Lecture 2.2

Syntax, POS Tagging & Parsing

Syntax

Syntax, is the study of grammatical relations between words and other units within the sentence.

- Study of structure of sentence in a language
- Word order or subconscious grammatical knowledge
- Refers to the way words are arranged together, and the relationship between them.
- Roughly, goal is to relate surface form (what we perceive when someone says something) to semantics (what that utterance means)
- Representational device is tree structure

Syntax useful for: -

- Grammar checkers
- Question answering
- Information extraction
- Machine translation

Constituency

- How would the blocks relate to one another? e.g.: I hit the man with a stick
  - Two possibilities:
    - I hit [the man with a stick]
    - I hit [the man] with a stick
- Af Somali Exercise:

Write down three examples of Somali constituency

Two kinds of ambiguity:

- She called her friend from Australia.
  - STRUCTURAL AMBIGUITY
    - [She called] [her friend] [from Australia].
    - [She called] [her friend from Australia].

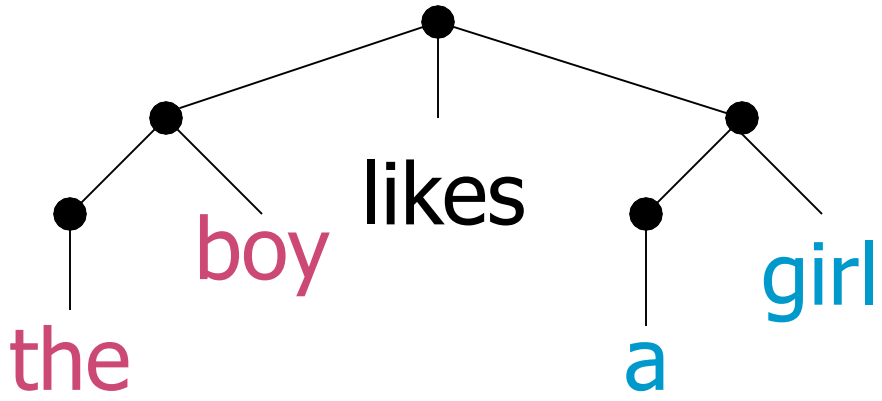
- We went down to the bank yesterday
  - LEXICAL AMBIGUITY
    - **[bank]** river bank
    - **[bank]** financial institute bank

Basic Word Order

- SVO (English, Chinese)
  - *The boy saw the man.*
- SOV (Amharic, Russian, Turkish, Japanese)
  - *Pensive poets painful vigils keep.* (*Pope*)
  - ልጁ ቤቱ ሄደ ። አበበ አልማዝን መታት
- VSO (Irish, Arabic, Welsh)
  - *Govern thou my song.* (*Milton*)

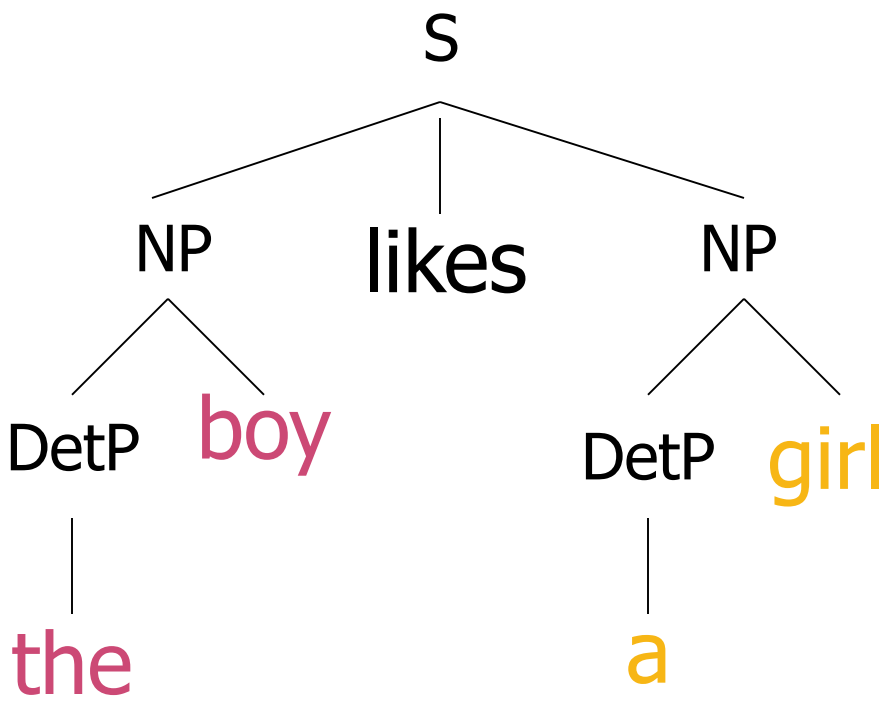
From Substrings to Trees

- (((the) boy) likes ((a) girl))



