Natural Language Processing
自然語言處理

黃瀚萱
Department of Computer Science
National Chengchi University
2020 Fall

Lesson 2

Linguistic Essentials

Schedule

Date	Topic
9/16	Introduction
9/23	**Linguistic Essentials**
9/30	Collocation
10/7	Language Model
10/14	Word Sense Disambiguation
10/21	NLP and Cybersecurity
10/28	Text Classification
11/4	POS Tagging
11/11	Midterm Exam

Schedule

Date	Topic
11/18	Chinese Word Segmentation
11/25	Word Embeddings
12/2	Neural Networks for NLP
12/9	Parsing
12/16	Discourse Analysis
12/23	Invited Talk
12/30	Final Project Presentation I
1/6	Final Project Presentation II
1/13	Final Exam

Agenda

- Core issue of NLP
- Basic linguistic concepts
  - Part-of-speech （詞性） and morphology（構詞）
  - Syntax （句法）
  - Semantics （語意） and pragmatics （語用）
- Basic text processing
  - Tokenization
  - Stemming
  - Lemmatization
  - POS Tagging

Core Issue of NLP

Ambiguity (歧義性)

- Possibility of multiple interpretations of a language use
  - The core issue in NLP
  - Highly inherited in real languages at all levels
    - It is hard to process with rule-based approaches.
- Humans always disambiguate (resolve ambiguity 消歧、解歧) in real time.

Ambiguity in Human Languages

- I made her duck.

  - I cooked a water bird for her.
  - I cooked the water bird belonging to her.

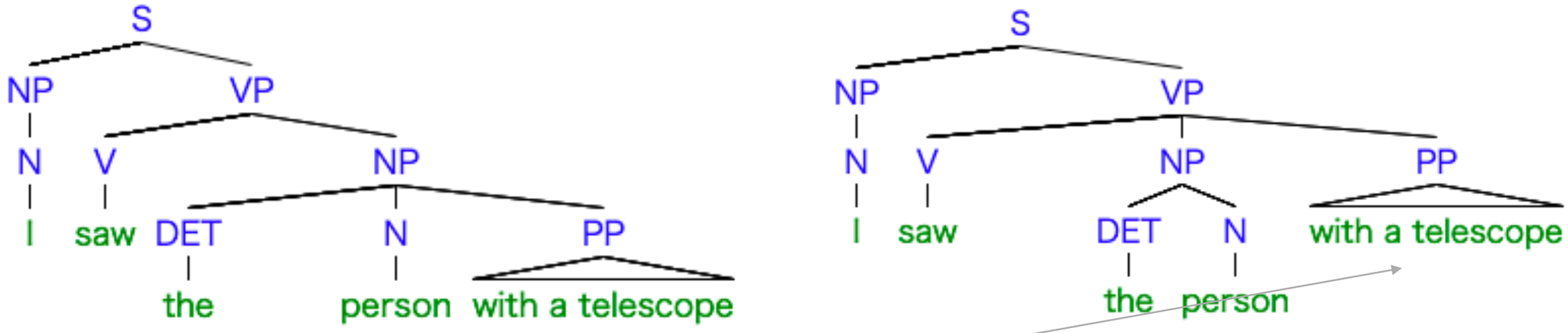
- I created the wooden duck for her.
- I caused her to quickly lower her head.
- I waved my magic wand and turned her into an undifferentiated water bird.

Reasons of the Ambiguity

- “her”
    - Dative pronoun (與格的人稱代詞)
    - Possessive pronoun (所有格).
- “duck”
    - Noun: the water bird
    - Verb: lower someones’ head or body to avoid a hit.
- “made”
    - Transitive verb: cooked someone’s water bird.
    - Ditransitive verb: cooked a water bird for someone.
    - Taking a direct object and a verb: caused someone to duck.

Syntactical Ambiguity

- I saw the person with a telescope.
  - I saw the person who was carrying a telescope
  - I saw the person through a telescope.
- I saw the person with a bag. <= Less ambiguous.



prepositional phrase (介繫詞片語)

http://mshang.ca/sy

Contextuality

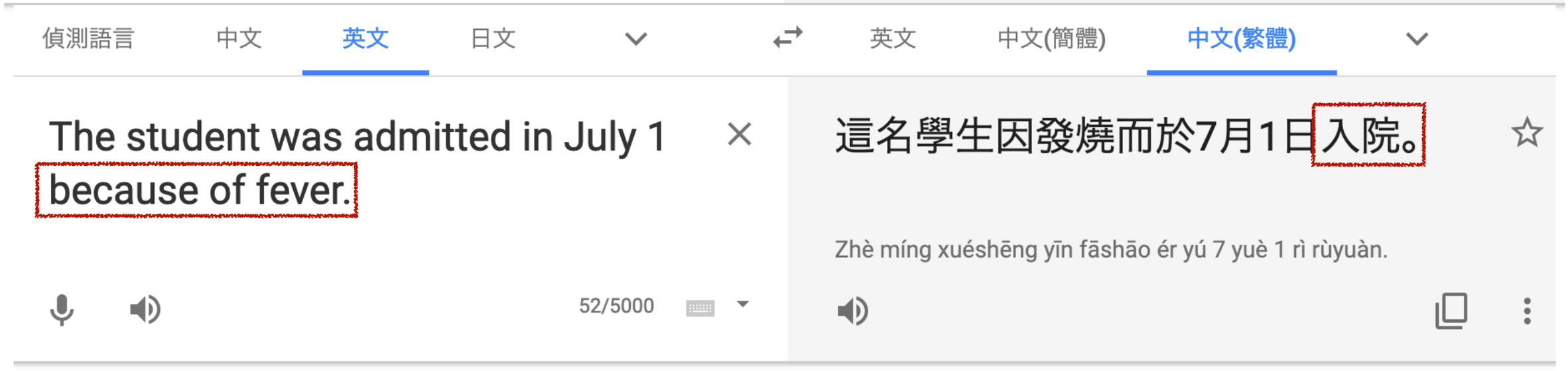
- The meaning of a use of language may be resolved only with contextual information.

- The kid was **admitted** in July 1.

- 這個小孩於七月1日入學

- 這個小孩於七月1日入院

Ambiguity in MT



Sources of Ambiguity

- Syntax
  - Part of speech (詞性) ambiguities (duck)
  - Attachment ambiguities (with a telescope)
- Semantics
  - Word sense ambiguities (admit)

Part I

Basic Linguistic Concepts

Training Stage

Training Data

(Arg1, Arg2)
Instances

Labels

Feature
Extraction

Feature Vectors

Learning Model

Naive Bayes

Logistic Regression

SVM

Neural Network

(Arg1, Arg2)
Instance

New Data

Bag of word

Word embedding

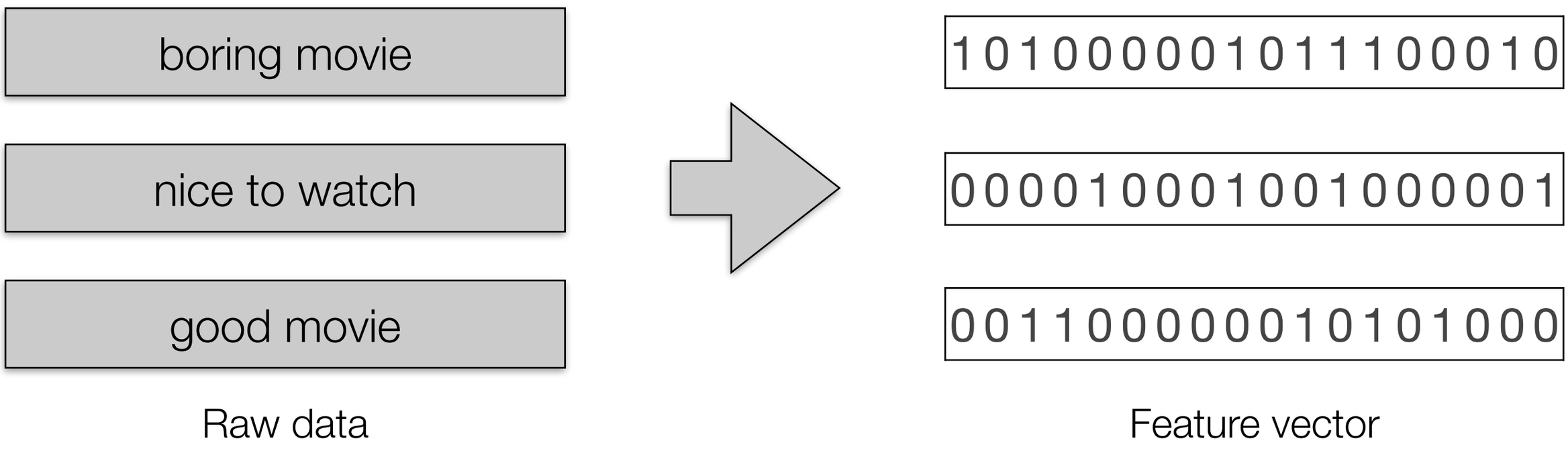
Sentence Encoder

Predicted
Label

Productive Stage

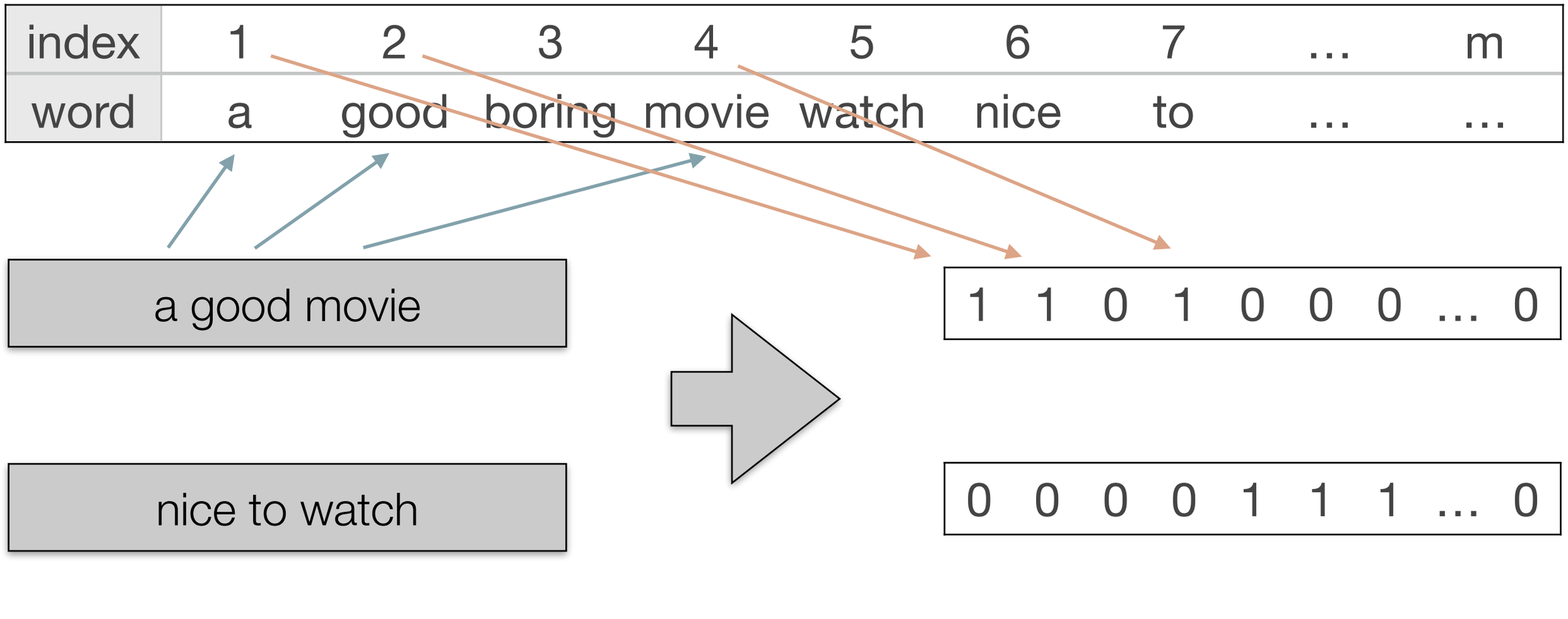
Feature Extraction

- Handcrafted linguistic features
  - Proposed by human
- Directly learning the representation from data
  - Co-optimized by data and the model



Bag of Word Representation

- Each word has a specific cell on the feature vectors
- The presence or absence of each word is denoted by the value of the cell.



