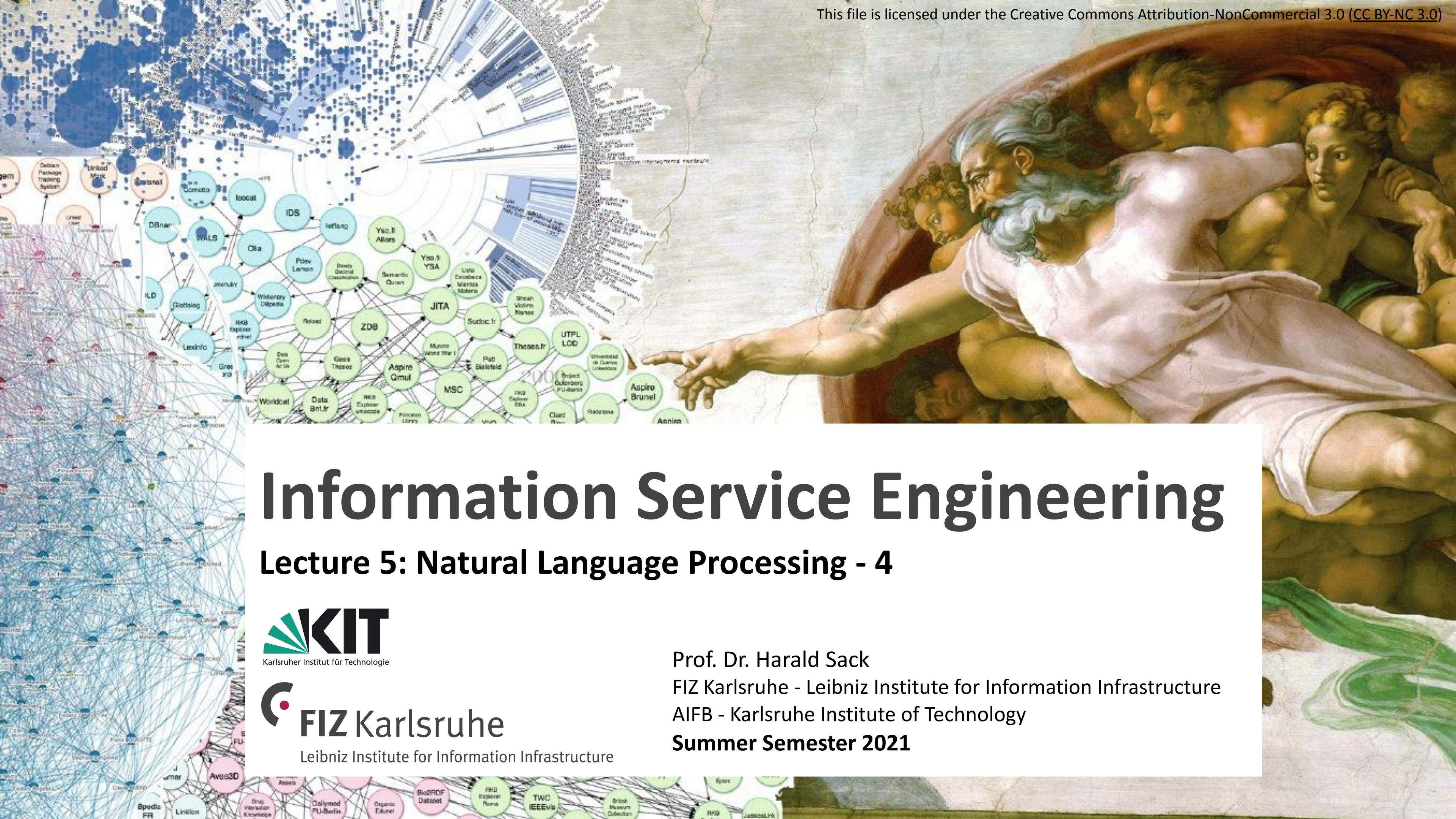


Information Service Engineering

Lecture 5: Natural Language Processing - 4

Prof. Dr. Harald Sack
FIZ Karlsruhe - Leibniz Institute for Information Infrastructure
AIFB - Karlsruhe Institute of Technology

Summer Semester 2021

Information Service Engineering

Last Lecture: Natural Language Processing (3)

2.0 What is Natural Language Processing?
2.1 NLP and Basic Linguistic Knowledge
2.2 Morphology
2.3 NLP Applications
2.4 NLP Techniques
2.5 NLP Challenges
2.6 Evaluation, Precision and Recall
2.7 Regular Expressions
2.8 Finite State Automata
2.9 Tokenization
2.10 Language Model and N-Grams
2.11 Part-of-Speech Tagging
2.12 Word Embeddings

- Finite State Automata
- Finite State Transducers
- Morphological Parsing with FST
- Tokenization
- Language Model and N-Grams

Information Service Engineering

Lecture 5: Natural Language Processing (4)

2.0 What is Natural Language Processing?
2.1 NLP and Basic Linguistic Knowledge
2.2 Morphology
2.3 NLP Applications
2.4 NLP Techniques
2.5 NLP Challenges
2.6 Evaluation, Precision and Recall
2.7 Regular Expressions
2.8 Finite State Automata
2.9 Tokenization
2.10 Language Model and N-Grams
2.11 Part-of-Speech Tagging
2.12 Word Embeddings

Language Model

$$P(S) = P(w_1, w_2, \cdots, w_n)$$

Probability of sequence of words

$$\Big\downarrow$$

Bayes Theorem and Chain Rule

$$P(S) = \prod_{i=1}^{n} P(w_i | w_1, \cdots, w_{i-1})$$

$$\Big\downarrow$$

Markov Assumption

$$P(S) = \prod_{i=1}^{n} P(w_i | w_{i-1})$$

N-gram model

Maximum Likelihood Estimation

$$P(w_n | w_{n-1}) = \frac{\#(w_{n-1} w_n)}{\sum_w \#(w_{n-1} w)} = \frac{\#(w_{n-1} w_n)}{\# w_{n-1}}$$

Normalized N-gram Frequencies

Maximum Likelihood Estimation

- **Example Corpus**:
  - <s> I saw the boy </s>
  - <s> the man is working </s>
  - <s> I walked in the street </s>

- **Vocabulary:**
  - V = {I,saw,the,boy,man,is,working,walked,in,street}

Estimating N-Gram Models

1. Bracket each sentence by **special start and end symbols <s>...</s>**:
   <s> to be or not to be **</s>**
2. Count the **frequency of each n-gram**:
   #(<s> to) = 1, #(to be) = 2,….
3. **Normalize** to get the **probability**:

$$P(not|or) = \frac{\#or\ not}{\#or}$$

This is the **relative frequency estimate**.

Maximum Likelihood Estimation

- **Example Corpus:**
  - <s> I saw the boy </s>
  - <s> the man is working </s>
  - <s> I walked in the street </s>

boy	I	in	is	man	saw	street	the	walked	working	<s>	</s>
1	2	1	1	1	1	1	3	1	1	3	3

unigram counts

Maximum Likelihood Estimation

	boy	I	in	is	man	saw	street	the	walked	working	<s>	</s>
boy	0	0	0	0	0	0	0	0	0	0	0	1
I	0	0	0	0	0	1	0	0	1	0	0	0
in	0	0	0	0	0	0	0	1	0	0	0	0
is	0	0	0	0	0	0	0	0	0	1	0	0
man	0	0	0	1	0	0	0	0	0	0	0	0
saw	0	0	0	0	0	0	0	1	0	0	0	0
street	0	0	0	0	0	0	0	0	0	0	0	1
the	1	0	0	0	1	0	1	0	0	0	0	0
walked	0	0	1	0	0	0	0	0	0	0	0	0
working	0	0	0	0	0	0	0	0	0	0	0	1
<s>	0	2	0	0	0	0	0	1	0	0	0	0
</s>	0	0	0	0	0	0	0	0	0	0	0	0

bigram counts

Maximum Likelihood Estimation

- **Estimation** of the Maximum Likelihood Estimation for a new sentence:
    - <s> I saw the man

$$P(S) = P(I| < s >) \cdot P(saw|I) \cdot P(the|saw) \cdot P(man|the)$$

$$= \frac{\#(< s > I)}{\#(< s >)} \cdot \frac{\#(I\ saw)}{\#(I)} \cdot \frac{\#(saw\ the)}{\#(saw)} \cdot \frac{\#(the\ man)}{\#(the)}$$

$$= \frac{2}{3} \cdot \frac{1}{2} \cdot \frac{1}{1} \cdot \frac{1}{3} = \frac{1}{9}$$

You may find a Collaborative Notebook with this example here.

Unknown Words

- What if a word occurs that is not part of our vocabulary?
  - <s> I saw the **girl** </s>

- **Closed Vocabulary Assumption**:
  The test set can contain only words from our vocabulary.

- **Open Vocabulary Assumption:**
  The test set can contain **unknown words** (**out of vocabulary words**, **OOV**) that are not part of our vocabulary.

