NLP Pipeline Building Blocks

- Transcribing
- Language identification
- Segmentation
- Normalization
  - *weird symbols, non-UTF symbols, curly quotation marks*
  - *truecasing*
  - *word wrap*
  - *spelling errors*
  - *slang*
  - *lemmatization, stemming, removing stopwords*

NLP Pipeline Building Blocks

- Transcribing
- Language identification
- Segmentation
- Normalization
- Text classification or topic modelling
- POS tagging
- Named-entity recognition
- Syntactic parsing
- Relation extraction
- Coreference resolution
- Semantic parsing…

Structural Linguistics 1: the *form*

Mariana Romanyshyn,
Computational Linguist at Grammarly

Contents

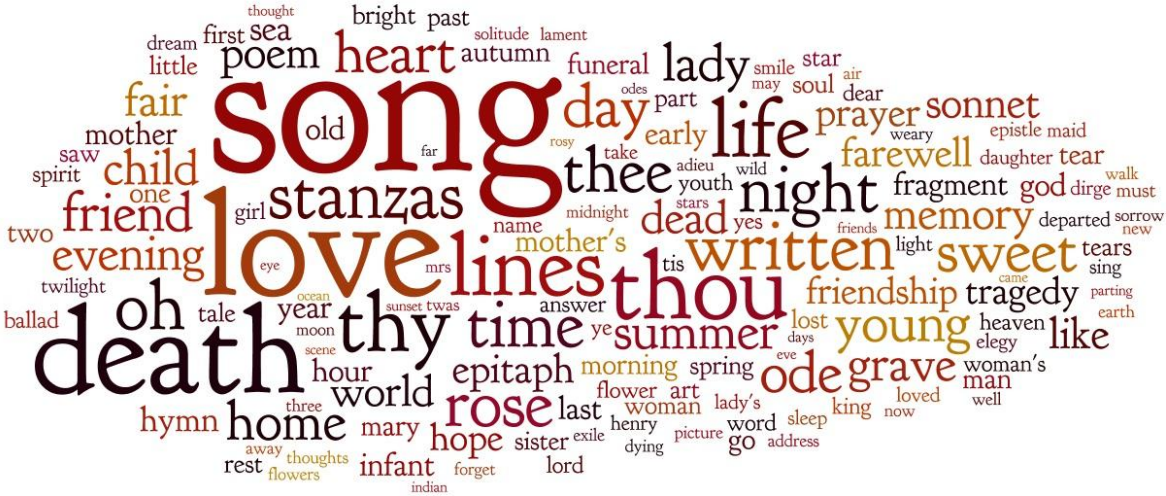
A word is its...

1. form
2. function
3. meaning

Contents

A word is its...

1. ***form***
2. function
3. meaning



1. Orthography

Orthography

Investigate the way the word is written:

- capitalization
- hyphen
- apostrophe

Capitalization

Capitalization

- Style
  - sentence start
  - book titles

- Proper names
  - *Cities like* ***The Hague****…* (en)
  - *Учора під* ***Верховною Радою Саакашвілі****…* (uk)
  - *Mamy w* ***Polsce*** *wybudowany dom…* (pl)

Capitalization

- Common nouns and adjectives
  - *On **Monday**, an **English Democrat**…* (en)
  - *Eine **Katze** kommt ins **Haus**.* (de)

- Nouns in the legal language
  - *… а **Замовник** має право...* (uk)
  - *The **Landlord** agrees to let the **Tenant** take…* (en)

