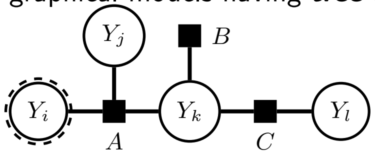
Probabilistic Graphical Models in Computer Vision (IN2329)

Csaba Domokos

Summer Semester 2017

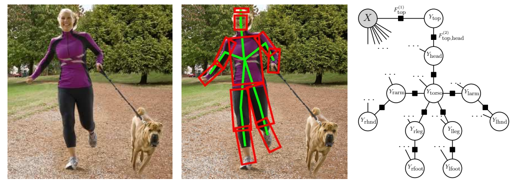
Agenda for today's lecture *

In the last lecture we learnt about **exact inference** methods on graphical models having **tree structure**.



Today we are going to learn about

- *Human-pose estimation*



Source: Nowozin and Lampert. Structured Learning and Prediction. 2011.

- *Mean-field approximation*: probabilistic inference via optimization (a.k.a. variational inference)

The model

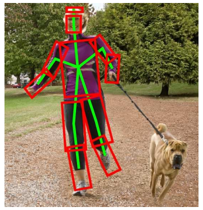
The goal is to recognize an articulated object with joints connecting different parts, here it is a *human body*.

An object is composed of a number of **rigid parts**. Each part is modeled as a rectangle parameterized by (x, y, s, θ), where

- (x, y) means the **center of the rectangle**,
- $s \in [0, 1]$ is a **scaling factor**, and
- **the orientation** is given by θ.

In overall, we have a four-dimensional pose space.

We denote the **locations** of two (connected) parts by $l_i = (x_i, y_i, s_i, \theta_i)$ and $l_j = (x_j, y_j, s_j, \theta_j)$, respectively.

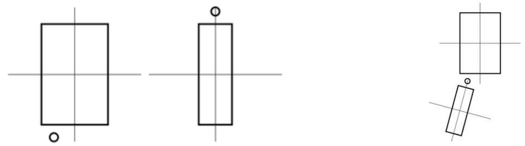


Source: Nowozin and Lampert. Structured Learning and Prediction. 2011.

The model (cont.)

An object (e.g., human body) is given by a configuration $\mathbf{l} = (l_1, \ldots, l_n)$, where l_i specifies the location of **part** v_i. The connections encode generic relationships such as "close to", "to the left of", or more precise geometrical constraints such as ideal joint angles.

- The **location of a joint** between v_i and v_j is specified by two points (x_{ij}, y_{ij}) and (x_{ji}, y_{ji}).
- The **relative orientation** is given by θ_{ij}, which is the difference between the orientation of the two parts.



Source: Felzenszwalb and Huttenlocher. Pictorial Structures for Object Recognition. IJCV, 2005.

In principle, all parts depend on each other, however, tree structured model can be considered for an articulated pose.

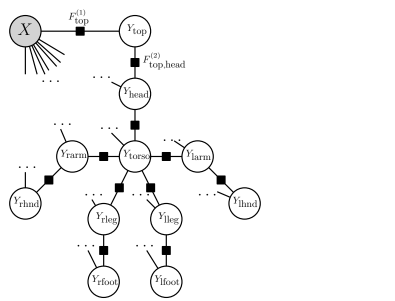
Graphical representation

The structure is encoded by a graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ corresponds to n parts, and there is an edge $(v_i, v_j) \in \mathcal{E}$ for each pair of connected parts v_i and v_j.

We want to minimize the following *energy function*:

$$\mathbf{l}^* \in \underset{\mathbf{l}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i, v_j) \in \mathcal{E}} d_{ij}(l_i, l_j) \right),$$

where $m_i(l_i)$ measures the degree of mismatch when the part v_i is placed at location l_i and $d_{ij}(l_i, l_j)$ measures the degree of deformation of the model when part v_i is placed at location l_i and part v_j is placed at location l_j.



Source: Nowozin and Lampert. Structured Learning and Prediction. 2011.

Note that MAP inference can be efficiently done by making use of *Max-sum algorithm*.

Image filters *

The **image filtering** is a technique for modifying or enhancing an image (e.g., smoothing, edge detection, sharpening). For example, the smoothing of an input signal means of removing (or filtering out) high-frequency components.

Here we consider **linear filtering** in which the value of an output pixel is a linear combination of the values of the pixels in the input pixel's neighborhood. In a spatially discrete setting, a linear filter is a weighted sum:

$$g(x_0, y_0) = [f * w](x_0, y_0) = \sum_{m,n} w(m,n) \cdot f(x_0 - m, y_0 - n)$$

which is also called **discrete convolution** of f and w. In practice this summation extends over a certain neighborhood. The matrix of weights $w(m,n)$ is called a **mask**.