Do NLP in 2010

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB

count_vect = CountVectorizer()
X_train = count_vect.fit_transform(twenty_train.data)

clf = MultinomialNB().fit(X_train, twenty_train.target)


docs_test = ['God is love',
             'OpenGL on the GPU is fast']

X_test = count_vect.transform(docs_test)
predicted = clf.predict(X_test)
```

https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html

Do NLP in 2020

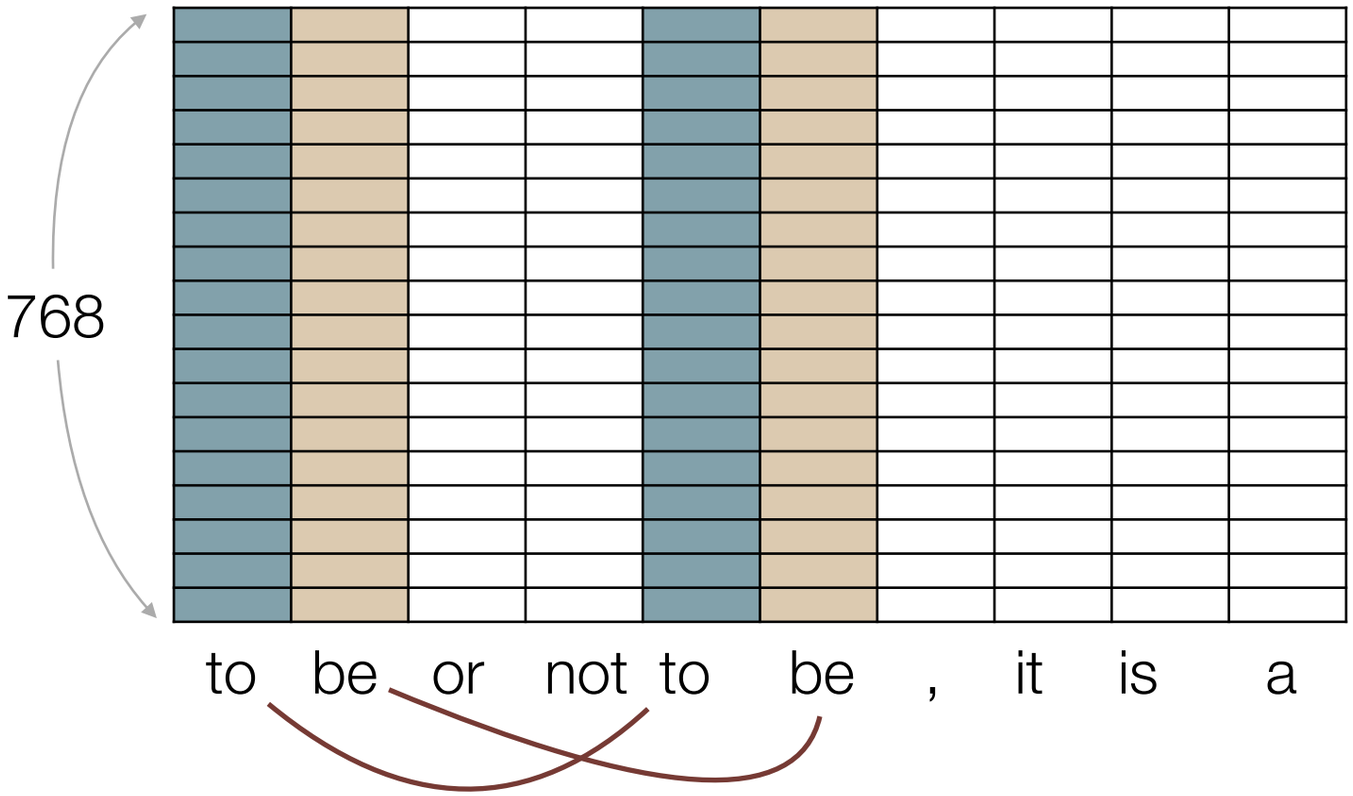
```
from simpletransformers.classification import ClassificationModel

train_data = [
    ['Example sentence belonging to class 1', 1],
    ['Example sentence belonging to class 0', 0]]
train_df = pd.DataFrame(train_data)
eval_data = [
    ['Example eval sentence belonging to class 1', 1],
    ['Example eval sentence belonging to class 0', 0]]
eval_df = pd.DataFrame(eval_data)

model = ClassificationModel('bert', 'bert-base')

model.train_model(train_df)

