Conditional Random Fields

Grant “Tractable Inference” Van Horn and
Milan “No Marginal” Cvitkovic

Recap: Discriminative vs. Generative Models

Suppose we have a dataset $\{(\mathbf{x}, \mathbf{y})^i\}_{i=1}^N$ drawn from $p(\mathbf{x}, \mathbf{y})$

Generative models try to learn $p(\mathbf{x}, \mathbf{y})$

- Hard problem; always requires assumptions
- Captures all the nuance of the data distribution (modulo our assumptions)

Discriminative models try to learn $p(\mathbf{y}|\mathbf{x})$

- Simpler problem; often all we care about
- Typically doesn't let us make useful modeling assumptions about $\mathbf{y}$

Recap: Graphical Models

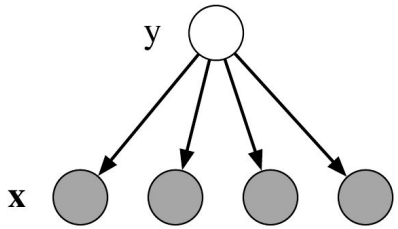
One way to make generative modeling tractable is to make simplifying assumptions about which variables affect which other variables.

These assumptions can often be represented graphically, leading to a whole zoo of models known collectively as **Graphical Models**.

Recap: Graphical Models

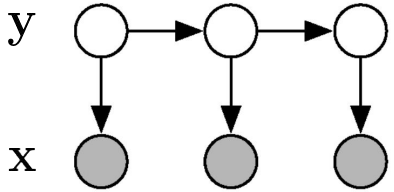
One zoo-member is the **Bayesian Network**, which you've met before:

Naive Bayes



$$p(\mathbf{x}, y) = p(y) \prod_i p(\mathbf{x}_i|y)$$

HMM



$$p(\mathbf{x}, \mathbf{y}) = \prod_i p(\mathbf{x}_i|\mathbf{y}_i)p(\mathbf{y}_i|\mathbf{y}_{i-1})$$

The vertices in a Bayes net are random variables.

An edge from A to B means roughly: "A causes B"
Or more formally: "B is independent of all other vertices when conditioned on its parents".

Figures from Sutton and McCallum, '12

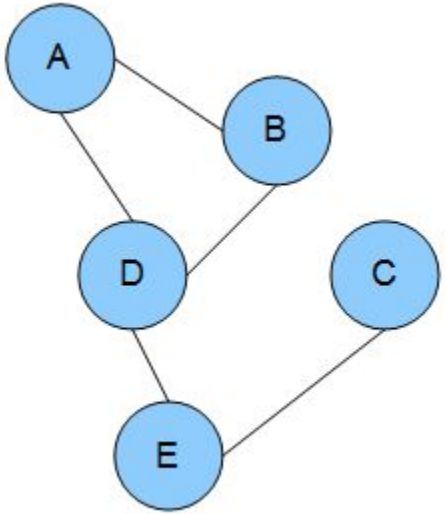
Recap: Graphical Models

Another is the **Markov Random Field (MRF)**, an undirected graphical model

Again vertices are random variables.

Edges show which variables depend on each other:

$$p(A|B,C,D,E) = p(A|B,D)$$



Thm: This implies that

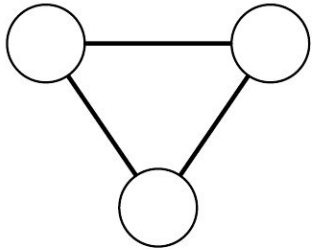
$$p(A,B,C,D,E) = F_1(A,B,D)F_2(C,E)F_3(D,E)$$

Where the $F_i(\mathbf{x})$ are called "factors". They measure how compatible an assignment of subset of random variables are.

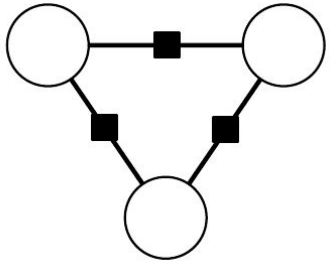
Figure from Wikipedia

Recap: Graphical Models

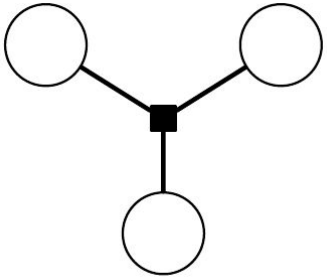
Then there are **Factor Graphs**, which are just MRFs with bonus vertices to remove any ambiguity about exactly how the MRF factors



An MRF



A factor graph for this MRF



Another factor graph for this MRF

Circles and squares are both vertices; circles are random variables, squares are factors.
The arguments to each factor/square are its neighboring random variables.

Figures from Sutton and McCallum, ‘12

Recap: Graphical Models

Thm: The joint distribution $p(\mathbf{x}, \mathbf{y})$ for any factor graph can be written as

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \prod_a e^{\Psi_a(\mathbf{x}_a, \mathbf{y}_a; \theta_a)} = \frac{1}{Z} \prod_a e^{\theta_a^T f_a(\mathbf{x}_a, \mathbf{y}_a)}$$

Where the $f_a(\mathbf{x}_a, \mathbf{y}_a)$ are "feature function",

the θ_a are vectors of weights,

the $\Psi_a(\mathbf{x}_a, \mathbf{y}_a; \theta_a) = \theta_a^T f_a(\mathbf{x}_a, \mathbf{y}_a)$ are "potential functions",

and $Z = \sum_{\mathbf{x},\mathbf{y}} \prod_a e^{\Psi_a(\mathbf{x}_a, \mathbf{y}_a; \theta_a)}$ is the "partition function" (which normalizes things)

Best of both worlds?