(For more details please refer to the course of **Computer Vision I: Variational Methods**.)

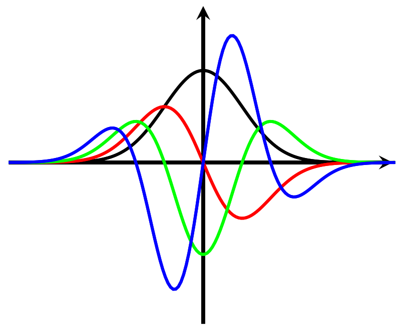
Derivatives of a Gaussian *

Let us consider the (one-dimensional) Gaussian density function:

$$f_X(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(x-\mu)^2}{2\sigma^2}) \, .$$

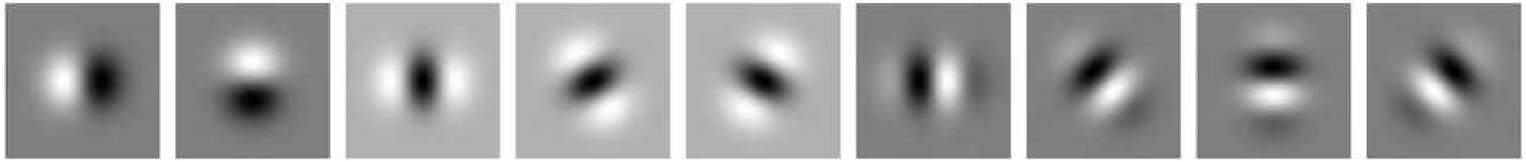
Assume that $\mu = 0$ and let us calculate the derivatives of $f_X(x; 0, \sigma)$ of different orders

$$\frac{\partial f_X(x;0,\sigma)}{\partial x} = \frac{-x}{\sigma^3\sqrt{2\pi}} \exp\left(\frac{-x^2}{2\sigma^2}\right)$$
$$\frac{\partial f_X(x;0,\sigma)}{\partial^2 x} = \frac{x^2-\sigma^2}{\sigma^5\sqrt{2\pi}} \exp\left(\frac{-x^2}{2\sigma^2}\right)$$
$$\frac{\partial f_X(x;0,\sigma)}{\partial^3 x} = \frac{x(3\sigma^2-x^2)}{\sigma^7\sqrt{2\pi}} \exp\left(\frac{-x^2}{2\sigma^2}\right)$$



Unary energies *

An image patch centered at some position is represented by a vector that collects all the responses of a set of Gaussian derivative filters of different orders, orientations and scales at that point. This vector is normalized and called the **iconic index** at that position.



The *unary energies* are defined as

$$m_i(l_i) = -\ln \mathcal{N}(\alpha(l_i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \ ,$$

where $\alpha(l_i)$ is the *iconic index* at location l_i in the image.

The parameters for each part (i.e. the mean vector $\boldsymbol{\mu}_i$ and the covariance matrix $\boldsymbol{\Sigma}_i$ for all $i = 1, \ldots, n$) can be obtained by maximum likelihood estimation for a given set of training samples.

Pairwise energies *

The pairwise energies have a special form as follows.

$$d_{ij}(l_i, l_j) = -\ln \mathcal{N}(T_{ji}(l_j) - T_{ij}(l_i), \mathbf{0}, \mathbf{D}_{ij}) \ ,$$

where T_{ij}, T_{ji} and $\mathbf{D}_{ij}$ are connection parameters

$$\begin{aligned} T_{ij}(l_i) =& (x'_i, y'_i, s_i, \cos(\theta_i + \theta_{ij}), \sin(\theta_i + \theta_{ij})), \\ T_{ji}(l_j) =& (x'_j, y'_j, s_j, \cos(\theta_j), \sin(\theta_j)), \\ \mathbf{D}_{ij} =& \text{diag}(\sigma_x^2, \sigma_y^2, \sigma_s^2, 1/k, 1/k) \ . \end{aligned}$$

$T_{ij}(l_i)$ and $T_{ji}(l_j)$ are one-to-one mappings encoding the set of possible transformed locations. θ_{ij} stands for the ideal relative angle between the ith and jth parts.