Capitalization

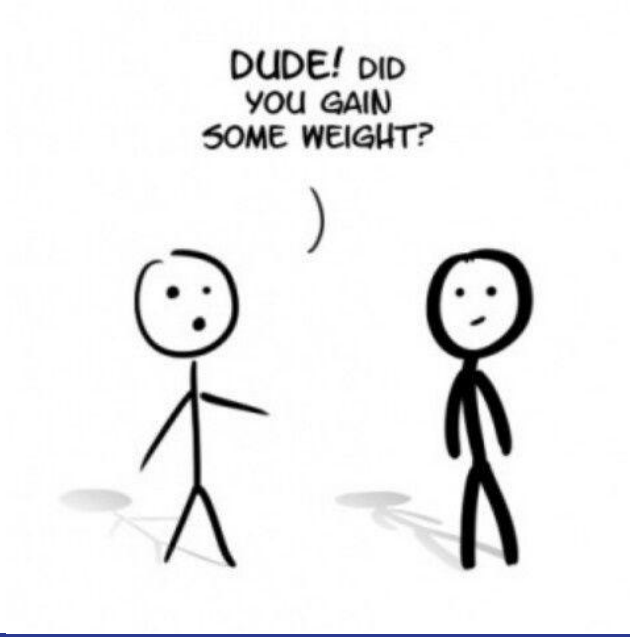
- Pronouns
  - ***Sie*** (de), ***Вы*** (ru), ***Шумо*** (tg), ***Vi*** (sl), *risponder**Le*** (it)
  - *follow **His** word* (en), *вчення **Його** сина* (uk)

- Initialisms and acronyms
  - ***EU*** (en), ***ЄС*** (uk), ***UE*** (pl)
  - ***LotR*** (en), ***GmbH*** (de)

Capitalization

Capitonyms - words that change meaning due to capitalization.

- ***Laut*** vs. ***laut*** (de)
- ***Morgen*** vs. ***morgen*** (de)
- ***March*** vs. ***march*** (en)
- ***Polish*** vs. ***polish*** (en)
- ***China*** vs. ***china*** (en)
- ***Роман*** vs. ***роман*** (uk)
- ***Рада*** vs. ***рада*** (uk)
- ***Peru*** vs. ***peru*** (pt)



Hyphenation

Hyphenation

- Compounds
  - ***Editor-in-Chief*** (en), ***Ростов-на-Дону*** (ru)
  - ***passer-by*** (en), ***механіко-математичний*** (uk)
  - *a* ***twenty-five-year-old*** *woman* (en)
  - *a* ***please-don't-ask-me*** *attitude* (en)

- Prefixation/Suffixation
  - ***co-worker*** (en), ***екс-чемпіон*** (uk), ***celui-ci*** (fr)

Hyphenation

- Suspense
  - ***pre- and post-operative*** (en)
  - ***радіо- і телепрограми*** (uk)

- Onomatopoeia
  - ***heh-heh*** (en), ***ціп-ціп-ціп*** (uk), ***fiu-fiu*** (pl)

Hyphenation

Hyphenation eliminates ambiguity:

- ***unionized*** vs. ***un-ionized***

Hyphenation

Hyphenation eliminates ambiguity:

- ***unionized*** vs. ***un-ionized***
- ***three-hundred-year-old*** *trees*

  vs.

