Polyhedral Outer Approximations with Application to Natural Language Parsing

André F. T. Martins[1,2] Noah A. Smith[1] Eric P. Xing[1]

[1]Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA

[2]Instituto de Telecomunicações
Instituto Superior Técnico
Lisboa, Portugal

ICML, Montréal, Québec, June 17th, 2009

In a Nutshell

- Structured prediction: models interdependence among outputs
  [Lafferty et al., 2001, Taskar et al., 2003, Tsochantaridis et al., 2004]

In a Nutshell

- Structured prediction: models interdependence among outputs
  [Lafferty et al., 2001, Taskar et al., 2003, Tsochantaridis et al., 2004]
- Exact inference only tractable w/ strong locality assumptions

In a Nutshell

- Structured prediction: models interdependence among outputs
  [Lafferty et al., 2001, Taskar et al., 2003, Tsochantaridis et al., 2004]
- Exact inference only tractable w/ strong locality assumptions
  - Often: better (non-local) models with approximate inference

In a Nutshell

- Structured prediction: models interdependence among outputs
  [Lafferty et al., 2001, Taskar et al., 2003, Tsochantaridis et al., 2004]
- Exact inference only tractable w/ strong locality assumptions
  - Often: better (non-local) models with approximate inference
  - Sometimes outputs are globally constrained (matchings, permutations, spanning trees)

In a Nutshell

- Structured prediction: models interdependence among outputs
  [Lafferty et al., 2001, Taskar et al., 2003, Tsochantaridis et al., 2004]
- Exact inference only tractable w/ strong locality assumptions
  - Often: better (non-local) models with approximate inference
  - Sometimes outputs are globally constrained (matchings, permutations, spanning trees)
- How does approximate inference affect learning?
  [Kulesza and Pereira, 2007, Finley and Joachims, 2008]

In a Nutshell

- Structured prediction: models interdependence among outputs
  [Lafferty et al., 2001, Taskar et al., 2003, Tsochantaridis et al., 2004]
- Exact inference only tractable w/ strong locality assumptions
  - Often: better (non-local) models with approximate inference
  - Sometimes outputs are globally constrained (matchings, permutations, spanning trees)
- How does approximate inference affect learning?
  [Kulesza and Pereira, 2007, Finley and Joachims, 2008]
- This paper: LP-relaxed inference and max-margin learning
  - Guarantees for algorithmic separability
  - New interpretation: balancing accuracy and computational cost
  - Learning bounds via polyhedral characterizations
- Application: dependency parsing with rich features

Example: Dependency Parsing

- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$

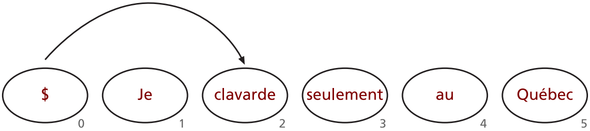
Example: Dependency Parsing

- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$

- Dependency trees: a syntactic representation that captures lexical relationships

Example: Dependency Parsing

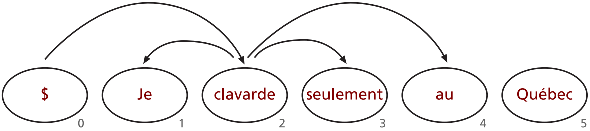
- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$



- Dependency trees: a syntactic representation that captures lexical relationships

Example: Dependency Parsing

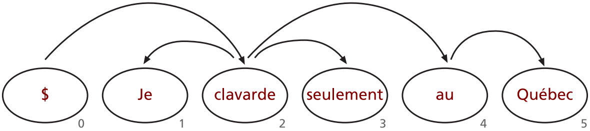
- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$



- Dependency trees: a syntactic representation that captures lexical relationships

Example: Dependency Parsing

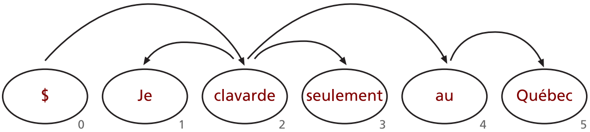
- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$



- Dependency trees: a syntactic representation that captures lexical relationships

Example: Dependency Parsing

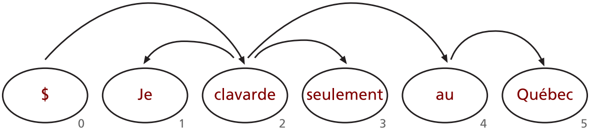
- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$



- Dependency trees: a syntactic representation that captures lexical relationships
- Let $\mathcal{Y}(x)$ be the set of legal dependency trees of x

Example: Dependency Parsing

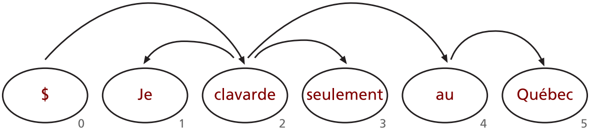
- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$



- Dependency trees: a syntactic representation that captures lexical relationships
- Let $\mathcal{Y}(x)$ be the set of legal dependency trees of x
  - Each $y \in \mathcal{Y}(x)$ is a spanning tree of the complete digraph linking all word pairs

Example: Dependency Parsing

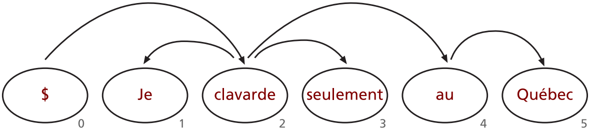
- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$



- Dependency trees: a syntactic representation that captures lexical relationships
- Let $\mathcal{Y}(x)$ be the set of legal dependency trees of x
  - Each $y \in \mathcal{Y}(x)$ is a spanning tree of the complete digraph linking all word pairs
- We want to learn a parser $h : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{Y} = \bigcup_{x \in \mathcal{X}} \mathcal{Y}(x)$

Example: Dependency Parsing

- Let x be a sentence in $\mathcal{X} \triangleq \Sigma^*$



- Dependency trees: a syntactic representation that captures lexical relationships
- Let $\mathcal{Y}(x)$ be the set of legal dependency trees of x
  - Each $y \in \mathcal{Y}(x)$ is a spanning tree of the complete digraph linking all word pairs
- We want to learn a parser $h : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{Y} = \bigcup_{x \in \mathcal{X}} \mathcal{Y}(x)$
- This is a structured classification problem involving non-local interactions among output variables

Outline

Outline

Notation

- Input set $\mathcal{X}$

Notation

- Input set $\mathcal{X}$
- Output set $\mathcal{Y}$

Notation

- Input set $\mathcal{X}$
- Output set $\mathcal{Y}$
- Labeled dataset $\mathcal{L} \triangleq \{(x_1, y_1), \ldots, (x_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$
    - drawn i.i.d. from $P(X, Y)$

Notation

- Input set $\mathcal{X}$
- Output set $\mathcal{Y}$
- Labeled dataset $\mathcal{L} \triangleq \{(x_1, y_1), \ldots, (x_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$
  - drawn i.i.d. from $P(X, Y)$
- Loss function $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_+$

Notation

- Input set $\mathcal{X}$
- Output set $\mathcal{Y}$
- Labeled dataset $\mathcal{L} \triangleq \{(x_1, y_1), \ldots, (x_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$
  - drawn i.i.d. from $P(X, Y)$
- Loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$
- Goal: learn $h : \mathcal{X} \to \mathcal{Y}$ with small expected loss $\mathbb{E}\ell(h(X); Y)$

Notation

- Input set $\mathcal{X}$
- Output set $\mathcal{Y}$
- Labeled dataset $\mathcal{L} \triangleq \{(x_1, y_1), \ldots, (x_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$
  - drawn i.i.d. from $P(X, Y)$
- Loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$
- Goal: learn $h : \mathcal{X} \to \mathcal{Y}$ with small expected loss $\mathbb{E}\ell(h(X); Y)$
- Here: linear classifiers

$$h_{\mathbf{w}}(x) = \arg\max_{y \in \mathcal{Y}} \mathbf{w}^\top \mathbf{f}(x, y)$$

- Hypothesis space $\mathcal{H} \triangleq \{h_{\mathbf{w}} \mid \mathbf{w} \in \mathcal{W}\}$, $\mathcal{W} \subseteq \mathbb{R}^d$ convex

