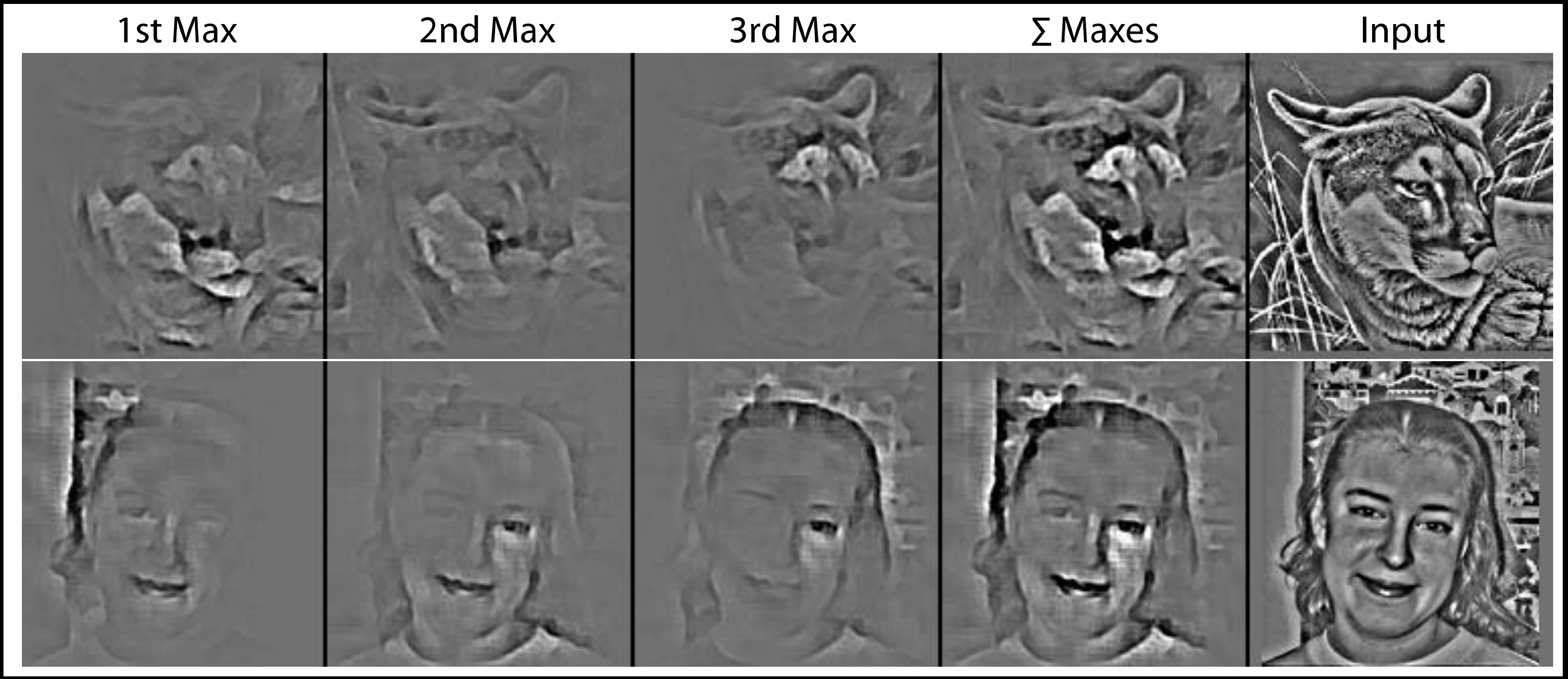


# Relative Size of Receptive Fields

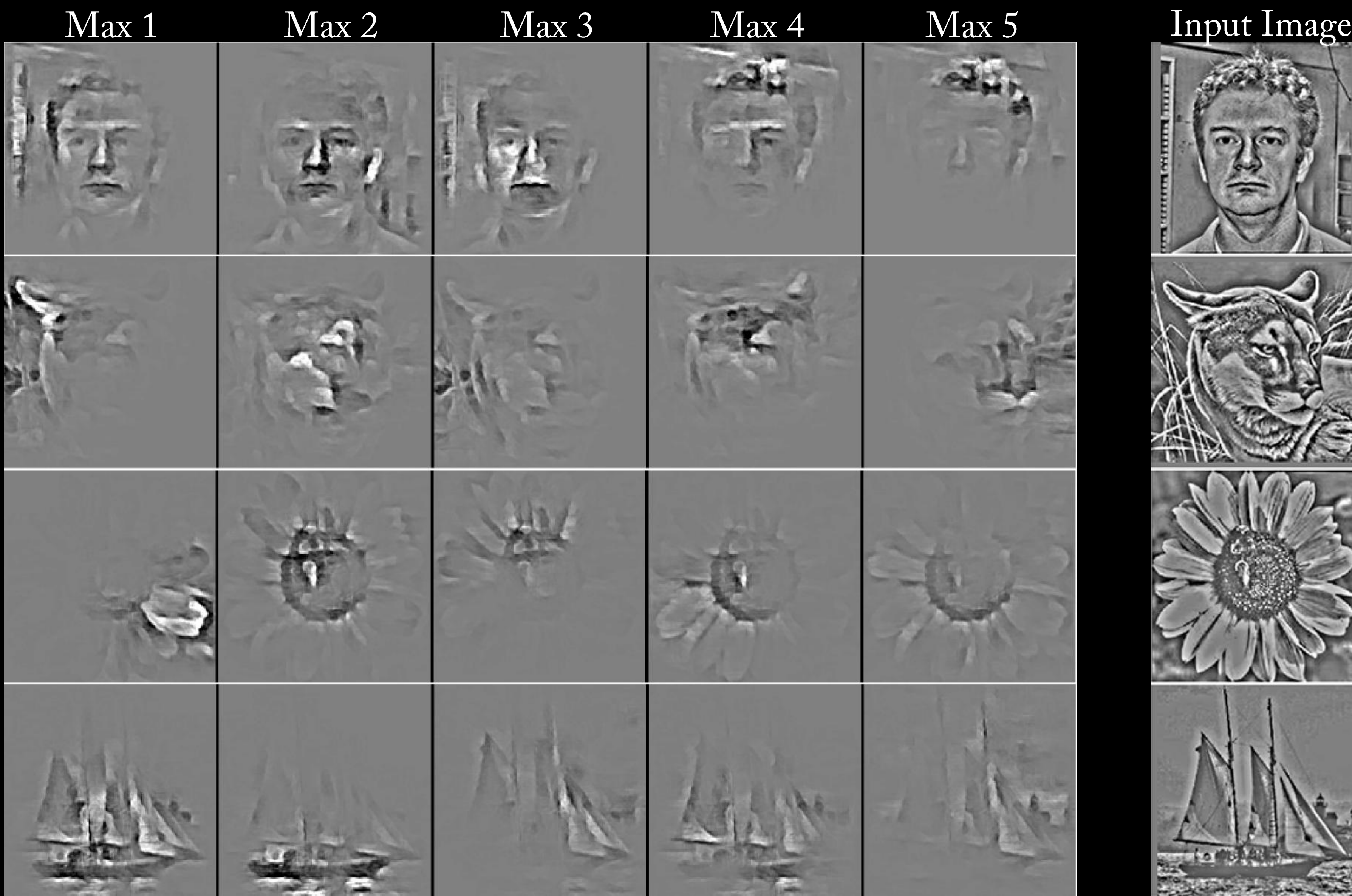

(to scale)

# Largest 3 activations at top layer

- Pixel space visualization from individual top layer maxes.

