

Lecture 10: Learning in Stochastic Image Grammar

Unsupervised learning and learning rate

Song-Chun Zhu

Center for Vision, Cognition, Learning and Arts
University of California, Los Angeles

At CVPR, Providence, Rhode Island
June 16, 2012

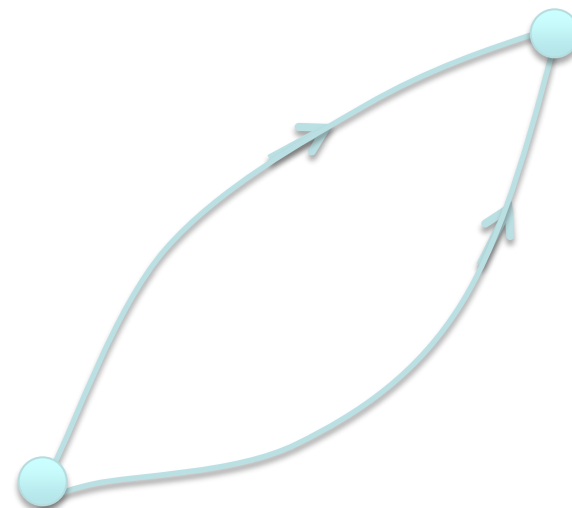
Two objectives of learning

1, Explaining the data

- Generative approaches
- Hierarchy, dictionary
- Max. likelihood, Min KL-divergence

2, Minimize the loss --- driven by task

- Discriminative approaches
- long features
- Max margin, Min loss



Both have some merits. In this lecture, we focus on learning hierarchical grammatical models to explain the data. One can boost the performance by Structured-SVM once the task and loss functions are known.

Outline of this lecture

1, Learning And-Or Trees from data matrix

case study 0: toy example

case study 1: object S-AOT

case study 2: event T-AOT

case study 3: causal C-AOT (in lecture 8)

2, A unifying principle: learning by information projection

--- pursuit of stochastic sets in the image universe

3, PAC-learning: the learning rate of AOG models

1, Case study 0: a toy example

Suppose we design a 2-layer And-Or graph (rewritten as tree) which generates many examples,

- Can we recover this AOG by unsupervised learning?
- What are the key factors affecting learning accuracy?

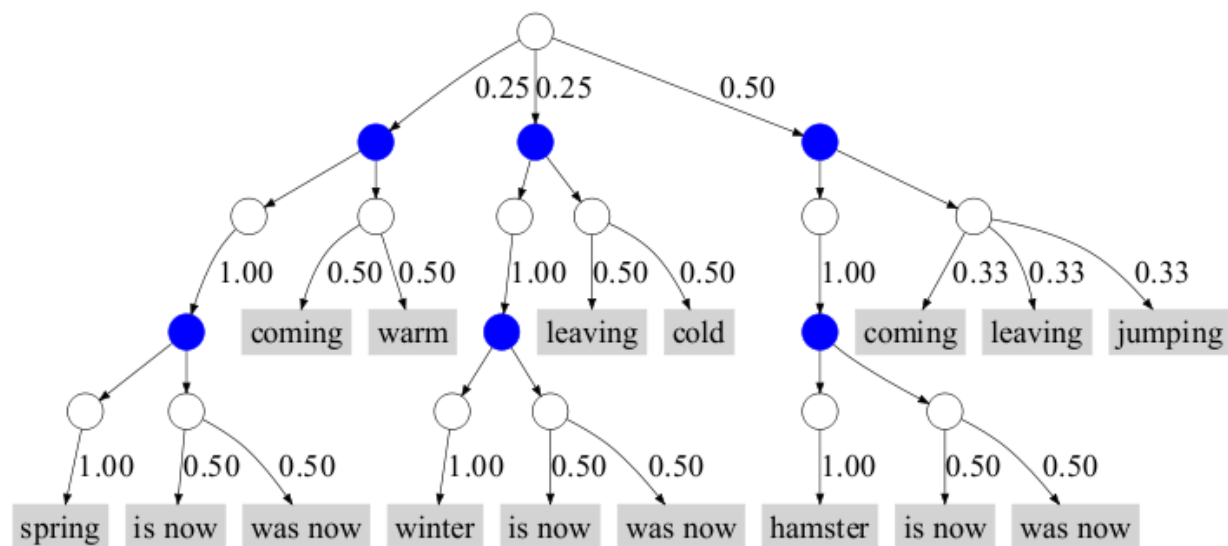


Figure 1. A toy AND/OR graph for completing a sentence, such as “hamster is now jumping”. There are certain constraints in forming the sentence, such as *spring* should not be *cold* or *leaving*. Shaded circles denote AND nodes. Empty circles means OR nodes. Shaded rectangles are terminal nodes.

Phrase instances –sampled from the AoG

Noisy letters of random length **s** are inserted between words.

rkspbwidfvq**hamster** b x **is** nowtncyjh**oleaving**pkqmlqzkgw
mpuhqpkysdhs**spring**jcgr **is** nowhuyjr**coming**wj qqptc
fyglgvwwvcoq**winter**vkixcc **as** nowkzteds**leaving**xtdlg qt
pojuptrjho **hamster**dhypjaa **is** cowlhoad**leaving**ciczcdz hf
lzxnsvh dy**hamster**qqdis **now**myqpzb**coming**eu ybaf irfdhey
l ygszzkzq**hamster**andx **is** nowrvxza**leaving**tdplhvvyjqcqxowd
qexxttrqdot tui**winter**bsodv**was** nowgjolz**coming**mrqnjmgmddqxxizu
cgkkaek**winter**fhhsbb **is** nowtdyqdl**leaving**xctsqcsc
spmdxqzh **aspring**gfesqi **is** now tca**coming**ggv ezeu
rimldbwdqowbt**dawinter**rusz **is** nowtgrgnq**leaving**ow nhstkb
horbsp**winter**v m **is** nowympvee**eaving**rmikrzzjyhpz
lyrszvsd**spring**hysctewas **now**nvigix **coming**xfbuxzg
nuxrpto**hamster**lnworzv**was** nowbo pnb**coming** vvm ddozops x
ekhqhtfats q**winter**rydwgznp**was** nowciz**leaving**evjzit
ufjwauxov w**hamster**cvaet**was** noweu**leaving**mdzyrnaptdg
lukzjxpe puxb**winter**dgiueagwas nowpdlrcr**comings**rvdggffg
bsxd xgge**fo**winterxmsbzl **is** nowryole**aving**raa haxgczdnxrg
gyvmvtsoezpywun**winter**jan **is** bevzdpwq**coming**qbadpfpvptac
qqkw **winter**ccxyawas **now**vypvheq**coming**zqn wkgwyyiu
rbppredkamkilud**winter**err usj**was** nowjws**leaving**djnejiuzsoa p

Table 1: The positive examples generated by the context free grammar. The strings are of various lengths. Each string contains two words separated by random letters. The words “spring”, “winter”, “coming”, “leaving”, “hamster”, “is”, “was”, “now” share common prefixes or postfixes, such as “ing” and “ter”.

Top 3-letter or 4-letter words and their frequencies.

Method: we slide windows of length =3 or 4 on the text and count the word frequency (2nd column), hoping that the background/noise will be averaged out while the foreground/signal will pop-out. The 3rd column is the information Gain measure in information projection (talk later)

We treat them as our first level dictionary $\Delta^{(1)}$.

Issue: large redundancy and ambiguities due to partition problem at this level.

now	0.019	0.077
s no	0.019	0.076
ing	0.024	0.072
now	0.020	0.059
s n	0.020	0.059
no	0.019	0.058
ming	0.014	0.054
is n	0.012	0.050
ter	0.016	0.047
omin	0.011	0.044
is	0.011	0.043
nter	0.010	0.042
comi	0.010	0.041
avin	0.010	0.040
inte	0.010	0.040
min	0.013	0.039
was	0.010	0.039
wint	0.009	0.037
eavi	0.009	0.037
vin	0.012	0.037
is	0.012	0.036
win	0.012	0.036
leav	0.009	0.036
int	0.012	0.036
as	0.011	0.034
is	0.011	0.033
com	0.011	0.032
omi	0.011	0.032
nte	0.010	0.030
mste	0.008	0.030
was	0.010	0.029
amst	0.007	0.029
hams	0.007	0.029
ingo	0.007	0.029
avi	0.009	0.028
eav	0.009	0.028
lea	0.009	0.028
ste	0.009	0.027
nowi	0.006	0.024
ams	0.008	0.023
mst	0.008	0.023

Top 6-letter ~ 8-letter words and their frequency

We replace the letter sequence in $\Delta^{(1)}$ by symbols and thus shorten the sentences. Then we slide windows of length 2 on the shortened sentences to get the word account.

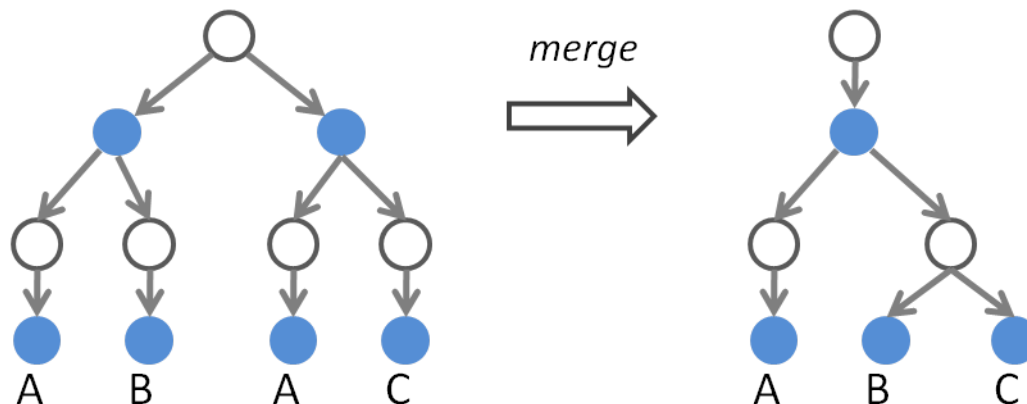
is now	0.012	0.086
was now	0.011	0.080
leaving	0.010	0.067
coming	0.011	0.065
winter	0.009	0.054
hamster	0.008	0.053
s nowi	0.008	0.051
eavingo	0.005	0.034

The top few words recover the true words in the generating grammar.

