A Multiple Kernel Learning Approach to Joint Multi-Class Object Detection

Christoph H. Lampert

Matthew B. Blaschko

Max-Planck-Institute for Biological Cybernetics
Department Schölkopf: Empirical Inference
Tübingen, Germany

June 11, 2008

Task: Object Detection in Images

Which of the following 20 categories occur in this image?

`aeroplane, bird, bicycle, boat, bottle, bus, car, cat, chair, cow, diningtable`
`dog, horse, motorbike, person, pottedplant, sheep, sofa, train, tv/monitor`

chair, diningtable, sofa, tv/monitor

Task: Object Detection in Images

Which of the following 20 categories occur in this image?

`aeroplane, bird, bicycle, boat, bottle, bus, car, cat, chair, cow, diningtable`
`dog, horse, motorbike, person, pottedplant, sheep, sofa, train, tv/monitor`

aeroplane, person

Task: Object Detection in Images

Which of the following 20 categories occur in this image?

`aeroplane, bird, bicycle, boat, bottle, bus, car, cat, chair, cow, diningtable`
`dog, horse, motorbike, person, pottedplant, sheep, sofa, train, tv/monitor`

horse, person

Object Co-Occurrence and Context

cat

person

chair, diningtable, sofa, tv/monitor

cow

aeroplane, person

horse, person

chair, sofa, person

chair, dog, sofa pottedplant

bicycle, person

train

chair, diningtable

car

motorbike, person

pottedplant

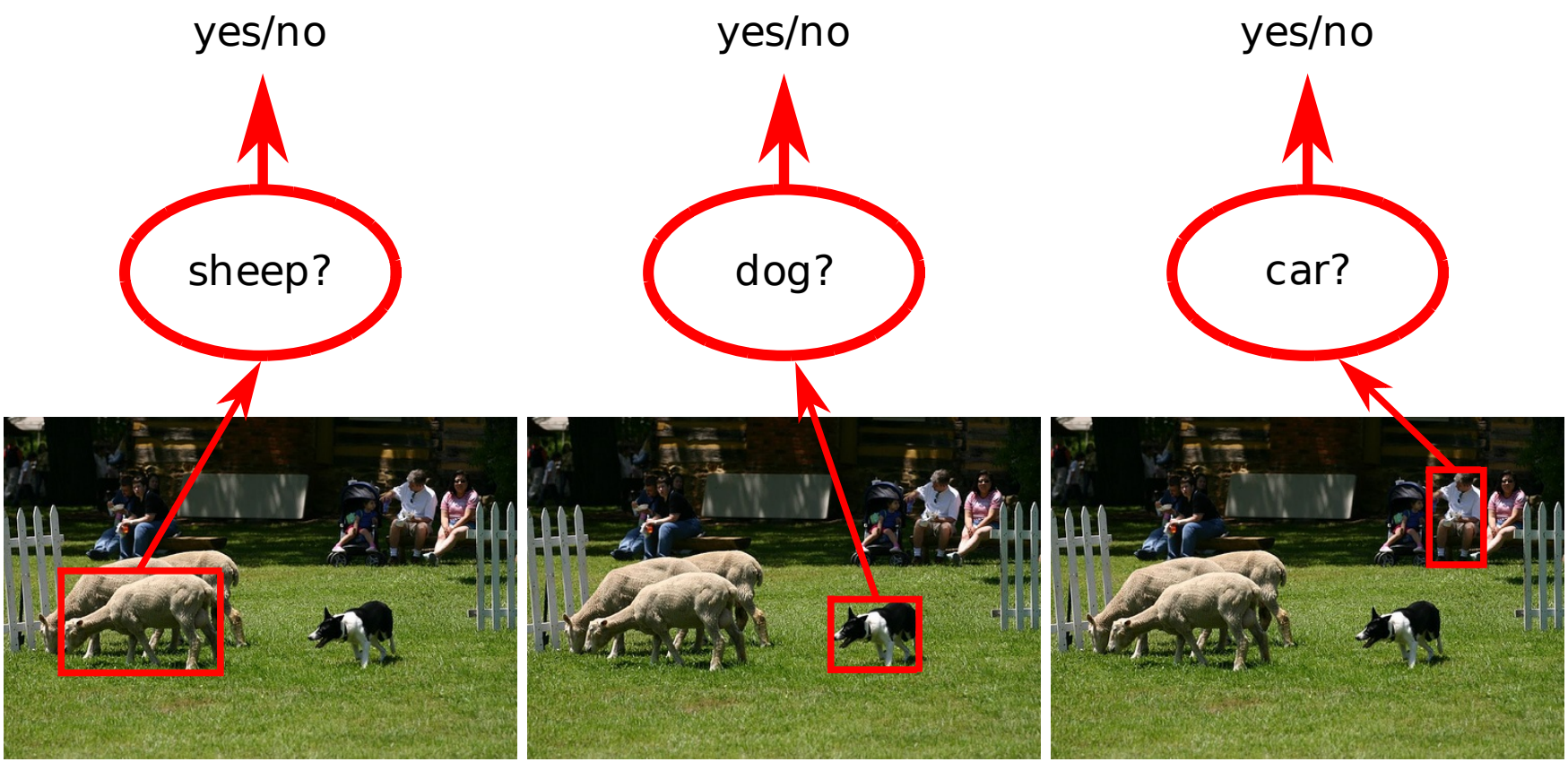
dog, sofa

Natural images often contain more than 1 object class.

Co-occurrence is not random, but e.g. due to location or function.

Image context gives valuable clues about object presence. . .

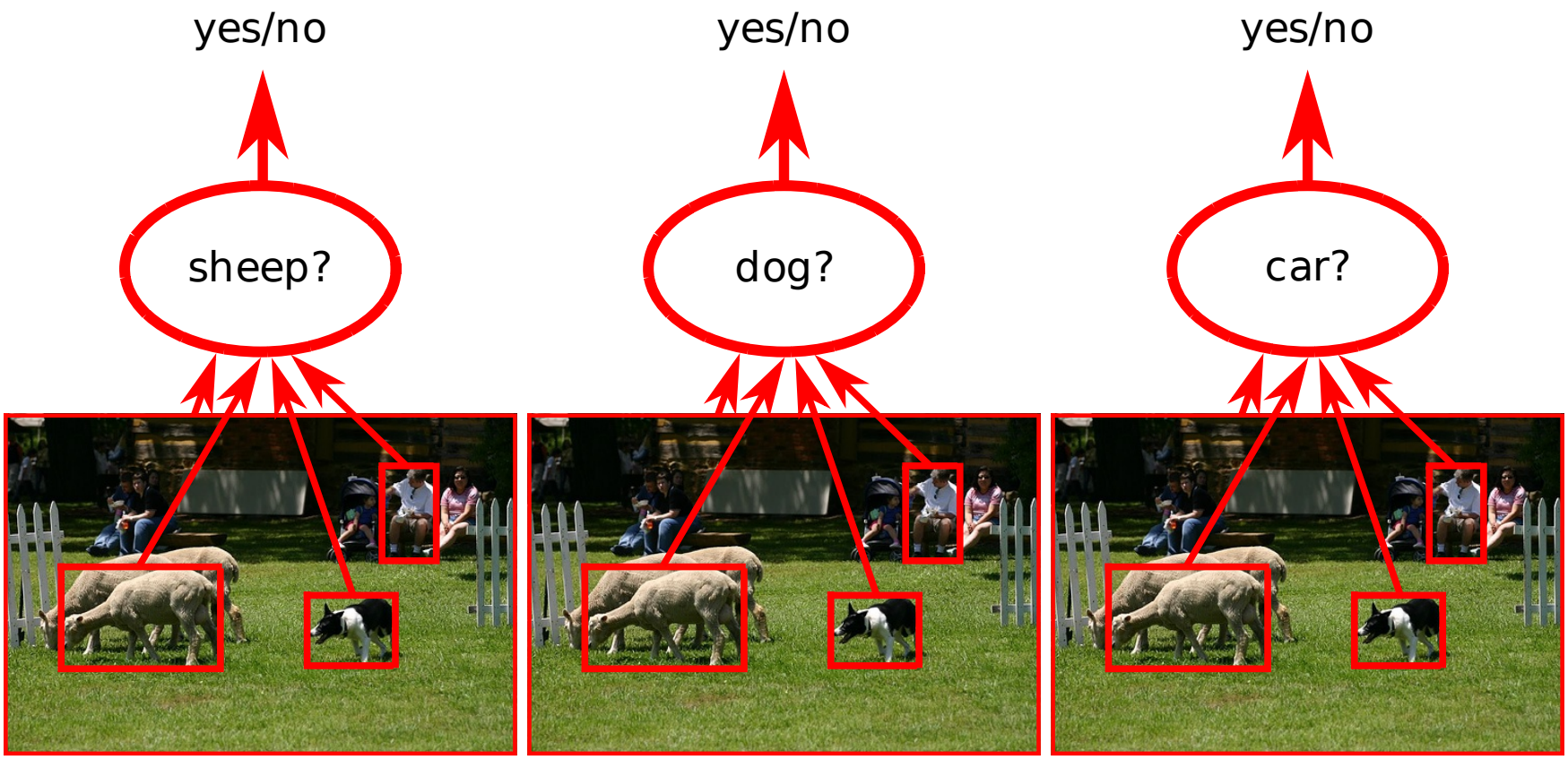
Single Class Object Detection



Separate *per-class object detection* classifies only *one candidate region* at a time.

