Structured Prediction Models

Juho Rousu

Seminar on Predicting Structured Data, March 13, 2008

Structured prediction in general

Given

- Input space $\mathcal{X}$, output space $\mathcal{Y}$, both arbitrary objects
- Sample $\{(x_i, y_i)\}_{i=1}^{m}$, $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ drawn according to some unknown joint distribution D
- Loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y}$, $\mathcal{L}(y, y')$ gives the loss incurred in predicting y' when y was correct.
- A model class M consisting of models $f : \mathcal{X} \mapsto \mathcal{Y}$

We wish to learn a model $f \in M$ that minimizes the expected loss

$$E_{(x,y)\in D}\mathcal{L}(f(x), y)$$

Structured prediction with linear models

We consider models that

- Map inputs and output to a joint inner product space

$$\varphi : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{F}_{XY}$$

- Assume a the form of a linear score function

$$F(x, y, w) = \langle w, \varphi(x, y) \rangle$$

- Predict $y* = f(x)$ by solving the *preimage problem*

$$f(x) = \mathbf{argmax}_{y \in \mathcal{Y}} F(x, y, w)$$

Learning criteria

Two approaches considered for learning w:

1. "Unsupervised" formulation: Try to maximize the minimum score of training data

   $$w* = \mathbf{argmax}_w \min_{i=1}^{m} F(x_i, y_i, w),$$

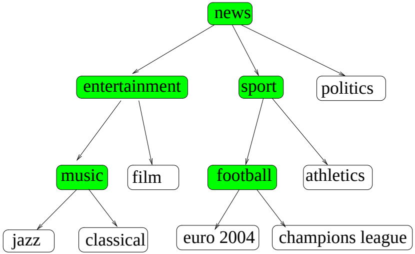
   i.e. try to separate training data from the origin with maximum margin (MMR, Szedmak et al. 2005)
2. "Supervised" formulation: Try to separate correct input-output pairs (x_i, y_i) from the incorrect ones (x_i, y) with maximum margin (Collins 2002; Taskar, 2003; Tsochantaridis, 2004)

   $$w* = \mathbf{argmax}_w \min_{i=1}^{m} \min_{y \neq y_i} F(x_i, y_i, w) - F(x_i, y, w).$$

We will concentrate on the latter in this talk.

Running example: Hierarchical Multilabel Classification

Goal: Given document x, and hierachy $T = (V, E)$, predict multilabel $\mathbf{y} \in \{+1, -1\}^k$ where the positive microlabels y_i form a union of partial paths in T




BBC News | ENTERTAINMENT | Football pundit accuses Posh

Front Page
World
UK
UK Politics
Business
Sci/Tech
Health
Education
Sport
Entertainment
New Music
Releases
Talking Point
In Depth
AudioVideo

Saturday, 8 January, 2000, 15:02 GMT

Football pundit accuses Posh

David and Victoria Beckham are permanently in the public eye

BBC football pundit Mark Lawrenson has accused David Beckham and his pop star wife Victoria of "courting publicity".

SPECIAL REPORT
World Club Championships

Lawrenson, an analyst on BBC1's Football Focus, spoke out during a discussion about Beckham's sending off in Thursday's World Club Championship match.

Joint feature maps & kernels

Two general approaches for creating a joint feature map:

- Tensor product of (global) input ($\phi(x)$) and output feature maps ($\psi(y)$):

$$\varphi(x, y) = \phi(x) \otimes \psi(y) = (\phi_k(x)\psi_l(y))_{k,l} \,,$$

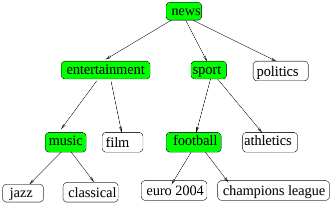
  consists all product features between the input and output
- Linear combination of local features:

$$\varphi(x, y) = \sum_l \phi_l(x) \otimes \psi_l(y),$$

  where l enumerates the components of the structure and $\phi_l(x)$ and $\psi_l(y)$ are input and output features relevant to the l'th component.
  - Assumes that input and output structures are already perfectly aligned, this is true in tasks such as sequence annotation.