Node Labels

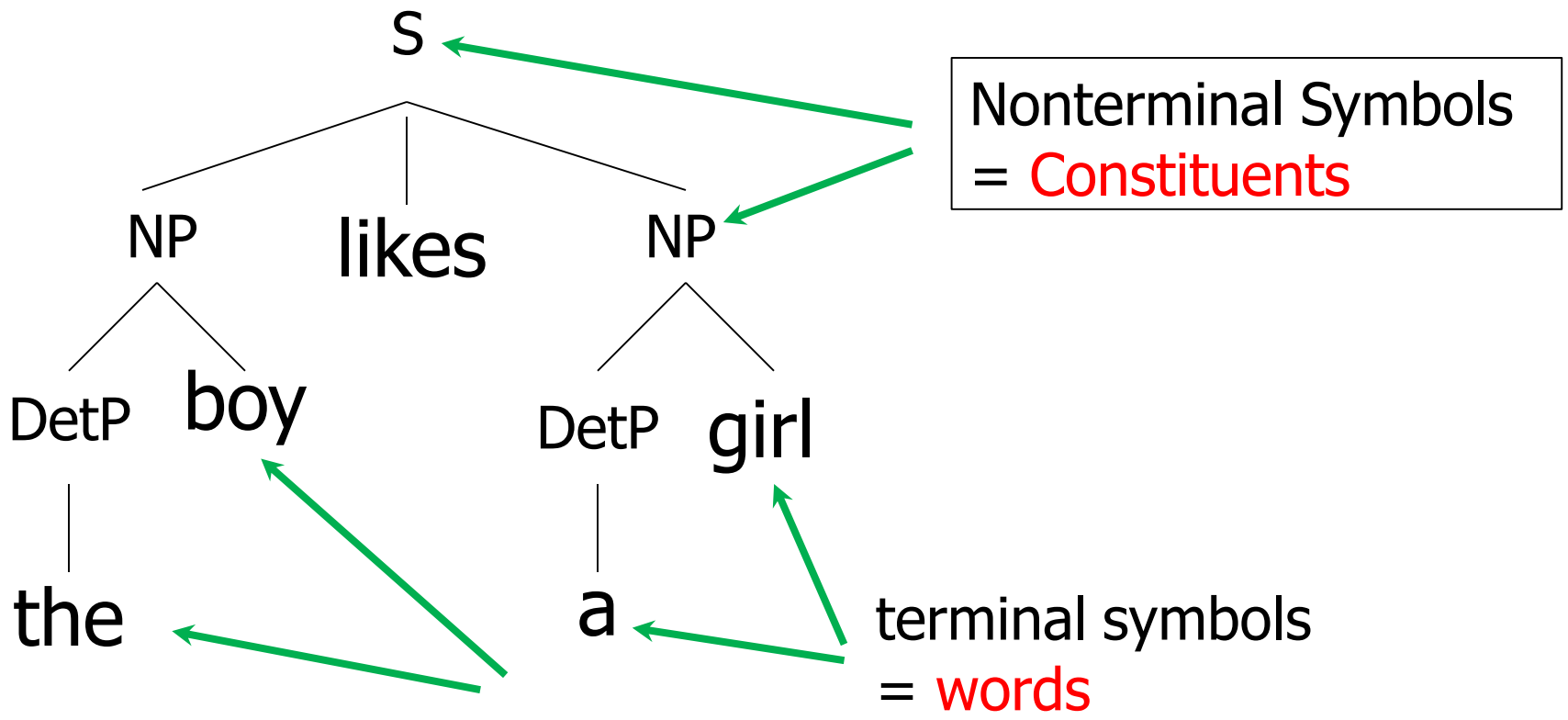
- (((the/Det) boy/N) likes/v ((a/Det) girl/N))



Types of Nodes

- (((the/Det) boy/N) likes/v ((a/Det) girl/N))

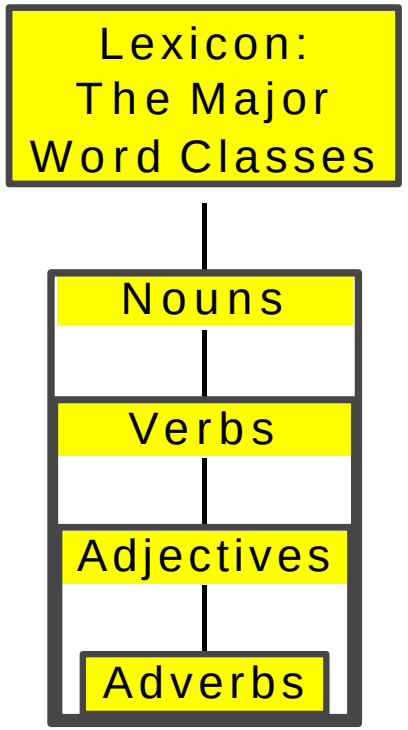
Phrase-structure tree



Determining Part-of-Speech

- Determining part of speech is crucial for building the hierarchical structure of sentences.

The Lexicon



Context-Free Grammars

- Defined in formal language theory
- Composed of
  - Terminals,
  - nonterminals,
  - start symbol, and
  - rules
- CFG is a String-rewriting system/method
- Start with start symbol, rewrite using rules, done until only terminals are left
- ***NOT A LINGUISTIC THEORY***, just a formal device

CFG: Example

- Many possible CFGs for English, here is an example (fragment):
  - S → NP VP
  - VP → V NP
  - NP → DetP N | AdjP NP
  - AdjP → Adj | Adv AdjP
  - N → boy | girl
  - V → sees | likes
  - Adj → big | small
  - Adv → very
  - DetP → a | the

the very small boy likes a girl

Derivations in a CFG

the boy likes a girl

S → NP VP

VP → V NP

NP → DetP N | AdjP

NP AdjP → Adj | Adv

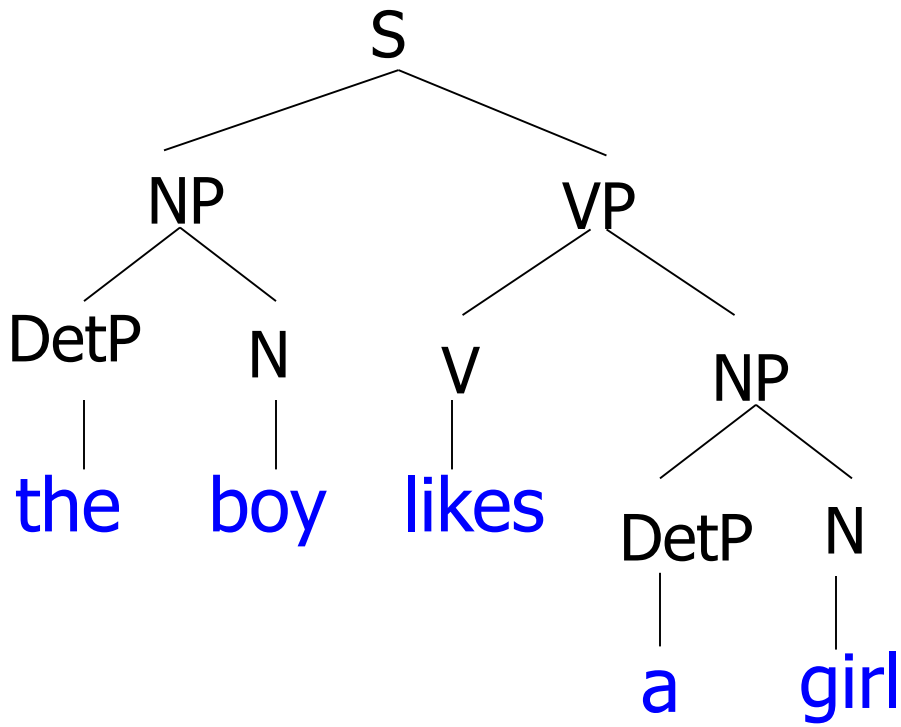
AdjP N → boy | girl

V → sees | likes

Adj → big | small

Adv → very

DetP → a | the



Part Of Speech Tagging

- Syntax requires word classes to be identified
- Words can be divided into classes that behave similarly.
  - Traditionally eight parts of speech:
    - ✓noun, verb, pronoun, preposition, adverb, conjunction, adjective and article
- They tell us a lot about a word (and the words near it).
- Tell us what words are likely to occur in the neighborhood
  - adjectives often followed by nouns
  - personal pronouns often followed by verbs (you, he, she, it..)
  - possessive pronouns by nouns (yours, his, hers, its,….