Decomposition Into Parts

- Assumption: each $y \in \mathcal{Y}$ decomposes into parts

Decomposition Into Parts

- Assumption: each $y \in \mathcal{Y}$ decomposes into parts
  - Example: clique assignments in a Markov network
  - Example: arcs in a dependency parse tree
  - Example: k-tuples of arcs in a dependency tree (up to some k)

Decomposition Into Parts

- Assumption: each $y \in \mathcal{Y}$ decomposes into parts
  - Example: clique assignments in a Markov network
  - Example: arcs in a dependency parse tree
  - Example: k-tuples of arcs in a dependency tree (up to some k)
- Define a set of parts $\mathcal{R}$
- Replace y by an indicator vector $\mathbf{z} \triangleq (z_r)_{r \in \mathcal{R}}$ with $z_r \triangleq \mathbb{I}(r \in y)$

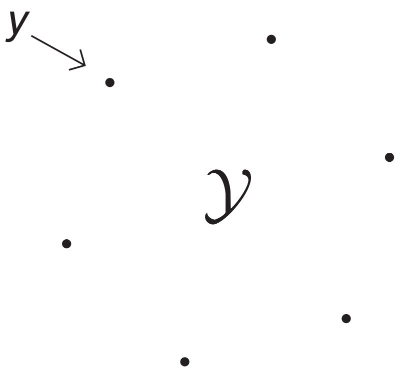
Decomposition Into Parts

- Assumption: each $y \in \mathcal{Y}$ decomposes into parts
  - Example: clique assignments in a Markov network
  - Example: arcs in a dependency parse tree
  - Example: k-tuples of arcs in a dependency tree (up to some k)
- Define a set of parts $\mathcal{R}$
- Replace y by an indicator vector $\mathbf{z} \triangleq (z_r)_{r \in \mathcal{R}}$ with $z_r \triangleq \mathbb{I}(r \in y)$
- Assumption: features decompose over the parts

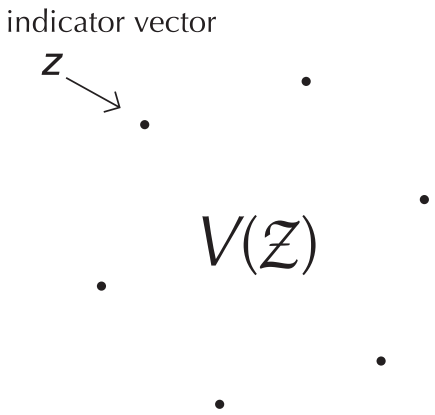
$$\mathbf{f}(x, y) \triangleq \sum_{r \in y} \mathbf{f}_r(x) = \sum_{r \in \mathcal{R}} z_r \mathbf{f}_r(x) = \mathbf{F}(x)\mathbf{z},$$

- $\mathbf{F}(x) \triangleq (\mathbf{f}_r(x))_{r \in \mathcal{R}}$ is a feature matrix (d-by-$|\mathcal{R}|$)

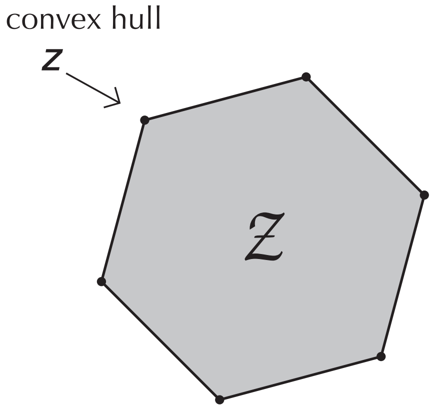
From the Output Set to a Polytope



From the Output Set to a Polytope



From the Output Set to a Polytope



Inference

- Minkowski-Weyl theorem: there is a representation

$$\mathcal{Z} = \{\mathbf{z} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{z} \leq \mathbf{b}\}$$

where $\mathbf{A}$ is a p-by-n matrix and $\mathbf{b}$ is a vector in $\mathbb{R}^p$ ($p, n \in \mathbb{N}$)

Inference

- Minkowski-Weyl theorem: there is a representation

$$\mathcal{Z} = \{\mathbf{z} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{z} \leq \mathbf{b}\}$$

  where $\mathbf{A}$ is a p-by-n matrix and $\mathbf{b}$ is a vector in $\mathbb{R}^p$ $(p, n \in \mathbb{N})$

- Inference becomes an LP [Taskar et al., 2004]:

$$\begin{aligned} \max_{y \in \mathcal{Y}} \mathbf{w}^\top \mathbf{f}(x, y) &= \max_{\mathbf{z} \in V(\mathcal{Z})} \mathbf{w}^\top \mathbf{F}(x)\mathbf{z} \\ &= \max_{\mathbf{z} \in \mathcal{Z}} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w} \end{aligned}$$

Inference

- Minkowski-Weyl theorem: there is a representation

$$\mathcal{Z} = \{\mathbf{z} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{z} \leq \mathbf{b}\}$$

  where $\mathbf{A}$ is a p-by-n matrix and $\mathbf{b}$ is a vector in $\mathbb{R}^p$ $(p, n \in \mathbb{N})$

- Inference becomes an LP [Taskar et al., 2004]:

$$\begin{aligned} \max_{y \in \mathcal{Y}} \mathbf{w}^\top \mathbf{f}(x, y) &= \max_{\mathbf{z} \in V(\mathcal{Z})} \mathbf{w}^\top \mathbf{F}(x)\mathbf{z} \\ &= \max_{\mathbf{z} \in \mathcal{Z}} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w} \end{aligned}$$

- Are we done?

Inference

- Minkowski-Weyl theorem: there is a representation

$$\mathcal{Z} = \{\mathbf{z} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{z} \leq \mathbf{b}\}$$

where $\mathbf{A}$ is a p-by-n matrix and $\mathbf{b}$ is a vector in $\mathbb{R}^p$ $(p, n \in \mathbb{N})$

- Inference becomes an LP [Taskar et al., 2004]:

$$\begin{aligned} \max_{y \in \mathcal{Y}} \mathbf{w}^\top \mathbf{f}(x, y) &= \max_{\mathbf{z} \in V(\mathcal{Z})} \mathbf{w}^\top \mathbf{F}(x)\mathbf{z} \\ &= \max_{\mathbf{z} \in \mathcal{Z}} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w} \end{aligned}$$

- Are we done? No: Finding $\mathbf{A}$ and $\mathbf{b}$ is problem dependent.

Inference

- Minkowski-Weyl theorem: there is a representation

$$\mathcal{Z} = \{\mathbf{z} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{z} \leq \mathbf{b}\}$$

where $\mathbf{A}$ is a p-by-n matrix and $\mathbf{b}$ is a vector in $\mathbb{R}^p$ $(p, n \in \mathbb{N})$

- Inference becomes an LP [Taskar et al., 2004]:

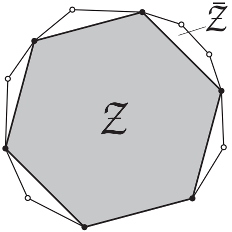
$$\begin{aligned} \max_{y \in \mathcal{Y}} \mathbf{w}^\top \mathbf{f}(x, y) &= \max_{\mathbf{z} \in V(\mathcal{Z})} \mathbf{w}^\top \mathbf{F}(x)\mathbf{z} \\ &= \max_{\mathbf{z} \in \mathcal{Z}} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w} \end{aligned}$$

- Are we done? No: Finding $\mathbf{A}$ and $\mathbf{b}$ is problem dependent.
- In general $p = O(\exp(n))$ (exponentially many constraints)

LP-Relaxed Inference and Outer Polytope

- Often: a concise representation of an outer polytope $\bar{\mathcal{Z}} \supseteq \mathcal{Z}$ such that $\bar{\mathcal{Z}} \cap \mathbb{Z}^n = V(\mathcal{Z})$

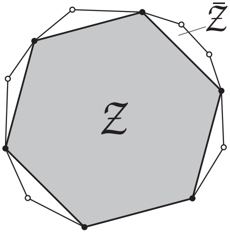
$$\max_{\mathbf{z} \in \mathcal{Z}} \mathbf{s}^\top \mathbf{z} = \max_{\mathbf{z} \in \bar{\mathcal{Z}}, \mathbf{z} \in \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z}$$



LP-Relaxed Inference and Outer Polytope

- Often: a concise representation of an outer polytope $\bar{\mathcal{Z}} \supseteq \mathcal{Z}$ such that $\bar{\mathcal{Z}} \cap \mathbb{Z}^n = V(\mathcal{Z})$

$$\begin{aligned} \max_{\mathbf{z} \in \mathcal{Z}} \mathbf{s}^\top \mathbf{z} &= \max_{\mathbf{z} \in \bar{\mathcal{Z}}, \mathbf{z} \in \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z} \\ &\leq \max_{\mathbf{z} \in \bar{\mathcal{Z}}} \mathbf{s}^\top \mathbf{z} \end{aligned}$$



Learning

- What about learning?