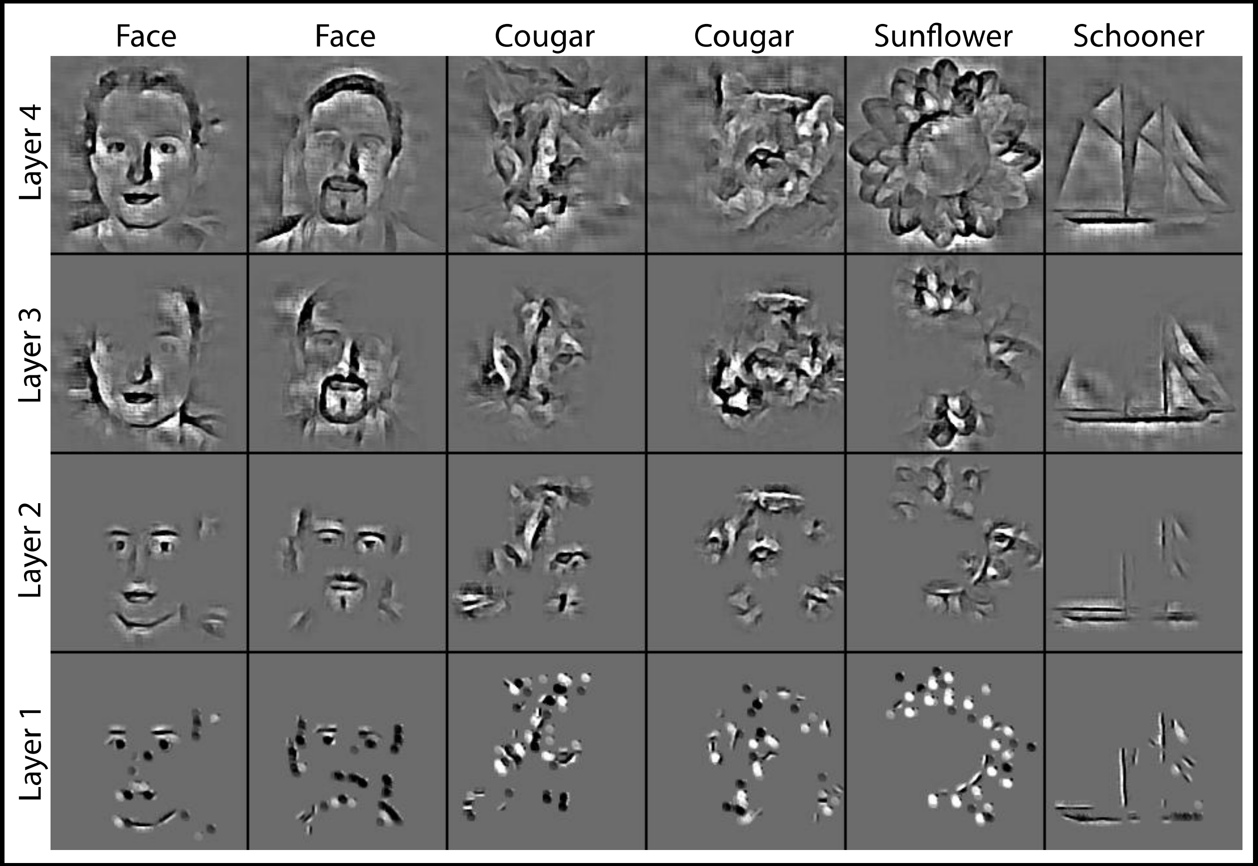


# Largest 5 activations at top layer

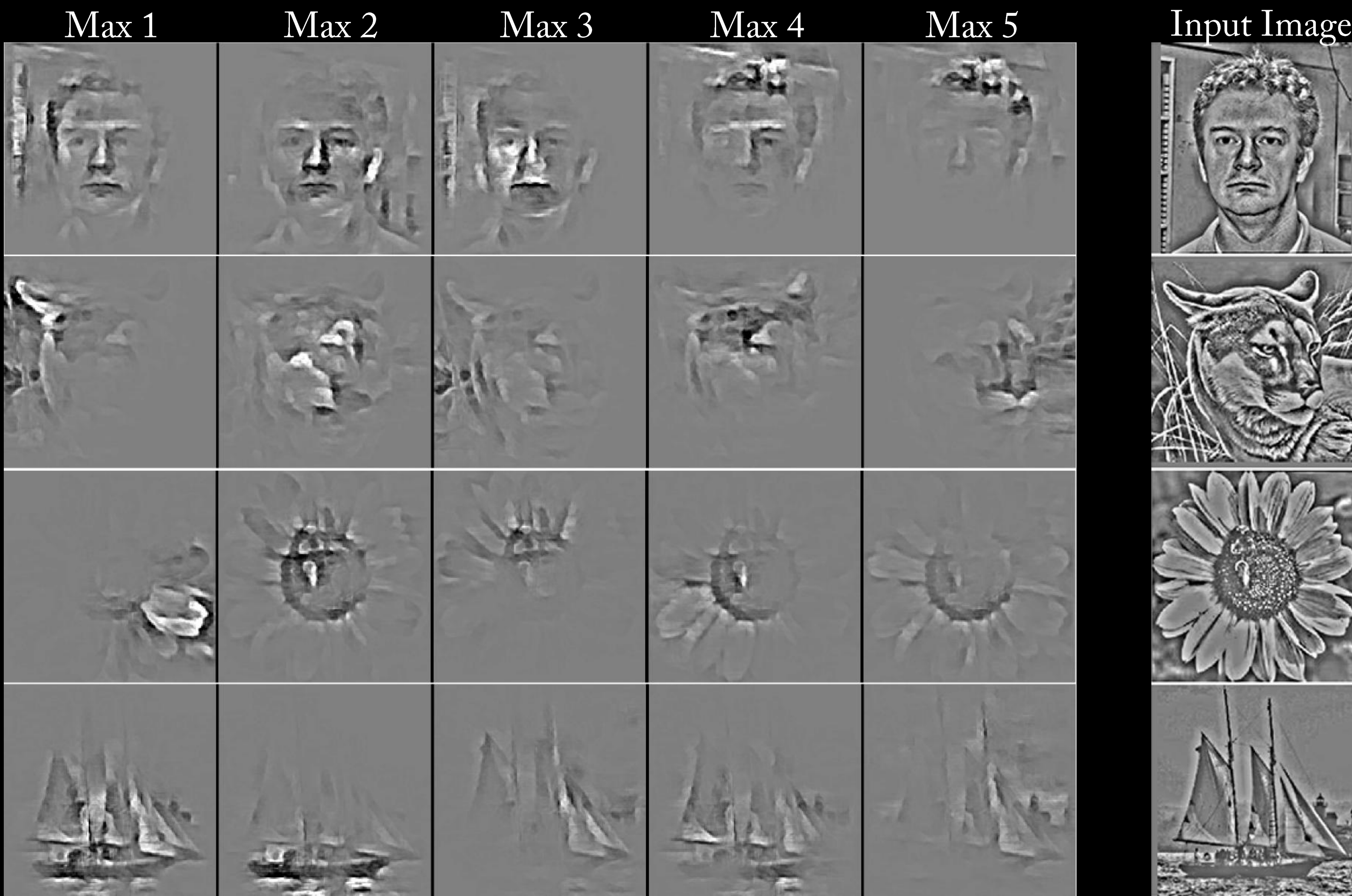


# Top-down Decomposition

- Pixel visualizations of strongest features activated from top-down reconstruction from single max in top layer.

