Lecture 15

BOW and Object detection

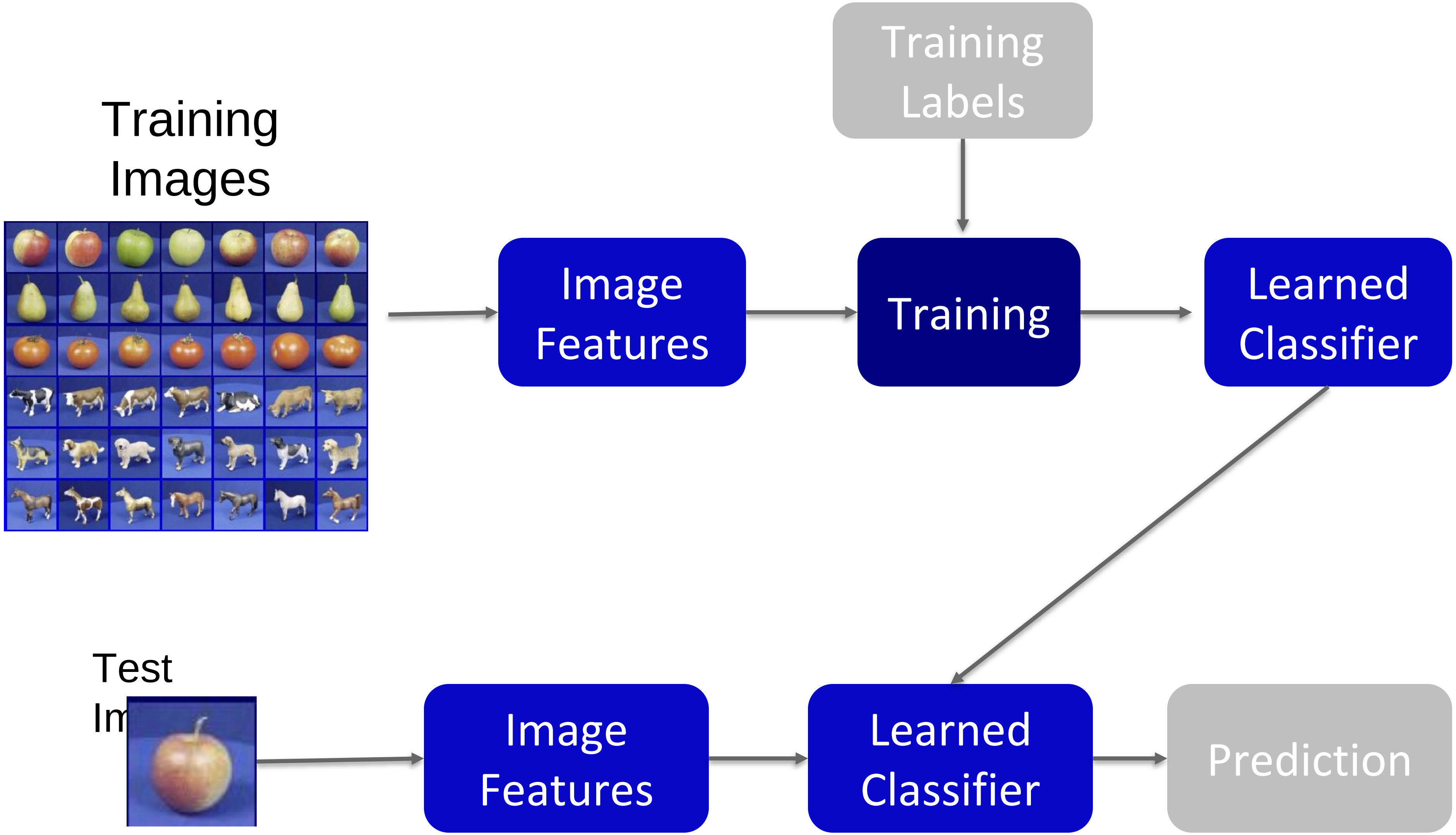
Linda Shapiro Nov 18, 2025

Administrative

A4 is out

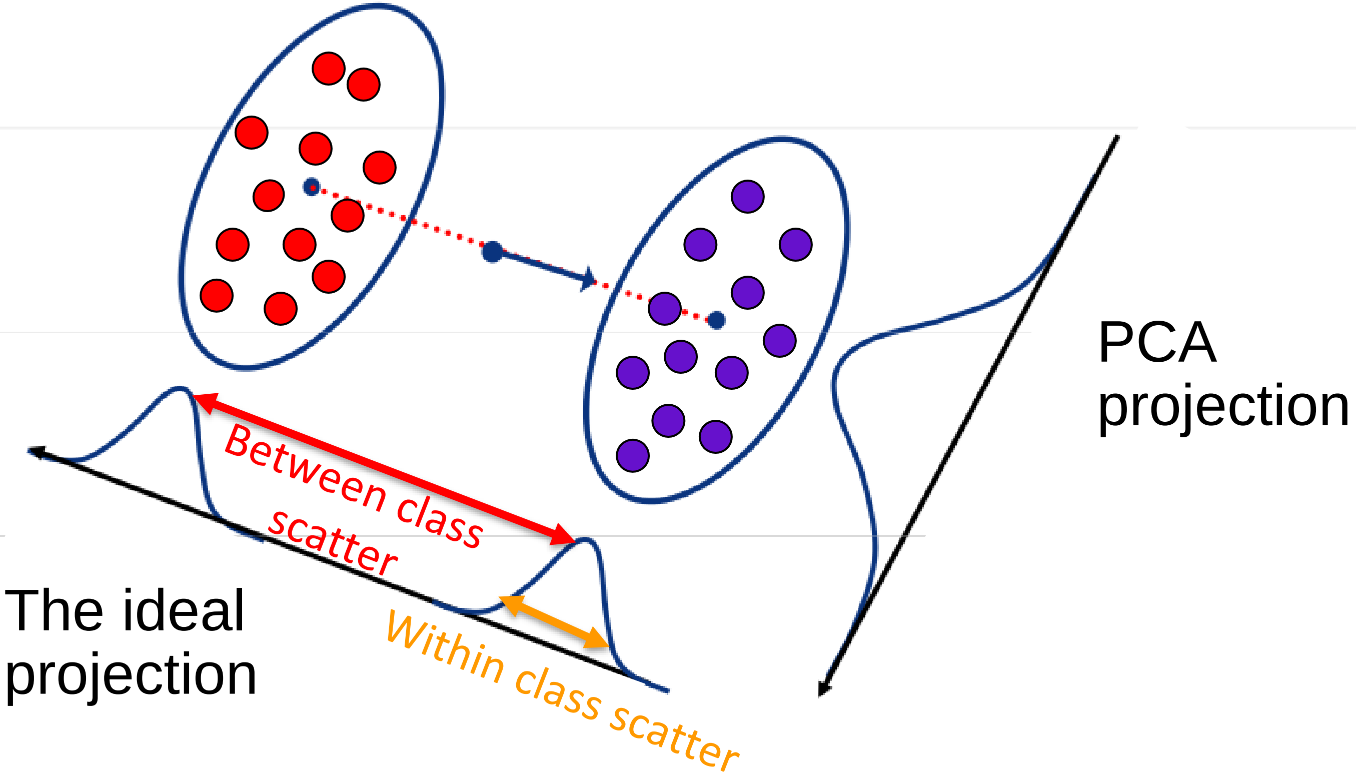
- Due Nov 25

So far: A simple recognition pipeline



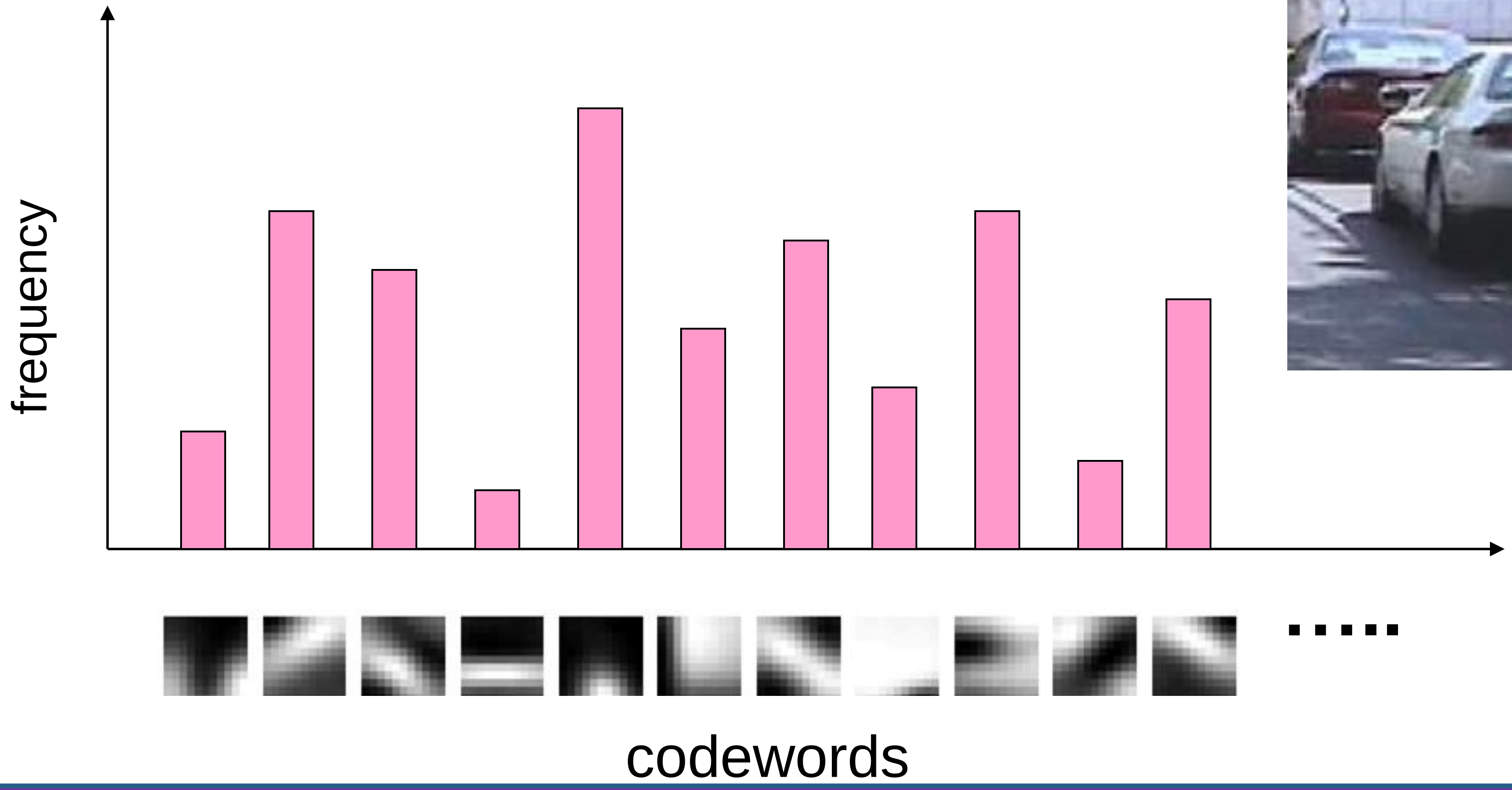
So far: PCA versus LDA WE DID NOT DO LDA BUT HERE IT IS!

We want a projection that maximizes: $J(w) = \max \frac{between\ \ class\ \ scatter}{within\ \ class\ \ scatter}$



Not YET: Bag of words features

- Every image now becomes a k-dimensional histogram representation.
- We can use these features for any recognition task.



Today’s agenda

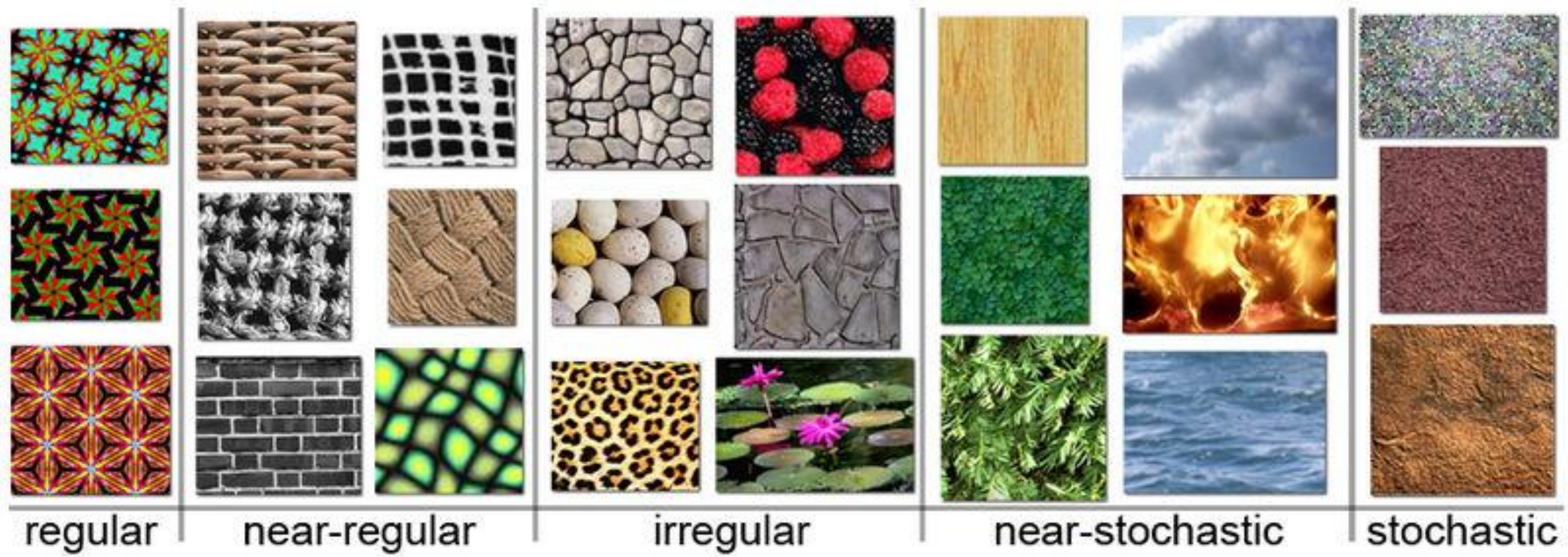
- **BOW**
- Spatial pyramids
- Object detection
  - Task and evaluation
- A simple detector
- Deformable parts model

Main idea: create a vocabulary of filters that would be able to recognize patches of specific objects

Object → Bag of ‘words’

The size of the vocabulary will determine the size of the feature dimension.

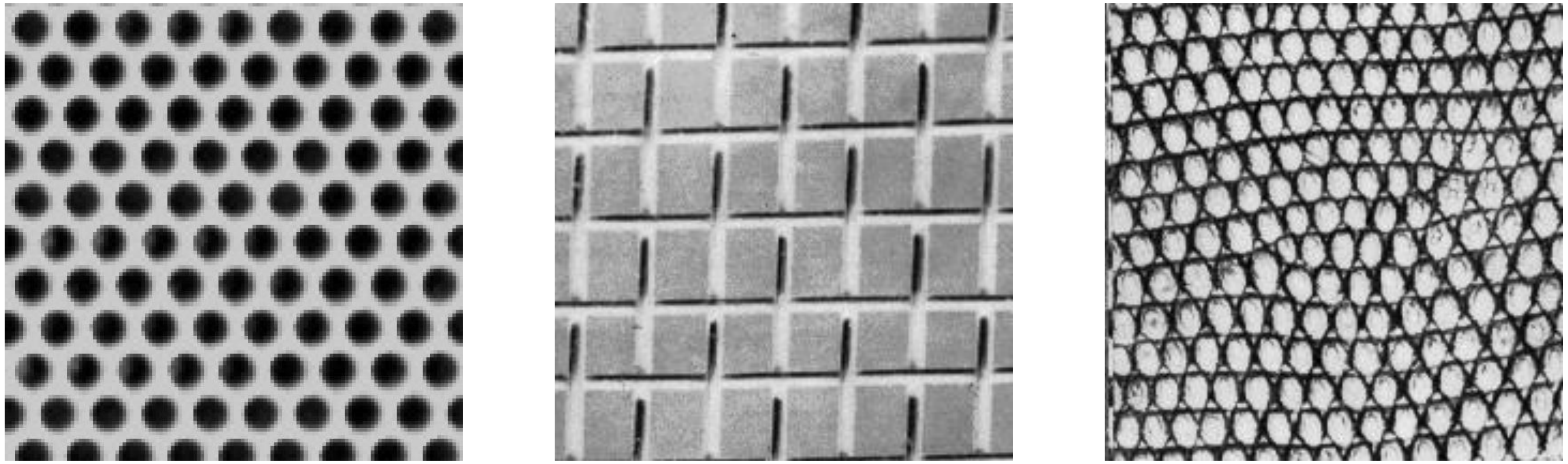
The idea originated from: **Texture Recognition**

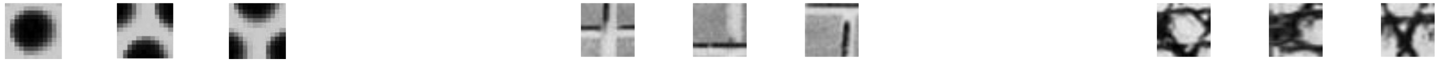


Example textures (from Wikipedia)

The idea originated from: Texture Recognition

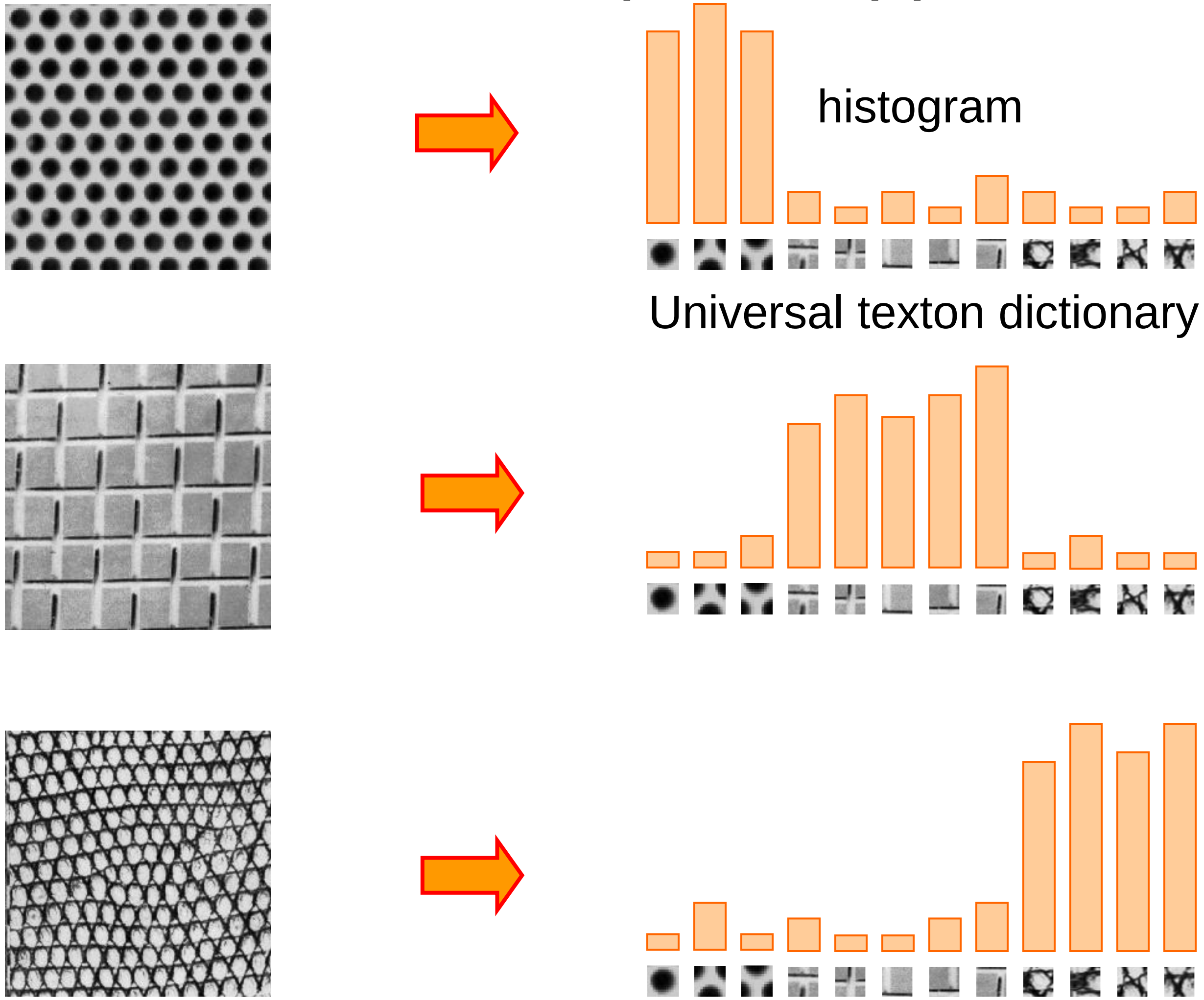
- Texture is characterized by the repetition of certain patches



Vocabulary: 

Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Every image is represented as fixed sized histogram of the number of times a patch appears



A similar idea is also used in natural language processing and called: Bag-of-words models

- Every word document is represented as the frequencies of words from a fixed vocabulary Salton & McGill (1983)

Visual bag of words for object recognition

face, flowers, building

- Works pretty well for recognition and for enabling image retrieval

Csurka et al. (2004), Willamowski et al. (2005), Grauman & Darrell (2005), Sivic et al. (2003, 2005)

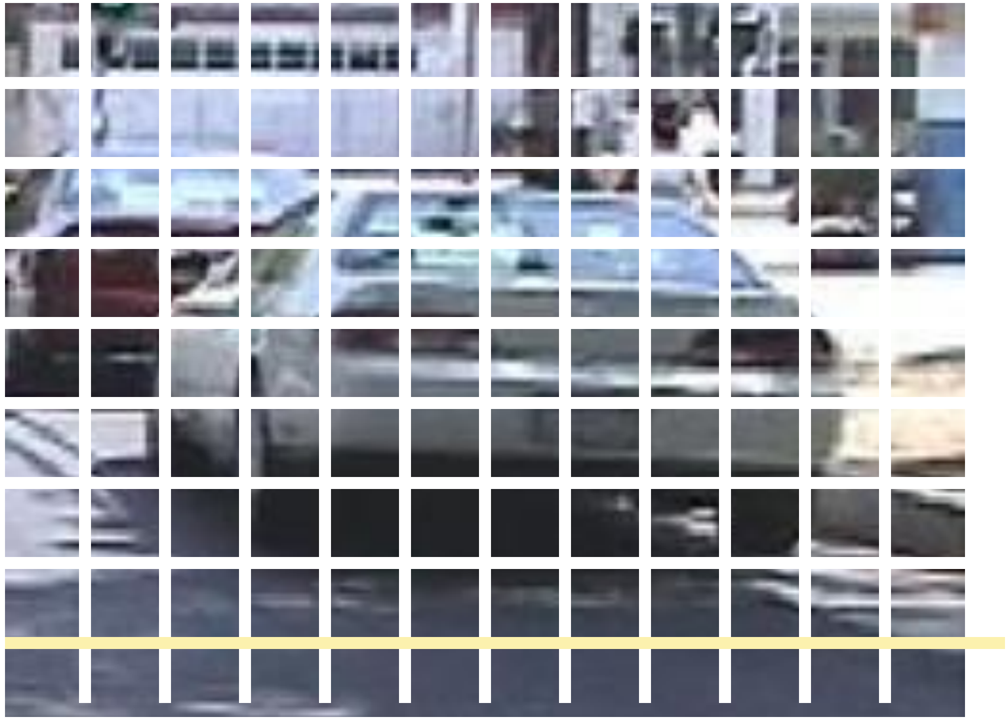
Bag of features

- First, take a bunch of images, extract features, and build up a “visual vocabulary” – a list of common features

- Given a new image, extract features and build a histogram of visual bag of words
  - for each patch in the image, find the closest visual word in the vocabulary and increment its corresponding value in the histogram

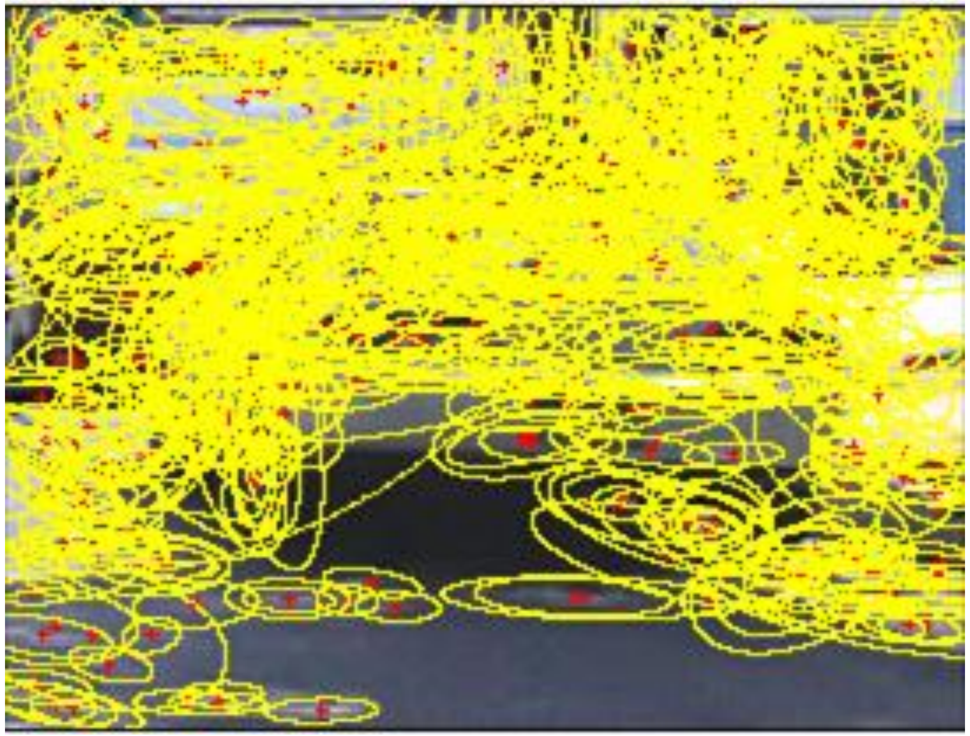
Step 1. Choose patches in a training dataset of images

- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005



Step 1. Choose patches in a training dataset of images

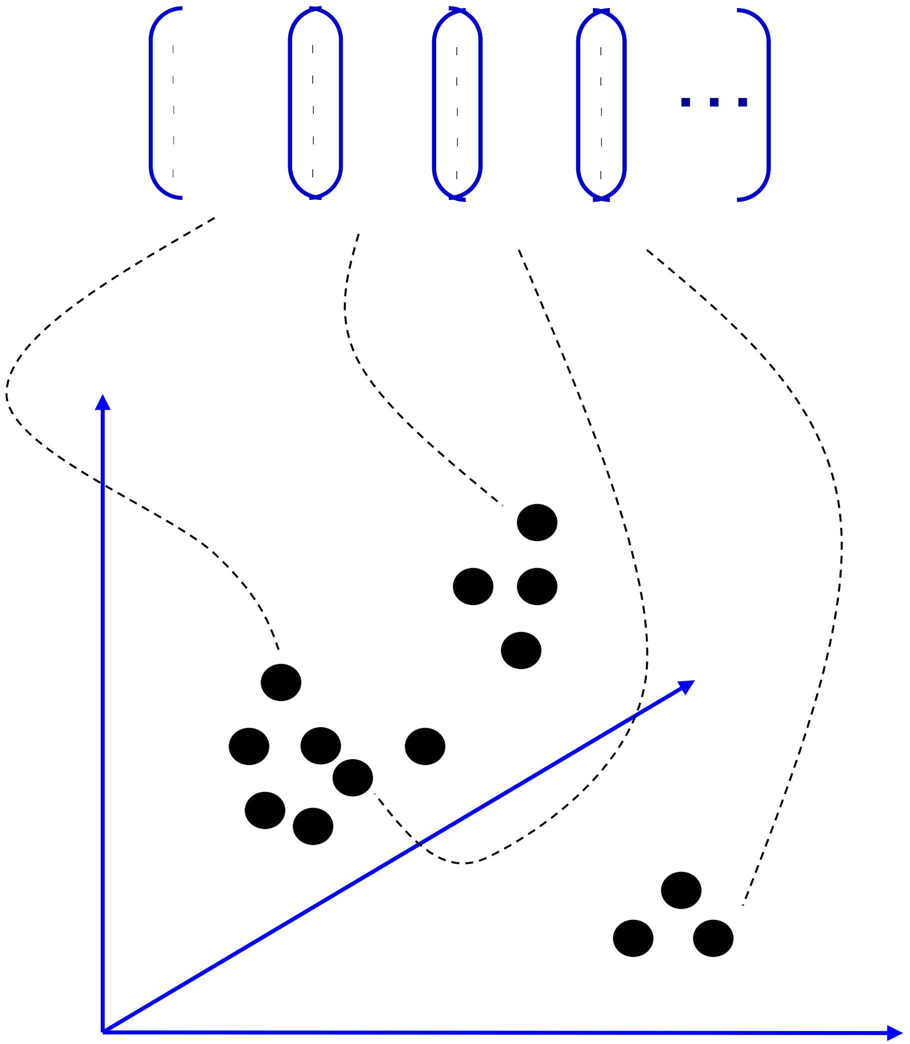
- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005



Step 1. Choose patches in a training dataset of images

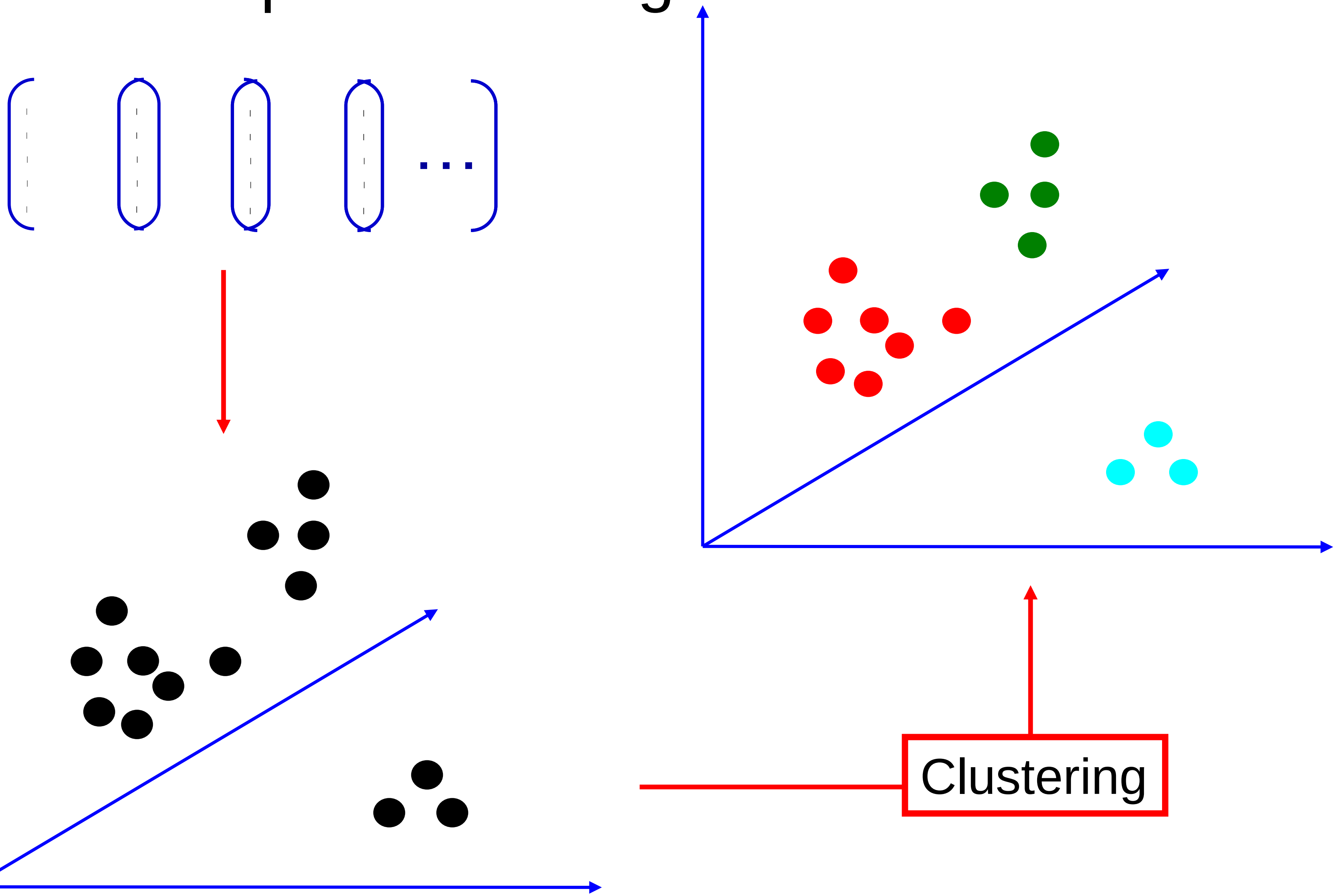
- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005
- Other methods
  - Random sampling (Vidal-Naquet & Ullman, 2002)
  - Segmentation-based patches (Barnard et al. 2003)

Step 2. Cluster the patches using k-means

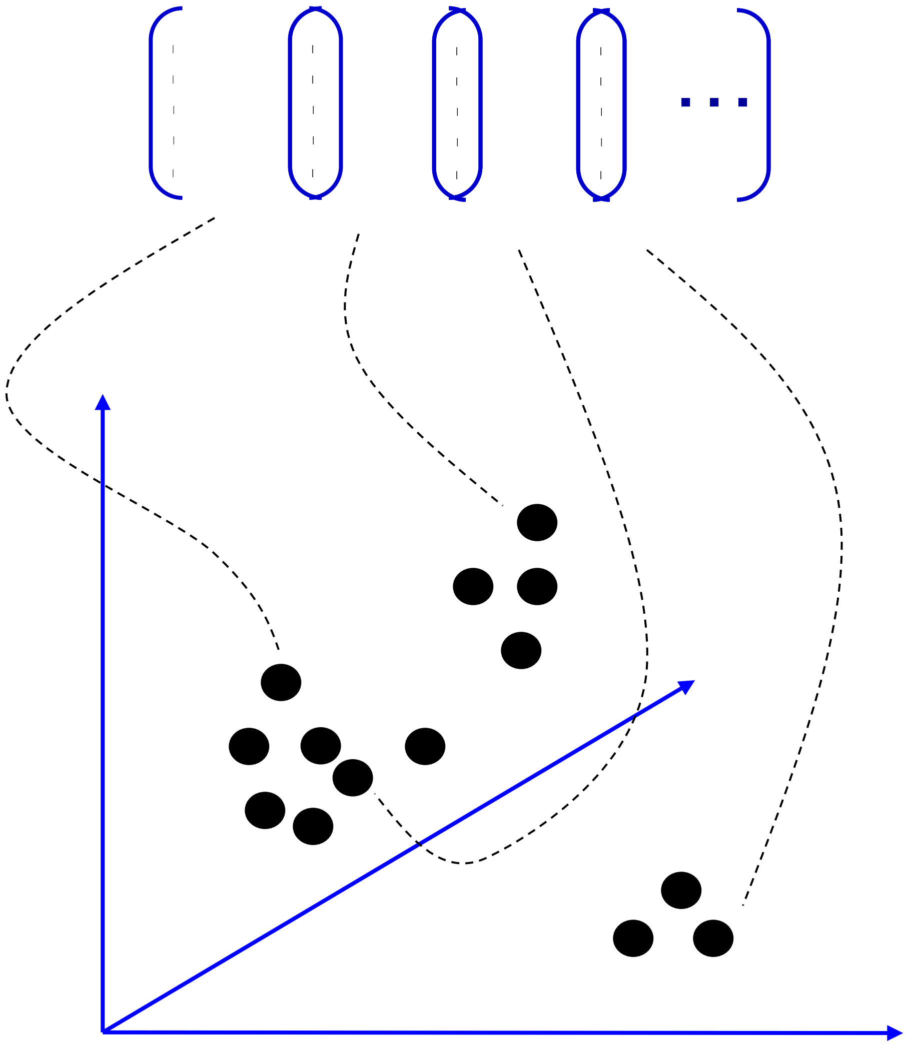


The k in k-means is the size of the vocabulary. It will determine the size of the features

Step 2. Cluster the patches using k-means

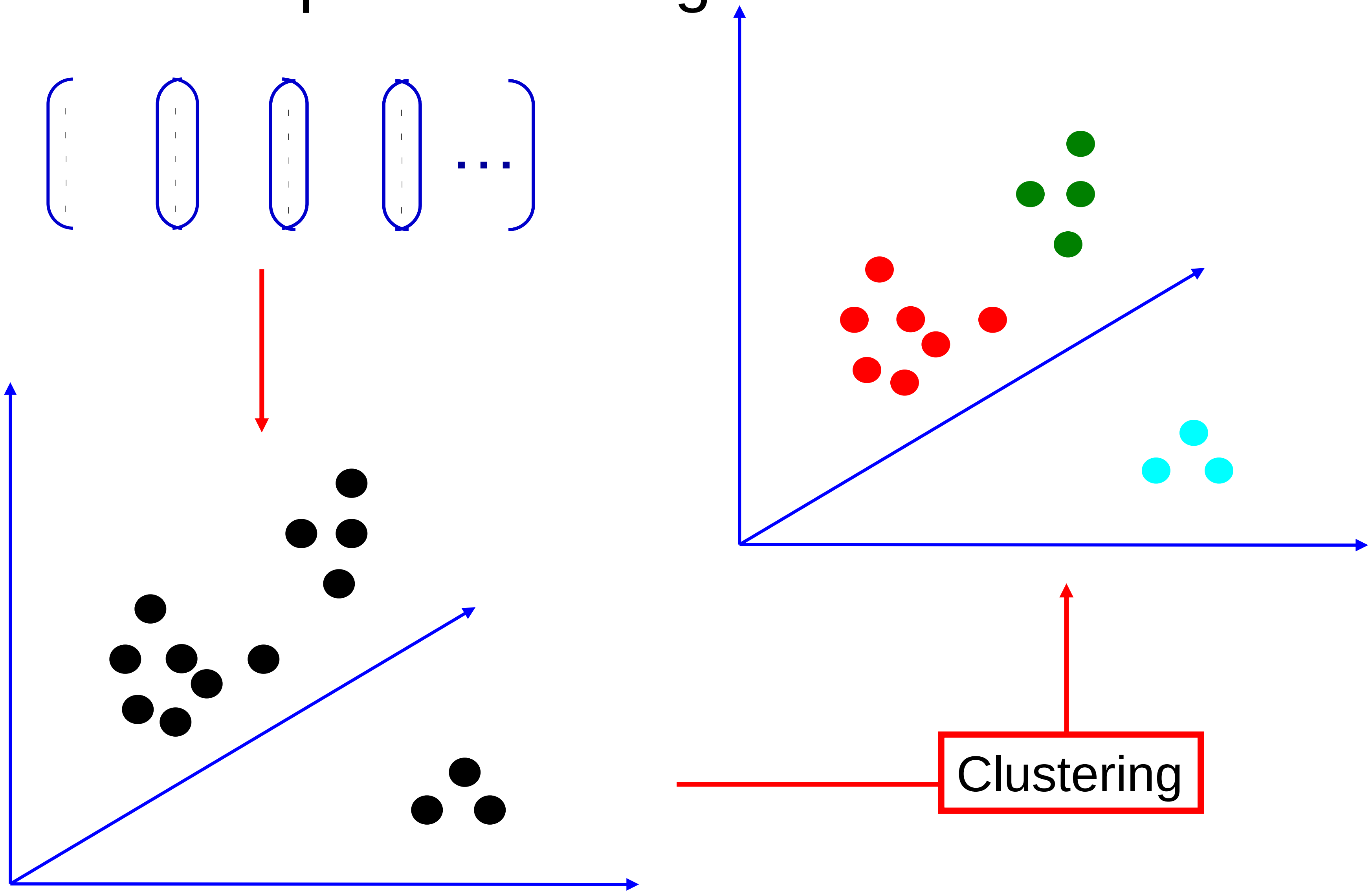


Step 2. Cluster the patches using k-means

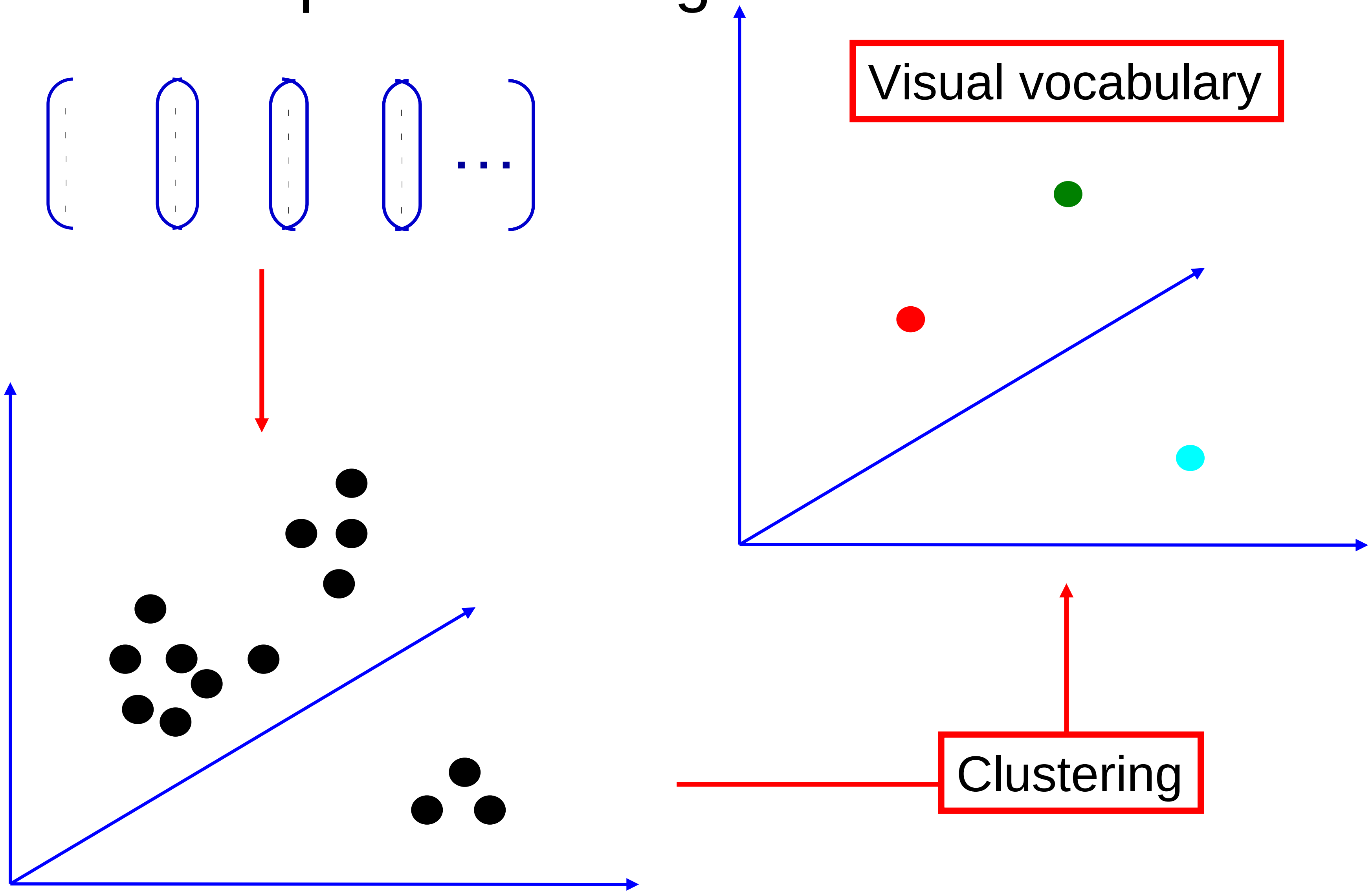


The k in k-means is the size of the vocabulary. It will determine the size of the bag of features

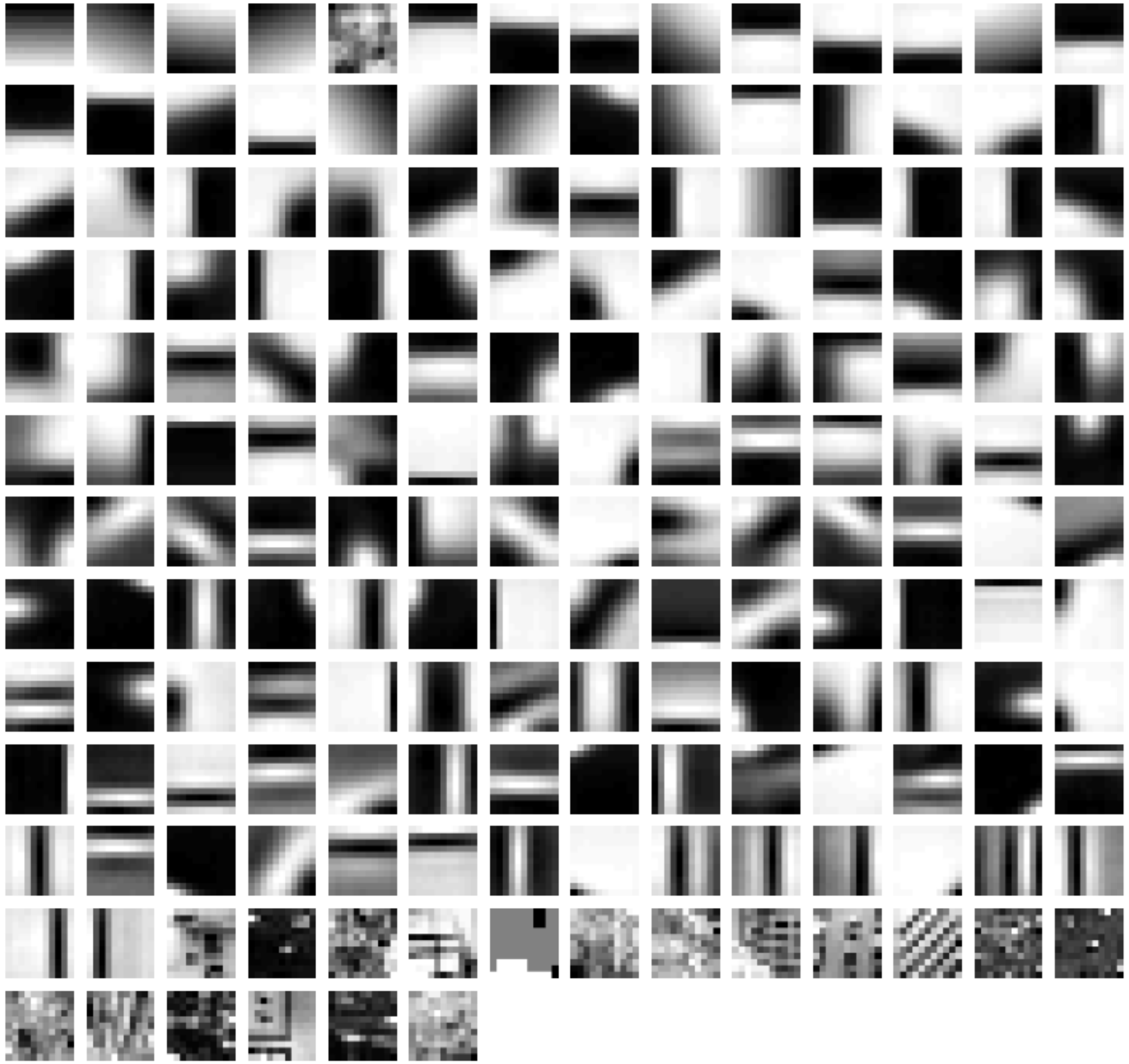
Step 2. Cluster the patches using k-means



Step 2. Cluster the patches using k-means

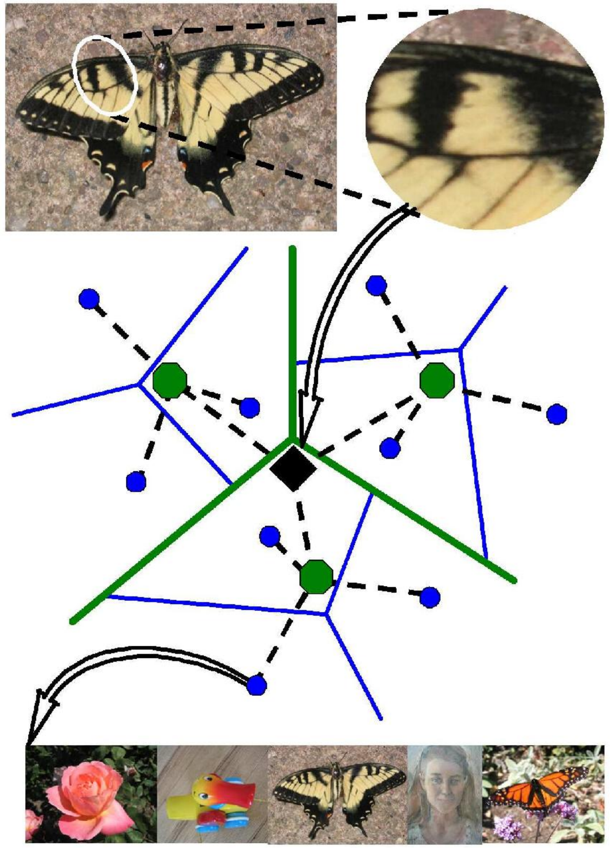


Example visual vocabulary



Visual vocabularies: Issues

- How to choose vocabulary size?
  - **Too small**: Most patches are just noisy and not useful
  - **Too large**: overfits to training images and doesn't generalize
- **Computational efficiency**
  - Try to choose as small of a vocabulary size as possible to reduce curse of dimensionality



Step 3. Convert every image into a histogram

- Every image now becomes a k-dimensional histogram representation.
- We can use these features for any recognition task.

