Towards Accurate Structured Output Learning and Prediction

Patrick Pletscher
ETH Zurich
October 25, 2012

pat@pletscher.org
Motivation: Segmentation Problems

Object Recognition in Computer Vision

- Given an image. Task: locate the objects in it.
Motivation: Segmentation Problems

Object Recognition in Computer Vision

- Given an image. Task: locate the objects in it.
- Strong dependencies among labels between closeby pixels. “pixel is likely grass if neighboring pixels are grass”
Motivation: Segmentation Problems

Object Recognition in Computer Vision

- Given an image. Task: locate the objects in it.
- Strong dependencies among labels between closeby pixels. "pixel is likely grass if neighboring pixels are grass"
- Key idea: Impose some structural constraints. Predict labels of the pixels jointly.
Motivation: Segmentation Problems

Object Recognition in Computer Vision

- Given an image. Task: locate the objects in it.
- Strong dependencies among labels between closeby pixels. "pixel is likely grass if neighboring pixels are grass"
- Key idea: Impose some structural constraints. Predict labels of the pixels jointly.
- This thesis: learning & prediction with structured data.
Additional Examples of Structured Data

- Computer Vision: Image denoising or stereo.
- Natural language processing: Parsing or part-of-speech tagging.
- Biology: Protein side-chain prediction and design.
Structured Output Prediction

Setting

- Given observed input variables $x \in \mathcal{X}$; usually $\mathcal{X} = \mathbb{R}^D$.
- Predict a multivariate discrete output variable $y \in \mathcal{Y}$.
- Learning: Find good predictor $f_w(x): \mathcal{X} \rightarrow \mathcal{Y}$. Parameterized by $w$.
- Energy $E(y, x, w)$
  - Cost function to score the different outputs.
  - Models the dependencies.
Structured Output Prediction

Setting

- Given observed input variables $x \in \mathcal{X}$; usually $\mathcal{X} = \mathbb{R}^D$.
- Predict a *multivariate* discrete output variable $y \in \mathcal{Y}$.
- Learning: Find good predictor $f_w(x) : \mathcal{X} \rightarrow \mathcal{Y}$. Parameterized by $w$.
- Energy $E(y, x, w)$
  - Cost function to score the different outputs.
  - Models the dependencies.

Standard Binary Classification

- Binary classification as a special case: $\mathcal{Y} = \{-1, 1\}$.
- Linear prediction function: $f_w(x) = \text{sign}(\langle w, x \rangle)$.
- Energy: $E(y, x, w) = -y \langle w, x \rangle$
**Prediction for an Input $x$**

Choose the best output: $y^* = f_w(x) = \arg\min_{y \in Y} E(y, x, w)$.

Due to dependencies no obvious way how to do this
Structured Models: Why Is It Difficult?

Prediction for an Input $x$

Choose the best output: $y^* = f_w(x) = \arg\min_{y \in \mathcal{Y}} E(y, x, w)$.

Due to dependencies no obvious way how to do this

Exhaustive Enumeration? Here: Object Recognition

- $M$ pixels, $K$ different object classes. $|\mathcal{Y}| = K^M$ possibilities.
- Usually: $K > 10, M \gg 10 \times 10$.
- Compare to: $10^{80}$ atoms in the universe.
- Prediction as a computational problem.

Need for Clever Algorithms and Approximations

- Some problems exactly tractable. But often only approximate.
- Learning the energy: Even harder as prediction a subprocedure.
SOP Overview – Energy

- “Blueprint of a model” $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

- Linear dependence on parameters:
  $$E(y, x, w) = -\langle w, \phi(x, y) \rangle$$

- Explicit way to write the energy:
  $$E(y) = \sum_{i \in \mathcal{V}} \theta_i(y_i) + \sum_{(i, j) \in \mathcal{E}} \theta_{ij}(y_i, y_j)$$

- Construct $\theta$ from $w$ and $x$.
SOP Overview – Loss

- **Loss:** the smaller the better!
  \[ \Delta \left( \begin{array}{c} \text{[Image]} \\ \text{[Image]} \end{array} \right) = 0 \]
  \[ \Delta \left( \begin{array}{c} \text{[Image]} \\ \text{[Image]} \end{array} \right) = 0.5 \]