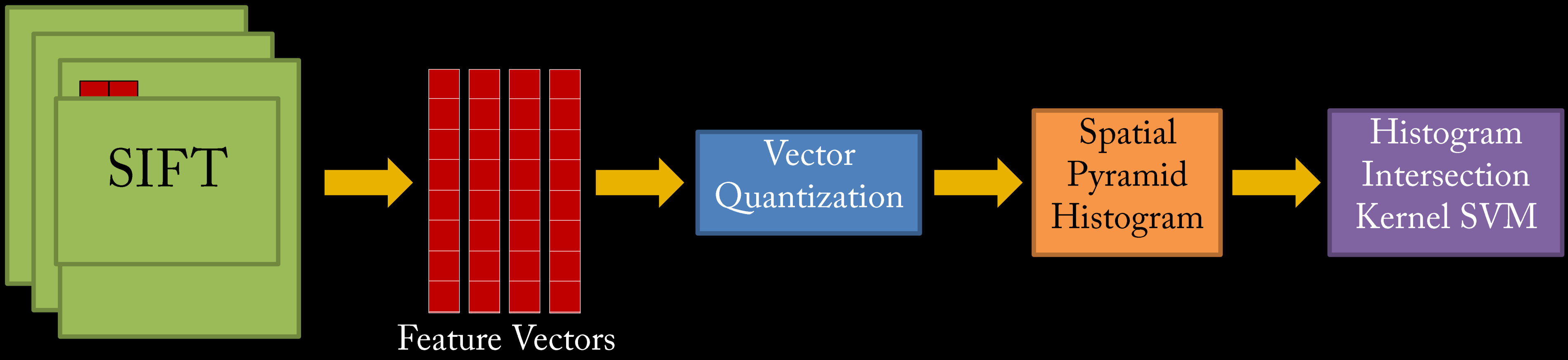


# Largest 5 activations at top layer



# Application to Object Recognition

- Use Spatial Pyramid Matching of Lazebnik et al. [CVPR'06]



# Classification Results: Caltech 101

- Use 1st layer activations as input to Spatial Pyramid Matching (SPM) of Lazebnik et al. [CVPR'06]

| | |
|---|---|
| Our model - layer 1 | $67.8 \pm 1.2\%$ |
| Chen *et al.* [3] layer-1+2 (ConvFA) | $65.7 \pm 0.7\%$ |
| Kavukcuoglu *et al.* [8] (ConvSC) | $65.7 \pm 0.7\%$ |
| Zeiler *et al.* [18] layer-1+2 (DN) | $66.9 \pm 1.1\%$ |
| Boureau *et al.* [2] (Macrofeatures) | $70.9 \pm 1.0\%$ |
| Jarrett *et al.* [7] (PSD) | $65.6 \pm 1.0\%$ |
| Lazebnik *et al.* [9] (SPM) | $64.6 \pm 0.7\%$ |
| Lee *et al.* [11] layer-1+2 (CDBN) | $65.4 \pm 0.5\%$ |

Convolutional Sparse Coding

Other approaches using SPM with Hard quantization

# Classification Results: Caltech 256

- Use 1st layer activations as input to Spatial Pyramid Matching (SPM) of Lazebnik et al. [CVPR'06]

| | |
|---|---|
| Our model - layer 1 | $31.2 \pm 1.0\%$ |
| Yang *et al.* [17] (SPM) | $29.5 \pm 0.5\%$ |

Other approaches using SPM with Hard quantization

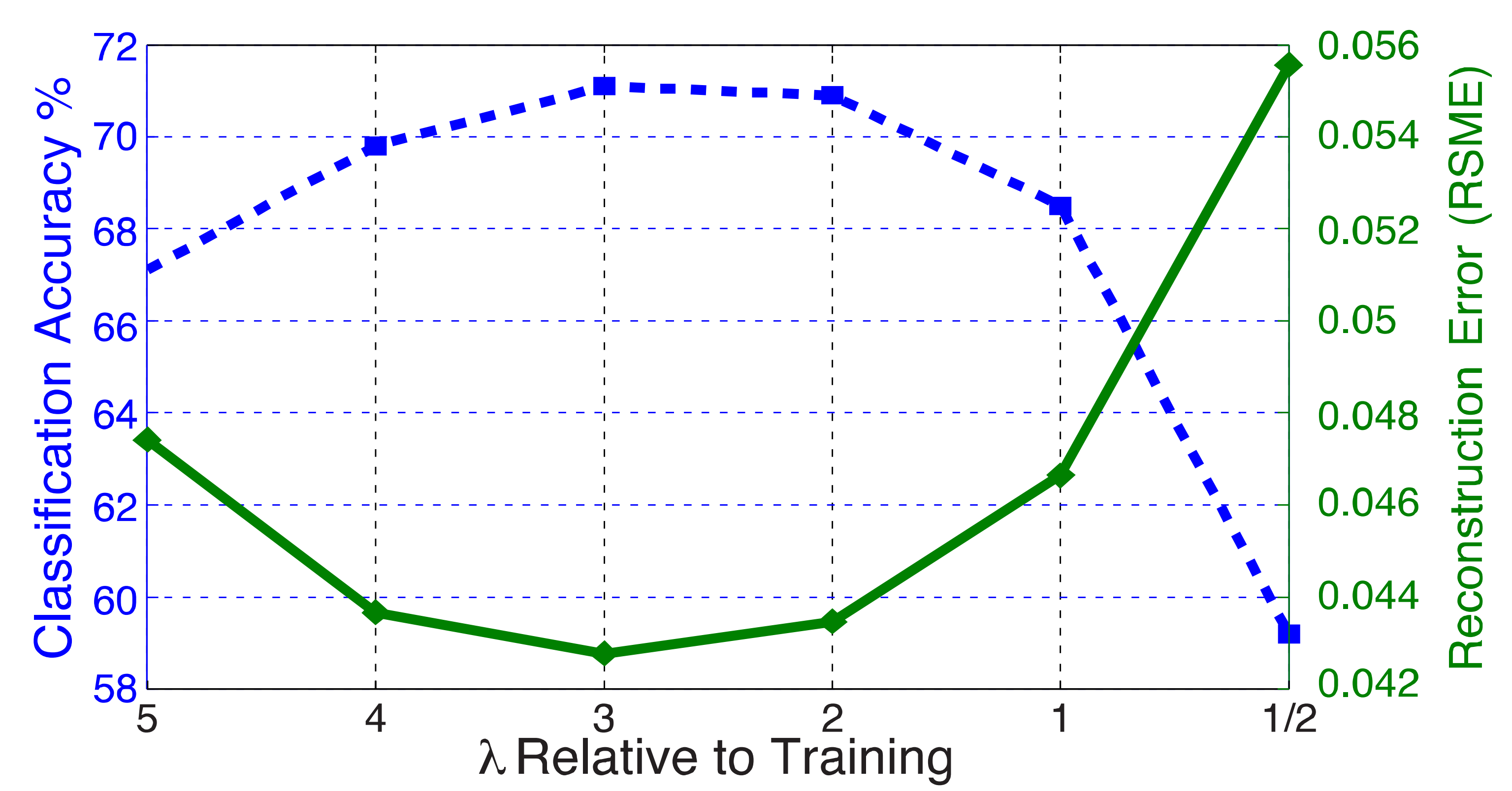
# Classification Results: Transfer Learning

- Training filters on one dataset, classify in another.

