A General-Purpose Algorithm for Constrained Sequential Inference

Daniel Deutsch,* Shyam Upadhyay,* and Dan Roth

*equal contribution
Co-Authors

Shyam Upadhyay

Dan Roth
Structured Prediction

• Structured prediction is everywhere
  – Parsing, tagging, generation
• Inference is hard: exponentially large search space
• Not all outputs are “valid”
Valid Output Structures: Parsing

**Input:** John kissed Mary

**Gold Parse:**

\[
(S \ (NP \ XX) \ (VP \ XX \ (NP \ XX) \ )\ )
\]
Valid Output Structures: Parsing

Input: John kissed Mary

Gold Parse: (S (NP XX ) (VP XX (NP XX ) ) )

Invalid Parse: (S (NP ) (VP XX XX (NP XX ) ) ) ❌

Empty phrase
Valid Output Structures: Parsing

Input: John kissed Mary

Gold Parse: \((S (NP \ XX \ ) (VP \ XX \ (NP \ XX \ ) \ ) \ )\)

Invalid Parse: \((S (NP \ ) (VP \ XX \ XX \ (NP \ XX \ ) \ ) \ )\) ×

Empty phrase

Invalid Parse: \((S (VP \ XX \ (NP \ XX \ ) \ ) \ )\) ×

Incorrect number of pre-terminals
Valid Output Structures: Parsing

Input: John kissed Mary

Gold Parse: (S (NP XX ) (VP XX (NP XX ) ) )

Invalid Parse: (S (NP ) (VP XX XX (NP XX ) ) ) ❌
 Empty phrase

Invalid Parse: (S (VP XX (NP XX ) ) ) ❌
 Incorrect number of pre-terminals

Invalid Parse: (S (NP XX ) (VP XX (NP XX ) ) ) ❌
 Unbalanced parentheses
Valid Output Structures: Semantic Role Labelling

A0 \approx \text{agent}
A1 \approx \text{patient}

\textbf{Input:} Alice Smith \textbf{gave} a flower to Bob Jones

\textbf{Gold Tags:} A0 A0 O O A1 O A2 A2
Valid Output Structures: Semantic Role Labelling

**Input:** Alice Smith gave a flower to Bob Jones

**Gold Tags:**

- Alice: A0
- Smith: A0
- gave: O
- a: O
- flower: A1
- to: O
- Bob: A2
- Jones: A2

**Invalid Tags:**

- Duplicate A0

\[\text{Duplicate A0}\]
Valid Output Structures: Semantic Role Labelling

**Input:** Alice Smith gave a flower to Bob Jones

**Gold Tags:**
- A0
- A0
- O
- O
- A1
- O
- A2
- A2

**Legal Args:** A0, A1, A2 (from Propbank for “gave”)

**Invalid Tags:**
- A0
- A0
- O
- O
- A0
- O
- A2
- A2

Duplicate A0
Valid Output Structures: Semantic Role Labelling

**Input:** Alice Smith gave a flower to Bob Jones

**Gold Tags:** A0 A0 O O A1 O A2 A2

**Legal Args:** A0, A1, A2 (from Propbank for “gave”)

**Invalid Tags:**
- A0 A0 O O A0 O A2 A2 (Duplicate A0)
- A3 A3 O O A1 O A2 A2 (Illegal argument A3)
Valid Output Structures: Semantic Role Labelling

Input: Alice Smith gave a flower to Bob Jones

Gold Tags: A0 A0 O O A1 O A2 A2

Legal Args: A0, A1, A2 (from Propbank for “gave”)

Spans: [Alice Smith] gave a flower to [Bob Jones]

Invalid Tags: A0 A0 O O A0 O A2 A2 ×
Duplicate A0

Invalid Tags: A3 A3 O O A1 O A2 A2 ×
Illegal argument A3
Valid Output Structures: Semantic Role Labelling

Input: Alice Smith gave a flower to Bob Jones

Gold Tags: A0 A0 O O A1 O A2 A2

Legal Args: A0, A1, A2 (from Propbank for “gave”)

Spans: [Alice Smith] gave a flower to [Bob Jones]

Invalid Tags: A0 A0 O O A0 O A2 A2 ✗
Duplicate A0

Invalid Tags: A3 A3 O O A1 O A2 A2 ✗
Illegal argument A3

Invalid Tags: A0 O O O A1 O A2 A2 ✗
Spans not respected
Sequential Inference

• Many tasks have converged on the same solution
  – Assume output structure is a sequence
  – Beam search from left-to-right
  – Seq2Seq

• How are structural constraints enforced?

