

# Basic Text Processing

# Regular Expressions



# Regular expressions

- A formal language for specifying text strings
- How can we search for any of these?
  - woodchuck
  - woodchucks
  - Woodchuck
  - Woodchucks





# Regular Expressions: Disjunctions

- Letters inside square brackets []

Pattern	Matches
[ wW ]oodchuck	Woodchuck, woodchuck
[ 1234567890 ]	Any digit

- Ranges [ A-Z ]

Pattern	Matches	
[ A-Z ]	An upper case letter	Drenched Blossoms
[ a-z ]	A lower case letter	my beans were impatient
[ 0-9 ]	A single digit	Chapter 1: Down the Rabbit Hole



# Regular Expressions: Negation in Disjunction

- Negations [ ^Ss ]
  - Carat means negation only when first in []

Pattern	Matches	
[ ^A-Z ]	Not an upper case letter	O <u>y</u> fn pripetchik
[ ^Ss ]	Neither 'S' nor 's'	I have no exquisite reason"
[ ^e^ ]	Neither e nor ^	Look <u>h</u> ere
a^b	The pattern a carat b	Look up <u>a^b</u> now



# Regular Expressions: More Disjunction

- Woodchucks is another name for groundhog!
- The pipe | for disjunction

Pattern	Matches
groundhog woodchuck	
yours mine	yours mine
a b c	= [abc]
[ gG ]roundhog  [ Ww ]oodchuck	



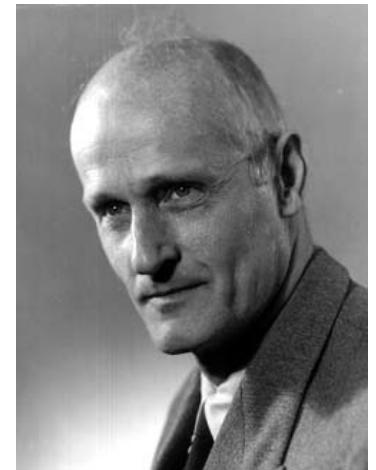
Photo D. Fletcher



# Regular Expressions: ? \* + .

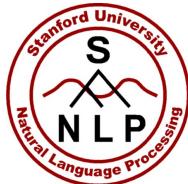
Pattern	Matches
colou?r	Optional previous char
oo*h!	0 or more of previous char
o+h!	1 or more of previous char
baa+	
beg.n	

The table shows regular expression patterns and their matches. The first four rows show standard patterns with their meanings and examples. The last two rows are blank.



Stephen C Kleene

Kleene \*, Kleene +



# Regular Expressions: Anchors $\wedge$ $\$$

Pattern	Matches
$\wedge [A-Z]$	<u>P</u> alo Alto
$\wedge [^A-Za-z]$	<u>1</u> " <u>Hello</u> "
$\backslash . \$$	The end <u>.</u>
$. \$$	The end <u>?</u> The end <u>!</u>



## Example

- Find me all instances of the word “the” in a text.

the

Misses capitalized examples

[tT]he

Incorrectly returns other or theology

[^a-zA-Z][tT]he[^a-zA-Z]



## Errors

- The process we just went through was based on **fixing two kinds of errors**
  - Matching strings that we should not have matched (**there, then, other**)
    - **False positives (Type I)**
  - Not matching things that we should have matched (**The**)
    - **False negatives (Type II)**



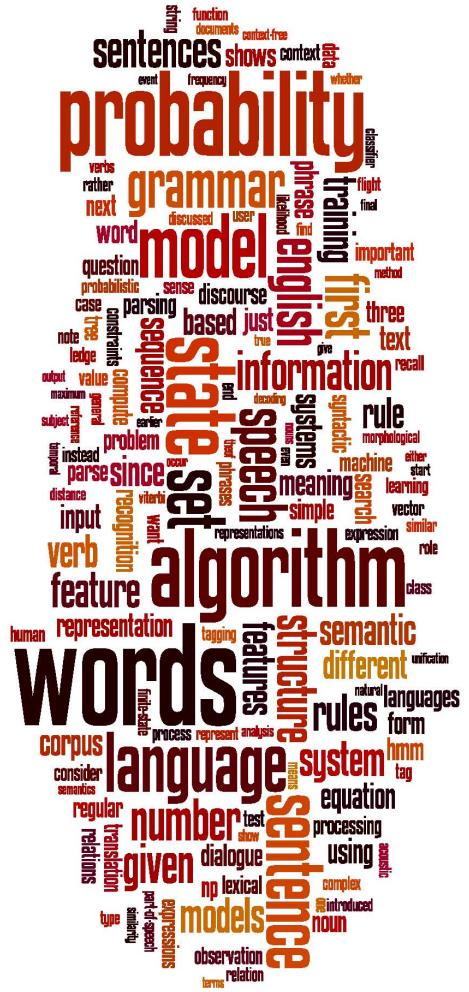
## Errors cont.

- In NLP we are always dealing with these kinds of errors.
- Reducing the error rate for an application often involves two antagonistic efforts:
  - Increasing accuracy or precision (minimizing false positives)
  - Increasing coverage or recall (minimizing false negatives).



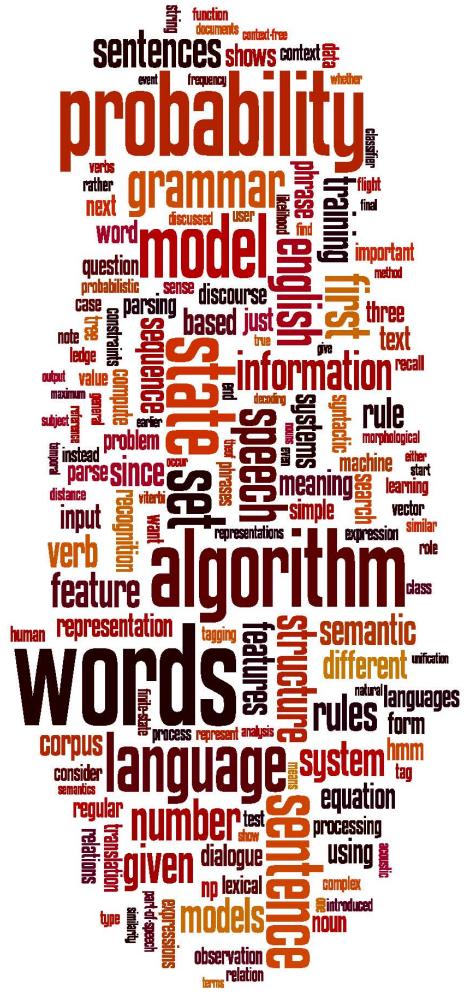
# Summary

- Regular expressions play a surprisingly large role
  - Sophisticated sequences of regular expressions are often the first model for any text processing task
- For many hard tasks, we use machine learning classifiers
  - But regular expressions are used as features in the classifiers
  - Can be very useful in capturing generalizations



# Basic Text Processing

# Regular Expressions



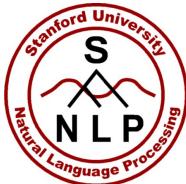
# Basic Text Processing

# Word tokenization



## Text Normalization

- Every NLP task needs to do text normalization:
  1. Segmenting/tokenizing words in running text
  2. Normalizing word formats
  3. Segmenting sentences in running text



## How many words?

- I do uh main- mainly business data processing
  - Fragments, filled pauses
- Seuss's **cat** in the hat is different from other **cats!**
  - **Lemma:** same stem, part of speech, rough word sense
    - **cat** and **cats** = same lemma
  - **Wordform:** the full inflected surface form
    - **cat** and **cats** = different wordforms



# How many words?

they lay back on the San Francisco grass and looked at the stars and their

- **Type:** an element of the vocabulary.
- **Token:** an instance of that type in running text.
- How many?
  - 15 tokens (or 14)
  - 13 types (or 12) (or 11?)



# How many words?

$N$  = number of tokens

$V$  = vocabulary = set of types

$|V|$  is the size of the vocabulary

Church and Gale (1990):  $|V| > O(N^{1/2})$

	Tokens = $N$	Types = $ V $
Switchboard phone conversations	2.4 million	20 thousand
Shakespeare	884,000	31 thousand
Google N-grams	1 trillion	13 million



# Simple Tokenization in UNIX

- (Inspired by Ken Church's UNIX for Poets.)
- Given a text file, output the word tokens and their frequencies

```
tr -sc 'A-Za-z' '\n' < shakes.txt      Change all non-alpha to newlines
| sort          Sort in alphabetical order
| uniq -c       Merge and count each type
```

1945	A	25	Aaron
72	AARON	6	Abate
19	ABBESSION	1	Abates
5	ABBOT	5	Abbess
...	...	6	Abbey
		3	Abbot
		....	...

Dan Jurafsky



# The first step: tokenizing

```
tr -sc 'A-Za-z' '\n' < shakes.txt | head
```

THE  
SONNETS  
by  
William  
Shakespeare  
From  
fairest  
creatures  
We  
...



## The second step: sorting

```
tr -sc 'A-Za-z' '\n' < shakes.txt | sort | head
```

A

A

A

A

A

A

A

A

A

...



## More counting

- Merging upper and lower case

```
tr 'A-Z' 'a-z' < shakes.txt | tr -sc 'A-Za-z' '\n' | sort | uniq -c
```

- Sorting the counts

```
tr 'A-Z' 'a-z' < shakes.txt | tr -sc 'A-Za-z' '\n' | sort | uniq -c | sort -n -r
```

23243	the
22225	i
18618	and
16339	to
15687	of
12780	a
12163	you
10839	my
10005	in
8954	d

What happened here?



## Issues in Tokenization

- Finland's capital → Finland Finlands Finland's ?
- what're, I'm, isn't → What are, I am, is not
- Hewlett-Packard → Hewlett Packard ?
- state-of-the-art → state of the art ?
- Lowercase → lower-case lowercase lower case ?
- San Francisco → one token or two?
- m.p.h., PhD. → ??



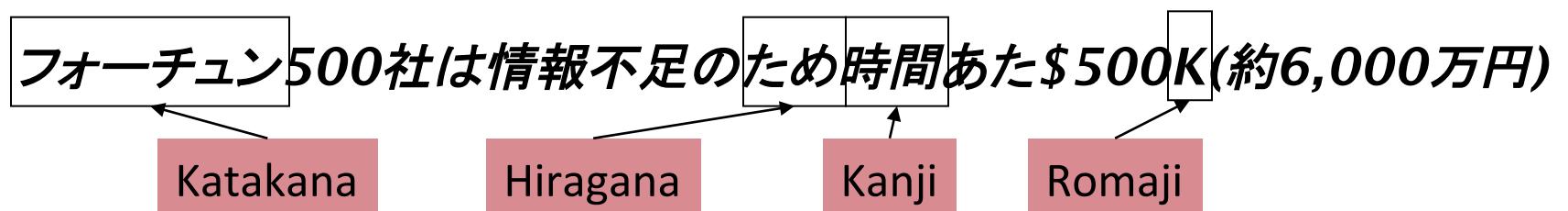
# Tokenization: language issues

- French
  - *L'ensemble* → one token or two?
    - *L* ? *L'* ? *Le* ?
    - Want *l'ensemble* to match with *un ensemble*
- German noun compounds are not segmented
  - *Lebensversicherungsgesellschaftsangestellter*
  - ‘life insurance company employee’
  - German information retrieval needs **compound splitter**



# Tokenization: language issues

- Chinese and Japanese no spaces between words:
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
  - Sharapova now lives in US southeastern Florida
- Further complicated in Japanese, with multiple alphabets intermingled
  - Dates/amounts in multiple formats



End-user can express query entirely in hiragana!



# Word Tokenization in Chinese

- Also called **Word Segmentation**
- Chinese words are composed of characters
  - Characters are generally 1 syllable and 1 morpheme.
  - Average word is 2.4 characters long.
- Standard baseline segmentation algorithm:
  - Maximum Matching (also called Greedy)

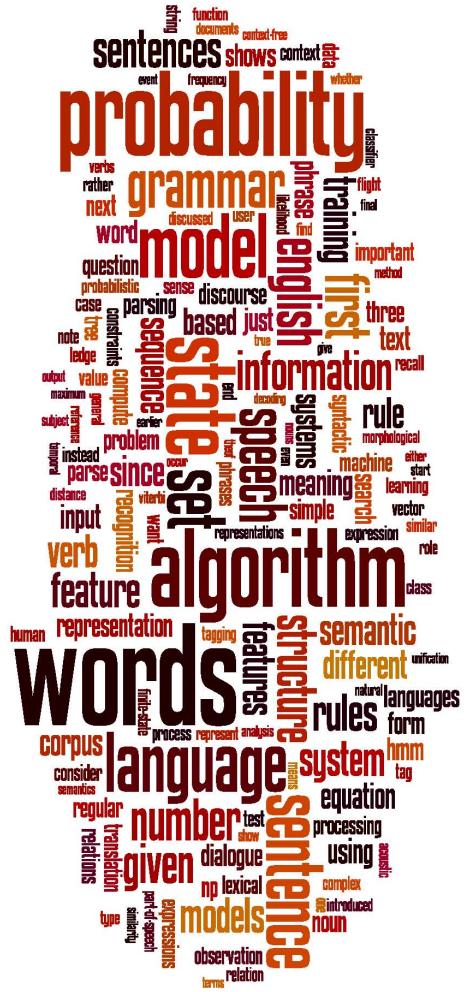


# Maximum Matching Word Segmentation Algorithm

- Given a wordlist of Chinese, and a string.
- 1) Start a pointer at the beginning of the string
  - 2) Find the longest word in dictionary that matches the string starting at pointer
  - 3) Move the pointer over the word in string
  - 4) Go to 2

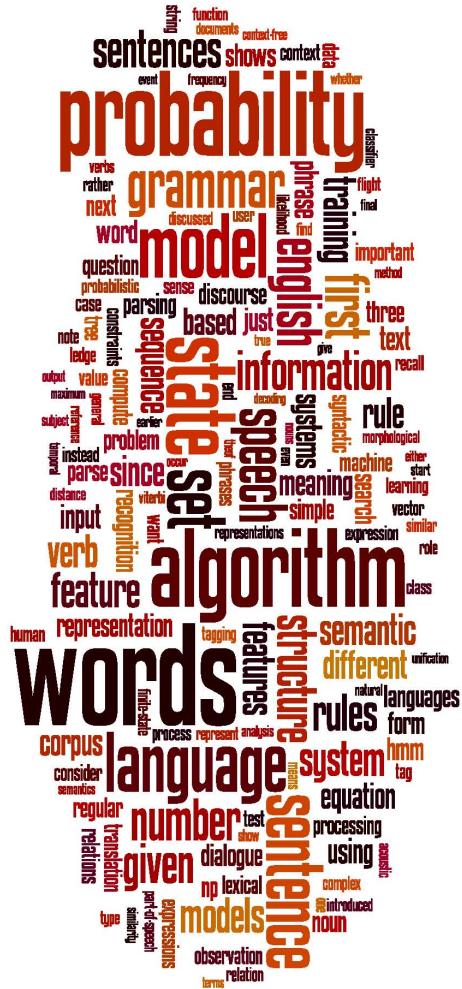


# Max-match segmentation illustration



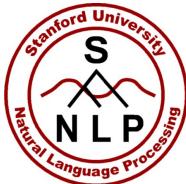
# Basic Text Processing

# Word tokenization



# Basic Text Processing

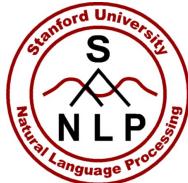
# Word Normalization and Stemming



# Normalization

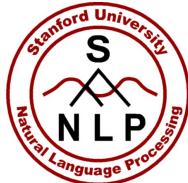
- Need to “normalize” terms
  - Information Retrieval: indexed text & query terms must have same form.
    - We want to match ***U.S.A.*** and ***USA***
- We implicitly define equivalence classes of terms
  - e.g., deleting periods in a term
- Alternative: asymmetric expansion:

• Enter: <b><i>window</i></b>	Search: <b><i>window, windows</i></b>
• Enter: <b><i>windows</i></b>	Search: <b><i>Windows, windows, window</i></b>
• Enter: <b><i>Windows</i></b>	Search: <b><i>Windows</i></b>
- Potentially more powerful, but less efficient



## Case folding

- Applications like IR: reduce all letters to lower case
  - Since users tend to use lower case
  - Possible exception: upper case in mid-sentence?
    - e.g., *General Motors*
    - *Fed* vs. *fed*
    - *SAIL* vs. *sail*
- For sentiment analysis, MT, Information extraction
  - Case is helpful (*US* versus *us* is important)



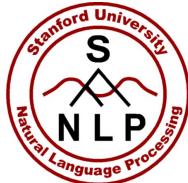
# Lemmatization

- Reduce inflections or variant forms to base form
  - *am, are, is* → *be*
  - *car, cars, car's, cars'* → *car*
- *the boy's cars are different colors* → *the boy car be different color*
- Lemmatization: have to find correct dictionary headword form
- Machine translation
  - Spanish **quiero** ('I want'), **quieres** ('you want') same lemma as **querer** 'want'



# Morphology

- **Morphemes:**
  - The small meaningful units that make up words
  - **Stems:** The core meaning-bearing units
  - **Affixes:** Bits and pieces that adhere to stems
    - Often with grammatical functions



# Stemming

- Reduce terms to their stems in information retrieval
- *Stemming* is crude chopping of affixes
  - language dependent
  - e.g., *automate(s)*, *automatic*, *automation* all reduced to *automat*.

for example compressed  
and compression are both  
accepted as equivalent to  
compress.



for exampl compress and  
compress ar both accept  
as equival to compress



# Porter's algorithm

## The most common English stemmer

### Step 1a

sses	$\rightarrow$	ss	caresses	$\rightarrow$	caress
ies	$\rightarrow$	i	ponies	$\rightarrow$	poni
ss	$\rightarrow$	ss	caress	$\rightarrow$	caress
s	$\rightarrow$	$\emptyset$	cats	$\rightarrow$	cat

### Step 1b

(*v*)ing	$\rightarrow$	$\emptyset$	walking	$\rightarrow$	walk
			sing	$\rightarrow$	sing
(*v*)ed	$\rightarrow$	$\emptyset$	plastered	$\rightarrow$	plaster
...					

### Step 2 (for long stems)

ational	$\rightarrow$	ate	relational	$\rightarrow$	relate
izer	$\rightarrow$	ize	digitizer	$\rightarrow$	digitize
ator	$\rightarrow$	ate	operator	$\rightarrow$	operate
...					

### Step 3 (for longer stems)

al	$\rightarrow$	$\emptyset$	revival	$\rightarrow$	reviv
able	$\rightarrow$	$\emptyset$	adjustable	$\rightarrow$	adjust
ate	$\rightarrow$	$\emptyset$	activate	$\rightarrow$	activ
...					



# Viewing morphology in a corpus

## Why only strip –ing if there is a vowel?

( \*v\* )ing → Ø walking → walk  
sing → sing



# Viewing morphology in a corpus

## Why only strip –ing if there is a vowel?

(*\*v\**)ing → Ø walking → walk  
sing → sing

```
tr -sc 'A-Za-z' '\n' < shakes.txt | grep 'ing$' | sort | uniq -c | sort -nr
```

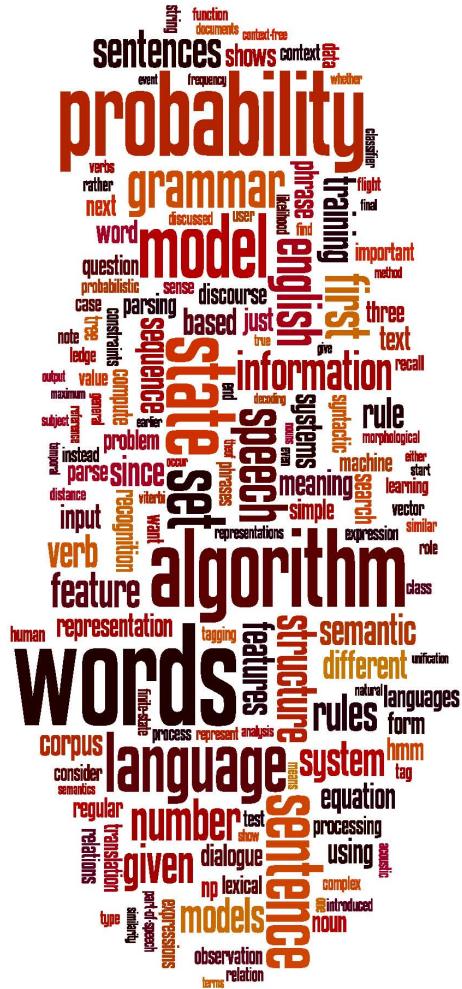
1312 King	548 being
548 being	541 nothing
541 nothing	152 something
388 king	145 coming
375 bring	130 morning
358 thing	122 having
307 ring	120 living
152 something	117 loving
145 coming	116 Being
130 morning	102 going

```
tr -sc 'A-Za-z' '\n' < shakes.txt | grep '[aeiou].*ing$' | sort | uniq -c | sort -nr
```



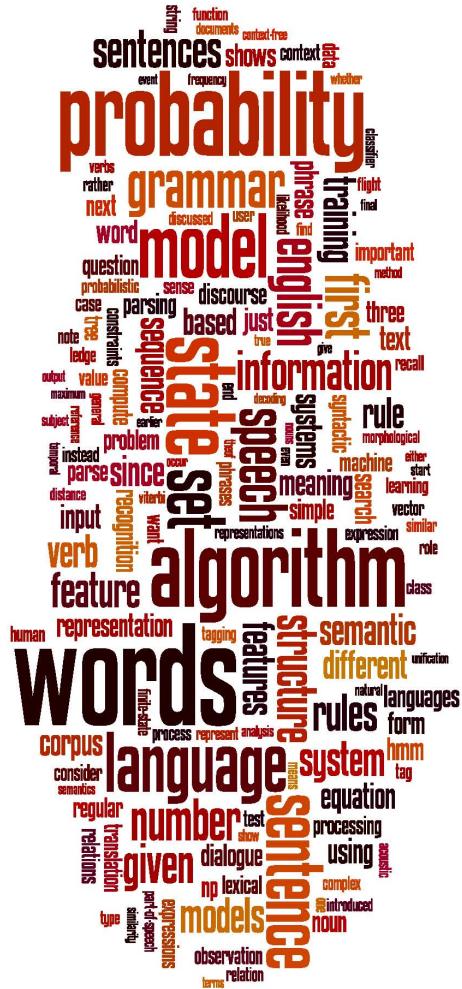
# Dealing with complex morphology is sometimes necessary

- Some languages require complex morpheme segmentation
  - Turkish
  - **Uygarlastiramadiklarimizdanmissinizcasina**
  - '(behaving) as if you are among those whom we could not civilize'
  - **Uygar** 'civilized' + **las** 'become'
    - + **tır** 'cause' + **ama** 'not able'
    - + **dik** 'past' + **lar** 'plural'
    - + **imiz** 'p1pl' + **dan** 'abl'
    - + **mis** 'past' + **siniz** '2pl' + **casina** 'as if'



# Basic Text Processing

# Word Normalization and Stemming



# Basic Text Processing

# Sentence Segmentation and Decision Trees

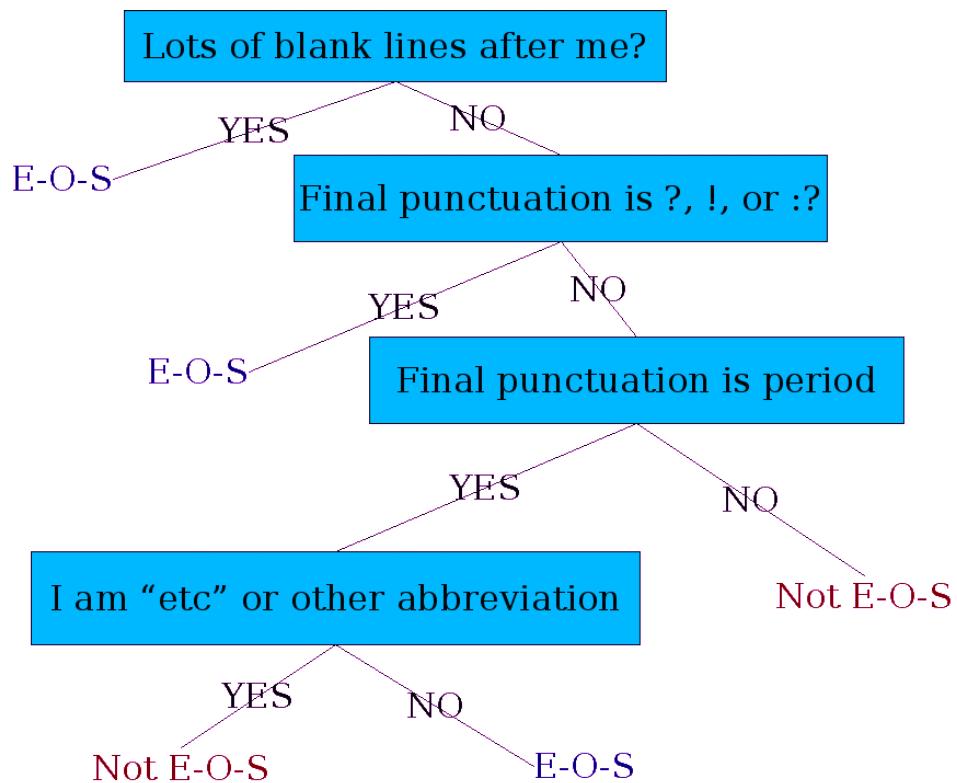


# Sentence Segmentation

- !, ? are relatively unambiguous
- Period “.” is quite ambiguous
  - Sentence boundary
  - Abbreviations like Inc. or Dr.
  - Numbers like .02% or 4.3
- Build a binary classifier
  - Looks at a “.”
  - Decides EndOfSentence/NotEndOfSentence
  - Classifiers: hand-written rules, regular expressions, or machine-learning



# Determining if a word is end-of-sentence: a Decision Tree





## More sophisticated decision tree features

- Case of word with “.”: Upper, Lower, Cap, Number
- Case of word after “.”: Upper, Lower, Cap, Number
- Numeric features
  - Length of word with “.”
  - Probability(word with “.” occurs at end-of-s)
  - Probability(word after “.” occurs at beginning-of-s)



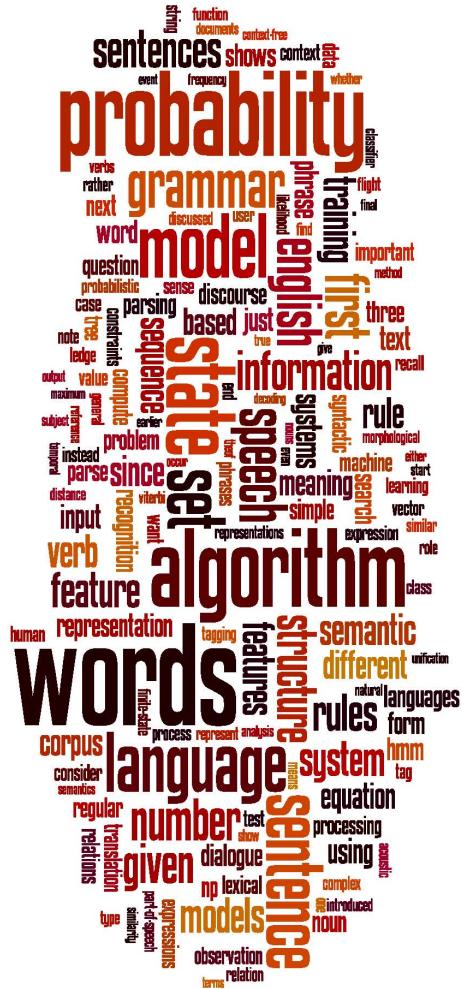
# Implementing Decision Trees

- A decision tree is just an if-then-else statement
- The interesting research is choosing the features
- Setting up the structure is often too hard to do by hand
  - Hand-building only possible for very simple features, domains
    - For numeric features, it's too hard to pick each threshold
  - Instead, structure usually learned by machine learning from a training corpus



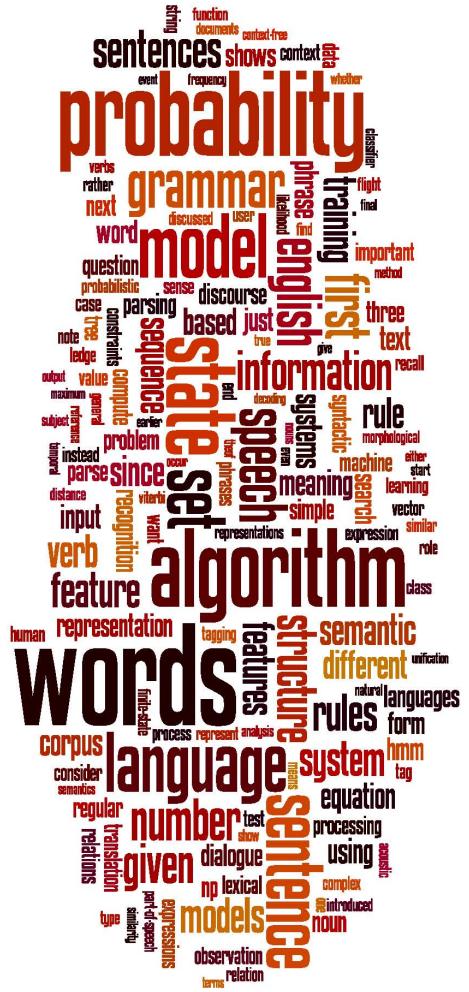
# Decision Trees and other classifiers

- We can think of the questions in a decision tree
- As features that could be exploited by any kind of classifier
  - Logistic regression
  - SVM
  - Neural Nets
  - etc.



# Basic Text Processing

# Sentence Segmentation and Decision Trees



# Language Modeling

# Introduction to N-grams



# Probabilistic Language Models

- Today's goal: assign a probability to a sentence
  - Machine Translation:
    - $P(\text{high winds tonite}) > P(\text{large winds tonite})$
  - Spell Correction
    - The office is about fifteen **minuets** from my house
      - $P(\text{about fifteen minutes from}) > P(\text{about fifteen minuets from})$
  - Speech Recognition
    - $P(\text{I saw a van}) \gg P(\text{eyes awe of an})$
  - + Summarization, question-answering, etc., etc.!!

Why?



# Probabilistic Language Modeling

- Goal: compute the probability of a sentence or sequence of words:

$$P(W) = P(w_1, w_2, w_3, w_4, w_5 \dots w_n)$$

- Related task: probability of an upcoming word:

$$P(w_5 | w_1, w_2, w_3, w_4)$$

- A model that computes either of these:

$P(W)$    or    $P(w_n | w_1, w_2 \dots w_{n-1})$       is called a **language model**.

- Better: **the grammar**      But **language model** or **LM** is standard



## How to compute $P(W)$

- How to compute this joint probability:
  - $P(\text{its, water, is, so, transparent, that})$
- Intuition: let's rely on the Chain Rule of Probability



## Reminder: The Chain Rule

- Recall the definition of conditional probabilities

Rewriting:

- More variables:

$$P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$$

- The Chain Rule in General

$$P(x_1, x_2, x_3, \dots, x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)\dots P(x_n|x_1, \dots, x_{n-1})$$



# The Chain Rule applied to compute joint probability of words in sentence

$$P(w_1 w_2 \dots w_n) = \prod_i P(w_i | w_1 w_2 \dots w_{i-1})$$

$P(\text{"its water is so transparent"}) =$

$P(\text{its}) \times P(\text{water} | \text{its}) \times P(\text{is} | \text{its water})$

$\times P(\text{so} | \text{its water is}) \times P(\text{transparent} | \text{its water is so})$



## How to estimate these probabilities

- Could we just count and divide?

$P(\text{the} \mid \text{its water is so transparent that}) =$

$\frac{\text{Count}(\text{its water is so transparent that the})}{\text{Count}(\text{its water is so transparent that})}$

- No! Too many possible sentences!
- We'll never see enough data for estimating these

Dan Jurafsky



# Markov Assumption

- Simplifying assumption:



Andrei Markov

$$P(\text{the} \mid \text{its water is so transparent that}) \approx P(\text{the} \mid \text{that})$$

- Or maybe

$$P(\text{the} \mid \text{its water is so transparent that}) \approx P(\text{the} \mid \text{transparent that})$$

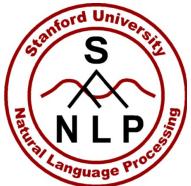


# Markov Assumption

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i | w_{i-k} \dots w_{i-1})$$

- In other words, we approximate each component in the product

$$P(w_i | w_1 w_2 \dots w_{i-1}) \approx P(w_i | w_{i-k} \dots w_{i-1})$$



## Simplest case: Unigram model

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i)$$

Some automatically generated sentences from a unigram model

fifth, an, of, futures, the, an, incorporated, a,  
a, the, inflation, most, dollars, quarter, in, is,  
mass

thrift, did, eighty, said, hard, 'm, july, bullish

that, or, limited, the



## Bigram model

- Condition on the previous word:

$$P(w_i | w_1 w_2 \dots w_{i-1}) \approx P(w_i | w_{i-1})$$

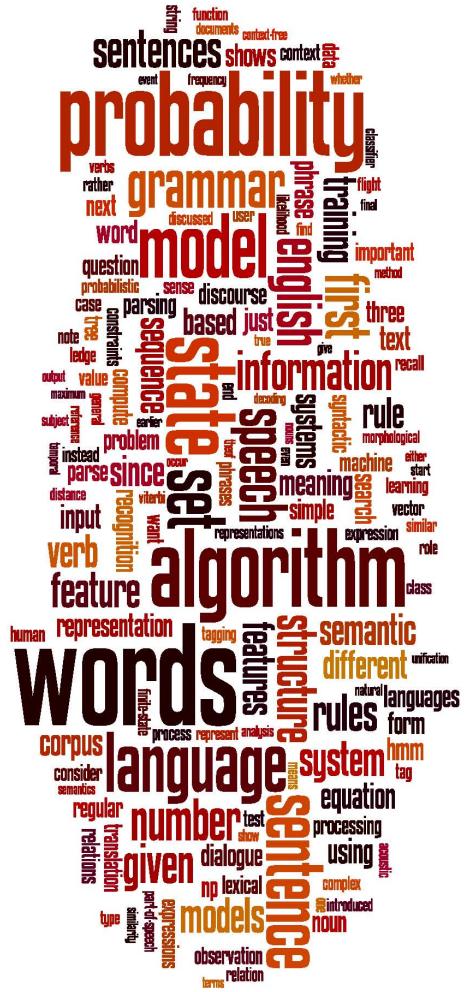
texaco, rose, one, in, this, issue, is, pursuing, growth, in,  
a, boiler, house, said, mr., gurria, mexico, 's, motion,  
control, proposal, without, permission, from, five, hundred,  
fifty, five, yen

outside, new, car, parking, lot, of, the, agreement, reached  
this, would, be, a, record, november



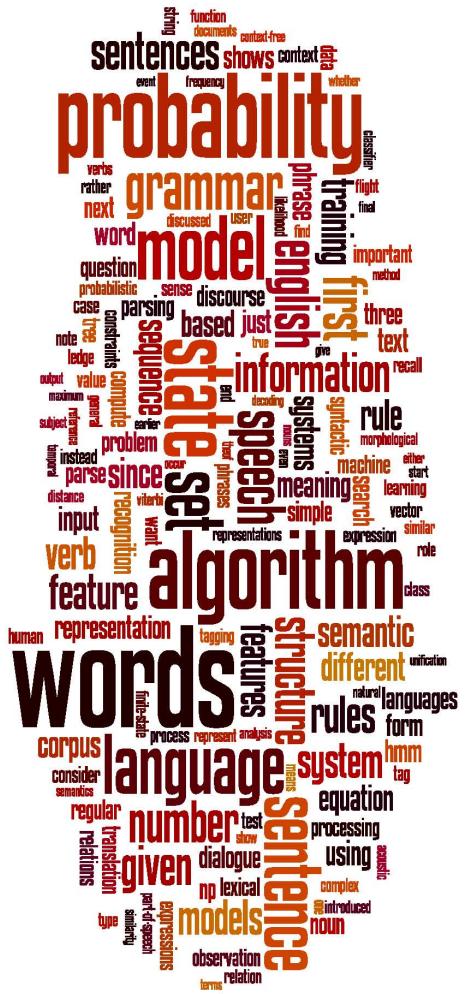
## N-gram models

- We can extend to trigrams, 4-grams, 5-grams
  - In general this is an insufficient model of language
    - because language has **long-distance dependencies**:
- “The computer which I had just put into the machine room on the fifth floor crashed.”
- But we can often get away with N-gram models



# Language Modeling

# Introduction to N-grams



# Language Modeling

Estimating N-gram  
Probabilities



## Estimating bigram probabilities

- The Maximum Likelihood Estimate

$$P(w_i | w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i)}{\text{count}(w_{i-1})}$$

$$P(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$



## An example

$$P(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

< s > I am Sam < /s >

< s > Sam I am < /s >

< s > I do not like green eggs and ham < /s >

$$P(\text{I} | \text{<s>}) = \frac{2}{3} = .67$$

$$P(\text{</s>} | \text{Sam}) = \frac{1}{2} = 0.5$$

$$P(\text{Sam} | \text{<s>}) = \frac{1}{3} = .33$$

$$P(\text{Sam} | \text{am}) = \frac{1}{2} = .5$$

$$P(\text{am} | \text{I}) = \frac{2}{3} = .67$$

$$P(\text{do} | \text{I}) = \frac{1}{3} = .33$$



## More examples: Berkeley Restaurant Project sentences

- can you tell me about any good cantonese restaurants close by
- mid priced thai food is what i'm looking for
- tell me about chez panisse
- can you give me a listing of the kinds of food that are available
- i'm looking for a good place to eat breakfast
- when is caffe venezia open during the day



# Raw bigram counts

- Out of 9222 sentences

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0



# Raw bigram probabilities

- Normalize by unigrams:

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

- Result:

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0



## Bigram estimates of sentence probabilities

$P(< s > | \text{I want english food } </ s >) =$

$$P(\text{I} | < s >)$$

$$\times P(\text{want} | \text{I})$$

$$\times P(\text{english} | \text{want})$$

$$\times P(\text{food} | \text{english})$$

$$\times P(</ s > | \text{food})$$

$$= .000031$$



## What kinds of knowledge?

- $P(\text{english} \mid \text{want}) = .0011$
- $P(\text{chinese} \mid \text{want}) = .0065$
- $P(\text{to} \mid \text{want}) = .66$
- $P(\text{eat} \mid \text{to}) = .28$
- $P(\text{food} \mid \text{to}) = 0$
- $P(\text{want} \mid \text{spend}) = 0$
- $P(\text{i} \mid \langle s \rangle) = .25$



## Practical Issues

- We do everything in log space
  - Avoid underflow
  - (also adding is faster than multiplying)

$$\log(p_1 \times p_2 \times p_3 \times p_4) = \log p_1 + \log p_2 + \log p_3 + \log p_4$$

Dan Jurafsky



## Language Modeling Toolkits

- SRILM
  - <http://www.speech.sri.com/projects/srilm/>

Dan Jurafsky



# Google N-Gram Release, August 2006

AUG

3

## All Our N-gram are Belong to You

Posted by Alex Franz and Thorsten Brants, Google Machine Translation Team

Here at Google Research we have been using word [n-gram models](#) for a variety of R&D projects,

...

That's why we decided to share this enormous dataset with everyone. We processed 1,024,908,267,229 words of running text and are publishing the counts for all 1,176,470,663 five-word sequences that appear at least 40 times. There are 13,588,391 unique words, after discarding words that appear less than 200 times.



# Google N-Gram Release

- serve as the incoming 92
- serve as the incubator 99
- serve as the independent 794
- serve as the index 223
- serve as the indication 72
- serve as the indicator 120
- serve as the indicators 45
- serve as the indispensable 111
- serve as the indispensible 40
- serve as the individual 234

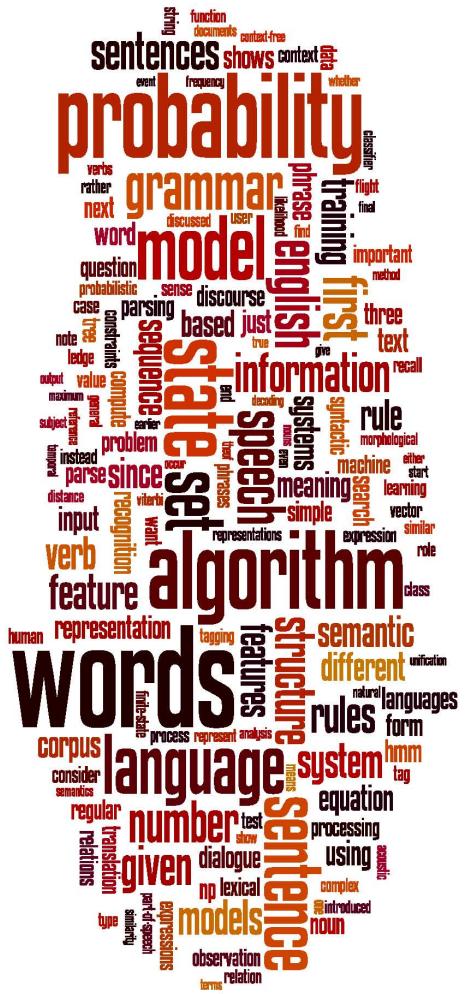
<http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html>

Dan Jurafsky



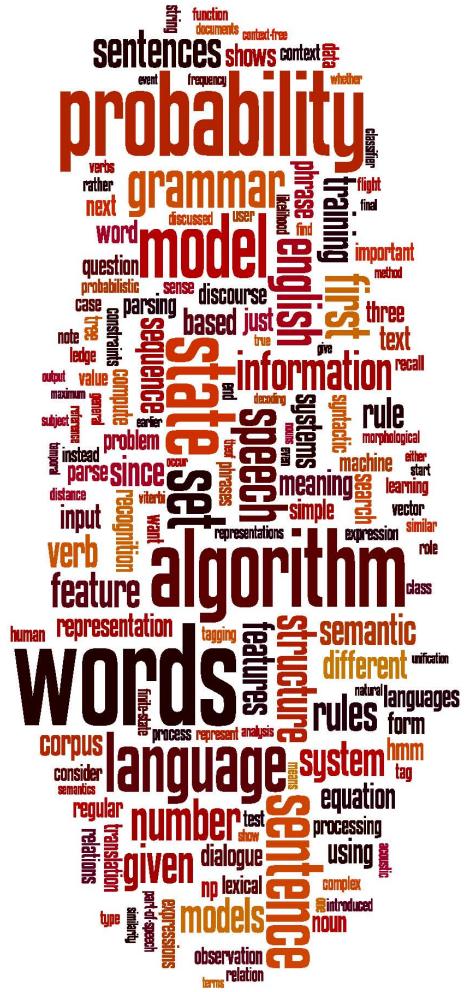
## Google Book N-grams

- <http://ngrams.googlecode.com/>



# Language Modeling

Estimating N-gram  
Probabilities



# Language Modeling

# Evaluation and Perplexity



# Evaluation: How good is our model?

- Does our language model prefer good sentences to bad ones?
  - Assign higher probability to “real” or “frequently observed” sentences
    - Than “ungrammatical” or “rarely observed” sentences?
- We train parameters of our model on a **training set**.
- We test the model’s performance on data we haven’t seen.
  - A **test set** is an unseen dataset that is different from our training set, totally unused.
  - An **evaluation metric** tells us how well our model does on the test set.



## Extrinsic evaluation of N-gram models

- Best evaluation for comparing models A and B
  - Put each model in a task
    - spelling corrector, speech recognizer, MT system
  - Run the task, get an accuracy for A and for B
    - How many misspelled words corrected properly
    - How many words translated correctly
  - Compare accuracy for A and B



# Difficulty of extrinsic (in-vivo) evaluation of N-gram models

- Extrinsic evaluation
  - Time-consuming; can take days or weeks
- So
  - Sometimes use **intrinsic** evaluation: **perplexity**
  - Bad approximation
    - unless the test data looks **just** like the training data
    - So **generally only useful in pilot experiments**
  - But is helpful to think about.



# Intuition of Perplexity

- The Shannon Game:
  - How well can we predict the next word?

I always order pizza with cheese and \_\_\_\_\_

The 33<sup>rd</sup> President of the US was \_\_\_\_\_

I saw a \_\_\_\_\_
  - Unigrams are terrible at this game. (Why?)
- A better model of a text
  - is one which assigns a higher probability to the word that actually occurs

mushrooms 0.1  
pepperoni 0.1  
anchovies 0.01  
....  
fried rice 0.0001  
....  
and 1e-100



# Perplexity

The best language model is one that best predicts an unseen test set

- Gives the highest  $P(\text{sentence})$

Perplexity is the inverse probability of the test set, normalized by the number of words:

Chain rule:

For bigrams:

$$PP(W) = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

$$PP(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_1 \dots w_{i-1})}}$$

$$PP(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_{i-1})}}$$

**Minimizing perplexity is the same as maximizing probability**



# The Shannon Game intuition for perplexity

- From Josh Goodman
- How hard is the task of recognizing digits '0,1,2,3,4,5,6,7,8,9'
  - Perplexity 10
- How hard is recognizing (30,000) names at Microsoft.
  - Perplexity = 30,000
- If a system has to recognize
  - Operator (1 in 4)
  - Sales (1 in 4)
  - Technical Support (1 in 4)
  - 30,000 names (1 in 120,000 each)
  - Perplexity is 53
- Perplexity is weighted equivalent branching factor



## Perplexity as branching factor

- Let's suppose a sentence consisting of random digits
- What is the perplexity of this sentence according to a model that assign  $P=1/10$  to each digit?

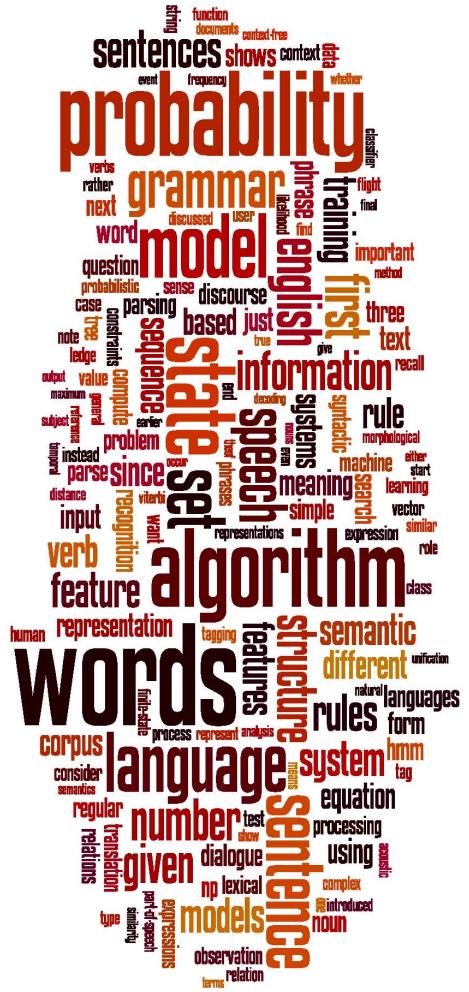
$$\begin{aligned} \text{PP}(W) &= P(w_1 w_2 \dots w_N)^{-\frac{1}{N}} \\ &= \left(\frac{1}{10}\right)^{-\frac{1}{N}} \\ &= \frac{1}{10}^{-1} \\ &= 10 \end{aligned}$$



## Lower perplexity = better model

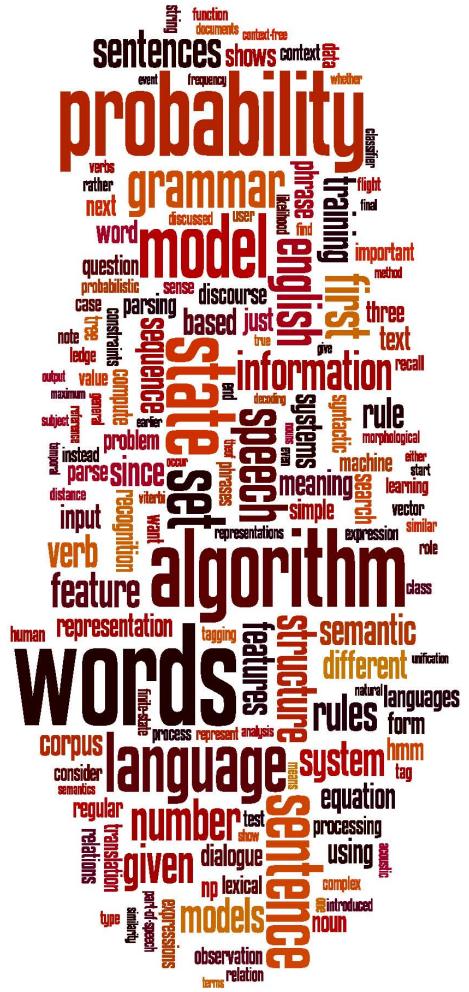
- Training 38 million words, test 1.5 million words, WSJ

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109



# Language Modeling

# Evaluation and Perplexity



# Language Modeling

# Generalization and zeros



# The Shannon Visualization Method

- Choose a random bigram ( $\langle s \rangle, w$ ) according to its probability
- Now choose a random bigram ( $w, x$ ) according to its probability
- And so on until we choose  $\langle /s \rangle$
- Then string the words together

$\langle s \rangle$  I  
I want  
want to  
to eat  
eat Chinese  
Chinese food  
food  $\langle /s \rangle$

I want to eat Chinese food



# Approximating Shakespeare

## Unigram

To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have  
Every enter now severally so, let  
Hill he late speaks; or! a more to leg less first you enter  
Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like

## Bigram

What means, sir. I confess she? then all sorts, he is trim, captain.  
Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.  
What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman?

## Trigram

Sweet prince, Falstaff shall die. Harry of Monmouth's grave.  
This shall forbid it should be branded, if renown made it empty.  
Indeed the duke; and had a very good friend.  
Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.

## Quadrigram

King Henry.What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;  
Will you not tell me who I am?  
It cannot be but so.  
Indeed the short and the long. Marry, 'tis a noble Lepidus.



## Shakespeare as corpus

- $N=884,647$  tokens,  $V=29,066$
- Shakespeare produced 300,000 bigram types out of  $V^2= 844$  million possible bigrams.
  - So 99.96% of the possible bigrams were never seen (have zero entries in the table)
- Quadrigrams worse: What's coming out looks like Shakespeare because it *is* Shakespeare



# The wall street journal is not shakespeare (no offense)

## Unigram

Months the my and issue of year foreign new exchange's september were recession ex-change new endorsed a acquire to six executives

## Bigram

Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her

## Trigram

They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions



## The perils of overfitting

- N-grams only work well for word prediction if the test corpus looks like the training corpus
  - In real life, it often doesn't
  - We need to train robust models that generalize!
  - One kind of generalization: Zeros!
    - Things that don't ever occur in the training set
      - But occur in the test set



# Zeros

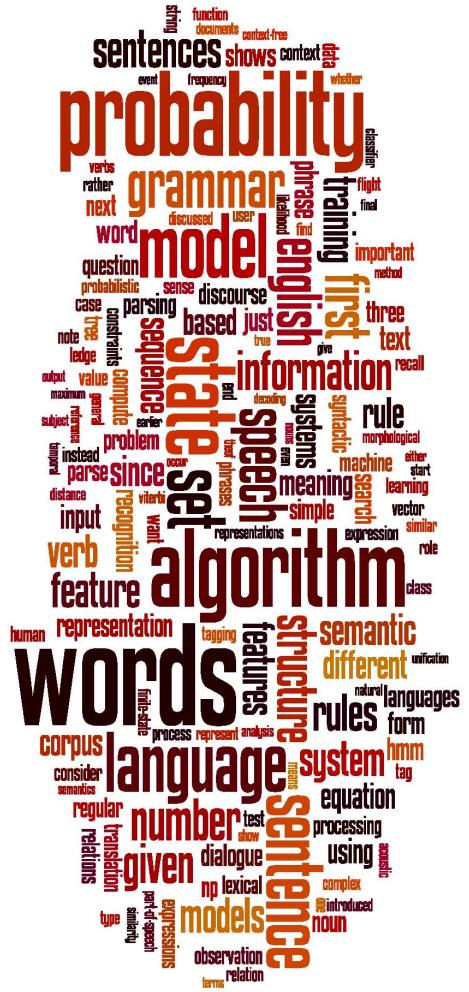
- Training set:
  - ... denied the allegations
  - ... denied the reports
  - ... denied the claims
  - ... denied the request
- Test set
  - ... denied the offer
  - ... denied the loan

$$P(\text{"offer"} \mid \text{denied the}) = 0$$



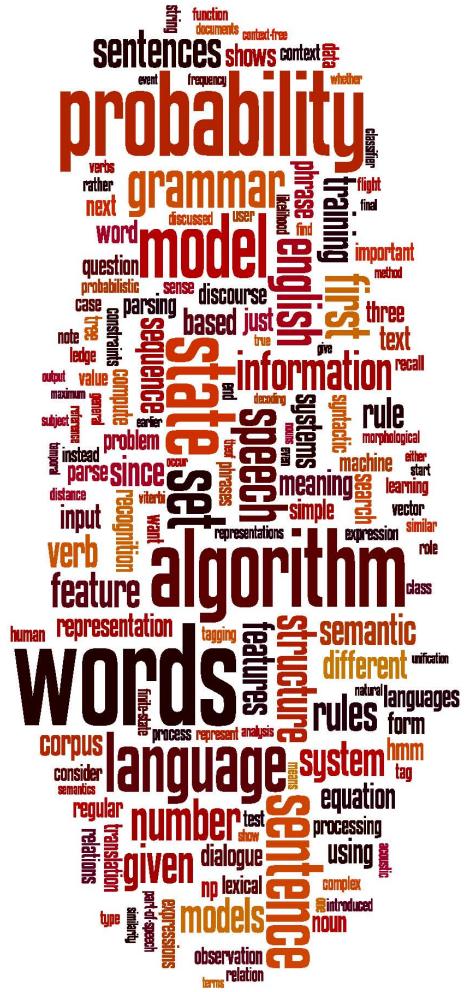
## Zero probability bigrams

- Bigrams with zero probability
  - mean that we will assign 0 probability to the test set!
- And hence we cannot compute perplexity (can't divide by 0)!



# Language Modeling

# Generalization and zeros



# Language Modeling

# Smoothing: Add-one (Laplace) smoothing



## The intuition of smoothing (from Dan Klein)

- When we have sparse statistics:

$P(w | \text{denied the})$

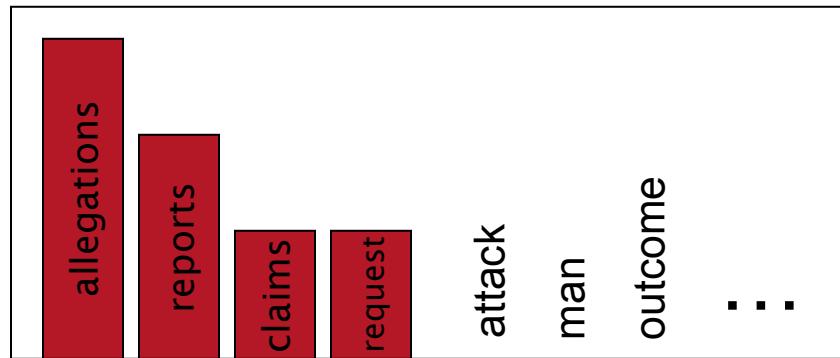
3 allegations

2 reports

1 claims

1 request

7 total



- Steal probability mass to generalize better

$P(w | \text{denied the})$

2.5 allegations

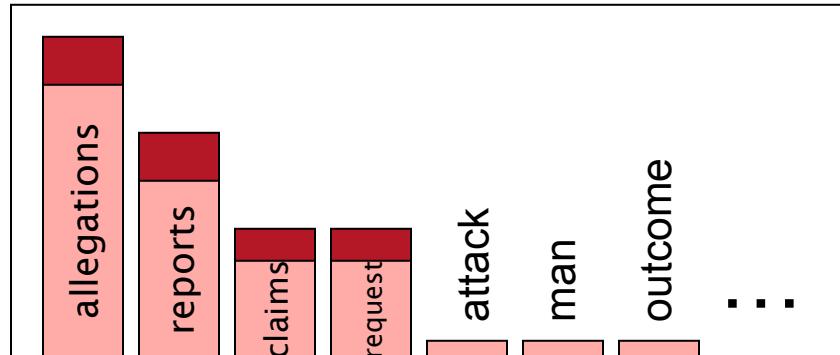
1.5 reports

0.5 claims

0.5 request

2 other

7 total





## Add-one estimation

- Also called Laplace smoothing
- Pretend we saw each word one more time than we did
- Just add one to all the counts!

$$P_{MLE}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

- MLE estimate:

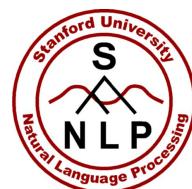
$$P_{Add-1}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + V}$$

- Add-1 estimate:



# Maximum Likelihood Estimates

- The maximum likelihood estimate
  - of some parameter of a model M from a training set T
  - maximizes the likelihood of the training set T given the model M
- Suppose the word “bagel” occurs 400 times in a corpus of a million words
- What is the probability that a random word from some other text will be “bagel”?
- MLE estimate is  $400/1,000,000 = .0004$
- This may be a bad estimate for some other corpus
  - But it is the **estimate** that makes it **most likely** that “bagel” will occur 400 times in a million word corpus.



# Berkeley Restaurant Corpus: Laplace smoothed bigram counts

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1



# Laplace-smoothed bigrams

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{C(w_{n-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	<b>0.00025</b>	0.0025	<b>0.00025</b>	<b>0.00025</b>	<b>0.00025</b>	0.00075
want	0.0013	<b>0.00042</b>	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	<b>0.00026</b>	0.0013	0.18	0.00078	<b>0.00026</b>	0.0018	0.055
eat	<b>0.00046</b>	<b>0.00046</b>	0.0014	<b>0.00046</b>	0.0078	0.0014	0.02	<b>0.00046</b>
chinese	0.0012	<b>0.00062</b>	<b>0.00062</b>	<b>0.00062</b>	<b>0.00062</b>	0.052	0.0012	<b>0.00062</b>
food	0.0063	<b>0.00039</b>	0.0063	<b>0.00039</b>	0.00079	0.002	<b>0.00039</b>	<b>0.00039</b>
lunch	0.0017	<b>0.00056</b>	<b>0.00056</b>	<b>0.00056</b>	<b>0.00056</b>	0.0011	<b>0.00056</b>	<b>0.00056</b>
spend	0.0012	<b>0.00058</b>	0.0012	<b>0.00058</b>	<b>0.00058</b>	<b>0.00058</b>	<b>0.00058</b>	<b>0.00058</b>



# Reconstituted counts

$$c^*(w_{n-1}w_n) = \frac{[C(w_{n-1}w_n) + 1] \times C(w_{n-1})}{C(w_{n-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16



# Compare with raw bigram counts

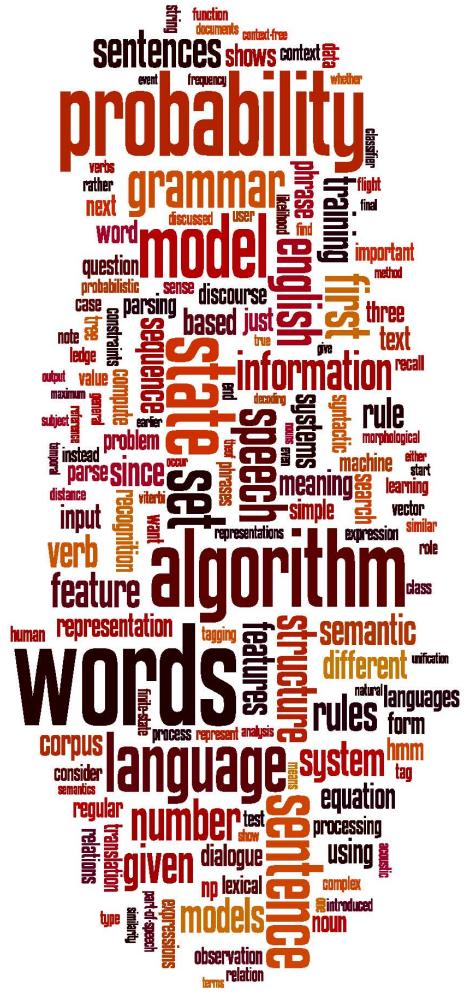
	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16



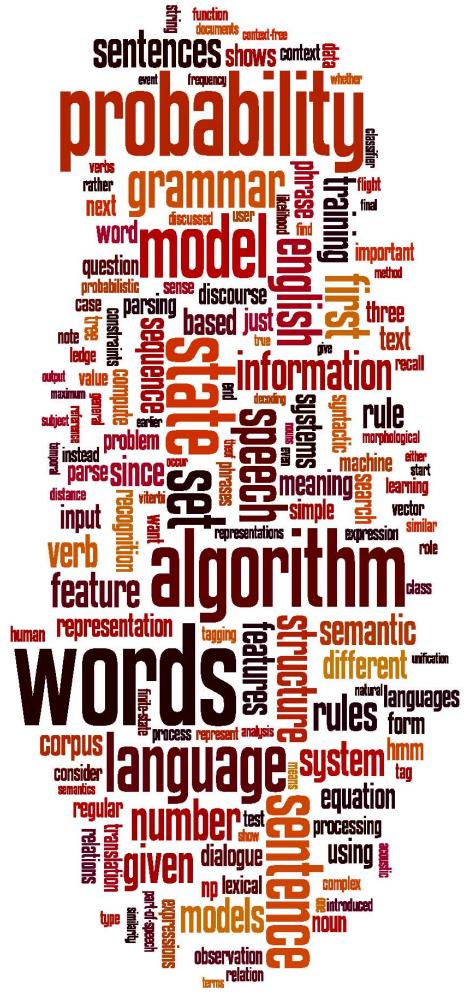
## Add-1 estimation is a blunt instrument

- So add-1 isn't used for N-grams:
  - We'll see better methods
- But add-1 is used to smooth other NLP models
  - For text classification
  - In domains where the number of zeros isn't so huge.



# Language Modeling

# Smoothing: Add-one (Laplace) smoothing



# Language Modeling

# Interpolation, Backoff, and Web-Scale LMs



# Backoff and Interpolation

- Sometimes it helps to use **less** context
  - Condition on less context for contexts you haven't learned much about
- **Backoff:**
  - use trigram if you have good evidence,
  - otherwise bigram, otherwise unigram
- **Interpolation:**
  - mix unigram, bigram, trigram
- Interpolation works better



# Linear Interpolation

- Simple interpolation

$$\begin{aligned}\hat{P}(w_n | w_{n-1} w_{n-2}) = & \lambda_1 P(w_n | w_{n-1} w_{n-2}) \\ & + \lambda_2 P(w_n | w_{n-1}) \\ & + \lambda_3 P(w_n)\end{aligned}$$

$$\sum_i \lambda_i = 1$$

- Lambdas conditional on context:

$$\begin{aligned}\hat{P}(w_n | w_{n-2} w_{n-1}) = & \lambda_1(w_{n-2}^{n-1}) P(w_n | w_{n-2} w_{n-1}) \\ & + \lambda_2(w_{n-2}^{n-1}) P(w_n | w_{n-1}) \\ & + \lambda_3(w_{n-2}^{n-1}) P(w_n)\end{aligned}$$



# How to set the lambdas?

- Use a **held-out** corpus

Training Data

Held-Out  
Data

Test  
Data

- Choose  $\lambda$ s to maximize the probability of held-out data:
  - Fix the N-gram probabilities (on the training data)
  - Then search for  $\lambda$ s that give largest probability to held-out set:

$$\log P(w_1 \dots w_n | M(\lambda_1 \dots \lambda_k)) = \sum_i \log P_{M(\lambda_1 \dots \lambda_k)}(w_i | w_{i-1})$$



# Unknown words: Open versus closed vocabulary tasks

- If we know all the words in advanced
  - Vocabulary  $V$  is fixed
  - Closed vocabulary task
- Often we don't know this
  - **Out Of Vocabulary** = OOV words
  - Open vocabulary task
- Instead: create an unknown word token <UNK>
  - Training of <UNK> probabilities
    - Create a fixed lexicon  $L$  of size  $V$
    - At text normalization phase, any training word not in  $L$  changed to <UNK>
    - Now we train its probabilities like a normal word
  - At decoding time
    - If text input: Use UNK probabilities for any word not in training



# Huge web-scale n-grams

- How to deal with, e.g., Google N-gram corpus
- Pruning
  - Only store N-grams with count > threshold.
    - Remove singletons of higher-order n-grams
    - Entropy-based pruning
- Efficiency
  - Efficient data structures like tries
  - Bloom filters: approximate language models
  - Store words as indexes, not strings
    - Use Huffman coding to fit large numbers of words into two bytes
  - Quantize probabilities (4-8 bits instead of 8-byte float)



## Smoothing for Web-scale N-grams

- “Stupid backoff” (Brants *et al.* 2007)
- No discounting, just use relative frequencies

$$S(w_i | w_{i-k+1}^{i-1}) = \begin{cases} \frac{\text{count}(w_{i-k+1}^i)}{\text{count}(w_{i-k+1}^{i-1})} & \text{if } \text{count}(w_{i-k+1}^i) > 0 \\ 0.4S(w_i | w_{i-k+2}^{i-1}) & \text{otherwise} \end{cases}$$

$$S(w_i) = \frac{\text{count}(w_i)}{N}$$



## N-gram Smoothing Summary

- Add-1 smoothing:
  - OK for text categorization, not for language modeling
- The most commonly used method:
  - Extended Interpolated Kneser-Ney
- For very large N-grams like the Web:
  - Stupid backoff

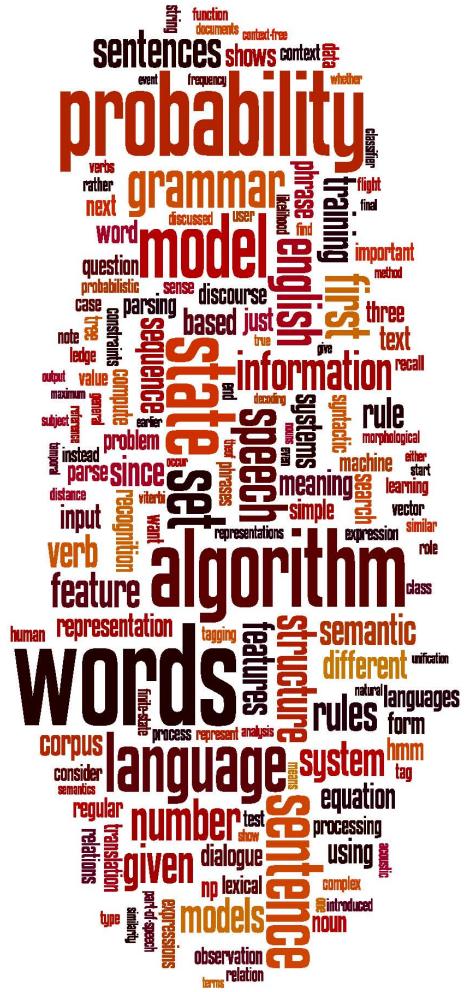


# Advanced Language Modeling

- Discriminative models:
  - choose n-gram weights to improve a task, not to fit the training set
- Parsing-based models
- Caching Models
  - Recently used words are more likely to appear

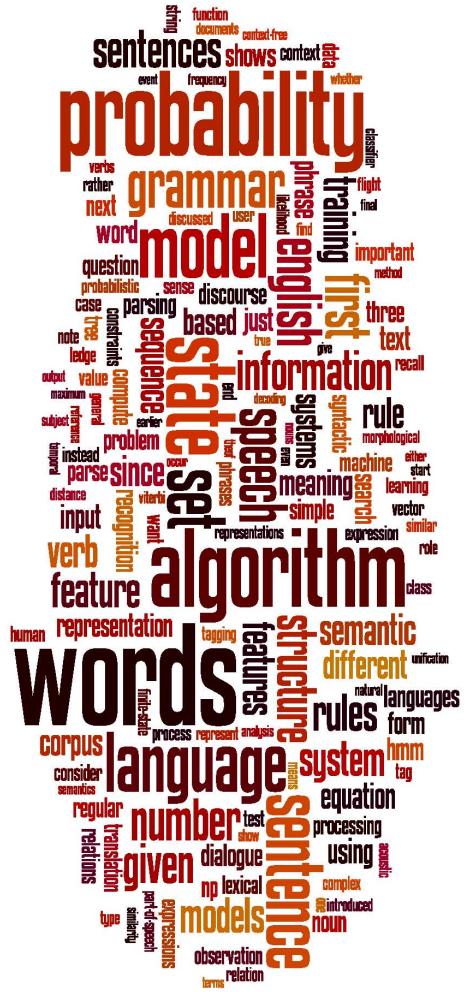
$$P_{CACHE}(w | history) = \lambda P(w_i | w_{i-2} w_{i-1}) + (1 - \lambda) \frac{c(w \in history)}{|history|}$$

- These perform very poorly for speech recognition (why?)



# Language Modeling

# Interpolation, Backoff, and Web-Scale LMs



# Language Modeling

# Advanced: Good Turing Smoothing



# Reminder: Add-1 (Laplace) Smoothing

$$P_{Add-1}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + V}$$



## More general formulations: Add-k

$$P_{Add-k}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i) + k}{c(w_{i-1}) + kV}$$

$$P_{Add-k}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i) + m(\frac{1}{V})}{c(w_{i-1}) + m}$$



# Unigram prior smoothing

$$P_{Add-k}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) + m(\frac{1}{V})}{c(w_{i-1}) + m}$$

$$P_{\text{UnigramPrior}}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) + mP(w_i)}{c(w_{i-1}) + m}$$



# Advanced smoothing algorithms

- Intuition used by many smoothing algorithms
  - Good-Turing
  - Kneser-Ney
  - Witten-Bell
- Use the count of things we've **seen once**
  - to help estimate the count of things we've **never seen**



## Notation: $N_c$ = Frequency of frequency c

- $N_c$  = the count of things we've seen c times
- Sam I am I am Sam I do not eat

I 3

sam 2

am 2

$$N_1 = 3$$

do 1

$$N_2 = 2$$

not 1

$$N_3 = 1$$

eat 1



# Good-Turing smoothing intuition

- You are fishing (a scenario from Josh Goodman), and caught:
  - 10 carp, 3 perch, 2 whitefish, **1 trout, 1 salmon, 1 eel** = 18 fish
- How likely is it that next species is trout?
  - $1/18$
- How likely is it that next species is new (i.e. catfish or bass)
  - Let's use our estimate of things-we-saw-once to estimate the new things.
  - $3/18$  (because  $N_1=3$ )
- Assuming so, how likely is it that next species is trout?
  - Must be less than  $1/18$
  - How to estimate?



