6.819 / 6.869: Advances in Computer Vision

Learning:

Introduction to Machine Learning for Vision

Website:
http://6.869.csail.mit.edu/fa15/

Instructor: Yusuf Aytar

Lecture TR 9:30AM – 11:00AM
(Room 34-101)

Key Concepts

Concepts

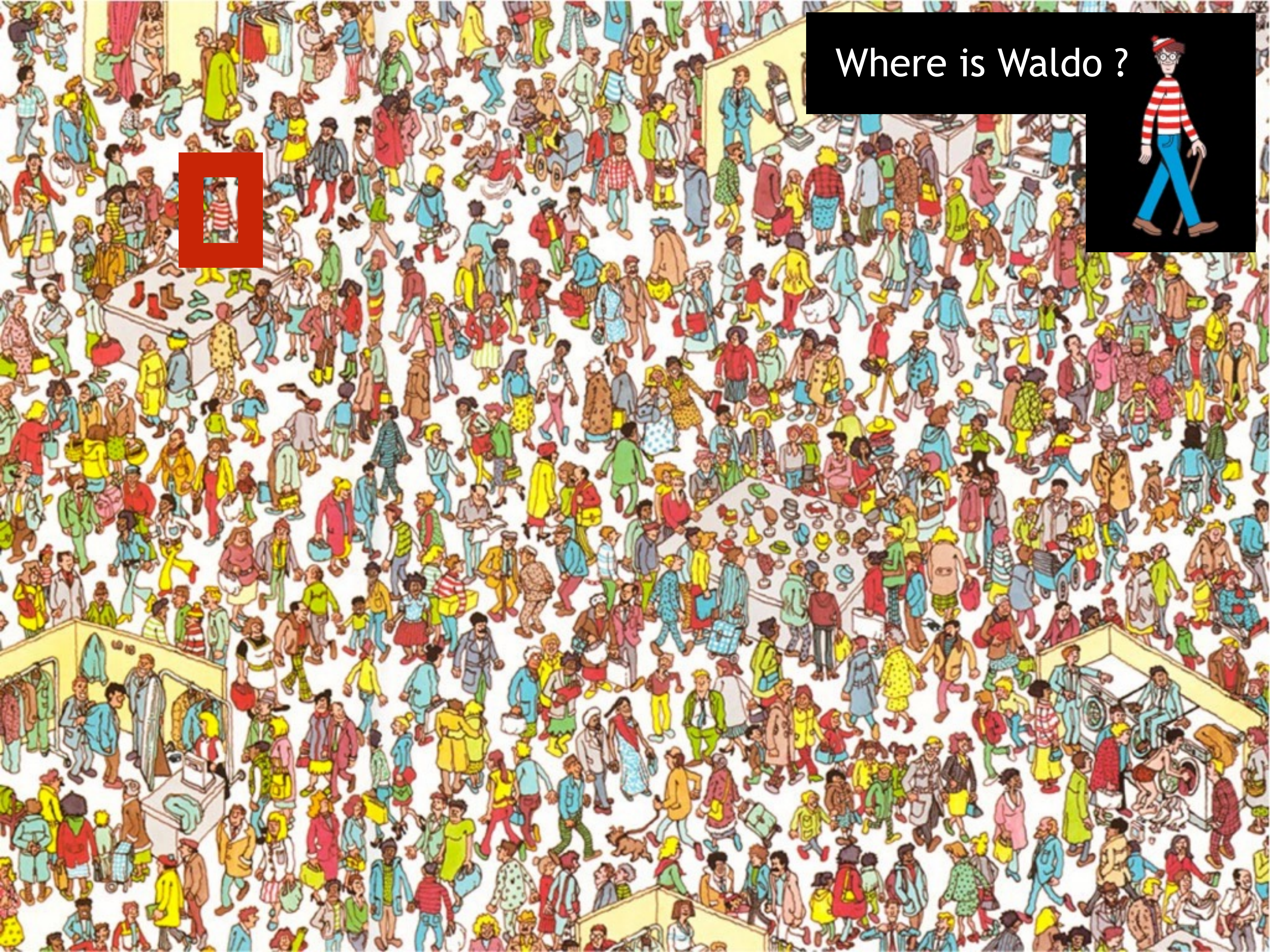
Pattern, Category/Class, Instance, Generalization, Classification

Classification

Feature space, Objective function, Regularization, Loss Function, Optimization

Support Vector Machines

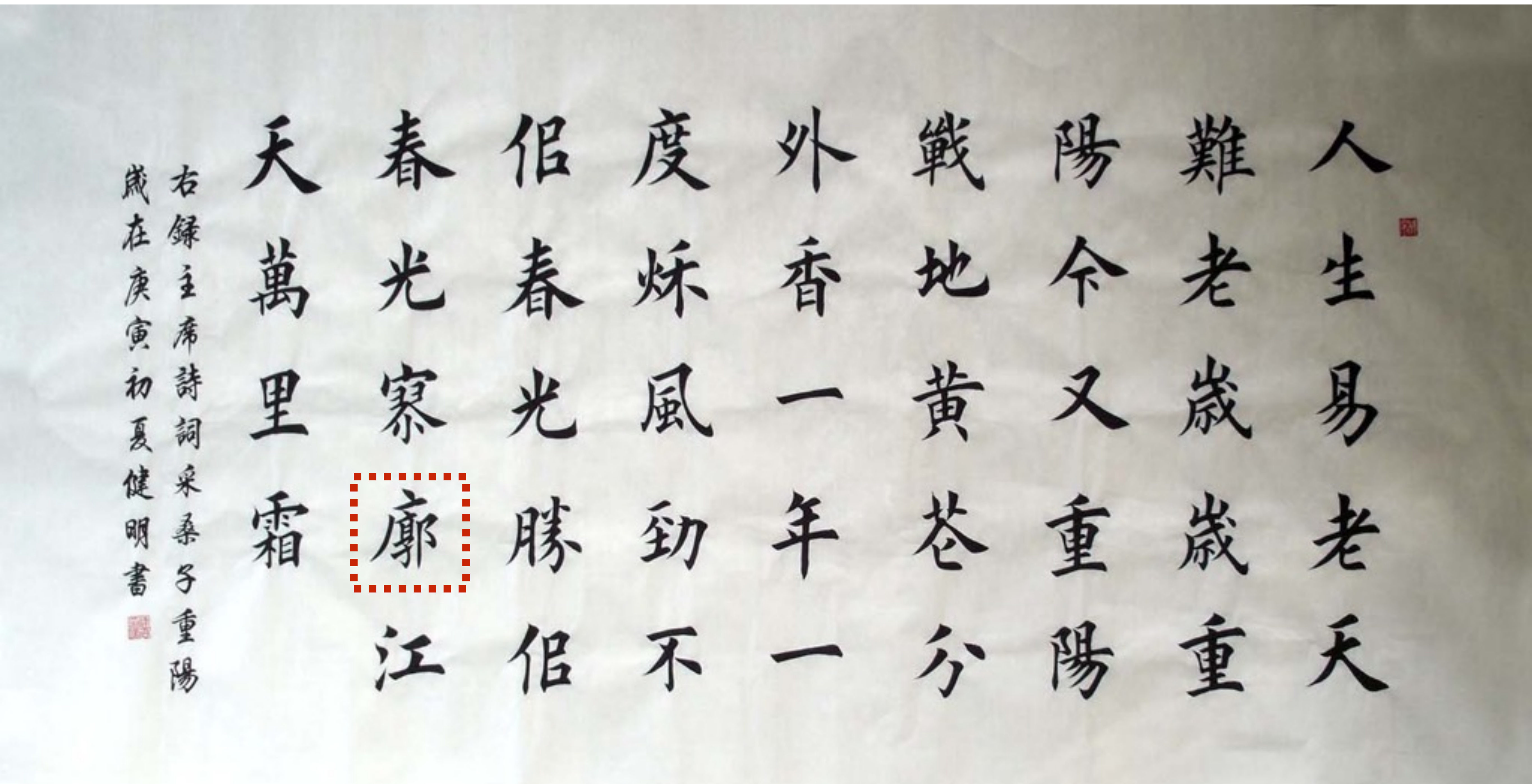
Where is the bottle ?


Where is Waldo ?

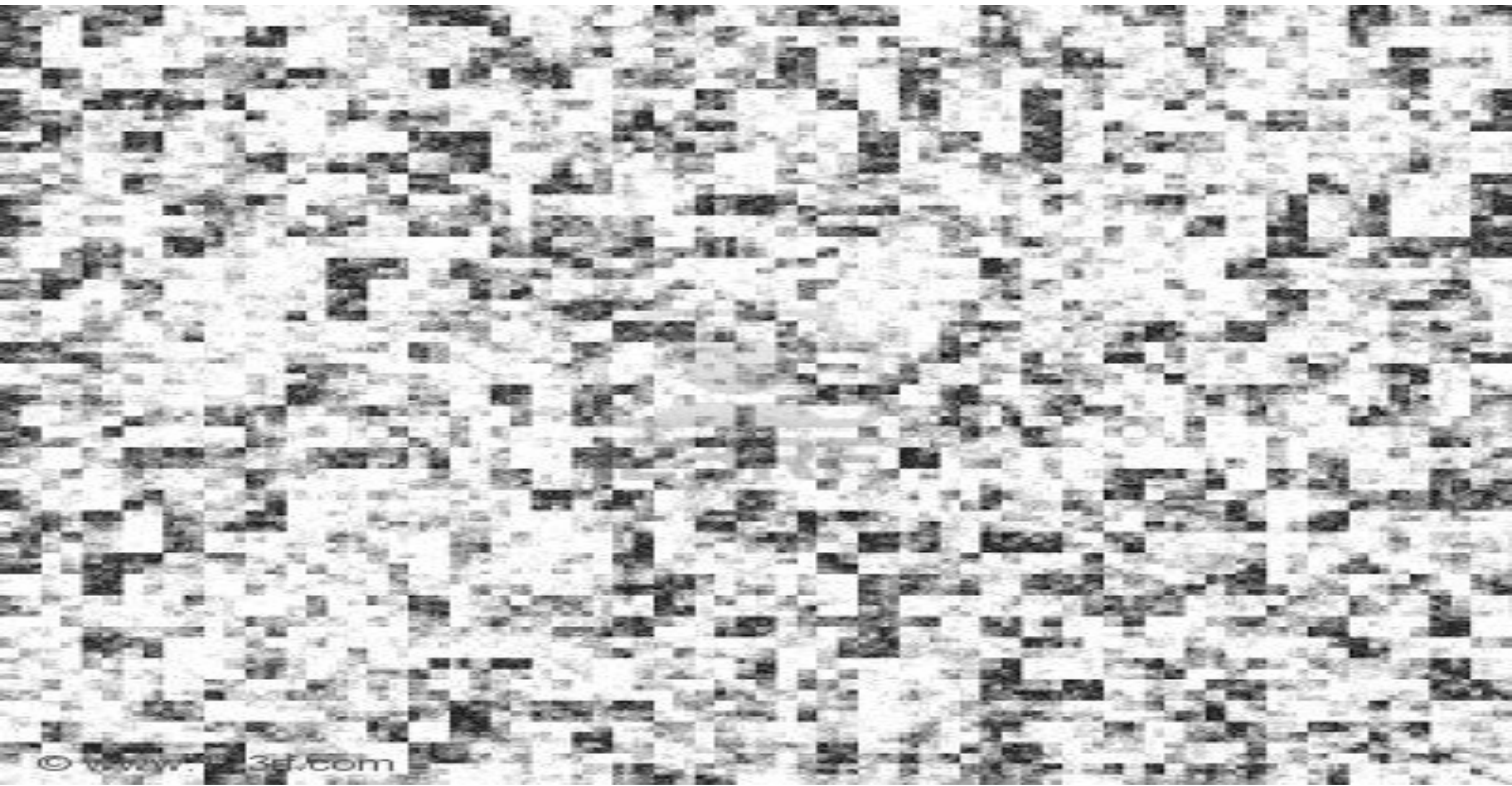
Find the same patch

Task: find the most similar patch in a second image

?


人生易老天
難老歲歲重
陽今又重陽
戰地黃花分
外香一年一
度秌風勁不
佀春光勝佀
春光寥廓江
天萬里霜
右錄主席詩詞采桑子重陽
歲在庚寅初夏健明書

 ?



