Discrete Inference and Learning
Lecture 5

MVA
2017 – 2018

http://thoth.inrialpes.fr/~alahari/disinflearn

Slides based on material from Nikos Komodakis, M. Pawan Kumar
Outline

• Last class
  – Primal-dual schema
  – Fast Primal-dual (FastPD) algorithm
  – Dual decomposition

• Today
  – Recap of the course
  – Learning parameters
Primal-dual schema

• Goal: Find integral-primal solution $x$, feasible dual solution $y$,
  – such that their primal-dual costs are “close enough”, e.g.,

\[
\frac{c^T x}{b^T y} \leq f^* \quad \text{and} \quad \frac{c^T x}{c^T x^*} \leq f^*
\]

$f^*$-approximation to the optimal $x^*$
Primal-dual schema

- Works iteratively

\[ \frac{c^T x^k}{b^T y^k} \leq f^* \]

- Easier to use relaxed complementary slackness, instead of working directly with costs

\[ c^T x^* \rightarrow \cdots \rightarrow c^T x^2 \rightarrow c^T x^1 \]

\[ b^T y^1 \rightarrow b^T y^2 \rightarrow \cdots \rightarrow b^T y^k \]
Primal-dual schema

- Relaxed complementary slackness

primal LP: \( \min c^T x \)
\[ \text{s.t. } A x = b, x \geq 0 \]

dual LP: \( \max b^T y \)
\[ \text{s.t. } A^T y \leq c \]

**Exact CS:** \( \forall 1 \leq j \leq n : x_j > 0 \Rightarrow \sum_{i=1}^{m} a_{ij} y_i = c_j \)

**Relaxed CS:** \( \forall 1 \leq j \leq n : x_j > 0 \Rightarrow \sum_{i=1}^{m} a_{ij} y_i \geq c_j / f_j \)
Dual decomposition

• Reduces MRF optimization to a simple projected subgradient method

• Combines solutions from sub-problems in a principled and optimal manner

• Applies to a wide variety of cases
Dual decomposition

- Decomposition into subproblems (slaves)
- Coordination of slaves by a master process
Dual decomposition

• Master
  – updates the parameters of the slave-MRFs by “averaging” the solutions returned by the slaves
  – tries to achieve consensus among all slave-MRFs
  – e.g., if a certain node is already assigned the same label by all minimizers, the master does not touch the MRF potentials of that node.
Comparison: TRW and DD

**TRW**
- Fast
- Local Maximum
- Requires Min-Marginals

**DD**
- Slow
- Global Maximum
- Requires MAP Estimate
Outline

• Recap of the course

• Learning parameters
Conditional Random Fields (CRFs)

- Ubiquitous in computer vision
  - segmentation  stereo matching
  - optical flow  image restoration
  - image completion object detection/localization
  ...

- and beyond
  - medical imaging, computer graphics, digital communications, physics...

- Really powerful formulation
Conditional Random Fields (CRFs)

• Key task: inference/optimization for CRFs/MRFs

• Extensive research for more than 20 years

• Lots of progress

• Many state-of-the-art methods:
  • Graph-cut based algorithms
  • Message-passing methods
  • LP relaxations
  • Dual Decomposition
  • ....
MAP inference for CRFs/MRFs

• Hypergraph $G = (\mathcal{V}, \mathcal{C})$
  – Nodes $\mathcal{V}$
  – Hyperedges/cliques $\mathcal{C}$

• High-order MRF energy minimization problem

$$\text{MRF}_G(U, H) \equiv \min_x \sum_{q \in \mathcal{V}} U_q(x_q) + \sum_{c \in \mathcal{C}} H_c(x_c)$$

unary potential (one per node)  high-order potential (one per clique)
CRF training

• But how do we choose the CRF potentials?

• Through training
  • Parameterize potentials by $w$
  • Use training data to learn correct $w$

• Characteristic example of structured output learning [Taskar], [Tsochantaridis, Joachims]

• Equally, if not more, important than MAP inference
  • Better optimize correct energy (even approximately)
  • Than optimize wrong energy exactly
Outline

• Supervised Learning

• Probabilistic Methods

• Loss-based Methods

• Results
Image Classification

Is this an urban or rural area?

Input: $d$  
Output: $x \in \{-1,+1\}$
Image Classification

Is this scan healthy or unhealthy?

Input: $d$  \hspace{1cm} Output: $x \in \{-1, +1\}$
Image Classification

Labeling $\mathbf{X} = \mathbf{x}$

Label set $\mathbf{L} = \{-1,+1\}$
Image Classification

Which city is this?

Input: \(d\)  
Output: \(x \in \{1,2,\ldots,h\}\)
Image Classification

What type of tumor does this scan contain?

Input: $d$  
Output: $x \in \{1,2,...,h\}$
Object Detection

Where is the object in the image?

Input: $d$  
Output: $x \in \{\text{Pixels}\}$
Object Detection

Where is the rupture in the scan?

Input: $d$  
Output: $x \in \{\text{Pixels}\}$
Object Detection

Labeling $X = x$  
Label set $L = \{1, 2, ..., h\}$
Segmentation

What is the semantic class of each pixel?

Input: $d$  
Output: $x \in \{1, 2, \ldots, h\}^{|\text{Pixels}|}$
Segmentation

What is the muscle group of each pixel?

