Learning Conditional Random Field parameters from training data

A review

Przemyslaw Polewski

Photogrammetry and Remote Sensing
Technische Universität München

Department of Geoinformatics
Hochschule München

09.11.2016 / PF-Seminar

Outline

1. Conditional Random Fields
2. Direct methods
   - Probabilistic modeling
   - Grid search
3. Maximum likelihood variations
   - Pseudo-likelihood
   - Marginal approximations
4. Max-margin learning
   - Structured SVM
5. Software libraries

Conditional Random Fields

Definition

- Consider set of objects $O = \{o_1, \ldots, o_n\}$ with features $X = \{x_1, \ldots, x_n\}$ and labels $Y = \{y_1, \ldots, y_n\}$.

Conditional Random Fields

Definition

- Consider set of objects $O = \{o_1, \ldots, o_n\}$ with features $X = \{x_1, \ldots, x_n\}$ and labels $Y = \{y_1, \ldots, y_n\}$.
- Let G be a discrete structure (graph,lattice) with a neighborhood relation $\sim$ on the elements of O

Conditional Random Fields

Definition

- Consider set of objects $O = \{o_1, \ldots, o_n\}$ with features $X = \{x_1, \ldots, x_n\}$ and labels $Y = \{y_1, \ldots, y_n\}$.
- Let G be a discrete structure (graph,lattice) with a neighborhood relation $\sim$ on the elements of O
- CRF: Undirected probabilistic graphical model which describes *conditional* probability (over G) of object labels given their features:

Conditional Random Fields

Definition

- Consider set of objects $O = \{o_1, \ldots, o_n\}$ with features $X = \{x_1, \ldots, x_n\}$ and labels $Y = \{y_1, \ldots, y_n\}$.
- Let G be a discrete structure (graph,lattice) with a neighborhood relation $\sim$ on the elements of O
- CRF: Undirected probabilistic graphical model which describes *conditional* probability (over G) of object labels given their features:

$$P(y|x) = \frac{1}{Z(x)} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y_c|x)] \tag{1}$$

Conditional Random Fields

Definition

- Consider set of objects $O = \{o_1, \dots, o_n\}$ with features $X = \{x_1, \dots, x_n\}$ and labels $Y = \{y_1, \dots, y_n\}$.
- Let G be a discrete structure (graph,lattice) with a neighborhood relation $\sim$ on the elements of O
- CRF: Undirected probabilistic graphical model which describes *conditional* probability (over G) of object labels given their features:

$$P(y|x) = \frac{1}{Z(x)} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y_c|x)] \tag{1}$$

Equivalent energy formulation:

$$E(y) = -\log P(y|x) = \sum_{c \in \mathscr{C}(G)} \psi_c(y_c|x) - \log Z(x) \tag{2}$$

Conditional Random Fields

Notation

$$E(y;\overbrace{\Theta}^{\text{param.}}) = \sum_{c \in \underbrace{\mathscr{C}(G)}_{\text{cliques of G}}} \overbrace{\psi_c(y_c|x;\Theta)}^{\text{clique potentials}} - \log \underbrace{Z(x;\Theta)}_{\text{partition function}}$$

Conditional Random Fields

Notation

$$E(y; \overbrace{\Theta}^{\text{param.}}) = \sum_{c \in \underbrace{\mathscr{C}(G)}_{\text{cliques of G}}} \overbrace{\psi_c(y_c | x; \Theta)}^{\text{clique potentials}} - \log \underbrace{Z(x;\Theta)}_{\text{partition function}}$$

- Θ: vector of parameters $\theta_1, \theta_2, \ldots, \theta_m$
- Z: function of data and parameters, but not of labels

Conditional Random Fields

Notation

$$E(y; \overbrace{\Theta}^{\text{param.}}) = \sum_{c \in \underbrace{\mathscr{C}(G)}_{\text{cliques of G}}} \overbrace{\psi_c(y_c | x; \Theta)}^{\text{clique potentials}} - \log \underbrace{Z(x; \Theta)}_{\text{partition function}}$$

- Θ: vector of parameters $\theta_1, \theta_2, \ldots, \theta_m$
- Z: function of data and parameters, but not of labels

Two complementary tasks:

Conditional Random Fields

Notation

$$E(y;\overbrace{\Theta}^{\text{param.}}) = \sum_{c \in \underbrace{\mathscr{C}(G)}_{\text{cliques of G}}} \overbrace{\psi_c(y_c|x;\Theta)}^{\text{clique potentials}} - \log \underbrace{Z(x;\Theta)}_{\text{partition function}}$$

- Θ: vector of parameters $\theta_1, \theta_2, \ldots, \theta_m$
- Z: function of data and parameters, but not of labels

Two complementary tasks:

- Inference: minimize E w.r.t. **labels** y given parameters

Conditional Random Fields

Notation

$$E(y;\overbrace{\Theta}^{\text{param.}}) = \sum_{c \in \underbrace{\mathscr{C}(G)}_{\text{cliques of G}}} \overbrace{\psi_c(y_c|x;\Theta)}^{\text{clique potentials}} - \log \underbrace{Z(x;\Theta)}_{\text{partition function}}$$

- Θ: vector of parameters $\theta_1, \theta_2, \ldots, \theta_m$
- Z: function of data and parameters, but not of labels

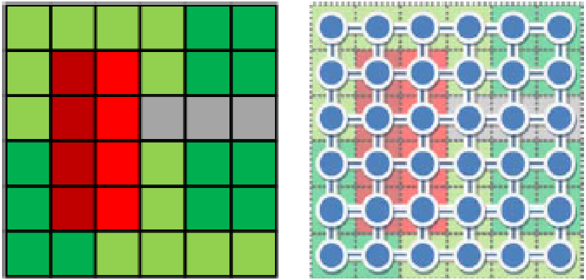
Two complementary tasks:

- Inference: minimize E w.r.t. **labels** y given parameters
- Learning: minimize E w.r.t. **parameters** Θ given training labels

Conditional Random Fields

Neighborhood examples

Raster image

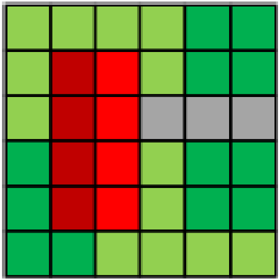


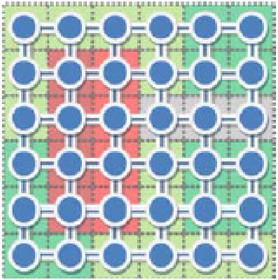
ⓒWegner, 2011

Conditional Random Fields

Neighborhood examples

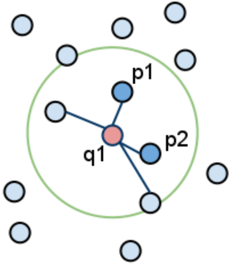
Raster image





©Wegner, 2011

Point cloud



©pointclouds.org

Literal interpretation of potentials

- Treat each clique potential as a classifier yielding a categorical distribution over possible class assignments to clique members

Literal interpretation of potentials

- Treat each clique potential as a classifier yielding a categorical distribution over possible class assignments to clique members
- Each classifier can be trained independently using the user's ML method of choice (e.g. SVM, LR, RF)

Literal interpretation of potentials

- Treat each clique potential as a classifier yielding a categorical distribution over possible class assignments to clique members
- Each classifier can be trained independently using the user's ML method of choice (e.g. SVM, LR, RF)
- Drawback: need a lot of training examples for rare class combinations

Literal interpretation of potentials

- Treat each clique potential as a classifier yielding a categorical distribution over possible class assignments to clique members
- Each classifier can be trained independently using the user's ML method of choice (e.g. SVM, LR, RF)
- Drawback: need a lot of training examples for rare class combinations
- E.g. for unary and binary cliques:

$$E(y) = -\sum_i \log P(y_i|x) - \sum_i \sum_{j \in N(i)} \log P(\underbrace{y_i, y_j}_{\text{for all class pairs}}|x) \tag{3}$$

Example

Classification of urban ALS point clouds [Niemeyer et al., 2014]

Example

Classification of urban ALS point clouds [Niemeyer et al., 2014]

- 7 unbalanced object classes: grassland, road, building with gable roof, low vegetation, façade, building with flat roof, tree
- Unary and binary cliques

Example

Classification of urban ALS point clouds [Niemeyer et al., 2014]

- 7 unbalanced object classes: grassland, road, building with gable roof, low vegetation, façade, building with flat roof, tree
- Unary and binary cliques

©Niemeyer et al. 2014

Example

Classification of urban ALS point clouds [Niemeyer et al., 2014]

- 7 unbalanced object classes: grassland, road, building with gable roof, low vegetation, façade, building with flat roof, tree
- Unary and binary cliques

> "A comparison of three different versions of a CRF-based classifier has shown that Random Forests are well suited for the computation of unary and pairwise potentials needed for CRFs(. . .)"
>
> — Niemeyer et al.