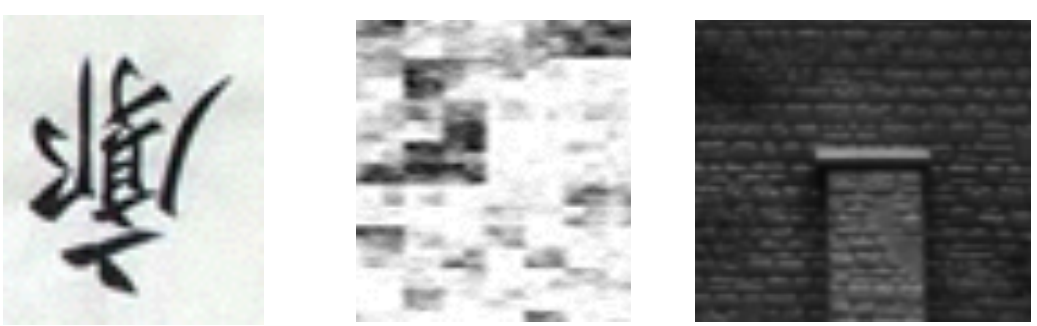
Pattern vs Category

Computers are good with patterns
and
We are good with categories

… but computers are also
getting better with categories

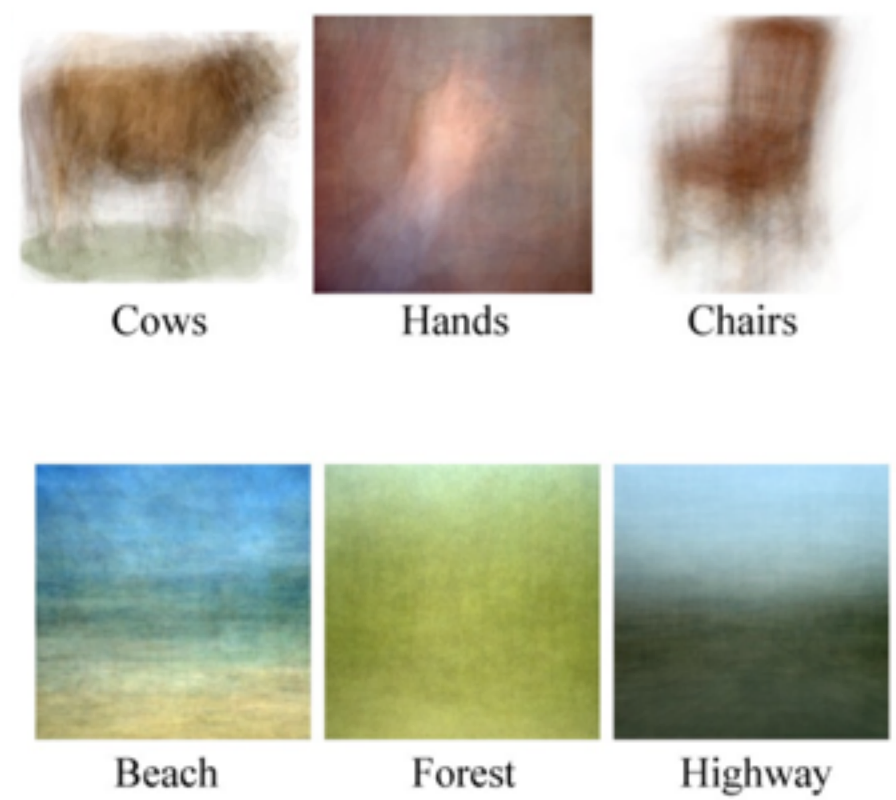
Pattern

Patterns have discriminative representations with less variation



Category

Categories also have discriminative representations, but with great variations



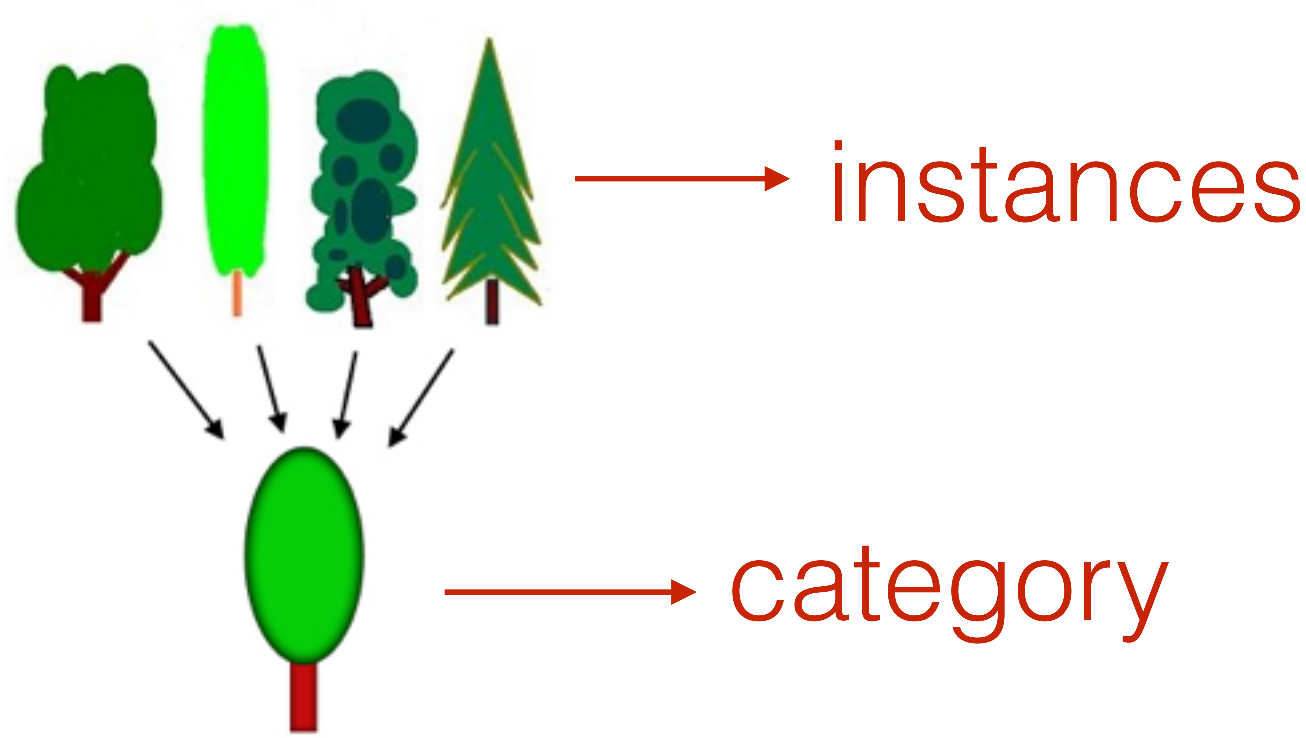
Oliva & Torralba (2007) TICS

Instance vs Category

Instances Find these two objects

Categories Find a bottle:

Generalization



Generalization: Extracting the essence of a concept based on its analysis of similarities from many discrete objects.

http://en.wikipedia.org/wiki/Generalization

Challenges of Generalization

Illumination(Hansen, 2012)

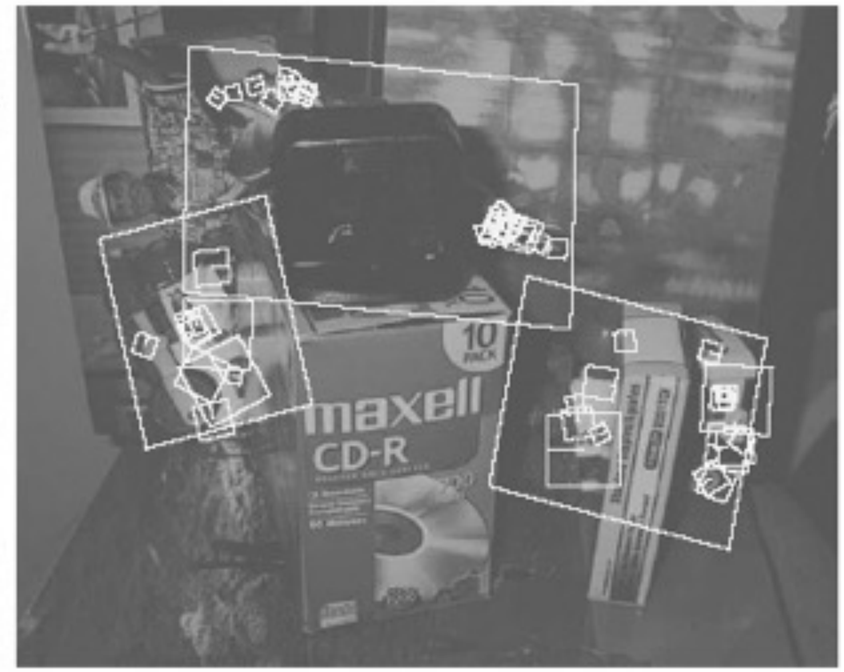
Scale, Viewpoint, Deformation (Xu et al., 2011)

Intra-class variance

A successful object category detector should be invariant to changes in illumination, occlusion, background clutter, scale, viewpoint, deformation and **intra-class variance**.

Object Instance Detection

Find the Object

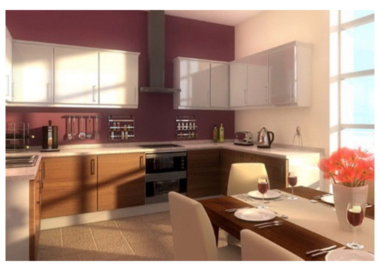


Which of the invariances below apply for the given object instance detection problem?

illumination, occlusion, background clutter, scale, viewpoint, deformation and **intra-class variance**.

Classification vs. Detection

Is this a … image ?

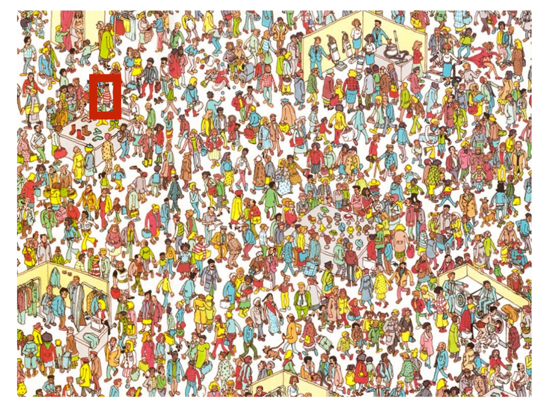


Kitchen

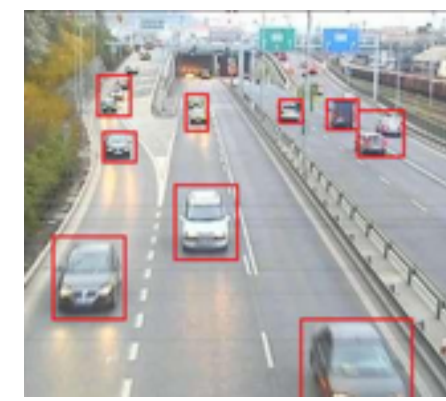
Table

Horse

Where is the …?
Localize the object.



Waldo

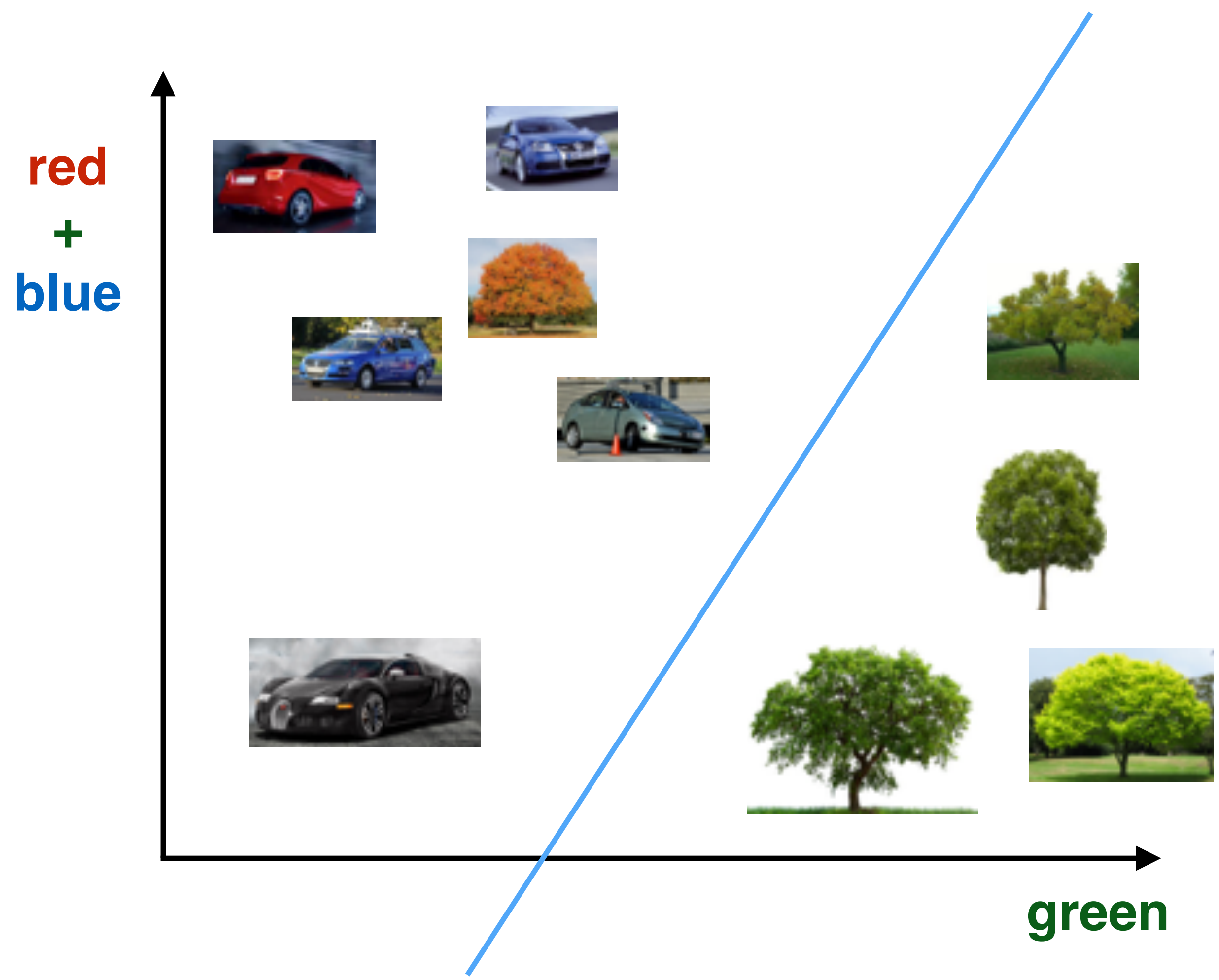


Car

Detection can be performed through a classifier,
i.e. sliding window search

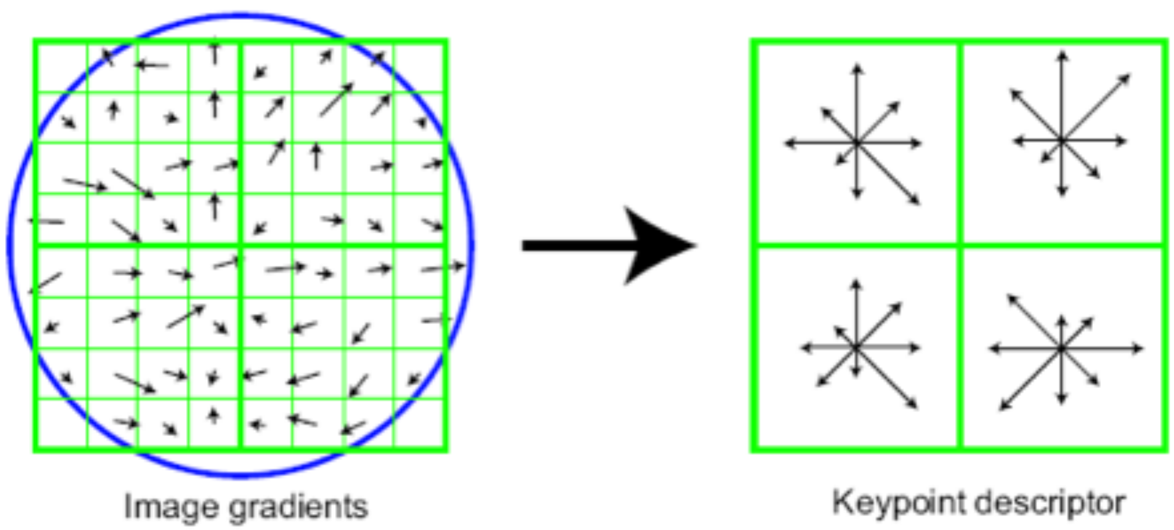
Feature Space

Every training sample is represented as a point in the feature space

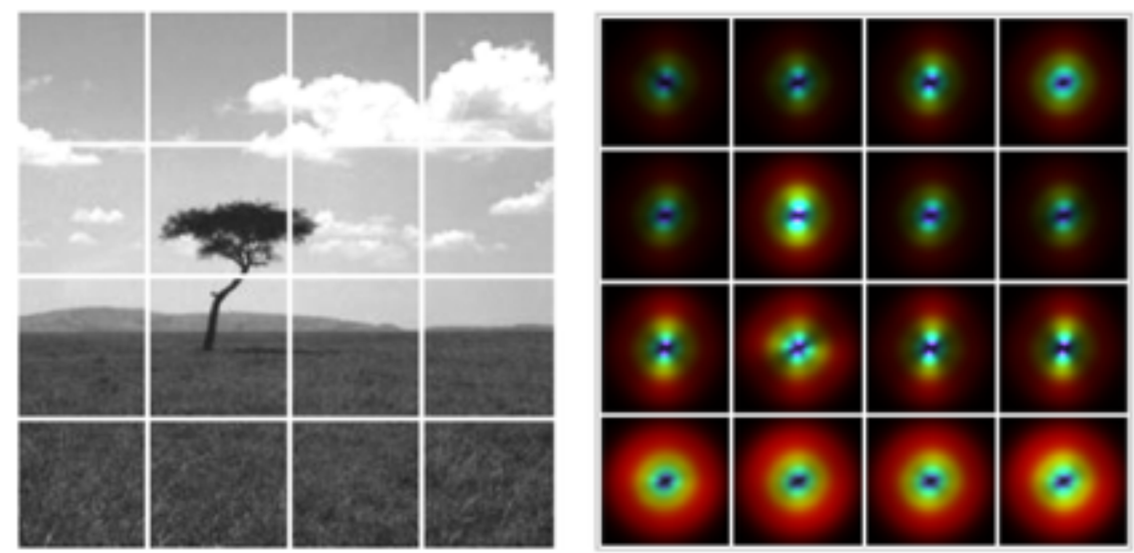


Example Feature Spaces

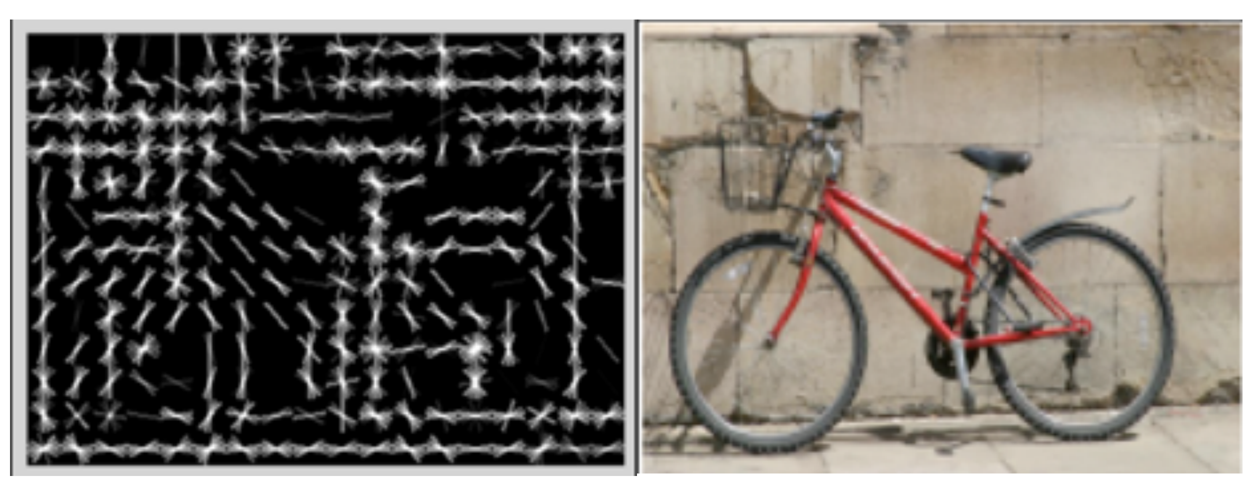
SIFT: Scale-Invariant Feature Transform
(Lowe, 1999)