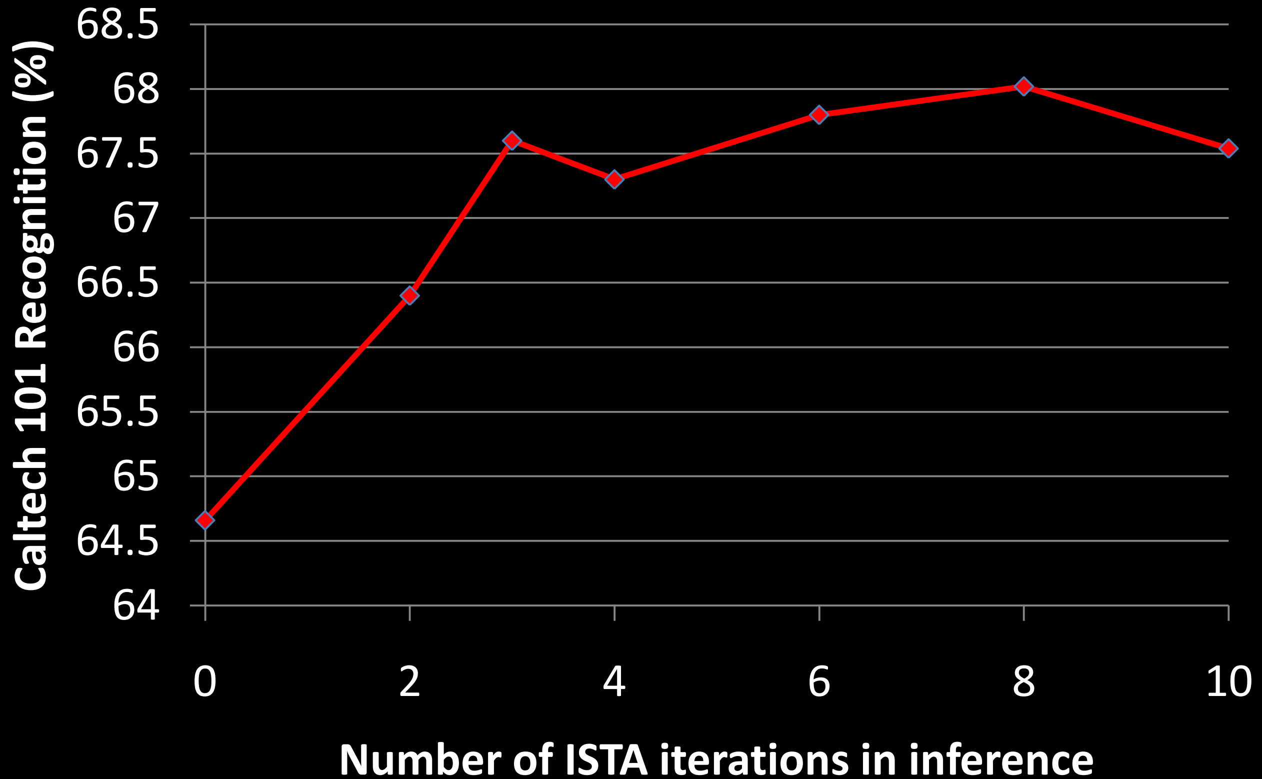
- Classifying Caltech 101
  - Using Caltech 101 Filters: 71.0 ± 1.0 %
  - Using Caltech 256 Filters: 70.5 ± 1.1 % (transfer)
- Classifying Caltech 256
  - Using Caltech 256 Filters: 33.2 ± 0.8 %
  - Using Caltech 101 Filters: 33.9 ± 1. 1 % (transfer)

# Classification/Reconstruction Relationship

- Caltech 101 classification for varying lambda.



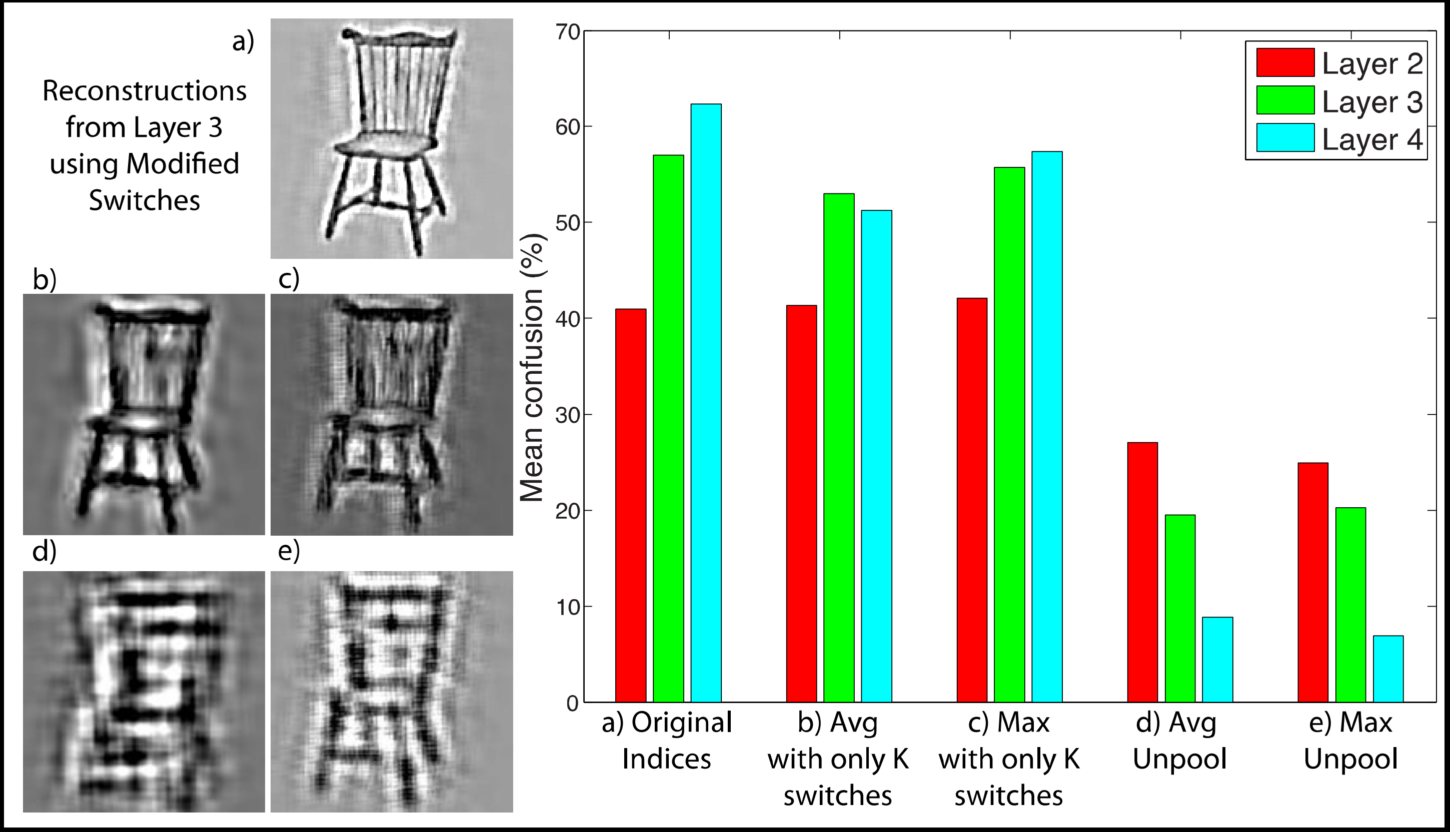
# Effect of Sparsity




- Explaining away, as induced by ISTA, helps performance
- But direct feed-forward (0 ISTA iterations) works pretty well
  - cf. Rapid object categorization in humans (Thorpe et al.)

# Analysis of Switch Settings

- Recons. and classification with various unpooling.