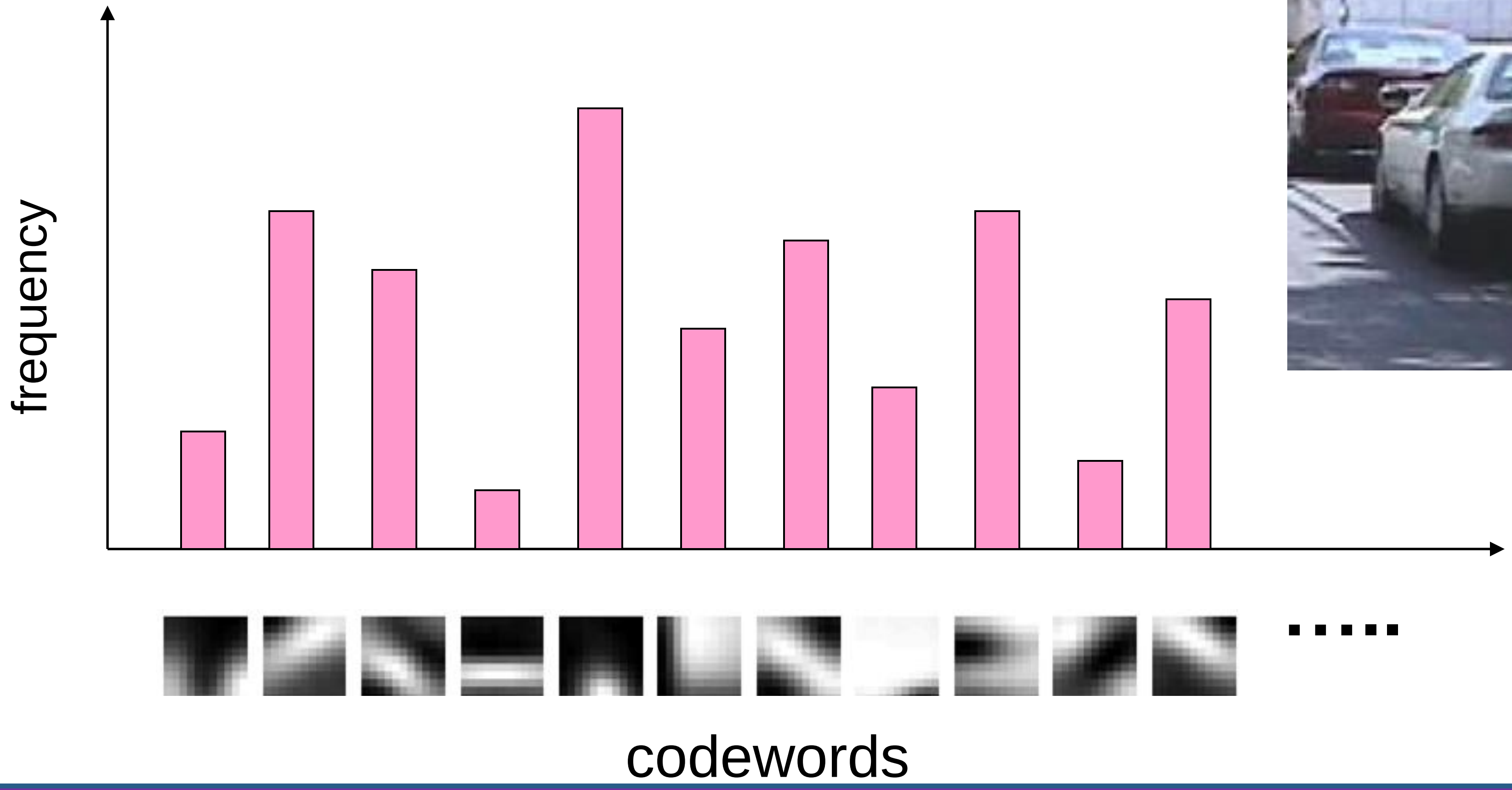
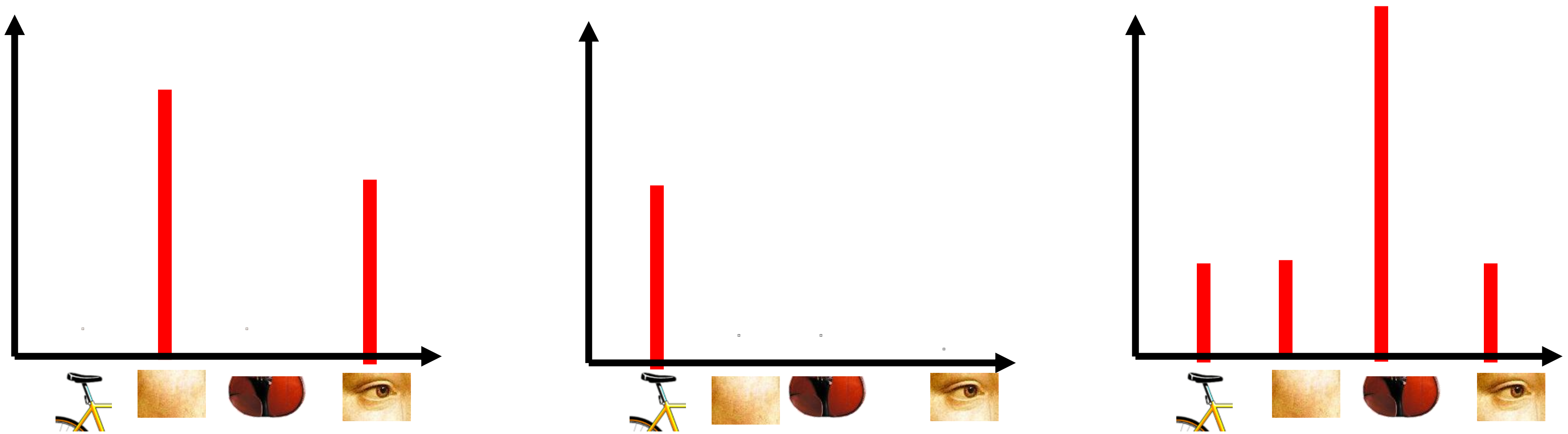


Image classification

- A histogram of bag-of-words features are very good at distinguishing between different categories.
- E.g., first image is a face, second is a bike, third is an instrument



Uses of BoW representation

- Treat as feature vector for standard classifier
  - e.g k-nearest neighbors

Visual bag of words works quite well for a fixed set of categories






class	bag of features Zhang et al. (2005)	bag of features Willamowski et al. (2004)	Parts-and-shape model Fergus et al. (2003)
airplanes	**98.8**	97.1	90.2
cars (rear)	98.3	**98.6**	90.3
cars (side)	**95.0**	87.3	88.5
faces	**100**	99.3	96.4
motorbikes	**98.5**	98.0	92.5
spotted cats	**97.0**	—	90.0

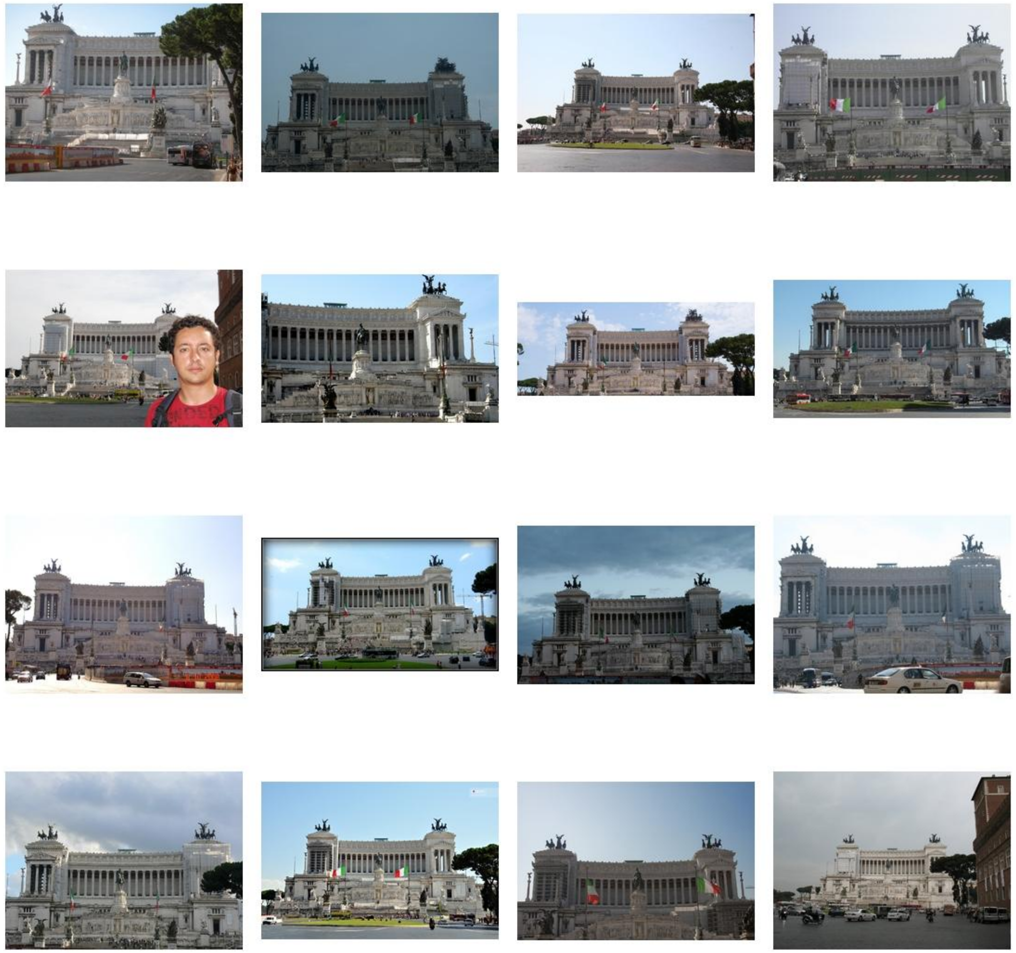
Bag of words can also enable search

query image

top 6 results

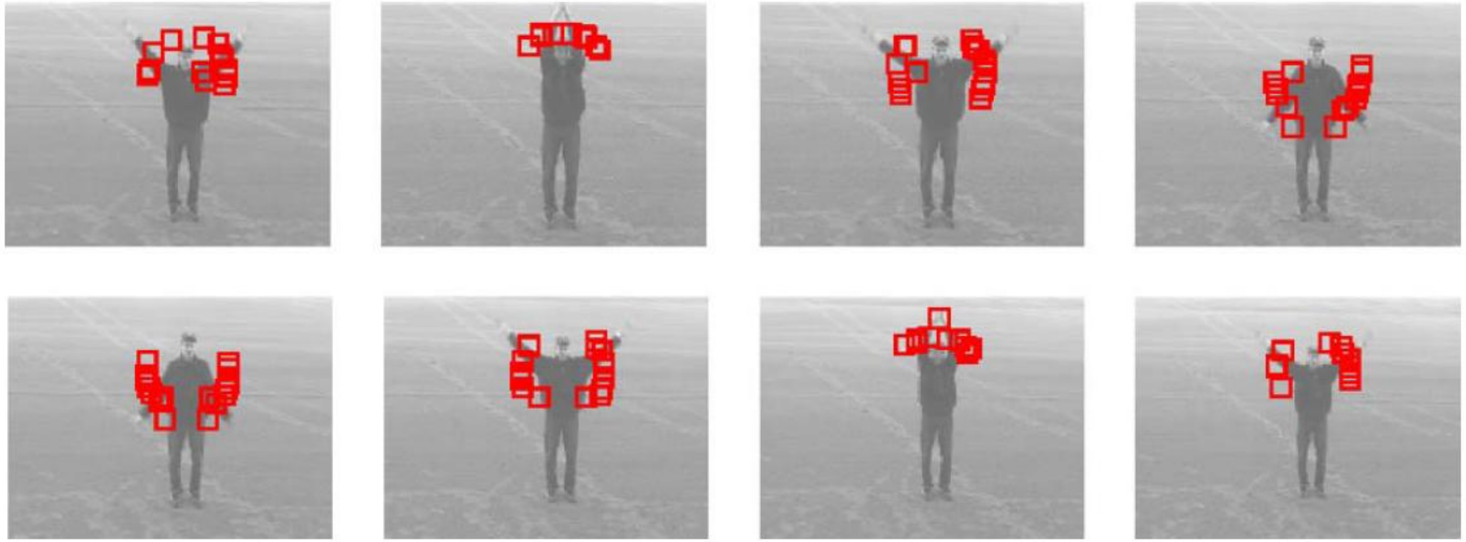
- Cons:
    - performance degrades as the database grows

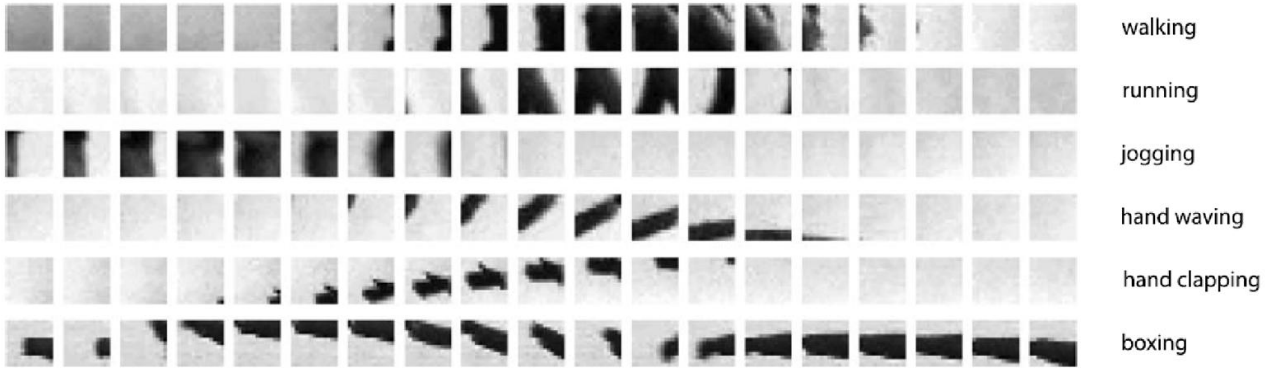
Example bag-of-words matches



Example bag-of-words matches

Bags of words in videos





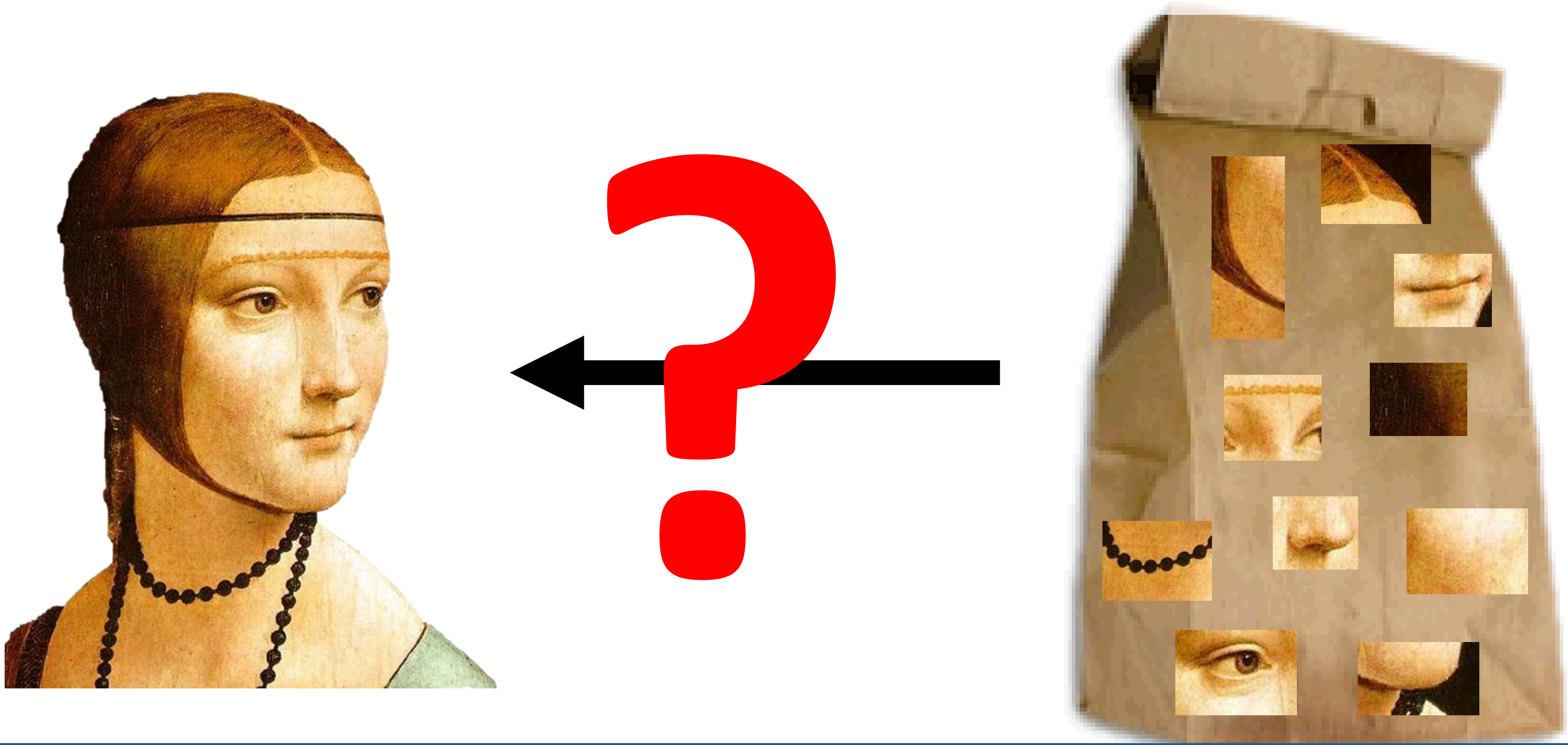
Juan Carlos Niebles, Hongcheng Wang and Li Fei-Fei, **Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words**, IJCV 2008.

Today’s agenda

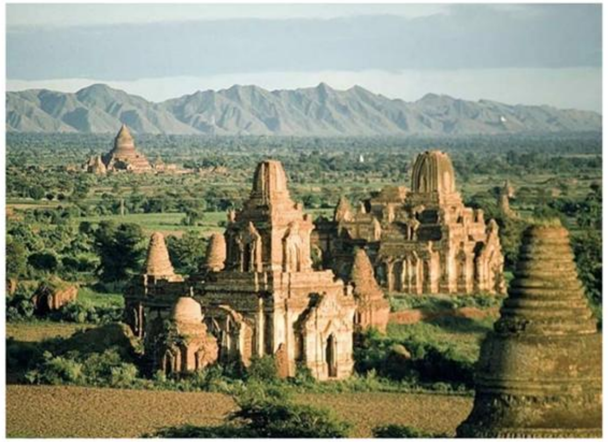
- Spatial pyramids
- Object detection
  - Task and evaluation
- A simple detector
- Deformable parts model

How do we choose the size of the patches?

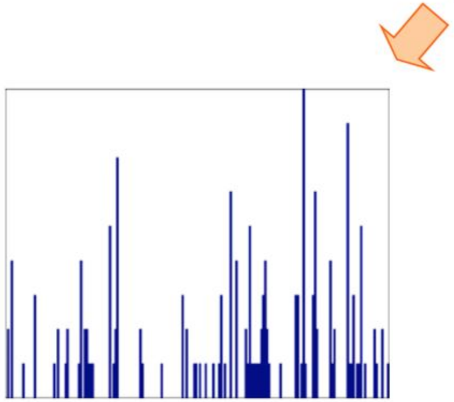
- If the object is close to the camera, larger patches are better
- If the object is really far away, smaller patches are better for finding it.



Bag of words + pyramids

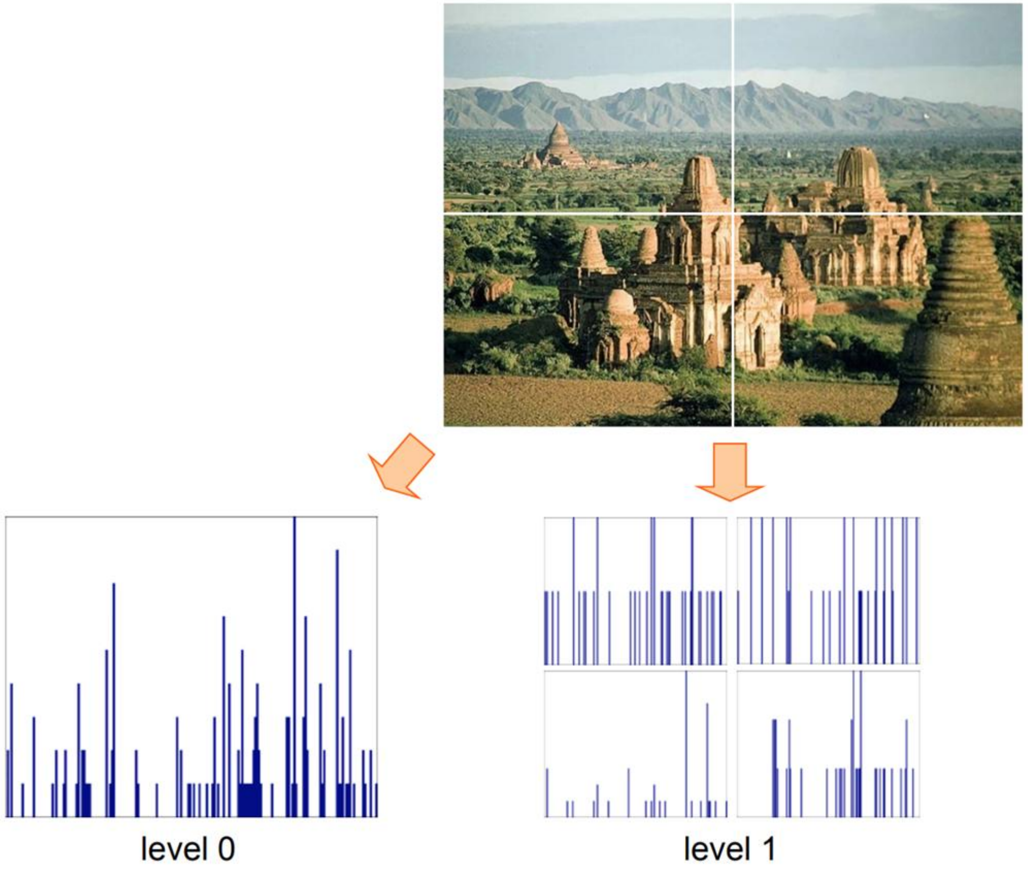


Locally orderless representation at several levels of spatial resolution



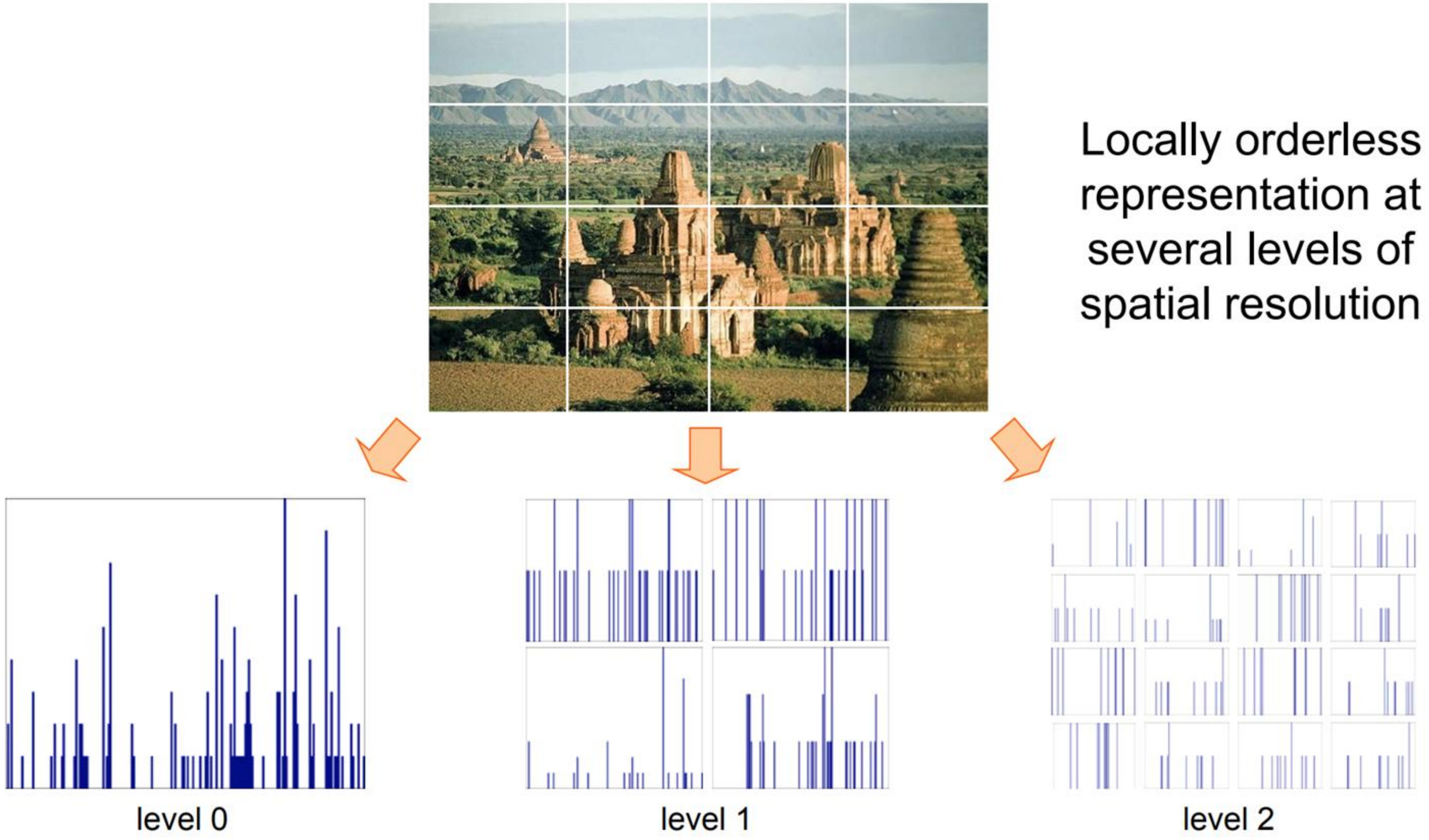
level 0

Bag of words + pyramids



Locally orderless representation at several levels of spatial resolution

Bag of words + pyramids

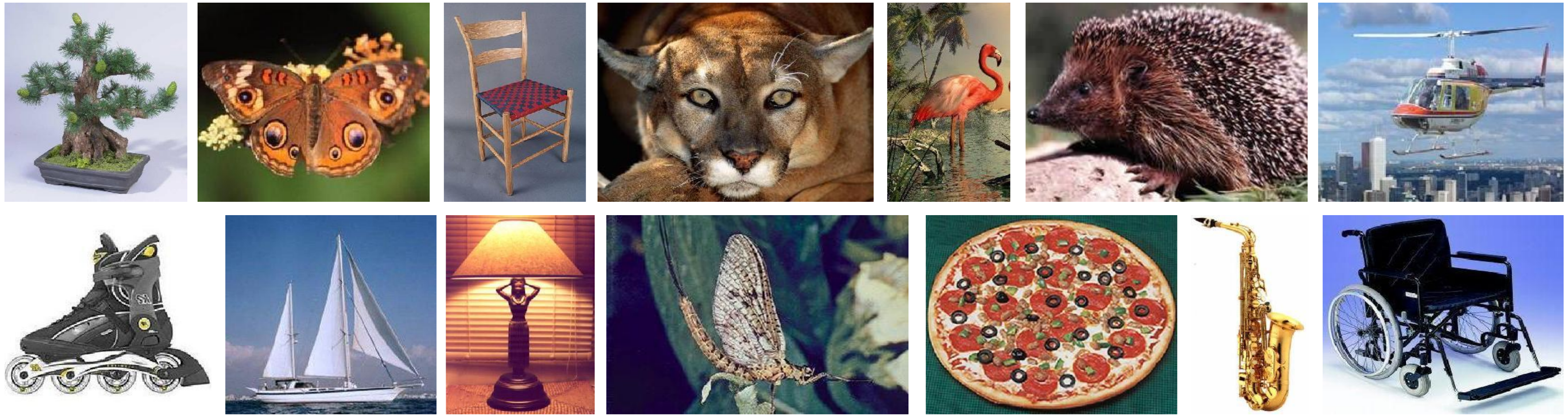


Pyramids are a general idea that is used in all vision models today

- Very useful for representing images.
- Pyramid is built by using multiple copies of image.
- Each level in the pyramid is 1/4 of the size of previous level.

