Ling/CSE 472: Introduction to Computational Linguistics

5/20/15

Statistical Parsing

Overview

- Why statistical parsing?
- PCFGs
- Estimating rule probabilities
- Probabilistic CKY
- Ways to improve PCFG
- Reading questions

Why statistical parsing?

- Parsing = making explicit structure that is inherent (implicit) in natural language strings

- Useful for: language modeling + any app that needs access to the meaning of sentences

- Most application scenarios that use parser output want just one parse
    - Have to choose among all the possible analyses

- Most application scenarios need robust parsers
    - Need some output for every input, even if its not grammatical

PCFGs

- N: a set of non-terminal symbols
- Σ: a set of terminal symbols (disjoint from N)
- R: a set of rules, of the form A -> β [p]
    - A: non-terminal
    - β: string of symbols from Σ or N
    - p: probability of β given A
- S: a designated start symbol

PCFGs

- How does this differ from CFG?
- How do we use it to calculate the probability of a parse?
- The probability of a sentence?
- What assumptions does that require?

PCFGs

- How does this differ from CFG? -- added probability to each rule
- How do we use it to calculate the probability of a parse? -- multiply probability of each rule used (= P(T|S) = P(T))
- The probability of a sentence? -- sum of probability of all trees
- What assumptions does that require? -- expansion of a node does not depend on the context

PCFGs: Why

- When would you want to know the probability of a parse?
- When would you want to know the probability of a sentence?

How to estimate the rule probabilities

- Get a Treebank
- Gather all instances of each non-terminal
- For each expansion of the non-terminal (= rule), count how many times it occurs

$$P(\alpha \rightarrow \beta \mid \alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)}$$

Using the probabilities for best-first parsing

- Probabilistic CKY: in each cell, store just the most probable edge for each non-terminal

- Probabilities based on rule probability plus daughter edge probabilities

function PROBABILISTIC-CKY(*words*,*grammar*) **returns** most probable parse
and its probability

for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
  for all $\{ A \mid A \rightarrow words[j] \in grammar \}$
    $table[j-1, j, A] \leftarrow P(A \rightarrow words[j])$
  for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
    for $k \leftarrow i+1$ **to** $j-1$ **do**
      for all $\{ A \mid A \rightarrow BC \in grammar,$
        and $table[i,k,B] > 0$ **and** $table[k,j,C] > 0 \}$
        if $(table[i,j,A] < P(A \rightarrow BC) \times table[i,k,B] \times table[k,j,C])$ **then**
          $table[i,j,A] \leftarrow P(A \rightarrow BC) \times table[i,k,B] \times table[k,j,C]$
          $back[i,j,A] \leftarrow \{k,B,C\}$
  return BUILD_TREE($back[1, $ LENGTH(*words*), $S]$), $table[1, $ LENGTH(*words*), $S]$

Work through an example: Kim adores snow in Oslo

$\mathrm{S} \rightarrow \mathrm{NP\ VP}$	$[.8]$	$\mathrm{NOM} \mid \mathrm{NP} \rightarrow \mathrm{Kim}$	$[.01]$
$\mathrm{VP} \rightarrow \mathrm{V\ NP}$	$[.2]$	$\mathrm{NOM} \mid \mathrm{NP} \rightarrow \mathrm{snow}$	$[.01]$
$\mathrm{VP} \rightarrow \mathrm{VP\ PP}$	$[.3]$	$\mathrm{NOM} \mid \mathrm{NP} \rightarrow \mathrm{Oslo}$	$[.01]$
$\mathrm{PP} \rightarrow \mathrm{P\ NP}$	$[.9]$	$\mathrm{V} \mid \mathrm{VP} \rightarrow \mathrm{adores}$	$[.02]$
$\mathrm{NP} \rightarrow \mathrm{NOM\ PP}$	$[.2]$	$\mathrm{V} \mid \mathrm{VP} \rightarrow \mathrm{snores}$	$[.01]$
		$\mathrm{P} \rightarrow \mathrm{in}$	$[.1]$

Why statistical parsing? (reprise)

- Most application scenarios that use parser output want just one parse
  - Have to choose among all the possible analyses
  - How does PCFG solve this problem?
- Most application scenarios need robust parsers
  - Need some output for every input, even if its not grammatical
  - How does PCFG solve this problem?

Problems with PCFG

- Independence assumption is wrong
  - What does “independence assumption” mean?
  - What is the evidence that it’s wrong?
- Not sensitive to lexical dependencies
  - What does that mean?

Ways to improve PCFGs

- Split the non-terminals
    - Rename each non-terminal based on its parent (NP-S vs. NP-VP)
    - Hand-written rules to split pre-terminal categories
        - Automatically search for optimal splits through split and merge algorithm
- Lexicalized PCFGs: add identity of lexical head to each node label
    - Data sparsity problem -> smoothing again

Overview

- Why statistical parsing?
- PCFGs
- Estimating rule probabilities
- Probabilistic CKY
- Ways to improve PCFG
- Reading questions

Reading Questions

- The book states that PCFG's don't allow for rule probability to be conditioned on surrounding context. Can you explain this in more detail? Does this mean n-gram probabilities can't be included on some level of the PCFG? Why not?

- Do PCFGs rely on a preset dictionary of terminal symbols to grammatical category? Would it be possible to train a PCFG if it did not know the category words belong to initially (e.g. the = determiner)?

Reading Questions

- In the "Probabilistic lexicalized CFG" part, why is it effective to condition the probability of a rule on the lexical head or nearby heads?

- What type of training data is needed for lexicalized CFG parsers? It seems like they need very specific and very in-depth data compared to other types of parsers, but I'm not totally clear on what exactly that is. Do they need the probabilities of both words (n-grams) and of syntactic structures?

Reading Questions

- I don't understand the generation method for the Collins Parser. I understand the purpose of generating to the left/right, but am confused as to how the probabilities at the end are determined from there.

- I am a little confused by the Collins Parser. Is it working like an N-Gram that just takes into account what comes before and after it? Does the tree structure really play a role in the resulting probability?

Reading Questions

- Does a parsing algorithm such as inside-outside choose sentences that are semantically more likely to happen as well? Isn't this algorithm inefficient when it comes to the time it takes to parse data/ how much do we prioritize efficiency over correctness?

- It seems like probabilistic parsers require a lot of time and stored information to work properly. How is it that there are applications using probabilistic parsing that are able to run in resource restricted environments like cell phones?

Reading Questions

- "humans prefer whichever parse is more probable" can I understand the probability of a parse as how frequently does it appear in real-life use?

- In the sentence "the time to recognize a word is influenced by entropy of the word and the entropy of the word's morphological paradigm", how should we understand the idea of "entropy"?

Reading Questions

- Why is it that 5-gram grammars do better at language modeling than parser-based language models given a large amount of data? It seems like having access to the syntactic dependencies in language would be an advantage for the parser-based model.