6.863J Natural Language Processing Lecture 16: the boundaries of syntax & semantics – towards constraint-based systems

Robert C. Berwick
berwick@ai.mit.edu

The Menu Bar

- Administrivia:
- Lab 4 due April 9? (what about Friday)
  - Start w/ final projects, unless there are objections
- *Agenda:*
  - Shallow instead of ‘deep’ semantics: MUC
  - Stochastic language use? Some examples
  - How to accommodate: towards constraint-based grammar

How to *integrate* all this stuff?

- We saw that we might want to partition syntactic knowledge from semantic...
- But we have to go farther – because both of these might be probabilistic in nature
- How to integrate?

Integration

- One way: semantic grammar (see below)
- The way we'll explore though:
  - Define linguistic structure
  - Place pr distributions on that structure

Example

- Syntactic rule = NP → Det N
- Semantic extension = NP :

  Apply(lambda (x) (DEF/SING x), N)
- Lexicon:
  - Art:the: DEF/SING
  - N:guy: Person
- Parse of the NP *the guy*:
  - NP → Det N ⇒ OK, NP contains article & noun
  - Apply (lambda (x) (DEF/SING x), N) ⇒ OK, NP contains DEF/SING article
  - Apply (lambda (x) (DEF/SING x), Person) ⇒ the N in the NP is Person
  - (DEF/SING Person) ⇒ result of applying lambda calculus ⇒ textual replacement of variable x with argument Person

Another Example

Syntax:
S → NP VP
VP → V NP
NP → Det N
Semantics:
S : Apply(VP, NP)
VP : Apply(V, NP)
NP : Apply(lambda (x) (DEF/SING x), N)

Lexicon:
V:kissed = lambda(o) lambda(x) (kiss past [agent x] [theme o])
N:guy = person
N:dog = DOG
Det:the = DEF/SING

Top-down parse sentence:
The guy kissed the dog

- S → NP VP — Apply(VP, NP)
- VP → V NP — Apply(Apply(V, NP), NP)
- Lexicon look-up — Apply(Apply(lambda(o) lambda(x) (kiss past [agent x] [theme o]), NP), NP)
- NP → ART N — Apply(Apply(lambda(o) lambda(x) (kiss past [agent x] [theme o]), Apply(lambda (x) DEF/SING x), N)), NP)
- Lexicon look-up — Apply(Apply(lambda(o) lambda(x) (kiss past [agent x] [theme o]), Apply(lambda (x) (DEF/SING x), DOG)), NP)
- Apply-operator — Apply(Apply(lambda(o) lambda(x) (kiss past [agent x] [theme o]), (DEF/SING DOG), NP)
- Apply-operator — Apply(lambda(x) (kiss past [agent x] [theme DEF/SING DOG]), NP)
  NP → ART N — Apply(lambda(x) (kiss past [agent x] [theme DEF/SING DOG]), Apply(lambda(x) (DEF/SING * x), N))
- Lexicon look-up — Apply(lambda(x) (kiss past [agent x] [theme DEF/SING DOG]), Apply(lambda(x) (DEF/SING * x), Person))
- Apply-operator — Apply(lambda(x) (kiss past [agent x] [theme DEF/SING DOG]), (DEF/SING Person))
- Apply-operator — (kiss past [agent DEF/SING Person] [theme DEF/SING DOG])

Semantic Grammar: Definition

- Syntactic and semantic processing is collapsed in a single framework
- Like a regular grammar but terminal symbols are replaced by semantic categories
- Example:
  - [VP read [NP a book]] or [write [a book]]

$$\begin{array}{ccc} VP & \rightarrow & V\ NP \\ & \Downarrow & \\ \text{READ-VP} & \rightarrow & \text{READ-VERB READ-STUFF} \\ \text{WRITE-VP} & \rightarrow & \text{WRITE-VERB WRITE-STUFF} \end{array}$$