Learning

- What about learning?
- Assumption: The loss function *also* decomposes over the parts

Learning

- What about learning?
- Assumption: The loss function *also* decomposes over the parts
- Example: Hamming loss

$$\begin{aligned}\ell(y'; y) &\triangleq \sum_{r \in \mathcal{R}} \left(\mathbb{I}(r \in y')\mathbb{I}(r \notin y) + \mathbb{I}(r \notin y')\mathbb{I}(r \in y)\right) \\ &= \|\mathbf{z}' - \mathbf{z}\|_1 \\ &= \mathbf{p}^\top \mathbf{z}' + q \quad \text{where } \mathbf{p} \triangleq \mathbf{1} - 2\mathbf{z} \text{ and } q \triangleq \mathbf{1}^\top \mathbf{z}\end{aligned}$$

- Hamming loss is an affine function of $\mathbf{z}$

Learning

- Structured SVM:

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{t=1}^{m} r_t(\mathbf{w})$$

where the slack $r_t(\mathbf{w})$ is the solution of the loss-augmented inference (LAI) problem

$$r_t(\mathbf{w}) = \max_{y_t' \in \mathcal{Y}} \mathbf{w}^\top \underbrace{\mathbf{f}(x_t, y_t')}_{\mathbf{F}_t \mathbf{z}_t'} - \mathbf{w}^\top \underbrace{\mathbf{f}(x_t, y_t)}_{\mathbf{F}_t \mathbf{z}_t} + \underbrace{\ell(y_t'; y_t)}_{\mathbf{p}_t \mathbf{z}_t' + q_t}$$

Learning

- Structured SVM:

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{t=1}^{m} r_t(\mathbf{w})$$

where the slack $r_t(\mathbf{w})$ is the solution of the loss-augmented inference (LAI) problem

$$\begin{aligned} r_t(\mathbf{w}) &= \max_{y'_t \in \mathcal{Y}} \mathbf{w}^\top \underbrace{\mathbf{f}(x_t, y'_t)}_{\mathbf{F}_t \mathbf{z}'_t} - \mathbf{w}^\top \underbrace{\mathbf{f}(x_t, y_t)}_{\mathbf{F}_t \mathbf{z}_t} + \underbrace{\ell(y'_t; y_t)}_{\mathbf{p}_t \mathbf{z}'_t + q_t} \\ &= \left(\max_{\mathbf{z}'_t \in \mathcal{Z}} (\mathbf{F}_t^\top \mathbf{w} + \mathbf{p}_t)^\top \mathbf{z}'_t \right) - (\mathbf{F}_t^\top \mathbf{w})^\top \mathbf{z}_t + q_t \end{aligned}$$

- Also an LP.

Learning

- Structured SVM:

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{t=1}^{m} r_t(\mathbf{w})$$

where the slack $r_t(\mathbf{w})$ is the solution of the loss-augmented inference (LAI) problem

$$\begin{aligned} r_t(\mathbf{w}) &= \max_{y'_t \in \mathcal{Y}} \mathbf{w}^\top \underbrace{\mathbf{f}(x_t, y'_t)}_{\mathbf{F}_t \mathbf{z}'_t} - \mathbf{w}^\top \underbrace{\mathbf{f}(x_t, y_t)}_{\mathbf{F}_t \mathbf{z}_t} + \underbrace{\ell(y'_t; y_t)}_{\mathbf{p}_t \mathbf{z}'_t + q_t} \\ &= \left(\max_{\mathbf{z}'_t \in \mathcal{Z}} (\mathbf{F}_t^\top \mathbf{w} + \mathbf{p}_t)^\top \mathbf{z}'_t \right) - (\mathbf{F}_t^\top \mathbf{w})^\top \mathbf{z}_t + q_t \end{aligned}$$

- Also an LP.

Outline

Exact and Relaxed Structured SVMs

- Structured SVM (exact):

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{t=1}^{m} r_t(\mathbf{w})$$

where the slack $r_t(\mathbf{w})$ is the solution of the exact LAI problem

$$r_t(\mathbf{w}) \quad = \quad \left(\max_{\mathbf{z}'_t \in \mathcal{Z}} (\mathbf{F}_t^\top \mathbf{w} + \mathbf{p}_t)^\top \mathbf{z}'_t\right) - (\mathbf{F}_t^\top \mathbf{w})^\top \mathbf{z}_t + q_t$$

Exact and Relaxed Structured SVMs

- Structured SVM (exact):

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{t=1}^{m} r_t(\mathbf{w})$$

where the slack $r_t(\mathbf{w})$ is the solution of the exact LAI problem

$$r_t(\mathbf{w}) \quad = \quad \left(\max_{\mathbf{z}'_t \in \mathcal{Z}} (\mathbf{F}_t^\top \mathbf{w} + \mathbf{p}_t)^\top \mathbf{z}'_t\right) - (\mathbf{F}_t^\top \mathbf{w})^\top \mathbf{z}_t + q_t$$

- Relax.

Exact and Relaxed Structured SVMs

- Structured SVM (relaxed):

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{t=1}^{m} \bar{r}_t(\mathbf{w})$$

where the slack $\bar{r}_t(\mathbf{w})$ is the solution of the relaxed LAI problem

$$\bar{r}_t(\mathbf{w}) = \left(\max_{\mathbf{z}'_t \in \bar{\mathcal{Z}}} (\mathbf{F}_t^\top \mathbf{w} + \mathbf{p}_t)^\top \mathbf{z}'_t\right) - (\mathbf{F}_t^\top \mathbf{w})^\top \mathbf{z}_t + q_t$$

Exact and Relaxed Structured SVMs

- Structured SVM (relaxed):

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{t=1}^{m} \bar{r}_t(\mathbf{w})$$

where the slack $\bar{r}_t(\mathbf{w})$ is the solution of the relaxed LAI problem

$$\begin{aligned} \bar{r}_t(\mathbf{w}) &= \left(\max_{\mathbf{z}'_t \in \bar{\mathcal{Z}}} (\mathbf{F}_t^\top \mathbf{w} + \mathbf{p}_t)^\top \mathbf{z}'_t \right) - (\mathbf{F}_t^\top \mathbf{w})^\top \mathbf{z}_t + q_t \\ &\geq r_t(\mathbf{w}) \quad \text{upper bounds the exact slack} \end{aligned}$$

Exact and Relaxed Structured SVMs

- Structured SVM (relaxed):

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{t=1}^{m} \bar{r}_t(\mathbf{w})$$

where the slack $\bar{r}_t(\mathbf{w})$ is the solution of the relaxed LAI problem

$$\begin{aligned} \bar{r}_t(\mathbf{w}) &= \left(\max_{\mathbf{z}'_t \in \bar{\mathcal{Z}}} (\mathbf{F}_t^\top \mathbf{w} + \mathbf{p}_t)^\top \mathbf{z}'_t \right) - (\mathbf{F}_t^\top \mathbf{w})^\top \mathbf{z}_t + q_t \\ &\geq r_t(\mathbf{w}) \quad \text{upper bounds the exact slack} \\ &\geq \ell(h_{\mathbf{w}}(x_t), \mathbf{z}_t) \quad \text{upper bounds the true loss.} \end{aligned}$$

Algorithmic Separability

- LP relaxed inference augments the output space: makes up artificial negative examples

Algorithmic Separability

- LP relaxed inference augments the output space: makes up artificial negative examples
- Equivalently: an approximate algorithm $\mathcal{A}_{\mathbf{w}}$ which sometimes returns fractional solutions