# Good Turing calculations

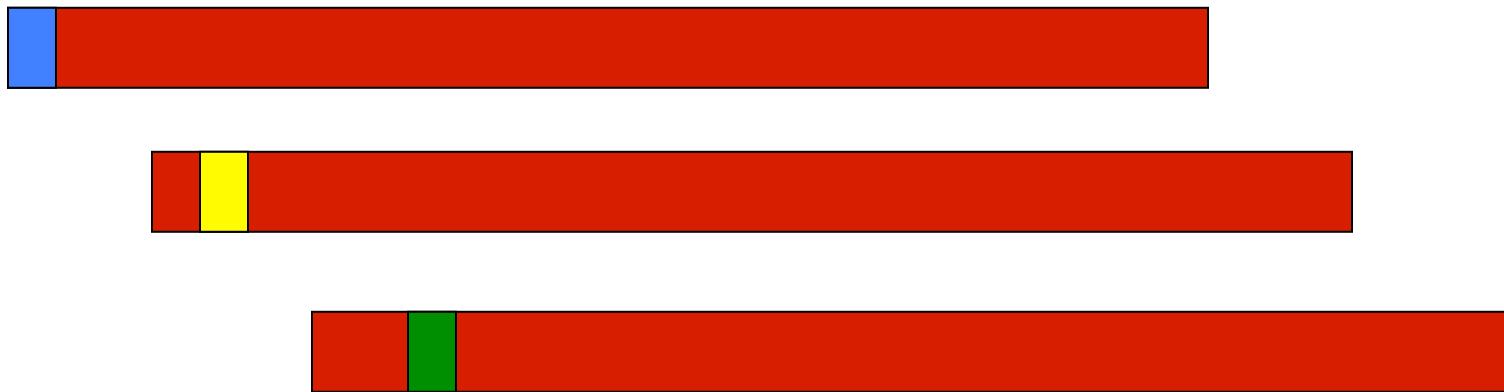
$$P_{GT}^*(\text{things with zero frequency}) = \frac{N_1}{N} \quad c^* = \frac{(c+1)N_{c+1}}{N_c}$$

- Unseen (bass or catfish)
  - $c = 0$ :
  - MLE  $p = 0/18 = 0$
  - $P_{GT}^*(\text{unseen}) = N_1/N = 3/18$
- Seen once (trout)
  - $c = 1$
  - MLE  $p = 1/18$
  - $C^*(\text{trout}) = 2 * N_2/N_1$   
 $= 2 * 1/3$   
 $= 2/3$
  - $P_{GT}^*(\text{trout}) = 2/3 / 18 = 1/27$



# Ney et al.'s Good Turing Intuition

H. Ney, U. Essen, and R. Kneser, 1995. On the estimation of 'small' probabilities by leaving-one-out.  
IEEE Trans. PAMI. 17:12,1202-1212



Held-out words:



## Ney et al. Good Turing Intuition (slide from Dan Klein)

- Intuition from leave-one-out validation
  - Take each of the  $c$  training words out in turn
  - $c$  training sets of size  $c-1$ , held-out of size 1
  - What fraction of held-out words are unseen in training?
    - $N_1/c$
  - What fraction of held-out words are seen  $k$  times in training?
    - $(k+1)N_{k+1}/c$
  - So in the future we expect  $(k+1)N_{k+1}/c$  of the words to be those with training count  $k$
  - There are  $N_k$  words with training count  $k$
  - Each should occur with probability:
    - $(k+1)N_{k+1}/c/N_k$
  - ...or expected count:

$$k^* = \frac{(k+1)N_{k+1}}{N_k}$$

Training

$N_1$
$N_2$
$N_3$
⋮
$N_{3511}$
$N_{4417}$

Held out

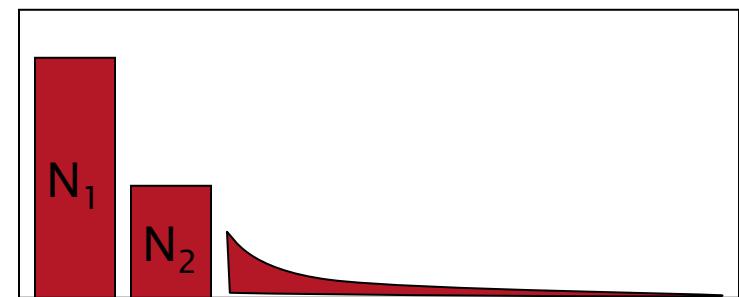
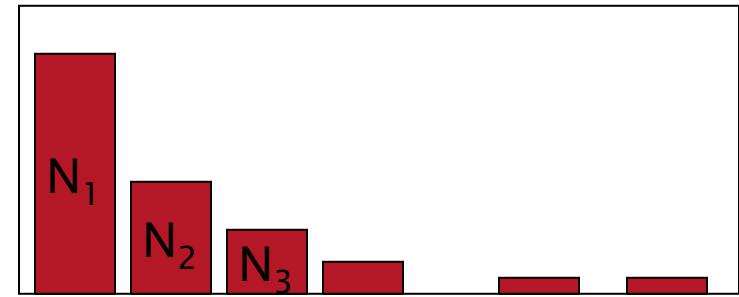
$N_0$
$N_1$
$N_2$
⋮
$N_{3510}$
$N_{4416}$



# Good-Turing complications

(slide from Dan Klein)

- Problem: what about “the”? (say  $c=4417$ )
  - For small  $k$ ,  $N_k > N_{k+1}$
  - For large  $k$ , too jumpy, zeros wreck estimates
- Simple Good-Turing [Gale and Sampson]: replace empirical  $N_k$  with a best-fit power law once counts get unreliable



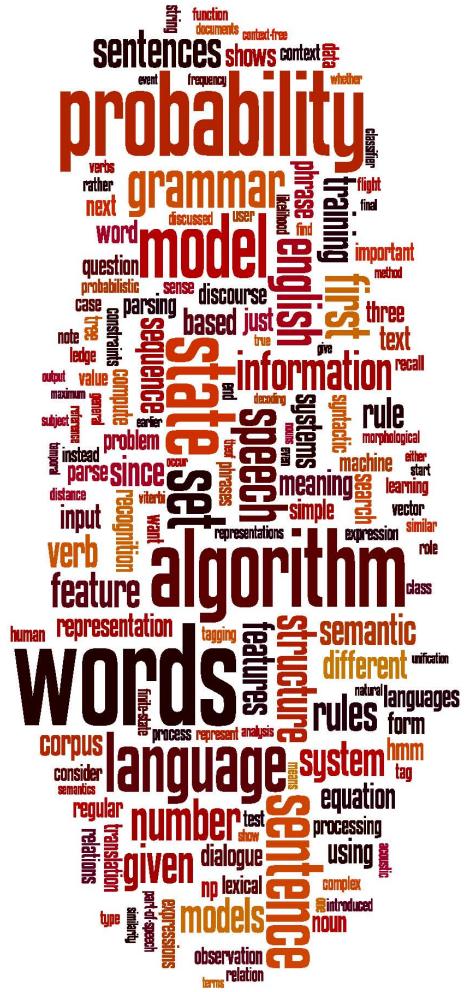


# Resulting Good-Turing numbers

- Numbers from Church and Gale (1991)
- 22 million words of AP Newswire

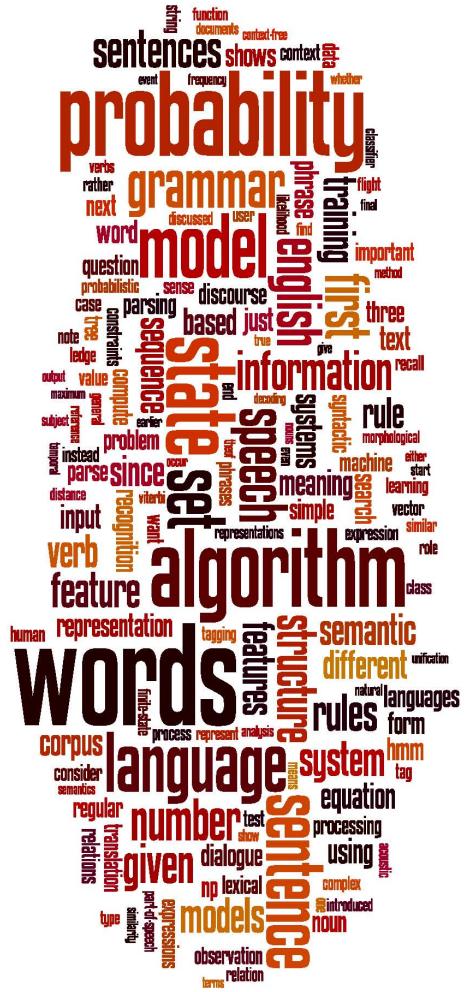
$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

Count c	Good Turing c*
0	.0000270
1	0.446
2	1.26
3	2.24
4	3.24
5	4.22
6	5.19
7	6.21
8	7.24
9	8.25



# Language Modeling

# Advanced: Good Turing Smoothing



# Language Modeling

# Advanced: Kneser-Ney Smoothing



# Resulting Good-Turing numbers

- Numbers from Church and Gale (1991)
- 22 million words of AP Newswire

$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

- It sure looks like  $c^* = (c - .75)$

Count c	Good Turing c*
0	.0000270
1	0.446
2	1.26
3	2.24
4	3.24
5	4.22
6	5.19
7	6.21
8	7.24
9	8.25



# Absolute Discounting Interpolation

- Save ourselves some time and just subtract 0.75 (or some d)!

$$P_{\text{AbsoluteDiscounting}}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) - d}{c(w_{i-1})} + \lambda(w_{i-1}) P(w)$$

discounted bigram      Interpolation weight  
↖ unigram

- (Maybe keeping a couple extra values of d for counts 1 and 2)
- But should we really just use the regular unigram  $P(w)$ ?



# Kneser-Ney Smoothing I

- Better estimate for probabilities of lower-order unigrams!
  - Shannon game: *I can't see without my reading* Fglasses? ?
  - “Francisco” is more common than “glasses”
  - ... but “Francisco” always follows “San”
- The unigram is useful exactly when we haven’t seen this bigram!
- Instead of  $P(w)$ : “How likely is  $w$ ”
- $P_{\text{continuation}}(w)$ : “How likely is  $w$  to appear as a novel continuation?
  - For each word, count the number of bigram types it completes
  - Every bigram type was a novel continuation the first time it was seen

$$P_{\text{CONTINUATION}}(w) \propto |\{w_{i-1} : c(w_{i-1}, w) > 0\}|$$



# Kneser-Ney Smoothing II

- How many times does  $w$  appear as a novel continuation:

$$P_{CONTINUATION}(w) \propto |\{w_{i-1} : c(w_{i-1}, w) > 0\}|$$

- Normalized by the total number of word bigram types

$$|\{(w_{j-1}, w_j) : c(w_{j-1}, w_j) > 0\}|$$

$$P_{CONTINUATION}(w) = \frac{|\{w_{i-1} : c(w_{i-1}, w) > 0\}|}{|\{(w_{j-1}, w_j) : c(w_{j-1}, w_j) > 0\}|}$$



# Kneser-Ney Smoothing III

- Alternative metaphor: The number of # of word types seen to precede w

$$|\{w_{i-1} : c(w_{i-1}, w) > 0\}|$$

- normalized by the # of words preceding all words:

$$P_{CONTINUATION}(w) = \frac{|\{w_{i-1} : c(w_{i-1}, w) > 0\}|}{\sum_{w'} |\{w'_{i-1} : c(w'_{i-1}, w') > 0\}|}$$

- A frequent word (Francisco) occurring in only one context (San) will have a low continuation probability



# Kneser-Ney Smoothing IV

$$P_{KN}(w_i | w_{i-1}) = \frac{\max(c(w_{i-1}, w_i) - d, 0)}{c(w_{i-1})} + \lambda(w_{i-1}) P_{CONTINUATION}(w_i)$$

$\lambda$  is a normalizing constant; the probability mass we've discounted

$$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})} |\{w : c(w_{i-1}, w) > 0\}|$$

the normalized discount

The number of word types that can follow  $w_{i-1}$   
 = # of word types we discounted  
 = # of times we applied normalized discount

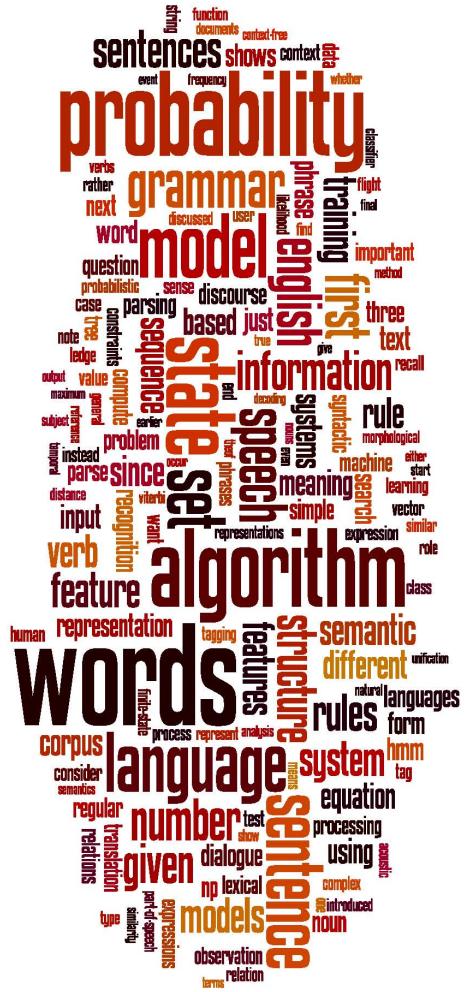


# Kneser-Ney Smoothing: Recursive formulation

$$P_{KN}(w_i \mid w_{i-n+1}^{i-1}) = \frac{\max(c_{KN}(w_{i-n+1}^i) - d, 0)}{c_{KN}(w_{i-n+1}^{i-1})} + \lambda(w_{i-n+1}^{i-1}) P_{KN}(w_i \mid w_{i-n+2}^{i-1})$$

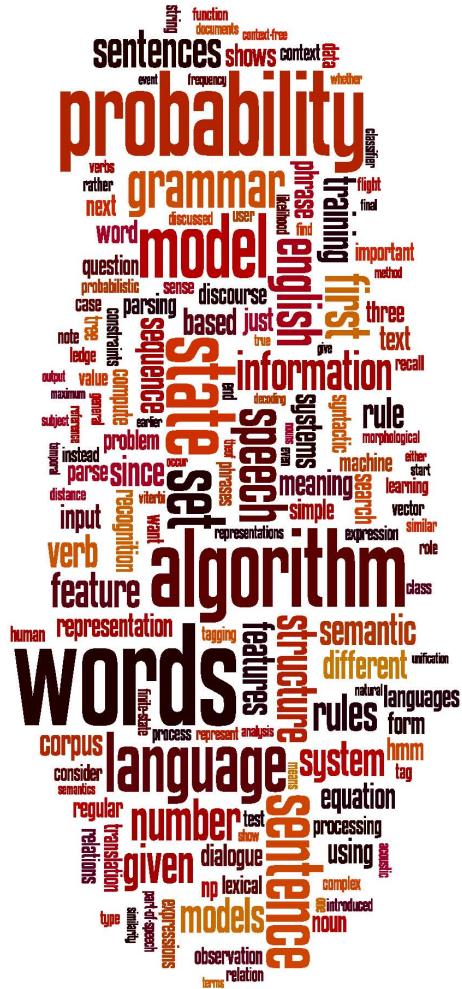
$$c_{KN}(\bullet) = \begin{cases} \text{count}(\bullet) & \text{for the highest order} \\ \text{continuation count}(\bullet) & \text{for lower order} \end{cases}$$

Continuation count = Number of unique single word contexts for  $\bullet$



# Language Modeling

# Advanced: Kneser-Ney Smoothing



# Text Classification and Naïve Bayes

# The Task of Text Classification

Dan Jurafsky



# Is this spam?

**Subject: Important notice!**

**From:** Stanford University <newsforum@stanford.edu>

**Date:** October 28, 2011 12:34:16 PM PDT

**To:** undisclosed-recipients:;

---

Greats News!

You can now access the latest news by using the link below to login to Stanford University News Forum.

<http://www.123contactform.com/contact-form-StanfordNew1-236335.html>

Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information about the new services.

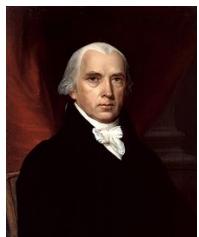
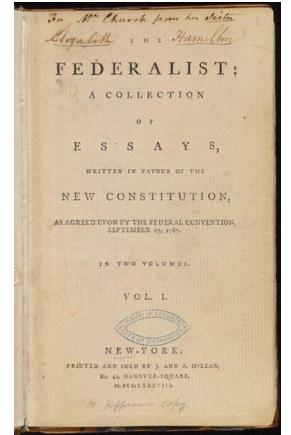
© Stanford University. All Rights Reserved.

Dan Jurafsky

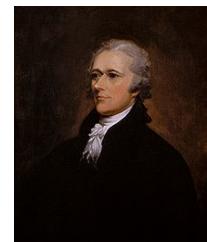


# Who wrote which Federalist papers?

- 1787-8: anonymous essays try to convince New York to ratify U.S Constitution: Jay, Madison, Hamilton.
- Authorship of 12 of the letters in dispute
- 1963: solved by Mosteller and Wallace using Bayesian methods



James Madison



Alexander Hamilton



## Male or female author?

1. By 1925 present-day Vietnam was divided into three parts under French colonial rule. The southern region embracing Saigon and the Mekong delta was the colony of Cochinchina; the central area with its imperial capital at Hue was the protectorate of Annam...
2. Clara never failed to be astonished by the extraordinary felicity of her own name. She found it hard to trust herself to the mercy of fate, which had managed over the years to convert her greatest shame into one of her greatest assets...

S. Argamon, M. Koppel, J. Fine, A. R. Shimoni, 2003. "Gender, Genre, and Writing Style in Formal Written Texts," *Text*, volume 23, number 3, pp. 321–346



## Positive or negative movie review?



- unbelievably disappointing



- Full of zany characters and richly applied satire, and some great plot twists



- this is the greatest screwball comedy ever filmed

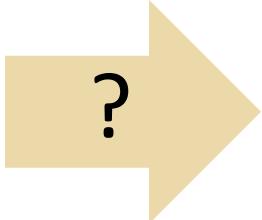


- It was pathetic. The worst part about it was the boxing scenes.



# What is the subject of this article?

## MEDLINE Article



## MeSH Subject Category Hierarchy

- Antagonists and Inhibitors
- Blood Supply
- Chemistry
- Drug Therapy
- Embryology
- Epidemiology
- ...



# Text Classification

- Assigning subject categories, topics, or genres
- Spam detection
- Authorship identification
- Age/gender identification
- Language Identification
- Sentiment analysis
- ...



## Text Classification: definition

- *Input:*
  - a document  $d$
  - a fixed set of classes  $C = \{c_1, c_2, \dots, c_J\}$
- *Output:* a predicted class  $c \in C$



# Classification Methods: Hand-coded rules

- Rules based on combinations of words or other features
  - spam: black-list-address OR (“dollars” AND “have been selected”)
- Accuracy can be high
  - If rules carefully refined by expert
- But building and maintaining these rules is expensive



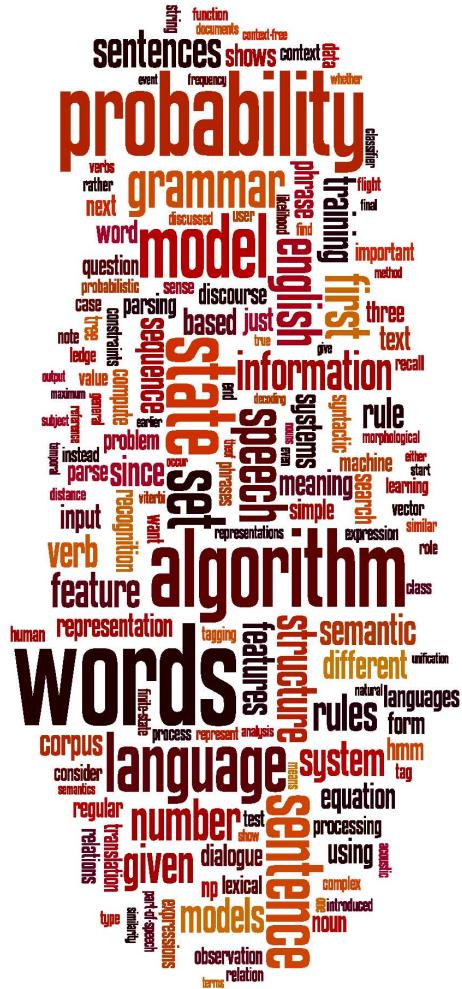
# Classification Methods: Supervised Machine Learning

- *Input:*
  - a document  $d$
  - a fixed set of classes  $C = \{c_1, c_2, \dots, c_J\}$
  - A training set of  $m$  hand-labeled documents  $(d_1, c_1), \dots, (d_m, c_m)$
- *Output:*
  - a learned classifier  $\gamma: d \rightarrow c$



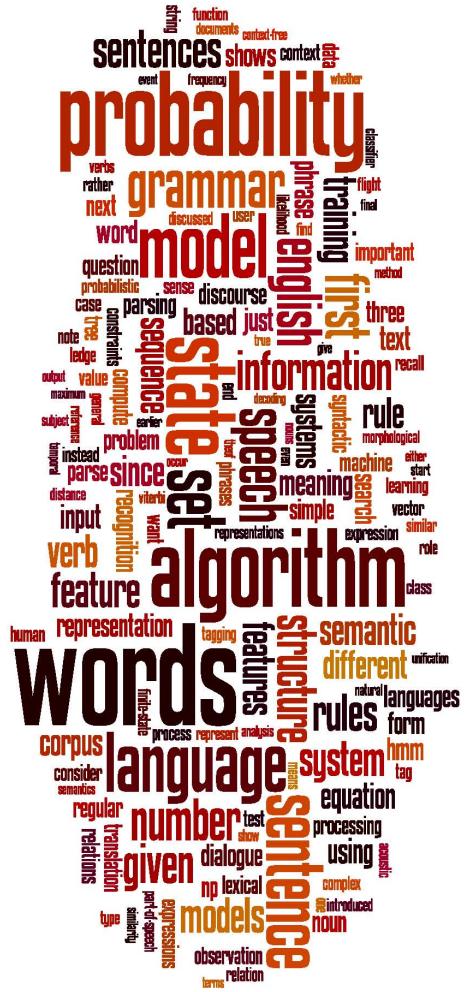
# Classification Methods: Supervised Machine Learning

- Any kind of classifier
  - Naïve Bayes
  - Logistic regression
  - Support-vector machines
  - k-Nearest Neighbors
  - ...



# Text Classification and Naïve Bayes

# The Task of Text Classification



# Text Classification and Naïve Bayes

# Naïve Bayes (I)



## Naïve Bayes Intuition

- Simple (“naïve”) classification method based on Bayes rule
- Relies on very simple representation of document
  - Bag of words



Y(

## The bag of words representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet.

)=C





# The bag of words representation

Y(

I **love** this movie! It's **sweet**, but with **satirical** humor. The dialogue is **great** and the adventure scenes are **fun**... It manages to be **whimsical** and **romantic** while **laughing** at the conventions of the fairy tale genre. I would **recommend** it to just about anyone. I've seen it **several** times, and I'm always **happy** to see it **again** whenever I have a friend who hasn't seen it yet.

)=C





# The bag of words representation: using a subset of words

Y(

```
x love xxxxxxxxxxxxxxxx sweet
xxxxxxxx satirical xxxxxxxxxxx
xxxxxxxxxx great xxxxxxx
xxxxxxxxxxxxxxxx fun xxxx
xxxxxxxxxxxxxx whimsical xxxx
romantic xxxx laughing
xxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxx recommend xxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx several xxxxxxxxxxxxxxxx
xxxxx happy xxxxxxxxx again
xxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxx
```

)=C





# The bag of words representation

$\gamma( \quad ) = C$

great	2
love	2
recommend	1
laugh	1
happy	1
...	...





# Bag of words for document classification

Test  
document

parser  
language  
label  
translation  
...

?

Machine  
Learning

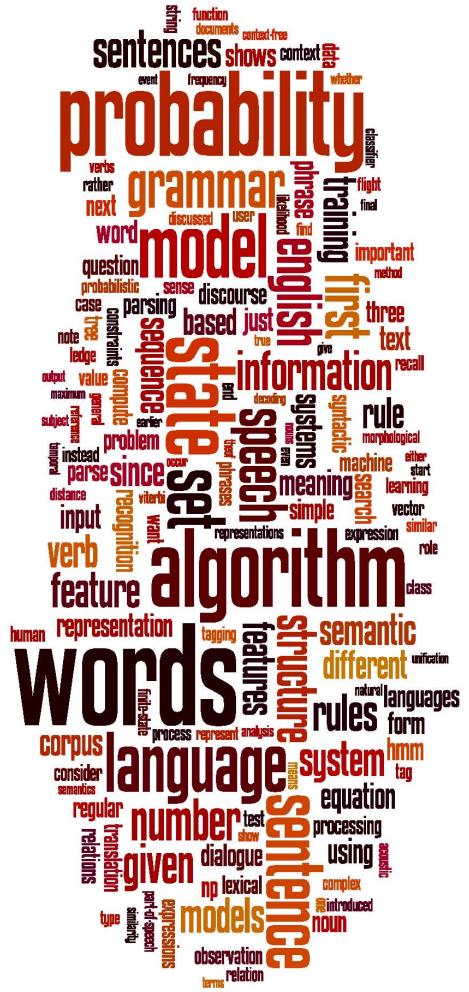
NLP

Garbage  
Collection

Planning

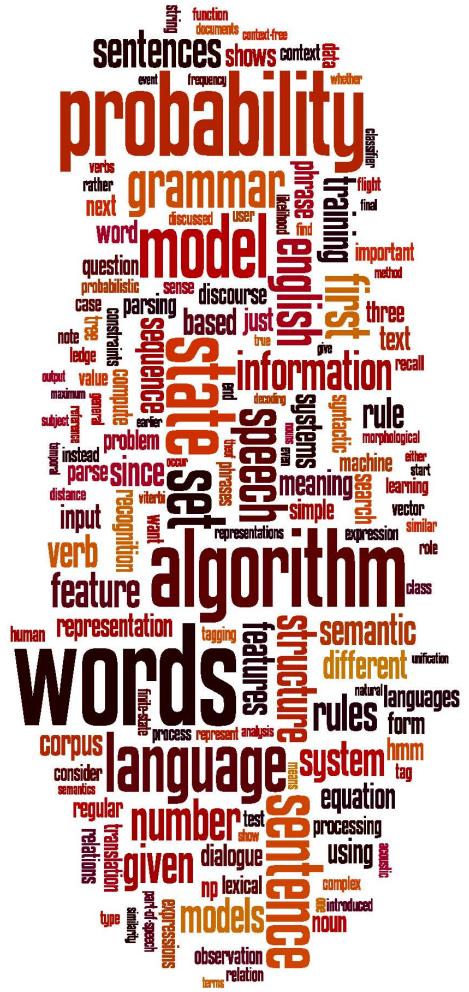
GUI

learning	<u>parser</u>	garbage	planning	...
<u>training</u>	tag	collection	temporal	
algorithm	training	memory	reasoning	
shrinkage	<u>translation</u>	optimization	plan	
network...	<u>language...</u>	region...	<u>language...</u>	



# Text Classification and Naïve Bayes

# Naïve Bayes (I)



# Text Classification and Naïve Bayes

# Formalizing the Naïve Bayes Classifier



# Bayes' Rule Applied to Documents and Classes

- For a document  $d$  and a class  $c$

$$P(c | d) = \frac{P(d | c)P(c)}{P(d)}$$



# Naïve Bayes Classifier (I)

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(c | d)$$

MAP is “maximum a posteriori” = most likely class

$$= \operatorname{argmax}_{c \in C} \frac{P(d | c)P(c)}{P(d)}$$

Bayes Rule

$$= \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

Dropping the denominator



## Naïve Bayes Classifier (II)

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

$$= \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c)P(c)$$

Document d  
represented as  
features  
x<sub>1..n</sub>



## Naïve Bayes Classifier (IV)

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c)P(c)$$

$O(|X|^n \cdot |C|)$  parameters

Could only be estimated if a very, very large number of training examples was available.

How often does this class occur?

We can just count the relative frequencies in a corpus



# Multinomial Naïve Bayes Independence Assumptions

$$P(x_1, x_2, \dots, x_n | c)$$

- **Bag of Words assumption:** Assume position doesn't matter
- **Conditional Independence:** Assume the feature probabilities  $P(x_i | c_j)$  are independent given the class  $c$ .

$$P(x_1, \dots, x_n | c) = P(x_1 | c) \bullet P(x_2 | c) \bullet P(x_3 | c) \bullet \dots \bullet P(x_n | c)$$



## Multinomial Naïve Bayes Classifier

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n \mid c)P(c)$$

$$c_{NB} = \operatorname{argmax}_{c \in C} P(c_j) \prod_{x \in X} P(x \mid c)$$



# Applying Multinomial Naive Bayes Classifiers to Text Classification

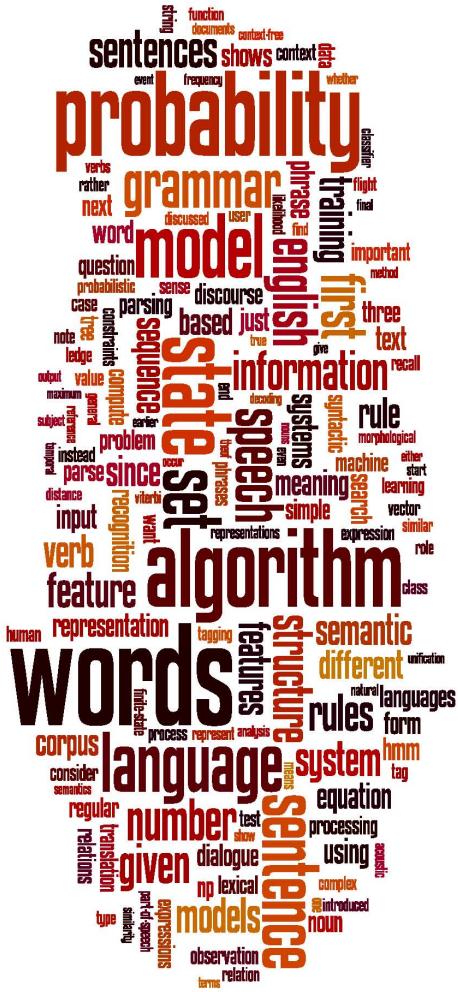
positions  $\leftarrow$  all word positions in test document

$$c_{NB} = \operatorname{argmax}_{c_j \in C} P(c_j) \prod_{i \in positions} P(x_i | c_j)$$



# Text Classification and Naïve Bayes

# Formalizing the Naïve Bayes Classifier



# Text Classification and Naïve Bayes

Naïve Bayes:  
Learning



## Learning the Multinomial Naïve Bayes Model

- First attempt: maximum likelihood estimates
  - simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i | c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$



# Parameter estimation

$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$

fraction of times word  $w_i$  appears  
among all words in documents of topic  $c_j$

- Create mega-document for topic  $j$  by concatenating all docs in this topic
  - Use frequency of  $w$  in mega-document



## Problem with Maximum Likelihood

- What if we have seen no training documents with the word ***fantastic*** and classified in the topic **positive (*thumbs-up*)?**

$$\hat{P}(\text{"fantastic"} \mid \text{positive}) = \frac{\text{count}(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} \text{count}(w, \text{positive})} = 0$$

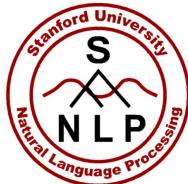
- Zero probabilities cannot be conditioned away, no matter the other evidence!

$$c_{MAP} = \operatorname{argmax}_c \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$



# Laplace (add-1) smoothing for Naïve Bayes

$$\hat{P}(w_i | c) = \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} (\text{count}(w, c) + 1)}$$
$$= \frac{\text{count}(w_i, c) + 1}{\left( \sum_{w \in V} \text{count}(w, c) \right) + |V|}$$



# Multinomial Naïve Bayes: Learning

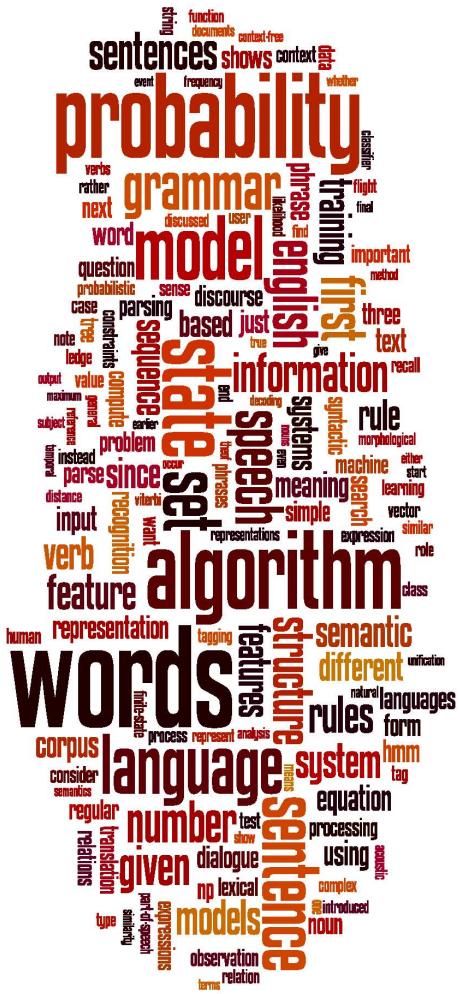
- From training corpus, extract *Vocabulary*
  - Calculate  $P(c_j)$  terms
    - For each  $c_j$  in  $C$  do  
 $docs_j \leftarrow$  all docs with class =  $c_j$
  - Calculate  $P(w_k | c_j)$  terms
    - $Text_j \leftarrow$  single doc containing all  $docs_j$
    - For each word  $w_k$  in *Vocabulary*  
 $n_k \leftarrow$  # of occurrences of  $w_k$  in  $Text_j$
- $$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$
- $$P(w_k | c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |\text{Vocabulary}|}$$



## Laplace (add-1) smoothing: unknown words

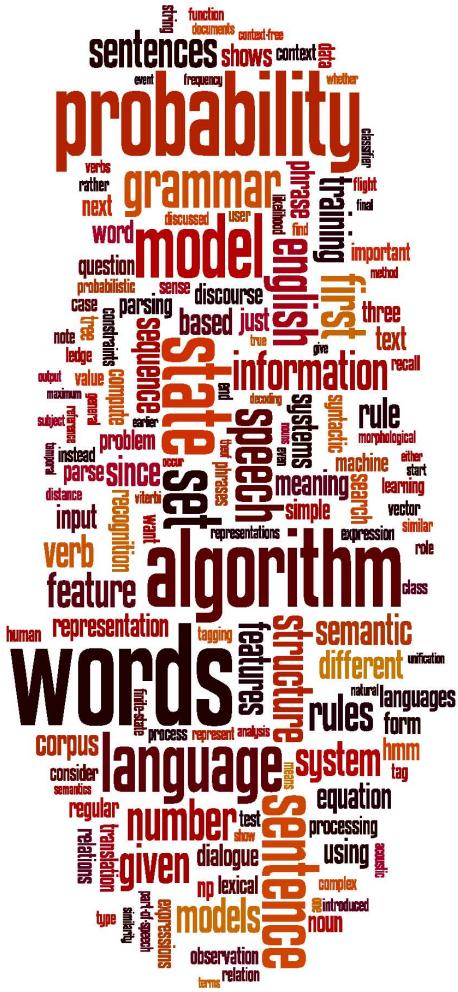
Add one extra word to the vocabulary, the “unknown word”  $w_u$

$$\begin{aligned}\hat{P}(w_u | c) &= \frac{\text{count}(w_u, c) + 1}{\left( \sum_{w \in V} \text{count}(w, c) \right) + |V + 1|} \\ &= \frac{1}{\left( \sum_{w \in V} \text{count}(w, c) \right) + |V + 1|}\end{aligned}$$



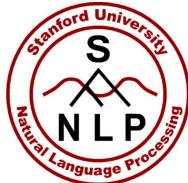
# Text Classification and Naïve Bayes

Naïve Bayes:  
Learning

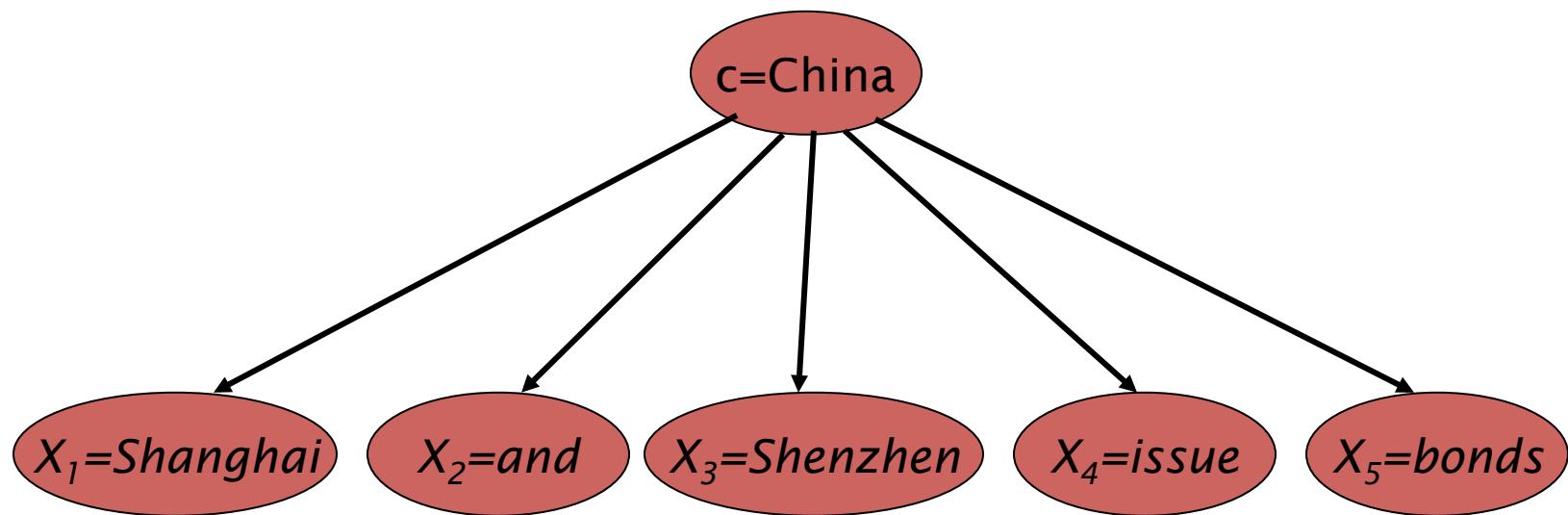


# Text Classification and Naïve Bayes

Naïve Bayes:  
Relationship to  
Language Modeling



# Generative Model for Multinomial Naïve Bayes





# Naïve Bayes and Language Modeling

- Naïve bayes classifiers can use any sort of feature
  - URL, email address, dictionaries, network features
- But if, as in the previous slides
  - We use **only** word features
  - we use **all** of the words in the text (not a subset)
- Then
  - Naïve bayes has an important similarity to language modeling.



## Each class = a unigram language model

- Assigning each word:  $P(\text{word} \mid c)$
- Assigning each sentence:  $P(s \mid c) = \prod P(\text{word} \mid c)$

Class *pos*

0.1	I		I	<u>love</u>	<u>this</u>	<u>fun</u>	<u>film</u>
0.1	love		0.1	0.1	.05	0.01	0.1
0.01	this						
0.05	fun						
0.1	film						

$$P(s \mid \text{pos}) = 0.0000005$$

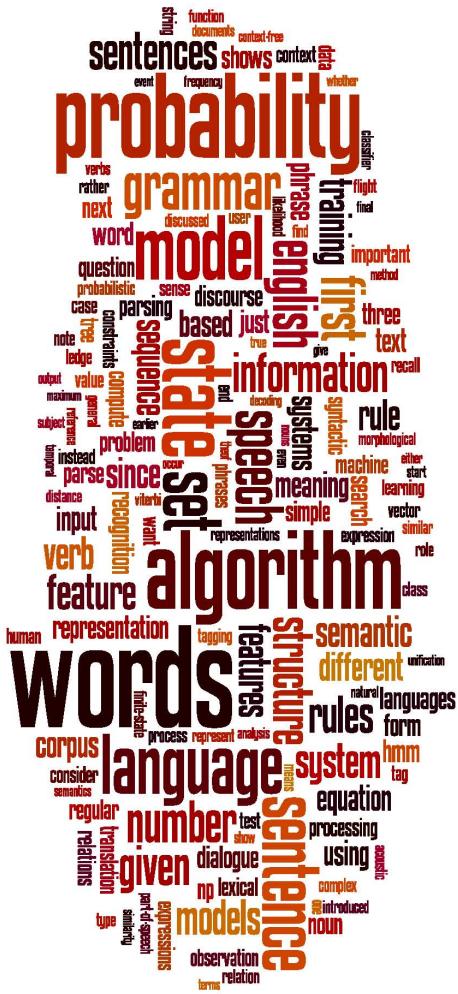


## Naïve Bayes as a Language Model

- Which class assigns the higher probability to s?

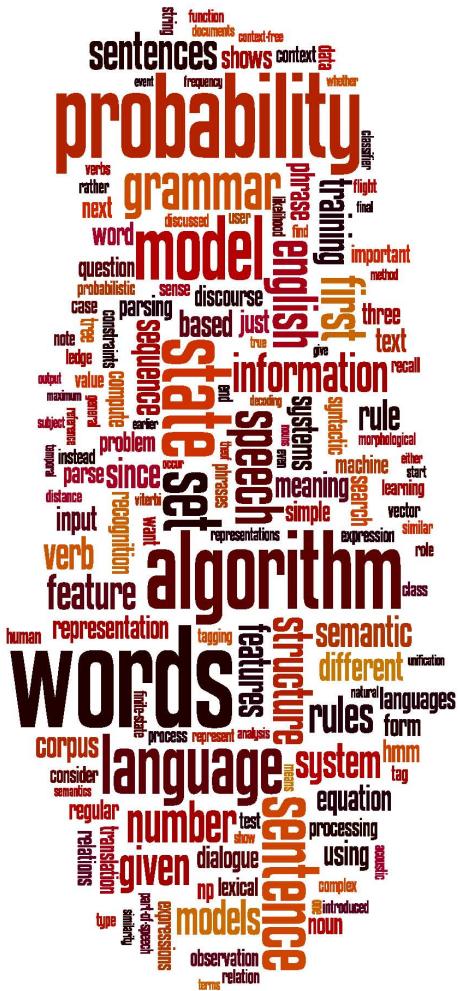
Model pos		Model neg						
0.1	I	0.2	I	I				
0.1	love	0.001	love	love				
0.01	this	0.01	this	this				
0.05	fun	0.005	fun	fun				
0.1	film	0.1	film	film				

$P(s|pos) > P(s|neg)$



# Text Classification and Naïve Bayes

Naïve Bayes:  
Relationship to  
Language Modeling



# Text Classification and Naïve Bayes

Multinomial Naïve  
Bayes: A Worked  
Example



$$\hat{P}(c) = \frac{N_c}{N}$$

$$\hat{P}(w|c) = \frac{\text{count}(w,c)+1}{\text{count}(c)+|V|}$$

**Priors:**

$$P(c) = \frac{3}{4}$$

$$P(j) = \frac{1}{4}$$

**Conditional Probabilities:**

$$P(\text{Chinese}|c) = (5+1) / (8+6) = 6/14 = 3/7$$

$$P(\text{Tokyo}|c) = (0+1) / (8+6) = 1/14$$

$$P(\text{Japan}|c) = (0+1) / (8+6) = 1/14$$

$$P(\text{Chinese}|j) = (1+1) / (3+6) = 2/9$$

$$P(\text{Tokyo}|j) = (1+1) / (3+6) = 2/9$$

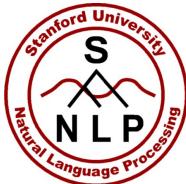
$$45 \quad P(\text{Japan}|j) = (1+1) / (3+6) = 2/9$$

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

**Choosing a class:**

$$P(c|d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 \\ \approx 0.0003$$

$$P(j|d5) \propto 1/4 * (2/9)^3 * 2/9 * 2/9 \\ \approx 0.0001$$



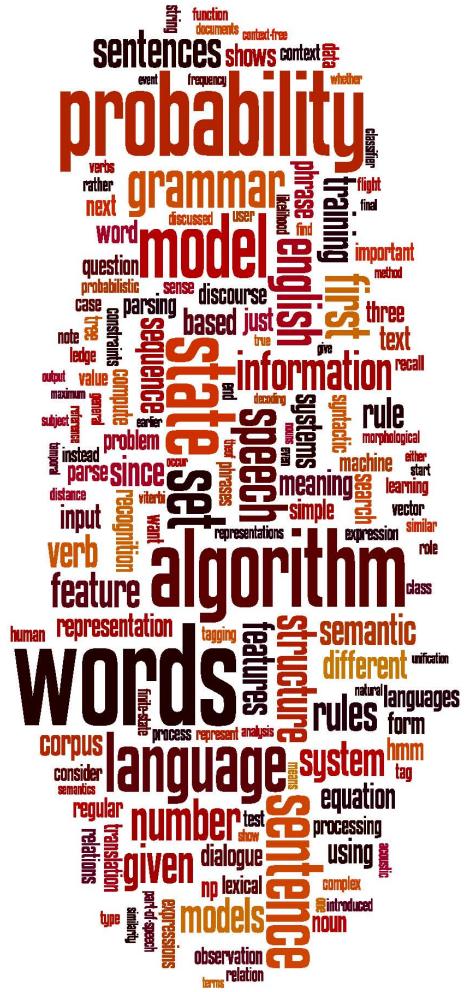
# Naïve Bayes in Spam Filtering

- SpamAssassin Features:
  - Mentions Generic Viagra
  - Online Pharmacy
  - Mentions millions of (dollar) ((dollar) NN,NNN,NNN.NN)
  - Phrase: impress ... girl
  - From: starts with many numbers
  - Subject is all capitals
  - HTML has a low ratio of text to image area
  - One hundred percent guaranteed
  - Claims you can be removed from the list
  - 'Prestigious Non-Accredited Universities'
  - [http://spamassassin.apache.org/tests\\_3\\_3\\_x.html](http://spamassassin.apache.org/tests_3_3_x.html)



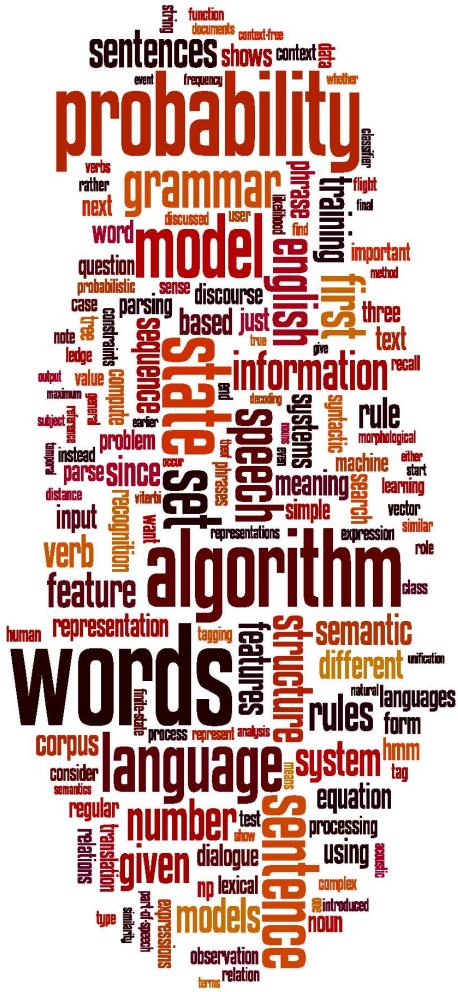
## Summary: Naive Bayes is Not So Naive

- Very Fast, low storage requirements
- Robust to Irrelevant Features
  - Irrelevant Features cancel each other without affecting results
- Very good in domains with many equally important features
  - Decision Trees suffer from *fragmentation* in such cases – especially if little data
- Optimal if the independence assumptions hold: If assumed independence is correct, then it is the Bayes Optimal Classifier for problem
- A good dependable baseline for text classification
  - **But we will see other classifiers that give better accuracy**



# Text Classification and Naïve Bayes

# Multinomial Naïve Bayes: A Worked Example



# Text Classification and Naïve Bayes

Precision, Recall, and  
the F measure



## The 2-by-2 contingency table

	correct	not correct
selected	tp	fp
not selected	fn	tn



# Precision and recall

- **Precision:** % of selected items that are correct  
**Recall:** % of correct items that are selected

	correct	not correct
selected	tp	fp
not selected	fn	tn



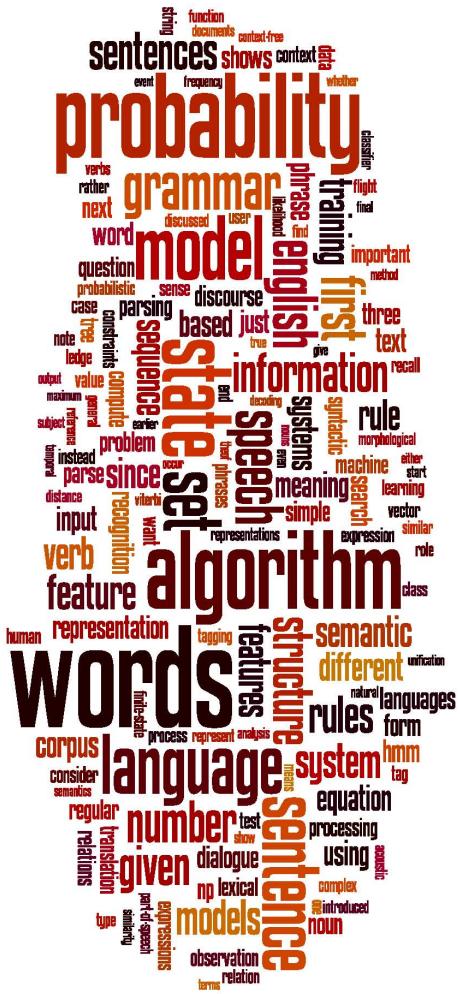
## A combined measure: F

- A combined measure that assesses the P/R tradeoff is F measure (weighted harmonic mean):

$$F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

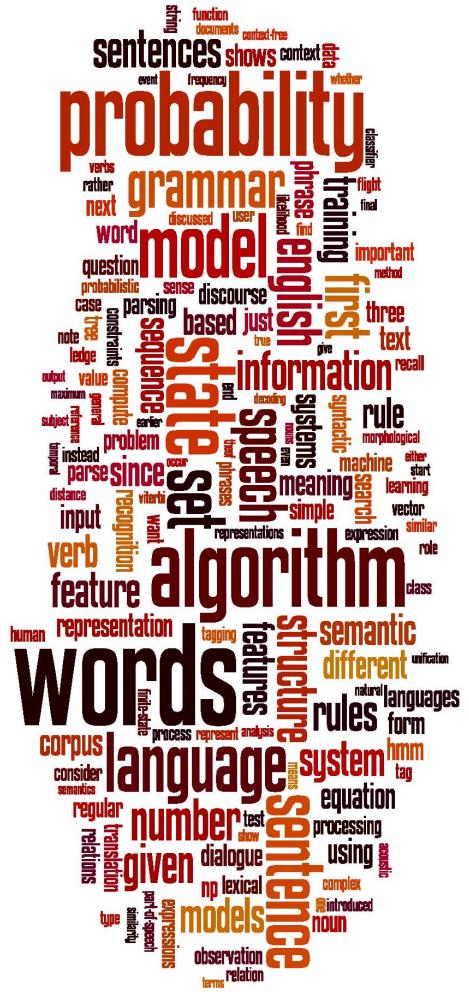
- The harmonic mean is a very conservative average; see *IIR* § 8.3
- People usually use balanced F1 measure
  - i.e., with  $\beta = 1$  (that is,  $\alpha = \frac{1}{2}$ ):

$$F = 2PR/(P+R)$$



# Text Classification and Naïve Bayes

Precision, Recall, and  
the F measure



# Text Classification and Naïve Bayes

# Text Classification: Evaluation



# More Than Two Classes: Sets of binary classifiers

- Dealing with **any-of** or **multivalue** classification
  - A document can belong to 0, 1, or >1 classes.
- For each class  $c \in C$ 
  - Build a classifier  $\gamma_c$  to distinguish  $c$  from all other classes  $c' \in C$
- Given test doc  $d$ ,
  - Evaluate it for membership in each class using each  $\gamma_c$
  - $d$  belongs to **any** class for which  $\gamma_c$  returns true



# More Than Two Classes: Sets of binary classifiers

- One-of or multinomial classification
  - Classes are mutually exclusive: each document in exactly one class
- For each class  $c \in C$ 
  - Build a classifier  $y_c$  to distinguish  $c$  from all other classes  $c' \in C$
- Given test doc  $d$ ,
  - Evaluate it for membership in each class using each  $y_c$
  - $d$  belongs to the one class with maximum score



# Evaluation: Classic Reuters-21578 Data Set

- Most (over)used data set, 21,578 docs (each 90 types, 200 tokens)
- 9603 training, 3299 test articles (ModApte/Lewis split)
- 118 categories
  - An article can be in more than one category
  - Learn 118 binary category distinctions
- Average document (with at least one category) has 1.24 classes
- Only about 10 out of 118 categories are large
  - Earn (2877, 1087)
  - Acquisitions (1650, 179)
  - Money-fx (538, 179)
  - Grain (433, 149)
  - Crude (389, 189)
  - Trade (369, 119)
  - Interest (347, 131)
  - Ship (197, 89)
  - Wheat (212, 71)
  - Corn (182, 56)

Common categories  
(#train, #test)

57



# Reuters Text Categorization data set (Reuters-21578) document

<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OL DID="12981" NEWID="798">

<DATE> 2-MAR-1987 16:51:43.42</DATE>

<TOPICS><D>livestock</D><D>hog</D></TOPICS>

<TITLE>AMERICAN PORK CONGRESS KICKS OFF TOMORROW</TITLE>

<DATELINE> CHICAGO, March 2 - </DATELINE><BODY>The American Pork Congress kicks off tomorrow, March 3, in Indianapolis with 160 of the nations pork producers from 44 member states determining industry positions on a number of issues, according to the National Pork Producers Council, NPPC.

Delegates to the three day Congress will be considering 26 resolutions concerning various issues, including the future direction of farm policy and the tax law as it applies to the agriculture sector. The delegates will also debate whether to endorse concepts of a national PRV (pseudorabies virus) control and eradication program, the NPPC said.

A large trade show, in conjunction with the congress, will feature the latest in technology in all areas of the industry, the NPPC added. Reuter

&#3; </BODY></TEXT></REUTERS>



# Confusion matrix c

- For each pair of classes  $\langle c_1, c_2 \rangle$  how many documents from  $c_1$  were incorrectly assigned to  $c_2$ ?
  - $c_{3,2}$ : 90 wheat documents incorrectly assigned to poultry

Docs in test set	Assigned UK	Assigned poultry	Assigned wheat	Assigned coffee	Assigned interest	Assigned trade
True UK	95	1	13	0	1	0
True poultry	0	1	0	0	0	0
True wheat	10	90	0	1	0	0
True coffee	0	0	0	34	3	7
True interest	-	1	2	13	26	5
True trade	0	0	2	14	5	10



## Per class evaluation measures

### Recall:

Fraction of docs in class  $i$  classified correctly:

$$\frac{c_{ii}}{\sum_j c_{ij}}$$

### Precision:

Fraction of docs assigned class  $i$  that are actually about class  $i$ :

$$\frac{c_{ii}}{\sum_j c_{ji}}$$

### Accuracy:

(1 - error rate)

Fraction of docs classified correctly:

$$\frac{\sum_i c_{ii}}{\sum_j \sum_i c_{ij}}$$



## Micro- vs. Macro-Averaging

- If we have more than one class, how do we combine multiple performance measures into one quantity?
- **Macroaveraging:** Compute performance for each class, then average.
- **Microaveraging:** Collect decisions for all classes, compute contingency table, evaluate.



# Micro- vs. Macro-Averaging: Example

Class 1

	Truth: yes	Truth: no
Classifier: yes	10	10
Classifier: no	10	970

Class 2

	Truth: yes	Truth: no
Classifier: yes	90	10
Classifier: no	10	890

Micro Ave. Table

	Truth: yes	Truth: no
Classifier: yes	100	20
Classifier: no	20	1860

- Macroaveraged precision:  $(0.5 + 0.9)/2 = 0.7$
- Microaveraged precision:  $100/120 = .83$
- Microaveraged score is dominated by score on common classes



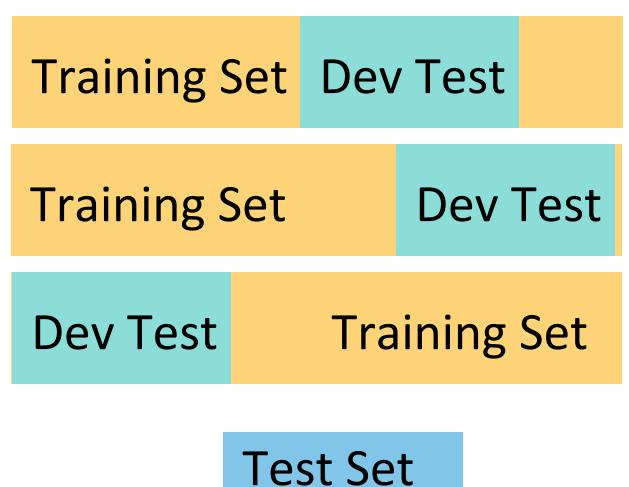
# Development Test Sets and Cross-validation

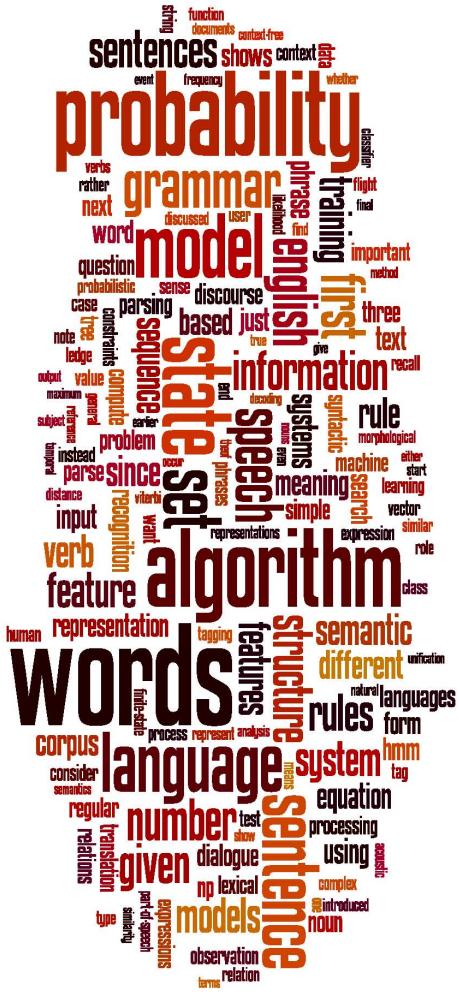
Training set

Development Test Set

Test Set

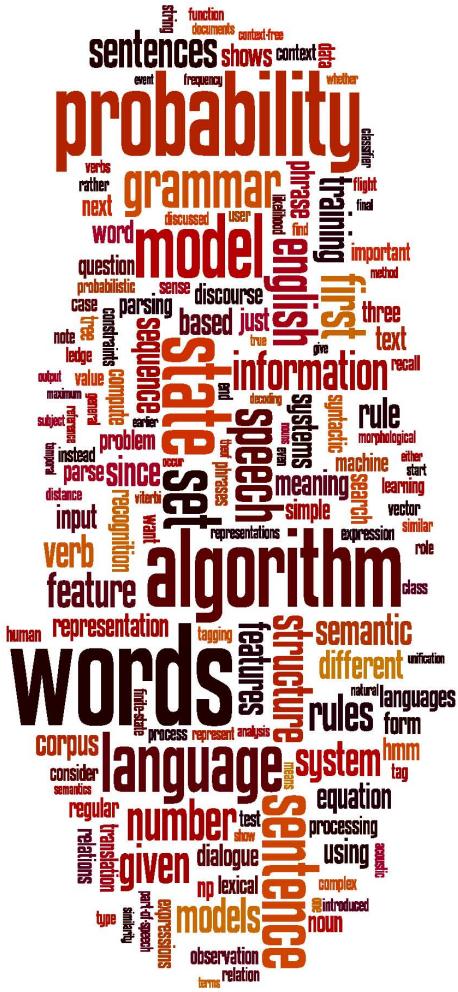
- Metric: P/R/F1 or Accuracy
- Unseen test set
  - avoid overfitting ('tuning to the test set')
  - more conservative estimate of performance
- Cross-validation over multiple splits
  - Handle sampling errors from different datasets
  - Pool results over each split
  - Compute pooled dev set performance





# Text Classification and Naïve Bayes

Text Classification:  
Evaluation



# Text Classification and Naïve Bayes

Text Classification:  
Practical Issues



# The Real World

- Gee, I'm building a text classifier for real, now!
- What should I do?



# No training data? Manually written rules

If (wheat or grain) and not (whole or bread) then  
Categorize as grain

- Need careful crafting
  - Human tuning on development data
  - Time-consuming: 2 days per class



## Very little data?

- Use Naïve Bayes
  - Naïve Bayes is a “high-bias” algorithm ([Ng and Jordan 2002 NIPS](#))
- Get more labeled data
  - Find clever ways to get humans to label data for you
- Try semi-supervised training methods:
  - Bootstrapping, EM over unlabeled documents, ...



## A reasonable amount of data?

- Perfect for all the clever classifiers
  - SVM
  - Regularized Logistic Regression
- You can even use user-interpretable decision trees
  - Users like to hack
  - Management likes quick fixes



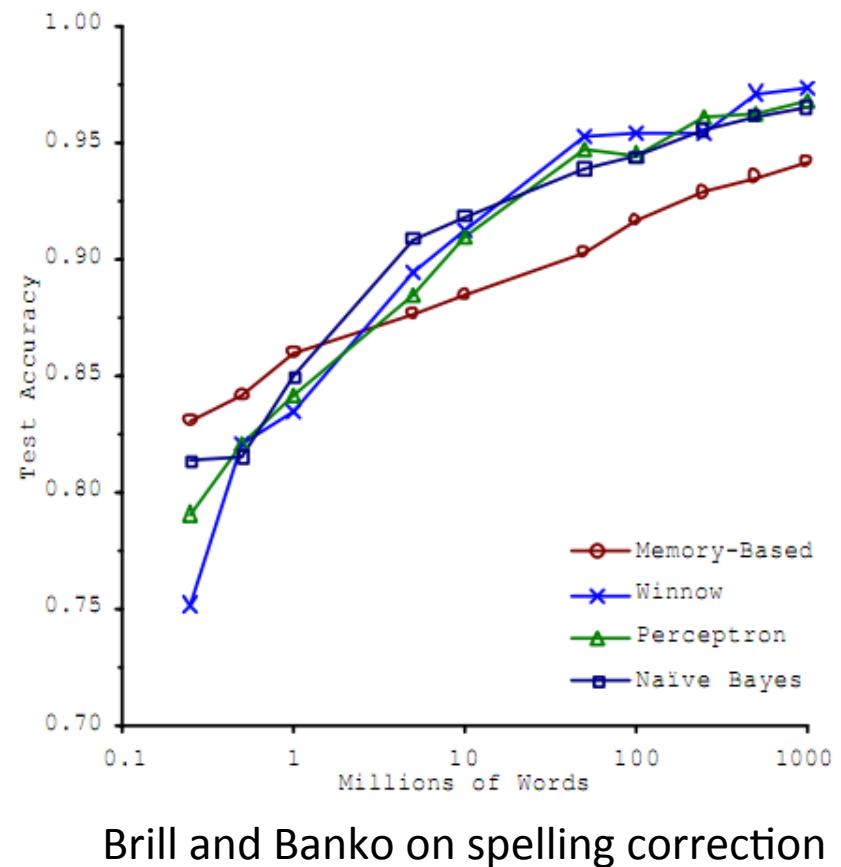
## A huge amount of data?

- Can achieve high accuracy!
- At a cost:
  - SVMs (train time) or kNN (test time) can be too slow
  - Regularized logistic regression can be somewhat better
- So Naïve Bayes can come back into its own again!



# Accuracy as a function of data size

- With enough data
  - Classifier may not matter





## Real-world systems generally combine:

- Automatic classification
- Manual review of uncertain/difficult/"new" cases



## Underflow Prevention: log space

- Multiplying lots of probabilities can result in floating-point underflow.
- Since  $\log(xy) = \log(x) + \log(y)$ 
  - Better to sum logs of probabilities instead of multiplying probabilities.
- Class with highest un-normalized log probability score is still most probable.

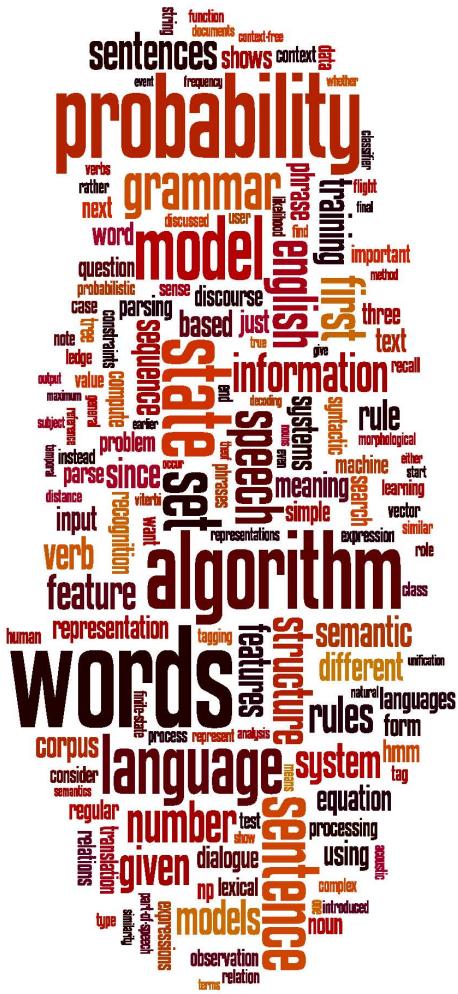
$$c_{NB} = \operatorname{argmax}_{c_j \in C} \log P(c_j) + \sum_{i \in positions} \log P(x_i | c_j)$$

- Model is now just max of sum of weights



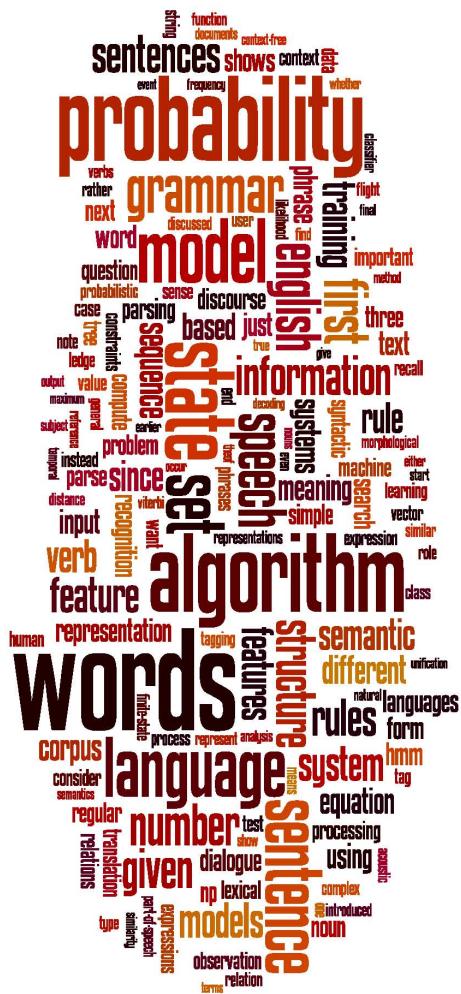
# How to tweak performance

- Domain-specific features and weights: *very* important in real performance
- Sometimes need to collapse terms:
  - Part numbers, chemical formulas, ...
  - But stemming generally doesn't help
- Upweighting: Counting a word as if it occurred twice:
  - title words ([Cohen & Singer 1996](#))
  - first sentence of each paragraph ([Murata, 1999](#))
  - In sentences that contain title words ([Ko et al, 2002](#))



# Text Classification and Naïve Bayes

Text Classification:  
Practical Issues



# Part-of-speech tagging

A simple but useful form of  
linguistic analysis

Christopher Manning



# Parts of Speech

- Perhaps starting with Aristotle in the West (384–322 BCE), there was the idea of having parts of speech
  - a.k.a lexical categories, word classes, “tags”, POS
- It comes from Dionysius Thrax of Alexandria (c. 100 BCE) the idea that is still with us that there are 8 parts of speech
  - But actually his 8 aren't exactly the ones we are taught today
    - Thrax: noun, verb, article, adverb, preposition, conjunction, participle, pronoun
    - School grammar: noun, verb, adjective, adverb, preposition, conjunction, pronoun, interjection

## Open class (lexical) words

### Nouns

#### Proper

*IBM*

*Italy*

#### Common

*cat / cats*

*snow*

### Verbs

#### Main

*see*

*registered*

### Adjectives

*old older oldest*

### Adverbs

*slowly*

### Numbers

*122,312*

*one*

*... more*

## Closed class (functional)

### Determiners

*the some*

### Conjunctions

*and or*

### Pronouns

*he its*

### Modals

*can*

*had*

### Prepositions

*to with*

### Particles

*off up*

*... more*

### Interjections

*Ow Eh*



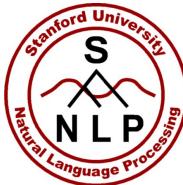
# Open vs. Closed classes

- Open vs. Closed classes
  - Closed:
    - determiners: *a, an, the*
    - pronouns: *she, he, I*
    - prepositions: *on, under, over, near, by, ...*
    - Why “closed”?
  - Open:
    - Nouns, Verbs, Adjectives, Adverbs.



