Statistical Machine Learning (BE4M33SSU)
Lecture 10: Markov Random Fields

Czech Technical University in Prague

- Markov Random Fields & Gibbs Random Fields
- Approximated Inference for MRFs
- (Generative) Parameter learning for MRFs

Motivation: Two Examples from Computer Vision

Example 1 (Image segmentation)

Recall the segmentation model used in the EM-Algorithm lab, where $\boldsymbol{x}\colon D \to \mathbb{R}^3$ denotes an image and $\boldsymbol{s}\colon D \to K$ denotes its segmentation (K – set of segment labels)

$$p(\boldsymbol{s}) = \prod_{i\in D} p(s_i) = \frac{1}{Z(u)} \exp \sum_{i\in D} u_i(s_i) \quad \text{and} \quad p(\boldsymbol{x} \mid \boldsymbol{s}) = \prod_{i\in D} p(x_i \mid s_i)$$

This model is pixelwise independent and, consequently, so is the inference.

We want to take into account that:

- neighbouring pixels belong more often than not to the same segment,
- the segment boundaries are in most places smooth, . . .

We may consider e.g. a prior model for segmentations

$$p(\boldsymbol{s}) = \frac{1}{Z(u)} \exp \Big[\sum_{i\in D} u_i(s_i) + \sum_{\{i,j\}\in E} u_{ij}(s_i, s_j) \Big],$$

where E are edges connecting neighbouring pixels in D.

Motivation: Two Examples from Computer Vision

Example 2 (Motion Flow)

Given two (consecutive) images $\boldsymbol{x}, \boldsymbol{x}' \colon D \to \mathbb{R}^3$ from a video, determine the motion flow, i.e. find a displacement vector v_i for each pixel $i \in D$.

- projections of the same 3D points look similar in $\boldsymbol{x}$ and $\boldsymbol{x}'$.
- 3D points projected onto neighbouring image pixels move more often than not coherently.
- Assume a discriminative model $p(\boldsymbol{v} \mid \boldsymbol{x}, \boldsymbol{x}')$ since the method does not intend to model the image appearance.

$$p(\boldsymbol{v} \mid \boldsymbol{x}, \boldsymbol{x}') = \frac{1}{Z(\boldsymbol{x}, \boldsymbol{x}')} \exp\Big[-\sum_{i \in D} \|\boldsymbol{x}_i - \boldsymbol{x}'_{i+v_i}\|^2 - \alpha \sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \Big]$$

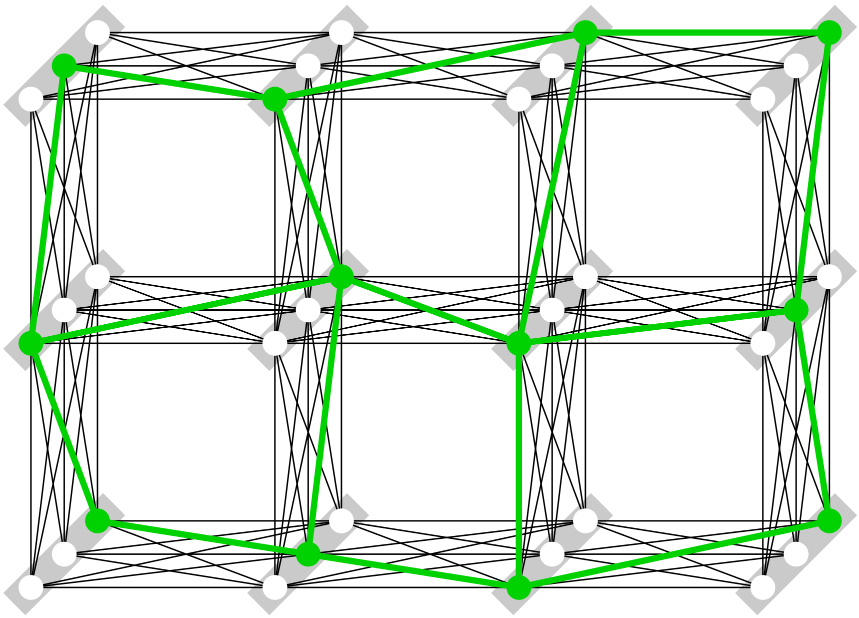
Such models can be generalised for stereo cameras and combined with segmentation approaches.

Markov Random Fields & Gibbs Random Fields

Let (V, E) denote an undirected graph and let $\boldsymbol{S} = \{S_i \mid i \in V\}$ be a field of random variables indexed by the nodes of the graph and taking values from a finite set K.