©Niemeyer et al. 2014

Grid search / empirical approach

- Applicable when number of parameters is small and their domain is known a priori better than $(-\infty;\infty)$ (e.g. [0;1])
- Grid search on parameter combinations $\Theta^k \in \mathbf{\Theta}$ with repeated inference:

Grid search / empirical approach

- Applicable when number of parameters is small and their domain is known a priori better than $(-\infty; \infty)$ (e.g. [0;1])
- Grid search on parameter combinations $\Theta^k \in \boldsymbol{\Theta}$ with repeated inference:

---

for $k = 1$ to $|\boldsymbol{\Theta}|$ **do**
    $y^k = \arg\min_y E(y|x; \Theta^k)$
    $e^k = \text{loss}(y^k, y^*)$
end for
$k^* = \arg\min_k e^k$
return Θ^{k^*}

Grid search / empirical approach

- Applicable when number of parameters is small and their domain is known a priori better than $(-\infty; \infty)$ (e.g. [0;1])
- Grid search on parameter combinations $\Theta^k \in \boldsymbol{\Theta}$ with repeated inference:

---

for $k = 1$ to $|\boldsymbol{\Theta}|$ **do**
    $y^k = \arg\min_y E(y|x; \Theta^k)$
    $e^k = \text{loss}(y^k, y^*)$
end for
$k^* = \arg\min_k e^k$
return Θ^{k^*}

Will (almost) never find the true optimal weights

- Discrete grid $\neq$ continuous weights

Empirical approach

Example

Contrast sensitive Potts model for MLS point clouds [Weinmann et al., 2015]

Empirical approach

Example

Contrast sensitive Potts model for MLS point clouds [Weinmann et al., 2015]

- 5 unbalanced object classes: wire, pole/trunk, facade, ground, vegetation
- Unary cliques: learned with Random Forest
- Binary cliques: contrast-sensitive Potts model:

Empirical approach

Example

Contrast sensitive Potts model for MLS point clouds [Weinmann et al., 2015]

- 5 unbalanced object classes: wire, pole/trunk, facade, ground, vegetation
- Unary cliques: learned with Random Forest
- Binary cliques: contrast-sensitive Potts model:

$$\psi(y_i, y_j; x) = [y_i = y_j] \cdot \boldsymbol{w_1} \cdot \frac{N_a}{N_{k_i}} \cdot [\boldsymbol{w_2} + (1 - \boldsymbol{w_2}) \cdot e^{-\frac{d_{ij}(x)^2}{2\sigma^2}}]$$

Empirical approach

Example

Contrast sensitive Potts model for MLS point clouds [Weinmann et al., 2015]

- 5 unbalanced object classes: wire, pole/trunk, facade, ground, vegetation
- Unary cliques: learned with Random Forest
- Binary cliques: contrast-sensitive Potts model:

$$\psi(y_i, y_j; x) = [y_i = y_j] \cdot \boldsymbol{w_1} \cdot \frac{N_a}{N_{k_i}} \cdot [\boldsymbol{w_2} + (1 - \boldsymbol{w_2}) \cdot e^{-\frac{d_{ij}(x)^2}{2\sigma^2}}]$$

"The weight parameters w_1 and w_2 could be set based on (...). Here, they are set to values that were found empirically."

— Weinmann et al.

Learning parameters with maximum likelihood

The problem

Likelihood function:

$$\mathscr{L}(\Theta) = P(y|x;\Theta) = \frac{1}{Z(x,\Theta)} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y_c|x;\Theta)]$$

Learning parameters with maximum likelihood

The problem

Likelihood function:

$$\mathscr{L}(\Theta) = P(y|x;\Theta) = \frac{1}{Z(x,\Theta)} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y_c|x;\Theta)]$$

Partition “constant” is a *function* of parameters !

Learning parameters with maximum likelihood

The problem

Likelihood function:

$$\mathscr{L}(\Theta) = P(y|x;\Theta) = \frac{1}{Z(x,\Theta)} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y_c|x;\Theta)]$$

Partition “constant” is a *function* of parameters !

$$Z(x,\Theta) = \underbrace{\sum_{y'}}_{\text{for all possible labelings}} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y'_c|x;\Theta)]$$

Learning parameters with maximum likelihood

The problem

Likelihood function:

$$\mathscr{L}(\Theta) = P(y|x;\Theta) = \frac{1}{Z(x,\Theta)} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y_c|x;\Theta)]$$

Partition “constant” is a *function* of parameters !

$$Z(x,\Theta) = \underbrace{\sum_{y'}}_{\text{for all possible labelings}} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y'_c|x;\Theta)]$$

Exact computation infeasible even for small training sets and binary labeling. . .

Learning parameters with maximum likelihood

The problem

Likelihood function:

$$\mathscr{L}(\Theta) = P(y|x;\Theta) = \frac{1}{Z(x,\Theta)} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y_c|x;\Theta)]$$

Partition “constant” is a *function* of parameters !

$$Z(x,\Theta) = \underbrace{\sum_{y'}}_{\text{for all possible labelings}} \exp[-\sum_{c \in \mathscr{C}(G)} \psi_c(y'_c|x;\Theta)]$$

Exact computation infeasible even for small training sets and binary labeling. . .

