Using and teaching logic and machine learning for modeling cognitive processes

Anders Søgaard

Center for Language Technology
University of Copenhagen
Njalsgade 140–2
DK-2300 Copenhagen S
Email: soegaard@hum.ku.dk

November 25 2009
Running example: dependency parsing

MA Programme in IT & Cognition

Student experiences 2007–9

Recent initiatives

Logic for dependency parsing

Machine learning for dependency parsing

Logic and machine learning?
 dependency parsing

- The 1-best parsing problem for projective dependency grammars is in $O(|G|n^3)$. Non-projective dependency parsing is NP-hard in general (e.g. by the Traveling Salesman Problem).
- Popular approximate parsing algorithms exist for both projective (deterministic transition-based; linear time) and non-projective dependency parsing (minimum spanning tree, $O(|G|n^2)$).
Graph-based dependency parsing (MST)

Edmonds (1969) introduced a two-step $\mathcal{O}(|G|n^2)$ minimum spanning tree algorithm for edge-factored models:

(i) greedy head selection

(ii) cycle contraction.
Graph-based dependency parsing (MST)

Edmonds (1969) introduced a two-step $\mathcal{O}(|G|n^2)$ minimum spanning tree algorithm for edge-factored models:

(i) greedy head selection

(ii) cycle contraction.

John smokes Lebanese $\Rightarrow$
Graph-based dependency parsing (MST)

Edmonds (1969) introduced a two-step $\mathcal{O}(|G|n^2)$ minimum spanning tree algorithm for edge-factored models:

(i) greedy head selection

(ii) cycle contraction.

\[ \text{John} \quad \text{smokes} \quad \text{Lebanese} \quad \implies \]

\[ \text{John} \quad (\text{smokes Lebanese}) \quad \implies \]
Graph-based dependency parsing (MST)

Edmonds (1969) introduced a two-step $O(|G|n^2)$ minimum spanning tree algorithm for edge-factored models:

(i) greedy head selection

(ii) cycle contraction.

$\text{John}$ $\text{smokes}$ $\text{Lebanese}$ $\implies$

$\text{John}$ $\text{(smokes Lebanese)}$ $\implies$

$\text{John}$ $\text{smokes}$ $\text{Lebanese}$
Transition-based dependency parsing

(i) \[ \text{SHIFT} \quad \ldots \quad w \ldots \]

\[ \quad \ldots w \quad \ldots \]
Transition-based dependency parsing

(i) \[ \text{SHIFT} \quad \ldots w \ldots \]
\[ \ldots w \ldots \]

(ii) \[ \text{REDUCE} \quad \ldots w \ldots \]
\[ \ldots \ldots \text{iff} \quad \exists v. v \to w \]
Transition-based dependency parsing

(i) \[
\text{SHIFT} \quad \ldots \\ \quad \ldots w \\ \quad \ldots \\ \]

(ii) \[
\text{REDUCE} \quad \ldots w \\ \quad \ldots \\ \quad \ldots \quad \text{iff} \quad \exists v. v \rightarrow w \\ \]

(iii) \[
\text{LEFT-ARC} \quad \ldots w \quad v \ldots \\ \quad \ldots \quad v \ldots \quad \text{add} \quad w \leftarrow v \\ \]
Transition-based dependency parsing

(i) **SHIFT**

\[
\begin{array}{c}
\vdash \\
\vdash \\
\vdash
\end{array}
\]

(ii) **REDUCE**

\[
\begin{array}{c}
\vdash \\
\vdash \\
\vdash \\
\vDash \\
\vdash
\end{array} \\
\iff \exists v. v \rightarrow w
\]

(iii) **LEFT-ARC**

\[
\begin{array}{c}
\vdash \\
\vdash \\
\vdash \\
\vdash \\
\vdash
\end{array} \\
\text{add } \vdash \leftarrow v
\]

(iv) **RIGHT-ARC**

\[
\begin{array}{c}
\vdash \\
\vdash \\
\vdash \\
\vdash \\
\vdash
\end{array} \\
\iff \# w'. w' \rightarrow w
How did John get to smoke Libanese?