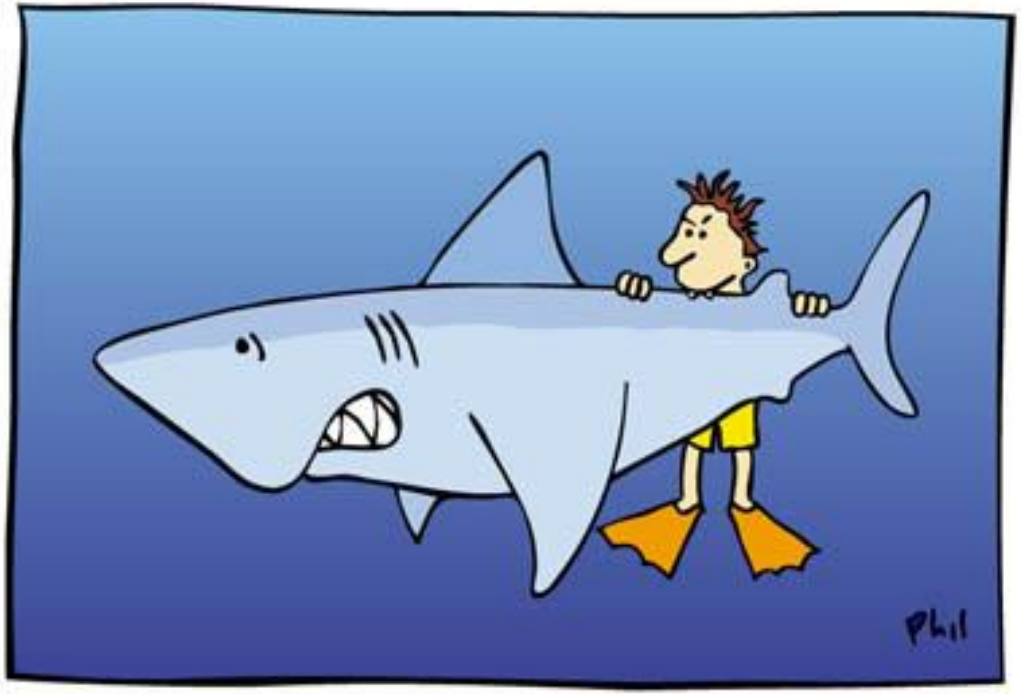
  three ***hundred-year-old*** *trees*

  vs.

  three hundred ***year-old*** *trees*

Hyphenation

*a **man-eating** shark* vs. *a **man eating** shark*



Apostrophes

Apostrophes

- Contractions
  - ***we'll, I'd've, 'cause, talkin'*** (en)
  - *Wie* ***geht's****?* (de), ***c'est*** (fr), ***l'opera*** (it), ***дит'ясла*** (uk)
- Possession
  - ***lady's, ladies'*** (en)
- Names
  - ***O'Doole, M'Gregor***
- Pronunciation
  - ***бур'ян*** (uk), ***ka'a*** (gn)

Usage in NLP

Capitalization:

- Named entity recognition & resolution
- Part-of-speech tagging
- Truecasing
- Title identification/formatting
- Detection of politeness level
- Word sense disambiguation
- Error correction

Usage in NLP

Hyphenation:

- Tokenization
- Part-of-speech tagging
- Word sense disambiguation

Apostrophes:

- Tokenization
- Formality detection and transfer

2. Tokenization

Tokenization

A token is:

- an independent word
- a number
- a punctuation mark

Tokenization

How many tokens are there in...

- *can’t, we’ll, Homer’s, l'opéra, geht’s*

Tokenization

How many tokens are there in...

- *can’t, we’ll, Homer’s, l'opéra, geht’s*
- *ladies’, ‘cause, ‘cause’*

Tokenization

How many tokens are there in...

- *can’t, we’ll, Homer’s, l'opéra, geht’s*
- *ladies’, ‘cause, ‘cause’*
- *twenty-five-year-old, жар-птиця, йди-но, такий-от*

Tokenization

How many tokens are there in...

- *can’t, we’ll, Homer’s, l'opéra, geht’s*
- *ladies’, ‘cause, ‘cause’*
- *twenty-five-year-old, жар-птиця, йди-но, такий-от*
- *gonna, lemme, dunno, sorta*

Tokenization

How many tokens are there in...

- *can't, we'll, Homer's, l'opéra, geht's*
- *ladies', 'cause, 'cause'*
- *twenty-five-year-old, жар-птиця, йди-но, такий-от*
- *gonna, lemme, dunno, sorta*
- *$3b, 1.35, 1/2/2018*
- *etc., U.S.A.*
- *:), www.example.com*