Part of Speech Tagging

- **PoS Tagging** is the process of annotating each word in a sentence with a part-of-speech marker.
- Lowest level of syntactic analysis is PoS Tagging.

John saw the saw and decided to take it to the table.
NNP VBD DT NN CC VBD TO VB PRP IN DT NN

- Useful for subsequent syntactic parsing and word sense disambiguation.

Tagging Terminology

- **Tagging**
  - The process of associating labels with each token (word) in a text
- **Tags**
  - The labels (Noun, Verb, Adjective, etc)
- **Tag Set**
  - The collection of tags used for a particular task

Tagging Example

- Typically a tagged text is a sequence of white-space separated base/tag tokens:

Example of Tagged Text

The/at, interior/nn , its/pp original/jj form/nn ,/, is/bez truly/ql majestic/jj and/cc an/at architectural/jj triumph/nn ./. Its/pp rotunda/nn forms/vbz a/at perfect/jj circle/nn whose/wp diameter/nn is/bez equal/jj to/in the/at height/nn from/in the/at floor/nn the/at

What does tagging do?

1. **Collapses Some Distinctions**
   - Lexical identity may be discarded
   - e.g. all personal pronouns tagged with PRP
   - e.g. if there are 2000 nouns, all we be tagged as N
2. **Tagging may Introduces Others tag lables**
   - Ambiguities may be removed
   - e.g. *deal* tagged with NN or VB
   - e.g. *deal* tagged with *DEAL1* or *DEAL2*
3. **Helps classification and prediction**

POS and Tagsets

- The choice of tagset greatly affects the difficulty of the problem
- Need to strike a balance between
  - Getting better information about context (best: introduce more distinctions)
  - Make it possible for classifiers to do their job (need to minimize distinctions)

Common Tagsets

- Brown corpus: 87 tags
- Penn Treebank: 45 tags
- Lancaster UCREL C5 (used to tag the British National Corpus - BNC): 61 tags
- Lancaster C7: 145 tags

Word Class/Categories

- Word categories: also called parts of speech
  - *Noun*: Names of things boy, cat, truth
  - *Verb*: Action or state become, hit
  - *Pronoun*: Used for noun like I, you, we
  - *Adjective*: modifies noun happy, clever
  - *Adverb*: modifies V, Adj, Adv sadly, very
  - *Conjunction*: Joins things and, but, while
  - *Preposition*: Relation of N to, from, into
  - *Interjection*: An outcry ouch, oh, alas, psst

Automatic Taggers

- Size of tag sets depends on:
  - language,
  - objectives and
  - purpose
- simple morphology = more ambiguity = fewer tags

- Part-of-Speech Tagging
  - *Rule-Based Tagger*
  - ***Stochastic Tagger: HMM-based***
  - ***Transformation-Based Tagger (Brill)***

- Some addresses the ambiguity problem
  - *The probabilistic approach tries to find **the more likely** tag sequence*

POS Tagging Approaches

- **Rule-Based**: Human crafted rules based on lexical and other linguistic knowledge.
- **Learning-Based**: Trained on human annotated corpora like the Penn Treebank.
  - **Statistical models**: Hidden Markov Model (HMM), Maximum Entropy Markov Model (MEMM), Conditional Random Field (CRF)
  - **Rule learning**: Transformation Based Learning (TBL)
- Generally, learning-based approaches have been found to be more effective overall, taking into account the total amount of human expertise and effort involved.

Stochastic Tagging

- Based on probability of certain tag occurring given various possibilities
  - ***Requires a training corpus***
  - ***No probabilities for words not in corpus.***
  - ***Training corpus may be different from test corpus.***

- Simple Method: Choose most frequent tag in training text for each word!
  - **Result: 90% accuracy**
  - **Unknown for words never encountered before**
  - **HMM is an example**

HMM Tagger

- The Whole Idea:
  - ***guess the most likely tag for a given word.***
- The issue is maximization……
- Assumption:
  - A word's tag only depends on the previous tag (***limited horizon***) and that this dependency does not change over time (***time invariance***)
- HMM Taggers selects a tag sequence that gives the maximum result from the following probability formula:
  - **P(word /tag) × P(tag/previous n tags)**

Markov Model Taggers

- **Bigram tagger**
  - Make predictions based on the **preceding** tag
  - The basic unit is the preceding tag and the current tag
- **Trigram tagger**
  - Predication based on the previous two tags
  - Expected to have more accurate predictions ….**how?**
    - RB(adverb) VBD(past tense) Vs RB (adverb) VBN(past participle) ?
      - E.g.:
      - "clearly marked"
      - Is clearly marked : P(BEZ RB VBN) > P(BEZ RB VBD)
      - He clearly marked : P(PN RB VBD) > P(PN RB VBN)

An Example Bigram

- *Secretariat/NNP is/VBZ expected/VBN to/TO **race**/VB tomorrow/NN People/NNS continue/VBP to/TO inquire/VB the DT reason/NN for/IN the/DT **race**/NN for/IN outer/JJ space/NN*
  - ✓ to/TO race/???
- $t_i = \text{argmax}_i\ P(t_i|t_{i-1})P(w_i|t_i)$
  - $\max[P(VB|TO)P(race|VB)\ ,\ P(NN|TO)P(race|NN)]$
- Brown reveals that:
  - $P(NN|TO) = .021 \times P(race|NN) = .00041$ **= 0.000007**
  - $P(VB|TO) = .34 \times P(race|VB) = .00003$ **= 0.00001**

Pair and exercise

- The sentences: "bad boys and girls"
  - Structure a) bad (boys and girls)
  - Structure b) (bad boys) and girls
- Develop a CFG grammar that will be used to generate and handle this construction.
- Test your grammar on **nltk**!!!

