A generative re-ranking model for dependency parsing

Federico Sangati, Willem Zuidema and Rens Bod

INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION
University of Amsterdam

November 9, 2009

Discriminative vs. Generative models

Discriminative Models

- Parsing as a classification task.
- Transition-based parsers. (Nivre and Hall , 2005)
- Graph-based parsers. (McDonald, 2006)
- STATE-OF-THE-ART! (Buchholz et al., 2006; Nivre et al., 2007)

Probabilistic Generative Models

- Define probabilities over structures. (Eisner, 1996)
- Perform more poorly... although not much represented in the last evaluation challenges.
- Very important for many NLP tasks (SR, MT, NLG, ...): need probabilities.

The idea

Is there a principled way of combining the two?

- Discriminative model provides the k-best candidates.
- Generative model computes the prob. of each candidate.
- Selects the one with max. probability (re-ranking).
- Generative model trained on the training corpus bus NOT on the output of the discriminative model.

Motivation

- Implement and compare different generative models...
- without implementing different parsers (we actually don't need any parser).
- 'Parser simulator'[a] methodology.

[a] Reut Tsarfaty terminology

Probabilistic Generative Models

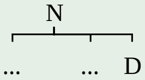
Decomposition

Reverse the process: we can decompose any given structure into **events** and corresponding **conditioning contexts.**

Example

A generative model chooses each dependent D of a node N conditioned on N and their relative position (left, right).

$$P\ (\ D\ \mid\ N\ \ \text{direction}\ )$$



Event : D is a right dependent of N. (**D N R**)

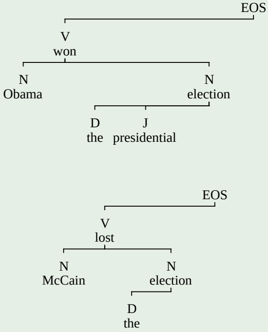
Conditioning context : N has a right dependent. (**N R**)

Training phase

Decomposition

Decompose each dependency structure in the training corpus, and keep track of the frequency of each event and conditioning context.

Training corpus



$$P\,(\;D \;\mid\; N \;\;\text{direction}\;)$$

Events	**Freq.**	**Cond. Contexts**	**Freq.**
won *EOS* L	1	*EOS* L	4
lost *EOS* L	1		
STOP EOS L	2		
STOP EOS R	2	*EOS* R	2
Obama won L	1	won L	2
STOP won L	1		
STOP Obama L	1	Obama L	1
STOP Obama R	1	Obama R	1
⋮		⋮	

Re-ranking phase

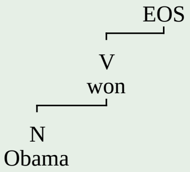
Decomposition

A given candidate structure can be decomposed into:

- events $(e_1, e_2, \ldots, e_n)$
- conditioning contexts $(c_1, c_2, \ldots, c_n)$.

The probability of the structure: $$\prod_{i=1}^{n} \frac{f(e_i)}{f(c_i)}$$

Test structure



$$P\ (\ D\ \mid\ N\ \ \text{direction}\ )$$

Events	Freq.	Cond. Contexts	Freq.	$f(e_i)/f(c_i)$
won *EOS* L	1	*EOS* L	4	1/4
STOP EOS L	2	*EOS* L	4	1/2
STOP EOS R	2	*EOS* R	2	1
Obama won L	1	won L	2	1/2
STOP won L	1	won L	2	1/2
STOP Obama L	1	Obama L	1	1
STOP Obama R	1	Obama R	1	1
				1/32

Probabilistic Generative Models

Important

The only thing to define: how a generative model decomposed a structure into events.

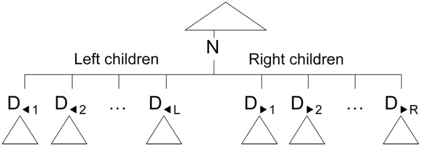
Provided

- a way of decomposing a given structure into events,
- a consistent way of representing them

both training and re-ranking phases can be **performed identically for many different generative models**.

Eisner model

Generative model inspired by the work of Eisner, 1996.



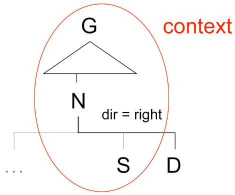
- Nodes are generated recursively in a **top-down** manner.
- Left and right children are generated as two separate **Markov sequences** of nodes, each conditioned on sibling and ancestral information (*context*).

$$P(T(N)) = \prod_{l=1}^{L} P(D_{\triangleleft l})|context) \cdot P(T(D_{\triangleleft l}))$$
$$\times \prod_{r=1}^{R} P(D_{\triangleright r})|context) \cdot P(T(D_{\triangleright r}))$$