Tokenization

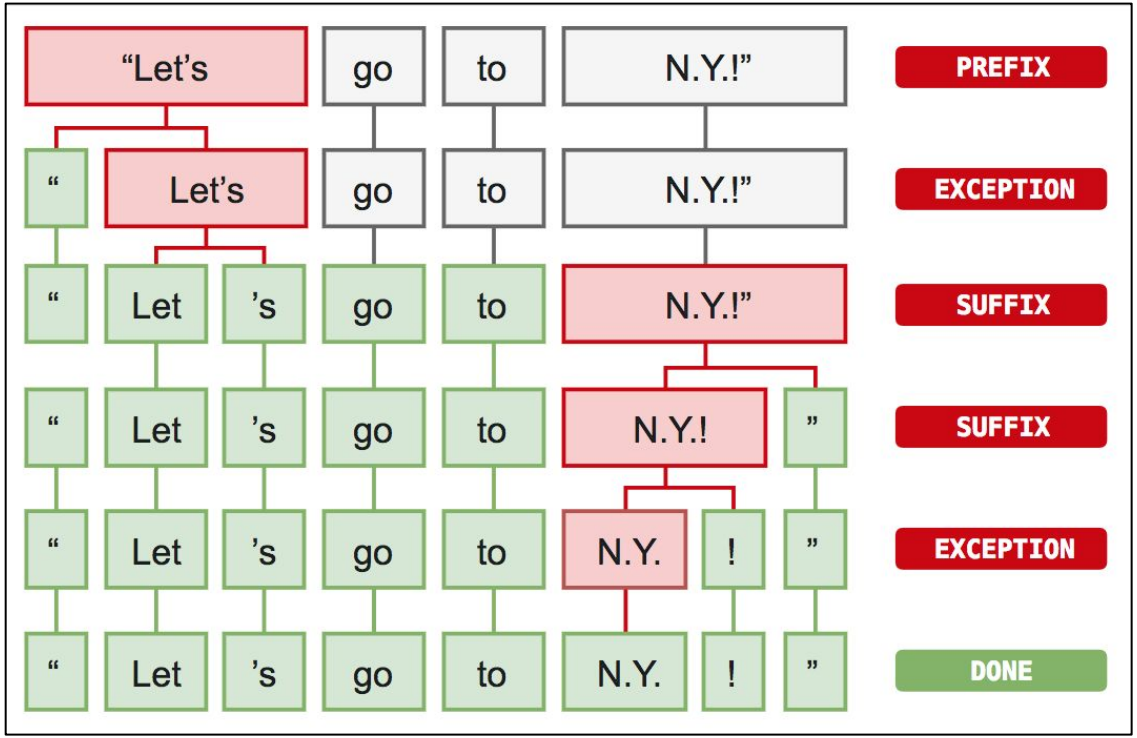
How many tokens are there in...

- *can't, we'll, Homer's, l'opéra, geht's*
- *ladies', 'cause, 'cause'*
- *twenty-five-year-old, жар-птиця, йди-но, такий-от*
- *gonna, lemme, dunno, sorta*
- *$3b, 1.35, 1/2/2018*
- *etc., U.S.A.*
- *:), www.example.com*
- *San Francisco, out of*

Tokenization Howto

- Spacy (7 languages; Python)
- Stanford CoreNLP (6 languages; wrappers for 10+ programming langs)
- StanfordNLP (53 languages; Python)
- OpenNLP (7 languages; Java)
- Emory NLP (English; Java)
- lang-uk/tokenize-uk (Ukrainian; Python)
- nlp_uk (Ukrainian; Groovy)
- cl-nlp (Common Lisp)
- nltk (English; Python) or TextBlob (English; Python)
- Write your own rules and regexps :)

Tokenization with spaCy



Taken from https://spacy.io/assets/img/tokenization.svg

Tokenization with spaCy

```
import en_core_web_md
```

```
nlp = en_core_web_md.load()
```

```
quote = nlp("Roads? Where we're going we don't need roads.")
print([token.text for token in quote])
```

```
['Roads', '?', 'Where', 'we', "'re", 'going', 'we', 'do', "n't", 'need', 'roads', '.']
```

```
print([token.shape_ for token in quote])
```

```
['Xxxxx', '?', 'Xxxxx', 'xx', "'xx", 'xxxx', 'xx', 'xx', "x'x", 'xxxx', 'xxxx', '.']
```