Pairwise energies (cont.) *

Let $\mathbf{R}_\theta$ be the matrix that performs a rotation of θ radians about the origin. Then,

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + s_i \mathbf{R}_{\theta_i} \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} x_j' \\ y_j' \end{bmatrix} = \begin{bmatrix} x_j \\ y_j \end{bmatrix} + s_j \mathbf{R}_{\theta_j} \begin{bmatrix} x_{ji} \\ y_{ji} \end{bmatrix} ,$$

where (x_i, y_i), (x_j, y_j) and (x_{ij}, y_{ij}), (x_{ji}, y_{ji}) are the positions of the joints in image and local coordinates, respectively.
We assume the following joint distributions:

- $\mathcal{N}(x_i - x_j, 0, \sigma_x^2)$ and $\mathcal{N}(y_i - y_j, 0, \sigma_y^2)$ which measures the horizontal and vertical distances, respectively, between the observed joint positions.
- $\mathcal{N}(s_i - s_j, 0, \sigma_s^2)$ measures the difference in foreshortening between the two parts.
- $\mathcal{M}(\theta_i - \theta_j, \theta_{ij}, k) \propto \exp(k \cos(\theta_i - \theta_j - \theta_{ij}))$ measures the difference between the relative angle of the two parts and the ideal relative angle.

These parameters can be also obtained by maximum likelihood estimation.

Inference

MAP inference provides a single (best) prediction of the overall pose. The factor–to–varaible messages can be written as

$$\begin{aligned} r_{F \to v_i}(l_i) &= \max_{\substack{(l_i', l_j') \in \mathcal{Y}_F, \\ l_i' = l_i}} \Big(\exp(-m_i(l_i') - d_{ij}(l_i', l_j')) + \sum_{k \in N(F) \setminus \{i\}} q_{v_k \to F}(l_k') \Big) \\ &= \max_{l_j \in \mathcal{Y}_j} \Big(\underbrace{\exp(-m_i(l_i))}_{\text{const.}} \exp(-d_{ij}(l_i, l_j)) + \underbrace{q_{v_j \to F}(l_j)}_{h(l_j)} \Big) . \end{aligned}$$

$\mathcal{Y}$ could be quite large ($\approx 1.5M$ possible states), hence $\mathcal{Y}_i \times \mathcal{Y}_j$ is too big. However a special form of pairwise energies is used, so that a message can be calculated in $\mathcal{O}(|\mathcal{Y}_i|)$ time.

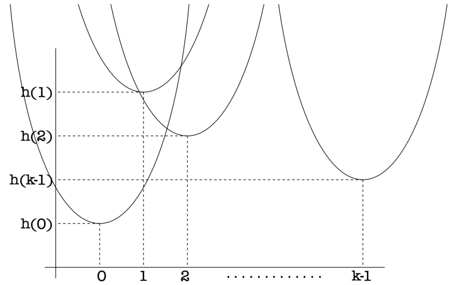
Efficient inference via min convolution

Assume that we have to compute a message for $(i, j) \in \mathcal{E}$, that is for a given $c \in \mathbb{R}$

$$r(l_i) = \min_{l_j}(c \cdot d_{ij}(l_i, l_j) + h(l_j)) = \min_{l_j}(c \cdot (l_i - l_j)^2 + h(l_j)) \; .$$

Here we only discuss the one-dimensional case, however, the extension for the multi-dimensional case is straightforward.

The basic idea is to calculate the **lower envelope**, which can be done in linear time w.r.t. the possible values of $l_i \in \mathcal{Y}_i$.



Source: Felzenszwalb and Huttenlocher. Pictorial Structures for Object Recognition. IJCV, 2005.

We consider parabolas rooted at $(l_i, h(l_i))$ (i.e. $y = c \cdot (x - l_i)^2 + h(l_i)$).

Calculating the lower envelope

Note that any two parabolas defining the lower envelope intersect at **exactly one point**. The x-coordinate of the intersection of the parabolas rooted at $(p, h(p))$ and $(q, h(q))$ can be calculated as

$$s = \frac{h(p) - h(q) + cp^2 - cq^2}{2c(p - q)}$$

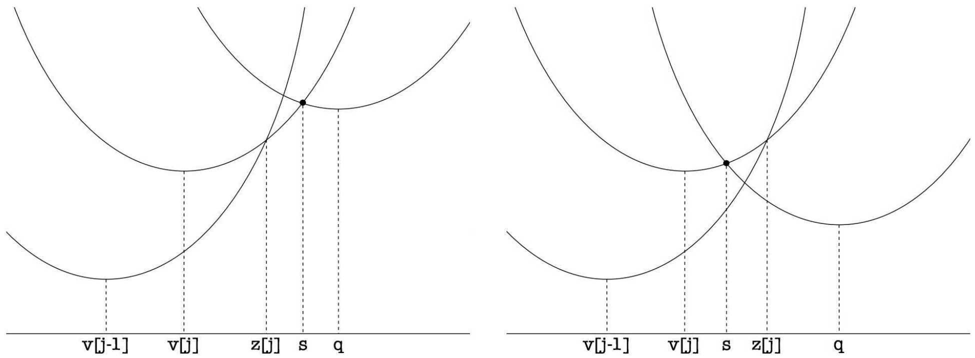
Note that when $q < p$ then the parabola coming from q is below the one coming from p to the left of the intersection point s, and above it to the right of s.