Input: \(d\) 

Output: \(x \in \{1,2,...,h\}^{\text{Pixels}}\)
Segmentation

Labeling $X = x$  
Label set $L = \{1, 2, \ldots, h\}$
Segmentation

Labeling $X = x$  
Label set $L = \{1, 2, \ldots, h\}$
CRF training

- Stereo matching:
  - $Z$: left, right image
  - $X$: disparity map

Goal of training: estimate proper $w$

\[ f = \arg\min_x \text{MRF}_G(x; u, h) \]

$w$ parameterized by $w$
CRF training

- Denoising:
  - $Z$: noisy input image
  - $X$: denoised output image

Goal of training: estimate proper $w$

$$f = \arg\min_x \text{MRF}_G(x; u, h)$$

(parameterized by $w$)
CRF training

- Object detection:
  - Z: input image
  - X: position of object parts

Goal of training: estimate proper \( w \)

\[
f = \arg\min_x \text{MRF}_G(x; u, h)
\]

parameterized by \( w \)
CRF training (some further notation)

$$\text{MRF}_G(x; u^k, h^k) = \sum_p u^k_p(x_p) + \sum_c h^k_c(x_c)$$

$$u^k_p(x_p) = w^T g_p(x_p, z^k), \quad h^k_c(x_c) = w^T g_c(x_c, z^k)$$

vector valued feature functions

$$\text{MRF}_G(x; w, z^k) = w^T \left( \sum_p g_p(x_p, z^k) + \sum_c g_c(x_c, z^k) \right) = w^T g(x, z^k)$$
Learning formulations
Risk minimization

\[ \hat{x}^k = \arg \min_x \text{MRF}_G(x; w, z^k) \]

\[ \min_w \sum_{k=1}^K \Delta (x^k, \hat{x}^k) \]

$K$ training samples \( \{(x^k, z^k)\}_{k=1}^K \)
Regularized Risk minimization

\[
\hat{x}^k = \arg \min_x MRF_G(x; w, z^k)
\]

\[
\min_w R(w) + \sum_{k=1}^{K} \Delta(x^k, \hat{x}^k)
\]

\[
R(w) = \|w\|^2, \|w\|_1, \text{etc.}
\]
Regularized Risk minimization

\[ \min_w R(w) + \sum_{k=1}^{K} L_G \left( x^k, z^k; w \right) \]

Replace \( \Delta(.) \) with easier to handle upper bound \( L_G \)
(e.g., convex w.r.t. \( w \))

\[ \min_w R(w) + \sum_{k=1}^{K} \Delta \left( x^k, \hat{x}^k \right) \]
Choice 1: Hinge loss

\[
\min_w R(w) + \sum_{k=1}^{K} L_G (x^k, z^k; w)
\]

\[
L_G (x^k, z^k; w) = MRF_G(x^k; w, z^k) - \min_x (MRF_G(x; w, z^k) - \Delta(x, x^k))
\]

- Upper bounds \(\Delta(.)\)

- Leads to max-margin learning
Max-margin learning

\[ \text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) \]
Max-margin learning

\[ \text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) \]

energy of
ground truth
Max-margin learning

\[ MRF_G(x^k; w, z^k) \leq MRF_G(x; w, z^k) - \Delta(x, x^k) \]

energy of ground truth \hspace{2cm} any other energy
Max-margin learning

\[ \text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) \]

energy of ground truth \hspace{1cm} any other energy \hspace{1cm} desired margin
Max-margin learning

\[ \text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k \]

energy of ground truth
any other energy
desired margin
slack
Max-margin learning

\[
\min_w \sum_k \xi_k
\]

subject to the constraints:

\[
\text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k
\]

- energy of ground truth
- any other energy
- desired margin
- slack
Max-margin learning

$$\min_{w} R(w) + \sum_{k} \xi_k$$

subject to the constraints:

$$MRF_G(x^k; w, z^k) \leq MRF_G(x; w, z^k) - \Delta(x, x^k) + \xi_k$$

- energy of ground truth
- any other energy
- desired margin
- slack
Max-margin learning

\[
\min_w R(w) + \sum_k \xi_k
\]

subject to the constraints:

\[
\text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k
\]
Max-margin learning

\[
\min_w R(w) + \sum_k \xi_k
\]

subject to the constraints:

\[
\text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k
\]

or equivalently

\[
\begin{align*}
\min_w R(w) + \sum_k \xi_k \\
\xi_k &= \text{MRF}_G(x^k; w, z^k) - \min_x \left( \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) \right)
\end{align*}
\]
Max-margin learning

\[
\min_w R(w) + \sum_k \xi_k
\]

subject to the constraints:

\[
\text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k
\]

or equivalently

\[
\xi_k = \text{MRF}_G(x^k; w, z^k) - \min_x \left( \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) \right)
\]
Max-margin learning

Minimize $R(w) + \sum_k \xi_k$

subject to the constraints:

$\text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k$

or equivalently

Minimize $R(w) + \sum_k \xi_k$

$\xi_k = \text{MRF}_G(x^k; w, z^k) - \min_x (\text{MRF}_G(x; w, z^k) - \Delta(x, x^k))$
Choice 2: logistic loss

\[
\min_w R(w) + \sum_{k=1}^{K} L_G(x^k, z^k; w)
\]

\[
L_G(x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) + \log \sum_x e^{-\text{MRF}_G(x; w, z^k)}
\]

