CS4740/5740
Introduction to Natural Language Processing

Final Exam Solutions
May 13, 2014

1 Parsing

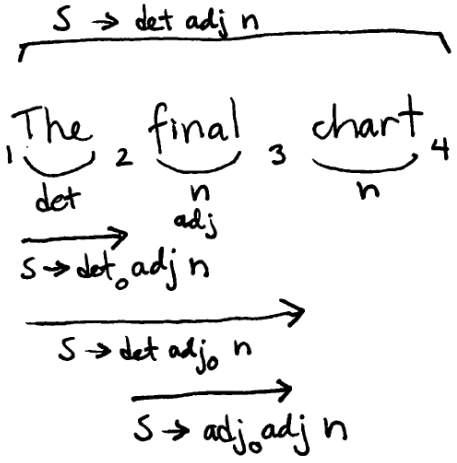

Figure 1: Final bottom-up chart.

1. See Figure 1.
2. See Figure 2.
3. The Earley algorithm incorporates (limited) top-down predictions to prevent the consideration of any complete edge (i.e., a part-of-speech for a token, a constituent type for a phrase) unless it was expected to occur at that position in the sentence (according to the top-down edges introduced into the chart).
4. (a) The specific probabilities don't matter; for each constituent type, they just need to add to 1.

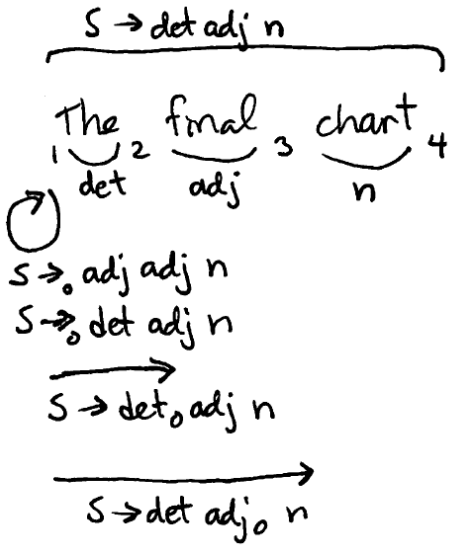

Figure 2: Final Earley chart.

production rule	probability
S → VP	1.0
VP → Verb NP	0.7
VP → Verb NP PP	0.3
NP → NP PP	0.3
NP → Det Noun	0.7
PP → Prep Noun	1.0

(b) We need a different PP attachment in each sentence, but the grammar and parser force the PP attachment to be the same in each case. Semantically, "with the binoculars" should attach to (i.e., modify) the noun phrase "the man" in the second sentence (since you can't use binoculars to call someone); in the first sentence, the PP attachment is ambiguous — the sentence's interpretation would depend on the context — but most readers prefer the reading that attaches "with the binoculars" to the verb (i.e., saw with the binoculars).

(c) Lexicalization of production rules can capture lexical specific preference of certain rule expensions. In order to mitigate the sparse data problem, we will lexicalize with respect to the head word of the left hand side of each production rule, instead of all nonterminals in each

production rule. In particular, the rules expending from VP should be modified as

production rule	probability
VP $(x) \rightarrow$ Verb NP	(p_x)
VP $(x) \rightarrow$ Verb NP PP	(q_x)

where
$x \in$ { Cut, Ask, Find, ... },
$p_x \stackrel{\text{def}}{=}$ P(VP $(x) \rightarrow$ Verb NP | VP, x),
$q_x \stackrel{\text{def}}{=}$ P(VP $(x) \rightarrow$ Verb NP PP | VP, x),
and $p_x + q_x = 1$.

Comment

Because we didn't restrict lexicalization to head words of the right hand side of rules, it is okey to propose lexicalized PCFGs in many different ways; in particular, you don't have to condition on the head word, you can condition on the entire combination of words for all non-terminals, as long as you made it clear what you are conditioning on, although it would be much less practical.

