Efficient, Feature-based, Conditional Random Field Parsing

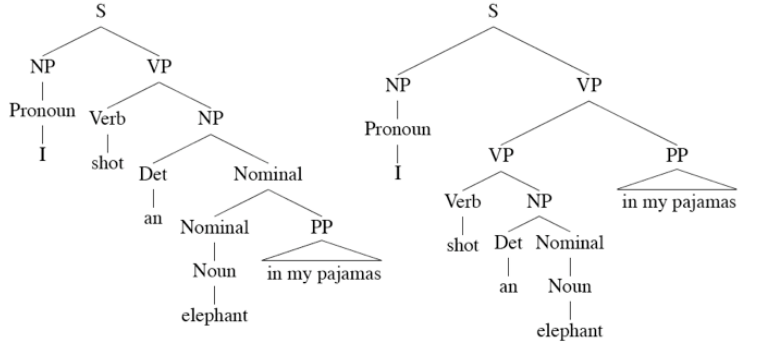
Paper by: Jenny Rose Finkel, Alex Kleeman, Christopher D. Manning

Published at ACL, 2008

Presentation by: Michael Zhang

Overview

Constituency Parsing



Objective

Given some sentence S, assign some parse tree T.

Overview

Objective

Given some sentence S, assign some parse tree T.

- Parsing models at the time were dominated by **generative** methods
- However, **discrimitive** models had been shown to outperform generative models in other NLP tasks
  - Discrimitive models have not surpassed generative models due to the computational complexity of the task

Overview

Prior work on discrimitive parsing fell into 3 categories:

- Reranking n-best outputs from a generative parser
- Parse by making a series of independent, discriminative decisions using either greedy search or beam search.
- Perform joint inference via dynamic programming algorithms for training and to find the globally best parse
  - Previous work in this vein has been limited to shorter sentences, or giving up on features

Overview

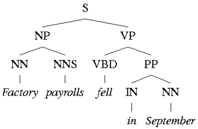
Contribution

- Created a feature based, discriminative model for parsing
- Made a practical CRF model that could be easily trained and perform on longer sentences

The model: CRF-CFG

Context Free Grammar

Consits of set of terminals $\{w^k\}$, non-terminals $\{N^k\}$, start symbol $\{ROOT\}$, rules $\{\rho = N^i \rightarrow \zeta^i\}$



(a) PCFG Structure

Phrasal rules

$r_1 = \mathrm{S}_{0,5} \rightarrow \mathrm{NP}_{0,2} \quad \mathrm{VP}_{2,5}$ | *Factory payrolls fell in September*

$r_3 = \mathrm{VP}_{2,5} \rightarrow \mathrm{VBD}_{2,3} \quad \mathrm{PP}_{3,5}$ | *Factory payrolls fell in September*

...

Lexicon rules

$r_5 = \mathrm{NN}_{0,1} \rightarrow$ *Factory* | *Factory payrolls fell in September*

$r_6 = \mathrm{NNS}_{1,2} \rightarrow$ *payrolls* | *Factory payrolls fell in September*

...

(b) Rules r

CRF-CFG

- Defines local potentials $\phi(r|s;\theta)$

$$P(t|s;\theta) = \frac{1}{Z_s} \prod_{r \in t} \phi(r|s;\theta)$$

$$Z_s = \sum_{t \in \tau(s)} \prod_{r \in t} \phi(r|s;\theta)$$

The model: CRF-CFG

The Objective Function

- clique potential function:

$$\phi(r|s;\theta) = exp \sum_i \theta_i f_i(r,s)$$

- Log conditional likelihood of training data $\mathcal{D}$ (with L_2 regularization term)

$$\mathcal{L}(\mathcal{D};\theta) = (\sum_{(t,s)\in\mathcal{D}} (\sum_{r\in t} \sum_i \theta_i f_i(r,s)) - Z_s) + \sum_i \frac{\theta_i^2}{2\sigma^2}$$

- Partial derivatives of log likelihood:

$$\frac{\partial\mathcal{L}}{\partial\theta_i} = (\sum_{(t,s)\in\mathcal{D}} (\sum_{r\in t} f_i(r,s)) - E_\theta[f_i|s]) + \frac{\theta_i}{\sigma^2}$$

Training and Optimizations

Z_s and $\dfrac{\partial \mathcal{L}}{\partial \theta_i}$ can both be efficiently computed in polynomial time with the inside-outside algorithm
(substituting non-negative potentials (ϕ) for probalilities)

Optimizations

- Parallelizaiton
- Chart Prefiltering
- Stochastic Optimizaition

Training and Optimizations

Parallelization

- Log likelihood and Partial Derivatives can be computed by summing over each tree individually
- Stochastic optimization methods mean they only compute the objective for a small number of sentences at a time (15-30)
- Clients compute relevant information for each sentence, then pass it to a a centeral server aggrigating data from each.
- **Bottleneck:** They note that the benefits of adding clients decreases rapidly as computation time is dominated by the longest sentence for each batch.