- Can be shown to lead to maximum likelihood learning
Max-margin vs Maximum-likelihood

\[ L_G (x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) - \min_x (\text{MRF}_G(x; w, z^k) - \Delta(x, x^k)) \]

\[ L_G (x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) + \log \sum_x e^{-\text{MRF}_G(x; w, z^k)} \]
Max-margin vs Maximum-likelihood

\[ L_G(x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) + \max_x \left( -\text{MRF}_G(x; w, z^k) + \Delta(x, x^k) \right) \]

\[ L_G(x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) + \log \sum_x e^{-\text{MRF}_G(x; w, z^k)} \]
Solving the learning formulations
Maximum-likelihood learning

\[
\min_{w} \frac{\mu}{2} ||w||^2 + \sum_{k=1}^{K} L_G (x^k, z^k; w)
\]

\[
L_G (x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) + \log \sum_{x} e^{-\text{MRF}_G(x; w, z^k)}
\]

- Differentiable & convex
- Global optimum via gradient descent, for example
Maximum-likelihood learning

\[ \min_{\mathbf{w}} \frac{\mu}{2} \| \mathbf{w} \|^2 + \sum_{k=1}^{K} L_G (\mathbf{x}^k, \mathbf{z}^k; \mathbf{w}) \]

\[ L_G (\mathbf{x}^k, \mathbf{z}^k; \mathbf{w}) = \text{MRF}_G (\mathbf{x}^k; \mathbf{w}, \mathbf{z}^k) + \log \sum_{\mathbf{x}} e^{-\text{MRF}_G (\mathbf{x}; \mathbf{w}, \mathbf{z}^k)} \]

gradient \[ \nabla_\mathbf{w} = \mathbf{w} + \sum_{k} \left( g(\mathbf{x}^k, \mathbf{z}^k) - \sum_{\mathbf{x}} p(\mathbf{x}|\mathbf{w}, \mathbf{z}^k) g(\mathbf{x}, \mathbf{z}^k) \right) \]

Recall that: \[ \text{MRF}_G (\mathbf{x}; \mathbf{w}, \mathbf{z}^k) = \mathbf{w}^T g(\mathbf{x}, \mathbf{z}^k) \]
Maximum-likelihood learning

$$\min_{\mathbf{w}} \frac{\mu}{2} \| \mathbf{w} \|^2 + \sum_{k=1}^{K} L_{G} (\mathbf{x}^k, \mathbf{z}^k; \mathbf{w})$$

$$L_{G} (\mathbf{x}^k, \mathbf{z}^k; \mathbf{w}) = \text{MRF}_G (\mathbf{x}^k; \mathbf{w}, \mathbf{z}^k) + \log \sum_{\mathbf{x}} e^{-\text{MRF}_G (\mathbf{x}; \mathbf{w}, \mathbf{z}^k)}$$

gradient $\nabla_{\mathbf{w}} = \mathbf{w} + \sum_{k} \left( g(\mathbf{x}^k, \mathbf{z}^k) - \sum_{\mathbf{x}} p(\mathbf{x}|\mathbf{w}, \mathbf{z}^k) g(\mathbf{x}, \mathbf{z}^k) \right)$

- Requires MRF probabilistic inference
- **NP-hard** (exponentially many $\mathbf{x}$): approximation via loopy-BP ?
Max-margin learning (UNCONSTRAINED)

\[
\min_w R(w) + \sum_{k=1}^{K} L_G (x^k, z^k; w)
\]

\[
L_G (x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) - \min_x (\text{MRF}_G(x; w, z^k) - \Delta(x, x^k))
\]

- Convex but non-differentiable
- Global optimum via subgradient method
Subgradient

Subgradient at $x_2 = \text{gradient at } x_2$

$g(x_2) + h_2 \cdot (x - x_2)$

Subgradient at $x_1$

$g(x_1) + h_1 \cdot (x - x_1)$
Subgradient

**Lemma.** Let $f(\cdot) = \max_{m=1,\ldots,M} f_m(\cdot)$, with $f_m(\cdot)$ convex and differentiable. A subgradient of $f$ at $y$ is given by $\nabla f_{\hat{m}}(y)$, where $\hat{m}$ is any index for which $f(y) = f_{\hat{m}}(y)$. 
Lemma. Let \( f(\cdot) = \max_{m=1,...,M} f_m(\cdot) \), with \( f_m(\cdot) \) convex and differentiable. A subgradient of \( f \) at \( y \) is given by \( \nabla f_{\hat{m}}(y) \), where \( \hat{m} \) is any index for which \( f(y) = f_{\hat{m}}(y) \).
**Subgradient**

**Lemma.** Let $f(\cdot) = \max_{m=1, \ldots, M} f_m(\cdot)$, with $f_m(\cdot)$ convex and differentiable. A subgradient of $f$ at $y$ is given by $\nabla f_{\hat{m}}(y)$, where $\hat{m}$ is any index for which $f(y) = f_{\hat{m}}(y)$. 

