Set up CMS!

Admin-meeting:

- [ ] today
- [X] 11/8
- [ ] 11/12
- [ ] 11/14
- [ ] 11/9
- [ ] 11/11

11/13/14
Lec 23.

- Do we want a "hackathon" next week (Tues, say) to organize any tasks that can be done jointly? (probably not a "hackathon" per se, but people might want to coordinate.)
- Since already passed some code along, as did Longi.
- Do we want in-class presentations? (5 groups, standing?)
- Pragmed to get initial input.

I think we're possible already past the last possible day when this meeting will be useful.

- I think 11/18-11/20

One possibility:

- 11/4-11/6: hand in Ch. 1
- 11/6: lecture
- 11/8: Ch. 2
- 11/12: Ch. 3
- 11/14: Ch. 4
- 11/19: project presentation
- 11/26: final

(If we don't)

Another:

- 11/4-11/6: hand in Ch. 1
- 11/6: lecture
- 11/8: Ch. 2
- 11/12: Ch. 3
- 11/14: Ch. 4
- 11/19: project presentation
- 11/26: final

30
150 = 75 x 2!!

- reading
- summarizing
- preparing
- things you can use:
- 1/15
- what's the diff.
- from Stanford
- TM toolkit:
  - Melod?
  - new tools:
  - bioregular genome
  - permission data.
  - Stanford paper:
    - referential relata
    - (p. 347)
    - high-card:
      - depending
Recall: I mentioned that this paper is really worth reading line-by-line for being inventive about finding new features for a new problem, and for being quite detailed.

Remember this problem:

It should serve as a source of inspiration for you when you try to think of interesting features for your interesting problems.

Today, I want to first quickly point at some tools/techniques of immediate potential use to you, but more I'm going to use the paper as a jumping-off point to at least touch on some more advanced topics.

3.2 "use of people"

- named-entity recognition - active area of research -
  - Ar. anwers [example from JT/M - AMR only unlikely to be in your dictionary]
  - [Stanford NER tool] (other - see resources mentioned on course webpage)
    - has a NE-model with 7 classes, time, location, money, parent, date
  
You may want to be able to identify these.

3.3 "beautifu language" - unusual word combinations

you should care b/c this may be relevant to your projects
(a talk on "what makes two types of long defn."
  - fightin' words - n-grams last time
  - her, syntax is taken into account... although really as a way to
    limit interesting n-grams.

<adjacent nouns as still n-grams> -> she's exgls on same ps.

Note reference to 2 defn. types of papers.
CFG-style:

Using Jurafsky & Martin textbook slides 9 (ch 14)
(hide irrelevant non-used slides, to make things easier to find)

- Slide 7: tree structure

  - Book [the brown flight]
  - Book [the brown flight]

  (explain that if "book [the brown] flight" is ok, so is a flight: "flight"
  (adjacent nouns in noun phrases, as before). Newkens are interested in
  whether these are interchangeable, but the 2nd is a possible parse.

- Constituents as labeled subtrees:

- Slide 5: CFG rules

  (can we blow up the "outline" page so there can be
  put side-by-side)

- Known parsing alg's for this

  - Relations btw constituents

  (skip category-splitting à la Klein, Manning, for time)

- But lexical info is important

- Slide 20: "into" acts differently than "of" w.mt attachment points "sick" vs. "caught"

- Slide 22: track lexical items + heads of phrases

Dependency-style: direct relation btw lexical items

- Fig 12.14 of J; M.

- Linear-time deterministic parsers - MALT parser, for example.

- Entropy characterizing a distribution

- $H \eta : \text{characterizing a distribution}

- $H \eta : \sum_i \eta_i \log \left( \frac{1}{\eta_i} \right)$

  "surprised"

  - Suppose: show $\log(0) = 0$ by limiting arguments
A reconstruction of what we talked about in class with accessory notes.

- We have intuitions that there are substructures to language sentences (sorts like how we inferred the presence of hidden substructures in discourse).

- Subjects / predicates, modifiers / modifies, etc.