The algorithm manages two arrays:

- The horizontal grid location of the ith parabola in the lower envelope is stored in $v[i]$
- The range in which the ith parabola of the lower envelope is below the others is given by $z[i]$ and $z[i + 1]$

Updating the lower envelope

There are two possible cases when adding a parabola from q to the lower envelope constructed so far:



$s > z[j]$ $\qquad$ $s \leqslant z[j]$

Source: Felzenszwalb and Huttenlocher. Pictorial Structures for Object Recognition. IJCV, 2005.

Pseudo-code of the min convolution *

1: $j \leftarrow 0$ ▻ Index of rightmost parabola in lower envelope
2: $v[0] \leftarrow 0$ ▻ Locations of parabolas in lower envelope
3: $z[0] \leftarrow -\infty$ ▻ Locations of boundaries between parabolas
4: $z[1] \leftarrow \infty$
5: **for** $q = 1 \rightarrow n-1$ **do** ▻ Compute lower envelope
6:     $s \leftarrow (h(q) - h(v[j]) + cq^2 - cv[j]^2)/(2c(q - v[j]))$
7:     **if** $s \leqslant z[j]$ **then**
8:         $j \leftarrow j - 1$ and goto 6
9:     **else**
10:         $j \leftarrow j + 1$
11:         $v[j] \leftarrow q$; $z[j] \leftarrow s$; $z[j+1] \leftarrow \infty$
12:     **end if**
13: **end for**
14: $j \leftarrow 0$
15: **for** $q = 0 \rightarrow n-1$ **do** ▻ Fill in values of min convolution
16:     **while** $z[j+1] < q$ **do**
17:         $j \leftarrow j + 1$
18:     **end while**
19:     $r(q) \leftarrow c(q - v[j])^2 + h(v[j])$
20: **end for**

KL divergence

Assume two discrete probability distributions p and q. One way to measure the *difference* between p and q is to calculate the **Kullback–Leibler (KL) divergence** (a.k.a. *relative entropy*) defined as

$$D_{\mathrm{KL}}(p\|q) = \sum_i p(i) \log \frac{p(i)}{q(i)} = \sum_i p(i) \log p(i) - \sum_i p(i) \log q(i)$$
$$= \mathbb{E}_p[\log p(i)] - \mathbb{E}_p[\log q(i)] \ .$$

It is defined iff for all i, $q(i) = 0$ implies $p(i) = 0$. If $p(i) = 0$, then the ith term is interpreted as 0. The KL divergence is always non-negative, moreover $D_{\mathrm{KL}}(p\|q) = 0$ iff $p = q$ *almost everywhere*. Nevertheless, it is neither symmetric nor does it satisfy the triangle inequality.

Interpretation (Information Theory): it is the amount of information lost when q is used to approximate p. It measures the expected number of extra bits required to code samples from p using a code optimized for q rather than the code optimized for p.

Motivation

For general (discrete) factor graph models, performing *probabilistic inference* is hard. Assume we are given an **intractable** distribution $p(\mathbf{y} \mid \mathbf{x})$. We consider an **approximate distribution** $q(\mathbf{y})$, which is *tractable*, for $p(\mathbf{y} \mid \mathbf{x})$.

One way of finding the best approximating distribution is to pose it as an **optimization problem** over probability distributions: given a distribution $p(\mathbf{y} \mid \mathbf{x})$ and a family Q of *tractable distributions* $q \in Q$ on $\mathcal{Y}$, we want to solve

$$q^* \in \operatorname*{argmin}_{q \in Q} D_{\mathrm{KL}}(q(\mathbf{y}) \| p(\mathbf{y} \mid \mathbf{x})) = \operatorname*{argmin}_{q \in Q} \sum_{\mathbf{y} \in \mathcal{Y}} q(\mathbf{y}) \log \frac{q(\mathbf{y})}{p(\mathbf{y} \mid \mathbf{x})}$$
$$= \operatorname*{argmin}_{q \in Q} \Big\{ \underbrace{\sum_{\mathbf{y} \in \mathcal{Y}} q(\mathbf{y}) \log q(\mathbf{y})}_{-H(q)} - \sum_{\mathbf{y} \in \mathcal{Y}} q(y) \log p(\mathbf{y} \mid \mathbf{x}) \Big\} \ .$$

The term $-\sum_{\mathbf{y} \in \mathcal{Y}} q(\mathbf{y}) \log q(\mathbf{y}) \triangleq H(q)$ is called the **entropy** of the distribution q.