Gist: Grid of gabors
(Oliva & Torralba, 2001)



HOG: Histograms of oriented gradients
(Dalal & Triggs CVPR 05)



Machine Learning Methods

	Supervised Learning	Unsupervised Learning
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Generative vs Discriminative

Generative

- Represent both the data and the labels
- Often, makes use of conditional independence and priors
- Examples
  - Naïve Bayes classifier
  - Bayesian network
- Models of data may apply to future prediction problems

Discriminative

- Learn to directly predict the labels from the data
- Often, assume a simple boundary (e.g., linear)
- Examples
  - Logistic regression
  - SVM
  - Boosted decision trees
- Often easier to predict a label from the data than to model the data

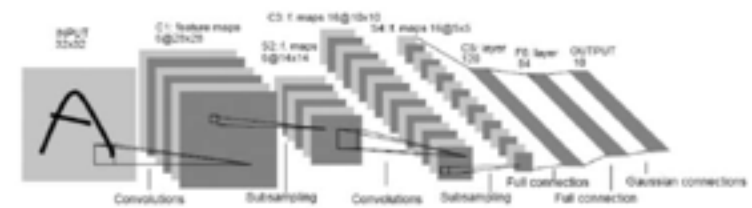
Discriminative Models

Nearest neighbor

10^6 examples

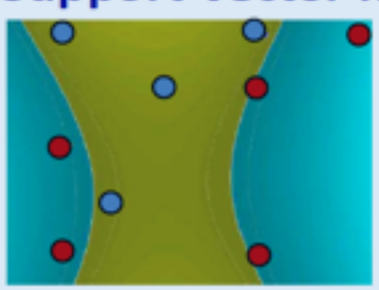
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005…

Neural networks



LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Support Vector Machines



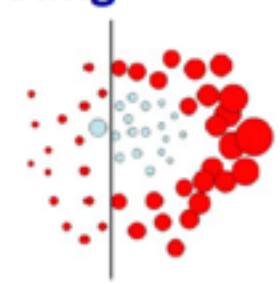
Guyon, Vapnik, Heisele,
Serre, Poggio…

Latent SVM

Structural SVM

Felzenszwalb 00
Ramanan 03…

Boosting



Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,…

Source: Vittorio Ferrari, Kristen Grauman, Antonio Torralba, Fei-Fei Li

Classification

- Apply a prediction function to a feature representation of the image to get the desired output:

f() = “apple”

f() = “tomato”

f() = “cow”

Classification Formulation

$$\mathbf{y} = f(\mathbf{x})$$

output prediction function image feature

- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function f by minimizing the prediction error on the training set
- **Testing:** apply f to a never before seen *test example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

Slide credit: L. Lazebnik

Learning Framework

Training

Training Images

Image Features

Training Labels

Training

Learned model

Testing

Test Image

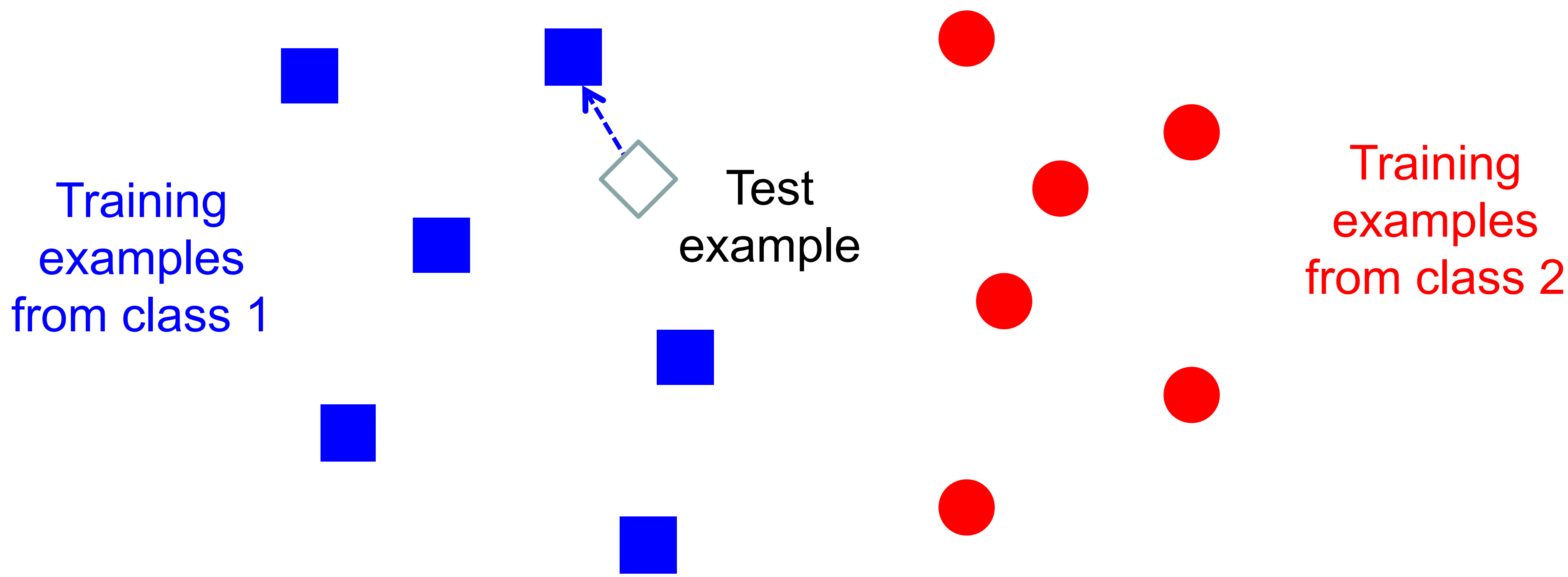
Image Features

Learned model

Prediction

Slide credit: D. Hoiem and L. Lazebnik

Nearest neighbor classification

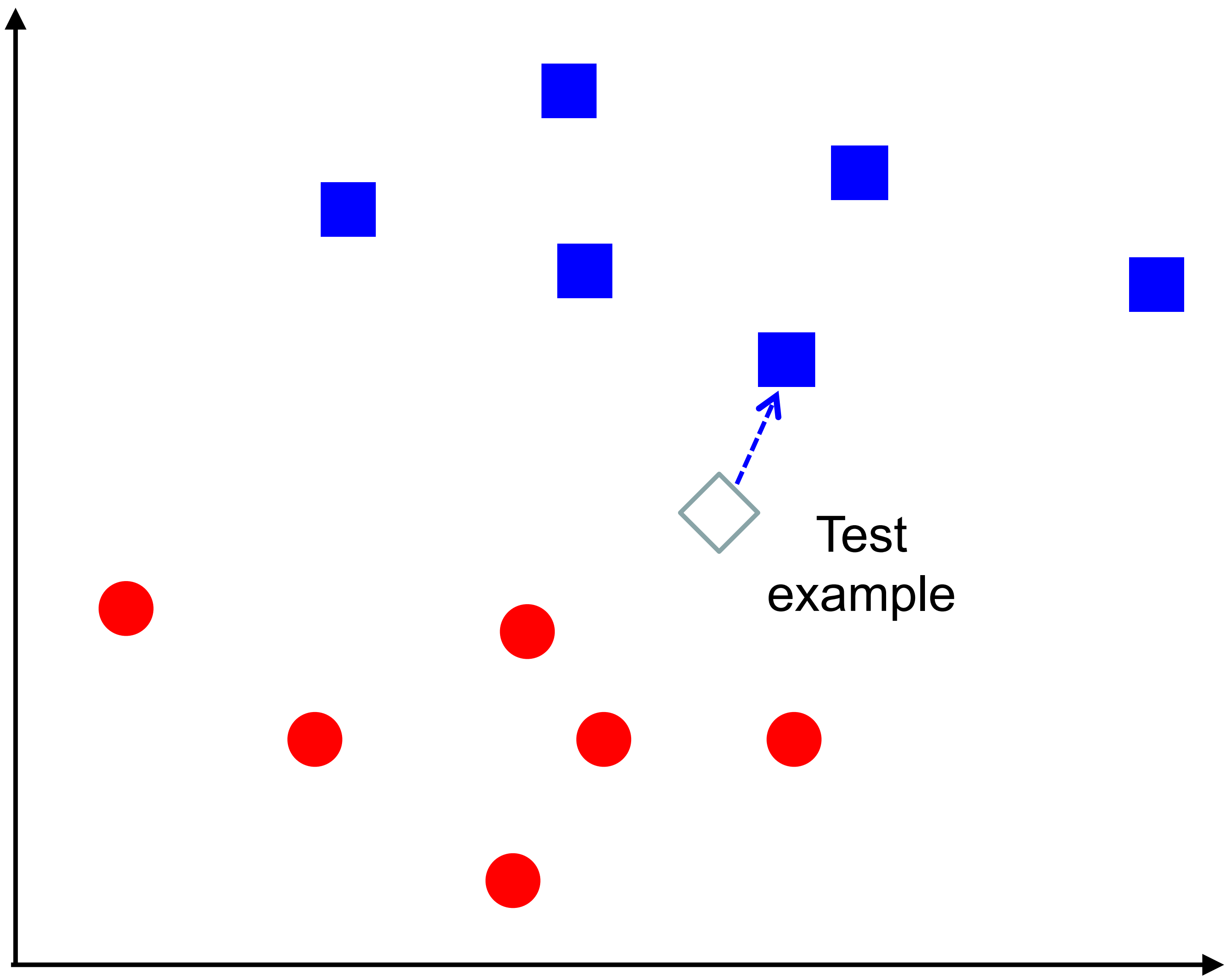


f(**x**) = label of the training example nearest to **x** in the feature space

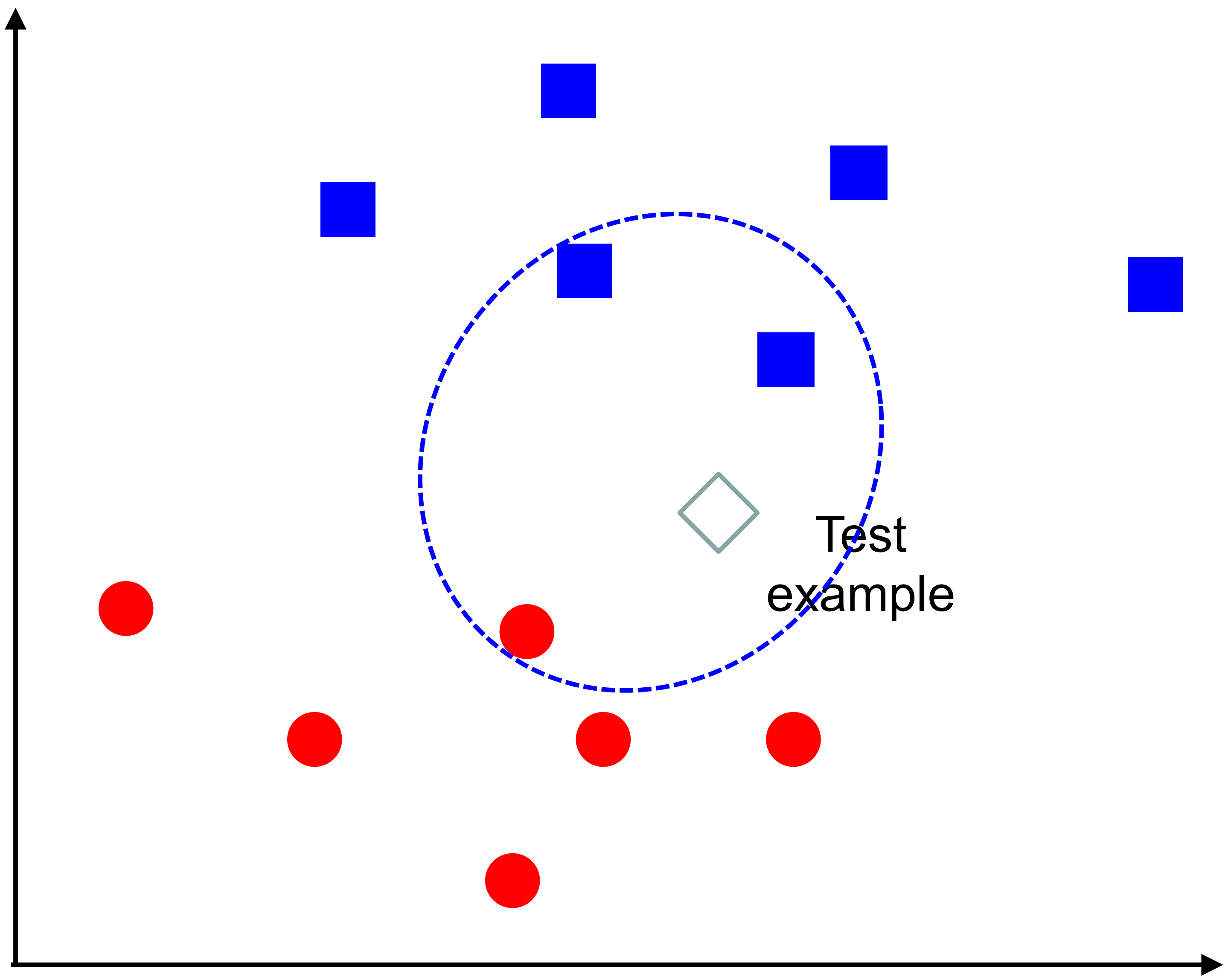
- All we need is a distance function for our inputs
- No training required!