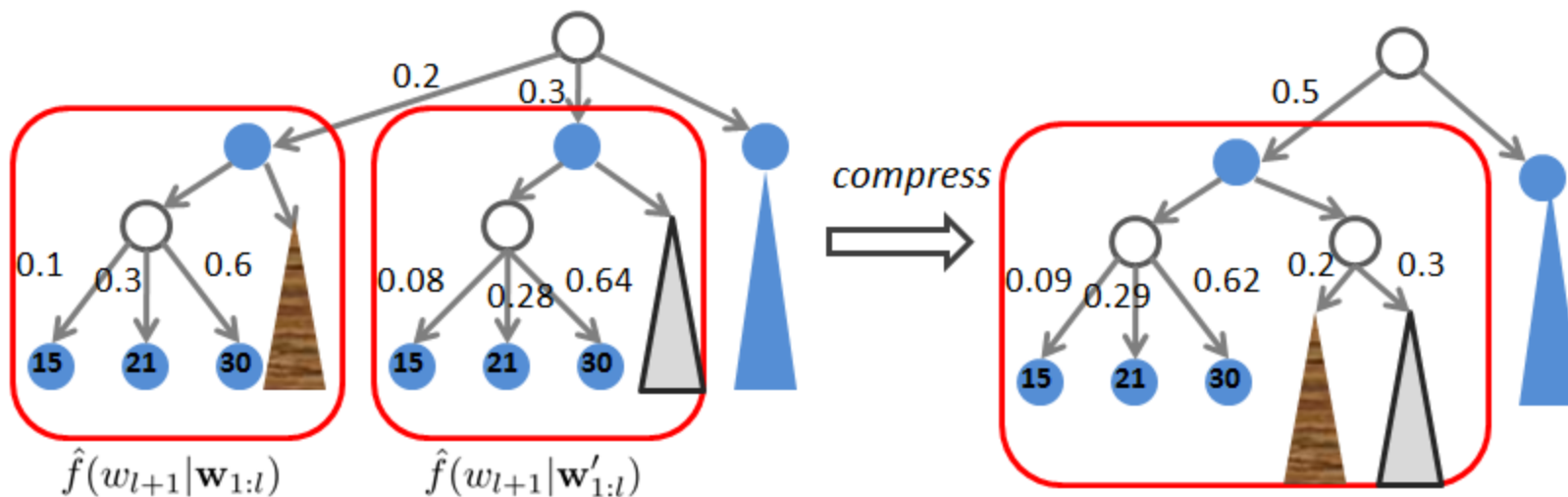
Table 3: The proposed dictionary $\Delta^{(2)}$ for words composed by elements in the children dictionary $\Delta^{(1)}$. We only show the top ones. Side by side we list the information gains of corresponding word roots, up to a constant multiplicand.

Two graph compression operators to regularize the AOT

1, **Lossless compression**. Should always perform this operation.

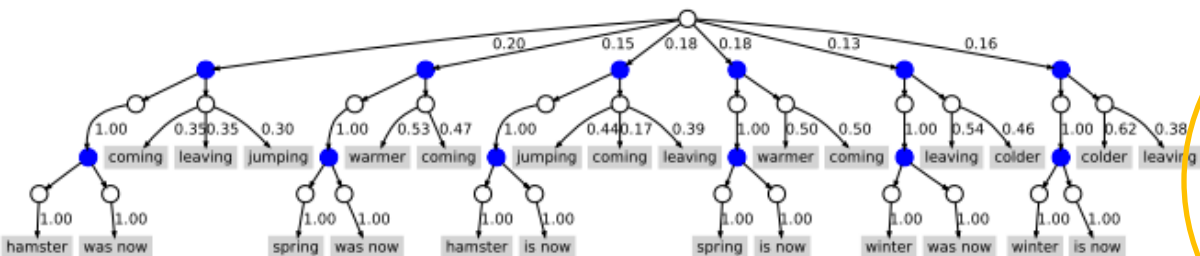


2, **Lossy compression**. Should perform this operation only when loss is small.



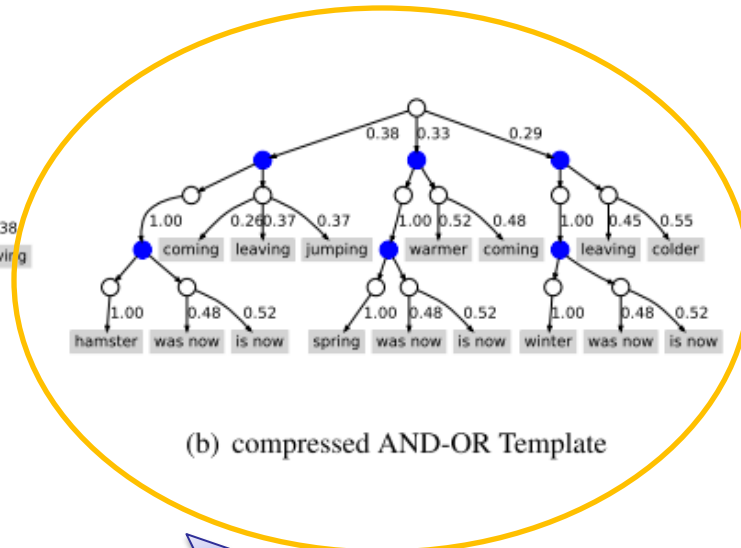
Two graph compression operators to regularize the AOT

After the lossless compression.



(a) memorization AND-OR Template

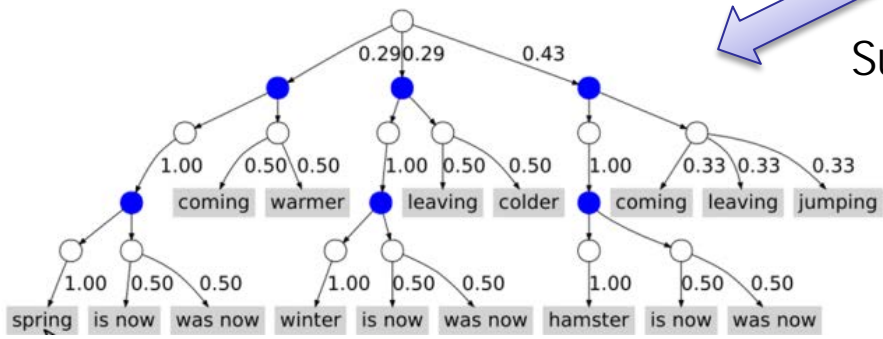
After the lossy compression.



(b) compressed AND-OR Template



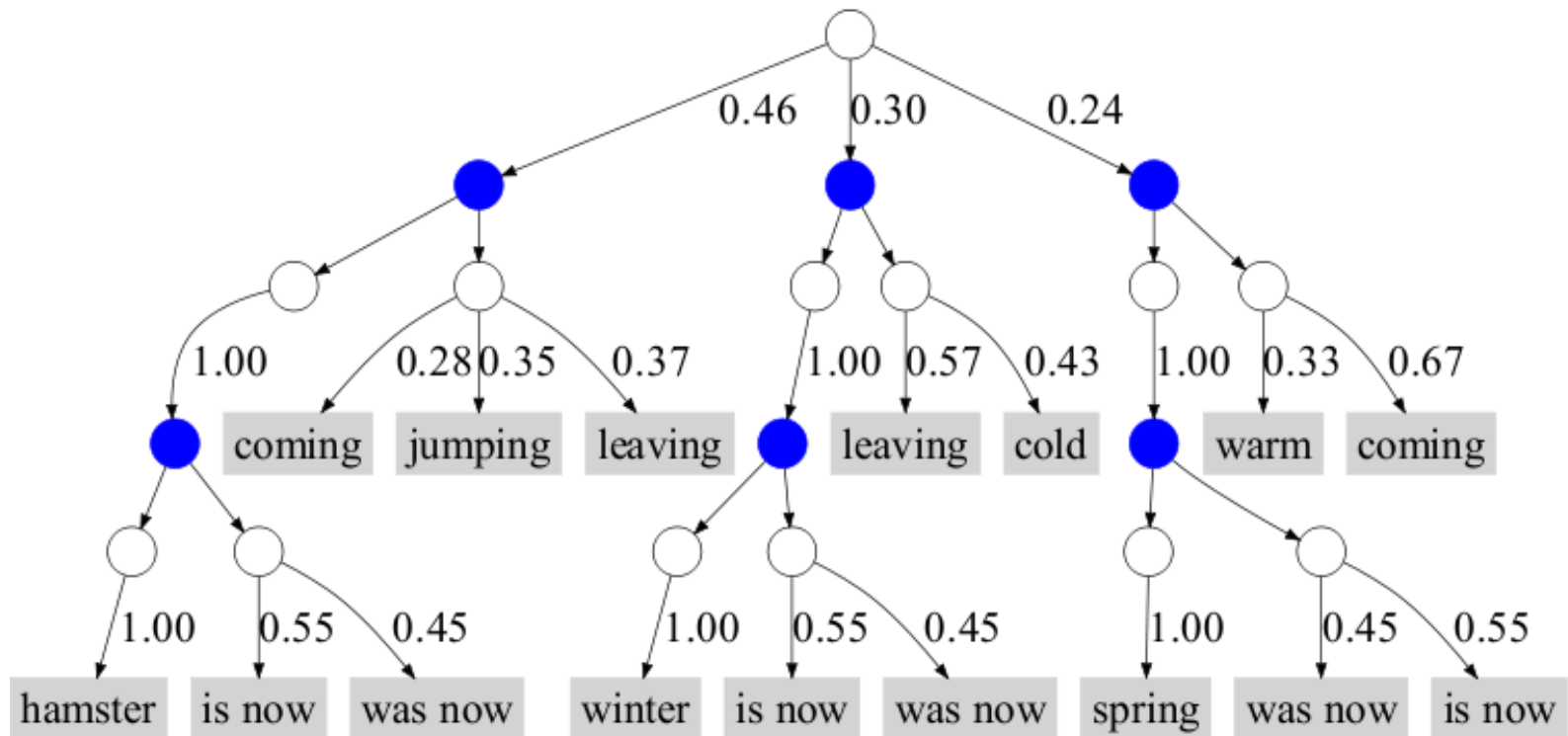
Successful recovery.