Algorithmic Separability

- LP relaxed inference augments the output space: makes up artificial negative examples
- Equivalently: an approximate algorithm $\mathcal{A}_{\mathbf{w}}$ which sometimes returns fractional solutions
- Some definitions [Kulesza and Pereira, 2007]
  - $\mathcal{L}$ is separable if $\exists \mathbf{w}$ s.t. $h_{\mathbf{w}}$ classifies all data correctly
  - $\mathcal{L}$ is alg. separable if $\exists \mathbf{w}$ s.t. $\mathcal{A}_{\mathbf{w}}$ classifies all data correctly

Algorithmic Separability

- LP relaxed inference augments the output space: makes up artificial negative examples
- Equivalently: an approximate algorithm $\mathcal{A}_{\mathbf{w}}$ which sometimes returns fractional solutions
- Some definitions [Kulesza and Pereira, 2007]
  - $\mathcal{L}$ is separable if $\exists \mathbf{w}$ s.t. $h_{\mathbf{w}}$ classifies all data correctly
  - $\mathcal{L}$ is alg. separable if $\exists \mathbf{w}$ s.t. $\mathcal{A}_{\mathbf{w}}$ classifies all data correctly
- Remark: $\mathcal{L}$ algorithmically separable $\Longrightarrow$ $\mathcal{L}$ separable

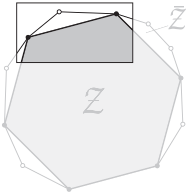
Algorithmic Separability

- LP relaxed inference augments the output space: makes up artificial negative examples
- Equivalently: an approximate algorithm $\mathcal{A}_{\mathbf{w}}$ which sometimes returns fractional solutions
- Some definitions [Kulesza and Pereira, 2007]
  - $\mathcal{L}$ is separable if $\exists \mathbf{w}$ s.t. $h_{\mathbf{w}}$ classifies all data correctly
  - $\mathcal{L}$ is alg. separable if $\exists \mathbf{w}$ s.t. $\mathcal{A}_{\mathbf{w}}$ classifies all data correctly
- Remark: $\mathcal{L}$ algorithmically separable $\implies$ $\mathcal{L}$ separable
- Margin of separation: Minimal γ s.t. $\forall (x_t, y_t) \in \mathcal{L}, y'_t \in \mathcal{Y}(x_t)$:

$$\mathbf{w}^\top \mathbf{f}(x_t, y_t) \geq \mathbf{w}^\top \mathbf{f}(x_t, y'_t) + \gamma \ell(y_t, y'_t) \quad \text{with } \|\mathbf{w}\| = 1.$$

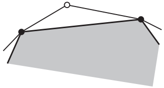
A Sufficient Condition for Algorithmic Separability

- Zooming around a fractional vertex of the outer polytope:



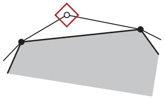
A Sufficient Condition for Algorithmic Separability

- Zooming around a fractional vertex of the outer polytope:



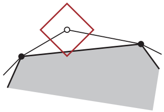
A Sufficient Condition for Algorithmic Separability

- Let L be the radius of the largest loss ball centered at a fractional vertex which does not contain any integer vertex



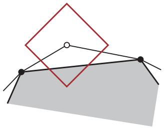
A Sufficient Condition for Algorithmic Separability

- Let L be the radius of the largest loss ball centered at a fractional vertex which does not contain any integer vertex



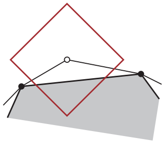
A Sufficient Condition for Algorithmic Separability

- Let L be the radius of the largest loss ball centered at a fractional vertex which does not contain any integer vertex



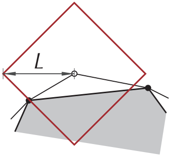
A Sufficient Condition for Algorithmic Separability

- Let L be the radius of the largest loss ball centered at a fractional vertex which does not contain any integer vertex



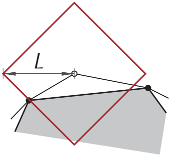
A Sufficient Condition for Algorithmic Separability

- Let L be the radius of the largest loss ball centered at a fractional vertex which does not contain any integer vertex



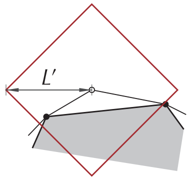
A Sufficient Condition for Algorithmic Separability

- Let L' ($\geq L$) be the radius of the largest loss ball centered at a fractional vertex which contains at most one integer vertex



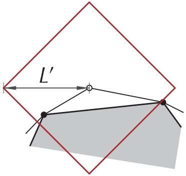
A Sufficient Condition for Algorithmic Separability

- Let L' ($\geq L$) be the radius of the largest loss ball centered at a fractional vertex which contains at most one integer vertex



A Sufficient Condition for Algorithmic Separability

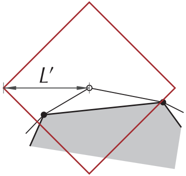
- Let L' ($\geq L$) be the radius of the largest loss ball centered at a fractional vertex which contains at most one integer vertex



  - Assume binary-valued features
  - Let N_f be the maximum number of active features per part

A Sufficient Condition for Algorithmic Separability

- Let L' ($\geq L$) be the radius of the largest loss ball centered at a fractional vertex which contains at most one integer vertex



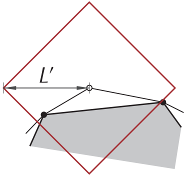
  - Assume binary-valued features
  - Let N_f be the maximum number of active features per part

Proposition

If $\mathcal{L}$ is separable with $\gamma \geq L'\sqrt{N_f}$ then it is algorithmically separable.

A Sufficient Condition for Algorithmic Separability

- Let L' ($\geq L$) be the radius of the largest loss ball centered at a fractional vertex which contains at most one integer vertex



  - Assume binary-valued features
  - Let N_f be the maximum number of active features per part

Proposition

If $\mathcal{L}$ is separable with $\gamma \geq L'\sqrt{N_f}$ then it is algorithmically separable.

  - In the paper: bounds for the nonseparable case
  - Also: bounds for ϵ-approximate algorithms

Balancing Accuracy and Runtime

- Typical goal: minimize expected loss $\mathbb{E}\ell(h(X), Y)$

Balancing Accuracy and Runtime

- Typical goal: minimize expected loss $\mathbb{E}\ell(h(X), Y)$
- Structured prediction: computational cost is also important
- Let $\ell_c(h, x)$ be the cost of computing $h(x)$

Balancing Accuracy and Runtime

- Typical goal: minimize expected loss $\mathbb{E}\ell(h(X), Y)$
- Structured prediction: computational cost is also important
- Let $\ell_c(h, x)$ be the cost of computing $h(x)$
- Let $\mathbb{E}\ell_c(h, X)$ be the average computational cost of h

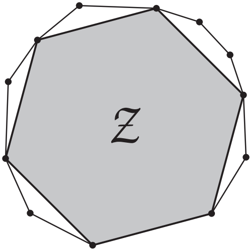
Balancing Accuracy and Runtime

- Typical goal: minimize expected loss $\mathbb{E}\ell(h(X), Y)$
- Structured prediction: computational cost is also important
- Let $\ell_c(h, x)$ be the cost of computing $h(x)$
- Let $\mathbb{E}\ell_c(h, X)$ be the average computational cost of h
- Alternative goal: minimize $\underbrace{\mathbb{E}\ell(h(X), Y)}_{\text{expected loss of } h} + \eta \cdot \underbrace{\mathbb{E}\ell_c(h, X)}_{\text{average cost of } h}$

Balancing Accuracy and Runtime

- Recall that exact inference in our setting is cast as an ILP:

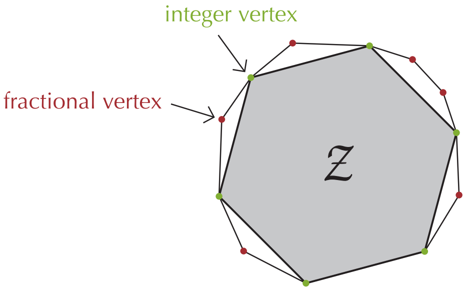
$$\max_{\mathbf{z} \in \bar{\mathcal{Z}} \cap \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w},$$