result, model_outputs, wrong_predictions = model.eval_model(eval_df)
```

https://github.com/ThilinaRajapakse/simpletransformers#minimal-start-for-binary-classification

Google BERT Encoder

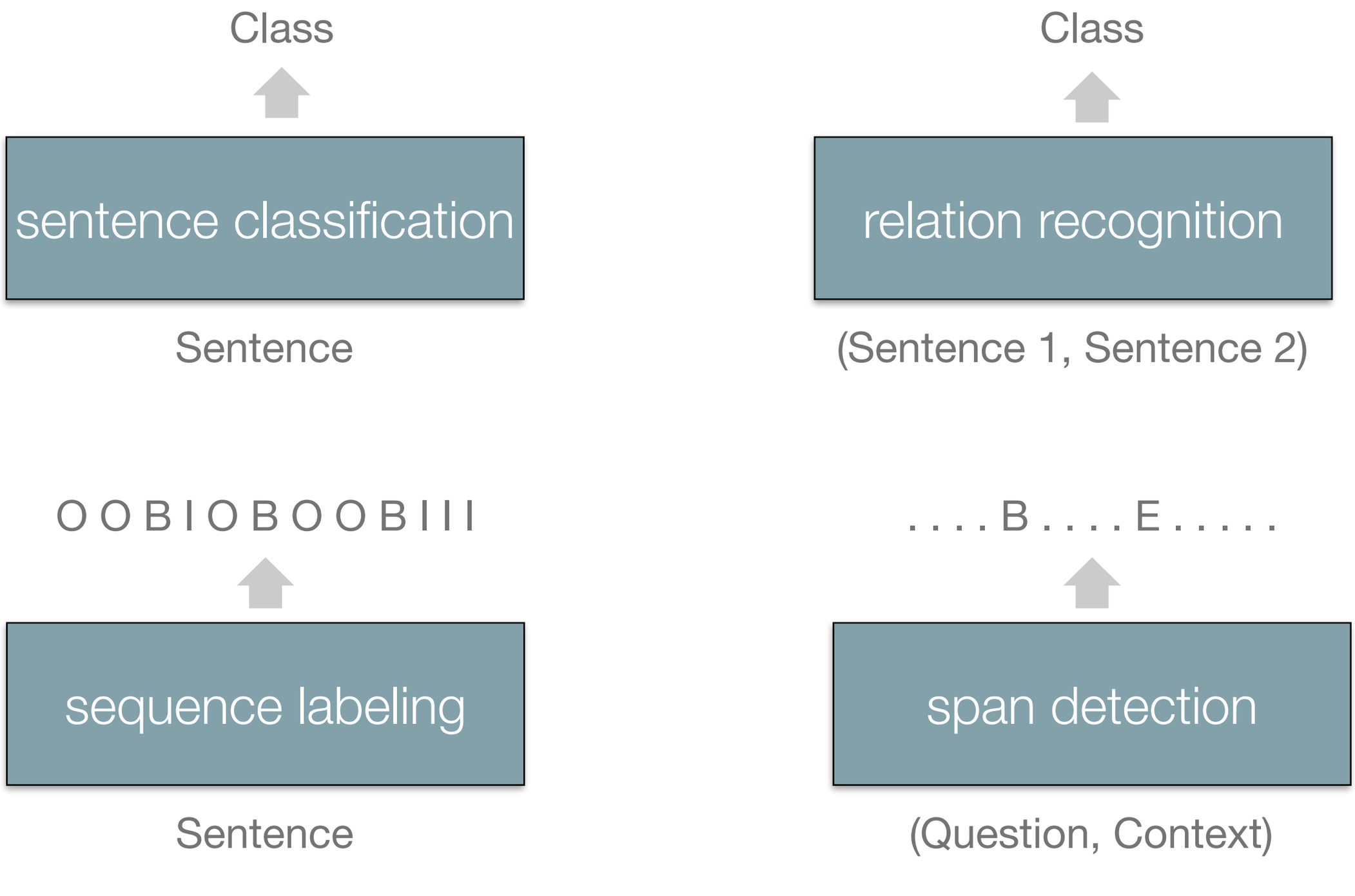
- Each sentence with *n* words is represented as a matrix with the dimension of 768 * *n*

- Words from more than 100 languages are projected to a single space.



The embedding of the same word differs according to the context

Four Fundamental Tasks in NLP

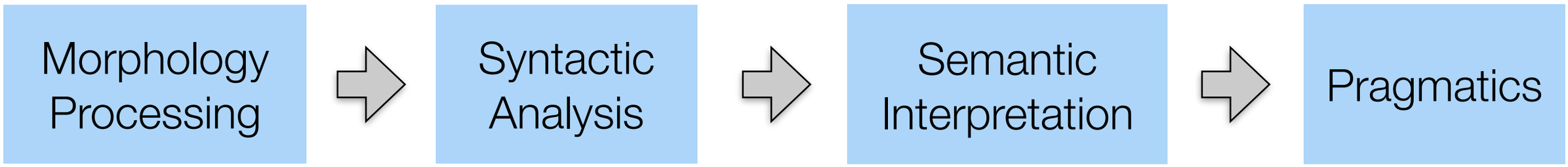


Levels of Knowledge

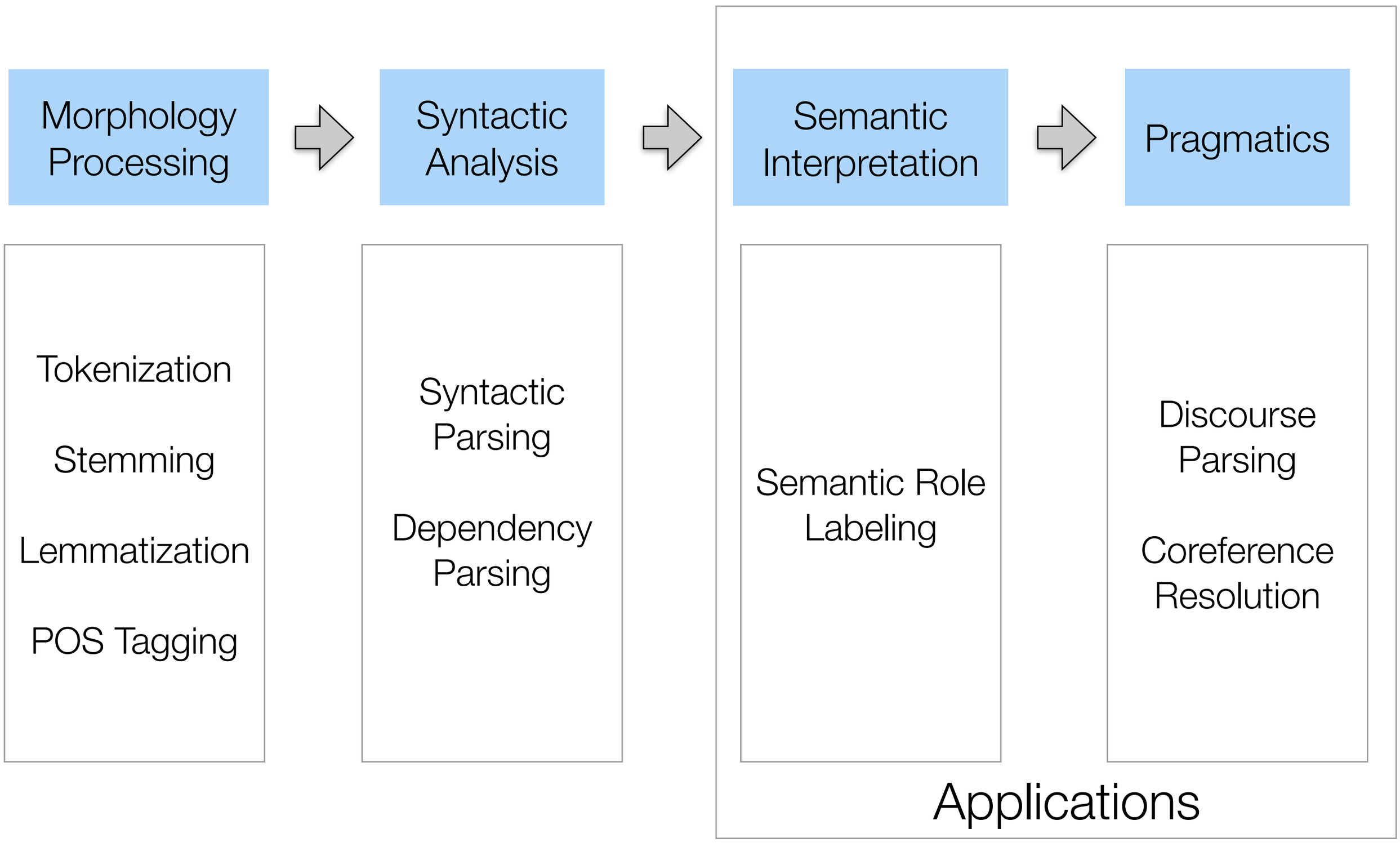
- Each kind of knowledge has associated with it a set of processes that make use of it.

- The process is often in a pipelined fashion.

  - tokenization -> POS tagging -> parsing…



Pipeline Architecture



Preprocessing Tasks

Tokenization

Sentence Segmentation

Part-of-Speech Tagging

Syntactic / Dependency Parsing

Stance Detection

Opinion Mining

Fake News Detection

Stock Prediction

…

Morphological Issues

- Upper and lower case
  - Should we regard black, Black, and BLACK the same?
  - Should we treat the words 峰 and 峯 differently?

Tokenization

- What is a word or a term?
  - Whitespace does not always work.
    - I’ve read the book.
    - I can’t do that.
- “Taipei City” or “Taipei” and “City”
- “Taipei City Hall”, “Taipei City” and “Hall”, or “Taipei” and “City Hall”.
- Chinese word segmentation

Hyphenation

- email or e-mail?
- cooperate or co-operate?
- **neural-network based approach** or **neural network-based approach?**

Part of Speech (詞性)

- Classification of words into a number of syntactic or grammatical categories.

Children eat sweet candy.

Noun: People

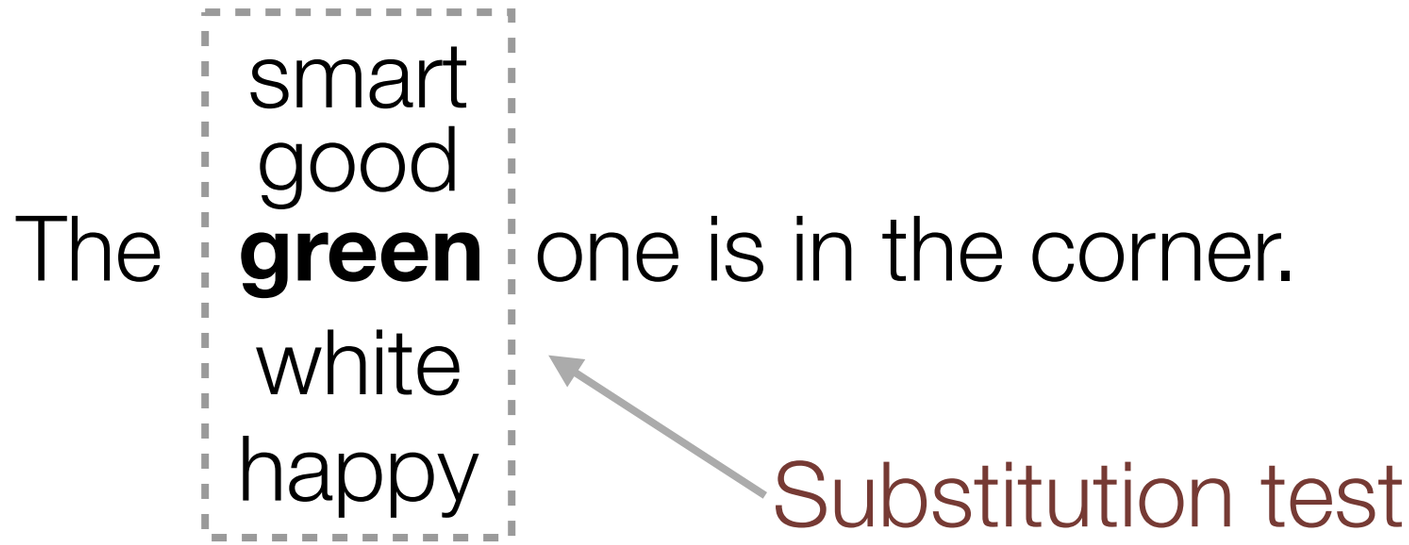
Verb: Action what *children* do with *candy*

Adjective: Property of *candy*

Noun: Type of Food

Three Important Parts of Speech

- Noun (名詞): People, animals, concepts, and things.
- Verb (動詞): Action
- Adjective (形容詞): Properties of Nouns
- Substitution test
  - Words belong to the same part of speech are grammatically interchangeable.



Words with Multiple POS

- kid
  - Noun: a child
  - Verb: to make jokes
  - Adjective: younger [sister/brother]
- though
  - Conjunction (連接詞): *Though I am almost 40, I still want to compete.*
  - Adverb (副詞): *Jason did one nice thing, though.*

Open Word Class vs Closed Word Class

Noun
kid machine book table chair cup award kettle screen power time bat speaker bike …

Verb
run hit jump climb improve swim fly fight perform cook build establish …

Adjective
red black bad nice good funny interesting cute fine delicious positive …

Open word classes

Conjunction
though and but while because …

Preposition
in on at with through …

Pronoun
you I me her they them …

Closed word classes

Morphological Process: Inflection

- Systematic modifications of a root form by adding prefixes and suffixes to indicate grammatical distinctions like singular (單數) and plural (複數).

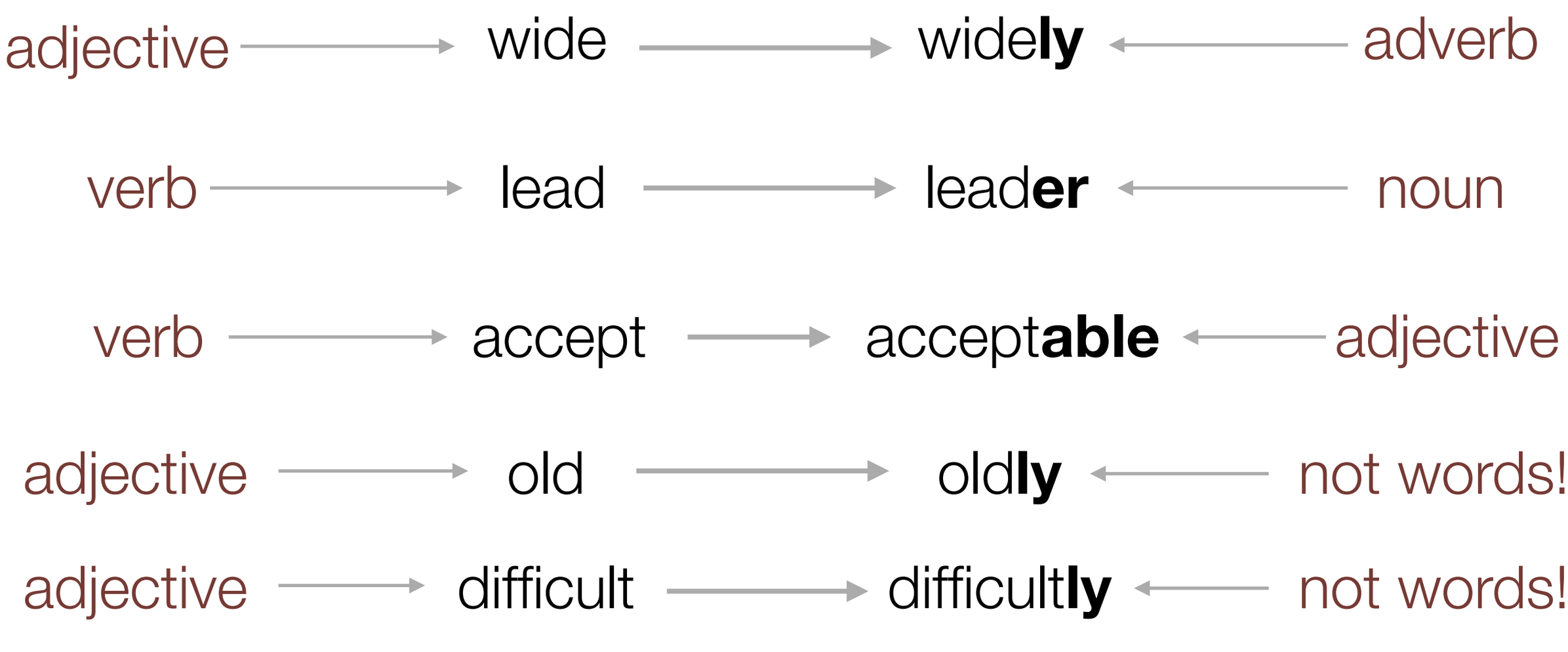
- Do not significantly change the word class or meaning.

dog → dog**s**

fish → fish**es**

Morphological Process: Derivation

- Less systematic
- Often involving a change in meaning and POS.



Morphological Process: Compounding

- Merging two or more words into a new word that denotes a single semantic concept.
- Noun-noun compounds (合成詞)
    - teacup
    - disk drive
    - headquarters
- Other compounds
    - downmarket (adjective + noun)
    - mad cow disease (狂牛症) (adjective + noun + noun)

Wordform Grouping

- Stemming (詞幹提取)
  - Stripping off affixes of a word
  - booked => book, books => book, book => book
- Lemmatization (字根還原)
  - Finding the lemma or lexeme of which one is looking at an inflected form.
  - ran => run

Stemming

- It is often taken to be a crude error that a stemming algorithm does not leave a real word after removing the stem.

- The purpose of stemming is to bring variant forms of a word together, not to map a word onto its root form.

- decomposing => decompos
- decomposes => decompos
- decomposed => decompos
- decompose => decompos

Lemmatization

- Removing the inflections and finding the word's root form.
  - decomposing => decompose
  - decomposes => decompose
  - decomposed => decompose
  - decompose => decompose

Noun and Pronouns (代名詞)

- Refer to entities in the world.
    - People, animals, and things.
- In English, only one inflection of the noun
    - Plural form vs Singular form
- Gender inflection in the third person singular pronoun
    - he, she, and it

Determiners (冠詞)

- Describing the particular reference of a noun.
  - Articles
    - a/an: Indicates the person/thing was not previously mentioned.

*A boat on the sea with clouds. **A** fisher stands checking her equipments.*

    - the: Already made reference to the noun, or if the reference is clear from context.

*A **boat** on the sea with clouds. **The** fisher stands checking her equipments.*

Verbs

- Verbs are used to describe actions.
- Usually the most important word in a sentence.

Form	Regular	Irregular
root / base	walk	write