Open Vocabulary

- In an **Open Vocabulary system**, unknown words are modeled by adding a pseudo-word **<UNK>**.
  1. **Choose** a (fixed) vocabulary.
  2. **Convert** in the **training** set any word not in the vocabulary (**OOV word**) into **<UNK>**.
  3. **Convert** in the **test set** any unknown word into **<UNK>**.
  4. **Estimate** probabilities for **<UNK>** like for any regular vocabulary word.

Strategies to Deal with Unknown N-grams

- **Corpus**:
    - <s> I saw the boy </s>
    - <s> the man is working </s>
    - <s> I walked in the street </s>

- **Test set:**
    - <s> I saw the man in the street </s>

$$P(S) = P(I|<s>) \cdot P(saw|I) \cdot P(the|saw) \cdot P(man|the) \cdot P(in|man) \cdot P(the|in) \cdot P(street|the)$$

$$P(S) = \frac{\#(<s> I)}{\#(<s>)} \cdot \frac{\#(I\ saw)}{\#(I)} \cdot \frac{\#(saw\ the)}{\#(saw)} \cdot \frac{\#(the\ man)}{\#(the)} \cdot \frac{\#(man\ in)}{\#(man)} \cdot \frac{\#(in\ the)}{\#(in)} \cdot \frac{\#(the\ street)}{\#(the)}$$

$$P(S) = \frac{2}{3} \cdot \frac{1}{2} \cdot \frac{1}{1} \cdot \frac{1}{3} \cdot \frac{0}{1} \cdot \frac{1}{1} \cdot \frac{1}{3} = 0$$

Strategies to Deal with Unknown N-grams

Example:

- Shakespeare corpus consists of N=884,647 word tokens and a vocabulary of V=29,066 word types.
- Only 30,000 word types occurred (of possible 475,000 referenced by Webster's 3rd New International Dictionary).
  - Words not in the training data ⇒ *P(unknown Word)=0*
- Only **0.04% of all possible 2-grams** occurred.
- The probability of **99.96% of all possible 2-grams is 0**.

Laplace Smoothing

- **Assign a small probability to all unknown N-grams** that do not occur in the test corpus
  - i.e. add 1

	boy	I	in	is	man	saw	street	the	walked	working
boy	1	1	1	1	1	1	1	1	1	1
I	1	1	1	1	1	2	1	1	2	1
in	1	1	1	1	1	1	1	2	1	1
is	1	1	1	1	1	1	1	1	1	2
man	1	1	1	2	1	1	1	1	1	1
saw	1	1	1	1	1	1	1	2	1	1
street	1	1	1	1	1	1	1	1	1	1
the	2	1	1	1	2	1	2	1	1	1
walked	1	1	2	1	1	1	1	1	1	1
working	1	1	1	1	1	1	1	1	1	1

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1}w_i)}{\#(w_{i-1})} \quad \Rightarrow \quad P(w_i|w_{i-1}) = \frac{\#(w_{i-1}w_i)+1}{\#(w_{i-1})+|V|}$$

Advanced Language Modelling

- Laplace Smoothing is not optimal.
  - OK for domains where there are not so many zeros.
  - OK for text classification.
- The most commonly used method for smoothing:
  - Extended Interpolated **Kneser-Ney Smoothing** (1995).
- For very large N-gram corpora like the Web:
  - **Stupid Backoff Smoothing** (2007).
  - Only store n-grams with count > threshold, or use entropy to prune less-important n-grams.

How to Evaluate a Language Model?

- How do we know whether one language model is better than another?
- There are two ways to evaluate models:
  - **Intrinsic evaluation** captures how well the model captures what it is supposed to capture (e.g. probabilities).
  - **Extrinsic (task-based) evaluation** captures how useful the model is in a particular task.
- Both cases require an **evaluation metric** that allows us to measure and compare the performance of different models.

How to Evaluate a Language Model?

- We want to measure how similar the **predictions** of a model are to **real text**.
- Divide the corpus into two parts:
  - **Training set** (“seen”) and **test set** (“unseen”).
- Build a **language model** from the **training set**
  - Compute word frequencies, etc.
- For intrinsic evaluation: estimate the **probability** of the **test set**, i.e. calculate the **average branching factor** of the test set.

Average Branching Factor

- How well can we predict the next word in a sequence?
  - I always order pizza with cheese and ____
  - The winner of the 2018 UEFA Europa League Final was ____
  - I saw a ____

mushrooms	**0.1**
pepperoni	**0.1**
salami	**0.1**
tuna	**0.05**
anchovies	**0.01**
….	
strawberries	**0.00001**
….	
and	**1e-100**

- A **better language model**
  - is one which assigns a **higher probability** to the word that **actually occurs**.

Average Branching Factor

- The **branching factor** of a language is the **number of possible next words that can follow any word**.
- A good language model should be able to **minimize** this number
    - i.e., give a higher probability to the words that occur in real texts.
- The **average branching factor** is referred to as **Perplexity (PP)** and is the most common evaluation metric for N-gram language models.
- **Perplexity** is a measurement of how well a probability model predicts a sample.

Perplexity

- The **perplexity** of a test set according to a language model is the **geometric mean of the inverse test set probability** computed by the model.

$$P(S) = P(w_1, w_2, \cdots, w_n)$$

test set probability

$$PP(S) = P(w_1, w_2, \cdots, w_n)^{-\frac{1}{n}} = \sqrt[n]{\frac{1}{P(w_1, w_2, \cdots, w_n)}}$$

geometric mean of inverse test set probability

$$PP(S) = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{P(w_i|w_1, w_2, \cdots, w_{i-1})}}$$

applying Bayes and chain rule

F. Jelinek et al., Perplexity — a measure of the difficulty of speech recognition tasks, JASA, 1977

Perplexity

- Since language model **probabilities are very small**, multiplying them together often yields to **underflow**. It is often better to **use logarithms instead**, so replace

$$PP(S) = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{P(w_i|w_1, w_2, \cdots, w_{i-1})}}$$

with

$$PP(S) = \exp\left(-\frac{1}{n}\sum_{i=1}^{n} \log_e P(w_i|w_1, ..., w_{i-1})\right)$$

use log for convenience

21

Perplexity Example

- Wall Street Journal (19,979 word vocabulary)
  - Training set: 38 million words
  - Test set: 1.5 million words

- Perplexity:

Unigram:	962
Bigram:	170
Trigram:	109

Extrinsic Evaluation

- Perplexity tells us which Language Model assigns a higher probability to unseen text.
- This doesn’t necessarily tell us which Language Model is actually better for a specific task (i.e. is better at scoring candidate sentences).
- **Task-based evaluation**:
  - Train model A, plug it into your system for performing task T.
  - Evaluate performance of system A on task T.
  - Train model B, plug it in, evaluate system B on same task T.
  - Compare scores of system A and system B on task T.

Word Error Rate

- Originally developed for speech recognition.
- How much does the predicted sequence of words differ from the actual sequence of words in the reference transcript?

$$WER = \frac{\#Insertions + \#Deletions + \#Substitutions}{\#words\ in\ transcript}$$

Same as the Levenshtein distance

- Example:
  - Ground truth: *Where no man has gone before*
  - Prediction: *Where no man stands alone*
  - 2 substitutions + 1 deletion: WER = 3/6 = 50%

Information Service Engineering

Lecture 5: Natural Language Processing (4)

2.0 What is Natural Language Processing?
2.1 NLP and Basic Linguistic Knowledge
2.2 Morphology
2.3 NLP Applications
2.4 NLP Techniques
2.5 NLP Challenges
2.6 Evaluation, Precision and Recall
2.7 Regular Expressions
2.8 Finite State Automata
2.9 Tokenization
2.10 Language Model and N-Grams
2.11 Part-of-Speech Tagging
2.12 Word Embeddings

Part-of-Speech

- **Category of words** which have similar grammatical properties.
- Words that are assigned to the **same word part of speech** generally display **similar behavior** in terms of **syntax**.
- Also referred to as
  - lexical categories, word classes, morphological classes, lexical tags.
- *Dionysius Thrax of Alexandria* [c. 100 BC] describes 8 parts-of-speech:
  - Noun
  - Verb
  - Pronoun
  - Preposition
  - Adverb
  - Conjunction
  - Participle
  - Article

(English) Word Classes

- **Nouns**
  - A word that functions as the **name of some specific thing or set of things**, as e.g. living creatures, objects, places, actions, qualities, states of existence, or ideas.
  - **Proper Nouns**
    Names of specific entities, as e.g. *Harald, Karlsruhe, KIT, etc*.
  - **Common Nouns**
    - **Count Nouns**
      Nouns that allow enumeration, as e.g. *one dog, two dogs, etc.*
    - **Mass Nouns**
      Conceptualization of a homogeneous group, as e.g. *snow, heat, salt, etc.*