Example of a Grammar

- RES-VP → RESERVING RES-MOD
- RES-VP → RESERVING
- DEP-VP → DEPARTING DEP-MODS
- RESERVING → RESERVE-VERB FLIGHT
- RES-MOD → for PERSON
- DEPARTING → DEPART-VERB
- DEPARTING → DEPART-VERB SOURCE-LOCATION
- DEP-MODS → DEP-MOD DEP-MODS
- DEP-MODS → DEP-MOD
- DEP-MOD → to DEST-LOCATION
- DEP-MOD → from SOURCE-LOCATION

Exercise

- Grammar:

  RES-VP → RESERVING
  RES-VP → RESERVING RES-MOD
  RESERVING → RESERVE-VERB FLIGHT-NP
  RES-MOD → for PERSON
  FLIGHT-NP → ART FLIGHT-NOUN
  FLIGHT-NP → ART FLIGHT-NOUN FLIGHT-MODS
  FLIGHT-MODS → FLIGHT-MOD FLIGHT-MODS
  FLIGHT-MODS → FLIGHT-MOD
  FLIGHT-MOD → from SOURCE-LOCATION
  FLIGHT-MOD → to DEST-LOCATION

- Lexicon:

  FLIGHT-NOUN:flight
  ART:a
  PERSON:me
  LOCATION:Boston
  LOCATION:Chicago
  RESERVE-VERB: book

Solution (1)

- FLIGHT-MOD → from SOURCE-LOCATION
  Book a flight [FLIGHT-MOD from SOURCE-LOCATION Boston] to Chicago for me

- FLIGHT-MOD → to DEST-LOCATION
  Book a flight [FLIGHT-MOD from SOURCE-LOCATION Boston] [FLIGHT-MOD to DEST-LOCATION Chicago] for me

- FLIGHT-MODS → FLIGHT-MOD
  Book a flight [FLIGHT-MOD from SOURCE-LOCATION Boston] [FLIGHT-MODS [FLIGHT-MOD to DEST-LOCATION Chicago]] for me

- FLIGHT-MODS → FLIGHT-MOD FLIGHT-MODS
  Book a flight [FLIGHT-MODS [FLIGHT-MOD from SOURCE-LOCATION Boston] [FLIGHT-MODS [FLIGHT-MOD to DEST-LOCATION Chicago]]] for me

Solution (2)

- FLIGHT-NP → ART FLIGHT-NOUN FLIGHT-MODS
  Book [FLIGHT-NP ART a FLIGHT-NOUN flight [FLIGHT-MODS [FLIGHT-MOD from SOURCE-LOCATION Boston] [FLIGHT-MODS [FLIGHT-MOD to DEST-LOCATION Chicago]]]] for me

- RESERVING → RESERVE-VERB FLIGHT-NP
  [RESERVING RESERVE-VERB Book [FLIGHT-NP ART a FLIGHT-NOUN flight [FLIGHT-MODS [FLIGHT-MOD from SOURCE-LOCATION Boston] [FLIGHT-MODS [FLIGHT-MOD to DEST-LOCATION Chicago]]]]] for me

- RES-MOD → for PERSON
  [RESERVING RESERVE-VERB Book [FLIGHT-NP ART a FLIGHT-NOUN flight [FLIGHT-MODS [FLIGHT-MOD from SOURCE-LOCATION Boston] [FLIGHT-MODS [FLIGHT-MOD to DEST-LOCATION Chicago]]]]] [RES-MOD for PERSON me]

- RES-VP → RESERVING RES-MOD
  [RES-VP [RESERVING RESERVE-VERB Book [FLIGHT-NP ART a FLIGHT-NOUN flight [FLIGHT-MODS [FLIGHT-MOD from SOURCE-LOCATION Boston] [FLIGHT-MODS [FLIGHT-MOD to DEST-LOCATION Chicago]]]]] [RES-MOD for PERSON me]]

Useful?

- Semantic grammars are useful in a limited domain
  - Dialogue system to book flights through the telephone
- For general use → too many rules!

Application Continuum

- Machine Translation
- Unrestricted language comprehension
- Summarization
- Information extraction
  - Find specific information: location, names of terrorists, ...
- Text classification
  - What is the text about (topic detection)?
- Information retrieval

Information Extraction

- Analyzing unrestricted, unstructured text
- Extracting specific structured information
- Enabling technology
  - Converting text to a database (data mining)
  - Summarization

Example from the terrorism domain

Input:

San Salvador, 19 Apr 89. Salvadoran President-elect Alfredo Cristiani condemned the terrorist killing of Attorney general Roberto Garcia Alvarado and accused the Farabundo Marti National Liberation Front (FMLN) of the crime. (...)