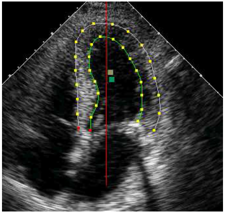
. . . **need an approximation !**

Taming the partition function

Method I

Case study: Heart Motion Abnormality Detection [Schmidt et al., 2008]

- Binary classification, up to pairwise cliques



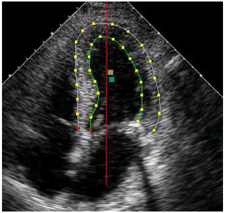
ⓒSchmidt et al. 2008

Taming the partition function

Method I

Case study: Heart Motion Abnormality Detection [Schmidt et al., 2008]

- Binary classification, up to pairwise cliques
- Assumes linear potentials:
  $\psi(y_i|x) = v^T F(x_i, y_i), \psi(y_i, y_j|x) = w^T F(x_{i,j}, y_i, y_j)$



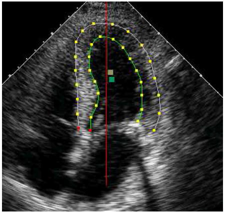
©Schmidt et al. 2008

Taming the partition function

Method I

Case study: Heart Motion Abnormality Detection [Schmidt et al., 2008]

- Binary classification, up to pairwise cliques
- Assumes linear potentials:
  $\psi(y_i|x) = v^T F(x_i, y_i), \psi(y_i, y_j|x) = w^T F(x_{i,j}, y_i, y_j)$
- Compound parameter: $\Theta = (v, w)$, compound features $F(x, y)$



©Schmidt et al. 2008

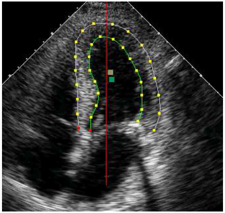
Taming the partition function

Method I

Case study: Heart Motion Abnormality Detection [Schmidt et al., 2008]

- Binary classification, up to pairwise cliques
- Assumes linear potentials:
  $\psi(y_i|x) = v^T F(x_i, y_i), \psi(y_i, y_j|x) = w^T F(x_{i,j}, y_i, y_j)$
- Compound parameter: $\Theta = (v, w)$, compound features $F(x, y)$

$$\ell(\Theta) = \log \mathscr{L}(\Theta) = \Theta^T F(x, y) - \log Z(x, \Theta)$$



ⓒSchmidt et al. 2008

Taming the partition function

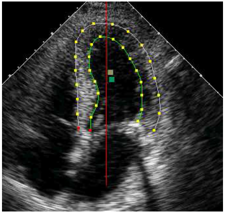
Method I

Case study: Heart Motion Abnormality Detection [Schmidt et al., 2008]

- Binary classification, up to pairwise cliques
- Assumes linear potentials:
  $\psi(y_i|x) = v^T F(x_i, y_i), \psi(y_i, y_j|x) = w^T F(x_{i,j}, y_i, y_j)$
- Compound parameter: $\Theta = (v, w)$, compound features $F(x, y)$

$$\ell(\Theta) = \log \mathscr{L}(\Theta) = \Theta^T F(x,y) - \log Z(x,\Theta)$$

$$\nabla \ell(\Theta) = F(x,y) - \underbrace{\sum_{y'} P(y'|x,\Theta) F(x,y')}_{\text{computationally infeasible}}$$



ⓒSchmidt et al. 2008

Taming the partition function

Method I

Solution: use **pseudo-likelihood** approximation:

Taming the partition function

Method I

Solution: use **pseudo-likelihood** approximation:

Let $Y = (Y_1, Y_2, \ldots, Y_m)$ be a random vector with graph $G = (V, E)$ defining conditional independence structure. Then:

Taming the partition function

Method I

Solution: use **pseudo-likelihood** approximation:
Let $Y = (Y_1, Y_2, \ldots, Y_m)$ be a random vector with graph $G = (V, E)$ defining conditional independence structure. Then:

$$P(y) \approx \prod_i P(y_i | y_k : (v_i, v_k) \in E)$$

Taming the partition function

Method I

Solution: use **pseudo-likelihood** approximation:
Let $Y = (Y_1, Y_2, \ldots, Y_m)$ be a random vector with graph $G = (V, E)$ defining conditional independence structure. Then:

$$P(y) \approx \prod_i P(y_i | y_k : (v_i, v_k) \in E)$$

Translating to our problem:

$$PL(\Theta) = \prod_i P(y_i | x, y_k : o_i \sim o_k ; \Theta)$$

Taming the partition function

Method I

Solution: use **pseudo-likelihood** approximation:
Let $Y = (Y_1, Y_2, \ldots, Y_m)$ be a random vector with graph $G = (V, E)$ defining conditional independence structure. Then:

$$P(y) \approx \prod_i P(y_i | y_k : (v_i, v_k) \in E)$$

Translating to our problem:

$$PL(\Theta) = \prod_i P(y_i | x, y_k : o_i \sim o_k; \Theta) = \prod_i \exp(\Theta^T \overbrace{F_i(x, y)}^{\text{sub-features induced by } \sim_i}) / Z_i$$