Definition 1 A joint probability distribution $p(\boldsymbol{s})$ is a Gibbs Random Field on the graph (V, E) if it factorises over the the nodes and edges, i.e.

$$p(\boldsymbol{s}) = \frac{1}{Z(u)} \exp\Big[\sum_{i \in V} u_i(s_i) + \sum_{\{i,j\} \in E} u_{ij}(s_i, s_j)\Big].$$



Remark 1 This can be generalised to Gibbs random fields on hypergraphs.

Markov Random Fields & Gibbs Random Fields

Definition 2 A probability distribution $p(\boldsymbol{s})$ is a Markov Random Field w.r.t. graph (V, E) if

$$p(\boldsymbol{s}_A, \boldsymbol{s}_B \mid \boldsymbol{s}_C) = p(\boldsymbol{s}_A \mid \boldsymbol{s}_C)\, p(\boldsymbol{s}_B \mid \boldsymbol{s}_C)$$

holds for any subsets $A, B \subset V$ and a separating set C.

Theorem 1 (Hammersley, Clifford, 1971)
If the distribution $p(\boldsymbol{s})$ is an MRF w.r.t. graph (V, E) and strictly positive, then it is a GRF on the hypergraph defined by all cliques of (V, E) and vice versa.

Remark 2 The following tasks for MRFs / GRFs are NP-complete

- Computing the most probable labelling $\boldsymbol{s}^* \in \arg\max\limits_{\boldsymbol{s} \in K^V} p(\boldsymbol{s})$.
- Computing the normalisation constant

$$Z(u) = \sum_{\boldsymbol{s} \in K^V} \exp\Big[\sum_{i \in V} u_i(s_i) + \sum_{\{i,j\} \in E} u_{ij}(s_i, s_j)\Big].$$

The same holds for computing marginal probabilities of $p(\boldsymbol{s})$.

Computing the most probable labelling, MRFs with boolean variables

Consider $\log p(\boldsymbol{s})$, replace $u \to -u$. The task reads then

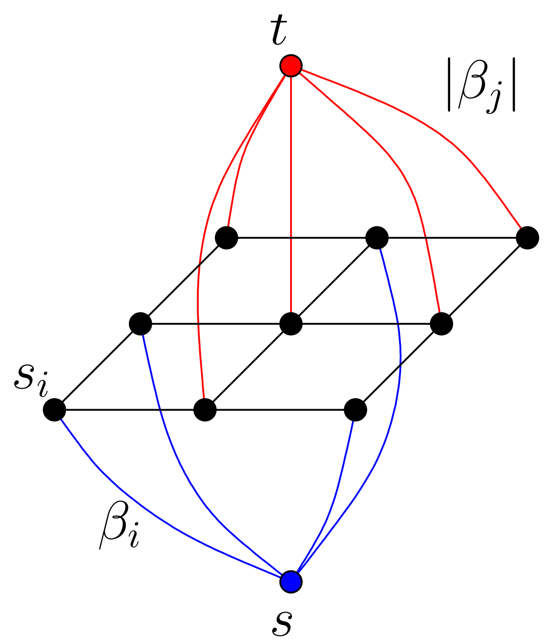
$$\sum_{i \in V} u_i(s_i) + \sum_{\{i,j\} \in E} u_{ij}(s_i, s_j) \to \min_{\boldsymbol{s} \in K^V}$$

If the variables s_i, $i \in V$ are boolean: the functions u_i, u_{ij} can be written as polynomials in the variables $s_i = 0, 1$, and, by re-defining the unary functions u_i if necessary, the task reads as

$$\begin{aligned} \boldsymbol{s}^* &= \arg\min_{\boldsymbol{s} \in K^V} \sum_{\{i,j\} \in E} \alpha_{ij} |s_i - s_j| + \sum_{i \in V} \beta_i s_i \\ &= \arg\min_{\boldsymbol{s} \in K^V} \sum_{\{i,j\} \in E} \alpha_{ij} |s_i - s_j| + \sum_{i \in V_+} \beta_i s_i + \sum_{i \in V_-} |\beta_i| (1 - s_i), \end{aligned}$$

where $V_+ = \{i \in V \mid \beta_i \geqslant 0\}$ and $V_- = V \setminus V_+$. This is a **MinCut-problem!**

Computing the most probable labelling, MRFs with boolean variables



- If all edge weights are non-negative, i.e. $\alpha_{ij} \geqslant 0$, $\forall \{i,j\} \in E$: the task can be solved via MinCut – MaxFlow duality,
- If some of the α-s are negative: apply approximation algorithms, e.g. relax the discrete variables to $s_i \in [0,1]$, consider an LP-relaxation of the task and solve the LP task e.g. by Tree-Reweighted Message Passing (Kolmogorov, 2006)