<table>
<thead>
<tr>
<th>Shift</th>
<th>...</th>
<th>John smokes Lebanese</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left-Arc</td>
<td>... John</td>
<td>smokes Libanese</td>
</tr>
<tr>
<td>Shift</td>
<td>...</td>
<td>smokes Libanese</td>
</tr>
<tr>
<td>Right-Arc</td>
<td>..., smokes</td>
<td>Lebanese</td>
</tr>
<tr>
<td>Reduce</td>
<td>... smokes, Libanese</td>
<td>...</td>
</tr>
<tr>
<td>Root</td>
<td>... smokes</td>
<td>...</td>
</tr>
</tbody>
</table>
MA Programme in IT & Cognition

Computer Science ——— Language Technology ---- Linguistics

Psychology
MA Programme in IT & Cognition

Computer Science ----------- Language Technology ---- Linguistics

Psychology

4  4  5
2007 → 2008 → 2009 → 2010
MA Programme in IT & Cognition

Computer Science — Language Technology — Linguistics

Psychology

<table>
<thead>
<tr>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCS (F)</td>
<td>Form.Ling.</td>
<td>RCS(A)</td>
<td>Thesis</td>
</tr>
<tr>
<td>Logic</td>
<td>CP(F)</td>
<td>CP(A)</td>
<td></td>
</tr>
<tr>
<td>Exp. mthd.</td>
<td>Stats</td>
<td>Adapt.Syst.</td>
<td></td>
</tr>
<tr>
<td>Linguistics</td>
<td>LT(F)</td>
<td>LT(A)</td>
<td></td>
</tr>
<tr>
<td>Progr(F/A)</td>
<td>HCI(F)</td>
<td>HCI(A)</td>
<td></td>
</tr>
</tbody>
</table>

RCS(F) and RCS(A) are compulsory.
## Student experiences 2007–9

<table>
<thead>
<tr>
<th></th>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coherence</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Flexibility</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Level</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Social</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Thesis support</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
Student experiences 2007–9

<table>
<thead>
<tr>
<th></th>
<th>GOOD</th>
<th>BAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coherence</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Flexibility</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Level</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Social</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Thesis support</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

- Students never *stayed* at the university after class to work in groups.
### Student experiences 2007–9

<table>
<thead>
<tr>
<th></th>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coherence</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Flexibility</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Level</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Social</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Thesis support</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

- Students never *stayed* at the university after class to work in groups.
- Students did not know much about each other.
Student experiences 2007–9

<table>
<thead>
<tr>
<th></th>
<th>GOOD</th>
<th>BAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coherence</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Flexibility</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Level</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Social</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Thesis support</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

- Students never *stayed* at the university after class to work in groups.
- Students did not know much about each other.
- Several students did not have a thesis topic ready after having completed the first 90 ECTS.
Recent initiatives
Recent initiatives

- MENTORING, i.e. monthly interviews about:
  - coherence (1st year)
  - courses and exams
  - extra-curricular activities
  - thesis (primarily 2nd year)
Recent initiatives

- **MENTORING**, i.e. monthly interviews about:
  - coherence (1st year)
  - courses and exams
  - extra-curricular activities
  - thesis (primarily 2nd year)

- **STUDENT GROUPS:**
  a. ensemble-based part-of-speech tagging
  b. text prediction
  c. text classification
  d. model-checking for extensions of modal logic
  e. word alignment in translated text

- **COLLABORATION:**
  a. Center for Language Technology
  b. Mikroværkstedet, Lund University
  d. University of Tübingen (Germany)
  e. Copenhagen Business School
• **Evening lectures:**
  
  → J. Hansen (RUC): “Dynamic epistemic logic”
  → P. Lindström (Lund, Sweden): “How children learn math”
  → M. Haulrich (CBS): “Repair in transition-based parsing”
  → R. Dekova (BAS, Bulgaria): “Lexical semantics and the mental lexicon”
• **Evening lectures:**
  → J. Hansen (RUC): “Dynamic epistemic logic”
  → P. Lindström (Lund, Sweden): “How children learn math”
  → M. Haulrich (CBS): “Repair in transition-based parsing”
  → R. Dekova (BAS, Bulgaria): “Lexical semantics and the mental lexicon”