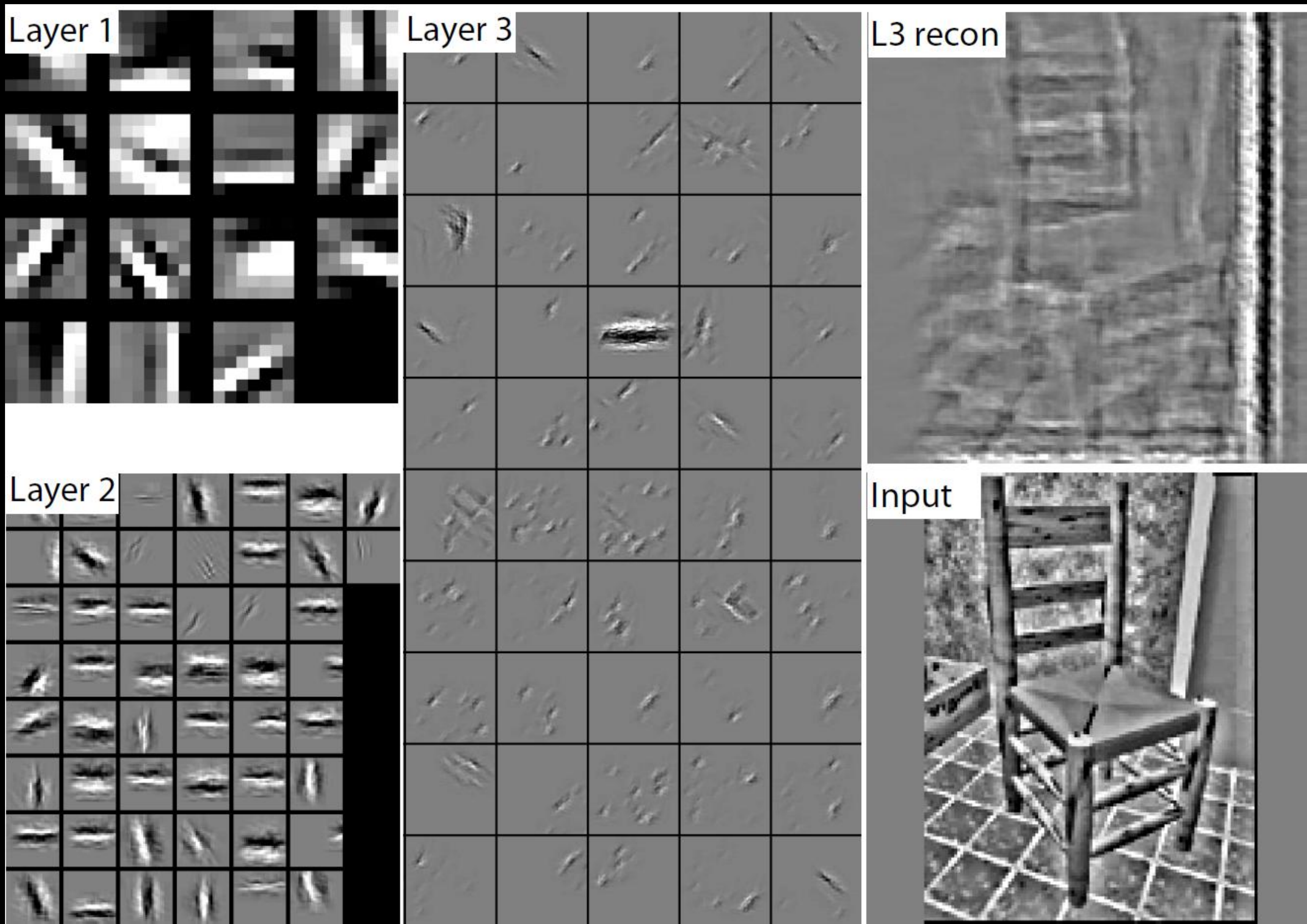


# Summary

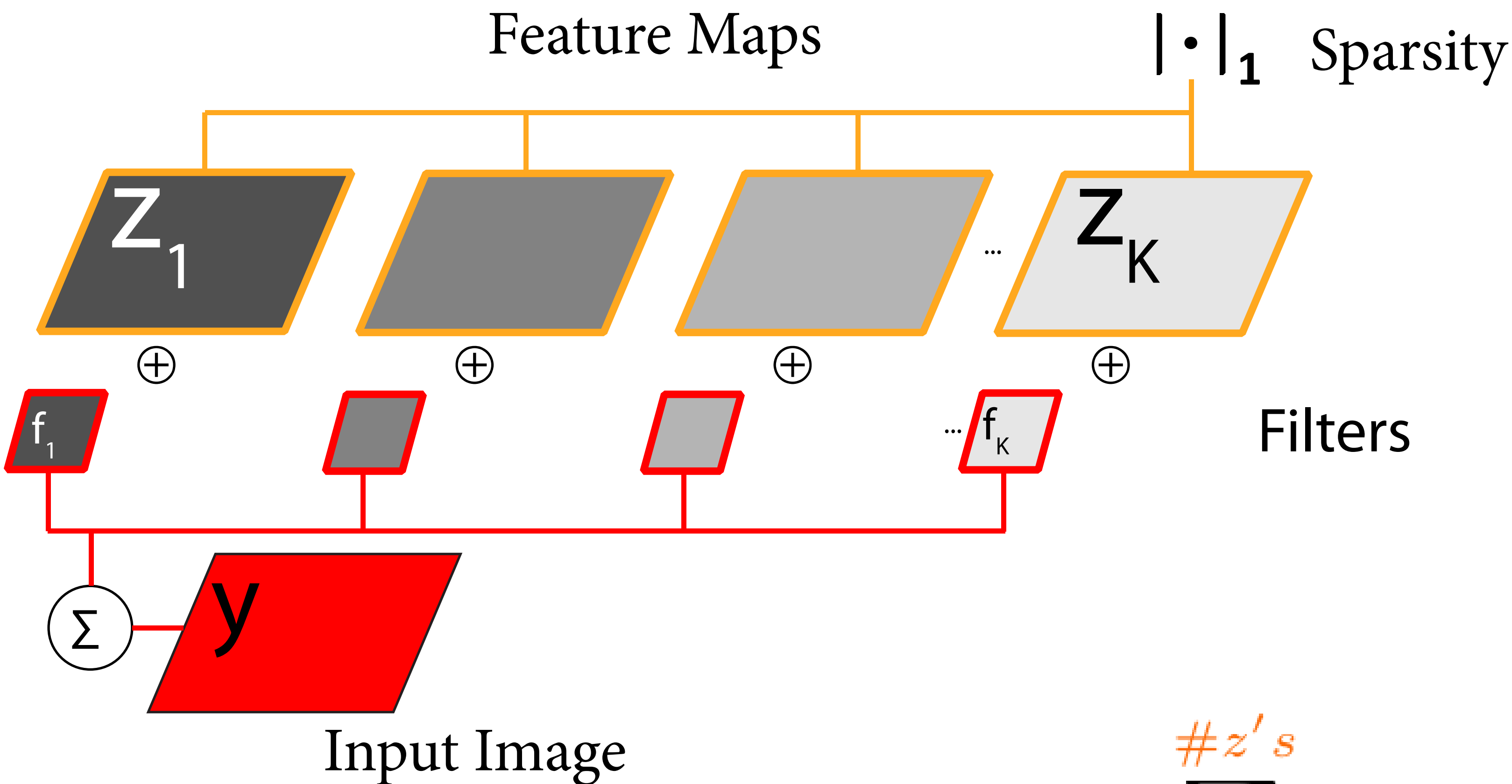
- Introduced multi-layer top-down model.
- Non-linearity induced by sparsity & pooling switches, rather than explicit function.
- Inference performed with quick ISTA iterations.
- Tractable for large & deep models.

- Obtains rich features, grouping and useful decompositions from 4-layer model.