Balancing Accuracy and Runtime

- Recall that exact inference in our setting is cast as an ILP:

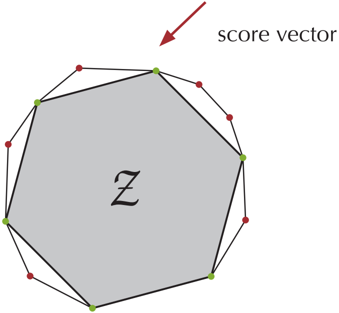
$$\max_{\mathbf{z} \in \bar{\mathcal{Z}} \cap \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w},$$



Balancing Accuracy and Runtime

- Recall that exact inference in our setting is cast as an ILP:

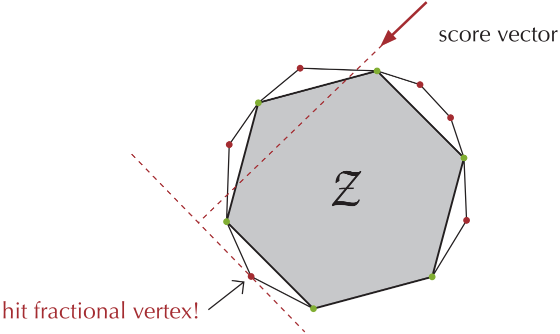
$$\max_{\mathbf{z} \in \bar{\mathcal{Z}} \cap \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w},$$



Balancing Accuracy and Runtime

- Recall that exact inference in our setting is cast as an ILP:

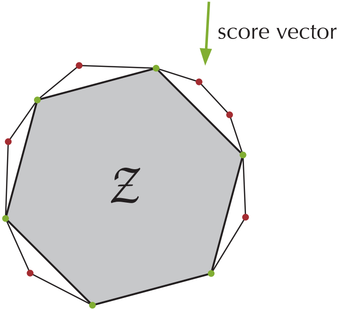
$$\max_{\mathbf{z} \in \bar{\mathcal{Z}} \cap \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w},$$



Balancing Accuracy and Runtime

- Recall that exact inference in our setting is cast as an ILP:

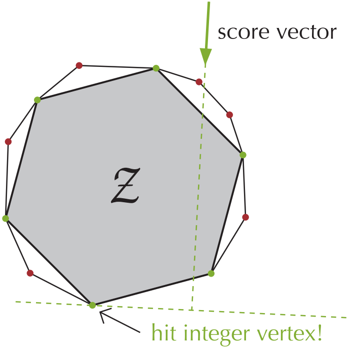
$$\max_{\mathbf{z} \in \bar{\mathcal{Z}} \cap \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w},$$



Balancing Accuracy and Runtime

- Recall that exact inference in our setting is cast as an ILP:

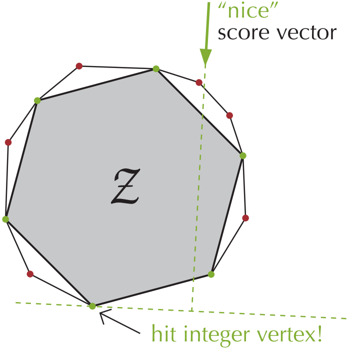
$$\max_{\mathbf{z} \in \bar{\mathcal{Z}} \cap \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w},$$



Balancing Accuracy and Runtime

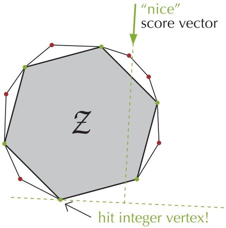
- Recall that exact inference in our setting is cast as an ILP:

$$\max_{\mathbf{z} \in \bar{\mathcal{Z}} \cap \mathbb{Z}^n} \mathbf{s}^\top \mathbf{z} \quad \text{with } \mathbf{s} = \mathbf{F}(x)^\top \mathbf{w},$$



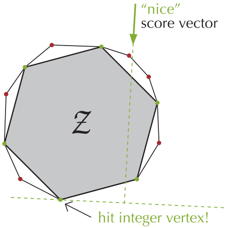
Nice Score Vectors and Low-Cost Hypotheses

- A "nice" score vector $\mathbf{s}$ is one which hits an integer vertex



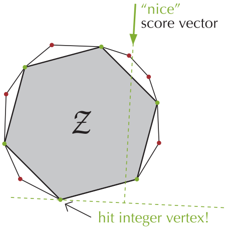
Nice Score Vectors and Low-Cost Hypotheses

- A "nice" score vector $\mathbf{s}$ is one which hits an integer vertex
- At test time: $\mathbf{s} \sim P(\mathbf{F}(X)^\top \mathbf{w})$ is a r.v. that depends on X (filtered by the parameters $\mathbf{w}$)



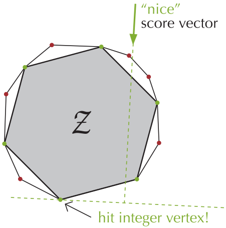
Nice Score Vectors and Low-Cost Hypotheses

- A "nice" score vector $\mathbf{s}$ is one which hits an integer vertex
- At test time: $\mathbf{s} \sim P(\mathbf{F}(X)^\top \mathbf{w})$ is a r.v. that depends on X (filtered by the parameters $\mathbf{w}$)
- A low-cost hypothesis $h_{\mathbf{w}}$ is one which yields $P(\mathbf{F}(X)^\top \mathbf{w})$ with large mass on "nice" score vectors



Nice Score Vectors and Low-Cost Hypotheses

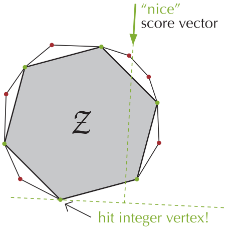
- A "nice" score vector $\mathbf{s}$ is one which hits an integer vertex
- At test time: $\mathbf{s} \sim P(\mathbf{F}(X)^\top \mathbf{w})$ is a r.v. that depends on X (filtered by the parameters $\mathbf{w}$)
- A low-cost hypothesis $h_{\mathbf{w}}$ is one which yields $P(\mathbf{F}(X)^\top \mathbf{w})$ with large mass on "nice" score vectors



- Idea: Approximate computational cost by relaxation gap: $\mathbb{E}\ell_c(h_{\mathbf{w}}, X) \approx \mathbb{E}\ell(h_{\mathbf{w}}(X), \bar{h}_{\mathbf{w}}(X))$

Nice Score Vectors and Low-Cost Hypotheses

- A "nice" score vector $\mathbf{s}$ is one which hits an integer vertex
- At test time: $\mathbf{s} \sim P(\mathbf{F}(X)^\top \mathbf{w})$ is a r.v. that depends on X (filtered by the parameters $\mathbf{w}$)
- A low-cost hypothesis $h_\mathbf{w}$ is one which yields $P(\mathbf{F}(X)^\top \mathbf{w})$ with large mass on "nice" score vectors



- Idea: Approximate computational cost by relaxation gap: $\mathbb{E}\ell_c(h_\mathbf{w}, X) \approx \mathbb{E}\ell(h_\mathbf{w}(X), \bar{h}_\mathbf{w}(X))$
  - Most ILP solvers (branch-and-bound, Gomory's cuts) converge faster as this gap is smaller

Balancing Accuracy and Runtime

- Add a empirical relaxation gap term to our learning objective: $\frac{1}{m}\sum_{t=1}^{m}(\bar{r}_t(\mathbf{w}) - r_t(\mathbf{w}))$

Balancing Accuracy and Runtime

- Add a empirical relaxation gap term to our learning objective: $\frac{1}{m}\sum_{t=1}^{m}(\bar{r}_t(\mathbf{w}) - r_t(\mathbf{w}))$
- The learning problem becomes

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \underbrace{\frac{1-\eta}{m}\sum_{t=1}^{m} r_t(\mathbf{w})}_{\text{Exact LAI}} + \underbrace{\frac{\eta}{m}\sum_{t=1}^{m}\bar{r}_t(\mathbf{w})}_{\text{Relaxed LAI}}.$$

Balancing Accuracy and Runtime

- Add a empirical relaxation gap term to our learning objective: $\frac{1}{m}\sum_{t=1}^{m}(\bar{r}_t(\mathbf{w}) - r_t(\mathbf{w}))$
- The learning problem becomes