• **Reading groups, workshops, etc.:**
  → CBS-RG Machine Learning
  → CBS-RG Natural Language Processing
  → Linguistic Circle of Copenhagen
  → ACL’10 (Uppsala, Sweden)
  → ESSLLI’10
Challenge: 90 ECTS and a diverse group of students

1. Students *do* stay at the university after class.
2. The student groups “average out” the students.
3. Student groups are also a chance for excellent students to excel.
4. Finally, however, we synchronized our courses a bit:
Challenge: 90 ECTS and a diverse group of students

1. Students *do* stay at the university after class.
2. The student groups “average out” the students.
3. Student groups are also a chance for excellent students to excel.
4. Finally, however, we synchronized our courses a bit:
   (a) RCS(F) and RCS(A) model topics introduced in CP(F).
**Challenge: 90 ECTS and a diverse group of students**

1. Students *do* stay at the university after class.

2. The student groups “average out” the students.

3. Student groups are also a chance for excellent students to excel.

4. Finally, however, we synchronized our courses a bit:
   
   (a) RCS(F) and RCS(A) model topics introduced in CP(F).
   (b) Topics from (a) are reused in other courses (Logic, LT(F), etc.).
Challenge: 90 ECTS and a diverse group of students

1. Students *do* stay at the university after class.

2. The student groups “average out” the students.

3. Student groups are also a chance for excellent students to excel.

4. Finally, however, we synchronized our courses a bit:
   
   (a) RCS(F) and RCS(A) model topics introduced in CP(F).
   (b) Topics from (a) are reused in other courses (Logic, LT(F), etc.).
   (c) All course exercises are in Python/Orange, also used in the student groups.
Logic for dependency parsing

In the dependency graph:

1 : \( p, \neg q \)
2 : \( p, \neg q \)
3 : \( \neg p, \neg q \)
4 : \( \neg p, q \)

- the formula \( \langle \prec \rangle \langle \leftarrow \rangle q \), i.e. the current node precedes a node whose syntactic head is in the denotation of \( q \), evaluates as true in nodes 1 and 2.

- the formula \( \langle \leftarrow \cap \prec ; \leftarrow \rangle \top \) is not true in any node.
Logic for dependency parsing (cont’d)

• A modal logic for dependency parsing was first introduced in Bröker (1997).
Logic for dependency parsing (cont’d)

- A modal logic for dependency parsing was first introduced in Bröker (1997).
- Kepser (2008) uses monadic second order logic to query dependency treebanks.
Logic for dependency parsing (cont’d)

- A modal logic for dependency parsing was first introduced in Bröker (1997).
- Kepser (2008) uses monadic second order logic to query dependency treebanks.
- Søgaard (2009) uses hybrid logic and crude repair to improve accuracy of dependency parsers.
Logic for dependency parsing (cont’d)

- A modal logic for dependency parsing was first introduced in Bröker (1997).
- Kepser (2008) uses monadic second order logic to query dependency treebanks.
- Søgaard (2009) uses hybrid logic and crude repair to improve accuracy of dependency parsers.
- Modal logics for other parsing formalisms are presented in Keller (1993), Søgaard (2007) and Søgaard and Lange (2009).
Machine learning for dependency parsing

- Dependency parsing is typically cast as supervised learning.
Machine learning for dependency parsing

- Dependency parsing is typically cast as supervised learning.
- Sufficient labeled data exists for a wide variety of languages.
  - The CONLL-X Shared Task used datasets from 12 languages.
  - The CONLL 2007 Shared Task used datasets from 10 languages (with three repeats).
  - Labeled data exists for other languages, incl. Hebrew, Latin, Romanian, Thai.
Machine learning for dependency parsing

- Dependency parsing is typically cast as supervised learning.
- Sufficient labeled data exists for a wide variety of languages.
  - The CONLL-X Shared Task used datasets from 12 languages.
  - The CONLL 2007 Shared Task used datasets from 10 languages (with three repeats).
  - Labeled data exists for other languages, incl. Hebrew, Latin, Romanian, Thai.
- The CONLL format and evaluation procedure are standard in the community.
Exercise from Lect. 1, RCS(F): Naive Bayes

Features: $\text{POS}(w')$. Class: $\text{POS}(w)$. Labeled data:

<table>
<thead>
<tr>
<th></th>
<th>John</th>
<th>NP</th>
<th>2</th>
<th>John</th>
<th>NP</th>
<th>3</th>
<th>walks</th>
<th>V</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>drives</td>
<td>V</td>
<td>0</td>
<td>is</td>
<td>V</td>
<td>3</td>
<td>and</td>
<td>CONJ</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>cars</td>
<td>NP</td>
<td>2</td>
<td>fast</td>
<td>ADJ</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If single-rooted:

1 John 2
2 drives 0
3 fast 2

\[ P(\leftarrow \text{Root}) \quad 3/11 \quad | \quad P(\leftarrow \text{V}) \quad 6/11 \quad | \quad P(\leftarrow \text{ADJ}) \quad 2/11 \]

\[
\begin{array}{ccc}
\text{d} = \text{NP} & \text{d} = \text{V} & \text{d} = \text{ADJ} \\
\hline
P(d|\leftarrow \text{Root}) & 0/3 & P(d|\leftarrow \text{Root}) & 2/3 & P(d|\leftarrow \text{Root}) & 1/3 \\
P(d|\leftarrow \text{V}) & 3/6 & P(d|\leftarrow \text{V}) & 1/6 & P(d|\leftarrow \text{V}) & 1/6 \\
P(d|\leftarrow \text{ADJ}) & 1/2 & P(d|\leftarrow \text{ADJ}) & 1/2 & P(d|\leftarrow \text{ADJ}) & 0/2 \\
0.09: \text{V} & 0.19: \text{Root} & 0.09: \text{Root}/\text{V} \\
\end{array}
\]
## Dependency parsing, now

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Learner</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaltParser</td>
<td>Transition-based</td>
<td>$O(</td>
</tr>
<tr>
<td>MSTParser</td>
<td>Graph-based</td>
<td>$O(</td>
</tr>
</tbody>
</table>

*Faster on average, since models are smaller.*
## Dependency parsing, now

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Learner</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaltParser</td>
<td>Transition-based</td>
<td>SVM</td>
</tr>
<tr>
<td>MSTParser</td>
<td>Graph-based</td>
<td>MIRA</td>
</tr>
</tbody>
</table>

*Faster on average, since models are smaller.

- Nivre and McDonald (2008) use *stacking* to combine the strengths of the two parsers.
Dependency parsing, now

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Learner</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaltParser</td>
<td>Transition-based</td>
<td>SVM</td>
</tr>
<tr>
<td>MSTParser</td>
<td>Graph-based</td>
<td>MIRA</td>
</tr>
</tbody>
</table>

*Faster on average, since models are smaller.

- Nivre and McDonald (2008) use *stacking* to combine the strengths of the two parsers.
- Martins et al. (2008) use *recursive stacking* to obtain previously best reported results.
**Dependency parsing, now**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Learner</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaltParser</td>
<td>Transition-based</td>
<td>SVM</td>
</tr>
<tr>
<td>MSTParser</td>
<td>Graph-based</td>
<td>MIRA</td>
</tr>
</tbody>
</table>

*Faster on average, since models are smaller.

- Nivre and McDonald (2008) use *stacking* to combine the strengths of the two parsers.
- Martins et al. (2008) use *recursive stacking* to obtain previously best reported results.
- Semisupervised methods have also been used to boost state-of-the-art (Koo et al., 2008; Sagae and Gordon, 2009; Suzuki et al., 2009).
Dependency parsing, now

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Learner</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaltParser</td>
<td>Transition-based</td>
<td>SVM</td>
</tr>
<tr>
<td>MSTParser</td>
<td>Graph-based</td>
<td>MIRA</td>
</tr>
</tbody>
</table>

*Faster on average, since models are smaller.

