Language Modeling

Julia Hirschberg

CS 4706

Approaches to Language Modeling

- Context-Free Grammars
  - Use in HTK
- Ngram Models

Context-Free Grammars

- Defined in formal language theory
  - Terminals: e.g. cat
  - Non-terminal symbols: e.g. NP, VP
  - Start symbol: e.g. S
  - Rewriting rules: e.g. S → NP VP
- Start with start symbol, rewrite using rules, done when only terminals left

A Fragment of English

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

Input: the cat is on the mat

Derivations in a CFG

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

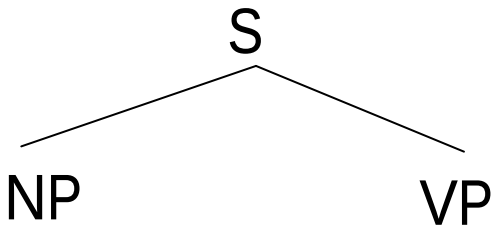
S

S

Derivations in a CFG

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

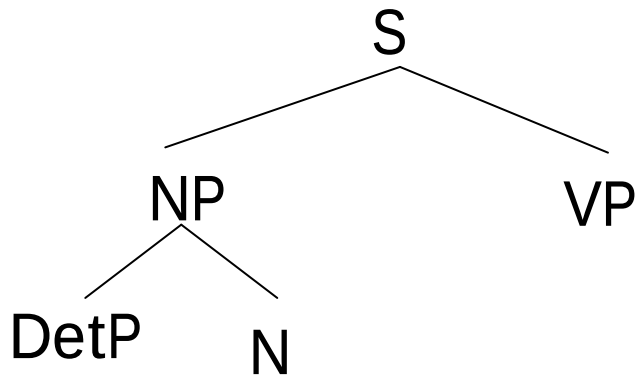
NP VP



Derivations in a CFG

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