Graphical models are great for letting us express relationships among our output variables. And we can still use use them for discriminative tasks by calculating

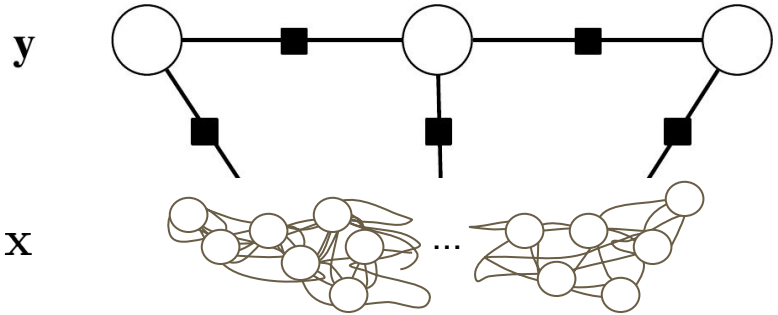
$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})}$$

But $p(\mathbf{x}, \mathbf{y})$ and (especially) the marginal $p(\mathbf{x})$ are so hard to model and compute, especially when all I care about is $p(\mathbf{y}|\mathbf{x})$!

Isn't there a way for me to *both* model relationships between my output variables *and* only train on the discriminative task I care about?

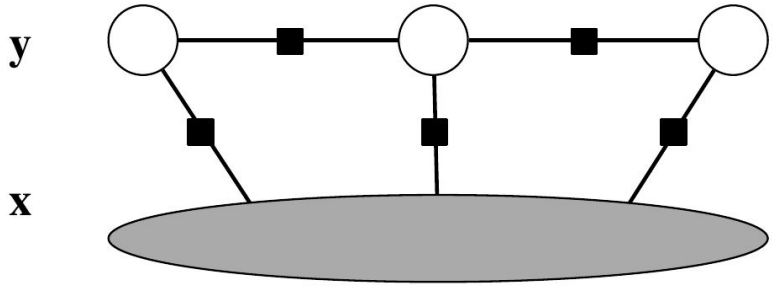
Conditional Random Fields

Sure! Just define a model that's a factor graph *when you condition on the input.* Then **Conditional** on your input, you've got a **Random Field**, or a **CRF.**



Gnarly, full joint modeling problem

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \prod_a e^{\Psi_a(\mathbf{x}_a, \mathbf{y}_a; \theta_a)}$$



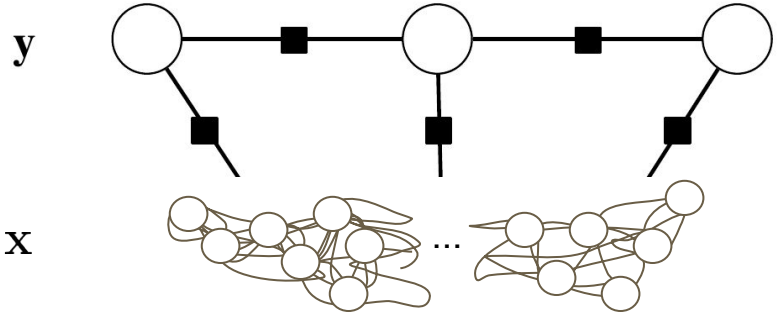
Easier, conditional modeling problem

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_a e^{\Psi_a(\mathbf{x}, \mathbf{y}_a; \theta_a)}$$

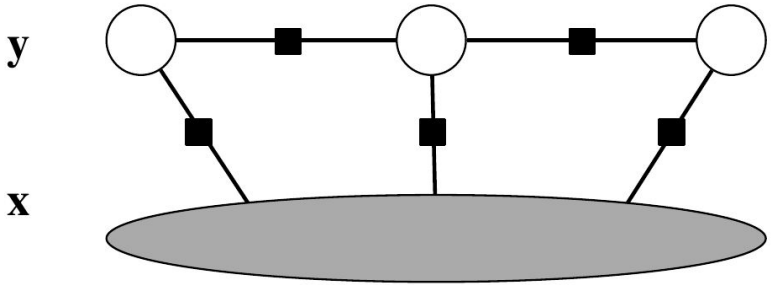
Figures from Sutton and McCallum, '12

Conditional Random Fields

Sure! Just define a model that's a factor graph *when you condition on the input.* Then **Conditional** on your input, you've got a **Random Field**, or a **CRF.**



Gnarly, full joint modeling problem



Easier, conditional modeling problem

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \prod_a e^{\Psi_a(\mathbf{x}_a, \mathbf{y}_a; \theta_a)}$$

To reiterate: this is a CRF. It's what you get when you model a conditional distribution as a product of factors

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_a e^{\Psi_a(\mathbf{x}, \mathbf{y}_a; \theta_a)}$$

Conditional Random Fields

Record scratch sound effect **Hold up. This seems super general. How do I actually implement this?**

> Uh...well...you still have all your work ahead of you: it's up to you to define the features and the factors in model. And since efficient inference isn't possible for every factor graph, people often restrict themselves to tree-structured graphs, for which there are fast inference algs.

So why does anyone care?

> It's a nice formalism that helps people apply graphical model techniques to prediction tasks.

That's it? How has this not been discovered before?

> Well it kinda has...

CRF Inference

Complicated topic that we'll talk about more this term.