Tensor product example: hierarchical document classification

- $\phi(x)$ is the bag of words of the document
- $\psi(y)$ is the vector of edge-label indicators: $\psi_{e,u}(y) = 1$ iff edge e is labeled u.
- $\varphi(x, y)$ contains all word/edge-labeling co-occurrences (or counts thereof) in example (x, y)



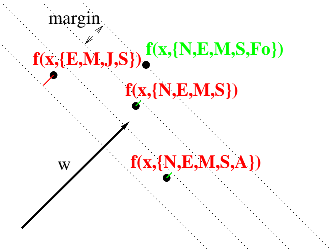
Tensor product features: statistical machine translation (SMT)

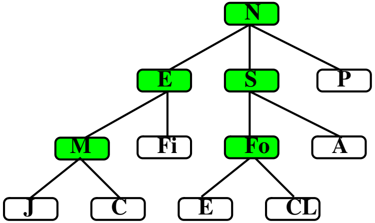
- $\phi(x)$ is the bag of phrases of the source sentence
- $\psi(y)$ is the bag of phrases of the target sentence
- $\varphi(x, y)$ contains all phrase co-occurrences (or counts thereof) in example (x, y)

(in SMT jargon, this is a kind of phrase book without alignment information)

Structured prediction framework (Taskar et al., 2004; Tsochantaridis et al., 2005)

- Map pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ into a joint feature space via $\varphi : \mathcal{X} \times \mathcal{Y} \mapsto \mathcal{F}_{xy}$
- Learn a weigth vector $\mathbf{w}$ to separate the correct $\mathbf{y}_i$ from the incorrect ones $\mathbf{y}'$ by a large margin.
- Prediction: given x, predict $\hat{y} = \mathbf{argmax}_y w^T \varphi(x, y)$





Optimization problem: primal form

$$\min_{\mathbf{w},\boldsymbol{\xi}\geq 0} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{m}\xi_i$$

$$\text{s.t. } \mathbf{w}^T(\varphi(\mathbf{x}_i, \mathbf{y}_i) - \varphi(\mathbf{x}_i, \mathbf{y})) \geq \ell(\mathbf{y}_i, \mathbf{y}) - \xi_i, \forall i, \mathbf{y} \in \{+1, -1\}^k$$

- Minimization of the norm $||\mathbf{w}||$ corresponds to maximizing the margin $\lambda = 1/||\mathbf{w}||$
- Margin scaling by the loss $\ell(\mathbf{y}_i, \mathbf{y})$: the more incorrect the output y, the larger the required margin
- Huge constraint set: one constraint per *pseudoexample* $(x_i, \mathbf{y}), i = 1, \ldots, m, y \in \mathcal{Y}$
- Cannot be solved by off-the-shelf QP solvers

Loss functions for hierarchies

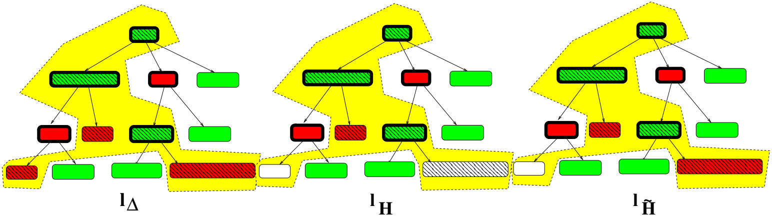
Consider vector-valued true output $\mathbf{y} = (y_1, \ldots, y_k) \in \{+1, -1\}^k$, and a predicted one $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_k)$. Many choices:

- **Zero-one loss**: $\ell_{0/1}(\mathbf{y}, \hat{\mathbf{y}}) = 1_{\{\mathbf{y} \neq \hat{\mathbf{y}}\}}$; treats all incorrect outputs alike $\implies$ not good, we would like to penalize very bad predictions more than almost correct ones
- **Hamming loss**: $\ell_{\Delta}(\mathbf{y}, \hat{\mathbf{y}}) = \sum_j 1_{\{y_j \neq \hat{y}_j\}}$; counts incorrectly predicted components $\implies$ better, but does not take the hierarchical structure of y_j's into account.