# POS Tagging

- Words often have more than one POS: *back*
  - The *back* door = JJ
  - On my *back* = NN
  - Win the voters *back* = RB
  - Promised to *back* the bill = VB
- The POS tagging problem is to determine the POS tag for a particular instance of a word.



# POS Tagging

- Input: Plays well with others
- Ambiguity: NNS/VBZ UH/JJ/NN/RB IN NNS
- Output: Plays/VBZ well/RB with/IN others/NNS
- Uses:
  - Text-to-speech (how do we pronounce “lead”?)
  - Can write regexps like (Det) Adj\* N+ over the output for phrases, etc.
  - As input to or to speed up a full parser
  - If you know the tag, you can back off to it in other tasks

Penn  
Treebank  
POS tags



# POS tagging performance

- How many tags are correct? (Tag accuracy)
  - About 97% currently
  - But baseline is already 90%
    - Baseline is performance of stupidest possible method
      - Tag every word with its most frequent tag
      - Tag unknown words as nouns
  - Partly easy because
    - Many words are unambiguous
    - You get points for them (*the*, *a*, etc.) and for punctuation marks!



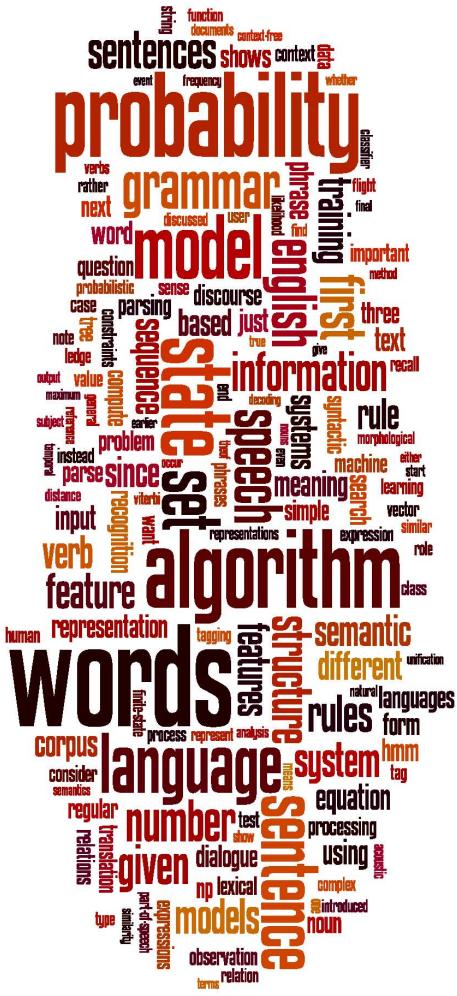
# Deciding on the correct part of speech can be difficult even for people

- Mrs/NNP Shaefer/NNP never/RB got/VBD **around/RP** to/TO joining/VBG
- All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB **around/IN** the/DT corner/NN
- Chateau/NNP Petrus/NNP costs/VBZ **around/RB** 250/CD



# How difficult is POS tagging?

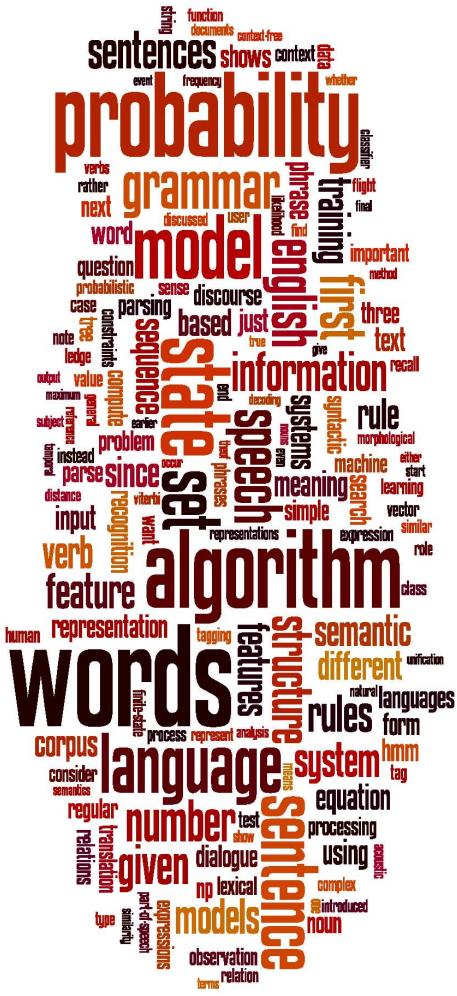
- About 11% of the word types in the Brown corpus are ambiguous with regard to part of speech
- But they tend to be very common words. E.g., *that*
  - I know *that* he is honest = IN
  - Yes, *that* play was nice = DT
  - You can't go *that* far = RB
- 40% of the word tokens are ambiguous



# Part-of-speech tagging

# A simple but useful form of linguistic analysis

# Christopher Manning



# Part-of-speech tagging revisited

# A simple but useful form of linguistic analysis

# Christopher Manning



# Sources of information

- What are the main sources of information for POS tagging?
  - Knowledge of neighboring words
    - Bill saw that man yesterday
    - NNP NN DT NN NN
    - VB VB(D) IN VB NN
  - Knowledge of word probabilities
    - *man* is rarely used as a verb....
- The latter proves the most useful, but the former also helps



# More and Better Features → Feature-based tagger

- Can do surprisingly well just looking at a word by itself:
  - Word                           the: the → DT
  - Lowercased word   Importantly: importantly → RB
  - Prefixes                       unfathomable: un- → JJ
  - Suffixes                       Importantly: -ly → RB
  - Capitalization               Meridian: CAP → NNP
  - Word shapes                  35-year: d-x → JJ
- Then build a maxent (or whatever) model to predict tag
  - Maxent  $P(t|w)$ :    93.7% overall / 82.6% unknown



# Overview: POS Tagging Accuracies

- Rough accuracies:
  - Most freq tag:
  - Trigram HMM:
  - Maxent  $P(t|w)$ :
  - TnT (HMM++):
  - MEMM tagger:
  - Bidirectional dependencies:
  - Upper bound:

~90% / ~50%

~95% / ~55%

93.7% / 82.6%

96.2% / 86.0%

96.9% / 86.9%

97.2% / 90.0%

~98% (human agreement)

Most errors  
on unknown  
words



# How to improve supervised results?

- Build better features!

PRP RB  
VBD IN RB IN PRP VBD .  
They left as soon as he arrived .

- We could fix this with a feature that looked at the next word

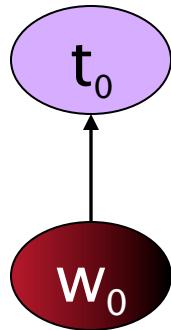
JJ  
NNP NNS VBD VBN .  
Intrinsic flaws remained undetected .

- We could fix this by linking capitalized words to their lowercase versions

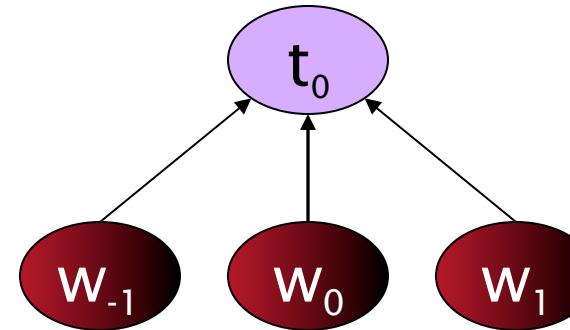


# Tagging Without Sequence Information

Baseline



Three Words



Model	Features	Token	Unknown	Sentence
Baseline	56,805	<b>93.69%</b>	82.61%	26.74%
3Words	239,767	<b>96.57%</b>	86.78%	48.27%

Using words only in a straight classifier works as well as a basic (HMM or discriminative) sequence model!!



# Summary of POS Tagging

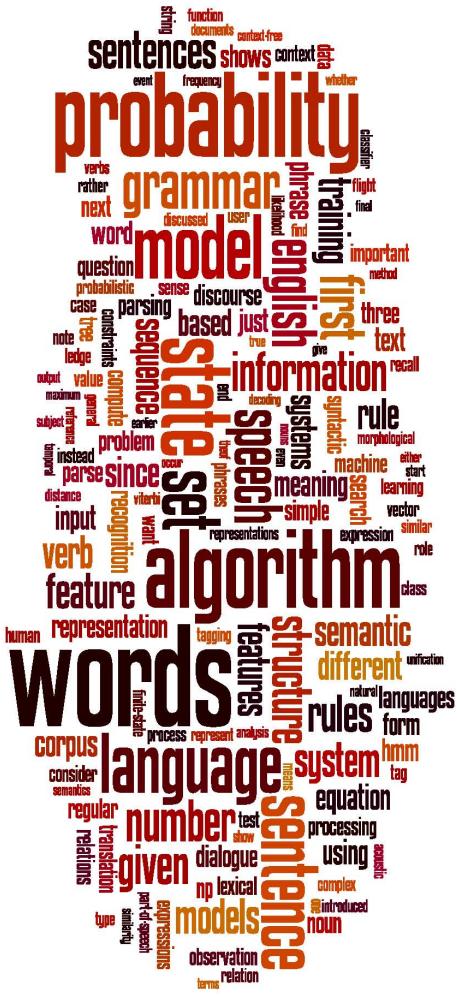
For tagging, the change from generative to discriminative model **does not by itself** result in great improvement

One profits from models for specifying dependence on **overlapping features of the observation** such as spelling, suffix analysis, etc.

An MEMM allows integration of rich features of the observations, but can suffer strongly from assuming independence from following observations; this effect can be relieved by adding dependence on following words

This additional power (of the MEMM ,CRF, Perceptron models) has been shown to result in improvements in accuracy

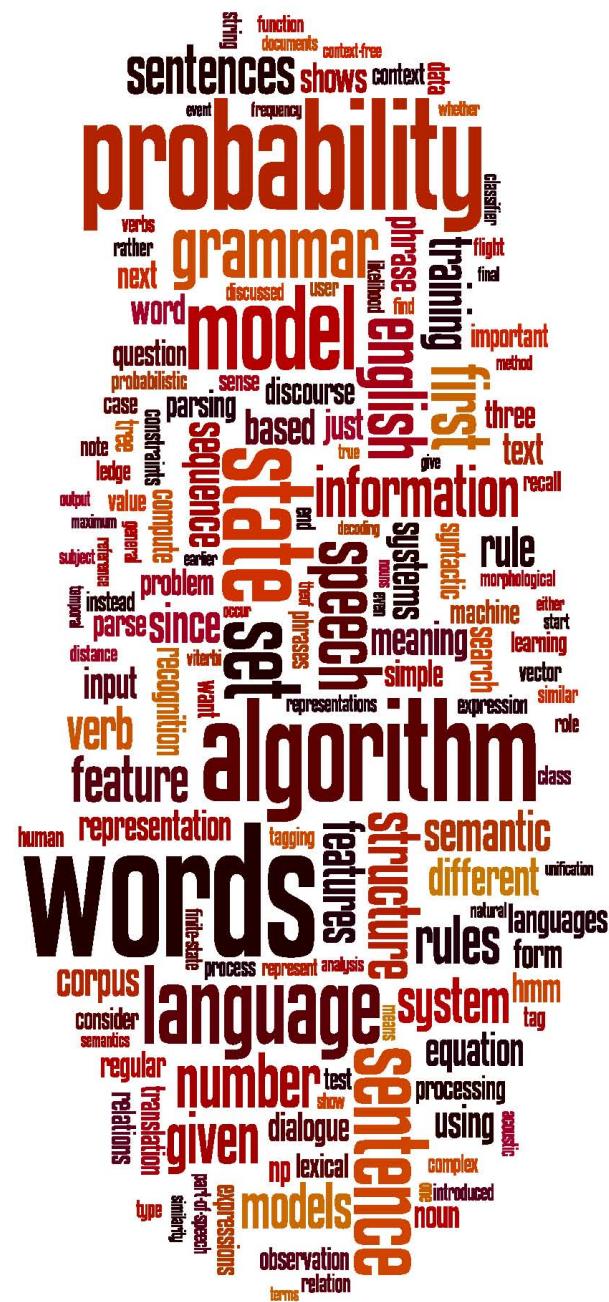
The **higher accuracy** of discriminative models comes at the price of **much slower training**



# Part-of-speech tagging revisited

# A simple but useful form of linguistic analysis

# Christopher Manning



# Statistical Natural Language Parsing

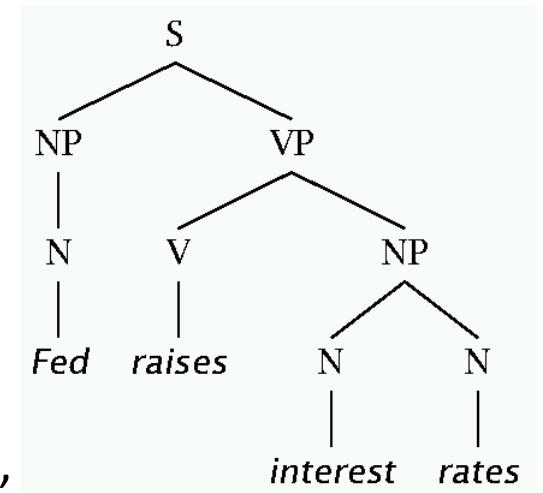
# Two views of syntactic structure



# Two views of linguistic structure:

## 1. Constituency (phrase structure)

- Phrase structure organizes words into nested constituents.
- How do we know what is a **constituent**? (Not that linguists don't argue about some cases.)
  - 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
  - Substitution/expansion/pro-forms:
    - I sat [on the box/right on top of the box/there].
  - Coordination, regular internal structure, no intrusion, fragments, semantics, ...

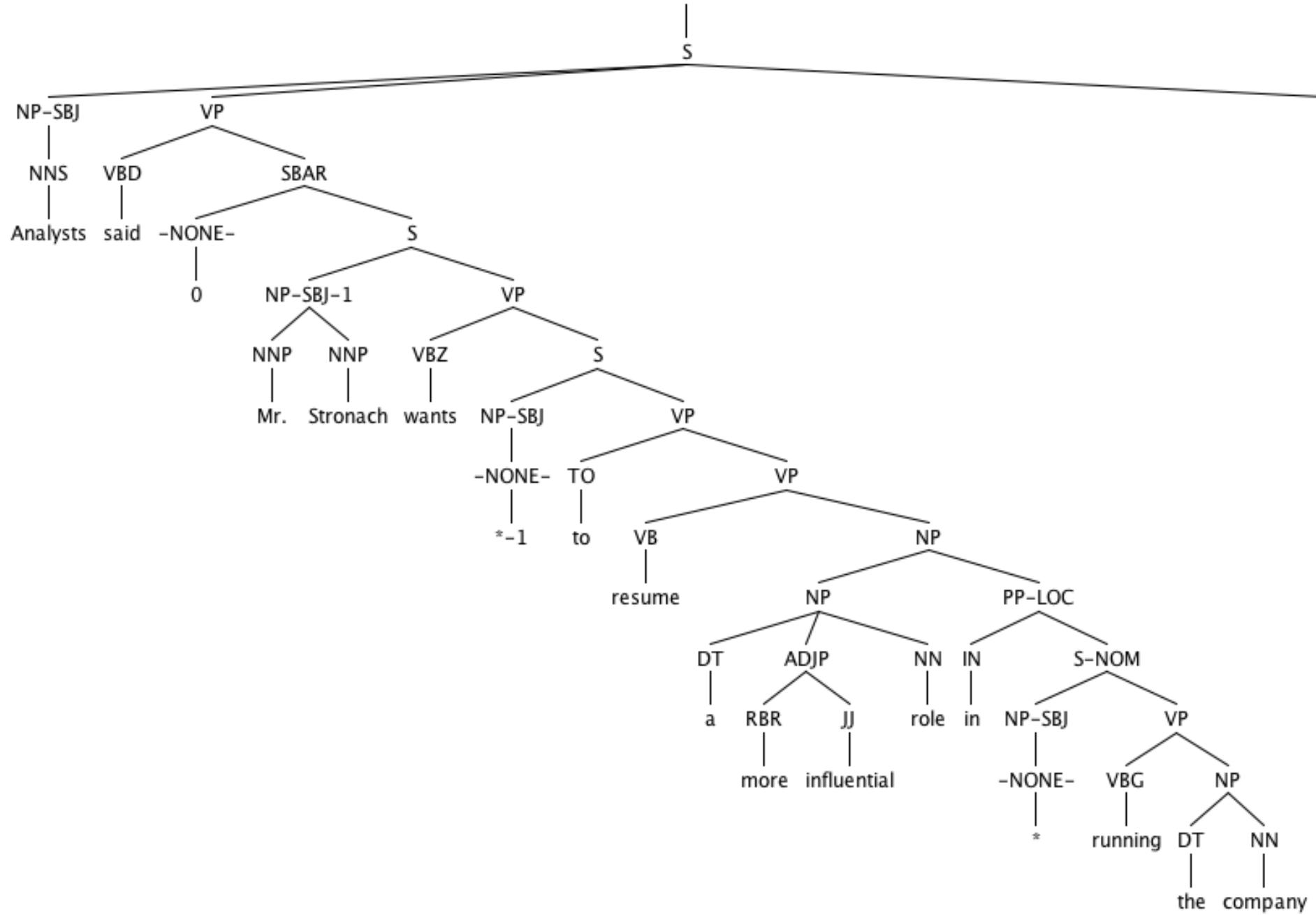




# Two views of linguistic structure:

## 1. Constituency (phrase structure)

- Phrase structure organizes words into nested constituents.
- How do we know what is a **constituent**? (Not that linguists don't argue about some cases.)
  - 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
  - Substitution/expansion/pro-forms:
    - I sat [on the box/right on top of the box/there].
  - Coordination, regular internal structure, no intrusion, fragments, semantics, ...





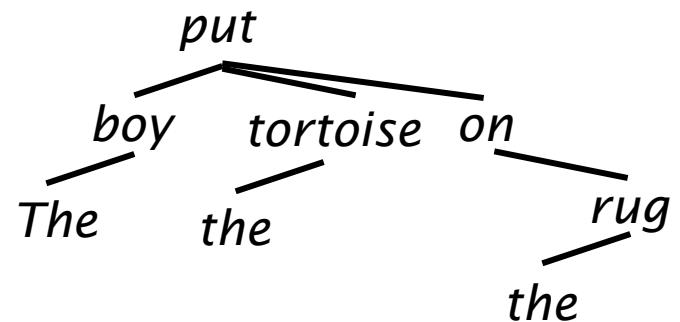
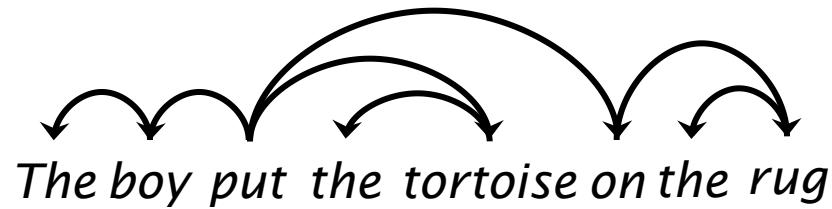
# Headed phrase structure

- VP → ... VB\* ...
- NP → ... NN\* ...
- ADJP → ... JJ\* ...
- ADVP → ... RB\* ...
- SBAR(Q) → S | SINV | SQ → ... NP VP ...
- Plus minor phrase types:
  - QP (quantifier phrase in NP), CONJP (multi word constructions: *as well as*), INTJ (interjections), etc.



## Two views of linguistic structure: 2. Dependency structure

- Dependency structure shows which words depend on (modify or are arguments of) which other words.

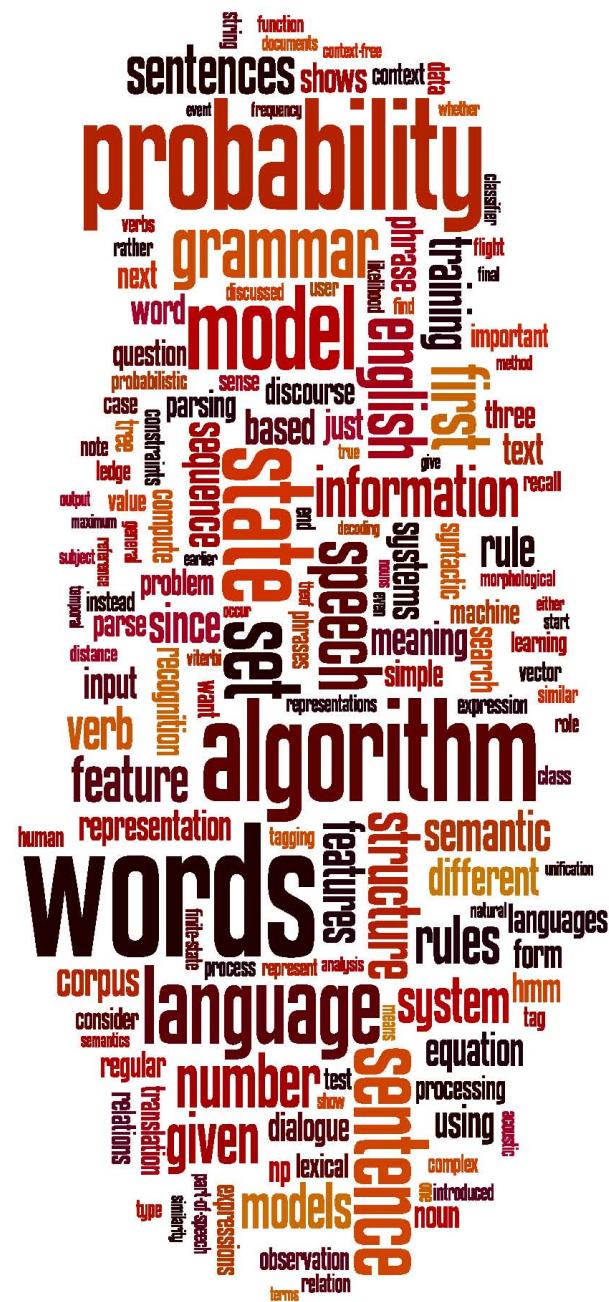




## Two views of linguistic structure: 2. Dependency structure

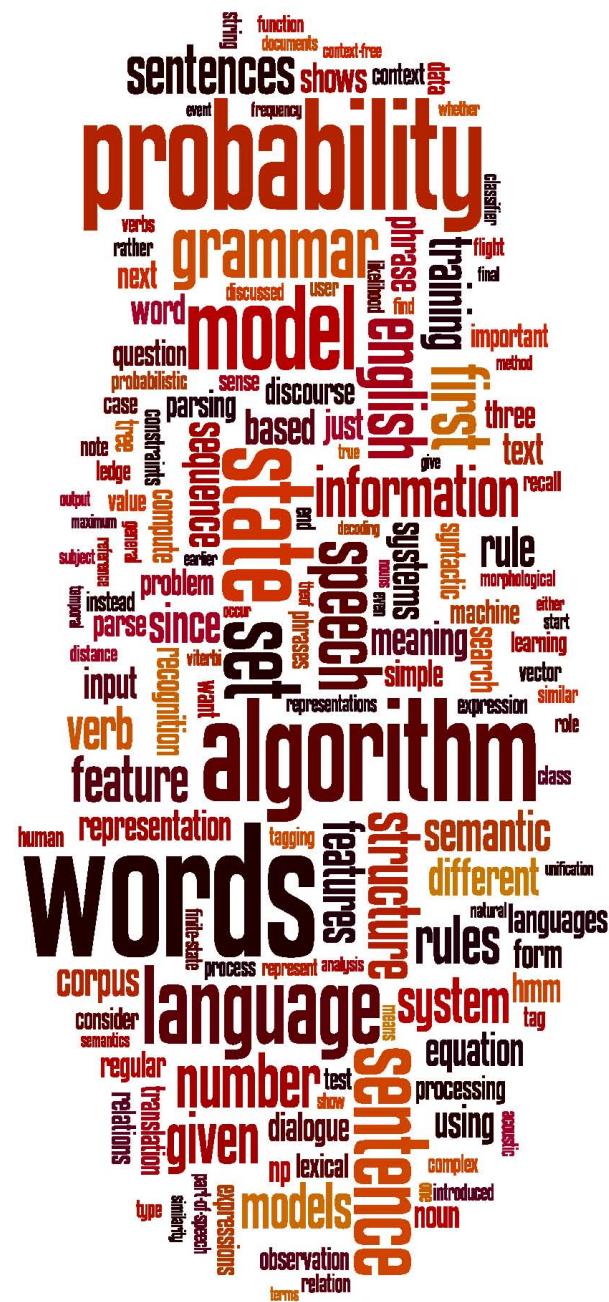
- Dependency structure shows which words depend on (modify or are arguments of) which other words.

*The boy put the tortoise on the rug*



# Statistical Natural Language Parsing

# Two views of syntactic structure



# Statistical Natural Language Parsing

# Parsing: The rise of data and statistics



## Pre 1990 (“Classical”) NLP Parsing

- Wrote symbolic grammar (CFG or often richer) and lexicon
  - $S \rightarrow NP\ VP$
  - $NP \rightarrow (DT)\ NN$
  - $NP \rightarrow NN\ NNS$
  - $NP \rightarrow NNP$
  - $VP \rightarrow V\ NP$
  - $NN \rightarrow interest$
  - $NNS \rightarrow rates$
  - $NNS \rightarrow raises$
  - $VBP \rightarrow interest$
  - $VBZ \rightarrow rates$
- Used grammar/proof systems to prove parses from words
- This scaled very badly and didn’t give coverage. For sentence:

*Fed raises interest rates 0.5% in effort to control inflation*

- Minimal grammar: 36 parses
- Simple 10 rule grammar: 592 parses
- Real-size broad-coverage grammar: millions of parses



# Classical NLP Parsing: The problem and its solution

- Categorical constraints can be added to grammars to limit unlikely/weird parses for sentences
  - But the attempt make the grammars not robust
    - In traditional systems, commonly 30% of sentences in even an edited text would have *no* parse.
- A less constrained grammar can parse more sentences
  - But simple sentences end up with ever more parses with no way to choose between them
- We need mechanisms that allow us to find the most likely parse(s) for a sentence
  - Statistical parsing lets us work with very loose grammars that admit millions of parses for sentences but still quickly find the best parse(s)



# The rise of annotated data: The Penn Treebank

[Marcus et al. 1993, *Computational Linguistics*]

```
( (S
  (NP-SBJ (DT The) (NN move))
  (VP (VBD followed)
    (NP
      (NP (DT a) (NN round))
      (PP (IN of)
        (NP
          (NP (JJ similar) (NNS increases))
          (PP (IN by)
            (NP (JJ other) (NNS lenders)))
          (PP (IN against)
            (NP (NNP Arizona) (JJ real) (NN estate) (NNS loans)))))))
    (,,)
  (S-ADV
    (NP-SBJ (-NONE- *))
    (VP (VBG reflecting)
      (NP
        (NP (DT a) (VBG continuing) (NN decline))
        (PP-LOC (IN in)
          (NP (DT that) (NN market)))))))
  (. .)))
```



# The rise of annotated data

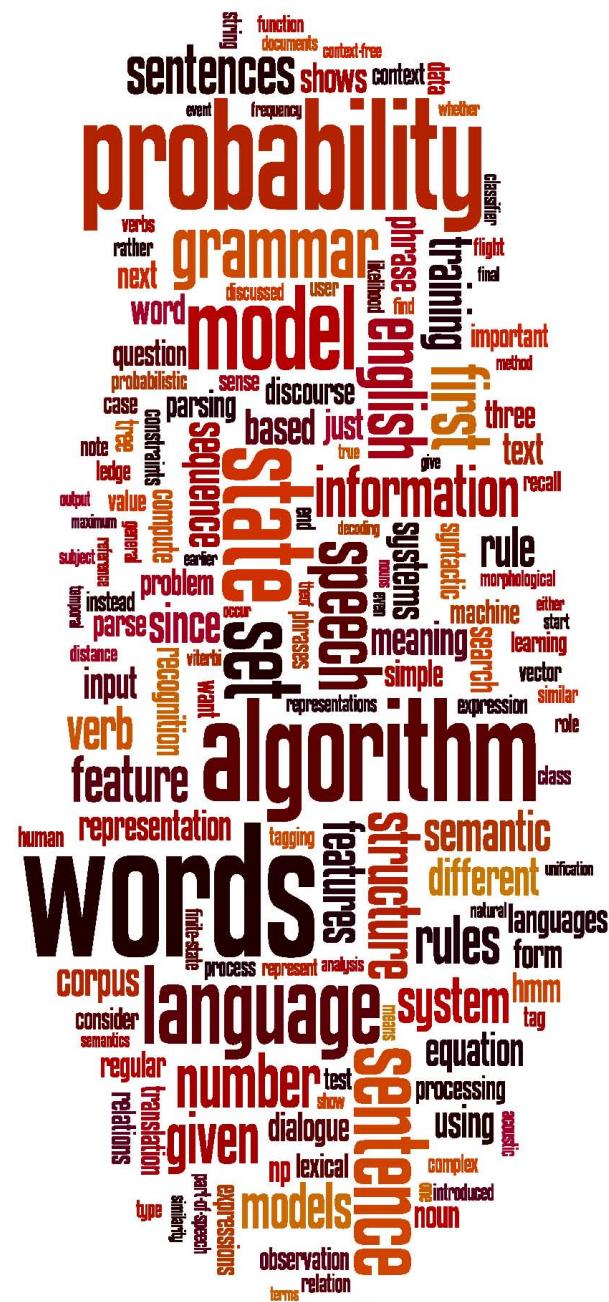
- Starting off, building a treebank seems a lot slower and less useful than building a grammar
- But a treebank gives us many things
  - Reusability of the labor
    - Many parsers, POS taggers, etc.
    - Valuable resource for linguistics
  - Broad coverage
  - Frequencies and distributional information
  - A way to evaluate systems



# Statistical parsing applications

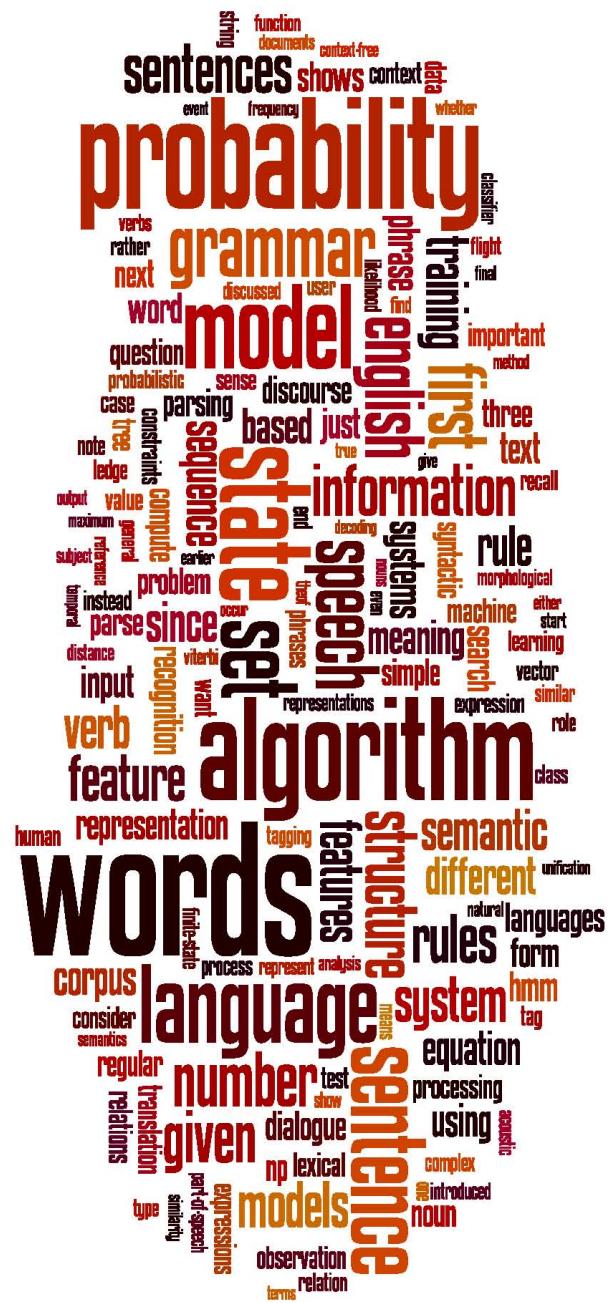
Statistical parsers are now robust and widely used in larger NLP applications:

- High precision question answering [Pasca and Harabagiu SIGIR 2001]
- Improving biological named entity finding [Finkel et al. JNLPBA 2004]
- Syntactically based sentence compression [Lin and Wilbur 2007]
- Extracting opinions about products [Bloom et al. NAACL 2007]
- Improved interaction in computer games [Gorniak and Roy 2005]
- Helping linguists find data [Resnik et al. BLS 2005]
- Source sentence analysis for machine translation [Xu et al. 2009]
- Relation extraction systems [Fundel et al. *Bioinformatics* 2006]



# Statistical Natural Language Parsing

# Parsing: The rise of data and statistics



# Statistical Natural Language Parsing

An exponential  
number of  
attachments



# Attachment ambiguities

- A key parsing decision is how we ‘attach’ various constituents
  - PPs, adverbial or participial phrases, infinitives, coordinations, etc.

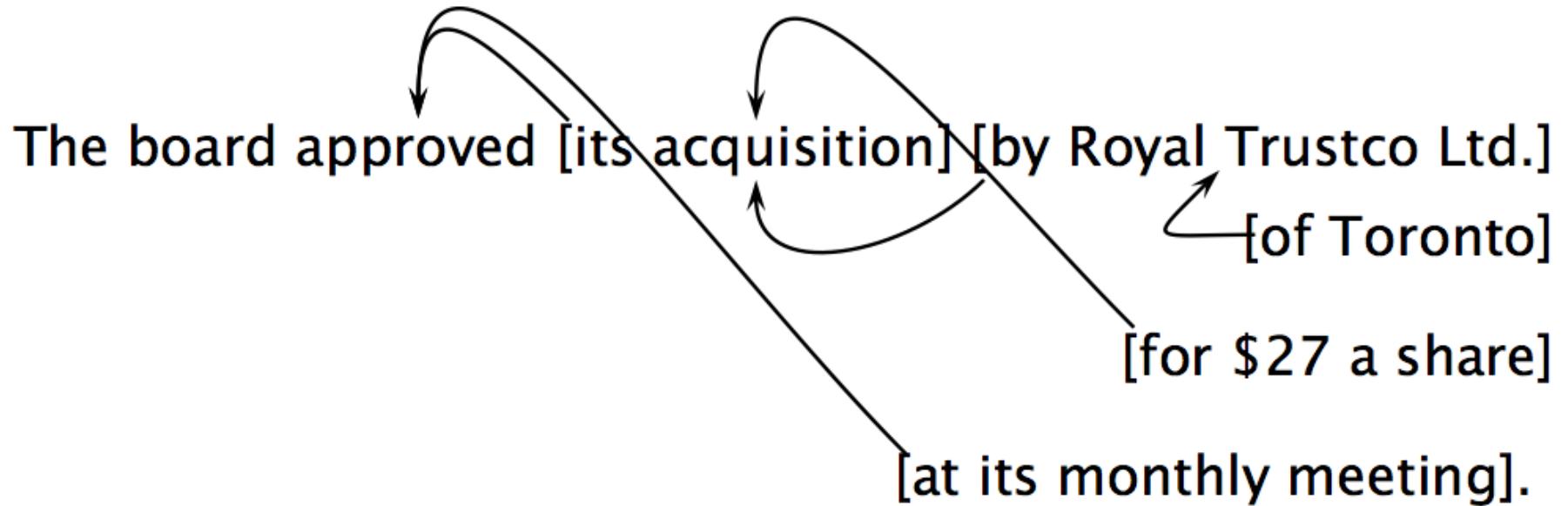
The board approved [its acquisition] [by Royal Trustco Ltd.]  
[of Toronto]  
[for \$27 a share]  
[at its monthly meeting].

- Catalan numbers:  $C_n = (2n)!/[(n+1)!n!]$
- An exponentially growing series, which arises in many tree-like contexts:
  - E.g., the number of possible triangulations of a polygon with  $n+2$  sides
  - Turns up in triangulation of probabilistic graphical models....



# Attachment ambiguities

- A key parsing decision is how we ‘attach’ various constituents
  - PPs, adverbial or participial phrases, infinitives, coordinations, etc.



- Catalan numbers:  $C_n = (2n)!/[(n+1)!n!]$
- An exponentially growing series, which arises in many tree-like contexts:
  - E.g., the number of possible triangulations of a polygon with  $n+2$  sides
  - Turns up in triangulation of probabilistic graphical models....



# Quiz Question!

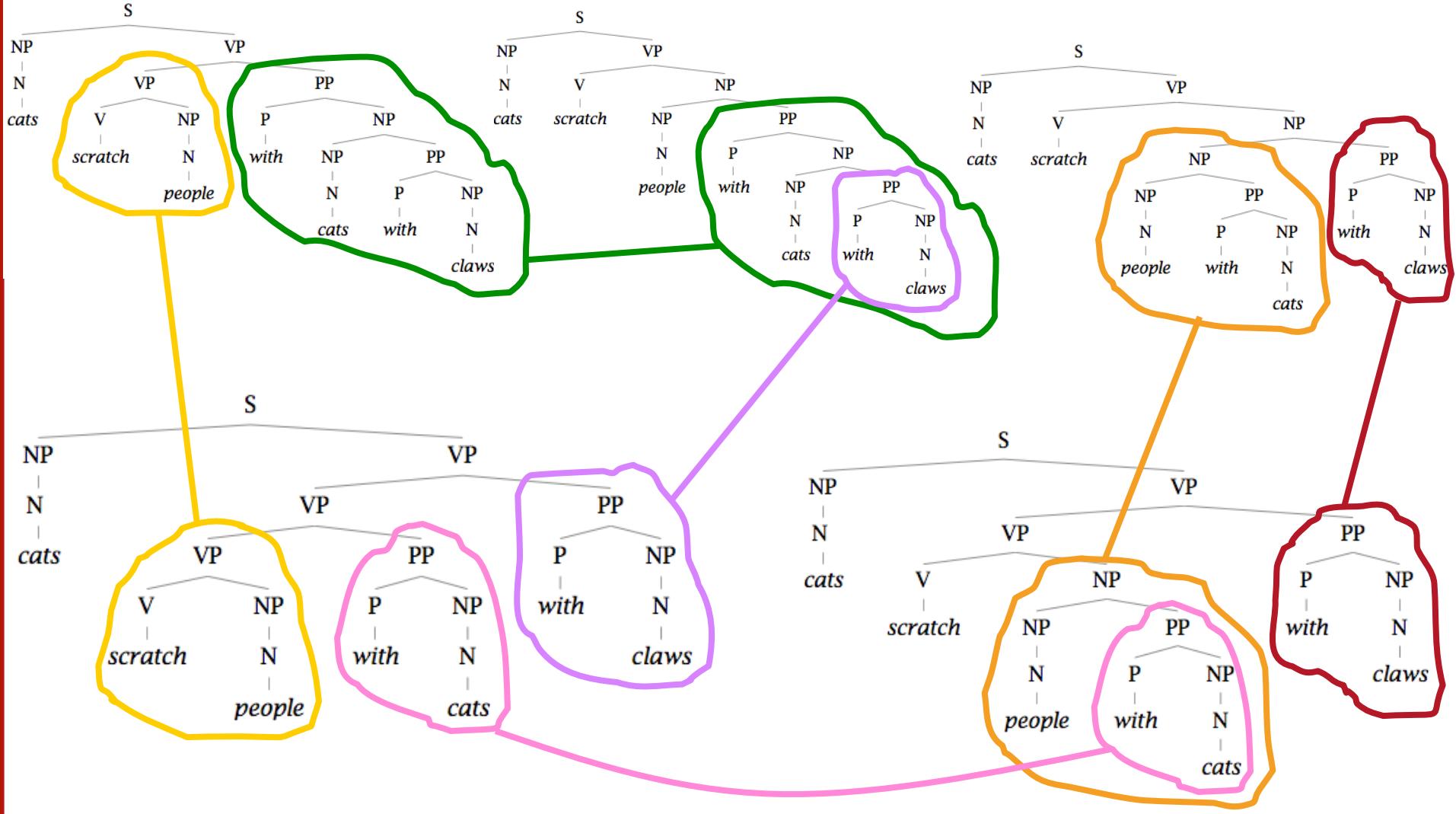
- How many distinct parses does the following sentence have due to PP attachment ambiguities?
  - A PP can attach to any preceding V or N within the verb phrase, subject only to the parse still being a tree.
    - (This is equivalent to there being no crossing dependencies, where if  $d_2$  is a dependent of  $d_1$  and  $d_3$  is a dependent of  $d_2$ , then the line  $d_2-d_3$  begins at  $d_2$  under the line from  $d_1$  to  $d_2$ .)

John wrote the book with a pen in the room.



# Two problems to solve:

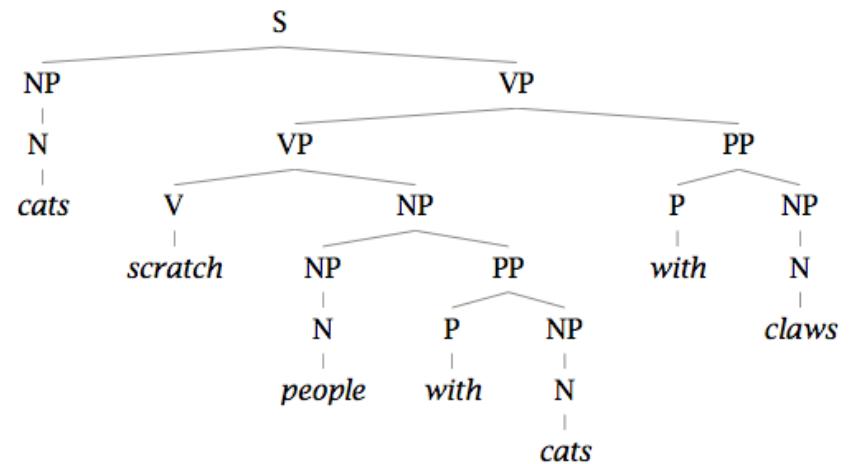
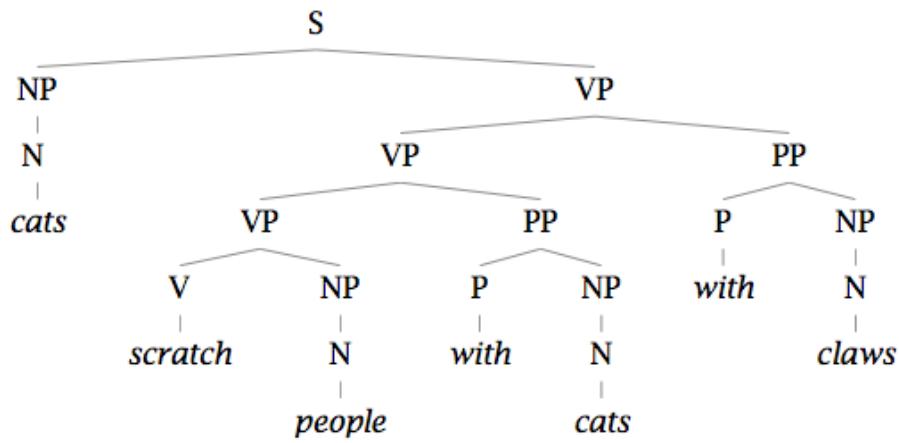
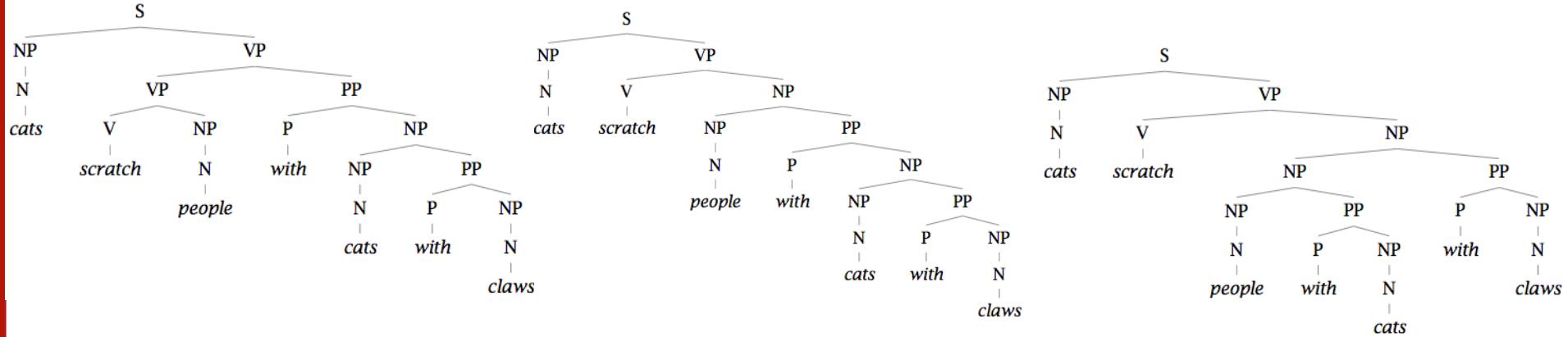
## 1. Repeated work...





# Two problems to solve:

## 1. Repeated work...

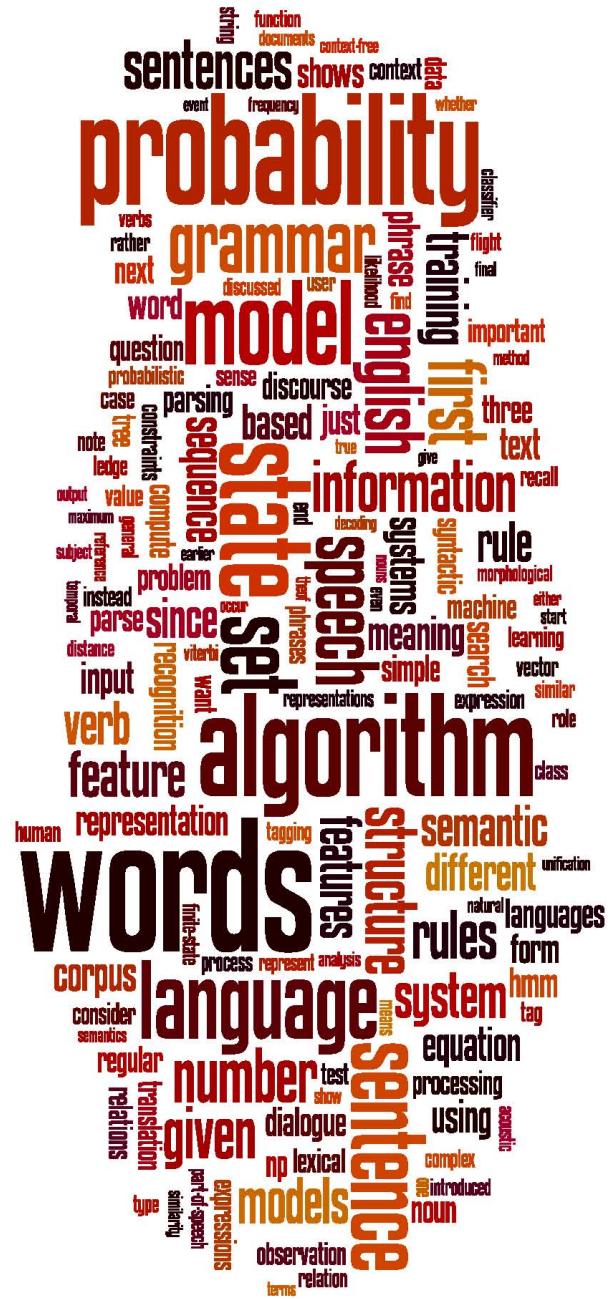




## Two problems to solve:

### 2. Choosing the correct parse

- How do we work out the correct attachment:
  - She saw the man with a telescope
  - Is the problem ‘AI complete’? Yes, but ...
  - Words are good predictors of attachment
    - Even absent full understanding
    - Moscow sent more than 100,000 soldiers into Afghanistan ...
    - Sydney Water breached an agreement with NSW Health ...
- Our statistical parsers will try to exploit such statistics.



# Statistical Natural Language Parsing

An exponential  
number of  
attachments

Welcome to:

## Classical Approaches of NLP



# Unit objectives

**After completing this unit, you should be able to:**

- Understand various concepts of database
- Gain knowledge on importance of structure of database
- Learn about various components of database
- Gain an insight into insurance database models

# Course description

- Purpose: To learn about CL & NLP. The various methodologies used in NLP along with their implantation. To understand the process of the various aspects of NLP, its applications in real life retrieval and extraction of information.
- Audience: Interested participants in acquiring more idea on Linguistics and concepts of NLP. The audience are expected to posses some interest in data processing.
- Perquisite: Should have a pre-requisite knowledge on Python programming language to implement the code processing of NLP.
- Course objectives: Learn about Text Processing, Lexical Analysis, Syntactic Parsing, Semantic Analysis,

Natural Language Generation

# Introduction (1 of 2)

- Computational linguistics:
  - Computational linguistics is an interdisciplinary field.
  - Concerned with the statistical or rule-based modeling.
  - Theoretical part is more relevant to the basic knowledge and foundation of linguistics.
  - Understanding the grammar of the languages and the morphology.

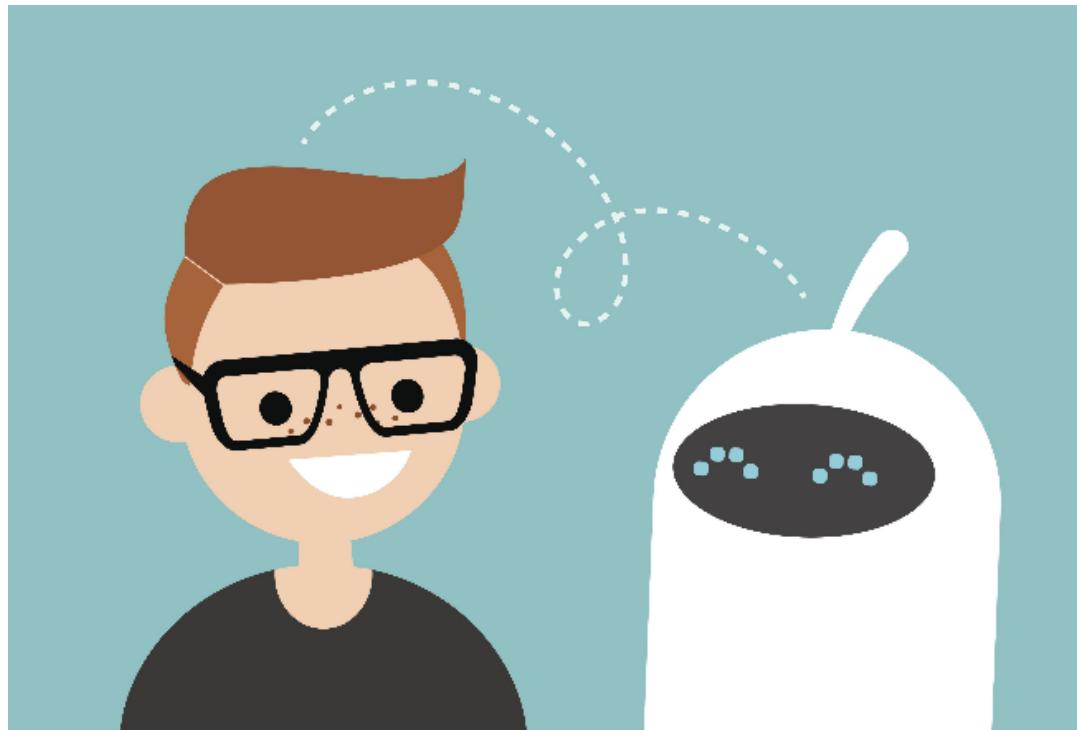


Figure: Computational Linguistics

Source: <https://chatbotsmagazine.com/how-computational-linguists-help-your-chatbot-understand-humans-8ec95ab903f6>

# Introduction (2 of 2)

- Natural language processing
  - Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, how to program computers to process and analyze large amounts of natural language data.
  - Sixth Sense activities
  - Speech recognition
  - Language understanding
  - Language generation
  - Techniques
  - Tagging
  - Hidden Markov models
  - Decision trees
  - Probabilistic values
  - Language models
  - Speech recognition

# Classical approaches to natural language processing



IBM ICE (Innovation Centre for Education)

- Goals of computational linguistics:
  - Formulation of grammatical framework which can check the semantics of the language.
  - Formulated by implementing syntactic and semantic analysis.
- Growth of computational linguistics:
  - System that can comprehend, recognize speech, tagging and parsing activities.
  - Neural network approach for the activity.
  - Good in the processing power.
  - Process information faster.
  - Both syntactic and semantic processing methodologies.
  - Extraction of clusters from any language, identifying the relational tuples, paraphrase sets are all important aspects of text corpora.
  - Computational linguistics and natural language processing algorithms are implemented in machine learning.

# Approaches to natural language processing (1 of 2)



IBM ICE (Innovation Centre for Education)

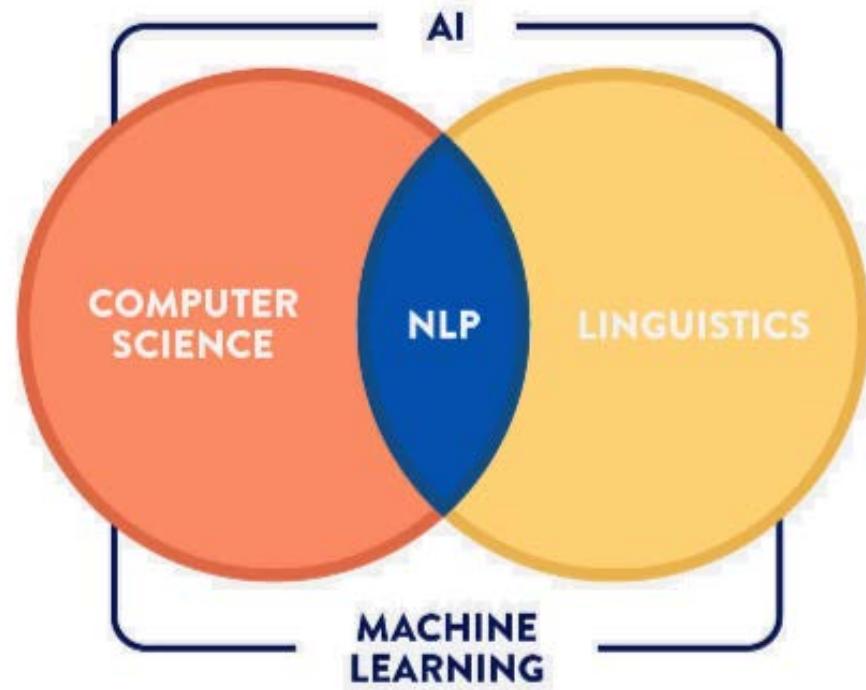
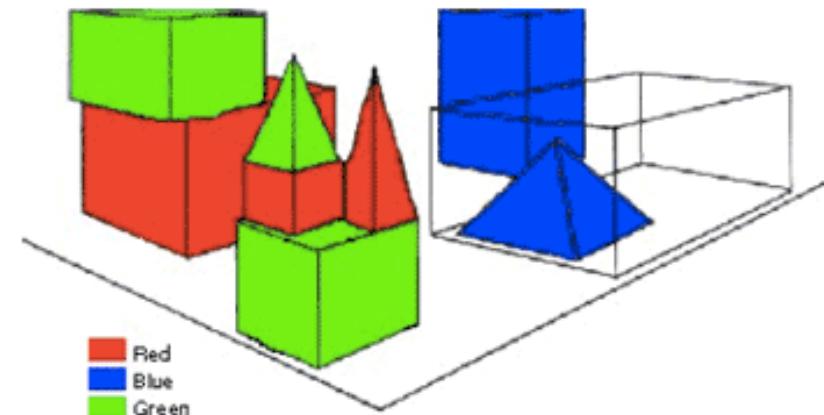


Figure: Approaches to natural language processing

Source: <https://towardsdatascience.com/introduction-to-natural-language-processing-nlp-323cc007df3d>



Person: Pick up a big red block.

Computer: OK.

Person: Grasp the pyramid.

Computer: I don't understand which pyramid you mean.

Figure: Approaches to natural language processing

Source: <https://www.topbots.com/4-different-approaches-natural-language-processing-understanding/>

# Approaches to natural language processing (2 of 2)



IBM ICE (Innovation Centre for Education)

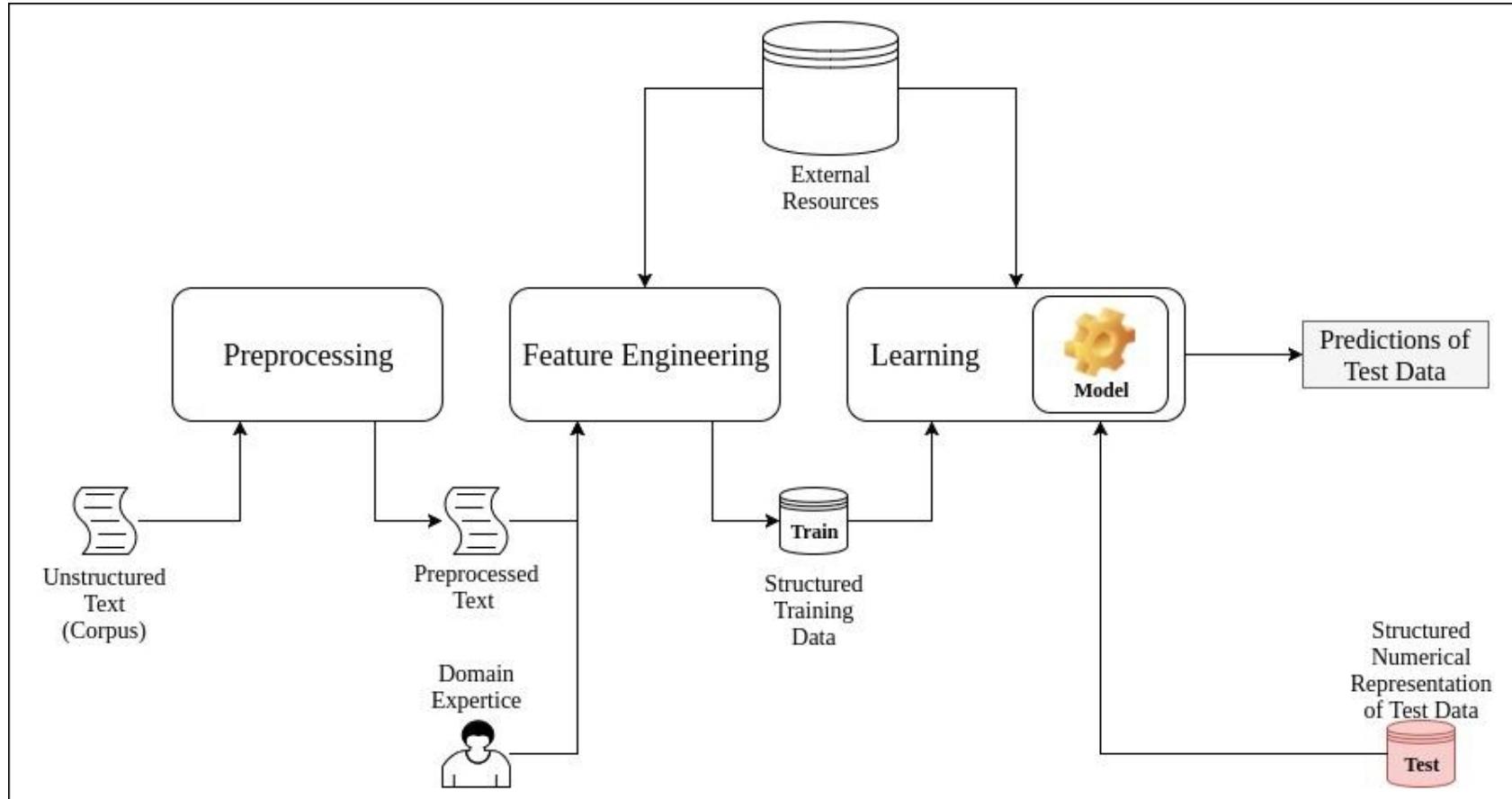


Figure: Shows the Classical approach to Natural Language Processing

Source: [https://subscription.packtpub.com/book/application\\_development/9781788478311/1/ch01lvl1sec12/the-traditional-approach-to-natural-language-processing](https://subscription.packtpub.com/book/application_development/9781788478311/1/ch01lvl1sec12/the-traditional-approach-to-natural-language-processing)

# Understanding linguistics

- Basic ideology starts by making the computer identify the different stages of learning a natural language.

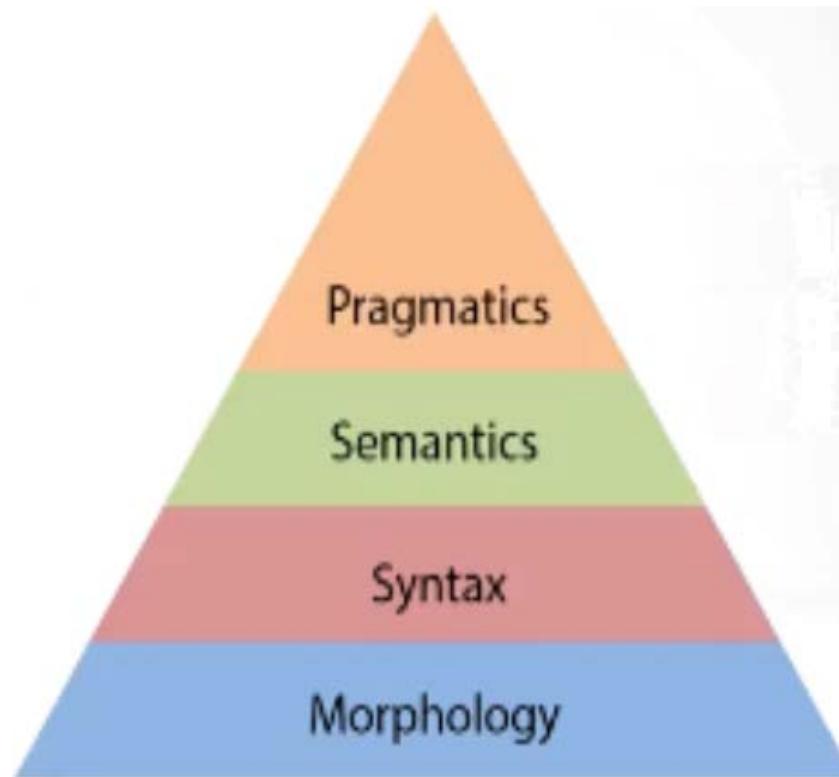


Figure: Major levels in linguistics analysis

Source: <https://towardsdatascience.com/linguistic-knowledge-in-natural-language-processing-332630f43ce1>

# Level 1: Morphology

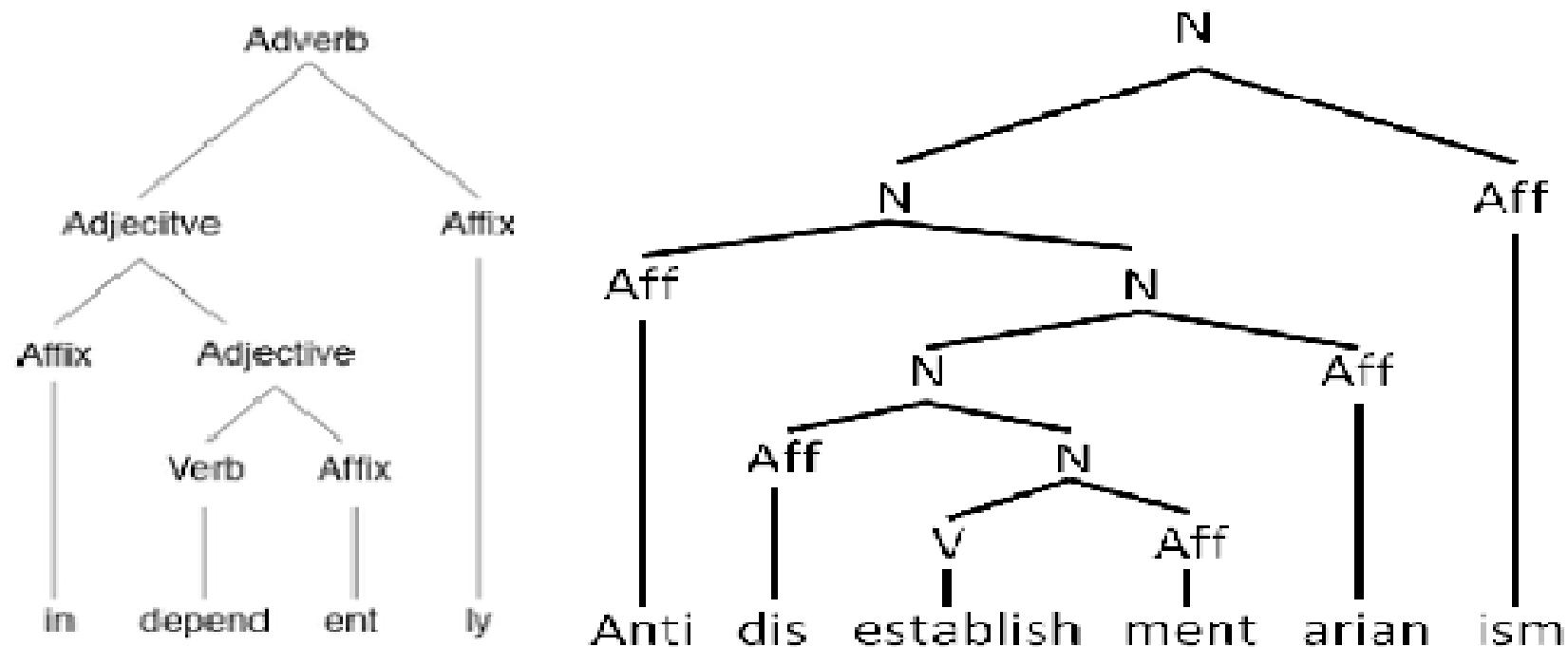


Figure: Morphology makes up the basic constructs

Source: [https://en.wikipedia.org/wiki/Morphology\\_\(linguistics\)](https://en.wikipedia.org/wiki/Morphology_(linguistics))

# Level 2: Syntax

- Syntax follows the grammar of the language.

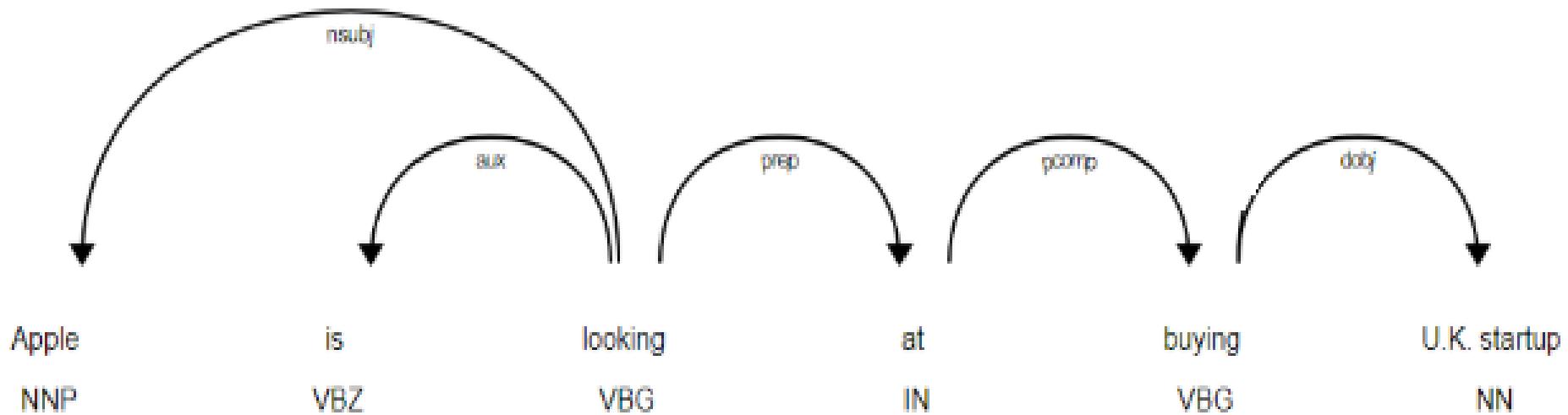


Figure: Syntactic analysis is done at the statement level

Source: <https://all-about-linguistics.group.shef.ac.uk/branches-of-linguistics/morphology/what-is-morphology/>

# Level 3: Semantics

- Deals with the meaning conveyed by creating sentences in that language.
- Semantics include tasks like named entity recognition and relationship extraction.



Figure: Semantics

Source: <https://ai.googleblog.com/2018/05/advances-in-semantic-textual-similarity.html>

# Level 4: Pragmatics

- Understanding the sentences created and to understand the conveyed meanings.
- Common problems that are associated with pragmatics:
  - Co-Referencing
  - Summarization
  - Modelling.
  - Question and answering.