If your factor graph is tree structures, you can use dynamic programming (a.k.a. message passing, a.k.a. belief propagation, a.k.a. forward-backward a.k.s.a backpropagation).

If your factor graph is something else, you can use approximation methods like loopy belief propagation.

CRF Training

This:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_a e^{\theta_a^T f_a(\mathbf{x}, \mathbf{y}_a)}$$

is convex in θ_a ! Wohoo!

But to compute the gradients we have to do inference. Boo.

Conditional Random Fields for Object Recognition

NIPS 2004

- Ariadna Quattoni
- Michael Collins
- Trevor Darrell

Object Recognition

Car Front

Car Side

Background

Given an image x, predict the label y

Mindset

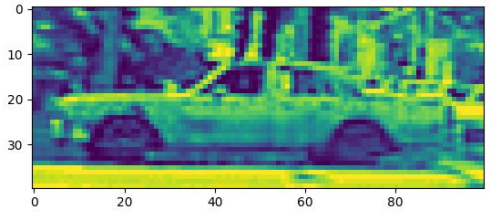
- Objects are rigid (e.g. cars, motorbikes, and faces).

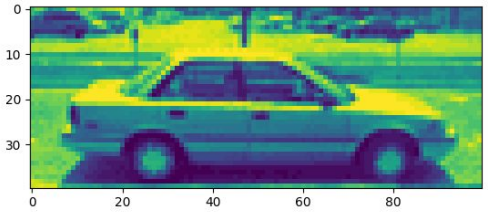
Rigid

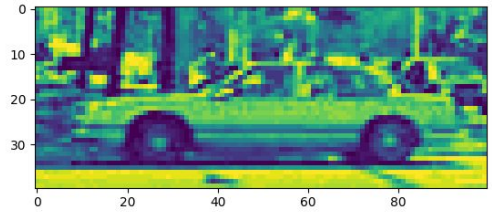
Non-Rigid

Mindset

- Objects are rigid (e.g. cars, motorbikes, and faces).
- View points are fixed (e.g. front & side)







Mindset

- Objects are rigid (e.g. cars, motorbikes, and faces).
- View points are fixed (e.g. front, side, back)
- We can use SIFT to find interesting regions of the image and extract fixed sized feature vectors.

x_i →  → $\{\phi(x_{i,1}), \ldots, \phi(x_{i,m})\}$

Lukas Mach at English Wikipedia

Question

How should we combine the feature vectors to make a class prediction?

Answer*

We'll use a CRF.

* Not the full answer

But...

The relationship between y_i and $x_{i,j}$ is not clearly defined.

Anwer

We'll use a hidden conditional random field (HCRF)

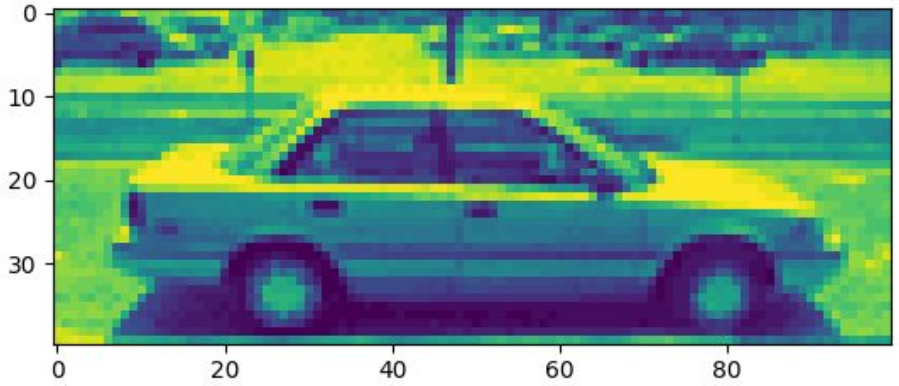
We'll model intermediate part labels $h_{i,j}$ as hidden variables in the model.

$$P(y, \mathbf{h}|\mathbf{x})$$

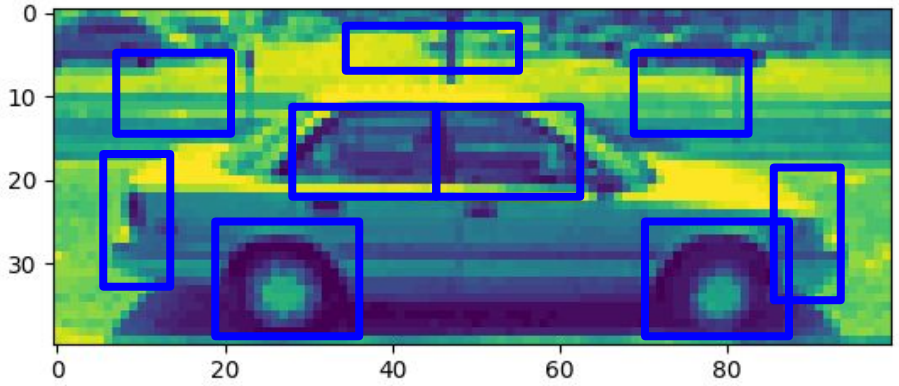
$$P(y|\mathbf{x}) = \sum_{\mathbf{h}} P(y, \mathbf{h}|\mathbf{x})$$

At a high level, what are we doing...