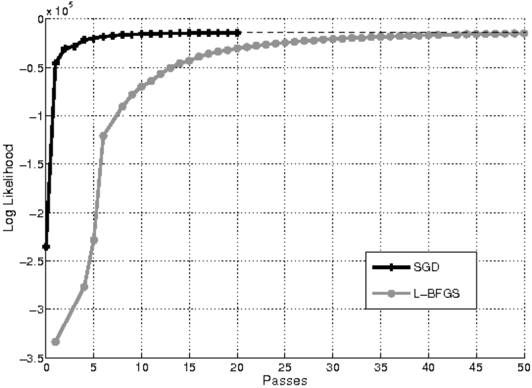
Training and Optimizations

Chart Prefiltering

- Not all rule decisions can properly tile the tree
- To avoid performing computations for these invalid trees, they prefilter on inside passes of the inside-outside algorithm
- They compute this information once by passing booleans instead of potentials, simotaneously calculating features for possible rules and save the entire datastructure to disk
- Allows them to avoid recalculating even on multiple passes through the data
- **3x speedup** on first iteration, **10x speedup** on successive iterations

Training and Optimizations

Stochastic Optimization



- Use SGD for optimization
- compared SGD to L-BFGS
- **7x speedup** on WSJ15

Features

n word class.

Lexicon Features	Grammar Features	
t		Binary-specific features
$b(t)$	ρ	
$\langle t, w \rangle$	$\langle b(p(r_p)), ds(w_s) \rangle$	$\langle b(p(r_p)), ds(w_{s-1}, dsw_s) \rangle$
$\langle t, lc(w) \rangle$	$\langle b(p(r_p)), ds(w_e) \rangle$	**PP feature:**
$\langle b(t), w \rangle$	**unary?**	if right child is a PP then $\langle r, w_s \rangle$
$\langle b(t), lc(w) \rangle$	**simplified rule:**	**VP features:**
$\langle t, ds(w) \rangle$	base labels of states	if some child is a verb tag, then rule,
$\langle t, ds(w_{-1}) \rangle$	**dist sim bigrams:**	with that child replaced by the word
$\langle t, ds(w_{+1}) \rangle$	all dist. sim. bigrams below	
$\langle b(t), ds(w) \rangle$	rule, and base parent state	Unaries which span one word:
$\langle b(t), ds(w_{-1}) \rangle$	**dist sim bigrams:**	
$\langle b(t), ds(w_{+1}) \rangle$	same as above, but trigrams	$\langle r, w \rangle$
$\langle p(t), w \rangle$	**heavy feature:**	$\langle r, ds(w) \rangle$
$\langle t, unk(w) \rangle$	whether the constituent is "big"	$\langle b(p(r)), w \rangle$
$\langle b(t), unk(w) \rangle$	as described in (Johnson, 2001)	$\langle b(p(r)), ds(w) \rangle$

- Their model allowed them to incoperate "lexicon features" (words over tags) and "grammar features" (local subtrees and corresponding span/split)
- Features had to be tuned on sentences ≤ 15 because ≤ 40 was infesible

Experiments

Model	P	R	F_1	Exact	Avg CB	0 CB	P	R	F_1	Exact	Avg CB	0 CB
	development set – length ≤ 15						test set – length ≤ 15					
Taskar 2004	89.7	90.2	90.0	–	–	–	89.1	89.1	89.1	–	–	–
Turian 2007	–	–	–	–	–	–	89.6	89.3	89.4	–	–	–
generative	86.9	85.8	86.4	46.2	0.34	81.2	87.6	85.8	86.7	49.2	0.33	81.9
discriminative	89.1	88.6	88.9	55.5	0.26	85.5	88.9	88.0	88.5	56.6	0.32	85.0
feature-based	90.4	89.3	89.9	59.5	0.24	88.3	91.1	90.2	90.6	61.3	0.24	86.8
relaxed	91.2	90.3	90.7	62.1	0.24	88.1	91.4	90.4	90.9	62.0	0.22	87.9

Table 3: Development and test set results, training and testing on sentences of length ≤ 15 from the Penn treebank.