- *Co-occurrence and context information is discarded.*

Joint Class Object Detection



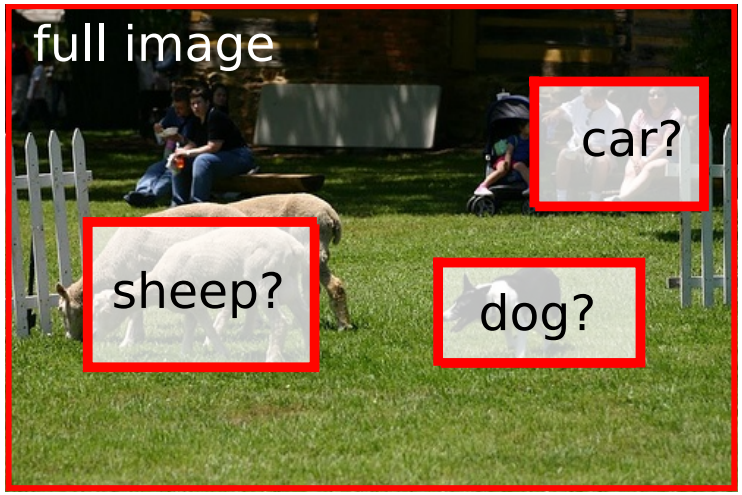
We propose a system that performs *joint object detection*.
It can use the information of

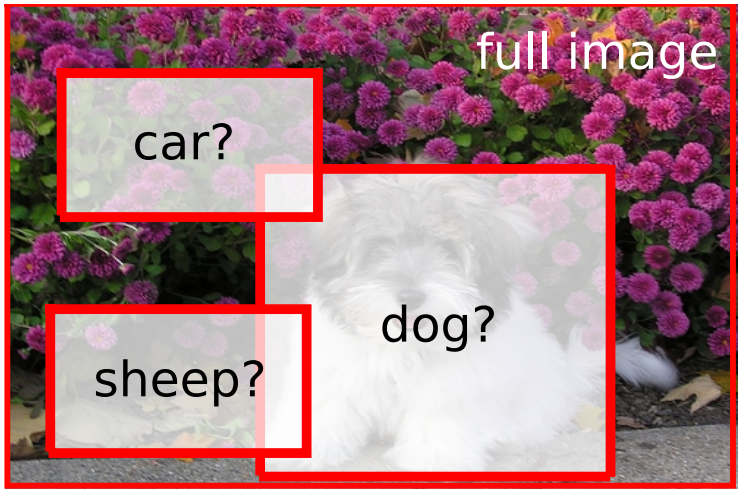
- *all candidate objects regions (co-occurance)*,
- *the full image (context)*.

Joint Training for Multi-Class Detection

Pattern Recognition Setup:

- Training images: $I_1, \ldots, I_n$
- Candidate regions for K classes: $R_i^1, \ldots, R_i^K$ for each image,
  - w.l.o.g. at most one object region per class per image
- Represent each image or region by a feature vector $x_i^k \in \mathcal{X}^k$.
- Labels $y_i^k = \begin{cases} +1 & \text{if } R_i^k \text{ is really of class } k \\ -1 & \text{otherwise, set } R_i^k \text{ to a background region.} \end{cases}$

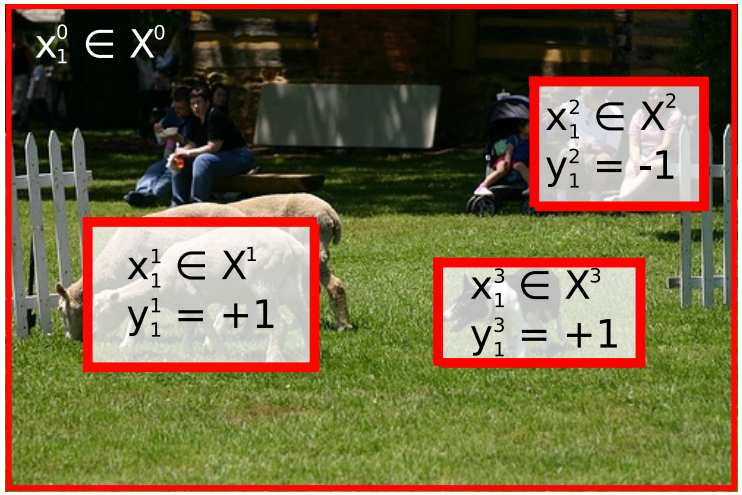


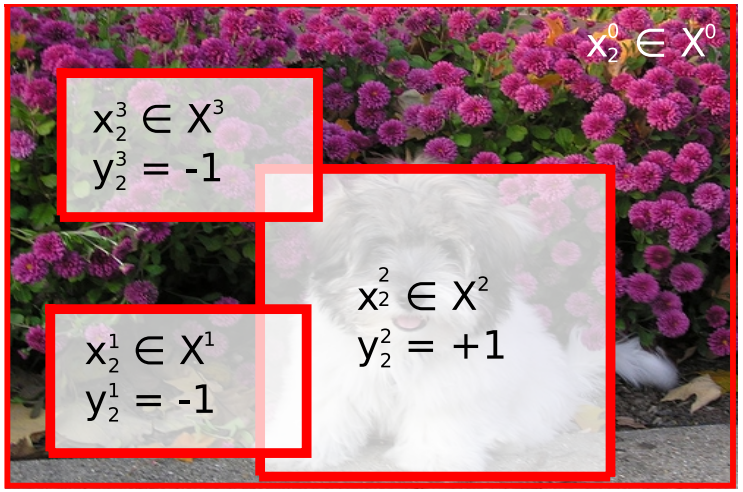


Joint Training for Multi-Class Detection

Pattern Recognition Setup:

- Training images: $I_1, \ldots, I_n$
- Candidate regions for K classes: $R_i^1, \ldots, R_i^K$ for each image,
  - w.l.o.g. at most one object region per class per image
- Represent each image or region by a feature vector $x_i^k \in \mathcal{X}^k$.
- Labels $y_i^k = \begin{cases} +1 & \text{if } R_i^k \text{ is really of class } k \\ -1 & \text{otherwise, set } R_i^k \text{ to a background region.} \end{cases}$



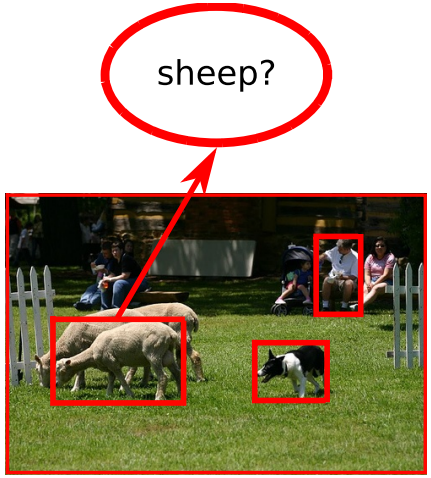


Joint Training for Multi-Class Detection

Separate Per-Class Training

Train classifiers $f_k : \mathcal{X}^k \to \mathbb{R}$, one per class.

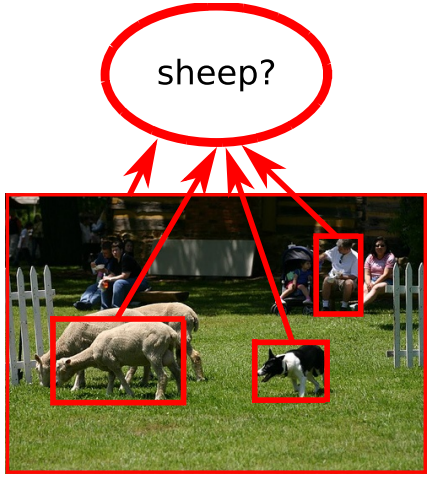
- input: candidate region for class k,
- output: confidence that the region is of object class k.



Context-Aware Joint Training

Train classifiers $\boldsymbol{f_k : \mathcal{X}^0 \times \cdots \times \mathcal{X}^K \to \mathbb{R}}$, one per class.

- input: candidate regions for *all classes* $k = 1, \ldots, K$ and the full image ($k = 0$),
- output: confidence that region k is of object class k.



Multi-Input SVM

- We model each f_k as a multi-input *support vector machine*:

$$f^k : \mathcal{X}^0 \times \cdots \times \mathcal{X}^K \to \mathbb{R}, \qquad \text{for } k = 1, \ldots, K.$$

- Define a joint kernel function

$$k^k(\mathcal{X}^0 \times \cdots \times \mathcal{X}^K) \times (\mathcal{X}^0 \times \cdots \times \mathcal{X}^K) \to \mathbb{R}.$$

$$k^k(\,(x^0, \ldots, x^K), (x'^0, \ldots, x'^K)\,) = \sum_{j=0}^{K} \beta_j^k \kappa_j(\,x^j, x'^j\,),$$

  where the κ_j are base kernels between image regions.

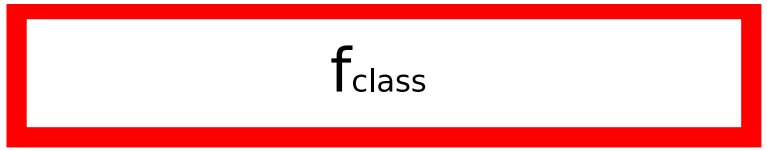
- Class dependent *mixing coefficients*: $\beta_j^k \in [0, 1]$, $\sum_{j=0}^{K} \beta_j^k = 1$.
- The trained f_k has the form

$$f^k(x^0, \ldots, x^K) = \sum_{j=0}^{K} \beta_j^k \sum_{i=0}^{n} y_i^k \alpha_i^k \kappa_j(\,x^j, x_i^j\,).$$

Mixing coefficients

The trained f_k has the form

$$f^k(x^0, \ldots, x^K) = \sum_{j=0}^{K} \beta_j^k \sum_{i=0}^{n} y_i^k \alpha_i^k \kappa_j(\, x^j, x_i^j \,).$$



Mixing coefficients β_j^k are crucial:

- $\beta_j^k = \delta_{k,j} \quad \Leftrightarrow$
  f_k depends only on x^k.
- $\beta_j^k \neq 0 \quad \Rightarrow$
  region x^j influences f_k
- The best choice of β_j^k is
  class and data dependent.