If we assume some sort of grammar describing the possible structures, then “parsing” is the process of recovering the possible structures of a sentence w.r.t. that grammar.

CFGs: constituents are different types (noun phrases NP, verb phrases VP, adjectives ADJ, etc.) of word subsequences; [lexical items on a disjoint finite set of words,] con constitutents distinguished start symbol S).

- think of them as depth-1 branches:

$$S \rightarrow NP \rightarrow VP \rightarrow \ldots$$

- a sentence can be made up of an NP then a VP

$$VP \rightarrow \ldots$$

([pro])

Valid trees have each branch being part of the form one of the finite set of "branch" rules, plus all the leaves are lexical items, and root is the distinguished start-symbol.

Sounds quite reasonable, but here’s one problem in practice:

- (Except in grade school), “volunteer” is an intransitive verb, (no direct object)

So we expect trees like this:

But since our lang also has transitive verbs:

We must have a rule for allowing a direct object:
... which is bad, b/c "volunteer" should not take a
direct object.

A technical fix is to say that there isn't just one category "VP", but to say
there are two types, "VP-trans" and "VP-intrans", where we wouldn't allow

and we can do this, although category proliferation is going to be a bit of a beast
that one will have to manage. And then are other issues,

<see posted notes from previous CS6740>
"also we didn't talk about"
"lexicalization should"

<one approach is feature-based context-free grammars>.

But sticking with CFGs has the advantage that there are parsing algorithms that
recover all possible structures for a sentence of length n in $O(n^3)$.
(implicitly)

Alternatively, we can move beyond CFGs, to formalisms where we take a hit in
parsing time, but get some significant linguistic modeling elegance in return.

"step (1): what about letting our rules be not single branches, but
larger sub-trees?"

"transfer substitution grammars"

"rules":

```
S --> NP VP
NP --> Det N
VP --> V NP
```

"combination" is substitution: of a tree with root labeled by some category X
into a tree's leaf when that leaf has the same label X.

So, clearly CFG's are a special case.

On this, I was wrong. In class, I wrongly stated that tree insertion grammars are
the same. They are not, but allow certain types of adjunction.
TSGs are weakly equivalent to CFGs: they generate the same set of strings. But who cares about generation? We're generally in a setting where we've been given some text; we have to analyze it.

It's the analysis, we care about.

TSGs are not strongly equivalent to CFGs: they generate different analysis sets.

E.g.: a TSG w/ only the "rule": $S \rightarrow S \cdot a$
generates only one tree structure.

An strongly equivalent CFG would have to have the rules $S \rightarrow S \cdot a$.

But then it must also allow the analysis:

\[
\begin{array}{c}
S \\
\downarrow \\
S \\
\downarrow \\
S \\
\downarrow \\
a
\end{array}
\]

\[ a \quad 2^\text{nd} \text{ tree. Contradicts strong equivalence, since it should only allow a single tree.} \]

You might also note that for TSGs, the derived tree—the analysis—doesn't match the derivation tree: the description of which rules were used to make the derived tree.

This would seem to complicate parsing. But since TSG is parseable in time cubic in string length, TSGs must be as well.

Aside: by the way, if checking whether a string is "legal" or not as fast as possible is your goal, you might want to go to string less complex than CFGs.

Finite-state grammars can be encoded as right-branching CFGs:

Finite-state models have this "flavor", and you have linear-time recognition.
This is an addition of new tree type: new operation (adjunction).

But some nice important linguistically-oriented properties, and some against that.

...
Idea: so you have some box structure already:

\[
S \rightarrow NP \rightarrow VP
\]

That is ADJP

Is bad

And then you'd like to modify:

Adjunction:

bar structure

split remove \( \times \) and replace w/ adjunction tree:

Now, where does the leftover \( \times \) go? That's what the foot is for!

Contrast what you'd have to do w/ a CFG or a TSG: b/c you need a leaf node avail to add "mildly" into, you have to decide from this stage:

Whether to do the modification, which is non-intuitive that the decision to insert "mildly" is done before you decide what "mildly" is going to modify ("bad")!