Loss functions for hierarchies

For hierarchies, we can construct two loss functions that take the hierarchy into account, yet allow us penalize basd predictions more than almost correct ones:

- **Edge loss**: assign loss on the edges of the hierarchy: $\ell_{\tilde{H}}(\mathbf{y}, \hat{\mathbf{y}}) = \sum_j c_j 1_{\{y_j \neq \hat{y}_j \text{ \& } y_{parent(j)} = \hat{y}_{parent(j)}\}}$; mistake in the child is penalized if the parent was correct.
- **Path loss** (Cesa-Bianchi et al. 2004): $\ell_H(\mathbf{y}, \hat{\mathbf{y}}) = \sum_j c_j 1_{\{y_j \neq \hat{y}_j \text{ \& } y_k = \hat{y}_k \forall k \in ancestors(j)\}}$; the first mistake along a path from root to leaf is penalized



Optimization problem: primal form

$$\min_{\mathbf{w},\boldsymbol{\xi}\geq 0} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{m}\xi_i$$
$$\text{s.t. } \mathbf{w}^T\left(\varphi(\mathbf{x}_i,\mathbf{y}_i) - \varphi(\mathbf{x}_i,\mathbf{y})\right) \geq \ell(\mathbf{y}_i,\mathbf{y}) - \xi_i, \forall i, \mathbf{y} \in \{+1,-1\}^k$$

- Minimization of the norm $||\mathbf{w}||$ corresponds to maximizing the margin $\lambda = 1/\,||\mathbf{w}||$
- Margin scaling by the loss $\ell(\mathbf{y}_i,\mathbf{y})$: the more incorrect the output y, the larger the required margin
- Huge constraint set: one constraint per *pseudoexample* $(x_i,\mathbf{y}), i = 1,\ldots,m, y \in \mathcal{Y}$
- Cannot be solved by off-the-shelf QP solvers

Optimization problem: dual form

The Lagrangian dual is given by

$$\max_{\boldsymbol{\alpha}>0} \sum_{i,\mathbf{y}} \alpha(x_i,\mathbf{y})\ell(\mathbf{y}_i,\mathbf{y}) - \frac{1}{2}\sum_{x_i,\mathbf{y}}\sum_{x_i',\mathbf{y}'} \alpha(x_i,\mathbf{y})^T K(x_i,\mathbf{y};x_i',\mathbf{y}')\alpha(x_i',\mathbf{y}')$$

$$\text{s.t.} \sum_{\mathbf{y}} \alpha(x_i,\mathbf{y}) \leq C, \forall i$$

- Joint kernel $K(x_i,\mathbf{y};x_j,\mathbf{y}') = \Delta\varphi(x_i,\mathbf{y})^T\Delta\varphi(x_j,\mathbf{y}')$, where $\Delta\varphi(x_i,\mathbf{y}) = \varphi(x_i,\mathbf{y}_i) - \varphi(x_j,\mathbf{y})$
- Many approaches to make the optimization tractable (Altun et al. 2003,Taskar et al, 2004)
- We will look at marginalization methods that will shrink the size of the QP to polynomial size in the dimension of the output space.

Marginalized dual problem

- The joint feature map $\phi(x) \otimes \psi(y)$ can be written as $(\phi(x) \otimes \psi_e(y))_{e \in E}$
- Joint kernel $K(x_i, \mathbf{y}; x_j, \mathbf{y}') = \Delta\varphi(x_i, \mathbf{y})^T \Delta\varphi(x_j, \mathbf{y}')$ decomposes by the edges

$$K(x_i, y; x_j, y') = \sum_e K_x(x_i, x_j)(\psi_e(\mathbf{y}_i) - \psi_e(\mathbf{y}))^T(\psi_e(\mathbf{y}_j) - \psi_e(\mathbf{y}'))$$

- The edge loss also decomposes similarly

$$\ell_{\tilde{H}}(\mathbf{y}, \hat{\mathbf{y}}) = \sum_e 1_{\{y_{child(e)} \neq \hat{y}_{child(e)} \ \& \ y_{parent(e)} = \hat{y}_{parent(e)}\}}$$

, where *child*(e) and *parent*(e) denote the nodes in the two ends of an edge

Marginalized dual problem

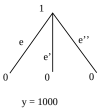
- The dual variables have edge-marginals, denoting the sum of dual variables $\alpha(x, \mathbf{y})$ where $\mathbf{y}$ has labeling u on edge e:

$$\mu_e(x, u) = \sum_{\mathbf{y}|u=\mathbf{y}_e} \alpha(x, \mathbf{y})$$

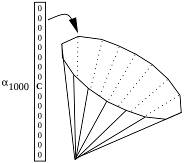
- Collecting all the equations in a matrix: $\boldsymbol{\mu} = M\boldsymbol{\alpha}$
- The feasible set of the marginalized problem is given by

$$\mathcal{M} = \{\mu | \exists \boldsymbol{\alpha} \in \mathcal{A} : \boldsymbol{\mu} = M\boldsymbol{\alpha}\},$$