Slide credit: L. Lazebnik

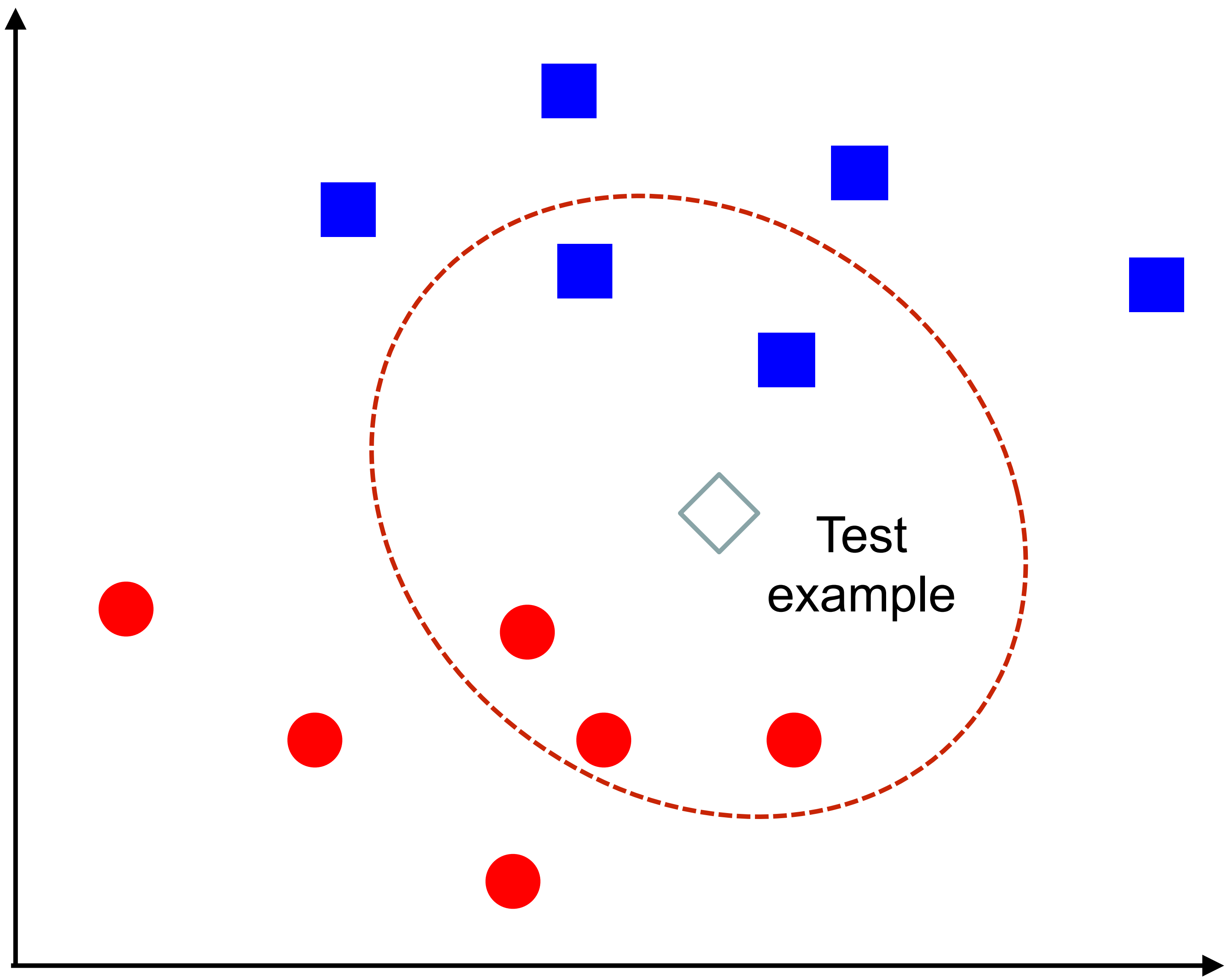
K-Nearest neighbor classification



3-Nearest neighbor classification



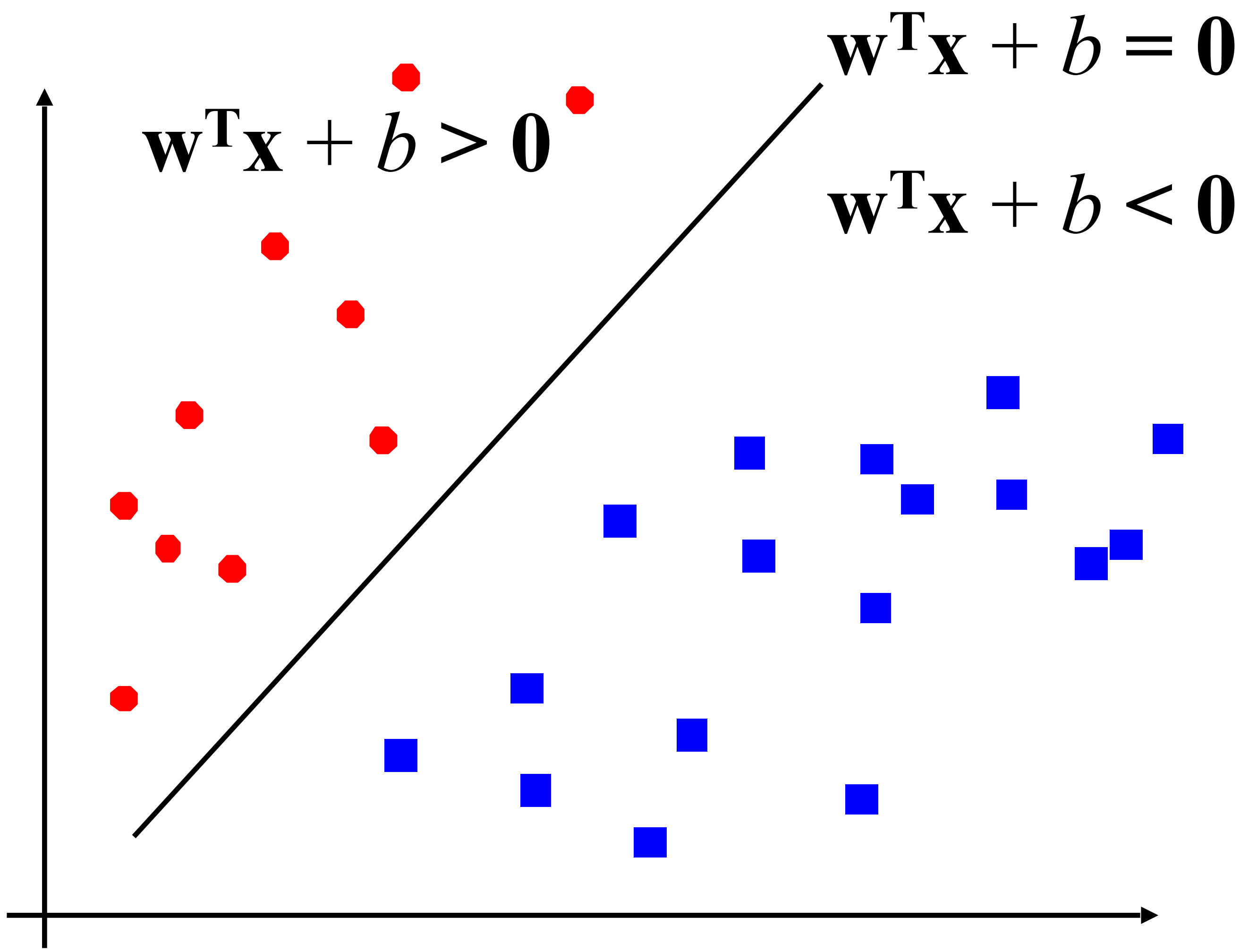
5-Nearest neighbor classification



Simple, a good one to try first

Linear Classifiers: Perceptron

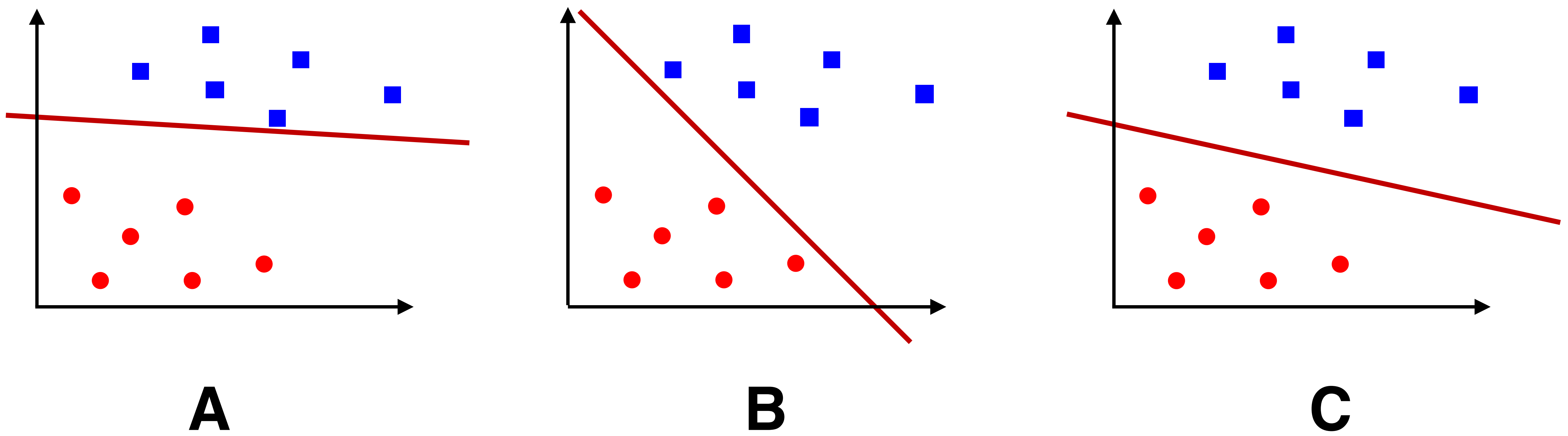
Binary classification can be viewed as the task of separating classes in feature space:



$$f(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x} + b)$$

Linear Classifiers

$$f(\mathbf{x_i}) = \mathrm{sign}(w^T x_i + b)$$

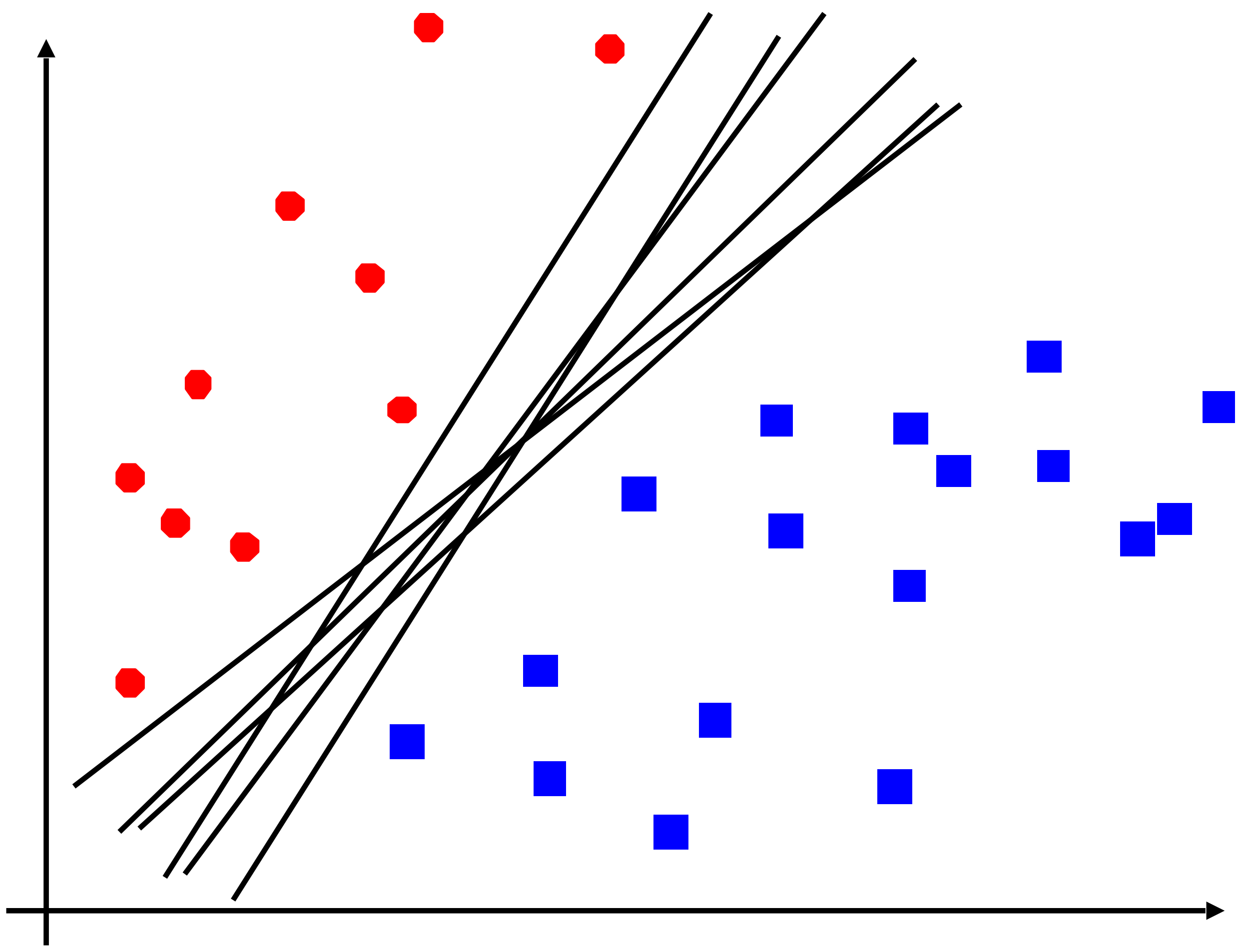


Which one is a better classifier?

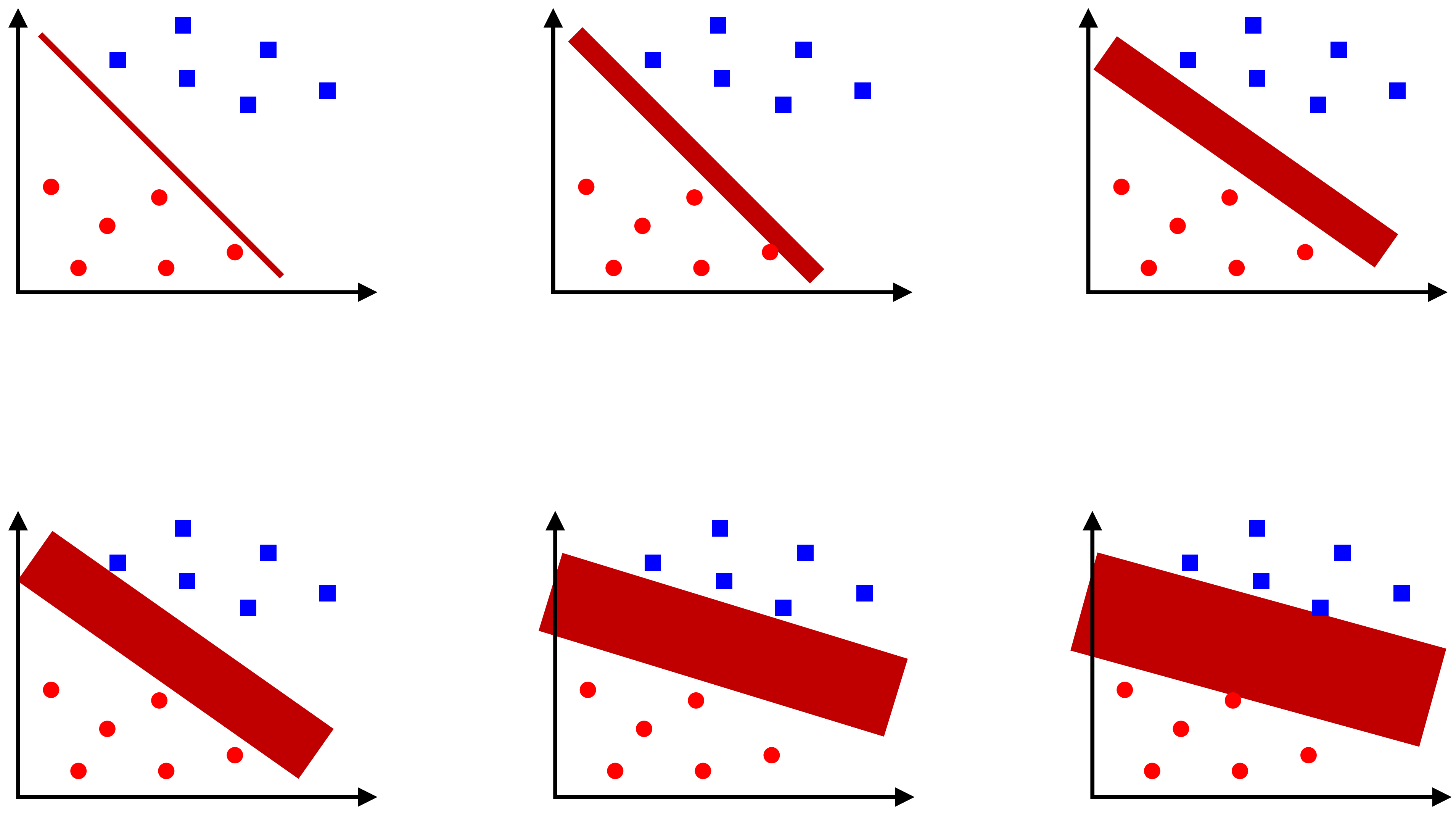
Linear Classifiers

$$f(\mathbf{x_i}) = \text{sign}(w^T x_i + b)$$

Which one is a better classifier?

