Towards Scalable Representations of Object Categories: Learning a Hierarchy of Parts

Sanja Fidler and Ales Leonardis

Faculty of Computer and Information Science

University of Ljubljana, Slovenia

Overview

1. Goal
2. Motivation
3. Hierarchical object description
4. Learning part compositions
5. Results

Goal

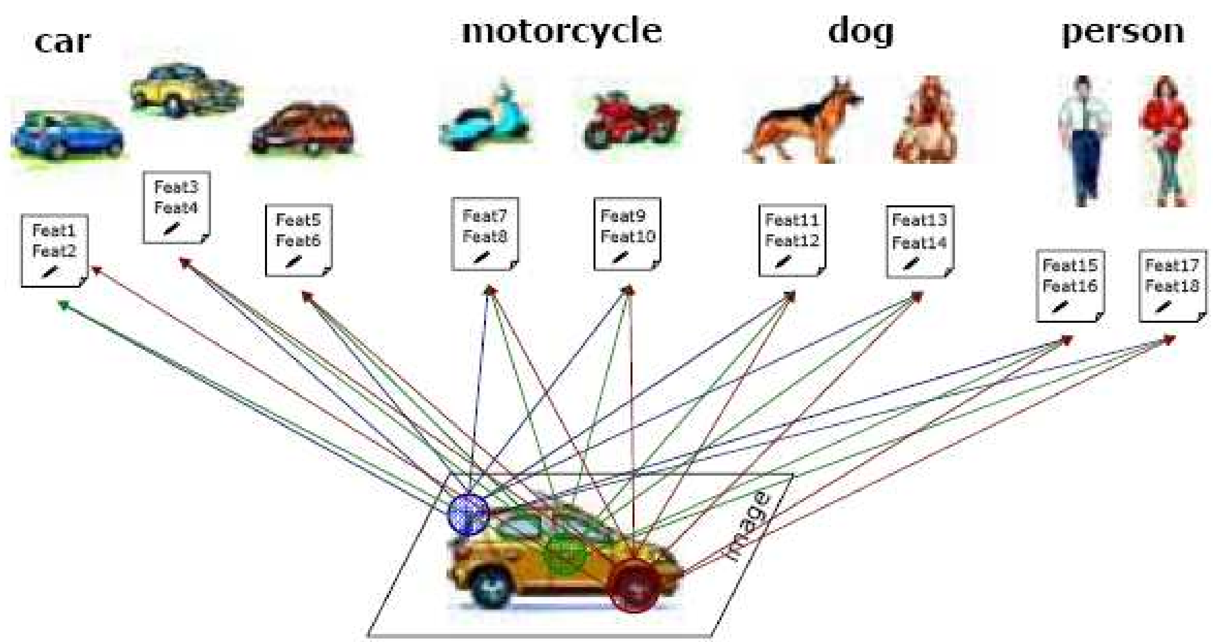
- Detection & Recognition of a large number of object categories

Desired Properties

- *Computational Plausibility*: Fast indexing & matching
- *Statistics driven learning*: Unsupervised learning of object parts for compact & concise representation
- *Robust detection*: Flexible, yet accurate models
- *Fast, Incremental Learning*: Easy addition of new object categories