# Model using layer-layer reconstruction

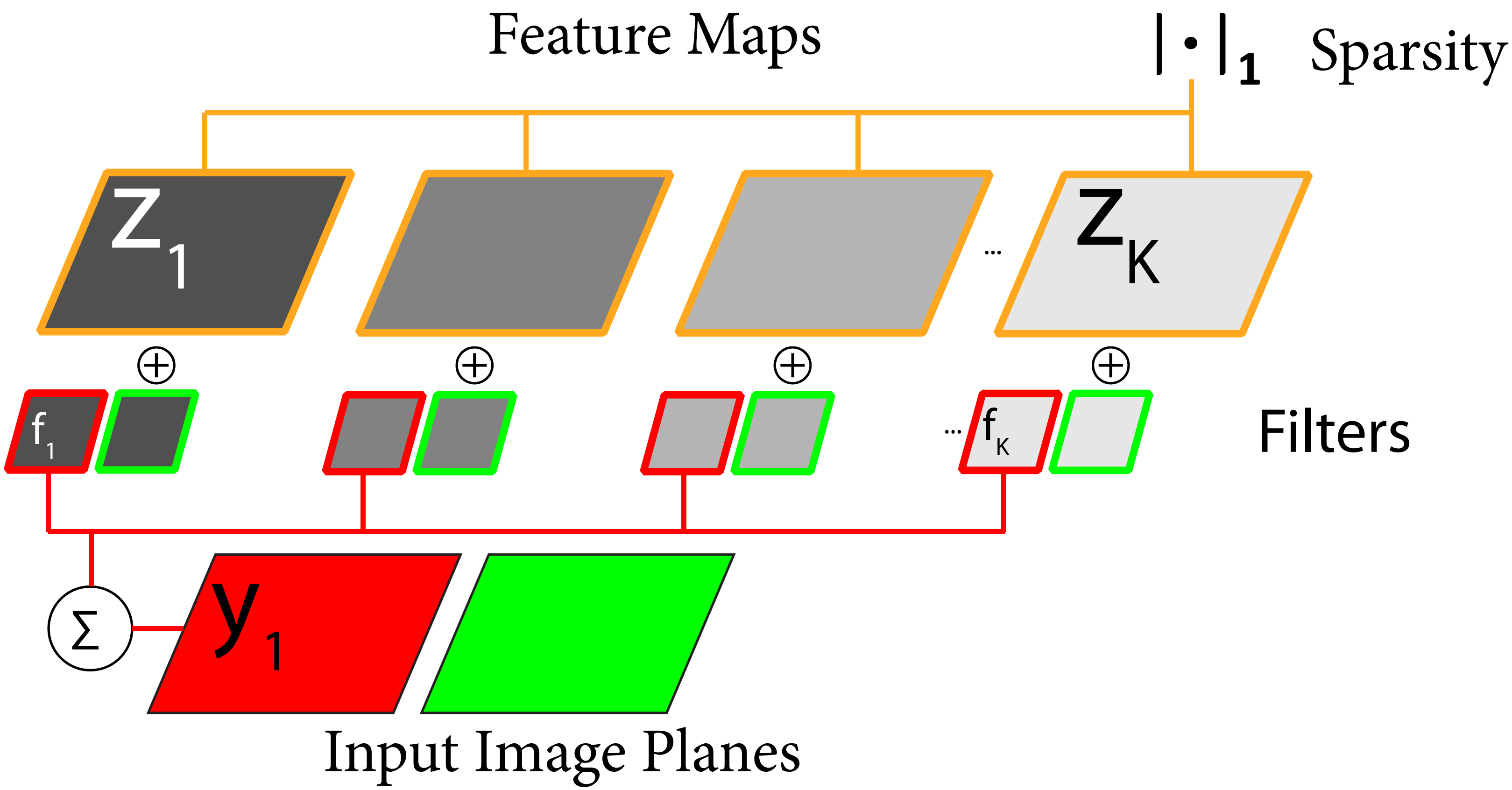


# Single Deconvolutional Layer

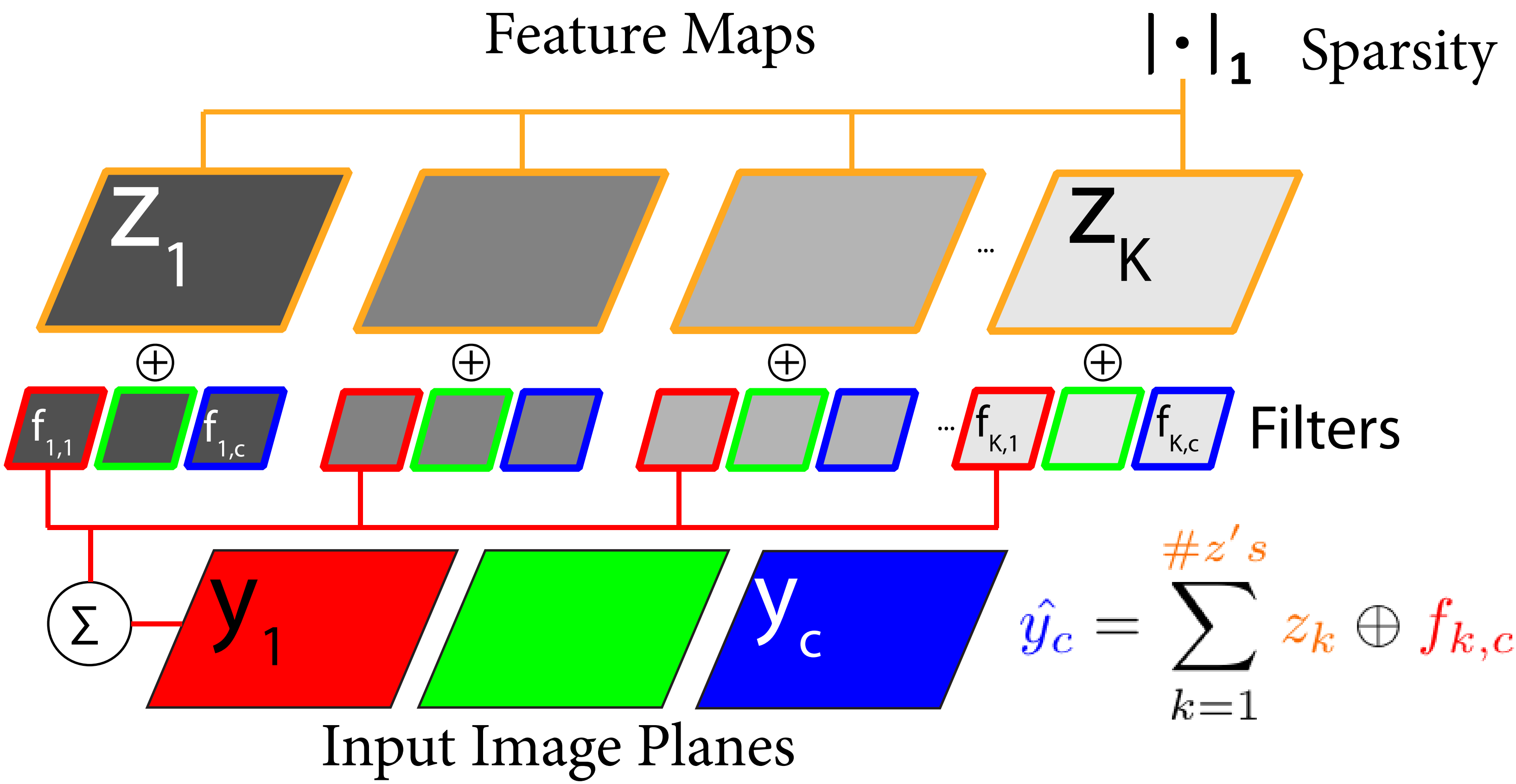



$$\hat{y} = \sum_{k=1}^{\#z's} z_k \oplus f_k$$