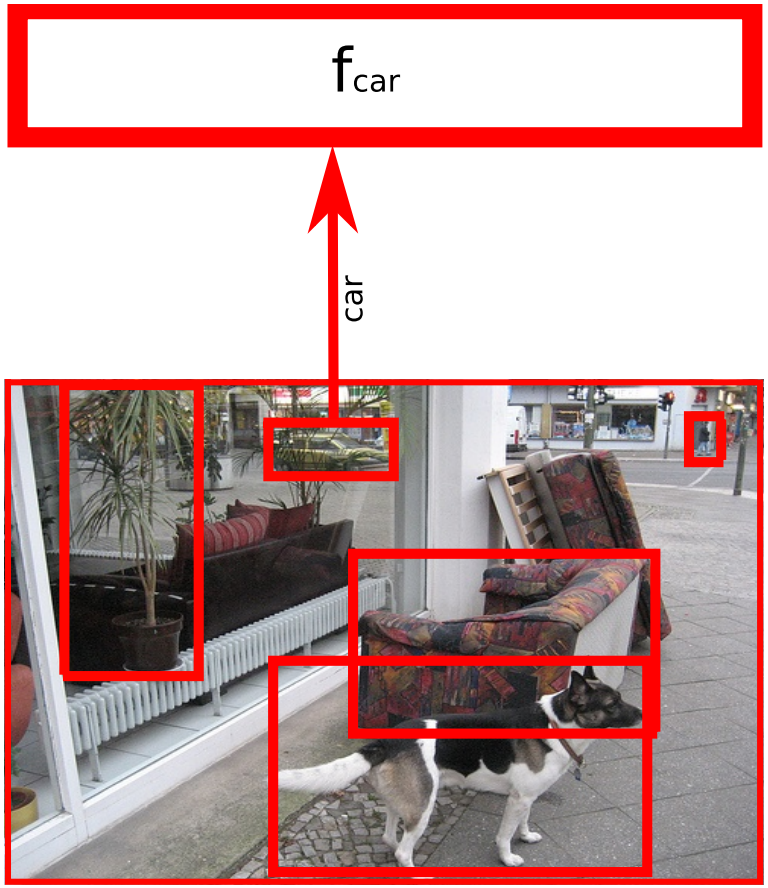
Mixing coefficients

The trained f_k has the form

$$f^k(x^0, \ldots, x^K) = \sum_{j=0}^{K} \beta_j^k \sum_{i=0}^{n} y_i^k \alpha_i^k \kappa_j(\, x^j, x_i^j \,).$$

Mixing coefficients β_j^k are crucial:

- $\beta_j^k = \delta_{k,j} \quad \Leftrightarrow$
  f_k **depends only on** x^k**.**
- $\beta_j^k \neq 0 \quad \Rightarrow$
  region x^j influences f_k
- The best choice of β_j^k is
  class and data dependent.



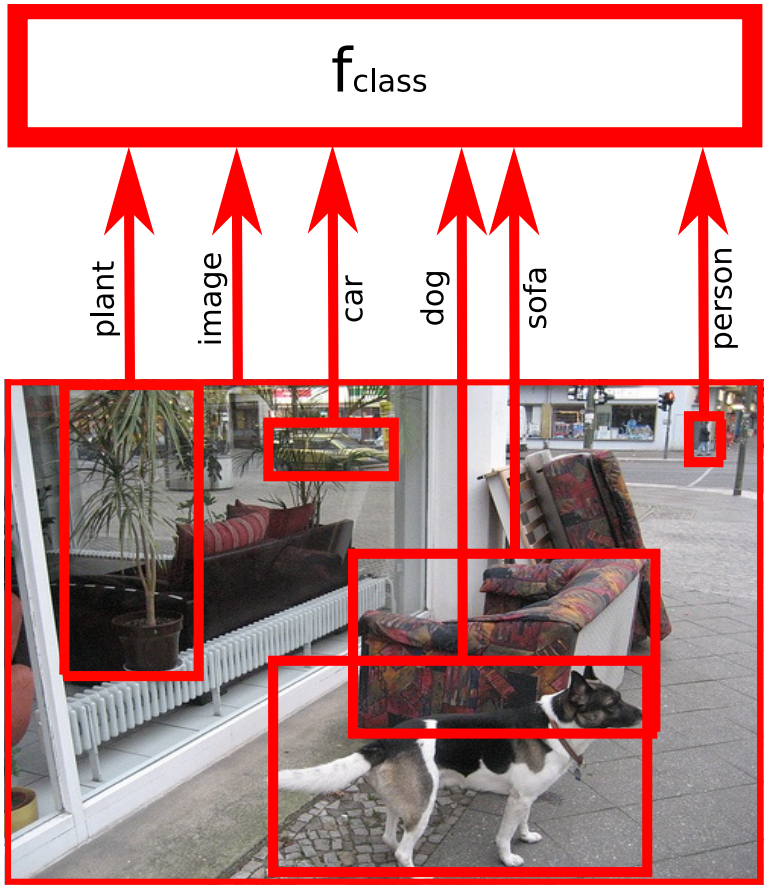
Mixing coefficients

The trained f_k has the form

$$f^k(x^0, \ldots, x^K) = \sum_{j=0}^{K} \beta_j^k \sum_{i=0}^{n} y_i^k \alpha_i^k \kappa_j(\, x^j, x_i^j \,).$$

Mixing coefficients β_j^k are crucial:

- $\beta_j^k = \delta_{k,j} \quad \Leftrightarrow$
  f_k depends only on x^k.
- $\beta_j^k \neq 0 \quad \Rightarrow$
  region x^j **influences** f_k
- The best choice of β_j^k is
  class and data dependent.



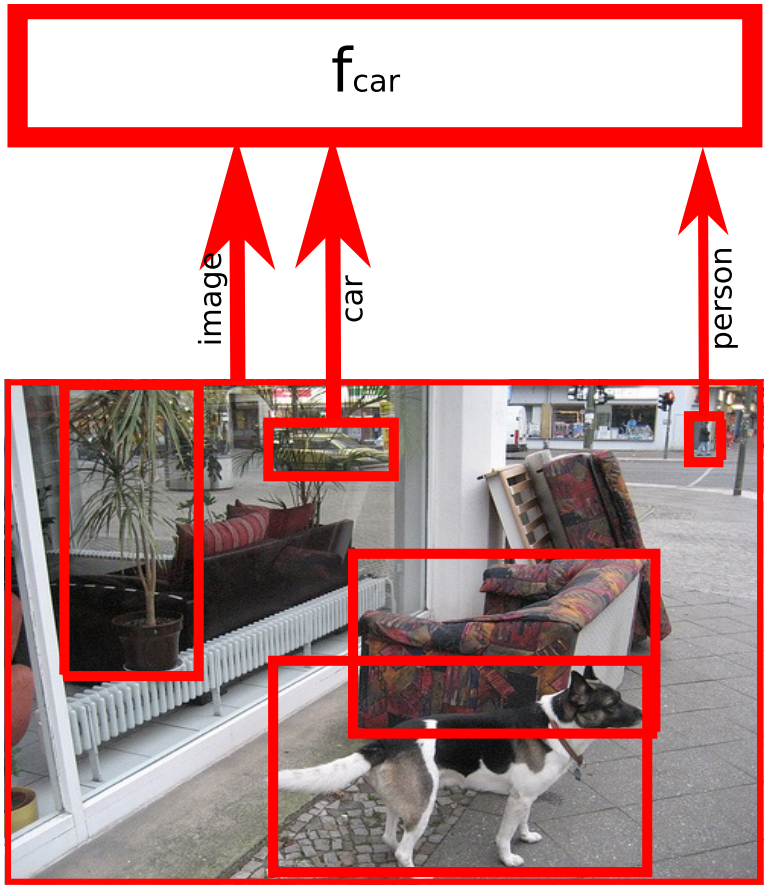
Mixing coefficients

The trained f_k has the form

$$f^k(x^0, \ldots, x^K) = \sum_{j=0}^{K} \beta_j^k \sum_{i=0}^{n} y_i^k \alpha_i^k \kappa_j(\, x^j, x_i^j \,).$$

Mixing coefficients β_j^k are crucial:

- $\beta_j^k = \delta_{k,j} \quad \Leftrightarrow$
  f_k depends only on x^k.
- $\beta_j^k \neq 0 \quad \Rightarrow$
  region x^j influences f_k
- **The best choice of β_j^k is *class and data dependent*.**