2 HMMs and Part-of-Speech Tagging

- Given that w_i is the ith word, and t_i is the ith part-of-speech tag, what we want here is $P(t_i|t_{i-1})$, estimated as $\frac{count-of(t_i,t_{i-1})}{count-of(t_{i-1})}$
  $P(V|N) = \frac{3}{9}$
  $P(N|ADJ) = \frac{4}{4}$
  $P(,|N) = \frac{2}{9}$
  $P(.|N) = \frac{3}{9}$
  $P(N|V) = \frac{1}{3}$
  $P(N|N) = \frac{1}{9}$
  $P(N|DET) = \frac{1}{3}$
  If you don't include sentence-boundary tokens, then $P(N|.) = \frac{1}{3}$
  If you do, then $P(N| < s >) = \frac{2}{3}$

Many students gave bigram probabilities for words, either as the joint $P(current - word \wedge previous - word)$ or as the conditional $P(current-word|previous-word)$. This might be reasonable in language modeling, but for HMMs we only want bigram probabilities for part-of-speech tags. Otherwise the probabilities cannot be combined into a single probability for the sentence and tags. Also note that joint probabilities cannot be combined at all - e.g. $P(w_0, w_1)*P(w_1, w_2)$ isn't a sensible probability.

- What we want here is $P(w_i|t_i)$, estimated as $\frac{count-of(w_i,t_i)}{count-of(t_i)}$
  $P(Mary|N) = \frac{2}{9}$
  $P(lamb|N) = \frac{4}{9}$
  $P(fleece|N) = \frac{1}{9}$
  $P(snow|N) = \frac{1}{9}$
  $P(white|N) = \frac{1}{9}$
  You can optionally include $0/9$ for all other words.

1. One VERY simple baseline is to select the most frequent part-of-speech for a particular word type.
2. 10 x 40^2 = 16,000
3. See Figure 3.

3 Information Extraction

1. See Figure 4.
2. See Figure 4.
3. See Figure 5 for one possible extraction pattern.
4. Noun phrase coreference resolution is important for IE because it allows information to be extracted w.r.t. a particular entity in sentences or clauses where it is not mentioned explicitly. It also important for allowing all of the information extracted w.r.t. an entity or event to be appropriately merged even when it is spread throughout a document.

