Markov Random Fields for Computer Vision (Part 3)

Machine Learning Summer School (MLSS 2011)

Stephen Gould
stephen.gould@anu.edu.au

Australian National University

13–17 June, 2011

Higher-Order Constraints

$$
\begin{aligned}
E(\mathbf{y}; \mathbf{x}) &= \sum_{c} \psi_c(\mathbf{y}_c; \mathbf{x}) \\
&= \underbrace{\sum_{i \in \mathcal{V}} \psi_i^U(y_i; \mathbf{x})}_{\text{unary}} + \underbrace{\sum_{ij \in \mathcal{E}} \psi_{ij}^P(y_i, y_j; \mathbf{x})}_{\text{pairwise}} + \underbrace{\sum_{c \in \mathcal{C}} \psi_c^H(\mathbf{y}_c; \mathbf{x})}_{\text{higher-order}}.
\end{aligned}
$$

Higher-order terms allow us to encode stronger constraints:

- encourage label consistency over regions [Kohli et al., 2007]
- limit global occurrence of labels [Ladicky et al., 2010]
- enforce global connectivity [Vicente et al., 2008; Norowin et al., 2009]
- prefer segmentation "tightness" [Lempitsky et al., 2009]

Minimizing 2nd-Order Binary Functions

[Boros and Hammer, 2001], [Kolmogorov and Zabih, 2004], [Freedman and Drineas, 2005], [Ishikawa, 2009]

Consider a cubic pseudo-Boolean function over $\mathbf{y} = (y_1, y_2, y_3)$,

$$E(y_1, y_2, y_3) = -y_1 y_2 y_3.$$

Minimizing 2nd-Order Binary Functions

[Boros and Hammer, 2001], [Kolmogorov and Zabih, 2004], [Freedman and Drineas, 2005], [Ishikawa, 2009]

Consider a cubic pseudo-Boolean function over $\mathbf{y} = (y_1, y_2, y_3)$,

$$E(y_1, y_2, y_3) = -y_1 y_2 y_3.$$

Introducing auxiliary binary variable z, we can write

$$\begin{aligned}\min_{\mathbf{y}} E(y_1, y_2, y_3) &= \min_{\mathbf{y}} -y_1 y_2 y_3 \\ &= \min_{\mathbf{y}} \min_{z \in \{0,1\}} -z(y_1 + y_2 + y_3 - 2)\end{aligned}$$

Minimizing 2nd-Order Binary Functions

[Boros and Hammer, 2001], [Kolmogorov and Zabih, 2004], [Freedman and Drineas, 2005], [Ishikawa, 2009]

Consider a cubic pseudo-Boolean function over $\mathbf{y} = (y_1, y_2, y_3)$,

$$E(y_1, y_2, y_3) = -y_1 y_2 y_3.$$

Introducing auxiliary binary variable z, we can write

$$\begin{aligned}
\min_{\mathbf{y}} E(y_1, y_2, y_3) &= \min_{\mathbf{y}} -y_1 y_2 y_3 \\
&= \min_{\mathbf{y}} \min_{z \in \{0,1\}} -z(y_1 + y_2 + y_3 - 2) \\
&= \min_{\mathbf{y}, z} \underbrace{\bar{z} + \bar{y}_1 z + \bar{y}_2 z + \bar{y}_3 z - 1}_{\text{submodular}}
\end{aligned}$$

Minimizing 2nd-Order Binary Functions

[Boros and Hammer, 2001], [Kolmogorov and Zabih, 2004], [Freedman and Drineas, 2005], [Ishikawa, 2009]

Consider a cubic pseudo-Boolean function over $\mathbf{y} = (y_1, y_2, y_3)$,

$$E(y_1, y_2, y_3) = -y_1 y_2 y_3.$$

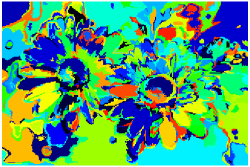
Introducing auxiliary binary variable z, we can write

$$\begin{aligned}
\min_{\mathbf{y}} E(y_1, y_2, y_3) &= \min_{\mathbf{y}} -y_1 y_2 y_3 \\
&= \min_{\mathbf{y}} \min_{z \in \{0,1\}} -z(y_1 + y_2 + y_3 - 2) \\
&= \min_{\mathbf{y}, z} \underbrace{\bar{z} + \bar{y}_1 z + \bar{y}_2 z + \bar{y}_3 z - 1}_{\text{submodular}}
\end{aligned}$$

The same trick applies to higher-order terms with negative coefficients. Reduction of terms with positive coefficients is possible, but the resulting energy function is non-submodular.