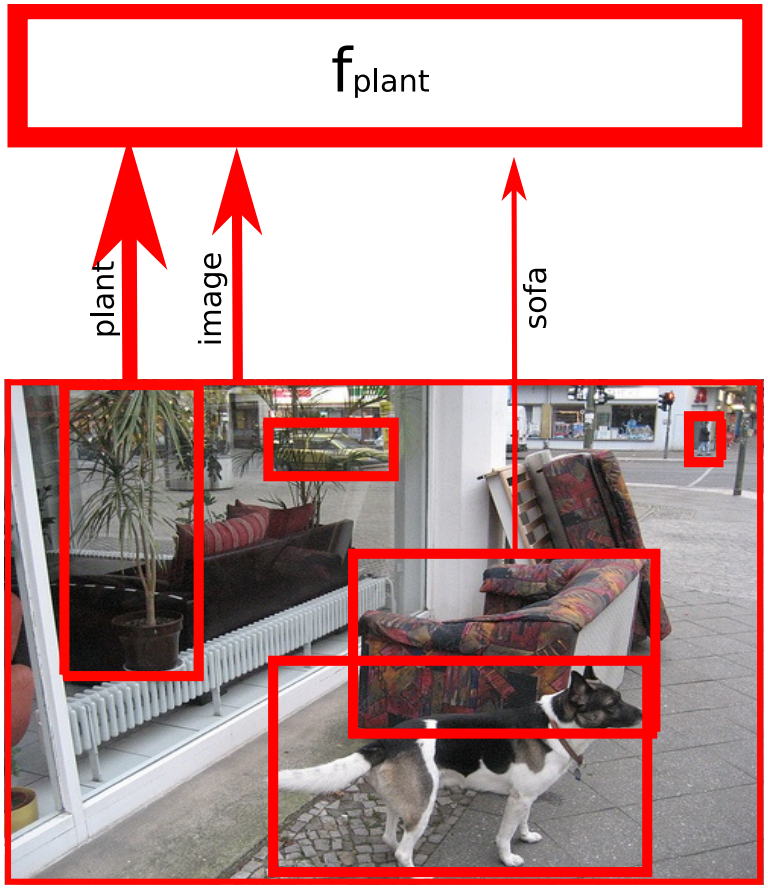
Mixing coefficients

The trained f_k has the form

$$f^k(x^0, \ldots, x^K) = \sum_{j=0}^{K} \beta_j^k \sum_{i=0}^{n} y_i^k \alpha_i^k \kappa_j(\, x^j, x_i^j \,).$$

Mixing coefficients β_j^k are crucial:

- $\beta_j^k = \delta_{k,j} \quad \Leftrightarrow$
  f_k depends only on x^k.
- $\beta_j^k \neq 0 \quad \Rightarrow$
  region x^j influences f_k
- **The best choice of β_j^k is *class and data dependent*.**



Choices for the mixing coefficients

We can set β_j^k based on *a priori* knowledge:

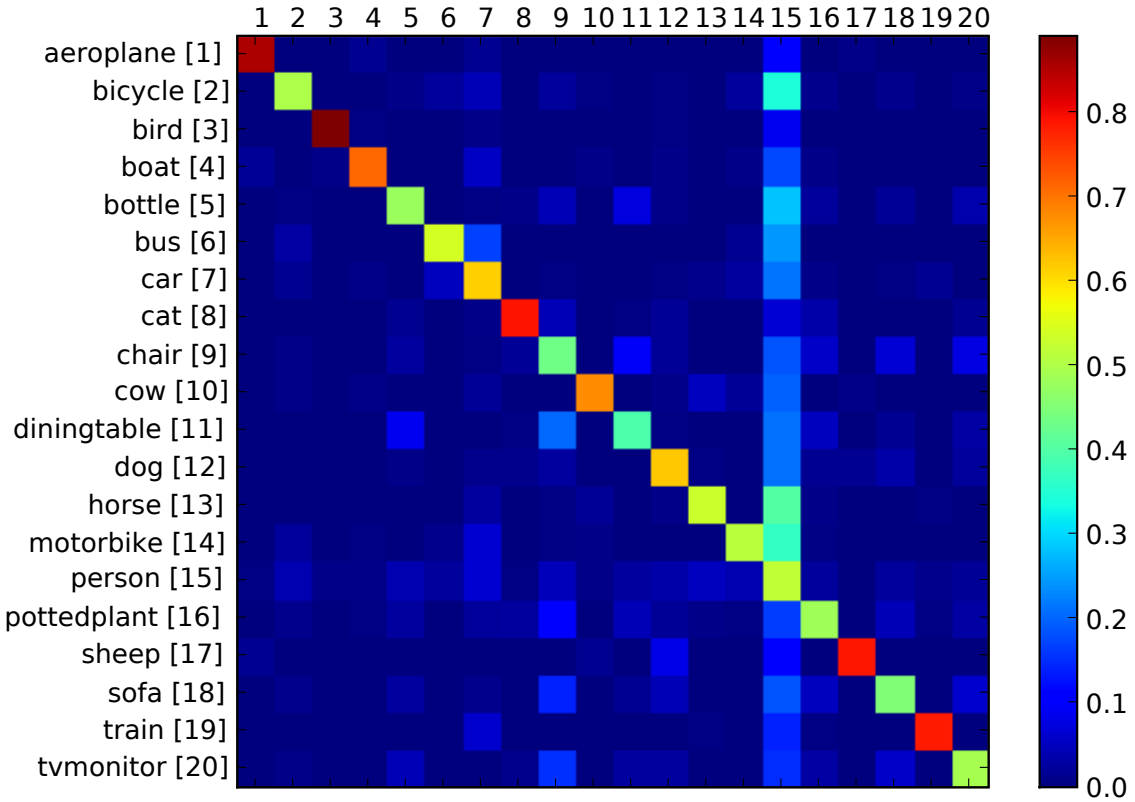
- *chairs* and *tables* tend to co-occur in images.
- *horses* and *persons* tend to co-occur in images.
- *cars* and *busses* tend to co-occur in images.
- ...

Problems:

- Requires human interaction.
- Requires semantic knowledge about the classes.

Choices for the mixing coefficients

We can set β_j^k based on (relative) *co-occurrence* in a dataset:

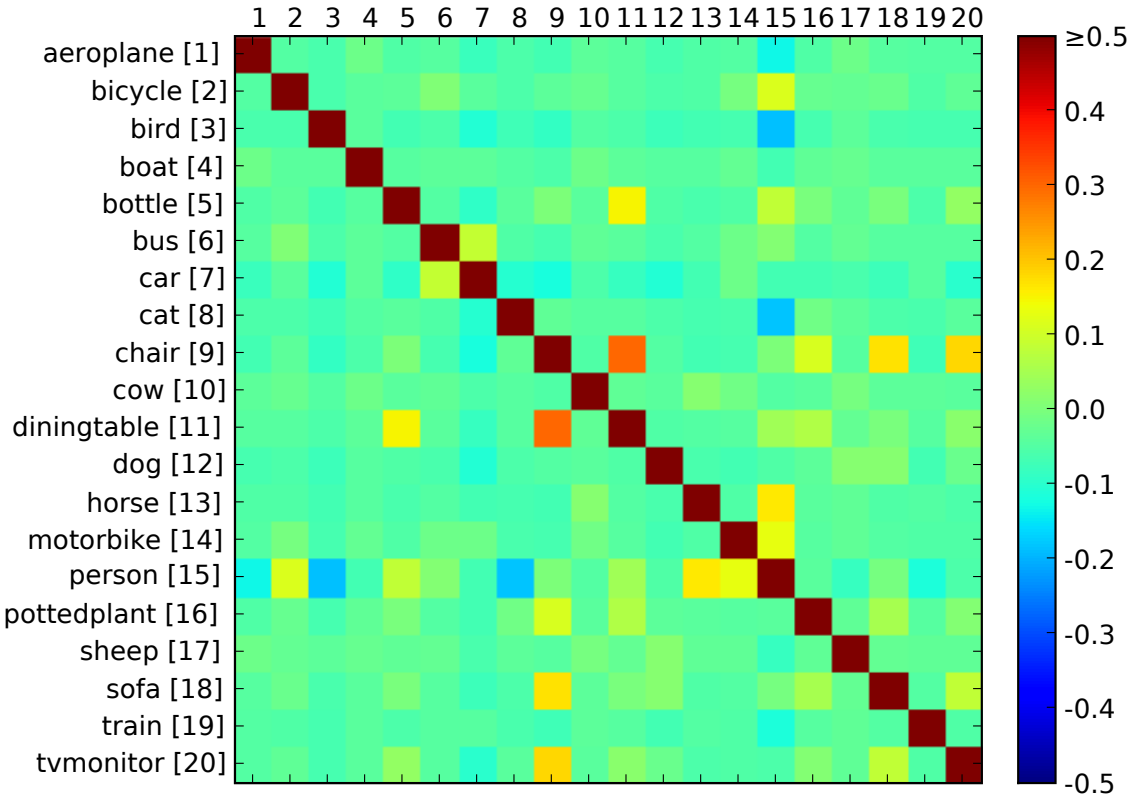


Problems:

- We discard *negative* dependencies: if e.g. *cats* and *aeroplanes* never co-occur, this can be used for prediction.

Choices for the mixing coefficients

We can set β_j^k based on class *correlation coefficients*:



Problems:

- This is generative, *a priori* information about the dataset.
- We need *discriminative* information about the base *classifiers*.

Learn the mixing coefficients

Learn the optimal β_j^k directly from training data.

- SVM optimization problem for *fixed* β

$$\min_{w_j \in \mathcal{H}_j,\ \xi_i^k \geq 0} \quad \frac{1}{2} \sum_{j=0}^{K} \frac{1}{\beta_j^k} \| w_j^k \|_{\mathcal{H}_j}^2 + C \sum_{i=1}^{n} \xi_i^k$$

subject to

$$y_i^k \sum_j \langle w_j^k, \varphi_j(x_i^j) \rangle_{\mathcal{H}_j} \geq 1 - \xi_i^k \qquad \text{for } i = 1, \ldots n.$$

- Convex problem in w_j^k, ξ_i^k.

Learn the mixing coefficients

Learn the optimal β_j^k directly from training data.