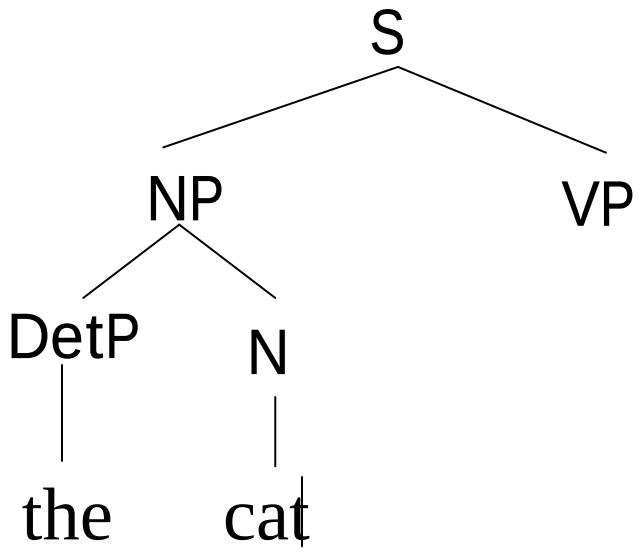
DetP N VP



Derivations in a CFG

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

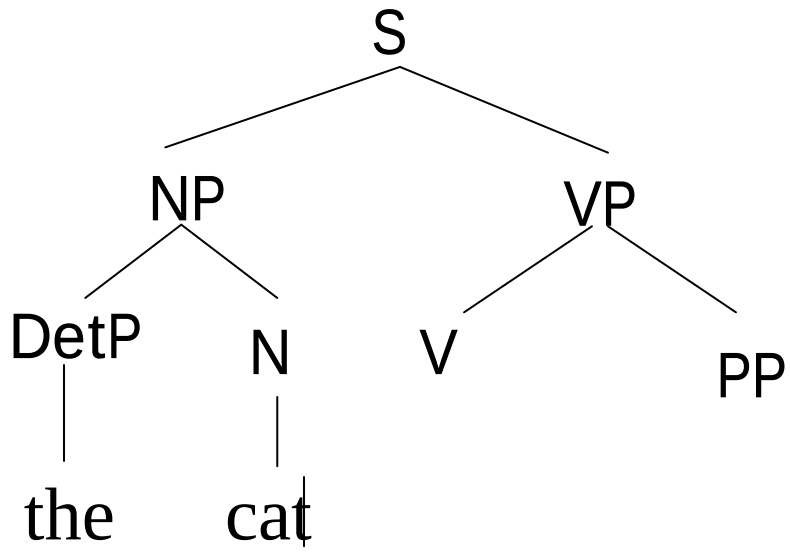
the cat VP



Derivations in a CFG

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

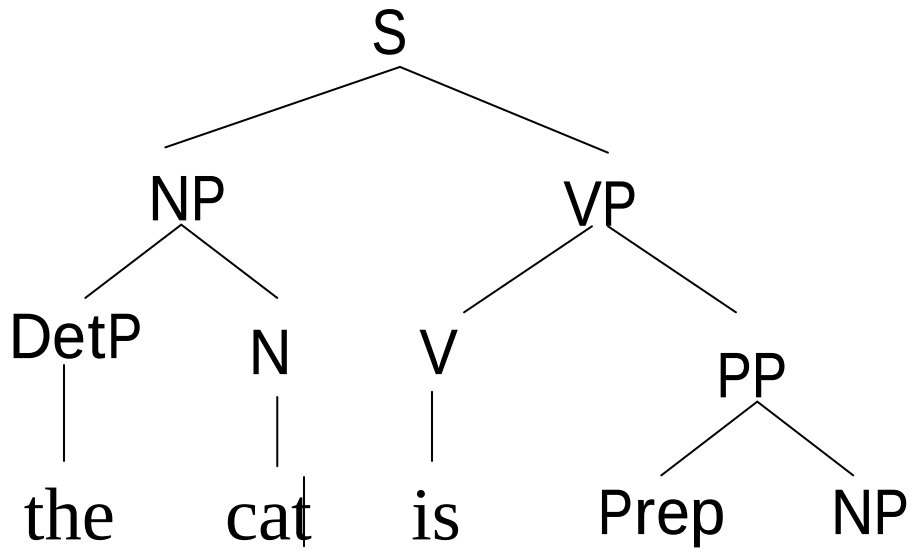
the cat V PP



Derivations in a CFG

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

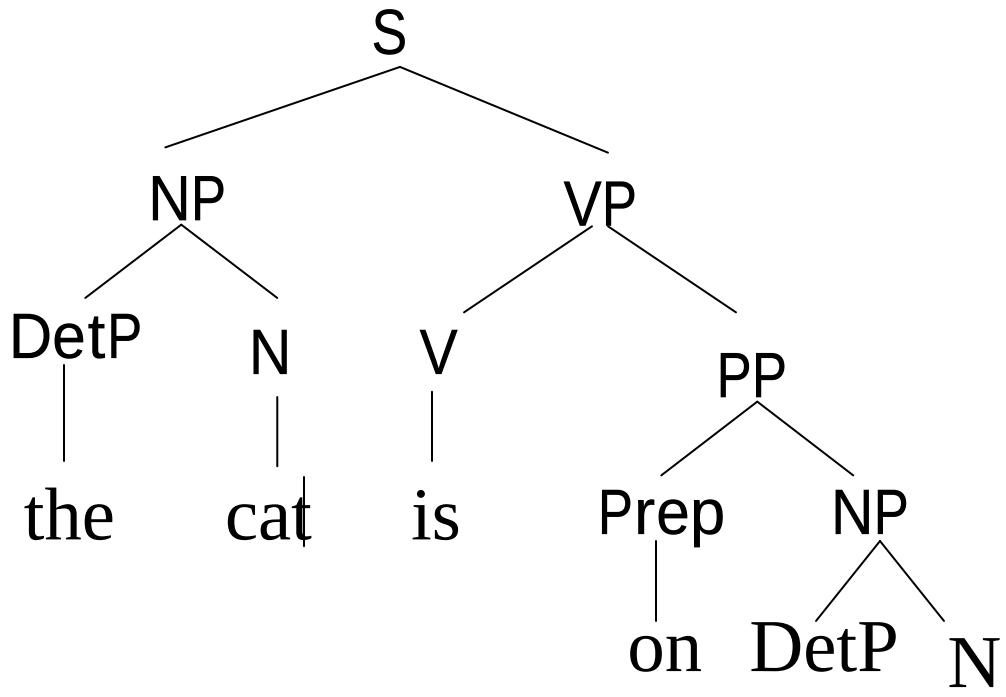
the cat is Prep NP



Derivations in a CFG

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

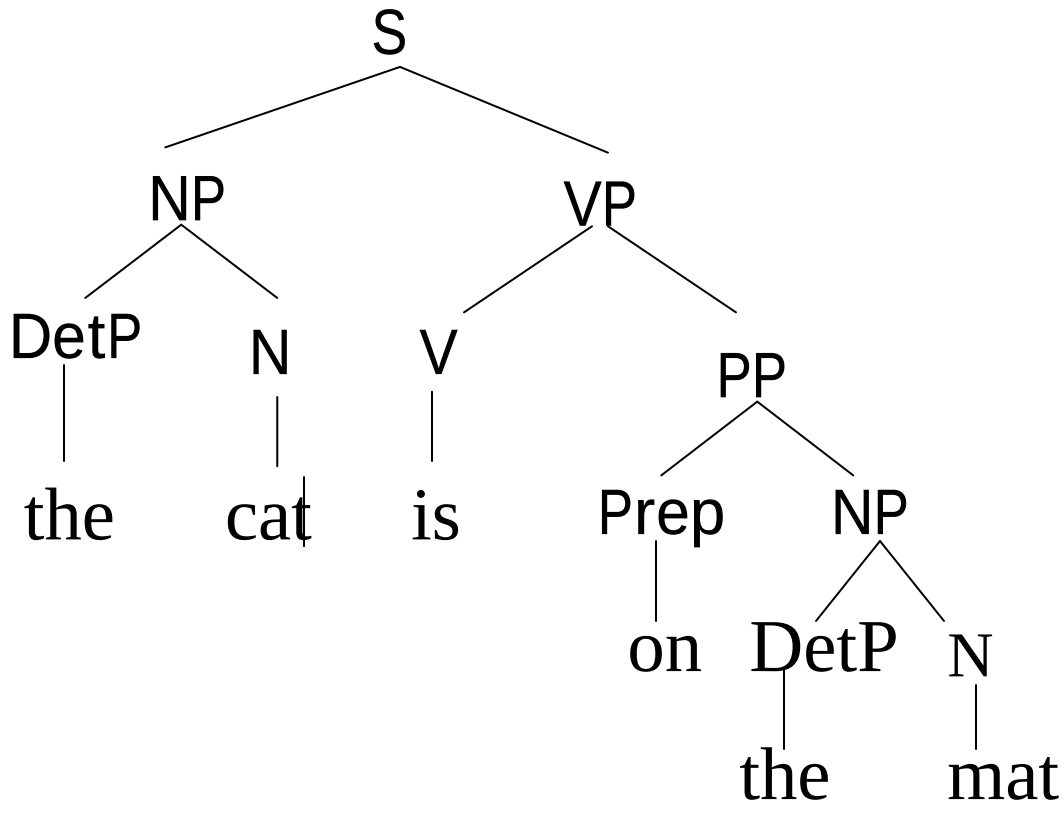
the cat is on Det N



Derivations in a CFG

S → NP VP
VP → V PP
NP → DetP N
N → cat | mat
V → is
PP → Prep NP
Prep → on
DetP → the

the cat is on the mat



A More Complicated Fragment of English

- S → NP VP
- **S → VP**
- VP → V PP
- **VP → V NP**
- **VP → V**
- NP → DetP NP
- **NP → N NP**
- **NP → N**
- PP → Prep NP

- N → cat | mat | **food | bowl | Mary**
- V → is | **likes | sits**
- Prep → on | **in | under**
- DetP → the | a

Mary likes the cat bowl.

Using CFGs in Simple ASR Applications

- LHS of rules is a semantic category:
  - LIST -> show me | I want | can I see|…
  - DEPARTTIME -> (after|around|before) HOUR
    | morning | afternoon | evening
  - HOUR -> one|two|three…|twelve (am|pm)
  - FLIGHTS -> (a) flight|flights
  - ORIGIN -> from CITY
  - DESTINATION -> to CITY
  - CITY -> Boston | San Francisco | Denver | Washington

HTK Grammar Format

- Variables start with $ (e.g., $city)
- Terminals must be in capital letters (e.g., FRIDAY, TICKET)
- X Y is concatenation (e.g., I WANT)
- (X | Y) means X or Y – e.g., (WANT | NEED)
- [X] means optional, (e.g., [ON] FRIDAY)
- <X> Kleene closure (e.g., <$digit>)