---

The red cat

El gato rojo
Enforcing Constraints

• No single way to enforce constraints
  – Not enforced at all (Lample et al., 2016; Choe and Charniak, 2016; Suhr et al., 2018)
  – Post-hoc (Andreas et al., 2013; Vinyals et al., 2015; Upadhyay et al., 2018)
  – Custom inference algorithm (Zhu et al., 2013)
This Work

• Propose a generic algorithm for enforcing constraints in sequential inference
• Abstracts over many custom inference procedures in NLP
• Built on expressing constraints as automata
• Automata guides inference to valid outputs
Representing Constraints

• Represent a constraint with an automaton
  – Finite-state automata and push-down automata
• Automaton’s language is all of the valid output structures that satisfy the constraint
• Great automaton libraries
  – OpenFST (Allauzen et al., 2007), Pynini (Gorman, 2016)
Representing Constraints as Automata

[Alice Smith] gave flowers to [Bob Jones]

= Any tag
Constrained Inference

- Automaton will be traversed in lock-step with beam search, guiding inference to find a valid structure.
Constrained Inference

Alice Smith gave flowers to Bob Jones

\[ \sum = \text{Any tag} \]
Constrained Inference

Alice Smith gave flowers to Bob Jones

O O O O O O O O
A0 A0 A0 A0 A0 A0
A1 A1 A1 A1 A1 A1
A2 A2 A2 A2 A2 A2
A3 A3 A3 A3 A3 A3

Σ = Any tag
## Constrained Inference

<table>
<thead>
<tr>
<th>Alice Smith gave flowers to Bob Jones</th>
</tr>
</thead>
<tbody>
<tr>
<td>O O O O O O O O O O A0 A0 A0 A0 A0 A0 A0 A0 A0 A0 A1 A1 A1 A1 A1 A1 A1 A1 A1 A1 A2 A2 A2 A2 A2 A2 A2 A2 A2 A2 A3 A3 A3 A3 A3 A3 A3 A3 A3 A3 A3</td>
</tr>
</tbody>
</table>

\[ \Sigma = \text{Any tag} \]
Constrained Inference

Alice Smith gave flowers to Bob Jones

\[ \Sigma = \text{Any tag} \]
Constrained Inference

Alice Smith gave flowers to Bob Jones

\[ \Sigma = \text{Any tag} \]
Constrained Inference

Alice Smith gave flowers to Bob Jones

\[ \Sigma \sum = \text{Any tag} \]
Constrained Inference

Alice Smith gave flowers to Bob Jones

$\Sigma = \text{Any tag}$
Constrained Inference

Alice Smith gave flowers to Bob Jones

A0 → A0
A1 → A1
A2 → A2
A3 → A3

A0 A0 A0 A0
A1 A1 A1 A1
A2 A2 A2 A2
A3 A3 A3 A3

Σ = Any tag

Σ

O O O O
O O O O
O O O O
O O O O
Constrained Inference

Alice Smith gave flowers to Bob Jones

\[ \sum = \text{Any tag} \]
Handling Multiple Constraints

• What about multiple constraints?
Handling Multiple Constraints

• What about multiple constraints?
• Can’t traverse jointly -- Need to intersect all of the automata
Handling Multiple Constraints

• What about multiple constraints?
• Can’t traverse jointly -- Need to intersect all of the automata
• Intersecting automata can be expensive
Handling Multiple Constraints

• What about multiple constraints?
• Can’t traverse jointly -- Need to intersect all of the automata
• Intersecting automata can be expensive
• Idea: Intersect automata only as necessary (Tromble and Eisner, 2006)
• Maintain an active set of enforced constraints
Active Set Algorithm