Caltech101 dataset

Multi-class classification results (30 training images per class)



Level	Single-level	Pyramid	Single-level	Pyramid
0	15.5 ±0.9		41.2 ±1.2	
1	31.4 ±1.2	32.8 ±1.3	55.9 ±0.9	57.0 ±0.8
2	47.2 ±1.1	49.3 ±1.4	63.6 ±0.9	**64.6** ±0.8
3	52.2 ±0.8	**54.0** ±1.1	60.3 ±0.9	64.6 ±0.7

Today's agenda

- Spatial pyramids
- Object detection
  - Task and evaluation
- A simple detector
- Deformable parts model

Object Detection

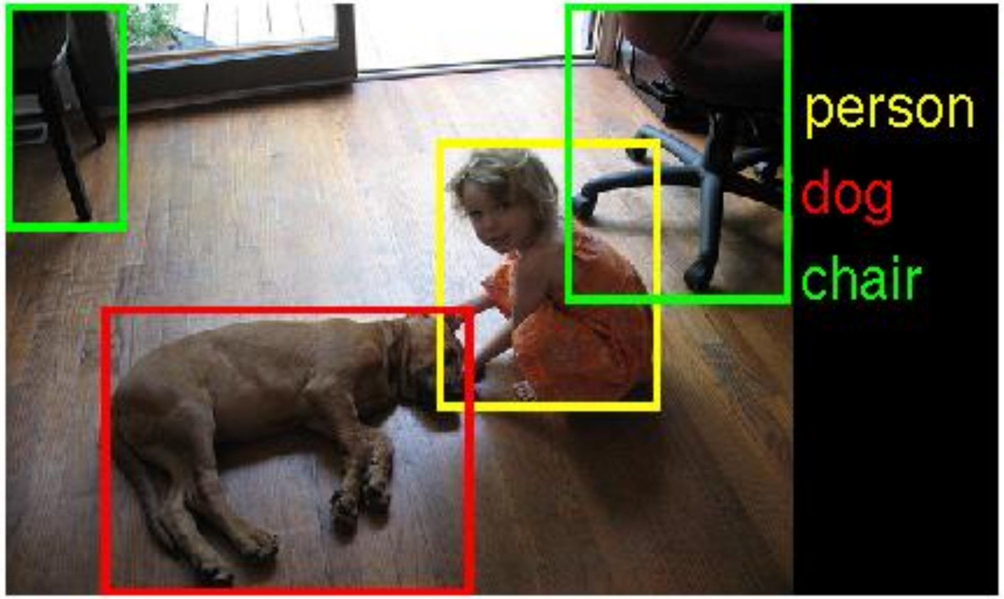
Credit: Flickr user neilalderney123

- What do you see in the image?

Object Detection

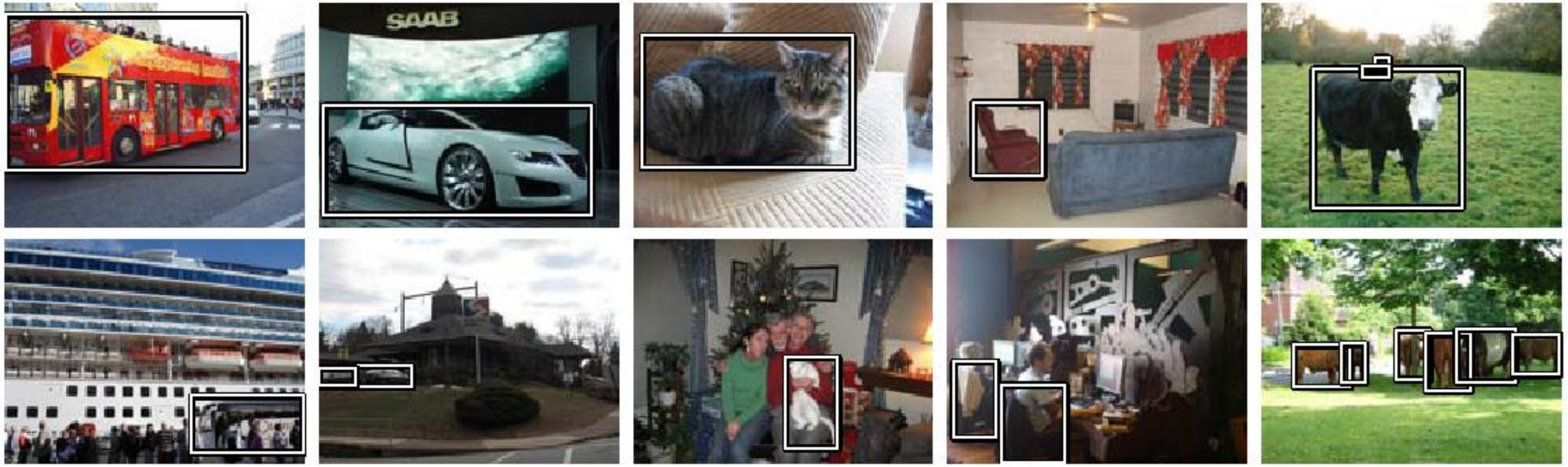
- **Problem**: Detecting and localizing generic objects from various categories, such as cars, people, etc.

- Challenges:
  - Illumination,
  - viewpoint,
  - deformations,
  - Intra-class variability



Object Detection Benchmarks

- PASCAL VOC (Visual Object Classes) Challenge



- 20 categories
- Annual classification, detection, segmentation, … challenges

Object Detection Benchmarks

- PASCAL VOC Challenge
- ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
  - 200 Categories for detection

Object Detection Benchmarks

- PASCAL VOC Challenge
- ImageNet Large Scale Visual Recognition Challenge (ILSVR)
- Common Objects in Context (COCO)
  - 80 Object categories

How do we evaluate object detection?

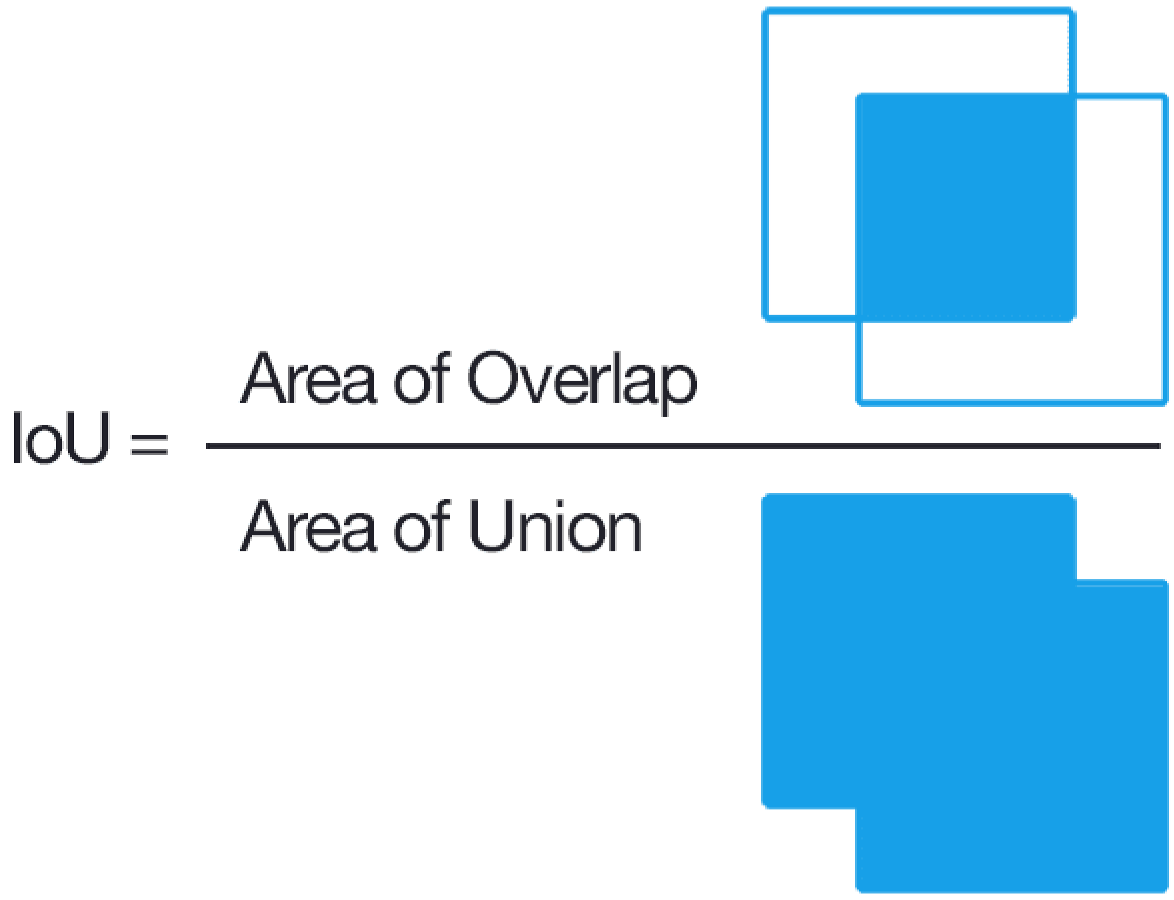
predictions
ground truth

Defining what is a good versus bad detection

IoU is a metric used to decide good from bad predictions.

Given a predicted box and and ground truth box:

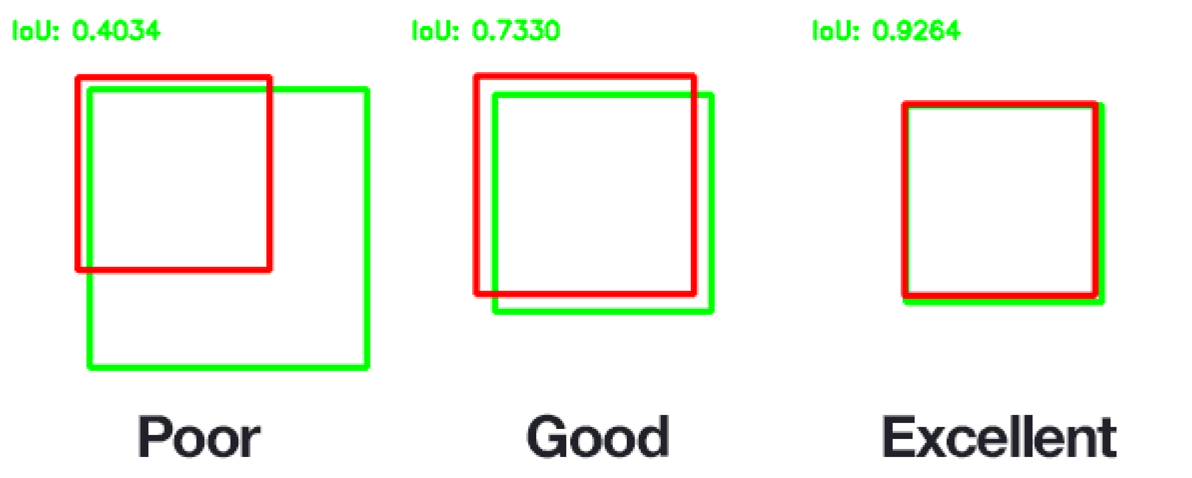
IoU = intersection between the two boxes over (divided by) the union of the two



Defining what is a good versus bad detection

We say a prediction was good if it has IoU > 0.5 with any of the ground truth boxes

0.5 is a threshold that is generally accepted as a good heuristic.



How do we evaluate object detection?

predictions

ground truth

True positive:
- The overlap of the prediction with the ground truth is MORE than 0.5

How do we evaluate object detection?

predictions

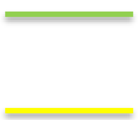
ground truth

True positive:

False positive:

- The overlap of the prediction with the ground truth is LESS than 0.5

How do we evaluate object detection?



True positive:

False positive:

False negative:

- The objects that our model doesn't find

How do we evaluate object detection?

predictions

ground truth

True positive:
False positive:
False negative:
- The objects that our model doesn't find

What is a True Negative?

	Predicted 1	Predicted 0
True 1	true positive	false negative
True 0	false positive	true negative

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

How do we evaluate object detection?

predictions

ground truth

True positive: 1
False positive: 2
False negative: 1

Q. What is the precision?

How do we evaluate object detection?

predictions

ground truth

True positive: 1
False positive: 2
False negative: 1

Q. What is the precision?

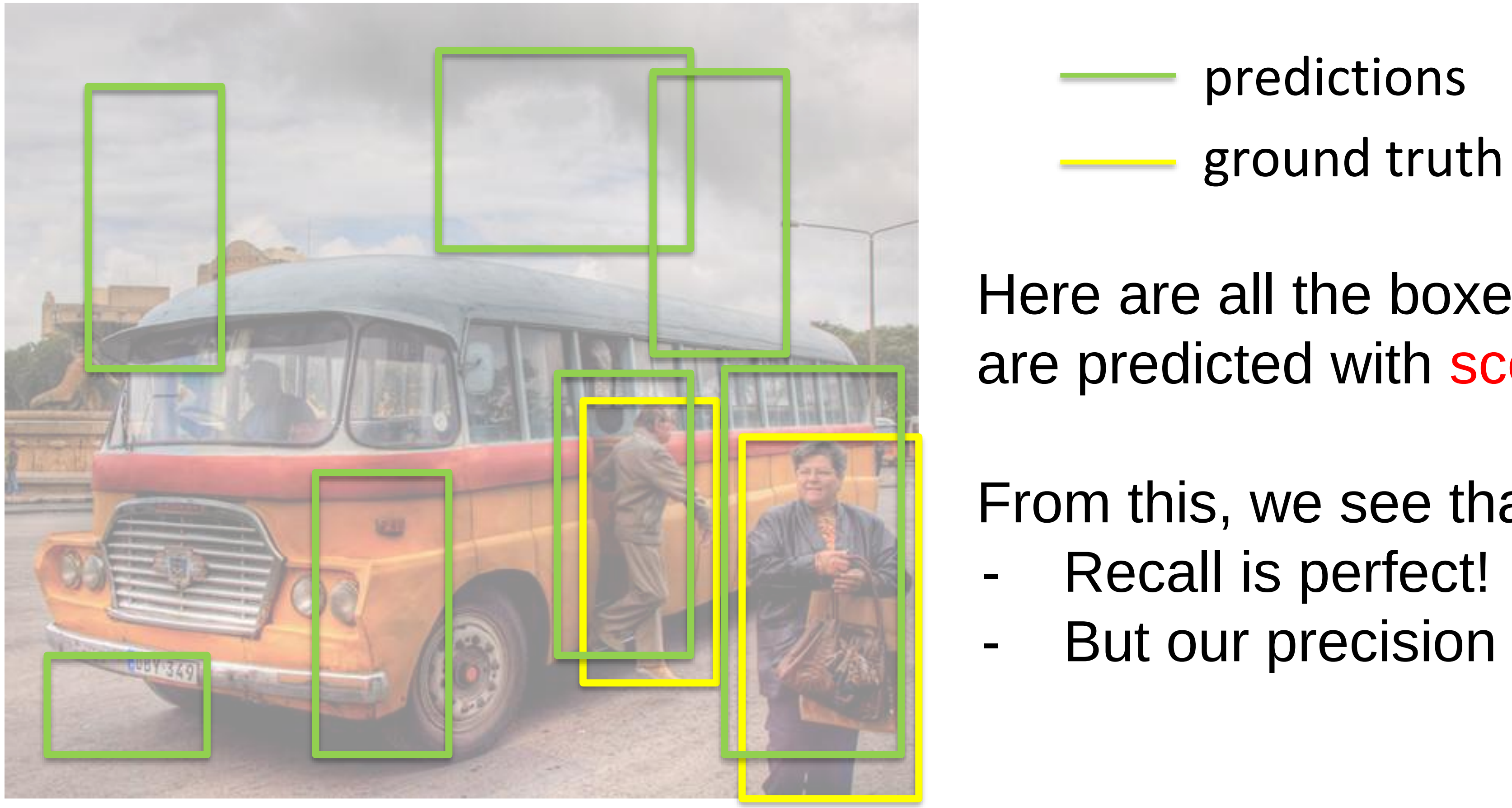
Q. What is the recall?

How to intuitively understand precision versus recall

- Precision:
  - how many of the predicted detections are correct?

- Recall:
  - how many of the ground truth objects are detected?

In reality, our model makes a lot of predictions with varying scores between 0 and 1



Here are all the boxes that are predicted with score > 0.

From this, we see that:

- Recall is perfect!
- But our precision is BAD!

How do we evaluate object detection?

predictions

ground truth

Here are all the boxes that are predicted with score > 0.5

We are using a threshold of 0.5

Q. What happens to precision if threshold is high?

How do we evaluate object detection?

predictions

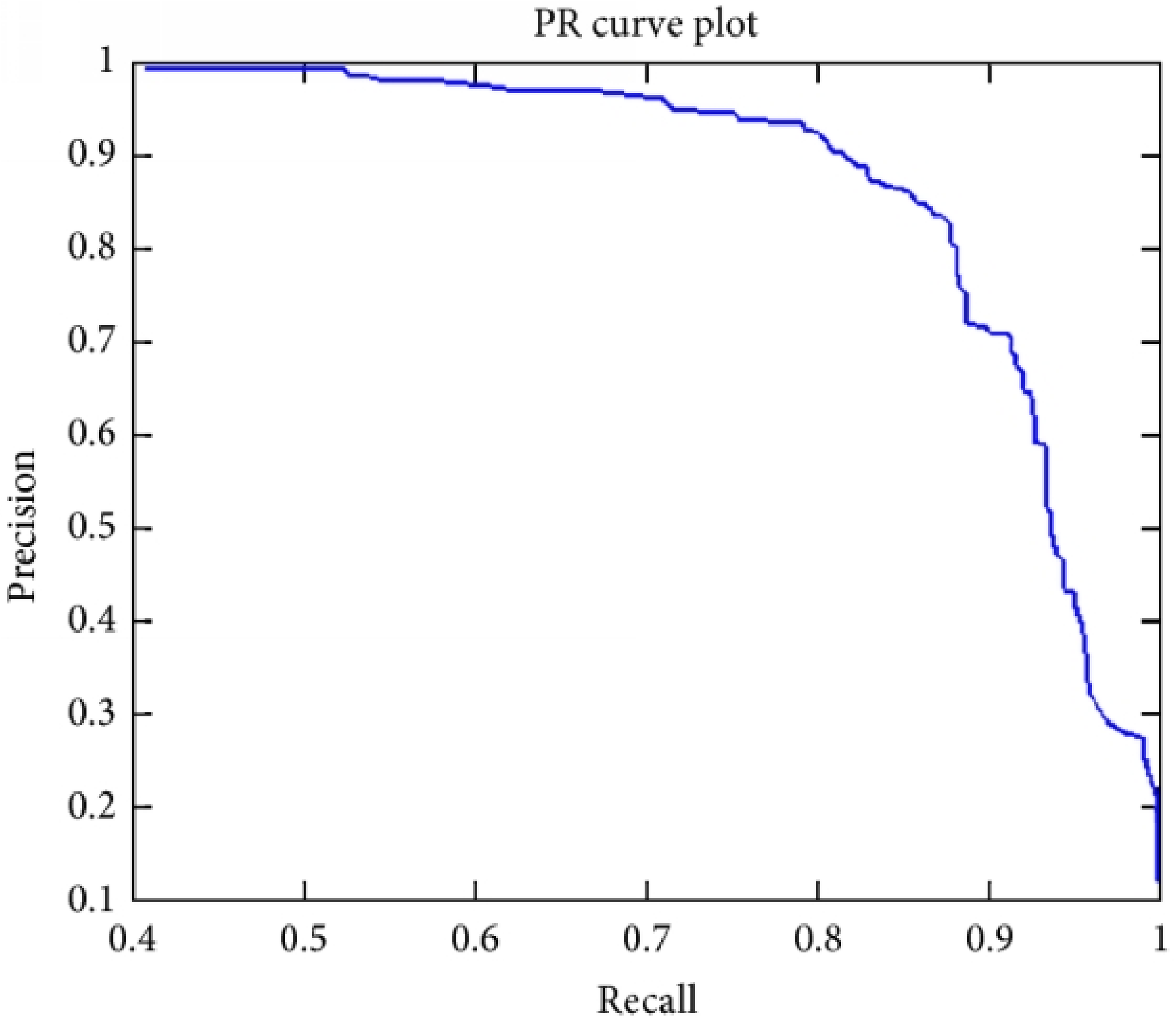
ground truth

Here are all the boxes that are predicted with score > 0.5

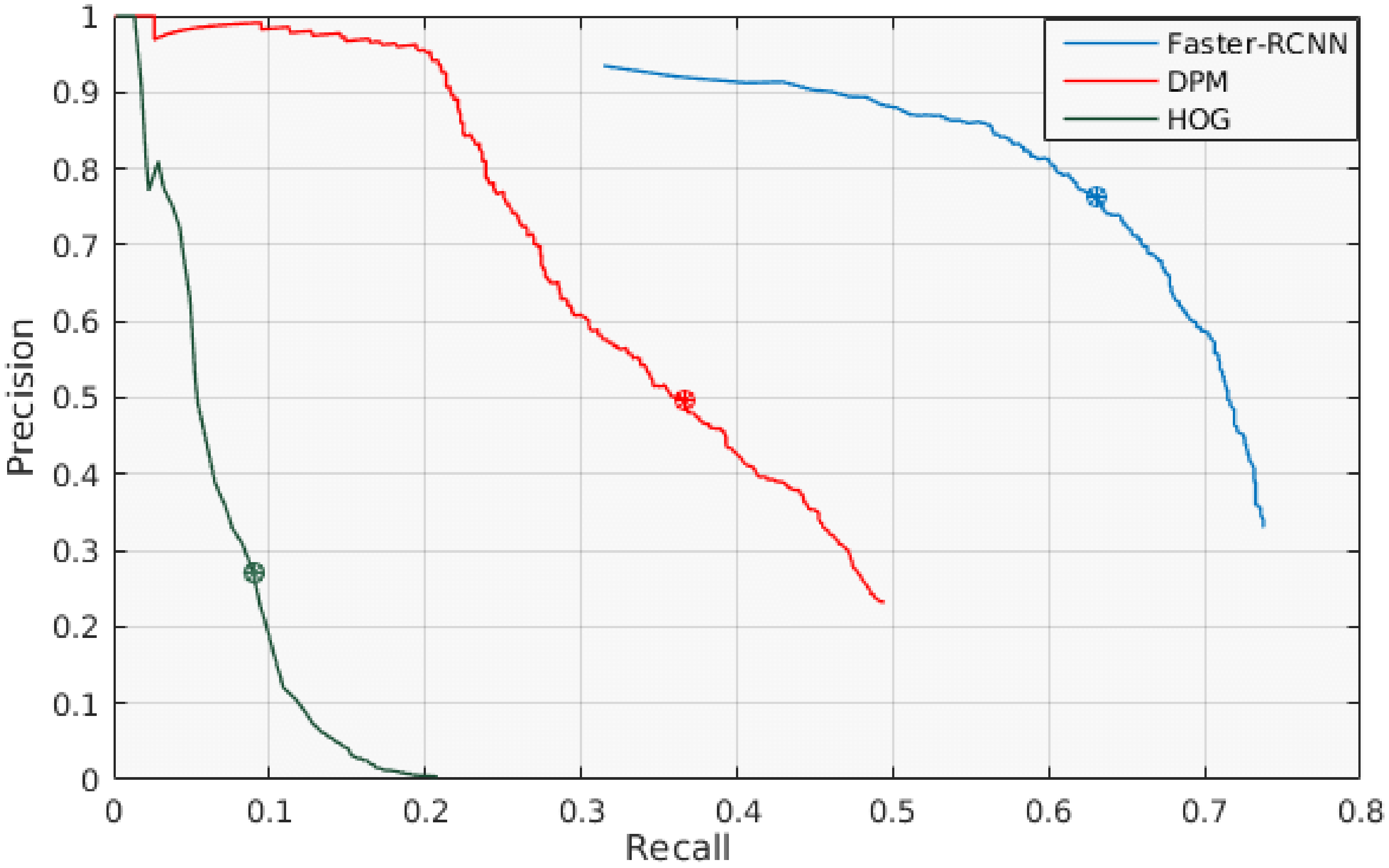
We are using a threshold of 0.5

Q. What happens to recall if threshold is high?

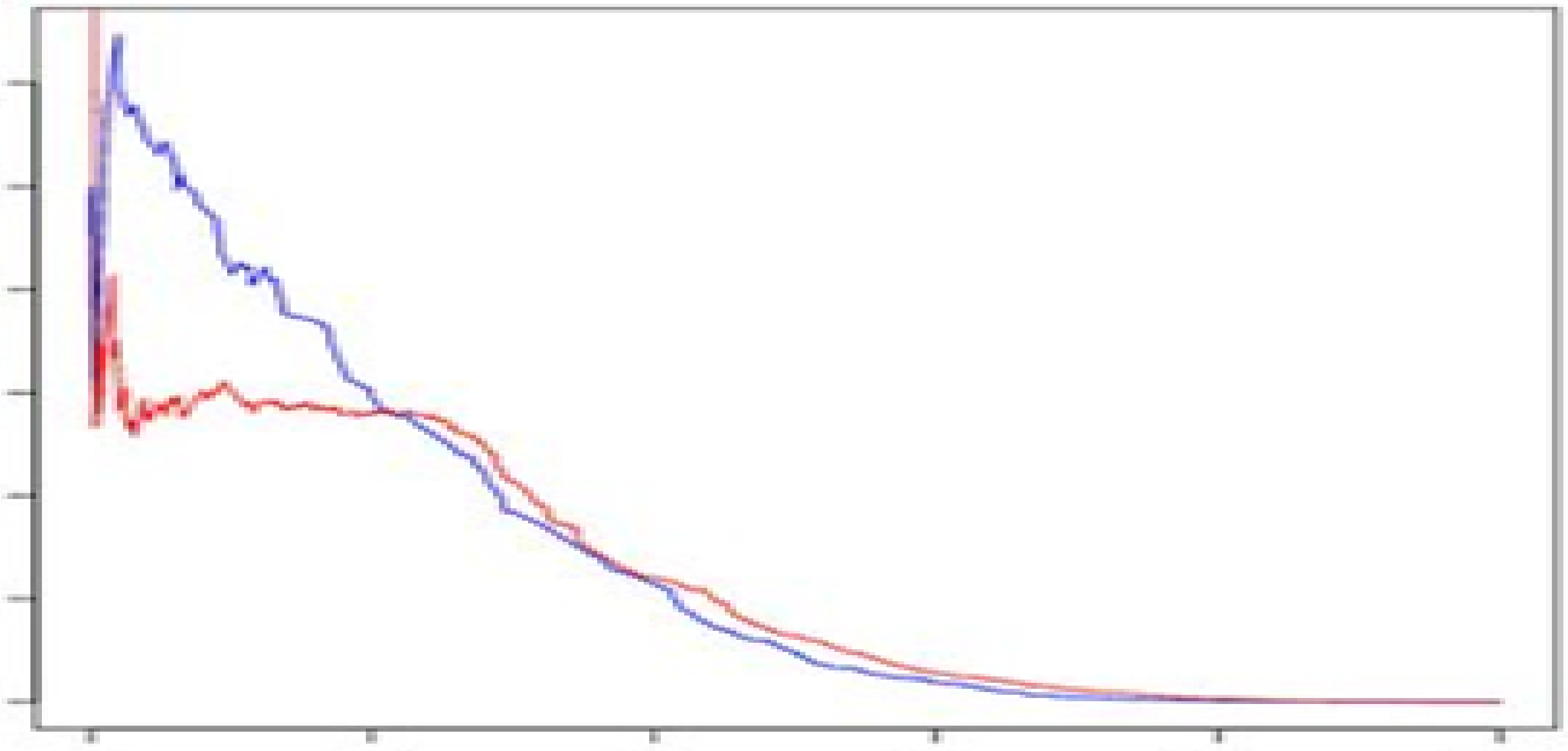
Precision – recall curve (PR curve)