Input Image

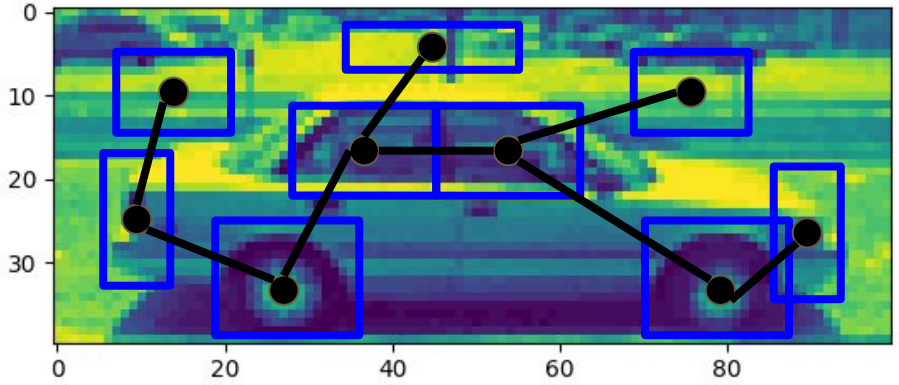


Detect Patches



Thanks David Lowe!

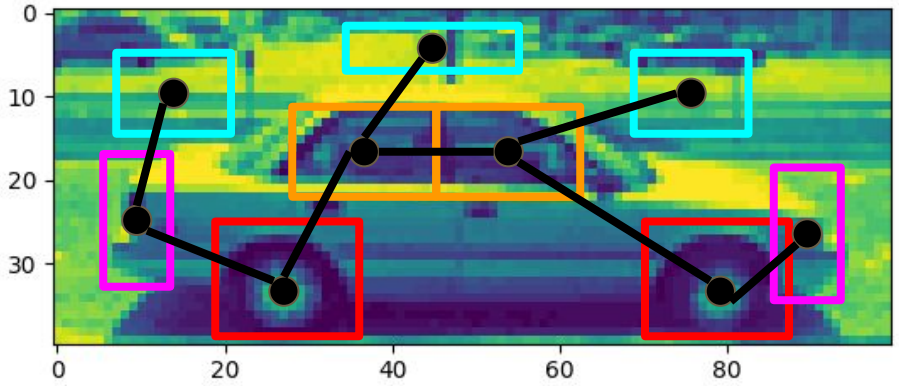
Determine Spatial Layout



Thanks Joseph Kruskal!

Run a minimum spanning **tree** algorithm over the parts, where the cost of an edge in the graph is taken to be the Euclidian distance between the corresponding image patches.

Label Patches with Parts



Wheel

Bumper

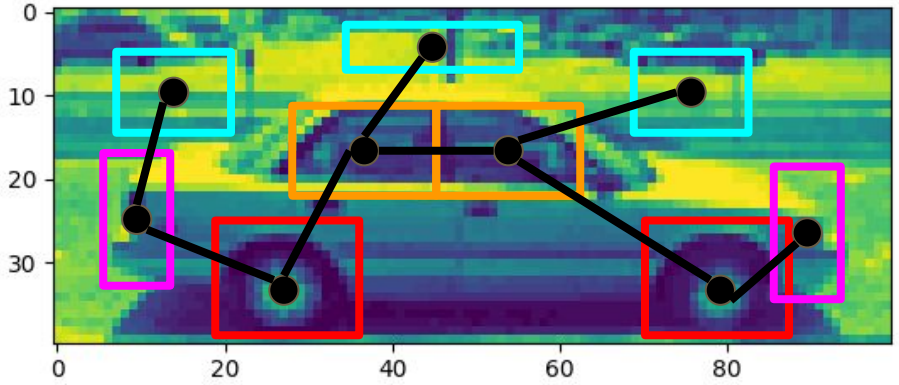
Window

Background

Label Patches with Parts: How?

- Assume we have a fixed number of parts, say 10.
- We would like to assign each image patch to a part.
- We want a measure of similarity between a part and a patch.
- So let's take a dot product between a "part vector" and the patch "feature vector."

Make a Prediction



Wheel

Bumper

Window

Background

Car!

Make a Prediction: How?

- Let's say we have two classes: car & background

- Lets focus on a single image patch. Conditioning on each class, we want to ask ourselves:
  - Does the most similar part for this patch make sense for this class?
  - Do the neighbors of this patch make sense for this class?
- The class that makes the most sense will be our prediction