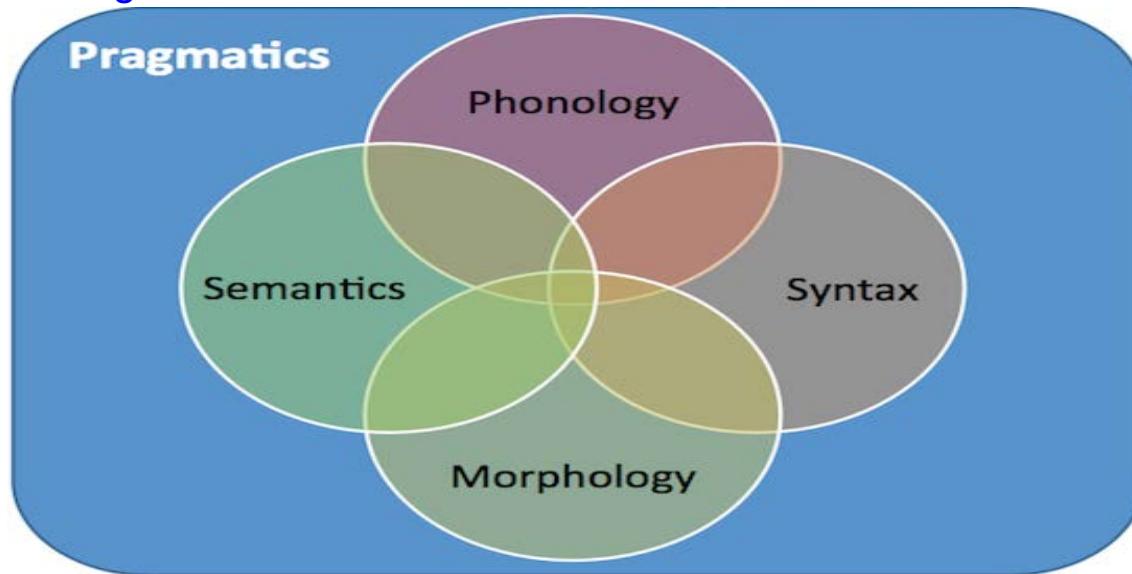


Figure: Pragmatics

Source: <https://medium.com/@paulomalvar/pragmatics-the-last-frontier-9d64351eea6f>

# Understanding linguistics

- The syntax and semantics go hand in hand and framing a sentence:
  - Hyponymy is used to convey the relationship between a general term and a specific instance. (Crocodile is an amphibian).
  - Meronymy used to convey that one part of a sentence is a part of another(fish has gills).

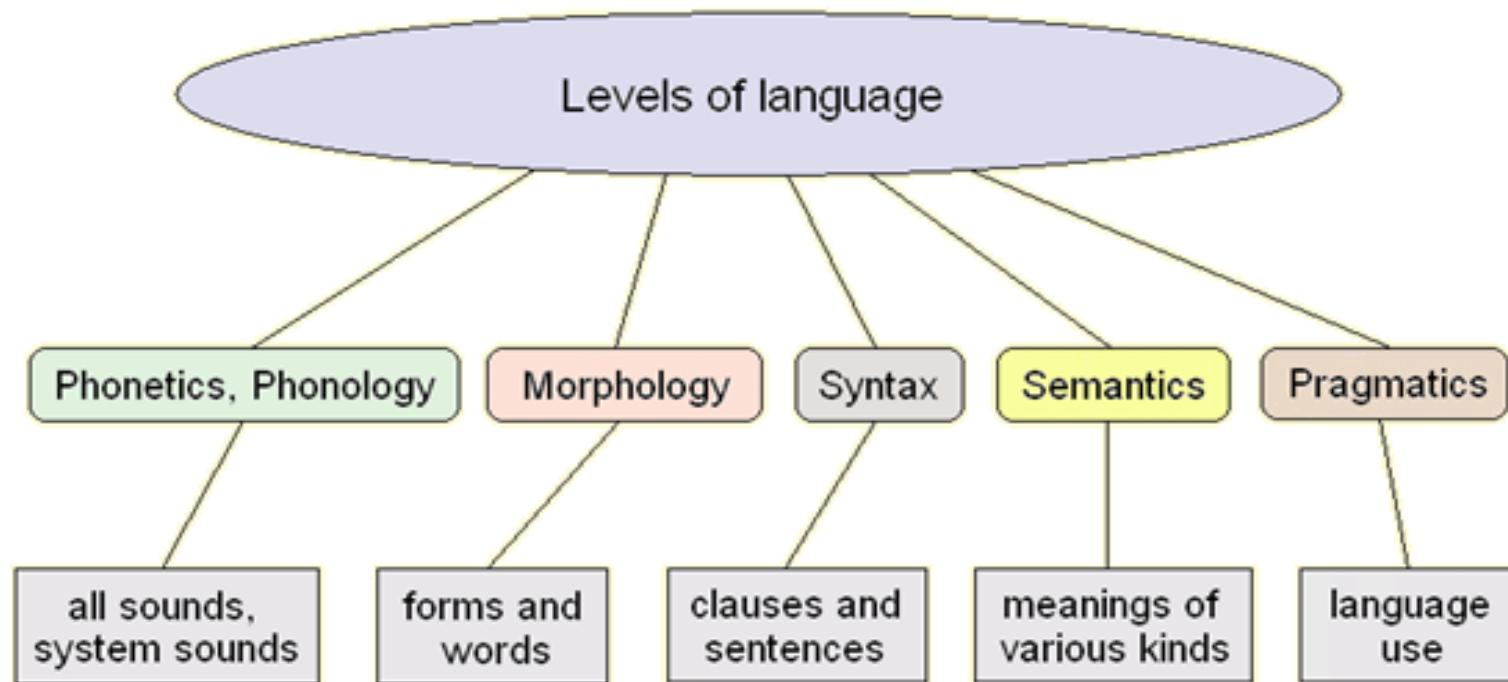


Figure: Linguistics

Source: [https://www.uni-due.de/SHE/REV\\_Levels\\_Chart.htm](https://www.uni-due.de/SHE/REV_Levels_Chart.htm)

# Traditional approach (1 of 2)

- Processing is considered as a sequence of steps.
- Separate and distinct processes take place.
- Preprocessing: Removal of unwanted data.
- Feature engineering: Understanding the numeral representation of the textual data.
- Machine learning algorithms: Learning the language using the training data.
- Predicting outputs: Identify the prediction with test data.

# Traditional approach (2 of 2)

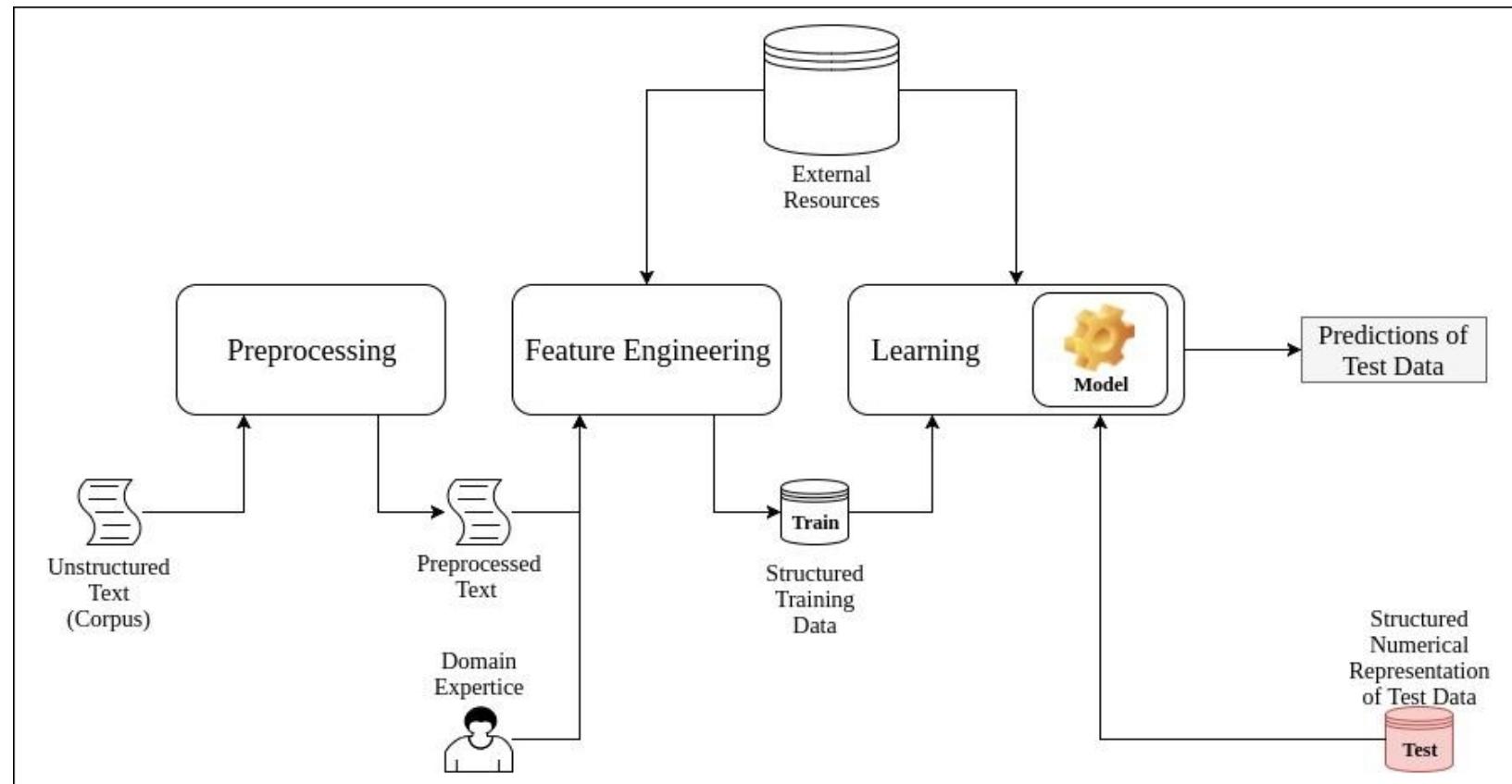


Figure: Traditional approach

Source: [https://subscription.packtpub.com/book/application\\_development/9781788478311/1/ch01lvl1sec12/the-traditional-approach-to-natural-language-processing](https://subscription.packtpub.com/book/application_development/9781788478311/1/ch01lvl1sec12/the-traditional-approach-to-natural-language-processing)

# Example: Automatic summarization using NLP



IBM ICE (Innovation Centre for Education)

- During a game let the NLP activity be automatic generation of the summary of the game. Will have multiple sets of statistics like scoring, penalties etc.
- The data collected contains the most relevant sentence to create the summary for every statistical parameter. Natural language processing algorithm should now create a summary of the game.
- Pre-processing steps:
  - Stemming: Choosing root verbs.
  - Removing distractions: Eliminating the punctuation.
  - Tokenization: Identifying simple words.

# Drawbacks

- Loss of information.
- Lengthy and tedious process.
- Domain expertise.
- External resources.
- Identification of external resource.

# Text processing

- Theory and practice of automating the creation or manipulation of electronic text.
- Text: Alphanumeric characters specified on the keyboard.
- Processing: Automated or mechanized processing.
- Representation of data:
  - Text
  - Images
  - Audio
  - Videos
- Analyzing the data which may be structured or unstructured to obtain structured information.

# What Is text processing?

- The textual information: Processed, analyzed and manipulated-machines learn.
- Text extraction and text classification.
- Extracting individual and small bits of information from large text data is called as text extraction.
- Assigning values to the text data depending upon the content is called as text classification.

The ultimate measure of a man is not where he stands in moments of comfort and convenience, but where he stands at times of challenge and controversy. The true neighbor will risk his position, his prestige, and even his life for the welfare of others. What he is, All he is, Who he is.

Figure: Text processing

Source: <https://www.wcpss.net/Page/8911>

# Text analysis vs. Text mining vs. Text analytics



IBM ICE (Innovation Centre for Education)

- Used to obtain data by statistical pattern learning.
  - Both text analysis and text mining are qualitative processes.
  - Text Analytics is quantitative process.
- 
- **Example:**
    - Banking service: Customer satisfaction.
    - Text analysis: Individual performance of the customer support executive. Text used in the feedback like "good", "bad".
    - Text analytics:
      - Overall performance of all the support executives.
      - Graph for visualizing the performance of the entire support team.
    - Text analytics for overall count of issues resolved.

# Tools and methodologies: Statistical methods



IBM ICE (Innovation Centre for Education)

- Statistical methods:
  - Word frequency: Identify the most regularly used expressions or words that are present in a specific text.
  - Collocation: Method for identifying the common words that appear together.
  - Concordance: Methodology to provide context to the natural language.
  - TF-IDF: Identifies the importance of words in a document.

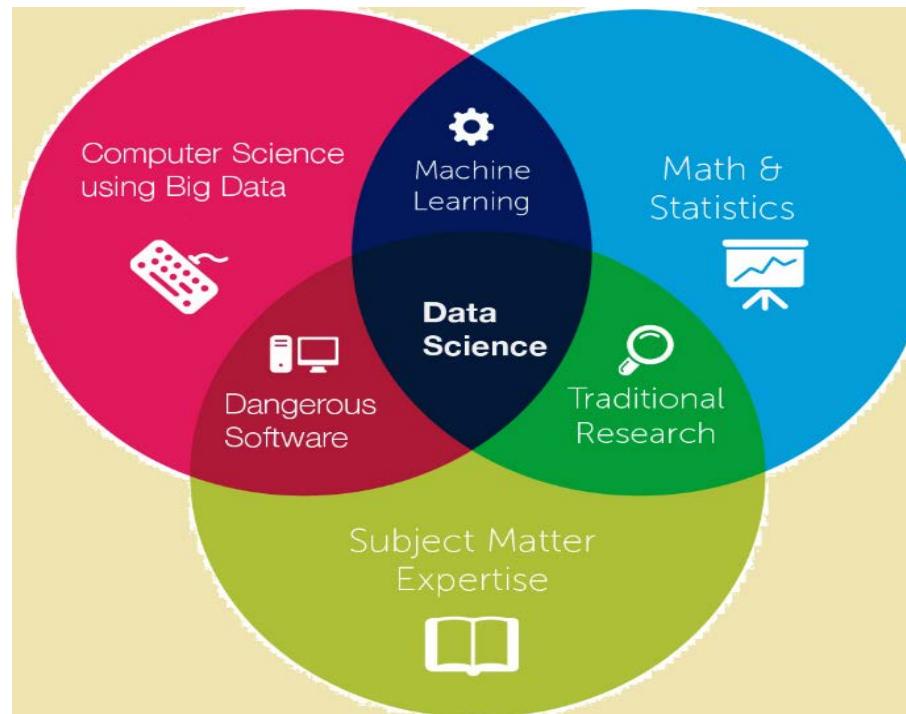


Figure: Statistical methods

Source: <http://grjenkin.com/articles/category/data-science/106322/big-data-data-science-and-machine-learning-explained>

# Tools and methodologies: Text classification (1 of 2)

- Text classification:
  - Content is analyzed and classified into multiple predefined groups based upon the analysis.

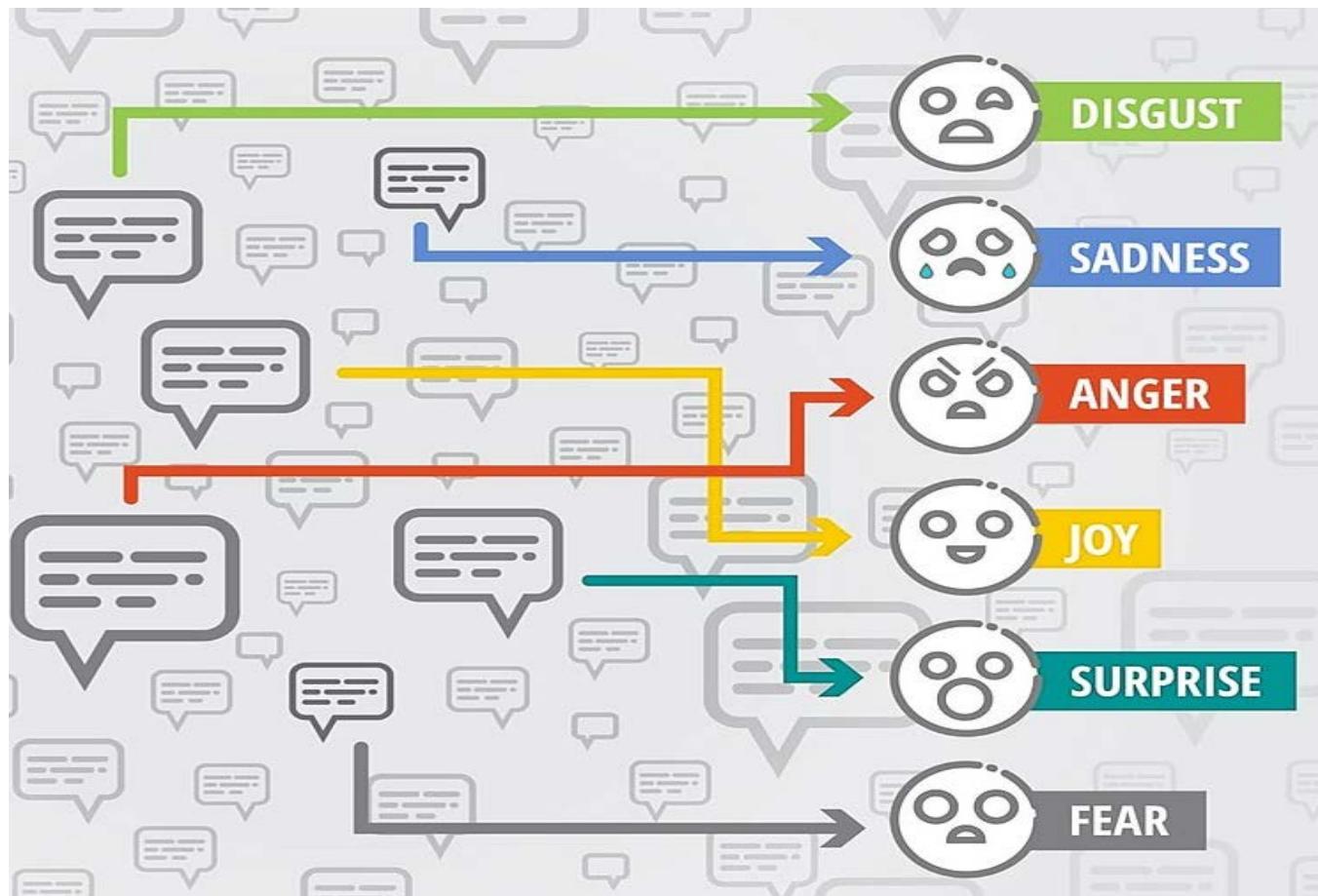


Figure: Text classification

Source: <https://hackernoon.com/text-classification-simplified-with-facebooks-fasttext-b9d3022ac9cb>

# Tools and methodologies: Text classification (2 of 2)



IBM ICE (Innovation Centre for Education)

- Topic analysis: Identify and interpret large collection of text according to the individual topics assigned.
- Sentiment analysis: Understanding the emotional feel represent in a textual message.

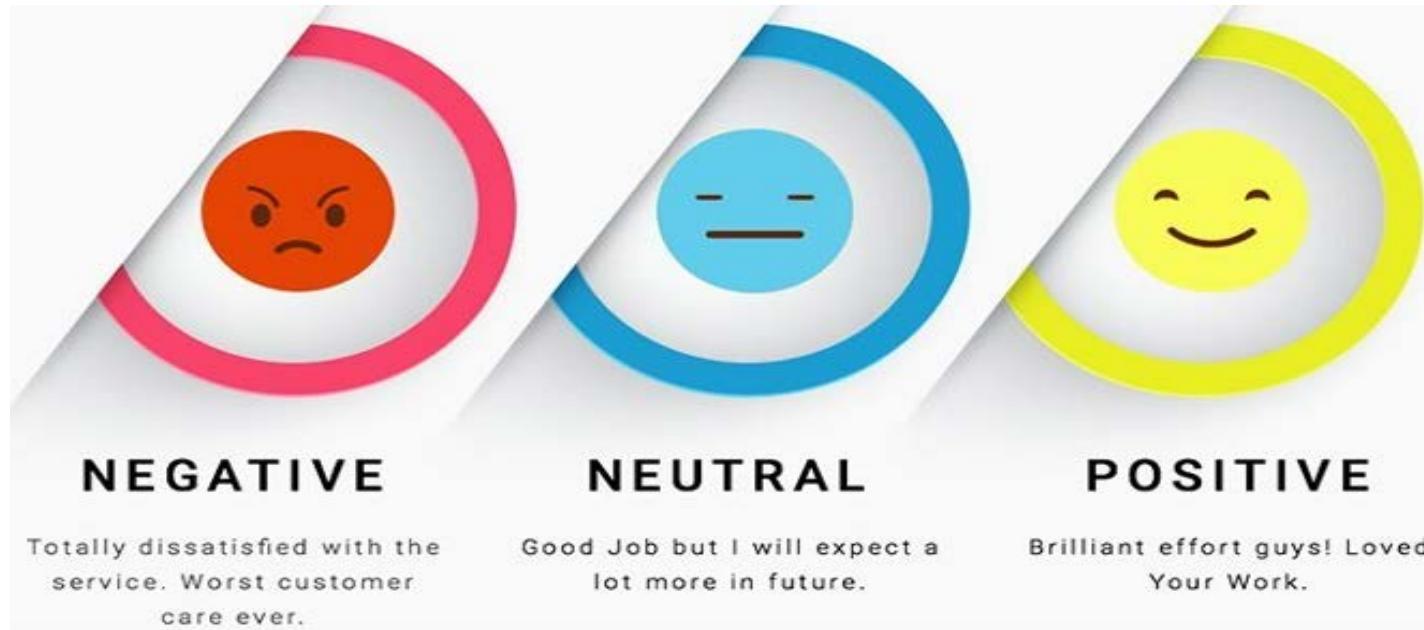


Figure: Language classification

Source: <https://www.kdnuggets.com/2018/03/5-things-sentiment-analysis-classification.html>

# Tools and methodologies: Text extraction



IBM ICE (Innovation Centre for Education)

- Text extraction: Process of gathering valuable pieces of information present within the text data.

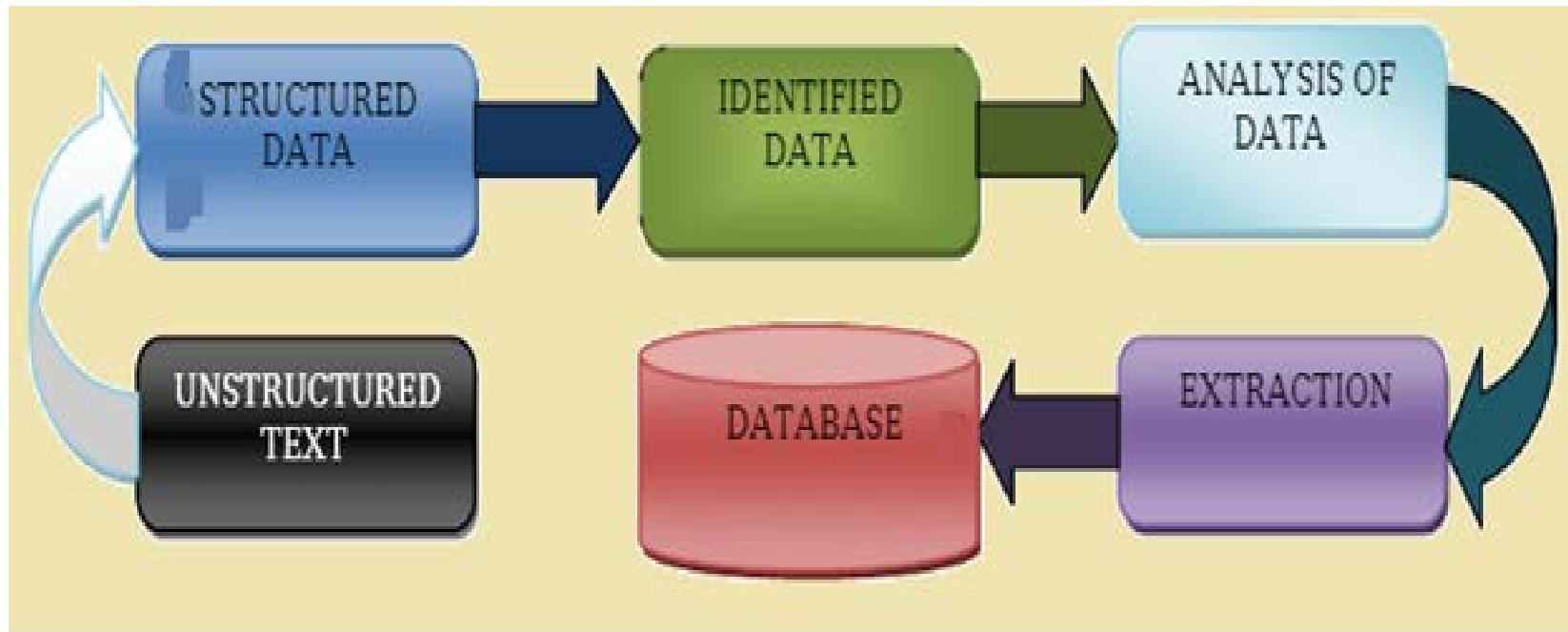


Figure: Text extraction

Source: <https://www.upgrad.com/blog/what-is-text-mining-techniques-and-applications/>

- Keyword extraction: Identifying and detecting the most relevant of the words inside a text.
- Entity extraction: Useful for gathering information on specific relevant elements and to discard all the other irrelevant elements.

# Tools and methodologies: Example

- Search engines use clustering.
- Web pages: Set of similar words are grouped and clustered.
- Pulling up the pages: High count of words relevant.

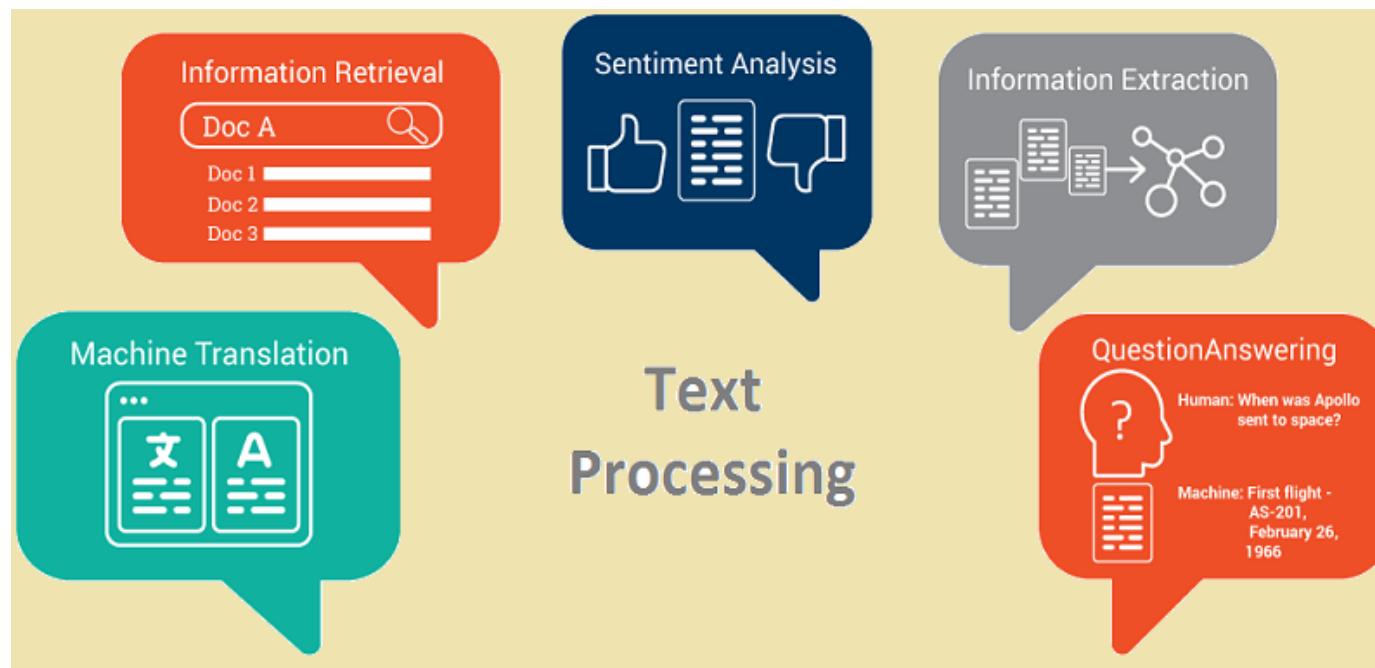


Figure: Test processing

Source: <https://towardsdatascience.com/machine-learning-text-processing-1d5a2d638958>

# Scope of text analysis/processing

- Large documents
  - Refer for a context
  - Cross examine multiple documents
- Individual sentences
  - Gathering specific information
  - Identify the emotional or intentional activities.
- Parts of the sentences
  - Sentiments of the words can be analyzed
  - Better understanding of the natural language
  - Provided for machine to analyze and understand

# Importance of text analysis

- Business growth:
  - Extraction of information to identify the customer
- Real time analysis:
  - Urgent requirements or complaint handled on a real-time basis.
  - Categorized as priority
  - Require multiple analysis
- Checking for consistency:
  - Detect latest models
  - Analyzing
  - Understanding
  - Sharing of the available data accurately

# Working principles of text analysis

- Data gathering
- Data preparation
- Data analysis

# Data gathering (1 of 2)

- Text analysis: Gathering the required data that need to be analyzed.
- Internal data:
  - Email
  - Chat messages
  - CRM tools
  - Databases
  - Surveys.
  - Spreadsheets
  - Product analysis report

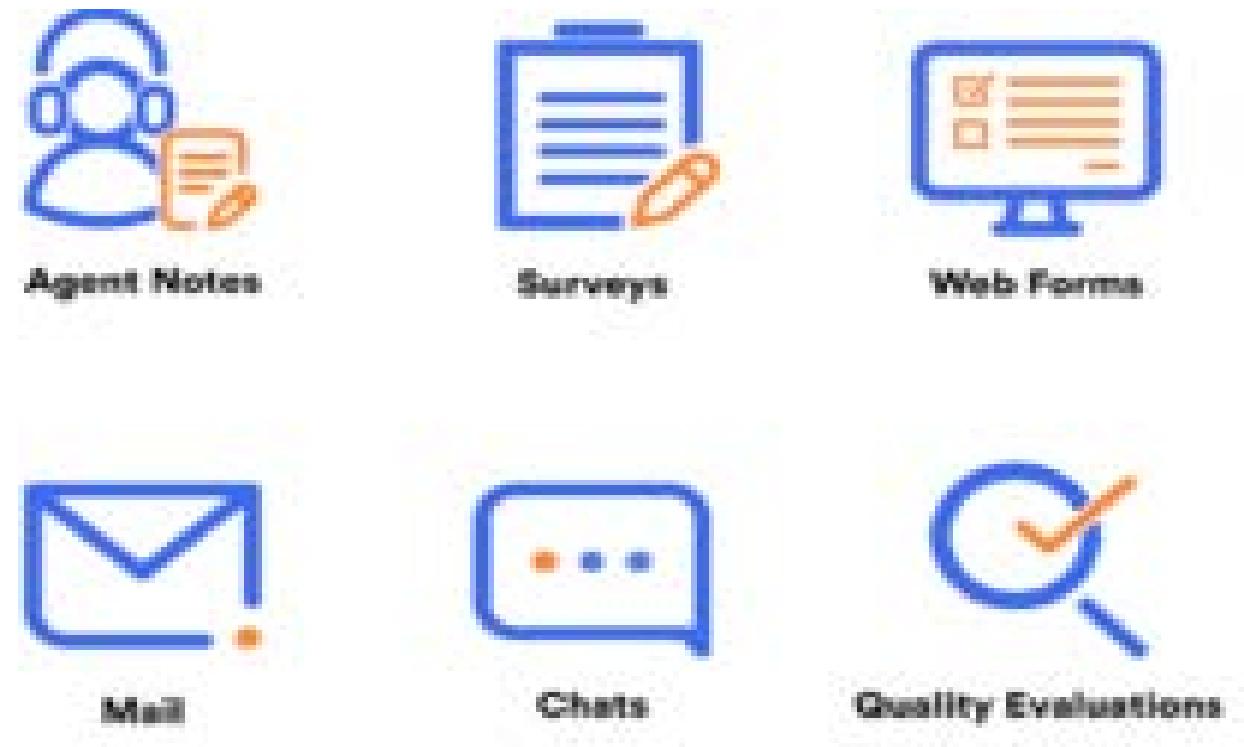


Figure: Text analysis

Source: <https://voziq.com/customer-retention/improving-customer-retention-strategies-with-unstructured-customer-data/attachment/common-sources-of-unstructured-data/>

# Data gathering (2 of 2)

- External data: The external data do not belong to the organization and are available freely through other sources.
- Web scraping tools.
- Open data.

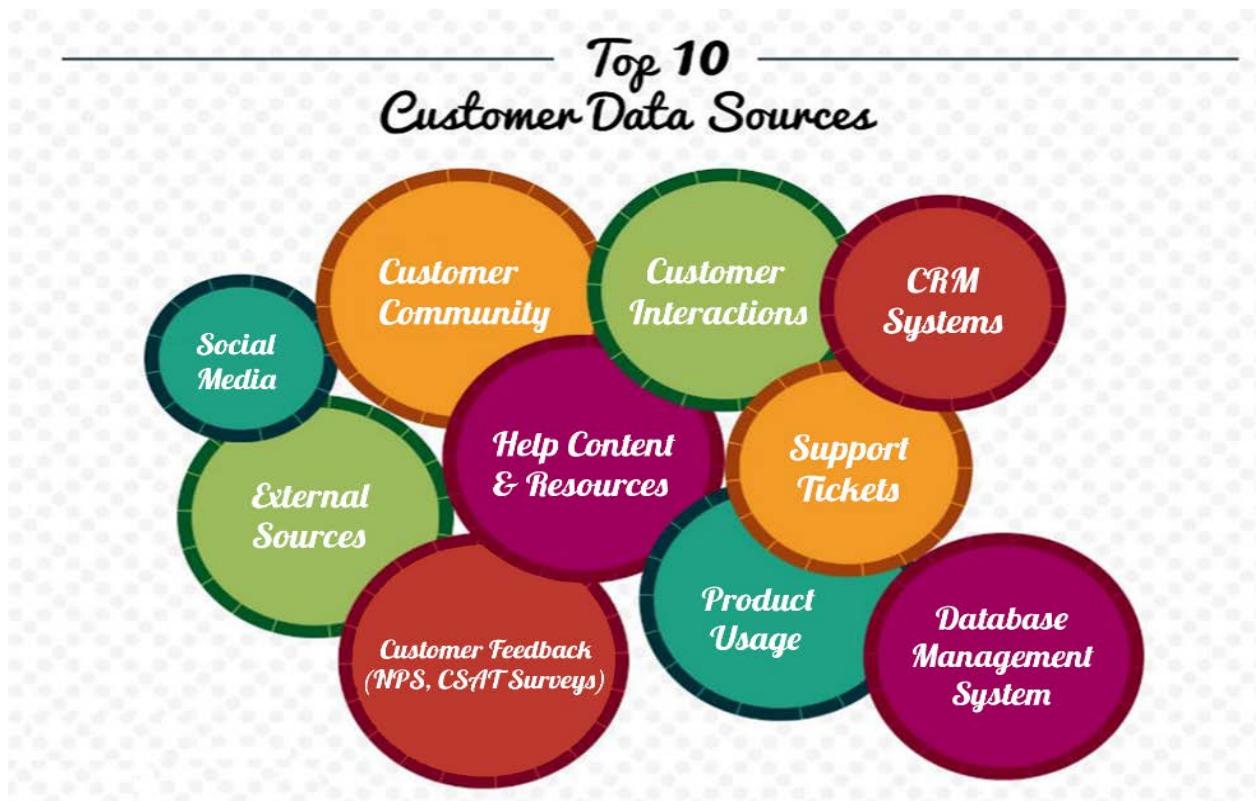


Figure: Web scraping tools

Source: <https://strikedeck.com/top-10-customer-data-sources/>

# Data preparation (1 of 2)

- Before text is analyzed by any machine learning algorithm, it needs to be prepared.
- Tokenization
  - Identify and recognize the unit of text
  - Process of breaking up text characters into meaningful elements
  - Analyze the meaningful parts of the text and discarding the meaningless sections.
  - Removes all the frequent words that can be found in a sentence
- Stemming
  - Used to reduces a word to its root to convey meaning.
  - Unnecessary character removal like prefix, suffix etc.
- Lemmatization
  - Identify parts of the speech not needed and removes the inflection.

# Data preparation (2 of 2)

- Constituency parsing
  - Uses syntactic structures: abstract notes associated to words and abstract categories.

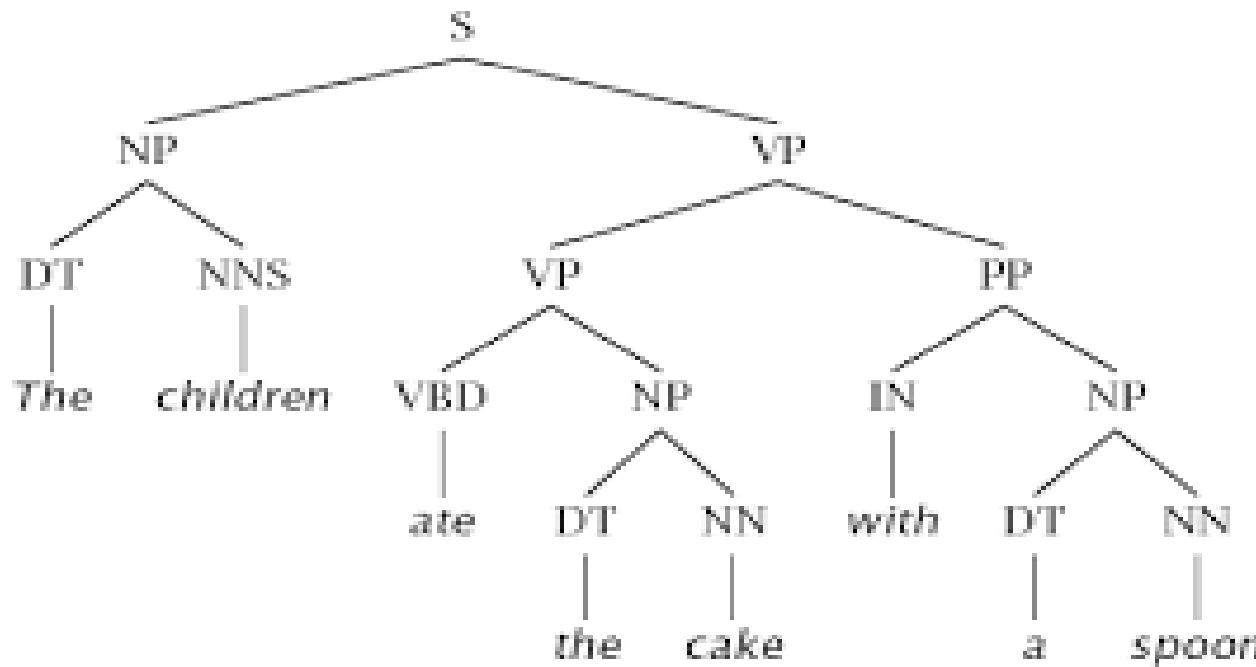


Figure: Constituency parsing

Source: <http://www.cs.cornell.edu/courses/cs5740/2017sp/lectures/13-parsing-const.pdf>

# Data preparation steps

- Step 1: Recognize the Tokens:

- "The sky is blue"
- Token 1:["The S","ky is", " blue"] Token 2:["The", "sky", "is", "blue"]

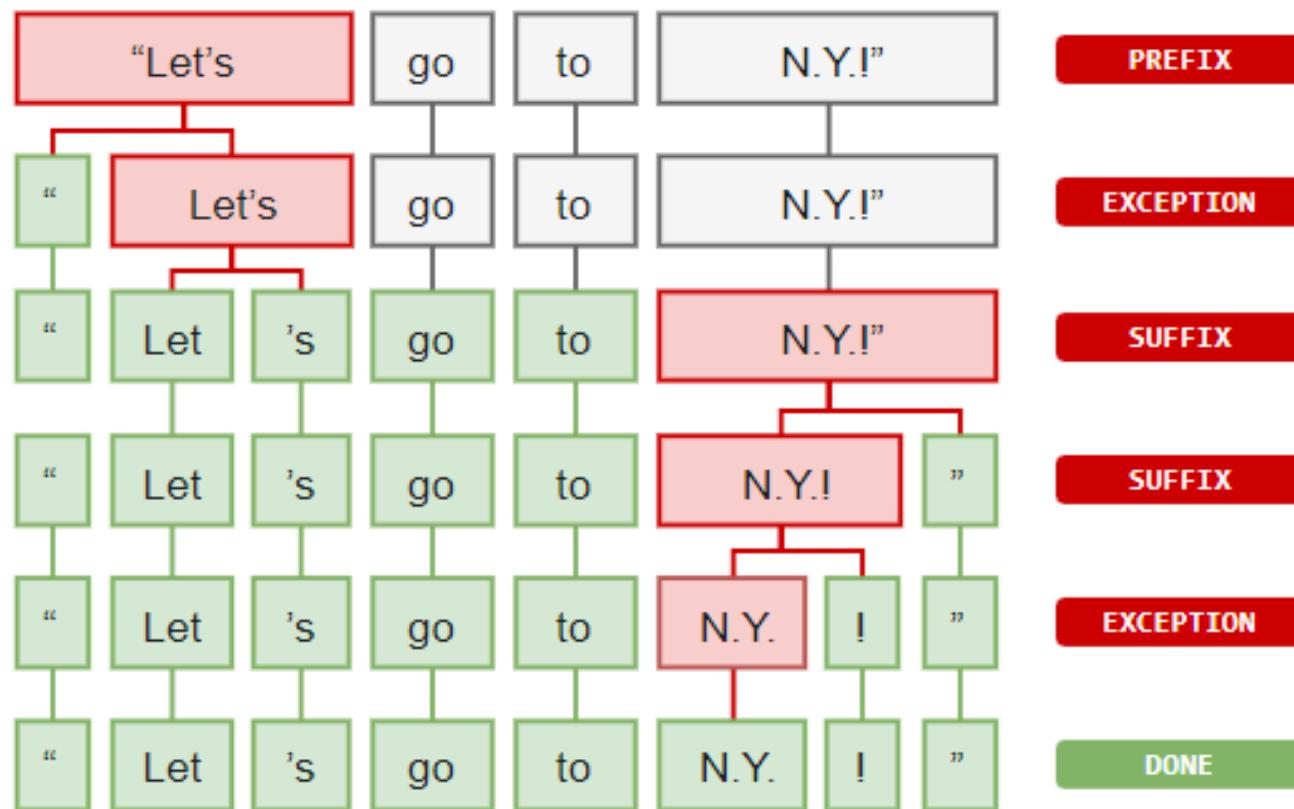


Figure: Recognize the tokens

Source: <https://medium.com/@makcedward/nlp-pipeline-word-tokenization-part-1-4b2b547e6a3>

# Data analysis (1 of 2)

- Deals with the unstructured text that has been source from multiple locations
- Two major processes: Text classification and Text extraction.
- Text classification:
  - Tags are assigned to the text based upon the content.
- Rule-based systems.
- Detect linguistic patterns in the text.
- Tag based upon the detection rules.
- Example:

( HDD | RAM | SSD | Memory ): Hardware

  - The classification algorithm will take any text that contains the word HDD, RAM, SSD, Memory and classify it under the common tag hardware.

# Data analysis (2 of 2)

- Machine learning based systems.
- Predict based upon the past observations.
- Require multiple samples of text and tags.
- Converted into vectors before training.
- Vectors: Extract the features that are relevant.

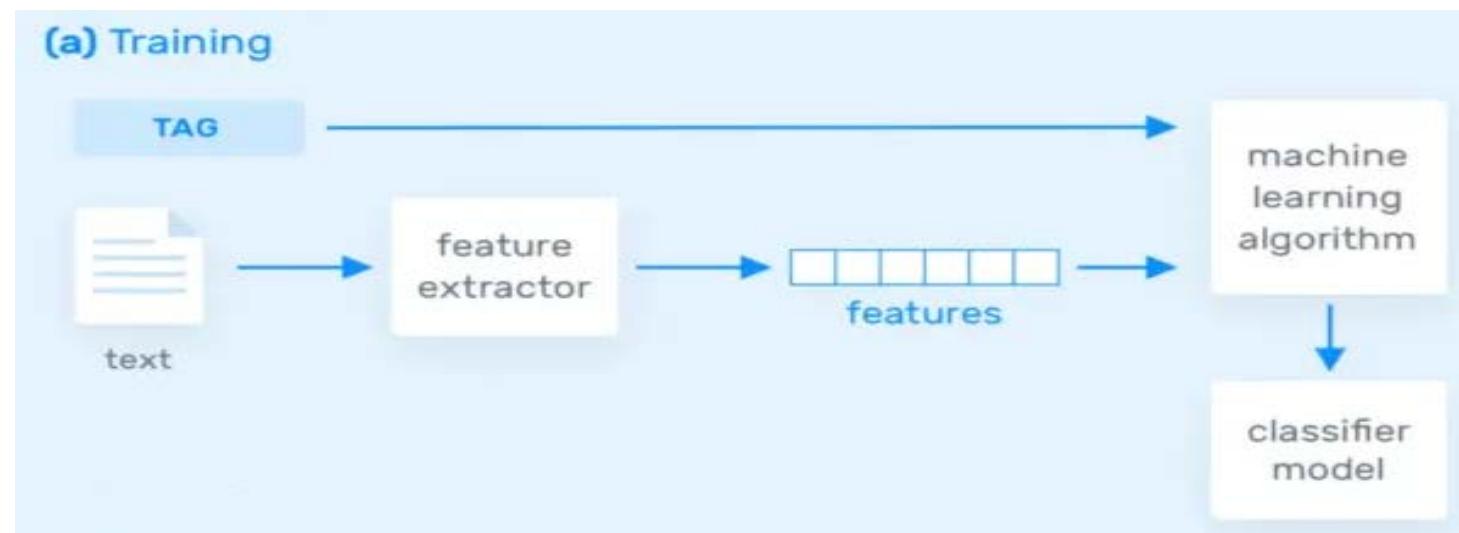


Figure: Training

Source: <https://monkeylearn.com/text-analysis/>

# Evaluation of text classification process

- Sample testing data: Separate test data set.
- Cross validation method: Splits training and testing data.
- Accuracy: Number of correct predictions against the total number of predictions.
- Precision: Total number of correctly predicted output against the total number of outputs predicted.

Evaluation metric	Formula
Precision	$TP / (TP + FP)$
Recall	$TP / (TP + FN)$
Accuracy	$(TP + TN)/N$
F Measure	$2 * Recall * Precision / (Recall + Precision)$

Figure: Evaluation metric formula

Source: [https://www.researchgate.net/figure/Text-Classification-Evaluation-Metrics\\_tbl1\\_329910331](https://www.researchgate.net/figure/Text-Classification-Evaluation-Metrics_tbl1_329910331)

# Text extraction

- Technique that can identify and gather valuable information from the data available inside any text.
- Keyword extraction:
  - Extracting the most relevant words or expressions that are available inside any text.
  - Identify and detect multiple words or expressions.
  - Extract content about any keyword.
- Entity extraction:
  - Identify the major entities that are relevant to any context

# Analysis in test extraction

- Regular expressions:
  - Identify any pattern of characters that can we search inside a large text data.
- Very fast.
- Results: Straight forward and almost accurate.
  - Complexity of the pattern leads to difficulty in coding.

```
(?i)\b(?:[a-zA-Z0-9_-]+)@(?:(:[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3}.)|(:(:[a-zA-Z0-9-]+.))+)(?:[a-zA-Z]{2,4}|[0-9]{1,3})(?:]?)\b
```

- Conditional random fields.
- Algorithm is trained to learn patterns that are to be extracted from the given source.
- Complex patterns can be used for encoding higher dimensional information.
- Machine learning algorithm can follow on any source data.

# Evaluation of text extraction process

- Metrics
  - Accuracy
  - Precision
  - Recall
  - F1 Score
- Must be a perfect match of the extracted segment.
- Should be able to capture partial results also.
- Recall Oriented Understudy For Gisting Evaluation (ROUGE) metric methodology: Identifying the performance of text extractors.
- Calculate the length and sequence numbers that overlap among the source text and the extracted text.

# Text analysis APIs

- Python
  - Large collection of libraries: Support numerical, scientific models is common for usage in identification and extraction process of text.
- Natural language toolkit
  - Can be used for analysis of text.
  - Predefined methods and library functions: Lot of operations needed in text analysis.
- Scikit learn
  - This library works with other modules like Numpy, Scipy and Matplotlib.
- Tensorflow
  - Library tool can also be used along with Python.
  - Multiple library functionalities and ready-made models are also available in tensorflow.
- PyTorch
  - Designing the neural network architecture.
  - Provide high performance code.
- Keras
  - Rapid iteration process needed for deep learning neural networks.

# Levels of NLP

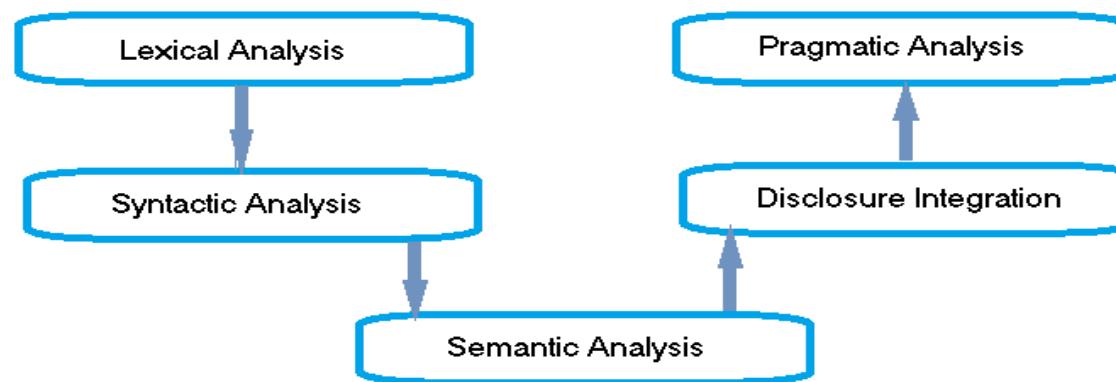


Figure: Levels of NLP

# Self evaluation: Exercise 1

- To continue with the training, after learning the basics of Linguistic Structure and various steps in Natural Language Processing, it is time to start writing code to perform tasks at a basic level. It is instructed to utilize the concepts of reading data from files to perform the following activity.
- You are instructed to write the following activities using Python code.
- Exercise 1: Python code to read data from a CSV file, Process the data and Create Files based on the Data.
-

# Self evaluation: Exercise 2

- To continue with the training, after learning the basics of Linguistic Structure and various steps in Natural Language Processing, it is time to start writing code to perform tasks at a basic level. It is instructed to utilize the concepts of reading data from files to perform the following activity.
- You are instructed to write the following activities using Python code.
- Exercise 2: Code to read data from a CSV file, Process, Subset, Merge and Clean the given Text Data.

# Lexical analysis

- Identifying and analyzing the structure of words.
- Lexicon: Collection of words and phrases in a language.
- Lexical analysis: Divide the whole chunk of txt into paragraphs, sentences, and words
- Token: Token name and optional token value.

Token name	Sample token values
<b>identifier</b>	x, color, UP
<b>keyword</b>	if, while, return
<b>separator</b>	}, (, ;
<b>operator</b>	+, <, =
<b>literal</b>	true, 6.02e23, "music"
<b>comment</b>	<i>/* Retrieves user data */, // must be negative</i>

```
--  
texto = re.sub(r'''['|'|`]m''', ' am', texto)  
texto = re.sub(r'''['|'|`]re''', ' are', texto)  
  
sent_tokenizer=nltk.data.load('tokenizers/punkt/english.pickle')  
sentences = sent_tokenizer.tokenize(texto)  
  
pattern =r''' (?x)  
    ([A-Z]\. )+  
    | \w+[!|`]s  
    | \w+(-\w+)*  
    | \$?d+(\.\d+)?%?  
    | \.\.\.  
    | [][\.,;"?():-_!]  
    ...  
    .  
token = []  
for sentence in sentences:  
    words = nltk.tokenize.regexp_tokenize(sentence, pattern)  
    token.append(words)
```

# POS tagging (1 of 2)

Abbreviation	Meaning
CC	coordinating conjunction
CD	cardinal digit
DT	determiner
EX	existential there
FW	foreign word
IN	preposition/subordinating conjunction
JJ	adjective (large)
JJR	adjective, comparative (larger)
JJS	adjective, superlative (largest)
LS	list marker
MD	modal (could, will)
NN	noun, singular (cat, tree)
NNS	noun plural (desks)
NNP	proper noun, singular (sarah)
NNPS	proper noun, plural (indians or americans)
PDT	predeterminer (all, both, half)
POS	possessive ending (parent's)
PRP	personal pronoun (hers, herself, him,himself)

PRP\$	possessive pronoun (her, his, mine, my, our )
RB	adverb (occasionally, swiftly)
RBR	adverb, comparative (greater)
RBS	adverb, superlative (biggest)
RP	particle (about)
TO	infinite marker (to)
UH	interjection (goodbye)
VB	verb (ask)
VBG	verb gerund (judging)
VBD	verb past tense (pleaded)
VBN	verb past participle (reunified)
VBP	verb, present tense not 3rd person singular(wrap)
VBZ	verb, present tense with 3rd person singular (bases)
WDT	wh-determiner (that, what)
WP	wh- pronoun (who)
WRB	wh- adverb (how)

# POS tagging (2 of 2)

- Lexical categories.
- Called as word classes.
- Tag set of that language.



Figure: Tag set of language.

# Syntactic parsing

- Third phase during text analysis process.
  - Identification of the meaningfulness present in each text.
  - Identifying the meaning of the text that belongs to a natural language
  - Within the boundary rules of the grammar of that language
- 
- Parser: Tokens from lexical analyzer: Meaningful representation.
  - Uses symbol table.

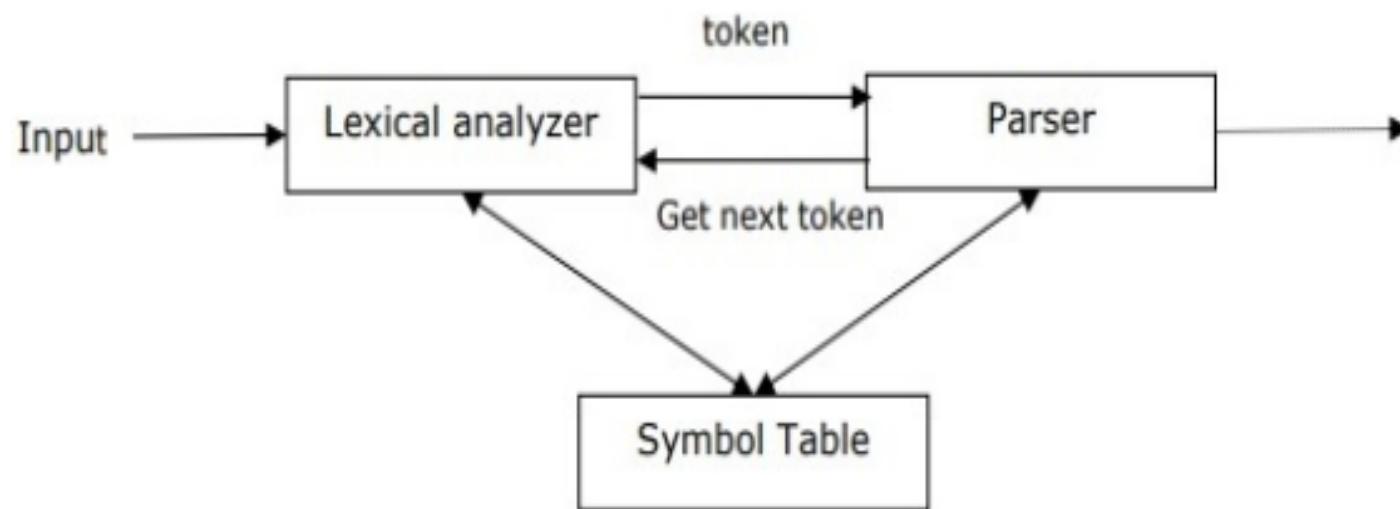


Figure: Syntactic parsing

Source: [https://www.tutorialspoint.com/natural\\_language\\_processing/natural\\_language\\_processing\\_syntactic\\_analysis.htm](https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_syntactic_analysis.htm)

# Types of parsing

## Top down parsing

- Starts at the start symbol and proceeds further.
- Every input symbol is read and recursively proceeds to process every element in the input.
- Due to the recursive nature backtracking occurs.

## Bottom up parsing

- Inverse of top down parsing.
- Derivation is defined as the production rules that are used during parsing.

# Derivation logic

- Leftmost derivation: The provided input string is scanned from left to right in sentential form.
- Rightmost derivation: The provided input string is scanned from right to left in sentential form.

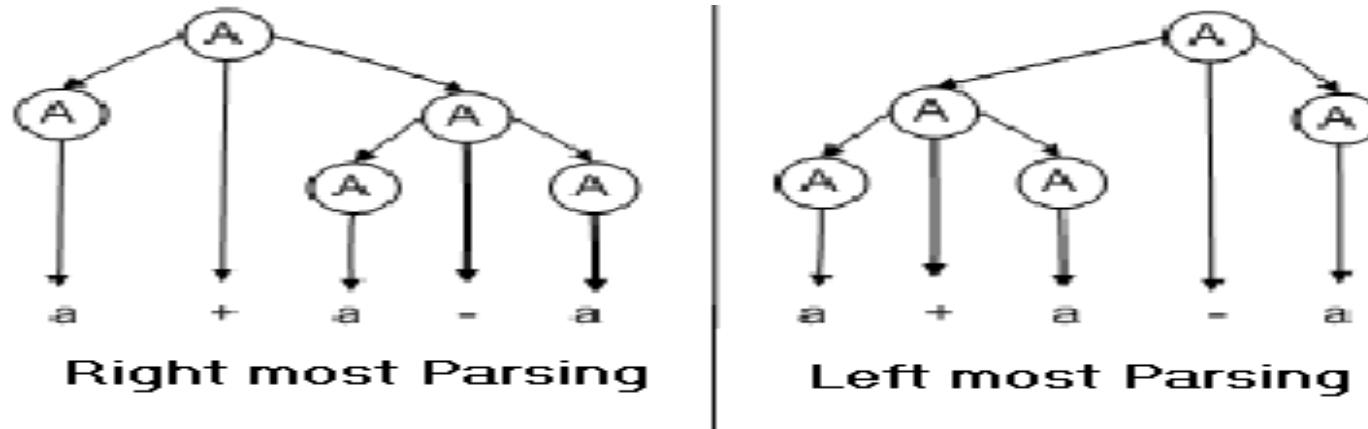


Figure: Parse trees

Source: SELF

# Grammar

- Grammar defined by G for any language is written in 4 tuple format as (N, T, S, P).

## Constituency grammar

- Noam Chomsky grammar: Subject predicate division methodology
- Major clauses: Noun phrase and verb phrase.

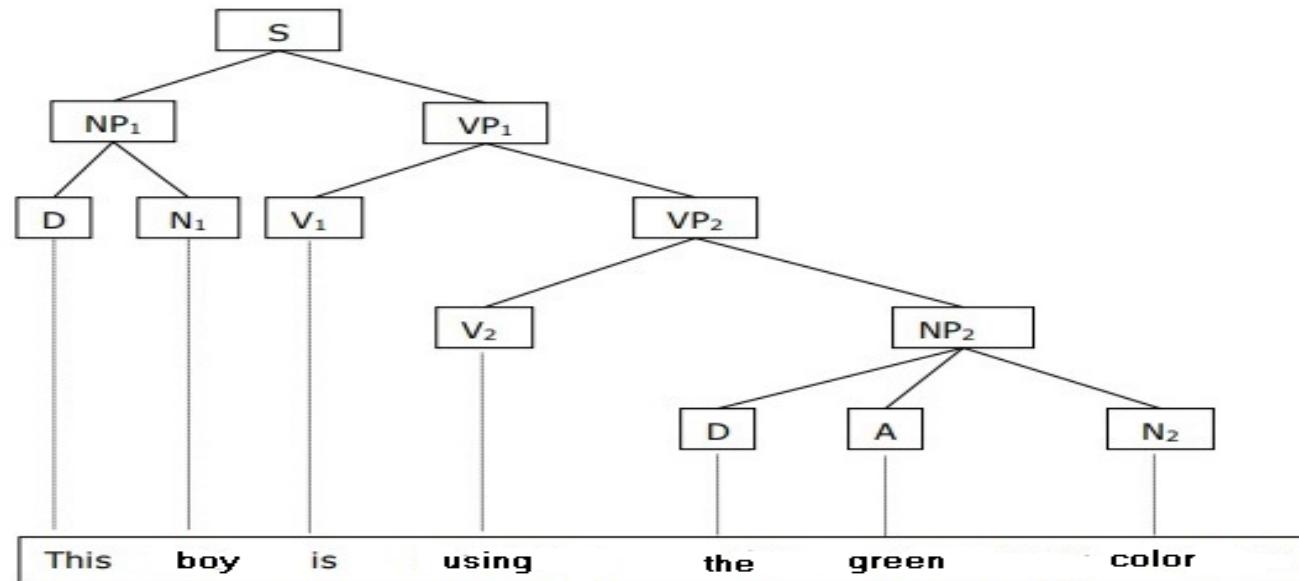


Figure: Noam Chomsky is based upon constituency relationship

Source: SELF

# Semantic analysis

- Concept identify and extract the meaning specified according to a language dictionary.
- Semantic analyzer: Extraction of the meaningfulness.
- Identify the meaning of individual words.
- Identify the meaning of the sentence: Combining the meaning of the individual words.

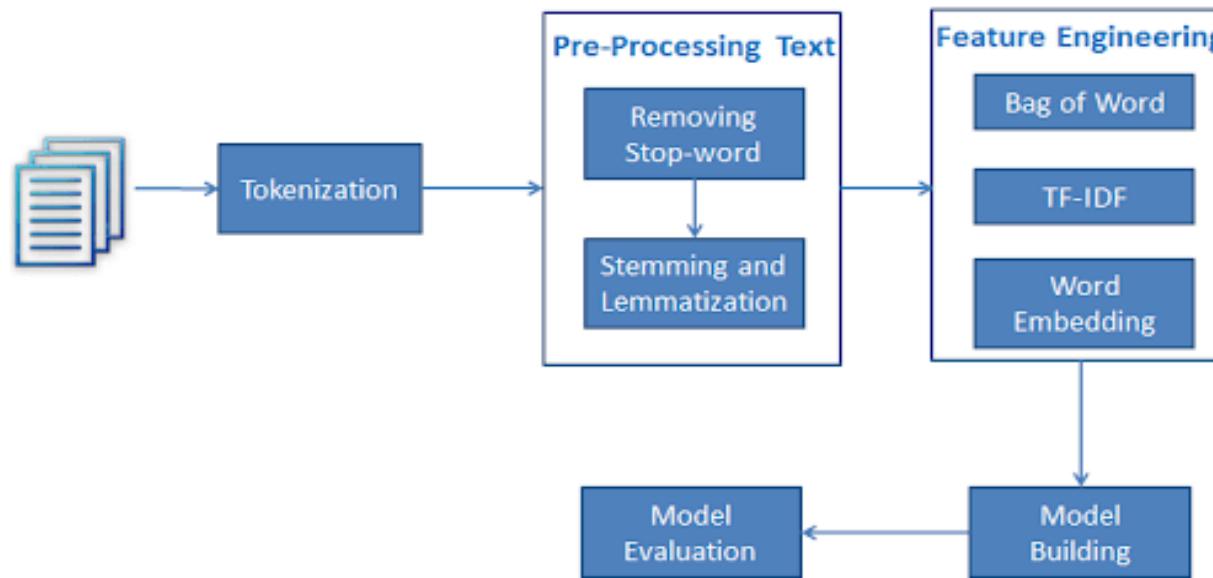


Figure: Semantic analysis

Source: <https://www.datacamp.com/community/tutorials/text-analytics-beginners-nltk>

# Semantic analysis elements

Synonymy: Relationship between two elements: expressing similar meaning.

- Antonymy: Relationship between two lexical items: provide dissimilarity in their meaning.
- Hyponymy: Represents the relationship between generic term: Instances.
  - Example:
  - Hypernym - Color
  - Hyponym - Red, Green, Blue
- Homonymy: Relationship between two or more words: similar spelling: different meaning
  - Example: Tablet - Medicine or Computer Device

# Representation in semantic analysis

- Semantic analysis core components
- Entities: Noun representation.
- Concepts: Generic noun representation.
- Relations: Relationship between entities and concepts.
- Predicates: Word representation.

# Self evaluation: Exercise 3

- To continue with the training, after learning the various steps involved in Natural Language Processing, it is instructed to utilize the concepts of Lexical, Syntactic and Semantic Analysis to process text from files to perform the following activity.
- You are instructed to write the following activities using Python code.
- Exercise 3: Read any available text information, perform Pre-processing on the data, Remove Stop words, Perform Tokenization, Classify the document words based on specific topics with Modelling.

# Natural language generation

- Natural language understanding: Identification of the natural language elements.
- Natural language processing: Understanding the nuances of language.
- Natural language generation: Generating simple natural language responses.

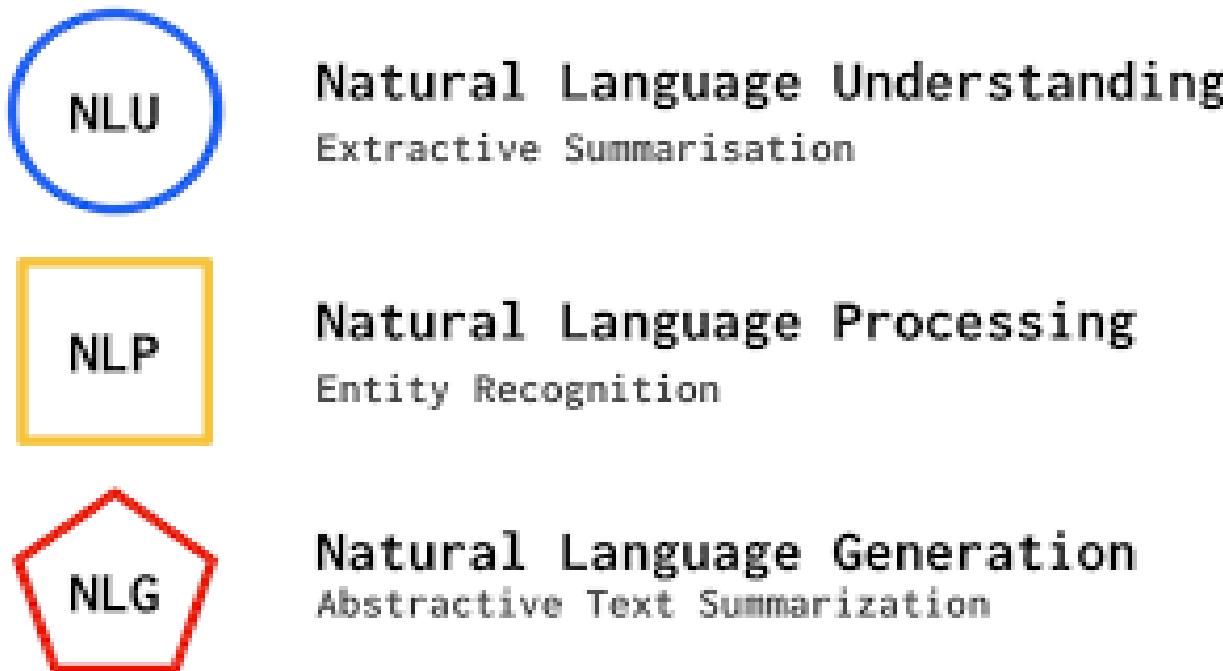


Figure: Natural language generation

Source: <https://wordlift.io/blog/en/advanced-seo-natural-language-processing/>

# NLP vs NLG

- NLP: Describe a machine's ability to ingest what is said.
- NLU: Subset of NLP: Handle unstructured inputs: Structured form.
- NLG: Processes turn structured data into text.

