Lecture 13.

Note Title 2/27/2010

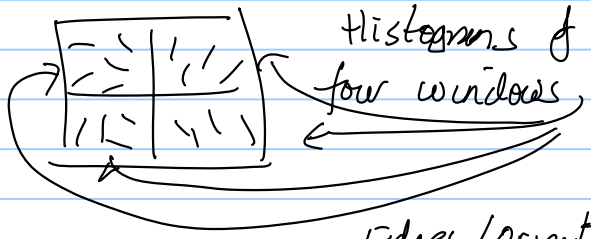
Object models are simpler if Interest Points (IP) are used.

These are sparser and more descriptive than edges, but they provide a weaker description of the object (often uninteresting).

Detect IP's → Brady-Kadir detector, SIFT detector. Whole literature on detectors. (scale)

Describe IP's → SIFT descriptor

Intuition

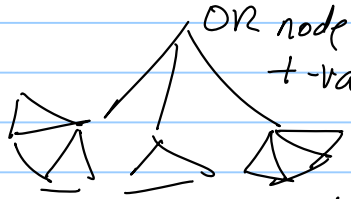


Histograms of edges inside four windows

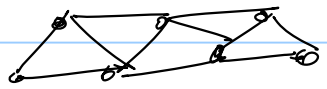
Edges/orientation roughly invariant to illumination

Histogram of edges – invariant to illumination & some viewpoint changes.

Model



The OR node selects between models of form



AND-models but composed sideways

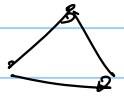
ordered set of nodes

$(1,2,\ldots,n)$

$(1,2,3),(2,3,4),(3,4,5),\ldots$

potentials

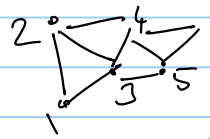
$\lambda^G \cdot \varphi^G(\omega_\mu, \omega_\nu, \omega_\rho)$



data terms for each node

$\lambda^D \cdot \varphi^D(\omega_\mu; I)$

"attracts" ω_μ to an IP.



Clique – set of vertices s.t. each pair of vertices are connected by an edge.

For OR node $\sum_{\nu \in ch(R_{or})} \lambda_\nu \delta_{t,\nu} \delta(\omega_n - f(\omega)_\nu)$

$f(\omega)$ → summarizes

Background model – generates remaining IP's in the image

(2) State variables

$\omega_n = (x_n, \theta_n, A_n)$ ← attribute. → specified by SIFT descriptor.
↑ position ↑ orientation

Model → full energy
= sum of data terms $\lambda^p \cdot \phi^p$
+ sum of geometric terms $\lambda^g \cdot \phi^g$
+ OR node.
+ Background.

Inference: if model is specified (graph structure + parameters λ's)
then can do inference by dynamic programming followed by exhaustive search to deal with the OR node.

Note: robustness — 2 out of 3 rule
if we miss the state of one node for a triangle clique then we can predict its position from the positions of the other two.
(Helpful if one IP is not detected because of occlusion or failure of the detector).

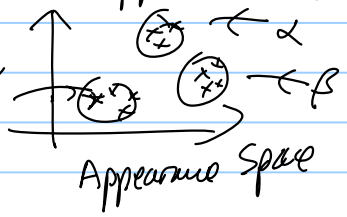
Learning
→ suppose the structure is known but we do not know the lambda λ parameters.
→ then we can learn from training data by maximum likelihood using the EM algorithm — the hidden variables, which we need to estimate, are which sub-model is used and what are the positions, orientations, and attributes of the nodes
Standard EM — we can use Dynamic Programming (DP) to make the learning practical. Initial conditions?

(3) Learning the Structure and Initializing Parameters.

Dictionaries →

Ignore spatial relations:

perform clustering on the appearance of the IP's (clustering - eg k-means)



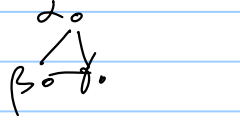
Treat each cluster as an element of an Appearance dictionary: $\mathcal{D}_A$

Each IP can be represented by an element of $\mathcal{D}_A$

Each IP is a very simple graphical model with single node. $\lambda_\alpha \cdot \phi_\alpha(x_\mu, A_\mu, \theta_\mu)$ $\alpha \in \mathcal{D}_A$

Note: in previous lecture, we defined a dictionary for edges → e.g. — | \ /.