- **Example:** Missclassified pixels.
  \[ \Delta_{y^*}(y) = \sum_{i \in V} y_i \neq y_i^* \]

- **More complex losses** (wrong direction):
  - Overlap of bounding boxes
  - \( F_1 \) score
  - Area-under-curve
SOP Overview – Learning

- **Goal:** Learn a good energy. *ground-truth has small energy*
- **Learning = Estimation of** $\mathbf{w}^*$.

$$\min_{\mathbf{w}} \frac{\lambda}{2} \| \mathbf{w} \|^2 + \frac{1}{N} \sum_{n=1}^{N} \ell(\mathbf{w}, \mathbf{x}^n, \mathbf{y}^n)$$

1. **Max-margin loss for** $(\mathbf{x}, \mathbf{y}^*)$:

$$E(\mathbf{y}^*, \mathbf{x}, \mathbf{w}) - \min_{\mathbf{y} \in \mathcal{Y}} [E(\mathbf{y}, \mathbf{x}, \mathbf{w}) - \Delta_{\mathbf{y}^*}(\mathbf{y})]$$

2. **Log-loss for** $(\mathbf{x}, \mathbf{y}^*)$:

$$E(\mathbf{y}^*, \mathbf{x}, \mathbf{w}) + \log \sum_{\mathbf{y} \in \mathcal{Y}} \exp(-E(\mathbf{y}, \mathbf{x}, \mathbf{w}))$$

**Diagram:**
- **Learning**
- **Prediction**
- **Evaluation**
- **Train data**
- **Unseen data**
- **Structure**
SOP Overview – Prediction

- Given input $x$ and weights $w^*$.
- Construct potentials: $w^*, x \mapsto \theta$.
- Minimize energy.

$$\min_{y \in \mathcal{Y}} \sum_{i \in \mathcal{V}} \theta_i(y_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j)$$
SOP Overview – Evaluation

Compare prediction $y$ to ground-truth $y^*$ using the loss function.

$$\Delta y^*(y)$$

Learning

Train data

Prediction

Unseen $x$

Evaluation

$y^*$

Happy?
SOP Overview – Tractability

Exact algorithms for loopy graphs only if

1. \( E(y) \) submodular and binary:
   \[
   \theta_{ij}(0, 0) + \theta_{ij}(1, 1) \leq \theta_{ij}(0, 1) + \theta_{ij}(1, 0)
   \]

2. Loss function “easy”.

My PhD Thesis

Learning:
- Pletscher & Kohli, AISTATS 2012.
- Pletscher & Ong, AISTATS 2012.
- Pletscher, Ong & Buhmann, ECML 2010.
- Pletscher, Ong & Buhmann, AISTATS 2009.

Prediction:
- Pletscher & Wulff, ICML 2012.

Evaluation:
- Pletscher, Nowozin, Kohli & Rother, DAGM 2011.
Novel Algorithm for Approximate Energy Minimization

- Compute minimum energy assignment for general pairwise energies:

\[
\min_y E(y) = \min_y \sum_{i \in \mathcal{V}} \theta_i(y_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j).
\]

- Combination of two relaxations:
  - Linear Programming (Schlesinger 1976).
  - Quadratic Programming (Ravikumar and Lafferty 2006).

- Efficient Message Passing Algorithms.
Linear and Quadratic Programming Relaxations

**LP for Marginal Polytope**

\[
\min_{\mu \in \mathcal{M}} \sum_{i \in \mathcal{V}} \theta_i^T \mu_i + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}^T \mu_{ij}
\]

But: \( \mathcal{M} \) is exponentially large!
**Linear and Quadratic Programming Relaxations**

**LP for Marginal Polytope**

\[
\min_{\mu \in \mathcal{M}} \sum_{i \in V} \theta_i^T \mu_i + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}^T \mu_{ij}
\]

But: \(\mathcal{M}\) is exponentially large!