Garcia Alvarado, 56, was killed when a bomb placed by urban guerrillas on his vehicle exploded as it came to a halt at an intersection in downtown San Salvador.

Vice President-elect Francisco Merino said that when the attorney-general's car stopped at a light on a street in downtown San Salvador, an individual placed a bomb on the roof of the armored vehicle. (...)

According to the police and Garcia Alvarado's driver, who escaped unscathed, the attorney general was traveling with two bodyguards. One of them was injured.

Example from the terrorism domain

Output template:

```
Incident: Date 19 APR 89
Incident: Location El Salvador: San Salvador
Incident: Type Bombing
Perpetrator: Individual ID urban guerrillas
Perpetrator: Organization ID FMLN
Perpetrator: Organization conf suspected or accused
Physical target: description vehicle
Physical target: effect some damage
Human target: name Roberto Garcia Alvarado
Human target: description attorney general
              Alvarado, driver, bodyguards
Human target: effect death: alvarado, no injury:
              driver, injury: bodyguards
```

Example System: FASTUS

- Finite-State Automaton Text Understanding System (SRI International)
- Cascaded non-deterministic finite-state automaton:
  - from linguistic to domain-dependent
  - from simple (word level) to complex (phrase level)
- Cascade = series of FS systems
  - Tokenization → Complex Words → Basic Phrases → Complex Phrases → Semantic Patterns → Merging
- MUC (Message Understanding Conference)
- Evaluation → recall and precision, F_{β}

Evaluation

- Precision $= \dfrac{\text{\# correct answers}}{\text{\# total answers}}$

- Recall $= \dfrac{\text{\# correct answers}}{\text{\# possible correct answers}}$

- $F_{ß} = \dfrac{((ß^2 + 1) \times P \times R)}{(ß^2 \times P \times R)}$

Architecture – Steps 1 to 3

- Tokenization → split words and punctuation
  - *He is mr. Jones!* → *He is mr. Jones !*
- Named-entity recognition & multi-word phrases
  - Multi-word phrases: *set up*, *joint venture*
  - Named entities: *Secretary General Annan*, *Prof. Dr. L. Steels*
- Find basic phrases:
  - Nominal and verbal phrases
  - Prepositions
  - Particles

After step 3 ...

- Bridgestone Sports Co. [company name] said [VG] Friday [NG] it [NG] had set up [VG] a joint venture [NG] in [Prep] Taiwan [location NG] with [Prep] a local concern [NG] and [Conj] a Japanese trading house [NG] to produce [VG] golf clubs [NG] to be shipped [VG] to [Prep] Japan [location NG]

Architecture – Step 4

- Construction of complex nominal and verbal groups
  - Apposition
    - *the secretary general Koffi Annan*
  - PP-attachment
    - *production of spice girl dolls*
  - Domain entities
    - Relationship: tie-up; jv-company: Bridgestone
    - Activity: production; product: spice girl dolls

Architecture – Step 5 & 6

- Recognition and construction of event structures:
  - <companies> <set-up> <joint-venture> with <companies>
  - <produce> <product>
  - <company> <capitalized> at <currency>
  - <company> <start> <activity> in/on <date>
- Fusion of event structures referring to the same event = (co-)reference resolution

FASTUS – Evaluation

- MUC-4:
  - 44% recall and 55% precision
- Human-level competence: 65-80% reliability
- Speed:
  - 2500 words per minute
  - 10 seconds per text
  - 9000 texts per day
- Simple, accurate, 2 x fast → fast in runtime & fast in development time
- Japanese version:
  - 34% recall
  - 56% precision

How to integrate in general Language & statistics

- “All grammars leak” – Sapir, 1921
- Is this true? What do we do?