Markov Model / Markov Chain

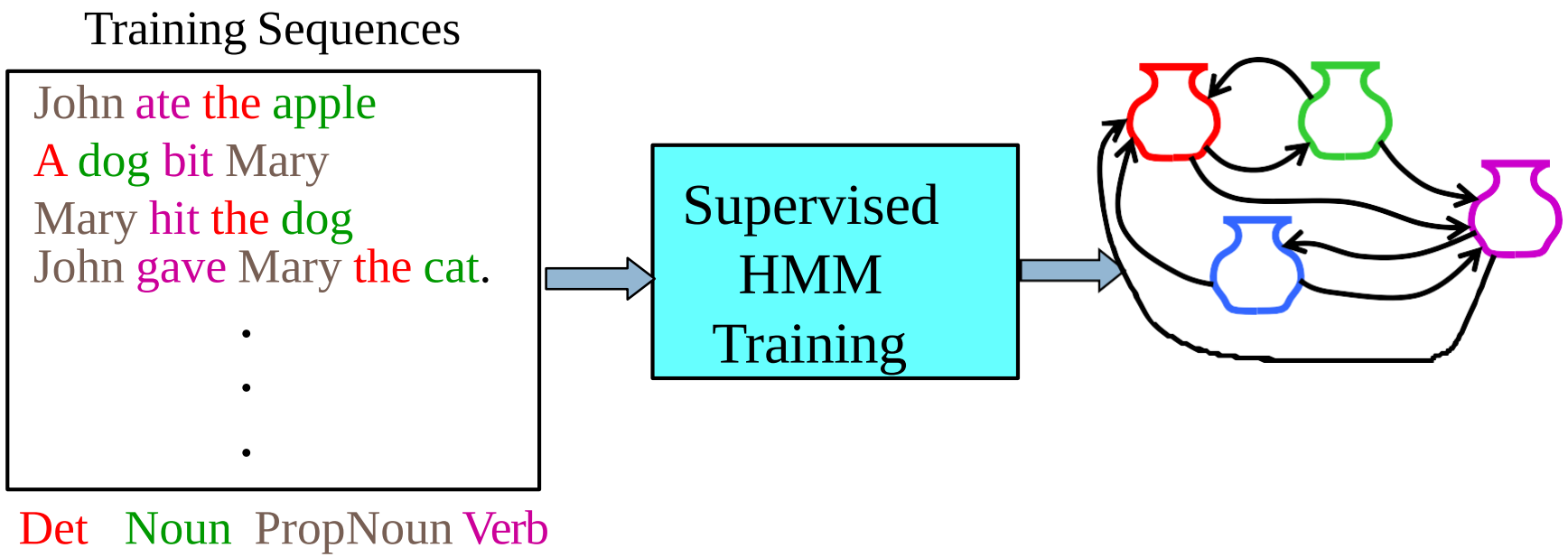
- A finite state machine with probabilistic state transitions.
- Makes Markov assumption that next state only depends on the current state and independent of previous history.

HMM Learning

- **Supervised Learning**: All training sequences are completely labeled (tagged).
- **Unsupervised Learning**: All training sequences are unlabelled (but generally know the number of tags, i.e. states).
- **Semisupervised Learning**: Some training sequences are labeled, most are unlabeled.

Supervised HMM Training

- If training sequences are labeled (tagged) with the underlying state sequences that generated them, then the parameters, λ={A,B} can all be estimated directly.