Mean field methods

$$
\begin{aligned}
D_{\mathrm{KL}}(q(\mathbf{y}) \| p(\mathbf{y} \mid \mathbf{x})) &= - H(q) - \sum_{\mathbf{y} \in \mathcal{Y}} q(\mathbf{y}) \log p(\mathbf{y} \mid \mathbf{x}) \\
&= -H(q) - \sum_{\mathbf{y} \in \mathcal{Y}} q(\mathbf{y}) \log \frac{1}{Z(\mathbf{x})} \prod_{F \in \mathcal{F}} \exp(-E_F(\mathbf{y}_F; \mathbf{x}_F)) \\
&= -H(q) + \sum_{\mathbf{y} \in \mathcal{Y}} q(\mathbf{y}) \sum_{F \in \mathcal{F}} E_F(\mathbf{y}_F; \mathbf{x}_F) + \log Z(\mathbf{x}) \\
&= -H(q) + \sum_{F \in \mathcal{F}} \sum_{\mathbf{y}_F \in \mathcal{Y}_F} \underbrace{\sum_{\substack{\mathbf{y}' \in \mathcal{Y}, \\ \mathbf{y}'_F = \mathbf{y}_F}} q(\mathbf{y}')}_{\mu_{F,\mathbf{y}_F}(q)} E_F(\mathbf{y}_F; \mathbf{x}_F) + \log Z(\mathbf{x}) \\
&= -H(q) + \sum_{F \in \mathcal{F}} \sum_{\mathbf{y}_F \in \mathcal{Y}_F} \mu_{F,\mathbf{y}_F}(q) E_F(\mathbf{y}_F; \mathbf{x}_F) + \log Z(\mathbf{x}) \ ,
\end{aligned}
$$

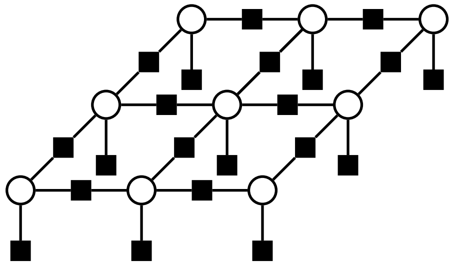
where $\mu_{F,\mathbf{y}_F}(q) = \sum_{\mathbf{y}' \in \mathcal{Y}, \mathbf{y}'_F = \mathbf{y}_F} q(\mathbf{y}')$ are the marginals of q.

Naïve mean field

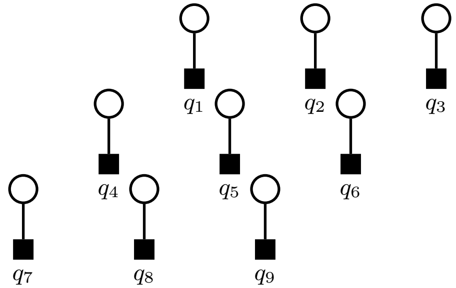
Take a set q as the set of all distributions in the form:

$$q(\mathbf{y}) = \prod_{i \in \mathcal{V}} q_i(y_i) \ .$$

For example, in case of the following factor graph:



Original factor graph



Mean field approximation

Naïve mean field *

Set q consists of all distributions in the form:

$$q(\mathbf{y}) = \prod_{i \in \mathcal{V}} q_i(y_i) \ .$$

Marginals $\mu_{F,\mathbf{y}_F}$ take the form

$$\mu_{F,\mathbf{y}_F}(q) = \sum_{\substack{\mathbf{y}' \in \mathcal{Y}, \\ \mathbf{y}'_F = \mathbf{y}_F}} q(\mathbf{y}) = q_{N(F)}(\mathbf{y}_F) = \prod_{i \in N(F)} q_i(y_i) \ .$$

Entropy $H(q)$ decomposes as

$$H(q) = \sum_{i \in \mathcal{V}} H_i(q_i) = -\sum_{i \in \mathcal{V}} \sum_{y_i \in \mathcal{Y}_i} q_i(y_i) \log q_i(y_i) \ .$$