Training Data

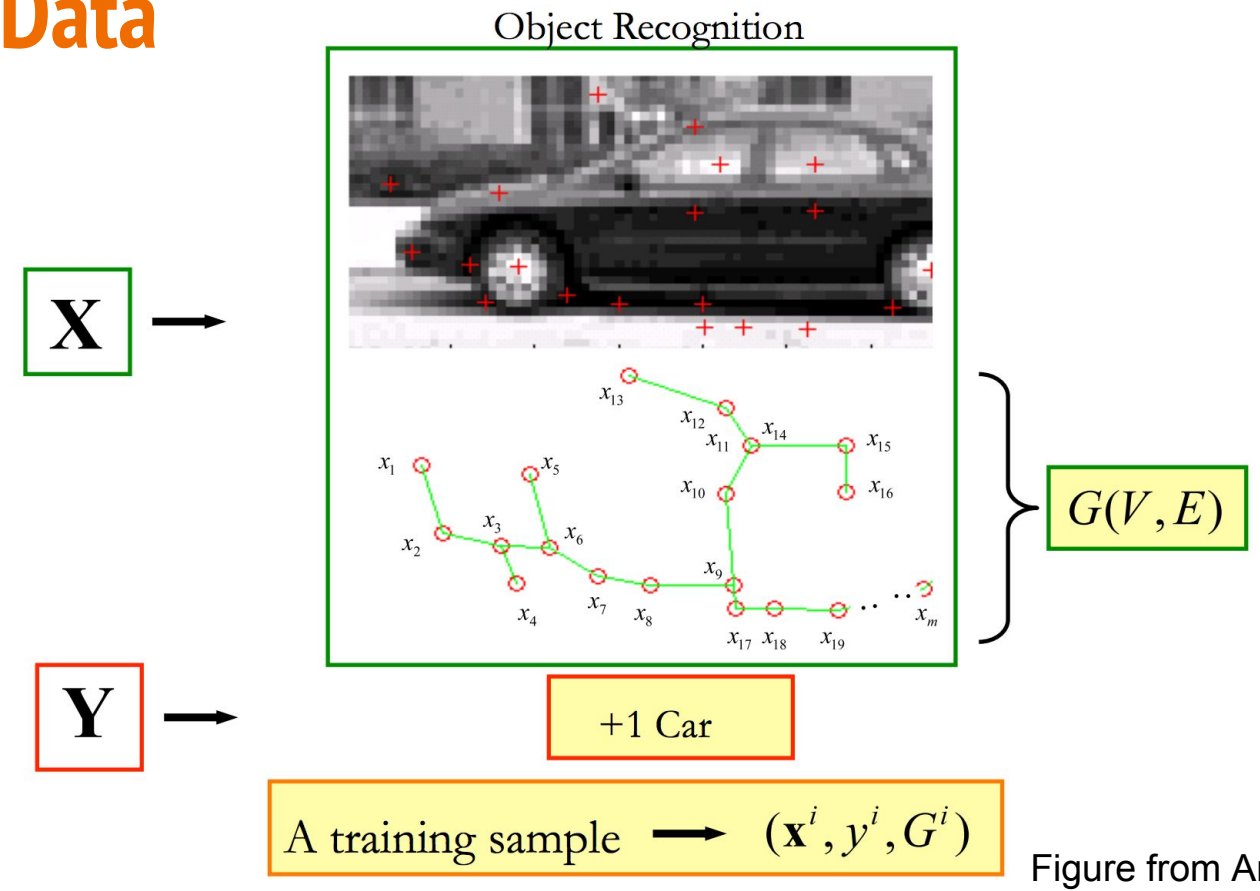


Figure from Ariadna Quattoni

Notation

$\mathcal{Y} = \{\text{background, car}\}$	**Class labels**
$\mathbf{x}_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,m}\}$	**Patches from image x**
$\mathbf{h} = \{h_1, h_2, \ldots, h_m\}$	**Part assignments**
$h_j \in \mathcal{H} \qquad \lvert\mathcal{H}\rvert = 5 \text{ or } 10$	**Finite set of parts in the model**
θ	**Model parameters**

The Conditional Probabilistic Model

$$P(y, \mathbf{h}|\mathbf{x}, \theta) = \frac{e^{\Psi(y,\mathbf{h},\mathbf{x};\theta)}}{\sum_{y',\mathbf{h}} e^{\Psi(y',\mathbf{h},\mathbf{x};\theta)}}$$

$$P(y|\mathbf{x}, \theta) = \sum_{\mathbf{h}} P(y, \mathbf{h}|\mathbf{x}, \theta) = \frac{\sum_{\mathbf{h}} e^{\Psi(y,\mathbf{h},\mathbf{x};\theta)}}{\sum_{y',\mathbf{h}} e^{\Psi(y',\mathbf{h},\mathbf{x};\theta)}}$$

Inference

θ^*	**trained model parameters**
$\mathbf{x}$	**new test image**
$\arg\max_{y \in \mathcal{Y}} P(y \mid \mathbf{x}, \theta^*)$	**predicted label**

How? With belief propagation (message passing)

Training Objective

$$L(\theta) = \sum_i P(y_i|\mathbf{x}_i, \theta) - \frac{1}{2\sigma^2} \left\|\theta^2\right\|$$

log-likelihood of the data

regularization; sigma is a hyper parameter

Define the potential function

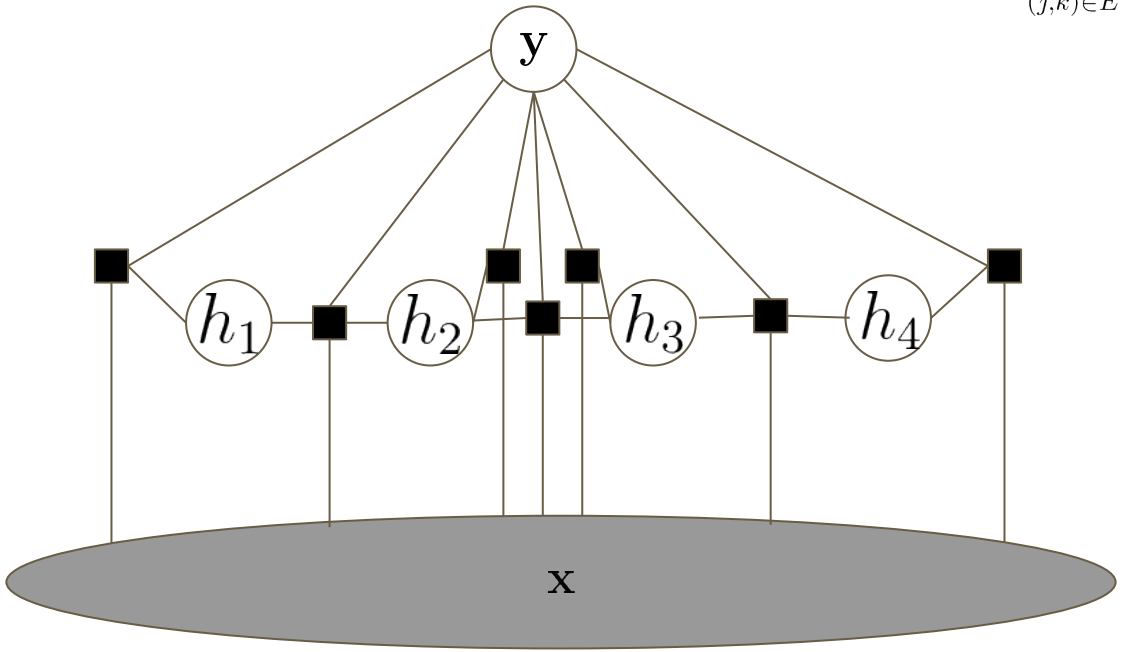
$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) =$$