Flat Representations

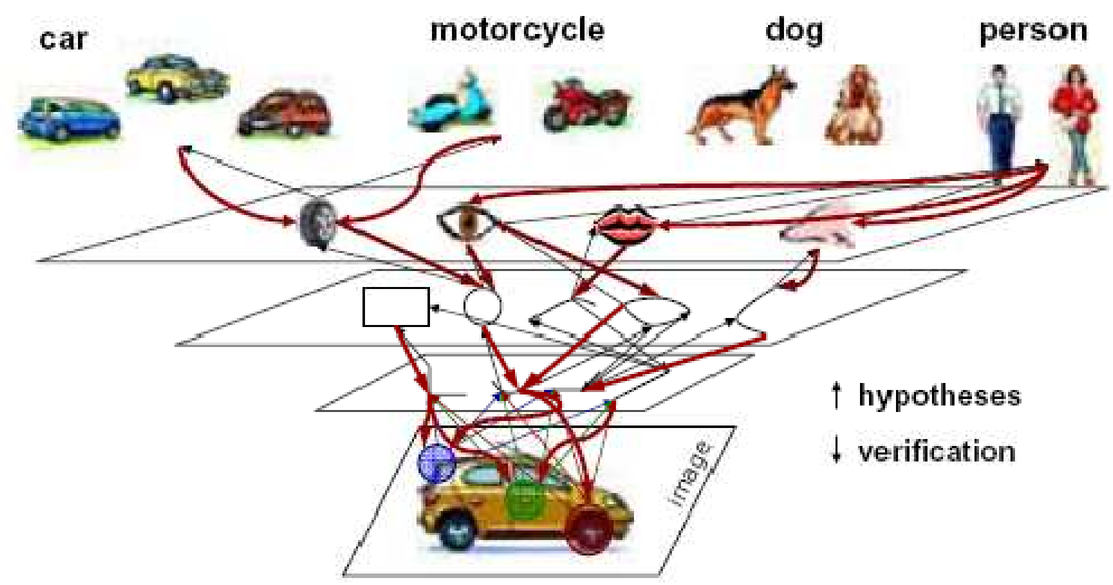
- Match each set of features to all object in the collection to find a good match
- Computationally demanding



Sanja Fidler and Aleš Leonardis. *Learning Hierarchical Representations of Object Categories*. EU Cognition meeting, Munich 2007

Hierarchical Representations

- A natural framework for indexing & matching

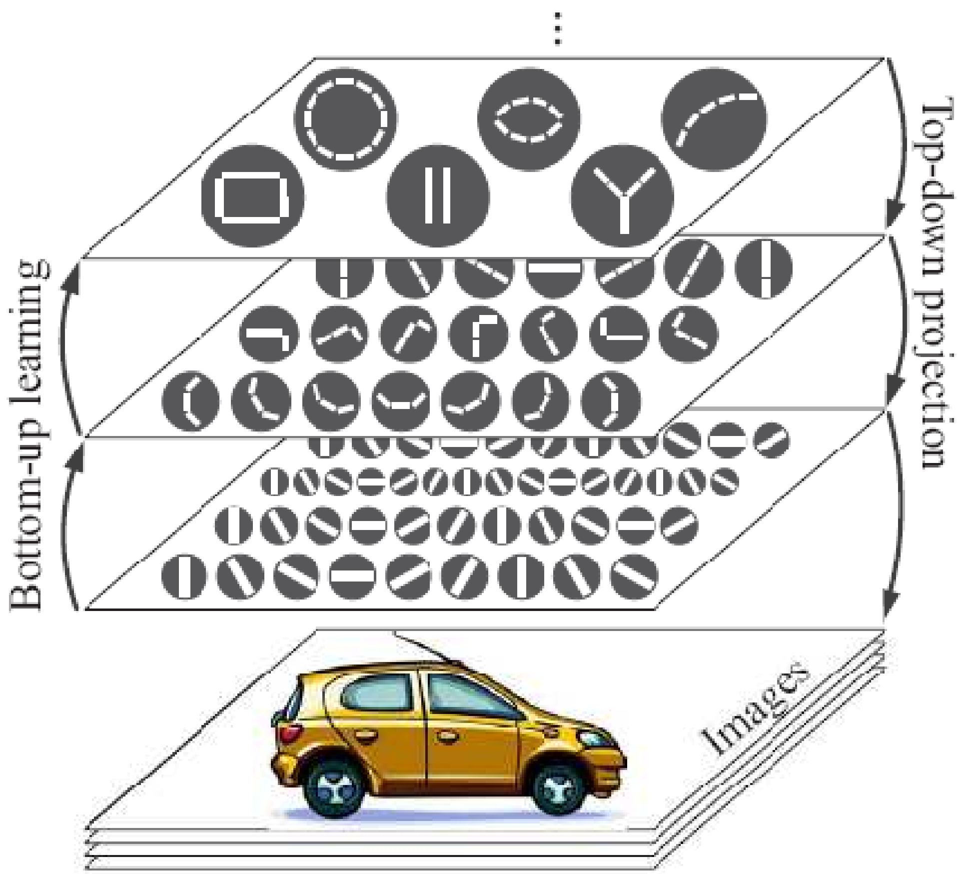


Sanja Fidler and Aleš Leonardis. *Learning Hierarchical Representations of Object Categories*. EU Cognition meeting, Munich 2007