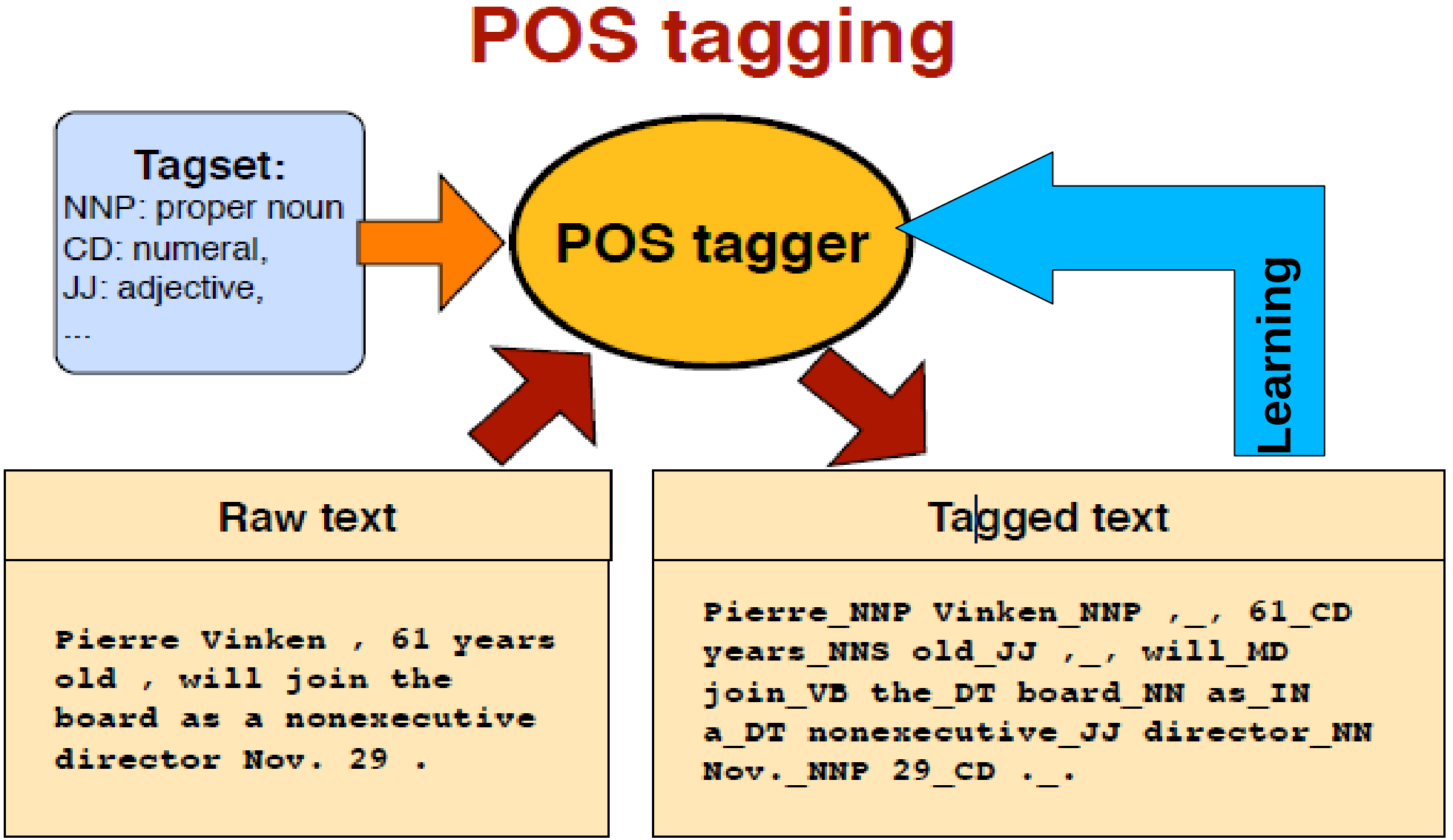


Tagging in NLTK

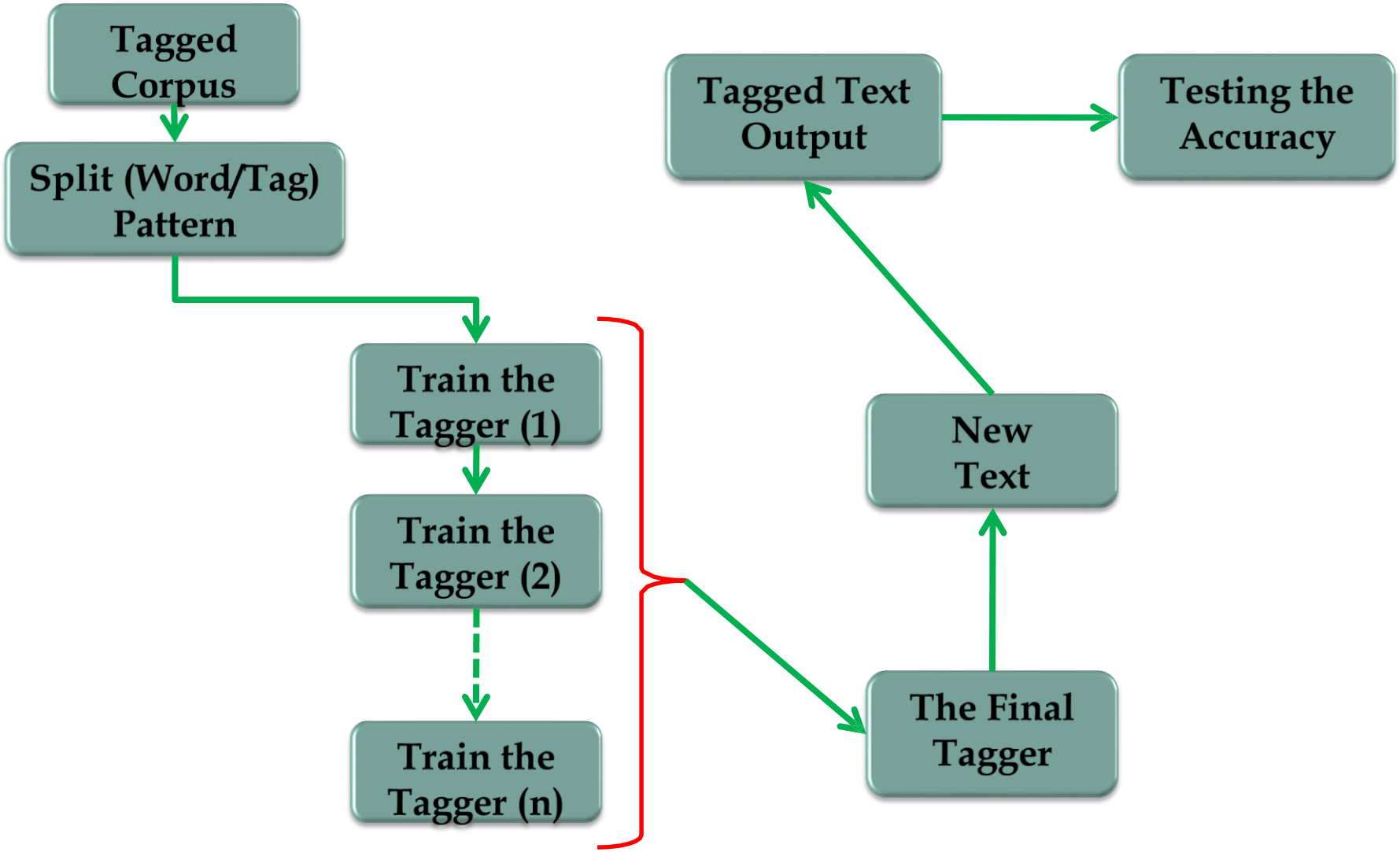
NLTK provides several means of developing a tagger:

- **Default Tagger**: the nltk default tagger works by assigning a default tag to all tokens.
- **Unigram** tagging :
  - assigning the most probable tag
- **Bigram** tagging :
  - assigning the most probable tag given a left-adjacent PoS
- **Regular Expression**: regular expressions (RE) can be used tag a string.
  - The expression should use part of a string to guess its part of speech.

Setting the Scene



Setting the Scene

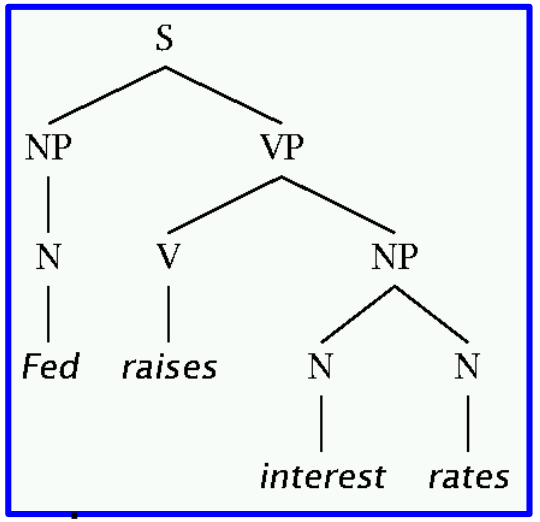


Parsing

- Parsing is the process of recognizing and assigning STRUCTURE
- Parsing a string with a CFG:
  - Finding a derivation of the string consistent with the grammar
  - The derivation gives us a **Parse Tree**