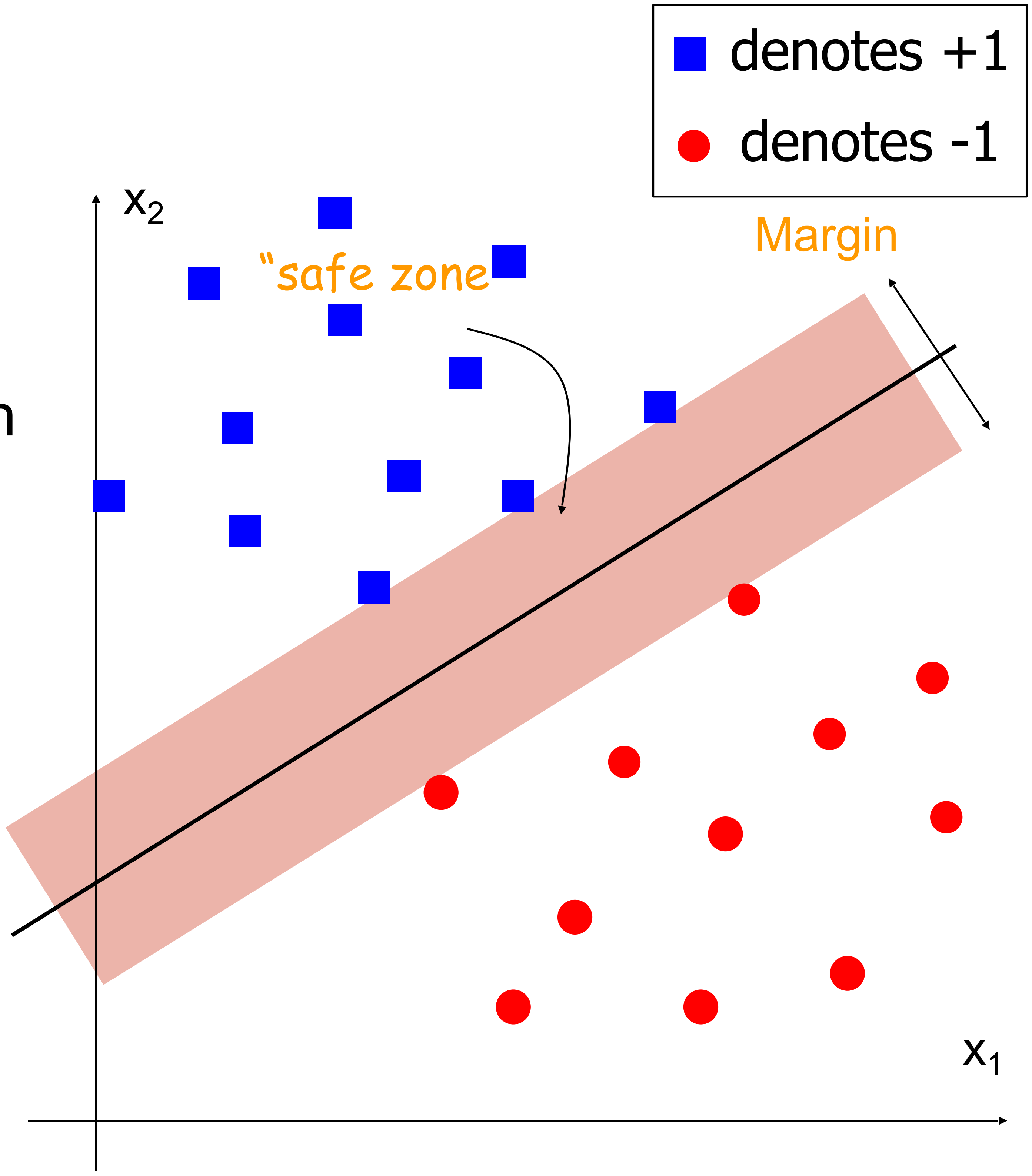


Support Vector Machines (Intuition)



Support Vector Machines

- The linear discriminant function (classifier) with the maximum margin is the best
- Margin is defined as the width that the boundary could be increased by before hitting a data point
- Why it is the best?
  - Robust to outliners and thus strong generalization ability



Slide Credit: Jinwei Gu

Support Vector Machines

- Given a set of data points:

$\{(\mathbf{x}_i, y_i)\},\ i = 1, 2, \ldots, n$, where

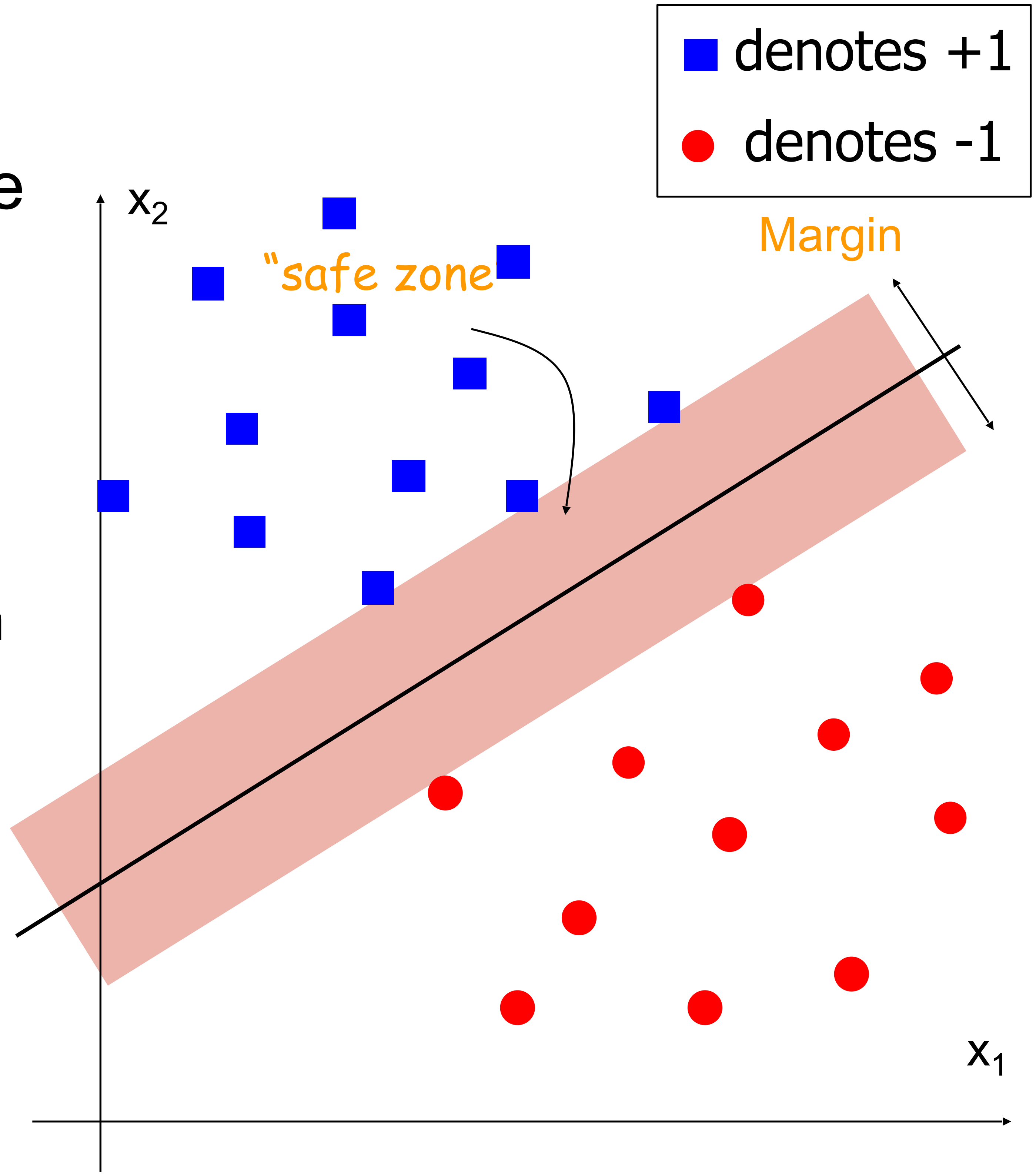
$$\text{For } y_i = +1,\quad \mathbf{w}^T\mathbf{x}_i + b > 0$$

$$\text{For } y_i = -1,\quad \mathbf{w}^T\mathbf{x}_i + b < 0$$

- With a scale transformation on both *w* and *b*, the above is equivalent to

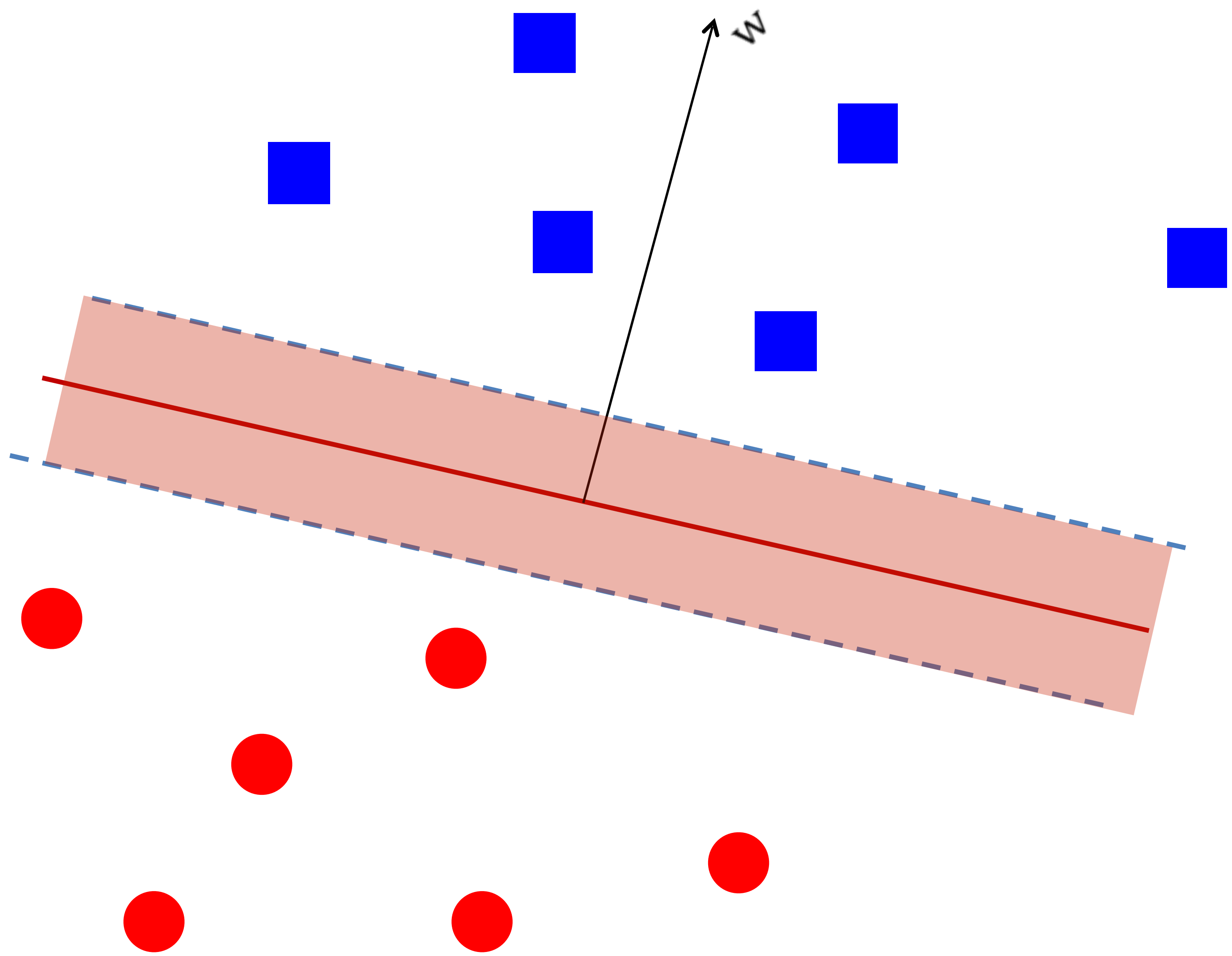
$$\text{For } y_i = +1,\quad \mathbf{w}^T\mathbf{x}_i + b \geq 1$$

$$\text{For } y_i = -1,\quad \mathbf{w}^T\mathbf{x}_i + b \leq -1$$



Slide Credit: Jinwei Gu

Support Vector Machines



$$\underbrace{\min_{w,b} \quad ||w||^2}_{\text{Regularizer}} + \underbrace{C \sum_i \max(0, 1 - y_i(w^T x_i + b))}_{\text{Loss Function (Hinge Loss)}}$$