- SVM optimization problem including β [Zien and Ong, 2007]:

$$\min_{\substack{w_j \in \mathcal{H}_j,\ \xi_i^k \geq 0 \\ \beta_j^k \in [0,1] \\ \sum_j \beta_j^k = 1}} \frac{1}{2} \sum_{j=0}^{K} \frac{1}{\beta_j^k} \| w_j^k \|_{\mathcal{H}_j}^2 + C \sum_{i=1}^{n} \xi_i^k$$

subject to

$$y_i^k \sum_j \langle w_j^k, \varphi_j(x_i^j) \rangle_{\mathcal{H}_j} \geq 1 - \xi_i^k \qquad \text{for } i = 1, \ldots n.$$

- Jointly convex in w_j^k, ξ_i^k and β_j^k: **Multiple-Kernel Learning**

Summary of Joint Multi-Object Learning

Training Setup

- Images x_i^0, object regions x_i^k, per-region labels y_i^k, $k = 1, \ldots, K$, $i = 1, \ldots, n$.

Multiple Kernel Learning (MKL)

- Learn coefficients α_i^k and β_j^k for classifiers

$$f^k(x^0, x^1, \ldots, x^K) = \sum_{j=0}^{K} \beta_j^k \sum_{i=0}^{n} y_i^k \alpha_i^k \kappa_j(\, x^j, x_i^j \,).$$

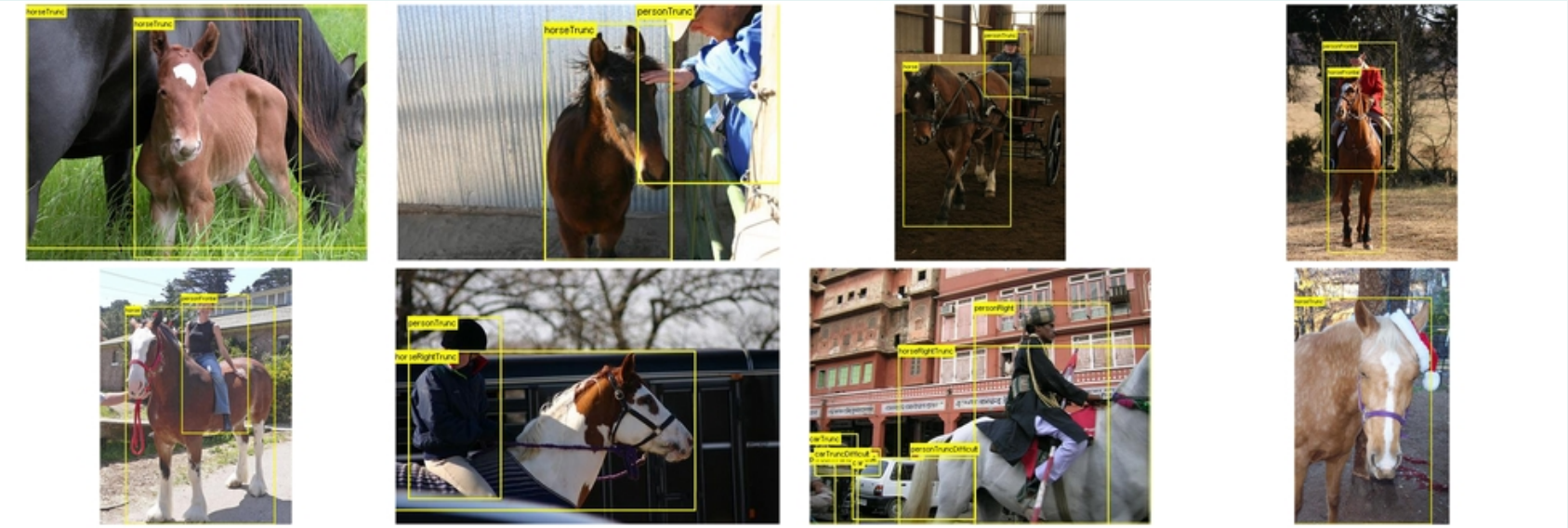
Evaluation Setup

- Image x'^0, predict candidate regions x'^k, $k = 1, \ldots, K$.
- Decide each region x'^k based on $\operatorname{sign} f^k(x'^0, x'^1, \ldots, x'^K)$.

Experimental Evaluation: PASCAL VOC

PASCAL VOC 2006

- Natural scenes, 5,304 images, 9,507 objects, 10 classes.
- Multiple classes, different viewpoints, truncations, occlusions



PASCAL VOC 2007

- Natural scenes, 9,963 images, 24,640 objects, 20 classes
- Multiple classes, very high intra-class variance

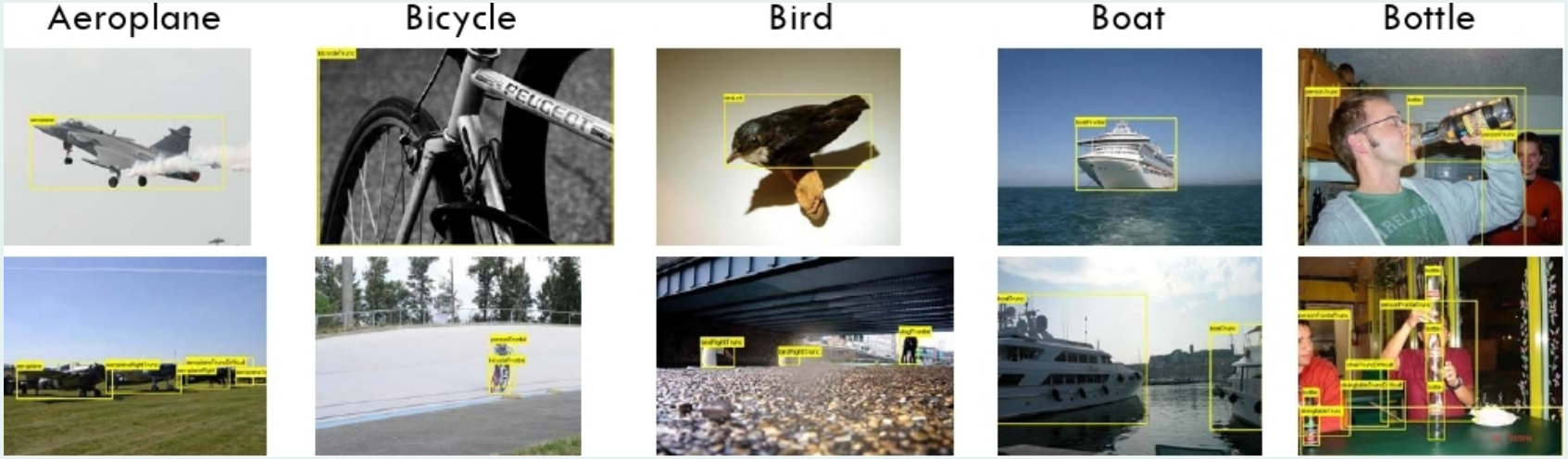
Experimental Evaluation: PASCAL VOC

PASCAL VOC 2006

- Natural scenes, 5,304 images, 9,507 objects, 10 classes.
- Multiple classes, different viewpoints, truncations, occlusions

PASCAL VOC 2007

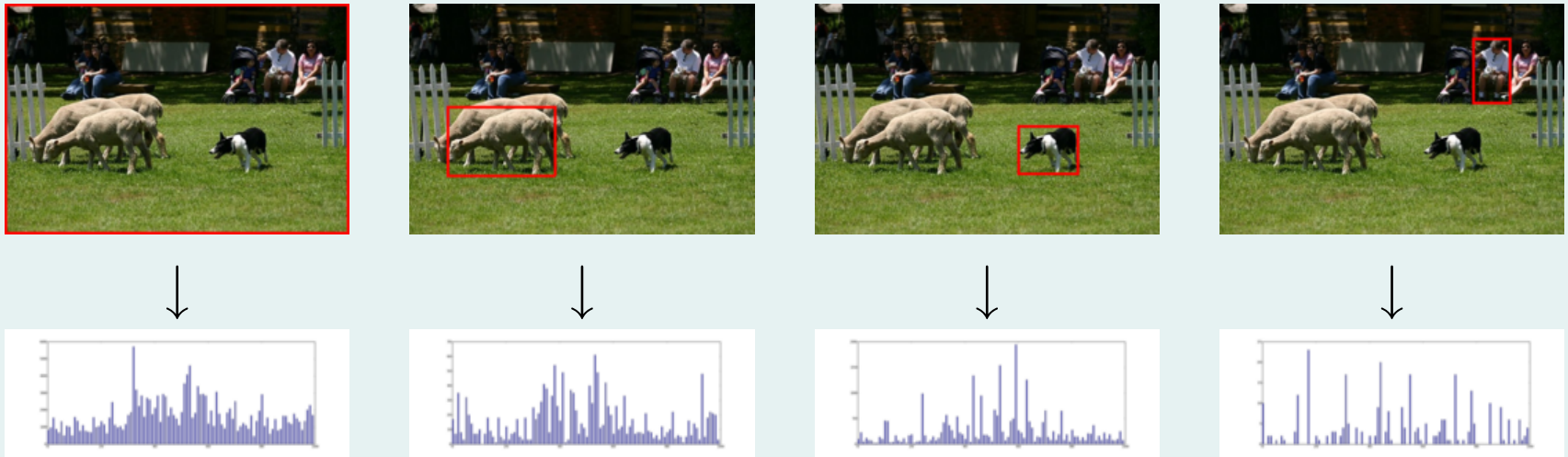
- Natural scenes, 9,963 images, 24,640 objects, 20 classes
- Multiple classes, very high intra-class variance