(English) Word Classes

- **Verbs**
  - A word that is **referring to actions, processes, occurrences, states of being**, etc.
  - **Main Verbs**
    provide the main semantic content of the clause, as e.g.
    *The dog **ate** my homework.*
  - **Auxiliary Verbs (Auxiliaries)**
    add functional or grammatical meaning to the clause in which it appears, such as to express *tense, aspect, modality, voice, emphasis*, etc; usually accompany a main verb, as e.g.
    ***Do** you want tea?*

(English) Word Classes

- **Adjectives**
  - An adjective **describes, modifies or gives more information about a noun or pronoun**.
  - **Examples:**
    big, happy, green, young, fun, crazy, three
- **Pronouns**
  - A pronoun is **used in place of a noun or noun phrase** to avoid repetition.
  - **Examples:**
    I, you, we, they, he, she, it, me, us, them, him, her, this, those

More Examples of POS Tags

- **Noun**: book/books, nature, Germany, Sony
- **Verb**: eat, wrote
- **Auxiliary**: can, should, have
- **Adjective**: new, newer, newest
- **Adverb**: well, urgently
- **Number**: 872, two, first
- **Article/Determiner**: the, some
- **Conjunction**: and, or
- **Pronoun**: he, my
- **Preposition**: to, in
- **Particle**: off, up
- **Interjection**: Ow, Eh

Open vs. Closed Word Classes

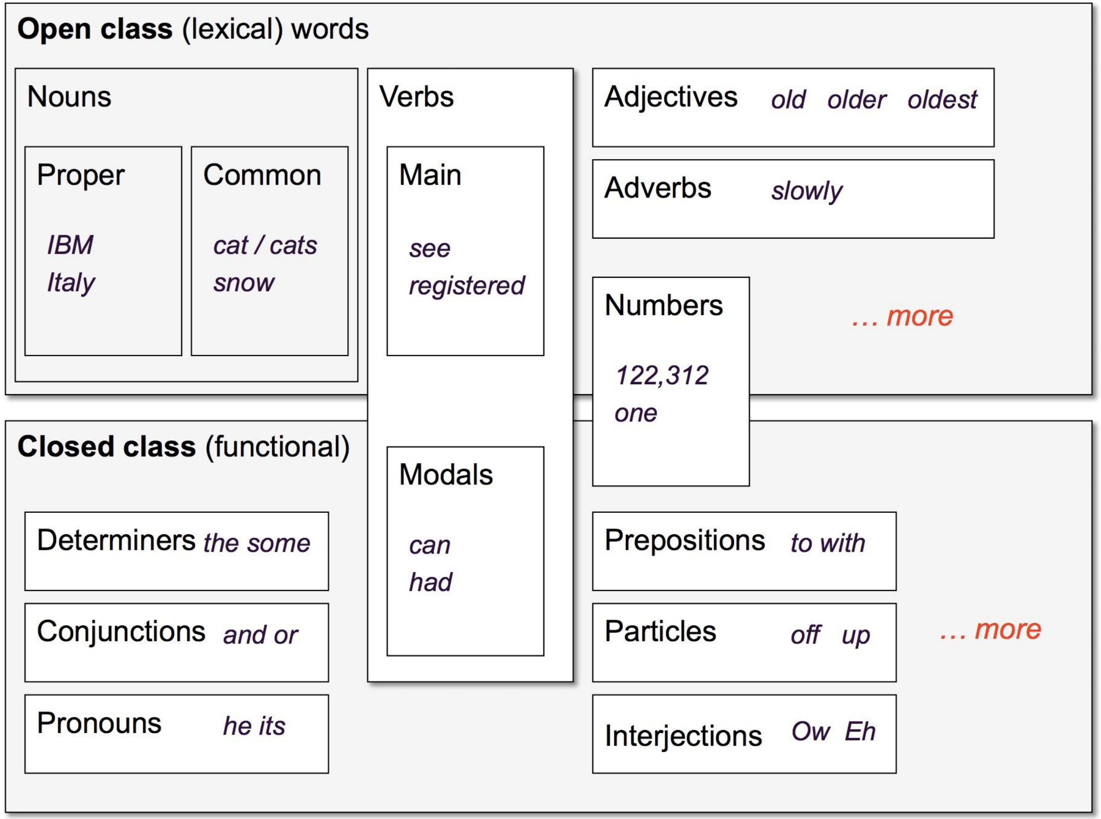
- **Closed Word Classes**
  - Limited number of words, usually do not grow, as e.g., Auxiliary, Article, Determiner, Conjunction, Pronoun, Preposition, Particle, Interjection.
  - Usually **function words** (i.e. short common words which play a role in grammar).
  - Examples (in English):
    - prepositions: on, under, over, …
    - particles: up, down, on, off, …
    - determiners: a, an, the, …
    - pronouns: she, who, I, ..
    - conjunctions: and, but, or, …
    - auxiliary verbs: can, may, should, …
    - numerals: one, two, three, third, …

Open vs. Closed Word Classes

- **Closed Word Classes** (functional words)
  - Limited number of words, usually do not grow, as e.g.,
    Auxiliary, Article, Determiner, Conjunction, Pronoun, Preposition, Particle, Interjection.

- **Open Word Classes** (lexical words)
  - Unlimited number of words, as e.g., (for English)
    Noun, Verb, Adverb, Adjective.

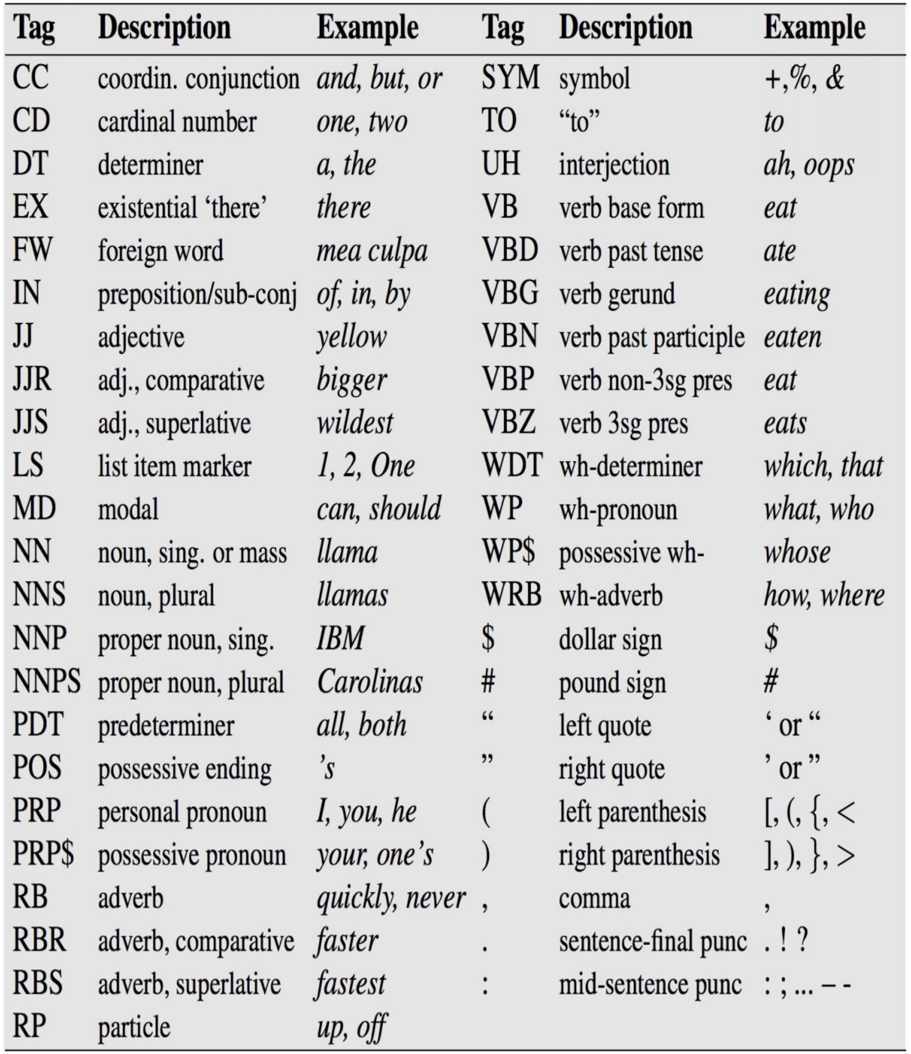
Open vs. Closed Word Classes



POS Tagsets

- There are various **POS tagsets** based on different granularity of tags.
  - **Brown tagset** (Francis and Kucera 1982, 87 tags)
    - Based on Brown corpus, i.e. Brown University Standard Corpus of Present-Day American English.
  - **C5 tagset** (61 tags)
  - **C7 tagset** (146 tags!)
  - **Penn TreeBank** (Marcus et al. 1993, 45 tags)
    - Simplified version of Brown tag set;
      de facto standard for English now.
  - **Prague Dependency Treebank** (Czech, Hajic 2006, 4452 tags)