Third singular present	walks	writes
Gerund / present particle	walking	writing
Past tesne	walked	wrote
past/passive particle	walked	written

Adverbs

- Adverbs modify a verb in the same way as the adjectives modify nouns.
    - Place: The guide finds a restaurant locally
    - Time: She often visits to hospital.
    - Manner: He grabbed her roughly.
    - Degree: I completely forgot that it’s his birthday today.

Other Parts of Speech

- Prepositions are mainly small words that prototypically express spatial relationships.
    - to, of, on, in…
- Conjunctions
    - Coordinating conjunctions: and, but, or, …
    - Subordinating conjunctions: because, although, that…

Full Part-of-Speech Tags

CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential *there*
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	*to*
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

Sizes of Tag Sets

- Penn Treebank, most widely used in computational work, is a simplified version of the Brown tag set.
- Many tag sets for other languages have also been developed.

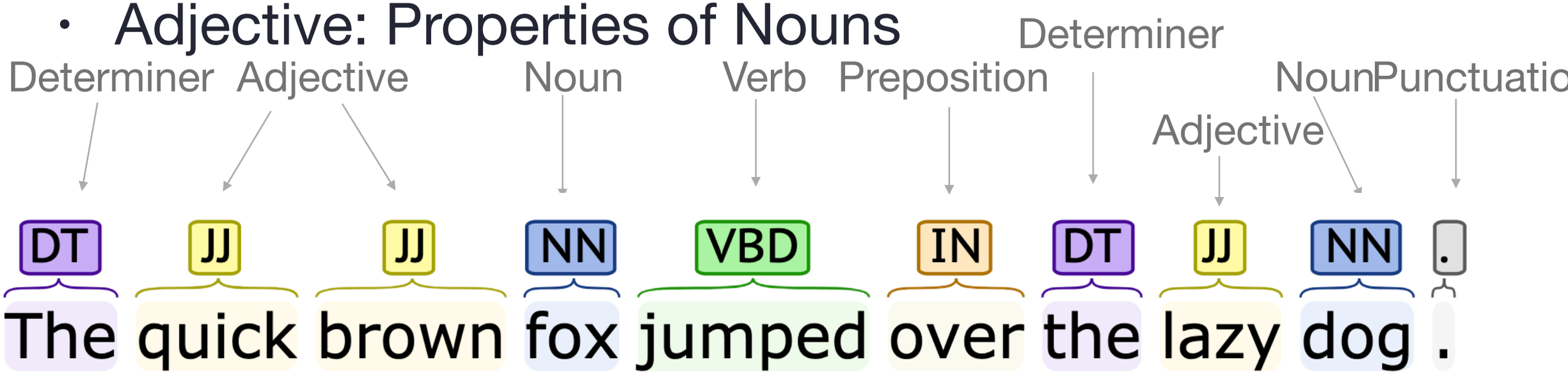
Set	Number of Tags
Brown	179
Penn	45
CLAWS1	132
CLAWS2	166
CLAWS c5	62
London-Lund	197

Different Tag Sets

Words	CLAWS c5	Brown	Penn TB	ICE
She	PNP	PPS	PRP	PRON
was	VBD	BEDZ	VBD	AUX
told	VVN	VBN	VBN	V
that	CJT	CS	IN	CONJUNC
the	AT0	AT	DT	ART
journey	NN1	NN	NN	N
might	VM0	MD	MD	AUX
kill	VVI	VB	VB	V
her	PNP	PRO	PRP	PRON
.	PUN	.	.	PUNC

Part-of-Speech Tagging

- Classification of words into a number of syntactic or grammatical categories. (詞性)
    - Noun: People, animals, concepts, and things.
    - Verb: Action
    - Adjective: Properties of Nouns



corenlp.ru

Ambiguity in POS Tagging

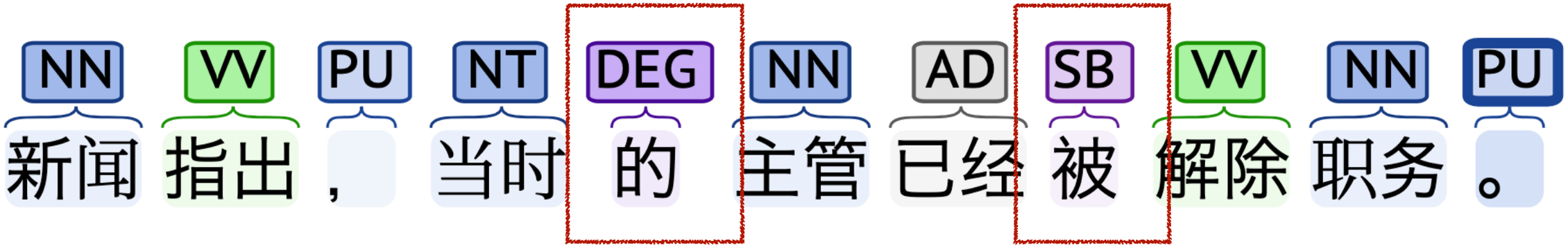
Original	I	made	her	duck
Tagging 1	Pronoun	Verb	Pronoun	Noun
Tagging 2	Pronoun	Verb	Possessive pronoun	Noun
Tagging 3	Pronoun	Verb	Pronoun	Verb

Part-of-Speech in Chinese

- Special part-of-speech tags are used in Chinese according to the Chinese grammar.

- Many toolkits for Chinese processing are based on Simplified Chinese.

- It is easy to perform the conversion from Traditional Chinese to Simplified



Sentences

- What is a sentence?
  - “something ending with period (.), question mark (?), or exclamation mark (!)”
- Multiple usages of the punctuation mark period.
  - Abbreviation (Dr. Ms. Mr.)
  - 3.1415926…

Sentences in Irregular Forms

- The scene is written with a combination of unbridled passion and sure-handed control: In the exchanges of the three characters and the rise and the fall of emotions, Mr. Weller has captured the heartbreaking inexorability of separation.

- “you remind me,” she remarked, “of your mother.”

    - Nested and not nicely sequential.

Sentence Boundaries

- Periods (.) are not only used for ending sentences.
  - Dr. Tsai was visiting patients.
  - I love the fruits like apple, banana, etc. and the meats like beef, pork, etc.
- The ill writing without the proper usage of punctation marks.
  - 這是有點霸道，但也有道理，因為他們是 上市公司，每一季要向美國證管會報告總公 司、附屬公司及子公司的營運及財務狀況， 帳都是照一套會計原則來做，所以很多時候 他們的要求，是出自一種單純的需要，而並不是故意要來欺負我們。
- The informal writing without punctuation marks.
  - 火星文並沒有嚴謹的定義運用上也不侷限於網路用語通常只要讓人無法立即判定理解的文句皆可泛稱為火星文

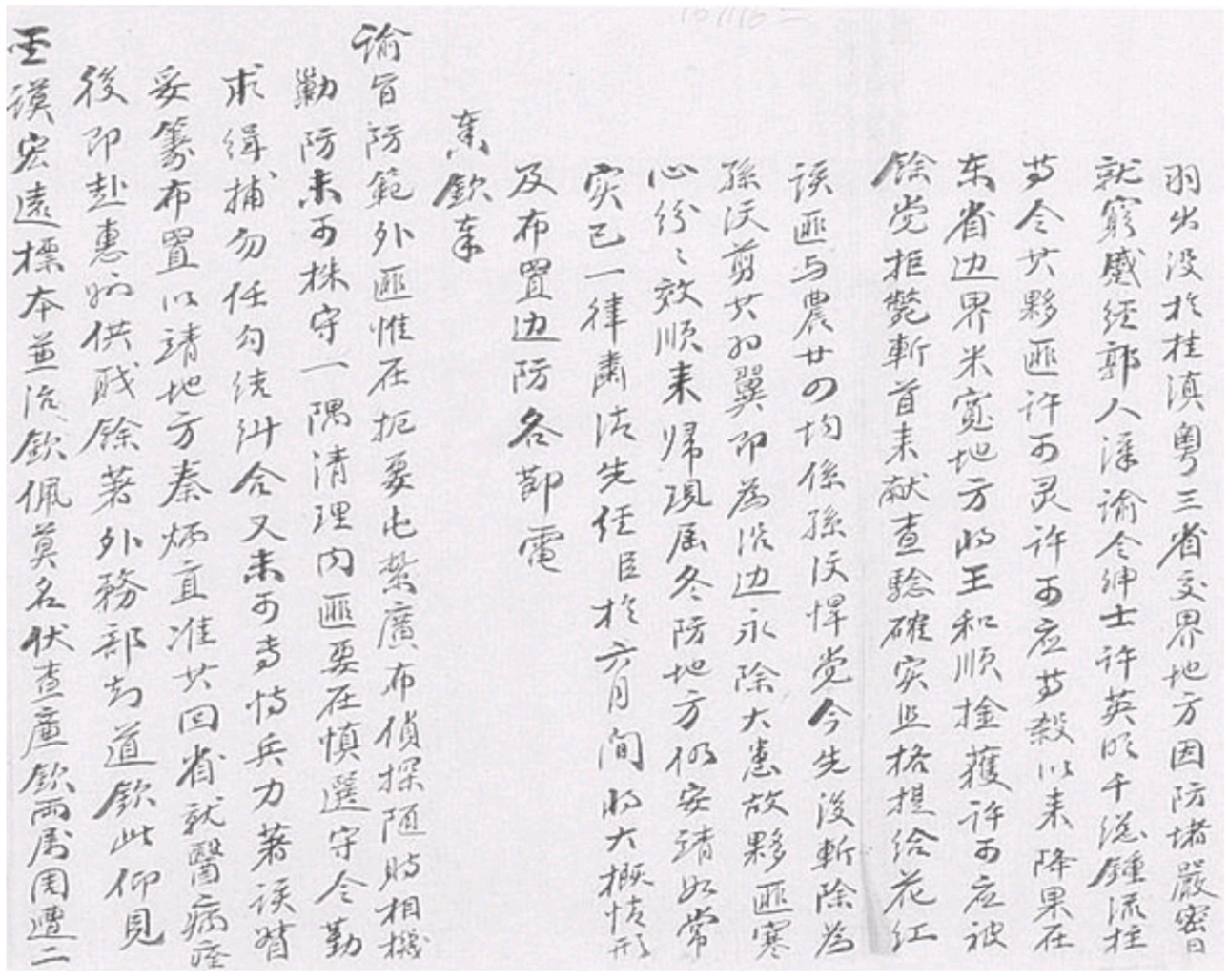
Abbreviation

Abbreviation

Double Roles

Fullstop

Classical Chinese Processing

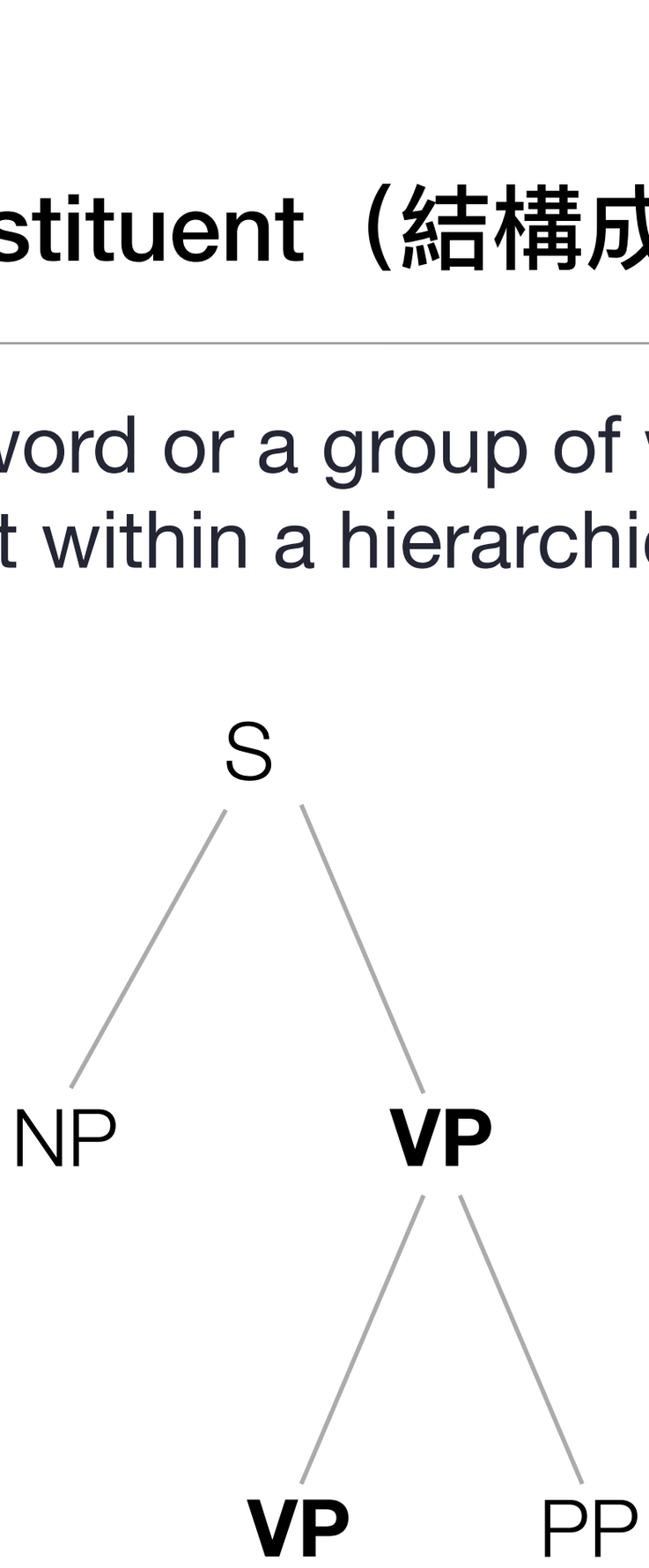


Word Order: Syntax

- Languages have constraints on word order.
  - the dog ate my cake
  - ate cake dog my the
  - the cake ate my dog
- A sequence of words without a proper order results an uninterpretable sentence.
- Syntax: the study of the regularities and constraints of word order and phrase structure.

Constituent（結構成份）

- A word or a group of words that functions as a single unit within a hierarchical structure.



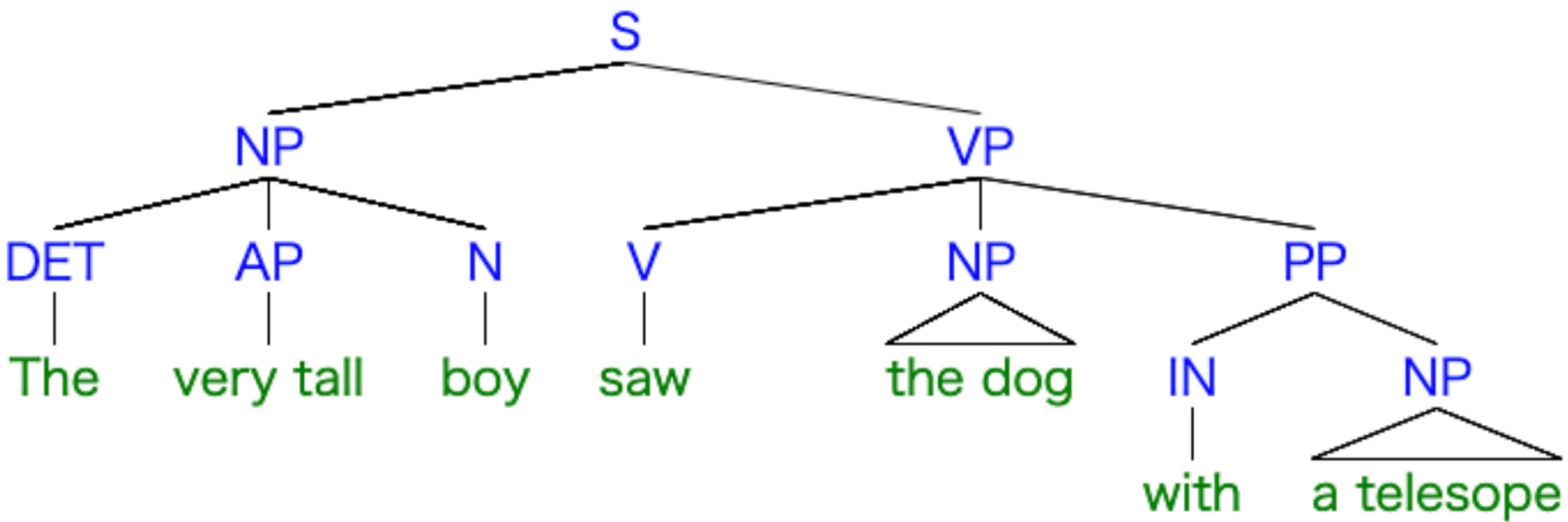
S → NP VP

NP → N
NP → DET N
NP → DET AP N
NP → **NP** PP

VP → V NP
VP → **VP** PP

Major Phrase Types

- S: whole sentence (S → NP VP)
- NP: noun phrase (名詞片語)
- PP: prepositional phrase (介繫詞片語)
- VP: verb phrase (動詞片語)
- AP: adjective phrase (形容詞片語)



Noun Phrases (NPs)

- A syntactic unit of the sentence, in which information about the noun is put together.

- The noun is the **head** of a noun phrase.

- NPs are often arguments of verbs.

  - Subject of an action

  - Object of an action

NP → N
NP → DET N
NP → DET AP N
NP → NP PP
…

Verb Phrases (VPs)

- Organizing all elements of the sentence that grammatically depend on the verb.
- The verb is the head of a verb phrase.
- The core of a sentence.

VP → V
VP → V NP
VP → V NP NP
VP → V NP AP
VP → VP PP
…

Adjective Phrases (APs)

- Complex adjective phrases are rare.

  *He is **pretty sure of himself**.*
  *She seemed a girl who was **quite certain to succeed**.*

AP → JJ (Adjective)
AP → AD (Adverb) JJ
AP → AP AP
…

Prepositional Phrases (PPs)

- Co-occur with NP, VP, and AP.
- PP → IN (Preposition) NP
- Usually used to express spatial and temporal locations and other attributes.

The boy saw the dog ***with a telescope****.*

I joined the conference held ***in the last year****.*