Tokenization with lang_uk

```
import tokenize_uk
```

```
text = "Обговорімо основні етапи купівлі та розмитнення \"євробляхи\"."
```

```
print(tokenize_uk.tokenize_words(text))
```

```
['Обговорімо', 'основні', 'етапи', 'купівлі', 'та', 'розмитнення', '"', 'євробляхи', '"', '.']
```

3. Morphology

Morphemes

A word consists of morphemes:

- stem+
- affix*

Morphemes

A word consists of morphemes:

- stem+
  - ***board***, ***blackboard*** (en)
  - ***рук****а*, ***рук****опис* (uk)
  - ***Lehr****er*, ***Lehrbuch*** (de)
- affix*
  - ***a****board, blackboard****s*** (en)
  - ***під****руч****ний****, рук****о****пис****ний*** (uk)
  - *Lehr****erin,*** *Lehrbüch****er*** (de)

Affixes

Affixes by position:

- prefix - ***over**estimate*, ***під**купити*
- suffix/postfix - *overestima**tion**, підкупити**ся***
- interfix - *speed**o**meter, рук**о**пис*
- circumfix/confix - ***ge**sproch**en**, **до**чекати**ся***

- infix - *abso**-bloody-**lutely, спат**оньк**и*

Affixes-shmaffixes

- Duplifix :)
  - *hokey-pokey, teenie-weenie*
  - *article-shmarticle, fancy-shmancy*
  - *helter-skelter*
  - *kitty-cat, chit-chat*
  - *bye-bye, choo-choo*

Affixes

Affixes by function:

- inflectional
- derivational

Affixes

Affixes by function:

- inflectional
  - *fox => fox**es**, стіл => стол**и**, Tasse => Tasse**n***
  - *high => high**er**, красив**ий** => **най**красив**іший***
  - *smell => smell**ing**, нюха**ти** => нюха**ю**, брати => **за**брати*
- derivational
  - *schön => Schön**heit**, краса => крас**ив**ий*
  - *happy => happi**ness**, хмара => **без**хмар**н**ий*
  - *sprechen => **ab**gesprochene*

Inflectional Affixes

- change the word form of a lexeme
- keep the part of speech

- examples:
  - *чит**ав**, чит**атиму**, **про**чит**ав**, чит**ай*** => ***читати*** (verb)
  - *high**er**, high**est*** => ***high*** (adj)

Synthetic vs. Analytic Languages

- Synthetic - relation between words is expressed by ***inflection***
    - *Я тобі напишу. Напишу я тобі. Тобі я напишу.*
    - *Я напишу тобі. Напишу тобі я. Тобі напишу я.*

- Analytic language - relation between words is expressed by ***word order*** and ***helper words***
    - *Ich werde dir schreiben. Dir werde ich schreiben.*
    - *I will write to you.*

Synthetic vs. Analytic Languages

- *the kind**est**, **most** rational* (en)
- *добр**ейш**ий, **самый** добрый* (ru)
- ***най**добр**іш**ий, **найбільш** добрий, ~~самий добрий~~* (uk)