Taming the partition function

Method I

Solution: use **pseudo-likelihood** approximation:
Let $Y = (Y_1, Y_2, \ldots, Y_m)$ be a random vector with graph $G = (V, E)$ defining conditional independence structure. Then:

$$P(y) \approx \prod_i P(y_i | y_k : (v_i, v_k) \in E)$$

Translating to our problem:

$$PL(\Theta) = \prod_i P(y_i | x, y_k : o_i \sim o_k; \Theta) = \prod_i \exp(\Theta^T \overbrace{F_i(x, y)}^{\text{sub-features induced by } \sim_i}) / Z_i$$

$$Z_i = \underbrace{\sum_{y_i'}}_{\text{only over current variable}} \exp(\Theta^T F_i(x, y_i'))$$

Taming the partition function

Method II

Instead of approximating entire likelihood, try to estimate the problematic sum [Kumar et al., 2005]. Assume linear potentials ($\Theta = (w, v)$):

Taming the partition function

Method II

Instead of approximating entire likelihood, try to estimate the problematic sum [Kumar et al., 2005]. Assume linear potentials ($\Theta = (w, v)$):

$$\psi(y_i|x) = \log P(y_i|x) = y_i w^T x_i$$

$$\psi(y_i, y_j|x) = \log P(y_i, y_j|x) = y_i y_j v^T x_{i,j}$$

Taming the partition function

Method II

Instead of approximating entire likelihood, try to estimate the problematic sum [Kumar et al., 2005]. Assume linear potentials ($\Theta = (w, v)$):

$$\psi(y_i|x) = \log P(y_i|x) = y_i w^T x_i$$

$$\psi(y_i, y_j|x) = \log P(y_i, y_j|x) = y_i y_j v^T x_{i,j}$$

Under this model, the (log) likelihood gradient is:

$$\frac{\partial \ell}{\partial w} = \frac{1}{2} \sum_i (y_i - < y_i >_{\Theta;x}) x_i$$

Taming the partition function

Method II

Instead of approximating entire likelihood, try to estimate the problematic sum [Kumar et al., 2005]. Assume linear potentials ($\Theta = (w, v)$):

$$\psi(y_i|x) = \log P(y_i|x) = y_i w^T x_i$$

$$\psi(y_i, y_j|x) = \log P(y_i, y_j|x) = y_i y_j v^T x_{i,j}$$

Under this model, the (log) likelihood gradient is:

$$\frac{\partial \ell}{\partial w} = \frac{1}{2} \sum_i (y_i - < y_i >_{\Theta;x}) x_i$$

Need to estimate the expectation:

$$< y_i >_{\Theta;x} = \sum_{y'} P(y'|x;\Theta) \cdot y'_i$$

Taming the partition function

Method II

- Marginal approximation:

Taming the partition function

Method II

- Marginal approximation:
  Obtain estimated marginal probabilities per variable:
  $P_i(y_i|x;\Theta) = \sum_{y_1} \ldots \sum_{y_{i-1}} \sum_{y_{i+1}} \ldots \sum_{y_n} P(y_1, \ldots, y_n|x;\Theta)$
  using loopy belief propagation.

Taming the partition function

Method II

- Marginal approximation:
  Obtain estimated marginal probabilities per variable:
  $P_i(y_i|x;\Theta) = \sum_{y_1} \dots \sum_{y_{i-1}} \sum_{y_{i+1}} \dots \sum_{y_n} P(y_1, \dots, y_n|x;\Theta)$
  using loopy belief propagation. Then, plug marginals into expectation:

$$< y_i >_{\Theta;x} = \sum_{y_i} y_i P_i(y_i|x;\Theta)$$

Taming the partition function

Method II

- Marginal approximation:
  Obtain estimated marginal probabilities per variable:
  $P_i(y_i|x;\Theta) = \sum_{y_1} \dots \sum_{y_{i-1}} \sum_{y_{i+1}} \dots \sum_{y_n} P(y_1, \dots, y_n|x;\Theta)$
  using loopy belief propagation. Then, plug marginals into expectation:

$$< y_i >_{\Theta;x} = \sum_{y_i} y_i P_i(y_i|x;\Theta)$$

- Saddle point approximation:
  Let $\hat{y} = \arg\max_y P(y|x;\Theta)$ be the most probable labeling.