The cup ***on the table*** *is hers.*

Rewrite Rules

- The regularities of word order are formulated by rewrite rules.
- The unit in the left side can be rewritten as the sequence of units in the right side.

S → NP VP
NP → DET NN
NP → NP PP
VP → VP PP
VP → V NP
PP → IN NP

DET → the | a | an
NN → cat | mouse | egg
V → ate | slept | ran
IN → in | of | on

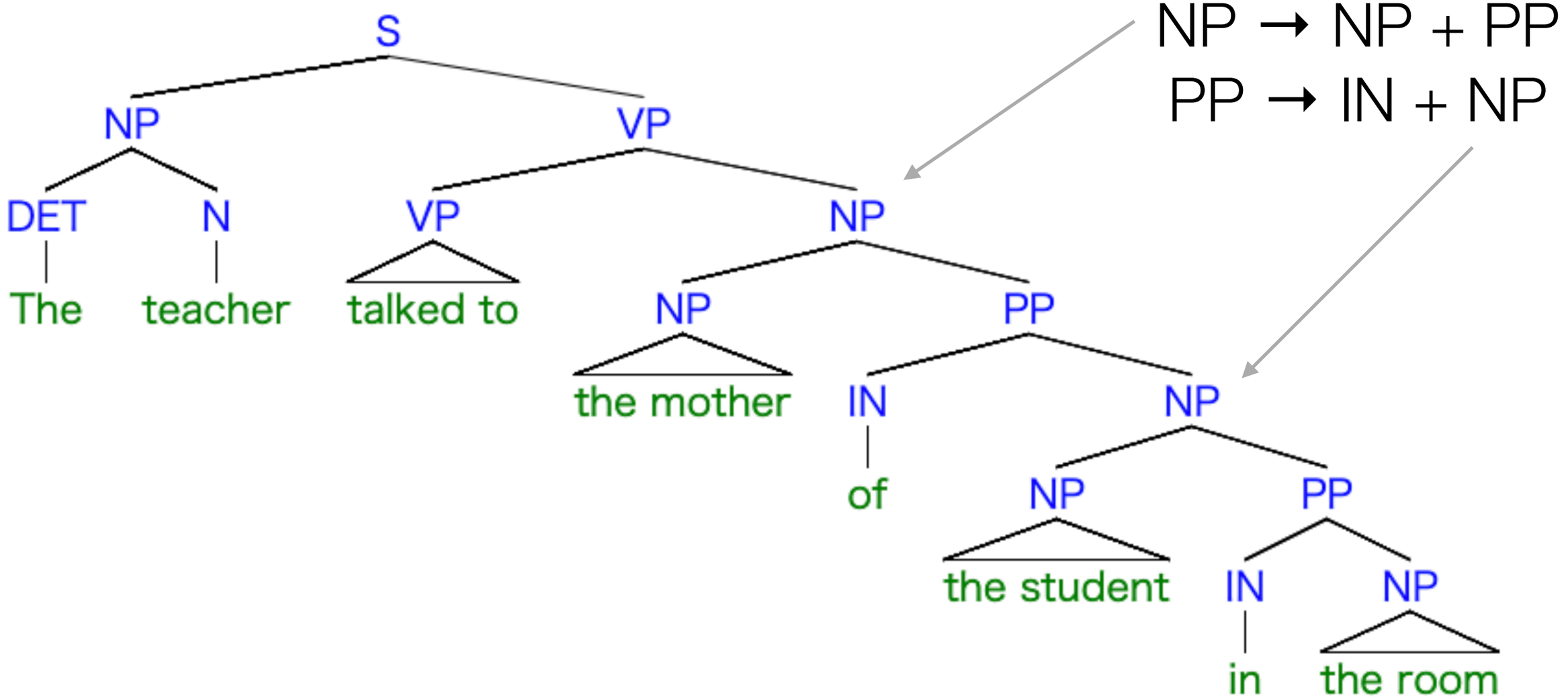
the mouse ate a cat on an egg **(valid)**
the cat ate a mouse ***and an egg*** **(invalid)**

Rewrite Rules and Backus Normal Form

- EBNF for equations
  - Equation → Expression = Expression
  - Expression → Term { (+|-) Term }
  - Term → Factor { * Factor }
  - Factor → Number | x | y | z | (Expression)
  - Number → Digit | Digit Number
  - Digit → 0 | 1 | ... | 9
- For parsing the equations like **(42 - 6 * 7) * x= 2 * 5 - 10**

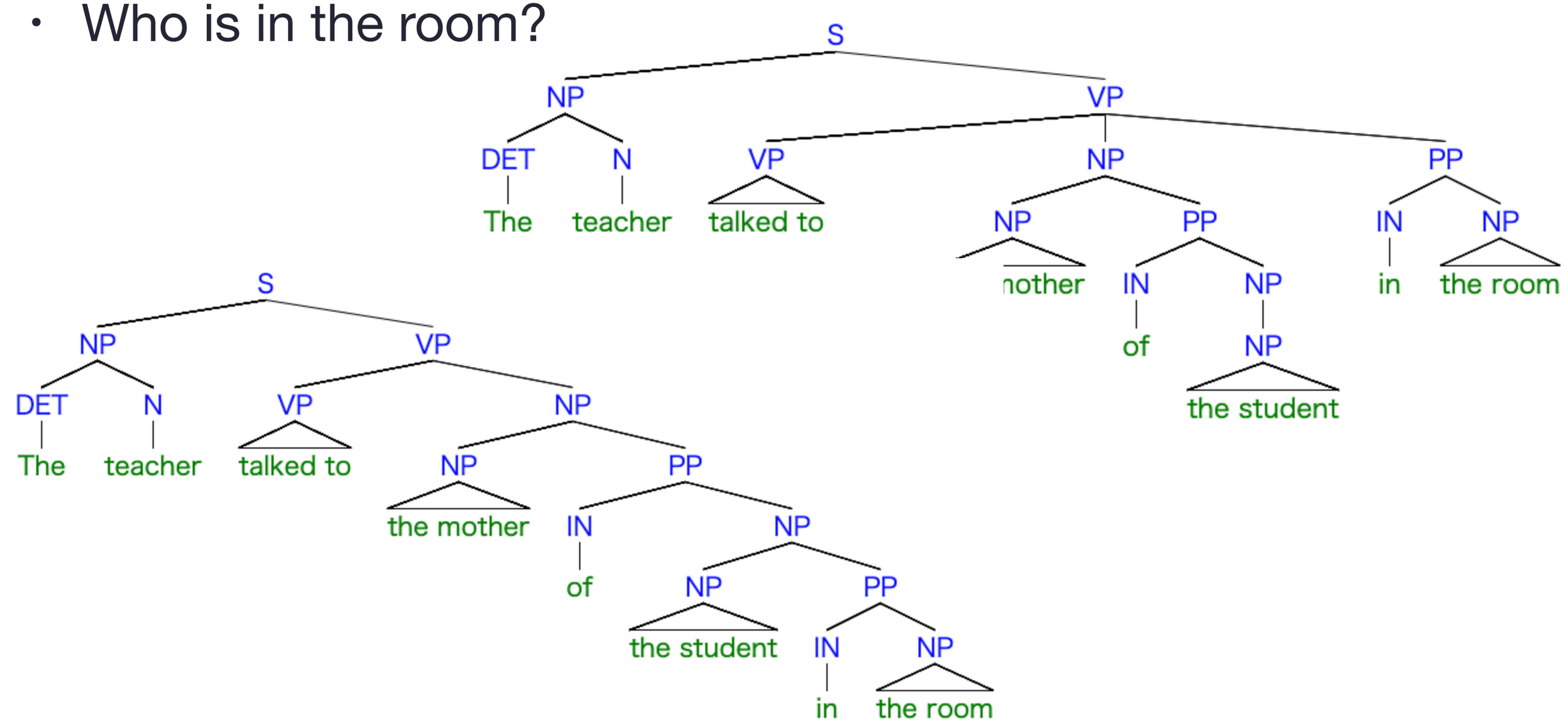
Recursive Constituent Structure

- Rewrite rules can be applied a number of times.
- VP and NP can be expanded to a large number of words.

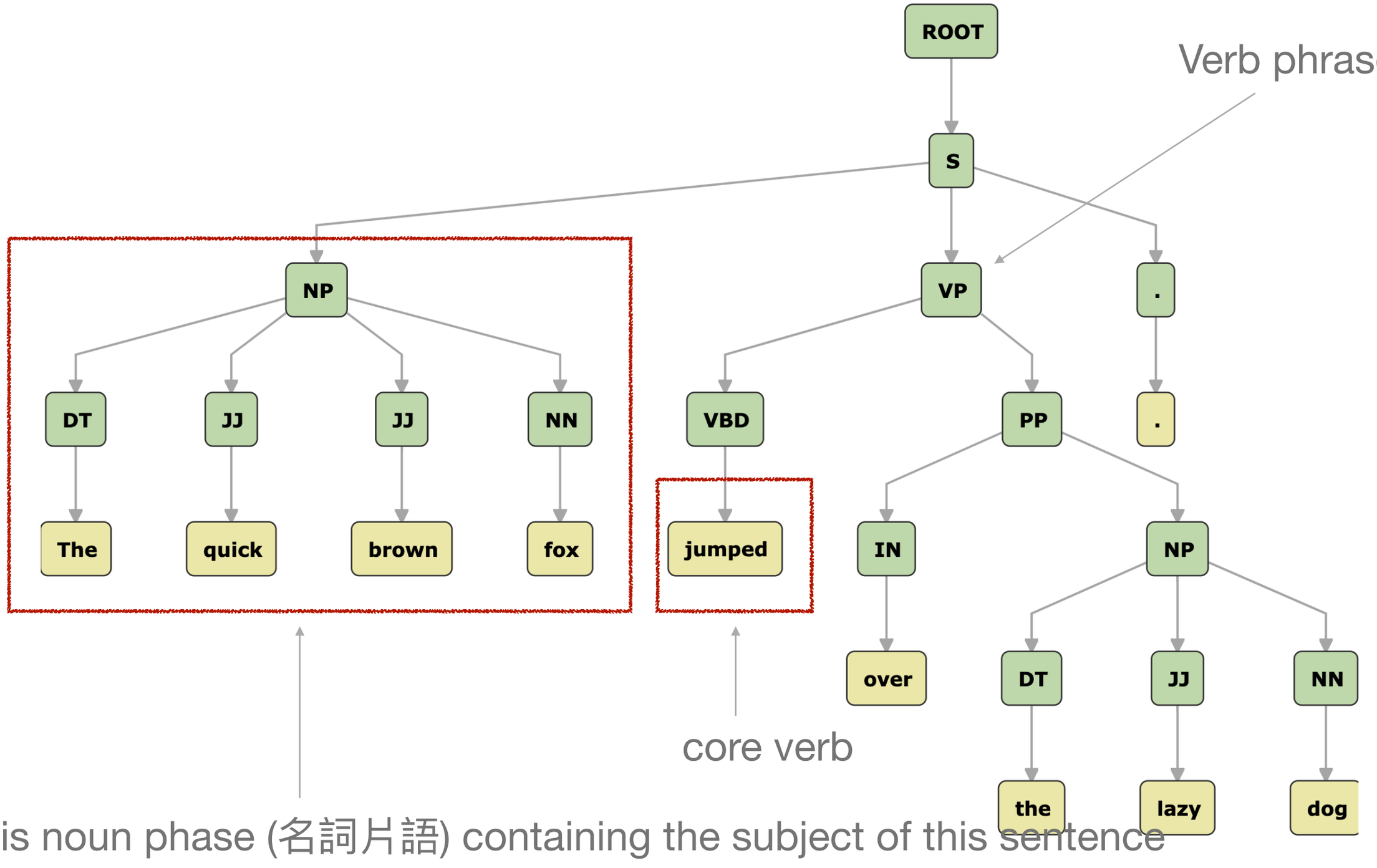


Syntactic Ambiguity

- Many different syntactic structure trees are feasible for a sentence.
- Who is in the room?

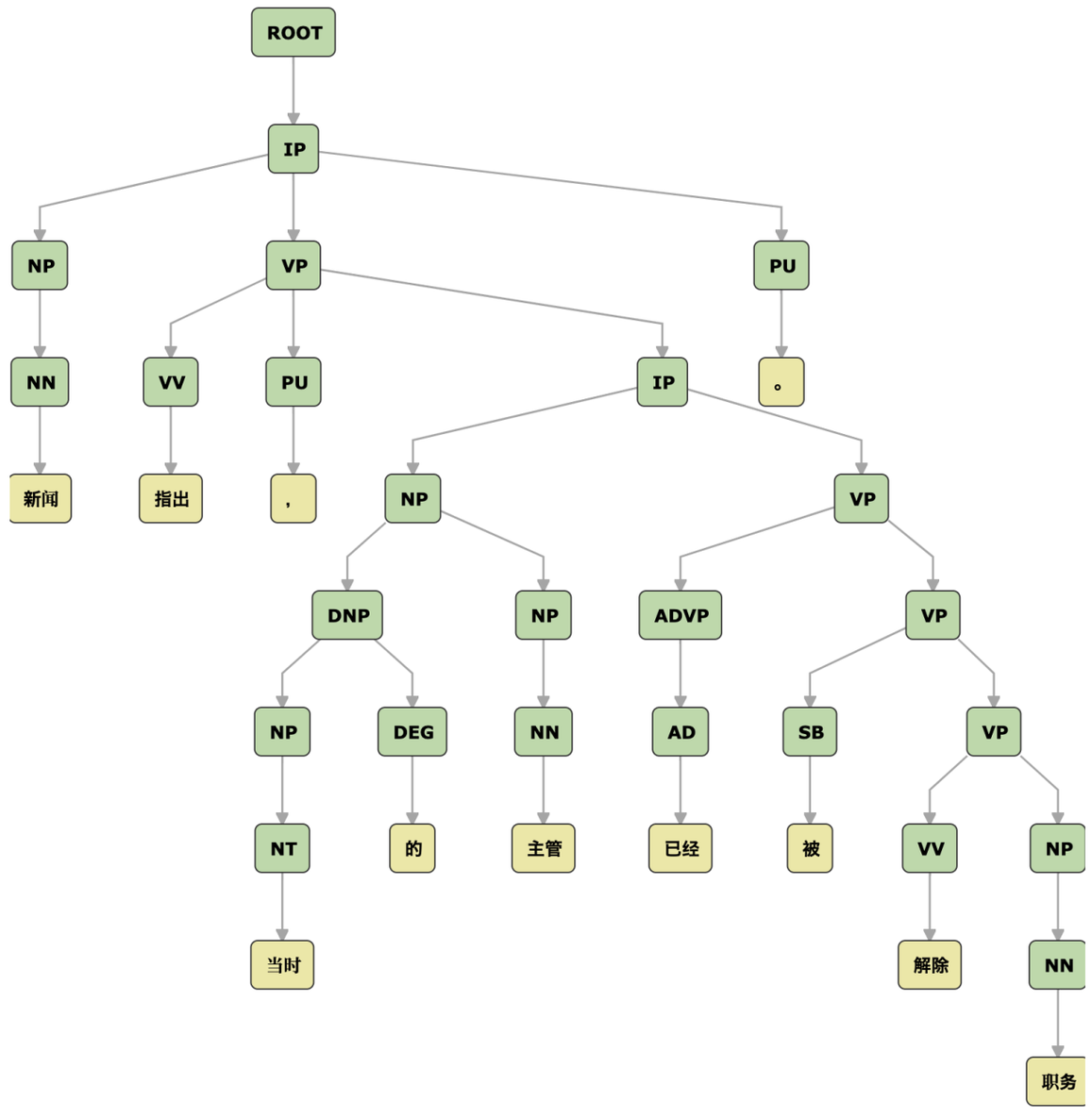


Syntactic Parsing



Chinese Syntactic Parsing

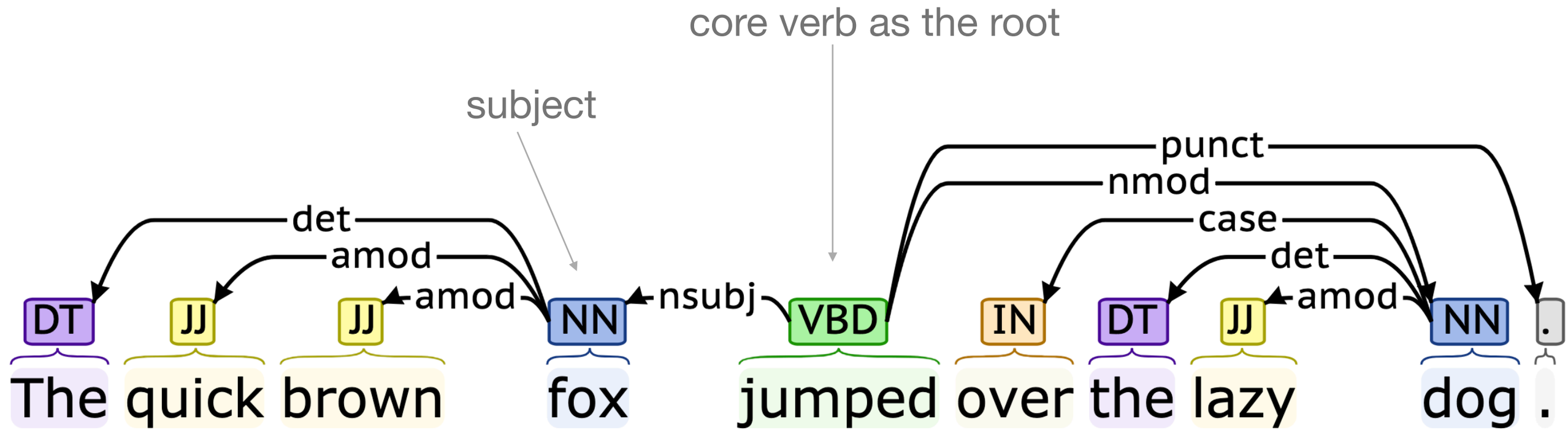
- The high level structure is less language dependent.



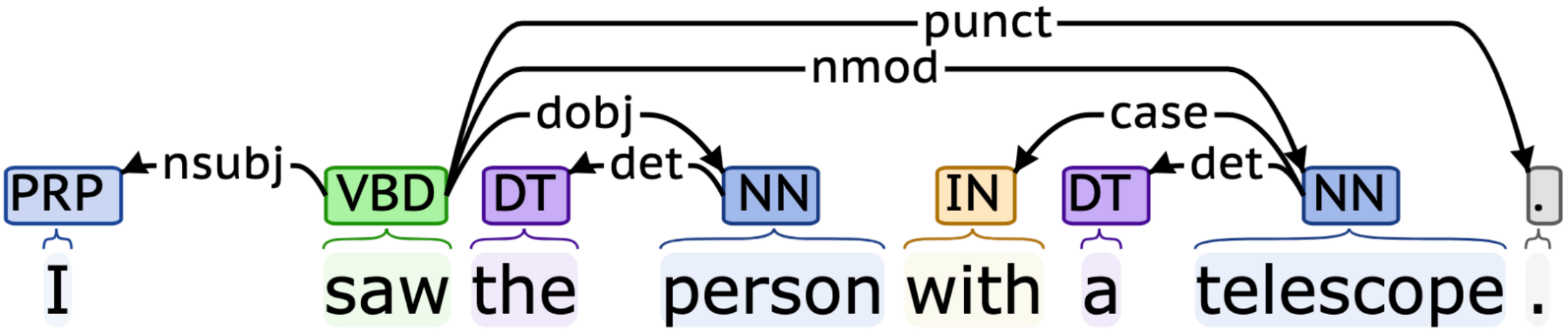
Dependency Parsing

- Instead of syntactic structure, dependency parsing provides information more focused on the relationship among words.

- Easier to process in pairs or triples.



Dependency Triples



(nsubj, saw, I)

(dobj, saw, person)

(nmod, saw, telescope)

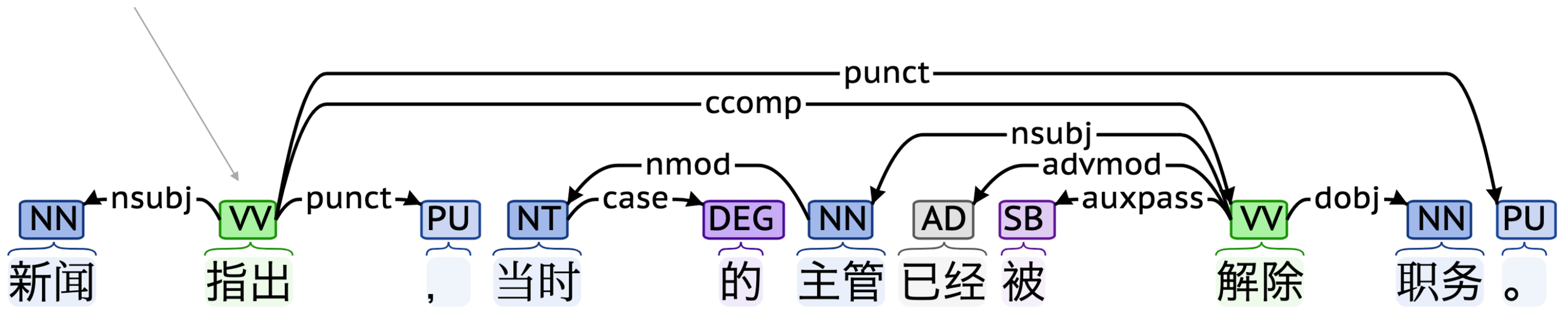
(punct, saw, .)

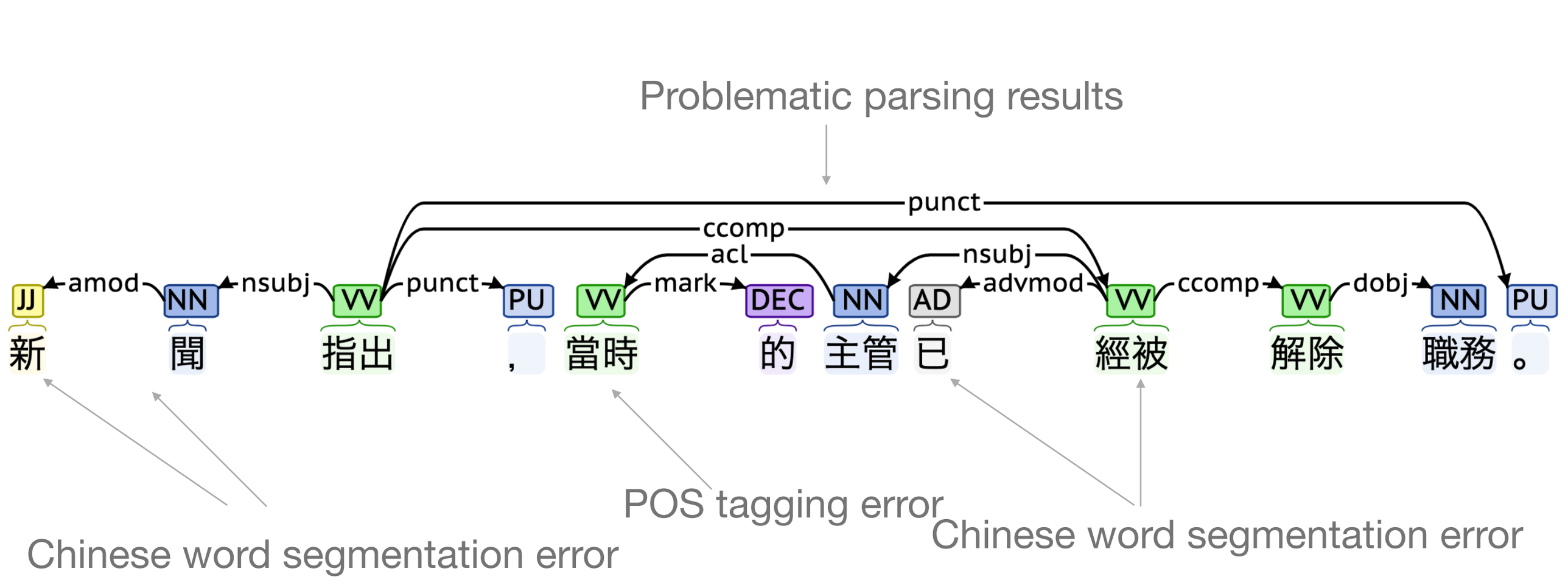
(det, person, the)

(det, telescope, a)

(case, telescope, with)

Chinese Dependency Parsing





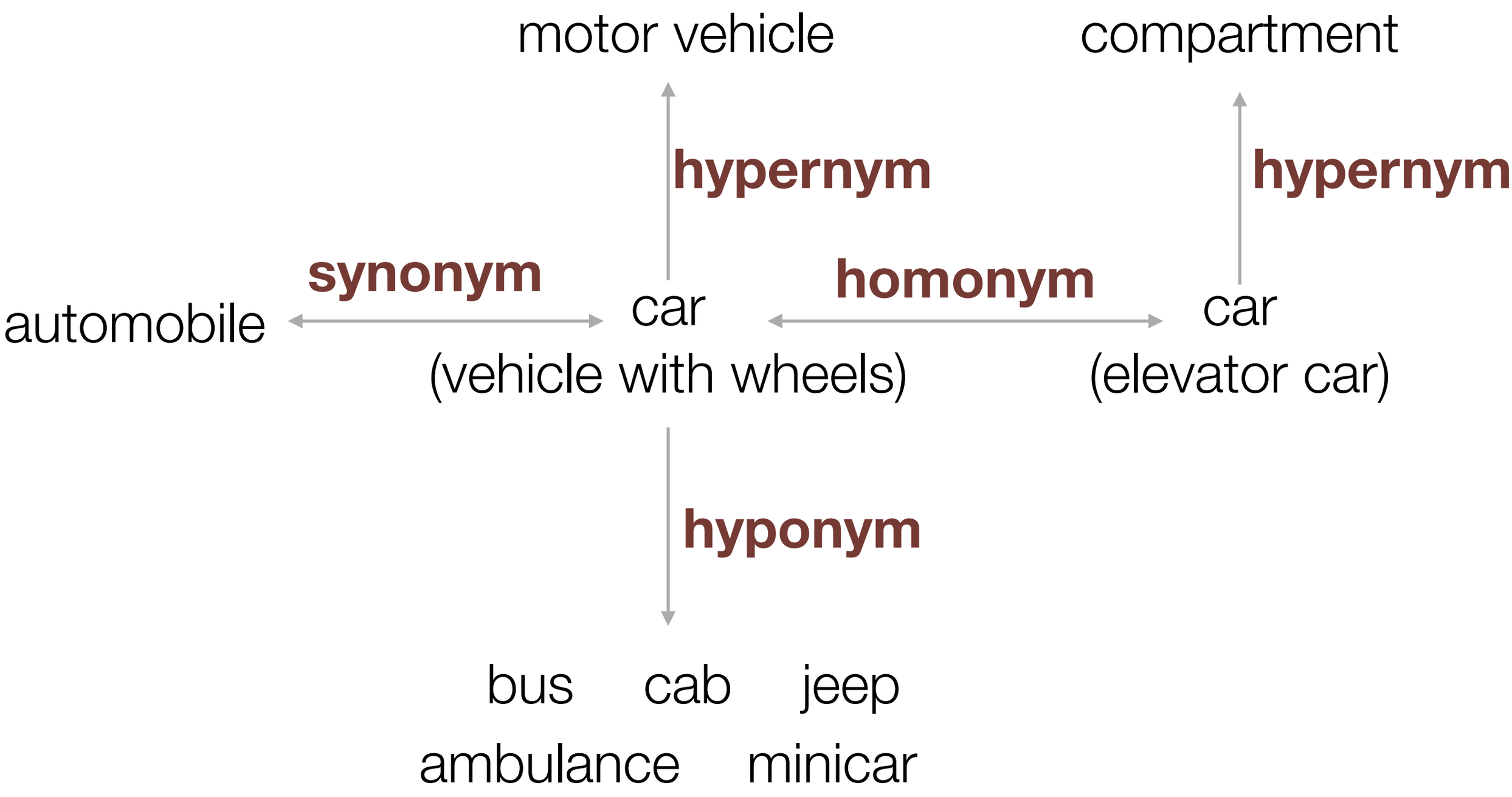
Semantics

- Lexical semantics
  - The study of the meaning of individual words.
- Sentence/discourse semantics
  - The study of how meanings of individual words are combined into the meaning of sentences or even larger units like discourse.

Relationship between Words

- Synonyms (同義詞)
    - car ↔ automobile
- Antonym (反義詞)
    - eval ↔ good
- Hypernym (上位語)
    - car → vehicle
- Hyponym (下位語)
    - car → bus
- Homonyms （同字異義）
    - bark (the sound of a dog) vs bark (the skin of a tree)

WordNet


motor vehicle
compartment
hypernym
hypernym
synonym
homonym
automobile
car
car
(vehicle with wheels)
(elevator car)
hyponym
bus
cab
jeep
ambulance
minicar

WordNet Browser

Word to search for: red Search WordNet

Display Options: (Select option to change) Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **red**, redness (red color or pigment; the chromatic color resembling the hue of blood)
  - *direct hyponym* / *full hyponym*
    - S: (n) sanguine (a blood-red color)
    - S: (n) chrome red (a red pigment used in paints; basic lead chromate)
    - S: (n) Turkey red, alizarine red (a bright orange-red color produced in cotton cloth with alizarine dye)
    - S: (n) cardinal, carmine (a variable color averaging a vivid red)
    - S: (n) crimson, ruby, deep red (a deep and vivid red color)
    - S: (n) dark red (a red color that reflects little light)
    - S: (n) purplish red, purplish-red (a red with a tinge of purple)
    - S: (n) cerise, cherry, cherry red (a red the color of ripe cherries)
    - S: (n) scarlet, vermilion, orange red (a variable color that is vivid red but sometimes with an orange tinge)
  - ***direct hypernym*** / *inherited hypernym* / *sister term*
    - S: (n) chromatic color, chromatic colour, spectral color, spectral colour (a color that has hue)
  - *derivationally related form*
- S: (n) **Red**, Red River (a tributary of the Mississippi River that flows eastward from Texas along the southern boundary of Oklahoma and through Louisiana)

http://wordnetweb.princeton.edu/perl/webwn?s=red&sub=Search+WordNet&o2=&o0=1&o8=1&o1=1&o7=&o5=&o9=&o6=&o3=&o4=&h=

Pragmatics

- The study of how the literal text interacts with the knowledge about the word and language conventions.

- Discourse analysis

  It's an old car, but it's reliable.

  old car implies unreliable by convention.

- Anaphor resolution

  *Tom helped **Jessica** get out of the cab. **She** thanked him.*

  Jessica is a woman's name.

Hurricane Hugo destroyed 20,000 Florida homes. At an estimated cost of one billion dollars, the disaster has been the most costly in the state's history.

1989 颶風

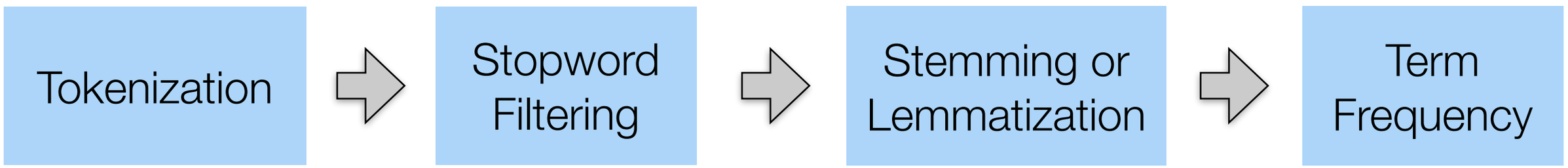
You have to know Hurricane Hugo is a disaster to understand this sentence.

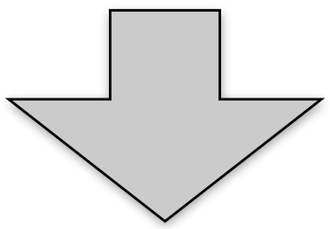
- NLP at pragmatics is usually difficult and yet to explore.

Part II

Basic Text Processing

Fundamental Text Processing





Tokenization

- Similar to Chinese word segmentation

Hindu refers to any person who regards themselves as culturally, ethnically, or religiously adhering to aspects of Hinduism.[1][2] It has historically been used as a geographical, cultural, and later religious