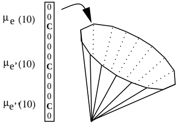
where $\mathcal{A}$ is the feasible set of the original dual problem



Hierarchy T



Dual polytope



Marginal dual polytope of T

Marginalized problem

$$\max_{\boldsymbol{\mu}\in\mathcal{M}} \sum_{e\in E} \boldsymbol{\mu}_e^T \boldsymbol{\ell}_e - \frac{1}{2}\sum_{e\in E} \boldsymbol{\mu}_e^T K_e \boldsymbol{\mu}_e$$

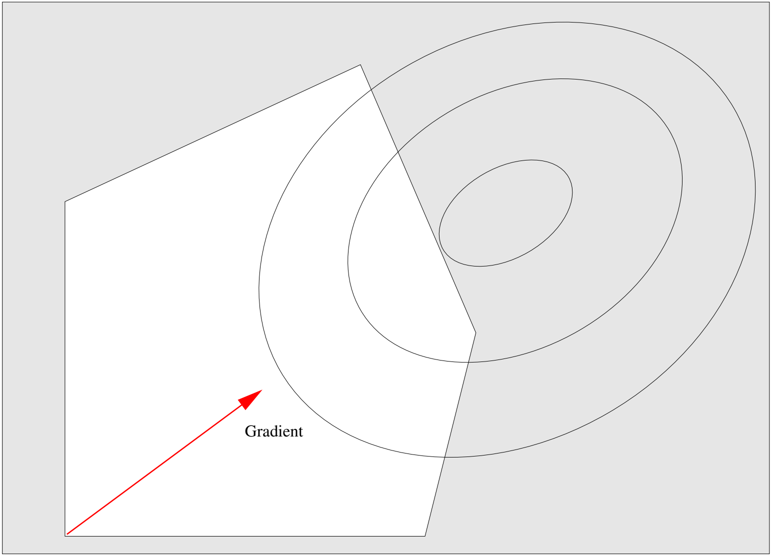
- The problem has now polynomial number of marginal dual variables $O(m|E|)$
- However, the constraints are now expressed in implicit form $\mu \in \mathcal{M}$
- Writing the constraint set out explicitly would make the problem again too large (although polynomial size)
- We deal with this problem after first looking at the solution algorithm

Conditional Gradient method

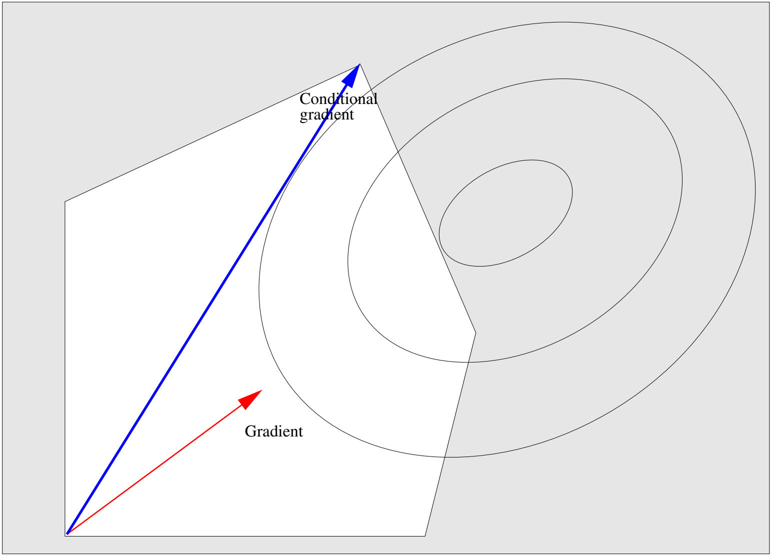
Conditional Gradient Descent (c.f. Bertsekas, 1999) can be used to optimize the marginalized dual problem Ingredients:

- Iterative gradient search in the feasible set
- Update direction is the highest feasible point assuming current gradient; found by solving a constrained linear program: $\max_{\boldsymbol{\mu} \in \mathcal{M}} (\boldsymbol{\ell} - K\boldsymbol{\mu}_0)^T \boldsymbol{\mu}$
- updates within single-example subspaces can be done independently, after obtaining an initial gradient.