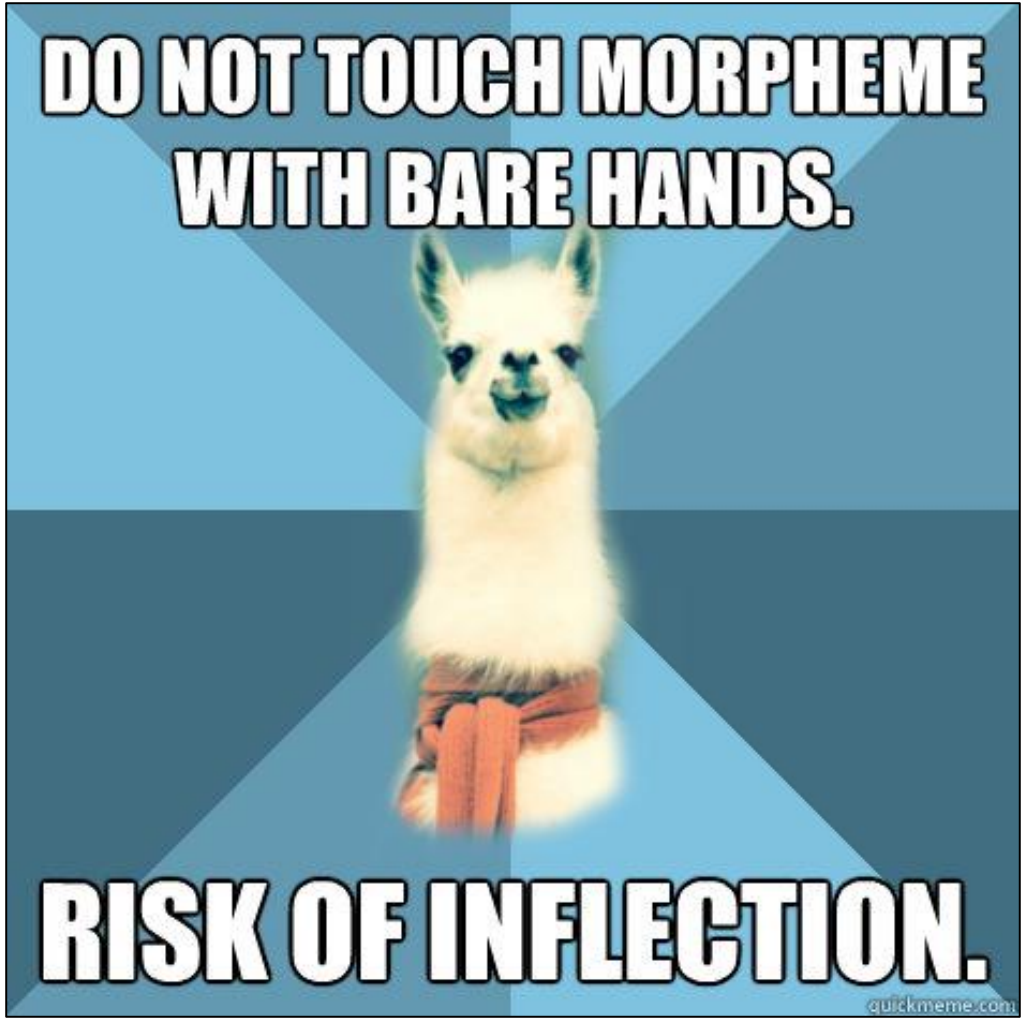
- ***will** write* (en)
- ***буду** писать* (ru)
- ***буду** писати, писати**му*** (uk)

Synthetic vs. Analytic Languages

- *the kind**est**, **most** rational* (en)
- *добр**ейш**ий, **самый** добрый* (ru)
- ***най**добр**іш**ий, **найбільш** добрий, ~~самий добрий~~* (uk)

- ***will** write* (en)
- ***буду** писать* (ru)
- ***буду** писати, писати**му*** (uk)

- *Evinizdeyim.* (tr) - Я є у Вас вдома.
- *Evinizdeymişim.* (tr) - Я був у Вас вдома.