Object Composition Hierarchy

- We wish to learn a hierarchical representation for the objects in an ***unsupervised*** manner
- Each object is made up of “parts” (*compositionability*)
- Parts appear in all levels of the hierarchy, where subsequent layers’ parts are compositions of parts from previous layers
  - All but the most basic parts are composed of parts

Part Hierarchy Demonstrated



Sanja Fidler and Aleš Leonardis. *Learning Hierarchical Representations of Object Categories*. EU Cognition meeting, Munich 2007

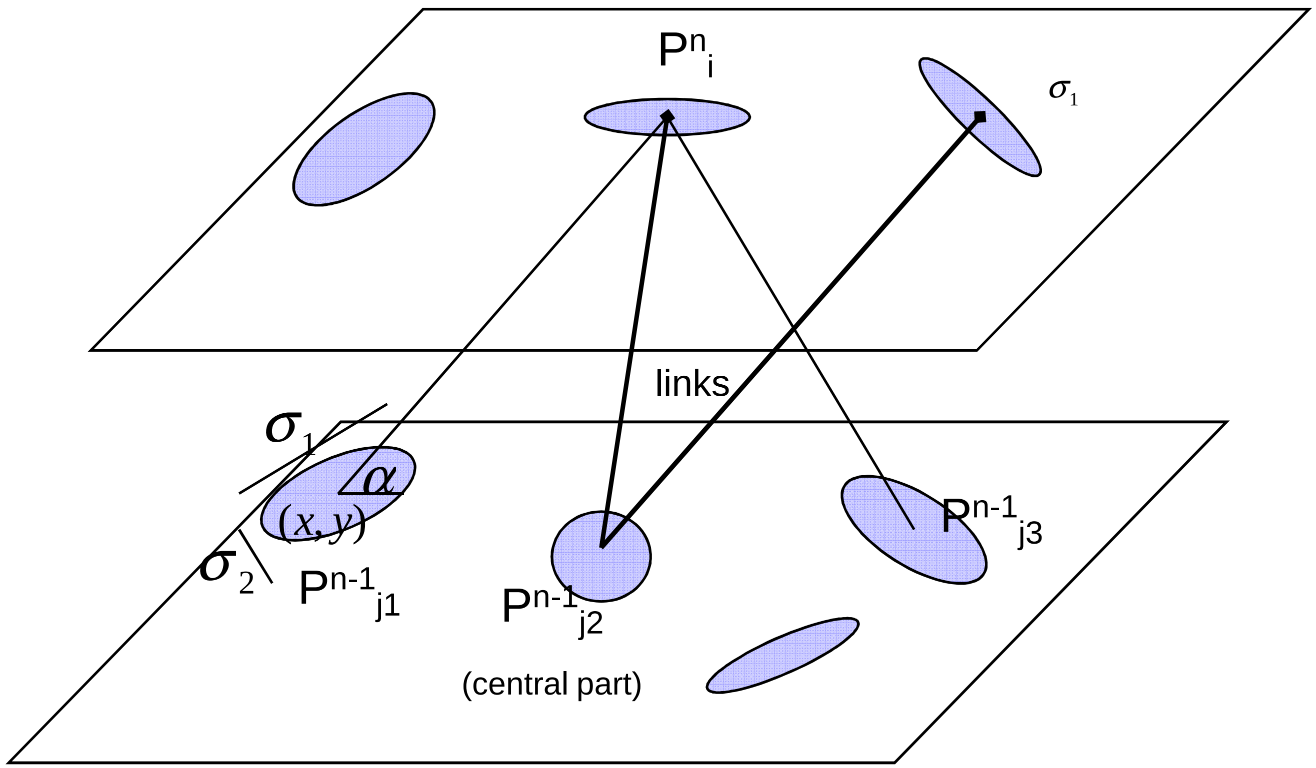
Hierarchy Structure

- L_n – n'th Layer
- P^n_i – i'th part of n'th layer, described by
  - Center of mass
  - Orientation
  - List of subparts from L_{n-1} , with position & orientation relative to P^n_i.