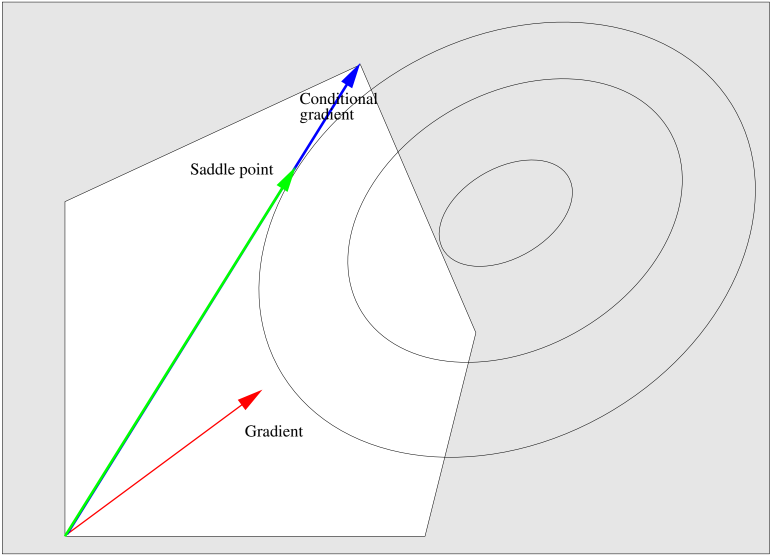
Conditional Gradient Algorithm



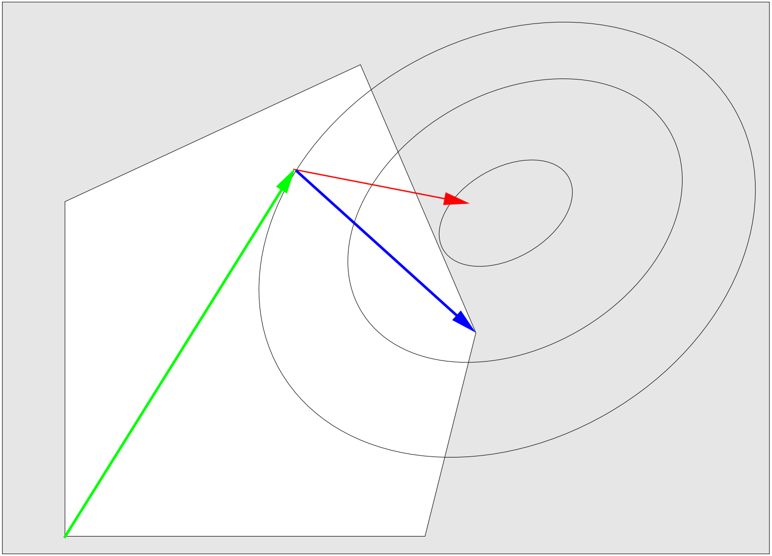
Conditional Gradient Algorithm



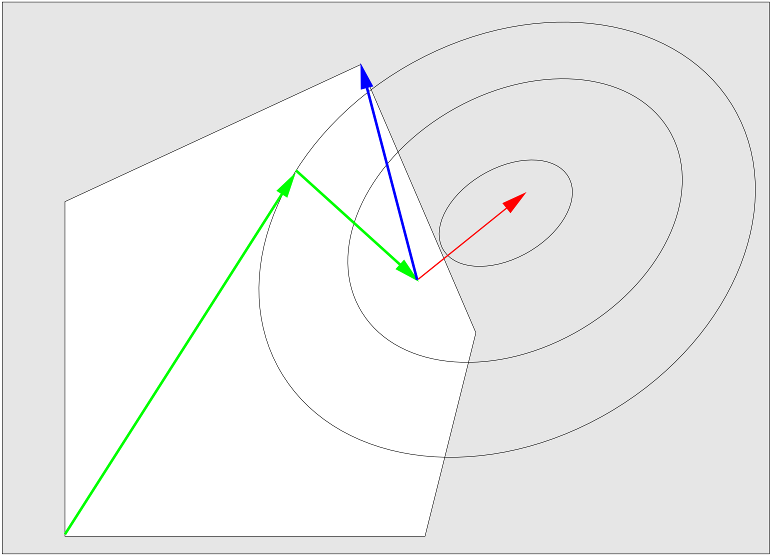
Conditional Gradient Algorithm



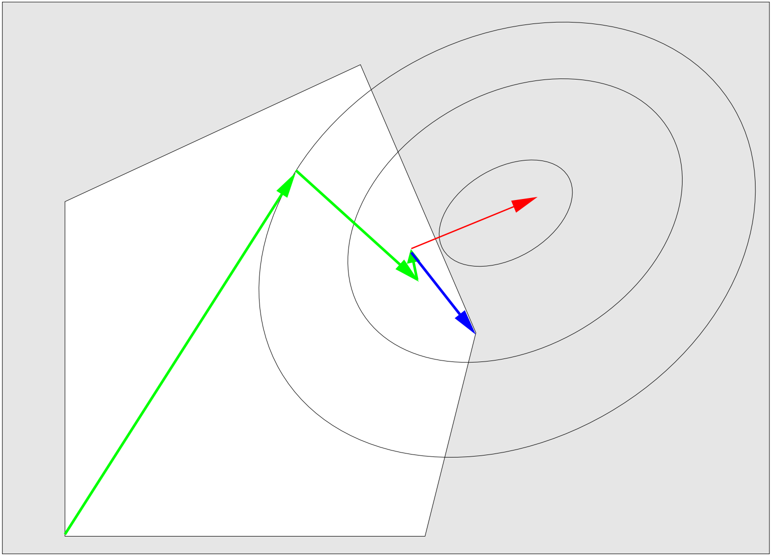
Conditional Gradient Algorithm



Conditional Gradient Algorithm

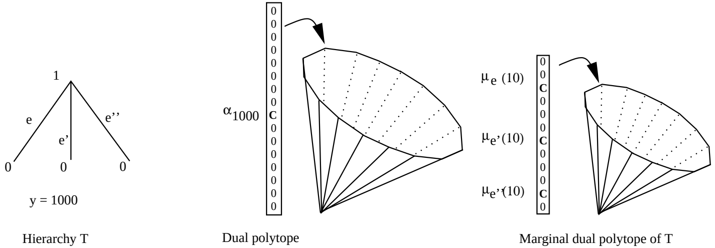


Conditional Gradient Algorithm



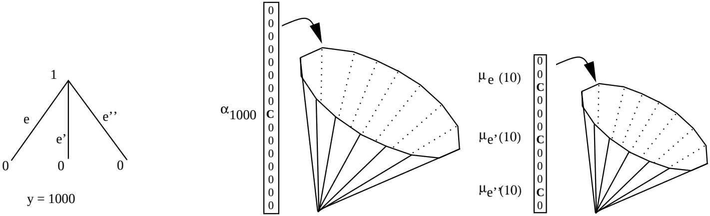
Finding update directions efficiently

- Solving the update direction $\max_{\boldsymbol{\mu}\in\mathcal{F}}(\boldsymbol{\ell} - K\boldsymbol{\mu}_0)^T\boldsymbol{\mu}$ with an LP solver will constitute a bottleneck for scalability