Now cluster triplets of IP by geometry.



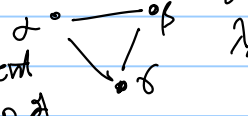
As in previous lecture.

Spatial relations - gaussian distribution on relative positions → gaussian on vector I made up of position, orientation, rotation.

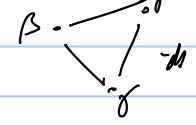
This gives a Triplet Dictionary $\mathcal{D}_T$

Note: this procedure is similar to learning a level-2 dictionary of edge structures from a level-1 dictionary of edgelets.

Each element of $\mathcal{D}_T$ is a small graphical model

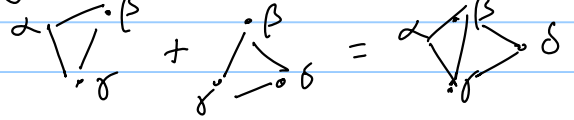


$\lambda_\alpha \phi_\alpha(\omega_\alpha; I) + \lambda_\beta \phi_\beta(\omega_\beta; I) + \gamma_\delta \phi_\delta(\omega_\delta; I) + \lambda_\sigma \cdot \phi_\sigma(\omega_\alpha, \omega_\beta, \omega_\delta)$



Each dictionary element has a score → no of times it occurs.

We can now build an object model by combining elements from the triplet-dictionary.



(4) # Learning the Structure & Initialization

Default Model: All data is generated by a background model.

$P(n) \prod_{\mu=1}^{n} P(x_\mu, \theta_\mu, A_\mu)$

n = no. of 1P's in each image.

$P(n)$ & $P(x_\mu, \theta_\mu, A_\mu)$ can be learnt from the examples.

Key point: this model assumes that the data points are generated independently from a distribution $P(x_\mu, \theta_\mu, A_\mu)$

(Note: in practice, little difference is we replace $P(x_\mu, \theta_\mu, A_\mu)$ by a uniform distribution).

First Step:

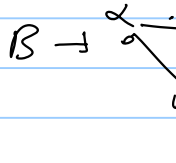
Select a triplet α β γ from the Triplet Directory

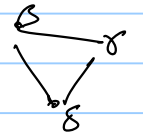
Use model:  + Background.

ie. 1P.'s in the image are either generated

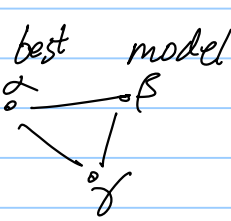
Estimate parameters using EM, by α β γ or by background (precalculated background) with initial values specified by dictionary.

Perform $\Rightarrow$ model selection

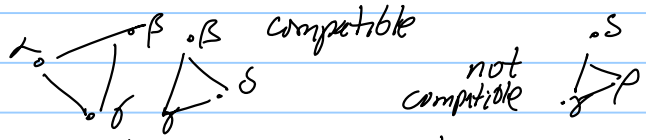
does B or $B \to$  best generate the data.

or $B \to$  or any other combination

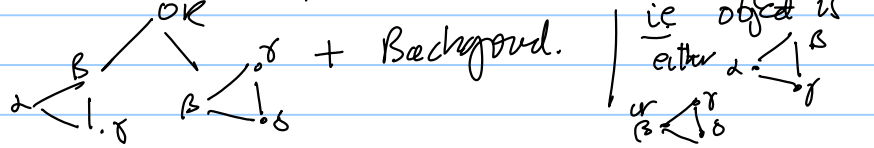
Select the best model

e.g. $B \to$ 

Second Step:

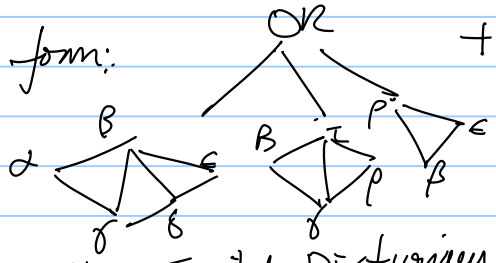
Two options $\Rightarrow$ either add a new triplet which is compatible with the first 

or add a new triplet as an alternative.