Support Vector Machines

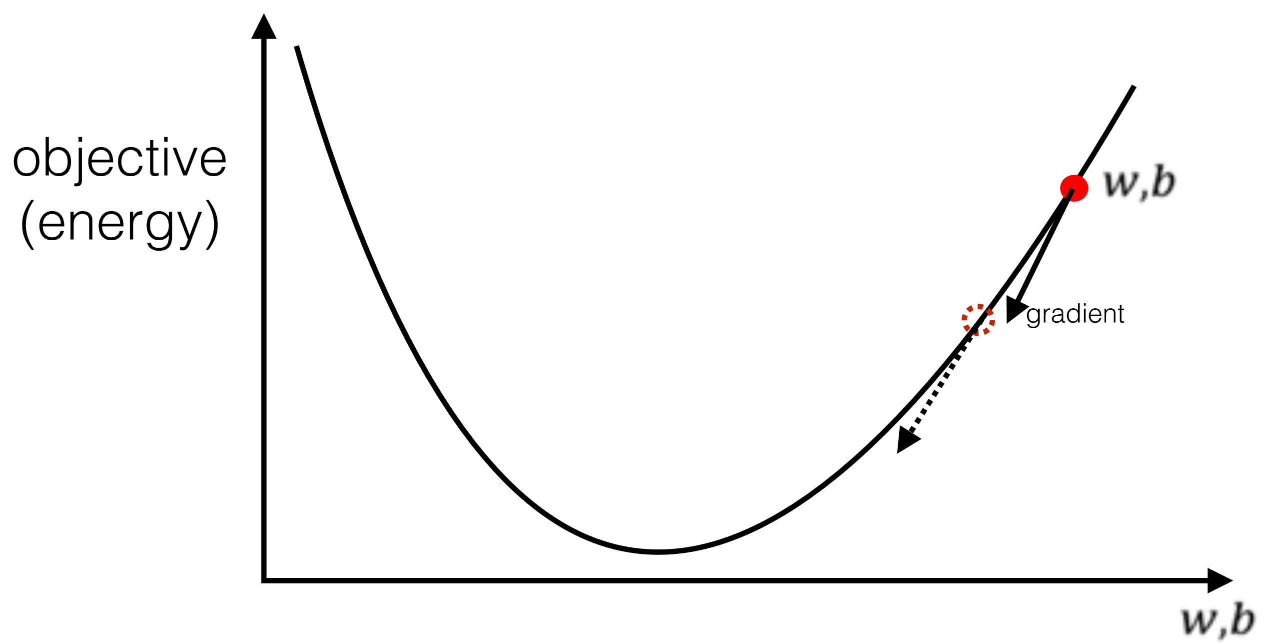
Objective Function

$$\min_{w,b} \quad \underbrace{||w||^2}_{\text{Regularizer}} \quad + \quad \underbrace{C\sum_i \max(0,1-y_i(w^T x_i + b))}_{\text{Loss Function (Hinge Loss)}}$$

Prediction Function

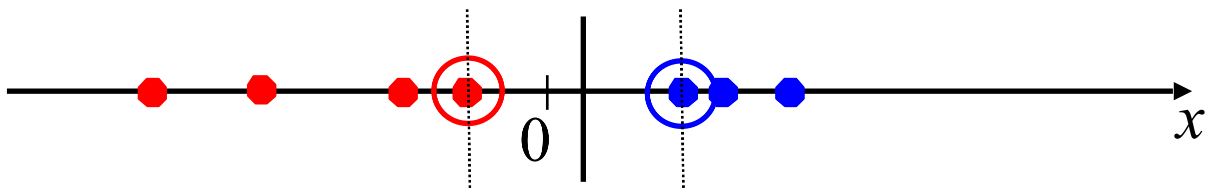
$$y = \text{sign}(\ w^T x_i + b\ )$$

Learning: Convex Optimization

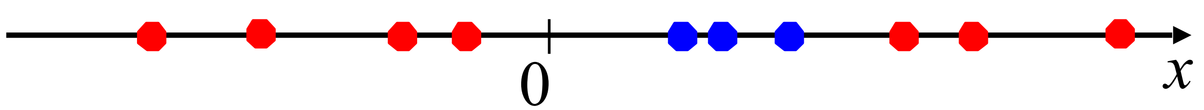


Non-Linear SVMs

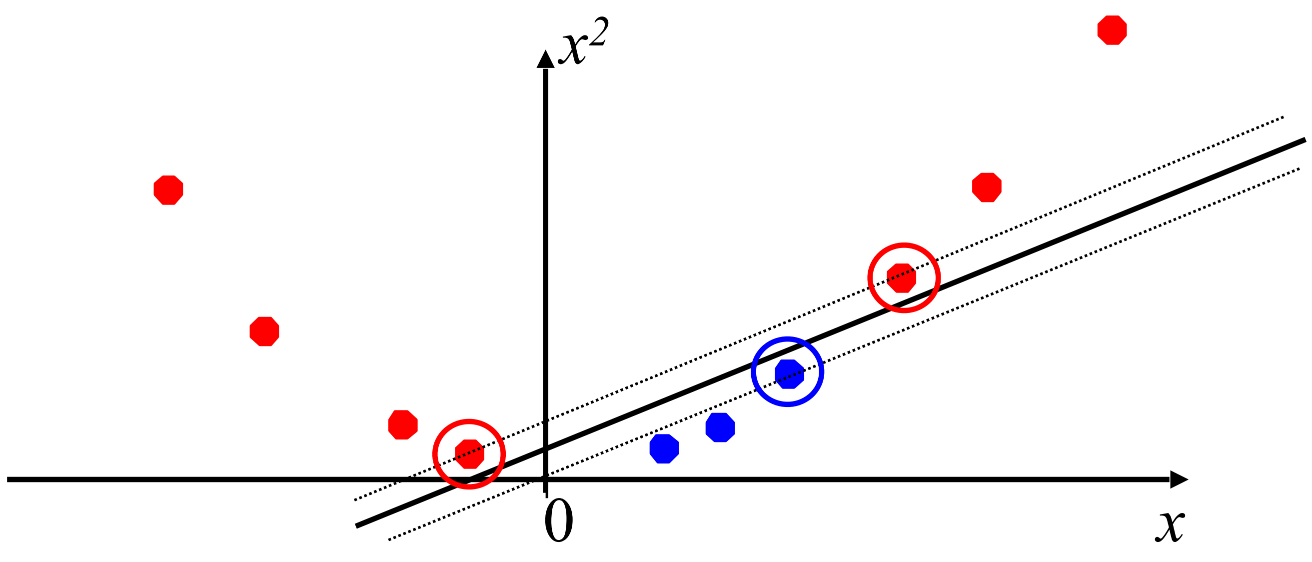
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?



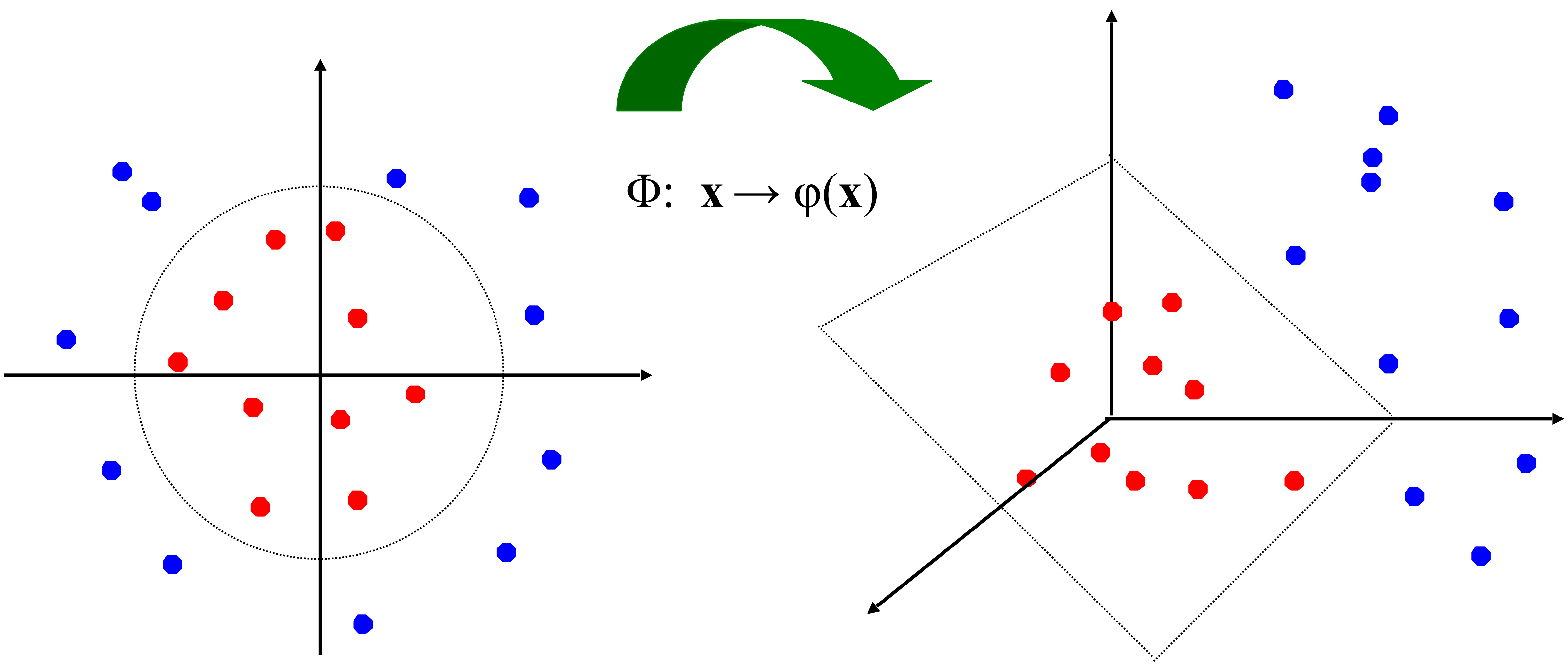
- We can map it to a higher-dimensional space:



Slide credit: Andrew Moore

Non-Linear SVMs

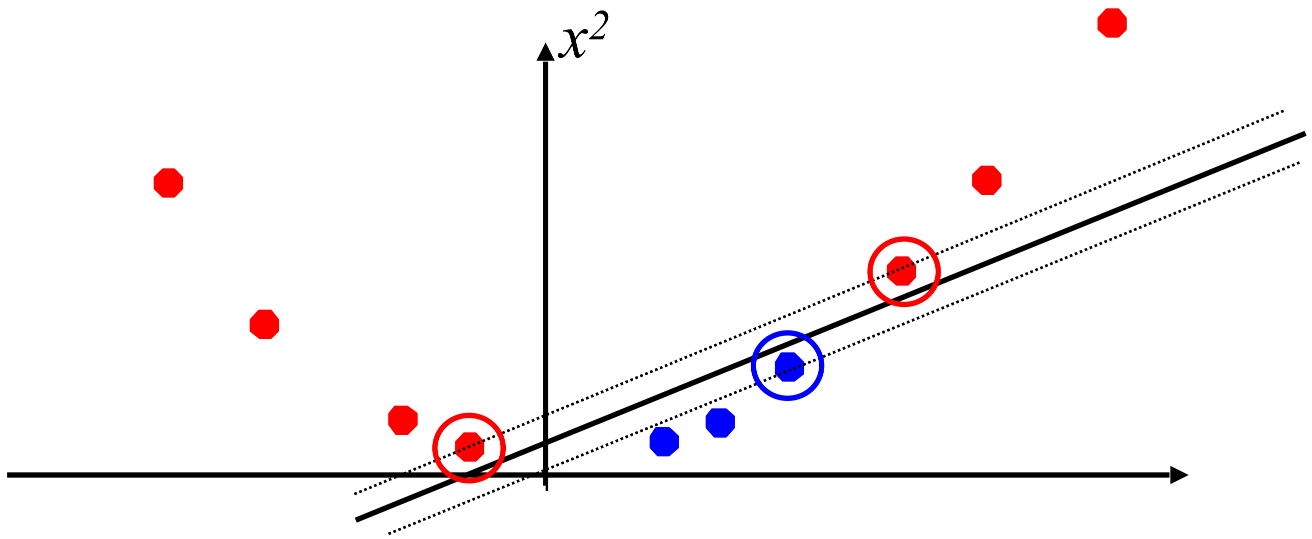
- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Slide credit: Andrew Moore

Non-Linear Kernel: Example

- Consider the mapping $\varphi(x) = (x, x^2)$



$$\varphi(x) \cdot \varphi(y) = (x, x^2) \cdot (y, y^2) = xy + x^2 y^2$$

$$K(x, y) = xy + x^2 y^2$$

Non-Linear SVMs

- ***The kernel trick***: instead of explicitly computing the lifting transformation $\boldsymbol{\varphi}(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\varphi}(\mathbf{x}_i) \cdot \boldsymbol{\varphi}(\mathbf{x}_j)$$

- This gives a nonlinear decision boundary in the original feature space:

$$\sum_i \alpha_i y_i \varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}) + b = \sum_i \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b$$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Common Kernel Functions

- Linear kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- Polynomial kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$
- Gaussian (Radial-Basis Function (RBF)) kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{2\sigma^2}\right)$$

- Sigmoid:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$$

- In general, functions that satisfy *Mercer's condition* can be kernel functions.

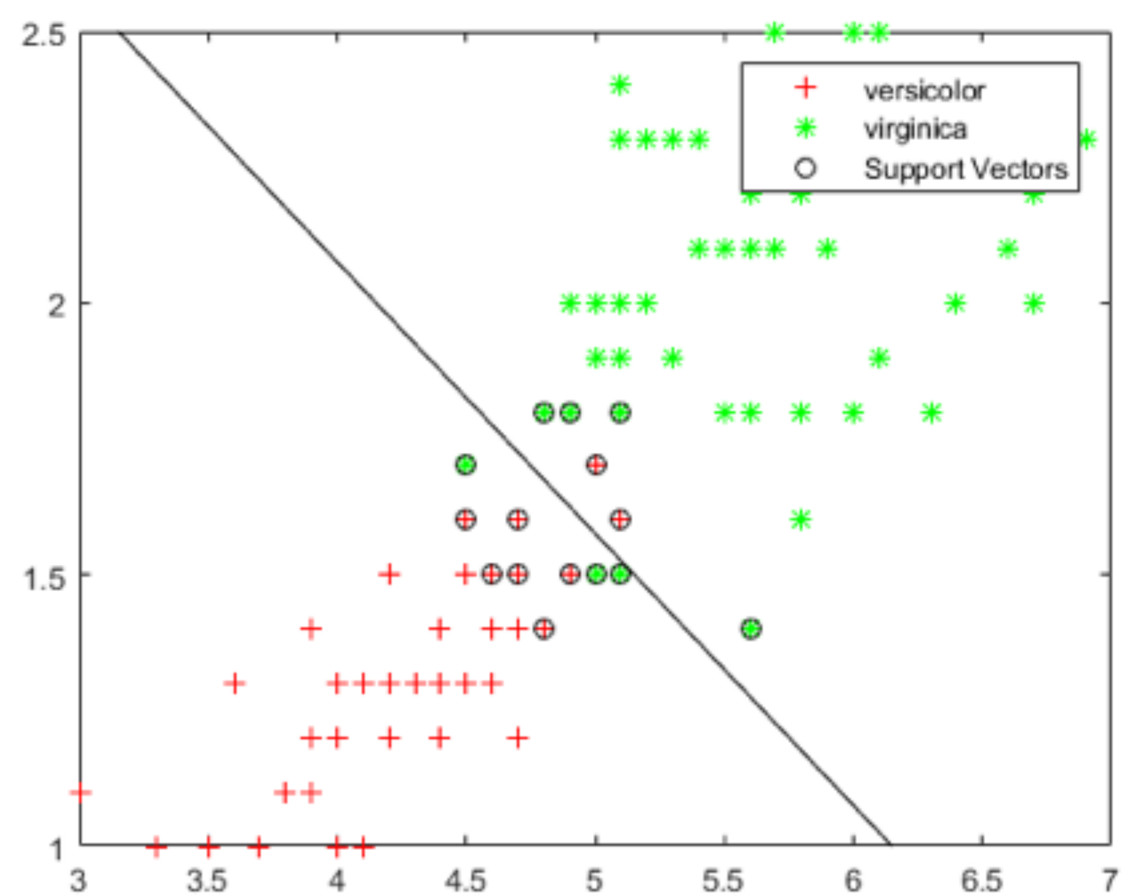
Summary: SVMs for image classification

1. Pick an image representation (HoG, SIFT+BOW, etc.)
2. Pick a kernel function for that representation
3. Compute the matrix of kernel values between every pair of training examples
4. Feed the kernel matrix into your favorite SVM solver to obtain support vectors and weights
5. At test time: compute kernel values for your test example and each support vector, and combine them with the learned weights to get the value of the decision function

Slide credit: L. Lazebnik

MATLAB SVM Example

```
load fisheriris
xdata = meas(51:end,3:4);
group = species(51:end);
svmStruct = svmtrain(xdata,group,'ShowPlot',true);
```



http://www.mathworks.com/help/stats/svmtrain.html

http://www.vlfeat.org/overview/svm.html

What about multi-class SVMs?