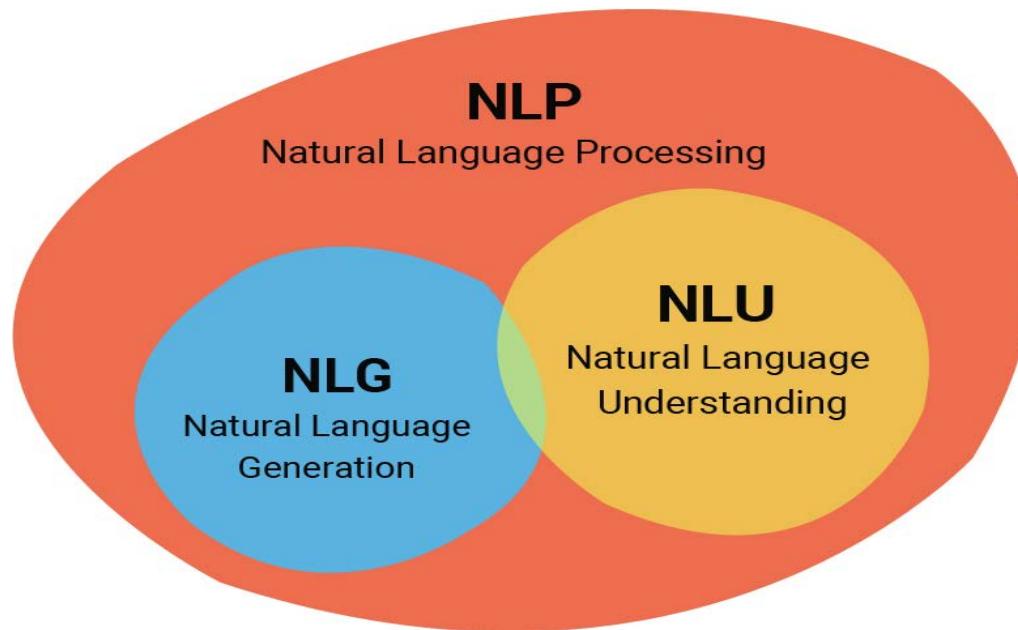


Figure: NLP vs NLG

Source: <https://www.aismartz.com/blog/the-past-and-the-presence-of-natural-language-generation/>

# History of NLG

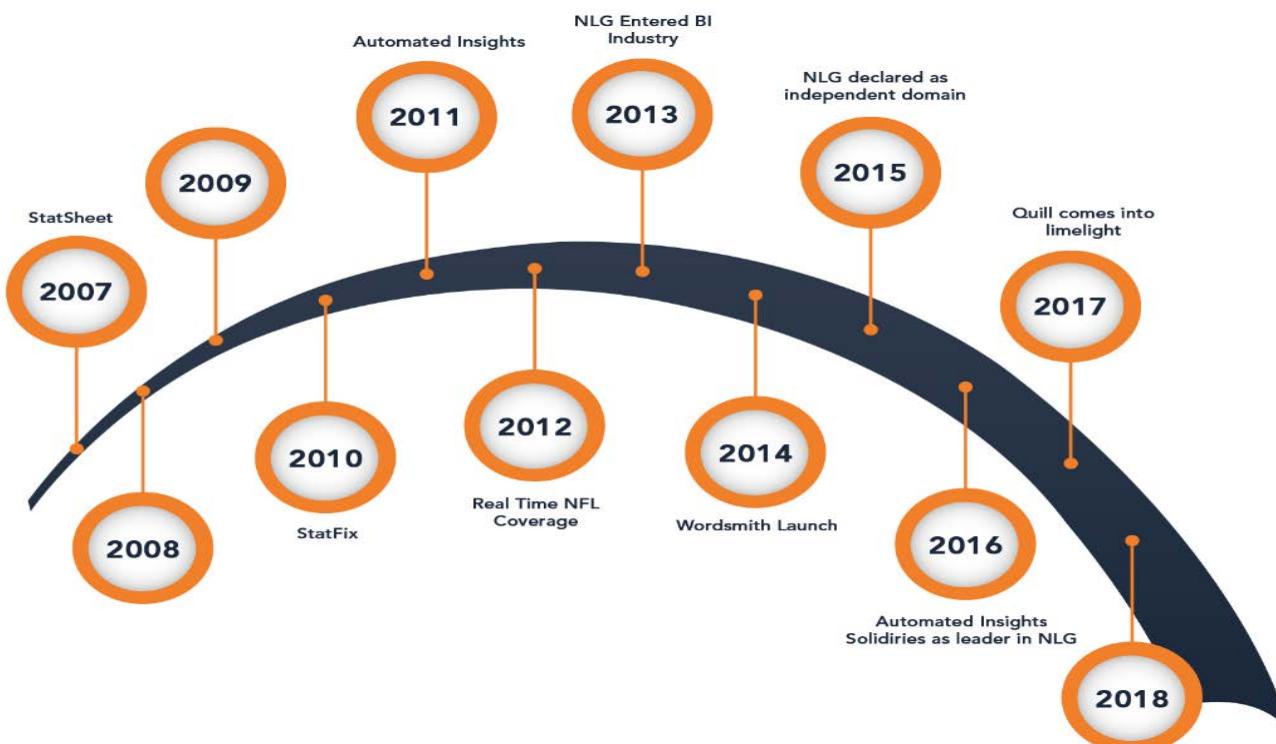


Figure: History of NLG

Source: <https://www.aismartz.com/blog/the-past-and-the-presence-of-natural-language-generation/>

# Working principle of natural language generation



IBM ICE (Innovation Centre for Education)

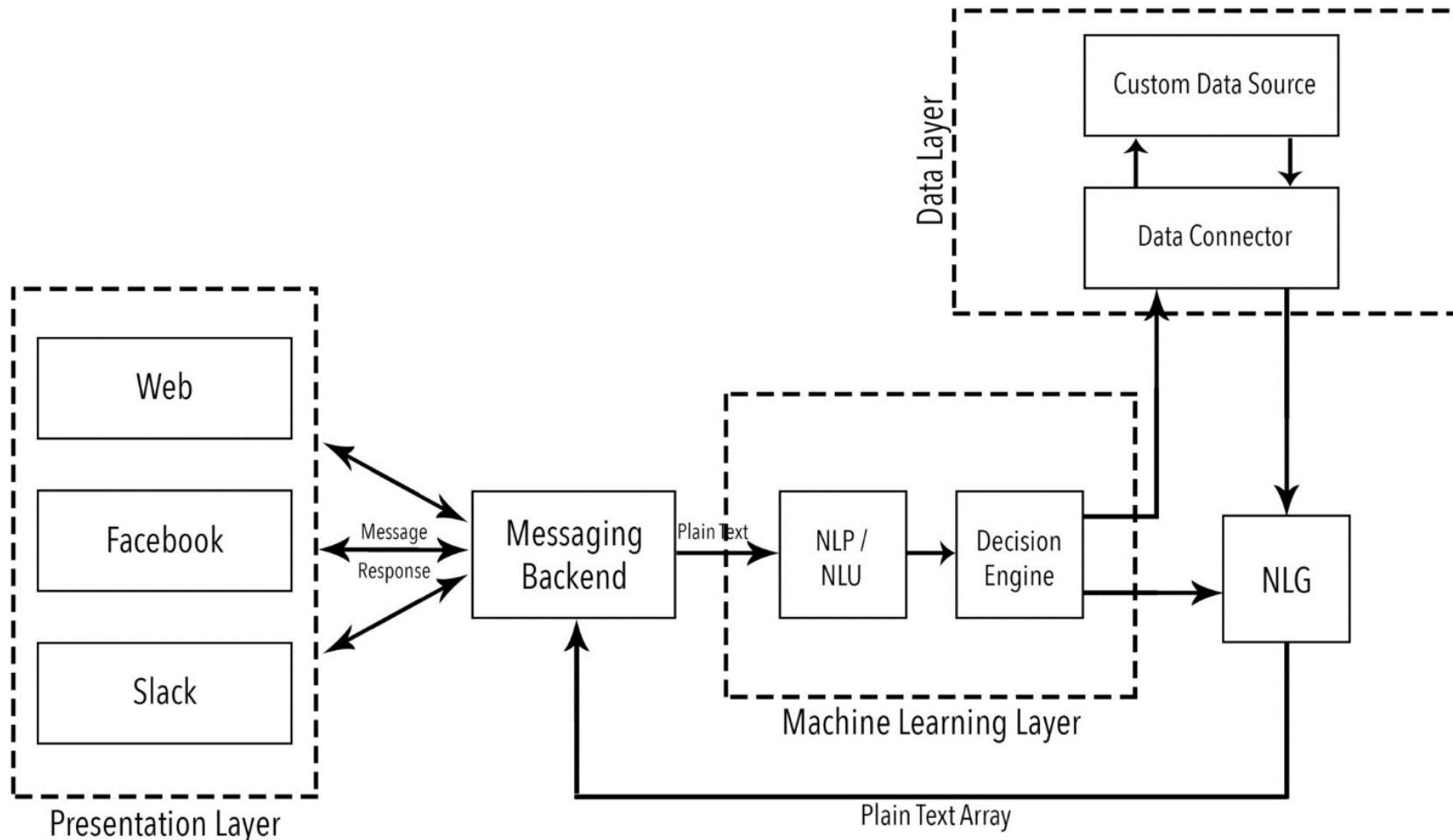


Figure: Working principle of natural language generation

Source: <https://chatbotslife.com/nlp-nlu-nlg-and-how-chatbots-work-dd7861dfc9df>

# Limitations in natural language generation



IBM ICE (Innovation Centre for Education)

- Difference in the stress level.
- Empathy not easily identified.
- Capture: Nuance in the communication.
- Response in natural language as per the context.

# Checkpoint (1 of 2)

## Multiple choice questions:

1. What is the field of Natural Language Processing (NLP)?
  - a) Computer Science
  - b) Artificial Intelligence
  - c) Linguistics
  - d) All of the above
  
2. What is Morphological Segmentation?
  - a) Does discourse analysis
  - b) Separate words into individual morphemes and identify the class of the morphemes
  - c) Is an extension of propositional logic
  - d) None of the above
  
3. Parts-of-Speech tagging determines \_\_\_\_\_
  - a) Part-of-speech for each word dynamically as per meaning of the sentence
  - b) Part-of-speech for each word dynamically as per sentence structure
  - c) All part-of-speech for a specific word given as input
  - d) All of the above

# Checkpoint solutions (1 of 2)

## Multiple choice questions:

1. What is the field of Natural Language Processing (NLP)?
  - a) Computer Science
  - b) Artificial Intelligence
  - c) Linguistics
  - d) **All of the above**
2. What is Morphological Segmentation?
  - a) Does discourse analysis
  - b) **Separate words into individual morphemes and identify the class of the morphemes**
  - c) Is an extension of propositional logic
  - d) None of the above
3. Parts-of-Speech tagging determines \_\_\_\_\_
  - a) Part-of-speech for each word dynamically as per meaning of the sentence
  - b) Part-of-speech for each word dynamically as per sentence structure
  - c) All part-of-speech for a specific word given as input
  - d) **All of the above**

# Checkpoint (2 of 2)

## Fill in the blanks:

1. Parsing generates \_\_\_\_\_ for a given sentence in graphical form.
2. Given a sound clip of a person or people speaking, \_\_\_\_\_ is the textual representation of the speech.
3. In linguistic morphology \_\_\_\_\_ is the process for reducing inflected words to their root form.

## True or False:

1. Modern NLP algorithms are based on machine learning, especially statistical machine learning. True/False
2. Speech segmentation is a subtask of speech recognition. True/False
3. Natural language generation is the main task of natural language processing. True/False

# Checkpoint solutions (2 of 2)

## Fill in the blanks:

1. Parsing generates **Parse Trees** for a given sentence in graphical form.
2. Given a sound clip of a person or people speaking, **Speech-to-text** is the textual representation of the speech.
3. In linguistic morphology **Stemming** is the process for reducing inflected words to their root form.

## True or False:

1. Modern NLP algorithms are based on machine learning, especially statistical machine learning. **True**
2. Speech segmentation is a subtask of speech recognition. **True**
3. Natural language generation is the main task of Natural language processing. **True**

# Question bank

## Two mark questions:

1. How is regular expression used in text extraction?
2. What are the text extraction methods?
3. Explain the semantic analysis elements
4. What is CFG?

## Four mark questions:

1. Differentiate text analysis, text mining, text analytics.
2. How is text classified?
3. Explain the evaluation of text classification process
4. What is syntactic parsing?

## Eight mark questions:

1. Discuss the levels of understanding the linguistics of any language.
2. Describe in detail the data processing activities of text process.

# Unit summary

**Having completed this unit, you should be able to:**

- Understand various concepts of database
- Gain knowledge on importance of structure of database
- Learn about various components of database
- Gain an insight into insurance database models

Welcome to:

# Empirical Approaches



# Unit objectives

**After completing this unit, you should be able to:**

- Understand the concepts of various empirical approaches
- Learn on how to work with creation of corpus, approaches and working principles
- Gain knowledge on nuances of annotation, know about treebank annotation and how to create and use it
- Understand the concepts various fundamental statistical techniques used in annotations and their principles
- Gain an insight into part-of-speech tagging principles along with the various techniques to implement it

# Corpus creation

- Basic reference location that the features of the natural language.
- Created based upon the grammar and context.
- Analysis of the Linguistics and Hypothesis testing.

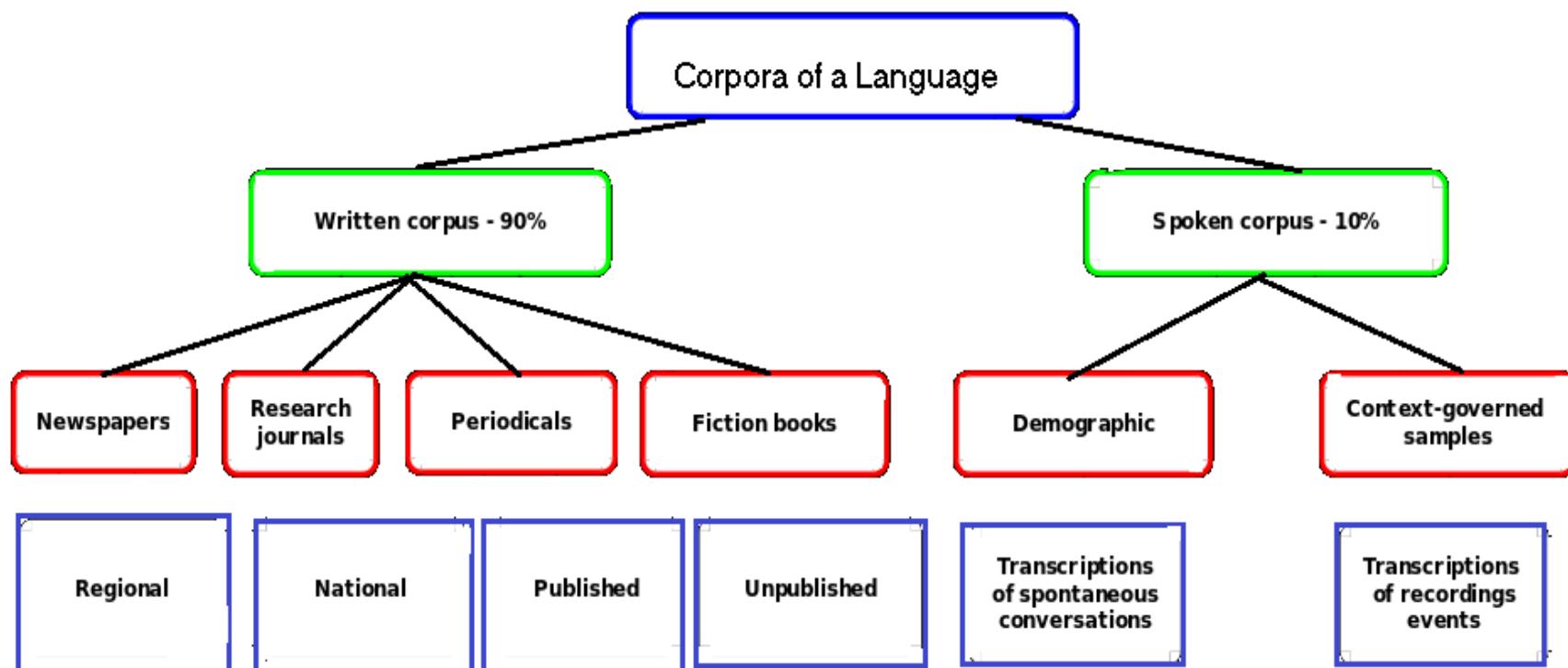


Figure: Corpus creation

Source: <https://devopedia.org/text-corpus-for-nlp>

# Corpus linguistics

- Collection of natural language elements that can characterize any natural language.
- Provide rules, grammar, and structure for the language.
- Corpus linguistics concepts:
  - Direct observation of the data.
  - Performance based upon the context.
  - Lexico grammatical approach.
  - Temporal activity based upon the context.
- Monolingual or Multilingual.
- Identify the state of any natural language.
- Teaching or research.
- Electronic Text Library (ETL).
- Holds a lot of text to identify patterns.

# Types of Corpora

- Language for general purposes corpora:
  - Economic corpora.
  - Legal corpora.
  - Medical corpora.
- Multilingual parallel corpora:
  - L1: L2 bidirectional.
  - L1 translation →L2.

# Lexicographical implementations in Corpora (1 of 2)



IBM ICE (Innovation Centre for Education)

- Focused on the compilation, design, processing and evaluation of the dictionaries.
- Dictionary for general language structure: LGP dictionary.
- Dictionary for specific elements: LSP dictionary.
- Word frequency information:
  - Calculate the number of occurrences of the words in each text sample.
- Collocation:
  - Words or terms that co-occur more often.
    - Example: move on, jump in.
- Set phrase:
  - Words that are grouped together that have a meaningful rendering.
    - Example: Day and night, grey area.

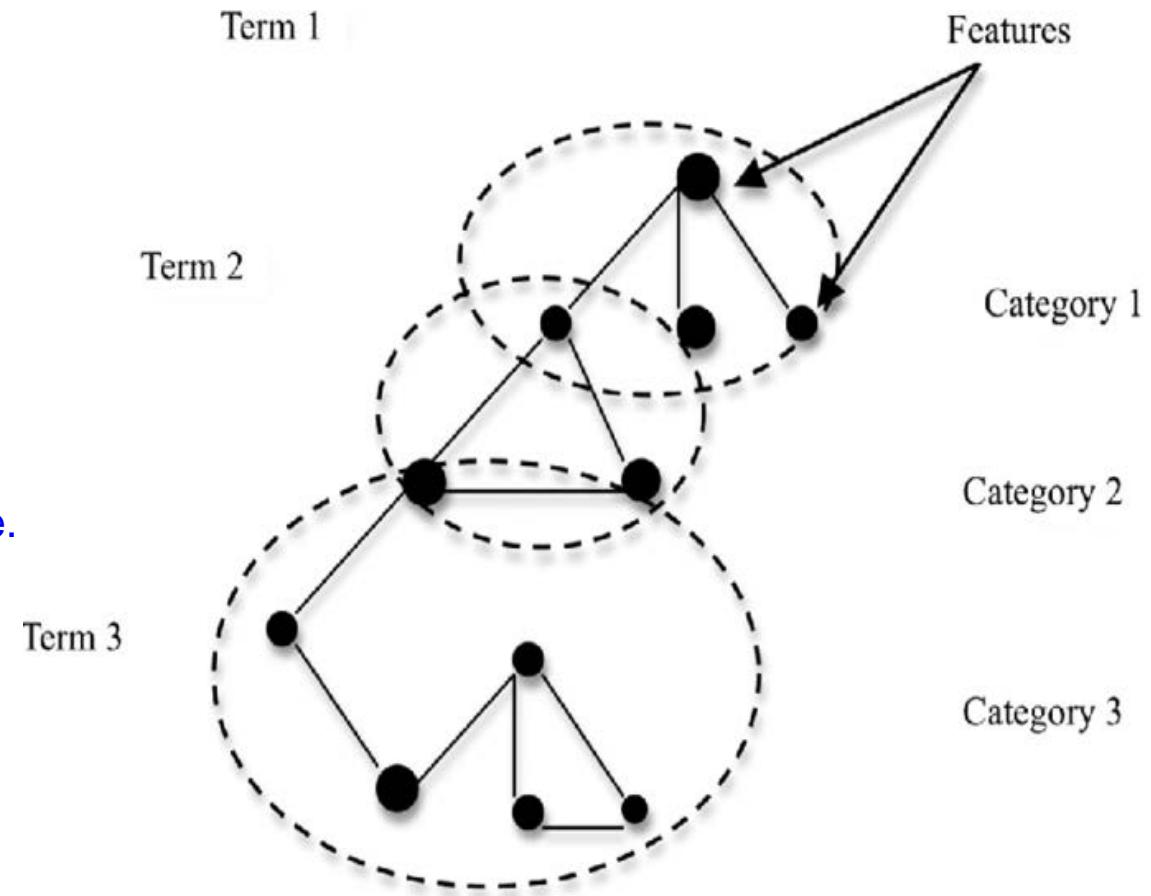
# Lexicographical implementations in Corpora (2 of 2)

- Linguistic theory:
  - Definite description explanations.

- Lexical studies:
  - Example: little, small.

- Sociolinguistics:
  - Example: guy / gal.

- Cultural identifications:
  - Example: Lorry / Truck, Biscuit / Cookie.



# Timeline of Corpus linguistics

- 1950s:
  - Collections are created based on spoken and written utterances of various languages from field research.
- 1960s:
  - Work in Information Retrieval (IR) develops techniques for statistical similarity of document content.
- 1970s:
  - Stochastic models developed from speech corpora make speech recognition systems possible. The vector space model is developed for document indexing.

# Usage areas of Corpora

- Translation
  - Parallel corpora
  - Native corpora
- Education
  - Data Driven Learning
  - Concordance usage
  - Generalization extraction from data
- General usages
  - Native speaker intuition
  - Frequency of occurrence
  - Relationship as per usage

# Traits of a good text corpus

- Depth
  - Example: Wordlist: Top 50000 words and not just top 5000 words.
- Recent
  - Example: outdated texts: Courting vs current age texts: Dating.
- Metadata
  - Example: Source, Genre, Type
- Genre
  - Example: Text from newspapers, journals, etc.
- Size
  - Example: half million English words
- Clean
  - Example: Noun – Flower, Fruit Verb – Eat, Smell

# Annotations in text Corpus

Several lines of evidence, based on the results of overexpression studies, indicate that PGC-1 $\alpha$  is sufficient to promote mitochondrial biogenesis and regulate mitochondrial respiratory capacity. First, PGC-1 $\alpha$  activates the transcription of mitochondrial uncoupling protein-1 (UCP-1) in BAT through interactions with the nuclear hormone receptors PPAR $\gamma$  and thyroid receptor [2]. Second, forced expression studies in adipogenic and myogenic mammalian cell lines demonstrated that PGC-1 $\alpha$  activates mitochondrial biogenesis through a group of transcription factor targets including nuclear respiratory factors 1 and 2 (NRF-1 and -2) and mitochondrial transcription factor A (Tfam), key transcriptional regulators of mitochondrial DNA transcription and replication [8]. Third, studies in primary cardiac myocytes in culture and in the hearts of transgenic mice have demonstrated that overexpression of PGC-1 $\alpha$  promotes mitochondrial biogenesis [10,16]. Lastly, forced expression of PGC-1 $\alpha$  in skeletal muscle of transgenic mice triggers mitochondrial proliferation and the formation of mitochondrial-rich type I, oxidative (“slow-twitch”) muscle fibers [17]. Collectively, these results indicate that PGC-1 $\alpha$  is sufficient to drive mitochondrial biogenesis.

Figure: Inline annotations

# Annotations in text Corpus

Suitable acid addition salts are formed from acids which form non-toxic salts. Examples include the acetate, adipate, aspartate, benzoate, besylate, bicarbonate/carbonate, bisulphate/sulphate, borate, camsylate, citrate, cyclamate, edisylate, esylate, formate, fumarate, gluceptate, gluconate, glucuronate, hexafluorophosphate, hibenzate, hydrochloride/chloride, hydrobromide/bromide, hydroiodide/iodide, isethionate, lactate, malate, maleate, malonate, mesylate, methylsulphate, naphthylate, 2-napsylate, nicotinate, nitrate, orotate, oxalate, palmitate, pamoate, phosphate/hydrogen phosphate/dihydrogen phosphate, pyroglutamate, saccharate, stearate, succinate, tannate, tartrate, tosylate, trifluoroacetate and xinofoate salts.

Figure: Annotations commonly used in any text corpora fall categories

# NLP task-specific training corpora

- POS Tagging:
  - Penn Treebank's WSJ section 45-tag tagset.
- Named entity recognition:
  - CoNLL 2003 NER task is newswire content from Reuters RCV1 corpus.
- Constituency parsing:
  - Penn Treebank's WSJ section has dataset.
- Semantic role labelling:
  - OntoNotes v5.0 with syntactic and semantic annotations.
- Sentiment analysis:
  - IMDb has released 50K movie reviews.
  - Amazon Customer Reviews of 130 million reviews
- Text Classification/Clustering:
  - Reuters-21578 news documents from 1987 indexed by categories.
- Question answering:
  - Jeopardy dataset of about 200000 Q & A

# Data sets used for natural language processing



IBM ICE (Innovation Centre for Education)

- Wordlists
  - Names or Stop words are useful for NLP work.
    - Example: BNC corpus provides Phrases in English (PIE)
- Tagsets
  - Essential for POS tagging, chunking, dependency parsing or constituency parsing.
    - Example: DKPro Core Tagset.
- Treebanks
  - A treebank is an annotated corpus.
    - Example: Penn Treebank and CHRISTINE Corpus.
- Word embedding
  - Real-valued vectors representations of words.
    - Example: Polyglot and Nordic Language Processing Laboratory (NLPL).

# Treebank annotation

- Metadata: Markup elements.
- Markup elements: Annotations.
- Natural language: Understood correctly: Metadata and Annotation should be accurate.



Figure: Treebank annotation

# Linguistic description layers

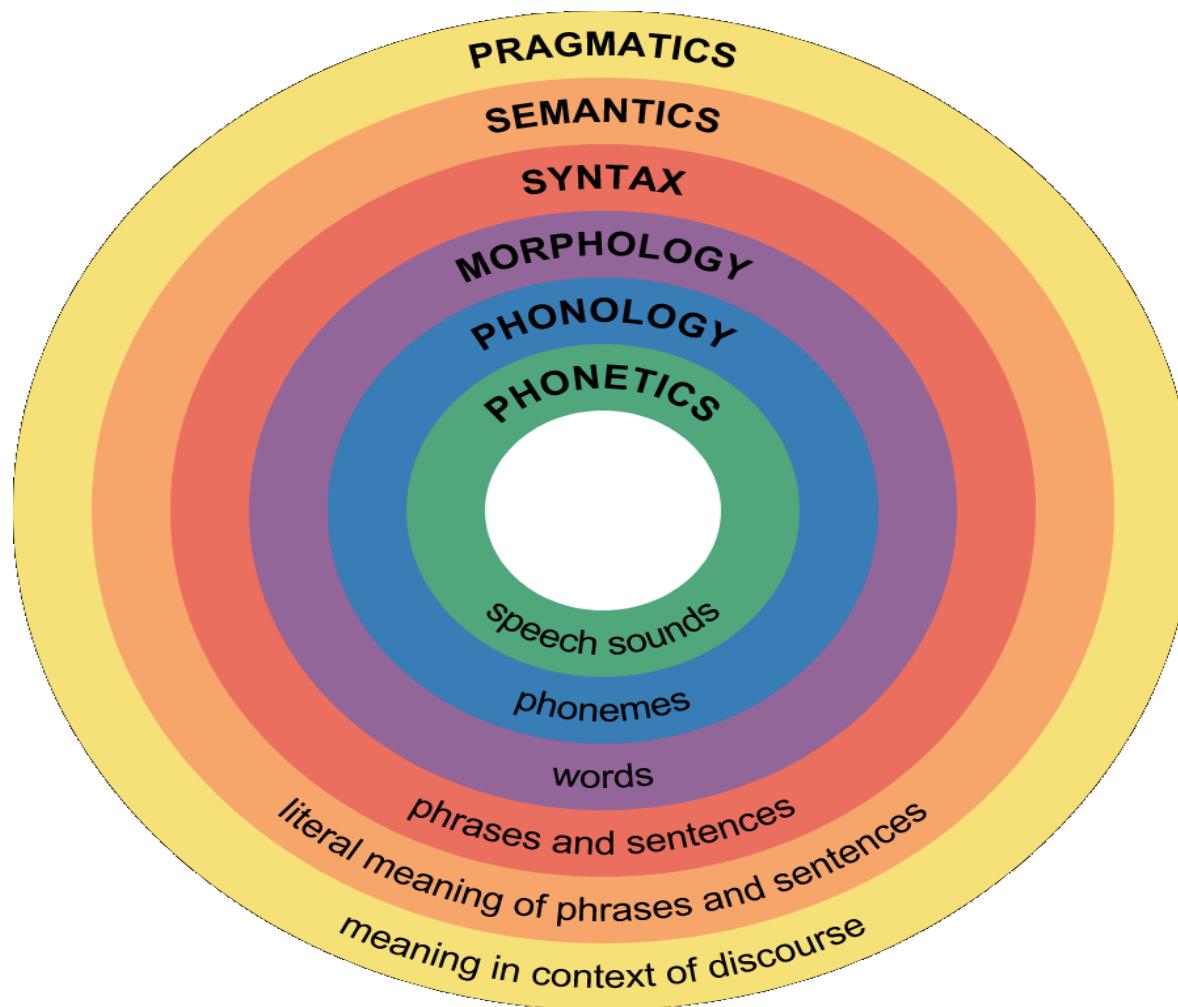


Figure: Linguistic description layers

# Areas using text annotations

- Question answering systems
  - Converse with the clients in any sort of environment wherever a question needs to be answered.
  - The questions and answers: Any natural language.
- Summarization
  - Provides very fast and elevated summary of very large documents in natural languages.
- Machine translation
  - The initial phases is NLP application of translation from one natural language to another.
- Speech recognition
  - Due to the automatic speech recognition systems, available most of the applications can understand the language in which the speech occurs.
- Classification of documents
  - Understanding of the natural language to identify the category of any document.
  - Large set of documents to be organized and arranged based upon the categories.

# Usage of annotations and corpora

- 1980's
  - Usage of compiling language: used in training the algorithms started.
- 1990
  - Hidden Markov models: Identification of Limited vocabulary.
  - Used in statistical language models to perform various natural language tasks.
- 21st century
  - Corpus creation.
  - Generate unigrams or collocations.

# Kinds of annotations

Tagset	Size	Date
Brown	77	1964
LOB	132	1980s
London-Lund Corpus	197	1982
Penn	36	1992

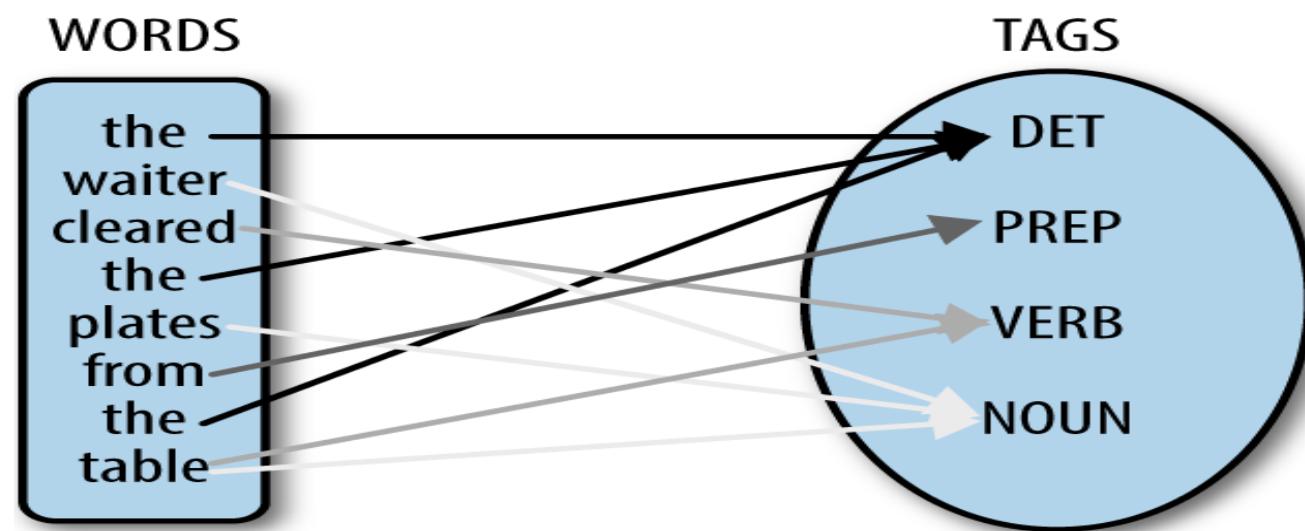
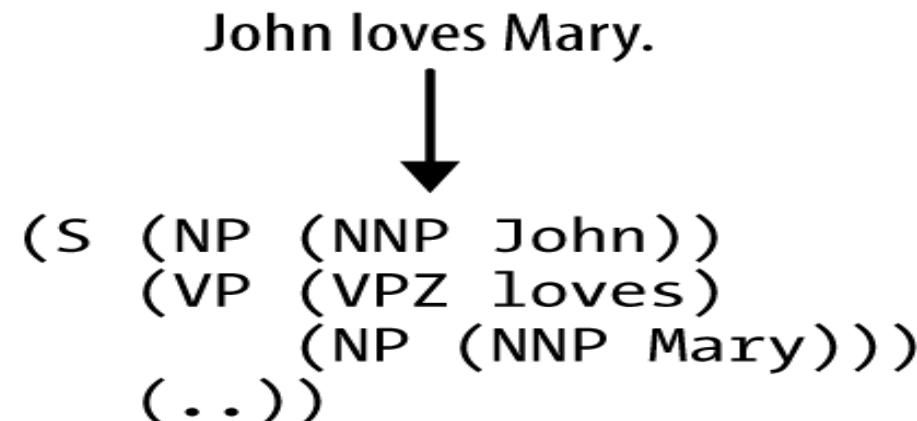


Figure: Part of speech, dependency structure, or phrase structure

# Kinds of annotations

- Speech synthesis
  - Words: A noun or a verb based upon the location of the word in a sentence.
- Parsing
  - Clean dishes are in the cabinet. - Noun phrase
  - Clean dishes before going to work! - Verb phrase
- Machine translation
  - Clean the dishes and put the clean dishes in the cabinet - Verb phrase and noun phrase consecutive usage.



# Annotation semantic labels

- Agent
- Theme
- Experiencer
- Source
- Goal
- Recipient
- Patient
- Instrument
- Location

Examples:

- [The man]<sub>agent</sub> painted [the wall]<sub>patient</sub> with [a paintbrush]<sub>instrument</sub>.
- [Mary]<sub>figure</sub> walked to [the cafe]<sub>goal</sub> from [her house]<sub>source</sub>.
- [John]<sub>agent</sub> gave [his mother]<sub>recipient</sub> [a necklace]<sub>theme</sub>.
- [My brother]<sub>theme</sub> lives in [Milwaukee]<sub>location</sub>.

# Annotations in machine learning

- Supervised learning
  - meta-data to be associated with the word are provided by humans.
- Semi-supervised learning
  - Inputs are generated from both the labelled data provided by the human.
- Unsupervised learning
  - No specific tags The machine tries to identify the structure of the inputs.

Algorithms	Tasks
Clustering	Genre classification, spam labeling
Decision trees	Semantic type or ontological class assignment, co-reference resolution
Naïve Bayes	Sentiment classification, semantic type or ontological class assignment
Maximum Entropy (MaxEnt)	Sentiment classification, semantic type, or ontological class assignment
Structured pattern induction (HMMs, CRFs, etc.)	POS tagging, sentiment classification, word sense disambiguation

# Annotation development cycle

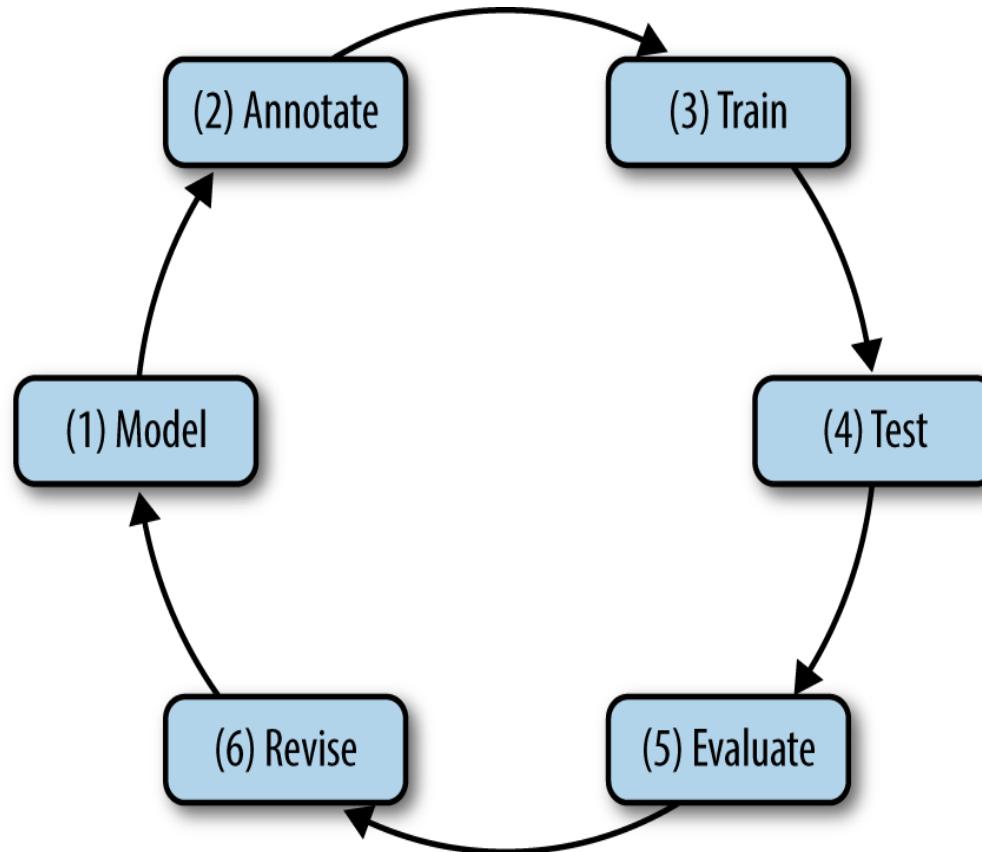


Figure: Annotation development cycle

# Model creation

- Task 1:
  - $T = \{\text{Document\_type}, \text{Spam}, \text{Not-Spam}\}$
  - $R = \{\text{Document\_type}::= \text{Spam} \mid \text{Not-Spam}\}$
  - $I = \{\text{Spam} = \text{"something we don't want!"}, \text{Not-Spam} = \text{"something we do want!"}\}$
- Task 2
  - $T = \{\text{Named\_Entity}, \text{Organization}, \text{Person}, \text{Place}, \text{Time}\}$
  - $R = \{\text{Named\_Entity}::= \text{Organization} \mid \text{Person} \mid \text{Place} \mid \text{Time}\}$
  - $I = \{\text{Organization} = \text{"list of organizations in a database"}, \text{Person} = \text{"list of people in a database"}, \text{Place} = \text{"list of countries, geographic locations, etc."}, \text{Time} = \text{"all possible dates on the calendar"}\}$

# Create annotations

- Annotations are created based on the tags.
- Creation of the tags: Two major types.
  - Consuming tag: Metadata: Content from the data set.
  - Non-consuming tag: Metadata: No content data set.

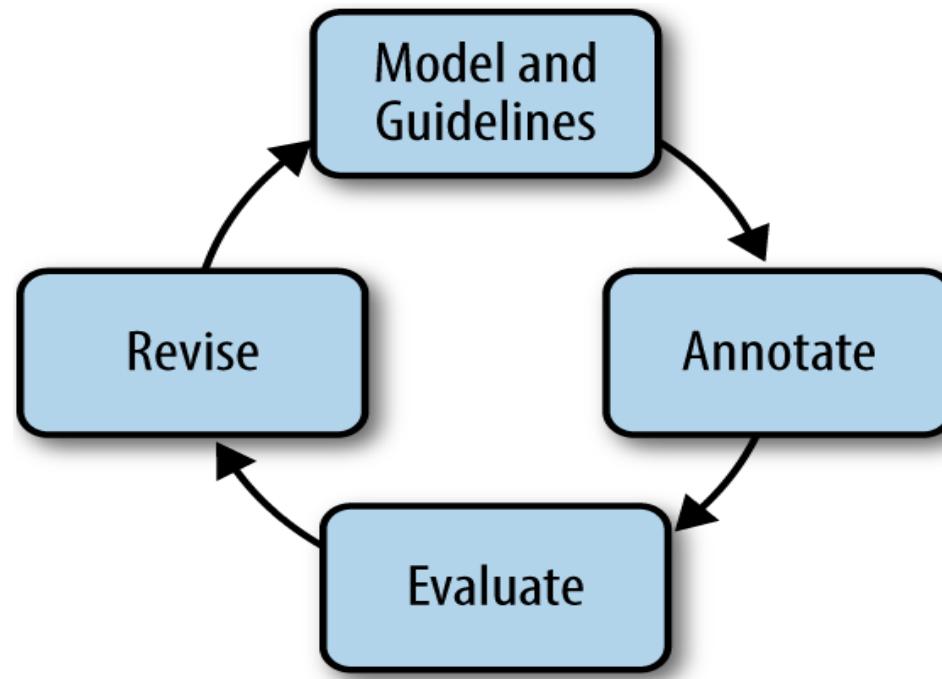


Figure: Annotations cycle

# Training and testing the algorithms

- Annotated Corpus is also split into two parts.
  - One part used for the training purpose is called as development corpus.
  - Other part used for the testing is called as test Corpus.
- Development corpus is again divided into two sets
  - Training set as a whole is used to train the machine algorithm.
  - Development test set is used for error analysis.



Figure: Training and testing the algorithms

# Result evaluation

- Evaluation of the algorithm is based upon the various metrics like Precision and Recall.

		Predicted Labeling	
		positive	negative
Gold Labeling	positive	true positive (tp)	false negative (fn)
	negative	false positive (fp)	true negative (tn)

Precision	$P = \frac{tp}{tp + fp}$
Recall	$R = \frac{tp}{tp + fn}$
Accuracy	$A = \frac{tp + tn}{tp + tn + fp + fn}$

# Revision of the model

- The metric calculations are performed on various categories.
- Depending upon the requirements of the Corpus and the task at hand the model-annotate cycle and the test train cycle continuous many number of times till the required performance is achieved.

# Tree banks and its construction (1 of 2)

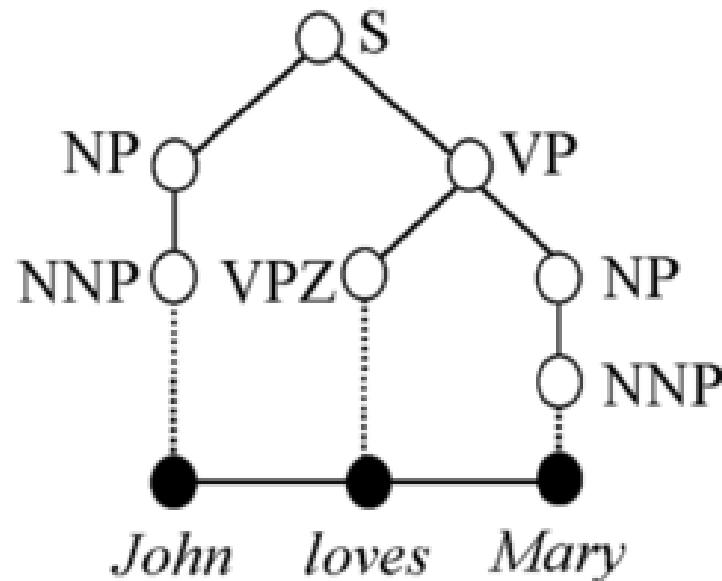
- Geoffrey Leech: Early 1980: Term tree bank.
- Structure: Tree architecture.
- Parsed text corpus.
- Annotates syntactic or semantic sentence.
- Started in early 1990s.
- Large-scale empirical data.
- Tree banks: Based on annotated Corpus: Part of speech tags.
- Manual process.

# Tree banks and its construction (2 of 2)

- Tree Banks are divided into two major types or groups. They are
  - Phrase structured or Constituency tree banks.
  - Dependency structured tree banks.

Example:

- Phrase/Constituency structured tree bank: Penn tree bank.
- Dependency structured tree bank: Prague dependency tree bank.



(S (NP (NNP John)))  
(VP (VPZ loves))  
(NP (NNP Mary))  
(. .))

# Need for tree bank

- Phonological ambiguity:
  - “too”, “two”, “to”
  - “ice cream” vs. “I scream”
- Morphological ambiguity:
  - unlockable: [[un-lock]-able] vs. [un-[lock-able]]
- Syntactic ambiguity:
  - John saw a man with a telescope
  - I saw her duck
- Lexical ambiguity:
  - “bank”, “saw”, “run”
- Semantic ambiguity:
  - every boy loves his mother
  - John and Mary bought a house
- Discourse ambiguity:
  - Susan called Mary. She was sick. (Co-reference resolution)
  - It is pretty hot here. (Intention resolution)

# Types of tree bank Corpus

- Semantic Treebanks.
- Sentence's semantic structure is used list of tree banks:
  - Robot commands treebank.
  - Geoquery, Groningen meaning bank.
  - RoboCup Corpus.
- Syntactic treebanks.
- Expressions of the formal language obtained from the conversion of parsed treebank data.
- Outputs: Predicate logic-based meaning representation.
- List of tree banks:
  - Penn Arabic Treebank, Columbia Arabic treebank: Arabic.
  - Sininca syntactic Treebank: Chinese.
  - Lucy, Susane and BLLIP WSJ: English.

# Types of tree bank Corpus

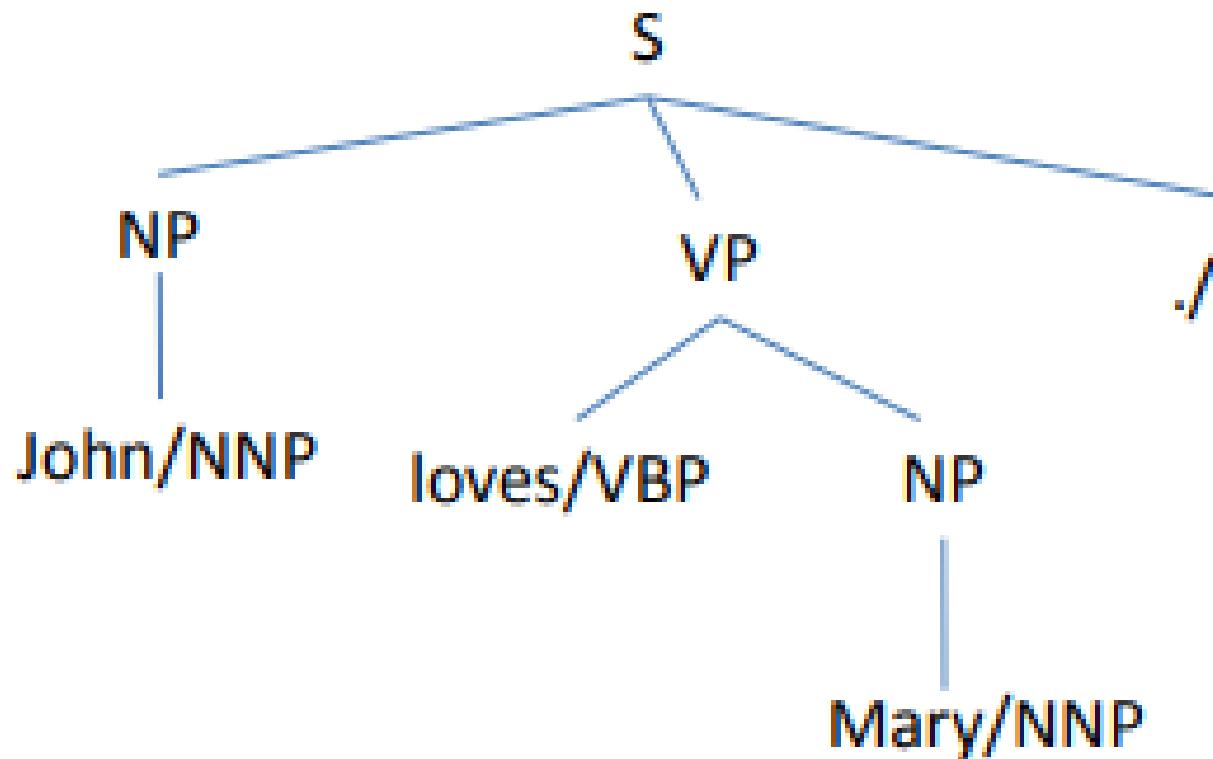


Figure: Tree bank representation

# Phrase structured vs dependency structured tree bank (1 of 2)

- Suitable for languages with fixed word order patterns and clear constituency structures.
- Dependency representations, is found adequate for languages which allow greater freedom of word order in which linearization is controlled more by pragmatic than by syntactic factors.

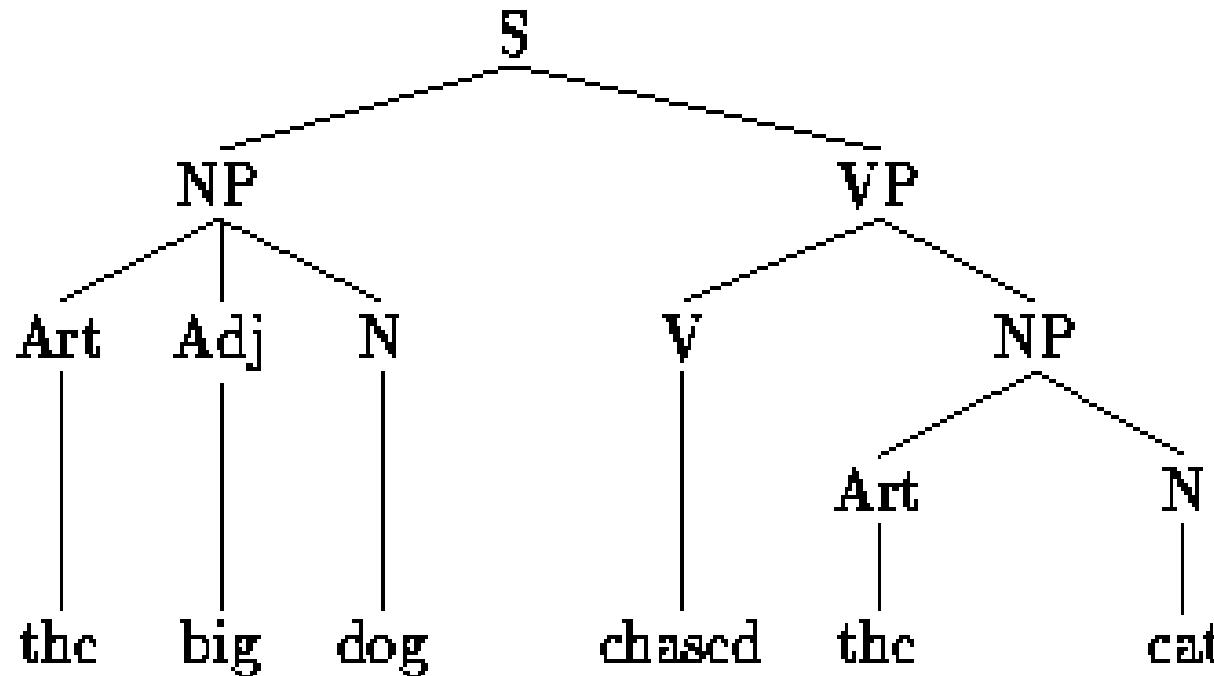


Figure: Phrase Structure tree

# Phrase structured vs dependency structured tree bank (2 of 2)



IBM ICE (Innovation Centre for Education)

- Dependency trees can be represented with arrows pointing from the head to the dependents or from the dependents to the heads.



Short Notation :

[V chased [N dog [det the] [Adj big]] [N cat [Det the]]]

Elaborate Notation :

[PRED chased

    [SUBJ dog  
        [DET the]  
        [ATTR big] ]  
    [DIROBJ cat  
        [DET the] ]

S.No	Word	Type	Dependent on
1	The		3
2	big		3
3	dog	Subj	4
4	chased	Main	V
5	the		6
6	cat	Obj	

# Self evaluation: Exercise 4

- To continue with the training, after learning the basics of Corpus and Corpora management and various steps in handling the Natural Language Text Processing, it is time to write code to work with Corpus Creation and use it to perform the activities in Computational Linguistics. It is instructed to utilize the concepts of reading data from files handling the text using beautiful soap 4 and to generate dump from genism library to perform the following activities.
- You are instructed to write the following activities using Python code.
- Exercise 4: Python code to create a text Corpora from Wikipedia through stripping down all markup and check the Corpus created.
- Exercise 5: Code to Create a Bag of Words Model to Process Raw Text.

# Self evaluation: Exercise 5

- To continue with the training, after learning the basics of Corpus and Corpora management and various steps in handling the Natural Language Text Processing, it is time to write code to work with Corpus Creation and use it to perform the activities in Computational Linguistics. It is instructed to utilize the concepts of reading data from files handling the text using beautiful soap 4 and to generate dump from genism library to perform the following activities.
- You are instructed to write the following activities using Python code.
- Exercise 4: Python code to create a text Corpora from Wikipedia through stripping down all markup and check the Corpus created.
- Exercise 5: Code to Create a Bag of Words Model to Process Raw Text.

# Fundamental statistical techniques

- Information extraction
  - Named entities
  - Relationships between entities
- Finding linguistic structure
  - Part-of-speech tagging
  - “Chunking” (low-level syntactic s
  - Parsing

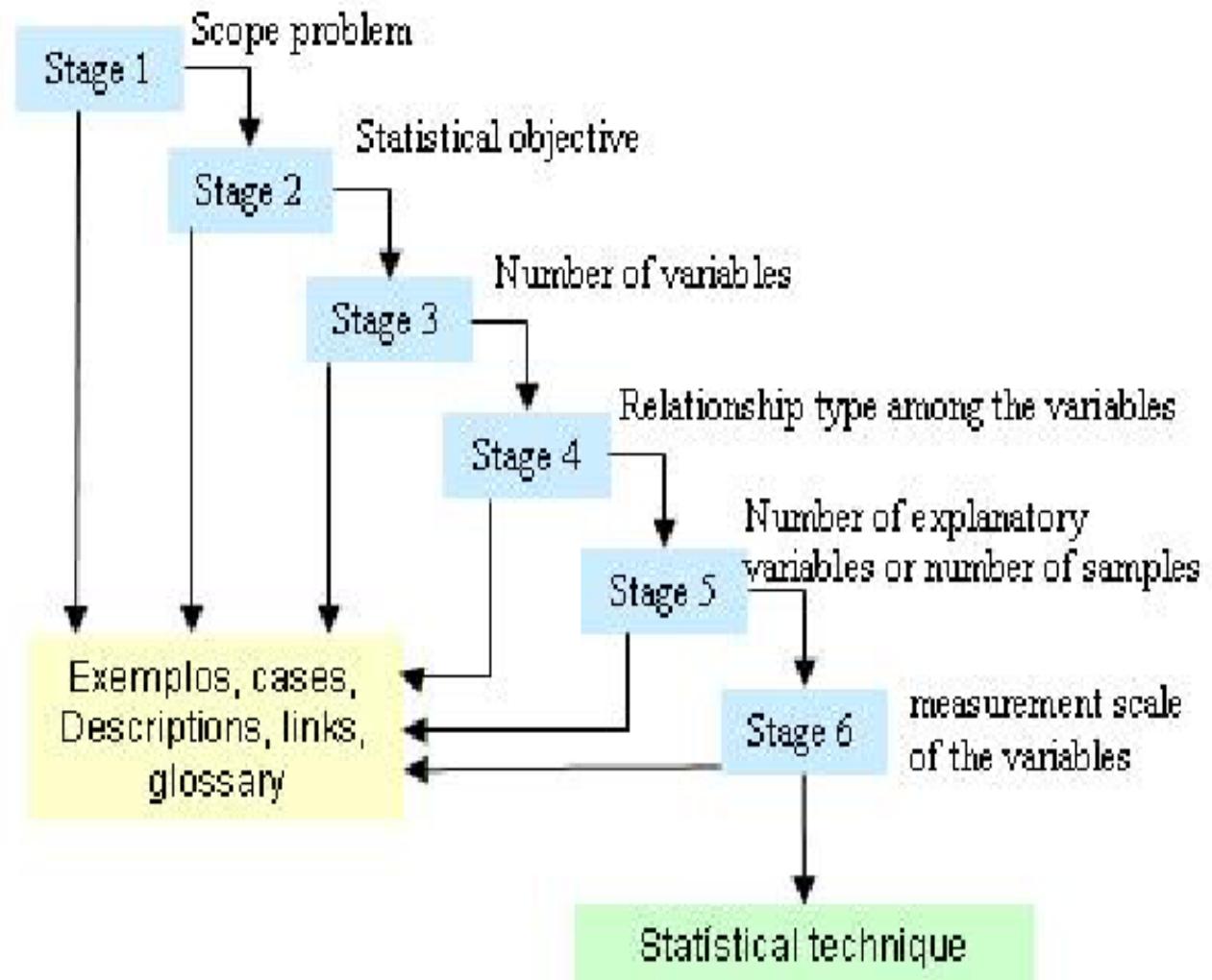


Figure: Machine translation

# Problems of the traditional approach

- Language acquisition.
- Language change.
- Language variation.
- Statistical distributions.
- Structural ambiguity.
- Robustness.
- Practicality.
- Brittleness.
- Instance of a knowledge representation.

# How statistics helps

- Disambiguation.
- Degrees of grammaticality.
- Structural preferences.
- Error tolerance.
- Learning on the fly.
- Lexical acquisition.

# Problems of the traditional approach and how statistics helps



IBM ICE (Innovation Centre for Education)

- Hidden Markov Model (HMM).
- Maximum likelihood estimation.
- Decision trees.
- N-gram.
- Maximum Entropy.
- Support vector machines.
- Conditional random fields.

# Hidden Markov model

- Based on probabilistic calculations.
- Initiating and assigning the joint probability value to the observation and label pair.

•

Positive data: Helps in the scaling.

- Elements needed to define an HMM are as follows
  - N – Number of distinct states in the model.
  - M – Number of distinct output symbols in the alphabet of the HMM.
  - $A = \{a_{ij}\}$  – State transition probability distribution.
  - $B = \{b_j(k)\}$  – Observation symbol probability distribution.
  - $\Pi = \{\Pi_i\}$  – Initial state distribution.

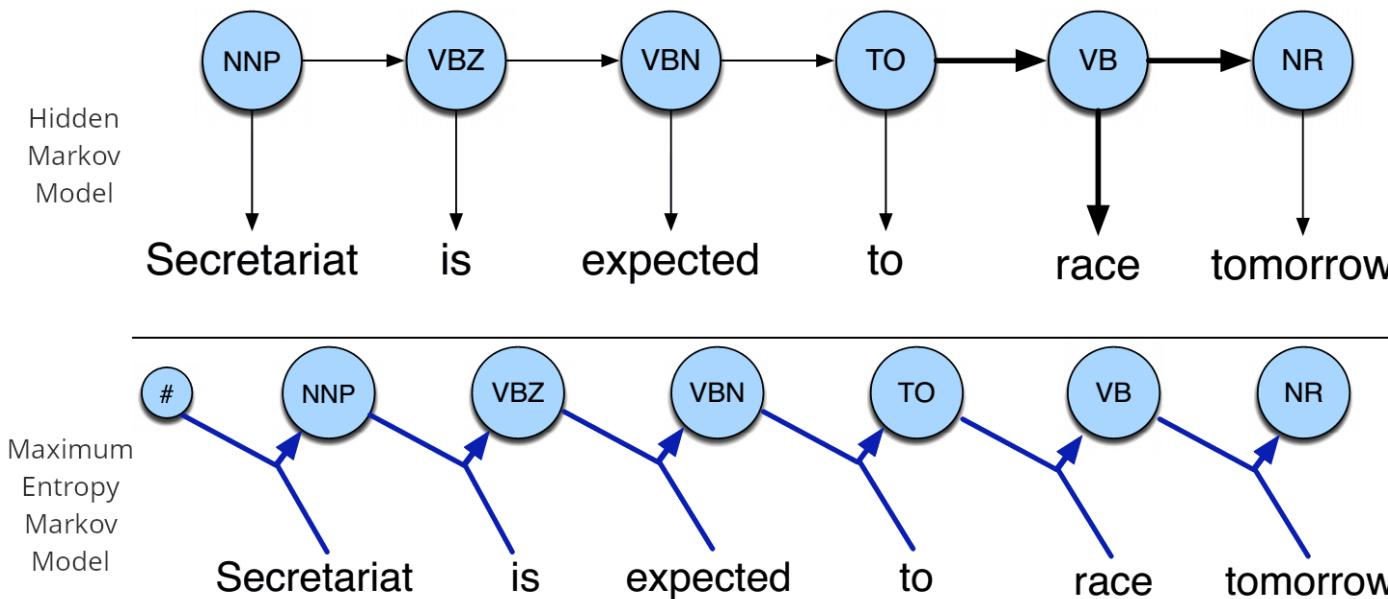
# Hidden Markov model

- Number of distinct states
  - N: Number of tags.
- Number of distinct output symbols
  - M: Number of words.
- State transition probability
  - A: Probability of the process from state “ i ” to state “ j ” in a single transition.
- Observation symbol probability
  - B: Probability that the kth output symbol will be emitted when the model is in state j.
- Initial state distribution
  - $\pi$ : Probability that the model will start in state i.

$$\operatorname{argmax}_{t_1, \dots, t_n} \prod_{i=1}^n (\underbrace{P(t_i | t_{i-1})}_{\text{Transition probability}} * \underbrace{P(w_i | t_i)}_{\text{Emission probability}})$$

# Maximum entropy Markov model

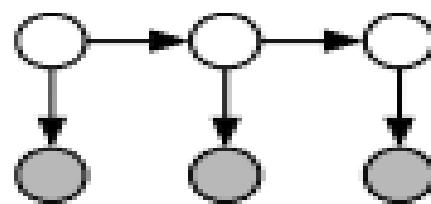
- It is a conditional probabilistic sequence model.
- Has the ability to work with long-term dependency models.
- Core principle to this model is maximum entropy.



$$\Pr(l|c) = \frac{1}{z(c)} \exp \left( \sum_{j=1}^k \lambda_j f_j(l, c) \right)$$

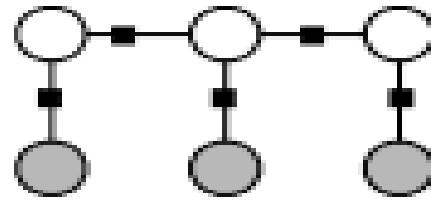
# Conditional random field model

- Used to solve sequence-labelling problem.
- Predict the sequences of labels or tags that can be used over any input data.
- CRFs are undirected graph-based models.

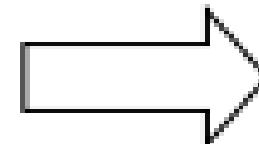


HMMs

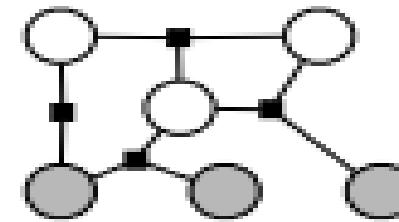
CONDITIONAL



Linear-chain CRFs



GENERAL  
GRAPHS



General CRFs

# Conditional random field model

- Higher the features higher the accuracy of the application model.

$$P(\Lambda(s|o)) = \frac{1}{z_0} \exp\left(\sum_{t=1}^T \sum_k \lambda_k f_k(s_{t-1}, s_t, o, t)\right)$$

# Support vector machine

- Used to solve two class pattern recognition problems of NLP.

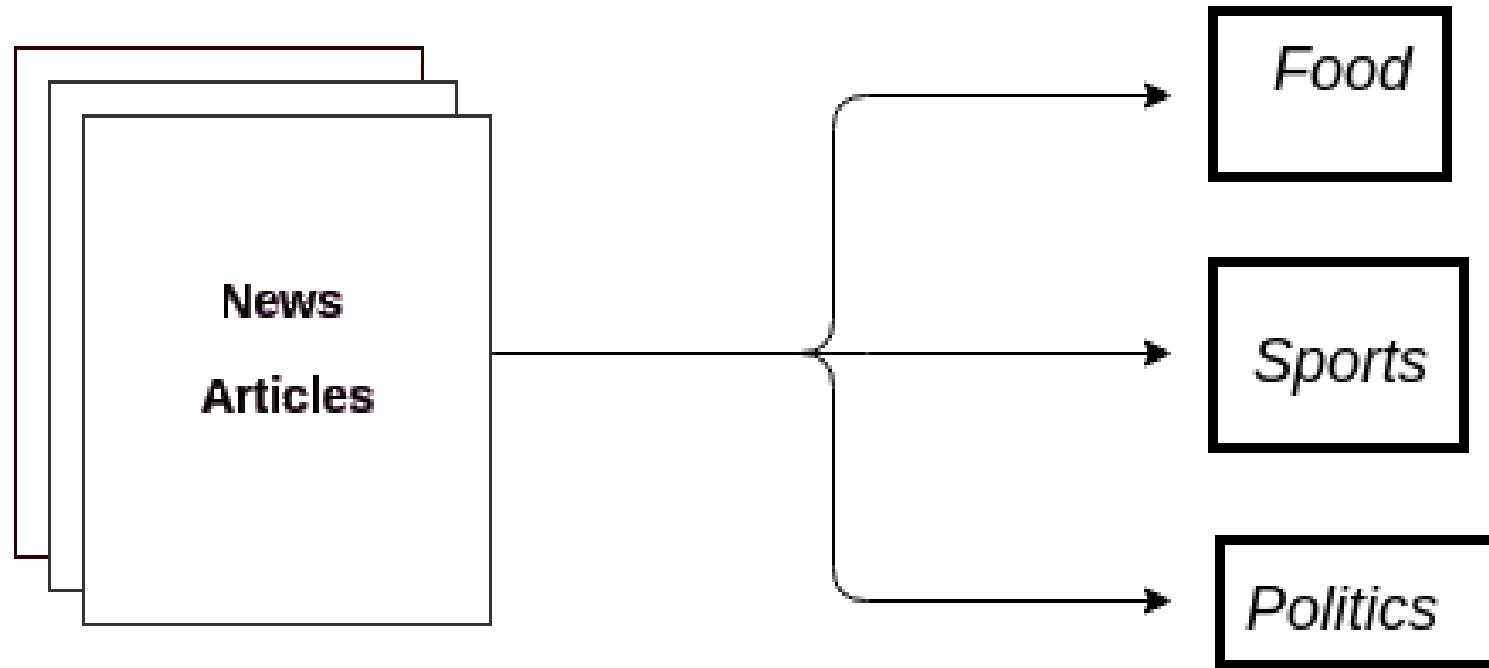
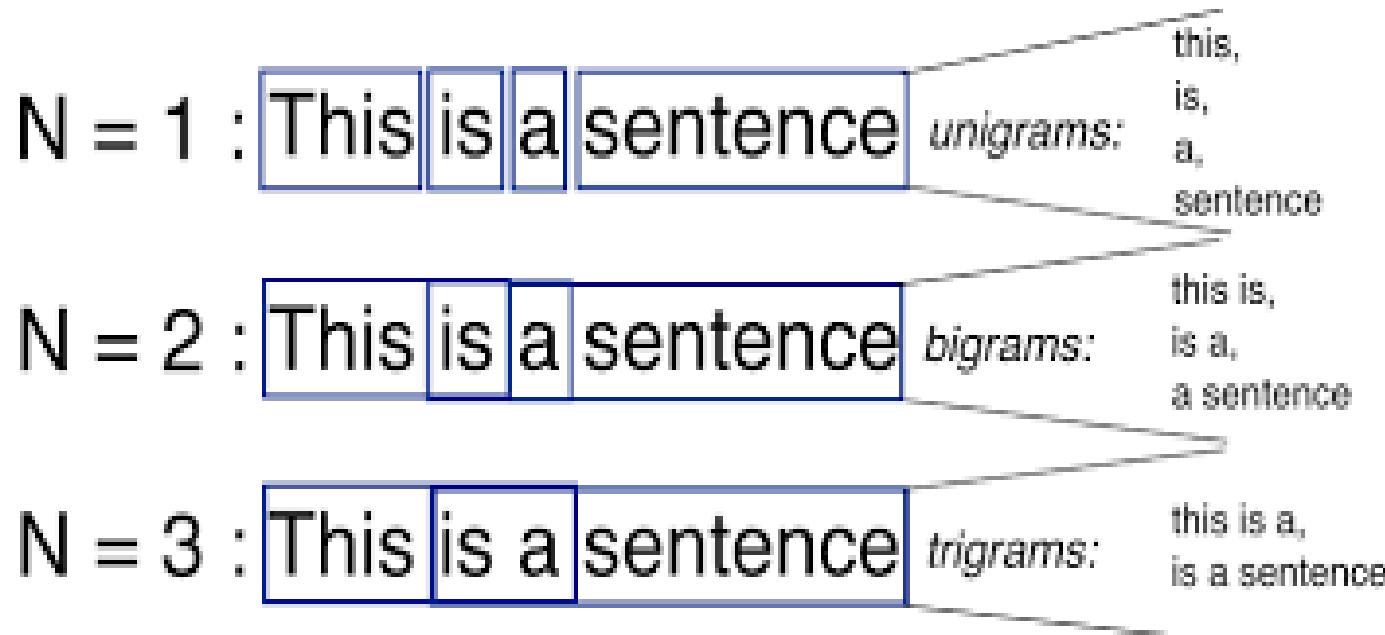


Figure: Solve two class pattern recognition problems of NLP

# N-GRAM

- Probability based statistical technique.
- Annotated corpus: Mandatory requirement.
- Accuracy increases: Increase in size and availability of the training corpus.



# Genetic algorithm

- Work towards identifying better solutions.
- Evolutionary computing: interaction between the agents.

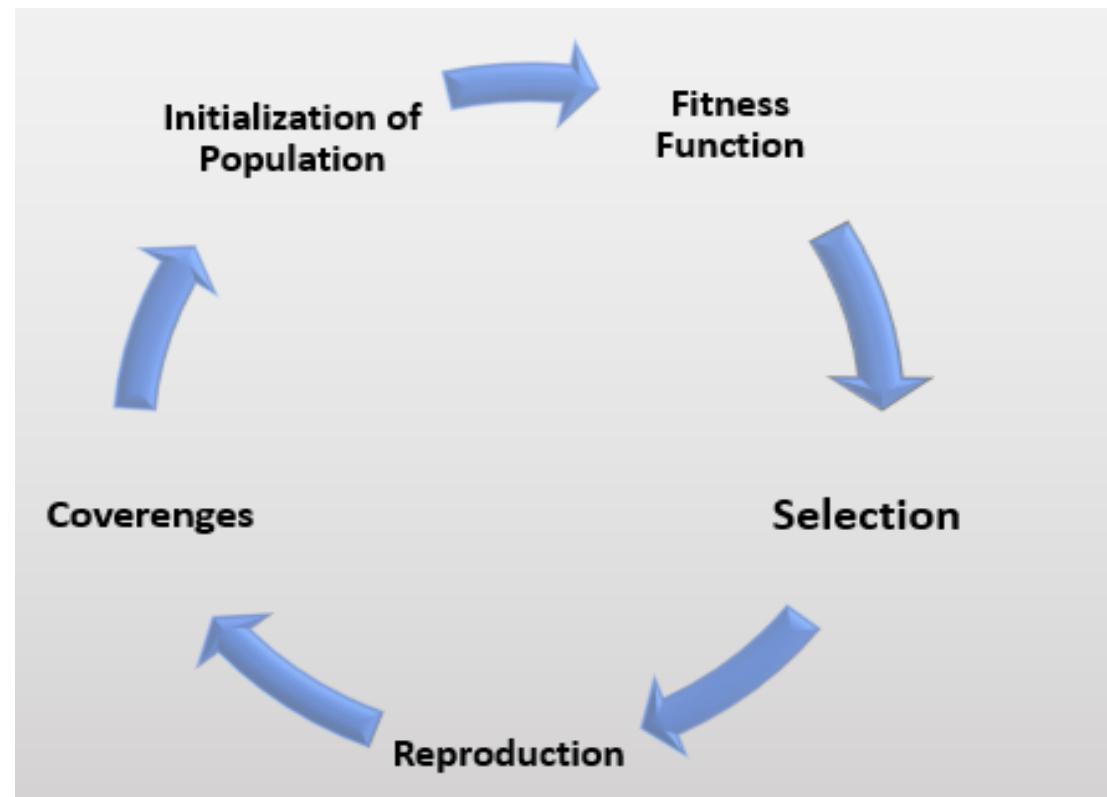


Figure: Genetic algorithm

Source: <https://www.educba.com/what-is-genetic-algorithm/>

# POS Tagging

- POS tagging: Word category disambiguation: Identifying the part of the speech.