$$\sum_{j=1}^{m} \sum_{l} f_l^1(j, y, h_j, \mathbf{x}) \theta_l^1 + \quad \text{Unary}$$

$$\sum_{(j,k) \in E} \sum_{l} f_l^2(j, k, y, h_j, h_k, \mathbf{x}) \theta_l^2 \quad \text{Pairwise}$$

Factor Graph

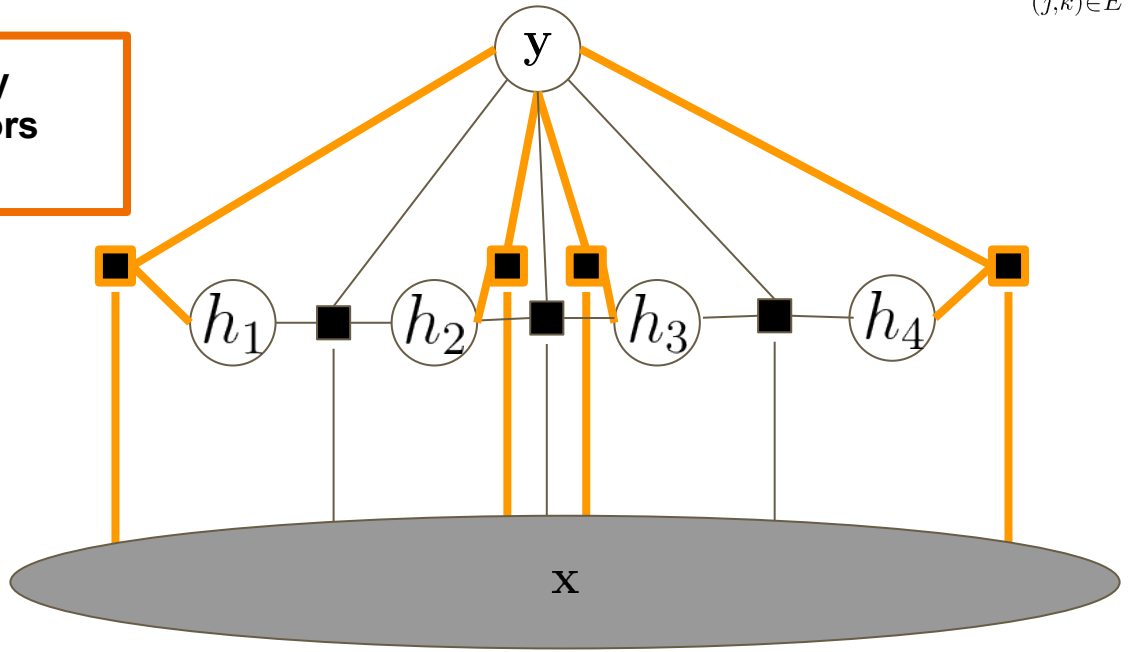
$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) =$$
$$\sum_{j=1}^{m} \sum_{l} f_l^1(j, y, h_j, \mathbf{x}) \theta_l^1 +$$
$$\sum_{(j,k) \in E} \sum_{l} f_l^2(j, k, y, h_j, h_k, \mathbf{x}) \theta_l^2$$



Factor Graph

$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) = \sum_{j=1}^{m} \sum_{l} f_l^1(j, y, h_j, \mathbf{x}) \theta_l^1 + \sum_{(j,k) \in E} \sum_{l} f_l^2(j, k, y, h_j, h_k, \mathbf{x}) \theta_l^2$$