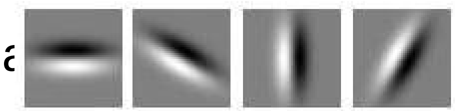
Hierarchy Structure ctd.

- *Central Part* – one specific subpart from L_{n-1} that indexes into $P^n_{i.}$ Its location and orientation are defined as (0,0), 0 resp.
- Contains a list of
- $\{P^{n-1}_j, \alpha_j, (x_j, y_j), (\sigma_{1j}, \sigma_{2j})\}_j$,denoting relative orientation, position, and position variance (via a gaussian) around (x_j, y_j).
- Links – a list of all parts from L_n that this part indexes to.

Hierarchy Structure



Initialization

- L_1 is a set of local oriented filters:
  - 8 Odd **Gabor** filters, oriented at 45 degrees a 
- At multiple scales
- L_1 parts are extracted from image via local maxima of filter responses (above threshold)
- Parts are denoted as $\{\pi^1_i\}_i$
- $\pi^n_i = \{P_i, \alpha_i, x_i, y_i\}$ is a realization of part *i* from layer *n,* with the orientation & position at which it was found in the image
- $\Lambda_n(\pi^n_i)$ – List of image locations to contribute to part π^n_i

Indexing & Matching

Algorithm 1 : Indexing and matching

INPUT: $\{\{\pi_i^{n-1}\}_i, \Lambda_{n-1}\}_{scale=1}^{n_{scales}}$

for each $scale$ **do**

  $\Pi_{scale} = \{\}$

  for each $\pi_i^{n-1} = \{\mathcal{P}_{i_k}^{n-1}, \alpha_i, x_i, y_i\}$ **do**

    Rotate the neighborhood of π_i^{n-1} by angle $-\alpha_i$

    for each part $\mathcal{P}^n \in Links(\mathcal{P}_{i_k}^{n-1})$ **do**

      Check for subparts of $\mathcal{P}^n$ according to their relative positions and spatial variance

      if subparts found **then**

        add $\pi^n = \{\mathcal{P}^n, \alpha_i, x_i, y_i\}$ to Π_{scale}, set $\Lambda_n(\pi^n) = \bigcup \Lambda_{n-1}(\pi_j^{n-1})$, where π_j^{n-1} are the found subparts of $\mathcal{P}^n$.

      end if

    end for

  end for

end for

Perform local inhibition over $\{\pi_i^n\}$

return $\{\{\pi_i^n\}_i, \Lambda_n\}_{s=1}^{n_{scales}}$

Learning Part Hierarchy

- We'd like to reduce computational complexity, by:
  - Choosing parts with few occurrences (reduces the subsequent matching process)
  - Create simple models (limit overall number of parts)
  - Perform local inhibition to remove part redundancy
- Learn layers and links sequentially:
  - Perform voting for each layer
  - Choose best composition of parts for higher layer
- In addition: Choose parts to cover images well

Incremental Learning of Layers

- L_1 : Oriented Gabor filters
- Subsequent layers: Learn compositions with *increasing complexity* (no. of parts), called *s-compositions.* Limit s to 4;
- An *s-composition* C^n_s is made up of s+1 parts (s parts + 1 central)

1-compositions

- Choose a part P^{n-1}_i with low avg. image frequency (N_i), to be the central part.
- Choose P^{n-1}_j s.t. $N_i \leq N_j$. From the neighboring features (neighborhood size chosen to minimize information loss)
- Perform Local inhibition to disregard parts having low novelty over central part
- $\{C^n_{s=1}\} = \{P^{n-1}_i, \{P^{n-1}_j, map_j\}\}$ is the set of possible 1-compositions.
- map_j – Spatial distribution of appearance of P^{n-1}_j conditioned on P^{n-1}_i being the central part.
- *Links*(P^{n-1}_i) – set of all compositions with P^{n-1}_i as the central part