Taming the partition function

Method II

- Marginal approximation:
  Obtain estimated marginal probabilities per variable:
  $P_i(y_i|x;\Theta) = \sum_{y_1} \dots \sum_{y_{i-1}} \sum_{y_{i+1}} \dots \sum_{y_n} P(y_1, \dots, y_n|x;\Theta)$
  using loopy belief propagation. Then, plug marginals into expectation:

$$< y_i >_{\Theta;x} = \sum_{y_i} y_i P_i(y_i|x;\Theta)$$

- Saddle point approximation:
  Let $\hat{y} = \arg\max_y P(y|x;\Theta)$ be the most probable labeling. Assume that the entire 'mass' of the partition function is concentrated at $\hat{y}$:

$$Z(\Theta;x) \approx \exp[\sum_i \psi(\hat{y}_i|x;\Theta) + \sum_i \sum_{j \in N(i)} \psi(\hat{y}_i, \hat{y}_j|x;\Theta)]$$

Taming the partition function

Method II

- Marginal approximation:
  Obtain estimated marginal probabilities per variable:
  $P_i(y_i|x;\Theta) = \sum_{y_1} \ldots \sum_{y_{i-1}} \sum_{y_{i+1}} \ldots \sum_{y_n} P(y_1, \ldots, y_n|x;\Theta)$
  using loopy belief propagation. Then, plug marginals into expectation:

$$< y_i >_{\Theta;x} = \sum_{y_i} y_i P_i(y_i|x;\Theta)$$

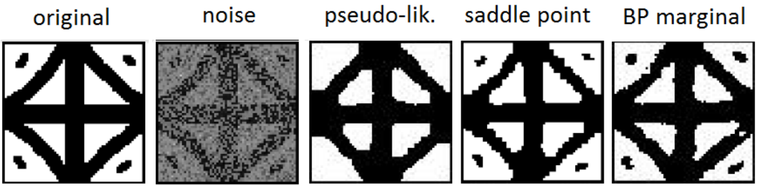
- Saddle point approximation:
  Let $\hat{y} = \arg\max_y P(y|x;\Theta)$ be the most probable labeling. Assume that the entire 'mass' of the partition function is concentrated at $\hat{y}$:

$$Z(\Theta;x) \approx \exp[\sum_i \psi(\hat{y}_i|x;\Theta) + \sum_i \sum_{j\in N(i)} \psi(\hat{y}_i, \hat{y}_j|x;\Theta)]$$

  Then, $< y_i >_{\Theta;x} \approx \hat{y}_i$.

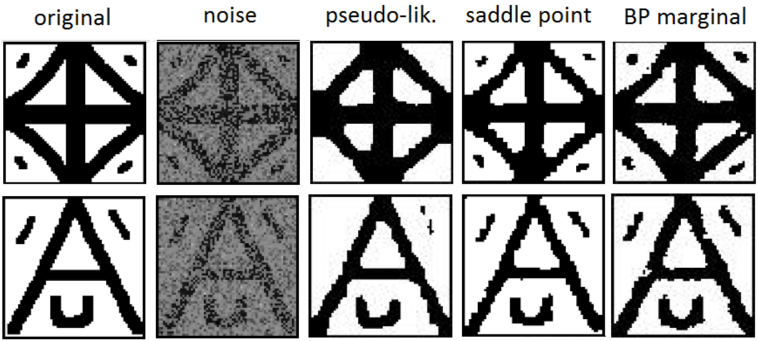
Taming the partition function

Learning/inference results



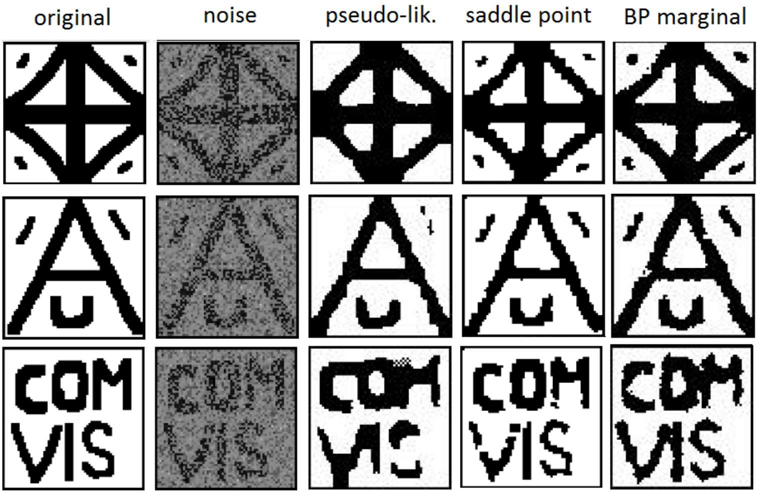
Taming the partition function

Learning/inference results



Taming the partition function

Learning/inference results



©Kumar et al., 2005

Max-margin learning

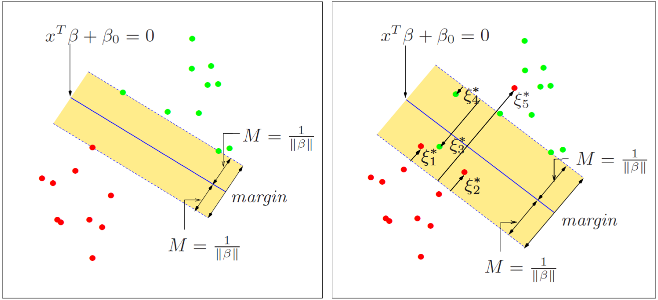
Introduction

- Abandons probabilistic perspective
- Aims at maximizing energy *margin* between object labels
- Uses *linear* potentials, can be kernelized

Max-margin learning

Introduction

- Abandons probabilistic perspective
- Aims at maximizing energy *margin* between object labels
- Uses *linear* potentials, can be kernelized
- Generalizes ideas from support vector machine



Max-margin learning

Basic formulation

Let $\psi(y_i) = w^T F(x_i, y_i), \psi(y_i, y_j) = v^T F(x_{i,j}, y_i, y_j)$ be the potentials.