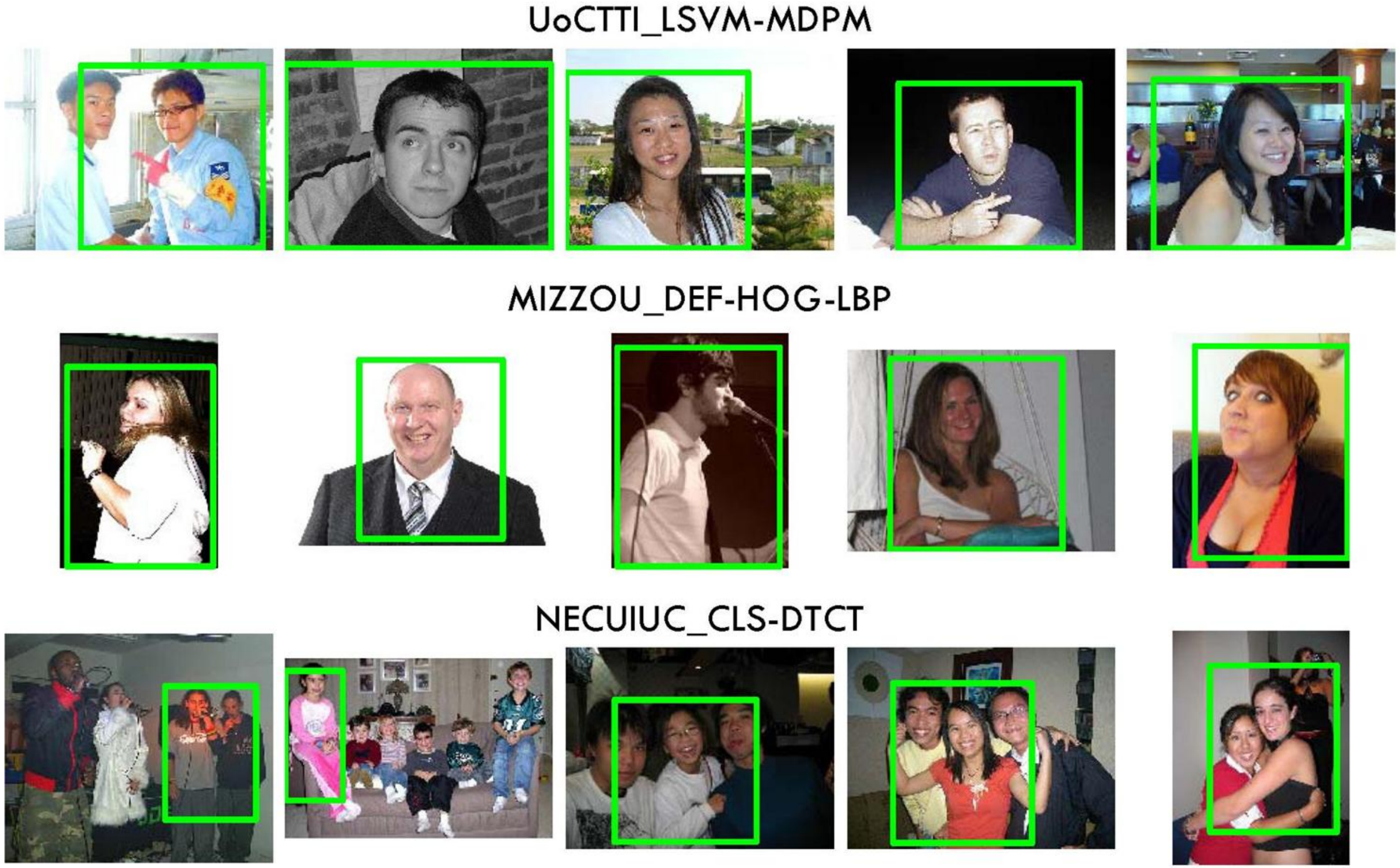
Which model is the best?



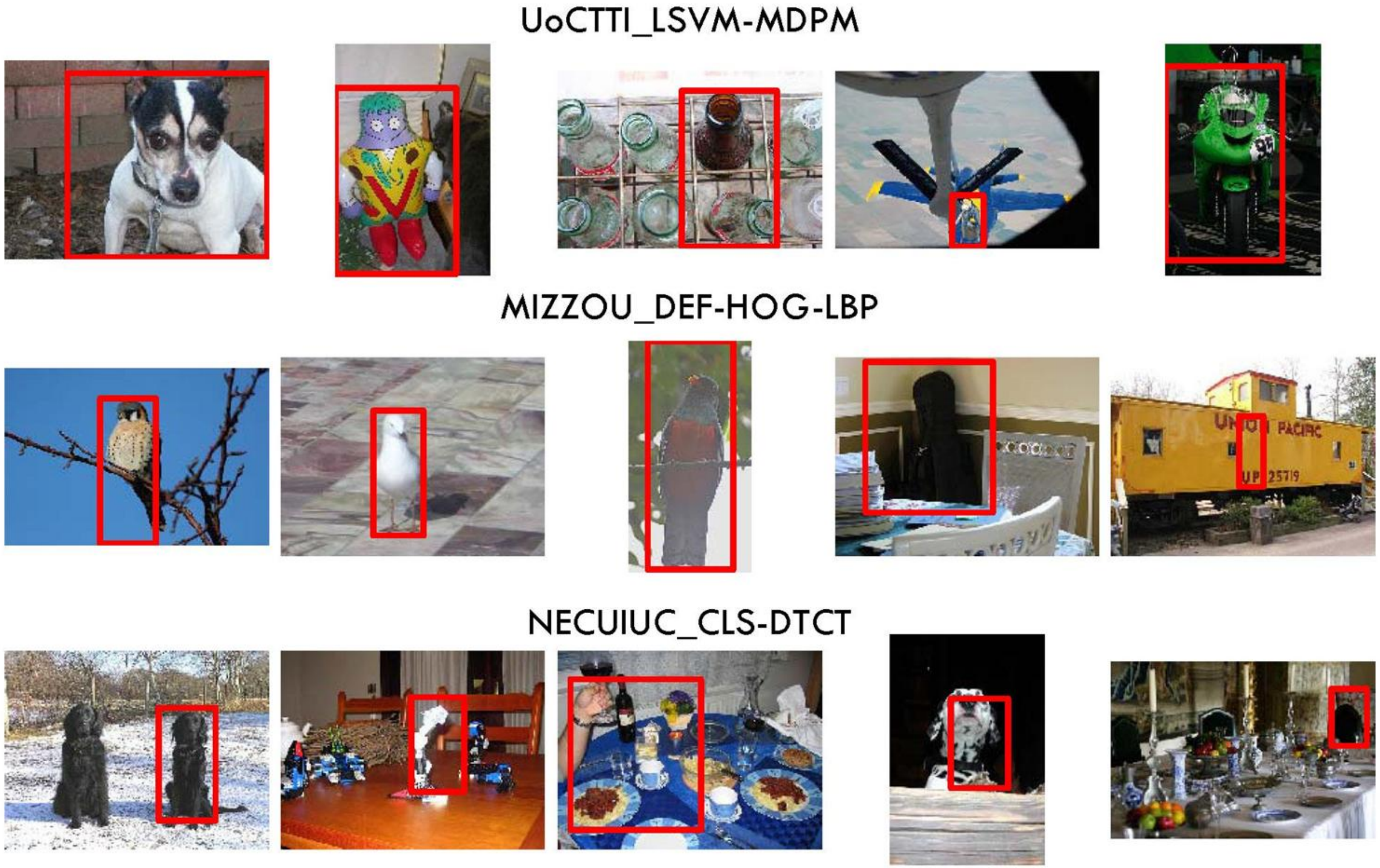
Which model is the best?



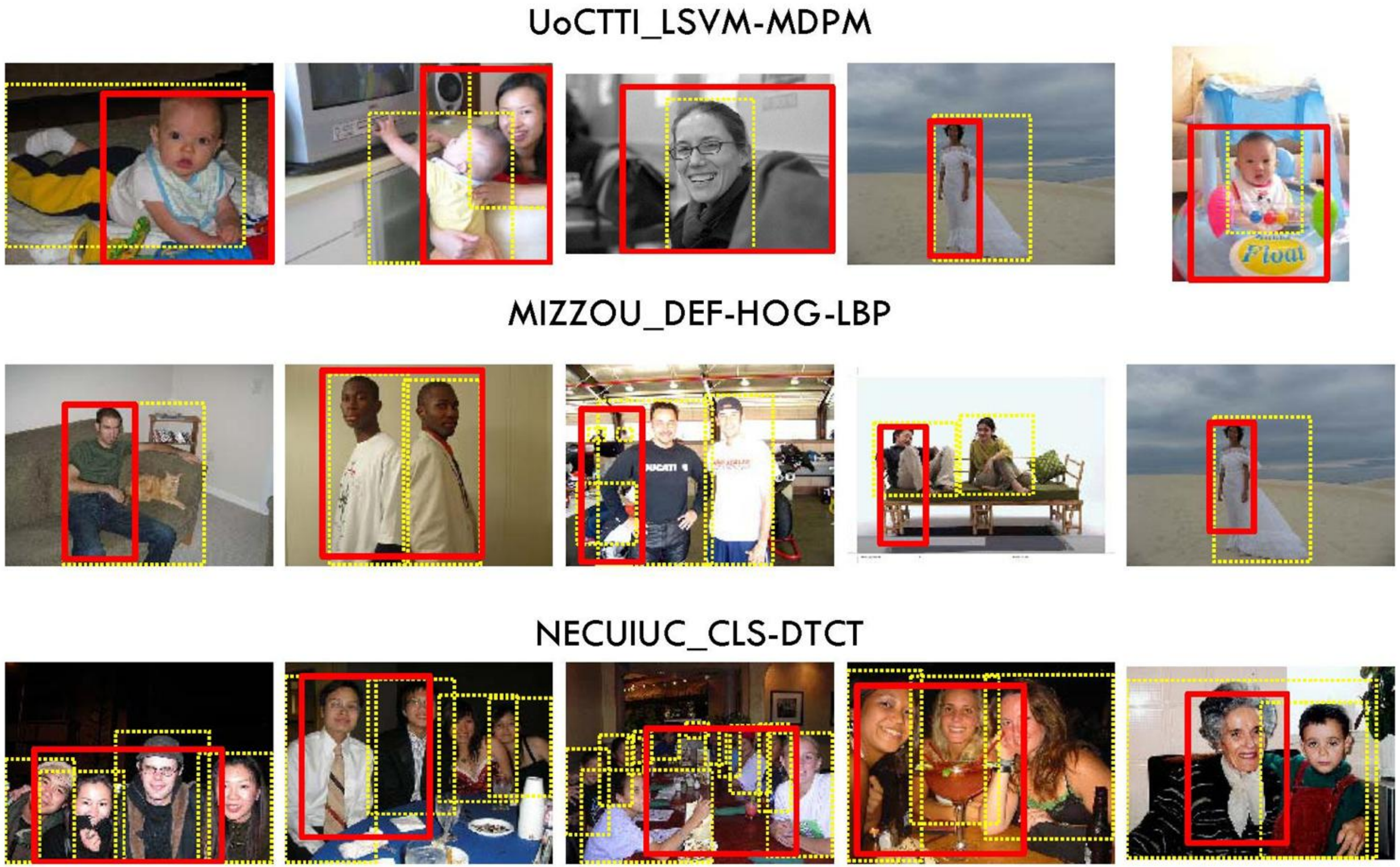
True positives - detecting person



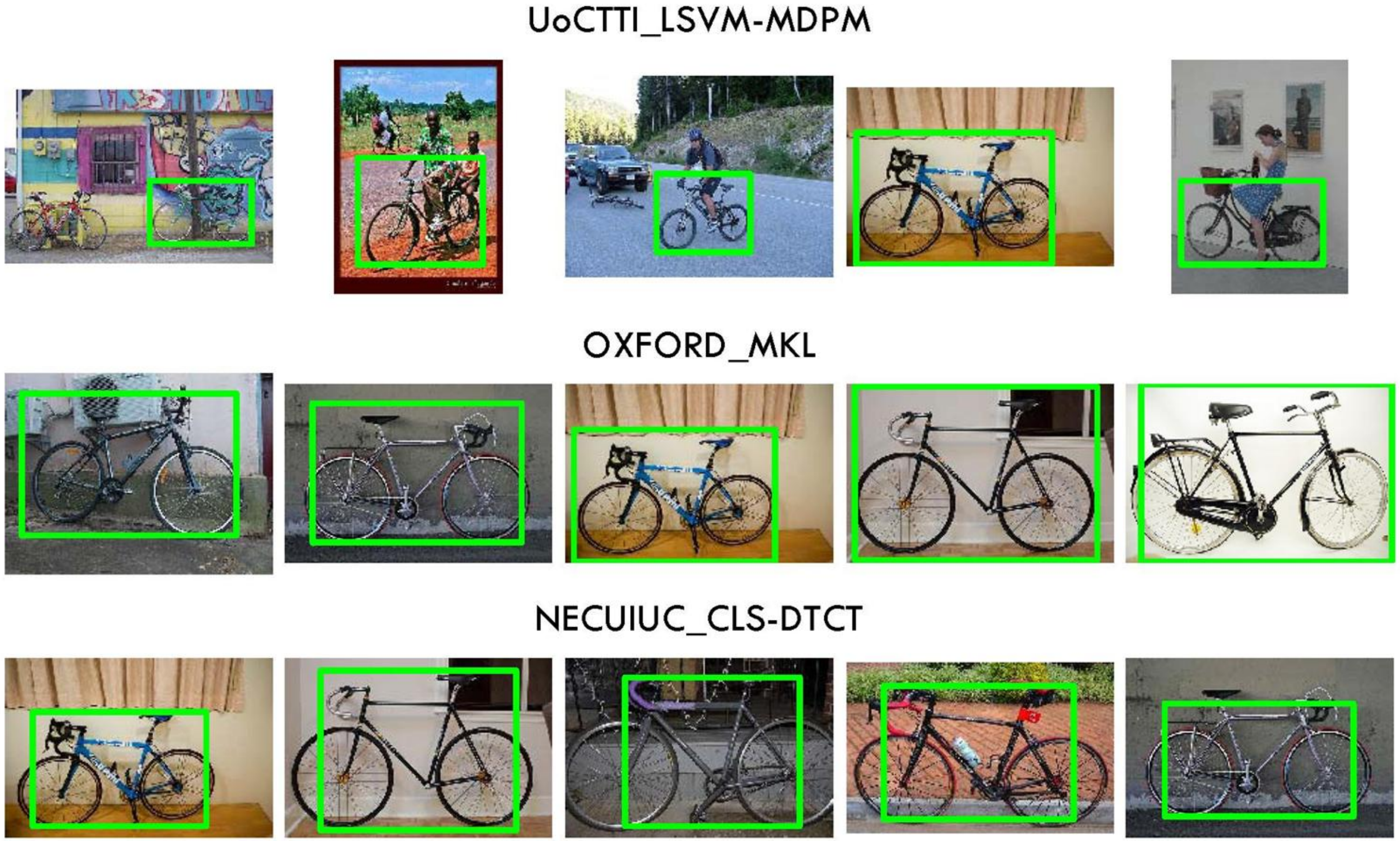
False positives - detecting person



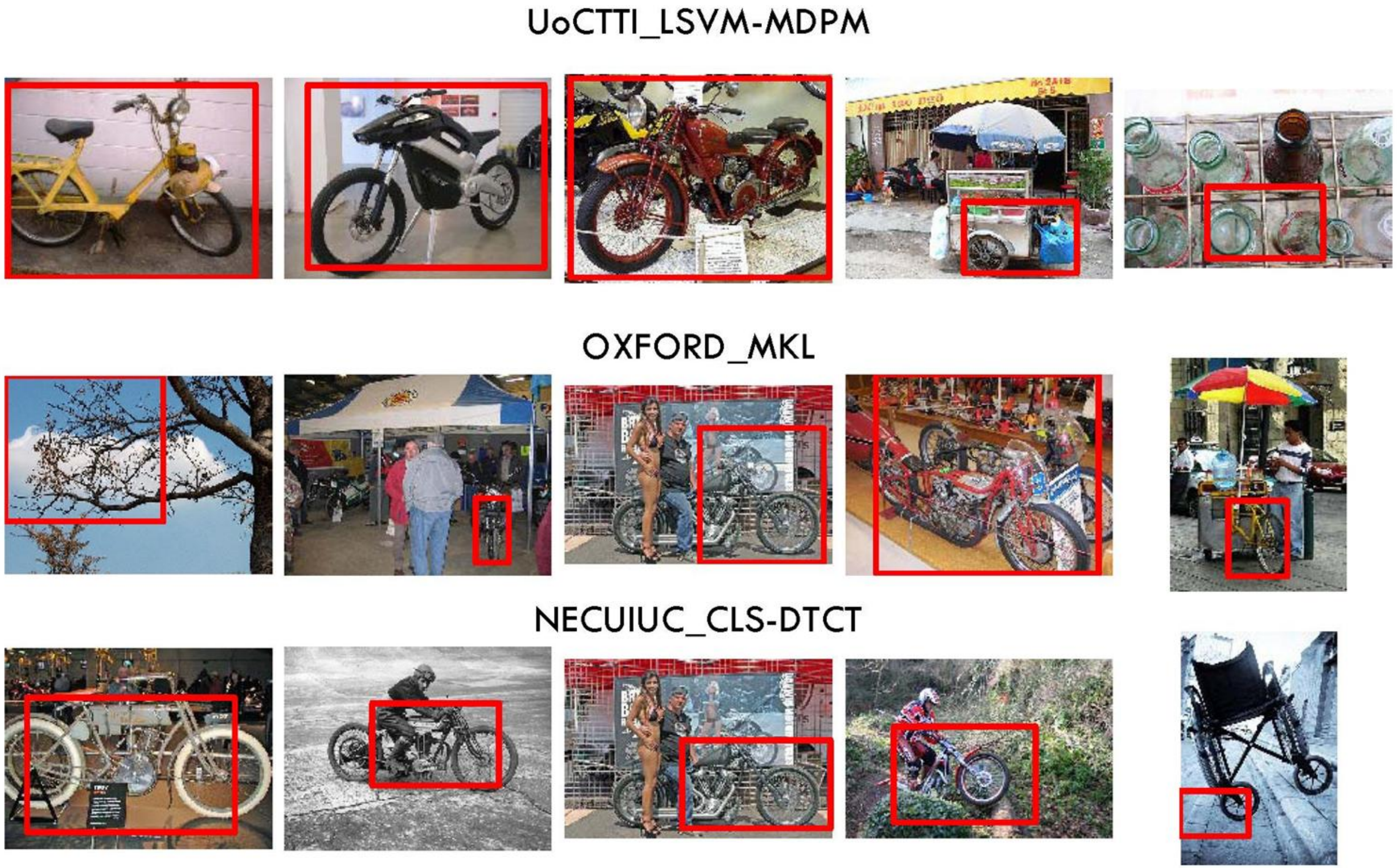
Near misses: IoU falls short of 0.5



True positives - detecting bicycle



False positives - detecting bicycle



Today's agenda

- Spatial pyramids
- Object detection
  - Task and evaluation
- A simple detector
- Deformable parts model

Dalal-Triggs method

sliding window

At every patch as the window slides

1. Convert the image patch into your favorite feature representation
    a. For example:
        i. HoG,
        ii. HoG with PCA,
        iii. RGB with LDA,
        iv. Bag of words on RGB
        v. etc.
2. Use a trained classifier to determine if it is a specific class
    a. e.g. kNN classifier
3. Accumulate the predictions over all the patches

Sliding window + hog features

- Slide through the image and check if there is an object at every location

No person here

Sliding window + hog features

YES!! Person match found

- Slide through the image and check if there is an object at every location

Sliding window + hog features

No bus found

- But what if we were looking for buses?

Sliding window + hog features

- But what if we were looking for buses?

No bus found

Sliding window + hog features

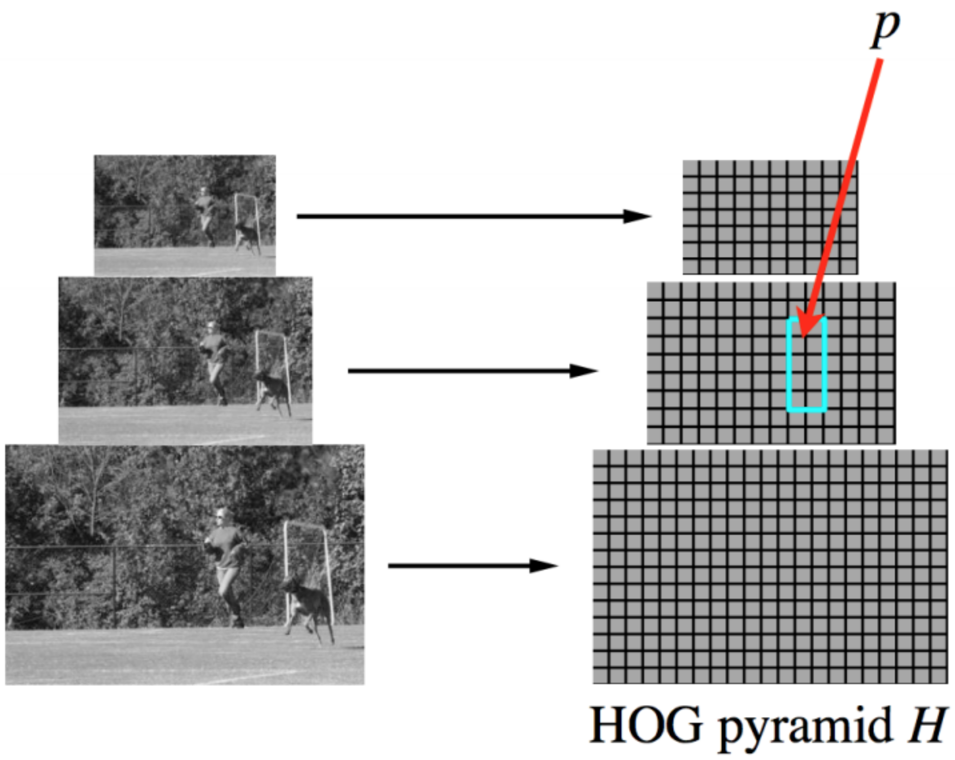
No bus found

- We will never find the object if we don’t choose our window size wisely!