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \underbrace{\frac{1-\eta}{m}\sum_{t=1}^{m} r_t(\mathbf{w})}_{\text{Exact LAI}} + \underbrace{\frac{\eta}{m}\sum_{t=1}^{m} \bar{r}_t(\mathbf{w})}_{\text{Relaxed LAI}}.$$

- In the paper: a stochastic adaptation of the online subgradient algorithm [Ratliff et al., 2006]

Balancing Accuracy and Runtime

- Add a empirical relaxation gap term to our learning objective: $\frac{1}{m}\sum_{t=1}^{m}(\bar{r}_t(\mathbf{w}) - r_t(\mathbf{w}))$
- The learning problem becomes

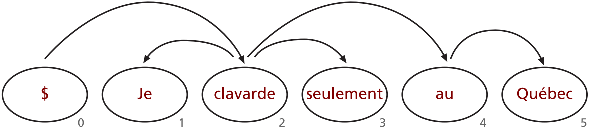
$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \underbrace{\frac{1-\eta}{m}\sum_{t=1}^{m} r_t(\mathbf{w})}_{\text{Exact LAI}} + \underbrace{\frac{\eta}{m}\sum_{t=1}^{m} \bar{r}_t(\mathbf{w})}_{\text{Relaxed LAI}}.$$

- In the paper: a stochastic adaptation of the online subgradient algorithm [Ratliff et al., 2006]
- A PAC bound with respect to the best exact learner
  - It measures the impact of the approximation in learning
  - Previous bounds were in terms of the approximate learner [Kulesza and Pereira, 2007]

Outline

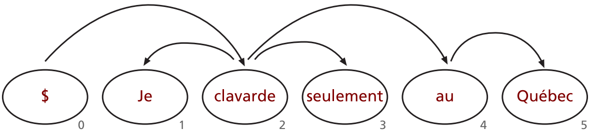
Experiments

- Dependency parsing for seven languages
  - Danish, Dutch, Portuguese, Slovene, Swedish, Turkish, English



Experiments

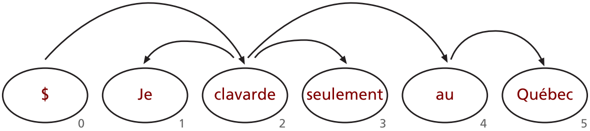
- Dependency parsing for seven languages
  - Danish, Dutch, Portuguese, Slovene, Swedish, Turkish, English



- Exact inference is efficient with a arc-factored model
  - Find a maximal spanning tree [McDonald et al., 2005]

Experiments

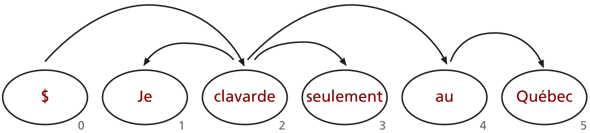
- Dependency parsing for seven languages
  - Danish, Dutch, Portuguese, Slovene, Swedish, Turkish, English



- Exact inference is efficient with a arc-factored model
  - Find a maximal spanning tree [McDonald et al., 2005]
- Beyond that: NP-hard [McDonald and Satta, 2007]

Experiments

- Dependency parsing for seven languages
  - Danish, Dutch, Portuguese, Slovene, Swedish, Turkish, English



- Exact inference is efficient with a arc-factored model
  - Find a maximal spanning tree [McDonald et al., 2005]
- Beyond that: NP-hard [McDonald and Satta, 2007]
- Our model: a ILP formulation with non-arc-factored features
  - Models grandparents/siblings interactions
  - Models valency and nonprojective arcs
  - Only $O(n^3)$ variables and constraints
  - More details: [Martins et al., 2009]

Experiments

- Experiment #1: Training with LP-relaxed LAI ($\eta = 1$)

Experiments

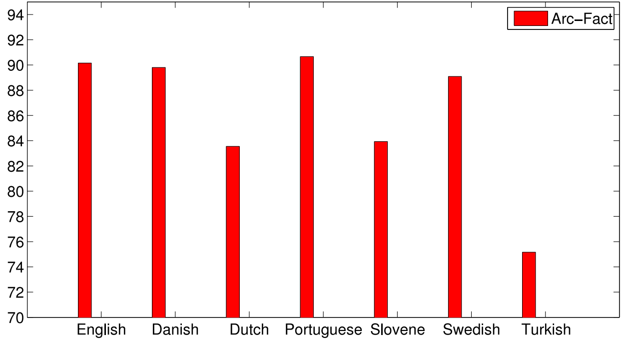
- Experiment #1: Training with LP-relaxed LAI ($\eta = 1$)
- Two different decoders at test time:
  - Exact decoder (solve an ILP)
  - Approximate decoder (solve the relaxed LP; if the solution is fractional, round it in polynomial time by finding a maximal spanning tree on the reweighted graph)

Experiments

- Experiment #1: Training with LP-relaxed LAI ($\eta = 1$)
- Two different decoders at test time:
  - Exact decoder (solve an ILP)
  - Approximate decoder (solve the relaxed LP; if the solution is fractional, round it in polynomial time by finding a maximal spanning tree on the reweighted graph)
- Strong baselines:
  - [MP06] — approximate second-order parser [McDonald and Pereira, 2006]
  - [MDSX08] — stacked parser [Martins et al., 2008]

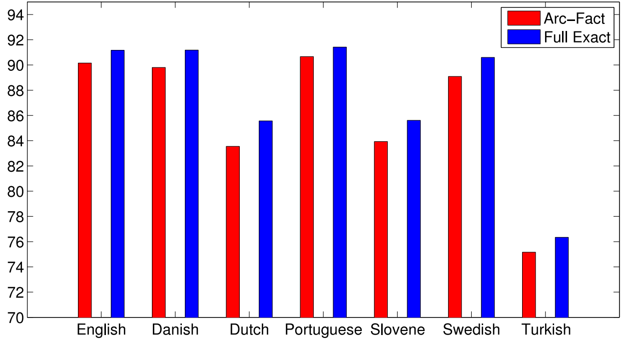
Experiments

- Our models:
  - Arc-factored



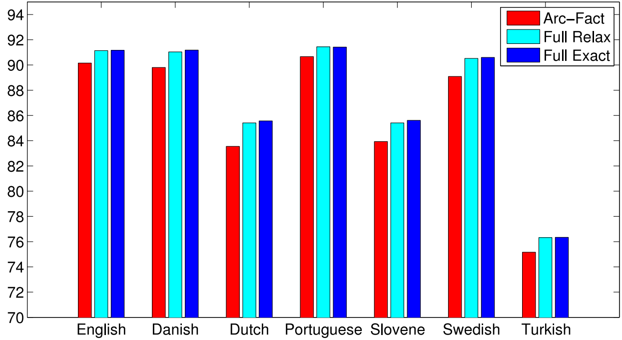
Experiments

- Our models:
  - Arc-factored
  - Full model (with exact decoding) — much better



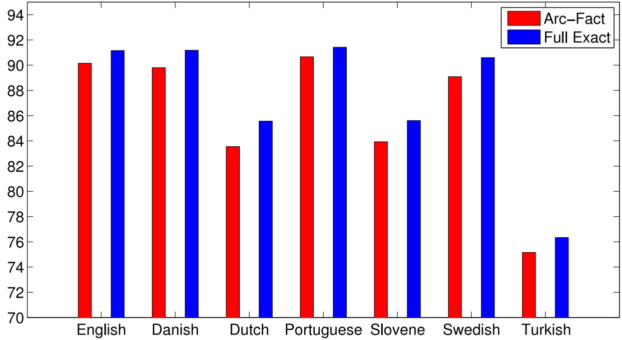
Experiments

- Our models:
  - Arc-factored
  - Full model (with exact decoding) — much better
  - Approximate decoding — did not considerably affect accuracy



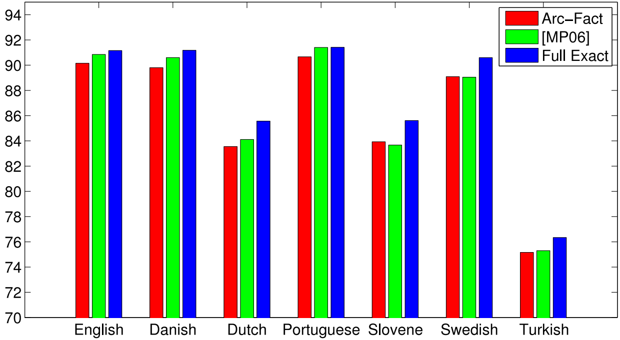
Experiments

- Where are the baselines?