Input

John kissed Mary
Active Set Algorithm

Input

Balanced Parentheses

Non-Empty Phrases

Correct Num. Pre-Terminals

John kissed Mary
### Active Set Algorithm

<table>
<thead>
<tr>
<th>Input</th>
<th>Balanced Parentheses</th>
<th>Non-Empty Phrases</th>
<th>Correct Num. Pre-Terminals</th>
</tr>
</thead>
<tbody>
<tr>
<td>John kissed Mary</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Constraint Automaton**

**Inference Output**

**Violated Constraints**
Active Set Algorithm

Input
John kissed Mary

Balanced Parentheses

Non-Empty Phrases

Correct Num. Pre-Terminals

Constraint Automaton
n/a

Inference Output
(S (VP XX (NP XX )) )

Violated Constraints
Correct Num. Pre-Terminals
Active Set Algorithm

Input: John kissed Mary

Balanced Parentheses

Non-Empty Phrases

Correct Num. Pre-Terminals

Constraint Automaton

Inference Output

Violated Constraints

Correct Num. Pre-Terminals

Balanced Parentheses

n/a

(S (VP XX (NP XX ) ) )

(S (NP XX ) (VP XX (NP XX ) ) )

(S (NP XX ) (VP XX (NP XX ) ) )

(S (VP XX (NP XX ) ) )

n/a

(S (NP XX ) (VP XX (NP XX ) ) )

(S (VP XX (NP XX ) ) )
Active Set Algorithm

<table>
<thead>
<tr>
<th>Input</th>
<th>Balanced Parentheses</th>
<th>Non-Empty Phrases</th>
<th>Correct Num. Pre-Terminals</th>
</tr>
</thead>
<tbody>
<tr>
<td>John kissed Mary</td>
<td>![Diagram]</td>
<td>![Diagram]</td>
<td>![Diagram]</td>
</tr>
</tbody>
</table>

For the input "John kissed Mary":

- Constraint Automaton: n/a
- Inference Output: (S (VP XX (NP XX )) )
- Violated Constraints: None

Correct Num. Pre-Terminals:
- (S (NP XX ) (VP XX (NP XX )) )

Balanced Parentheses:
- (S (NP XX ) (VP XX (NP XX )) )
- (S (NP XX ) (VP XX (NP XX )) )

Correct Num. Pre-Terminals:
- (S (NP XX ) (VP XX (NP XX )) )

None
Generality of Algorithm

• Abstract over constraint-specific inference algorithms:
  – BIO tagging
  – Shift-reduce parsing
  – Require/disallow n-grams
  – Constraining verbalizations in text-to-speech (Zhang et al., 2019)
Constituency Parsing Experiments

- Seq2Seq constituency parsing on the Penn Treebank

John kissed Mary
Necessity of Constraints

• Do we need constraints at all?
• Enforce three different constraints
  – Balanced parentheses
  – Correct number of pre-terminals
  – No empty phrases
Necessity of Constraints

The graph shows the percentage of training data against the percent satisfied. Different lines representing different categories:
- **NONEMPTY**
- **BAL**
- **#PRETERM**
- **All**

The data indicates an upward trend as the percentage of training data increases, reaching a plateau at higher percentages.
Necessity of Constraints

![Graph showing the percentage of training data against percent satisfied. The graph includes lines for 'NONEMPTY', 'BAL', '#PRETERM', and 'All'.]
Necessity of Constraints

![Graph showing the relationship between the percentage of training data and the percentage of satisfied cases. The graph includes lines for different categories: **NONEMPTY**, **BAL**, **#PRETERM**, and **All**.](image)

- **Percent Satisfied** on the y-axis.
- **Percentage of Training Data** on the x-axis.
Correct number of pre-terminals isn’t learned with full Penn Treebank
Inference Algorithm Comparison

• Compare how well different algorithms satisfy constraints
  – Unconstrained
  – Constrained
  – Post-Hoc
Post-Hoc Inference