```
['Hindu', 'refers', 'to', 'any', 'person', 'who',
'regards', 'themselves', 'as', 'culturally', ',',
'ethnically', ',', 'or', 'religiously',
'adhering', 'to', 'aspects', 'of', 'Hinduism',
'.', '[', '1', ']', '[', '2', ']', 'It', 'has',
'historically', 'been', 'used', 'as', 'a',
'geographical', ',', 'cultural', ',', 'and',
'later', 'religious', 'identifier', 'for',
'people', 'indigenous', 'to', 'the', 'Indian',
'subcontinent', '.', '[', '3', ']', '[', '4',
']', 'The', 'historical', 'meaning', 'of', 'the',
'term', 'Hindu', 'has', 'evolved', 'with',
'time', '.', 'Starting', 'with', 'the',
'Persian', 'and', 'Greek', 'references', 'to',
'the', 'land', 'of', 'the', 'Indus', 'in', 'the',
'1st', 'millennium', 'BCE', 'through', 'the',
'texts', 'of', 'the', 'medieval', 'era', ',',
'[', '5', ']', 'the', 'term', 'Hindu', 'implied',
'a', 'geographic', ',', 'ethnic', 'or',
'cultural', 'identifier', 'for', 'people',
'living', 'in', 'the', 'Indian', 'subcontinent',
'around', 'or', 'beyond', 'the', 'Sindhu', '(',
'Indus', ')', 'river', '.', '[', '6', ']', 'By',
'the', '16th', 'century', ',', 'the', 'term',
'began', 'to', 'refer', 'to', 'residents', 'of',
'the', 'subcontinent', 'who', 'were', 'not',
'Turkic', 'or', 'Muslims', ...]
```

Tokenization with the split() function

- split(): Return a list of the words in the string, using sep as the delimiter string.