Parsing

- Phrase structure organizes words into nested constituents.
- How do we know what is a constituent?
  - Distribution: a constituent behaves as a unit that can appear in different places:
    - John talked [to the children] [about drugs].
    - John talked [about drugs] [to the children].
    - *John talked drugs to the children about [Wrong]
  - Substitution/expansion:
    - I sat [on the box/right on top of the box/there].
  - Coordination, regular internal structure, no intrusion, fragments, semantics, …



Parsing

- *A parser processes input sentences according to the productions of a grammar, and builds one or more constituent structures that conform to the grammar.*
- *A parser is a procedural interpretation of the grammar. It searches through the space of trees allowed by a grammar to find one that has the required sentence along its edge.*

Parsing

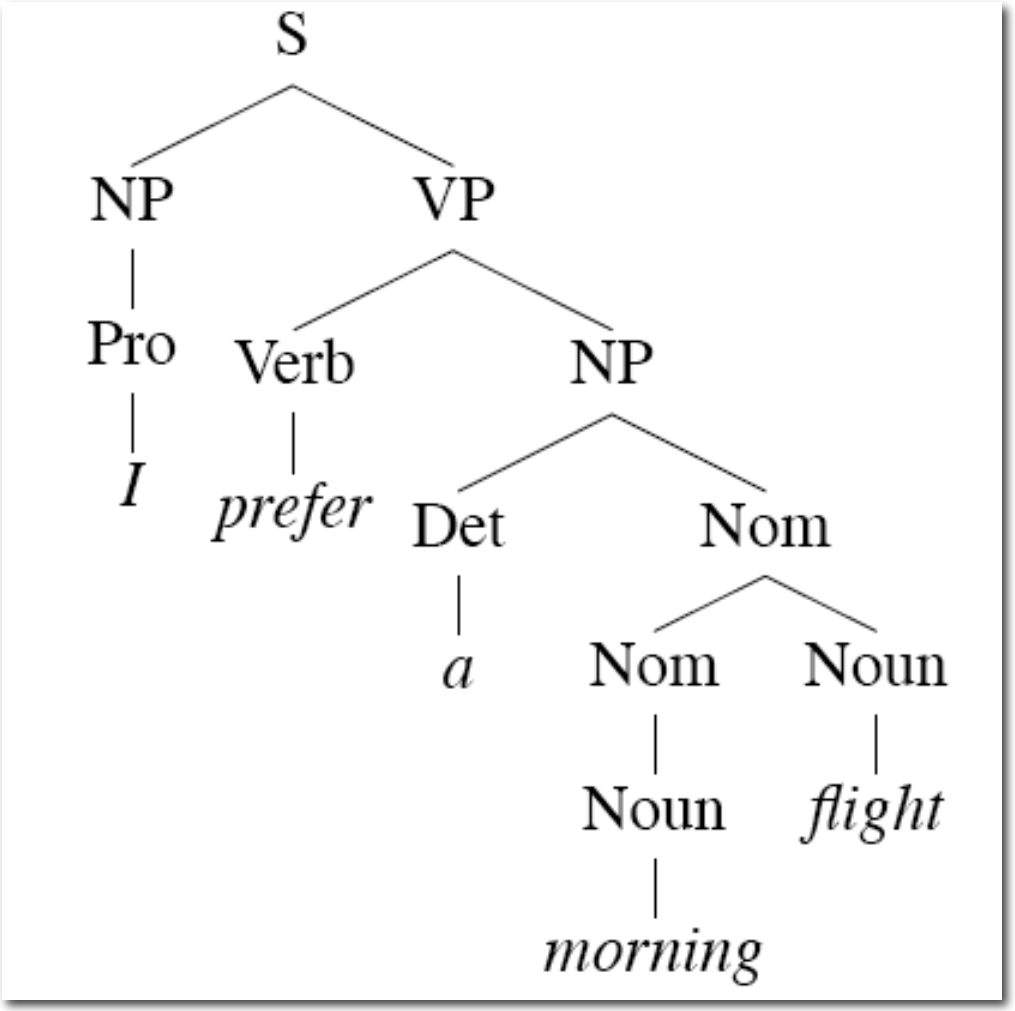
- *Parsing is the process of taking a string and a grammar and **returning parse tree(s)** for that string*
- A parser permits a grammar to be evaluated against a collection of test sentences
- A parser can also be used to check the permissibility of a sentences
- *A parser can serve as a model of psycholinguistic processing, helping to **explain the difficulties that humans have with processing certain syntactic constructions**.*

Parsing as Search

- Search within a space defined by
  - Start State
  - Goal State
  - State to state transformations
- Two distinct parsing strategies:
  - Top down
  - Bottom up
- Different parsing strategy, different state space, different problem.

Derivations

- A *derivation* is a sequence of rules applied to a string that *accounts* for that string (sequence of words)
  1. **Covers all the elements in the string**
  2. **Covers only the elements in the string**



Top-Down Parsing Method

- Recursive Descent Parsing
  - ✓ break a high-level goal into several lower-level sub-goals
  - ✓ First question will be how to break the top level goal?
  - ✓ The top-level goal is to find an **S …… Sentences**.
  - ✓ For the grammar, the $S \rightarrow NP\ VP$ production permits the parser to replace this goal with two sub-goals:
    - ✓ *find an NP, then*
    - ✓ *find a VP.*
    - ✓ *Then replace VP and NP with others until we reach a terminal*

Top-Down Parsing Method

- Recursive Descent Parsing
  - ✓ Keep doing this until a terminal is found and compare the terminal with the input string.
    - ✓ If no match then backup and look other alternatives
  - ✓ Once a parse has been found, we can get the parser to look for additional parses.
    - ✓ … in case the sentences has more than one possible structure
  - ✓ Top-down parsers use a grammar to predict what the input will be, before inspecting the input.
    - ✓ Check the part of speech before the word itself
  - Demo: **nltk.app.rdparser()**
- Recursive Descent Parsing in NLTK:
  - **nltk.RecursiveDescentParser(yourGrammar)**

Bottom-Up Parsing Method

Shift-Reduce Parsing

- shift-reduce parser tries to find sequences of words and phrases that correspond to the right hand side of a grammar production, and replace them with the left-hand side, until the whole sentence is reduced to an **S**.
- Since the input is available to the parser all along, it would be more sensible to consider the input sentence from the very beginning.
- This approach is called bottom-up parsing