Lemma. Let $f(\cdot) = \max_{m=1,\ldots,M} f_m(\cdot)$, with $f_m(\cdot)$ convex and differentiable. A subgradient of $f$ at $y$ is given by $\nabla f_{\hat{m}}(y)$, where $\hat{m}$ is any index for which $f(y) = f_{\hat{m}}(y)$. 
Lemma. Let \( f(\cdot) = \max_{m=1,\ldots,M} f_m(\cdot) \), with \( f_m(\cdot) \) convex and differentiable. A subgradient of \( f \) at \( y \) is given by \( \nabla f_{\hat{m}}(y) \), where \( \hat{m} \) is any index for which \( f(y) = f_{\hat{m}}(y) \).
Lemma. Let $f(\cdot) = \max_{m=1,...,M} f_m(\cdot)$, with $f_m(\cdot)$ convex and differentiable. A subgradient of $f$ at $y$ is given by $\nabla f_{\hat{m}}(y)$, where $\hat{m}$ is any index for which $f(y) = f_{\hat{m}}(y)$. 
**Subgradient**

**Lemma.** Let \( f(\cdot) = \max_{m=1, \ldots, M} f_m(\cdot) \), with \( f_m(\cdot) \) convex and differentiable. A subgradient of \( f \) at \( y \) is given by \( \nabla f_{\hat{m}}(y) \), where \( \hat{m} \) is any index for which \( f(y) = f_{\hat{m}}(y) \).
**Lemma.** Let \( f(\cdot) = \max_{m=1,...,M} f_m(\cdot) \), with \( f_m(\cdot) \) convex and differentiable. A subgradient of \( f \) at \( y \) is given by \( \nabla f_{\hat{m}}(y) \), where \( \hat{m} \) is any index for which \( f(y) = f_{\hat{m}}(y) \).
Lemma. Let \( f(\cdot) = \max_{m=1,\ldots,M} f_m(\cdot) \), with \( f_m(\cdot) \) convex and differentiable. A subgradient of \( f \) at \( y \) is given by \( \nabla f_{\hat{m}}(y) \), where \( \hat{m} \) is any index for which \( f(y) = f_{\hat{m}}(y) \).

\[
L_G(x^k, z^k; w) = \operatorname{MRF}_G(x^k; w, z^k) - \min_x \left( \operatorname{MRF}_G(x; w, z^k) - \Delta(x, x^k) \right)
\]

\[
\operatorname{MRF}_G(x; w, z^k) = w^T g(x, z^k)
\]

The subgradient of \( L_G \) is

\[
\hat{x}^k = \arg \min_x \left( \operatorname{MRF}_G(x; w, z^k) - \Delta(x, x^k) \right)
\]

\[
\nabla L_G = g(x^k, z^k) - g(\hat{x}^k, z^k)
\]
Max-margin learning (UNCONSTRAINED)

\[
\min_w R(w) + \sum_{k=1}^{K} L_G(x^k, z^k; w)
\]

\[
L_G(x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) - \min_x (\text{MRF}_G(x; w, z^k) - \Delta(x, x^k))
\]

<table>
<thead>
<tr>
<th>Subgradient algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Repeat</strong></td>
</tr>
<tr>
<td>1. compute global minimizers ( \hat{x}^k ) at current ( w )</td>
</tr>
<tr>
<td>2. compute <strong>total subgradient</strong> at current ( w )</td>
</tr>
<tr>
<td>3. update ( w ) by taking a step in the negative total subgradient direction</td>
</tr>
<tr>
<td><strong>until convergence</strong></td>
</tr>
</tbody>
</table>

\[
\text{total subgr.} = \text{subgradient}_w[R(w)] + \sum_{k} (g(x^k, z^k) - g(\hat{x}^k, z^k))
\]
Max-margin learning (UNCONSTRAINED)

\[
\min_{w} R(w) + \sum_{k=1}^{K} L_{G}(x^{k}, z^{k}; w)
\]

\[
L_{G}(x^{k}, z^{k}; w) = MRF_{G}(x^{k}; w, z^{k}) - \min_{x} (MRF_{G}(x; w, z^{k}) - \Delta(x, x^{k}))
\]

**Stochastic subgradient algorithm**

Repeat

1. pick \( k \) at random
2. compute global minimizer \( \hat{x}^{k} \) at current \( w \)
3. compute **partial subgradient** at current \( w \)
4. update \( w \) by taking a step in the negative partial subgradient direction

until convergence

**partial subgradient** = \( \text{subgradient}_{w}[R(w)] + g(x^{k}, z^{k}) - g(\hat{x}^{k}, z^{k}) \)

MRF-MAP estimation per iteration (unfortunately NP-hard)
Max-margin learning (CONSTRAINED)

$$\min_w R(w) + \sum_k \xi_k$$

subject to the constraints:

$$\text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k$$
Max-margin learning (CONstrained)

\[
\min \limits_{w} \frac{\mu}{2} \|w\|^2 + \sum_{k} \xi_k
\]

subject to the constraints:

\[
MRF_G(x^k; w, z^k) \leq MRF_G(x; w, z^k) - \Delta(x, x^k) + \xi_k
\]
Max-margin learning (CONSTRAINED)

\[
\min_w \frac{\mu}{2} \|w\|^2 + \sum_k \xi_k
\]

subject to the constraints:

\[
\text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k
\]

- Quadratic program (great!)
- But exponentially many constraints (not so great)
Max-margin learning (CONSTRAINED)

- What if we use only a small number of constraints?
  - Resulting QP can be solved
  - But solution may be infeasible

- **Constraint generation** to the rescue
  - only few constraints **active** at optimal solution!!
    (variables much fewer than constraints)
  - Given the active constraints, rest can be ignored
  - Then let us try to find them!
Constraint generation