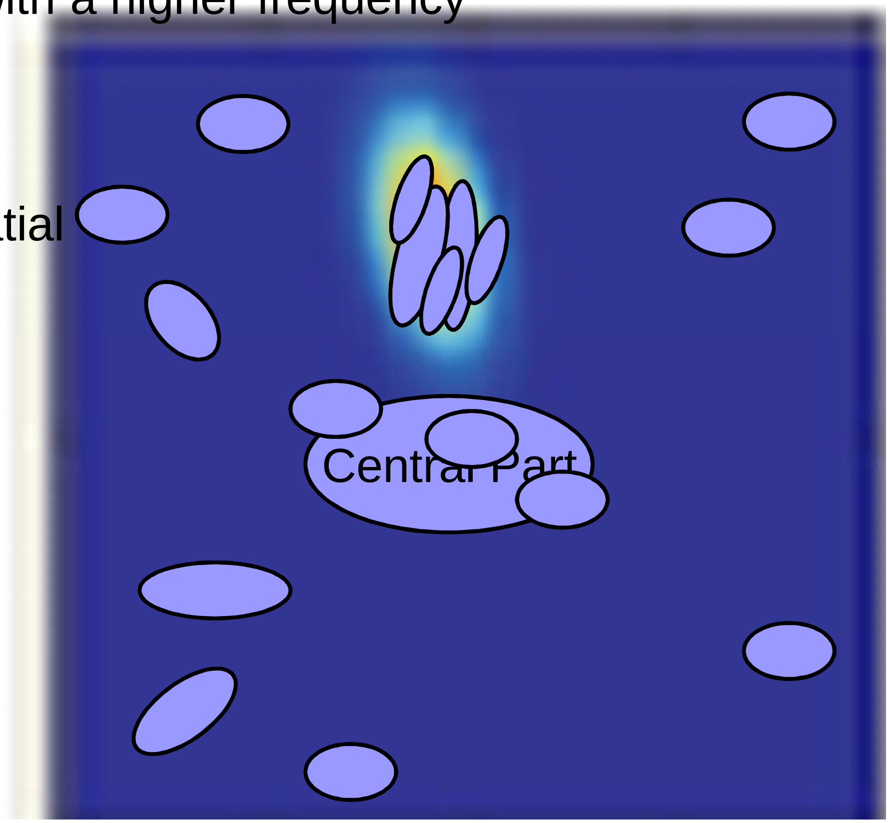
Formation of spatial maps

Candidate Parts with a higher frequency

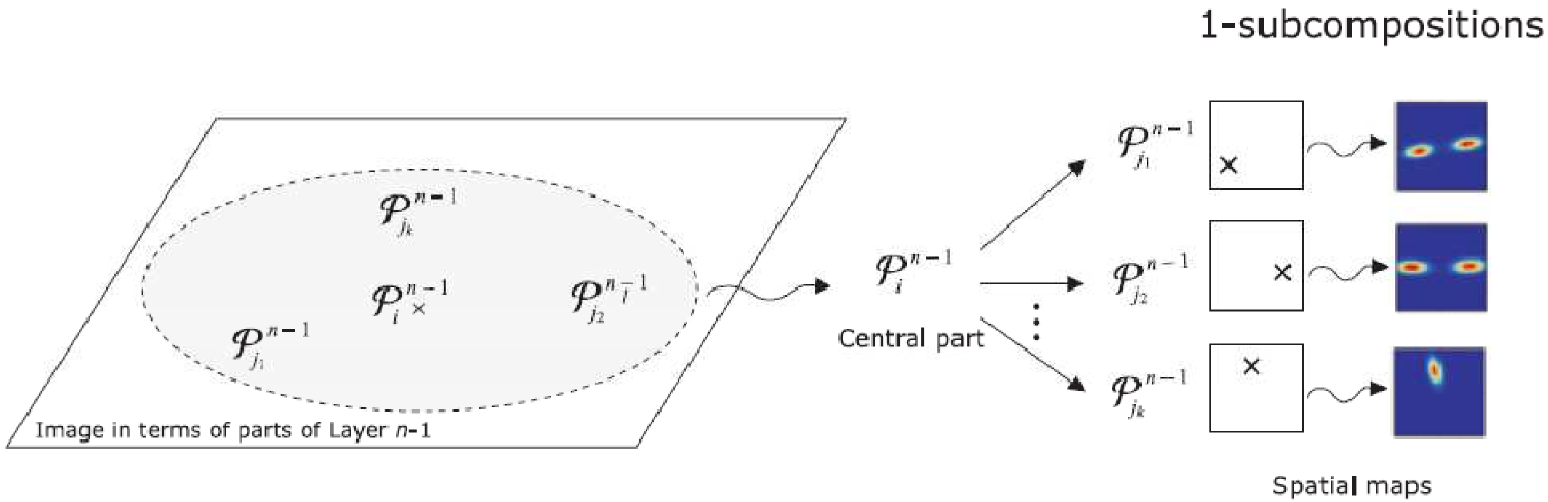
Local Inhibition

Find Peaks in Spatial Distribution

Model as Gaussian Density



Spatial Maps



- $(\sigma_{1j}, \sigma_{2j})\}$ – represent the spatial variability of the distribution of $P^{n-1}{}_j$ conditioned on the position of $P^{n-1}{}_i$

Spatial Maps ctd.

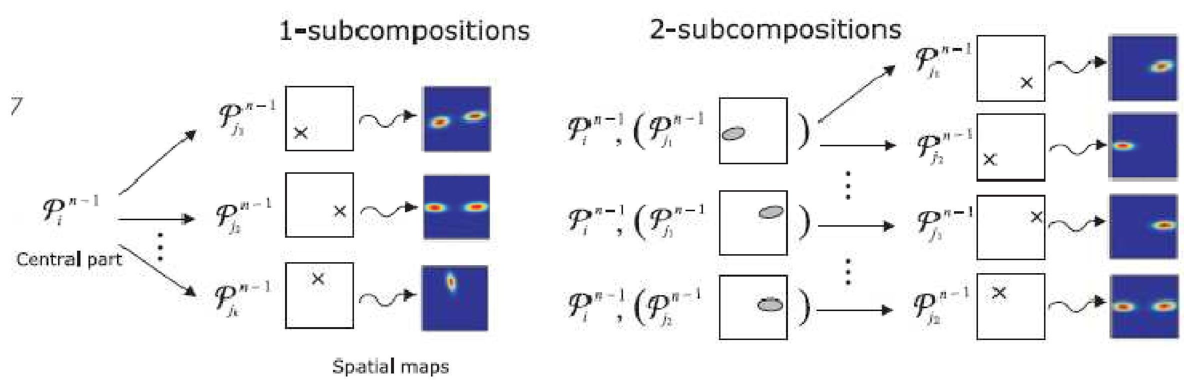
- probability for composition = sum of votes within area of variability / total inspected neighborhoods
- Keep only statistically significant 1-compositions:
  - $\Pr(C^n{}_1) \gg \Pr(P^{n-1}{}_i)\Pr(P^{n-1}{}_j)$
  - $N(C^n{}_1) > thresh_{n-1}$

S-Subcompositions

Central part

Additional part

- $\{C^n{}_s\} = \{P^{n-1}{}_i, \{P^{n-1}{}_{jm}, (x_{jm}, y_{jm}), ((\sigma_{1jm}, \sigma_{2jm})\}_{m=1,s-1}, \{P^{n-1}{}_j, map_j\}\}$
- i.e, build compositions using the previously learned s-1 compositions and one additional part.
- map_j is updated whenever *all* parts forming a certain composition are found in the local image neighborhood.
- Prune possible combinations similarly to 1-compositions.
- *When no new decompositions pass the set statistical significance threshold, the layer learning ends.*