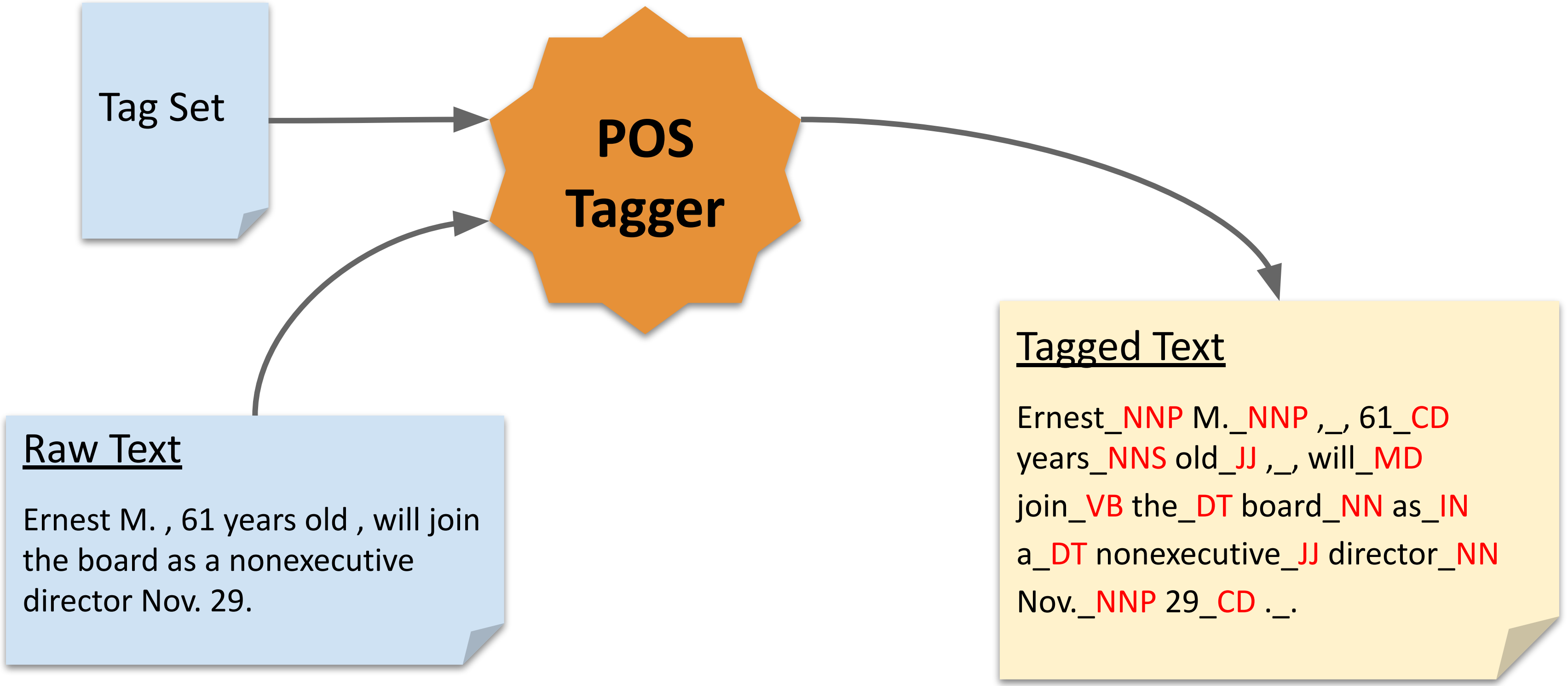
Penn Treebank Tagset



Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two*	TO	“to”	*to*
DT	determiner	*a, the*	UH	interjection	*ah, oops*
EX	existential ‘there’	*there*	VB	verb base form	*eat*
FW	foreign word	*mea culpa*	VBD	verb past tense	*ate*
IN	preposition/sub-conj	*of, in, by*	VBG	verb gerund	*eating*
JJ	adjective	*yellow*	VBN	verb past participle	*eaten*
JJR	adj., comparative	*bigger*	VBP	verb non-3sg pres	*eat*
JJS	adj., superlative	*wildest*	VBZ	verb 3sg pres	*eats*
LS	list item marker	*1, 2, One*	WDT	wh-determiner	*which, that*
MD	modal	*can, should*	WP	wh-pronoun	*what, who*
NN	noun, sing. or mass	*llama*	WP$	possessive wh-	*whose*
NNS	noun, plural	*llamas*	WRB	wh-adverb	*how, where*
NNP	proper noun, sing.	*IBM*	$	dollar sign	$
NNPS	proper noun, plural	*Carolinas*	#	pound sign	#
PDT	predeterminer	*all, both*	“	left quote	‘ or “
POS	possessive ending	*’s*	”	right quote	’ or ”
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	possessive pronoun	*your, one’s*	)	right parenthesis	],), }, >
RB	adverb	*quickly, never*	,	comma	,
RBR	adverb, comparative	*faster*	.	sentence-final punc	. ! ?
RBS	adverb, superlative	*fastest*	:	mid-sentence punc	: ; ... – -
RP	particle	*up, off*			

http://www.clips.ua.ac.be/pages/mbsp-tags

Part-of-Speech Tagging



Raw Text

Ernest M. , 61 years old , will join the board as a nonexecutive director Nov. 29.

Tagged Text

Ernest_NNP M._NNP ,_, 61_CD years_NNS old_JJ ,_, will_MD join_VB the_DT board_NN as_IN a_DT nonexecutive_JJ director_NN Nov._NNP 29_CD ._.

Part-of-Speech Tagging

- The process of assigning a part of speech to each word in a text.
- **Challenge**: words often have **more than one POS** (ambiguity).
- Examples:
  - On my **back**$_{\text{[NN]}}$ (noun)
  - The **back**$_{\text{[JJ]}}$ door (adjective)
  - Win the voters **back**$_{\text{[RB]}}$ (adverb)
  - Promised to **back**$_{\text{[VB]}}$ the bill (verb)
- Typical output of a POS-Tagger:
  - The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

Why Part-of-Speech Tagging?

POS tagging is a prerequisite for further analysis:

- **Speech synthesis:**
  - How to pronounce "*lead*"?
  - INsult or inSULT, OBject or objECT, OVERflow or overFLOW, DIScount or disCOUNT, CONtent or conTENT.
- **Parsing:**
  - What words are in the sentence?
  - Unique tag to each word reduces the number of required parses.
- **Information extraction**:
  - Finding names, relations, etc.
- **Machine Translation**:
  - The *noun* "content" may have a different translation from the *adjective.*

Ambiguity in POS Tags

- 45-tags, Brown corpus
  - Unambiguous (1 tag): 38,857
  - Ambiguous: 8,844
    - 2 tags: 6,731
    - 3 tags: 1,621
    - 4 tags: 357
    - 5 tags: 90
    - 6 tags: 32
    - 7 tags: 6 (well, set, round, open, fit, down)
    - 8 tags: 4 ('s, half, back, a)
    - 9 tags: 3 (that, more, in)

Vanilla Baseline Method

1. Tagging **unambiguous words** with the **correct label**.
2. Tagging **ambiguous words** with their **most frequent label**.
3. Tagging **unknown words** as a **noun**.

- This method (*Baseline*) performs with around **90% accuracy**.
- **State-of-the-art POS tagger** achieve around **97% accuracy**.
- **Humans** (*Ceiling*) perform around **97% accuracy**.

POS Tagging, State-of-the-Art Wiki

The Jabberwocky Contest

Lewis Caroll, Jabberwocky (1855)

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.

"Beware the Jabberwock, my son!
The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
The frumious Bandersnatch!"