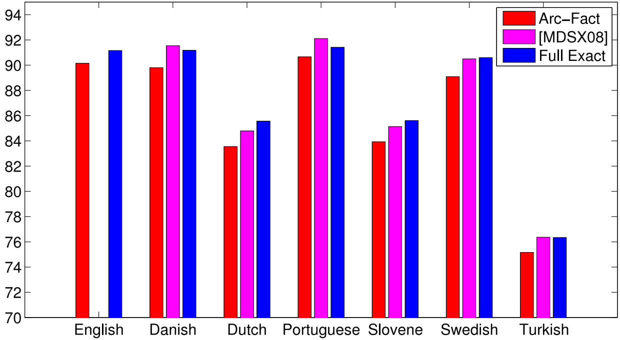
Experiments

- Where are the baselines?
  - ≫ [MP06] – second-order parser [McDonald and Pereira, 2006]



Experiments

- Where are the baselines?
  - $\gg$ [MP06] – second-order parser [McDonald and Pereira, 2006]
  - $\approx$ [MDSX08] — stacked parser [Martins et al., 2008]

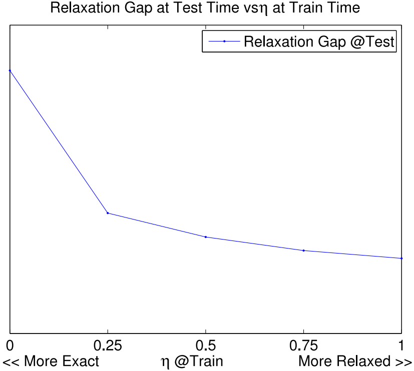


Experiments

- Experiment #2: does η really penalize computational cost?
  - Slovene dataset (with a reduced set of features)

Experiments

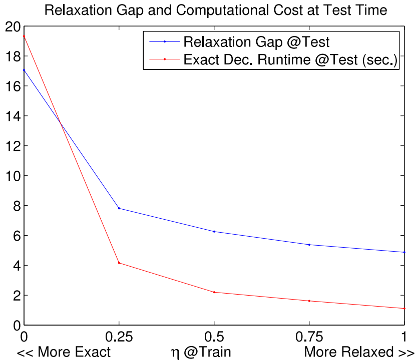
- Experiment #2: does η really penalize computational cost?
  - Slovene dataset (with a reduced set of features)



- As η increases, the model learns to avoid fractional solutions

Experiments

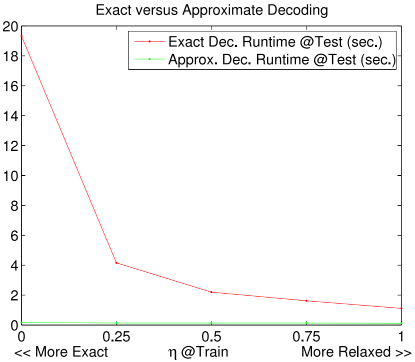
- Experiment #2: does η really penalize computational cost?
  - Slovene dataset (with a reduced set of features)



- As η increases, the model learns to avoid fractional solutions
- Runtime does correlate with the relaxation gap

Experiments

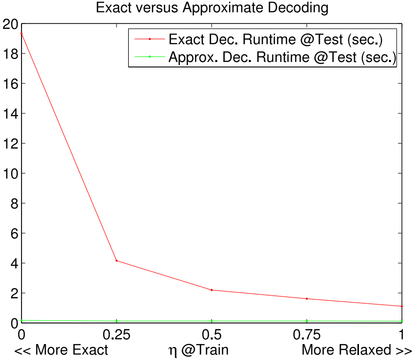
- Experiment #2: does η really penalize computational cost?
  - Slovene dataset (with a reduced set of features)



- As η increases, the model learns to avoid fractional solutions
- Runtime does correlate with the relaxation gap
- Yet the approximate decoder is significantly faster

Experiments

- Experiment #2: does η really penalize computational cost?
  - Slovene dataset (with a reduced set of features)



- As η increases, the model learns to avoid fractional solutions
- Runtime does correlate with the relaxation gap
- Yet the approximate decoder is significantly faster
- Our full model: Same order of magnitude as the baselines ($\approx$ 0.632 sec.)

Outline

Conclusions and Future Work

- We studied the impact of LP relaxed inference in max-margin learning

Conclusions and Future Work

- We studied the impact of LP relaxed inference in max-margin learning
- We established sufficient conditions for algorithmic separability

Conclusions and Future Work

- We studied the impact of LP relaxed inference in max-margin learning
- We established sufficient conditions for algorithmic separability
- As a by-product: a new learning algorithm that penalizes computational cost

Conclusions and Future Work

- We studied the impact of LP relaxed inference in max-margin learning
- We established sufficient conditions for algorithmic separability
- As a by-product: a new learning algorithm that penalizes computational cost
- We demonstrated the effectiveness of these techniques in dependency parsing with non-arc-factored features

Conclusions and Future Work

- We studied the impact of LP relaxed inference in max-margin learning
- We established sufficient conditions for algorithmic separability
- As a by-product: a new learning algorithm that penalizes computational cost
- We demonstrated the effectiveness of these techniques in dependency parsing with non-arc-factored features
- Future work: polyhedral characterizations that guarantee tighter bounds

Conclusions and Future Work

- We studied the impact of LP relaxed inference in max-margin learning
- We established sufficient conditions for algorithmic separability
- As a by-product: a new learning algorithm that penalizes computational cost
- We demonstrated the effectiveness of these techniques in dependency parsing with non-arc-factored features
- Future work: polyhedral characterizations that guarantee tighter bounds
- Conditions for vanishing relaxation gap in online learning

Conclusions and Future Work

- We studied the impact of LP relaxed inference in max-margin learning
- We established sufficient conditions for algorithmic separability
- As a by-product: a new learning algorithm that penalizes computational cost
- We demonstrated the effectiveness of these techniques in dependency parsing with non-arc-factored features
- Future work: polyhedral characterizations that guarantee tighter bounds
- Conditions for vanishing relaxation gap in online learning
- Connections with regularization

References I

Chu, Y. J. and Liu, T. H. (1965).
On the shortest arborescence of a directed graph.
Science Sinica, 14:1396–1400.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006).
Online Passive-Aggressive Algorithms.
JMLR, 7:551–585.

Edmonds, J. (1967).
Optimum branchings.
Journal of Research of the National Bureau of Standards, 71B:233–240.

Eisner, J. (1996).
Three new probabilistic models for dependency parsing: An exploration.
In *COLING*.

Finley, T. and Joachims, T. (2008).
Training structural SVMs when exact inference is intractable.
In *ICML*.

Kulesza, A. and Pereira, F. (2007).
Structured Learning with Approximate Inference.
NIPS.

Lafferty, J., McCallum, A., and Pereira, F. (2001).
Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
In *Proc. of ICML*.

References II

Magnanti, T. and Wolsey, L. (1994).
Optimal Trees.
Technical Report 290-94, Massachusetts Institute of Technology, Operations Research Center.

Martins, A. F. T., Das, D., Smith, N. A., and Xing, E. P. (2008).
Stacking dependency parsers.
In *Proc. of EMNLP*.

Martins, A. F. T., Smith, N. A., and Xing, E. P. (2009).
Concise integer linear programming formulations for dependency parsing.
In *Proc. of ACL-IJCNLP*.

McDonald, R. and Satta, G. (2007).
On the complexity of non-projective data-driven dependency parsing.
In *Proc. of IWPT*.

McDonald, R. T., Pereira, F., Ribarov, K., and Hajic, J. (2005).
Non-projective dependency parsing using spanning tree algorithms.
In *Proc. of HLT-EMNLP*.

McDonald, R. T. and Pereira, F. C. N. (2006).
Online learning of approximate dependency parsing algorithms.
In *Proc. of EACL*.