Experimental Evaluation: PASCAL VOC

Bag-of-Feature Representation

- Extract local SURF descriptors at keypoint locations
- Quantize the descriptors using a visual codebook
- Represent each image or image region by the histogram of codebook IDs

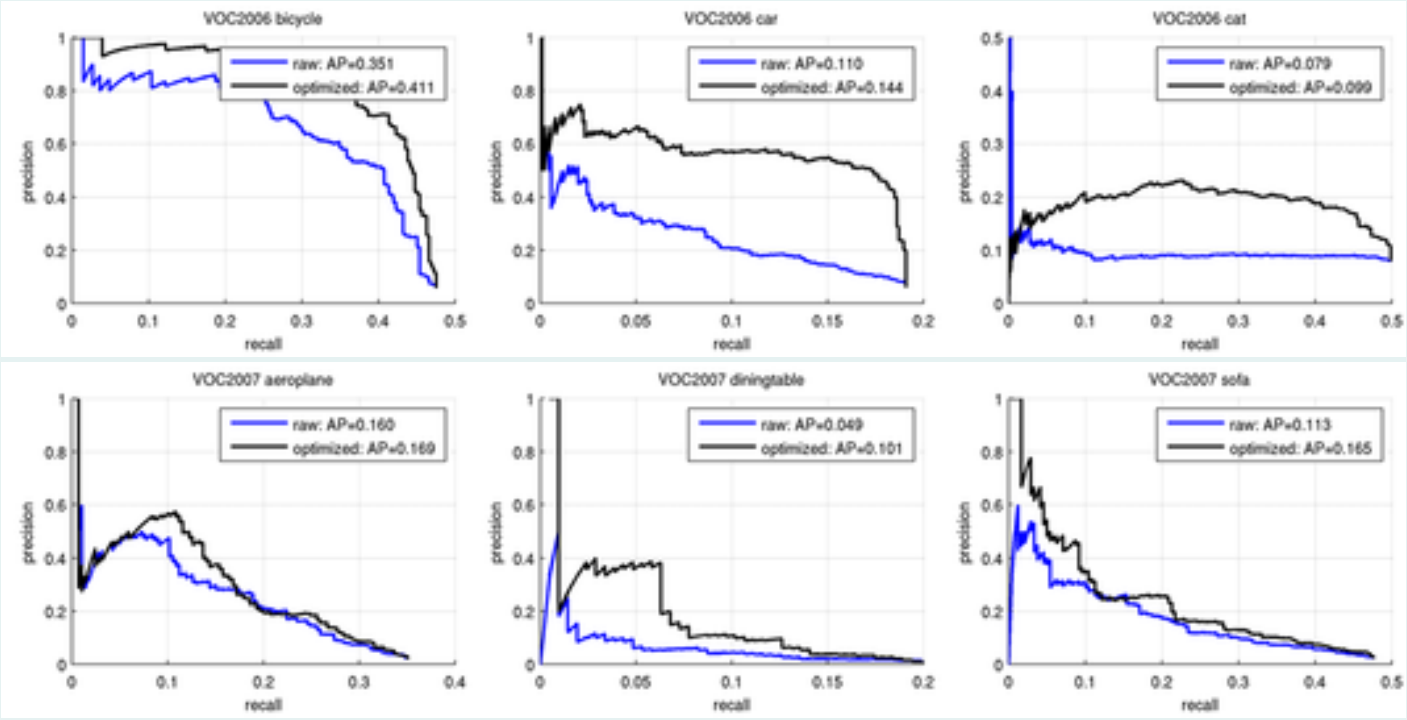


- Define χ^2 base kernels: $\kappa_k(h, h') = \exp\Big(-\frac{1}{\gamma_k}\sum_j \frac{(h_j - h'_j)^2}{|h_j + h'_j|}\Big)$.

Experimental Evaluation: PASCAL VOC

Evaluation

- Predict candidate regions using a sliding window approach.
- Rank detections over all test image using the MKL-SVM.
- Evaluate by precision-recall curves:

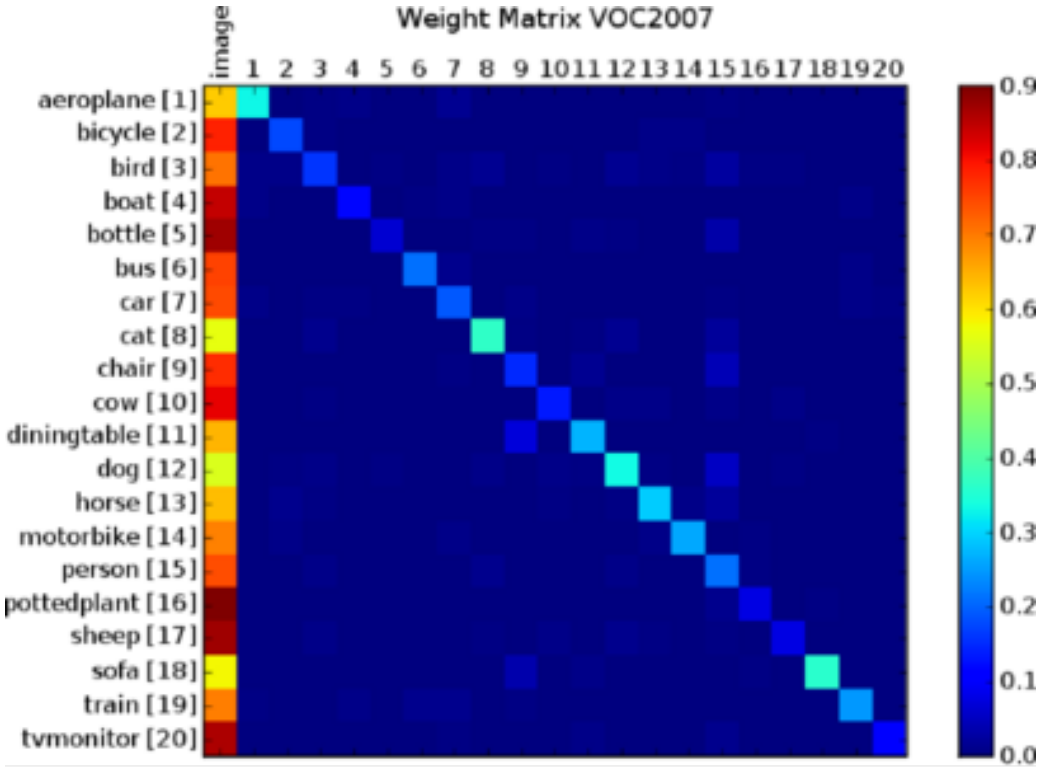


- Joint learning improves detection accuracy (black vs. blue).

Interpretation of Learned Weights

How do the weights β_j^k learned by MKL look like?

- Every class depends on *object box*.
- High *image* weights: $\longrightarrow$ *scene* classification?
- Intuitive connections:
  chair $\longrightarrow$ *diningtable*,
  person $\longrightarrow$ *bottle*,
  person $\longrightarrow$ *dog*.
- Several classes depend on the *person* class.
- β_k^j are sparse.

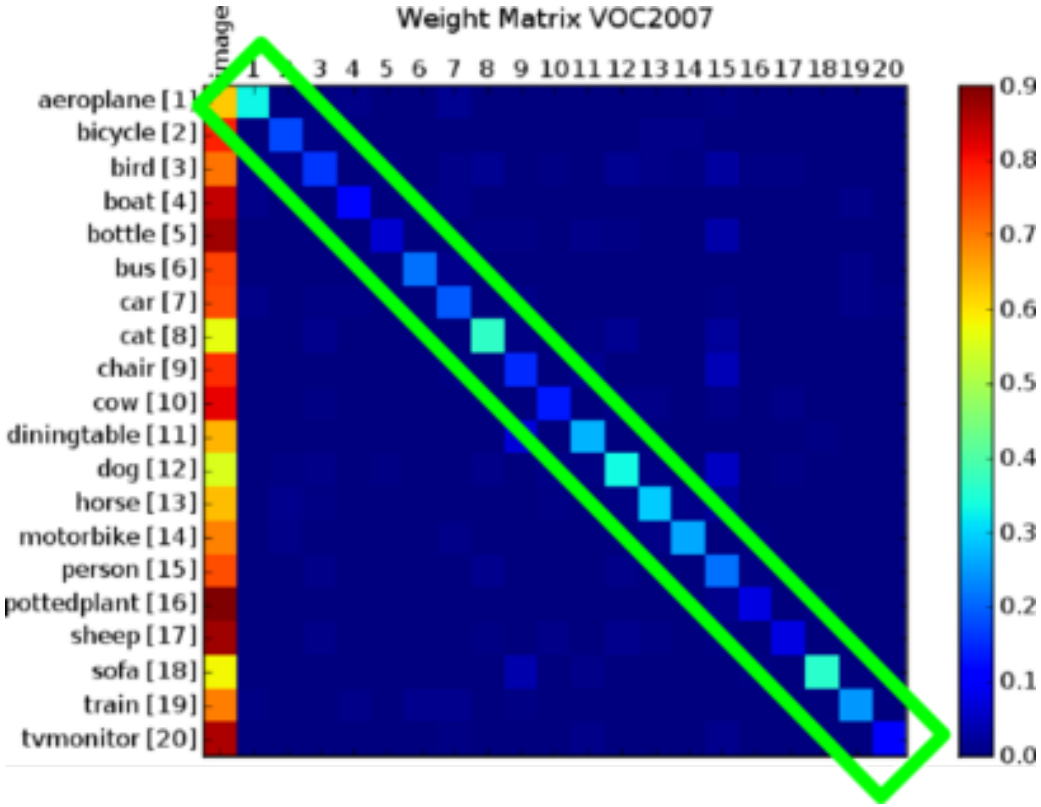


rows: class to be detected
columns: class candidate boxes

Interpretation of Learned Weights

How do the weights β_j^k learned by MKL look like?