Max-margin learning

Basic formulation

Let $\psi(y_i) = w^T F(x_i, y_i), \psi(y_i, y_j) = v^T F(x_{i,j}, y_i, y_j)$ be the potentials.

- $\Theta = (w, v)$: compound weight, $F(x, y)$: compound feature
- $\Delta(y, z) \geq 0$: distance between two labelings y and z
- Energy: $E(y|x; \Theta) = \Theta^T F(x, y)$

Max-margin learning

Basic formulation

Let $\psi(y_i) = w^T F(x_i, y_i), \psi(y_i, y_j) = v^T F(x_{i,j}, y_i, y_j)$ be the potentials.

- $\Theta = (w, v)$: compound weight, $F(x, y)$: compound feature
- $\Delta(y, z) \geq 0$: distance between two labelings y and z
- Energy: $E(y|x; \Theta) = \Theta^T F(x, y)$

Main idea: bias the weights Θ so that the energy of training labels is lower than energy of any other configuration *by the maximum margin*

Max-margin learning

Basic formulation

Let $\psi(y_i) = w^T F(x_i, y_i), \psi(y_i, y_j) = v^T F(x_{i,j}, y_i, y_j)$ be the potentials.

- $\Theta = (w, v)$: compound weight, $F(x, y)$: compound feature
- $\Delta(y, z) \geq 0$: distance between two labelings y and z
- Energy: $E(y|x; \Theta) = \Theta^T F(x, y)$

Main idea: bias the weights Θ so that the energy of training labels is lower than energy of any other configuration *by the maximum margin*

Structured SVM [Taskar et al., 2003, Tsochantaridis et al., 2005]

$$
\begin{aligned}
&\text{maximize } \gamma \\
&\qquad \text{s.t. } ||\Theta|| \leq 1 \\
&\qquad\qquad \forall_{y' \neq y} \underbrace{\Theta^T F(x, y')}_{E(y')} - \underbrace{\Theta^T F(x, y)}_{E(y)} \geq \Delta(y, y') \cdot \gamma
\end{aligned}
$$

Max-margin learning

Slack variable formulation

Training set may not be separable - no solution. Need to add slack variables:

Max-margin learning

Slack variable formulation

Training set may not be separable - no solution. Need to add slack variables:

$$\text{minimize } \frac{1}{2}||\Theta||_2^2 + \frac{C}{N}\sum_i \zeta_i$$

$$\text{s.t. } \zeta_i \geq 0$$

$$\forall_{x_i}\forall_{y_i' \neq y_i} \underbrace{\Theta^T F(x_i, y_i')}_{E(y_i')} - \underbrace{\Theta^T F(x_i, y_i)}_{E(y_i)} \geq \Delta(y_i, y_i') - \zeta_i$$

Max-margin learning

Slack variable formulation

Training set may not be separable - no solution. Need to add slack variables:

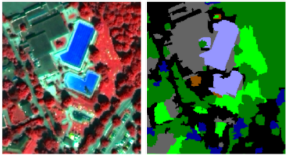
$$\text{minimize } \frac{1}{2}||\Theta||_2^2 + \frac{C}{N}\sum_i \zeta_i$$

$$\text{s.t. } \zeta_i \geq 0$$

$$\forall_{x_i}\forall_{y_i' \neq y_i} \underbrace{\Theta^T F(x_i, y_i')}_{E(y_i')} - \underbrace{\Theta^T F(x_i, y_i)}_{E(y_i)} \geq \Delta(y_i, y_i') - \zeta_i$$

Example: semantic segmentation of urban scenes [Volpi and Ferrari, 2015]

> "In our setting, the potentials employed are linear with respect to the parameter vector.(...)To learn the CRF parameters **w** we adopt the margin rescaling variant of the SSVM(...)"
>
> — Volpi and Ferrari



ⓒVolpi and Ferrari

Max-margin learning

Main challenge

Constraints revisited:

$$\forall_{y' \neq y} \Theta^T F(x, y') - \Theta^T F(x, y) \geq \Delta(y, y') \cdot \gamma$$

Max-margin learning

Main challenge

Constraints revisited:

$$\forall_{y' \neq y} \Theta^T F(x, y') - \Theta^T F(x, y) \geq \Delta(y, y') \cdot \gamma$$

- One constraint for every possible label assignment
- Exponential in input data length ! Infeasible to add all constraints at once