- Unfortunately, there is no “definitive” multi-class SVM formulation
- In practice, we have to obtain a multi-class SVM by combining multiple two-class SVMs
- One vs. others
  - Training: learn an SVM for each class vs. the others
  - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value
- One vs. one
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM “votes” for a class to assign to the test example

Slide credit: L. Lazebnik

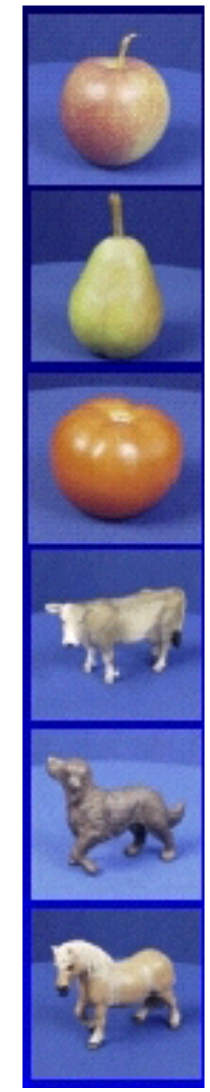
SVMs: Pros and cons

- Pros
  - Many publicly available SVM packages (LibSVM, Liblinear,etc): http://www.kernel-machines.org/software
  - Kernel-based framework is very powerful, flexible
  - SVMs work very well in practice, even with very small training sample sizes

- Cons
  - No “direct” multi-class SVM, must combine two-class SVMs
  - Computation, memory
    - During training time, must compute matrix of kernel values for every pair of examples
    - Learning can take a very long time for large-scale problems

Generalization

Training set (labels known)



Test set (labels unknown)

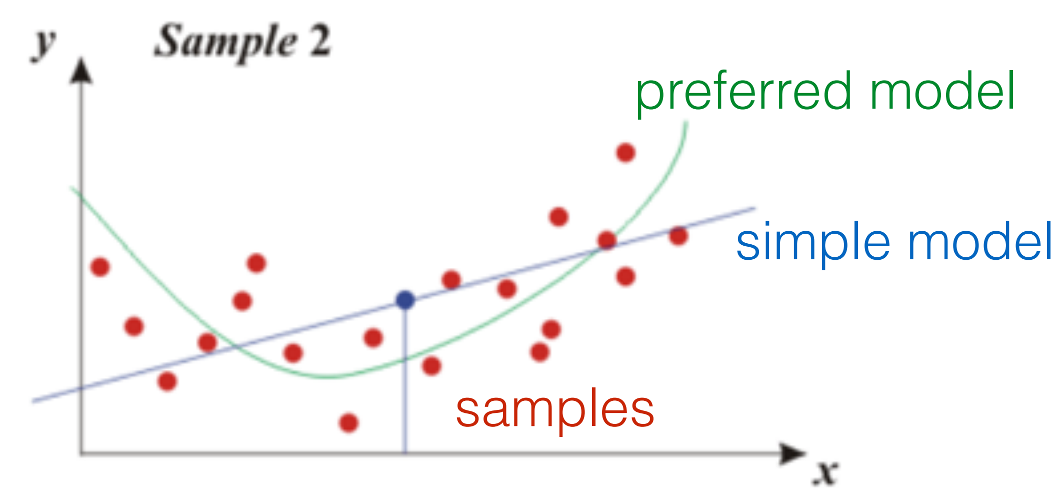
- How well does a learned model generalize from the data it was trained on to a new test set?

Slide credit: L. Lazebnik

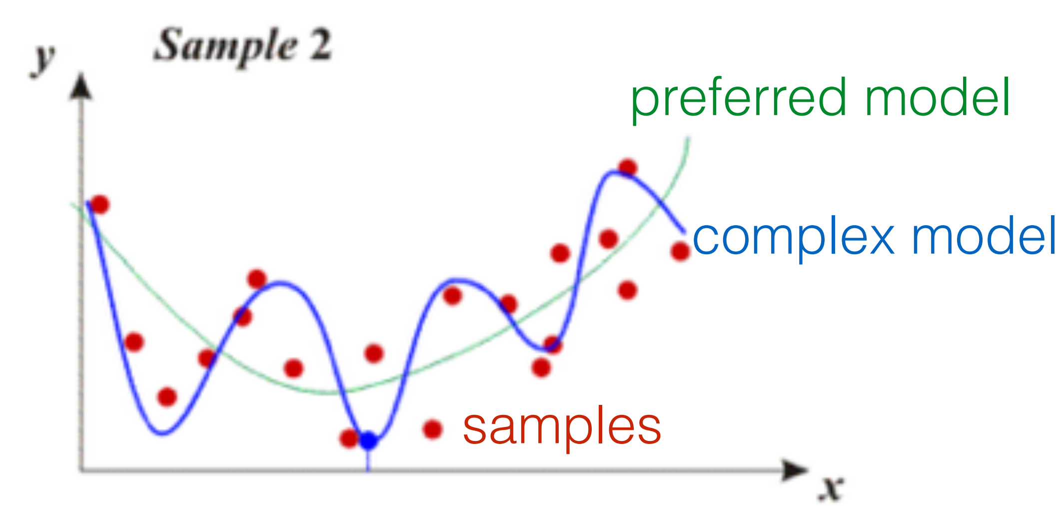
Overfitting vs Underfitting

"Everything should be made as simple as possible, but not simpler."
Albert Einstein

Underfitting: model is too "simple" to represent all the relevant class characteristics

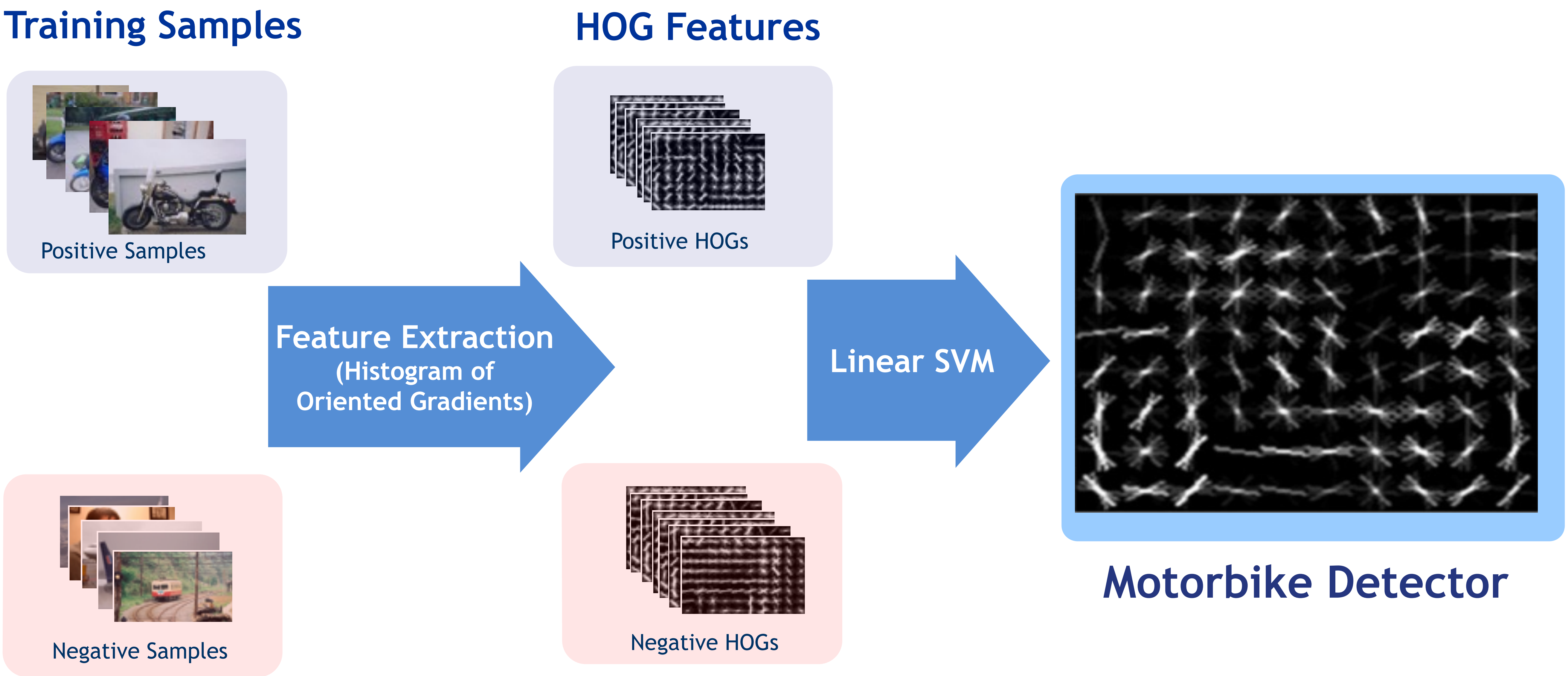


Overfitting: model is too "complex" and fits irrelevant characteristics (noise) in the data



Use Case: Linear SVMs over HoG

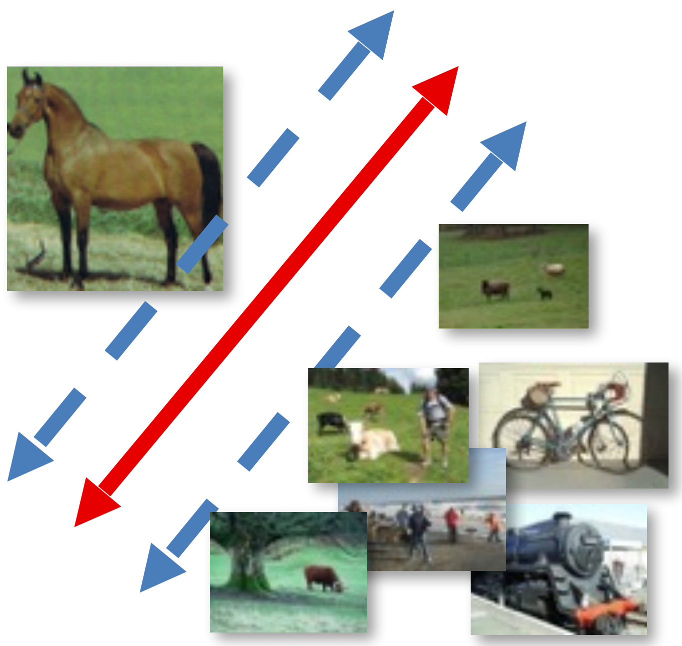
Traditional Detector Training (Motorbike)



[Dalal *et al.* CVPR'05]
[Felzenszwalb *et al.* CVPR'08]

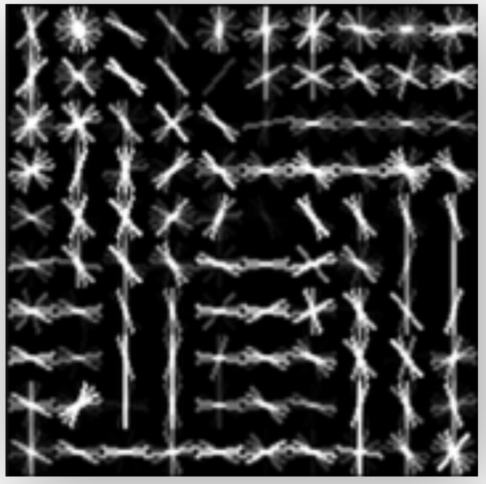
Use Case: Exemplar SVMs

Training an SVM with a single positive and many negative samples

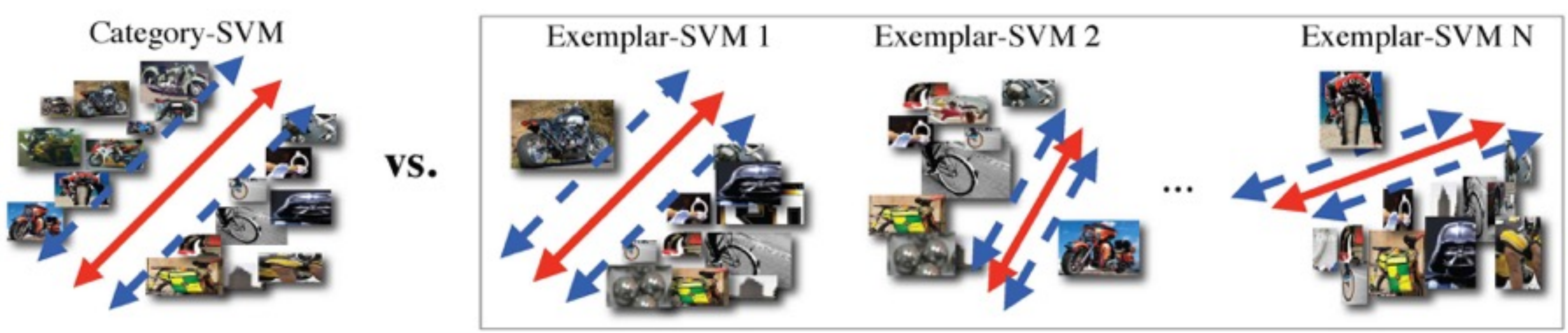


Linear SVMs
over
HoG features

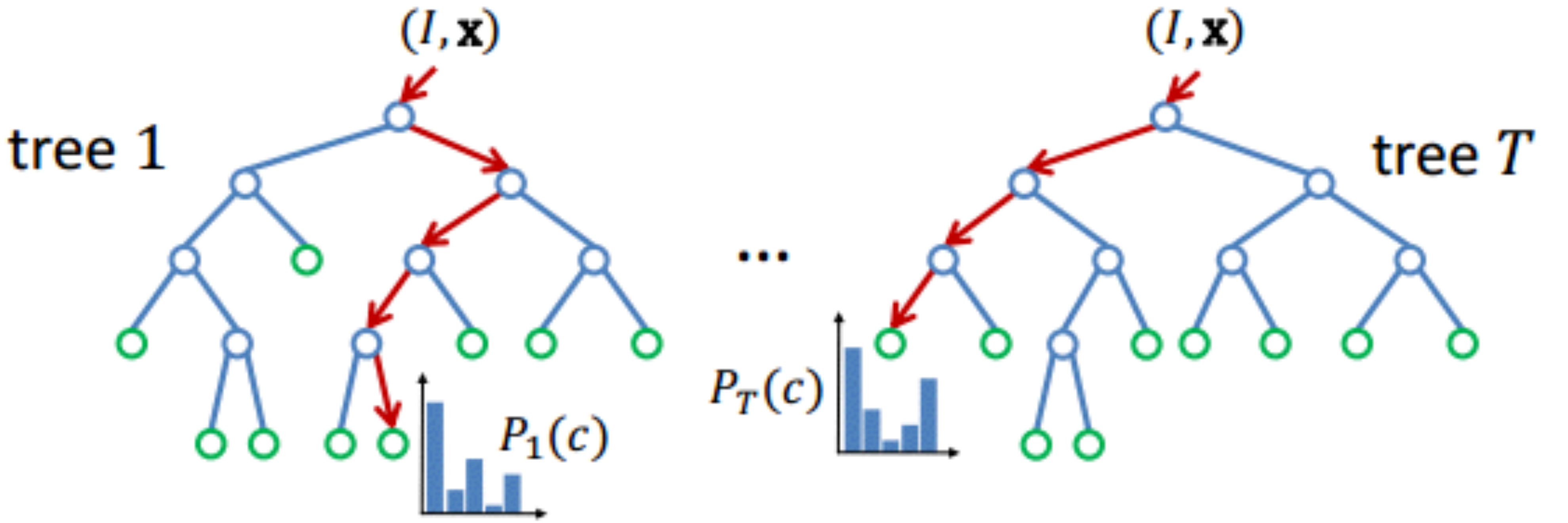
[Dalal &Triggs'05],
[Felzenszwalb'08]
[Malisiewicz'11]