- **Every class depends on *object box*.**
- High *image* weights:
  $\longrightarrow$ *scene* classification?
- Intuitive connections:
  chair $\longrightarrow$ *diningtable*,
  person $\longrightarrow$ *bottle*,
  person $\longrightarrow$ *dog*.
- Several classes depend on the *person* class.
- β_k^j are sparse.

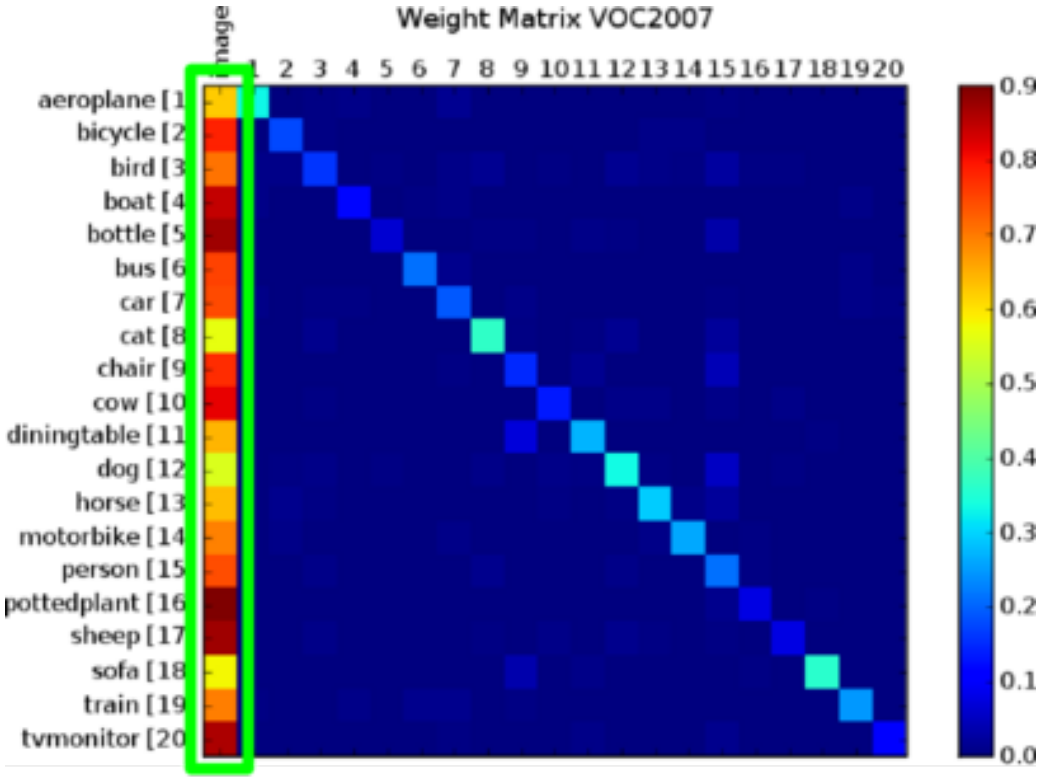


rows: class to be detected
columns: class candidate boxes

Interpretation of Learned Weights

How do the weights β_j^k learned by MKL look like?

- Every class depends on *object box*.
- **High *image* weights:**
  $\longrightarrow$ *scene* classification?
- Intuitive connections:
  chair $\longrightarrow$ *diningtable*,
  person $\longrightarrow$ *bottle*,
  person $\longrightarrow$ *dog*.
- Several classes depend on the *person* class.
- β_k^j are sparse.

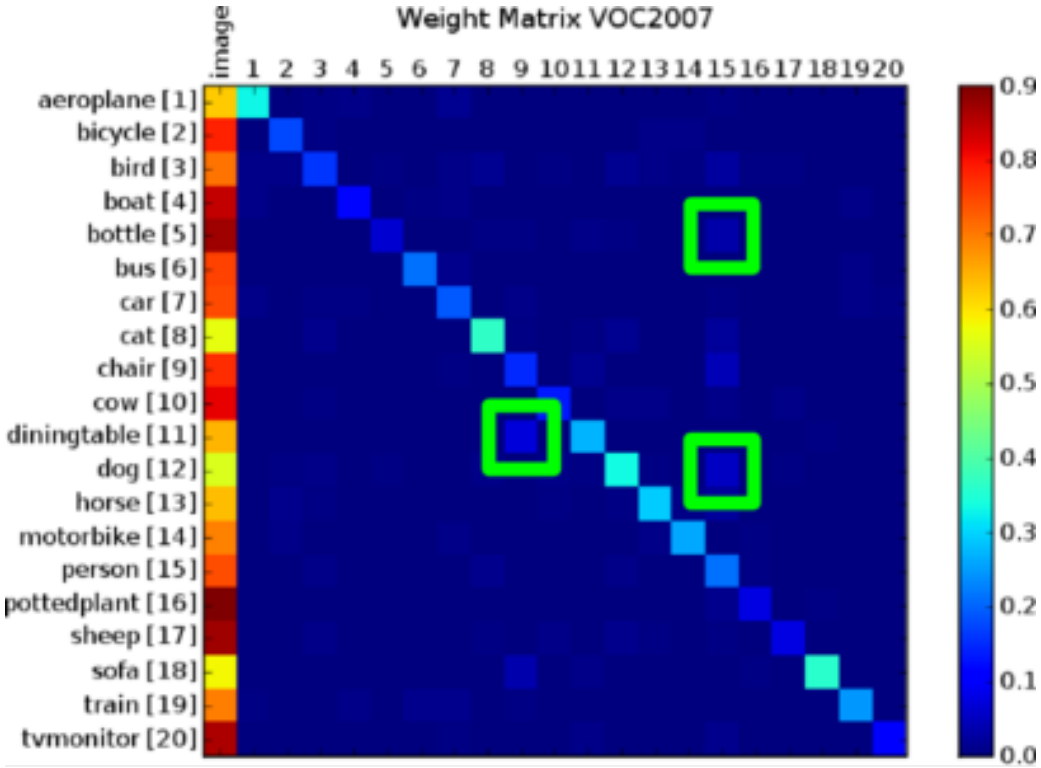


rows: class to be detected
columns: class candidate boxes

Interpretation of Learned Weights

How do the weights β_j^k learned by MKL look like?

- Every class depends on *object box*.
- High *image* weights: $\longrightarrow$ *scene* classification?
- **Intuitive connections:**
  diningtable $\leftarrow$ ***chair***,
  bottle $\leftarrow$ ***person***,
  dog $\leftarrow$ ***person***.
- Several classes depend on the *person* class.
- β_k^j are sparse.

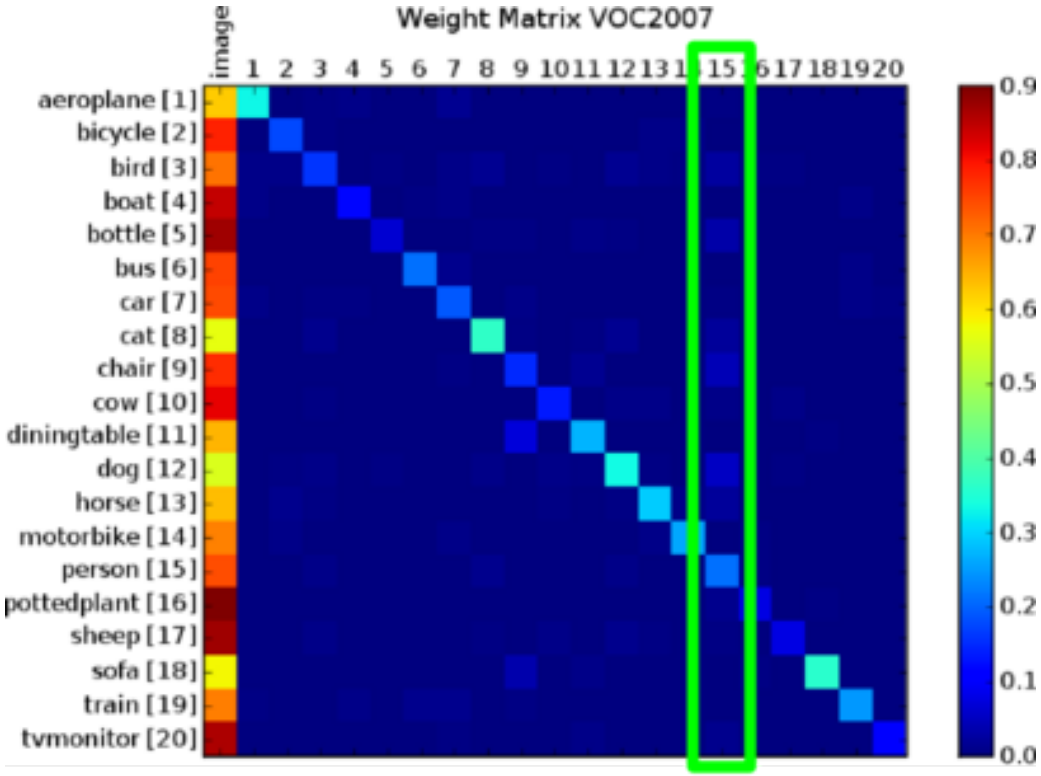


rows: class to be detected
columns: class candidate boxes

Interpretation of Learned Weights

How do the weights β_j^k learned by MKL look like?