Learning of S-subcompositions

Algorithm 2 : Learning of $s-$subcompositions

INPUT: Collection of images
for each image and each $scale$ **do**
  Preprocessing:
  process image with $\mathcal{L}_1$ parts to produce $\{\{\pi_i^1\}_i, \Lambda_1\}$
  for $k = 2$ to $n - 1$ **do**
    $\{\{\pi_i^k\}_i, \Lambda_k\}$ = Algorithm 1($\{\{\pi_i^{k-1}\}_i, \Lambda_{k-1}\}$)
  end for
  Learning:
  for each $\pi_i^{n-1} = \{\mathcal{P}^{n-1}, x_i, y_i\}$ **do**
    for each $\mathcal{C}_s^n \in Links(\mathcal{P}^{n-1})$ **do**
      Find all parts π^{n-1} within the neighborhood
      Match the first $(s-1)$-subparts contained within the subcomposition relative to the central part
      Perform local inhibition: $\Lambda(neigh.parts) := \Lambda(neigh.parts) \setminus \bigcup \Lambda(found\,subparts)$. Keep parts that have $|\Lambda(\pi^{n-1})| \geq thresh \cdot |\Lambda(\pi_i^{n-1})|$. We use $thresh = 0.5$.
      If all $s - 1$ subparts are found and s-th subpart appears anywhere in the neighborhood, update the spatial map for the s-th subpart.
    end for
  end for
end for

Part Selection & grouping

- To control the complexity, compositions are removed if parts within them index too many parts in subsequent layers
- Usually 10-20 links per part
- Determined by computational resources
- Parts are deemed equal if average part overlap over set of images is large enough; this removes different yet perceptually similar parts

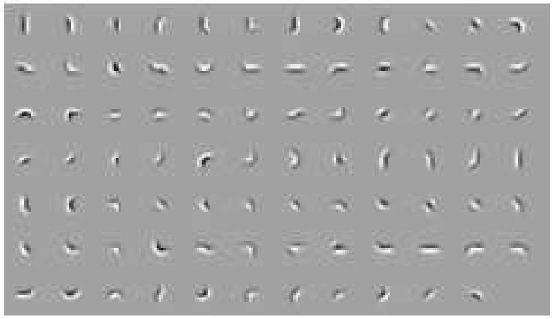
Learning process

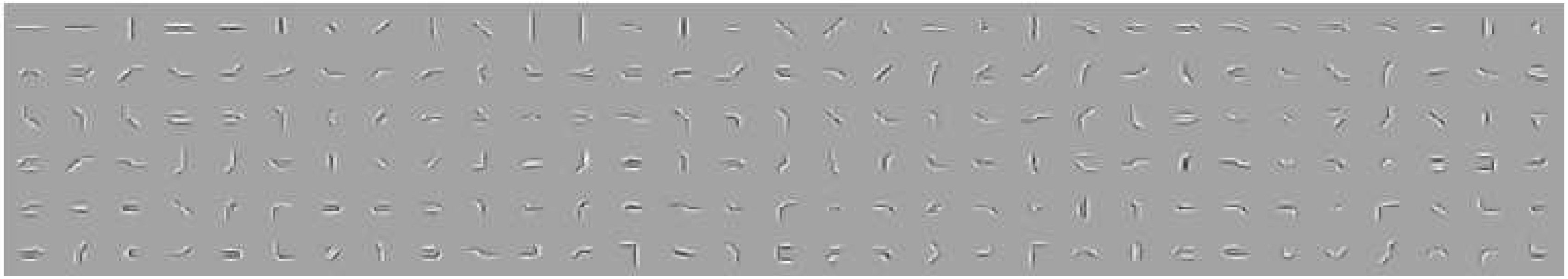
- Lower layers: Category independent, containing parts shared among many object classes
→Learn a union of image classes
- Higher layers: Number of part combinations increases rapidly. On the other hand, part combinations “specialize” for object categories
→ learn for each category by itself

Results

- Learned a collection of 3200 images from 15 categories (cars, faces, mugs, dogs…)
- Results are comparable with current approaches regarding object *Localization* for single-scale, and slightly better for multi-scale.