Proof. Exercise. □

Naïve mean field

Putting all together,

$$\begin{aligned}
q^* &\in \operatorname*{argmin}_{q \in Q} D_{\mathrm{KL}}(q(\mathbf{y}) \| p(\mathbf{y} \mid \mathbf{x})) \\
&= \operatorname*{argmin}_{q \in Q} \left\{ -H(q) + \sum_{F \in \mathcal{F}} \sum_{\mathbf{y}_F \in \mathcal{Y}_F} \mu_{F,\mathbf{y}_F}(q) E_F(\mathbf{y}_F; \mathbf{x}_F) + \log Z(\mathbf{x}) \right\} \\
&= \operatorname*{argmax}_{q \in Q} \left\{ H(q) - \sum_{F \in \mathcal{F}} \sum_{\mathbf{y}_F \in \mathcal{Y}_F} \mu_{F,\mathbf{y}_F}(q) E_F(\mathbf{y}_F; \mathbf{x}_F) \right\} \\
&= \operatorname*{argmax}_{q \in Q} \left\{ -\sum_{i \in \mathcal{V}} \sum_{y_i \in \mathcal{Y}_i} q_i(y_i) \log q_i(y_i) - \sum_{F \in \mathcal{F}} \sum_{\mathbf{y}_F \in \mathcal{Y}_F} \Big(\prod_{i \in N(F)} q_i(y_i) \Big) E_F(\mathbf{y}_F; \mathbf{x}_F) \right\} .
\end{aligned}$$

Optimizing over Q means to optimize over all q_i such that $q_i(y_i) \geqslant 0$ and $\sum_{y_i \in \mathcal{Y}_i} q_i(y_i) = 1$ for all $i \in \mathcal{V}$.

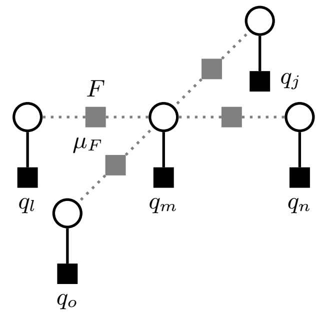
Optimization

$$\underset{q\in Q}{\operatorname{argmax}}\left\{\underbrace{-\sum_{i\in\mathcal{V}}\sum_{y_i\in\mathcal{Y}_i} q_i(y_i)\log q_i(y_i)}_{\text{entropy}} - \sum_{F\in\mathcal{F}}\sum_{\mathbf{y}_F\in\mathcal{Y}_F}\Big(\prod_{i\in N(F)} q_i(y_i)\Big)E_F(\mathbf{y}_F;\mathbf{x}_F)\right\}.$$

The *entropy* term is concave and the second term is non-concave due to products of variables occurring in the expression. Therefore solving this non-concave maximization problem globally is hard in general.

Remedy: **block coordinate ascent**

We hold all variables fixed except for a single block q_m, then we obtain a tractable concave maximization problem
→ closed-form update for each q_m.



Lagrange multipliers *

To obtain closed form solution, we define the *Lagrangian function*:

$$L(q_i, \lambda) = \Bigg\{ -\sum_{i\in\mathcal{V}} \sum_{y_i\in\mathcal{Y}_i} q_i(y_i) \log q_i(y_i)$$

$$- \sum_{F\in\mathcal{F}} \sum_{\mathbf{y}_F\in\mathcal{Y}_F} \Big(\prod_{i\in N(F)} q_i(y_i) \Big) E_F(\mathbf{y}_F; \mathbf{x}_F) + \lambda \Big(\sum_{y_i\in\mathcal{Y}_i} q_i(y_i) - 1 \Big) \Bigg\}.$$

Setting the derivatives of L w.r.t. q_i to 0, we obtain

$$\frac{\partial L}{\partial q_i(y_i)} = 0 = -\left(\log q_i(y_i) + 1\right) - \sum_{F\in M(i)} \sum_{\substack{\mathbf{y}'_F\in\mathcal{Y}_F,\\ y'_i=y_i}} \Big(\prod_{j\in N(F)\setminus\{i\}} q_j(y'_j) \Big) E_F(\mathbf{y}'_F; \mathbf{x}_F) + \lambda$$

$$q_i^*(y_i) = \exp\left(-1 - \sum_{F\in M(i)} \sum_{\substack{\mathbf{y}'_F\in\mathcal{Y}_F,\\ y'_i=y_i}} \Big(\prod_{j\in N(F)\setminus\{i\}} q_j(y'_j) \Big) E_F(\mathbf{y}'_F; \mathbf{x}_F) + \lambda \right).$$

Lagrange multipliers *

λ can be calculated as follows.