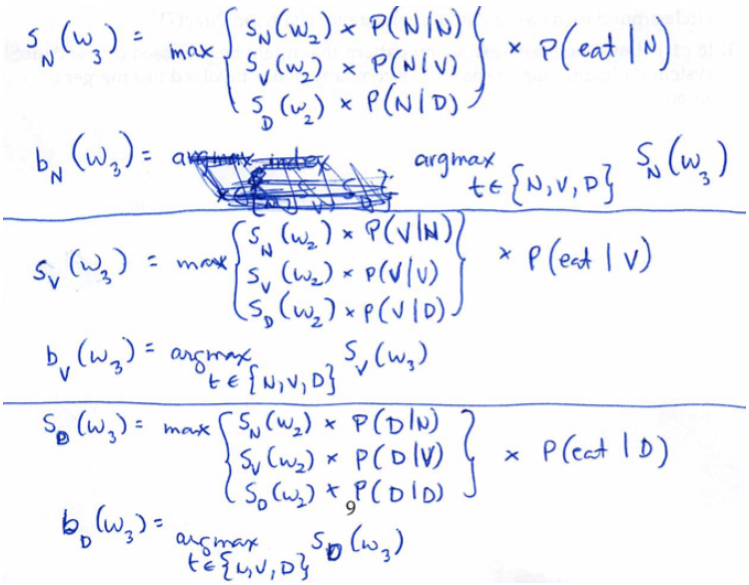

Figure 3: Viterbi Calculations

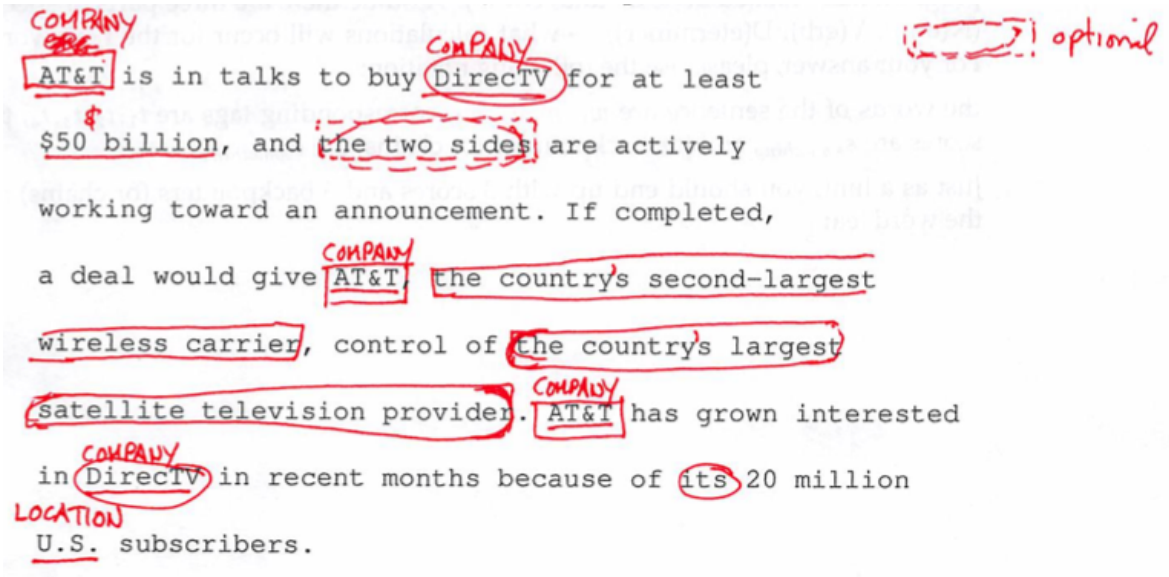

Figure 4: NEs and NP coref chains.

5. Some examples: (1) "the country's largest satellite television provider" = DirecTV. This would allow the system to know that DirectTV is the

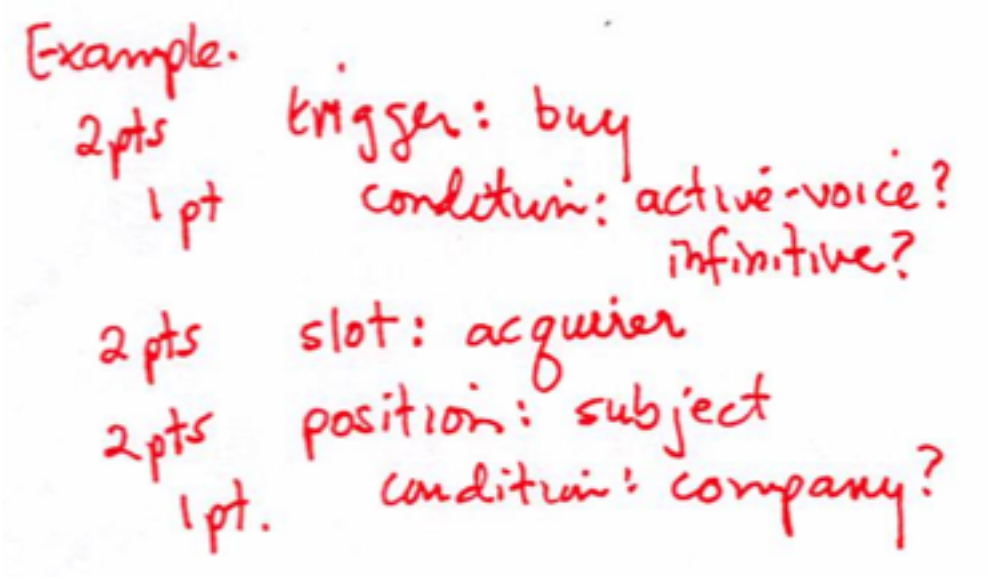

Figure 5: Extraction pattern.

entity being controlled in: "control of the country's largest satellite television provider".

(2) "its" in "its 20 million U.S. subscribers" refers to DirecTV. This would potentially allow the IE system to identify the reasons for the acquisition.

4 Grab Bag

1. False. Zipf's law relates a term's frequency to its rank in a (large) corpus, but does not state anything about the difference in term frequencies in different corpora.

2. False. WordNet is fundamentally arranged around the synset, not the lexeme. A lexeme is one surface form with several related senses, whereas one synset is one sense with several possible surface forms.

3. Language modeling can be used in language identification by cre-

ating a language model for each of the target languages. Remember that an n-gram language model gives the probability of a word given the n-1 preceding words – $P(x_i|x_{i-1}, x_{i-2}, , x_{i-(n-1)})$. A language model can be used to compute the probability of a sequence of words by combining the probabilities of individual words by multiplying, adding, or using another form of combination. This way we can compute the probability of a sequence of words (text) as

$$P(x_0, x_1, , x_k) = P(x_0|x_{-1}, x_{-2}, , x_{-(n-1)}).P(x_1|x_0, x_{-1}, , x_{-n})...P(x_n|x_{k-1}, x_{k-2}, , x_{k-(n-1)})$$

for a specific language model.

Using this formula, and given text and a language model for each of the target languages we can compute the probability that each of the language models assigns for the text and decide that the text is from the language associated with the language model that gives the highest probability.

For training we will need a set of (untagged) texts from each of the target languages.

Note that if we decide to use unigram language models, this is equivalent to counting the relative frequence of words in each of the target languages and using this relative frequence as the probability of each word.