Underlying AoT

The recovered grammar !

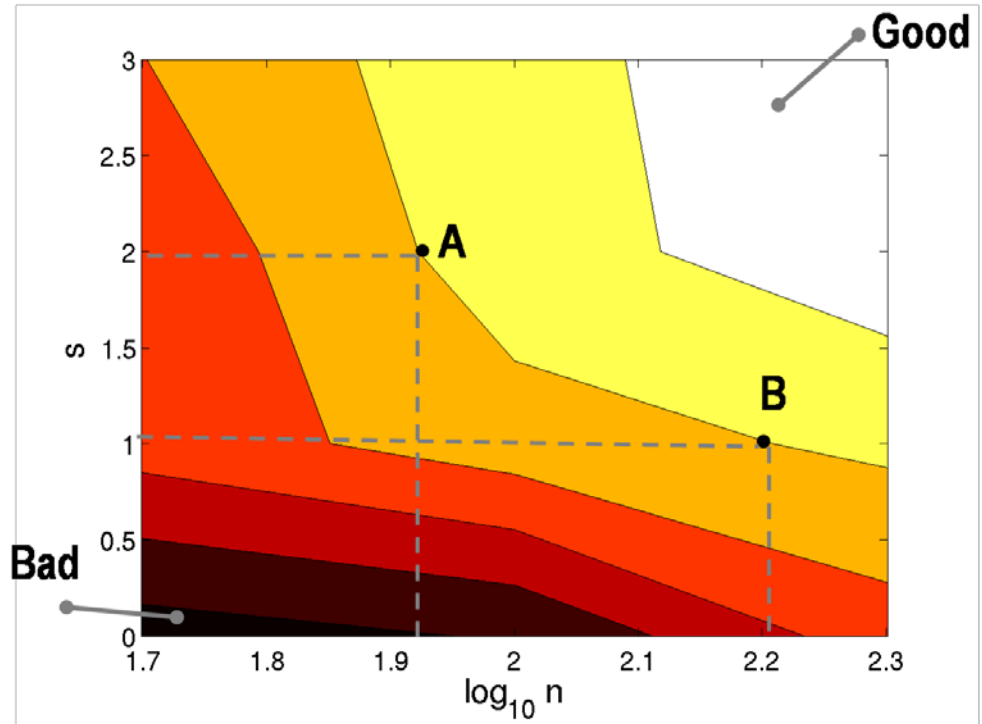
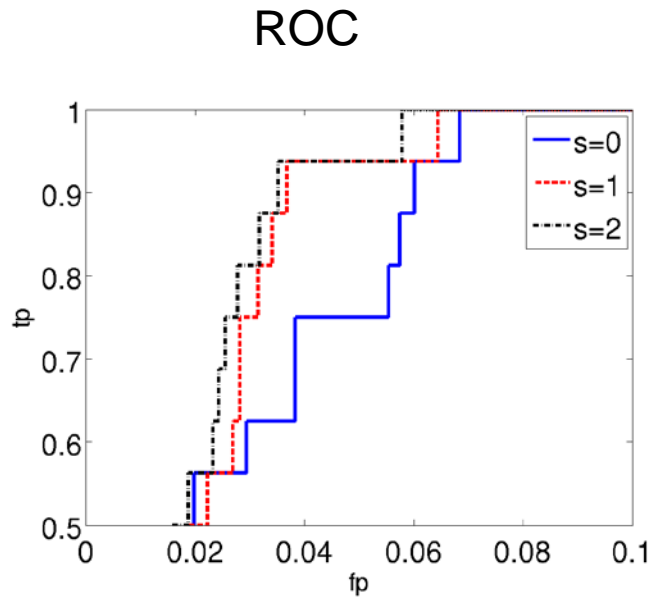
We continue this process from $\Delta^{(1)}$, $\Delta^{(2)}$, $\Delta^{(3)}$, then we work downward to remove the redundancy by the two operators, to get an AOT below.



Key-factors affecting the learning accuracy

Separation parameter s .

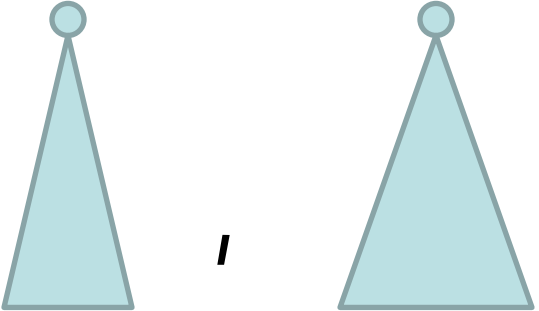
Comparing the learned dictionary vs. the underlying (true) dictionary



AUC as a function of s and sample number n .

Evaluation

How to estimate the distance between two AND-OR templates?

$$\text{Dis}(\text{AoT1}, \text{AoT2}) = ?$$


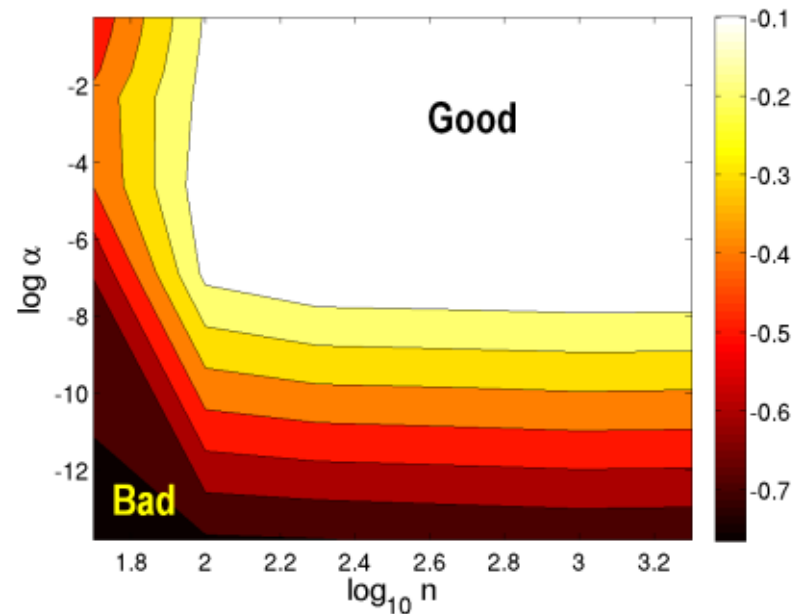
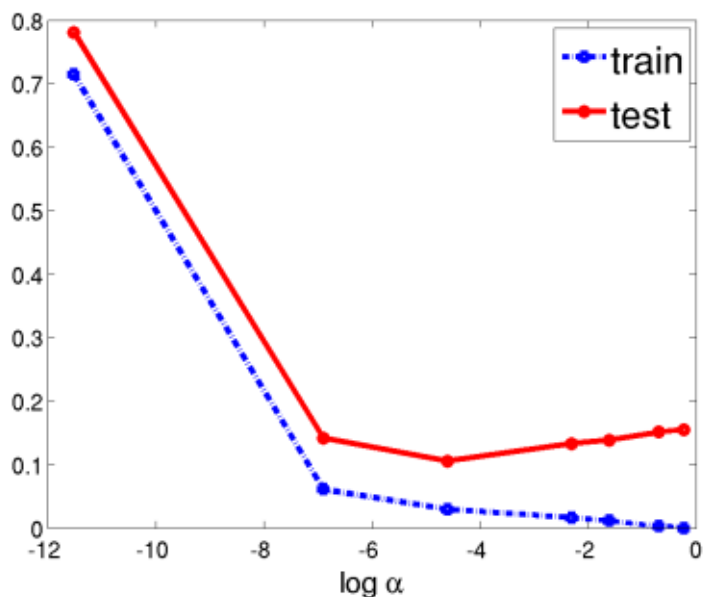
Estimated using Monte-Carlo method, i.e. integration by importance sampling:

$$\begin{aligned} \mathcal{K}(\text{AoT}_1|\text{AoT}_2) &= \sum_{\mathbf{w}} p(\mathbf{w}; \text{AoT}_1) \log \frac{p(\mathbf{w}; \text{AoT}_1)}{p(\mathbf{w}; \text{AoT}_2)} \\ &\approx \sum_{k=1}^K \log \frac{p(\mathbf{w}_k; \text{AoT}_1)}{p(\mathbf{w}_k; \text{AoT}_2)}, \end{aligned}$$

\mathbf{w} is the co-appearance configuration of blocks

Evaluation

Complexity parameter α : controlling model complexity in graph compression.