$$\sum_{y_i \in \mathcal{Y}_i} q_i(y_i) = \sum_{y_i \in \mathcal{Y}_i} \exp\left(-1 - \sum_{F \in M(i)} \sum_{\substack{\mathbf{y}'_F \in \mathcal{Y}_F, \\ y'_i = y_i}} \Big(\prod_{j \in N(F) \setminus \{i\}} q_j(y'_j) \Big) E_F(\mathbf{y}'_F; \mathbf{x}_F) + \lambda \right)$$

$$\exp(1-\lambda) = \underbrace{\sum_{y_i \in \mathcal{Y}_i} \exp\left(- \sum_{F \in M(i)} \sum_{\substack{\mathbf{y}'_F \in \mathcal{Y}_F, \\ y'_i = y_i}} \Big(\prod_{j \in N(F) \setminus \{i\}} q_j(y'_j) \Big) E_F(\mathbf{y}'_F; \mathbf{x}_F) \right)}_{Z_i(\mathbf{x}_F)}$$

$$\lambda - 1 = -\log Z_i(\mathbf{x}_F) \ ,$$

where $Z_i(\mathbf{x}_F)$ is a normalizing constant for q_i.

Update equation

By substituting, we obtain the obtain the update equation for the *Naïve mean field method*

$$q_i^*(y_i) = \exp\left(- \sum_{F \in M(i)} \sum_{\substack{\mathbf{y}'_F \in \mathcal{Y}_F, \\ y'_i = y_i}} \Big(\prod_{j \in N(F) \setminus \{i\}} q_j(y'_j) \Big) E_F(\mathbf{y}'_F; \mathbf{x}_F) - \log Z_i(\mathbf{x}_F) \right)$$

$$= \frac{1}{Z_i(\mathbf{x}_F)} \exp\left(- \sum_{F \in M(i)} \sum_{\substack{\mathbf{y}'_F \in \mathcal{Y}_F, \\ y'_i = y_i}} \Big(\prod_{j \in N(F) \setminus \{i\}} q_j(y'_j) \Big) E_F(\mathbf{y}'_F; \mathbf{x}_F) \right) .$$

Semantic segmentation

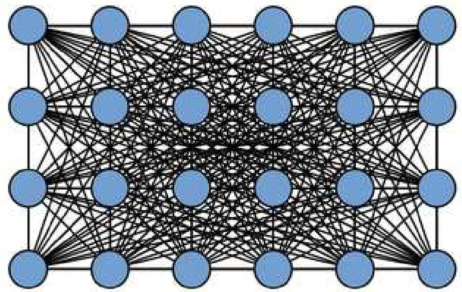
Krähenbühl and Koltun proposed an efficient approximate inference in fully connected CRF model by applying *Naïve mean field* approach.

Semantic segmentation: assign a label from the set of labels $\mathcal{L} = \{l_1, l_2, \ldots, l_k\}$ for each pixel on the image regarding their semantic meaning.

For each pixel on the image a random variable is assigned taking a value from $\mathcal{L}$. A fully connected pairwise CRF model $G = (\mathcal{V}, \mathcal{E})$ is considered, where the corresponding energy function is given by

$$E(\mathbf{y}) = \sum_{i \in \mathcal{V}} E_i(y_i) + \sum_{(i,j) \in \mathcal{E}} E_{ij}(y_i, y_j) ,$$

where $\mathcal{E} = \{(i,j) \in \mathcal{V} \times \mathcal{V} \mid i < j\}$.



Energy functions

- **Unary energies** $E_i(y_i)$ are computed independently for each pixel as $E_i(y_i) = -\log P_i(y_i)$ measures the degree of disagreement between labelling y_i and the image at pixel i.
- **Pairwise energies** (**contrast–sensitive Potts–model**), measuring the extent to which the labelling y is not piecewise smooth, have the form (p_i and I_i denote the pixel coordinates and intensity, respectively).

$$
\begin{aligned}
E_{ij}(y_i, y_j) =& [\![y_i \neq y_j]\!] \sum_m w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) \\
=& [\![y_i \neq y_j]\!] \sum_m w^{(m)} \exp\left(-\frac{1}{2}(\mathbf{f}_i - \mathbf{f}_j)^T \mathbf{\Sigma}^{(m)} (\mathbf{f}_i - \mathbf{f}_j)\right) \\
=& [\![y_i \neq y_j]\!] \Big\{ w^{(1)} \exp\Big(- \frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2} \Big) \\
&+ w^{(2)} \exp\Big(- \frac{|p_i - p_j|^2}{2\theta_\gamma^2} \Big) \Big\} .
\end{aligned}
$$

The parameters $\theta_\alpha, \theta_\beta$ and θ_γ are estimated on a set of training images.