# Single Deconvolutional Layer

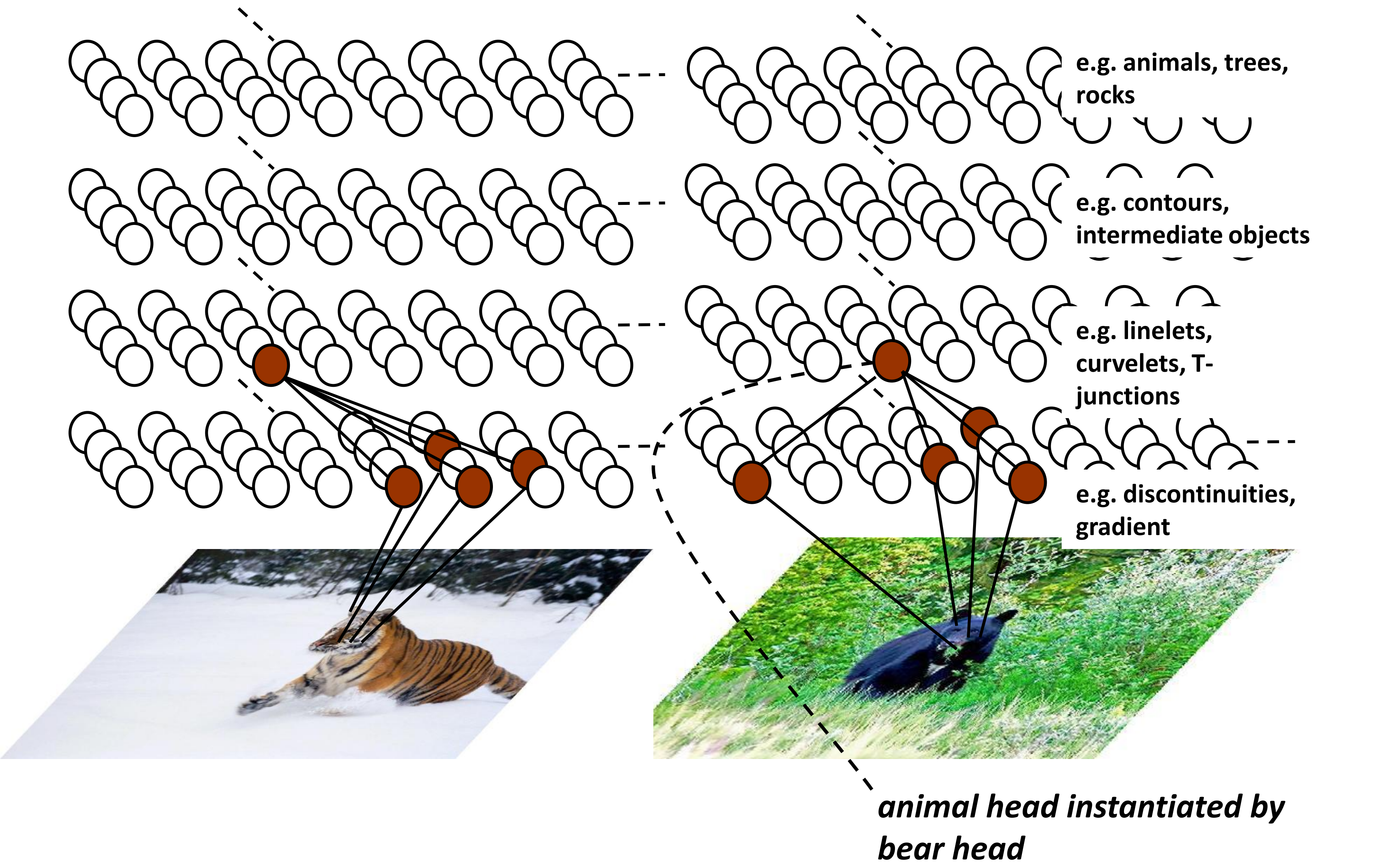


# Single Deconvolutional Layer



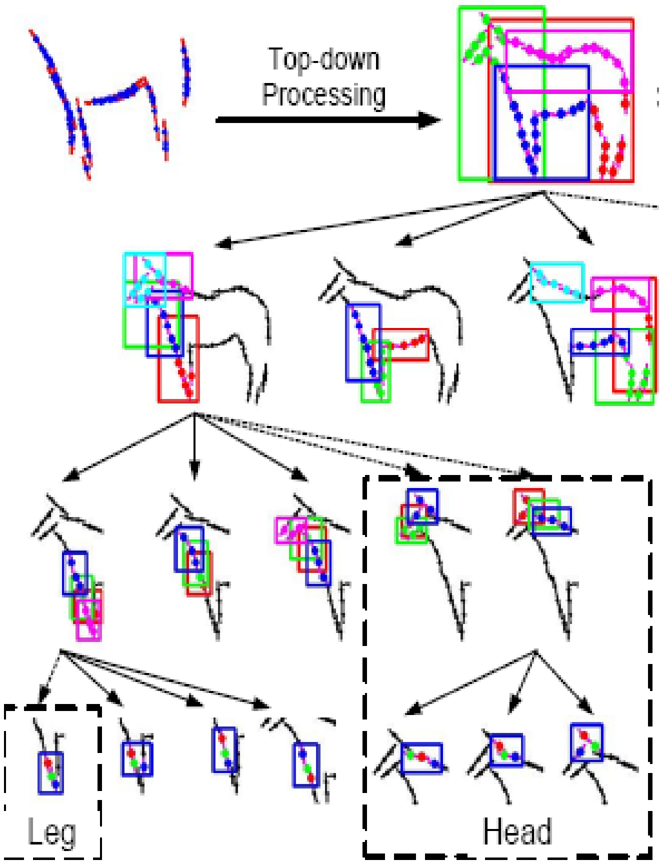
# Context and Hierarchy in a Probabilistic Image Model

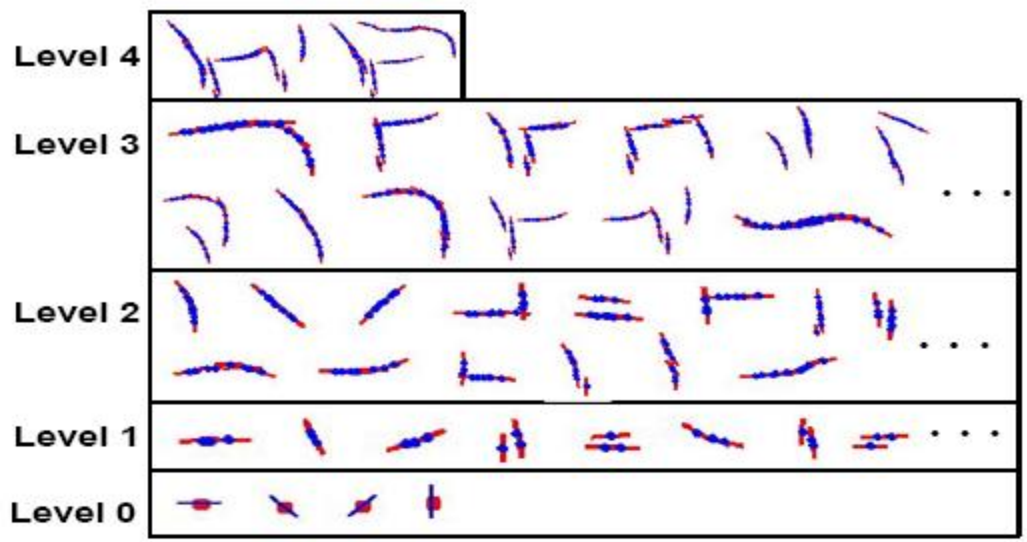
Jin & Geman (2006)