Higher-Order Consistency Constraints

image

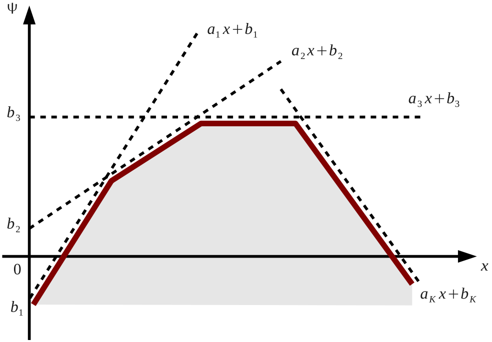


superpixels

segmentation

Lower Linear Envelopes [Kohli and Kumar, 2010]

$$\psi_c^H(\mathbf{y}_c) \triangleq \min_k \left\{ a_k \sum_{i \in \mathcal{C}} w_i [\![y_i = \ell_k]\!] + b_k \right\}$$

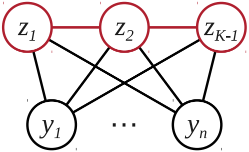


Minimizing Binary Lower Linear Envelopes [Gould, 2011]

$$\psi_c^H(\mathbf{y}_c) \triangleq \min_k \left\{ a_k \sum_{i \in \mathcal{C}} y_i + b_k \right\} = \min_k \{ f_k(\mathbf{y}_c) \}$$

Assume sorted on a_k. Introduce auxiliary binary random variables $\mathbf{z} = (z_1, \ldots, z_{K-1})$ such that $z_k \geq z_{k+1}$. Then

$$\min_{\mathbf{y}_c} \psi_c^H(\mathbf{y}_c) = \min_{\mathbf{y}_c, \mathbf{z}} \underbrace{f_1(\mathbf{y}_c) + \sum_k z_k \left(f_{k+1}(\mathbf{y}_c) - f_k(\mathbf{y}_c) \right)}_{\text{submodular binary pairwise MRF}}$$



Integer Programming

- Let us represent multi-label variable $y_i \in \mathcal{L}$ by a binary vector $(z_{i;1}, \ldots, z_{i;L})$ such that $z_{i;a} = 1$ if, and only if, $y_i = a$.

Integer Programming

- Let us represent multi-label variable $y_i \in \mathcal{L}$ by a binary vector $(z_{i;1}, \ldots, z_{i;L})$ such that $z_{i;a} = 1$ if, and only if, $y_i = a$.
- Let $\theta_{i;a} \triangleq \psi_i^U(y_i = a; \mathbf{x})$ and $\theta_{ij;ab} \triangleq \psi_{ij}^P(y_i = a, y_j = b; \mathbf{x})$.

Integer Programming

- Let us represent multi-label variable $y_i \in \mathcal{L}$ by a binary vector $(z_{i;1}, \ldots, z_{i;L})$ such that $z_{i;a} = 1$ if, and only if, $y_i = a$.
- Let $\theta_{i;a} \triangleq \psi_i^U(y_i = a; \mathbf{x})$ and $\theta_{ij;ab} \triangleq \psi_{ij}^P(y_i = a, y_j = b; \mathbf{x})$.

Then we can formulate energy minimization as a *binary integer programming* problem,

$$\text{minimize}_{\mathbf{y} \in \mathcal{L}^n} E\left(\mathbf{y}; \mathbf{x}\right)$$

$$\Updownarrow$$

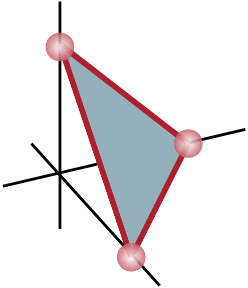
$$\begin{array}{ll}
\text{minimize} & \sum_{i \in \mathcal{V}} \sum_{a \in \mathcal{L}} \theta_{i;a} z_{i;a} + \sum_{ij \in \mathcal{E}} \sum_{a,b \in \mathcal{L}} \theta_{ij;ab} z_{ij;ab} \\
\text{subject to} & \sum_{a \in \mathcal{L}} z_{i;a} = 1 \\
& \sum_{a \in \mathcal{L}} z_{ij;ab} = z_{j;b} \\
& \sum_{b \in \mathcal{L}} z_{ij;ab} = z_{i;a} \\
& z_{i;a} \in \{0, 1\}, \; z_{ij;ab} \in \{0, 1\}
\end{array}$$

Integer Programming Relaxation

$$\begin{array}{ll} \text{minimize} & \sum_{i \in \mathcal{V}} \sum_{a \in \mathcal{L}} \theta_{i;a} z_{i;a} + \sum_{ij \in \mathcal{E}} \sum_{a,b \in \mathcal{L}} \theta_{ij;ab} z_{ij;ab} \\ \text{subject to} & \sum_{a \in \mathcal{L}} z_{i;a} = 1 \\ & \sum_{a \in \mathcal{L}} z_{ij;ab} = z_{j;b} \\ & \sum_{b \in \mathcal{L}} z_{ij;ab} = z_{i;a} \\ & z_{i;a} \in \{0,1\}, \; z_{ij;ab} \in \{0,1\} \end{array}$$