Model	P	R	F_1	Exact	Avg CB	0 CB	P	R	F_1	Exact	Avg CB	0 CB
	test set – length ≤ 40						test set – all sentences					
Petrov 2007	–	–	88.8	–	–	–	–	–	88.3	–	–	–
generative	83.5	82.0	82.8	25.5	1.57	53.4	82.8	81.2	82.0	23.8	1.83	50.4
generative-all	83.6	82.1	82.8	25.2	1.56	53.3	–	–	–	–	–	–
discriminative	85.1	84.5	84.8	29.7	1.41	55.8	84.2	83.7	83.9	27.8	1.67	52.8
feature-based	89.2	88.8	89.0	37.3	0.92	65.1	88.2	87.8	88.0	35.1	1.15	62.3

Table 4: Test set results, training on sentences of length ≤ 40 from the Penn treebank. The *generative-all* results were trained on all sentences regardless of length

- Discriminatively trained model: Lexicon Featrues, no Grammer Features
- Feature-based model: Lexicon Featrues and Grammer Features
- Relaxed model: Feature model with rules not seen in training

Experiments

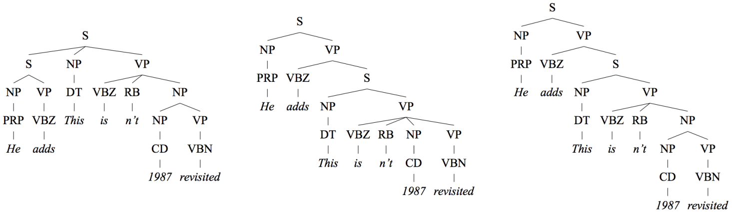
Model	P	R	F_1	Exact	Avg CB	0 CB	P	R	F_1	Exact	Avg CB	0 CB
	test set – length ≤ 40						test set – all sentences					
Petrov 2007	–	–	88.8	–	–	–	–	–	88.3	–	–	–
generative	83.5	82.0	82.8	25.5	1.57	53.4	82.8	81.2	82.0	23.8	1.83	50.4
generative-all	83.6	82.1	82.8	25.2	1.56	53.3	–	–	–	–	–	–
discriminative	85.1	84.5	84.8	29.7	1.41	55.8	84.2	83.7	83.9	27.8	1.67	52.8
feature-based	89.2	88.8	89.0	37.3	0.92	65.1	88.2	87.8	88.0	35.1	1.15	62.3

Table 4: Test set results, training on sentences of length ≤ 40 from the Penn treebank. The *generative-all* results were trained on all sentences regardless of length

Performance

- On WSJ15: (20 passes)
  - Discriminatively trained generative model (*discriminative*): 1 machine; 3 gigabytes of RAM; 12 min/pass
  - Feature-based model (*feature-based*): 1 machine; 3 gigabytes of RAM; 35 min/pass
- On WSJ40: (10 passes)
  - Discriminatively trained generative model (*discriminative*): 2 machines; 16 gigabytes of RAM each; 1 day/pass
  - Feature-based model (*feature-based*): 4 machines; 16 gigabytes of RAM each; 3 day/pass

Experiments

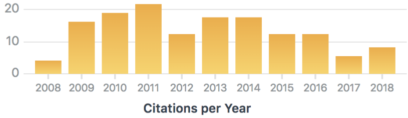


(a) generative output (b) feature-based discriminative output (c) gold parse

Figure 3: Example output from our generative and feature-based discriminative models, along with the correct parse.

- They highlight the models ability to capture the right-branching tendencies of English
- Specifically the "heavy" feature encouraging long constituants at ends of sentences

Impact



146 Citations

Semantic Scholar estimates that this publication has **146** citations based on the available data.

See our **FAQ** for additional information.

Why was this work influential?

- Defined a discriminative, feature based model that's simple, effective, and faster to train than previous methods.
- "Looking at how other tasks, such as named entity recognition and part-of-speech tagging, have evolved over time, it is clear that greater gains are to be gotten from developing better features than from better models."

Thanks! Question?