Examples

```
$city = BOSTON | NEWYORK | WASHINGTON |
    BALTIMORE;
$time = MORNING | EVENING;
$day = FRIDAY | MONDAY;
(SENT-START
    ((WHAT TRAINS LEAVE) | (WHAT TIME CAN I
        TRAVEL) | (IS THERE A TRAIN))
    (FROM|TO) $city (FROM | TO) $city ON $day [$time])
SENT-END)
```

Problems for Larger Vocabulary Applications

- CFGs complicated to build and hard to modify to accommodate new data:
  - Add capability to make a reservation
  - Add capability to ask for help
  - Add ability to understand greetings
  - …
- Parsing input with large CFGs is slow for real-time applications
- So…for large applications we use ngram models

Next Word Prediction

The air traffic
control supervisor who admitted falling
asleep while on duty at
Reagan National Airport has been suspended, and the
head of the Federal Aviation Administration on Friday ordered new rules to ensure a similar incident doesn't
take place. FAA chief Randy Babbitt said he has directed controllers at regional radar facilities to contact the towers of airports where there
is only one controller on duty at night before
sending planes on for landings. Babbitt also said regional
controllers have been
told that if no controller can be raised at the airport, they must offer pilots the option of diverting to another
airport. Two commercial
jets were unable to contact the control
tower early Wednesday and
had to land without gaining clearance.

Word Prediction

- How do we know which words occur together?
  - Domain knowledge
  - Syntactic knowledge
  - Lexical knowledge
- Can we model this knowledge computationally?
  - Simple statistical techniques do a good job when trained appropriately
  - Most common way of constraining ASR predictions to conform to probabilities of word sequences in the language – Language Modeling via N-grams

N-Gram Models of Language

- Use the previous N-1 words in a sequence to predict the next word
- Language Model (LM)
  - unigrams, bigrams, trigrams,…
- How do we train these models to discover co-occurrence probabilities?

Finding Corpora

- Corpora are online collections of text and speech
  - Brown Corpus
  - Wall Street Journal, AP newswire, web
  - DARPA/NIST text/speech corpora (Call Home, Call Friend, ATIS, Switchboard, Broadcast News, TDT, Communicator)

Tokenization: Counting Words in Corpora

- What is a word?
  - e.g., are cat and cats the same word? Cat and cat?
  - September and Sept?
  - zero and oh?
  - Is _ a word? * ? ‘(‘ ? Uh ?
  - Should we count parts of self-repairs? (go to fr- france)
  - How many words are there in don’t? Gonna?
  - Any token separated by white space from another?
    - In Japanese, Thai, Chinese text -- how do we identify a word?

Terminology

- Sentence: unit of written language (SLU)
- Utterance: unit of spoken language (prosodic phrase)
- Wordform: inflected form as it actually appears in the corpus
- Lemma: an abstract form, shared by word forms having the same stem, part of speech, and word sense – stands for the class of words with stem X
- Types: number of distinct words in a corpus (vocabulary size)
- Tokens: total number of words

Simple Word Probability

- Assume a language has T word ***types*** and N ***tokens***, how likely is word y to follow word x?
  - Simplest model: 1/T
    - But is every word equally likely?
  - Alternative 1: estimate likelihood of y occurring in new text based on its general frequency of occurrence estimated from a corpus (unigram probability)

    ct(y)/N
    - But is every word equally likely in every context?
  - Alternative 2: condition the likelihood of y occurring on the context of previous words ct(x,y)/ct(x)

Computing Word Sequence (Sentence) Probabilities

- Compute probability of a word given a preceding sequence
  - P(the mythical unicorn…) = P(the|<start>) P(mythical|<start> the) * P(unicorn|<start> the mythical)…
- Joint probability: $P(w_{n-1},w_n) = P(w_n \mid w_{n-1})\, P(w_{n-1})$
  - Chain Rule: Decompose joint probability, e.g. $P(w_1,w_2,w_3)$ as

    $P(w_1,w_2, ...,w_n) = P(w_1)\, P(w_2|w_1) \ldots P(w_n|w_{1 \text{ to } n-1})$
- But…the longer the sequence, the less likely we are to find it in a training corpus

  P(Most biologists and folklore specialists believe that in fact the mythical unicorn horns derived from the **narwhal**)