- An example – subcategorization & thematic roles

A leak

- “By the time their son was born, though, Honus Whiting was beginning to understand and privately share his wife’s opinion, at least as it pertained to Empire Falls”

Subcategorization: what we have been doing

- Eat vs. devour:

  John ate the meal/John ate

  Bill devoured the meal/Bill devoured

- Verb selects category of its complements – at least the syntactic category

Subcategorization?

- Example:
  - *consider:* __ NP[acc] {AdjP, NP, VP[inf]}
  - *regard:* __ NP[acc] *as* {NP, AdjP}
  - *think:* __ CP[*that*], __ NP[acc] NP

There are standard examples for these – cf. Lab 3.

Example – consider w/ no ‘as’

- John considers vanilla to be an acceptable flavor
- John considers vanilla an acceptable flavor
- John considers vanilla quite an acceptable flavor
- John considers vanilla among the most acceptable flavors

But what about these?

- John considers vanilla as an acceptable flavor
- John considers vanilla as quite acceptable
- John considers vanilla as among the most acceptable flavors
- John considers vanilla as being among the most acceptable flavors

Compare “regard”

- John regards vanilla as an acceptable flavor
- John regards vanilla to be an acceptable flavor

- (supposed to be the opposite of “consider”!)

Or consider “turn out”

- Takes AdjP but not a present participle:
  - John turned out political
  - John turned out doing all the work

The ‘paper of record’ – the NY times – doesn’t support the linguists

Consider as:

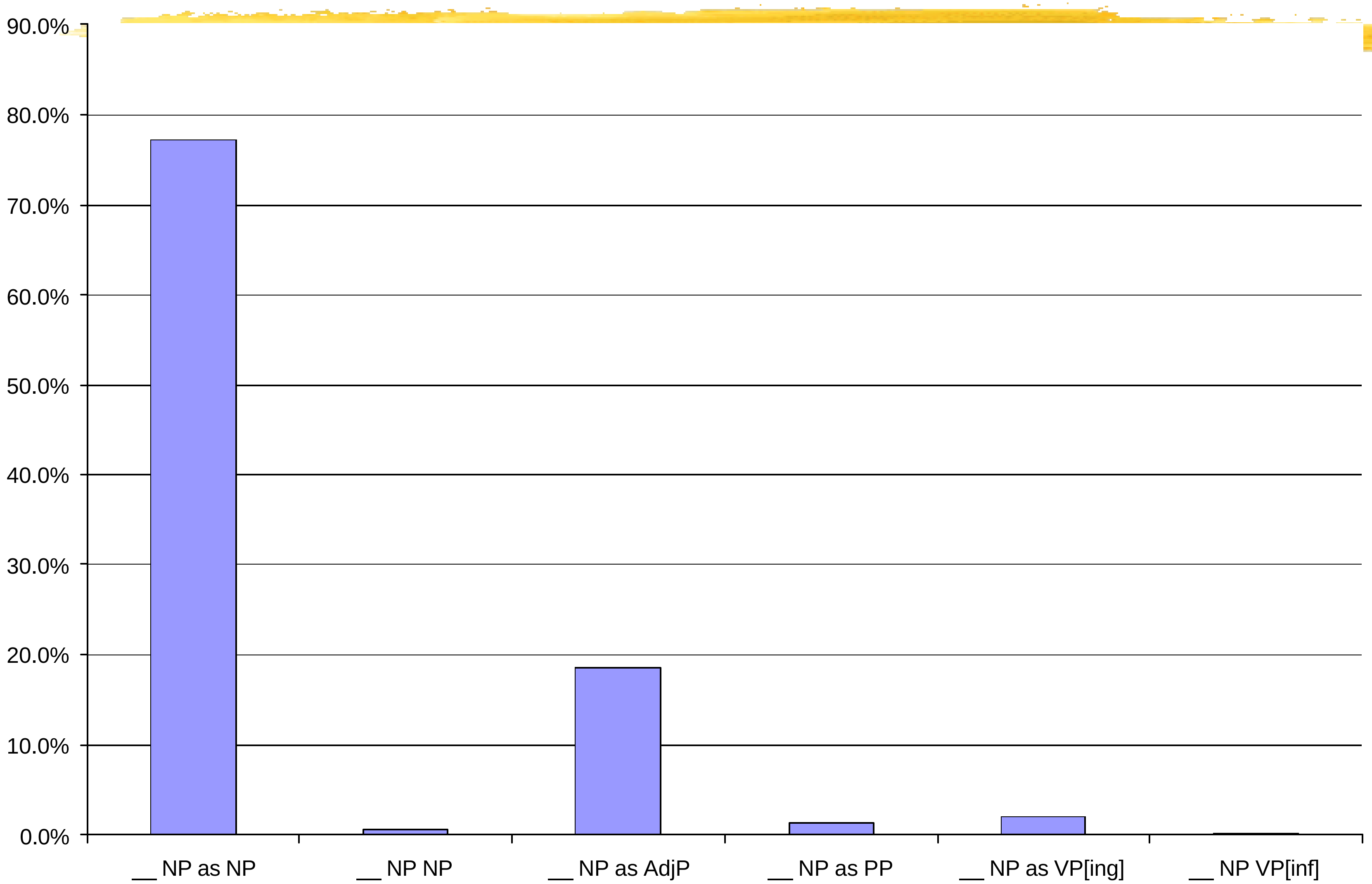
- The boys consider her as family and she participates in everything we do.
- Greenspan said, “I don't consider it as something that gives me great concern.
- “We consider that as part of the job,” Keep said.
- Although the Raiders missed the playoffs for the second time in the past three seasons, he said he considers them as having championship potential.
- Culturally, the Croats consider themselves as belonging to the “civilized” West, ...