The feature space

$$P(D|context) =$$

$$P(\underset{\text{distance}}{\underset{\uparrow}{dist(N,D)}}, \underset{\text{is terminal?}}{\underset{\uparrow}{term(D)}}, word(D), tag(D) | N, \underset{\text{previous sister}}{\underset{\uparrow}{S}}, \overset{\text{grand parent}}{\overset{\downarrow}{G}}, dir)$$



Breaking down

$$\begin{aligned} P(dist(N,D), term(D), word(D), tag(D) | N, S, G, dir) &= \\ P(tag(D) | H, S, G, dir) &\times \\ P(word(D) | tag(D), H, S, G, dir) &\times \\ P(term(D) | word(D), tag(D), H, S, G, dir) &\times \\ P(dist(P,D) | term(D), word(D), tag(D), H, S, G, dir) & \end{aligned}$$

Backoff

$P(tag(D) | H, S, G, dir)$

reduction list:

$wt(H), wt(S), wt(G), dir$
$wt(H), wt(S), t(G), dir$
$\left\{ \begin{array}{l} wt(H), t(S), t(G), dir \\ t(H), wt(S), t(G), dir \end{array} \right.$
$t(H), t(S), t(G), dir$

$P(word(D) | tag(D), H, S, G, dir)$

reduction list:

$wt(H), t(S), dir$
$t(H), t(S), dir$

$P(term(D) | word(D), tag(D), H, S, G, dir)$

reduction list:

$tag(D), wt(H), t(S), dir$
$tag(D), t(H), t(S), dir$

$P(dist(P,D) | term(D), word(D), tag(D), H, S, G, dir)$

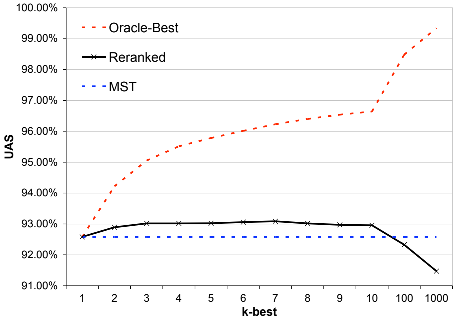
reduction list:

$word(D), tag(D), t(H), t(S), dir$
$tag(D), t(H), t(S), dir$

Results

Unlabeled Parsing

- Corpus: **Penn WSJ-40** converted to dependency structure according to Collins (1999).
- Training/Test: sec 02-21 / sec 22 (gold pos-tags)
- **UAS**: Unlabeled attachment score
- Discriminative model: **MST parser**, 2[nd] order (McDonald, 2006)



k-best	Oracle best	Oracle worst	Reranked
1	**92.58**	92.58	92.58
2	94.22	88.66	92.89
3	95.05	87.04	93.02
4	95.51	85.82	93.02
5	95.78	84.96	93.02
6	96.02	84.20	93.06
7	96.23	83.62	**93.09**
8	96.40	83.06	93.02
9	96.54	82.57	92.97
10	96.64	82.21	92.96
100	98.48	73.30	92.32
1000	99.34	64.86	91.47

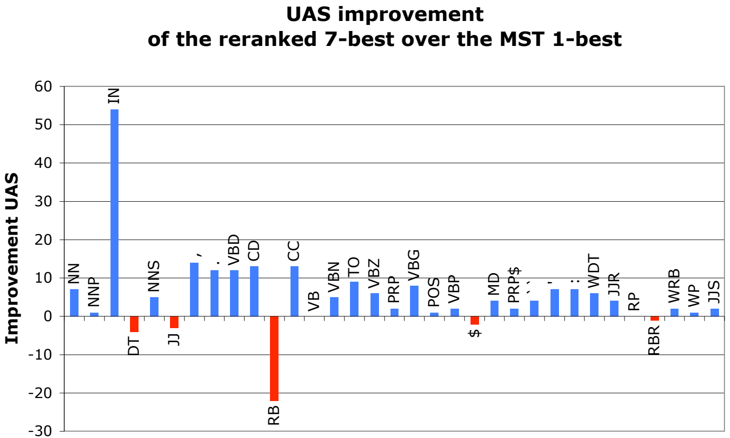
Conclusions

- Combining discriminative and generative models: improvements over state-of-the-art results.
- Open question: can we come up with a better generative model?
- Efficiency:
    - MST parser: training + parse 1-best test $\rightarrow$ 6 h.
    - Our method: training + re-ranking 100-best $\rightarrow$ 5 min!
- 'Parser simulator': efficient framework to evaluate many different generative models.
- Explore different feature spaces.

Thank you!

http://staff.science.uva.nl/~fsangati

{f.sangati,zuidema,rens.bod}@uva.nl


UAS improvement
of the reranked 7-best over the MST 1-best
Improvement UAS
60
50
40
30
20
10
0
-10
-20
-30
NN
NNP
IN
DT
NNS
JJ
,
.
VBD
CD
RB
CC
VB
VBN
TO
VBZ
PRP
VBG
POS
VBP
$
MD
PRP$
``
''
:
WDT
JJR
RP
RBR
WRB
WP
JJS