Inference

The inference is based on *Naïve mean field approximation*, where the update equation is given by

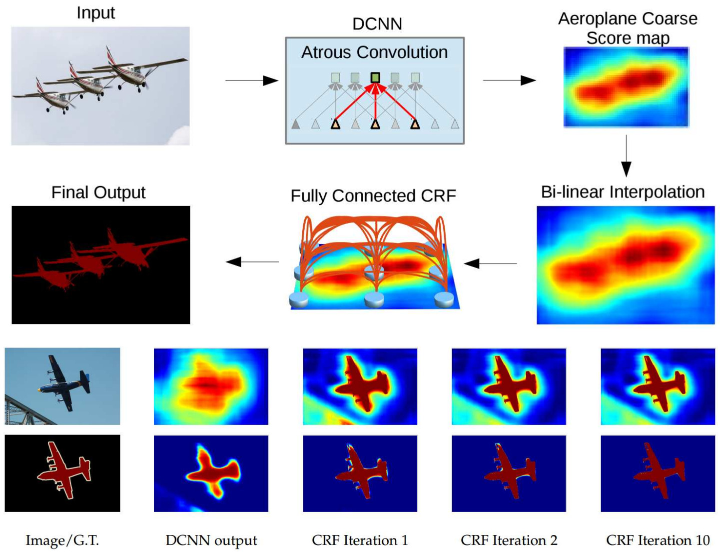
$$q_i(y_i) = \frac{1}{Z_i} \exp\left\{ -E_i(y_i) - \sum_{l' \in \mathcal{L}} [\![y_i \neq y_j]\!] \sum_{m=1}^{K} w^{(m)} \sum_{i \neq j} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) q_j(l') \right\} .$$

The inference is performed in average 0.2 seconds for 500.000 variables (in contrast to 36 hours).
The main idea: the message passing step can be expressed as a convolution with a Gaussian kernel $G_{\mathbf{\Sigma}^{(m)}}$ in feature space:

$$\sum_{j \in \mathcal{V}} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) q_j(l) - q_i(l) = [G_{\mathbf{\Sigma}^{(m)}} * q(l)](\mathbf{f}_i) - q_i(l) .$$

Note that the convolution sums over all variables, while message passing does not sum over q_i. This convolution can be efficiently calculated in $\mathcal{O}(|\mathcal{V}|)$ time (instead of $\mathcal{O}(|\mathcal{V}|^2)$).

DeepLab: CRF as post-processing



Source: Chen et al.. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. ICLR, 2015.

Summary *

Mean field approximation: instead of an *intractable* distribution $p(\mathbf{y} \mid \mathbf{x})$, we consider an *approximate distribution* $q(\mathbf{y})$, which minimizes the KL divergence.

In case of *naïve mean field approximation* $q(\mathbf{y})$ is defined as

$$q(\mathbf{y}) = \prod_{i \in \mathcal{V}} q_i(y_i) ,$$

which is tractable.

A local optimal solution can be obtained by applying the update equation:

$$q_i^*(y_i) = \frac{1}{Z_i(\mathbf{x}_F)} \exp\left(- \sum_{F \in M(i)} \sum_{\substack{\mathbf{y}'_F \in \mathcal{Y}_F, \\ y'_i = y_i}} \Big(\prod_{j \in N(F) \setminus \{i\}} q_j(y_j) \Big) E_F(\mathbf{y}_F; \mathbf{x}_F) \right) .$$

Next lecture *

In the **next lecture** we will learn about

- Sampling of a distribution ($p(\mathbf{y} \mid \mathbf{x})$) via *Gibbs sampling*.
- **Parameter learning**
  Consider an *energy function* for a *parameter vector* $\mathbf{w} = [w_1, w_2]^T$:

$$E(\mathbf{y}; \mathbf{x}, \mathbf{w}) = w_1 \sum_{i \in \mathcal{V}} E_i(y_i; x_i) + w_2 \sum_{(i,j) \in \mathcal{E}} E_{ij}(y_i, y_j) .$$

  We aim to estimate *optimal parameter vector* $\mathbf{w}$ consisting of (positive) weighting factors (like $w_1, w_2 \in \mathbb{R}^+$) for $E(\mathbf{y}; \mathbf{x}, \mathbf{w})$.

Literature *

Human pose estimation

1. Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, January 2005

Mean field approximation

2. Sebastian Nowozin and Christoph H. Lampert. Structured prediction and learning in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(3–4), 2010
3. Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009
4. Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Proceedings of Advances in Neural Information Processing Systems*, pages 109–117, Granada, Spain, December 2011. MIT Press
5. Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *Proceedings of International Conference on Learning Representations*, San Diego, CA, USA, May 2015