Regarding the NY Times

- As 70 to 80 percent of the cost of blood tests, like prescriptions, is paid for by the state, neither physicians nor patients regard expense to be a consideration
- Conservatives argue that the Bible regards homosexuality to be a sin
- But it turned out having a greater impact than any of us dreamed
- On the big night, Horatio ended up flattened on the ground like a fried egg with the yolk broken

How to solve this? Use probabilities...!

NY Times



90.0%
80.0%
70.0%
60.0%
50.0%
40.0%
30.0%
20.0%
10.0%
0.0%
__ NP as NP
__ NP NP
__ NP as AdjP
__ NP as PP
__ NP as VP[ing]
__ NP VP[inf]

And in general

- Instead of using a *subset* of the data
- Use a superset...and add distributional pr weights

Mutual aid

- Most formal models: no frequency information, and so grammaticality judgments or exploration of a factorial typology cross-linguistically is used
- Most “corpus linguistics”: there is frequency information, but an insufficiently developed theory of abstract syntax (“hidden structure”) for the frequency information to interact productively with a formal theory
- Goal: to get productive mutual feedback

Incorporating knowledge

- Do density estimation

  P(form | meaning context)

Application: retire

- Step 1: look at what dictionary or wordnet has for subcat
  - Result: intrans; transitive NP; PP (to, from)
- Step 2: see whether these examples attested (viz., Wall Street Journal)
  - Mr Riley plans to retire to the 1.5million dollar ranch he is building in Cody, Wyoming
  - Mr Frey, 64, remains chairman but plans to retire from that post in June
  - To all those wishing to retire in Mexico, let me offer these suggestions
  - Donald Tanselle, 62, will retire as VP of banking
  - A worker contributing 10% of earnings will be able to retire on a pension equal to 2/3 of their salary

What now?

- Step 3: do some statistics
- PP [on] is ‘monetary support’ – so are these real argument or adjuncts?
- Answer: don’t decide!
- Calculate conditional statistics
- Look at 1987 WSJ, and we get this:

WSJ conditional pr's

- P(NP[subj]|V=retire) = 1.0
- P(NP[obj]|V=retire) = 0.52
- P(PP[*from*]|V=retire) = 0.05
- P(PP[*as*]|V=retire) = 0.06

...