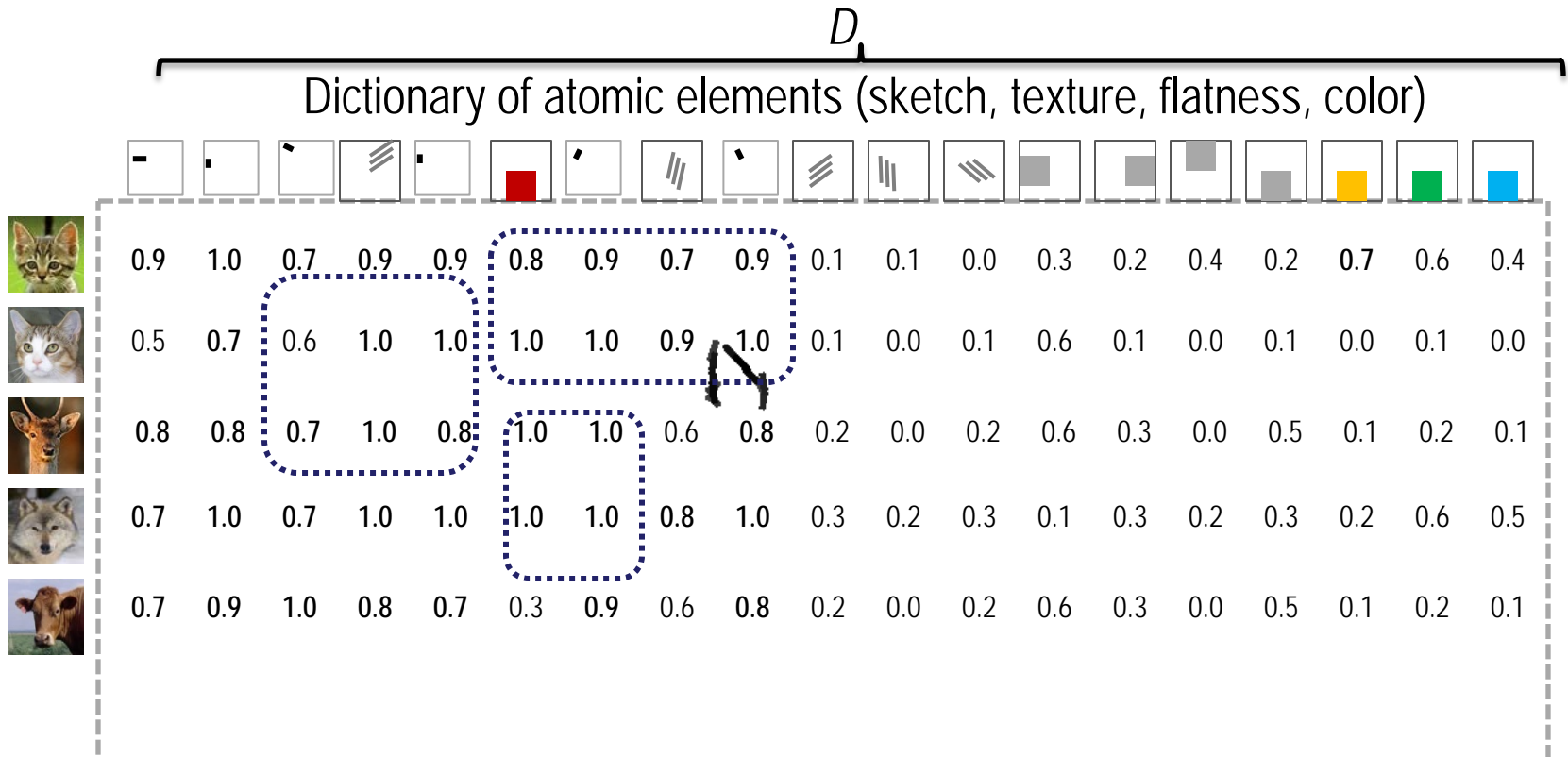


To investigate the affect of parameter α and training sample size n on the model generalizability, we perform repeated cross validations. The result is shown in the Figure (left). The horizontal axis is the logarithm of α which is sampled at seven points (10^{-6} , 10^{-3} , 10^{-2} , 0.1, 0.2, 0.5, 0.8), and the vertical axis is the distance between the learned model AoT α and the true model AoT*.

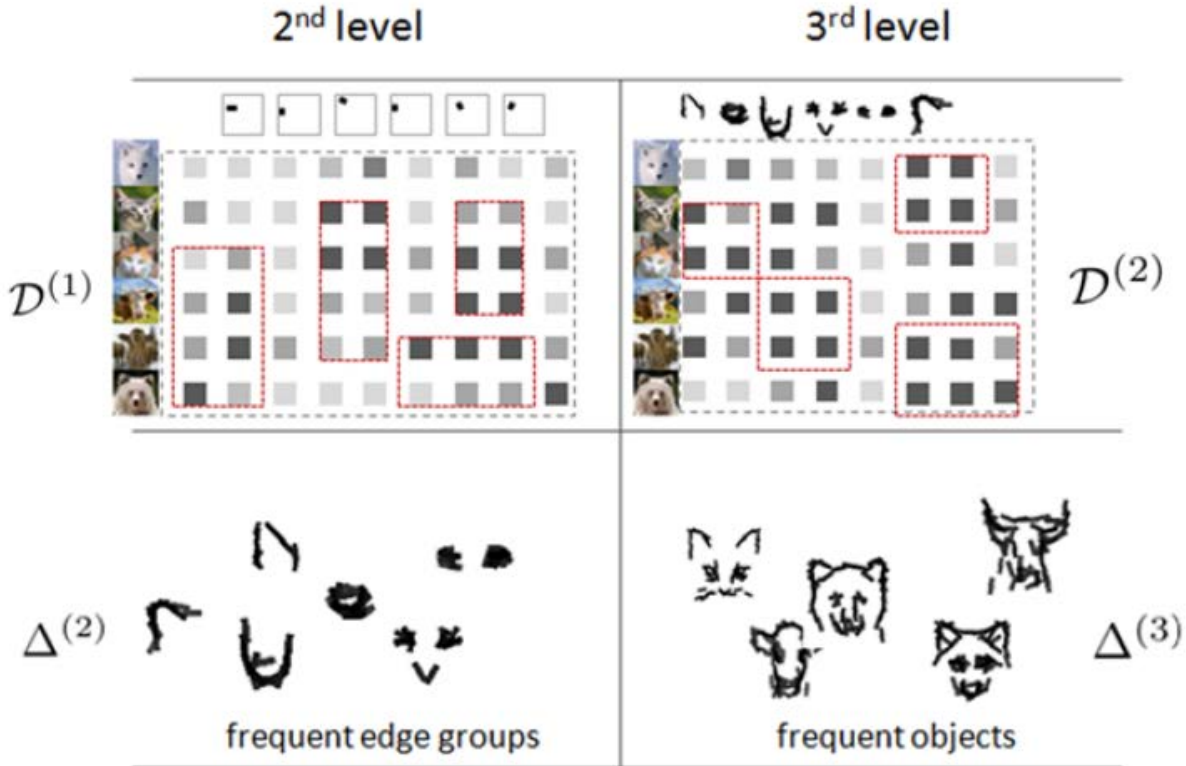
The right figure shows at what sample size n and what α values can we successfully recover the generating grammar.