DO NOT TOUCH MORPHEME
WITH BARE HANDS.
RISK OF INFLECTION.
quickmeme.com

Affixes

Affixes by function:

- inflectional
  - *fox => fox**es**, стіл => стол**и**, Tasse => Tasse**n***
  - *high => high**er**, красив**ий** => **най**красив**іший***
  - *smell => smell**ing**, нюха**ти** => нюха**ю**, брати => **за**брати*
- derivational
  - *schön => Schön**heit**, краса => крас**ив**ий*
  - *happy => happ**iness**, хмара => **без**хмар**н**ий*
  - *sprechen => **ab**gesprochen*

Derivational Affixes

- create a new lexeme, but keep the stem
- may change the part of speech

- examples:
  - *schön* => *Schön**heit*** (*schön*, adj => noun)
  - *хмара* => ***без**хмар**ний*** (*хмар*, noun => adj)
  - *sprechen* => ***ab**gesprochen* (*sprech*, verb => adj)

Derivational Affixes

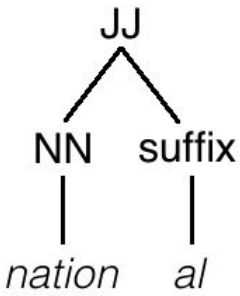
Word derivation: how to get from ***nation*** to ***denationalisation***?

NN

nation

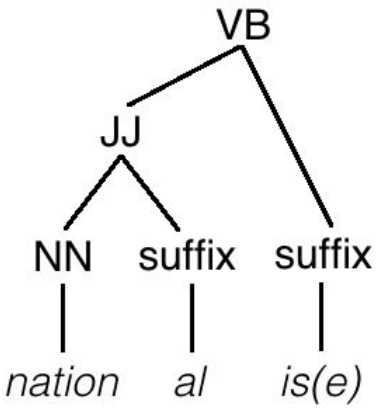
Derivational Affixes

Word derivation: how to get from ***nation*** to ***denationalisation***?



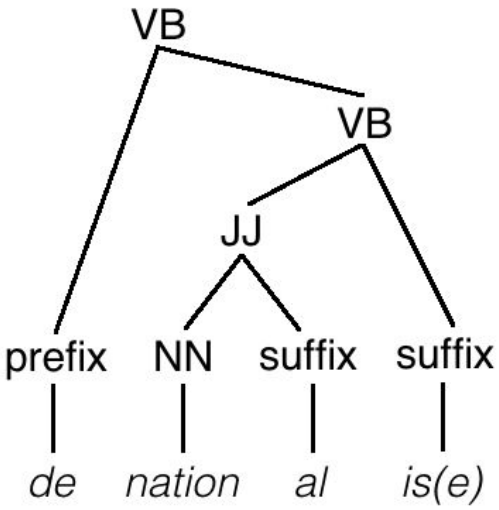
Derivational Affixes

Word derivation: how to get from ***nation*** to ***denationalisation***?



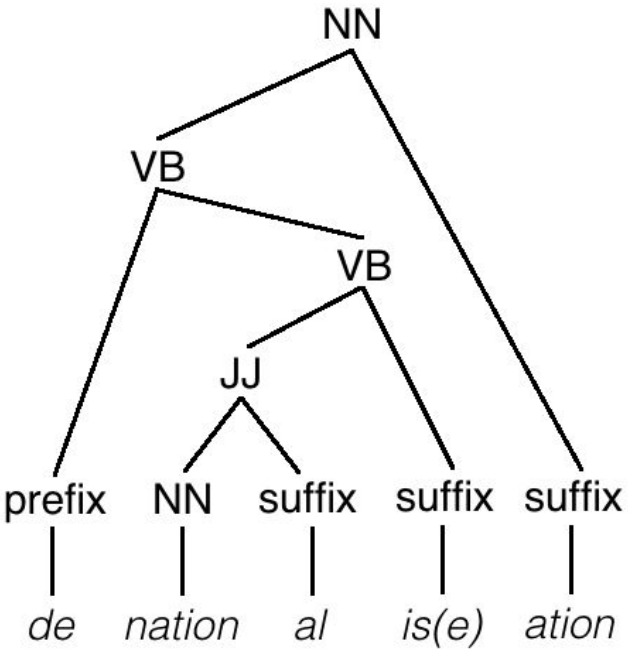
Derivational Affixes

Word derivation: how to get from ***nation*** to ***denationalisation***?



Derivational Affixes

Word derivation: how to get from ***nation*** to ***denationalisation***?