- Every class depends on *object box*.
- High *image* weights:
  $\longrightarrow$ *scene* classification?
- Intuitive connections:
  chair $\longrightarrow$ *diningtable*,
  person $\longrightarrow$ *bottle*,
  person $\longrightarrow$ *dog*.
- **Several classes depend on the *person* class.**
- β_k^j are sparse.

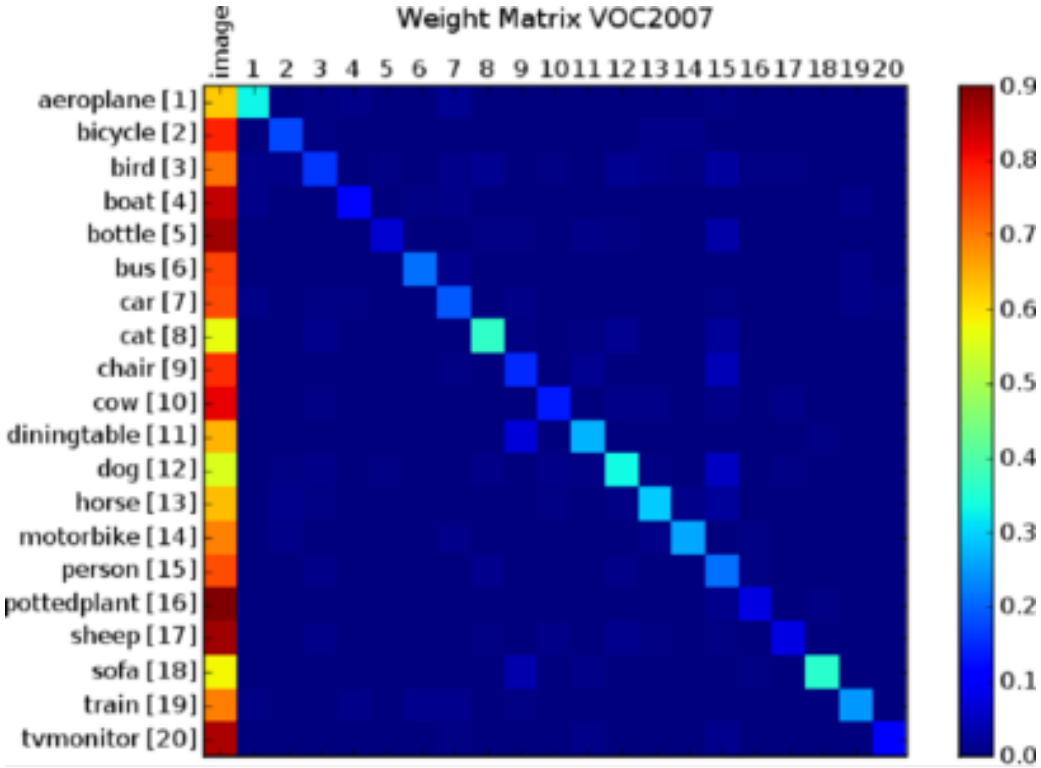


rows: class to be detected
columns: class candidate boxes

Interpretation of Learned Weights

How do the weights β_j^k learned by MKL look like?

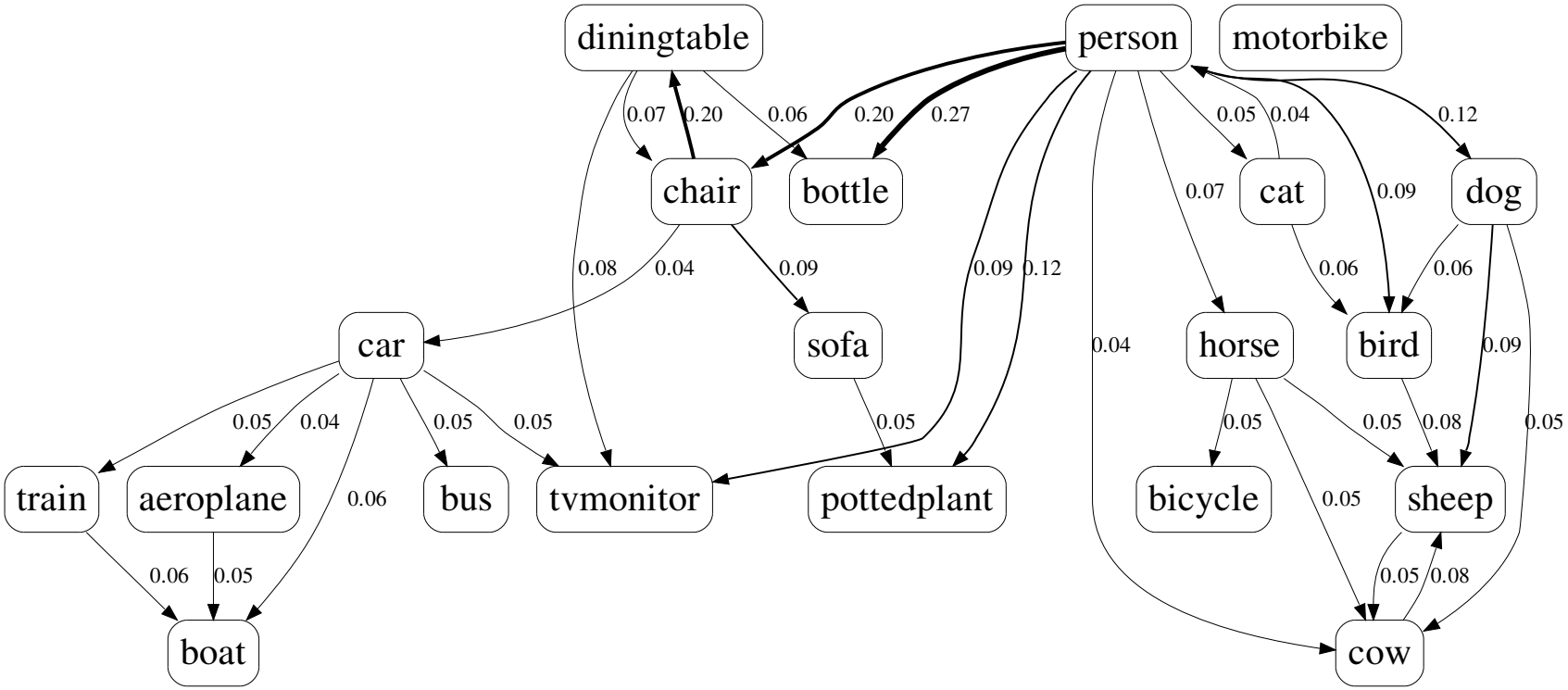
- Every class depends on *object box*.
- High *image* weights: $\longrightarrow$ *scene* classification?
- Intuitive connections:
  chair $\longrightarrow$ *diningtable*,
  person $\longrightarrow$ *bottle*,
  person $\longrightarrow$ *dog*.
- Several classes depend on the *person* class.
- $\boldsymbol{\beta_k^j}$ **are sparse.**



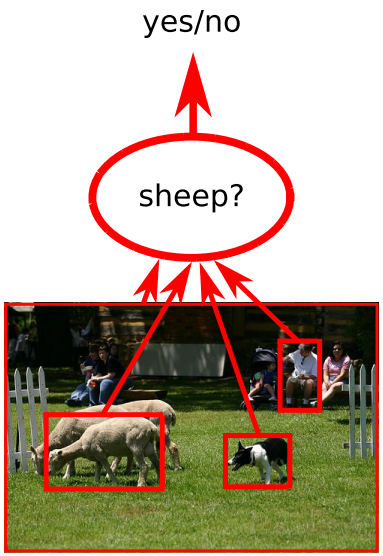
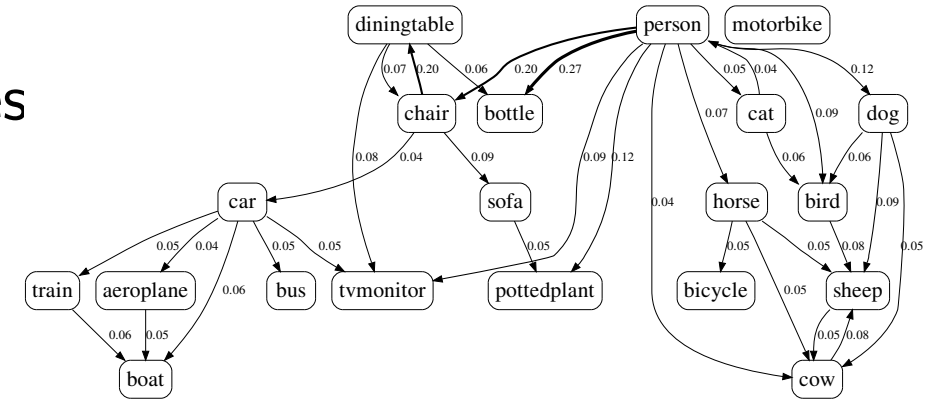
rows: class to be detected
columns: class candidate boxes

Visualization of Class Dependencies

We can turn the non-zero *weights* into a sparse *dependency graph*:



- Relative weights $\beta_k^j / \sum_{j=1}^K \beta_k^j$, thresholded at 0.04
- $i \rightarrow j$ means *"Class i is used to predict class j."*
- Clusters have semantic interpretation: *vehicles*, *indoor*, *animals*.

Summary

- **Joint object detection.** Decision for class j can depend on all other class regions.
- **Multiple Kernel Learning** to learn influence of classes.
- **Improved detection quality.**
- **Interpretable** dependencies between classes.