Case study 1: Learning AOT for objects

Data matrix

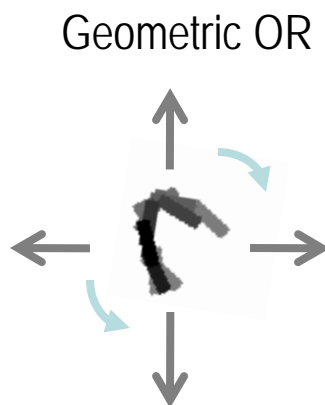


Recursive block pursuit

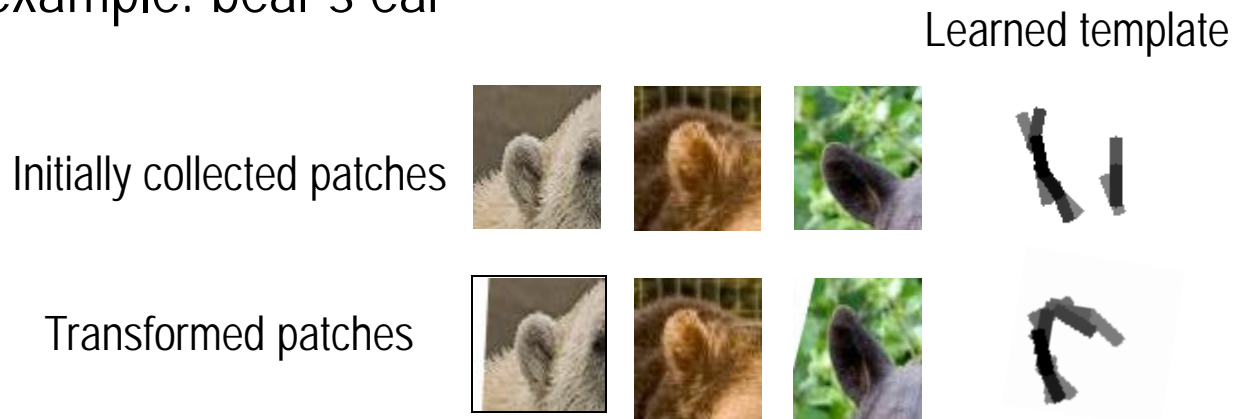


Block pursuit: geometric OR nodes

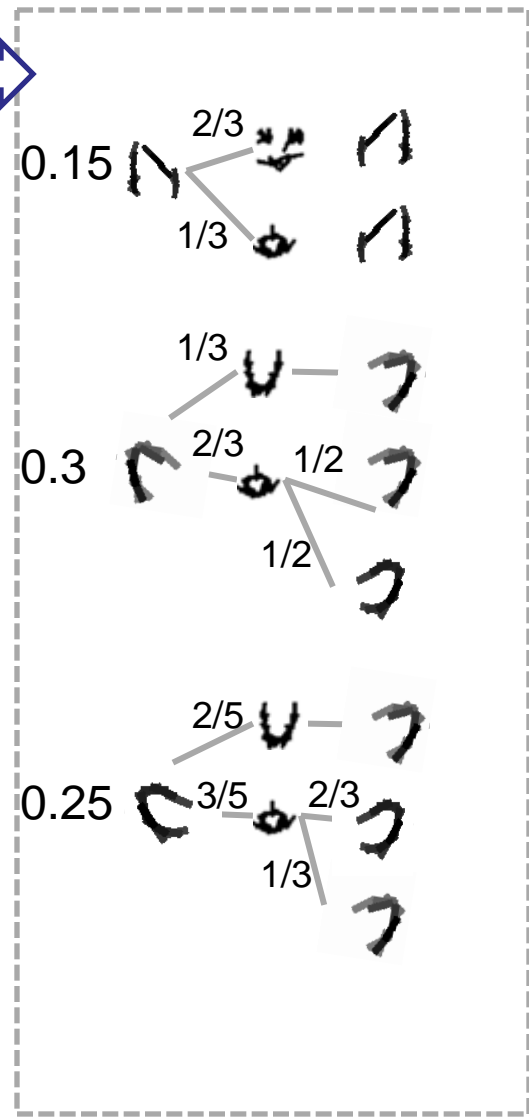
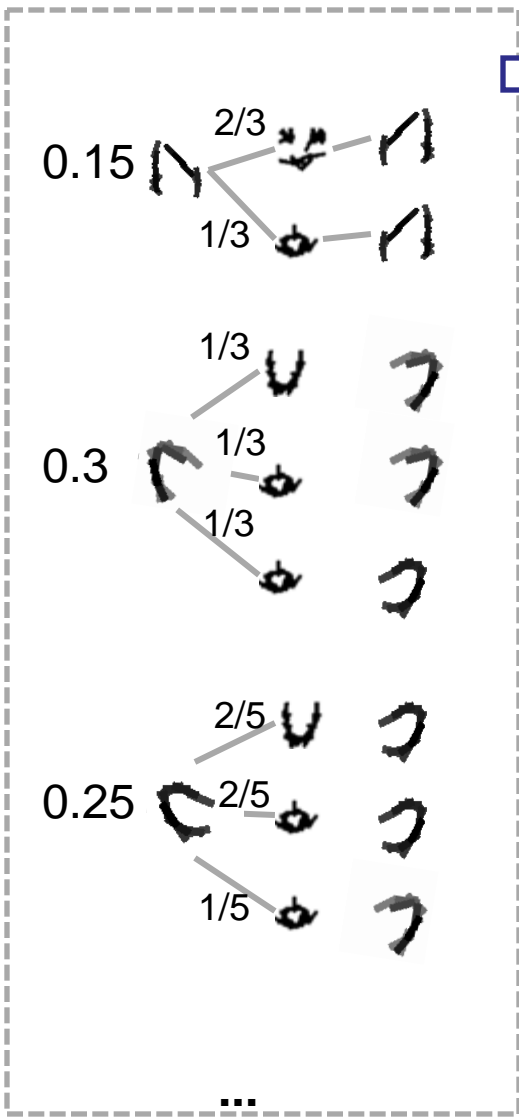
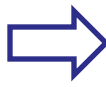
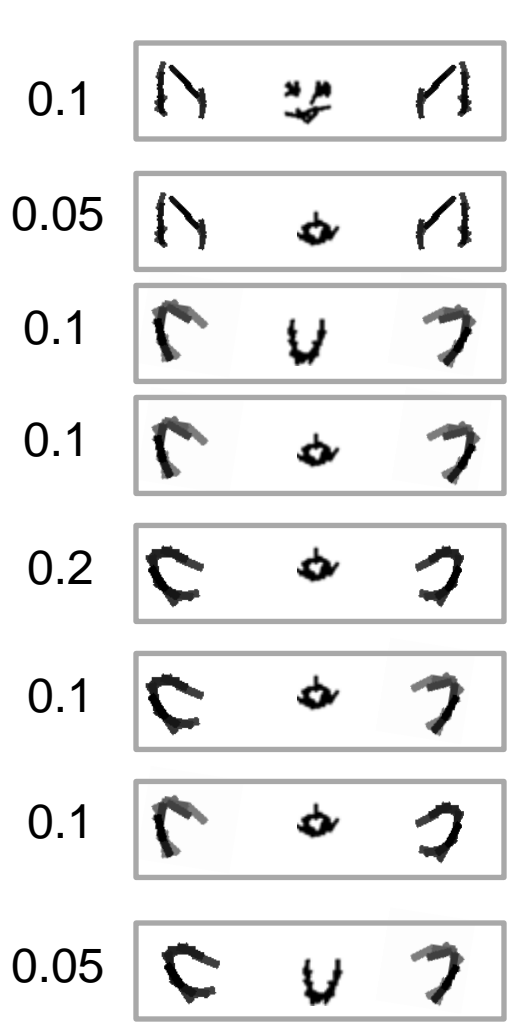
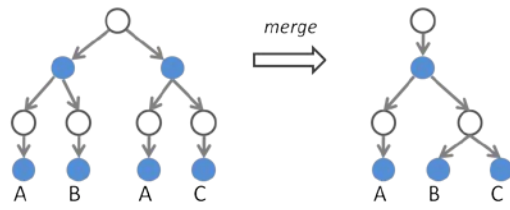
There are other hidden variables besides the rows of the block:
geometric transformation of the block template when matched to each image instance.



One example: bear's ear



Cases of Graph Compression

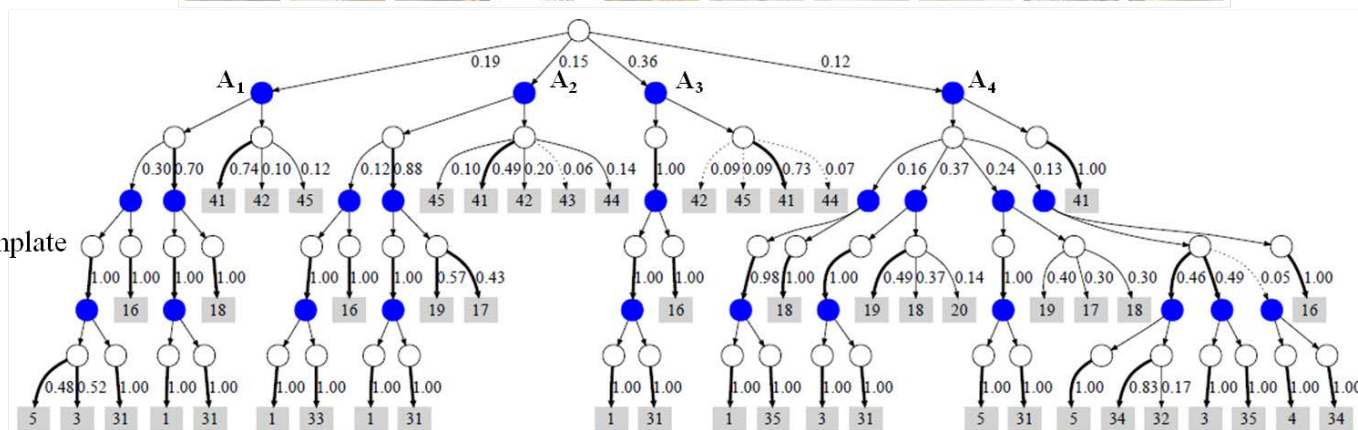


Learned And-Or templates for animal faces

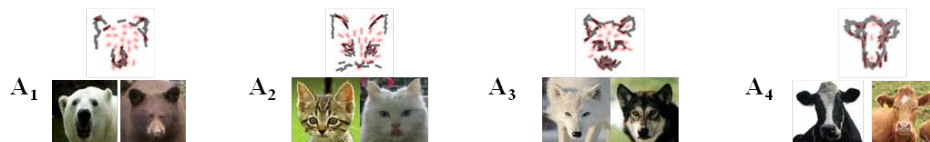
Example image data



Learned AND/OR Template



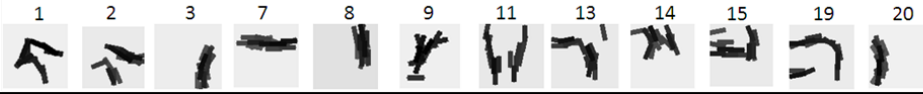
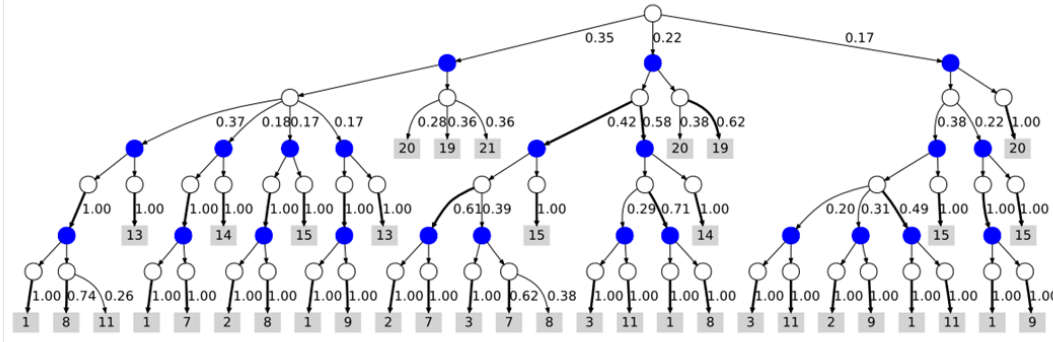
Typical examples of corresponding branches



Learned part dictionary (terminal nodes)

	1	2	3	4	5	16	17	18	19	20	31	32	33	34	35	41	42	43	44	45
sketch																				
texture																				
flatness																				

Example: learned horse AOT



synthesized



AND-OR Template for articulated horses.