- To find a better method, we need to look at the relationship of the original dual (in terms of αs) and the marginalized problem (in terms of μ_e's)



Finding update directions efficiently

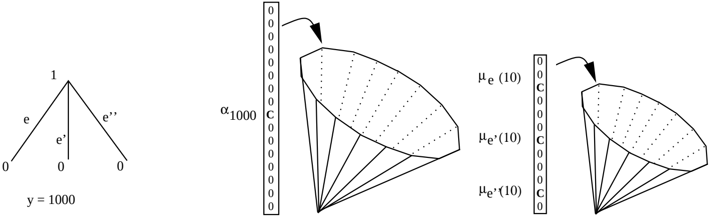
- Consider the marginal dual variables of a single example i: $\mu_e(i, \mathbf{y}_e) = \sum_{\mathbf{y}} 1_{\{\mathbf{y}|\mathbf{u}_e = \mathbf{y}_e\}} \alpha(i, \mathbf{y})$, denote $M = \left(1_{\{\mathbf{y}|\mathbf{u}_e = \mathbf{y}_e\}}\right)_{(e, \mathbf{u}_e), \mathbf{y}}$
- α's and μ's are tied by $M\boldsymbol{\alpha} = \boldsymbol{\mu}$, for each $\boldsymbol{\alpha}$ we have unique $\boldsymbol{\mu}$
- In particular, if $\boldsymbol{\alpha}$ is a vertex of the dual feasible set, $\boldsymbol{\mu} = M\boldsymbol{\alpha}$ is a vertex on the marginal polytope