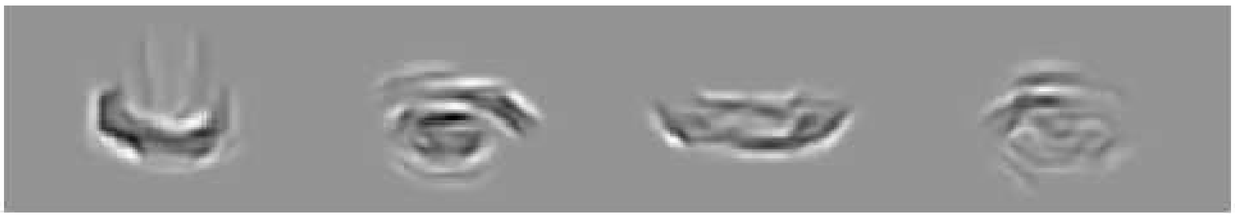
L_2 ,L_3 (non-specific)



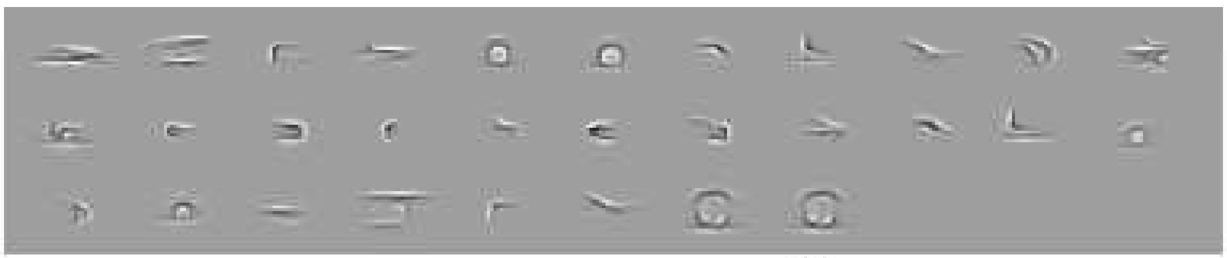


L_4 (category specific)

faces

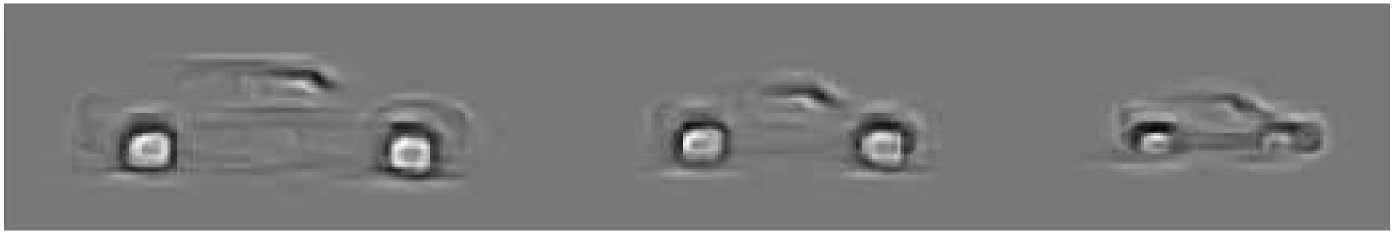


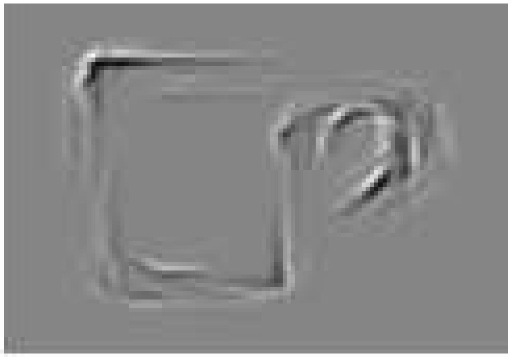
cars



mugs

L_5





Conclusions & Properties

- Low-Level parts are mostly category independent
- Mid-Level parts take on intuitive, familiar shapes (wheels, eyes, handles)
- High levels still require supervision…
- Number of indicative parts per image drops significantly for higher layers

Summary

- A hierarchical representation for efficient indexing & matching
- High level sparseness allows for a large number of visual categories
- Adding new objects is easy since most low-level features are shared between objects

Questions?