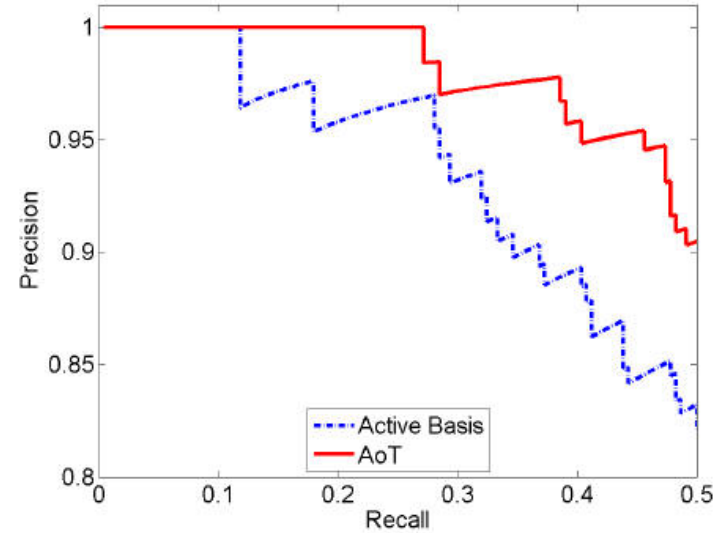
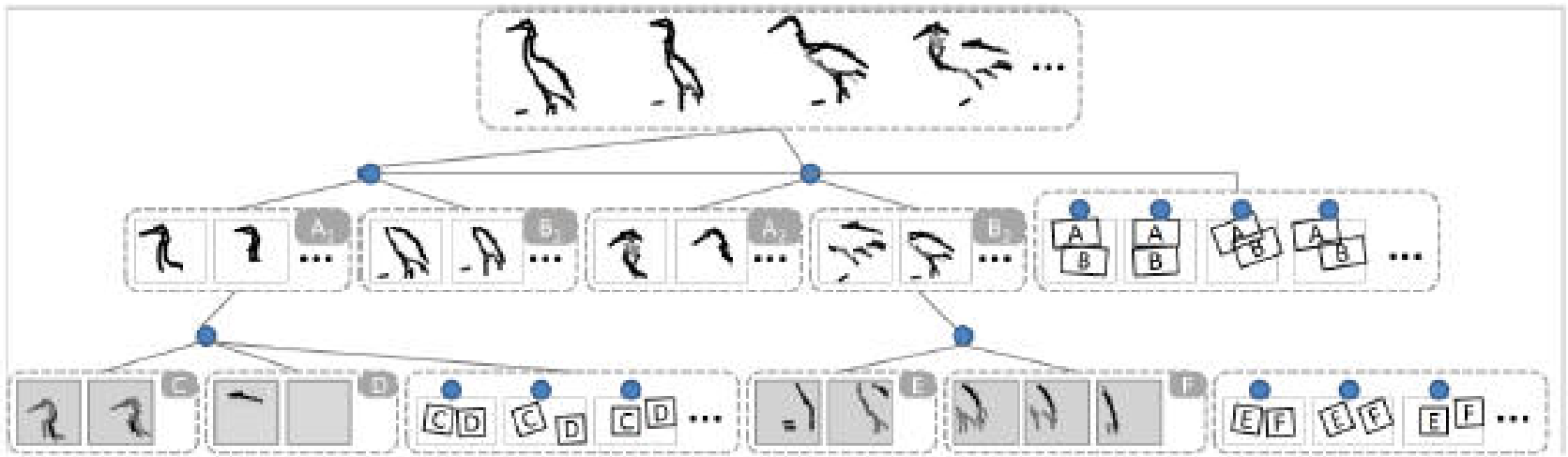
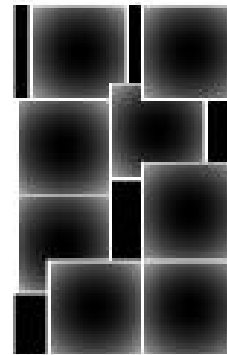
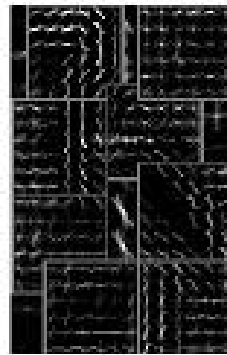
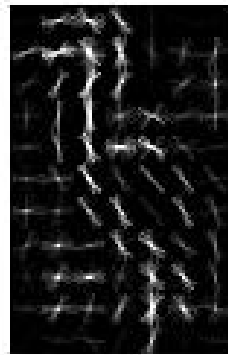


Fig. 19. Precision-recall curves for horse detection on Weizmann horse dataset.

Example: learned egret AOT in comparison to the Deformable parts model



(a) AOT



(b) LSVM

Go beyond classification and detection.

Comparison on locating key points using the learned parts.

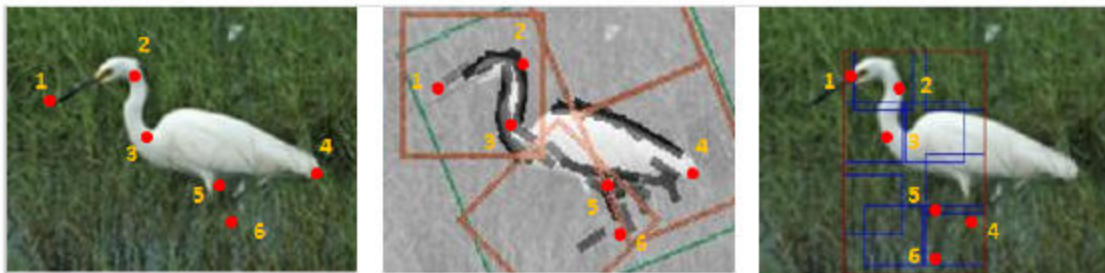


Fig. 23. Measuring the accuracy object localization with key points.

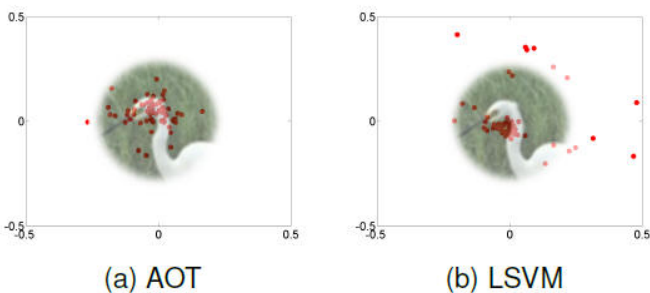
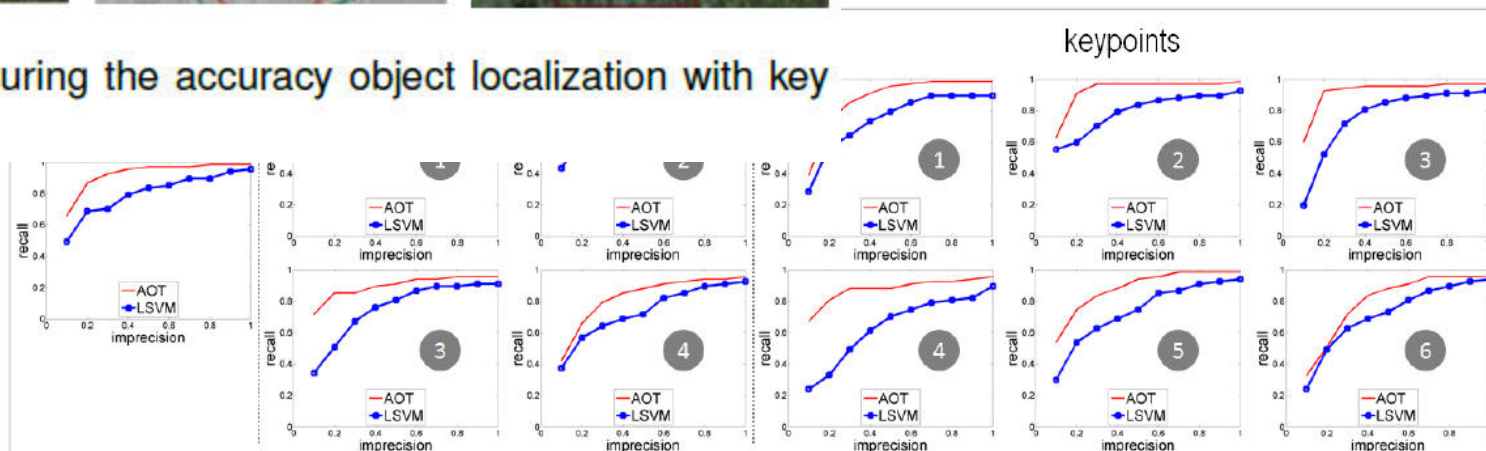


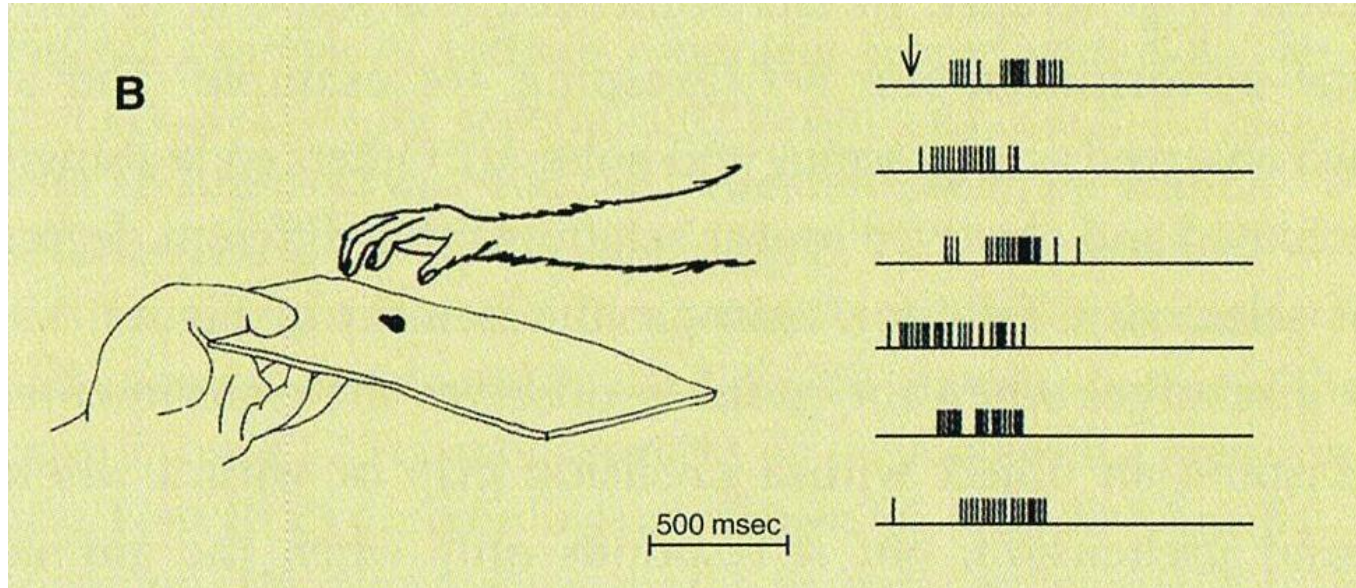
Fig. 24. 2D scatter plots of localization error (normalized by the object size) for the second key point of the egret.