1. Start with some constraints
2. Solve QP
3. Check if solution is feasible w.r.t. to all constraints
4. If yes, we are done!
5. If not, pick a violated constraint and add it to the current set of constraints. Repeat from step 2. (optionally, we can also remove inactive constraints)
Constraint generation

• **Key issue:** we must always be able to find a violated constraint if one exists

• Recall the constraints for max-margin learning

\[ MRF_G(x^k; w, z^k) \leq MRF_G(x; w, z^k) - \Delta(x, x^k) + \xi_k \]

• To find violated constraint, we therefore need to compute:

\[ \hat{x}^k = \arg \min_x (MRF_G(x; w, z^k) - \Delta(x, x^k)) \]

(just like subgradient method!)
1. Initialize set of constraints $C$ to empty

2. Solve QP using current constraints $C$ and obtain new $(w, \xi)$

3. Compute global minimizers $\hat{x}^k$ at current $w$

4. For each $k$, if the following constraint is violated then add it to set $C$:
   \[
   \text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(\hat{x}^k; w, z^k) - \Delta(\hat{x}^k, x^k) + \xi_k
   \]

5. If no new constraint was added then terminate. Otherwise go to step 2.

MRF-MAP estimation per sample (unfortunately NP-hard)
Max-margin learning (CONSTRAINED)

\[
\min_w \frac{\mu}{2} \|w\|^2 + \sum_k \xi_k
\]

subject to the constraints:
\[
\text{MRF}_G(x^k; w, z^k) \leq \text{MRF}_G(x; w, z^k) - \Delta(x, x^k) + \xi_k
\]

• Alternatively, we can solve above QP in the dual domain
• dual variables ↔ primal constraints
• Too many variables, but most of them zero at optimal solution
• Use a working-set method
  (essentially dual to constraint generation)
CRF training

• Existing max-margin (maximum likelihood) methods:
  • use MAP inference (probabilistic inference) w.r.t. an equally complex CRF as subroutine
  • have to call subroutine many times during learning

• Suboptimal
  • computational efficiency ?
  • accuracy ?
  • theoretical guarantees/properties ?

• Key issue: can we exploit the CRF structure more aptly during training?
CRF Training via Dual Decomposition

- Efficient max-margin training method

- Reduces training of complex CRF to parallel training of a series of easy-to-handle slave CRFs

- Handles arbitrary pairwise or higher-order CRFs

- Uses very efficient projected subgradient learning scheme

- Allows hierarchy of structured prediction learning algorithms of increasing accuracy
Dual Decomposition for MRF Optimization
(another recap)
MRF Optimization via Dual Decomposition

• Very general framework for MAP inference [Komodakis et al. ICCV07, PAMI11]

• Master = coordinator (has global view)
  Slaves = subproblems (have only local view)
MRF Optimization via Dual Decomposition

• Very general framework for MAP inference \cite{Komodakis07, Komodakis11}

• Master = \( \text{MRF}_G(u, h) \) (MAP-MRF on hypergraph \( G \))
  \[
  = \min \text{MRF}_G(x; u, h) := \sum_{p \in V} u_p(x_p) + \sum_{c \in C} h_c(x_c)
  \]
MRF Optimization via Dual Decomposition

- Very general framework for MAP inference [Komodakis et al. ICCV07, PAMI11]

Set of slaves = \( \{ \text{MRF}_{G_i}(\theta^i, h) \} \)

(MRFs on sub-hypergraphs \( G_i \) whose union covers \( G \))

- Many other choices possible as well
MRF Optimization via Dual Decomposition

- Very general framework for MAP inference [Komodakis et al. ICCV07, PAMI11]

- Optimization proceeds in an iterative fashion via master-slave coordination
MRF Optimization via Dual Decomposition

Set of slave MRFs
\( \{ \text{MRF}_{G_i}(\theta^i, h) \} \)

convex dual relaxation

\[
\text{DUAL}_{\{G_i\}}(u, h) = \max_{\{\theta^i\}} \sum_i \text{MRF}_{G_i}(\theta^i, h) \\
\text{s.t. } \sum_{i \in \mathcal{I}_p} \theta^i(\cdot) = u_p(\cdot)
\]

For each choice of slaves, master solves (possibly different) dual relaxation

- Sum of slave energies = lower bound on MRF optimum
- Dual relaxation = maximum such bound
MRF Optimization via Dual Decomposition

Set of slave MRFs
\( \{ \text{MRF}_{G_i}(\theta^i, h) \} \)

\[ \text{DUAL}_{\{G_i\}}(u, h) = \max_{\{\theta^i\}} \sum_i \text{MRF}_{G_i}(\theta^i, h) \]

\[ \text{s.t. } \sum_{i \in \mathcal{I}_p} \theta^i(\cdot) = u_p(\cdot) \]