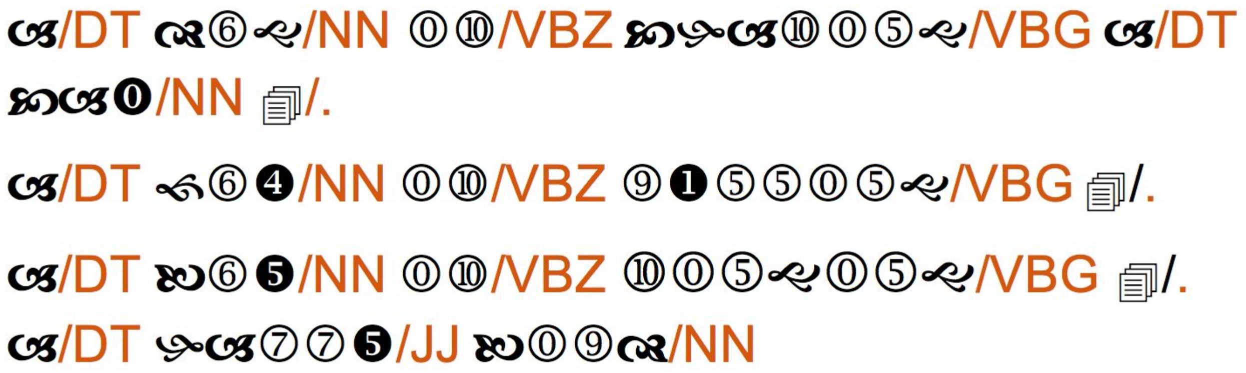
lJohn Tenniel (1820-1914), lustration on the poem Jabberwocky, public domain

Algorithms for POS Tagging

- **Rule-based POS Tagging**
  - 2-step approach:
    1. Dictionary or FST to assign a list of potential tags
    2. Hand-written rules to restrict to a POS tag (hundreds needed…)
- **Stochastic (Probabilistic Tagging, Machine Learning)**
  - **Hidden Markov Models**
  - MEMMs (Maximum Entropy Markov Models)
- **Transformation Based Tagging (Machine Learning)**
  - Combination of Rule-based and Stochastic Tagging
  - Rules are learned from data

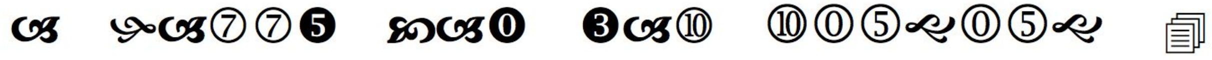
Stochastic POS Tagging

- What is the **most likely sequence of tags** for the **given sequence of words w**?
- Let's try:



/DT /NN /VBZ /VBG /DT
/NN /.
/DT /NN /VBZ /VBG /.
/DT /NN /VBZ /VBG /.
/DT /JJ /NN

- What is the most likely tag sequence for



Stochastic POS Tagging

- Let’s try:

  🙵/DT 🙰⑥🙤/NN ⓪⑩/VBZ 🙱🙢🙵⑩⓪⑤🙤/VBG 🙵/DT 🙱🙵⓿/NN 🗐/.

  - a/DT dog/NN is/VBZ chasing/VBG a/DT cat/NN ./.

  🙵/DT 🙚⑥➍/NN ⓪⑩/VBZ ⑨➊⑤⑤⓪⑤🙤/VBG 🗐/.

  - a/DT fox/NN is/VBZ running/VBG ./.

  🙵/DT 🙲⑥➎/NN ⓪⑩/VBZ ⑩⓪⑤🙤⓪⑤🙤/VBG 🗐/.

  - a/DT boy/NN is/VBZ singing/VBG ./.

  🙵/DT 🙢🙵⑦⑦➎/JJ 🙲⓪⑨🙰/NN

  - a/DT happy/JJ bird/NN

  🙵 🙢🙵⑦⑦➎ 🙱🙵⓿ ➌🙵⑩ ⑩⓪⑤🙤⓪⑤🙤 🗐

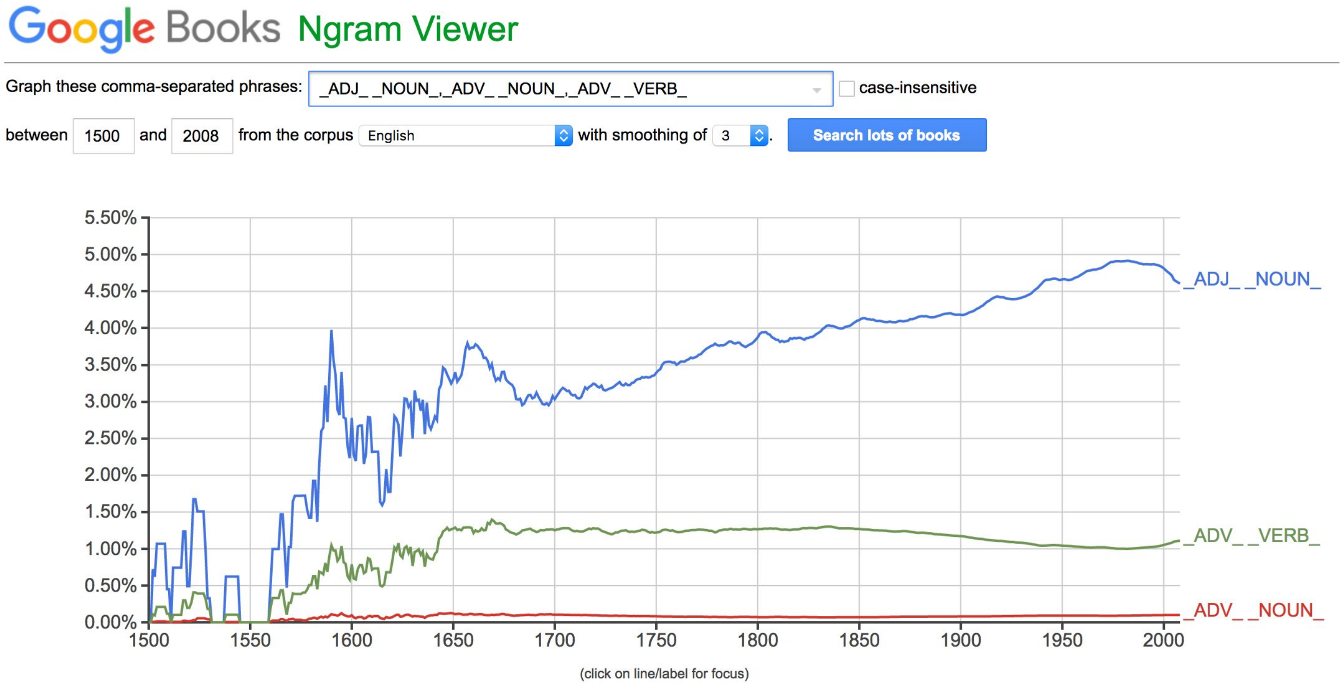
  - a happy cat was singing .

Stochastic POS Tagging

- How do you predict tags?
- Two types of information are useful
  - Relations between **words and tags**
    - as e.g. a/DT, dog/NN, is/VBZ, chasing/VBG, ...
  - Relations between **tags and tags**
    - as e.g., DT NN, DT JJ NN, …

Stochastic POS Tagging

- Model POS tagging as **sequence tagging problem**.
- Some POS-Tag sequences are more likely than others.



Graph these comma-separated phrases: _ADJ_ _NOUN_,_ADV_ _NOUN_,_ADV_ _VERB_ case-insensitive

between 1500 and 2008 from the corpus English with smoothing of 3.

Google N-gram Viewer Example

Stochastic POS Tagging

- How can we further resolve POS-Tag ambiguity?
- Making a decision based on:
  - **Current Observation**:
    - Word (W_0): *„the“* -> DT
    - Prefix, Suffix: *„unfathomable“* *„un“* -> JJ, *„able“* -> JJ
    - Lowercased word: *„New“* *„new“* -> JJ
    - Word shape: *„35-years-old“* *„d-a-a“* -> JJ
  - **Surrounding observations**
    - Words (W_{+1}, W_{-1})
  - **Previous decisions**
    - POS tags (T_{-1}, T_{-2})

Hidden Markov Models (HMM)

- Finding the best **sequence of tags (t_1,...,t_n)** that corresponds to the **sequence of observations (w_1,...,w_n)**.

She$_1$ promised$_2$ to$_3$ back$_4$ the$_5$ bill$_6$

$w =$ w_1 w_2 w_3 w_4 w_5 w_6

$t =$ t_1 t_2 t_3 t_4 t_5 t_6

RPR$_1$ VBD$_2$ TO$_3$ VB$_4$ DT$_5$ NN$_6$

Hidden Markov Models (HMM)

- Finding the best **sequence of tags (t_1,...,t_n)** that corresponds to the **sequence of observations (w_1,...,w_n)**.
- Probabilistic View
  - Considering all possible sequences of tags.
  - **Choosing** the tag sequence from this universe of sequences, which is **most probable given the observation sequence**.