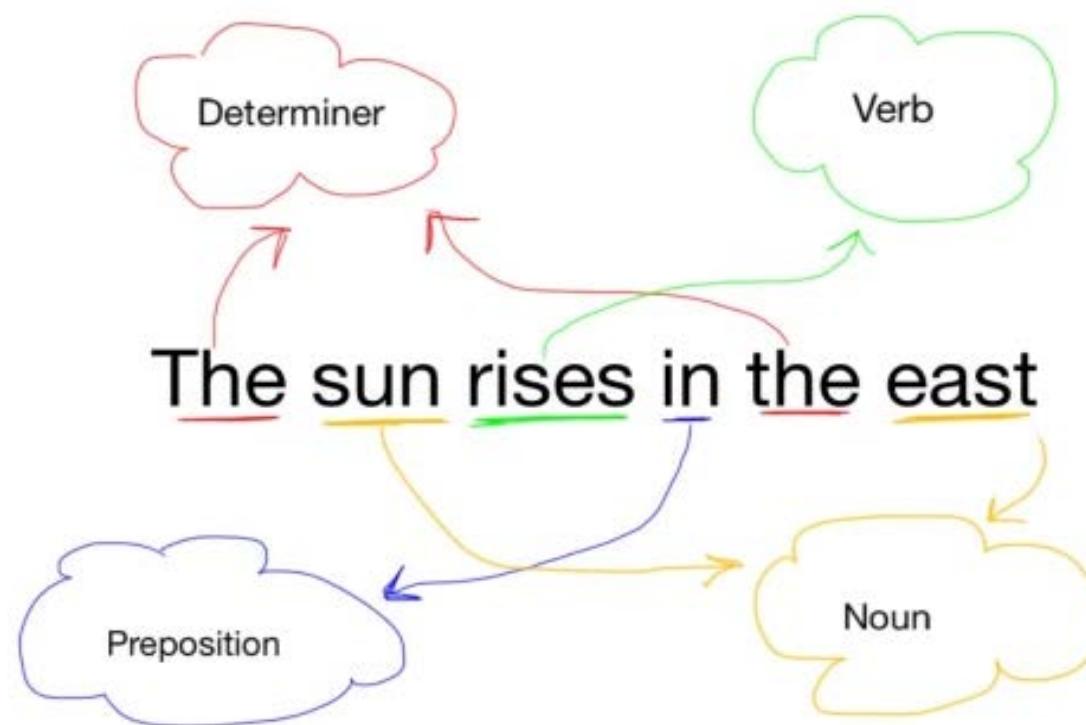


Figure: POS Tagging

Source: <https://nlp1000.wordpress.com/2016/12/19/pos-tagging-scikit-learn/>

# Word sense disambiguation

- Pronunciation of the words are important to understand the relevance and the meaning half a word in the sentence.
- Example: They refuse to permit us to obtain the refuse permit.
  - Verb: (/rə'fyooōz/): denial
  - Noun: (/ref,yoōs/): disposal

# Word sense disambiguation

- Example: Time Flies like an arrow.
- Various interpretations depending upon the parts of the speech that are tagged to individual words.

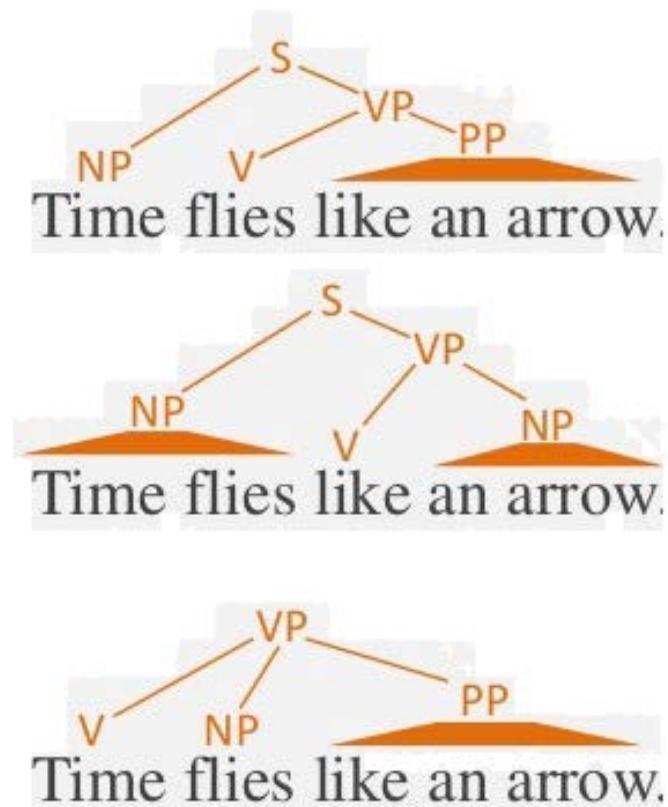


Figure: Word sense disambiguation

Source: <https://www.freecodecamp.org/news/an-introduction-to-part-of-speech-tagging-and-the-hidden-markov-model-953d45338f24/>

# POS tag and tagsets

- Special label is used to indicate and identify the part of the speech any word belongs to.
- All POS tags: Corpus is called as a tagset.
- POS tags: Automate text processing.
- POS tagging: POS annotation.
  - Manual annotation.
  - Automatic annotation.

# Types of POS taggers

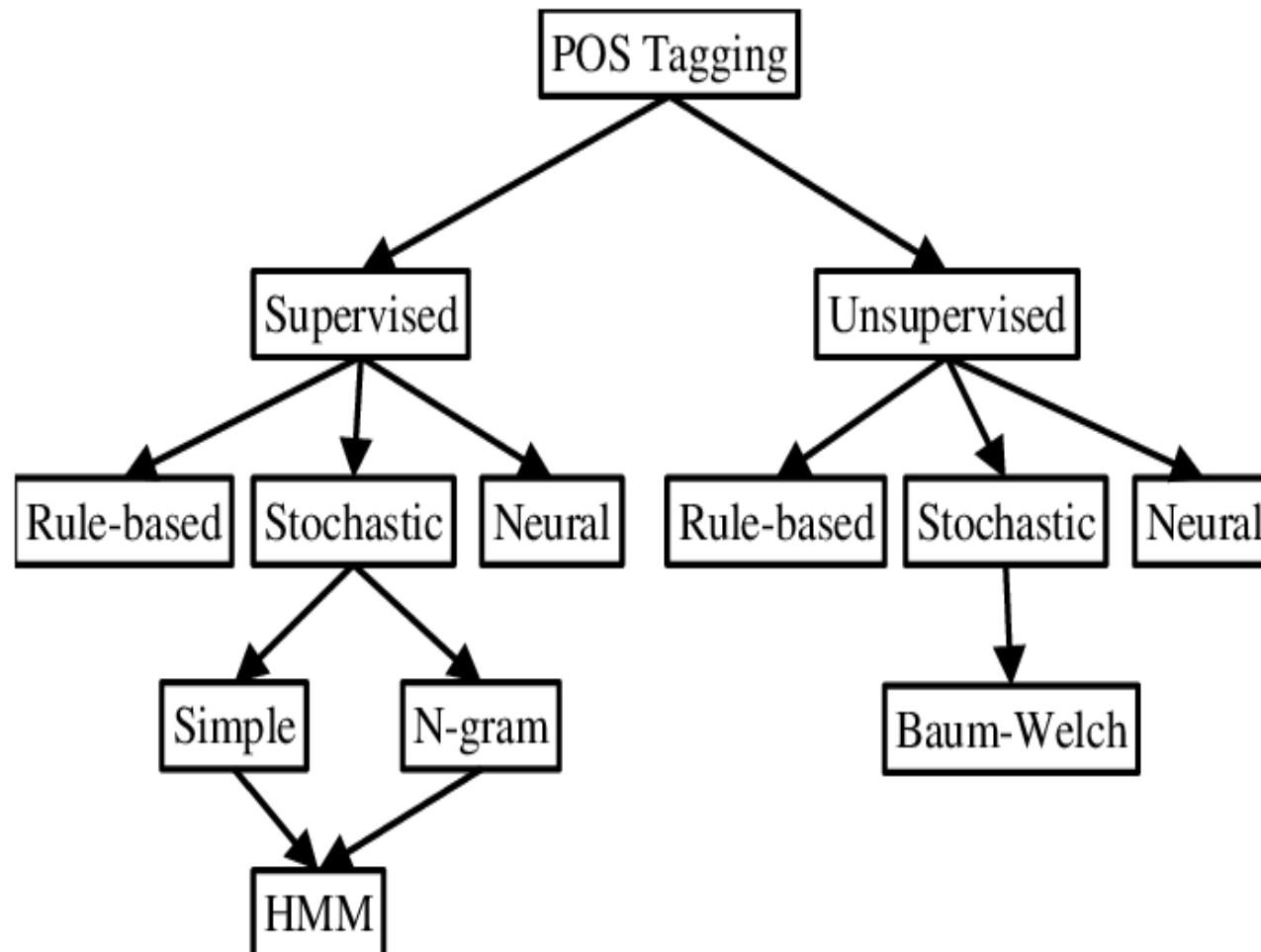


Figure: POS taggers types

Source: [https://www.researchgate.net/figure/Methods-for-implementing-POS-tagger\\_fig1\\_331684946](https://www.researchgate.net/figure/Methods-for-implementing-POS-tagger_fig1_331684946)

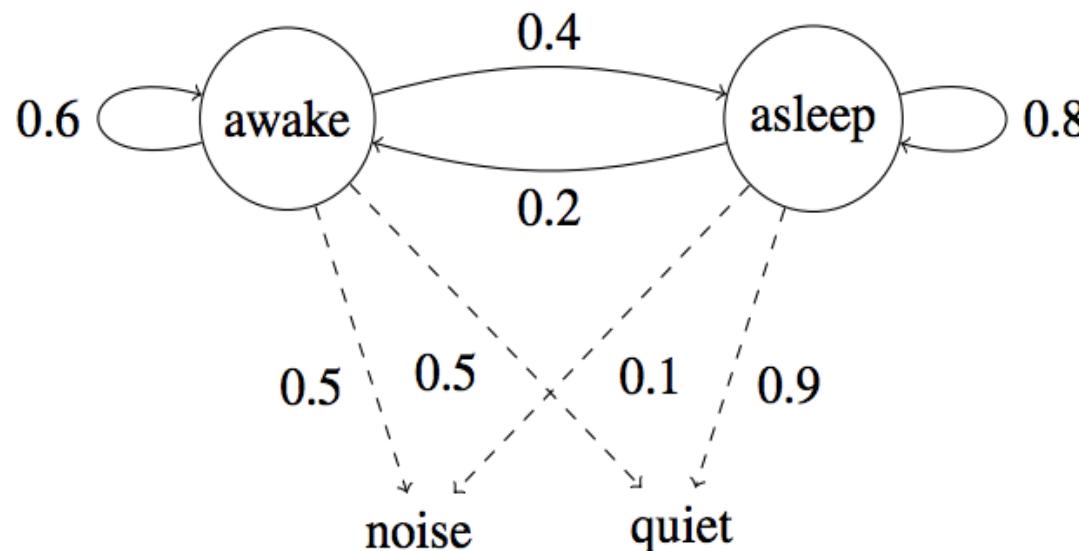
# Markovian model

- Prediction of elements can be achieved better than by other methods.
- Example: Common Weather conditions:
  - Rainy
  - Cloudy
  - Sunny
- Previous weather condition pattern:  
{ Sunny, Rainy, Cloudy, Cloudy, Sunny, Sunny, Sunny, Rainy }

$$P(q_1, \dots, q_n) = \prod_{i=1}^n P(q_i | q_{i-1})$$

# Hidden Markov model

- To identify the state of a model, simple Markovian design with the following states, observation and probabilities are provided.



	noise	quiet
awake	0.5	0.5
asleep	0.1	0.9

	awake	asleep
awake	0.6	0.4
asleep	0.2	0.8

Figure: Hidden Markov model

Source: <https://www.freecodecamp.org/news/an-introduction-to-part-of-speech-tagging-and-the-hidden-markov-model-953d45338f24/>

# POS tagging using HMM

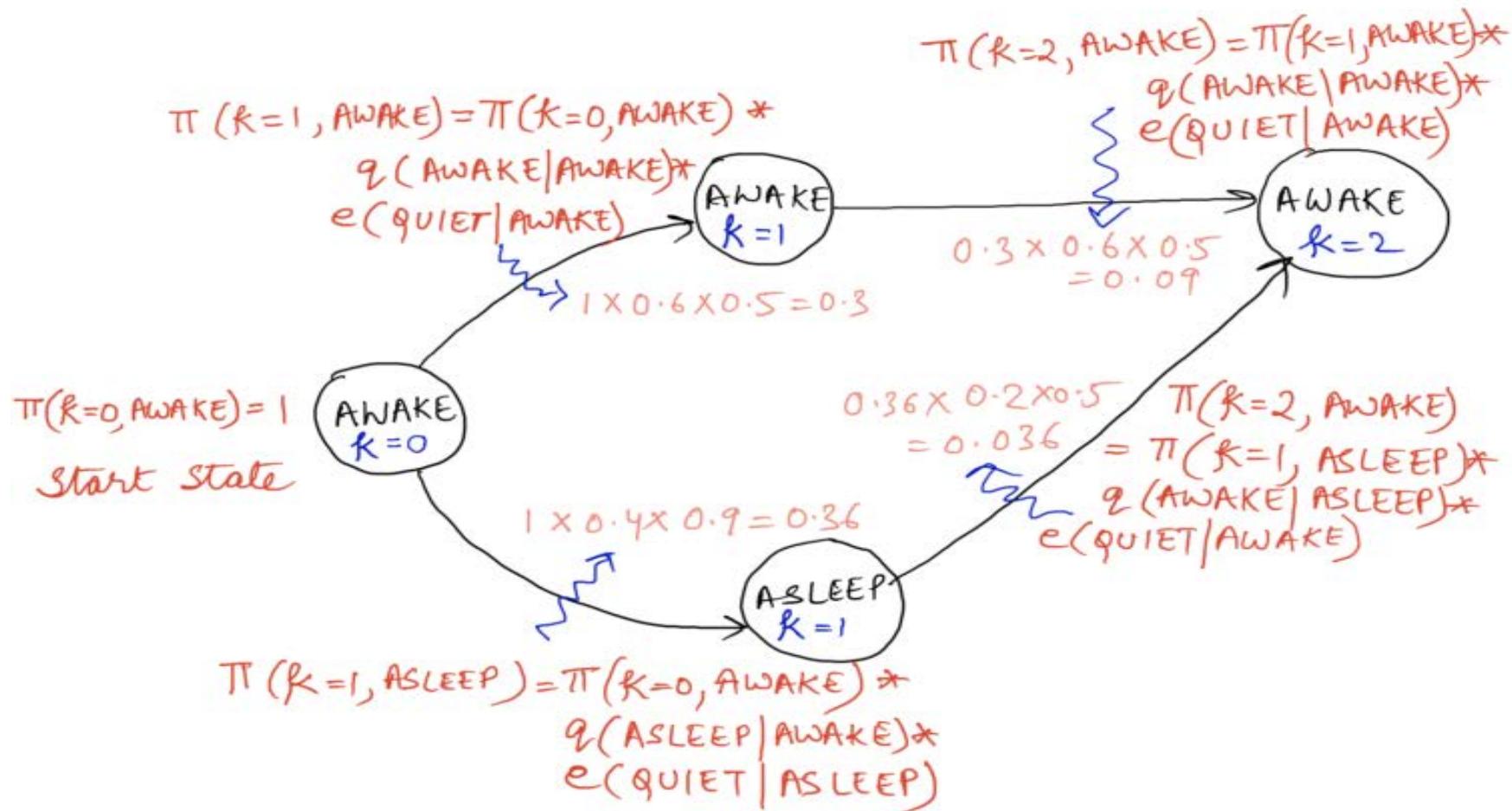


Figure: Hidden Markov Model statistical approach parts of speech

Source: <https://www.freecodecamp.org/news/a-deep-dive-into-part-of-speech-tagging-using-viterbi-algorithm-17c8de32e8bc/>

# Self evaluation: Exercise 6

- To continue with the training, after learning the various statistical methods and POS tagging in detail and how it works in Natural Language Text Processing, it is time to write code to work POS tagging using any statistical model and use it to perform the tagging activities in NLP Corpora. It is instructed to utilize the concepts of SKLEARN, Metrics, CRF model and POS tagging and create Transition and State Features to perform the following activities.
- You are instructed to write the following activities using Python code.
- Exercise 6: Code to Create Part of Speech Tags using Conditional Random Fields.

# Checkpoint (1 of 2)

## Multiple choice questions:

1. Traits of a good text corpus are.
  - a) Metadata
  - b) Genre
  - c) Size
  - d) All the above
  
2. Where does the additional variables are added in HMM?
  - a) Temporal model
  - b) Reality model
  - c) Probability model
  - d) All the above
  
3. How many bi-grams can be generated from given sentence:  
“IBM learning is a great place to learn Computational Linguistics”
  - a) 7
  - b) 8
  - c) 9
  - d) 10

# Checkpoint solutions (1 of 2)

## Multiple choice questions:

1. Traits of a good text corpus are.
  - a) Metadata
  - b) Genre
  - c) Size
  - d) All the above

2. Where does the additional variables are added in HMM?
  - a) Temporal model
  - b) Reality model
  - c) Probability model
  - d) All the above

3. How many bi-grams can be generated from given sentence:

“IBM learning is a great place to learn Computational Linguistics”

- a) 7
- b) 8
- c) 9
- d) 10

# Checkpoint (2 of 2)

## Fill in the blanks:

1. Treebanks are necessarily constructed according to the \_\_\_\_\_ of a language
2. An example of annotating a corpus is \_\_\_\_\_.
3. Hidden Markov models are statistical models based on \_\_\_\_\_ design.

## True or False:

1. Treebank is a linguistically annotated corpus based on POS tags. True/False
2. POS tagging can be performed without proper analysis of the Corpus structure. True/False
3. N – gram model is a probability-based model. True/False

# Checkpoint solutions (2 of 2)

## Fill in the blanks:

1. Treebanks are necessarily constructed according to the grammar of a language
2. An example of annotating a corpus is part-of-speech tagging.
3. Hidden Markov models are statistical models based on generative design.

## True or False:

1. Treebank is a Linguistically Annotated Corpus based on POS tags. **True**
2. POS tagging can be performed without proper analysis of the Corpus structure. **False**
3. N – gram model is a probability-based model. **True**

# Question bank

## Two mark questions:

1. What is the purpose of creating Corpus for any language?
2. What are the uses of genetic algorithms in NLP?
3. What are the various types of Tree Banks?
4. What are stochastic POS Taggers?

## Four mark questions:

1. Discuss the steps involved in the construction of a treebank.
2. What are the linguistic description layers used in NLP?
3. Differentiate phrase structured and dependency structured tree bank.
4. Write about N-gram and SVM statistical model.

## Eight mark questions:

1. Explain annotation life cycle with an example in detail.
2. Explain with example the working of POS tagging with HMM model.

# Unit summary

**Having completed this unit, you should be able to:**

- Understand the concepts of various empirical approaches
- Learn on how to work with creation of corpus, approaches and working principles
- Gain knowledge on nuances of annotation, know about treebank annotation and how to create and use it
- Understand the concepts various fundamental statistical techniques used in annotations and their principles
- Gain an insight into part-of-speech tagging principles along with the various techniques to implement it

# Welcome to:

# Statistical Approaches



# Unit objectives

- After completing this unit, you should be able to:
- Understand what is statistical parsing and the core concepts involved in it
- Learn about multiword expressions and how to handle them
- Understand the concepts of word similarity and the relativity calculations done
- Gain knowledge on word sense disambiguation and why it is needed in NLP
- Gain an insight into modern speech recognition techniques with an idea of the forerunners in the field
- Understand what statistical machine translation means and the guidelines needed to perform SMT

# Parsing (1 of 2)

- An activity in computational linguistics and natural language processing identification and understanding of the syntax and semantics based on the grammar of the natural language.
- Parser is a tool for computation that can process any input sentence within the boundaries of the productions in the grammar and build structures called as parse trees within the confinement of the grammatical rule.
- Example: Sentence- Tom ate an apple.

```
sentence -> noun_phrase, verb_phrase  
noun_phrase -> proper_noun  
noun_phrase -> determiner, noun  
verb_phrase -> verb, noun_phrase  
proper_noun -> [Tom]  
noun -> [apple]  
verb -> [ate]  
determiner -> [an]
```

Figure: Grammar

# Parsing (2 of 2)

- Parsing of the statement constructs a parse tree with root, intermediate nodes, which are also called as non-terminal nodes, and leaf nodes called as terminal nodes.

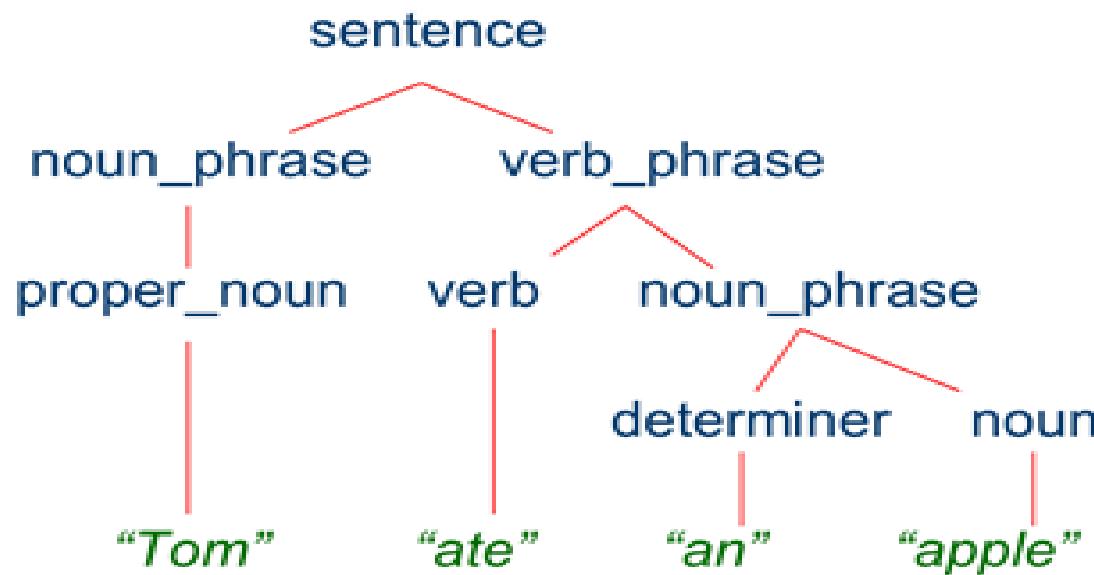


Figure: Parse Tree

Source: <https://forum.huawei.com/enterprise/en/what-is-parsing-in-nlp/thread/571685-100429>

# Statistical parsing (1 of 2)

- Association between grammar of the natural language and the probability of its occurrence.
- Statistical parsing associates every grammar rule with a probability value.
- Example: Sentence: The can can hold water.

# Statistical parsing (2 of 2)

- Large amount of grammatical rules → Very large search space.
- Optimization on the subsets of the parse trees generated.
- Dissimilar parse trees → Frequency identification.
- Similar parse trees → Discarded.
- Statistical parsing → Better performance → Vocabulary is very large.
- Lexicalized and Statistical Parsing (LSP) → Balance the vocabulary.

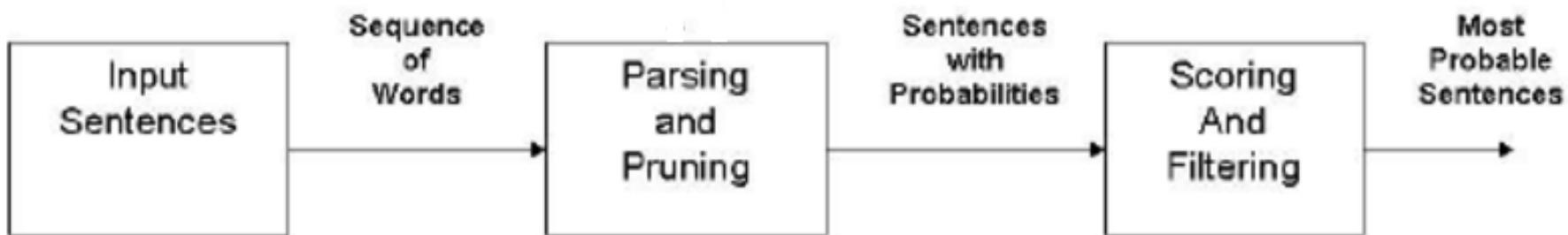


Figure: Statistical Parsing Steps

Source: [https://www.researchgate.net/figure/Framework-of-Lexicalized-and-Statistical-Parser\\_fig1\\_220155897](https://www.researchgate.net/figure/Framework-of-Lexicalized-and-Statistical-Parser_fig1_220155897)

# Approaches to parsing

- Understands syntax and semantics.
- Parser → Process an input sentence according to the production rules → Parse trees.

Structural Approach:

- Context free grammar (CFG) → Group of consecutive words.

	Statistical	Structural
Foundation	Statistical decision theory	Human perception and cognition
Description	Quantitative features Fixed number of features Ignores feature relationships Semantics from feature position	Morphological primitives Variable number of primitives Captures primitive relationships Semantics from primitive encoding
Classification	Statistical classifiers	Parsing with syntactic grammars

Figure: Structural Approach vs Statistical Approach

Source: [https://www.byclb.com/TR/Tutorials/neural\\_networks/ch1\\_1.htm](https://www.byclb.com/TR/Tutorials/neural_networks/ch1_1.htm)

# Statistical approach

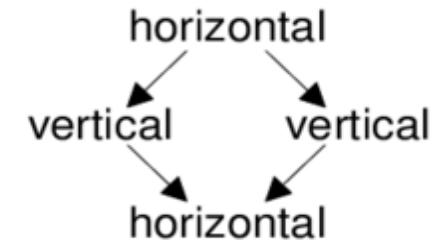
- Statistical approaches are data driven.
- Concentrate upon short-term relationship between the words in a sentence.

## Statistical

Number of segments: 4  
Number of horizontal segments: 2  
Number of vertical segments: 2  
Number of diagonal segments: 0



## Structural



Number of segments: 3  
Number of horizontal segments: 1  
Number of vertical segments: 0  
Number of diagonal segments: 2

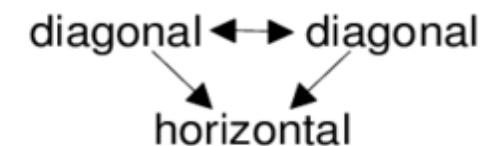


Figure: Analysis in Structural Approach vs Statistical Approach

Source: [https://www.researchgate.net/figure/The-statistical-and-structural-approaches-to-pattern-recognition-applied-to-a-common-fig3\\_228558473](https://www.researchgate.net/figure/The-statistical-and-structural-approaches-to-pattern-recognition-applied-to-a-common-fig3_228558473)

# Lexicalized statistical parsing (1 of 2)

- Context free grammar is augmented using a probabilistic component and ambiguity is resolved in lexicalized statistical parsing.
- CFG is designed for adopting the probabilistic component into itself and is called as Probabilistic Context Free Grammar (PCFG).
- The performance of PCFG enhanced by adding a conditional rule for the lexical head.

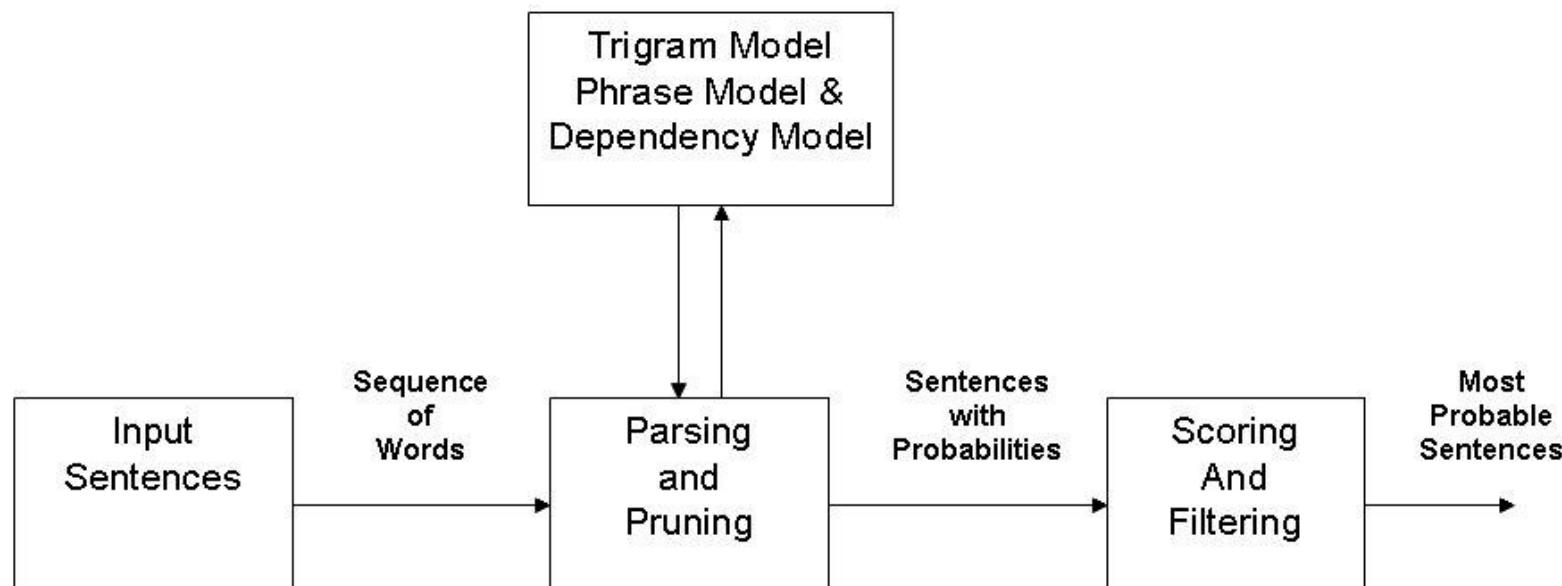


Figure: Lexicalized Statistical Parsing

# Lexicalized statistical parsing (2 of 2)

## Step 1: Lexicalization

- Remove the beginning and ending markers in a sentence.
- Removal of special characters and punctuations.
- Create a tree bank.

## Step 2: Language model construction

- Tree bank → Phrase structure or dependency structure.
- Tree bank → Generate the features, probabilities of the words.
- Model calculation → Relations between the words.
- Dependency association.

## Step 3: Statistical Parsing Implementation

- The syntax, semantics, relationship of words → Parse tree.
- Long-term relation → Higher level through the complex structures.

# Top-down parsing

- The top down parsing method begins on the top with the start symbol "S".

## Example:

- Sentence: Maybe john walks

## Grammar:

$s \Rightarrow sadv, s$

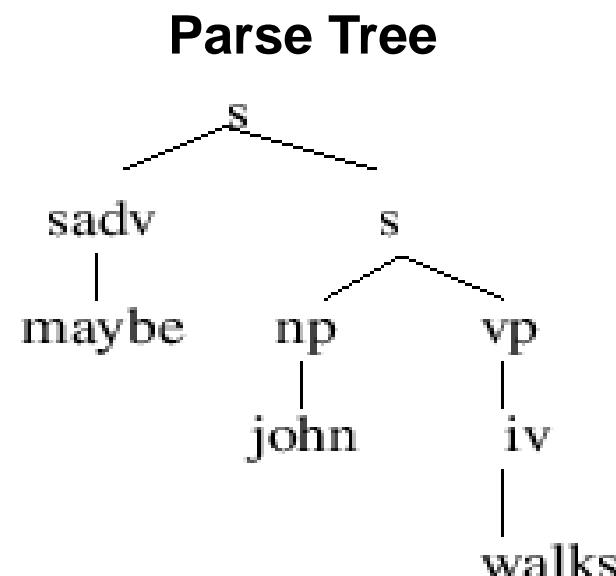
$s \Rightarrow np, vp$

$np \Rightarrow john$

$vp \Rightarrow iv$

$iv \Rightarrow walks$

$sadv \Rightarrow maybe$



# Bottom-up parsing

- The process starts from the non-terminal symbols and continues upwards replacing the individual words in two sentence phrases until it reaches the root symbol.

## Example:

- Sentence: maybe john walks

### Grammar:

$s \Rightarrow sadv, s$

$s \Rightarrow np, vp$

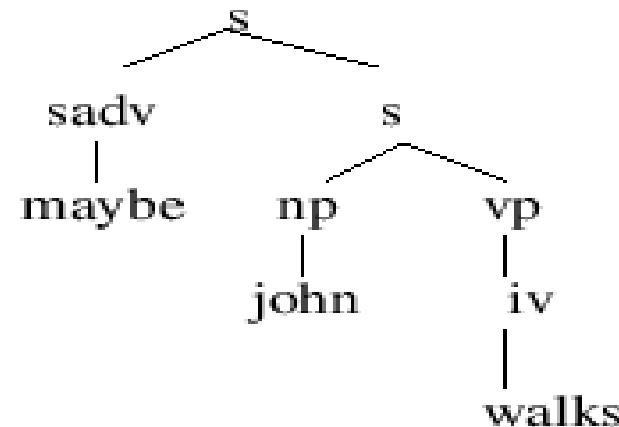
$np \Rightarrow john$

$vp \Rightarrow iv$

$iv \Rightarrow walks$

$sadv \Rightarrow maybe$

### Parse Tree



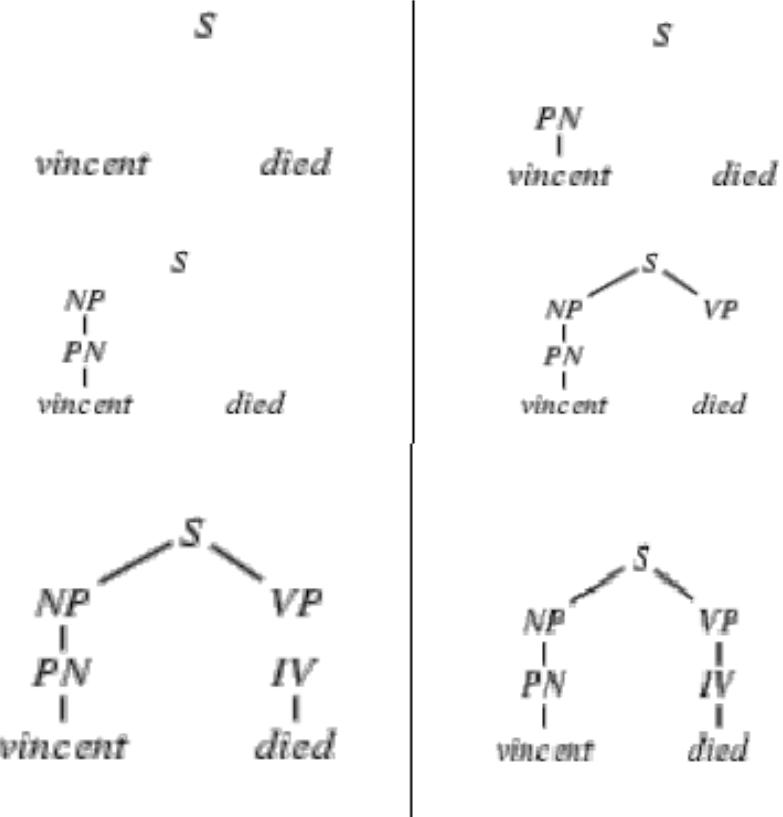
# Left corner parsing method

- Example 1: Input Statement: vincent died.

## Steps:

- Input: vincent Recognize an s. (Top-down prediction.)
- First word → pn. (Bottom-up)
- pn at left corner: np → pn. (Bottom-up)
- np at left corner: s → np vp (Bottom-up)

- LHS = RHS.
- Input: died. Recognize a vp. (Top-down.)
- First word → iv. (Bottom-up.)
- iv at left corner: vp → iv. (Bottom-up.)
- LHS = RHS



# Statistical parsing: Probabilistic parser

## Probabilistic CFG (PCFG)

- A CFG in which its re-writing rules are associated with a probability.  $p \rightarrow$  Probability of a non-terminal A expanded to sequence  $\beta$ .

$$A \rightarrow \beta [p]$$

- Probability of an expansion RHS  $\beta$  given the LHS A.

$$P(\text{RHS} | \text{LHS})$$

S → NP VP [.80]  
S → Aux NP VP [.15]  
S → VP [.05]  
NP → Pronoun [.35]  
NP → Proper-Noun [.30]  
NP → Det Nominal [.20]  
NP → Nominal [.15]

Figure: PCFG Representation

# Multiword expressions

- Multi word expressions are Idiosyncratic.
- The word that syntactically or semantically similar.

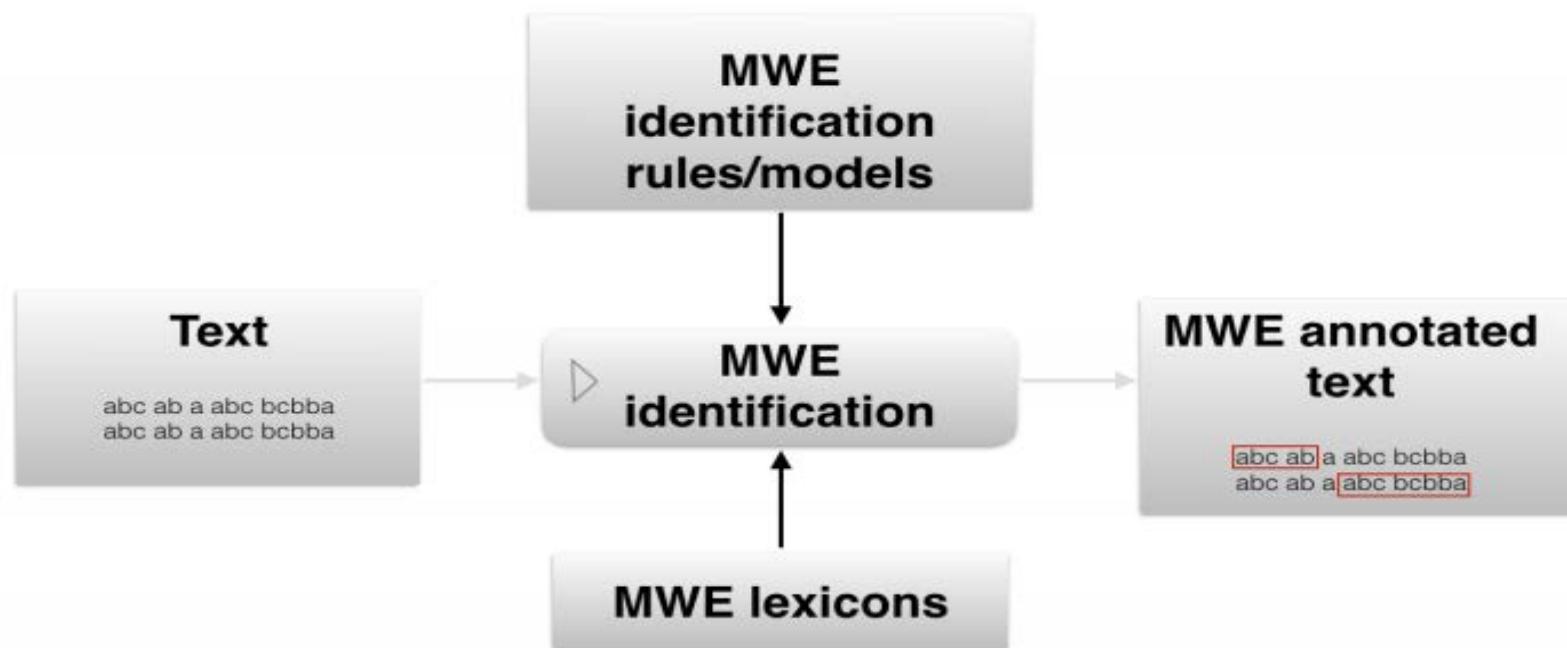


Figure: Multiword Expressions Process Outline

Source: [https://www.mitpressjournals.org/doi/pdf/10.1162/COLI\\_a\\_00302](https://www.mitpressjournals.org/doi/pdf/10.1162/COLI_a_00302)

# Features of MWE

- The MWE Can be decomposed into multiple lexemes.
- These can be represented through syntactic, semantic, pragmatic or lexical units.

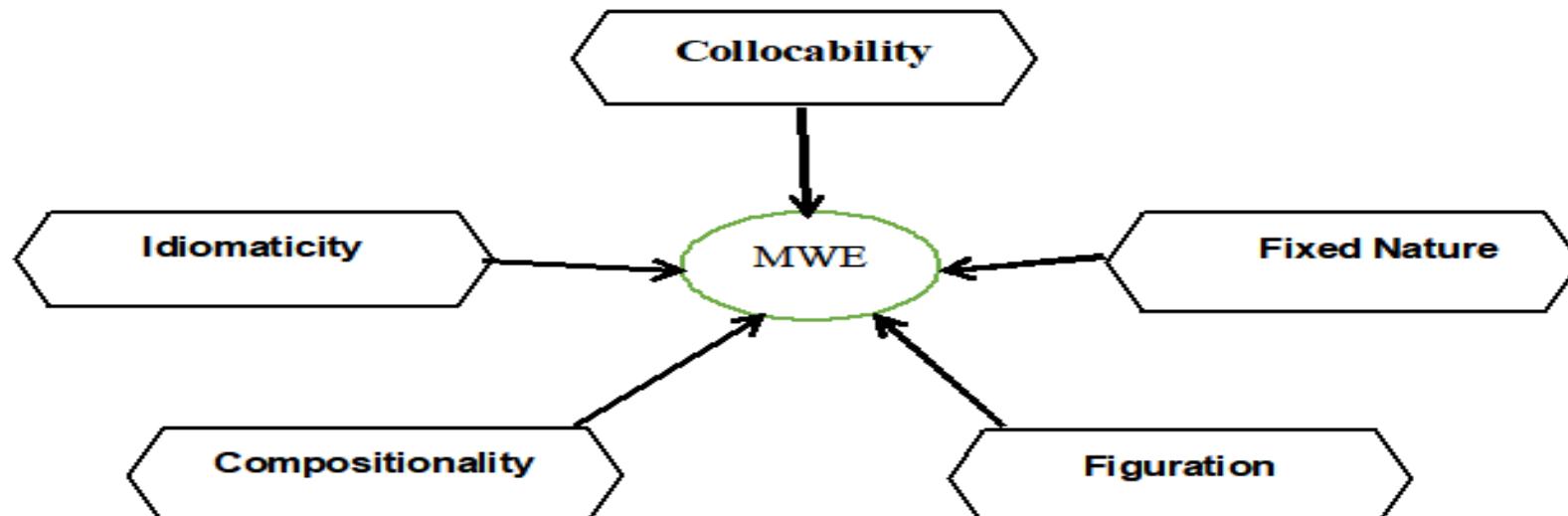


Figure: MWE Features

# Types of multi word expressions

Atkins and Rundell (2008)	Bergenholz and Gouws (2014)	Baldwin and Kim (2010)	IV. PROVERBS		
I. COLLOCATIONS			proverbs too many cooks ....	proverbs half a loaf is better than no bread	sentence-like units good-morning
collocations risk one's life	collocations severe criticism	collocations impeccable performance	quotations to be or not to be	winged words One small step for man ...	
II. FIXED PHRASES & IDIOMS			greetings good morning	routine formulas how do you do	
phrasal idioms to have a heart of gold	idioms to have eyes in the back of one's head	verb-noun idiomatic – combinations Kick the bucket	phatic phrases have a nice day	expletive constructions give him an inch and ...	
fixed phrases ham and eggs	non-pictorial idiomatic MWE round the clock		catch phrases horses for courses		
similes drunk as a lord	twin formula day and night		V. PHRASAL VERBS		
	comparative MWE as right as rain		phrasal verbs get up, see through	nonidiomatic particle verb to run all/no back in	verb-particle constructions take off
	MWEs from foreign languages ad hoc			nonidiomatic reflexive verb to enjoy yourself/to prostitute yourself	prepositional verbs refer to
	(non)idiomatic MWEs with a unique component to and fro		VI. LIGHT-VERB CONSTRUCTIONS		
	MWEs with an odd inflection		support verb constructions to take a decision	noun phrase with semantically void verb set in motion	light-verb constructions to take a walk
III. COMPOUNDS			VII. PREPOSITIONAL PHRASES		
figurative compounds lame duck	semi-terms magic eye	nominal compounds golf club, connecting flight	compound prepositions In spite of	MWEs with syntactic function with regard to	prepositional phrases in bed, in jail
semi-figurative compounds high school					complex prepositions on top of
functional compounds police dog					

# Multi word verbs

- The verbs that contain more than one word are called as multi word verbs.

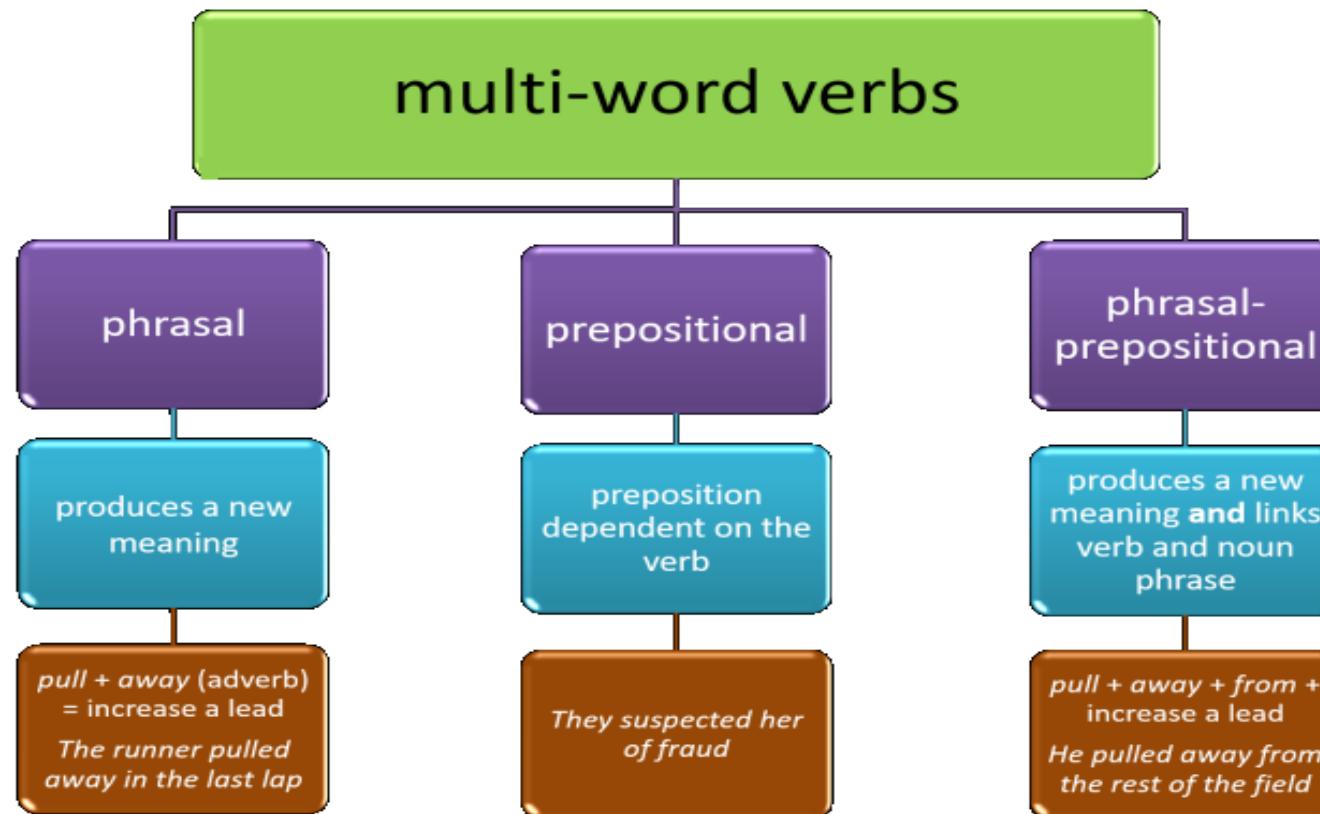


Figure: Multi Word Verbs

Source: <https://www.eltconcourse.com/training/inservice/verbs/mwvs.html>

# Word similarity and text similarity

- Helps in determining the closeness of two or more words/text.
- bag of words, TF-IDF, word2vec etc. are used to encode the input text data.



Figure: Semantic similarity and dissimilarity

Source: <https://ai.googleblog.com/2018/05/advances-in-semantic-textual-similarity.html>

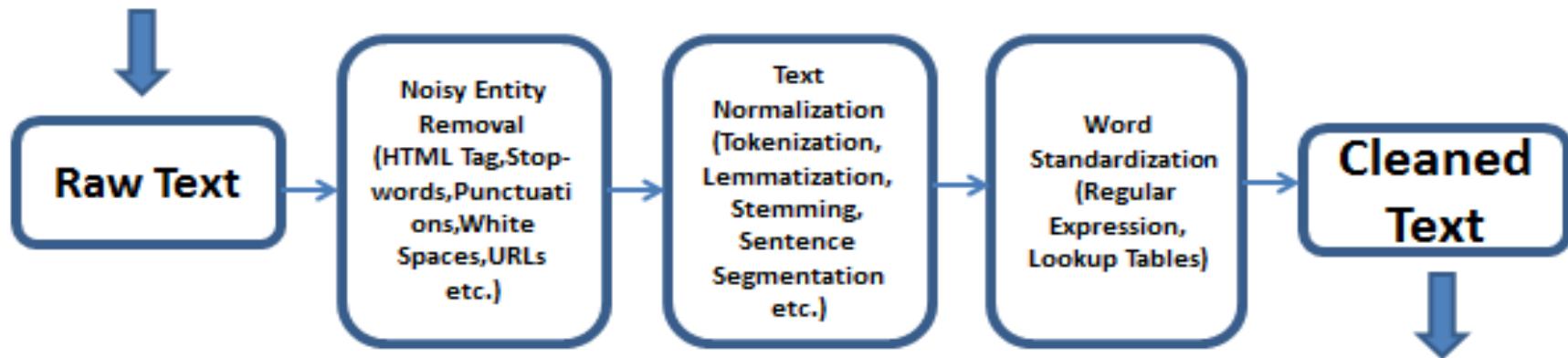


Figure: Pre-Processing of Text

Source: <http://www.vanaudelanalytix.com/python-blog/pre-processing-text-for-nlp>

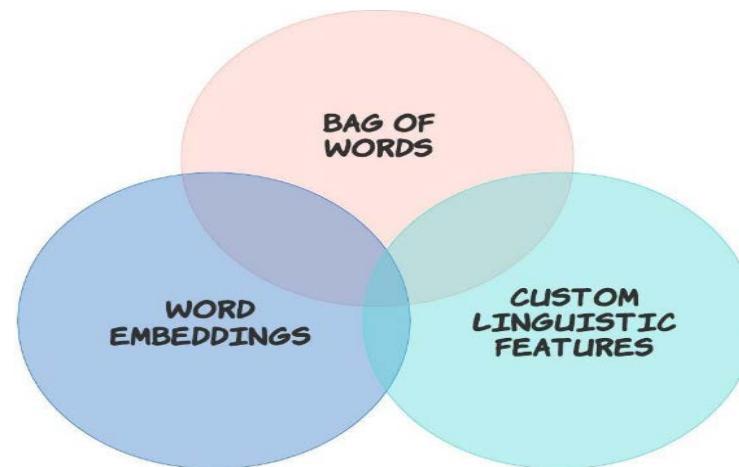


Figure: Feature extraction

Source: <https://amp.flipboard.com/@tdatascience/artificial-intelligence-8qhakstrz/the-triune-pipeline-for-three-major-transformers-in-nlp-a-WXncOskwRTGZbgY5j66HcA%3Aa%3A2892075988-6fae262963%2Ftowardsdatascience.com>

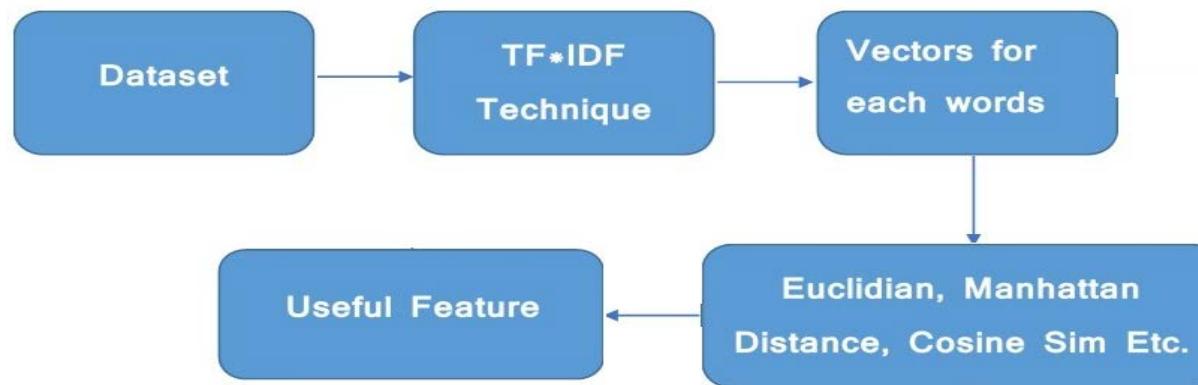


Figure: Similarity of vectors

Source: <https://medium.com/@Intellica.AI/comparison-of-different-word-embeddings-on-text-similarity-a-use-case-in-nlp-e83e08469c1c>

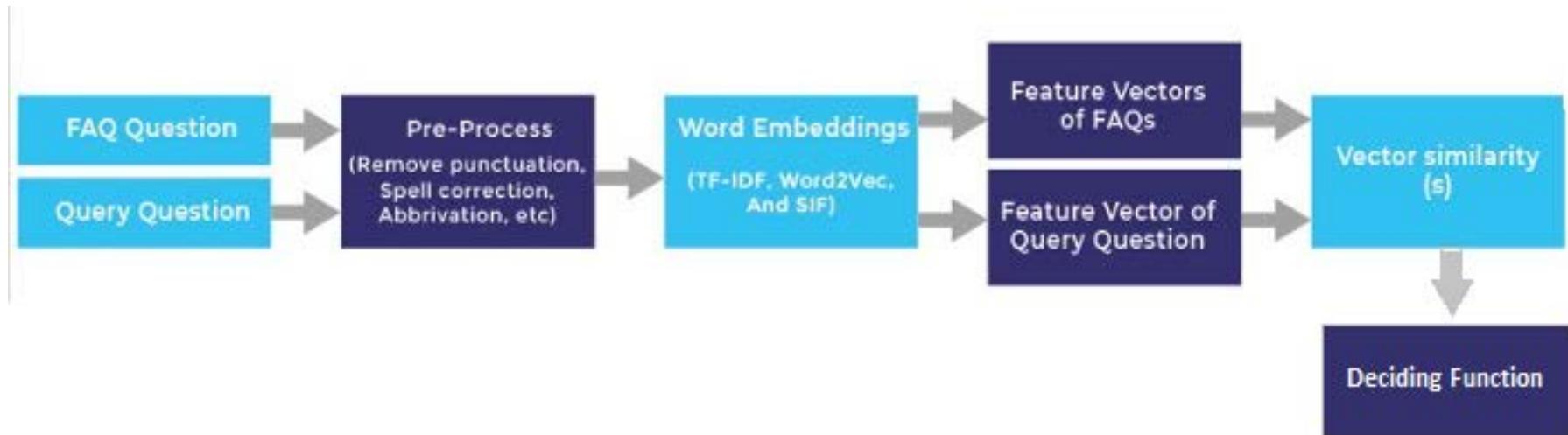


Figure: Deciding function

Source: <https://medium.com/@Intellica.AI/comparison-of-different-word-embeddings-on-text-similarity-a-use-case-in-nlp-e83e08469c1c>

# Jaccard similarity

- Simple representation of two text sentences that have common elements intersection over Union method.

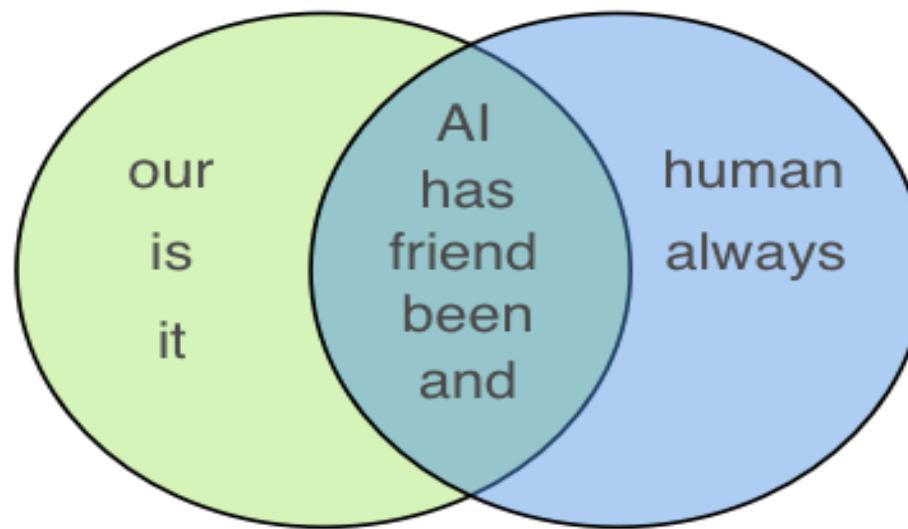


Figure: Jaccard distance

Source: <https://medium.com/@adriensieg/text-similarities-da019229c894>

# K-means

- Usage of K means algorithm for conversion of the words into appropriate vector.

	Document	Category	ClusterLabel
0	The sky is blue and beautiful.	weather	2
1	Love this blue and beautiful sky!	weather	2
2	The quick brown fox jumps over the lazy dog.	animals	1
3	A king's breakfast has sausages, ham, bacon, eggs, toast and beans	food	3
4	I love green eggs, ham, sausages and bacon!	food	3
5	The brown fox is quick and the blue dog is lazy!	animals	1
6	The sky is very blue and the sky is very beautiful today	weather	2
7	The dog is lazy but the brown fox is quick!	animals	1
8	President greets the press in Chicago	politics	4
9	Obama speaks in Illinois	politics	4

Figure: K – Means

Source: <https://medium.com/@adriensieg/text-similarities-da019229c894>

# Cosine similarity

- Uses the cos angle between the vectors.
- Measure of similarity between two non-zero vectors based on the inner product of cosine angles.

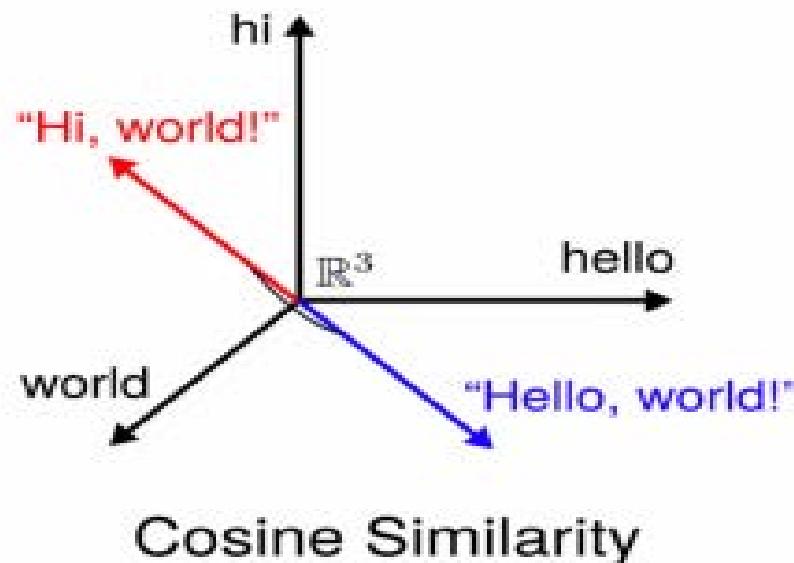


Figure: Cosine Similarity

Source: <https://medium.com/@adriensieg/text-similarities-da019229c894>

# Word Mover's distance

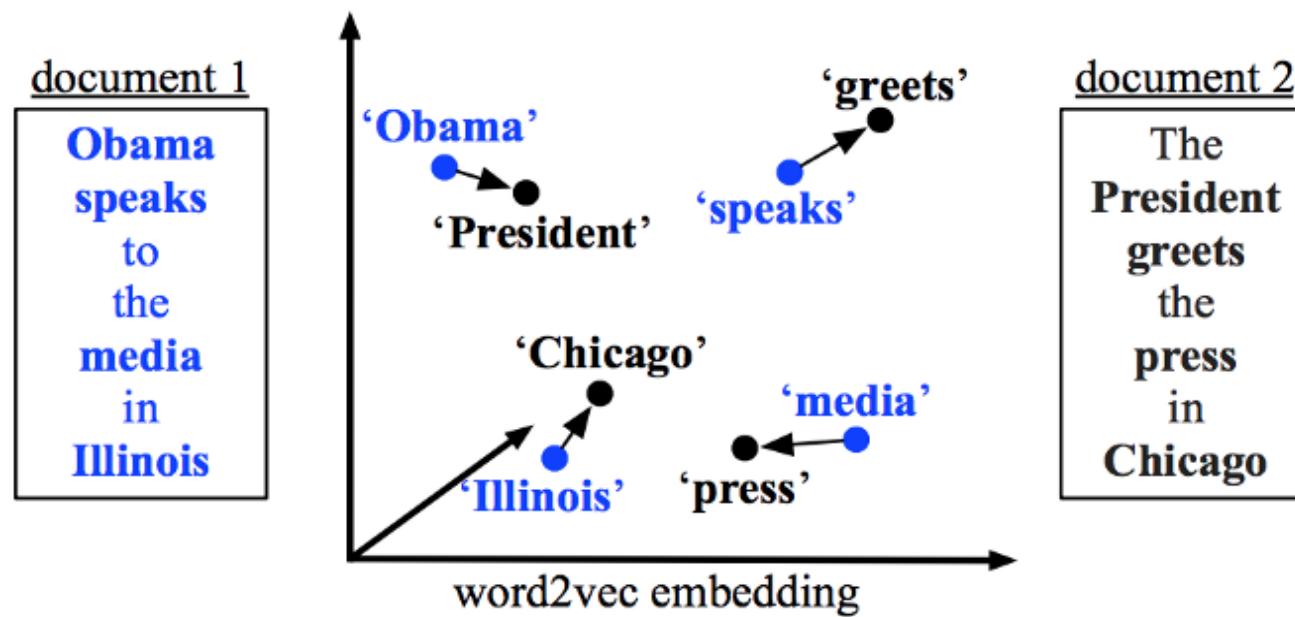


Figure: Word Mover's Distance

Source: <https://medium.com/@adriensieg/text-similarities-da019229c894>

# Variational auto encoders

- Used to identify text based upon the same text as input.
- The auto-encoder uses neural network to an approximate value relative to the input.

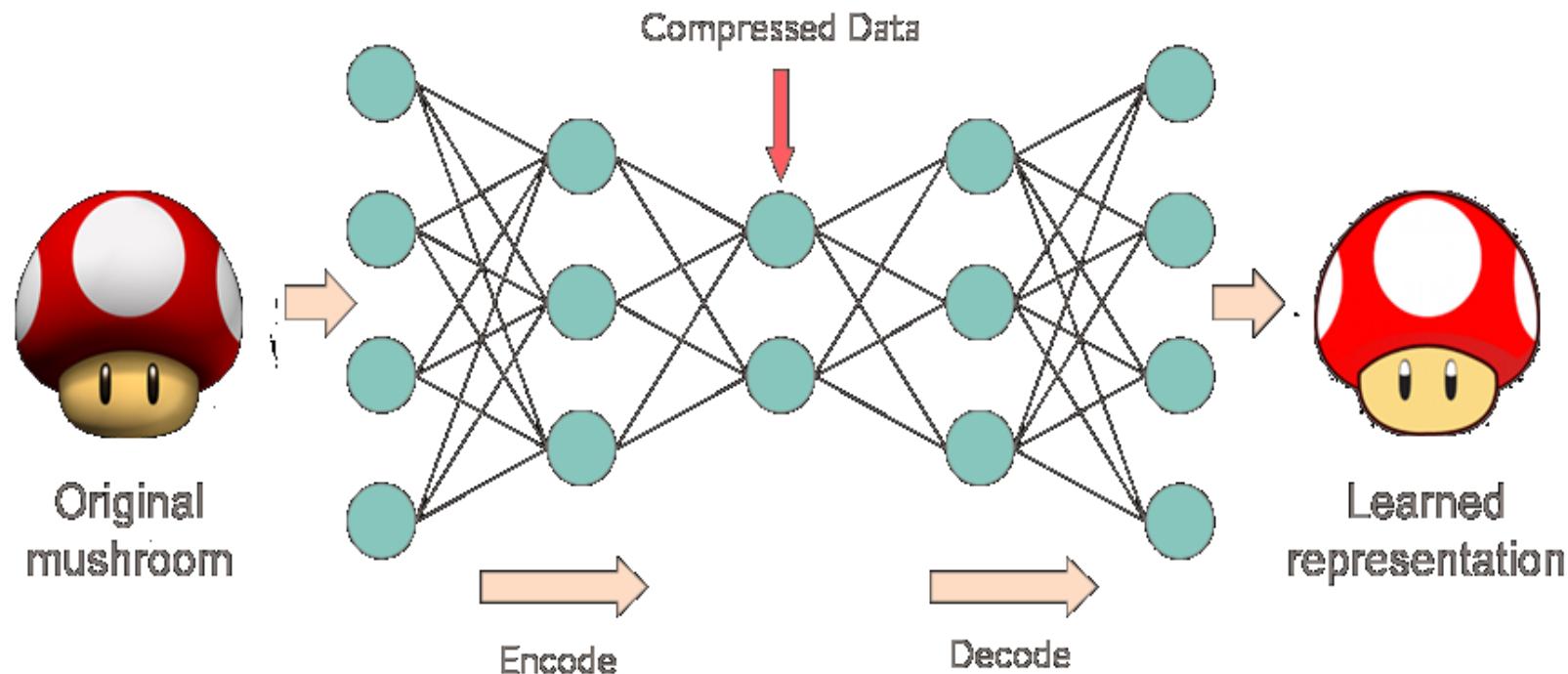


Figure: Variational Auto Encoders

Source: <https://medium.com/@adriensieg/text-similarities-da019229c895>

# Pre-trained sentence encoders

- Used to encode the basic text into higher dimension vectors.
- Pre-trained encoders are trained on both supervised learning and unsupervised learning to identify both the syntactic and semantic information.



Figure: Pre-Trained Sentence Encoders

Source: <https://medium.com/@adriensieg/text-similarities-da019229c896>

# Bidirectional Encoder Representations from Transformers (BERT) with cosine distance



IBM ICE (Innovation Centre for Education)

- The BERT model uses word vectors that can adapt depending upon the surroundings.

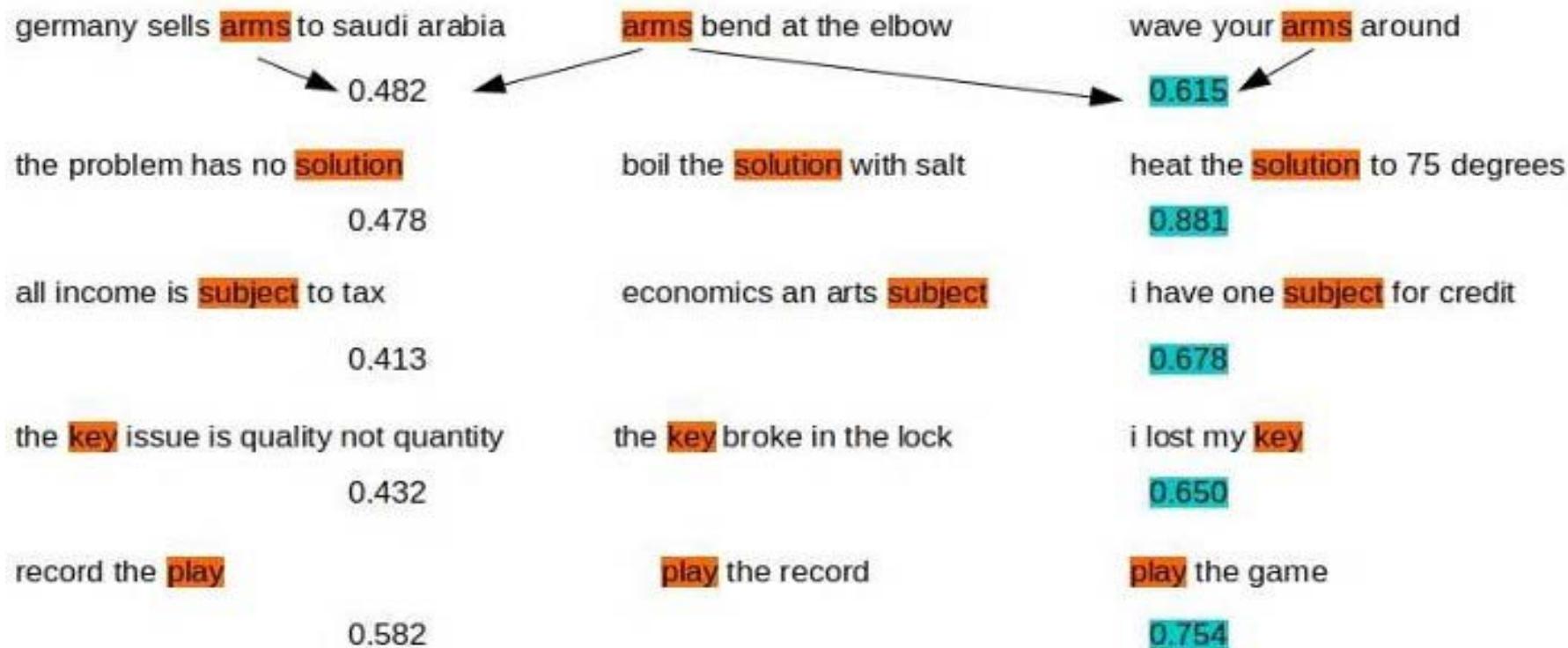


Figure: BERT Similarity

Source: <https://medium.com/@adriensieg/text-similarities-da019229c897>

# Word sense disambiguation

- Word sense disambiguation related to identify the sense of any word use tree in a sentence.
- Identify and determine the meanings of the words in any context.