Bigram Model

- Markov assumption: the probability of a word depends only *on the probability of a limited history*
- Approximate $P(w_n|w_1^{n-1})$ by $P(w_n|w_{n-1})$
  - P(unicorn|the mythical) by P(unicorn|mythical)
- Generalization: *the probability of a word depends only on the probability of the **n** previous words*
  - trigrams, 4-grams, 5-grams…
  - the higher n is, the more training data needed

- From
  - P(the mythical unicorn…) = P(the|<start>) P(mythical|<start> the) * P(unicorn|<start> the mythical)…
- To
  - P(the,mythical,unicorn) = P(unicorn|mythical) P(mythical|the) P(the|<start>)

Bigram Counts (fragment)

n / n-1	<S>	eats	honey	mythical	cat	unicorn	the	a	<end>
<S>	0	0	5	10	0	2	80	90	0
eats	0	0	5	5	10	3	10	10	10
honey	0	0	1	0	2	0	5	3	5
mythical	0	0	2	2	8	5	0	0	5
cat	0	0	0	0	0	0	0	1	5
unicorn	0	4	3	0	1	0	2	2	7
the	0	0	10	8	15	10	2	0	0
a	0	0	2	5	10	12	0	3	0
<end>	999	0	0	0	0	0	0	0	0

Determining Bigram Probabilities

- Normalization: divide each row's counts by appropriate unigram counts for w_{n-1}

<start>	a	mythical	cat	eats	honey	<end>
1000	200	35	60	25	50	1000

- Computing the bigram probability of mythical mythical
  - C(m,m)/C(all m-initial bigrams)
  - p (m|m) = 2 / 35 = .05714
- Maximum Likelihood Estimation (MLE): relative frequency of e.g. $\frac{freq(w_1, w_2)}{freq(w_1)}$

A Simple Example

- P(a mythical cat…) = P(a | <start>) P(mythical | a) P(cat | mythical) … P(<end>|…) = 90/1000 * 5/200 * 8/35 …
- Needed:
  - Bigram counts for each of these word pairs (x,y)
  - Counts for each unigram (x) to normalize
  - P(y|x) = ct(x,y)/ct(x)
- Why do we usually represent bigram probabilities as log probabilities?
- What do these bigrams intuitively capture?

Training and Testing

- N-Gram probabilities come from a training corpus
  - overly narrow corpus: probabilities don't generalize
  - overly general corpus: probabilities don't reflect task or domain
- A separate test corpus is used to evaluate the model, typically using standard metrics
  - held out test set; development (dev) test set
  - cross validation
  - results tested for statistical significance – how do they differ from a baseline? Other results?

Evaluating Ngram Models: Perplexity

- Information theoretic, intrinsic metric that usually correlates with extrinsic measures (e.g. ASR performance)
- At each choice point in a grammar or LM
  - Weighted average branching factor: Average number of choices y following x, weighted by their probabilities of occurrence
  - Or, if LM(1) assigns more probability to test set sentences than LM(2), the lower is LM(1)'s perplexity and the better it models the test set

Ngram Properties

- As we *increase the value of N*, the accuracy of an ngram model increases – why?
- Ngrams are quite sensitive to the corpus they are trained on
- A few events (words) occur with high frequency, e.g.?
  - Easy to collect statistics on these
- A very large number occur with low frequency, e.g.?
  - You may wait an arbitrarily long time to get valid statistics on these
  - Some of the zeroes in the table are really zeros
  - Others are just low frequency events you haven't seen yet
  - How to allow for these events in unseen data?

Ngram Smoothing

- Every n-gram training matrix is sparse, even for very large corpora
  - Zipf's law: a word's frequency is approximately inversely proportional to its rank in the word distribution list
- Solution:
  - ***Estimate*** the likelihood of ***unseen n-grams***
  - Problem: how do to adjust the rest of the corpus to accommodate these 'phantom' n-grams?
  - Many techniques described in J&M

Backoff methods (e.g. Katz ‘87)

- For e.g. a trigram model
  - Compute unigram, bigram and trigram probabilities
  - In use:
    - Where trigram unavailable back off to bigram if available, o.w. unigram probability
    - E.g An omnivorous ***unicorn***

LM toolkits

- The CMU-Cambridge LM toolkit (CMULM)
  - http://www.speech.cs.cmu.edu/SLM/toolkit.html
- The SRILM toolkit
  - http://www.speech.sri.com/projects/srilm/

Next

- Evaluating ASR systems