^ means "our estimate of the best one"

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} P(t_1^n | w_1^n)$$

argmax f(x) means "the x such that f(x) is maximized"

Hidden Markov Models (HMM)

- Using the **Bayes Rule**:

$$\hat{t}_1^n = \text{argmax}_{t_1^n} P(t_1^n | w_1^n)$$

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

$$P(t_1^n | w_1^n) = \frac{P(w_1^n | t_1^n) \cdot P(t_1^n)}{P(w_1^n)}$$

$$\hat{t}_1^n = \text{argmax}_{t_1^n} \underbrace{P(w_1^n | t_1^n)}_{\text{Likelihood of word sequence}} \cdot \underbrace{P(t_1^n)}_{\text{Prior probability of tag sequence}}$$

Hidden Markov Models (HMM)

- Using the **Markov Assumption**:

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} P(w_1^n|t_1^n) \cdot P(t_1^n)$$

Emission Probability

$$P(w_1^n|t_1^n) \simeq \prod_{i=1}^{n} P(w_i|t_i)$$

only depends on its POS tag
independent of other words

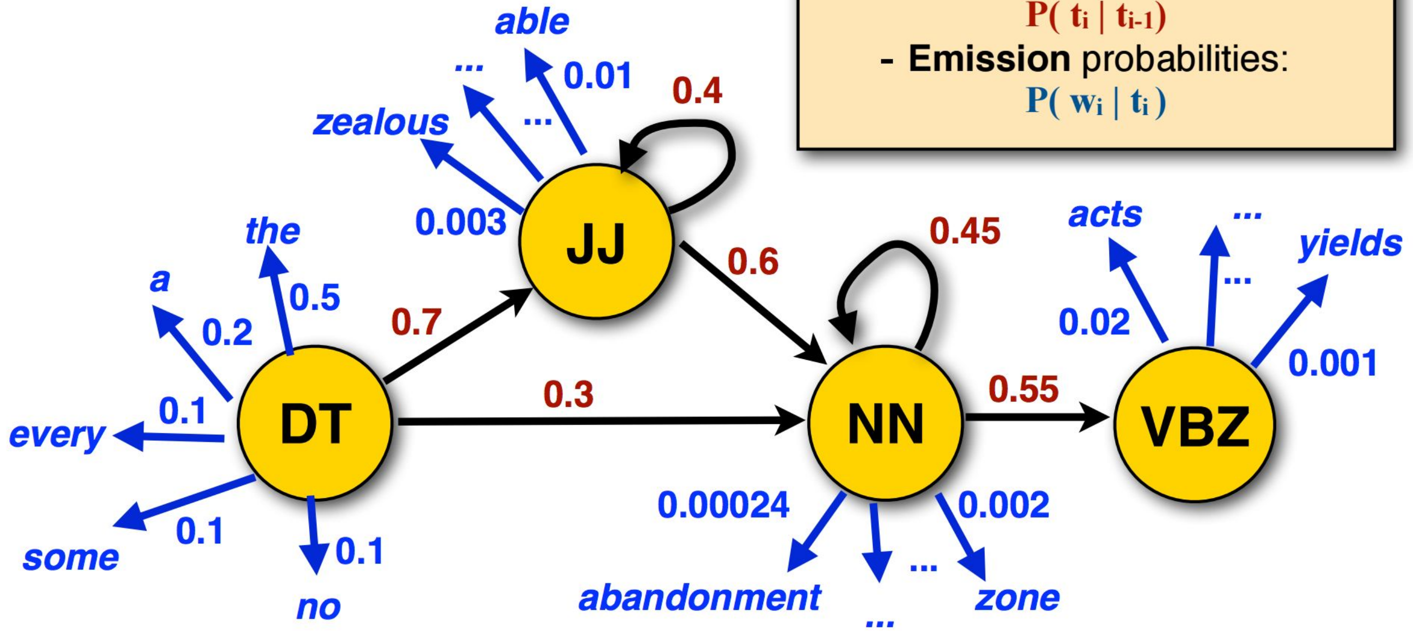
Transmission Probability

$$P(t_1^n) \simeq \prod_{i=1}^{n} P(t_i|t_{i-1})$$

only depends on previous POS tag
i.e. bigram

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} \prod_{i=1}^{n} P(w_i|t_i) \cdot P(t_i|t_{i-1})$$

Hidden Markov Models (HMM)



An HMM defines

- **Transition** probabilities:

  $P(t_i \mid t_{i-1})$

- **Emission** probabilities:

  $P(w_i \mid t_i)$

Information Service Engineering

Lecture 5: Natural Language Processing (4)

2.0 What is Natural Language Processing?

2.1 NLP and Basic Linguistic Knowledge

2.2 Morphology

2.3 NLP Applications

2.4 NLP Techniques

2.5 NLP Challenges

2.6 Evaluation, Precision and Recall

2.7 Regular Expressions

2.8 Finite State Automata

2.9 Tokenization

2.10 Language Model and N-Grams

2.11 Part-of-Speech Tagging

2.12 Word Embeddings

How to represent Textual Data in the Computer?

- For sake of simplicity we are focussing on the question
  How to represent words in the computer?
- Traditional solution:
  - represent words as **unique integers** associated with words:

```
{1=movie, 2=hotel, 3=apple, 4=movies, 5=art}
```

- Equivalent solution: **1-Hot Encoding**

```
movie = [1, 0, 0, 0, 0]
hotel = [0, 1, 0, 0, 0]
. . .
art   = [0, 0, 0, 0, 1]
```

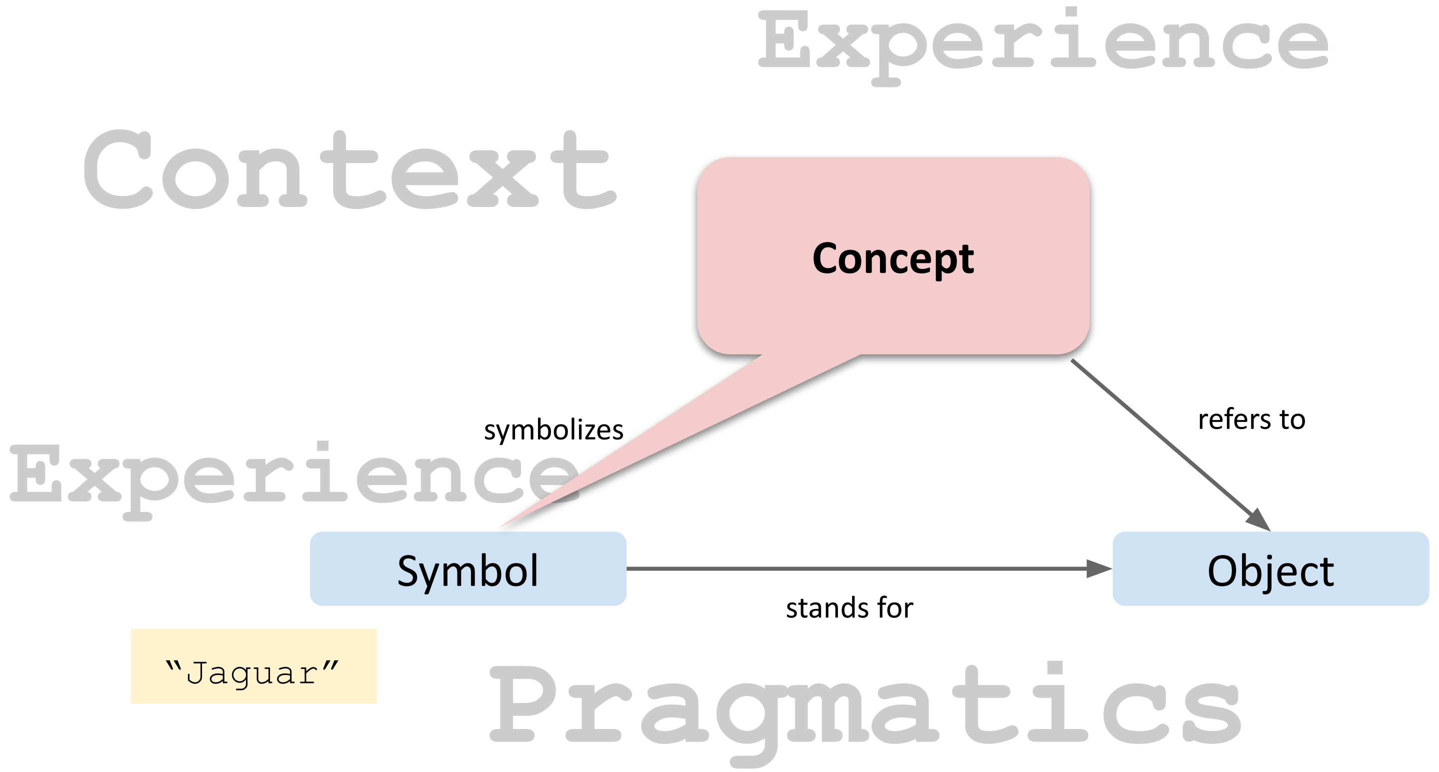
1-Hot Encoding

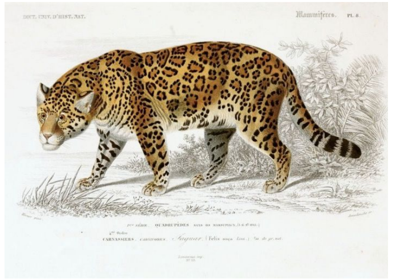
- Most basic representation of any textual unit
- **Vectorspace**: word vectors constitute an orthogonal base
  - orthogonal ($x^T y = 0$)
  - normalized ($x^T x = 1$)
- **Problem 1:** No relation to semantics
  - E.g. *car* and *automobile* are different (orthogonal) vectors.
  - All words are equidistant:
    $\|cat - dog\| = \|proton - carrier\|$
- **Problem 2**: polysemy
  - Should *jaguar (the cat)* have the same vector as *jaguar (the car)*?