/ outer approximation

**LP for Local Marginal Polytope**

\[
\min_{\mu \in \mathcal{L}_G} \sum_{i \in V} \theta_i^T \mu_i + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}^T \mu_{ij}
\]
Linear and Quadratic Programming Relaxations

LP for Marginal Polytope

\[ \min_{\mu \in \mathcal{M}} \sum_{i \in V} \theta_i^T \mu_i + \sum_{(i, j) \in \mathcal{E}} \theta_{ij}^T \mu_{ij} \]

But: \( \mathcal{M} \) is exponentially large!

LP for Local Marginal Polytope

\[ \min_{\mu \in \mathcal{L}_G} \sum_{i \in V} \theta_i^T \mu_i + \sum_{(i, j) \in \mathcal{E}} \theta_{ij}^T \mu_{ij} \]

Quadratic Programming

\[ \min_{\mu \in \mathcal{L}_G} \sum_{i \in V} \theta_i^T \mu_i + \sum_{(i, j) \in \mathcal{E}} \theta_{ij}^T \mu_{ij} \]

\[ \text{s.t.} \quad \mu_{ij} = \text{vec}(\mu_i \mu_j^T) \quad \forall (i, j) \in \mathcal{E} \]
LPQP: Combine LP and QP relaxations

Joint LP and QP Objective

$$\min_{\mu \in \mathcal{L}} \theta^T \mu + \rho \sum_{(i,j) \in \mathcal{E}} D_{KL}(\mu_{ij}, \mu_i \mu_j^T).$$

encourages consistency

Numerical Solution

- Non-convex KL divergence: use the Concave-Convex Procedure.
- Iteratively solve convex optimization problems.
- Efficient message-passing algorithms.
- Gradual increase of $\rho$. 

Patrick Pletscher Towards Accurate Structured Output Learning and Prediction 9/16
Decision Tree Fields
LPQP Results in Low Energy Solutions

LPQP

SA

51.6 46.4 45.9 43.5 42.5
High-order Loss Functions
Pletscher & Kohli, AISTATS 2012

Exact Max-margin Learning for High-order Losses

• High-order loss $\Delta_{y^*}(y)$: does not factorize into unaries

$$\left| \sum_{i \in \mathcal{V}} y_i - \sum_{i \in \mathcal{V}} y_i^* \right|$$ vs. $$\sum_{i \in \mathcal{V}} y_i \neq y_i^*$$

Label-count loss

Hamming loss

• Maximum margin $\Leftrightarrow$ Energy Minimization:

$$\ell_{mm}(w, x, y^*) = E(y^*, x, w) - \min_{y \in \mathcal{Y}} [E(y, x, w) - \Delta_{y^*}(y)]$$

• We characterize a family of tractable high-order loss functions.
Label-count Loss: Introduce Auxiliary Variable

\[
\min_{y, z \in \{0, 1\}} E(y, x, w) + \left( 2z \left( \sum_{i \in V} y_i^* - \sum_{i \in V} y_i \right) + \sum_{i \in V} y_i - \sum_{i \in V} y_i^* \right)
\]

Label-count loss: pairwise representation

\[ -\Delta_{y^*}^{\text{Count}} (y) \]

Label-count loss

Pairwise graphical model
Label-count Loss: Results

\[ \text{loss ratio} = \frac{\Delta_{y^*}(f_w^{\text{Hamming}}(x))}{\Delta_{y^*}(f_w^{\text{Count}}(x))} \]

Label-count loss evaluation

4-connected vs 8-connected
Summary & Future Work

Summary

• Work in most aspects of structured output prediction and learning.
• Goal: fast and accurate approximations for training and prediction.

Future Work

• Smooth-max for direct loss minimization.
• Max-margin learning on a relaxed polytope. Use similar ideas as in LPQP for enforcing consistency.
• Applications. Currently working with ImageNet, a 2 TB dataset.
Thanks To My Collaborators

thanks also to:
Committee, ETHZ ML Groups (JB+AK), Friends/Family
Discussion
Lacoste-Julien, Simon et al. (2012). “Stochastic Block-Coordinate Frank-Wolfe Optimization for Structural SVMs”.
References II