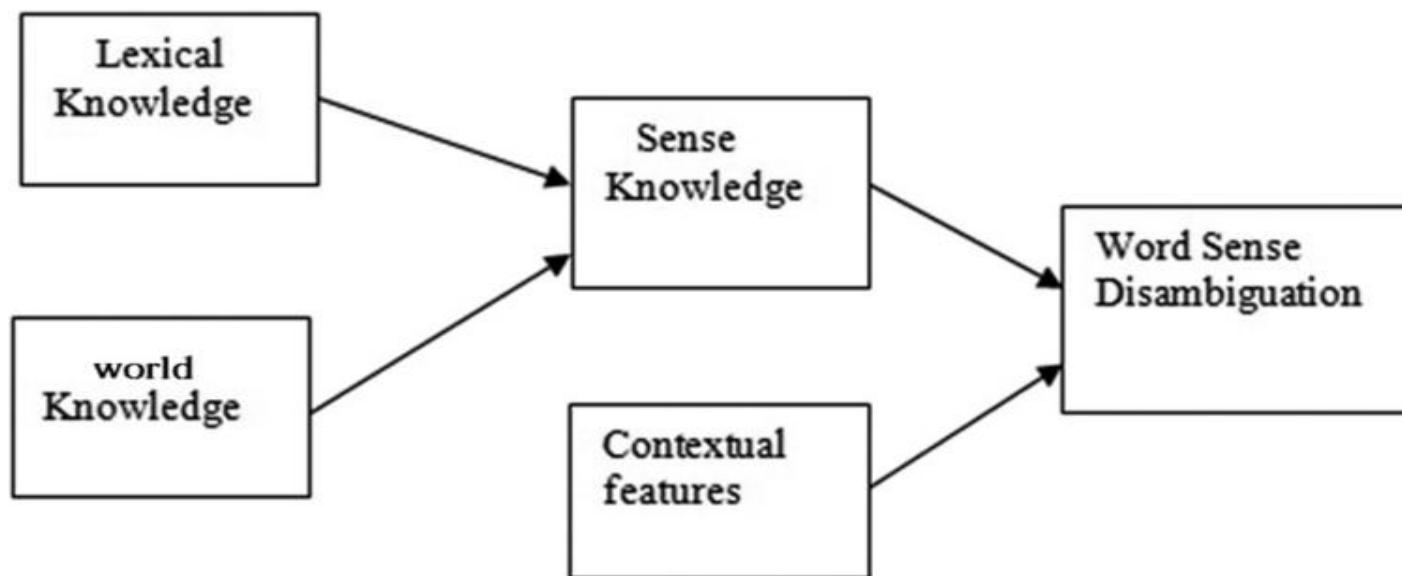


Figure: Word Sense Disambiguation Process

Source: <https://content.iospress.com/articles/international-journal-of-knowledge-based-and-intelligent-engineering-systems/kes190399>

# Complications in WSD

- Dictionary differences: Word identified with appropriate senses.
- POS tagging: Positioning of the words in the sentence to understand the tag and the meaning.
- Inter-judge variance: Human sensing of words and their meaning → Hard to decipher.
  - Sense cannot be same for all.
- Pragmatics: Identifying the meaning of the context based upon ontology.
- **Example:**
  - Sentence 1: Alex and John are fathers - Independent relationship.
  - Sentence 2: Alex and John are brothers - Dependent relationship.

# Methods in WSD

- Deep approach.
- Shallow approach.

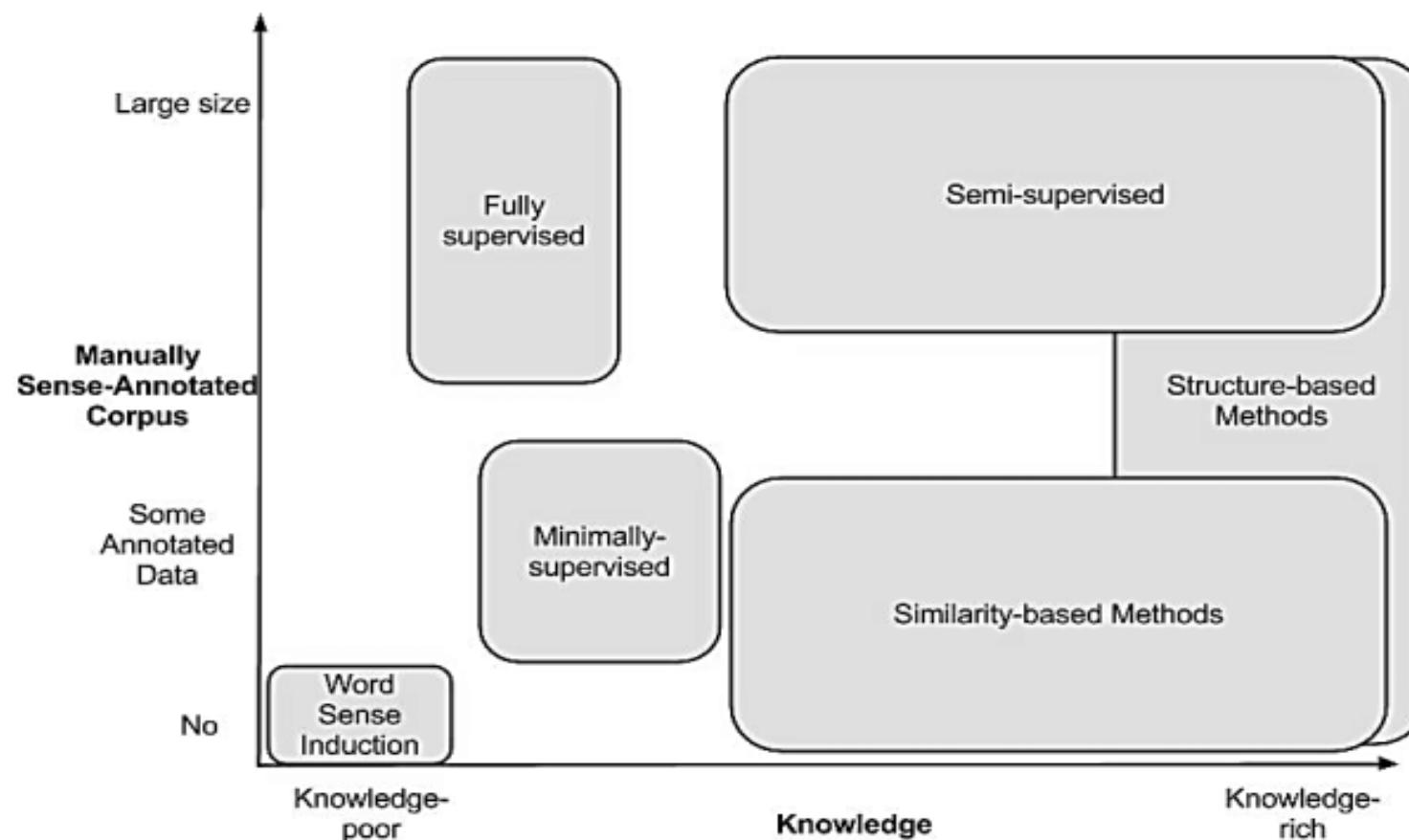


Figure: Methods in WSD

Source: [https://www.researchgate.net/figure/Word-Sense-Disambiguation-systems-Data-versus-Knowledge-Schwab-2013-personal-notes\\_fig2\\_257409694](https://www.researchgate.net/figure/Word-Sense-Disambiguation-systems-Data-versus-Knowledge-Schwab-2013-personal-notes_fig2_257409694)

# Evaluation of WSD

- Very hard to evaluate → Every word can have different sense based upon the context.
- Requires large amount of hand annotated Corpus.

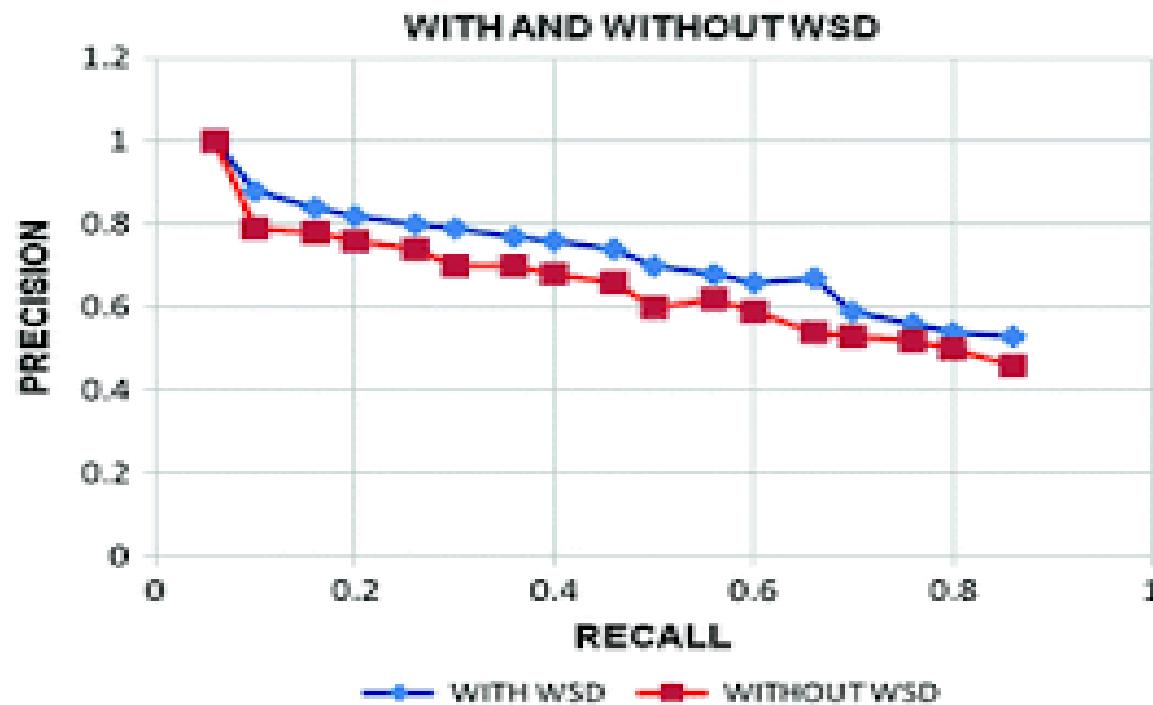


Figure: Evaluation of WSD

Source: [https://link.springer.com/chapter/10.1007/978-981-10-2471-9\\_64](https://link.springer.com/chapter/10.1007/978-981-10-2471-9_64)

# Self evaluation: Exercise 7

- To continue with the training, after learning the concepts of Parsing, Tokenization, Word Sense Disambiguation, Stop Word Removal in Natural Language Text Processing, it is time to write code to work with Tokenization, WSD and use it to compare similarities. It is instructed to utilize the concepts of reading data from files tokenization, word Similarity, WSD and perform the following activity.
- You are instructed to write the following activities using Python code.
- **Exercise 7:** Python code to Read two set of documents, perform Tokenization, map dictionaries, create corpus and calculate the similarity between the documents.

# Self evaluation: Exercise 8

- To continue with the training, after learning the concepts of Parsing, Tokenization, Word Sense Disambiguation, Stop Word Removal in Natural Language Text Processing, it is time to write code to work with Tokenization, WSD and use it to compare similarities. It is instructed to utilize the concepts of reading data from files tokenization, word Similarity, WSD and perform the following activity.
- You are instructed to write the following activities using Python code.
- **Exercise 8:** Python code to Read words and sentences and perform Word Sense Disambiguation using WordNet and LESK.

# Self evaluation: Exercise 9

- To continue with the training, after learning the concepts of Parsing, Tokenization, Word Sense Disambiguation, Stop Word Removal in Natural Language Text Processing, it is time to write code to work with Tokenization, WSD and use it to compare similarities. It is instructed to utilize the concepts of reading data from files tokenization, word Similarity, WSD and perform the following activity.
- You are instructed to write the following activities using Python code.
- **Exercise 9:** Python code to read text from files, perform pre-processing activities like word sense disambiguation, tokenization, stop word removal. Create a question answer context where the program can read queries from the user and respond as per the words and sentences in the question.

# History of speech recognition technology

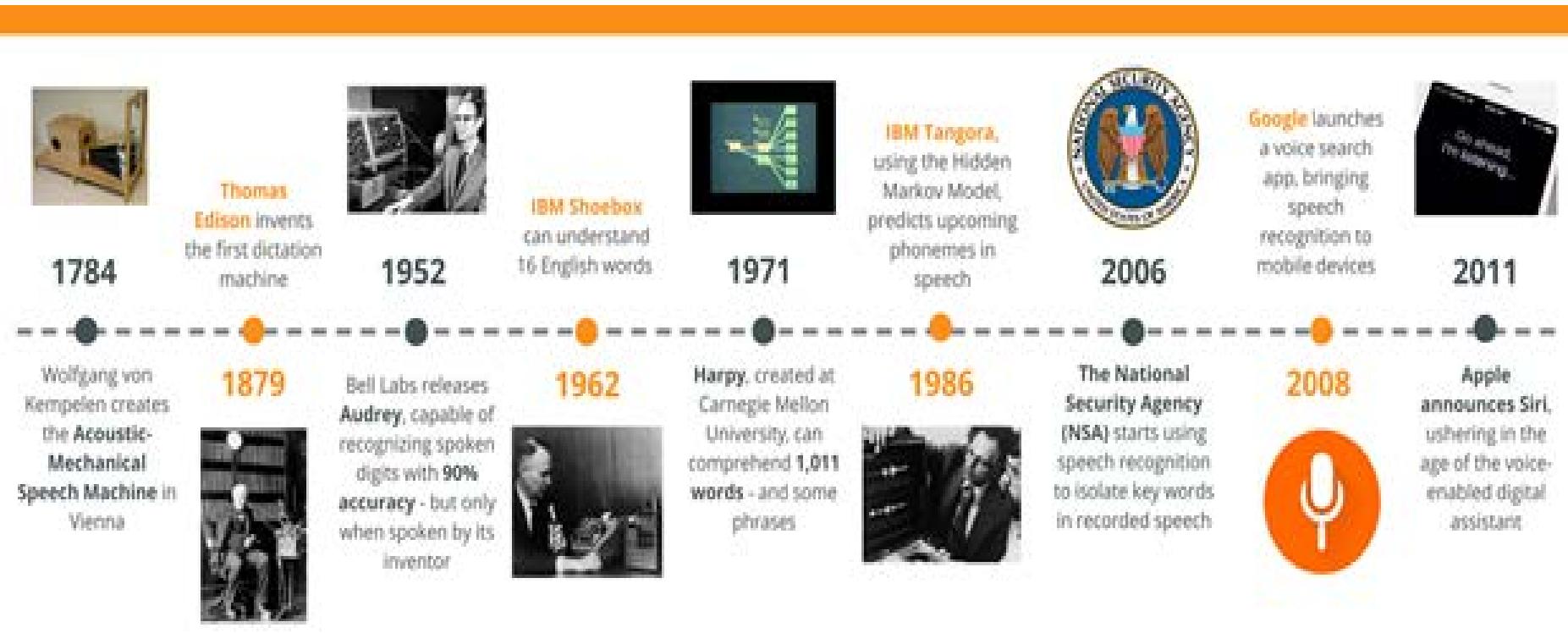


Figure: History of Speech Recognition Technology

Source: <https://medium.com/swlh/the-past-present-and-future-of-speech-recognition-technology-cf13c179aaaf>

# Working principle in voice recognition

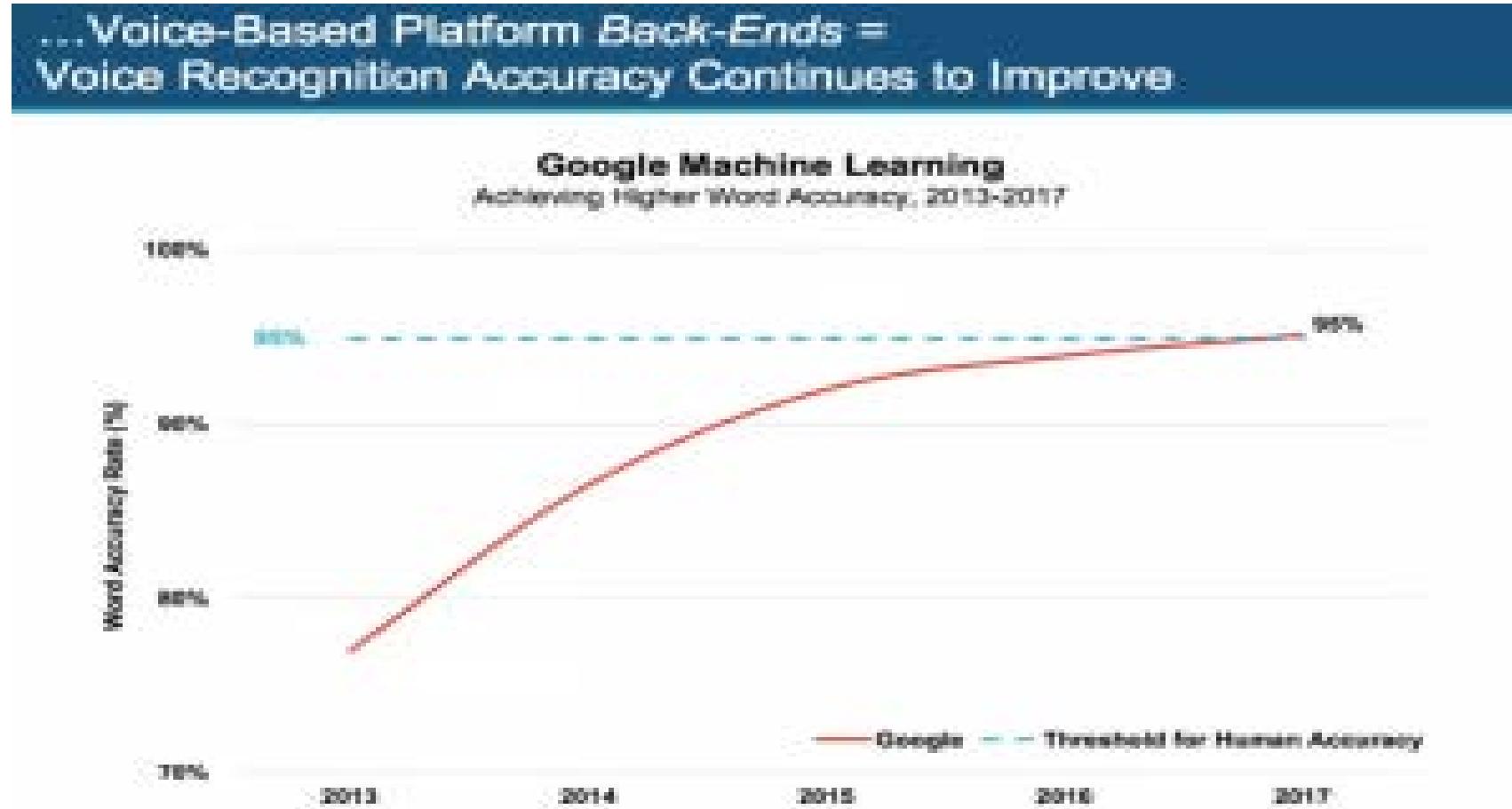


Figure: Voice Recognition Accuracy

Source: <https://www.globalme.net/blog/the-present-future-of-speech-recognition/>

# Major leaders in speech recognition and voice assistant



IBM ICE (Innovation Centre for Education)

## Apple's Siri

- Hardware: Apple HomePod (Due to launch in 2018 at \$349), iPhone, MacBooks, AirPods.
- Digital assistant: Siri
- Usage statistics:
  - 42.5% of smartphones have Apple's Siri digital assistant installed (Highervisibility).
  - 41.4 million monthly active users in the U.S. as of July 2017, down 15% on the previous year (Verto Analytics).
  - 19% of iPhone users engage with Siri at least daily (HubSpot).



Figure: Apple's Siri

Source: <https://osxdaily.com/2016/05/03/improve-hey-siri-voice-training-ios/>

# Amazon Alexa

## Amazon Alexa

- Hardware: Echo, Echo Dot, Echo Show, Fire TV Stick, Kindle..
- Digital Assistant: Alexa
- Usage Statistics:
  - “Tens of millions of Alexa-enabled devices” sold worldwide over the 2017 holiday season (Amazon).
  - 75% of all smart speakers sold to date are Amazon devices (Tech Republic).
  - There are now over 25,000 skills available for Alexa (Amazon).



Figure: Amazon Alexa

Source: <https://www.imore.com/how-improve-amazon-alexa-voice-recognition>

# Microsoft Cortana

## Microsoft Cortana

- Hardware: Harman/kardon Invoke speaker, Windows smartphones, Microsoft laptops
- Digital Assistant: Cortana
- Usage Statistics:
  - 5.1% of smartphones have the Cortana assistant installed
  - Cortana now has 133 million monthly users (Tech Radar)
  - 25% of Bing searches are by voice (Microsoft).



Figure: Cortana

Source: <https://www.pcworld.com/article/2984791/how-to-enable-windows-10s-hey-cortana-voice-commands.html>

# Google Assistant

## Google Assistant

- Hardware: Google Home, Google Home Mini, Google Home Max, Pixelbook, Pixel smartphones, Pixel Buds, Chromecast, Nest smart home products.
- Digital Assistant: Google Assistant.
- Usage Statistics:
  - Google Home has a 24% share of the US smart speaker market (eMarketer).
  - There are now over 1,000 Actions for Google Home (Google).
  - Google Assistant is available on over 225 home control brands and more than 1,500 devices (Google).



Google Assistant



Figure: Google assistant

Source: <https://www.indiatvnews.com/technology/news-google-assistant-devices-voice-match-default-speaker-know-what-is-it-625568>

# Machine translation

- Translate a text in one natural language to another natural language.

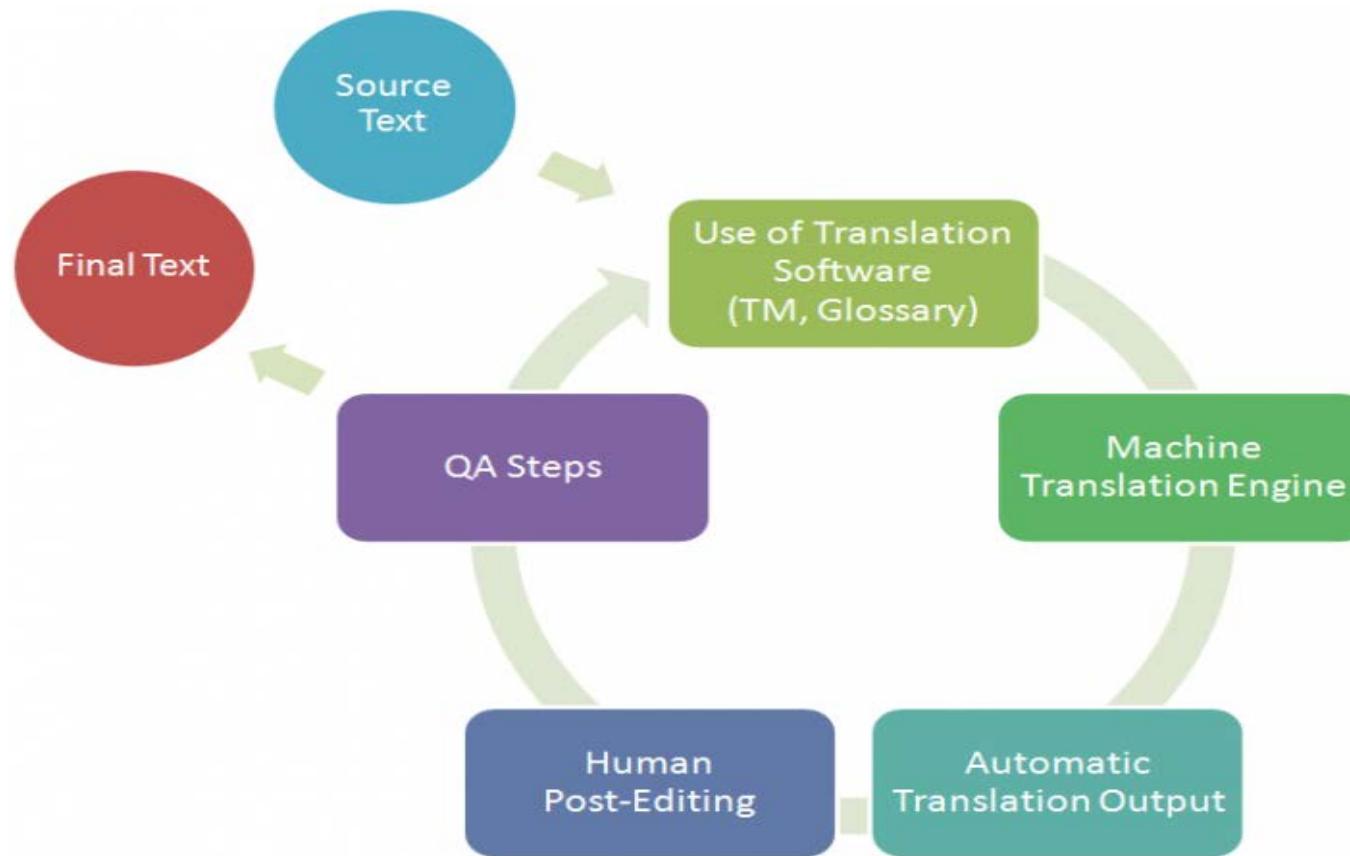


Figure: Machine Translation

Source: <https://langsolin.com/machine-translation-and-confidentiality/>

# Rule-based machine translation

- Parse through the text → Create a representation of parse tree.
- Parse tree → Text for the target language.
- Map linguistic universals (i.e., grammar) between languages.

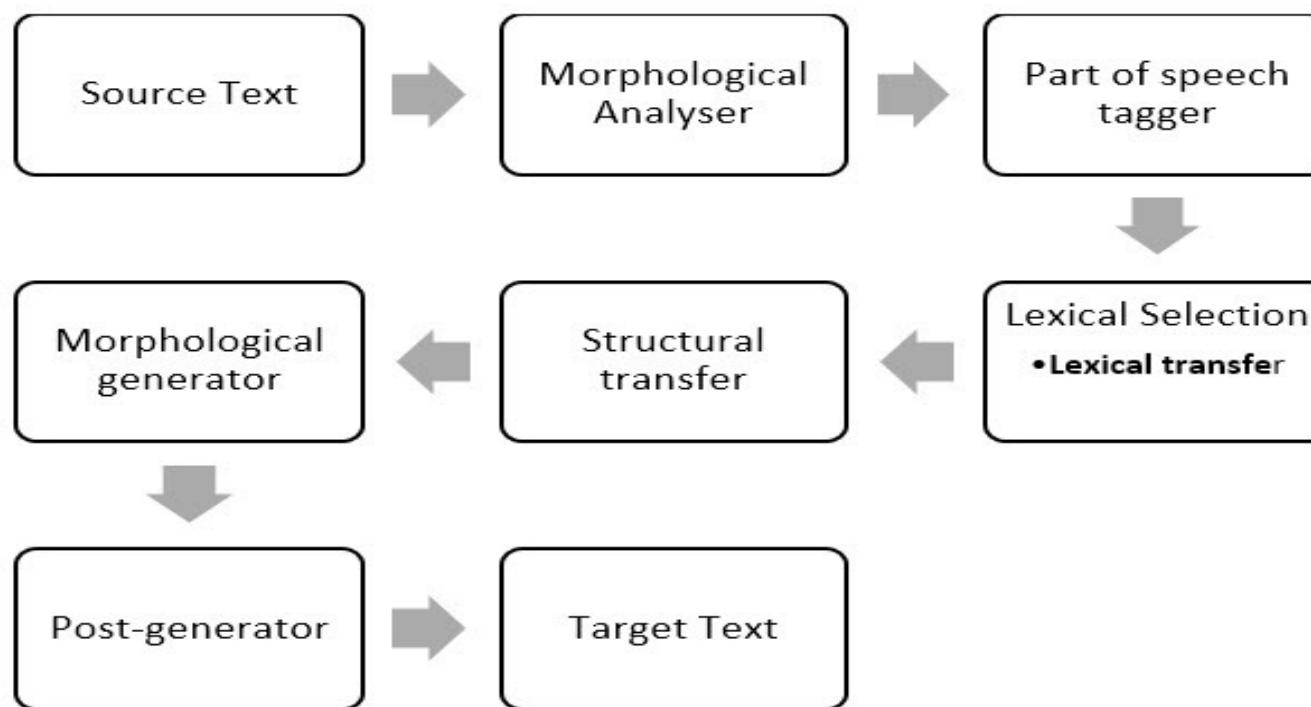


Figure: Rule-Based Machine Translation

Source: [https://www.researchgate.net/figure/Rule-based-Machine-Translation\\_fig1\\_320730405](https://www.researchgate.net/figure/Rule-based-Machine-Translation_fig1_320730405)

# Statistical machine translation

- Language has an inherent logic that could be treated in the same way as any logical mathematical challenge.

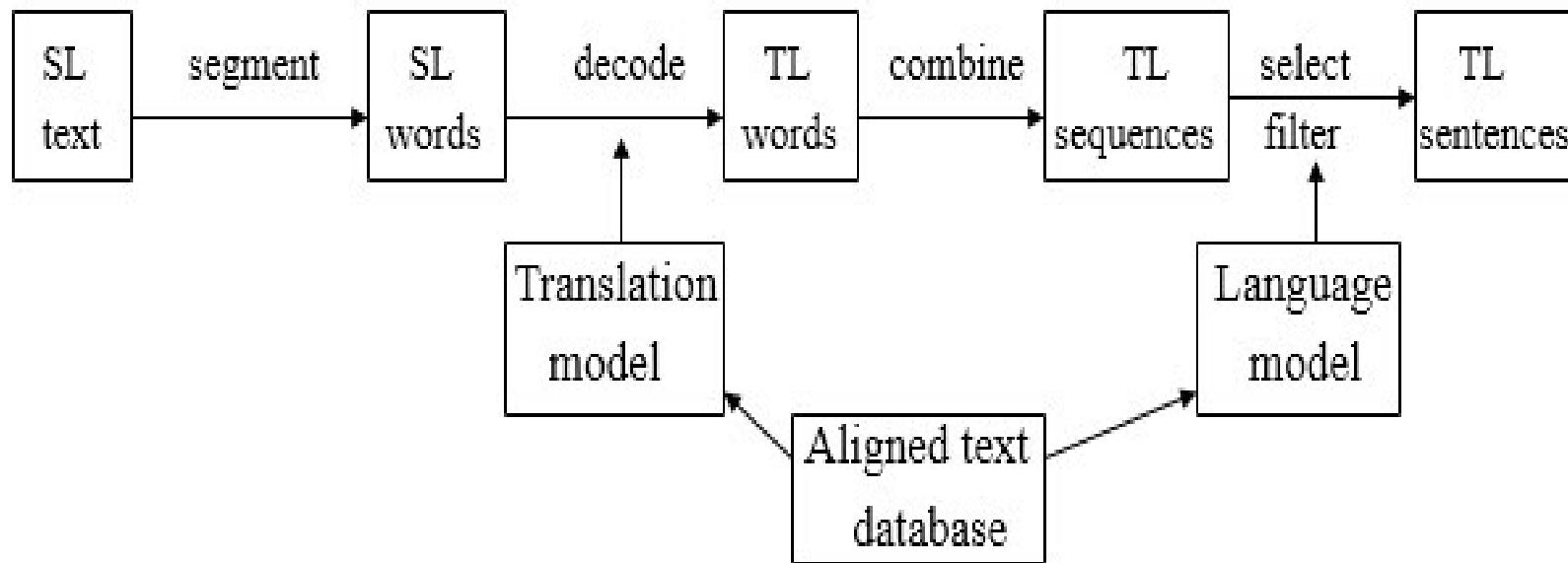


Figure: Statistical Machine Translation

Source: [https://www.researchgate.net/figure/Statistical-Machine-Translation\\_fig2\\_320730405](https://www.researchgate.net/figure/Statistical-Machine-Translation_fig2_320730405)

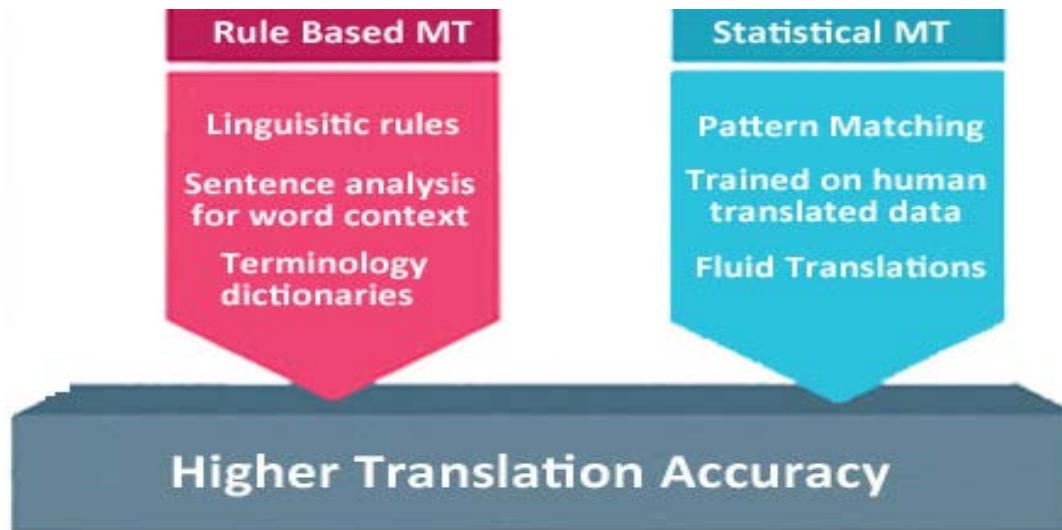


Figure: Rule-Based MT vs. Statistical MT

Source: <https://www.translationsoftware4u.com/enterprise-global.php>

Rule-Based MT	Statistical MT
Consistent and predictable quality	Unpredictable translation quality
Out-of-domain translation quality	Poor out-of-domain quality
Knows grammatical rules	Does not know grammar
High performance and robustness	High CPU and disk space requirements
Consistency between versions	Inconsistency between versions
Lack of fluency	Good fluency
Hard to handle exceptions to rules	Good for catching exceptions to rules
High development and customization costs	Rapid and cost-effective development costs provided the required corpus exists

# Working principle of SMT (1 of 2)

- Very large data set of approved translations.
- Translation Model → Frequency of phrases.
- More frequently a phrase is repeated → More probable the target translation is correct.
- Probability model → Target translation.

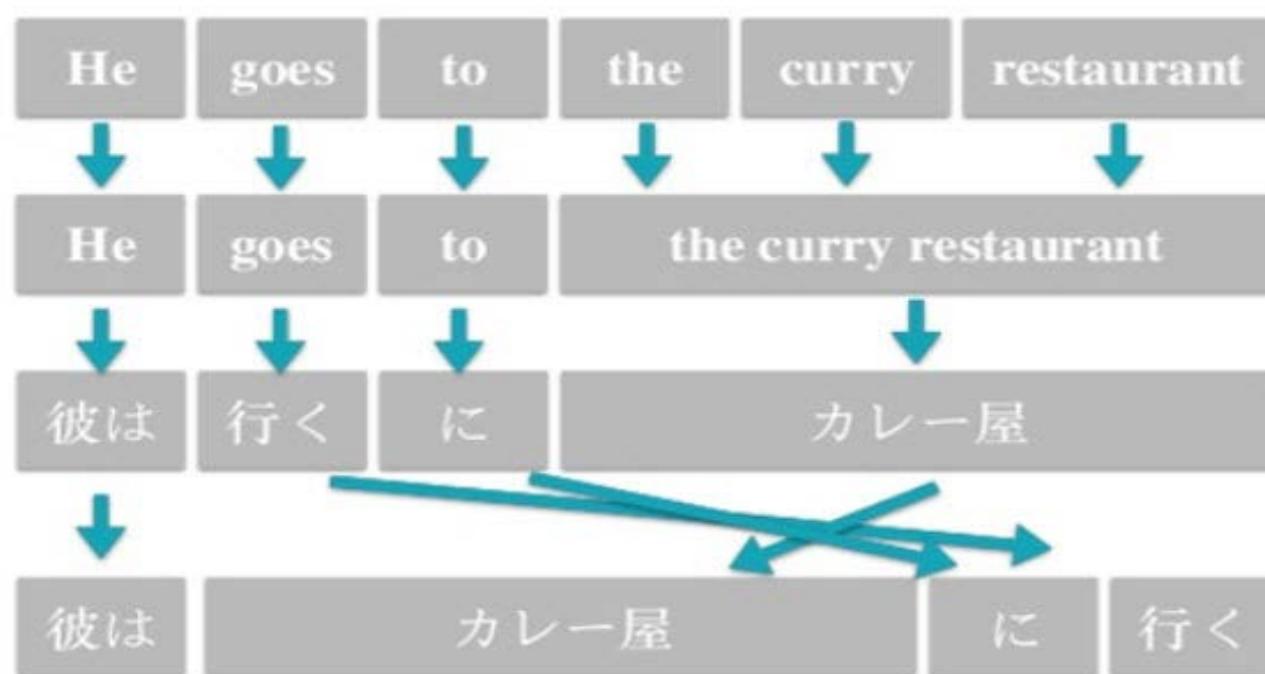


Figure: Working principles of SMT

Source: <https://kantanmtblog.com/2019/04/02/a-short-introduction-to-the-statistical-machine-translation-model/>

# Working principle of SMT (2 of 2)

- Statistical machine translation.
- Decoding process.
- Speech to speech machine translation.

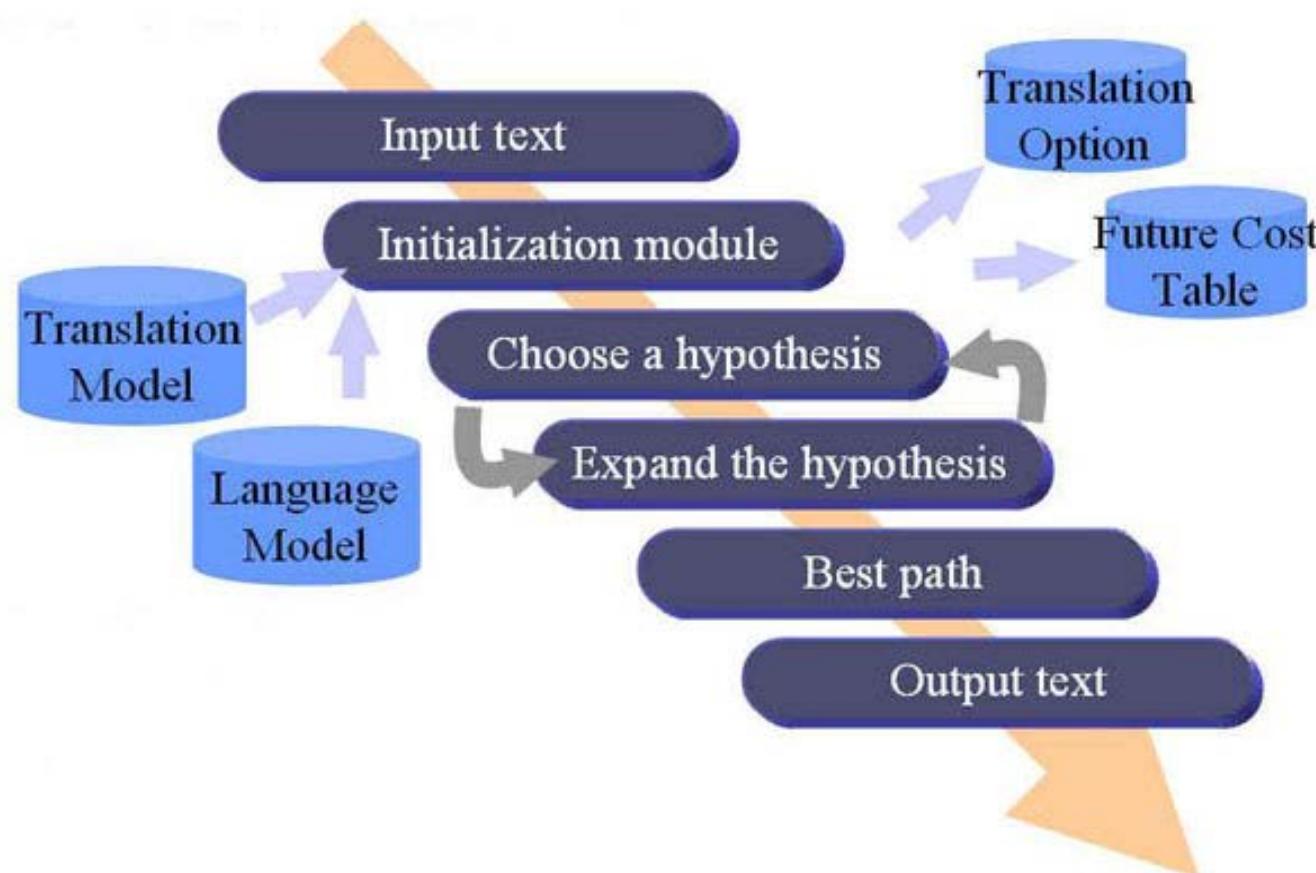


Figure: Working principles of SMT

Source: [http://nlp.postech.ac.kr/research/previous\\_research/smt/](http://nlp.postech.ac.kr/research/previous_research/smt/)

# Challenges with statistical machine translation



IBM ICE (Innovation Centre for Education)

- Sentence alignment
  - Single sentences → Translated into several sentences.
- Word alignment
  - Words have no clear equivalent in the target language.
  - "John does not live here," → "John wohnt hier nicht."
- Statistical anomalies
  - Override translations → Proper nouns.
  - "I took the train to Berlin" = "I took the train to Paris".
- Idioms
  - Idioms may not translate "idiomatically".
  - "hear" → "Bravo!" Parliament "Hear, Hear!" becomes "Bravo!".
- Different word orders
  - Word order in languages differ.
  - SVO or VSO languages.
- Out Of Vocabulary (OOV) words
  - Different word forms as separate symbols without any relation.

# Checkpoint (1 of 2)

## Multiple choice questions:

1. What is the main challenge/s of NLP?
  - a) Handling ambiguity of sentences
  - b) Handling tokenization
  - c) Handling POS-tagging
  - d) All the above
  
2. What is machine translation?
  - a) Converts one human language to another
  - b) Converts human language to machine language
  - c) Converts any human language to English
  - d) Converts Machine language to human language
  
3. What is morphological segmentation?
  - a) Does discourse analysis
  - b) Separate words into individual morphemes and identify the class of the morphemes
  - c) Is an extension of propositional logic
  - d) None of the above

# Checkpoint solutions (1 of 2)

## Multiple choice questions:

1. What is the main challenge/s of NLP?
  - a) **Handling ambiguity of sentences**
  - b) Handling tokenization
  - c) Handling POS-tagging
  - d) All the above
  
2. What is machine translation?
  - a) **Converts one human language to another**
  - b) Converts human language to machine language
  - c) Converts any human language to English
  - d) Converts Machine language to human language
  
3. What is morphological segmentation?
  - a) Does discourse analysis
  - b) **Separate words into individual morphemes and identify the class of the morphemes**
  - c) Is an extension of propositional logic
  - d) None of the above

# Checkpoint (2 of 2)

## Fill in the blanks:

1. Many words have more than one meaning; we must select the meaning which makes the most sense in context. This can be resolved by \_\_\_\_\_.
2. In a rule-based system, the form of procedural domain knowledge \_\_\_\_\_.
3. Types are available in machine learning are \_\_\_\_\_.
4. \_\_\_\_\_ are used to identify text based upon the same text as input.

## True or False:

1. Speech segmentation is a subtask of speech recognition. True/False
2. Modern NLP algorithms are based on machine learning, especially statistical machine learning. True/False
3. Statistical machine translation uses algorithms for learning how to analyze the human translations. True/False

# Checkpoint solutions (2 of 2)

## Fill in the blanks:

1. Many words have more than one meaning; we must select the meaning which makes the most sense in context. This can be resolved by Word sense disambiguation.
2. In a rule-based system, the form of procedural domain knowledge production rules.
3. Types are available in machine learning are 3.
4. Variational auto encoders are used to identify text based upon the same text as input.

## True or False:

1. Speech segmentation is a subtask of speech recognition. **True**
2. Modern NLP algorithms are based on machine learning, especially statistical machine learning. **True**
3. Statistical machine translation uses algorithms for learning how to analyze the human translations. **True**

# Question bank

## Two mark questions:

1. What are multi word expressions?
2. What is cosine similarity?
3. What is WSD? Why is it needed?
4. What are parse trees?

## Four mark questions:

1. Explain left corner parsing with examples.
2. How are multi word expressions classified?
3. Explain the methods of identifying text similarity.
4. What are the complications in word sense disambiguation?

## Eight mark questions:

1. Write in detail about the concepts and process involved in statistical machine translation.
2. Discuss in detail about the various approaches to parsing.

# Unit summary

**Having completed this unit, you should be able to:**

- Understand what is statistical parsing and the core concepts involved in it
- Learn about multiword expressions and how to handle them
- Understand the concepts of word similarity and the relativity calculations done
- Gain knowledge on word sense disambiguation and why it is needed in NLP
- Gain an insight into modern speech recognition techniques with an idea of the forerunners in the field
- Understand what statistical machine translation means and the guidelines needed to perform SMT

Welcome to:

# Applications of Natural Language Processing



# Unit objectives

**After completing this unit, you should be able to:**

- Understand what is information retrieval and the concepts
- Learn about work with the steps in IR and perform IR
- Gain knowledge on information answering, the various types of QA, how to model a QA
- Understand the concepts of information extraction, basic ideas and operations in IE
- Learn about what is ontology construction, the types, categories and steps involved in OC

# Information Retrieval

- Storage and access to the information.
- “Information Retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers)”.
- Information retrieval system ➔ Software to store and retrieve information.

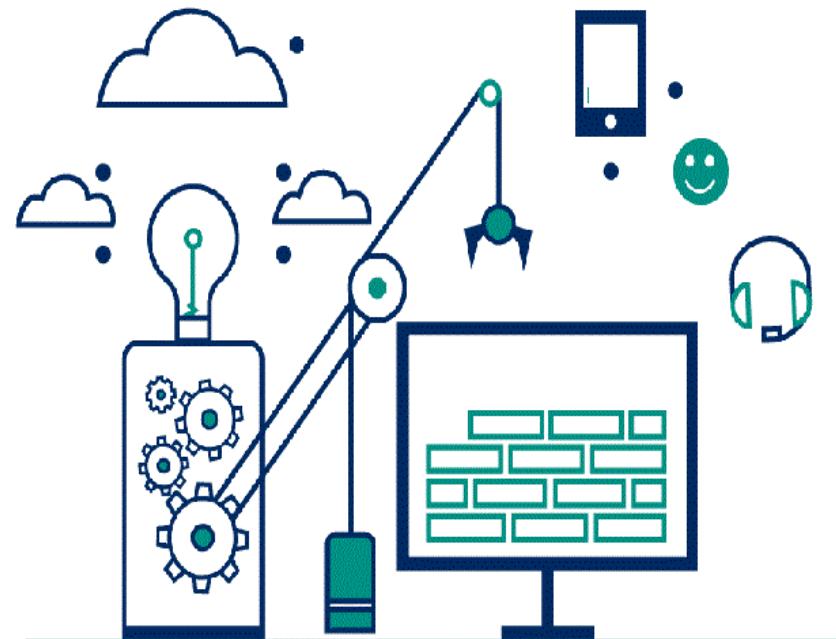


Figure: Information Retrieval

Source: <https://itexperttraining.com/core/courses/information-retrieval/>

# Information Retrieval in NLP

- Natural Language Processing → Understanding, processing, retrieval and generation of automatic responses to natural language queries.
  - Large amount of corpus → Automation of natural language processing.
  - Components → Searching, translation and response generation.
- 
- User inputs a Query.
  - Process → Natural language.
  - Identify the relevant information.
  - Process the query.

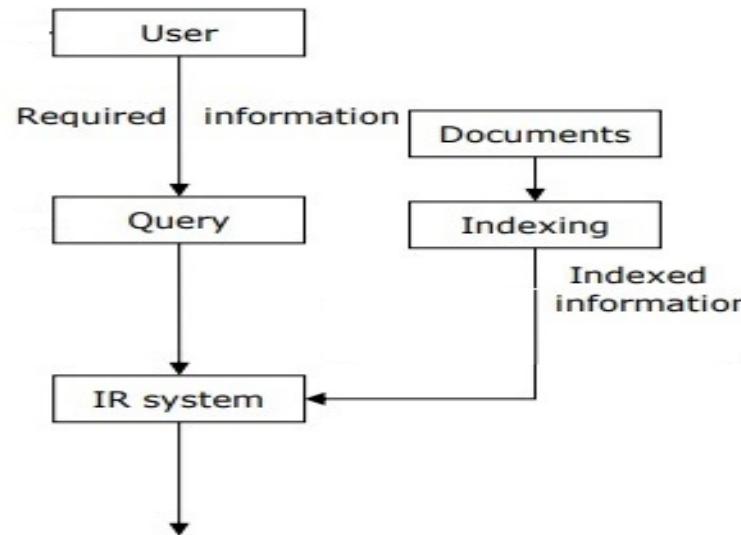


Figure: IR system

Source: SELF

# IR development (1 of 2)

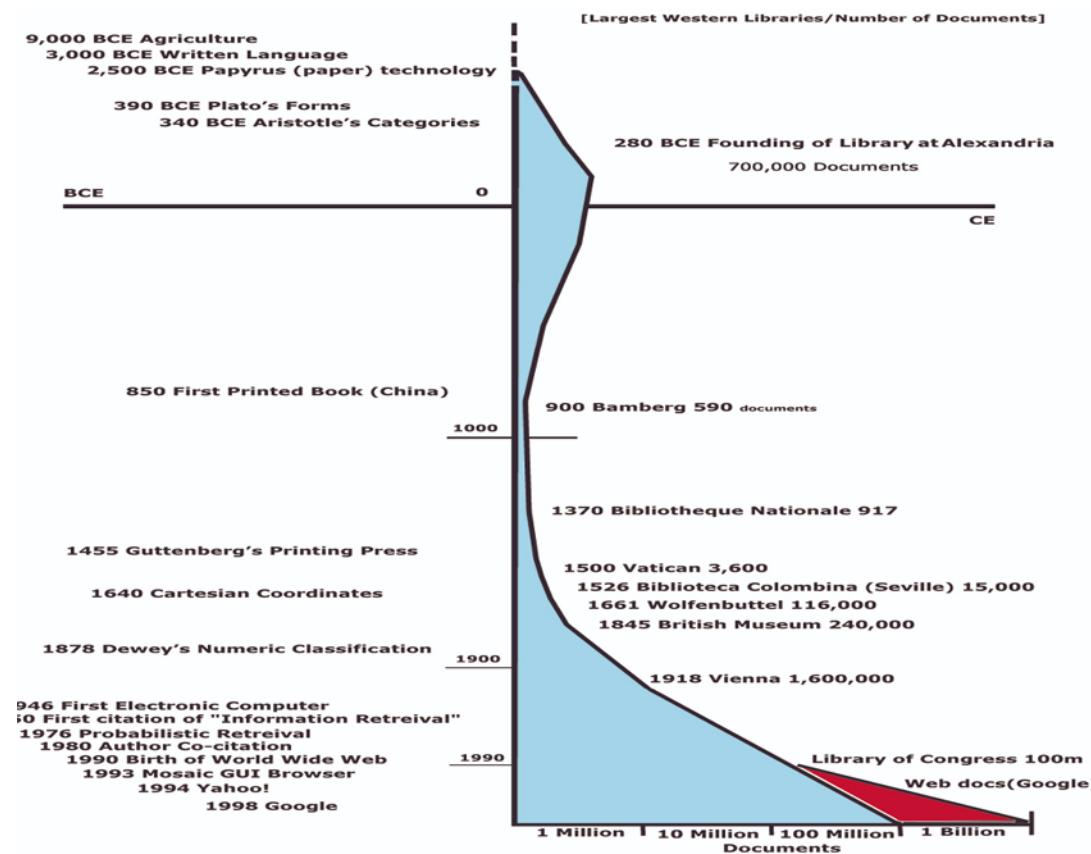


Figure: Availability of Data over timeline

Source: [https://www.researchgate.net/figure/Rough-timeline-of-the-generations-of-information-retrieval-in-digital-libraries-The\\_fig1\\_14214241](https://www.researchgate.net/figure/Rough-timeline-of-the-generations-of-information-retrieval-in-digital-libraries-The_fig1_14214241)

# IR development (2 of 2)

- 1920 to 1930 → First document storage and retrieval system.
- 1950 → Searching for information through selective process.
- 1970 → Large-scale retrieval system.
- 1990 → Internet and World Wide Web → Information retrieval process.
- Search engines → Retrieval systems → Data within seconds.
- 21st century → Large amount of data.

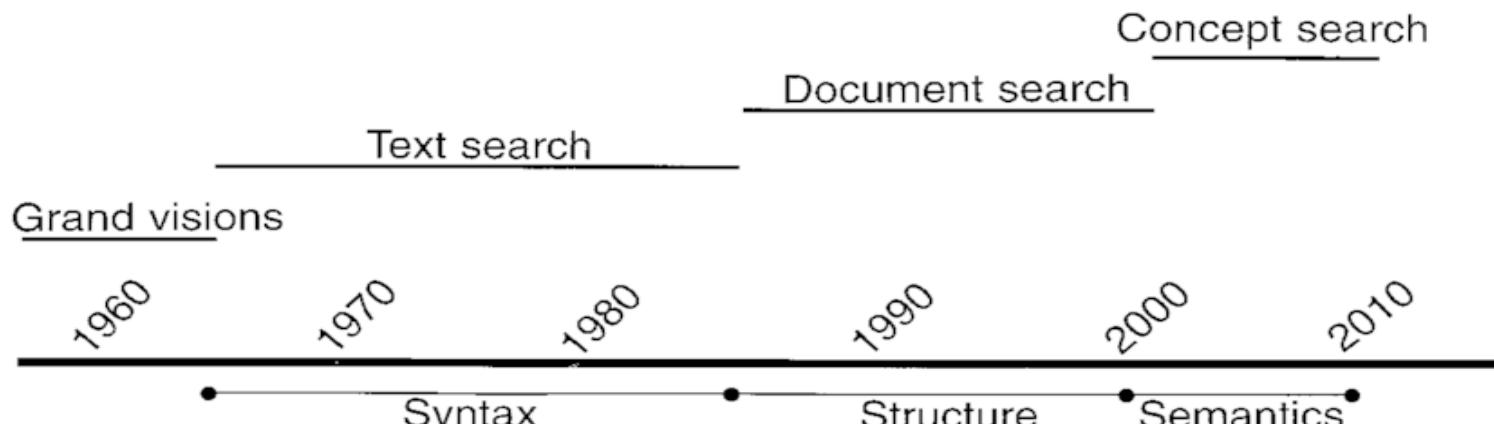


Figure: IR Timeline

Source: <http://online.sfsu.edu/fielden/hist.htm>

# Model types

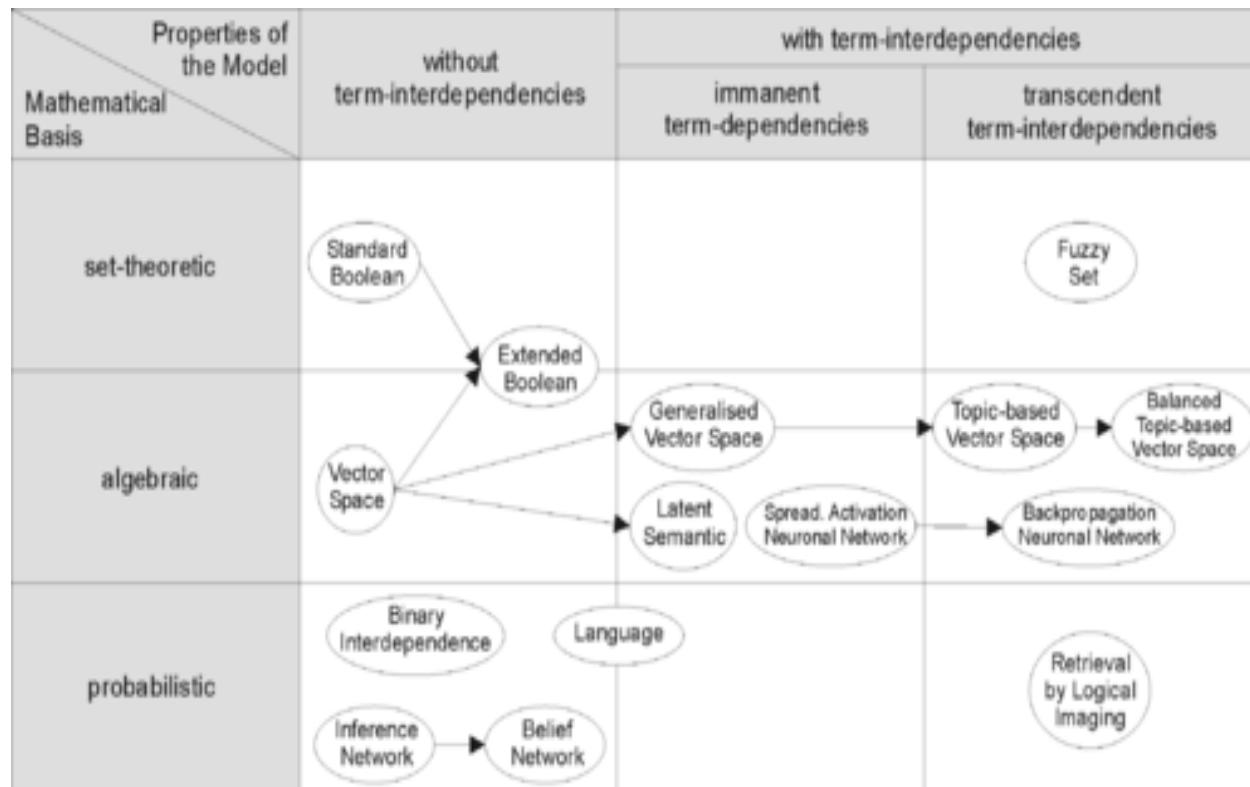


Figure: Models

Source: [https://en.wikipedia.org/wiki/Information\\_retrieval#Model\\_types](https://en.wikipedia.org/wiki/Information_retrieval#Model_types)

# Model types: Mathematical basis model

- Set-theoretic models:
  - Words → Sets.
- Standard Boolean.
- Extended Boolean.
- Fuzzy retrieval.
- Algebraic models:
  - Documents → Matrix, Vector or Tuples.
  - Similarity values → Match document to query.
- Vector space.
- Generalized vector space.
- Topic-based vector space.
- Extended Boolean.
- Latent semantic indexing.

# Problems with NLP in information retrieval



IBM ICE (Innovation Centre for Education)

## ➤ Linguistic variation

- Different words → Same concept.
- Linguistic variation → Document silence.
- Document silence → Omission of the relevant documents.
- Different words → Convey the idea.

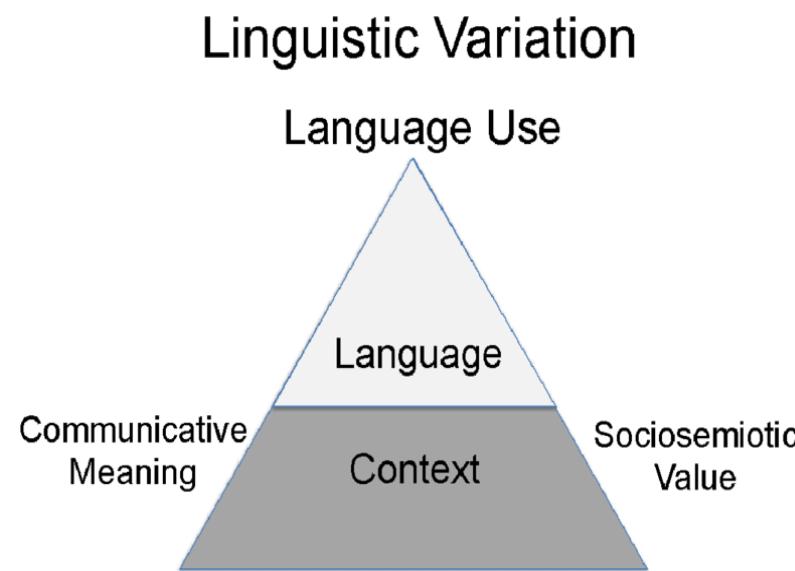


Figure: Linguistic variation

Source: [https://www.researchgate.net/figure/Discourse-and-context-in-linguistic-variation\\_fig1\\_235951304](https://www.researchgate.net/figure/Discourse-and-context-in-linguistic-variation_fig1_235951304)

# NLP in information retrieval

- Indexing.
- Query analysis.
- Comparison.
- Result processing.

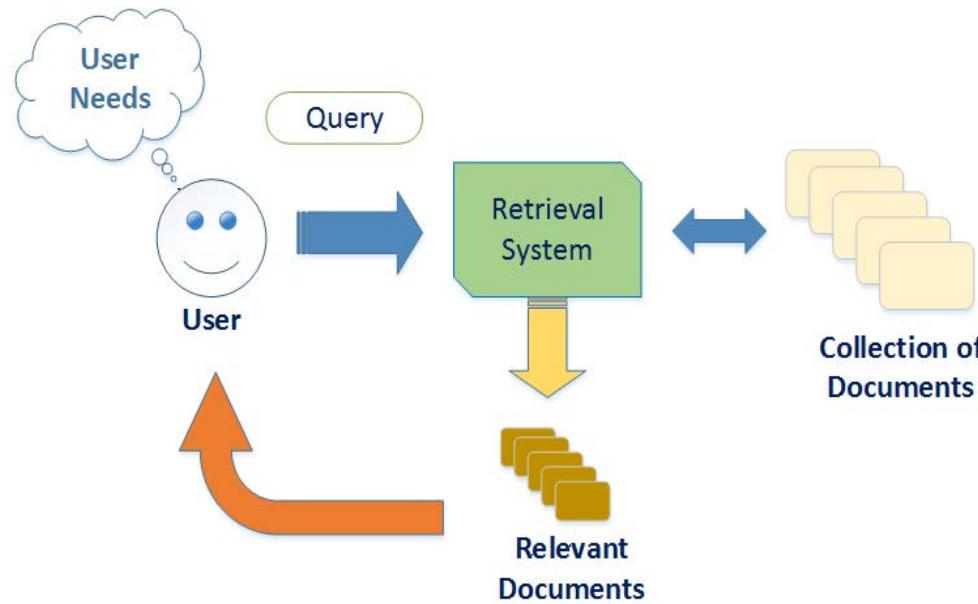


Figure: Information Retrieval system outline

Source: <https://ir.cs.ui.ac.id/new/>

# IR evaluation metrics

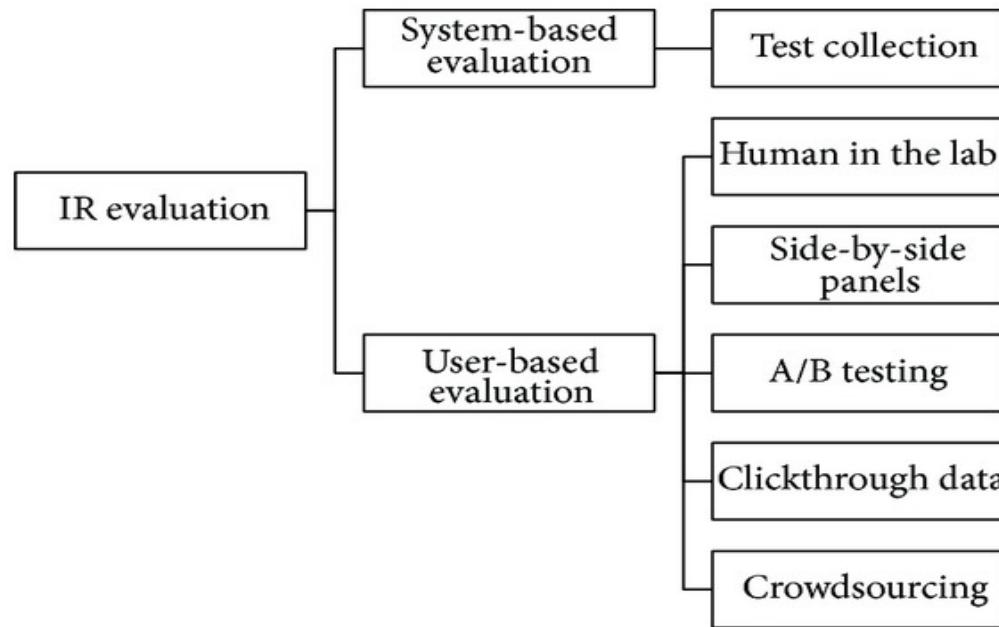


Figure: IR Evaluation Methods

Source: [https://www.researchgate.net/figure/Classification-of-IR-evaluation-methods\\_fig1\\_263517579](https://www.researchgate.net/figure/Classification-of-IR-evaluation-methods_fig1_263517579)

# Information Retrieval (IR) model and types

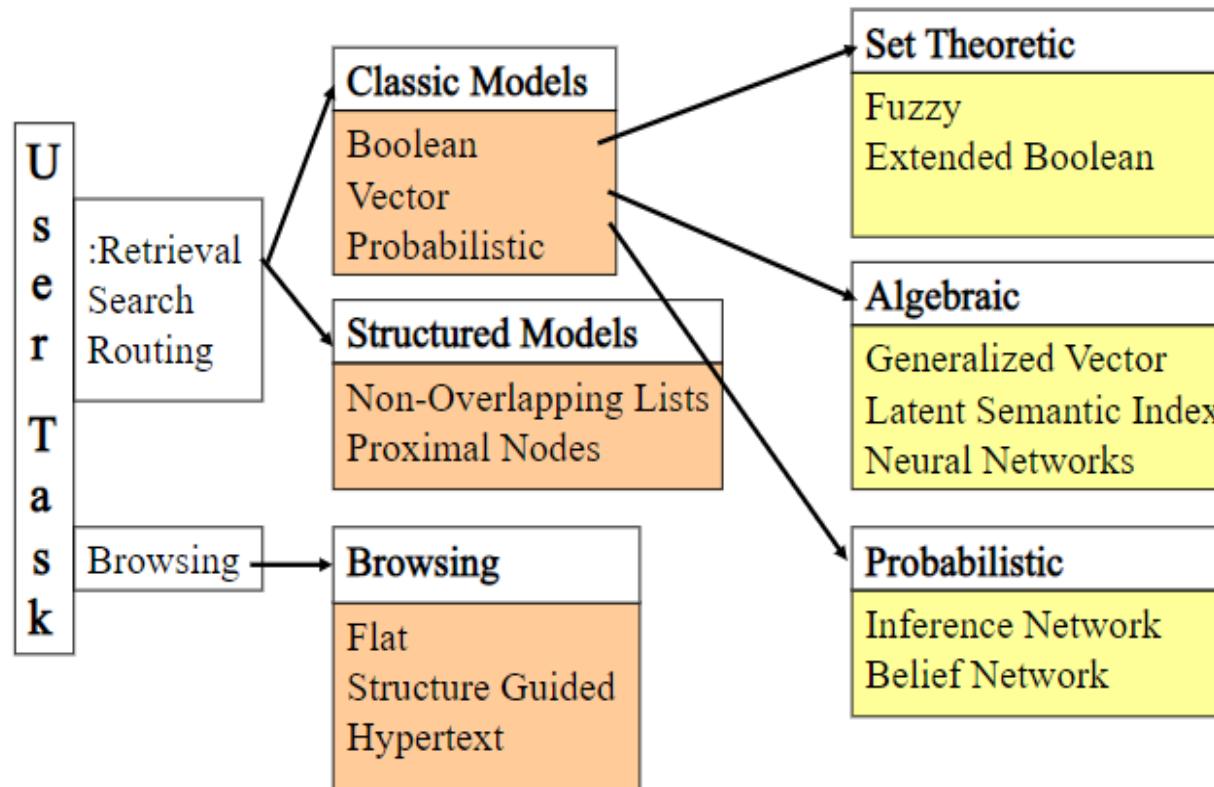


Figure: Information Retrieval Model

Source: <https://slideplayer.com/slide/4905733/>

# Design features of IR systems

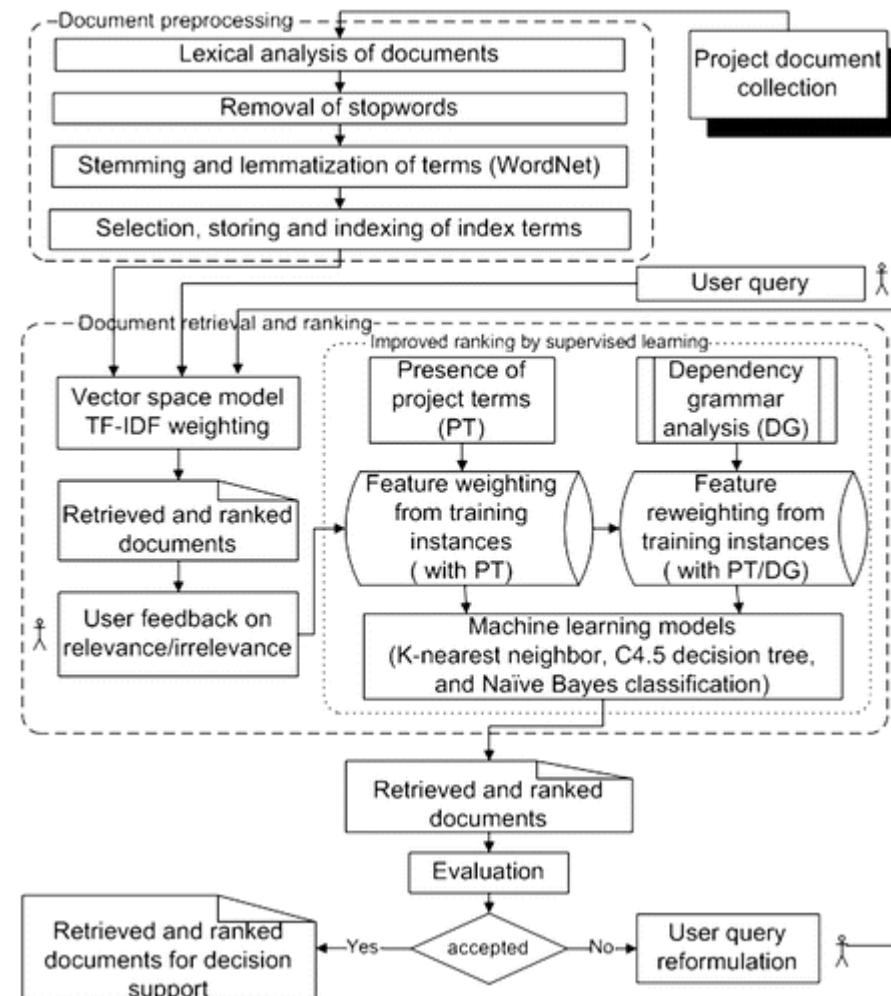


Figure: Design features of IR systems

Source: <https://ascelibrary.org/doi/10.1061/%28ASCE%29ME.1943-5479.0000341>

# Design features of IR systems

- NLP in IR → Large computational cost.
- NLP and Information retrieval → Same algorithm for better performance.
- Success rate → Length of the queries.