Feature Based Representation of Words

- Words can also be represented with handcrafted **features and relations**
- Potential features:
  - Morphological features: *prefix, suffix, stem, lemma, …*
  - Grammatical features: *part-of-speech, gender, number, …*
  - Structural features: *capitalization, hyphen, digit(s),...*
- Potential relations:
  - *Synonyms, antonyms, hyper- and hyponyms, meronyms and holonyms,...*
- **Problems**:
  - Annotation requires high manual effort, annotator disagreement, accuracy, scalability,...

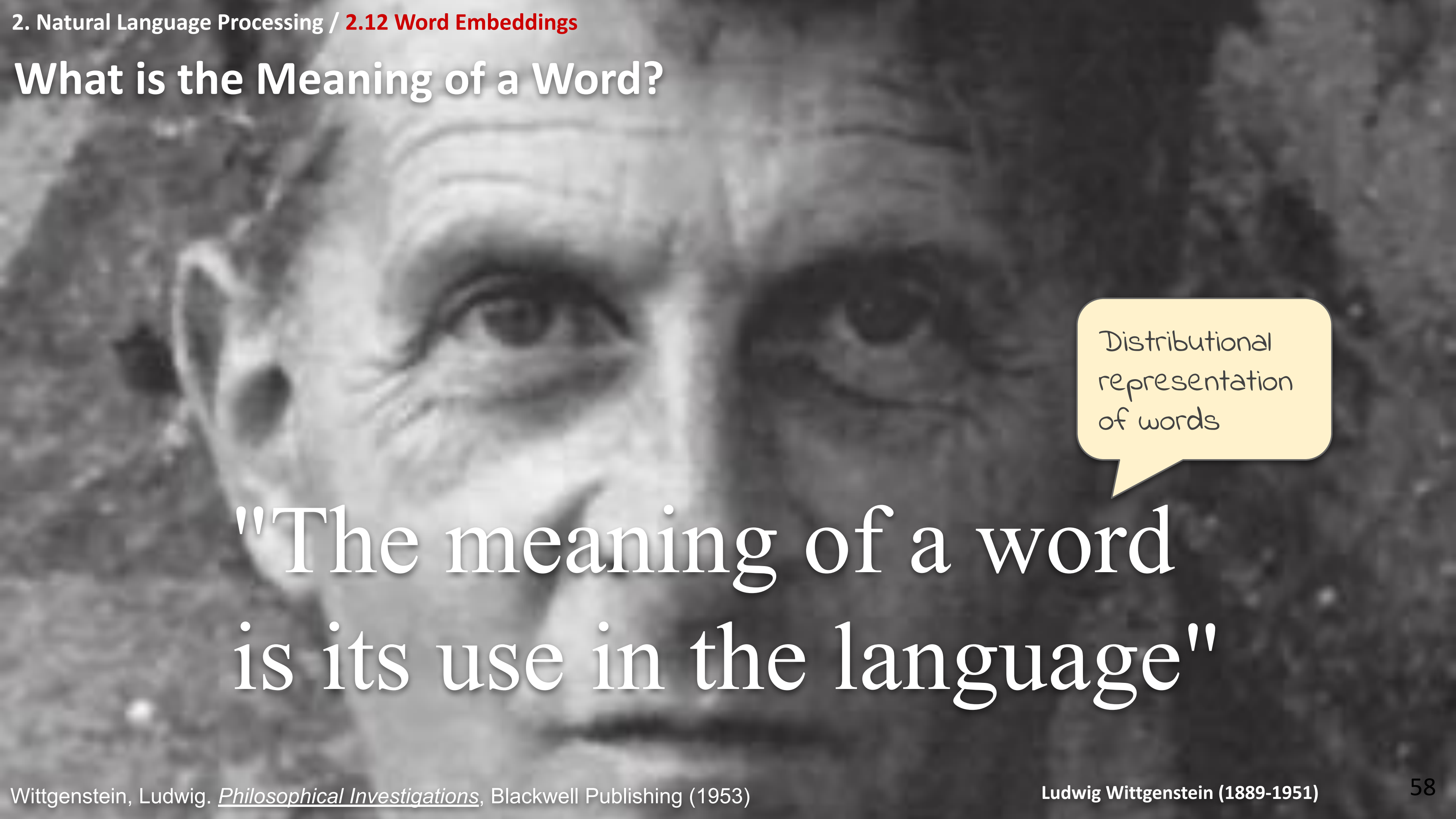
What is the Meaning of a Word?





Ogden, Richards: The **Meaning of Meaning**: A Study of the Influence of Language upon Thought and of the Science of Symbolism, 1923

What is the Meaning of a Word?



Distributional representation of words

"The meaning of a word is its use in the language"

Wittgenstein, Ludwig. *Philosophical Investigations*, Blackwell Publishing (1953)

Ludwig Wittgenstein (1889-1951)

Let’s Define Words by their Usage

- In particular, words are defined by their environments (i.e. the words around them).

- *“If [words] A and B have almost **identical environments** […] we say that they are **synonyms**.”*
  Zellig S. Harris (1954)

- Thereby: semantic representations for words can be derived through analysis of patterns of lexical co-occurrence in large language corpora.

"You shall know a word by the company it keeps"

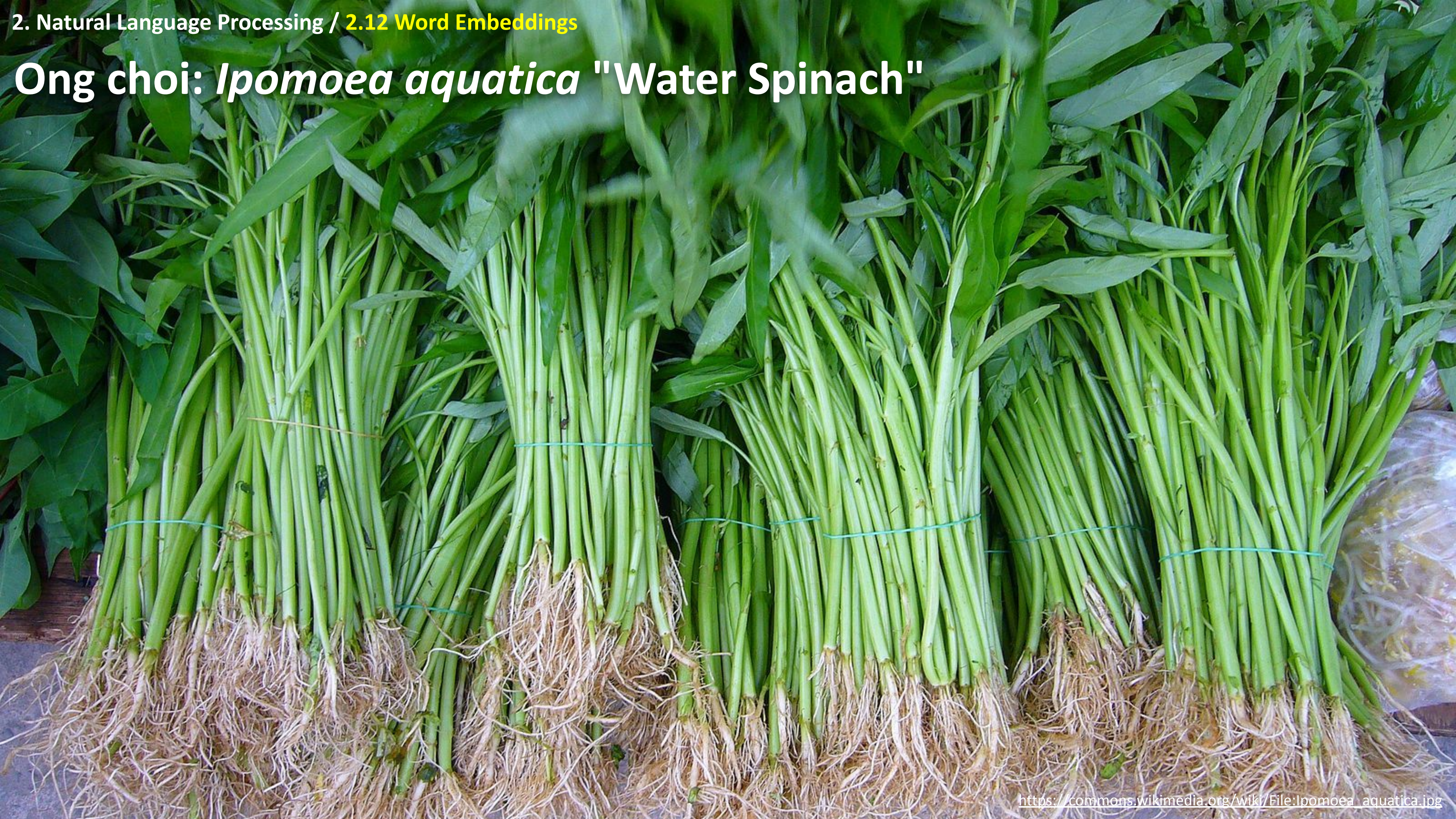
(J.R. Firth, 1957)

Zellig S. Harris (1954) Distributional Structure, WORD, 10:2-3, 146-162, DOI: 10.1080/00437956.1954.11659520
J.R. Firth (1957) A synopsis of linguistic theory, Studies in linguistic analysis, Blackwell, Oxford

What Does “Ong Choi” Mean?