Bottom-Up Parsing Method

- Shift–reduce parsing is a bottom up derivation strategy, that is, it starts from the words in the string, and tries to work upwards towards the root symbol in the grammar.
- parse(sent):
  - if sent is [S] then finish
  - otherwise, for every rule, check if the RHS of the rule matches any substring of the sentence
  - if it does, replace the substring in the LHS of the rule
  - continue with this sentence
- Demo: **nltk.app.srparser()**

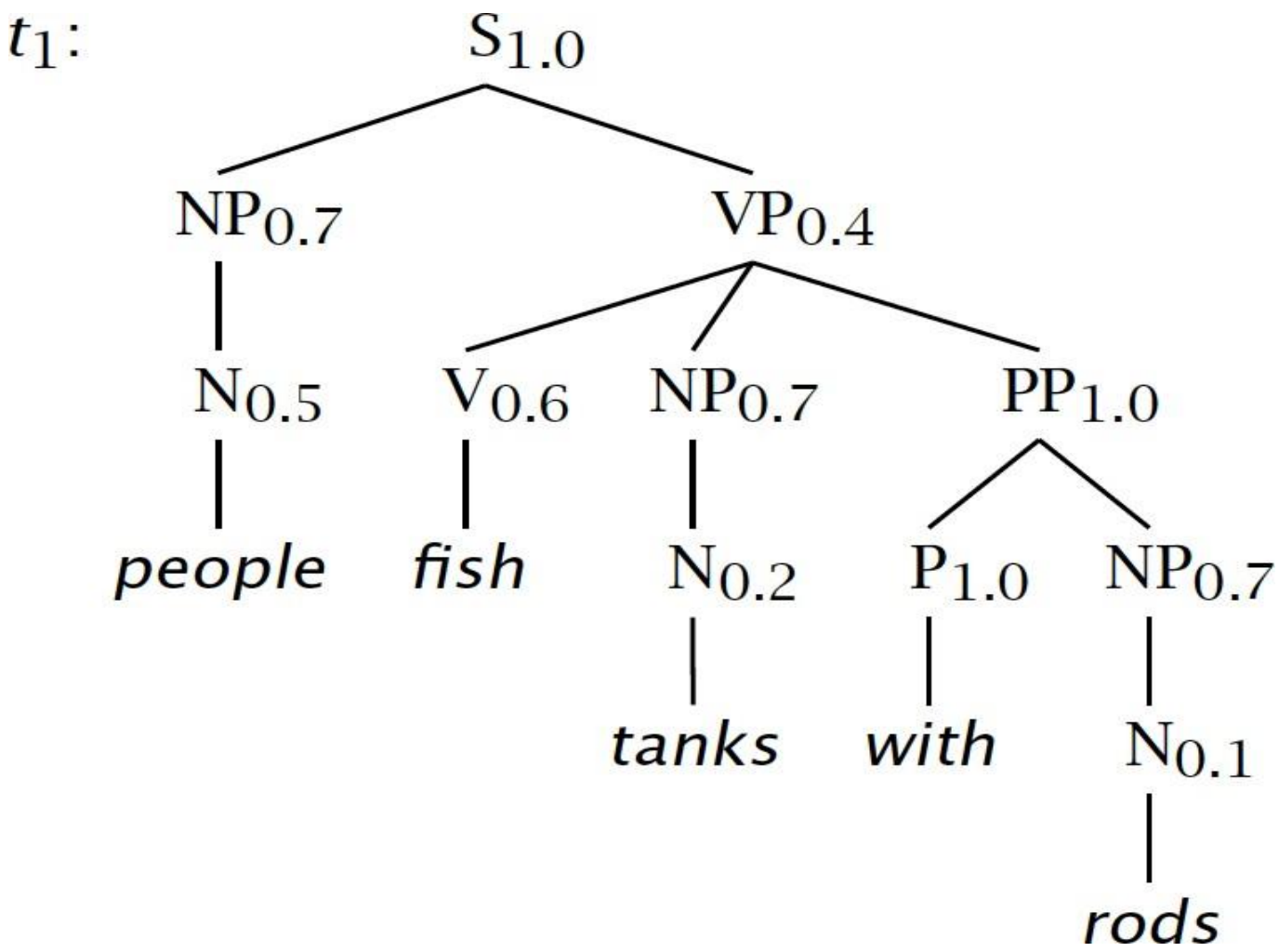
A PCFG (Probabilistic)