(Pr of having certain argument adds to 1)

(assumes independence between arguments – chance getting PP[as] indep of getting PP[from])

We can recalculate entire frame

- P(NP[subj]____|V=retire) = 0.25
- P(NP[subj]____NP[obj]|V=retire) = 0.50
- P(NP[subj]____PP[from]|V=retire) = 0.04
- P(NP[subj]____PP[from]PP[after]|V=retire) = 0.003

…

(Sum of pr's of all <u>frames</u> adds to 1)

Then we can do things like this

- Integrate pr of a 'frame' into the syntactic structure
- Pr that a VP is headed by a certain verb, and arguments surrounding that verb:
- $P(VP \rightarrow V[retire]\ PP[from]) =$

$$P(head=retire|VP)$$

$$\times P(VP \rightarrow V\ \ PP|VP,\ head=retire)$$

- Actually, it's more than surface subcat info
- Consider:
  - Martinez will retire next year

General model for verb subcat

- Want: P(*Subcat* = *f* | *Verb* = *v*)
- We model subcategorization at the level of the argument structure *a*, which groups data
- Decompose as:
  - $P(f \mid v) = P(a,m \mid v) = P(a \mid v)P(m \mid a,v)$
- Mappings *m* (including deletions, insertions) are few, and fairly consistent for <u>semantic</u> roles
- Verb classes: $$P(a \mid v) = \sum_{vc} P(vc \mid v) P(a \mid vc)$$

So…

- Payoff: this knowledge builds parsers that do very, very well – the best
- How can we acquire this info automatically?

Lerner (Brent 1993)

- Cues
  - A pattern that can be matched against unrestricted text
  - NP NP → (OBJ|SUBJ_OBJ|CAP) (PUNC|CC)
    - [...] *greet Peter*, [...]
    - [...] *see him.* [...]
    - [...] *love it, if* [...]
    - [...] *came Thursday*, [...] → error!
- Hypothesis Testing
  - Initial or null hypothesis (H_0) → frame is not appropriate for verb
  - If cue indicates with high probability that frame is appropriate → reject H_0

Hypothesis Testing

$$p_E = P\left(v^i\left(f^j\right) = 0 \middle| C\left(v^i, c^j\right) \geq m\right) = \sum_{r=m}^{n} \binom{n}{r} e_j^{\,r} \left(1 - e_j\right)^{n-r}$$

- Verb v^i occurs n times, and there are $m = n$ occurrences with a cue c^j for frame f^j
- $v^i(f^j) = 0$ → frame f^j is <u>not</u> appropriate for verb v^i
- $C(v^i, c^j)$ → number of times verb v^i occurs with cue c^j
- e_j → error rate of cue c^j → the probability that the cue matches, but that it is not evidence for a frame
- Determine a threshold a
- P_E < a → reject H_0
- P_E > a → H_0 is correct

Hypothesis Testing – Example

$$p_E = P\left(v^i\left(f^j\right) = 0 \middle| C\left(v^i, c^j\right) \geq m\right) = \sum_{r=m}^{n} \binom{n}{r} e_j^{\,r} \left(1 - e_j\right)^{n-r}$$

- Verb = greet → occurs 80 times (n = 80)
- Cue = (OBJ|SUBJ_OBJ|CAP) (PUNC|CC) → has *e* = 0.25
- Frame = NP__ NP
- C(greet,(OBJ|SUBJ_OBJ|CAP) (PUNC|CC)) = 11 (m = 11 and r = 11)

Hypothesis Testing – Example

$$p_E = P\left(\text{greet}\left(\text{NP NP}\right) = 0 \middle| C\left(\text{greet,((OBJ|SUBJ\_OBJ|CAP)(PUNC|CC))}\right) \geq 11\right)$$

$$p_E = \sum_{r=11}^{80} \binom{80}{r} 0.25^r \left(1-0.25\right)^{80-r}$$

$$p_E = \binom{80}{11} 0.25^{11} \left(1-0.25\right)^{80-11} + \ldots + \binom{80}{80} 0.25^{80} \left(1-0.25\right)^{80-80}$$

$$p_E = 0.011$$

- Threshold = 0.02
- Do we accept or reject the H_0 (= frame NP__NP is <u>not</u> appropriate for the verb *greet*)?
- Reject → $P_E = 0.011 < 0.02$

Evaluating Lerner

- Very high precision → always close to 100%
- Recall is lower → only 60%
- Only for six frames ...

We can start adding pr's to everything...

- PP attachment ('eat ice-cream with a spoon')
- Selectional preference (eat$\rightarrow$ theme $\rightarrow$ food)