- Nivre and McDonald (2008) use *stacking* to combine the strengths of the two parsers.
- Martins et al. (2008) use *recursive stacking* to obtain previously best reported results.
- Semisupervised methods have also been used to boost state-of-the-art (Koo et al., 2008; Sagae and Gordon, 2009; Suzuki et al., 2009).
- Søgaard (t.a.) *combines* ensemble-based and semisupervised methods to obtain best reported results.
Søgaard (t.a.)

\[
\begin{array}{ccc}
C_1(F) & C_2(F) & C_3(F) \\
C_1(R_1) & C_2(R_1) & C_3(R_1) \\
C_1(R_1 + U_{2,3}) & C_2(R_1 + U_{1,3}) & C_3(R_1 + U_{1,2}) \\
C_1(R_2) & C_2(R_2) & C_3(R_2) \\
\vdots
\end{array}
\]
## CONLL-X datasets

<table>
<thead>
<tr>
<th>Language</th>
<th>C06</th>
<th>Mar08</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>66.91</td>
<td>69.12</td>
<td>70.12</td>
</tr>
<tr>
<td>Danish</td>
<td>84.79</td>
<td>86.79</td>
<td>86.47</td>
</tr>
<tr>
<td>Dutch</td>
<td>79.19</td>
<td>81.51</td>
<td>81.87</td>
</tr>
<tr>
<td>German</td>
<td>87.34</td>
<td>88.68</td>
<td>89.08</td>
</tr>
<tr>
<td>Japanese</td>
<td>91.65</td>
<td>91.61</td>
<td>92.28</td>
</tr>
<tr>
<td>Portuguese</td>
<td>76.60</td>
<td>88.30</td>
<td>88.76</td>
</tr>
<tr>
<td>Slovene</td>
<td>76.12</td>
<td>76.72</td>
<td>77.98</td>
</tr>
<tr>
<td>Spanish</td>
<td>82.25</td>
<td>83.73</td>
<td>84.67</td>
</tr>
<tr>
<td>Swedish</td>
<td>84.58</td>
<td>85.16</td>
<td>85.92</td>
</tr>
<tr>
<td>Turkish</td>
<td>65.68</td>
<td>65.21</td>
<td>67.42</td>
</tr>
</tbody>
</table>
Logic and machine learning? (2do-list)

Checked items:

- Model-checking is used to verify labeled data.

Non-checked items:
Logic and machine learning? (2do-list)

Checked items:

- Model-checking is used to verify labeled data.
- Crude repair is used to improve parsing quality.

Non-checked items:
Logic and machine learning? (2do-list)

Checked items:

- Model-checking is used to verify labeled data.
- Crude repair is used to improve parsing quality.
- Finally, logic is used to study the properties of linguistic theories (Blackburn and Spaan, 1993; Søgaard, 2007).

Non-checked items:
Logic and machine learning? (2do-list)

Checked items:

- Model-checking is used to verify labeled data.
- Crude repair is used to improve parsing quality.
- Finally, logic is used to study the properties of linguistic theories (Blackburn and Spaan, 1993; Søgaard, 2007).

Non-checked items:

- Transition-based dependency parsing in Cross-style probabilistic modal logic.
Logic and machine learning? (2do-list)

Checked items:

• Model-checking is used to verify labeled data.
• Crude repair is used to improve parsing quality.
• Finally, logic is used to study the properties of linguistic theories (Blackburn and Spaan, 1993; Søgaard, 2007).

Non-checked items:

• Transition-based dependency parsing in Cross-style probabilistic modal logic.
• Learning crude repair functions.
Logic and machine learning? (2do-list)

Checked items:

- Model-checking is used to verify labeled data.
- Crude repair is used to improve parsing quality.
- Finally, logic is used to study the properties of linguistic theories (Blackburn and Spaan, 1993; Søgaard, 2007).

Non-checked items:

- Transition-based dependency parsing in Cross-style probabilistic modal logic.
- Learning crude repair functions.
- Modal characterizations of mildly non-projective dependency grammars.
Logic and machine learning? (2do-list)

Checked items:

- Model-checking is used to verify labeled data.
- Crude repair is used to improve parsing quality.
- Finally, logic is used to study the properties of linguistic theories (Blackburn and Spaan, 1993; Søgaard, 2007).

Non-checked items:

- Transition-based dependency parsing in Cross-style probabilistic modal logic.
- Learning crude repair functions.
- Modal characterizations of mildly non-projective dependency grammars.
- Model-checking polyadic dynamic logic.
Data-driven dependency parsing in collaborative research projects at CST

- **Question answering:**
  - MOSES (university websites); led by Patrizia Paggio.
  - ESICT (patient diagnosis); led by Bente Maegaard.

- **Machine translation:**
  - ESSMT (practical); led by me.
  - EMCOTT (theoretical; under review); led by Jürgen Wedekind and me.