# A Hierarchical Compositional System for Rapid Object Detection

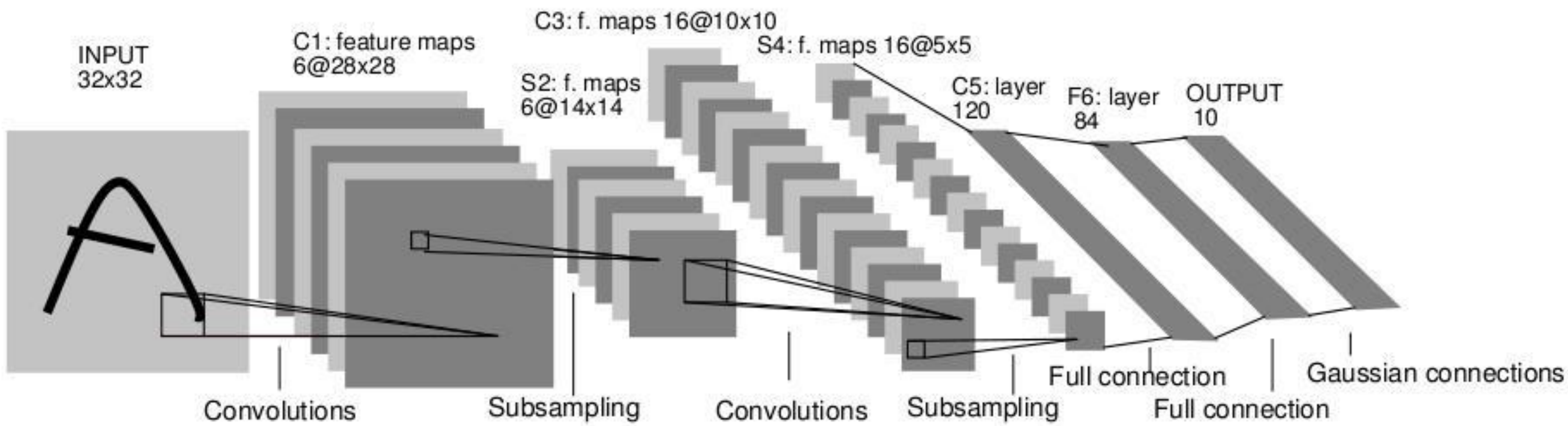
Long Zhu, Alan L. Yuille, 2007.




Able to learn #parts at each level

# Comparison: Convolutional Nets




LeCun *et al.* 1989

**Convolutional Networks**

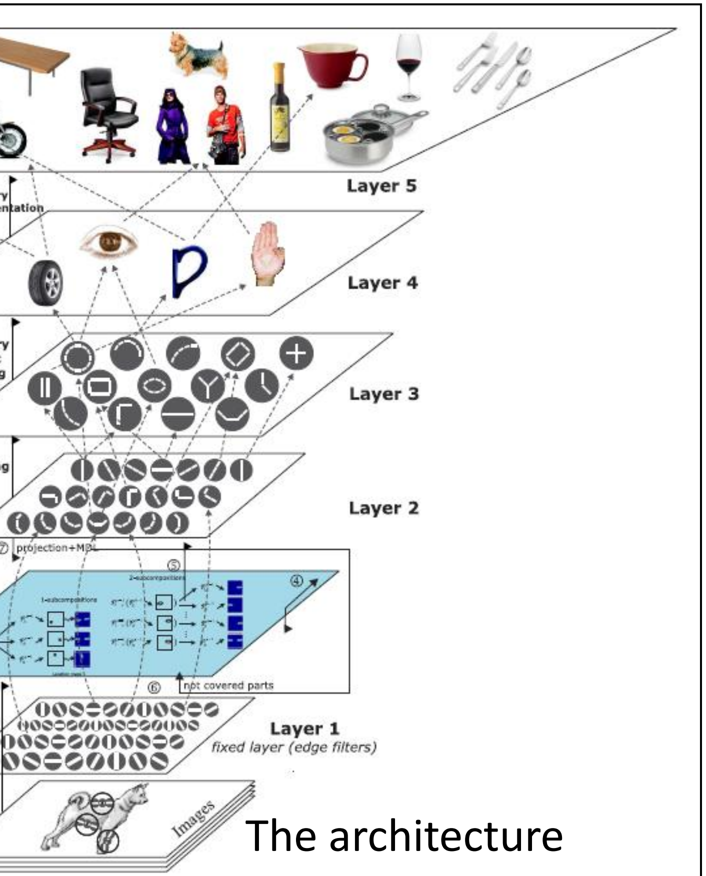
- Bottom-up filtering with convolutions in image space.
- Trained supervised requiring labeled data.

**Deconvolutional Networks**

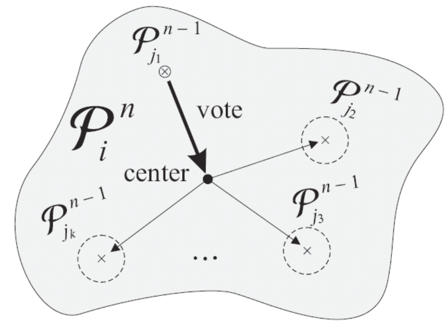
- Top-down decomposition with convolutions in feature space.
- Non-trivial unsupervised optimization procedure involving sparsity.

# Learning a Compositional Hierarchy of Object Structure

Fidler & Leonardis, CVPR'07; Fidler, Boben & Leonardis, CVPR 2008




The architecture




Parts model

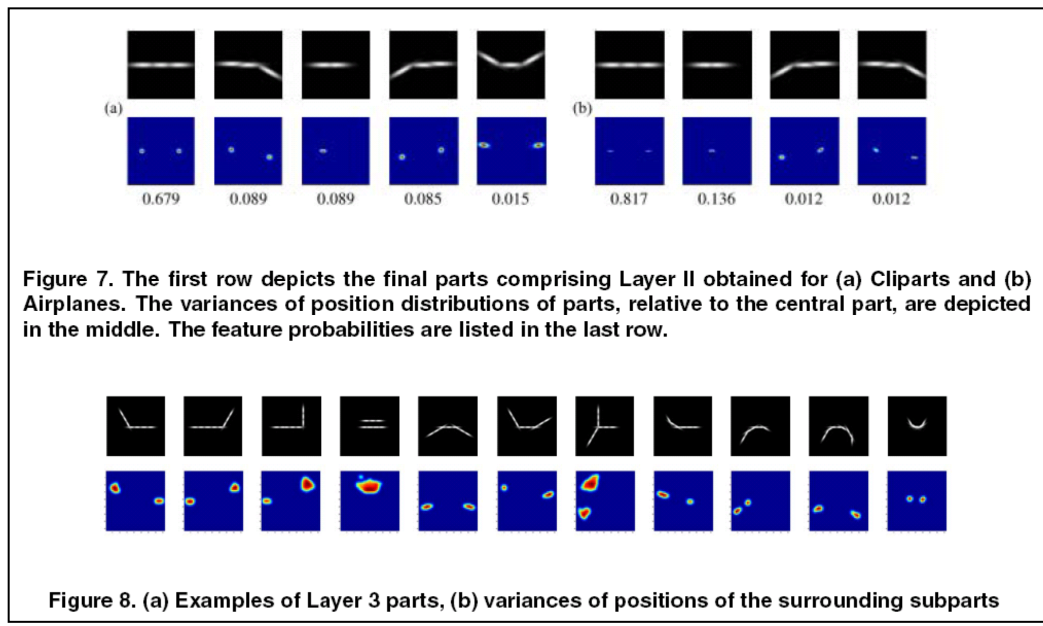



Figure 7. The first row depicts the final parts comprising Layer II obtained for (a) Cliparts and (b) Airplanes. The variances of position distributions of parts, relative to the central part, are depicted in the middle. The feature probabilities are listed in the last row.

Figure 8. (a) Examples of Layer 3 parts, (b) variances of positions of the surrounding subparts

Learned parts