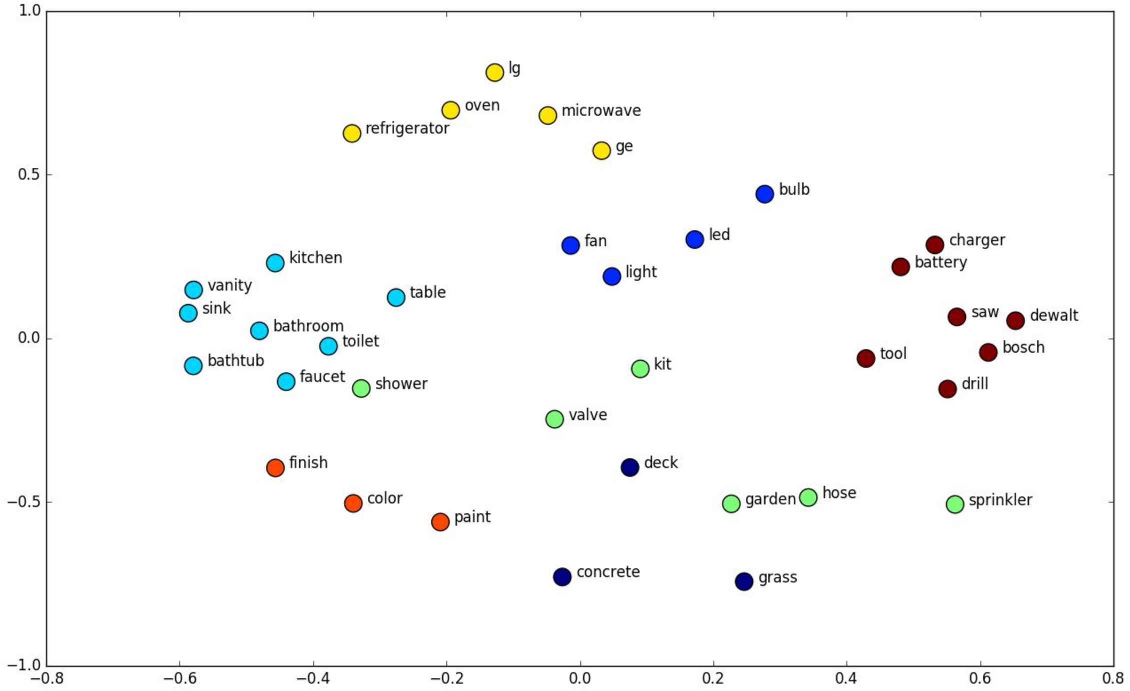
- Suppose you see these sentences:
  - **Ong choi** is delicious sautéed with garlic.
  - **Ong choi** is superb over rice.
  - **Ong choi** leaves with salty sauces...
- And you've also seen these:
  - **...spinach** sautéed with garlic over rice.
  - **Chard** stems and leaves are delicious.
  - **Collard greens** and other salty leafy greens...
- Conclusion:
  - Ong choi is a **leafy green** like **spinach**, **chard**, or **collard greens**.

Ong choi: *Ipomoea aquatica* "Water Spinach"



https://commons.wikimedia.org/wiki/File:Ipomoea_aquatica.jpg

We Define a Word as a Vector



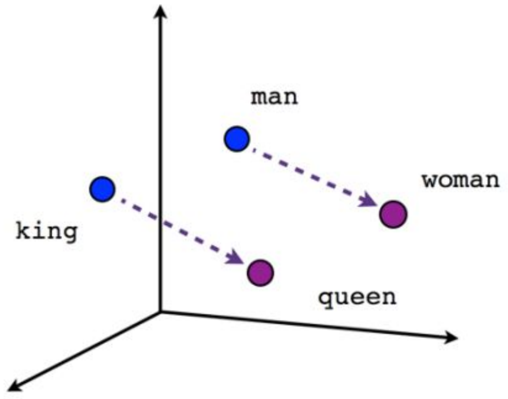
- Combines **distributional intention** (statistical language model) and **vector intuition.**
- Semantically similar words are nearby in a vector space.
- Called an "**embedding**" because it's embedded into a vector space.
- The standard way to represent meaning in NLP.

Sparse vs Dense Vectors

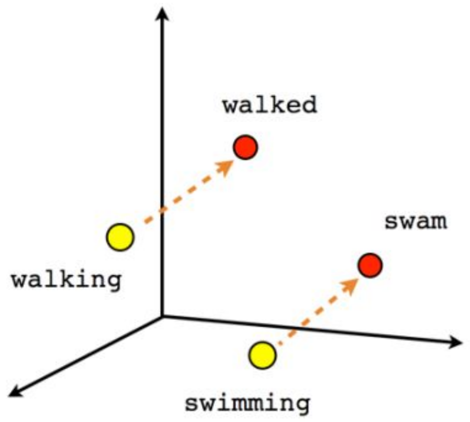
- **tf-idf**
  - THE standard in information retrieval.
  - Words are represented by a simple function of the counts of nearby words.
  - **Long** vectors (length |V|= 20,000 to 50,000)
  - **Sparse** vectors (almost all elements are zero)

- **word2vec** (Mikolov et al, https://code.google.com/archive/p/word2vec/)
  - Representation is created by training a classifier to distinguish nearby and far-away words.
  - **Short** vectors (length 50-1000)
  - **Dense** vectors (most elements are non-zero)

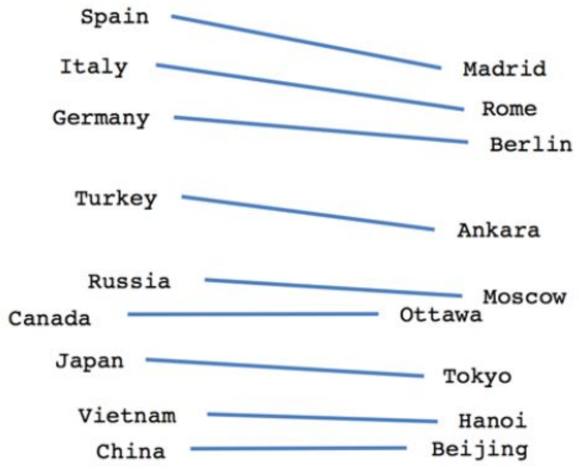
word2vec Properties



Male-Female



Verb tense



Country-Capital

To be continued… [Chap 4. Machine Learning]

Information Service Engineering

Lecture 5: Natural Language Processing (4)

2.0 What is Natural Language Processing?
2.1 NLP and Basic Linguistic Knowledge
2.2 Morphology
2.3 NLP Applications
2.4 NLP Techniques
2.5 NLP Challenges
2.6 Evaluation, Precision and Recall
2.7 Regular Expressions
2.8 Finite State Automata
2.9 Tokenization
2.10 Language Model and N-Grams
2.11 Part-of-Speech Tagging
2.12 Word Embeddings

2. Natural Language Processing - 4

Bibliography

- Google Research Blog, *All Our N-gram are Belong to You*, August 03, 2006
- The Penn TreeBank Project, June 14, 1997
- POS Tagging (State-of-the-Art), April, 24, 2021
- D. Jurafsky, J. H. Martin, *Speech and Language Processing, 3rd ed.*, 2009,
  - Section 3, *N-gram Language Models*, 3.1 - 3.4
  - Section 8, *Part-of-Speech Tagging*, 8.1 - 8.5.1
    (please note that this refers to the 3rd ed.)
- For deeper insights into word2vec:
  - Word2vec @ Google Code Archive, https://code.google.com/archive/p/word2vec/
  - Mikolov, Tomas; et al. (2013). "*Efficient Estimation of Word Representations in Vector Space*". arXiv:1301.3781

2. Natural Language Processing - 4

Syllabus Questions

- What is the purpose of a language model?
- How can the probability of occurrence of words and word sequences be determined?
- What are the important factors for the quality of a language model?
- How do we evaluate the quality of a language model?
- What does the average branching factor indicate?
- What is the difference of „word form“ and „lemma“?
- What is the Markov Assumption used for in a language model?
- What is the Maximum Likelihood Estimation used for in a language model?
- How can unknown words be treated in a language model?
- What is POS-tagging?
- What is the difference between proper nouns, count nouns, and mass nouns?
- What is POS-tagging used for and why is POS-tagging important?
- How does the baseline method for POS-tagging work?
- How does (in principle) stochastic POS-tagging (Hidden Markov Model) work?
- How can we represent words in the computer?
- Explain the problems related to the different word representations
- What are word embeddings?