Computing the most probable labelling (general case)

Approximation algorithms for the general case, when $s_i \in K$

$$u(\boldsymbol{s}) = \sum_{i \in V} u_i(s_i) + \sum_{\{i,j\} \in E} u_{ij}(s_i, s_j) \to \min_{\boldsymbol{s} \in K^V}$$

Move making algorithms:

Construct a sequence of labellings $\boldsymbol{s}^{(t)}$ with decreasing values of the objective function $u(\boldsymbol{s}^{(t)})$:

- Define neighbourhoods $\mathcal{N}(\boldsymbol{s}) \subset K^V$ such that the task

  $$\operatorname*{arg\,min}_{\boldsymbol{s} \in \mathcal{N}(\boldsymbol{s}')} \sum_{\{i,j\} \in E} u_{ij}(s_i, s_j) + \sum_{i \in V} u_i(s_i)$$

  is tractable for every $\boldsymbol{s}'$.

- Iterate

  $$\boldsymbol{s}^{(t+1)} \in \operatorname*{arg\,min}_{\boldsymbol{s} \in \mathcal{N}(\boldsymbol{s}^{(t)})} \sum_{\{i,j\} \in E} u_{ij}(s_i, s_j) + \sum_{i \in V} u_i(s_i)$$

  until no further improvement possible.

Computing the most probable labelling (general case)

α-**Expansions** (Boykov et al., 2001)

- Define the neighbourhoods by choosing a label $\alpha \in K$ and setting

$$\mathcal{N}_\alpha(\boldsymbol{s}) = \left\{ \boldsymbol{s}' \in K^V \;\middle|\; s'_i = \alpha \text{ if } s'_i \neq s_i \right\}.$$

  Notice that $|\mathcal{N}_\alpha(\boldsymbol{s})| \sim 2^V$.

- The task

$$\operatorname*{arg\,min}_{\boldsymbol{s} \in \mathcal{N}_\alpha(\boldsymbol{s}')} \sum_{\{i,j\} \in E} u_{ij}(s_i, s_j) + \sum_{i \in V} u_i(s_i)$$

  can be encoded as labelling problem with boolean variables.

- It can be solved by MinCut-MaxFlow if

$$u_{ij}(k, k') + u_{ij}(\alpha, \alpha) \leqslant u_{ij}(\alpha, k') + u_{ij}(k, \alpha)$$

  holds for all pairwise functions u_{ij} and all $k, k' \in K$.

Learning parameters of MRFs

Learning task: Given i.i.d. training data $\mathcal{T}^m = \{\boldsymbol{s}^\ell \in K^V \mid \ell = 1, \ldots, m\}$, estimate the parameters u_i, u_{ij} of the MRF.

The maximum likelihood estimator reads

$$\log p_u(\mathcal{T}^m) = \frac{1}{m}\sum_{\ell=1}^{m}\Big[\sum_{\{i,j\}\in E} u_{ij}(s_i^\ell, s_j^\ell) + \sum_{i\in V} u_i(s_i^\ell)\Big] - \log Z(u) \to \max_{u_i, u_{ij}}.$$

It is intractable: the objective function is concave in u, but we can compute neither $\log Z(u)$ nor its gradient (in polynomial time).

We may use the **pseudo-likelihood** estimator (Besag, 1975) instead. It is based on the following observation

- Let $\mathcal{N}_i$ denote the neighbouring nodes of $i \in V$.
- We can compute the conditional distributions

$$p(s_i \mid s_{V\setminus i}) \stackrel{!}{=} p(s_i \mid s_{\mathcal{N}_i}) \sim e^{u_i(s_i)} \prod_{j\in\mathcal{N}_i} e^{u_{ij}(s_i, s_j)}$$

Learning parameters of MRFs

The pseudo-likelihood of an single example $\boldsymbol{s} \in \mathcal{T}^m$ is defined by

$$L_p(u) = \sum_{i \in V} \log p_u(s_i \mid s_{\mathcal{N}_i})$$
$$= 2 \sum_{\{i,j\} \in E} u_{ij}(s_i, s_j) + \sum_{i \in V} u_i(s_i) - \sum_{i \in V} \log \sum_{s_i \in K} \exp\Big[u_i(s_i) + \sum_{j \in \mathcal{N}_i} u_{ij}(s_i, s_j)\Big]$$

The pseudo-likelihood estimator is

- a concave function of the parameters u,
- tractable, i.e. both $L_p(u, \mathcal{T}^m)$ and its gradient are easy to compute,
- consistent.