Sliding window + hog features

- We need to do multi-scale sliding windows with pyramids

Computationally, we first resize the image to different sizes and then extract features at each size



Today's agenda

- Spatial pyramids
- Object detection
  - Task and evaluation
- A simple detector
- Deformable parts model

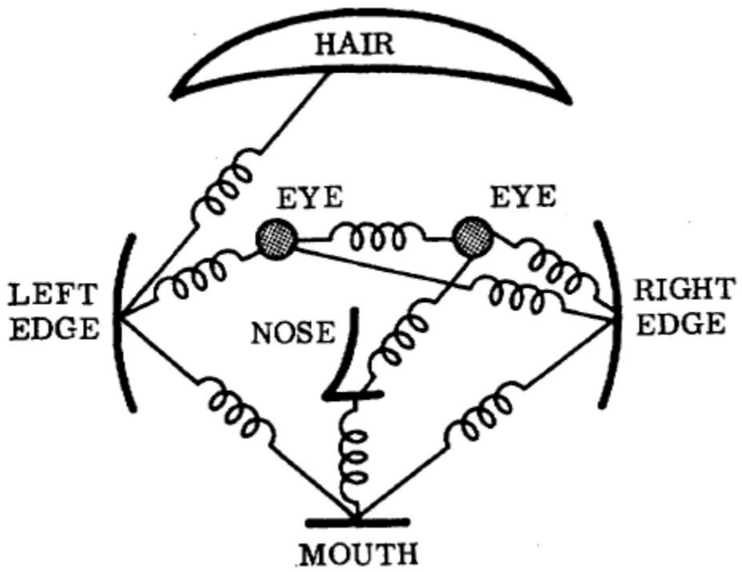
Recap – bag of words

- We can present images as a set of “words”
  - Where each word represents a **part** of the image.

- Can we use the location of these patches to find objects within those images?

Deformable Parts Model

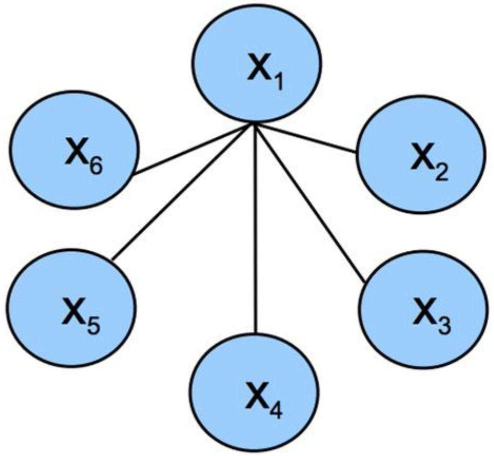
- Represents an object as a "collection of parts" arranged in a "deformable configuration"
- Each part represents local appearances
- Spring-like connections between certain pairs of parts

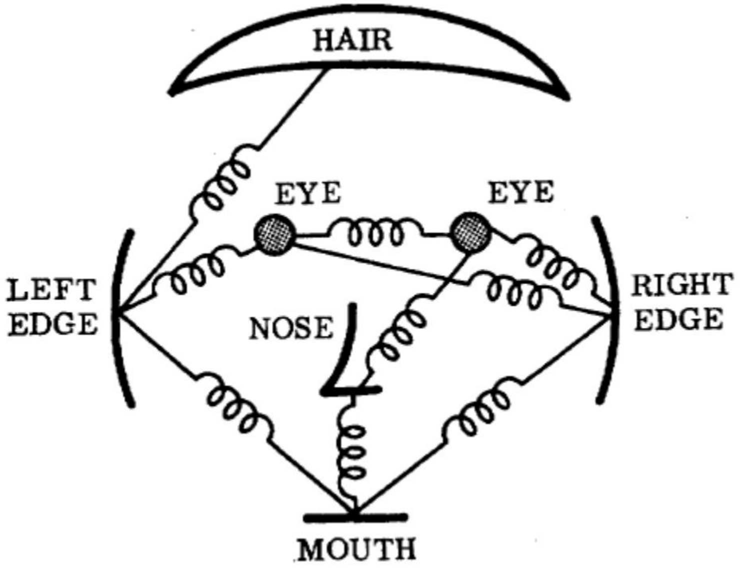


Fischler and Elschlager, Pictoral Structures, 1973

Deformable parts model

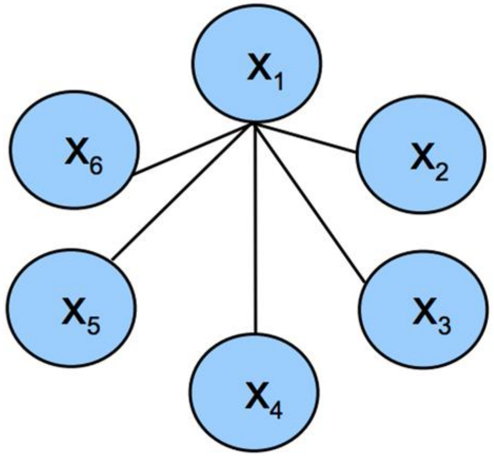
- The parts of an object form pairwise relationships.
- We can model this using a “star model”
  - where every part is defined relative to a root.





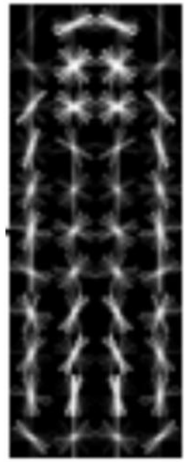
Detecting a person with their parts

- For example, a person can be modelled as having a head, left arm, right arm, etc.
- All parts can be modelled relative to the global person detector, which acts as the root.



Deformable parts model

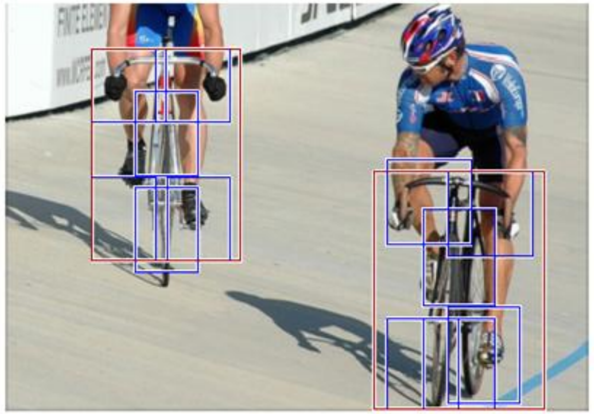
- Each model will have a global filter. And a set of part filters. Here is an example of a global person filter with its ‘head’ part filter:

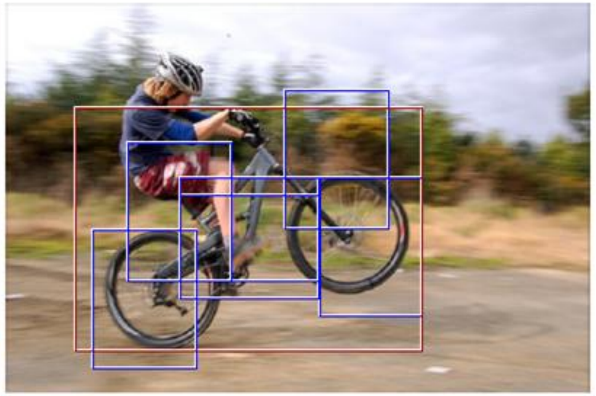


Global/root filter

Part filter

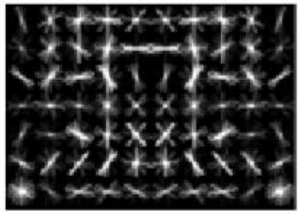
5-part bicycle model



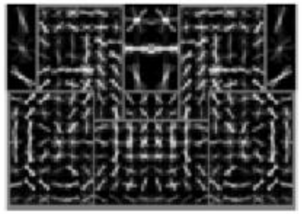


"side view" bike model component

Root filter

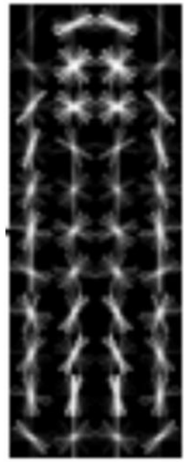


Part filters

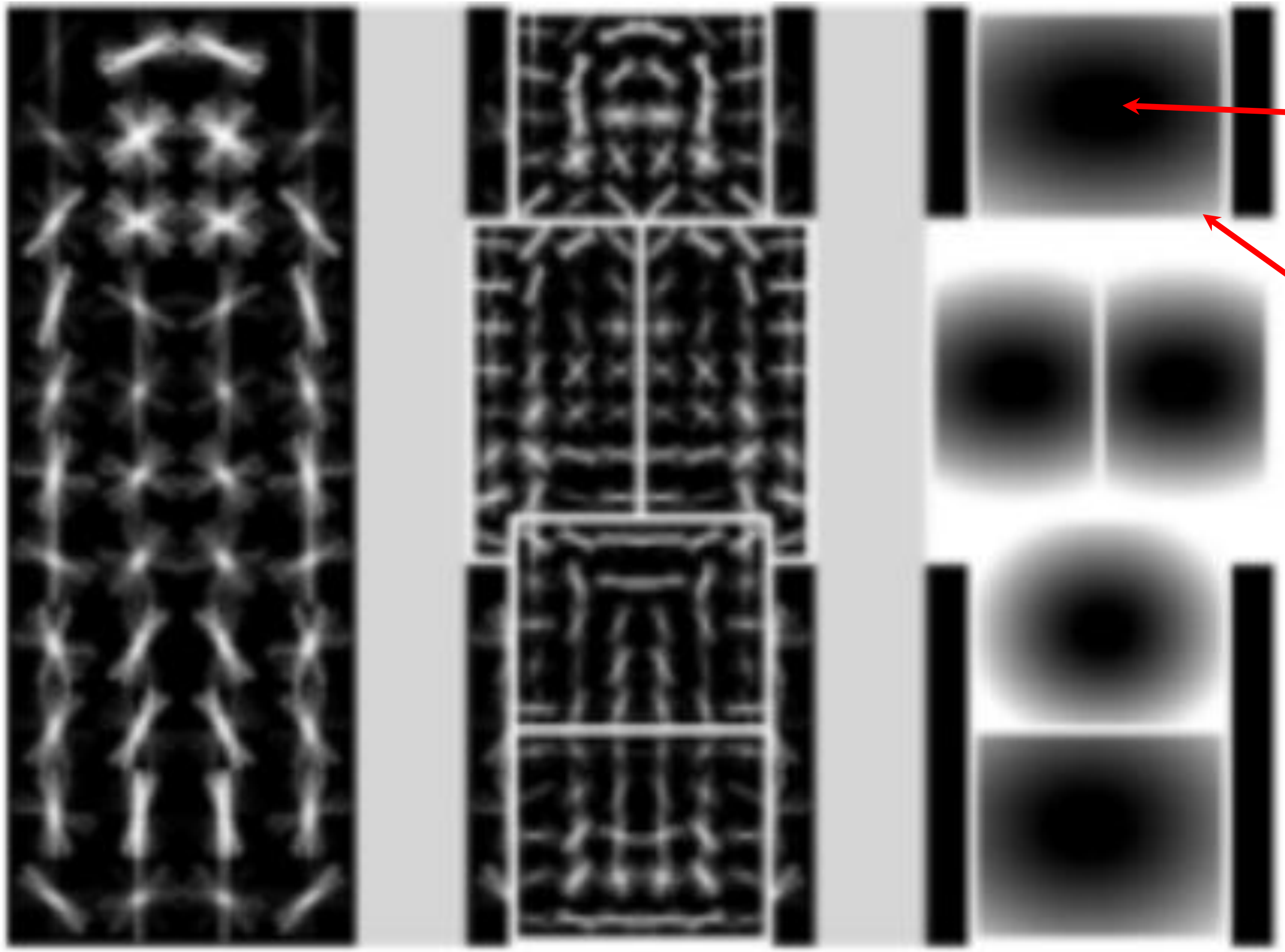


Deformable parts model

- Mixture of deformable part models
- Each component has global component + deformable parts
- Part filters have finer details



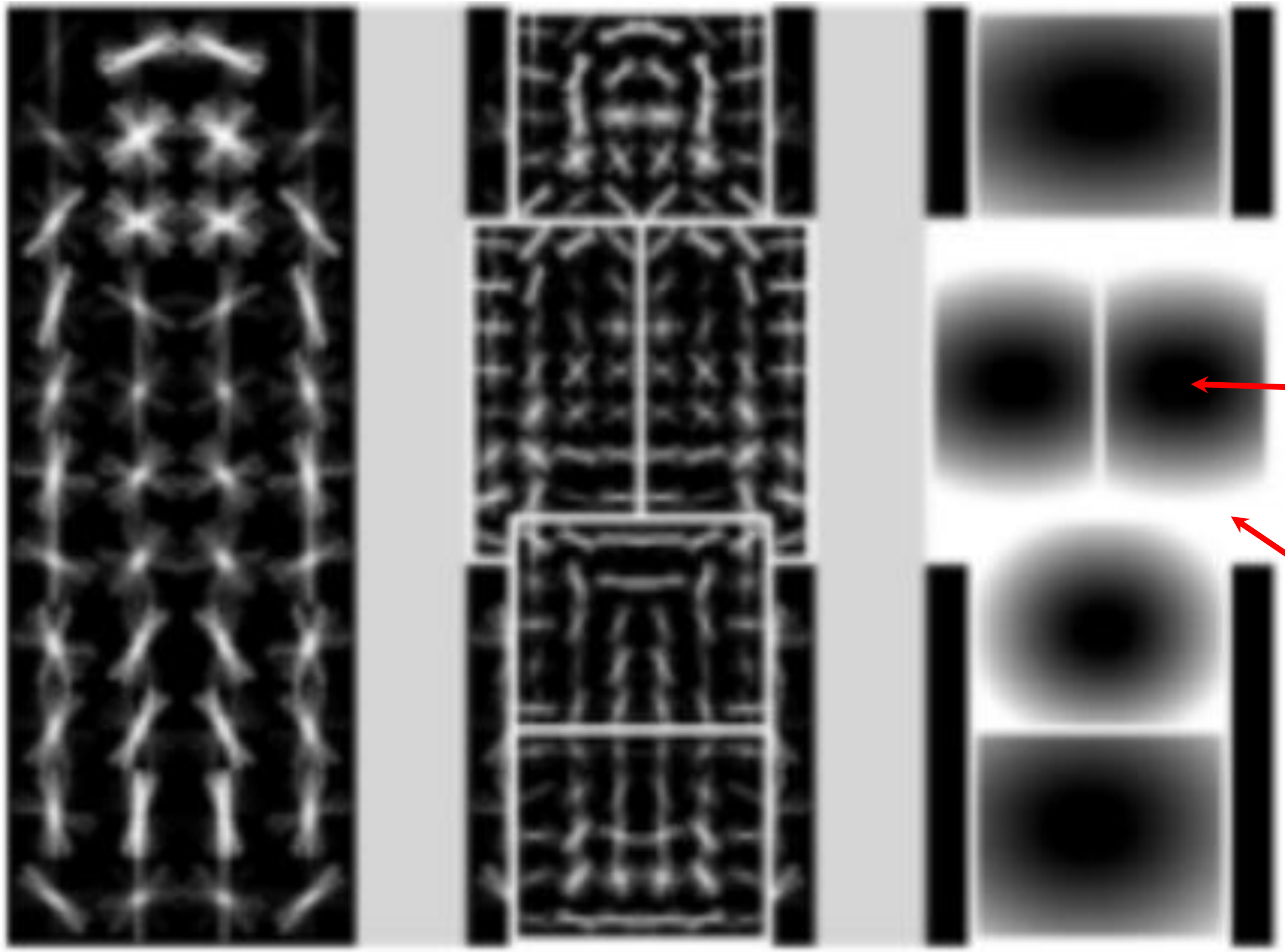
DPM for person model with 5 parts



If the head is here,
the penalty is low

If the head is here,
the penalty is high

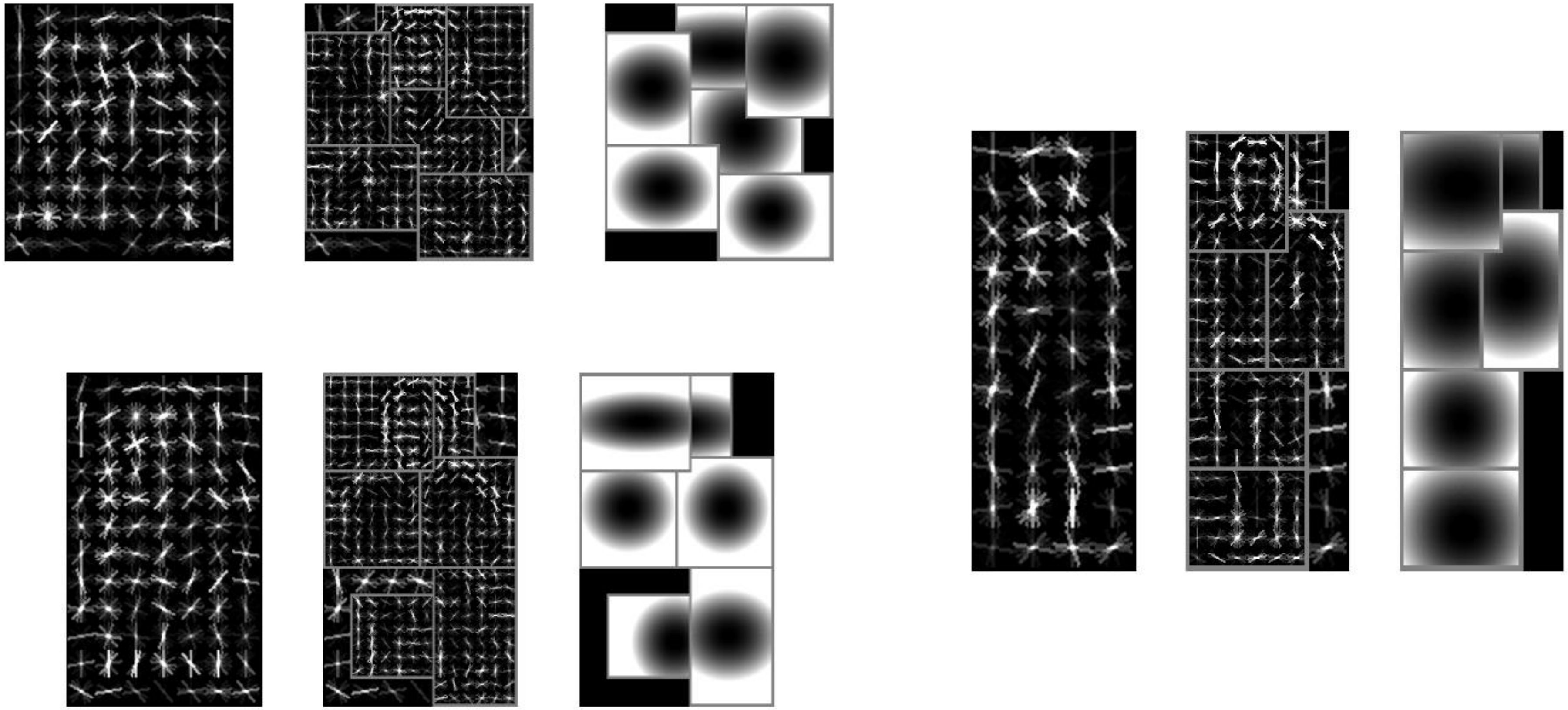
DPM for person model with 5 parts



If the arm is here,
the penalty is low

If the arm is here,
the penalty is high

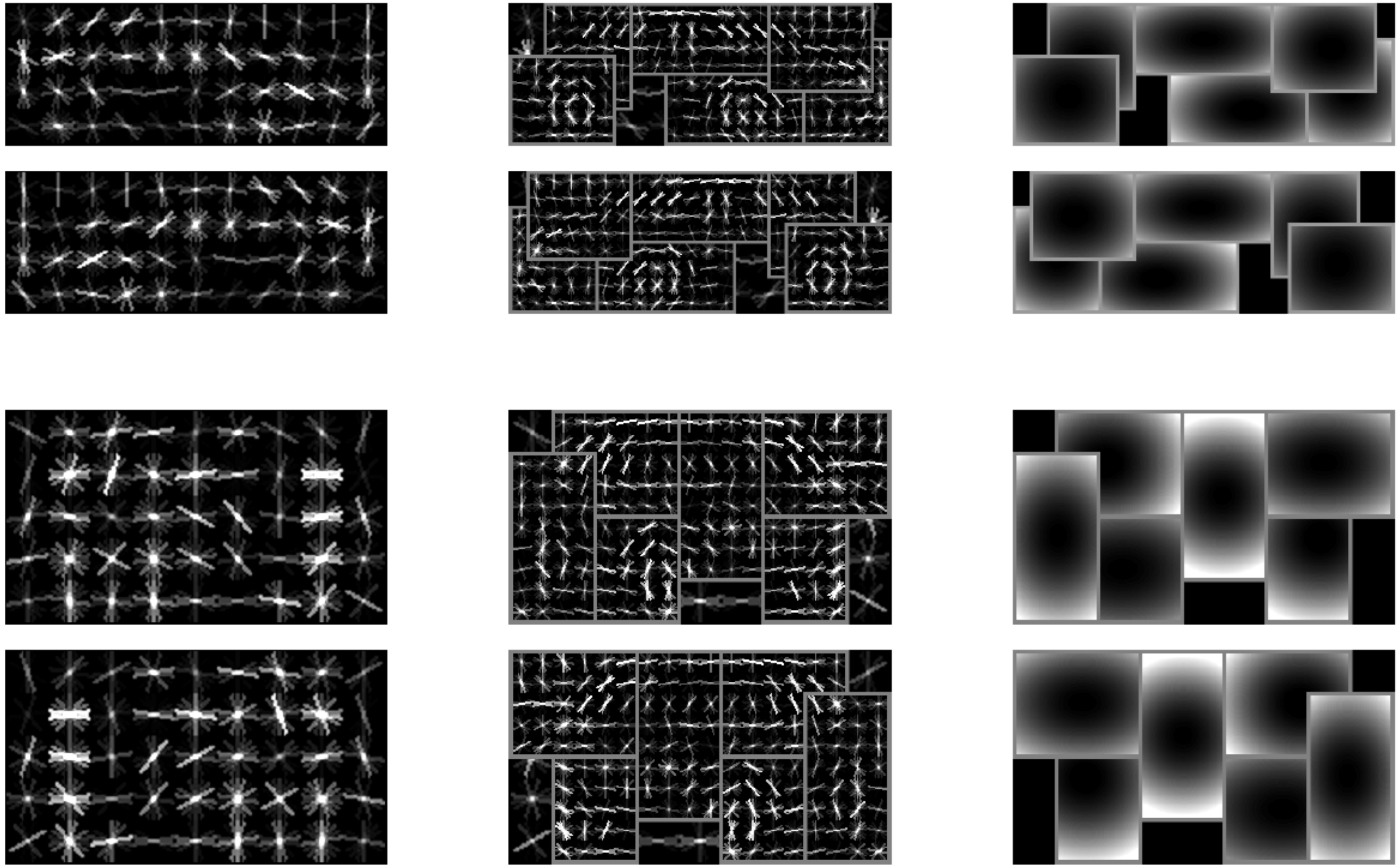
Multiple DPM for person model with 6 parts



DPM for car with 6 parts

side view

frontal view

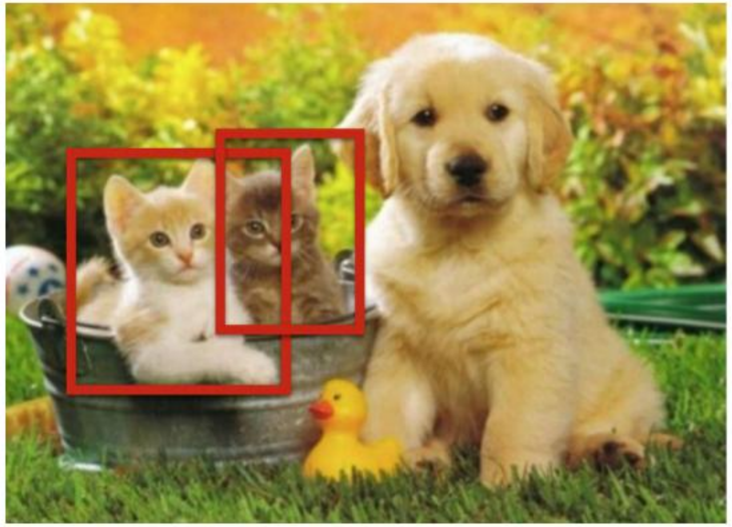


root filters (coarse) part filters (fine) deformation models

How do we use the parts to make a detection?

Intuition:

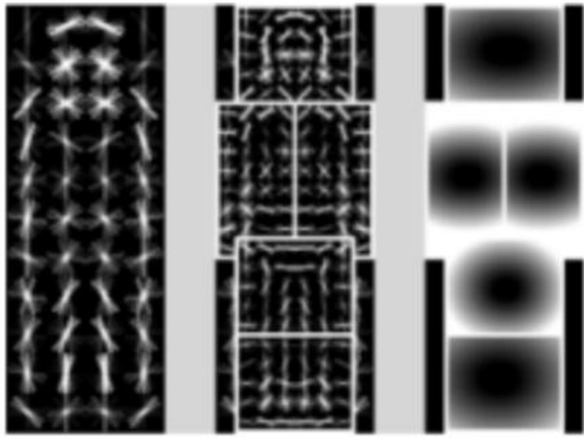
1. First, use the sliding windows at different pyramid scales to detect each part (and the root).
2. Each part gives you a score for where the person might be
3. Accumulate the global and part scores and penalize the deformation of the parts.