- Run unconstrained beam search with a beam of size $k$
- Return highest-scoring valid output

<table>
<thead>
<tr>
<th>Score</th>
<th>Tree Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.32</td>
<td>(S (NP ) (VP XX XX (NP XX ) ) )</td>
</tr>
<tr>
<td>0.28</td>
<td>(S (VP XX (NP XX ) ) )</td>
</tr>
<tr>
<td>0.25</td>
<td>(S (NP XX ) (VP XX (NP XX ) ) )</td>
</tr>
<tr>
<td>0.10</td>
<td>(S (NP XX ) (VP XX (NP XX ) ) )</td>
</tr>
</tbody>
</table>
Parsing Learning Curve

![Graph showing the relationship between the percentage of training data and the percent satisfied. The graph includes three lines: Unconstrained, Constrained, and PostHoc.]
Parsing Learning Curve

![Graph showing the parsing learning curve with three lines: unconstrained, constrained, and posthoc. The x-axis represents the percentage of training data, ranging from 0% to 100%, and the y-axis represents the percentage satisfied, ranging from 0% to 100%. The unconstrained line starts lower and reaches a higher percentage satisfied than the constrained and posthoc lines. The constrained line starts higher than the posthoc line and both reach a similar percentage satisfied at higher percentages of training data. The posthoc line is the lowest initially and then climbs steadily.]
Parsing Learning Curve

- **UNCONSTRAINED**
- **CONSTRAINED**
- **POSTHOC**
Parsing Learning Curve

F\textsubscript{1} details available in the paper
Semantic Role Labeling Experiments

- Model of He et al. (2017) on CoNLL 2005
- Assume gold predicates

\[ \text{A0} \rightarrow \text{A0} \rightarrow \text{O} \rightarrow \text{O} \rightarrow \text{A1} \rightarrow \text{O} \rightarrow \text{A2} \rightarrow \text{A2} \]

Alice Smith gave a flower to Bob Jones
Semantic Role Labeling Learning Curve

• Impose constraints at varying levels of supervision
  – Disallow duplicate arguments
  – Disallow invalid arguments (via Propbank)
  – Require spans to have the same label
Semantic Role Labeling Learning Curve

Percentage of Training Data

CoNLL $F_1$

- **UNCONSTRAINED**
- **+NoDUP**
- **+LEGALARGS**
- **+SPANLABEL** (Fully Constrained)
Semantic Role Labeling Learning Curve

![Learning Curve Graph](image)

- **CoNLL F₁**
- **Percentage of Training Data**

Legend:
- **Unconstrained**
- **+NoDUP**
- **+LegalArgs**
- **+SpanLabel (Fully Constrained)**

Penn Engineering
Semantic Role Labeling Learning Curve

Percentage of Training Data vs. CoNLL F₁ scores for different constrained and unconstrained settings.
Semantic Role Labeling Learning Curve

CoNLL F₁ vs Percentage of Training Data

- **Unconstrained**
- **+NoDUP**
- **+LegalArgs**
- **+SpanLabel (Fully Constrained)**
Constraints consistently help at all levels of supervision

Constraints help most in low-supervision settings

- **UNCONSTRAINED**
- **+NoDUP**
- **+LEGALARGS**
- **+SPANLABEL (Fully Constrained)**

---

**Semantic Role Labeling Learning Curve**

![Graph showing the improvement in CoNLL F1 score with increasing percentage of training data.](image-url)

- Constraints help consistently across all levels of supervision.
- They are most effective in low-supervision settings.

---

59
Active Set Size & Speedup

• Measured number of constraints added to the working set until a valid output found

• Calculated speed of inference relative to intersecting all constraints then running inference
Fewer constraints are necessary to impose at higher levels of supervision, which results in larger speedups.
Conclusion

• Presented a general-purpose algorithm to enforce constraints in sequential inference
• Demonstrated enforcing constraints is necessary and beneficial at all levels of supervision
• Showed the active set method for imposing multiple constraints results in faster inference
Thank you!

https://github.com/CogComp/gcd
References