Rule	Prob.
S → NP VP	1.0
VP → V NP	0.6
VP → V NP PP	0.4
NP → NP NP	0.1
NP → NP PP	0.2
NP → N	0.7
PP → P NP	1.0

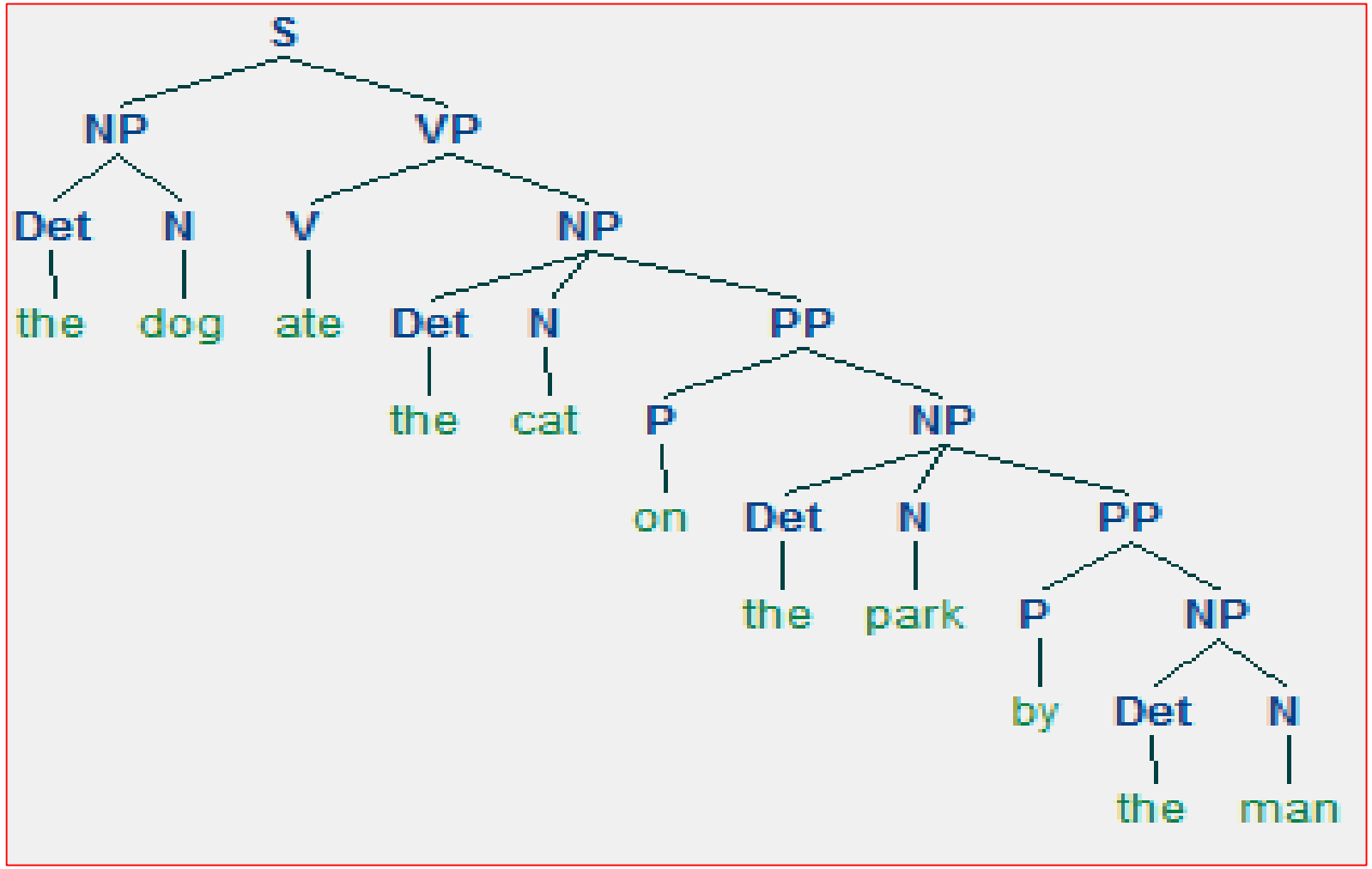
Rule	Prob.
N → *people*	0.5
N → *fish*	0.2
N → *tanks*	0.2
N → *rods*	0.1
V → *people*	0.1
V → *fish*	0.6
V → *tanks*	0.3
P → *with*	1.0

*Note that the sum of the probability of each category (word class) is **1***

A PCFG Tree


t_1:
$S_{1.0}$
$NP_{0.7}$
$VP_{0.4}$
$N_{0.5}$
$V_{0.6}$
$NP_{0.7}$
$PP_{1.0}$
people
fish
$N_{0.2}$
$P_{1.0}$
$NP_{0.7}$
tanks
with
$N_{0.1}$
rods

CFG Parsing in NLTK



Probabilistic Grammar

S → NP VP	0.8
S → Aux NP VP	0.1
S → VP	0.1
NP → Pronoun	0.2
NP → Proper-Noun	0.2
NP → Det Nominal	0.6
Nominal → Noun	0.3
Nominal → Nominal Noun	0.2
Nominal → Nominal PP	0.5
VP → Verb	0.2
VP → Verb NP	0.5
VP → VP PP	0.3
PP → Prep NP	1.0

Top Down vs Bottom Up Searching

- The search has to be guided by the INPUT and the Grammar
- TOP-DOWN search: the parse tree has to be rooted in the start symbol S
  - EXPECTATION-DRIVEN parsing
- BOTTOM-UP search: the parse tree must be an analysis of the input
  - DATA-DRIVEN parsing

Applications of parsing

- Machine translation (Alshawi 1996, Wu 1997, ...)

- Speech synthesis from parses (Prevost 1996)

  The government plans to raise income tax.
  The government plans to raise income tax the imagination.

- Speech recognition using parsing (Chelba et al 1998)

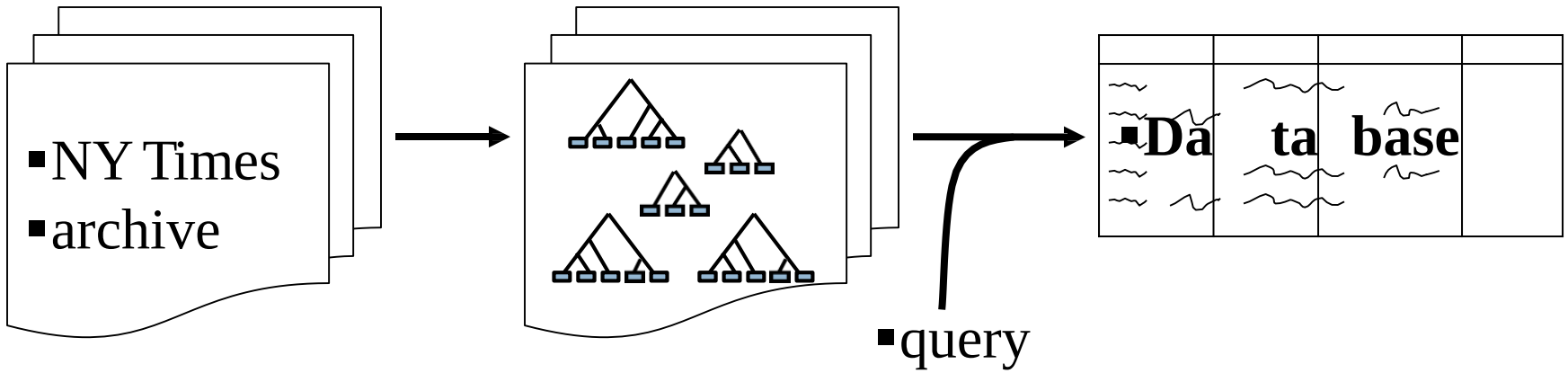
  Put the file in the folder.

  Put the file and the folder.

Applications of parsing

- Grammar checking (Microsoft)
- Indexing for information retrieval (Woods 1997)

  ... washing a car with a hose ... ⟶ vehicle maintenance
- Information extraction (Hobbs 1996)



Practical Sessions of this Lecture!

Let's try to do:

CFG in NLP

POS Tagging in NLP

N-grams in NLP

Plagiarism Checker in NLP