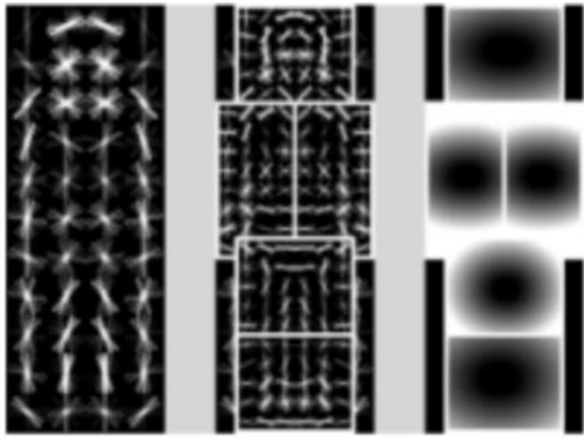
Example for detecting people

Image input

A feature template for person

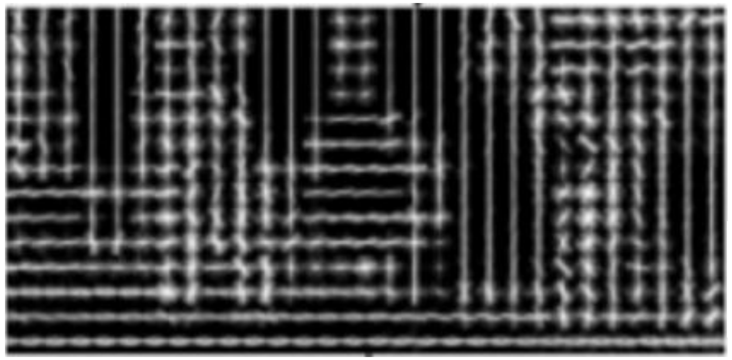


First extract features

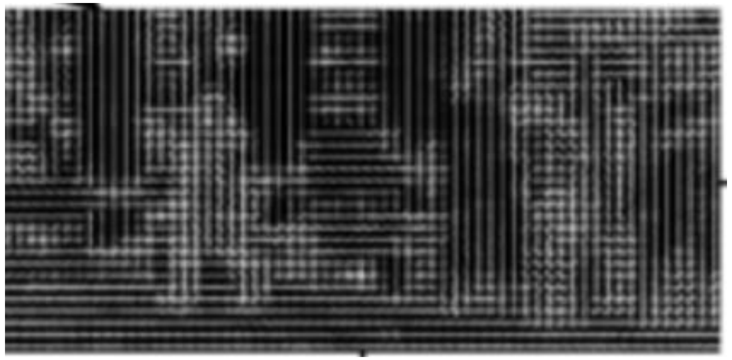


A feature template for person

Image input

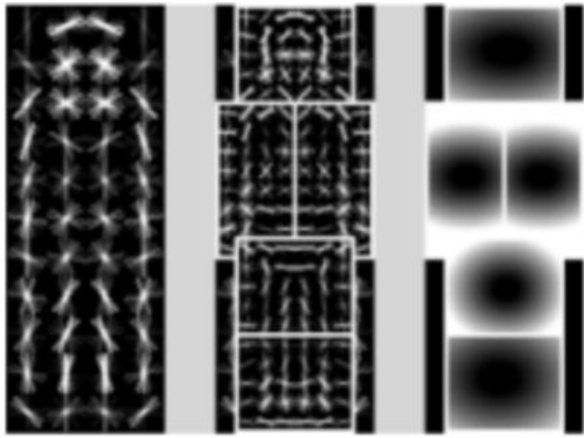


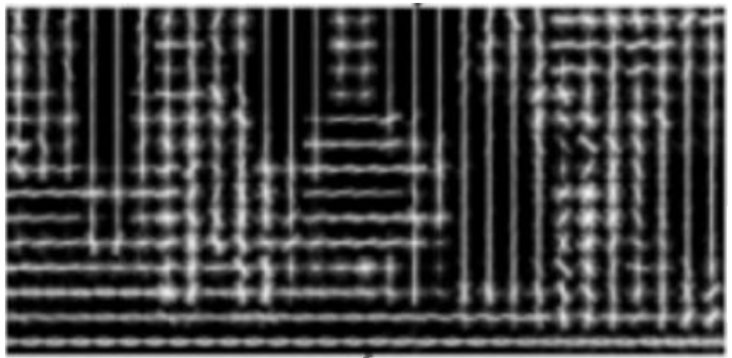
Features



Features at 2x resolution

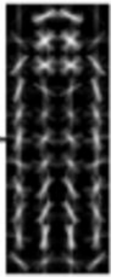
Calculate scores for part templates

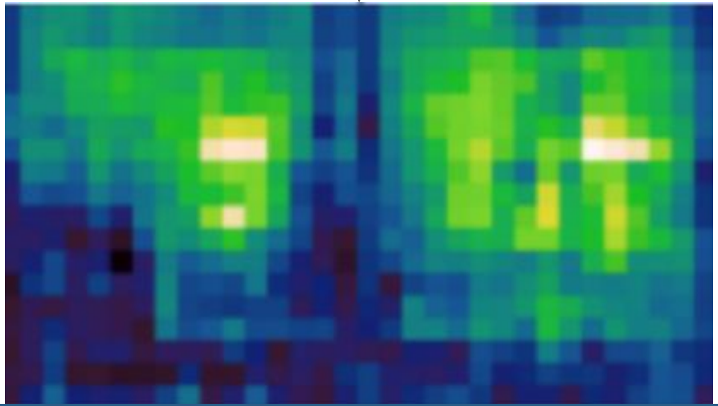




Features

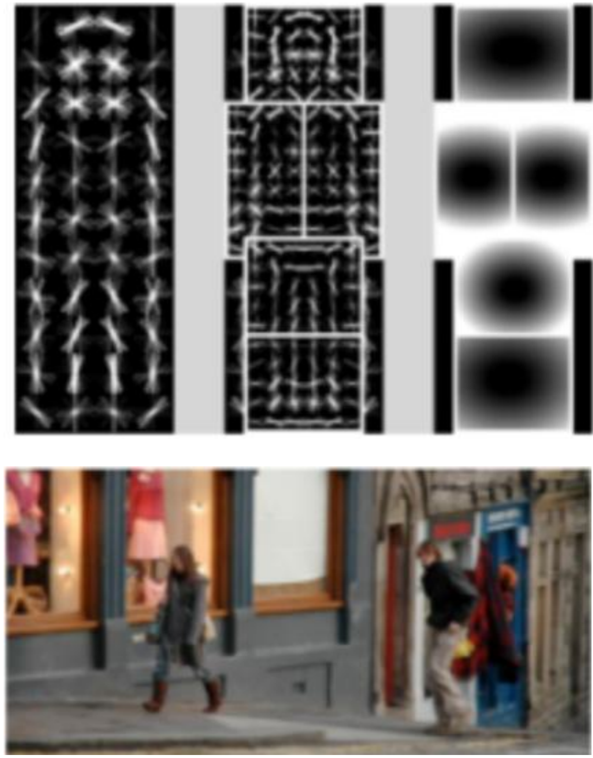
convolution

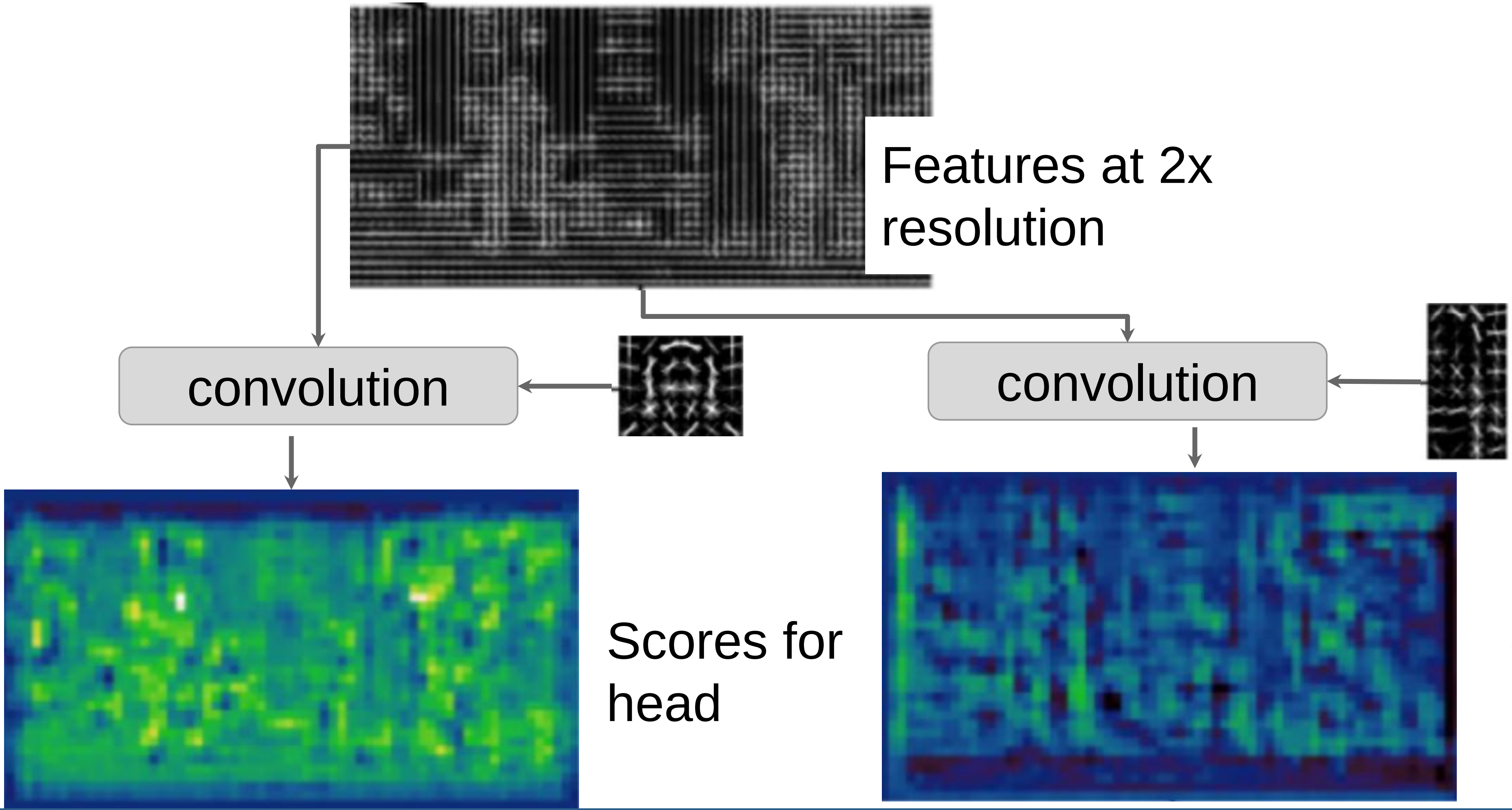




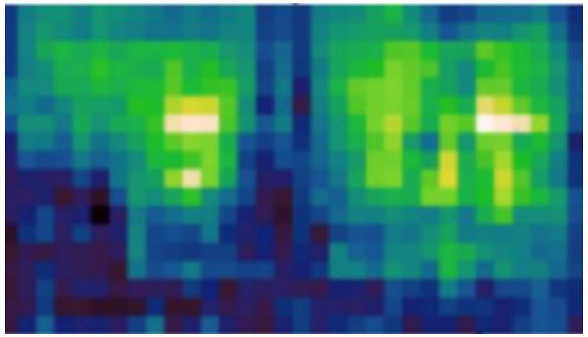
Global scores for where a person might be

Calculate scores for global template

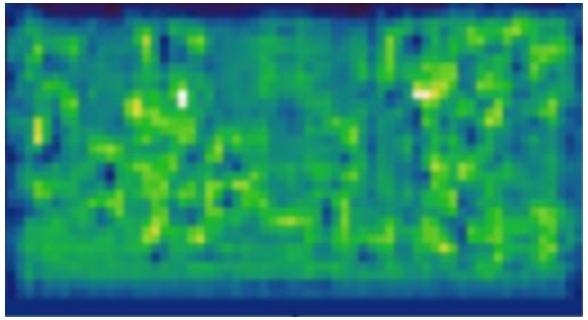




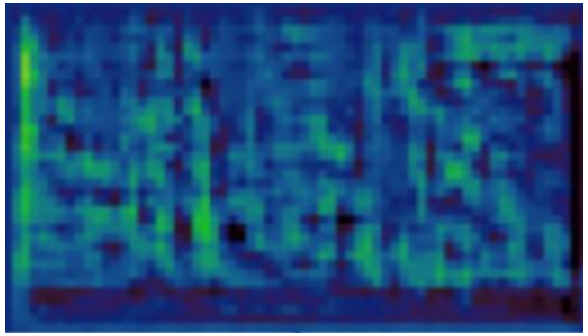
After step 1, we have scores for all parts and global template



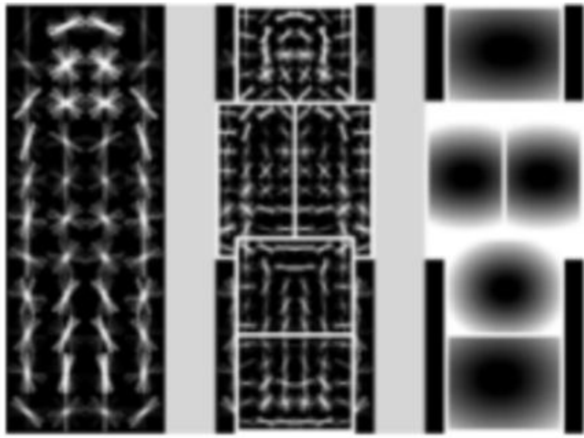
Global scores



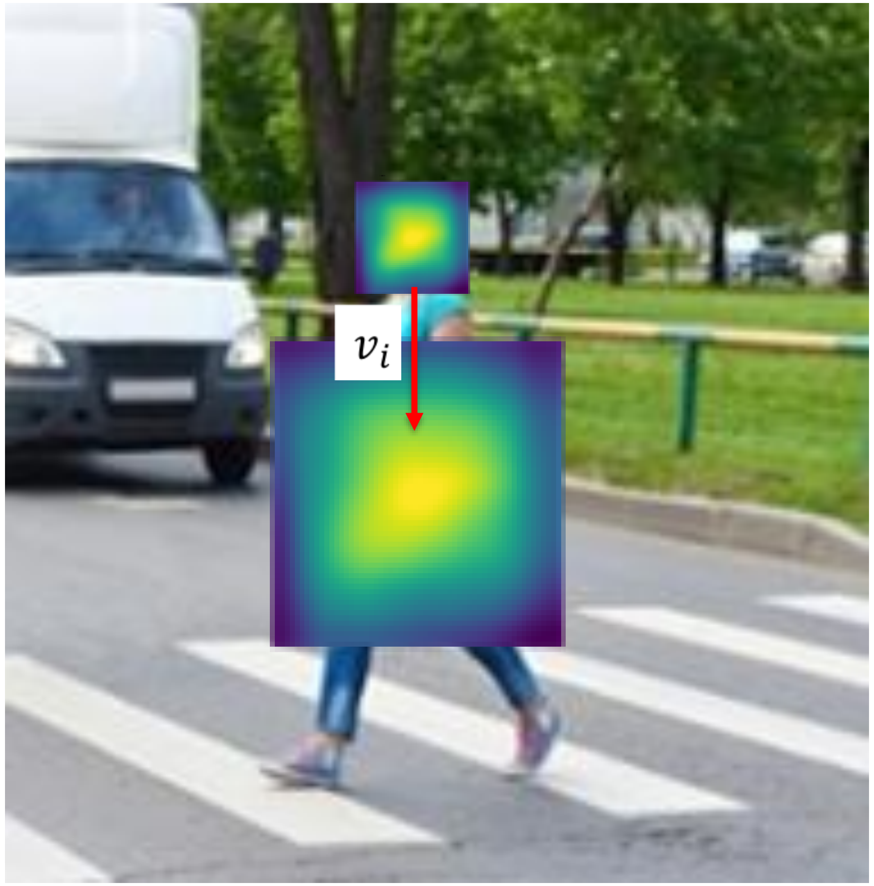
Scores for head



Scores for right arm

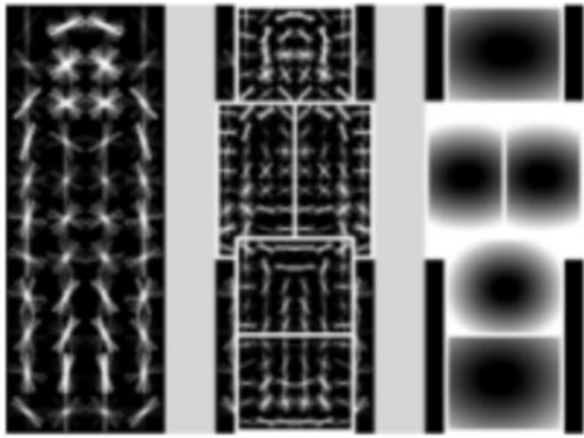


Allowing each part to deform and guess where the entire body is.

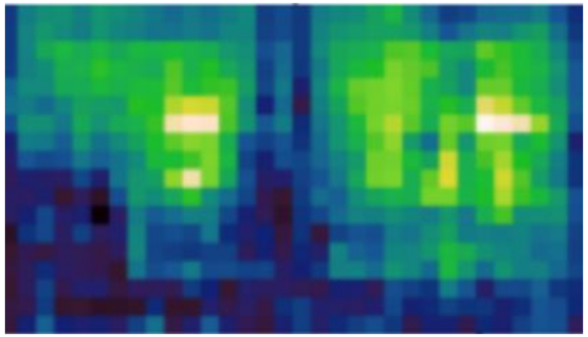


- Given the location for the detected head, we can guess where the body should be.
- The body should be in the direction (v_i) predefined in the model
- Bodies can be of different sizes and shapes. So we allow it to deform by some variable d_i
- This deformation spreads the scores to potential locations of the body

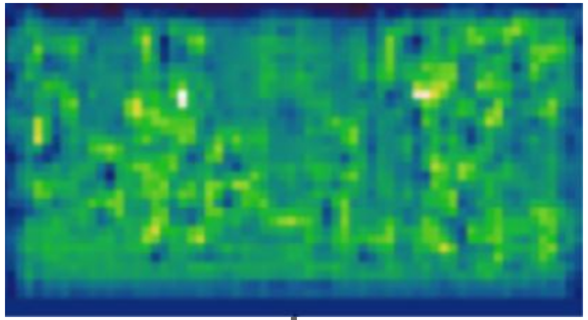
Step 2: each part gives you a score for where the person might be

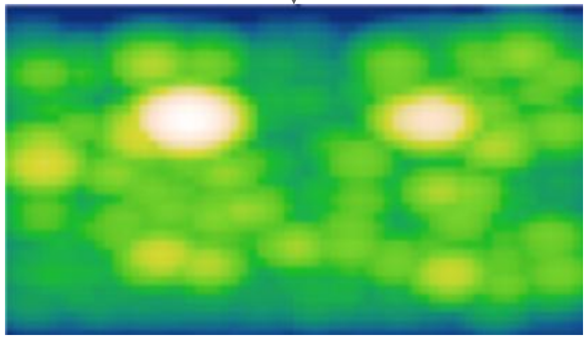


Global scores

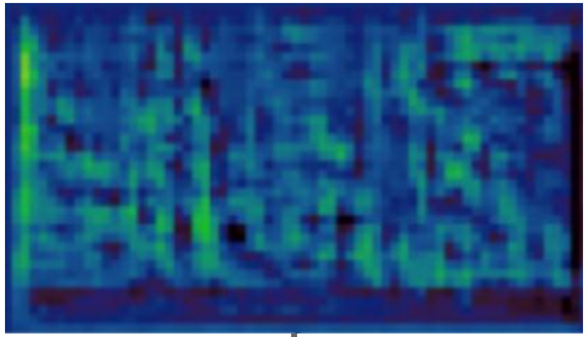


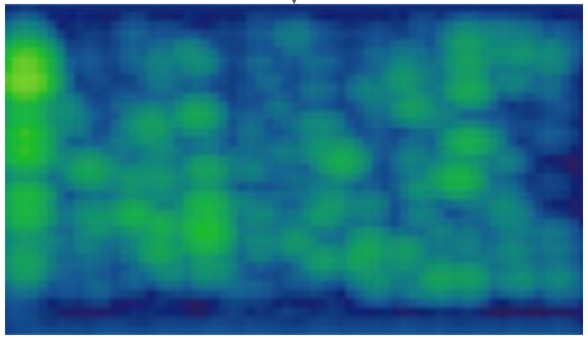
Scores for head





Scores for right arm





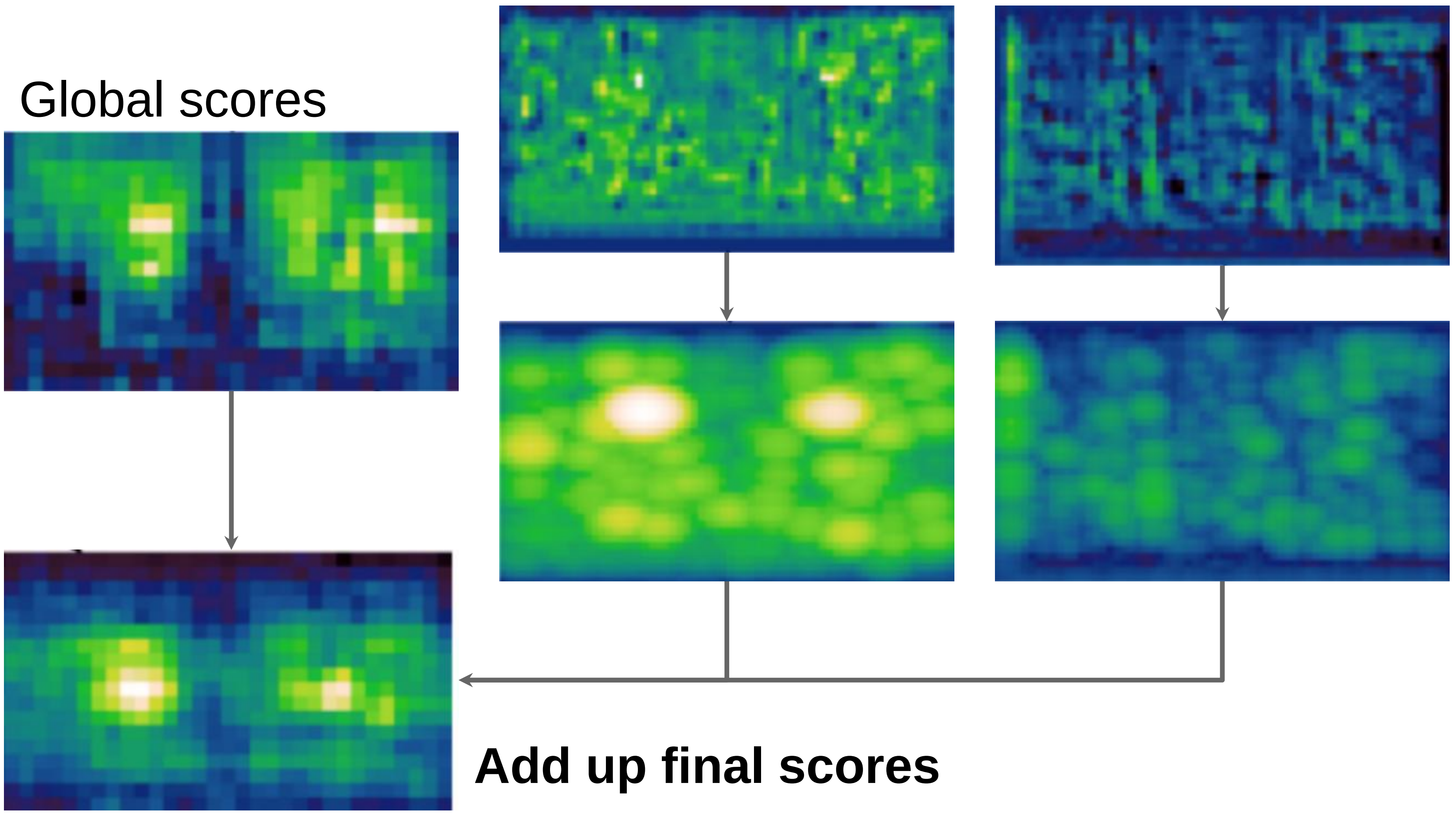
Each part is allowed to deform. So it deforms to where the person might be.

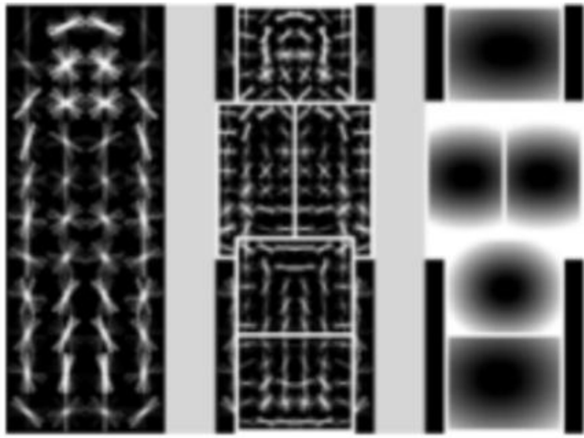
Intuition: If the head is here, where is the whole person likely to be?