Hierarchy T   Dual polytope   Marginal dual polytope of T

Finding update directions efficiently

- If $\boldsymbol{\alpha} \neq 0$ is a vertex, it has a single non-zero $\alpha(i, \mathbf{y}^*)$.
- The marginal image of this vector $\boldsymbol{\mu}(\mathbf{y}^*) = M\boldsymbol{\alpha}$ is a vertex
- To find the conditional gradient $\mathbf{argmax}_{\boldsymbol{\mu} \in \mathcal{F}} (\boldsymbol{\ell} - K\boldsymbol{\mu}_0)^T \boldsymbol{\mu}$ we can instead look for $\mathbf{argmax}_{\mathbf{y}} (\boldsymbol{\ell} - K\boldsymbol{\mu}_0)^T \boldsymbol{\mu}(\mathbf{y})$
- This is a inference problem on the hierachy! Can be solved in linear time using dynamic programming.



Hierarchy T

Dual polytope

Marginal dual polytope of T

Experiments

Datasets:

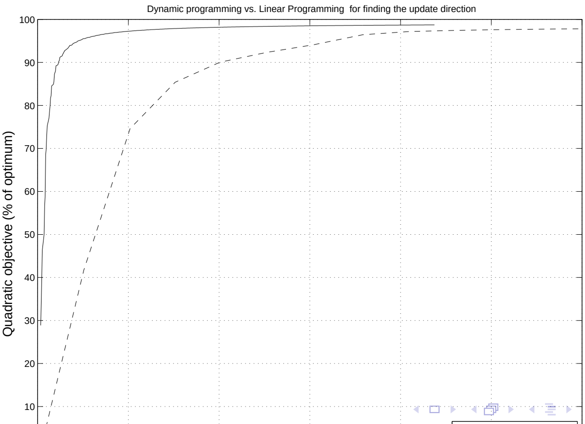
- Reuters Corpus Volume 1 ('CCAT' family), 34 microlabels, maximum tree depth 3, bag-of-words with TFIDF wieghting, 2500 documents were used for training and 5000 for testing.
- WIPO-alpha patent dataset (D section), 188 microlabels, maximum tree depth 4, 1372 documents for training, 358 for testing.

Algorithms:

- Our algorithm: H-M^3 ('Hierarchical Maximum Margin Markov')
- Comparison: Flat SVM, hierarchically trained SVM, hierarchical regularized least squares algorithm (Cesa-Bianchi et al. 2004)
- Implementation in MATLAB 7, LIPSOL solver used in the gradient ascent
- Tests run on a high-end Pentium PC with 1GB RAM

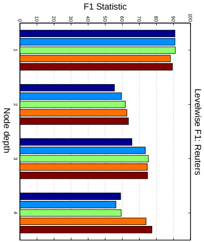
Optimization efficiency

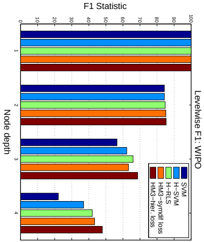
Optimization efficiency on WIPO dataset (1372 training examples, 188 nodes in the hierarchy) on a 3GHZ Pentium 4, 1GB main memory
LP = update directions via linear programming DP = update directions via dynamic programmming inference



Prediction accuracy: Levelwise F1

F1 statistics computed for each node depth separately for Reuters (left) and WIPO (right)





Flat SVM is poor in recalling deep nodes, $\text{H-M}^3\text{-}\ell_{\tilde{H}}$ is the best prediction method in the leaves.

References

Rousu, J., Saunders, C., Szedmak, S., Shawe-Taylor, J. (2006). Kernel-Based Learning of Hierarchical Multilabel Classification Models. JMLR 7, pp. 1601–1626

Sandor Szedmak, John Shawe-Taylor and Emilio Parado-Hernandez (2005). Learning via Linear Operators: Maximum Margin Regression Technical Report. PASCAL, Southampton, UK, Southampton, UK.

Taskar, B., Guestrin, C. and Koller, D. Max-Margin Markov Networks (2003). NIPS'2003

Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. ICML'2004.