Choosing more difficult slaves \( \Rightarrow \) tighter lower bounds
\( \Rightarrow \) tighter dual relaxations
CRF training via Dual Decomposition
Max-margin learning via dual decomposition

$$\min_w R(w) + \sum_{k=1}^{K} L_G (x^k, z^k; w)$$

$$L_G (x^k, z^k; w) = \text{MRF}_G(x^k; w, z^k) - \min_x (\text{MRF}_G(x; w, z^k) - \Delta(x, x^k))$$
Max-margin learning via dual decomposition

\[
\min_{w} R(w) + \sum_{k=1}^{K} L_G \left( x^k, z^k ; w \right)
\]

\[
L_G \left( x^k, z^k ; w \right) = \text{MRF}_G(x^k; u^k, h^k) - \min_{x} \left( \text{MRF}_G(x; u^k, h^k) - \Delta(x, x^k) \right)
\]
Max-margin learning via dual decomposition

$$\min_{w} R(w) + \sum_{k=1}^{K} L_G(x^k, z^k; w)$$

where

$$L_G(x^k, z^k; w) = \text{MRF}_G(x^k; u^k, h^k) - \min_x (\text{MRF}_G(x; u^k, h^k) - \Delta(x, x^k))$$

$$\Delta(x, x^k) = \sum_p \delta_p(x_p, x_p^k) + \sum_c \delta_c(x_c, x_c^k)$$

and $$\Delta(x, x) = 0$$

$$\tilde{u}_p^k(\cdot) = u_p^k(\cdot) - \delta_p(\cdot, x_p^k)$$

$$\tilde{h}_c^k(\cdot) = h_c^k(\cdot) - \delta_c(\cdot, x_c^k)$$

loss-augmented potentials
Max-margin learning via dual decomposition

\[
\min_w R(w) + \sum_{k=1}^K L_G(x^k, z^k; w)
\]

\[
L_G(x^k, z^k; w) = MRF_G(x^k; u^k, h^k) - \min_x MRF_G(x; \bar{u}^k, \bar{h}^k)
\]

\[
\Delta(x, x^k) = \sum_p \delta_p(x_p, x_p^k) + \sum_c \delta_c(x_c, x_c^k)
\]

\[
\bar{u}_p^k(\cdot) = u_p^k(\cdot) - \delta_p(\cdot, x_p^k)
\]

\[
\bar{h}_c^k(\cdot) = h_c^k(\cdot) - \delta_c(\cdot, x_c^k)
\]

\[
\Delta(x, x) = 0
\]

\[
\delta_p(x_p, x_p) = 0
\]

\[
\delta_c(x_c, x_c) = 0
\]

loss-augmented potentials
Max-margin learning via dual decomposition

\[
\min_w R(w) + \sum_{k=1}^{K} L_G(x^k, z^k; w)
\]

\[
L_G(x^k, z^k; w) = \text{MRF}_G(x^k; \tilde{u}^k, \tilde{h}^k) - \min_x \text{MRF}_G(x; \tilde{u}^k, \tilde{h}^k)
\]

\[
\Delta(x, x^k) = \sum_p \delta_p(x_p, x^k_p) + \sum_c \delta_c(x_c, x^k_c)
\]

\[
\Delta(x, x) = 0
\]

\[
\tilde{u}_p^k(\cdot) = u_p^k(\cdot) - \delta_p(\cdot, x^k_p)
\]

\[
\tilde{h}_c^k(\cdot) = h_c^k(\cdot) - \delta_c(\cdot, x^k_c)
\]

loss-augmented potentials
Max-margin learning via dual decomposition

\[
\min_{w} R(w) + \sum_{k=1}^{K} L_G(x^k, \bar{u}^k, \bar{h}^k; w)
\]

\[
L_G(x^k, \bar{u}^k, \bar{h}^k; w) = \text{MRF}_G(x^k; \bar{u}^k, \bar{h}^k) - \min_{x} \text{MRF}_G(x; \bar{u}^k, \bar{h}^k)
\]

\[
\Delta(x, x^k) = \sum_{p} \delta_p(x_p, x_p^k) + \sum_{c} \delta_c(x_c, x_c^k)
\]

\[
\Delta(x, x) = 0
\]

\[
\bar{u}_p^k(\cdot) = u_p^k(\cdot) - \delta_p(\cdot, x_p^k)
\]

\[
\bar{h}_c^k(\cdot) = h_c^k(\cdot) - \delta_c(\cdot, x_c^k)
\]

loss-augmented potentials
Max-margin learning via dual decomposition

\[
\min_w R(w) + \sum_{k=1}^{K} L_G(x^k, \bar{u}^k, \bar{h}^k; w)
\]

\[
L_G(x^k, \bar{u}^k, \bar{h}^k; w) = \text{MRF}_G(x^k; \bar{u}^k, \bar{h}^k) - \min_x \text{MRF}_G(x; \bar{u}^k, \bar{h}^k)
\]

Problem
Learning objective intractable due to this term
Max-margin learning via dual decomposition

\[
\min_w R(w) + \sum_{k=1}^{K} L_G(x^k, \bar{u}^k, \bar{h}^k; w)
\]

\[
L_G(x^k, \bar{u}^k, \bar{h}^k; w) = \text{MRF}_G(x^k; \bar{u}^k, \bar{h}^k) - \min_x \text{MRF}_G(x; \bar{u}^k, \bar{h}^k)
\]

**Solution:** approximate this term with dual relaxation from decomposition \(\{G_i = (V_i, \mathcal{C}_i)\}_{i=1}^{N}\)

\[
\min_x \text{MRF}_G(x; \bar{u}^k, \bar{h}^k) \approx \text{DUAL}_{\{G_i\}}(\bar{u}^k, \bar{h}^k)
\]
Max-margin learning via dual decomposition

\[
\min_w R(w) + \sum_{k=1}^{K} L_G(x^k, \bar{u}^k, \bar{h}^k; w)
\]

\[
L_G(x^k, \bar{u}^k, \bar{h}^k; w) = \text{MRF}_G(x^k; \bar{u}^k, \bar{h}^k) - \min_x \text{MRF}_G(x; \bar{u}^k, \bar{h}^k)
\]