```
text = "Hindu refers to any person who regards
themselves as culturally, ethnically, or religiously
adhering to aspects of Hinduism.[1][2]"

tokens = text.split(" ")

['Hindu', 'refers', 'to', 'any', 'person', 'who', 'regards',
'themselves', 'as', 'culturally,', 'ethnically,', 'or',
'religiously', 'adhering', 'to', 'aspects', 'of', 'Hinduism.
[1][2]']
```

Tokenization with NLTK

- A better solution
  - Free NLTK book: https://www.nltk.org/book/
- pip install nltk

```
from nltk.tokenize import word_tokenize
tokens = word_tokenize(" ")
```

```
['Hindu', 'refers', 'to', 'any', 'person', 'who', 'regards',
'themselves', 'as', 'culturally', ',', 'ethnically', ',', 'or',
'religiously', 'adhering', 'to', 'aspects', 'of', 'Hinduism', '.',
'[', '1', ']', '[', '2', ']']
```

Punctuation Marks

```
import string
print(string.punctuation)

!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~

# ASCI only
```

Removal of Punctuation Marks

```
def remove_punctuation_marks(tokens):
    clean_tokens = []
    for tok in tokens:
        if tok not in string.punctuation:
            clean_tokens.append(tok)
    return clean_tokens
```

```
['Hindu', 'refers', 'to', 'any', 'person', 'who', 'regards',
'themselves', 'as', 'culturally', 'ethnically', 'or',
'religiously', 'adhering', 'to', 'aspects', 'of', 'Hinduism',
'1', '2', 'It', 'has', 'historically', 'been', 'used', 'as',
'a', 'geographical', 'cultural', 'and', 'later', 'religious',
'identifier', 'for', 'people', 'indigenous', 'to', 'the',
'Indian', 'subcontinent', '3', '4', ...]
```

Removing All Non-alphabet Tokens

```
def remove_punctuation_marks(tokens):
    clean_tokens = []
    for tok in tokens:
        if tok.isalpha():
            clean_tokens.append(tok)
    return clean_tokens
```