Max-margin learning

Main challenge

Constraints revisited:

$$\forall_{y' \neq y} \Theta^T F(x, y') - \Theta^T F(x, y) \geq \Delta(y, y') \cdot \gamma$$

- One constraint for every possible label assignment
- Exponential in input data length ! Infeasible to add all constraints at once

Possible solution [Tsochantaridis et al., 2005]:

- Add constraints iteratively
- At every step, find most violated constraint
- Re-solve with augmented constraint set

Max-margin learning

Determining most violated constraint

Find the labeling minimizing the loss-augmented energy:

$$\hat{y} = \arg\max_{y'} E(y'|x, \underbrace{\Theta}_{\text{current est.}}) - \Delta(y', y)$$

Max-margin learning

Determining most violated constraint

Find the labeling minimizing the loss-augmented energy:

$$\hat{y} = \arg\max_{y'} E(y'|x, \underbrace{\Theta}_{\text{current est.}}) - \Delta(y', y)$$

Assume Δ decomposes according to nodes: $\Delta(y, y') = \sum_i \delta(y_i, y_i')$. Define augmented unary potentials:

$$\bar{\psi}(y_i') = \psi(y_i') - \delta(y_i, y_i')$$

Max-margin learning

Determining most violated constraint

Find the labeling minimizing the loss-augmented energy:

$$\hat{y} = \arg\max_{y'} E(y'|x, \underbrace{\Theta}_{\text{current est.}}) - \Delta(y', y)$$

Assume Δ decomposes according to nodes: $\Delta(y, y') = \sum_i \delta(y_i, y_i')$. Define augmented unary potentials:

$$\bar{\psi}(y_i') = \psi(y_i') - \delta(y_i, y_i')$$

Perform inference in new network with energy $E(y'; \bar{\psi}(\cdot), \psi(\cdot, \cdot))$ using preferred method (QPBO, linear programming relaxation, simulated annealing,...)

Available software libraries

- UGM: Matlab code for undirected graphical models (`http://www.cs.ubc.ca/~schmidtm/Software/UGM.html`)
- SVM$^{\text{struct}}$: Structured SVM API in Python, C++, Matlab (`http://www.cs.cornell.edu/people/tj/svm_light/svm_struct.html`)
- PyStruct - Structured Learning in Python (`http://pystruct.github.io/`)
- OpenGM: a C++ template library for discrete factor graph models (`http://hciweb2.iwr.uni-heidelberg.de/opengm/`)

Available software libraries

- UGM: Matlab code for undirected graphical models (`http://www.cs.ubc.ca/~schmidtm/Software/UGM.html`)
- SVM[struct]: Structured SVM API in Python, C++, Matlab (`http://www.cs.cornell.edu/people/tj/svm_light/svm_struct.html`)
- PyStruct - Structured Learning in Python (`http://pystruct.github.io/`)
- OpenGM: a C++ template library for discrete factor graph models (`http://hciweb2.iwr.uni-heidelberg.de/opengm/`)

Summary

- Numerous learning methods with varying degrees of complexity and capabilities

Summary

- Numerous learning methods with varying degrees of complexity and capabilities
- Two main paradigms: probabilistic perspective, max-margin learning

Summary

- Numerous learning methods with varying degrees of complexity and capabilities
- Two main paradigms: probabilistic perspective, max-margin learning
- Many specializations have been developed (chain graphs, trees, submodular potentials,. . .)

Summary

- Numerous learning methods with varying degrees of complexity and capabilities
- Two main paradigms: probabilistic perspective, max-margin learning
- Many specializations have been developed (chain graphs, trees, submodular potentials,. . .)
- Direct methods may still be useful

Kumar, S., August, J., and Hebert, M. (2005).
Exploiting Inference for Approximate Parameter Learning in Discriminative Fields: An Empirical Study, pages 153–168.
Springer Berlin Heidelberg, Berlin, Heidelberg.

Niemeyer, J., Rottensteiner, F., and Soergel, U. (2014).
Contextual classification of lidar data and building object detection in urban areas.
ISPRS Journal of Photogrammetry and Remote Sensing, 87:152 – 165.

Schmidt, M., Murphy, K., Fung, G., and Rosales, R. (2008).
Structure learning in random fields for heart motion abnormality detection.
In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8.

Taskar, B., Guestrin, C., and Koller, D. (2003).
Max-margin markov networks.
MIT Press.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005).
Large margin methods for structured and interdependent output variables.
J. Mach. Learn. Res., 6:1453–1484.

Volpi, M. and Ferrari, V. (2015).
Semantic segmentation of urban scenes by learning local class interactions.
In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Weinmann, M., Schmidt, A., Mallet, C., Hinz, S., Rottensteiner, F., and Jutzi, B. (2015).
Contextual classification of point cloud data by exploiting individual 3d neigbourhoods.
ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, II-3/W4:271–278.

The end

Thank you for your attention.