**Solution:** approximate this term with dual relaxation from decomposition \( \{G_i = (\mathcal{V}_i, \mathcal{C}_i)\}_{i=1}^{N} \)

\[
\min_x \text{MRF}_G(x; \bar{u}^k, \bar{h}^k) \approx \text{DUAL}_{\{G_i\}}(\bar{u}^k, \bar{h}^k)
\]

\[
\text{DUAL}_{\{G_i\}}(\bar{u}^k, \bar{h}^k) = \max_{\{\theta^{(i,k)}\}} \sum_i \text{MRF}_{G_i}(\theta^{(i,k)}, \bar{h}^k)
\]

s.t. \( \sum_{i \in \mathcal{I}_p} \theta^{(i,k)}(\cdot) = \bar{u}^k_p(\cdot) \)
Max-margin learning via dual decomposition

Solution: approximate this term with dual relaxation from decomposition \( \{G_i = (V_i, C_i)\}_{i=1}^{N} \)

\[
\min_{\mathbf{x}} \text{MRF}_{G}(\mathbf{x}; \mathbf{u}^k, \mathbf{h}^k) \approx \text{DUAL}_{\{G_i\}}(\mathbf{u}^k, \mathbf{h}^k)
\]

\[
\text{DUAL}_{\{G_i\}}(\mathbf{u}^k, \mathbf{h}^k) = \max_{\{\theta^{(i,k)}\}} \sum_{i} \text{MRF}_{G_i}(\theta^{(i,k)}, \mathbf{h}^k)
\]

s.t. \( \sum_{i \in I_p} \theta^{(i,k)}(\cdot) = \bar{u}_p^k(\cdot) \)
Max-margin learning via dual decomposition

\[
\min_{w, \{\theta^{(i,k)}\}} \ R(w) + \sum_{k} \sum_{i} L_{G_i}(x^k, \theta^{(i,k)}, \bar{h}^k; w)
\]

s.t. \[ \sum_{i \in I_p} \theta^{(i,k)}(\cdot) = \bar{u}^k_p(\cdot). \]

now

\[
\min_{w} \ R(w) + \sum_{k=1}^{K} L_{G}(x^k, \bar{u}^k, \bar{h}^k; w)
\]

before
Max-margin learning via dual decomposition

\[ \min_{w, \{\theta^{(i,k)}\}} R(w) + \sum_k \sum_i L_{G_i}(x^k, \theta^{(i,k)}, \bar{h}^k; w) \]

\[ \text{s.t. } \sum_{i \in I_p} \theta_p^{(i,k)}(\cdot) = \bar{u}_p^k(\cdot). \]

Now

\[ \min_w R(w) + \sum_{k=1}^K L_G(x^k, \bar{u}^k, \bar{h}^k; w) \]

Before

Essentially, training of complex CRF decomposed to parallel training of easy-to-handle slave CRFs !!!
Max-margin learning via dual decomposition

- Global optimum via projected subgradient method (slight variation of subgradient method)

<table>
<thead>
<tr>
<th>Projected subgradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repeat</td>
</tr>
<tr>
<td>1. compute subgradient at current $w$</td>
</tr>
<tr>
<td>2. update $w$ by taking a step in the negative subgradient direction</td>
</tr>
<tr>
<td>3. project into feasible set</td>
</tr>
<tr>
<td>until convergence</td>
</tr>
</tbody>
</table>
Projected subgradient learning algorithm

- Resulting learning scheme:
  - Very efficient and very flexible
  - Requires from the user only to provide an optimizer for the slave MRFs
  - Slave problems freely chosen by the user
  - Easily adaptable to further exploit special structure of any class of CRFs
Choice of decompositions \( \{G_i\} \)

\[
\mathcal{F}_0 = \text{true loss (intractable)} \\
\mathcal{F}\{G_i\} = \text{loss when using decomposition } \{G_i\}
\]

- \( \mathcal{F}_0 \leq \mathcal{F}\{G_i\} \)  
  (upper bound property)

- \( \{G_i\} < \{\tilde{G}_j\} \)  
  (hierarchy of learning algorithms)
Choice of decompositions \( \{ G_i \} \)

- \( G_{\text{single}} = \{ G_c \}_{c \in \mathcal{C}} \) denotes following decomposition:
  - One slave per clique \( c \in \mathcal{C} \)
  - Corresponding sub-hypergraph \( G_c = (\mathcal{V}_c, \mathcal{C}_c) \):
    \[
    \mathcal{V}_c = \{ p | p \in c \}, \quad \mathcal{C}_c = \{ c \}
    \]

- Resulting slaves often easy (or even trivial) to solve even if global problem is complex and NP-hard
  - leads to widely applicable learning algorithm

- Corresponding dual relaxation is an LP
  - Generalizes well known LP relaxation for pairwise MRFs (at the core of most state-of-the-art methods)
Choice of decompositions $\{G_i\}$

• But we can do better if CRFs have special structure...

• Structure means:
  • More efficient optimizer for slaves (speed)
  • Optimizer that handles more complex slaves (accuracy)

  (Almost all known examples fall in one of above two cases)

• We are essentially adapting decomposition to exploit the structure of the problem at hand
Choice of decompositions $\{G_i\}$

- But we can do better if CRFs have special structure...