Step 3: Add up the scores for the final detections

Scores for head

Scores for right arm





Formally, DPM is defined as:

- A model for an object with n parts is a $(n+2)$ tuple:

$$(F_0, P_1, \cdots, P_n, b)$$

Root filter   Model for 1st part   Bias term

- Each part-based model defined as:

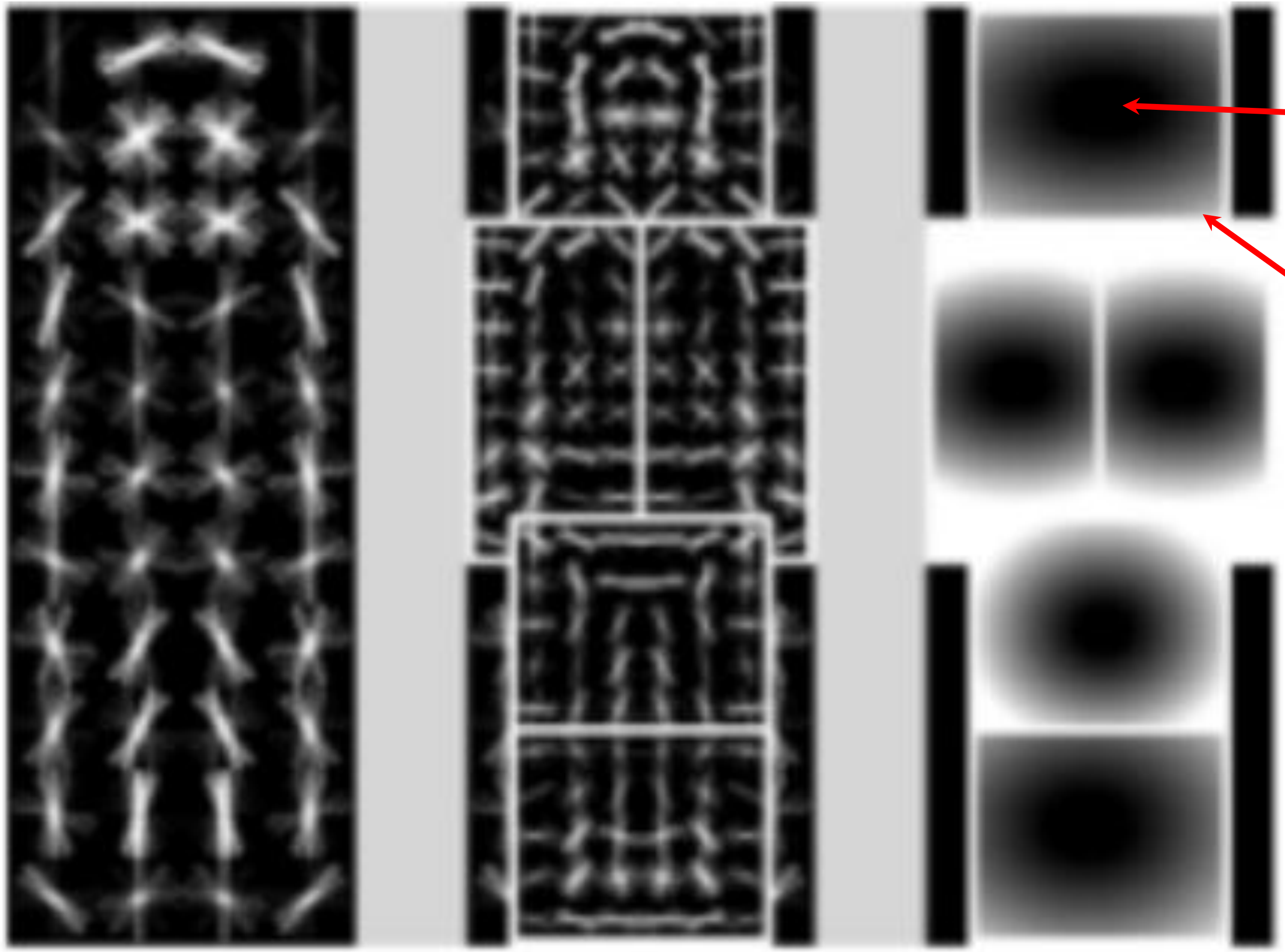
$$(F_i, v_i, d_i)$$

F_i filter for the *i*-th part

v_i “anchor” position for part *i* relative to the root position

d_i defines a deformation cost for each possible placement of the part relative to the anchor position

d_i can be defined in many ways. We will use a Gaussian filter to define it.



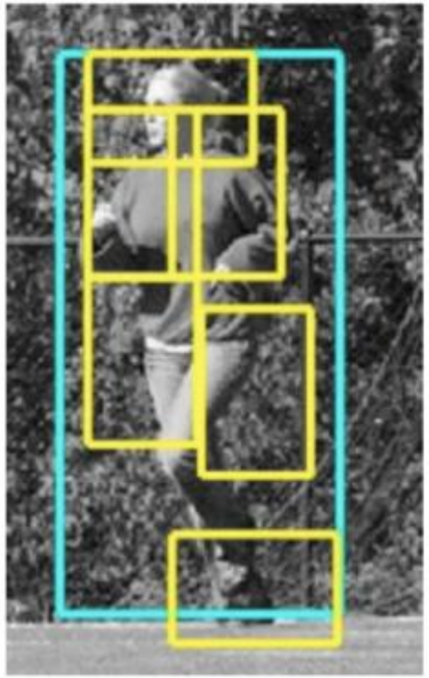
If the head is here, the penalty is low

If the head is here, the penalty is high

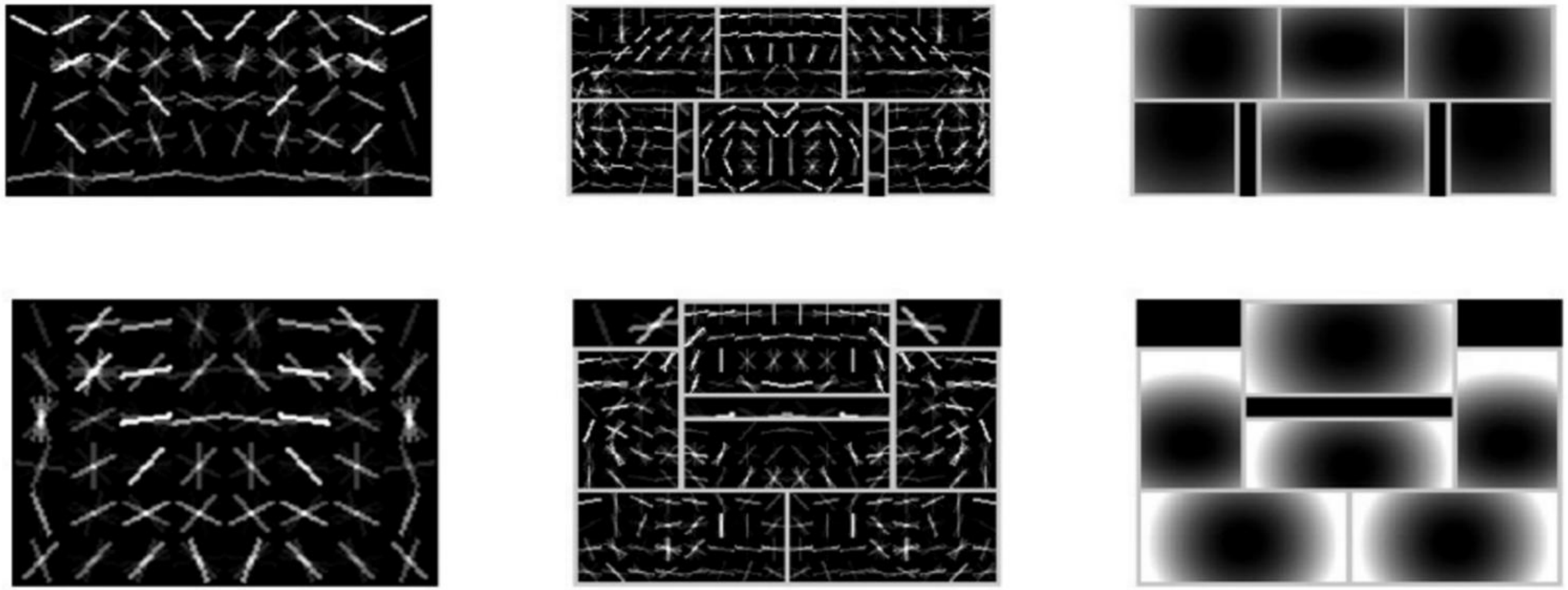
Calculating the score for a detection

The score for a detection is defined as the sum of scores for the global and part detectors *minus* the sum of deformation costs for each part.

This means that if a detection's parts are really far away from where they should be, it's probably a false positive.



Deformable Parts Model (DPM) - bicycle

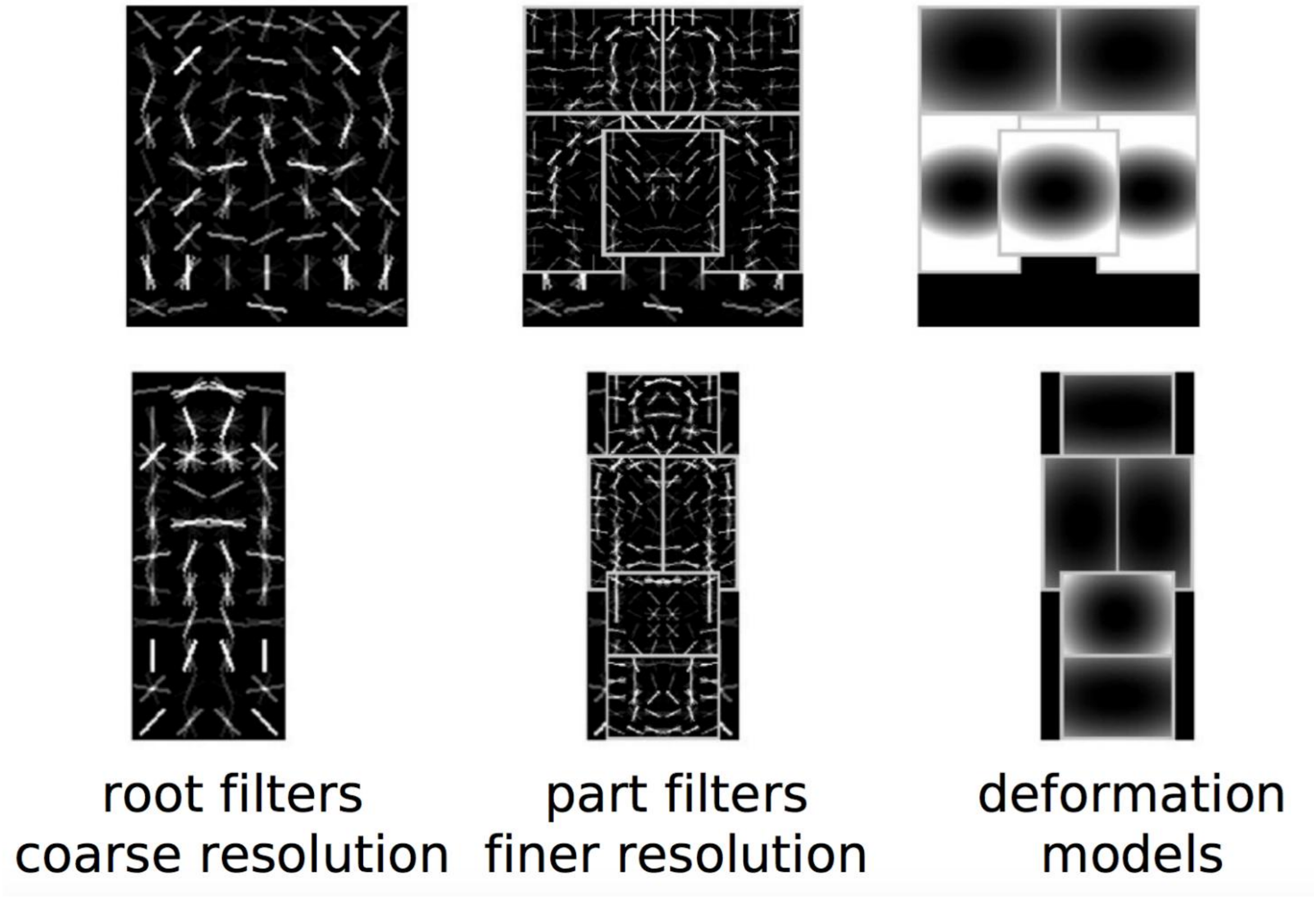


root filters
coarse resolution

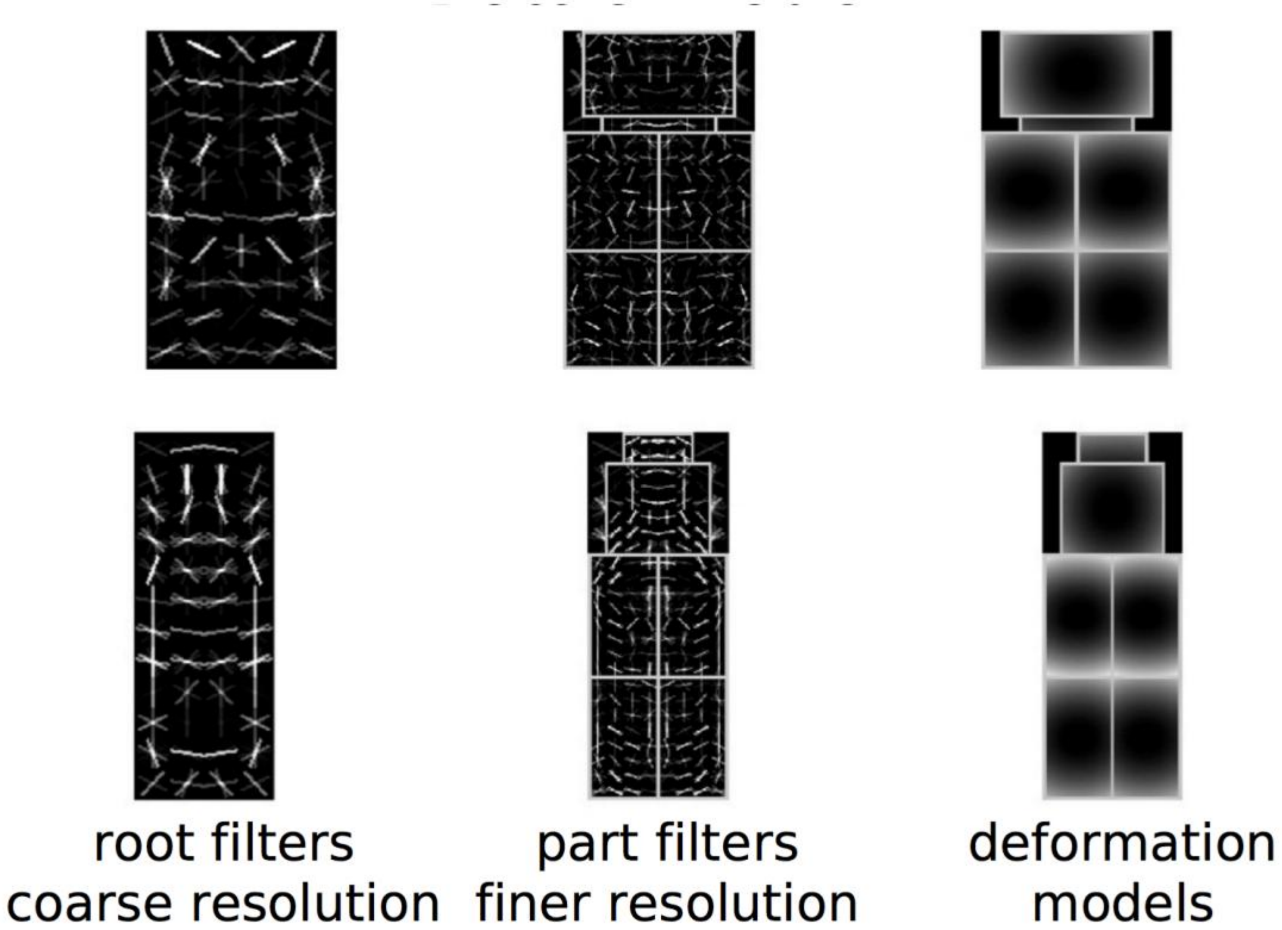
part filters
finer resolution

deformation
models

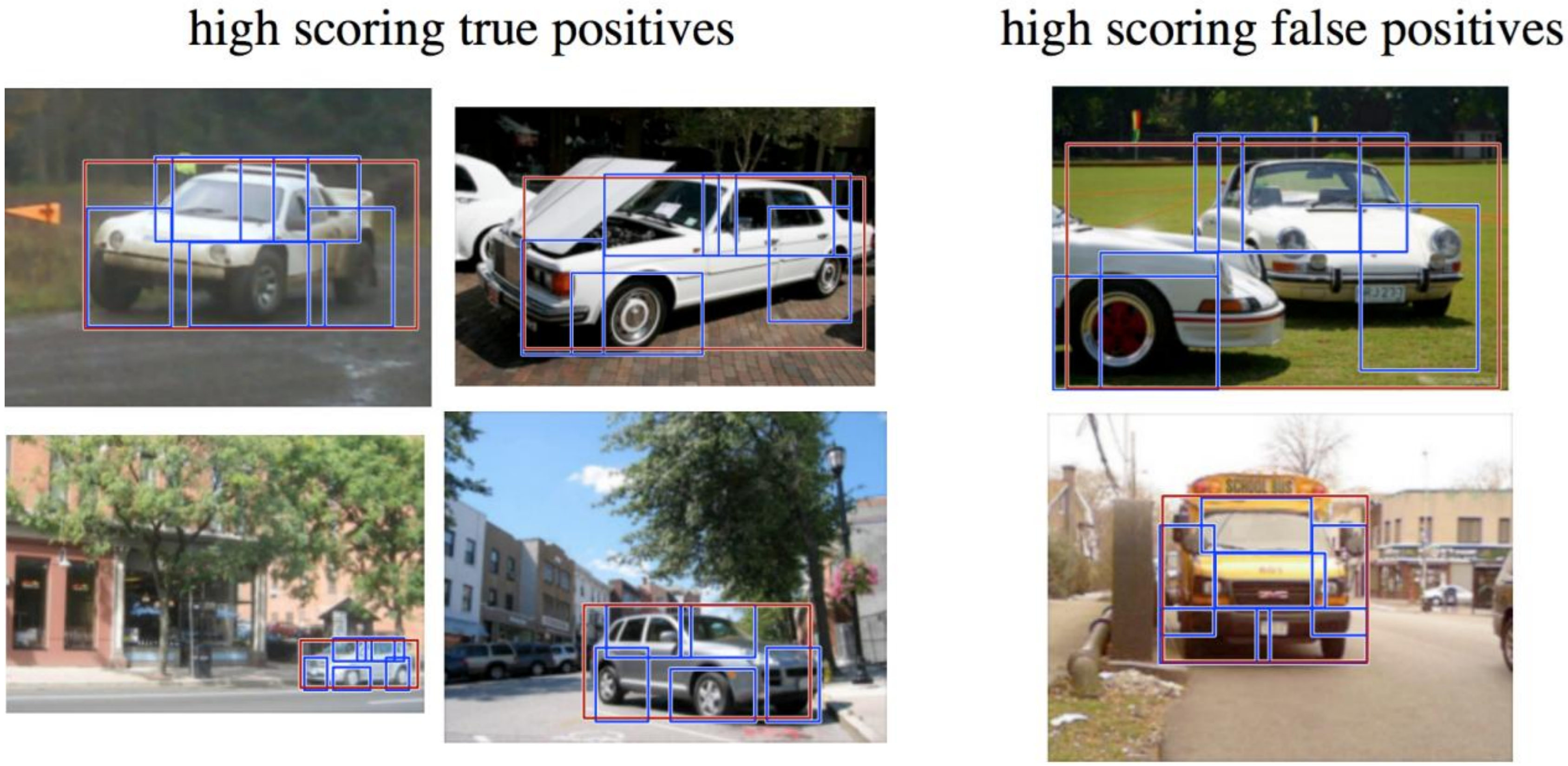
DPM with HoG features - person



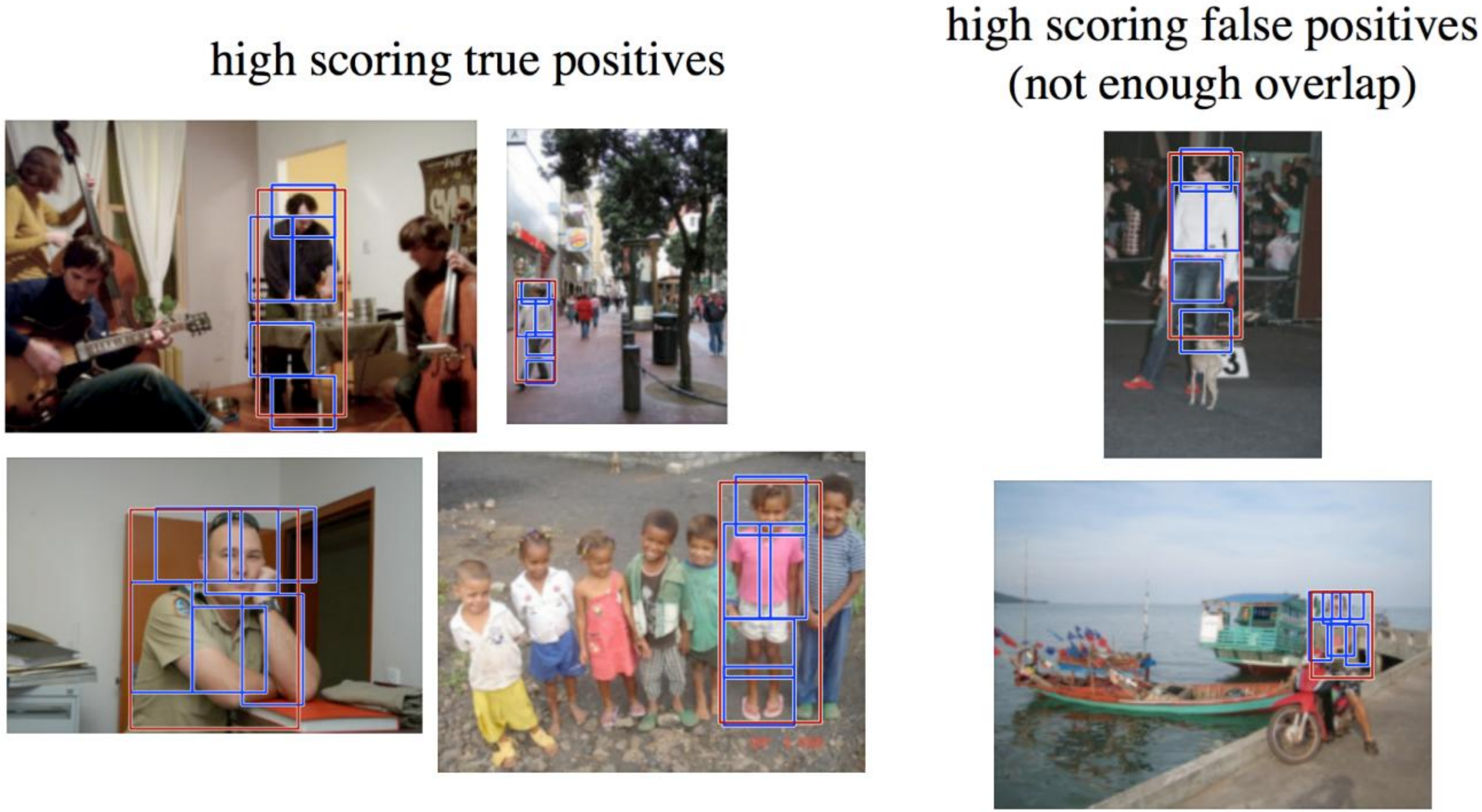
DPM - bottle


root filters
coarse resolution
part filters
finer resolution
deformation
models

Results – car detection



Results – Person detection

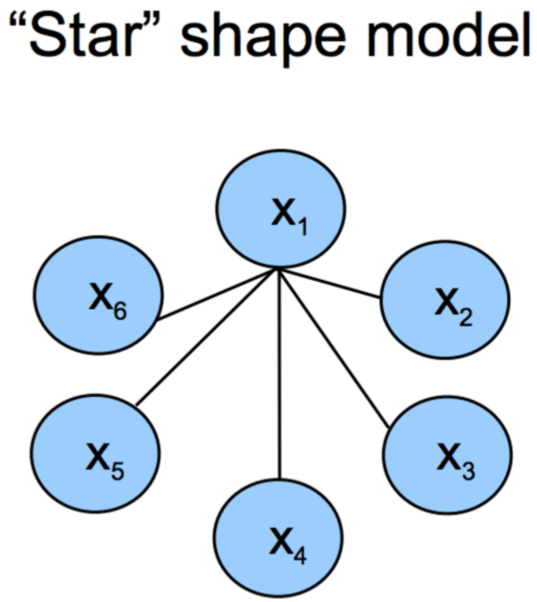


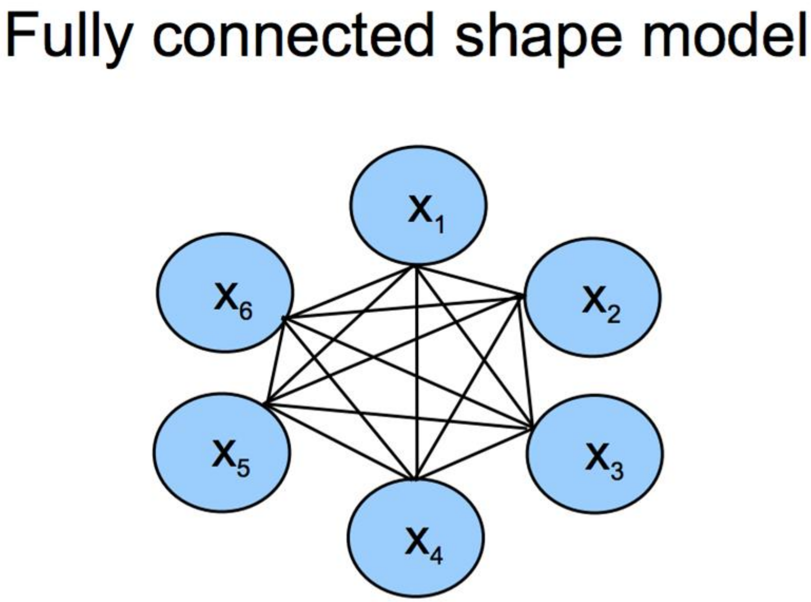
Results – horse detection

DPM - discussion

- **Approach**
  - Manually selected set of parts - Specific detector trained for each part
  - Spatial model trained on part activations
  - Evaluate joint likelihood of part activations
- **Pros**
  - Parts have intuitive meaning.
  - Standard detection approaches can be used for each part.
  - Works well for specific categories.
- **Disadvantages**
  - Parts need to be selected manually
  - Some parts don’t have a simple appearance
  - No guarantee that some important part hasn’t been missed
  - When adding a new category, it takes a lot of manual effort

Extensions - From star shaped model to constellation model





Today’s agenda

- Spatial pyramids
- Object detection
  - Task and evaluation
- A simple detector
- Deformable parts model

Next lecture

Linear Classifiers and Backpropagation