Integer Programming Relaxation

$$\begin{array}{ll} \text{minimize} & \sum_{i \in \mathcal{V}} \sum_{a \in \mathcal{L}} \theta_{i;a} z_{i;a} + \sum_{ij \in \mathcal{E}} \sum_{a,b \in \mathcal{L}} \theta_{ij;ab} z_{ij;ab} \\ \text{subject to} & \sum_{a \in \mathcal{L}} z_{i;a} = 1 \\ & \sum_{a \in \mathcal{L}} z_{ij;ab} = z_{j;b} \\ & \sum_{b \in \mathcal{L}} z_{ij;ab} = z_{i;a} \\ & z_{i;a} \in [0,1], \; z_{ij;ab} \in [0,1] \end{array}$$



Integer Programming Relaxation

MAP Linear Programming Relaxation

$$\text{minimize} \quad \boldsymbol{\theta}^T \mathbf{z} \quad \text{subject to} \quad \mathbf{z} \in \mathcal{M}^{\text{local}}$$

Integer Programming Relaxation

MAP Linear Programming Relaxation

$$\text{minimize} \quad \boldsymbol{\theta}^T \mathbf{z} \quad \text{subject to} \quad \mathbf{z} \in \mathcal{M}^{\text{local}}$$

Advantages:

- tractable
- provides a lower bound
- more "stable" for learning

Disadvantages:

- LP is typically very large
- solution is not integral (i.e., needs rounding)

A number of specialized techniques have been developed to solve the large-scale linear programs found in computer vision (e.g., [Wainwright et al., 2005; Werner, 2005; Yanover et al., 2006; Globerson and Jaakkola, 2007; Komodakis et al., 2007]).

Dual Decomposition [Komodakis and Paragios, 2009]

Main idea:

- start with integer program
- introduce duplicate variables and split into tractable *slaves*
- add *coupling constraints* between duplicated variables
- maximize the *dual problem*

Dual Decomposition [Komodakis and Paragios, 2009]

Main idea:

- start with integer program

$$\text{minimize}_{\mathbf{y}} \ \sum_{c \in \mathcal{C}} \psi_c(\mathbf{y}; \mathbf{x})$$

- introduce duplicate variables and split into tractable *slaves*

- add *coupling constraints* between duplicated variables

- maximize the *dual problem*

Dual Decomposition [Komodakis and Paragios, 2009]

Main idea:

- start with integer program

$$\text{minimize}_{\mathbf{y}} \ \sum_{c \in \mathcal{C}} \psi_c(\mathbf{y}; \mathbf{x})$$

- introduce duplicate variables and split into tractable *slaves*

$$\text{minimize}_{\mathbf{y}} \ \sum_{c \in \mathcal{C}} \psi_c(\mathbf{y}^{(c)}; \mathbf{x})$$

- add *coupling constraints* between duplicated variables

$$\text{subject to} \ \ y_i^{(c)} = y_i$$

- maximize the *dual problem*

Dual Decomposition [Komodakis and Paragios, 2009]

Main idea:

- start with integer program

$$\text{minimize}_{\mathbf{y}} \ \sum_{c \in \mathcal{C}} \psi_c(\mathbf{y}; \mathbf{x})$$

- introduce duplicate variables and split into tractable *slaves*

$$\text{minimize}_{\mathbf{y}} \ \sum_{c \in \mathcal{C}} \psi_c(\mathbf{y}^{(c)}; \mathbf{x})$$

- add *coupling constraints* between duplicated variables

$$\text{subject to} \ \ y_i^{(c)} = y_i$$

- maximize the *dual problem*

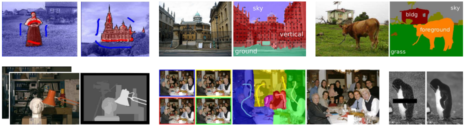
$$\begin{array}{ll} \text{maximize}_{\boldsymbol{\lambda}} & \sum_{c \in \mathcal{C}} \min_{\mathbf{y}^{(c)}} \left\{ \psi_c(\mathbf{y}^{(c)}; \mathbf{x}) - \boldsymbol{\lambda}_c^T \mathbf{y}^{(c)} \right\} \\ \text{subject to} & \sum_{c \in \mathcal{C}} \boldsymbol{\lambda}_c = \mathbf{0} \end{array}$$

Open Problems

- What is the class of multi-label or higher-order functions that can be transformed into submodular quadratic pseudo-Boolean functions?
- Are there better max-flow algorithms for solving energy minimization problems with high connectivity (i.e., large neighbourhoods)?
- How best to solve large-scale integer programs for computer vision applications?
- How can we learn the parameters from data? (next tutorial)
- ...

Summary

- Pixel labeling CRFs
- Pseudo-boolean fcns
- Higher-order terms
- Integer programming



Please feel free to contact me if you are interested in research at the intersection between computer vision and machine learning.

stephen.gould@anu.edu.au