- e.g., **pattern-based** high-order potentials (for a clique $c$) [Komodakis & Paragios CVPR09]

$$H_c(x) = \begin{cases} 
\psi_c(x) & \text{if } x \in \mathcal{P} \\
\psi_c^\text{max} & \text{otherwise}
\end{cases}$$

$\mathcal{P}$ subset of $\mathcal{L}^{|c|}$ (its vectors called **patterns**)

Choice of decompositions \( \{G_i\} \)

- Tree decomposition \( G_{\text{tree}} = \{T_i\}_{i=1}^{N} \) 
  \( (T_i \text{ are spanning trees that cover the graph}) \)

- No improvement in accuracy
  \[ \text{DUAL}_{G_{\text{tree}}} = \text{DUAL}_{G_{\text{single}}} \Rightarrow \mathcal{F}_{G_{\text{tree}}} = \mathcal{F}_{G_{\text{single}}} \]

- But improvement in speed
  \( (\text{DUAL}_{G_{\text{tree}}} \text{ converges faster than } \text{DUAL}_{G_{\text{single}}}) \)
Image denoising

- Piecewise constant images

\[ Z \]
\[ X \]

- Potentials:
  \[ u_p^k (x_p) = \left| x_p - z_p \right| \]
  \[ h_{pq}^k (x_p, x_q) = V \left( \left| x_p - x_q \right| \right) \]

- Goal: learn pairwise potential \( V(\cdot) \)
Image denoising

learnt potential

- Pairwise potential
- Primal objective function
- Subgradient
- DLPW
- Our method
Image denoising

![Graphs showing intensity difference and time versus primal objective function and average test error for different methods.]
Stereo matching

- Potentials:
  \[ u_p^k(x_p) = \left| I_{\text{left}}(p) - I_{\text{right}}(p - x_p) \right| \]
  \[ h_{pq}^k(x_p, x_q) = f\left(\left| \nabla I_{\text{left}}(p) \right| \right) \left[ x_p \neq x_q \right] \]

- Goal: learn function \( f(\cdot) \) for gradient-modulated Potts model
Stereo matching

• Potentials: 
  \[ u_p^k(x_p) = \left| I_{\text{left}}(p) - I_{\text{right}}(p - x_p) \right| \]
  \[ h_{pq}^k(x_p, x_q) = f\left(\left| \nabla I_{\text{left}}(p) \right|\right) \left[ x_p \neq x_q \right] \]

• Goal: learn function \( f(\cdot) \) for gradient-modulated Potts model

“Venus” disparity using \( f(\cdot) \) as estimated at different iterations of learning algorithm

[Middlebury dataset]
Stereo matching

• Potentials: 
  \[ u_p^k(x_p) = \left| I^{\text{left}}(p) - I^{\text{right}}(p - x_p) \right| \]
  \[ h_{pq}^k(x_p, x_q) = f \left( \left| \nabla I^{\text{left}}(p) \right| \right) \left[ x_p \neq x_q \right] \]

• Goal: learn function \( f(\cdot) \) for gradient-modulated Potts model

![Middlebury dataset]
Stereo matching

- Potentials:
  \[ u^k_p(x_p) = \left| I_{\text{left}}(p) - I_{\text{right}}(p - x_p) \right| \]
  \[ h^k_{pq}(x_p, x_q) = f \left( \nabla I_{\text{left}}(p) \right) \left[ x_p \neq x_q \right] \]

- Goal: learn function \( f(\cdot) \) for gradient-modulated Potts model
High-order $P^n$ Potts model

Goal: learn high order CRF with potentials given by

$$h_c(x) = \begin{cases} 
\beta^c_i & \text{if } x_p = l, \forall p \in c \\
\beta^c_{\text{max}} & \text{otherwise}
\end{cases}$$

$$\beta^c_i = w_l \cdot z^c_l$$

[Kohli et al. CVPR07]

Cost for optimizing slave CRF: $O(|L|) \Rightarrow$ Fast training

- 100 training samples
- 50x50 grid
- clique size 3x3
- 5 labels ($|L|=5$)
Learning to cluster
Clustering

- A fundamental task in vision and beyond

- Typically formulated as an optimization problem based on a given distance function between datapoints

- Choice of distance crucial for the success of clustering

- **Goal 1**: learn this distance automatically based on training data

- **Goal 2**: learning should also handle the fact that the number of clusters is typically unknown at test time
Exemplar based clustering formulation

\[ \min_{Q \subseteq S} E(Q) = \sum_{p \notin Q} \min_{q \in Q} d_{p,q} + \sum_{q \in Q} d_{q,q} \]

- distance between datapoints \( p \) and \( q \)
- penalty for choosing \( q \) as exemplar (cluster center)

set of exemplars (cluster centers)

set of datapoints

The above formulation allows to:

- automatically estimate the number of clusters (i.e. size of \( Q \))
- use arbitrary distances (e.g., non-metric, asymmetric, non-differentiable)
Exemplar based clustering formulation

\[
\min_{Q \subseteq S} E(Q) = \sum_{p \notin Q} \min_{q \in Q} d_{p,q} + \sum_{q \in Q} d_{q,q}
\]

- Distance between datapoints \( p \) and \( q \)
- Penalty for choosing \( q \) as exemplar (cluster center)

Inference can be performed efficiently using:

Clustering via LP-based Stabilities [Komodakis et al., NIPS 2008]