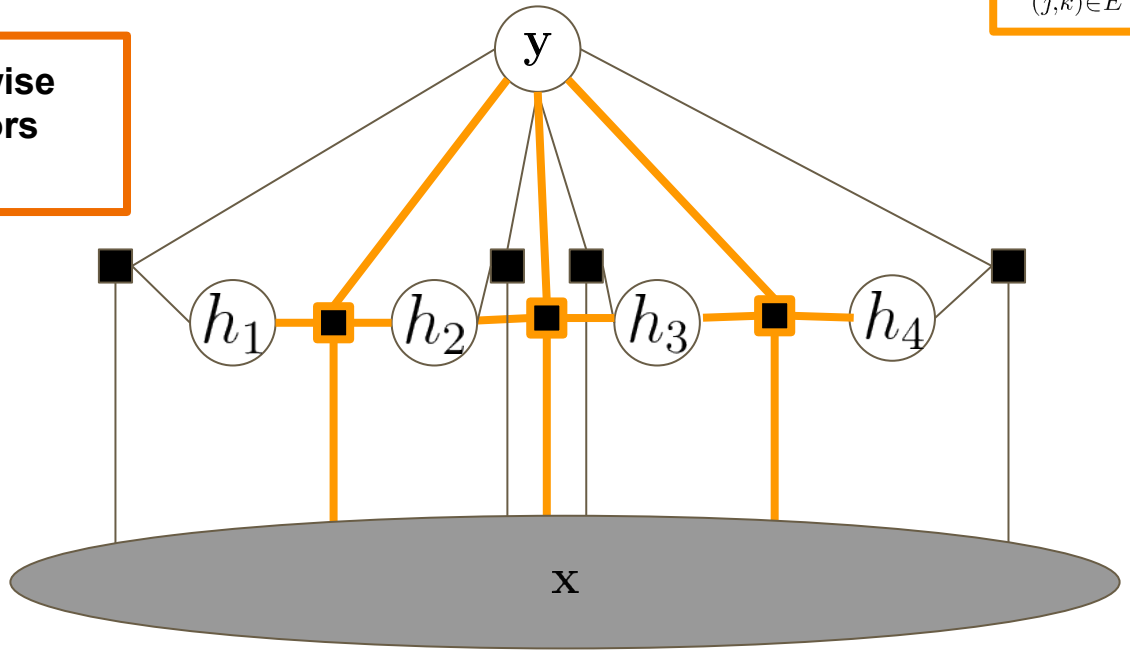
Unary Factors



Factor Graph

$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) =$$
$$\sum_{j=1}^{m} \sum_{l} f_l^1(j, y, h_j, \mathbf{x}) \theta_l^1 +$$
$$\sum_{(j,k) \in E} \sum_{l} f_l^2(j, k, y, h_j, h_k, \mathbf{x}) \theta_l^2$$

Pairwise Factors



Actual Potential Function

$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) =$$

$$\sum_j \phi(x_j) \cdot \theta(h_j) +$$

Compatibility between the patch and the part label

$$\sum_j \theta(y, h_j) +$$

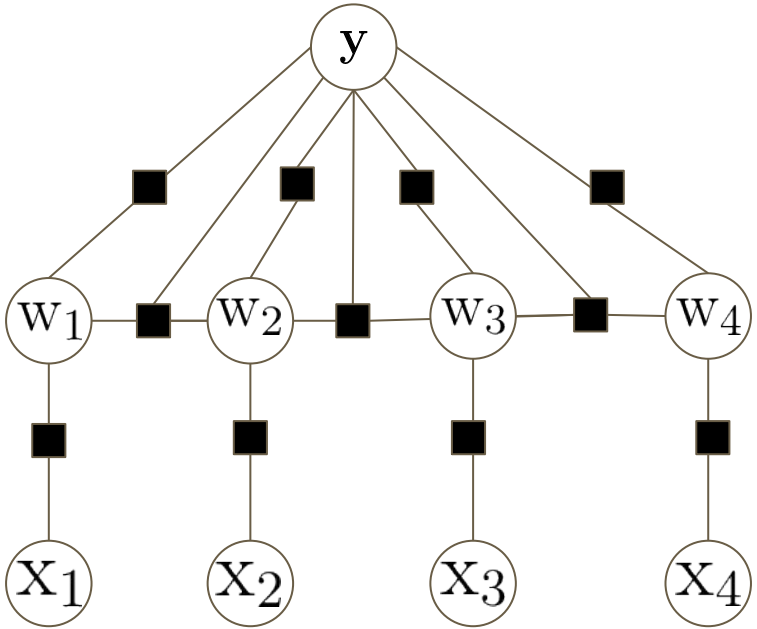
Compatibility between the part label and the class label

$$\sum_{(j,k) \in E} \theta(y, h_j, h_k)$$

Compatibility between an edge and the class label

Factor Graph

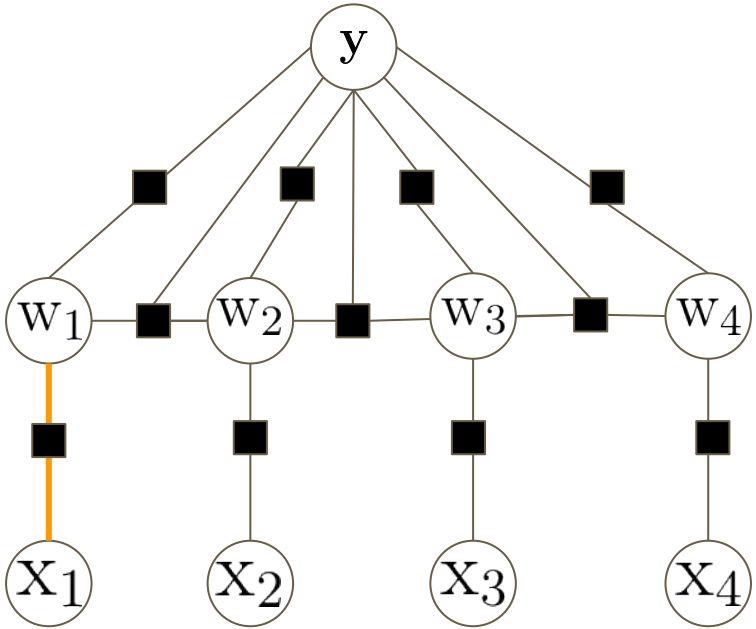
$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) =$$
$$\sum_j \phi(x_j) \cdot \theta(h_j) +$$
$$\sum_j \theta(y, h_j) +$$
$$\sum_{(j,k) \in E} \theta(y, h_j, h_k)$$