```
['Hindu', 'refers', 'to', 'any', 'person', 'who', 'regards', 'themselves',
'as', 'culturally', 'ethnically', 'or', 'religiously', 'adhering', 'to',
'aspects', 'of', 'Hinduism', 'It', 'has', 'historically', 'been', 'used', 'as',
'a', 'geographical', 'cultural', 'and', 'later', 'religious', 'identifier',
'for', 'people', 'indigenous', 'to', 'the', 'Indian', 'subcontinent', 'The',
'historical', 'meaning', 'of', 'the', 'term', 'Hindu', 'has', 'evolved',
'with', 'time', 'Starting', 'with', 'the', 'Persian', 'and', 'Greek',
'references', 'to', 'the', 'land', 'of', 'the', 'Indus', 'in', 'the', '1st',
'millennium', 'BCE', 'through', 'the', 'texts', 'of', 'the', 'medieval', 'era',
'the', 'term', 'Hindu', 'implied', 'a', 'geographic', 'ethnic', ...]
```

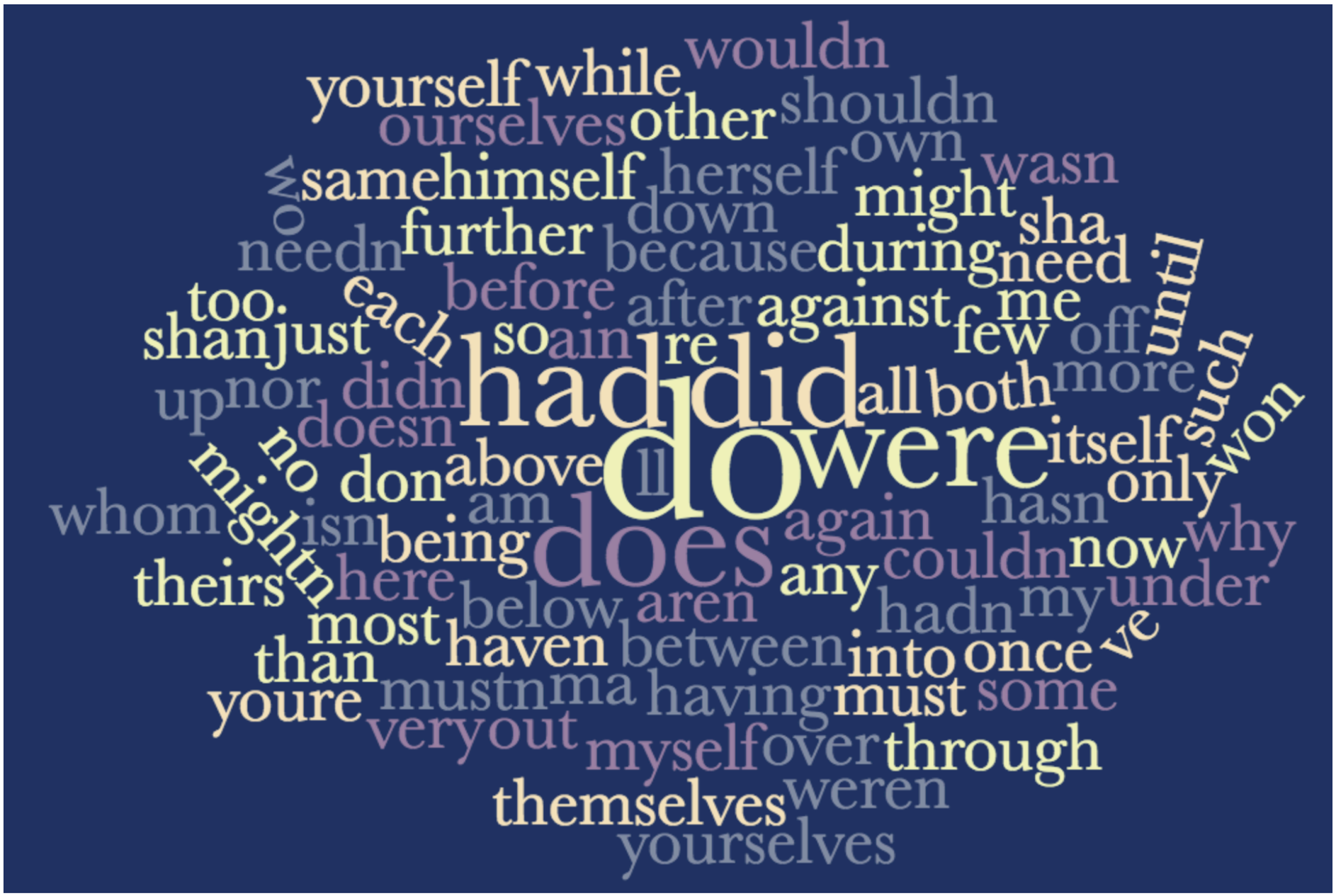
Using Python Generator

```
def remove_punctuation_marks(tokens):
        return [tok for tok in tokens if tok.isalpha()]
```

- ['Hindu', 'refers', 'to', 'any', 'person', 'who', 'regards', 'themselves', 'as', 'culturally', 'ethnically', 'or', 'religiously', 'adhering', 'to', 'aspects', 'of', 'Hinduism', 'It', 'has', 'historically', 'been', 'used', 'as', 'a', 'geographical', 'cultural', 'and', 'later', 'religious', 'identifier', 'for', 'people', 'indigenous', 'to', 'the', 'Indian', 'subcontinent', 'The', 'historical', 'meaning', 'of', 'the', 'term', 'Hindu', 'has', 'evolved', 'with', 'time', 'Starting', 'with', 'the', 'Persian', 'and', 'Greek', 'references', 'to', 'the', 'land', 'of', 'the', 'Indus', 'in', 'the', '1st', 'millennium', 'BCE', 'through', 'the', 'texts', 'of', 'the', 'medieval', 'era', 'the', 'term', 'Hindu', 'implied', 'a', 'geographic', 'ethnic', ...]

Stopwords

- Words without contributions to our task.



Content words vs Function words

Function words ? Content words

Prepositions
of, at, in, without, between

Pronouns
he, they, anybody, it, one

Determiners
the, a, that, my, more, much, either, neither

Auxiliary
will, have, would, can

Particles
as

Light Verbs
Do, make, have, get

Conjunctions
if, because, but, however, and...

Negatives
no, not, neither, nor...

Nouns
land, sea, bank, coach

Proper Nouns
Taiwan, Pacific, English

Verbs
write, listen, hold, run

Adjectives
better, black, sad, fast

Adverbs
smoothly, significantly, fast

Stopword List

```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
stopword_list = stopwords.words('english')
print(stopword_list)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've",
"you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself',
'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them',
'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll",
'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has',
'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or',
'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against',
'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from',
'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once',
'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more',
'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than',
'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now',
'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn',
"didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn',
"isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't",
'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn',
"wouldn't"]
```

Removal of Stopwords

```
def remove_stopwords(tokens):
    tokens_clean = []
    for tok in tokens:
        if tok not in stopword_list:
            tokens_clean.append(tok)
    return tokens_clean
print(remove_stopwords(tokens))
```

```
['Hindu', 'refers', 'person', 'regards', 'culturally', 'ethnically', 'religiously',
'adhering', 'aspects', 'Hinduism', 'It', 'historically', 'used', 'geographical',
'cultural', 'later', 'religious', 'identifier', 'people', 'indigenous', 'Indian',
'subcontinent', 'The', 'historical', 'meaning', 'term', 'Hindu', 'evolved', 'time',
'Starting', 'Persian', 'Greek', 'references', 'land', 'Indus', '1st', 'millennium',
'BCE', 'texts', 'medieval', 'era', 'term', 'Hindu', 'implied', 'geographic', 'ethnic',
'cultural', 'identifier', 'people', 'living', 'Indian', 'subcontinent', 'around',
'beyond', 'Sindhu', 'Indus', 'river', 'By', '16th', 'century', 'term', 'began', 'refer',
'residents', 'subcontinent', 'Turkic', 'Muslims', 'b', 'The', 'historical',
'development', 'Hindu', 'self-identity', 'within', 'local', 'South', 'Asian',
'population', 'religious', 'cultural', 'sense', 'unclear', 'Competing', 'theories',
'state', 'Hindu', 'identity', 'developed', 'British', 'colonial', 'era', 'developed',
'post-8th', 'century', 'CE', 'Islamic', 'invasion', 'medieval', 'Hindu-Muslim', 'wars',
```

Capitalization in English

- In NLP and text mining, it is usually to convert all letter to lowercase.
- However, some information would be lost.
- In Python, it is very easy to case conversion.
  - str.lower(): Return a copy of the string with all the cased characters converted to lowercase.
  - str.upper(): Return a copy of the string with all the cased characters converted to uppercase.

Converting All Characters to Lowercase

```
def lowercase(tokens):
    tokens_lower = []
    for tok in tokens:
        tokens_lower.append(tok.lower())
    return tokens_lower

print(remove_stopwords(lowercase(tokens)))
```

```
['hindu', 'refers', 'person', 'regards', 'culturally', 'ethnically', 'religiously',
'adhering', 'aspects', 'hinduism', 'historically', 'used', 'geographical', 'cultural',
'later', 'religious', 'identifier', 'people', 'indigenous', 'indian', 'subcontinent',
'historical', 'meaning', 'term', 'hindu', 'evolved', 'time', 'starting', 'persian',
'greek', 'references', 'land', 'indus', '1st', 'millennium', 'bce', 'texts',
'medieval', 'era', 'term', 'hindu', 'implied', 'geographic', 'ethnic', 'cultural',
'identifier', 'people', 'living', 'indian', 'subcontinent', 'around', 'beyond',
'sindhu', 'indus', 'river', '16th', 'century', 'term', 'began', 'refer', 'residents',
'subcontinent', 'turkic', 'muslims', 'b', 'historical', 'development', 'hindu', 'self-
identity', 'within', 'local', 'south', 'asian', 'population', 'religious', 'cultural',
'sense', 'unclear', 'competing', 'theories', 'state', 'hindu', 'identity',
'developed', 'british', 'colonial', 'era', 'developed', 'post-8th', 'century', 'ce',
'islamic', 'invasion', 'medieval', 'hindu-muslim', 'wars', ...]
```

Removal of Stopwords with Capitalization Handling

```
def remove_stopwords(tokens):
    tokens_clean = []
    for tok in tokens:
        if tok.lower() not in stopword_list:
            tokens_clean.append(tok)
    return tokens_clean
print(remove_stopwords(tokens))
```

```
['Hindu', 'refers', 'person', 'regards', 'culturally', 'ethnically', 'religiously',
'adhering', 'aspects', 'Hinduism', 'historically', 'used', 'geographical', 'cultural',
'later', 'religious', 'identifier', 'people', 'indigenous', 'Indian', 'subcontinent',
'historical', 'meaning', 'term', 'Hindu', 'evolved', 'time', 'Starting', 'Persian',
'Greek', 'references', 'land', 'Indus', '1st', 'millennium', 'BCE', 'texts',
'medieval', 'era', 'term', 'Hindu', 'implied', 'geographic', 'ethnic', 'cultural',
'identifier', 'people', 'living', 'Indian', 'subcontinent', 'around', 'beyond',
'Sindhu', 'Indus', 'river', '16th', 'century', 'term', 'began', 'refer', 'residents',
'subcontinent', 'Turkic', 'Muslims', 'b', 'historical', 'development', 'Hindu', 'self-
identity', 'within', 'local', 'South', 'Asian', 'population', 'religious', 'cultural',
'sense', 'unclear', 'Competing', 'theories', 'state', 'Hindu', 'identity',
'developed', 'British', 'colonial', 'era', 'developed', 'post-8th', 'century', 'CE',
'Islamic', 'invasion', 'medieval', 'Hindu-Muslim', 'wars', ...]
```

Stemming（詞幹提取）

- Not to distinguish the morphological affixes from words
- Lookup table
  - There is a dictionary for looking up the stem for a given word.
  - Out-of-vocabulary issue.
  - Precise.
- Rule-base
  - If the word ends in 'ed', remove the 'ed'
  - If the word ends in 'ing', remove the 'ing'
  - If the word ends in 'ly', remove the 'ly'
  - Prone to exceptional cases

Stemming with NLTK

- from nltk.stem.snowball import SnowballStemmer
- snowball_stemmer = SnowballStemmer("english")
- print(snowball_stemmer.stem('opened'))
  - open

Stemming Results

Input	Stemmed
open	open
opens	open
opened	open
opening	open
unopened	unopen
talk	talk
talks	talk
talked	talk
talking	talk
decompose	decompos
decomposes	decompos
decomposed	decompos
decomposing	decompos

Input	Stemmed
do	do
does	doe
did	did
wrote	wrote
written	written
ran	ran
gave	gave
held	held
went	went
gone	gone
lied	lie
lies	lie
lay	lay
lain	lain
lying	lie

More Stemming Results

Input	Stemmed
cats	cat
people	peopl
feet	feet
smoothly	smooth
firstly	first
secondly	second
install	instal
installed	instal
uninstall	uninstal

Input	Stemmed
internalization	intern
internationalization	internation
decontextualization	decontextu
decontextualized	decontextu
decentralization	decentr
decentralized	decentr

Lemmatization（字形還原）

- Grouping together the inflected forms of a word as a single lemma.
- A process more complex than stemming.
- Most based on dictionary.
- The outcome is more readable.

Lemmatization with NLTK

```
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

print(lemmatizer.lemmatize('opened', pos = 'v'))
```

```
open
```

Lemmatization with NLTK Regardless of POS

- A lazy function that tries all possible parts-of-speech for lemmatization.

```
def lemmatize(token):
    # ADJ (a), ADJ_SAT (s), ADV (r), NOUN (n) or VERB (v)
    for p in ['v', 'n', 'a', 'r', 's']:
        l = wordnet_lemmatizer.lemmatize(token, pos=p)
        if l != token:
            return l
    return token
```

Stemming vs Lemmatization

Input	Stemmed	Lemmatized
unopened	unopen	unopened
decompose	decompos	decompose
decomposes	decompos	decompose
decomposed	decompos	decompose
decomposing	decompos	decompose
does	doe	do
did	did	do
wrote	wrote	write
written	written	write
ran	ran	run
gave	gave	give
held	held	hold
went	went	go
gone	gone	go
lain	lain	lie

Stemming vs Lemmatization

Input	Stemmed	Lemmatized
people	peopl	people
feet	feet	foot
women	women	woman
smoothly	smooth	smoothly
firstly	first	firstly
secondly	second	secondly
install	instal	install
uninstall	uninstal	uninstall
internalization	intern	internalization
internationalization	internation	internationalization
decontextualization	decontextu	decontextualization
decontextualized	decontextu	decontextualized
decentralization	decentr	decentralization
decentralized	decentr	decentralize

Stemming/Lemmatization is Sensitive to Final Task

- Important information can be lost during stemming/ lemmatization.
    - Tense (時態)
    - Plural and singular (單複數)
- To do or not to do
    - It depends on your final task.

Register your attendance

Undergraduate

Graduate