More exercise?

What is the derivation of…

- *counterproductiveness*
- *antirepresentationalist*

- *поназдоганяти*
- *перекотиполе*

- *учительствовать*
- *сверхприбыльный*

Affixes

Affixes by function:

- inflectional
  - change the word form of a lexeme
  - keep the part of speech

- derivational
  - create a new lexeme, but keep the stem
  - may change the part of speech

Affixes

Affixes by function:

- inflectional => lemmatization
- derivational => stemming

Lemmatization

Lemma - the base form of the word.

- *foxes => fox*
- *найкрасивіший => красивий*
- *smelling => smell*

How to lemmatize?

- NLP toolkits mentioned previously
- pymorphy2 (Russian, Ukrainian)
- write your own lemmatizer :)

Lemmatization in spaCy

```
import en_core_web_md

nlp = en_core_web_md.load()
```

```
quote = nlp("Roads? Where we're going we don't need roads.")
" ".join([token.lemma_ for token in quote])
```

```
'road ? where -PRON- be go -PRON- do not need road .'
```

Lemmatization in spaCy

```
import en_core_web_md

nlp = en_core_web_md.load()
```

```
quote = nlp("Roads? Where we're going we don't need roads.")
" ".join([token.lemma_ for token in quote])
```

```
'road ? where -PRON- be go -PRON- do not need road .'
```

```
quote = nlp("You're gonna need a bigger boat.")
" ".join([token.lemma_ for token in quote])
```

```
'-PRON- be go to need a big boat .'
```

Lemmatization in pymorphy2

```
import tokenize_uk
import pymorphy2
```

```
text = "Обговорімо основні етапи купівлі та розмитнення \"єврobляхи\"."
morph = pymorphy2.MorphAnalyzer(lang='uk')
```

```
tokenize_uk.tokenize_words(text)
print(tokens)
```

```
['Обговорімо', 'основні', 'етапи', 'купівлі', 'та', 'розмитнення', '"', 'єврobляхи', '"', '.']
```

```
print([morph.parse(word)[0].normal_form for word in tokens])
```

```
['обговорити', 'основний', 'етап', 'купівля', 'та', 'розмитнення', '"', 'єврobляха', '"', '.']
```

Stemming

Stem - the base morpheme of the word.

- *Schönheit => schön*
- *безхмарний => хмар*
- *abgesprochene => sprech*

How to stem?

- SnowballStem (English, Russian, and 16 more languages)
- PorterStemmer in nltk (English)
- Ukr_stemming (Ukrainian)
- write your own stemmer :)

Stemming in SnowballStem

In [7]:
```
import en_core_web_md
import snowballstemmer

nlp = en_core_web_md.load()
stemmer = snowballstemmer.stemmer('english')
```

In [8]:
```
quote = nlp("This is the beginning of a beautiful friendship.")
print(" ".join([token.lemma_ for token in quote]))
print(" ".join(stemmer.stemWords([token.text.lower()
                                  for token in quote])))
```

```
this be the beginning of a beautiful friendship .
this is the begin of a beauti friendship .
```

Usage in NLP

Lemmatization & Stemming

- text classification
- information retrieval
- dimensionality reduction

Lemmatization

- any task that needs a dictionary lookup

Stemming

- complex word identification

4. Other parts of the word

Word Parts

A word also consists of:

- characters
- syllables
- sounds/phonemes

Usage in NLP

Characters, syllables, sounds:

- Language identification
- Text readability/fluency
- Generation of rhymed poetry
- Text complexity
- Spelling correction
- Automatic hyphenation
- Onomastics

Conclusion

Features

- Capitalization, hyphenation, apostrophes
- Number of tokens, position of a token
- Lemma, stem
- Number of stems
- Number and types of affixes
- Length of the word/lemma/stem
- Number of syllables in a word
- Ratio of vowels vs. consonants
- Voiced vs. voiceless consonants
- All possible frequencies

Questions?