Factor Graph

$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) = \sum_j \phi(x_j) \cdot \theta(h_j) + \sum_j \theta(y, h_j) + \sum_{(j,k) \in E} \theta(y, h_j, h_k)$$

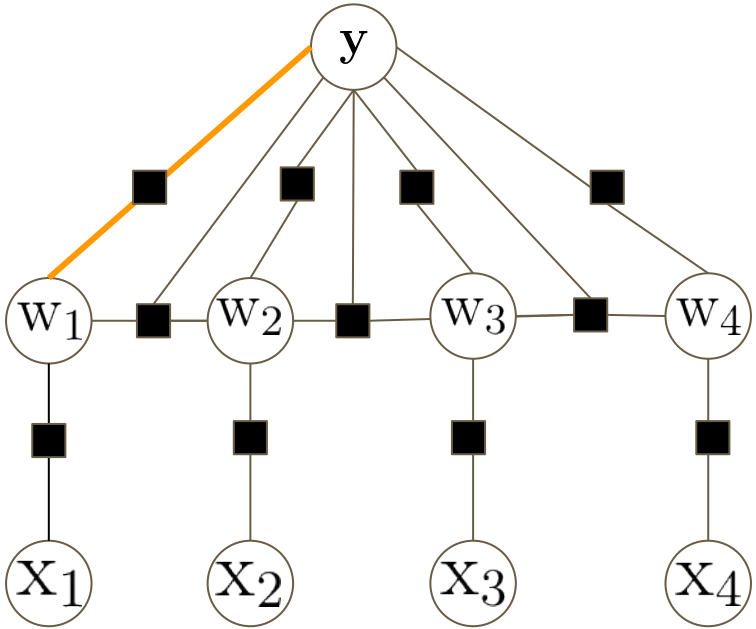
Compatibility between a patch and a part label



Factor Graph

$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) = \sum_j \phi(x_j) \cdot \theta(h_j) + \sum_j \theta(y, h_j) + \sum_{(j,k) \in E} \theta(y, h_j, h_k)$$

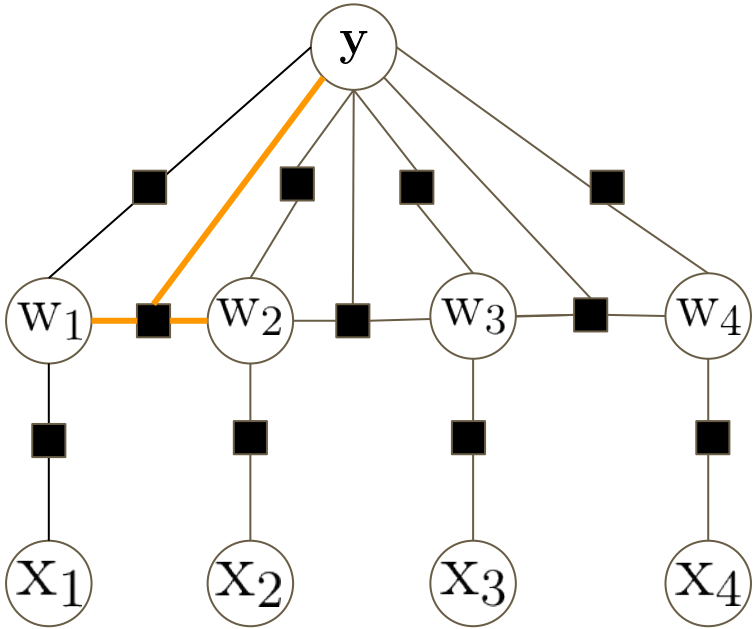
Compatibility between the part label and the class label



Factor Graph

$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) =$$
$$\sum_j \phi(x_j) \cdot \theta(h_j) +$$
$$\sum_j \theta(y, h_j) +$$
$$\sum_{(j,k) \in E} \theta(y, h_j, h_k)$$

Compatibility between an edge and the class label



Model Parameters

$$\theta(h_j) \in \mathbb{R}^d$$

Like an appearance vector for the part

Model Parameters

$$\theta(h_j) \in \mathbb{R}^d$$

$$\theta(y, k) \in \mathbb{R} \text{ for } y \in \mathcal{Y}, k \in \mathcal{H}$$

Does class y have part k?

Model Parameters

$$\theta(h_j) \in \mathbb{R}^d$$

$$\theta(y, k) \in \mathbb{R} \text{ for } y \in \mathcal{Y}, k \in \mathcal{H}$$

$$\theta(y, k, l) \in \mathbb{R} \text{ for } y \in \mathcal{Y}, \text{ and } k, l \in \mathcal{H}$$

Does class y have an edge between parts k and l?

Training & Inference

$$E \text{ is a tree}$$

$$\Psi \text{ is linear in the parameters } \theta$$

Exact methods exist for inference and parameter estimation. Use belief propagation.

Training & Inference

The gradient of $L(\theta)$ with respect to θ_l^1 and θ_l^2 can be calculated efficiently.

You can use a gradient method to optimize $L(\theta)$. See the paper for details.

Results

Data set	5 parts	10 parts
Car Side	94%	99%
Car Rear	91%	91.7%

Results

Data set	HCRF	Generative Model
Car Side	99%	-
Car Rear	94.6%	90.3%
Face	99%	96.4%
Plane	96%	90.2%
Motorbike	95%	92.5%

Learned Parts

Figure from Ariadna Quattoni

Questions?