Ratliff, N., Bagnell, J., and Zinkevich, M. (2006).
Subgradient methods for maximum margin structured learning.
In *ICML Workshop on Learning in Structured Outputs Spaces*.

References III

Smith, D. A. and Eisner, J. (2008).
Dependency parsing by belief propagation.
In *Proc. of EMNLP*.

Taskar, B., Chatalbashev, V., and Koller, D. (2004).
Learning associative Markov networks.
In *ICML*. ACM New York, NY, USA.

Taskar, B., Guestrin, C., and Koller, D. (2003).
Max-margin markov networks.
In *NIPS*.

Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. (2004).
Support vector machine learning for interdependent and structured output spaces.
In *ICML*. ACM New York, NY, USA.

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** $m = |\mathcal{L}|$ **do**

end for

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** $m = |\mathcal{L}|$ **do**
  Pick $\sigma_t \sim \text{Bernoulli}(\eta_t)$

end for

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** $m = |\mathcal{L}|$ **do**
  Pick $\sigma_t \sim \text{Bernoulli}(\eta_t)$
  if $\sigma_t = 1$ **then**
    Solve relaxed LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\bar{\mathbf{z}}'_t \in \bar{\mathcal{Z}}} \mathbf{w}_t^\top \mathbf{F}_t(\bar{\mathbf{z}}'_t - \mathbf{z}_t) + \ell(\bar{\mathbf{z}}'_t; \mathbf{z}_t)$
  else

  end if

end for

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** $m = |\mathcal{L}|$ **do**
  Pick $\sigma_t \sim \text{Bernoulli}(\eta_t)$
  if $\sigma_t = 1$ **then**
    Solve relaxed LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\bar{\mathbf{z}}'_t \in \bar{\mathcal{Z}}} \mathbf{w}_t^\top \mathbf{F}_t(\bar{\mathbf{z}}'_t - \mathbf{z}_t) + \ell(\bar{\mathbf{z}}'_t; \mathbf{z}_t)$
  else
    Solve exact LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\mathbf{z}'_t \in \mathcal{Z}} \mathbf{w}_t^\top \mathbf{F}_t(\mathbf{z}'_t - \mathbf{z}_t) + \ell(\mathbf{z}'_t; \mathbf{z}_t)$
  end if

end for

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** $m = |\mathcal{L}|$ **do**
  Pick $\sigma_t \sim \text{Bernoulli}(\eta_t)$
  if $\sigma_t = 1$ **then**
    Solve relaxed LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\bar{\mathbf{z}}'_t \in \bar{\mathcal{Z}}} \mathbf{w}_t^\top \mathbf{F}_t(\bar{\mathbf{z}}'_t - \mathbf{z}_t) + \ell(\bar{\mathbf{z}}'_t; \mathbf{z}_t)$
  else
    Solve exact LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\mathbf{z}'_t \in \mathcal{Z}} \mathbf{w}_t^\top \mathbf{F}_t(\mathbf{z}'_t - \mathbf{z}_t) + \ell(\mathbf{z}'_t; \mathbf{z}_t)$
  end if
  Compute the subgradient $\mathbf{g}_t \leftarrow \lambda \mathbf{w}_t + \mathbf{F}_t(\hat{\mathbf{z}}_t - \mathbf{z}_t)$

end for

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** $m = |\mathcal{L}|$ **do**
  Pick $\sigma_t \sim \text{Bernoulli}(\eta_t)$
  if $\sigma_t = 1$ **then**
    Solve relaxed LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\bar{\mathbf{z}}'_t \in \bar{\mathcal{Z}}} \mathbf{w}_t^\top \mathbf{F}_t(\bar{\mathbf{z}}'_t - \mathbf{z}_t) + \ell(\bar{\mathbf{z}}'_t; \mathbf{z}_t)$
  else
    Solve exact LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\mathbf{z}'_t \in \mathcal{Z}} \mathbf{w}_t^\top \mathbf{F}_t(\mathbf{z}'_t - \mathbf{z}_t) + \ell(\mathbf{z}'_t; \mathbf{z}_t)$
  end if
  Compute the subgradient $\mathbf{g}_t \leftarrow \lambda \mathbf{w}_t + \mathbf{F}_t(\hat{\mathbf{z}}_t - \mathbf{z}_t)$
  Project and update $\mathbf{w}_{t+1} \leftarrow \text{Proj}_{\mathcal{W}}(\mathbf{w}_t - \alpha_t \mathbf{g}_t)$
end for

Balancing Accuracy and Runtime

Stochastic Online Subgradient Algorithm (based on [Ratliff et al., 2006])

Input: $\mathcal{L}$, $\langle \eta_t \rangle_t$, learning rate sequence $\langle \alpha_t \rangle_t$
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** $m = |\mathcal{L}|$ **do**
  Pick $\sigma_t \sim \text{Bernoulli}(\eta_t)$
  if $\sigma_t = 1$ **then**
    Solve relaxed LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\bar{\mathbf{z}}'_t \in \bar{\mathcal{Z}}} \mathbf{w}_t^\top \mathbf{F}_t(\bar{\mathbf{z}}'_t - \mathbf{z}_t) + \ell(\bar{\mathbf{z}}'_t; \mathbf{z}_t)$
  else
    Solve exact LAI, $\hat{\mathbf{z}}_t \leftarrow \arg\max_{\mathbf{z}'_t \in \mathcal{Z}} \mathbf{w}_t^\top \mathbf{F}_t(\mathbf{z}'_t - \mathbf{z}_t) + \ell(\mathbf{z}'_t; \mathbf{z}_t)$
  end if
  Compute the subgradient $\mathbf{g}_t \leftarrow \lambda \mathbf{w}_t + \mathbf{F}_t(\hat{\mathbf{z}}_t - \mathbf{z}_t)$
  Project and update $\mathbf{w}_{t+1} \leftarrow \mathrm{Proj}_{\mathcal{W}}(\mathbf{w}_t - \alpha_t \mathbf{g}_t)$
end for
Return the averaged model $\hat{\mathbf{w}} \leftarrow \frac{1}{m}\sum_{t=1}^{m} \mathbf{w}_t$.

Generalization Bound

Proposition

Setting $\alpha_t = 1/(\lambda t)$, $\lambda = \Theta\left(\sqrt{\frac{1+\log m}{m}}\right)$ *and* $\eta_t = \Theta(t^{-1/2})$*:*

$$\mathbb{E}\ell(h_{\hat{\mathbf{w}}}(X), Y) \leq \frac{1}{m}\sum_{t=1}^{m} r_t(\mathbf{w}^*) + \frac{L \cdot o(m)}{m} + O\left(\sqrt{\frac{1}{m}\ln\frac{1}{\delta}}\right)$$

holds with probability $\geq 1 - \delta$.

Generalization Bound

Proposition

Setting $\alpha_t = 1/(\lambda t)$, $\lambda = \Theta\left(\sqrt{\frac{1+\log m}{m}}\right)$ *and* $\eta_t = \Theta(t^{-1/2})$*:*

$$\mathbb{E}\ell(h_{\hat{\mathbf{w}}}(X), Y) \leq \frac{1}{m}\sum_{t=1}^{m} r_t(\mathbf{w}^*) + \frac{L \cdot o(m)}{m} + O\left(\sqrt{\frac{1}{m}\ln\frac{1}{\delta}}\right)$$

holds with probability $\geq 1 - \delta$.

- Remark: The bound is in terms of what could be achieved with the best exact learner
  - It measures the impact of the approximation in learning

Generalization Bound

Proposition

Setting $\alpha_t = 1/(\lambda t)$, $\lambda = \Theta\left(\sqrt{\frac{1+\log m}{m}}\right)$ *and* $\eta_t = \Theta(t^{-1/2})$:

$$\mathbb{E}\ell(h_{\hat{\mathbf{w}}}(X), Y) \leq \frac{1}{m}\sum_{t=1}^{m} r_t(\mathbf{w}^*) + \frac{L \cdot o(m)}{m} + O\left(\sqrt{\frac{1}{m}\ln\frac{1}{\delta}}\right)$$

holds with probability $\geq 1 - \delta$.

- Remark: The bound is in terms of what could be achieved with the best exact learner
  - It measures the impact of the approximation in learning
- Previous bounds were in terms of the approximate learner

  [Kulesza and Pereira, 2007]