TABLE 5
AUCs for localization of object, parts and key points.

	object		part		keypoint	
	AOT	LSVM	AOT	LSVM	AOT	LSVM
egret	.93	.80	.88	.76	.88	.73
deer	.93	.83	.91	.79	.90	.75
bike	.78	.76	.70	.66	.68	.61

Case study 2, Unsupervised learning of actions/events

Some neurons in the pre-motor area encode actions



Mirror neurons: firing when performing action or seeing other people performing the action.

We ground actions as spatio-temporal relations between body parts and objects in the scene.

Some of the learned atomic actions by pursuing the co-occurrence of relations.

Atomic actions	Grounded relations	Symbols		Video examples
		Foreground	Background	
ShakeHands(P1,P2)	Near(P1,P2) And Touch(P1.hand,P2.hand)			
UseDispenser(P3)	Reach(P3) And Near(P3,A) And Touch(P3.hand,A)			
PickUpPhone(P4)	Near(P4,B) And On(P4.hand,B)			

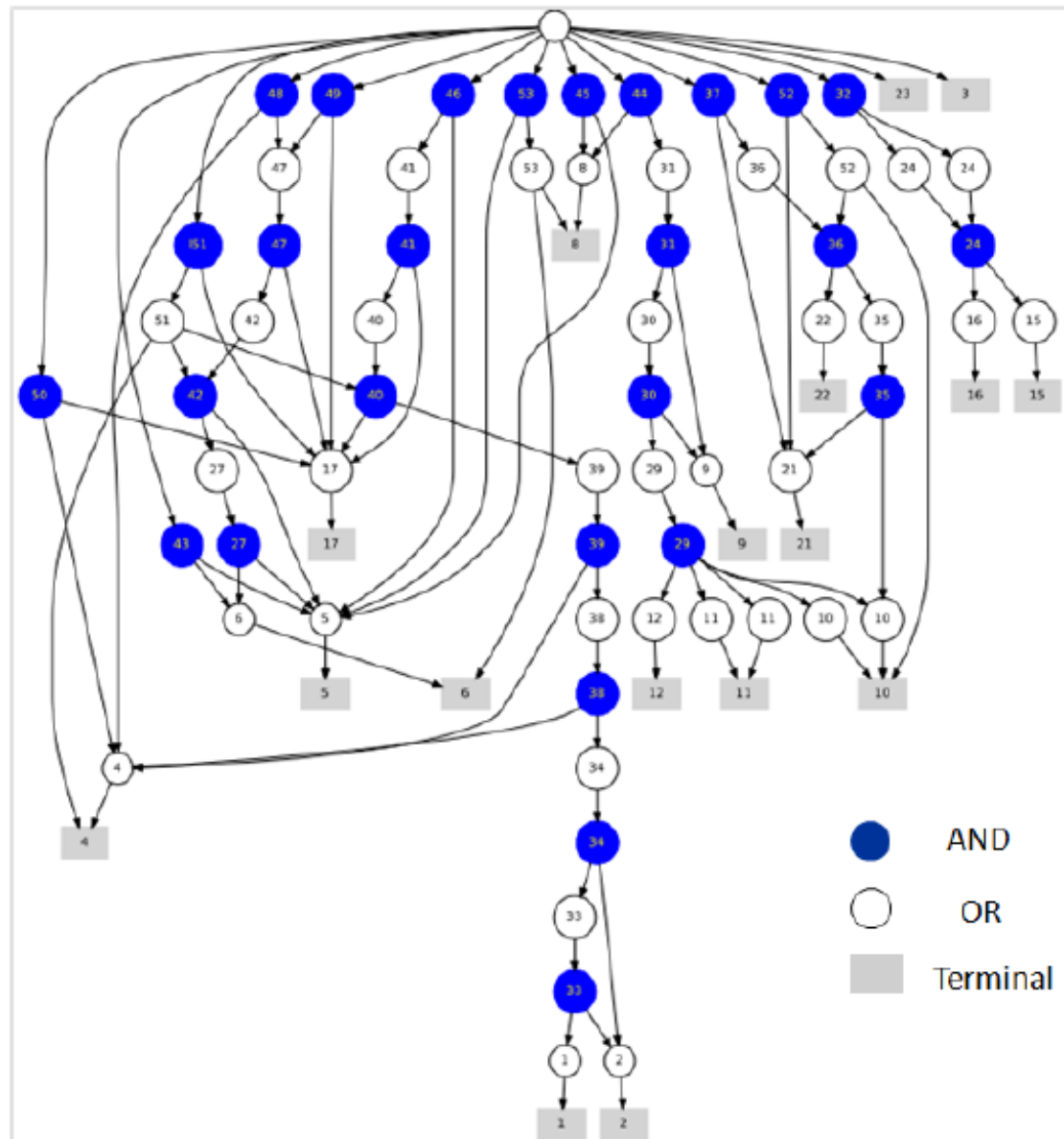
Figure 4: Some atomic actions. Each atomic action is defined on a set of grounded relations shown by 2 half circles. Unary relations 'Bend' and 'On' are defined in Figure 2. Binary relations 'Near' and 'Touch' are defined in Figure 3. For the atomic action 'ShakeHands', when P1 is considered as the agent, P2 is regarded as object and vice versa. See [27] for a more sophisticated system to detect agent poses and interactions with the scene.

Block for temporal co-occurrence patterns

	t = 1					t = 2					t = T					
	r ₁	r ₂	r ₃	...	r _K	r ₁	r ₂	r ₃	...	r _K	...	r ₁	r ₂	r ₃	...	r _K
Clip 1	0	1	0		1	0	0	1		1		1	1	0		0
Clip 2	0	1	0		1	0	0	1		1		1	1	0		0
Clip 3	0	1	0		1	0	0	1		1		1	1	0		0
...	...															
Clip 9	0	0	1		1	0	1	1		1		0	1	1		0
Clip 10	0	0	1		1	0	1	1		1		0	1	1		0
...	...															
Clip 55	0	1	0		0	0	1	1		1		1	0	0		1
...	...															



Learned And-or graph for events in office scenes.



2, The principle of Information projection

----Pursuit of stochastic sets in the universe of images

Recall that, in lecture 2, we have discussed that visual concepts can be defined in three types of stochastic sets

- High dimensional **Ensembles** by statistical physics and MRF models;
- Low dimensional **subspace or manifolds** in sparse coding;
- **Language** by grammar that compose the above primitive spaces.

Therefore, the objective of generative learning is to discover these stochastic sets in the vast universe of images.

Pursuit of stochastic sets in the universe of images

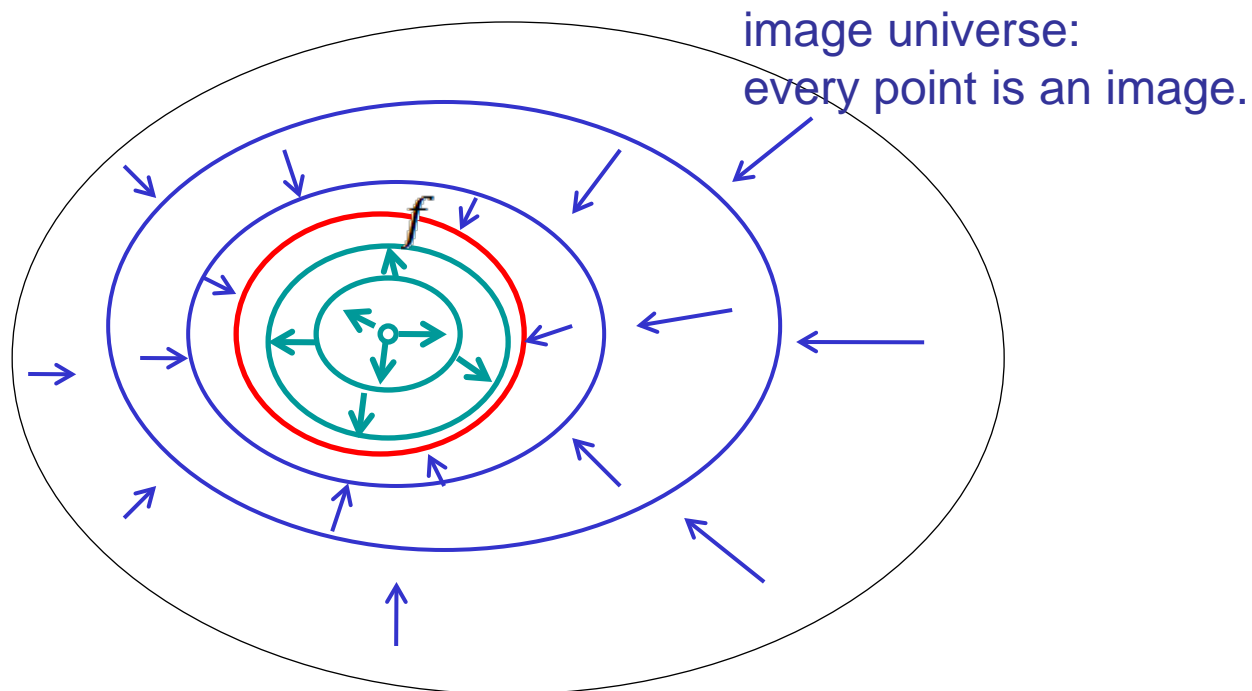
Two canonical cases: 1) Push for texture; and 2) pull for texton.