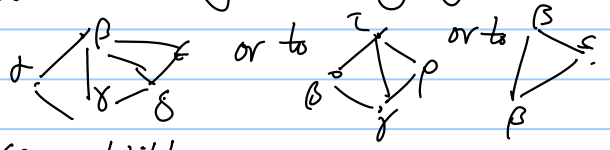


(5) More Generally:

After N stages:
model is of form:

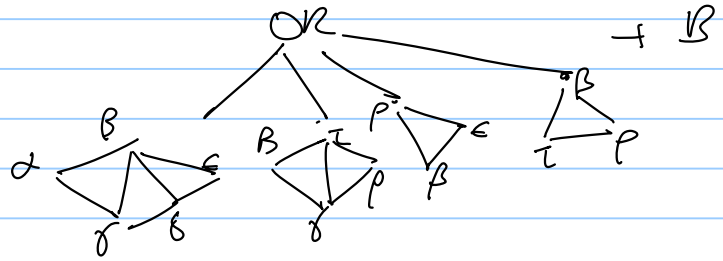


(1) Can add triple from the Triplet Dictionary to any of the OR models – e.g. to



in each case – must ensure compatibility.

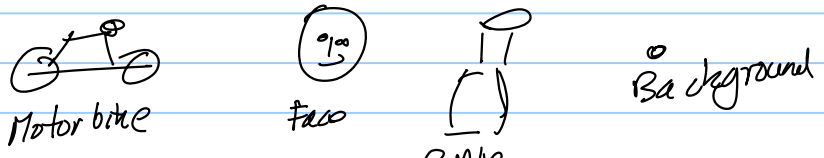
(2) or can add a new triplet



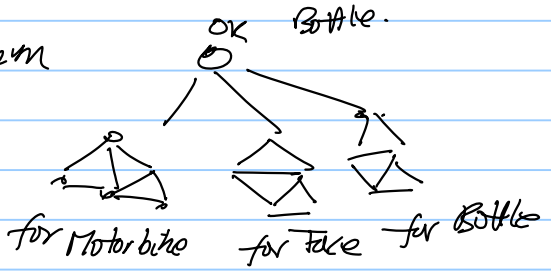
For all cases, perform model selection to determine whether the new model is better than the old model.
If not, stop growing the model.

What does the OR's do?

Suppose the dataset consists of several different types of objects?

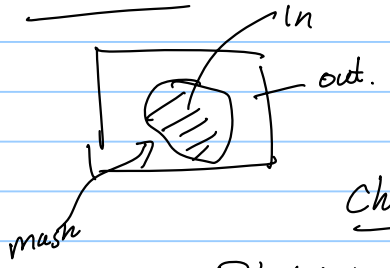


This procedure (should) learn



Learning OR's is possible because the Interest Points are distinctive.

(6) Enhancing the model by adding a mask.



Label:
$L(x)=1$, if x is in the object
$=0$, if x is outside the object.

Choose image feature $f(I)$.

$$P(\{f(I)\} \mid L) = \prod_{x:L(x)=1} P(f(I)(x) \mid L(x)=1) \prod_{x:L(x)=0} P(f(I)(x) \mid L(x)=0)$$

ie. assume the statistics of $f(I)$ are different inside and outside the object.

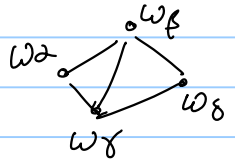
$$P(L) = \prod_x P(L(x) \mid M)$$

where M is a mask

Hence $P(L \mid \{f(I)\}) \propto P(\{f(I)\} \mid L) P(L)$.

But, how to deal with changes in size, orientation, and position?
How to learn the mask → ie. $P(L(x) \mid M)$?

Couple the mask to a model with IP's.



$\omega = f(\omega_2, \omega_\beta, \omega_\gamma, \omega_\delta)$
summary terms

state of ω gives the position, orientation, and scale of the mask

Strategy: learn the model with IP's
learn a mask model using IP' model to help train it.