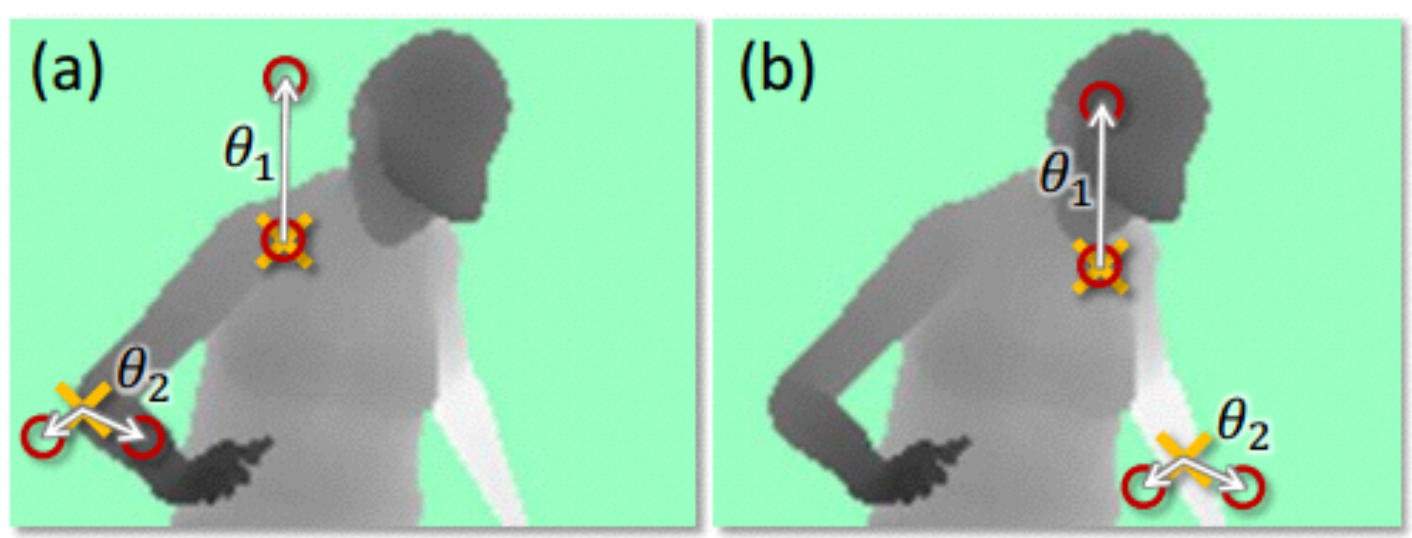


Exemplar SVM



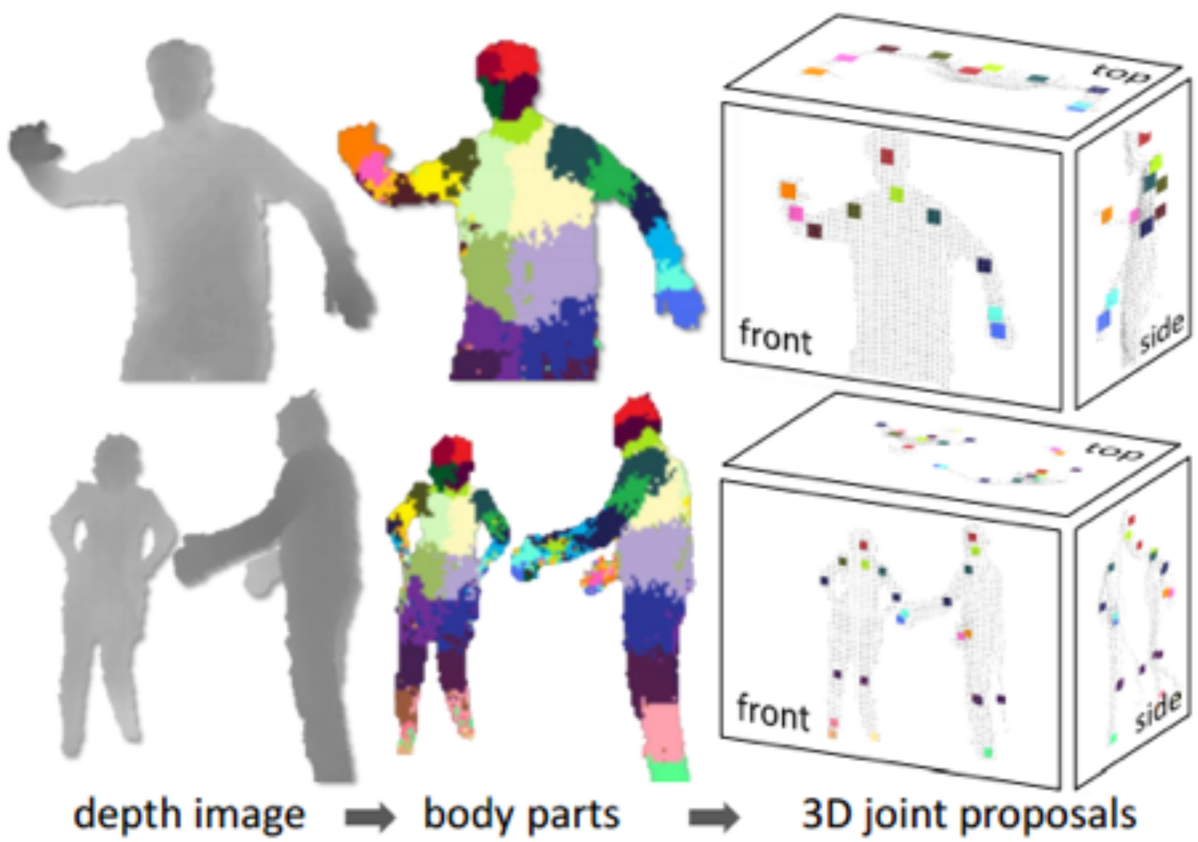
Another Classifier: Randomized Decision Forests

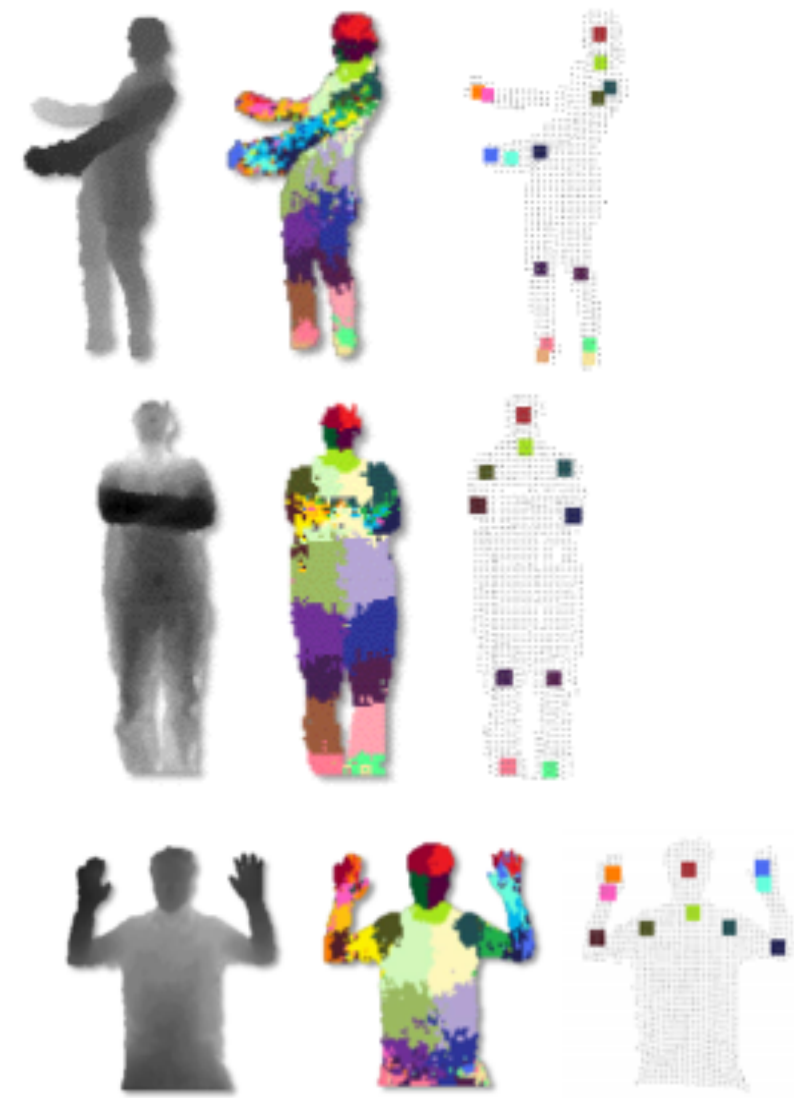




Shotton et.al., Real-Time Human Pose Recognition in Parts from Single Depth Images, CVPR, 2011

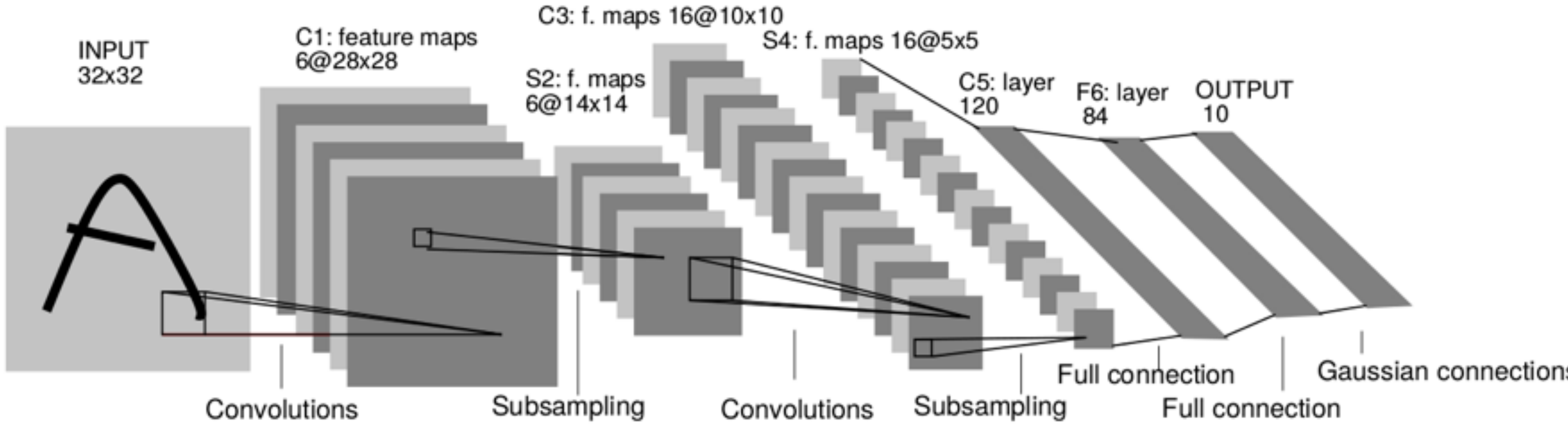
Body Part Classification with Randomized Decision Forests

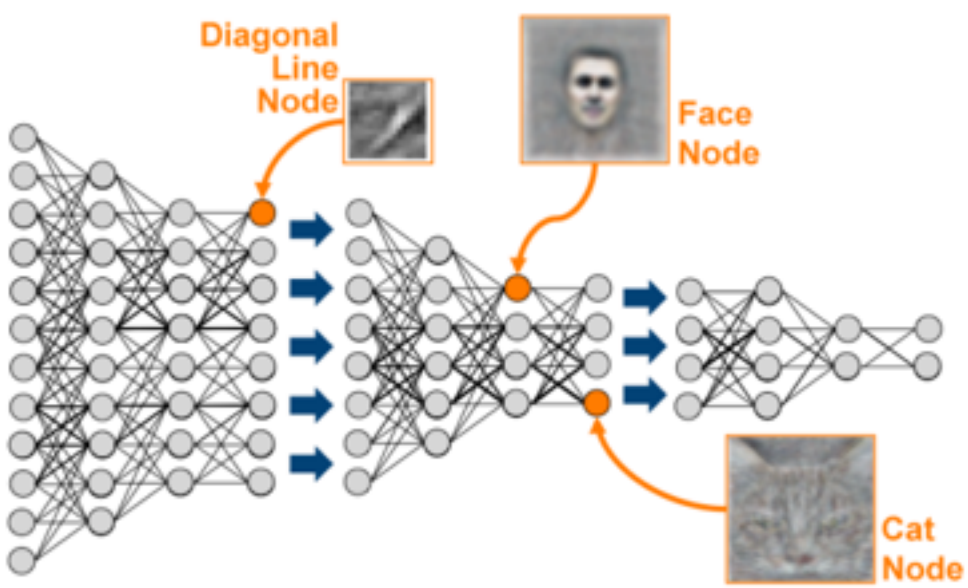


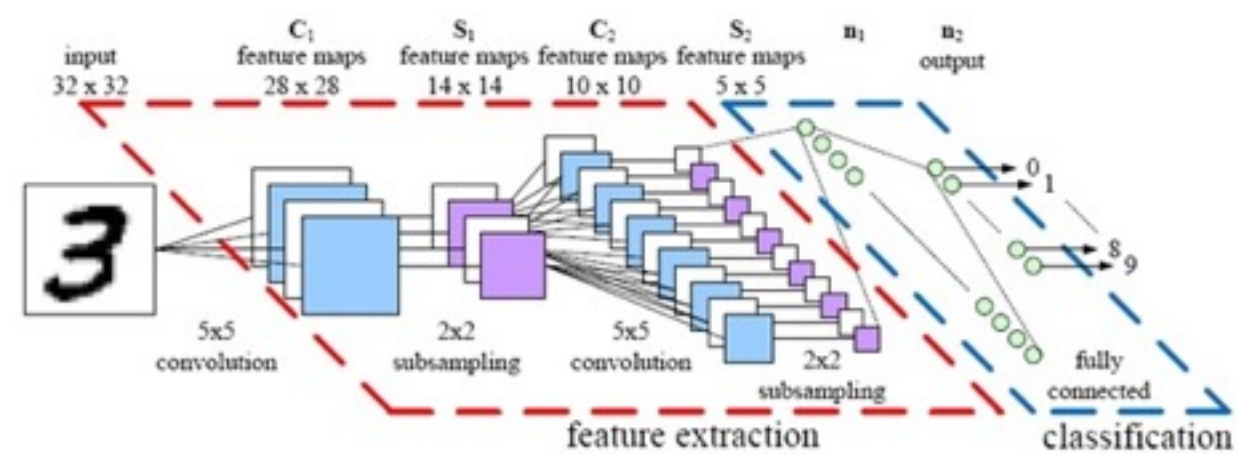


Shotton et.al., Real-Time Human Pose Recognition in Parts from Single Depth Images, CVPR, 2011

Another Classifier: Deep Learning







Next Lecture …