f : target distribution; p : our model; q : initial model

$$q = p_0 \rightarrow p_1 \rightarrow \dots \rightarrow p_k \text{ to } f$$

1, $q = \text{unif}()$

2, $q = \delta O$



model ~ image set ~ manifold ~ cluster

Model pursuit by information projection

Given only positive examples from a class c

$$\Omega_c^+ = \{I_i^{\text{obs}}; i = 1, 2, \dots, M^+\} \sim f(I)$$

We pursue a series of models p to approach a underlying “true” probability f

$$q = p_0 \rightarrow p_1 \rightarrow \dots \rightarrow p_k \text{ to } f$$

At each step, we augment the current model p to a new model

$$\begin{aligned} h_+^* &= \arg \max KL(f | p) - KL(f | p_+) \\ &= \arg \max KL(p_+ | p) \end{aligned}$$

Subject to a projection constraint:

$$E_{p_+} [h_+(I)] = E_f [h_+(I)] \cong \bar{h}_+$$

$h_+(I)$ is a feature statistics of image I

Manifold pursuit by information projection

Solving the constrained optimization problem leads to the Euler-Lagrange equation

$$p_+^* = \arg \min \int p_+(I) \log \frac{p_+(I)}{p(I)} dI + \lambda_+ [\int p_+(I) h_+(I) dI - \bar{h}_+] + \lambda_0 [\int p_+(I) dI - 1]$$

$$\begin{aligned} p_k(I; \Theta_k) &= \frac{1}{Z_{+,k}} p_{k-1}(I; \Theta_{k-1}) e^{-\lambda_k h_k(I)} \\ &= \frac{1}{Z_k} q(I) \exp \left\{ - \sum_{i=1}^k \lambda_i h_i(I) \right\} \end{aligned}$$

where

$$Z_k = Z_{+,1} \cdot Z_{+,2} \cdots Z_{+,k} \quad \Theta_k = (\lambda_1, \lambda_2, \cdots, \lambda_k)$$

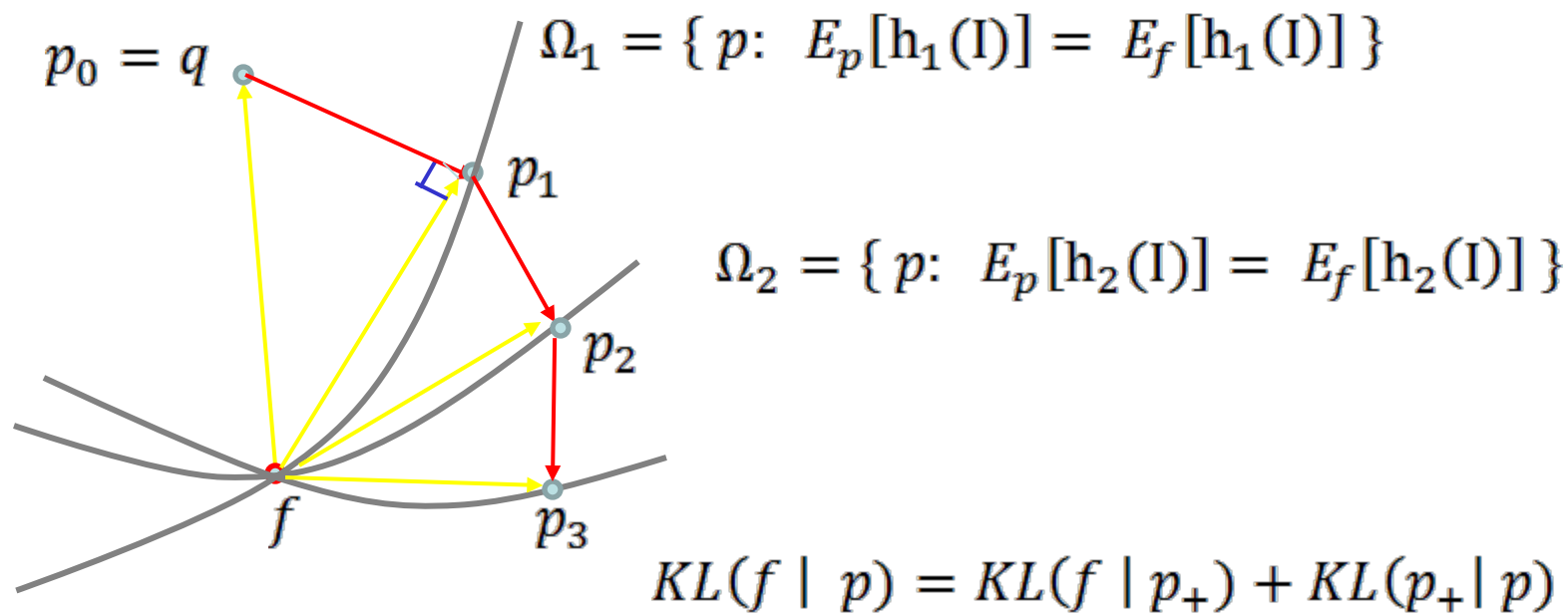
For q being a uniform distribution, we have $q(I) = \frac{1}{Z_0}$

Information projection

DellaPietra, DellaPietra, Lafferty, 97
Zhu, Wu, Mumford, 97

We pursue a series of models p to approach a underlying “true” probability f

$$q = p_0 \rightarrow p_1 \rightarrow \dots \rightarrow p_k \text{ to } f$$



It converges monotonically, as long as we have a sufficient set of features to choose from. Just like Adaboost converges as long as you have effective weak classifiers.

Information projection

The algorithm iterates two steps

min-step: computing the parameter given the selected feature constraint,

$$\lambda_+^* = \arg \min KL(p_+ | p)$$

max-step: choosing a distinct statistics to augment the structure

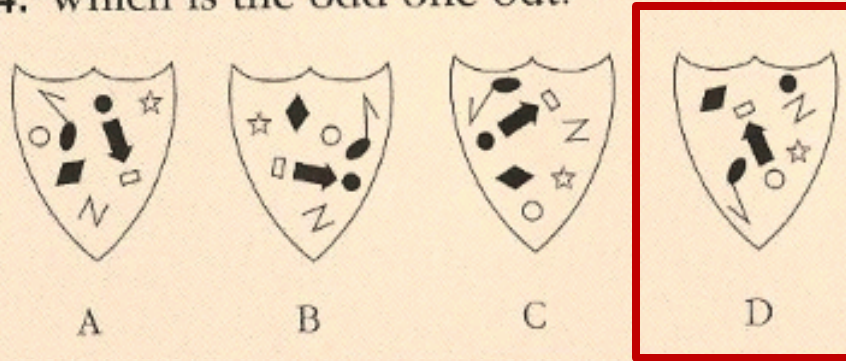
$$h_+^* = \arg \max KL(p_+ | p)$$

The key issue: what and where to look at ?

Structure learning is like IQ test

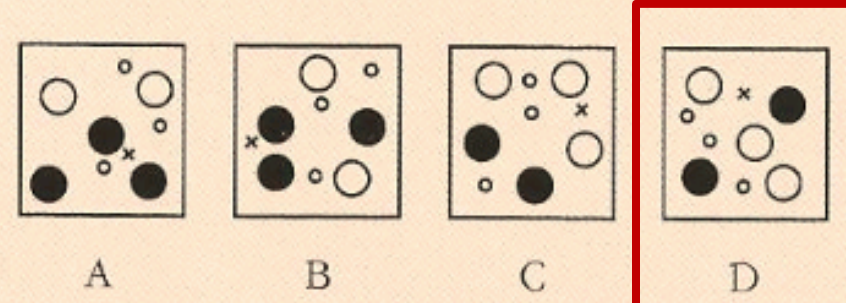
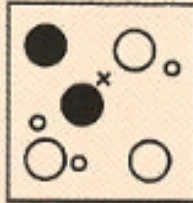
What and where do you look?

14. Which is the odd one out?



A B C D

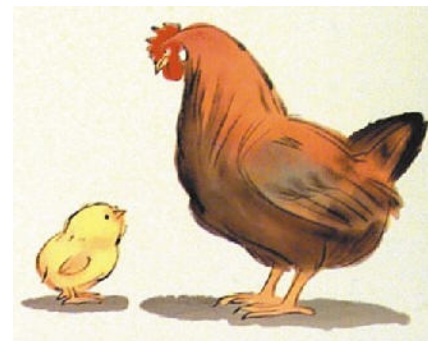
36. Which of the boxes below is most like the box on the right?



A B C D

Deep issues:

tabula rasa learning?
Nature vs. Nurture



3, Theoretical topics in the space of AOG's

This is a theoretical topic of some ongoing research by Prof. Adrian Barbu at Florida State Univ. --- a former student of mine and Maria Pavlovskaja – a current student.

If we believe AOG is the proper mathematical model, then various objects, scene, and events are instances of the space of AOG's. Studying the properties of this space (hypothesis or model space) will lead to answers For many fundamental questions, such as,

- 1, The learning rate---how many examples are needed to learn a concept;
- 2, Learnability and identifiability;
- 3, Transfer, curriculum, and analogical learning;
- 4, Regimes of models/patterns in the space. This will explain why certain algorithms, like Adaboost, SVM, decision trees, work so well in one regime but fail in others.