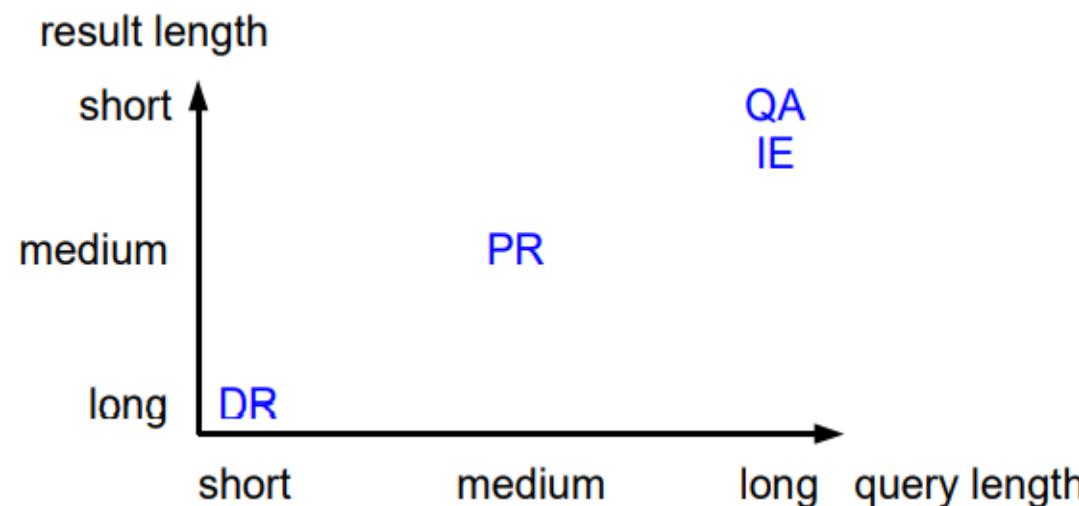


Figure: Classification of Document Retrieval, Passage Retrieval, Question Answering and Information Retrieval according to query length and result length

Source: <https://pdfs.semanticscholar.org/8721/f2a087ff35318a056a5814ba287a37df0ec8.pdf>

# Question answering systems

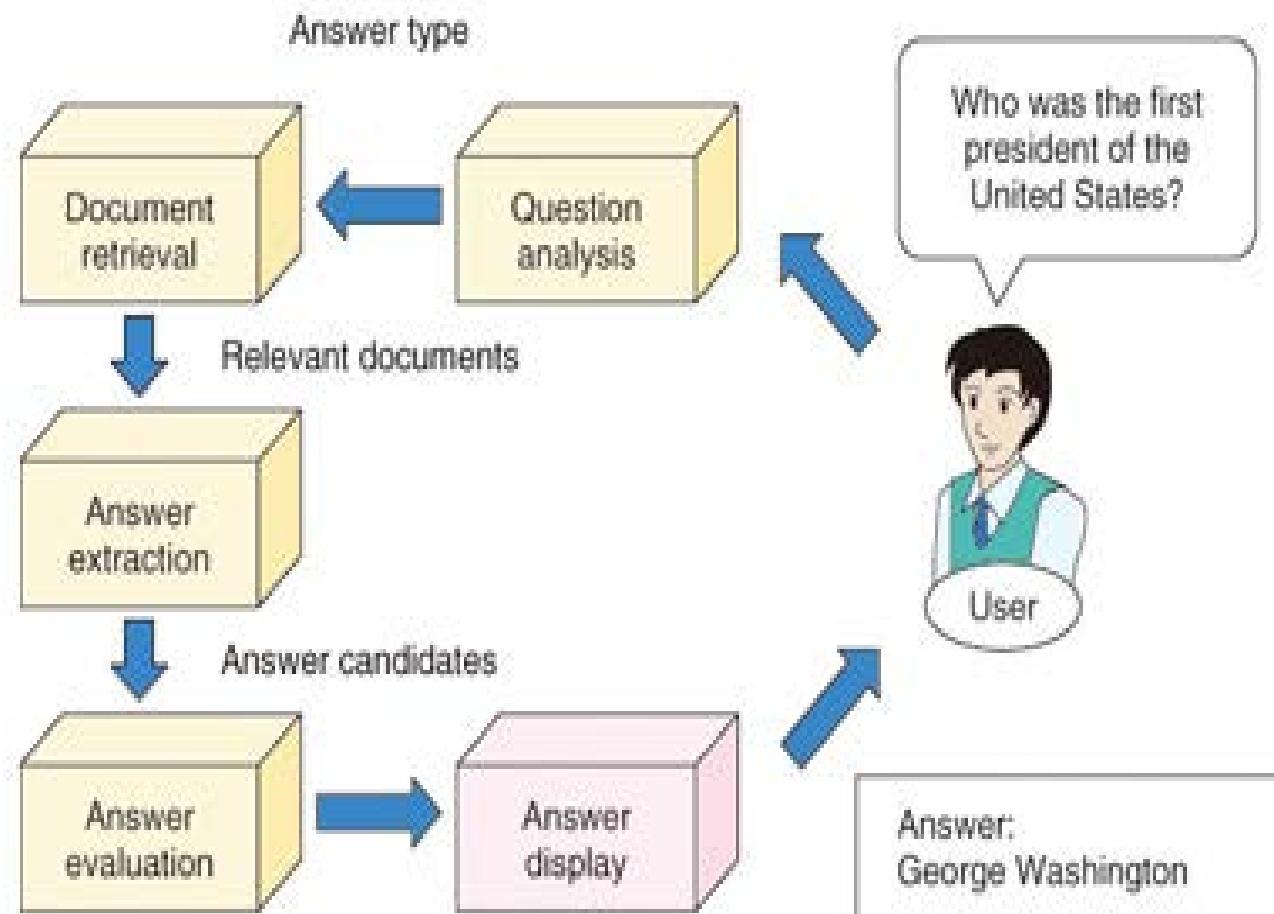


Figure: QA system Outline

Source: <https://www.ntt-review.jp/archive/ntttechnical.php?contents=ntr201307fa4.html>

# QA system architecture

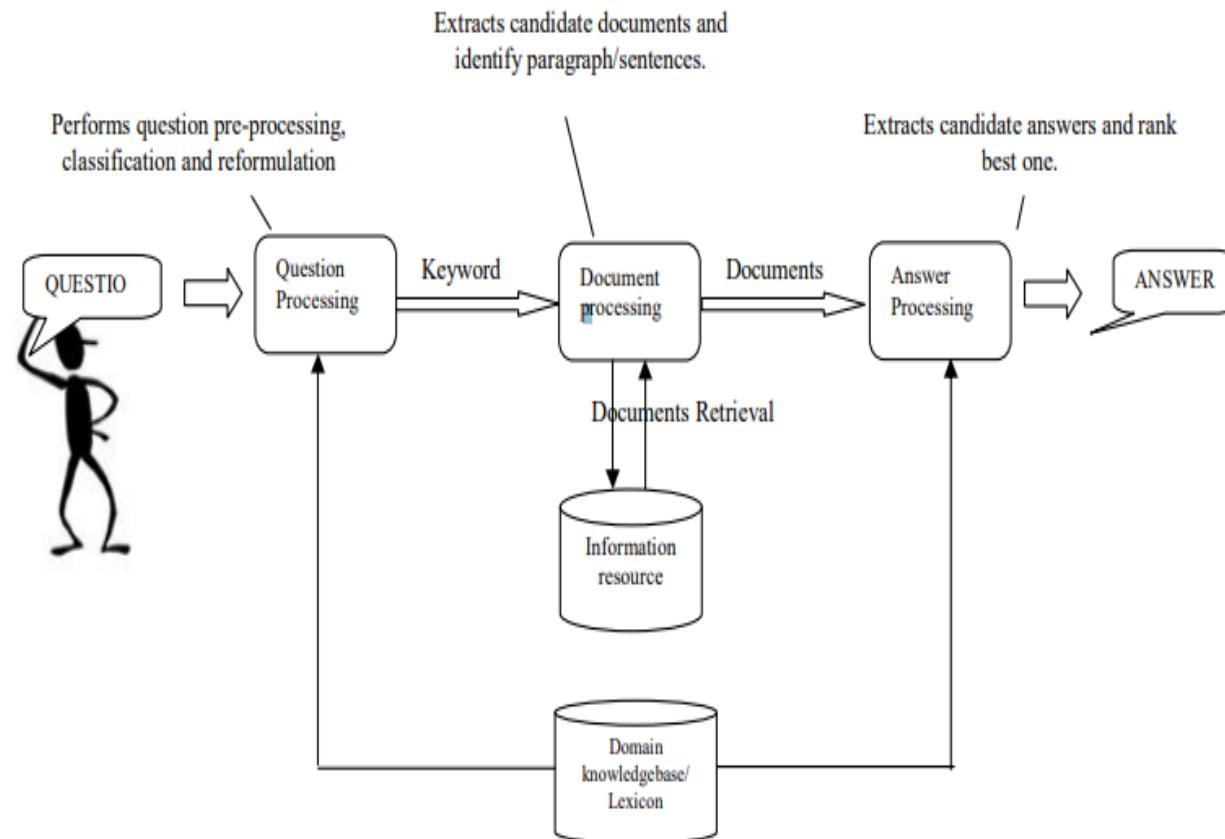


Figure: QA System Architecture

Source:

[https://www.researchgate.net/publication/323729727 Different Facets of Text Based Automated Question Answering System/link/5aca58a20f7e9bcd5198adf1/download](https://www.researchgate.net/publication/323729727_Different_Facets_of_Text_Based_Automated_Question_Answering_System/link/5aca58a20f7e9bcd5198adf1/download)

# QA system types

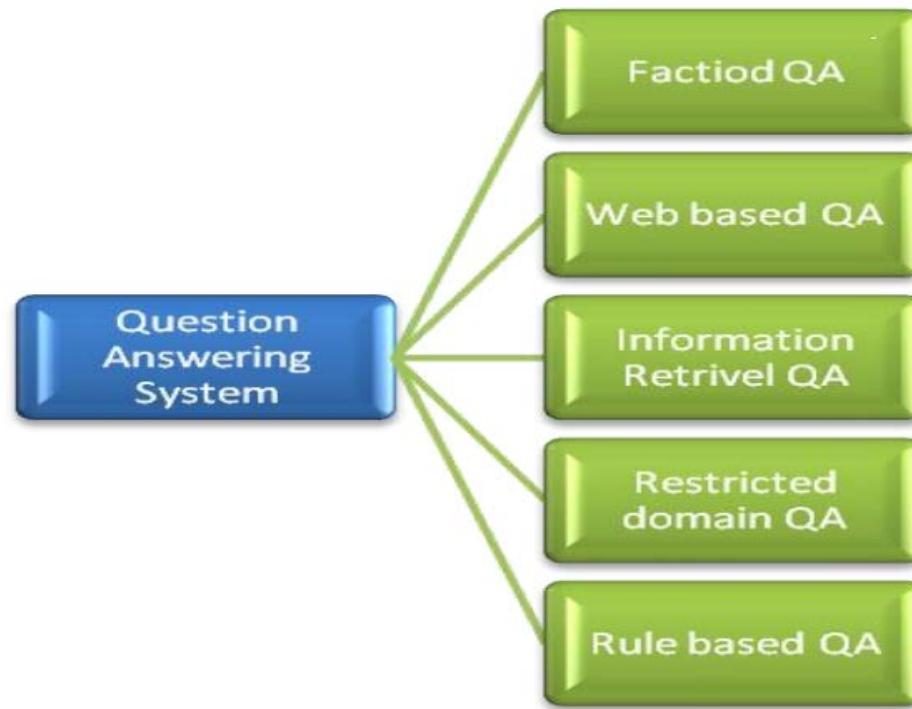


Figure: QA System Types

Source: [https://www.researchgate.net/publication/320978810\\_An\\_Overview\\_of\\_Question\\_Answering\\_System](https://www.researchgate.net/publication/320978810_An_Overview_of_Question_Answering_System)

# Text based QA systems

	Standard QA	Multilingual QA	Community QA	Interactive QA
Question type	Single sentence questions (Mostly factoid).	Single sentence questions in different accepted language	Multi sentence questions (Usually non-factoid).	Series of single sentence questions to have better understanding of the subject.
Question Reformulation	Automatic	Automatic	Manual	Automatic but user guided
Question Understanding	Depends on techniques (Shallow or deep linguistics) implemented	Depends on techniques implemented.	Depends on the understanding of community members responding to the asked question.	Good understanding as question representation is improved by real time interaction.
Answer Resource	Corpus, Knowledge base, Web documents	Corpus, Knowledge base, Web documents available for different languages	User (Expert) generated	Corpus, Knowledge base, Web documents
Answer representation	Short	Short	Long answers (or as required to the question)	Mixed answers
Answer reliability	Usually high	Average	Depends on potential experts	Average
Time lag	Immediate	Immediate	Have to wait until an answer is posted.	Real time response

Figure: Various QA System Representation

Source:

[https://www.researchgate.net/publication/323729727\\_Different\\_Facets\\_of\\_Text\\_Based\\_Automated\\_Question\\_Answering\\_System/link/5aca58a20f7e9bcd5198adf1/download](https://www.researchgate.net/publication/323729727_Different_Facets_of_Text_Based_Automated_Question_Answering_System/link/5aca58a20f7e9bcd5198adf1/download)

# Factoid question answering system

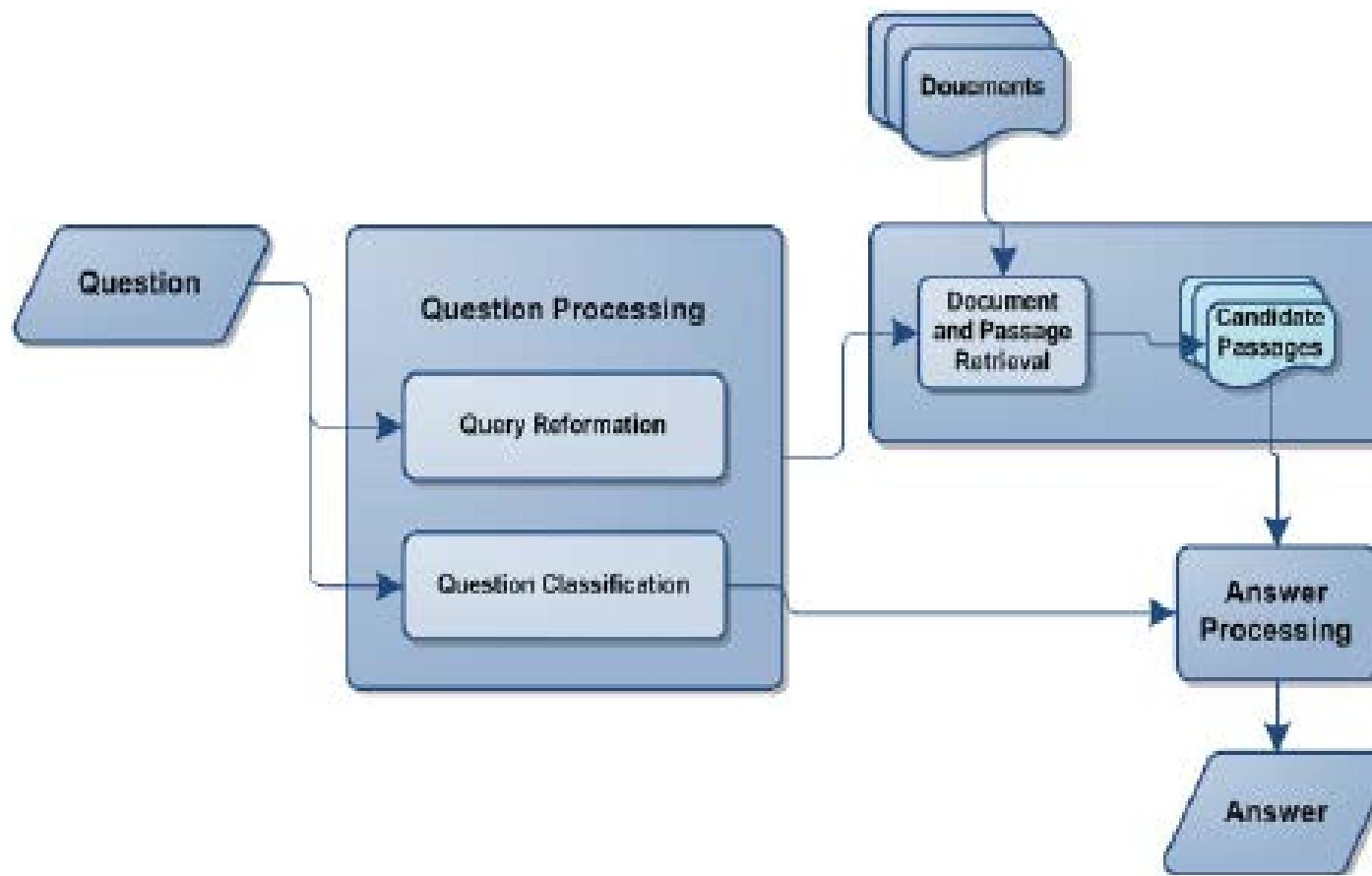


Figure: Factoid Question Answering system

Source: [https://www.researchgate.net/figure/The-common-architecture-of-a-factoid-question-answering-system\\_fig1\\_241886726](https://www.researchgate.net/figure/The-common-architecture-of-a-factoid-question-answering-system_fig1_241886726)

# Web based question answering system

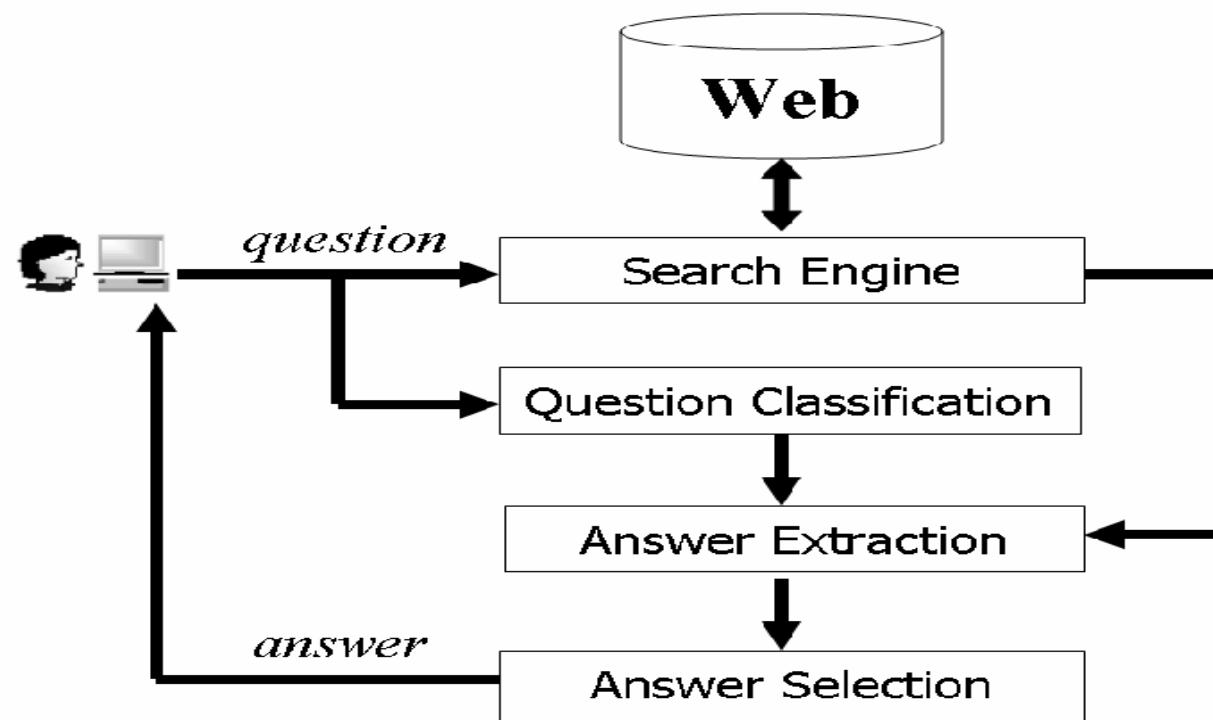


Figure: Web Based Question Answering System

Source: <https://www.semanticscholar.org/paper/A-Web-based-Question-Answering-System-Zhang-Lee/ad23647c895d57668fc202259dccbf29edb9e683>

# Information retrieval or information extraction based QA systems



IBM ICE (Innovation Centre for Education)

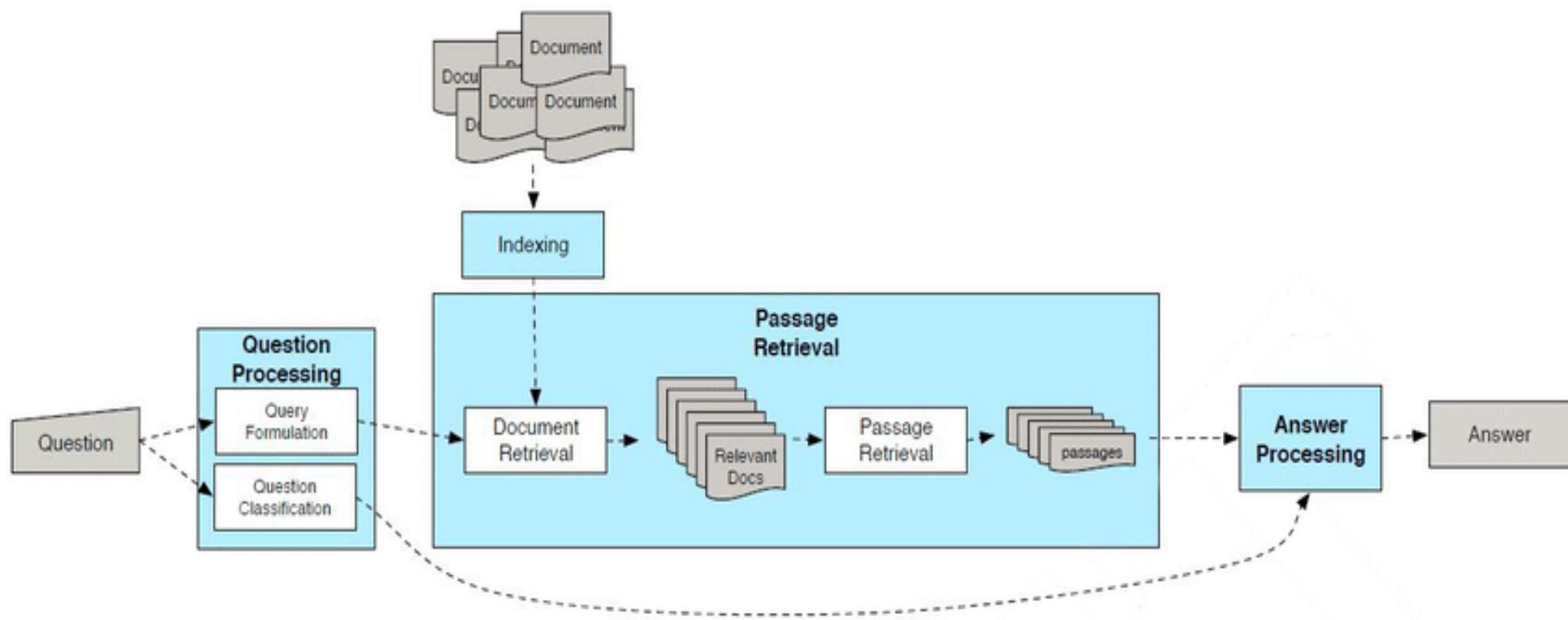


Figure: Information Retrieval or Information Extraction based QA systems

Source: [https://www.researchgate.net/figure/Information-retrieval-based-QA-system-procedure-115\\_fig1\\_273122359](https://www.researchgate.net/figure/Information-retrieval-based-QA-system-procedure-115_fig1_273122359)

# Restricted domain question answering

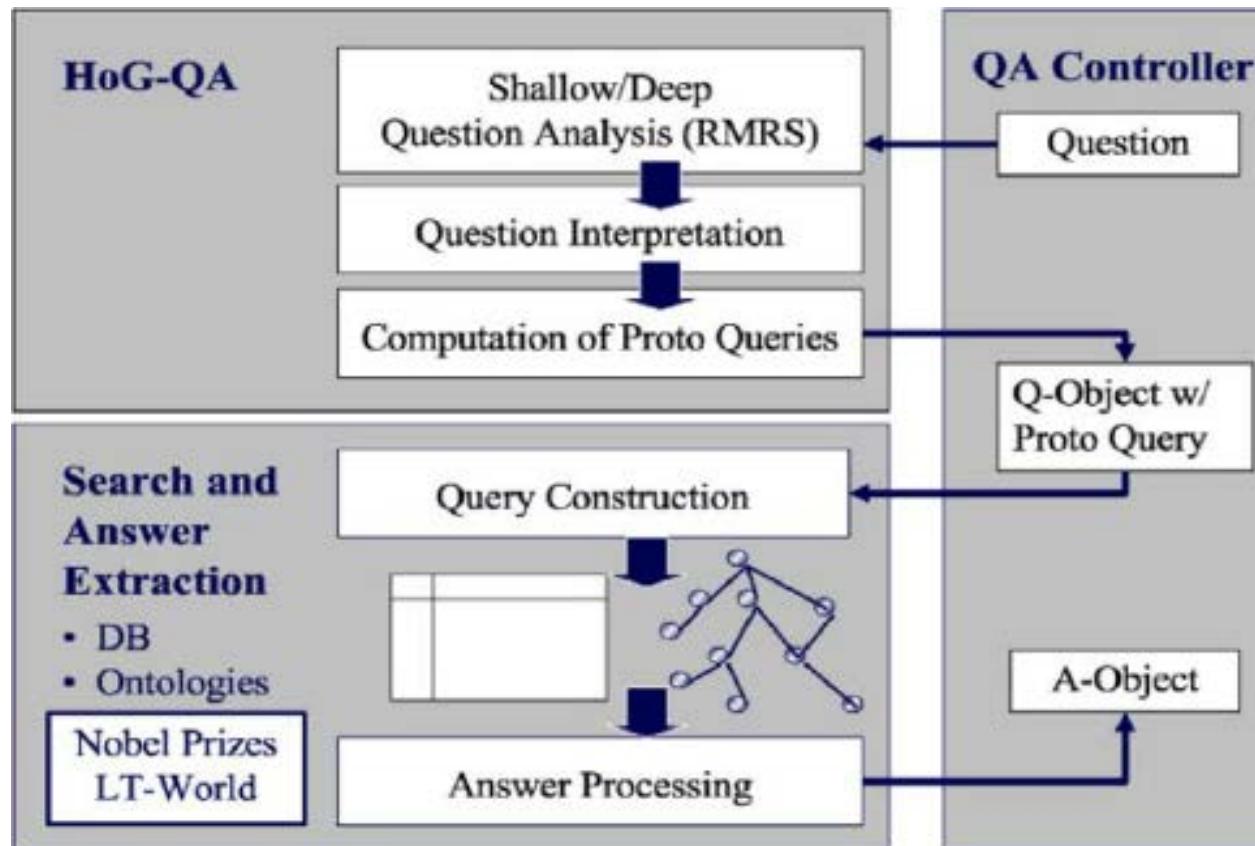


Figure: Restricted Domain Question Answering

Source: [https://www.researchgate.net/figure/Architecture-of-Domain-Restricted-question-answering-system\\_fig3\\_258651905](https://www.researchgate.net/figure/Architecture-of-Domain-Restricted-question-answering-system_fig3_258651905)

# Rule based question answering systems



IBM ICE (Innovation Centre for Education)

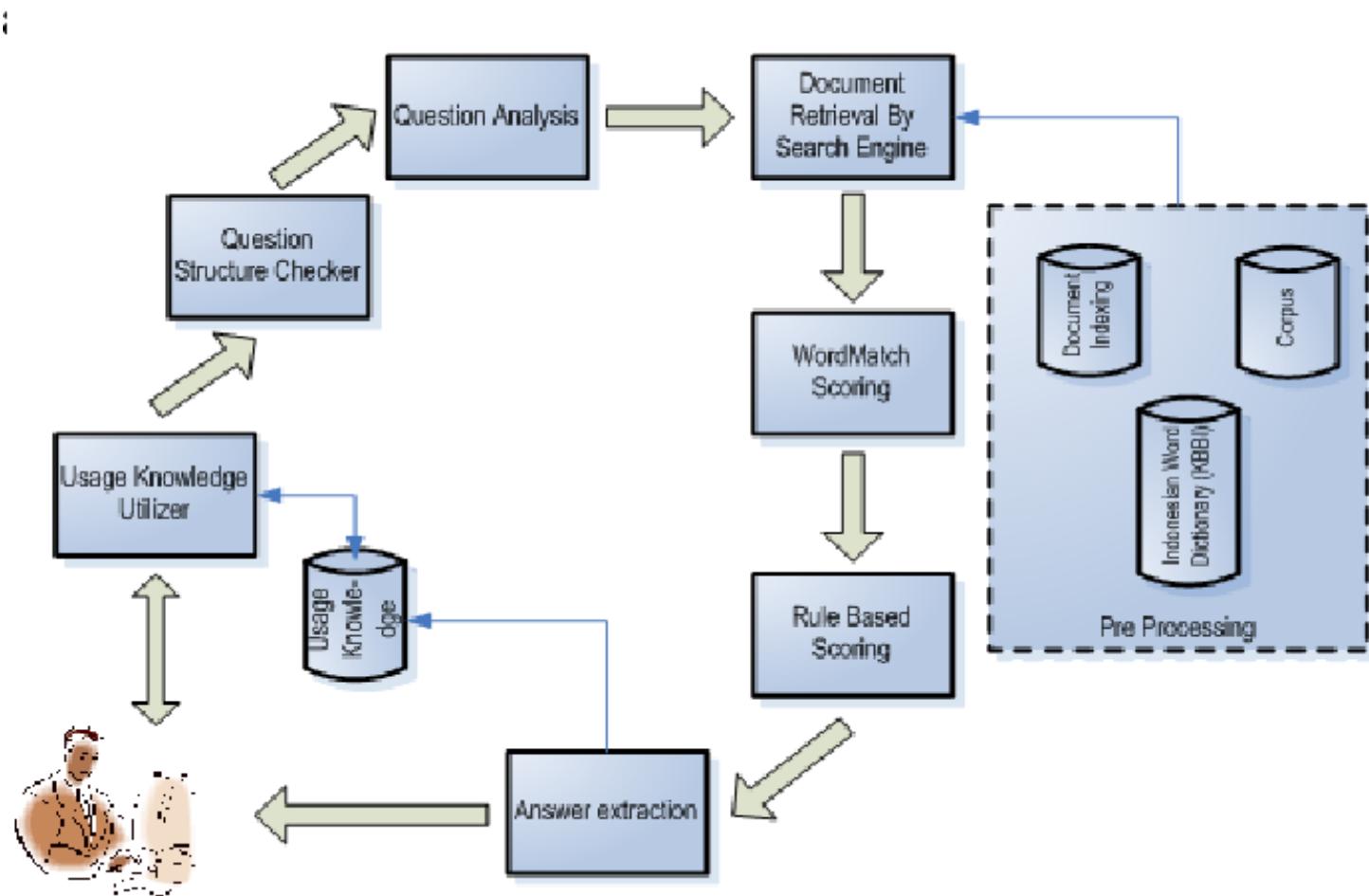


Figure: Rule Based Question Answering Systems

Source: <https://www.semanticscholar.org/paper/A-rule-based-question-answering-system-on-relevant-Gusmita-Durachman/995e84a3c0e4c5877df0e214f7bfc8355d0c616f>

# Self evaluation: Exercise 10

- To continue with the training, after learning the concepts of Information Retrieval and Question Answering in Natural Language Text Processing, it is time to write code to work with IR in NLP using the earlier topics implementing POS tagging, Tokenization and use it to compare similarities. It is instructed to utilize the concepts of reading data from files Tokenization, Word Similarity, POS tags and perform the following activity
- You are instructed to write the following activities using Python code.
- Exercise 10: Python code to build a recommendation system with text data and perform Information retrieval through queries.

# Self evaluation: Exercise 11

- To continue with the training, after learning the concepts of Information Retrieval and Question Answering in Natural Language Text Processing, it is time to write code to work with IR in NLP using the earlier topics implementing POS tagging, Tokenization and use it to compare similarities. It is instructed to utilize the concepts of reading data from files Tokenization, Word Similarity, POS tags and perform the following activity
- You are instructed to write the following activities using Python code.
- Exercise 11: Python code to Create a Rule-Based Chat bot.

# Information extraction (1 of 2)

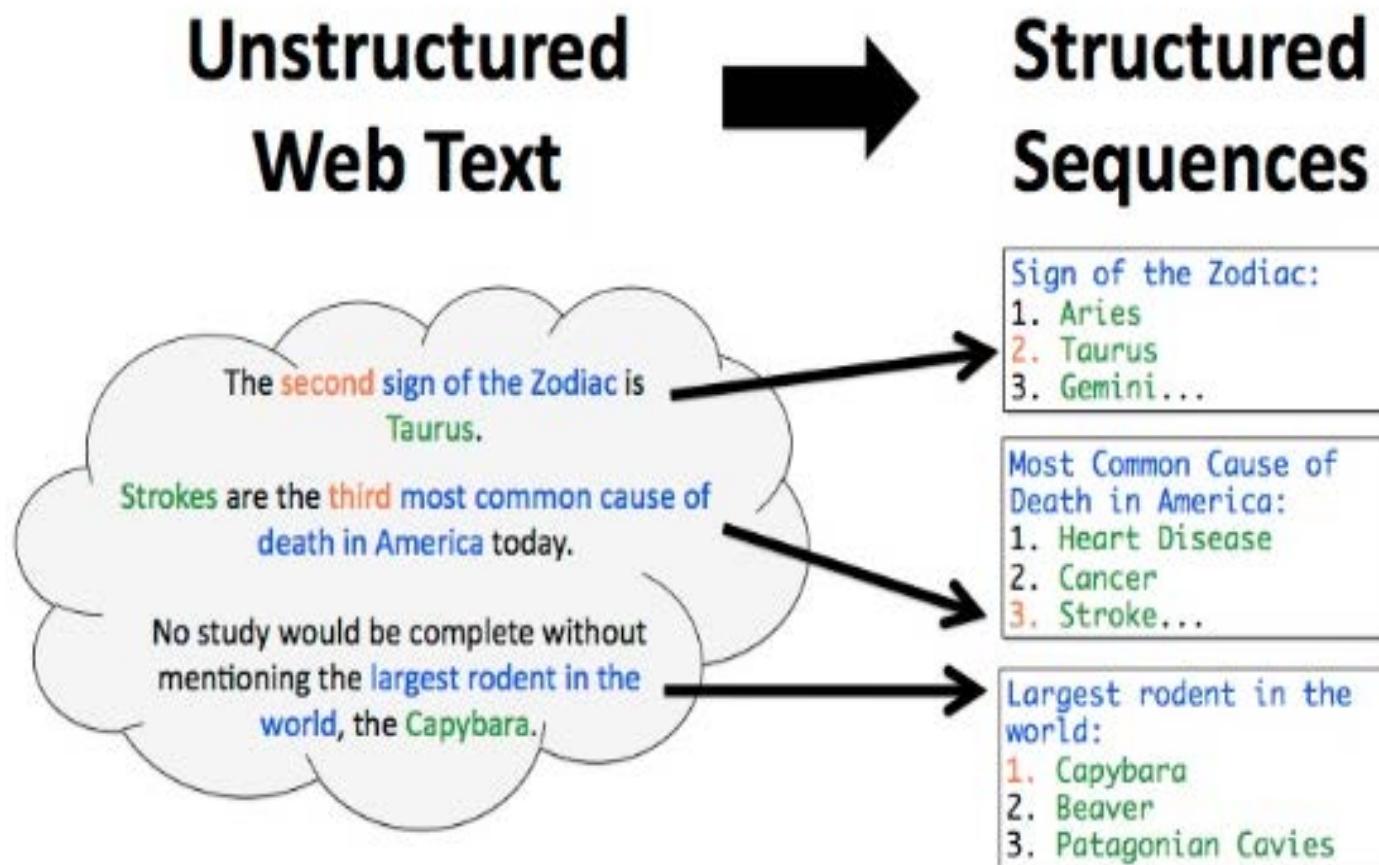


Figure: Information Extraction

Source: <https://www.slideshare.net/rubenizquierdobevia/information-extraction-45392844>

# Information extraction (2 of 2)

Indian captain Virat Kohli was dismissed cheaply for just 2 in Wellington on Friday by debutant Kyle Jamieson extending a rare lull in the batsman's stellar career. Throughout the ongoing New Zealand tour, Kohli has managed to score just a single fifty across 8 innings in all 3 international formats.

Figure: Sample Document

Source: <https://www.analyticsvidhya.com/blog/2020/06/nlp-project-information-extraction/>

- The following information can be extracted from the text:

Country – India, Captain – Virat Kohli

Batsman – Virat Kohli, Runs – 2

Bowler – Kyle Jamieson

Match venue – Wellington

Match series – New Zealand

Series highlight – single fifty, 8 innings, 3 formats

Figure: Extracted Information

Source: [SELF](#)

# Working of information extraction

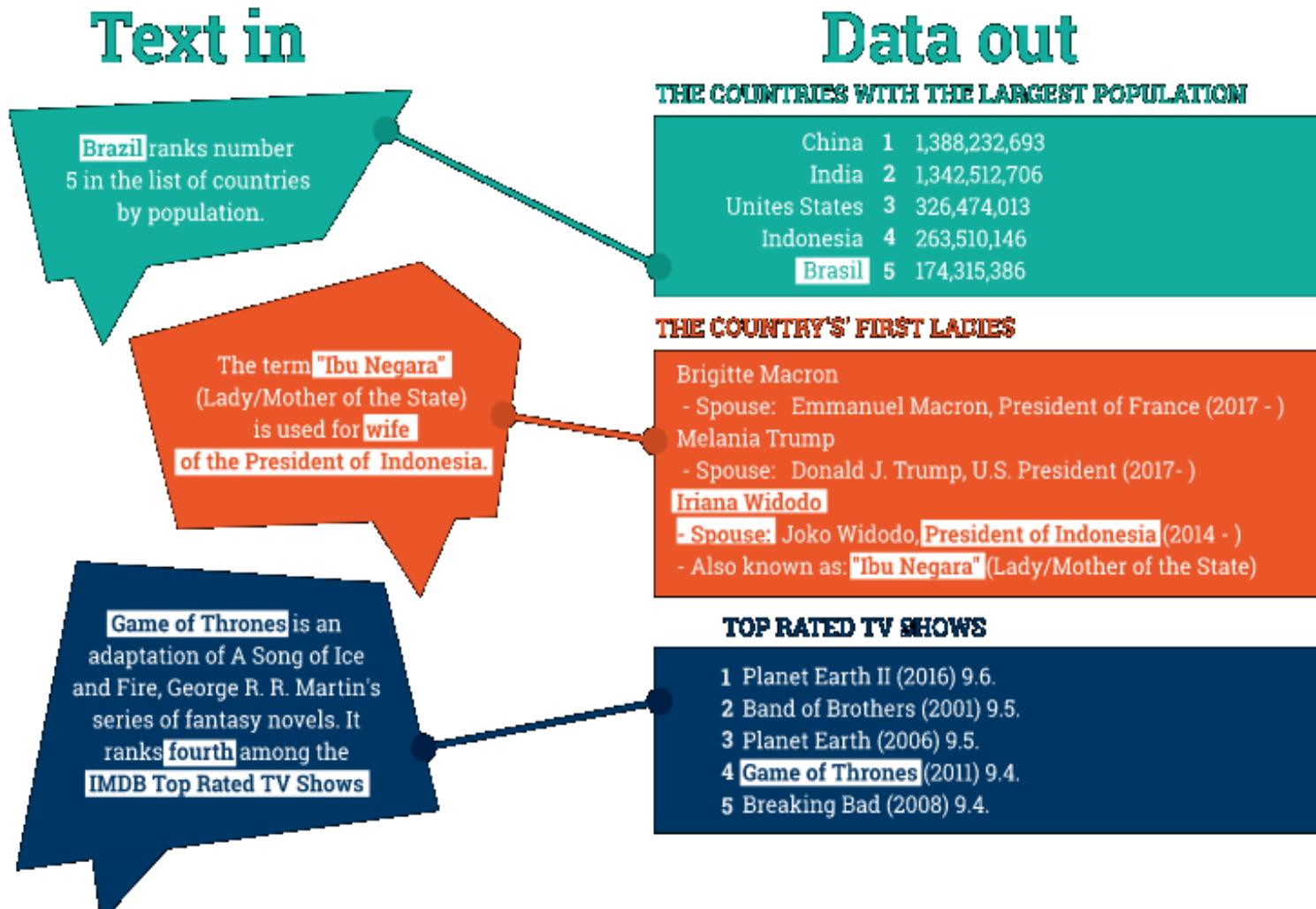


Figure: Simple IE Activity

# Information extraction applications (1 of 2)



IBM ICE (Innovation Centre for Education)

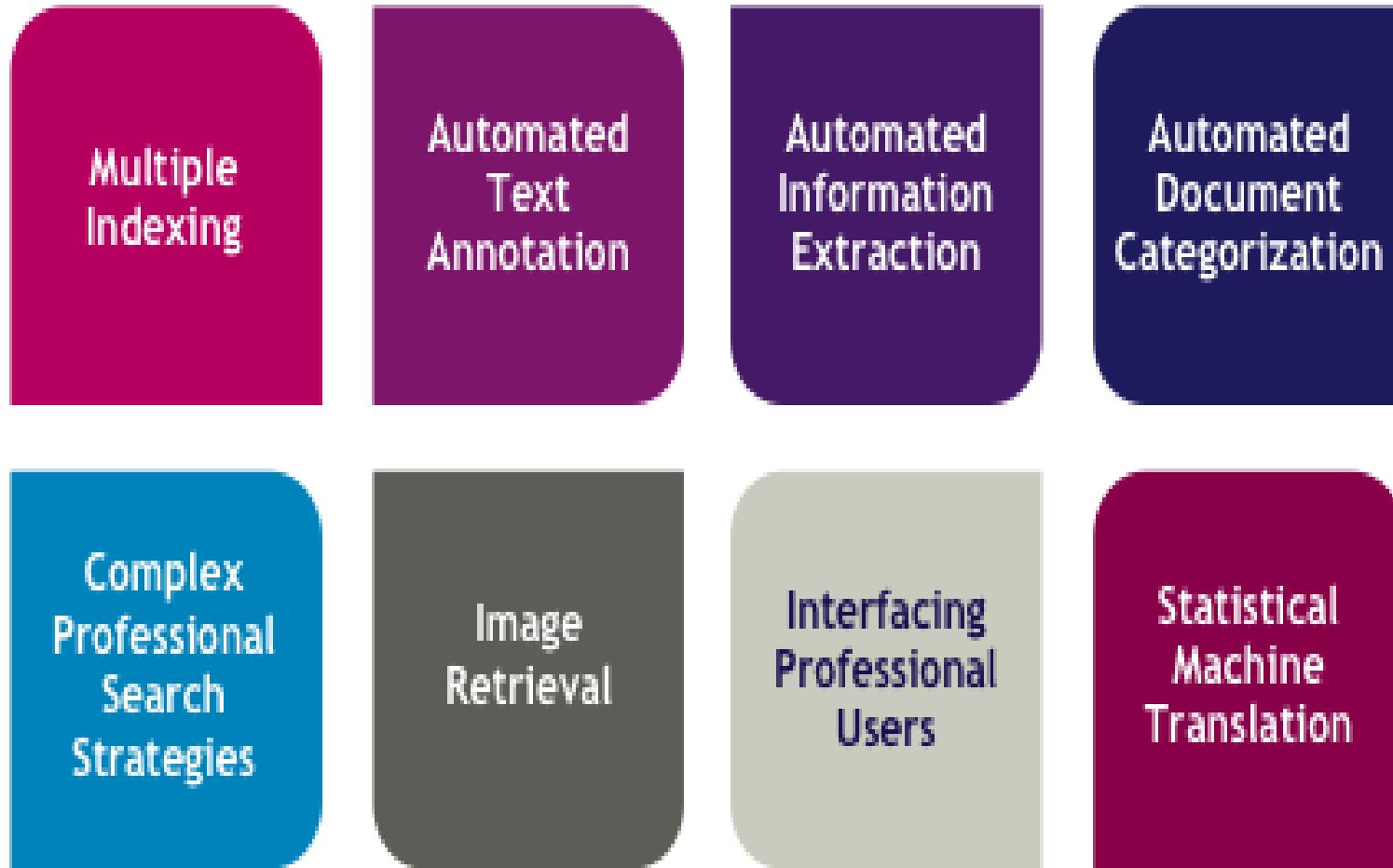


Figure: IE Sectors

Source: <https://www.ir-facility.org/research-areas>

# Information extraction architecture (2 of 2)



IBM ICE (Innovation Centre for Education)

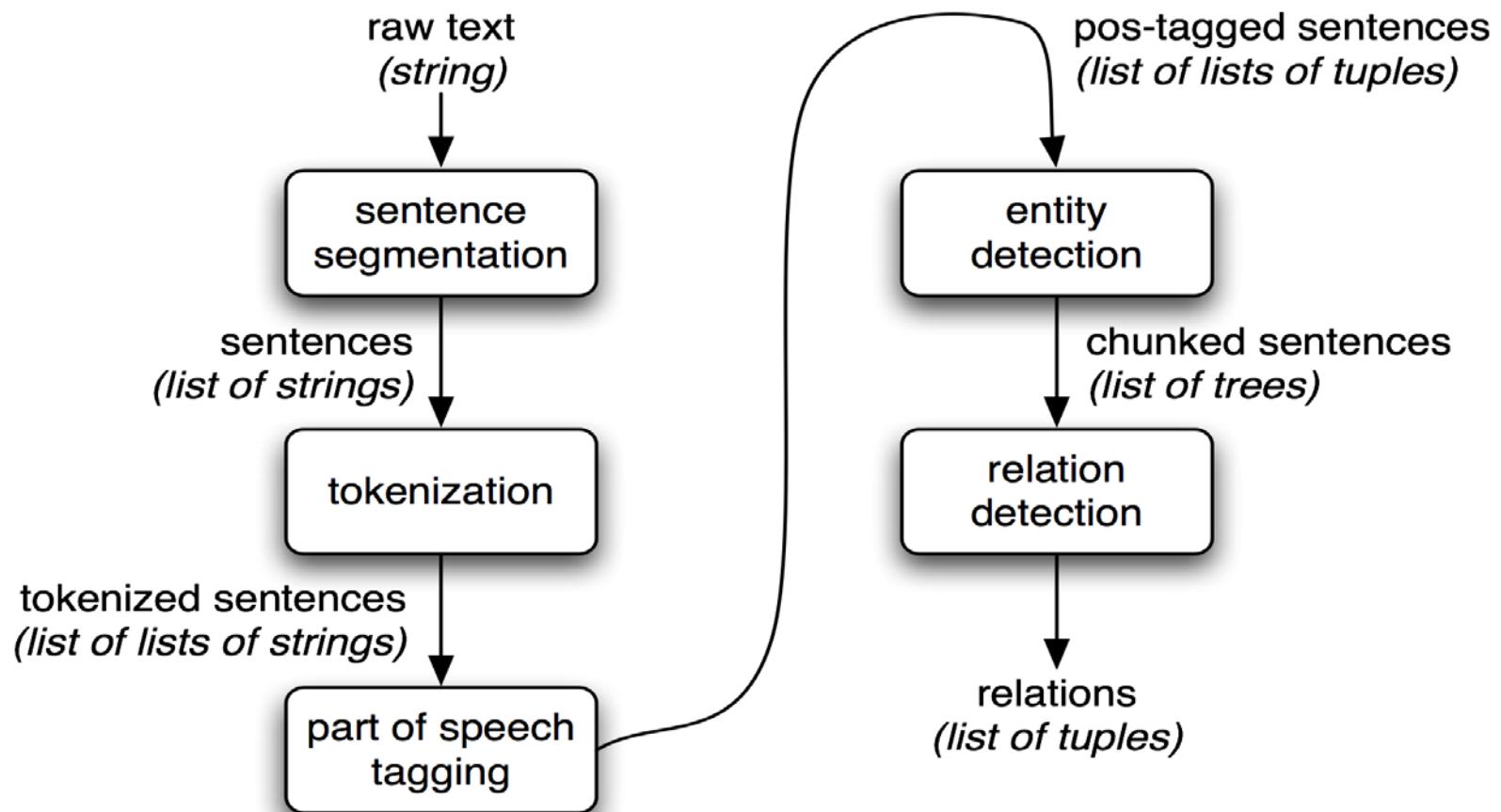


Figure: Information Extraction Architecture

Source: <https://www.nltk.org/book/ch07.html>

# Chunking (1 of 2)

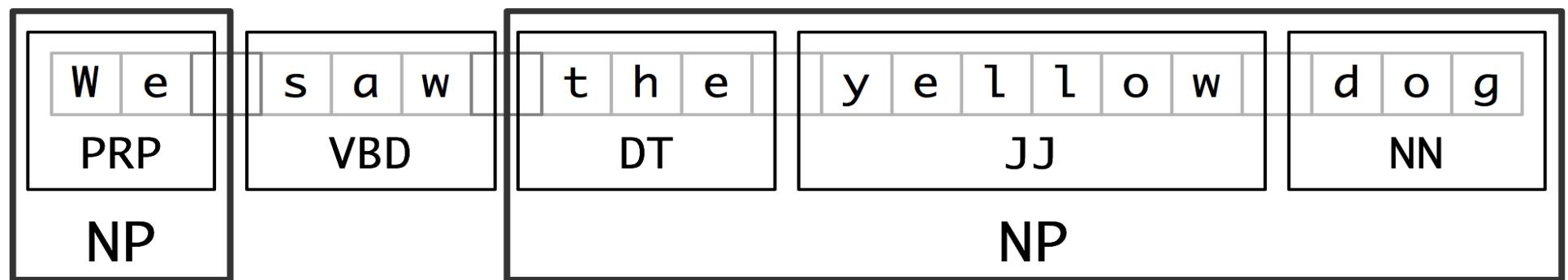


Figure: Simple Tokens and Chunks

Source: <https://www.nltk.org/book/ch07.html#sec-ner>

# Chunking (2 of 2)

- Tag patterns
- Regular expression Chunking

```
nouns = [("money", "NN"), ("market", "NN"), ("fund", "NN")]
```

```
grammar = "NP: {<NN><NN>} # Chunk two consecutive nouns"
```

```
nltk.RegexpParser(grammar)
```

- Structure: (S (NP money/NN market/NN) fund/NN).

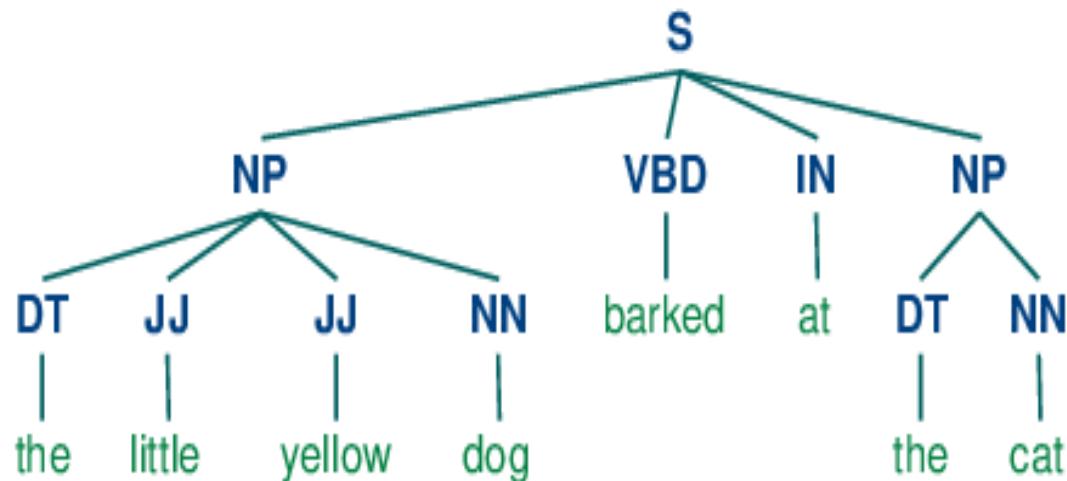


Figure: Parse Tree

Source: <https://www.nltk.org/book/ch07.html#sec-ner>

# Representing chunks: Tags vs trees

W e	s a w	t h e	y e l l o w	d o g
PRP	VBD	DT	JJ	NN
B-NP	O	B-NP	I-NP	I-NP

Figure: Chunks with Tags

Source: <https://www.nltk.org/book/ch07.html#sec-ner>

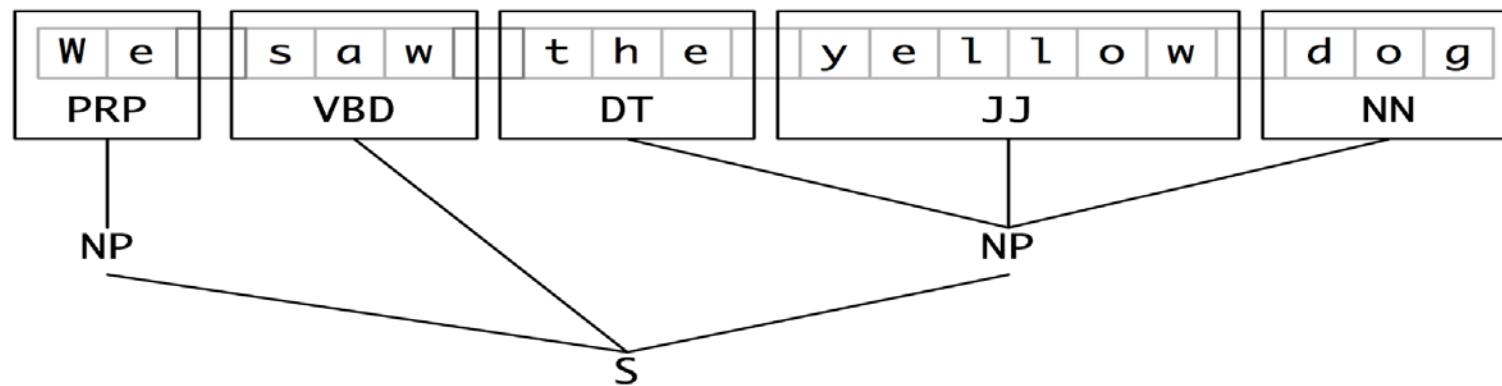


Figure: Chunks with Trees

Source: <https://www.nltk.org/book/ch07.html#sec-ner>

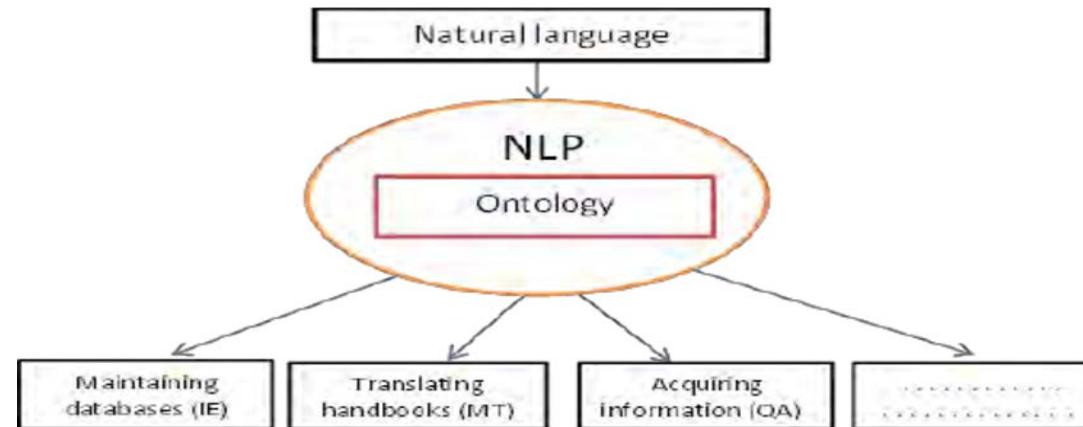


Figure: Ontology Representation

Source: <https://www.slideshare.net/athmanhajhamou/use-of-ontologies-in-natural-language-processing-2>

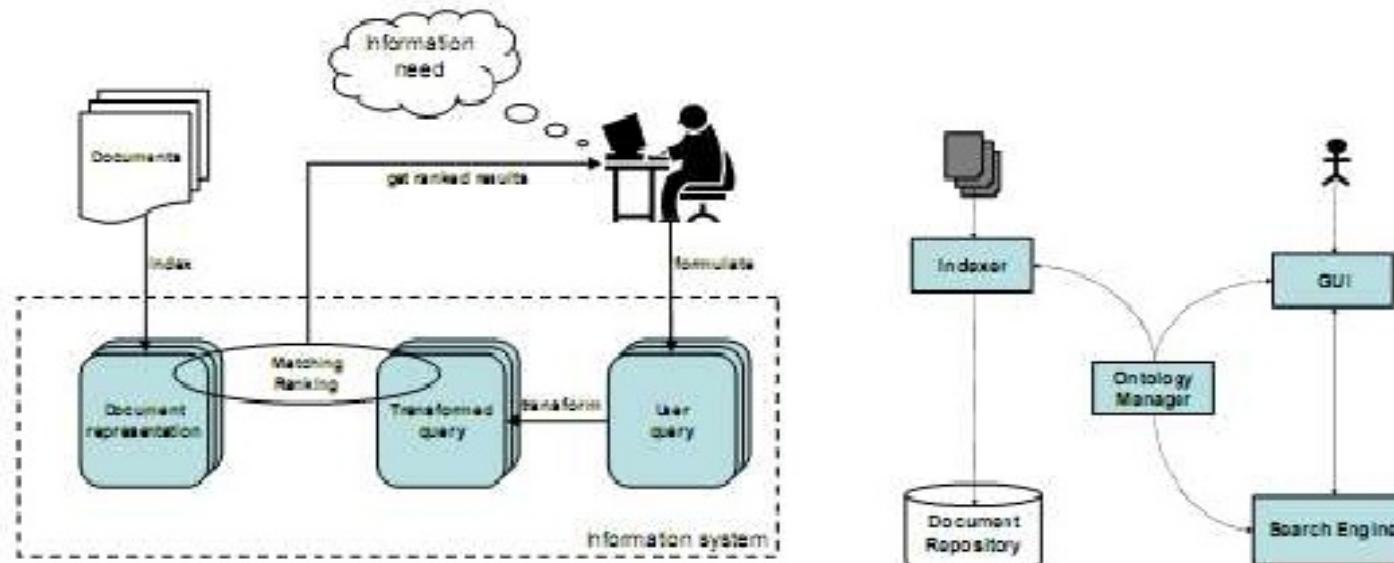


Figure: Ontology Outline in NLP

Source: [https://www.researchgate.net/figure/Relation-between-natural-language-NLP-and-ontology\\_fig1\\_270471292](https://www.researchgate.net/figure/Relation-between-natural-language-NLP-and-ontology_fig1_270471292)

# Ontology classifications and process

- 

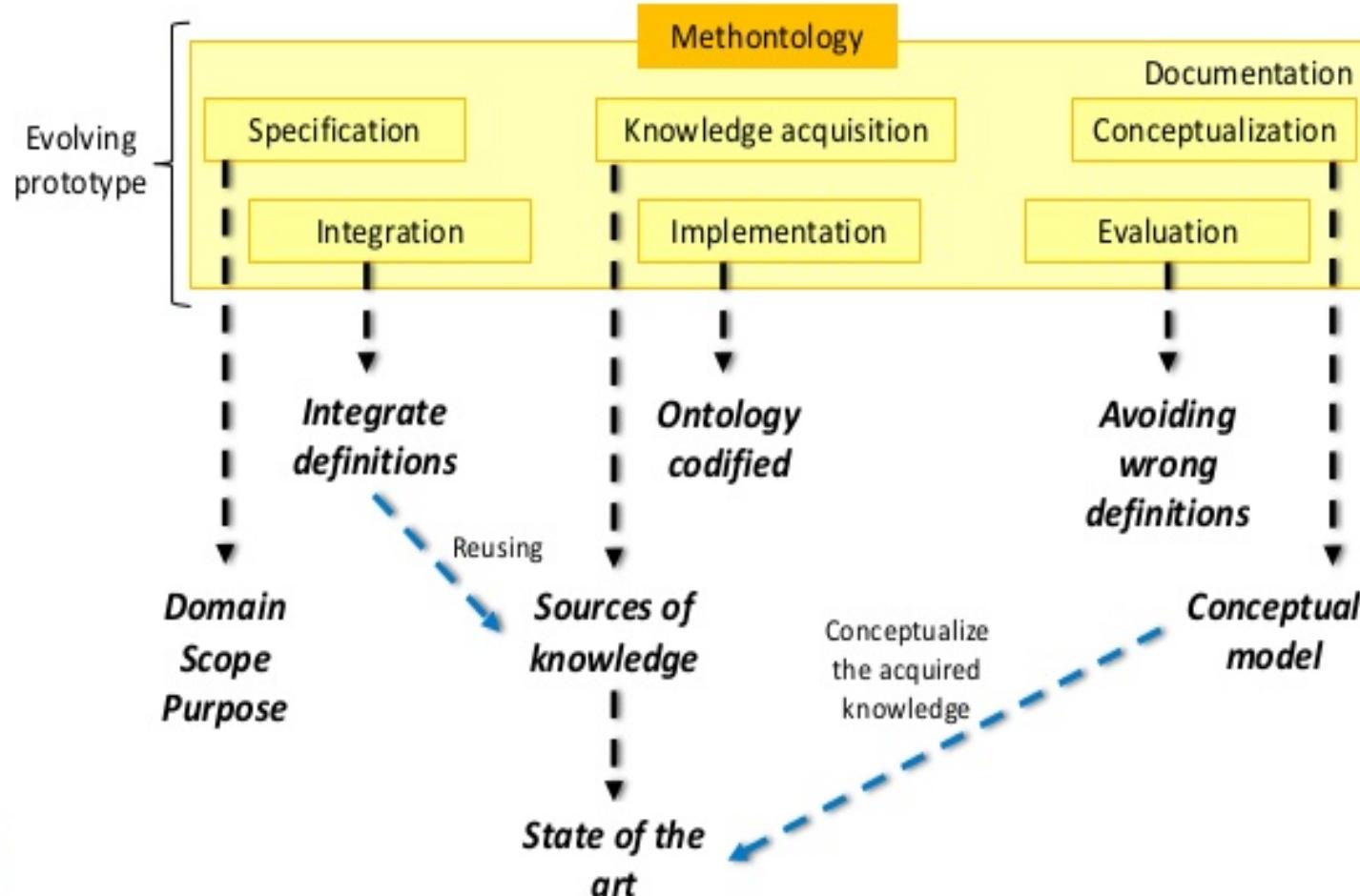


Figure: Ontology Process

Source: <https://www.slideshare.net/gessiupc/rcis2014-35471364>

# Why ontology and its advantages

Traditional CMS	Contributions of Ontology
Match-making often ineffective because of rigid definition of contents (e.g., categories) predefined by service providers	Shared and agreed ontology provides common, flexible, and extensible definitions of multimedia contents for match-making and subsequent business processes
Difficult to specify unclear types of multimedia content out of predefined categories	Complicated use requirements can be decomposed into simple genres for elicitation of options

Figure: Advantages of Ontologies

Source: <https://slideplayer.com/slide/5014295/>

# Ontology components

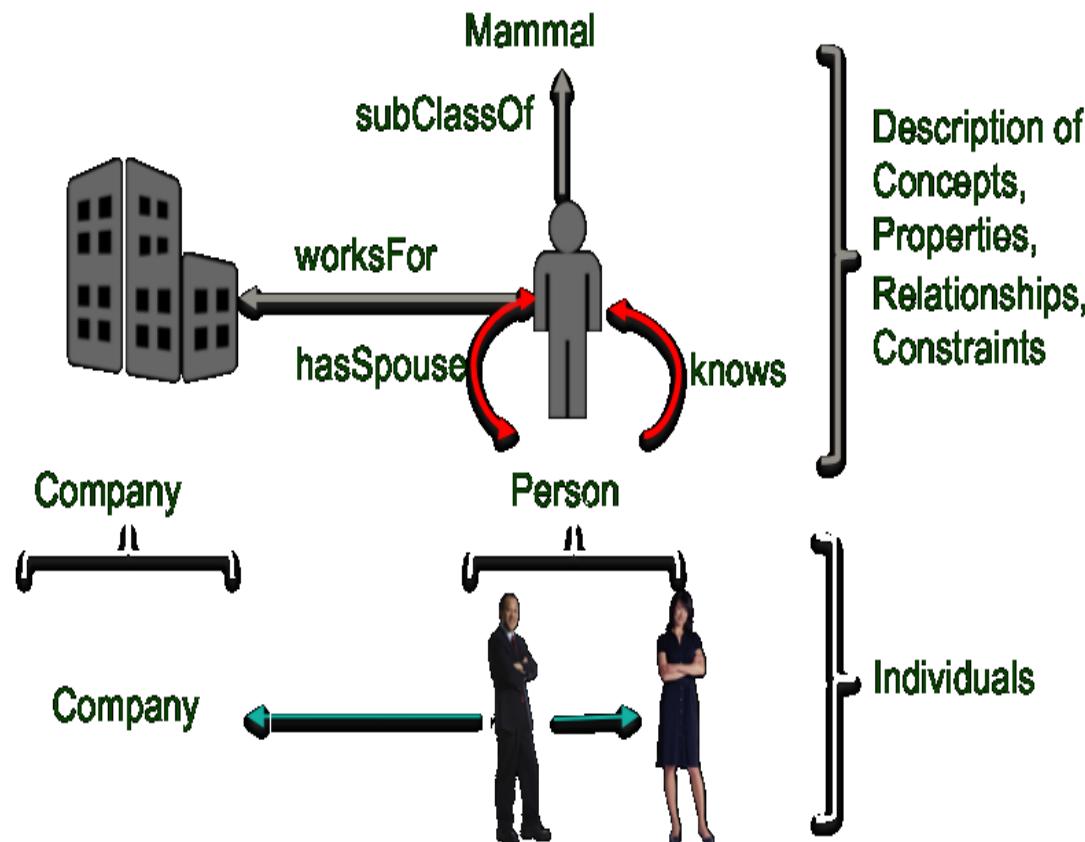
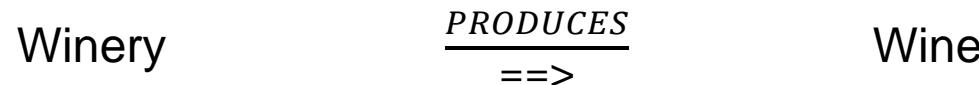


Figure: Ontology Components

Source: <http://graphdb.ontotext.com/documentation/enterprise/devhub/ontologies.html>

# Levels of formality

- Highly informal
  - Represented in simple natural language.
  - Wine is a winery product
- Semi informal
  - Represented in structured form of natural language
  - Wine PRODUCED IN Winery
- Semi formal: Represented in a formally defined language.



# Ontology construction approaches

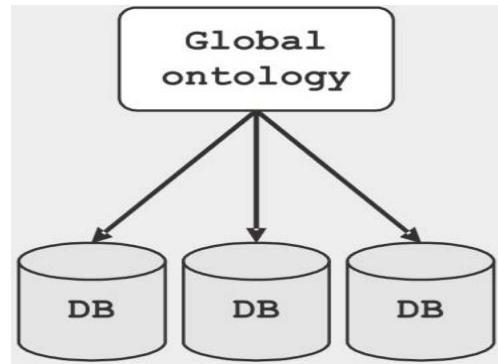


Figure: Single ontology approach

Source: [https://www.researchgate.net/publication/220327569/figure/fig1/AS\\_411993979801600@1475238427128/Single-Ontology-Approach\\_Q640.jpg](https://www.researchgate.net/publication/220327569/figure/fig1/AS_411993979801600@1475238427128/Single-Ontology-Approach_Q640.jpg)

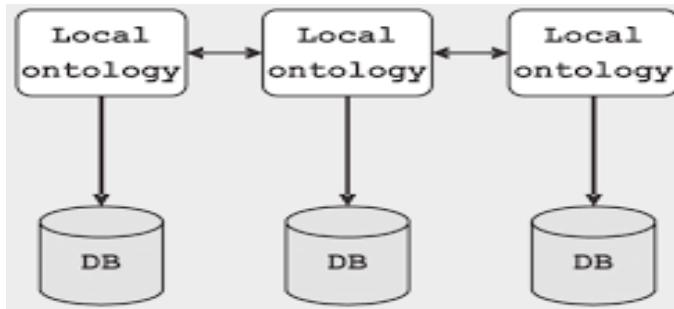


Figure: Multiple ontology approach

Source: [https://www.researchgate.net/publication/220327569/figure/fig2/AS\\_411993979801601@1475238427796/Multiple-Ontology-Approach.png](https://www.researchgate.net/publication/220327569/figure/fig2/AS_411993979801601@1475238427796/Multiple-Ontology-Approach.png)

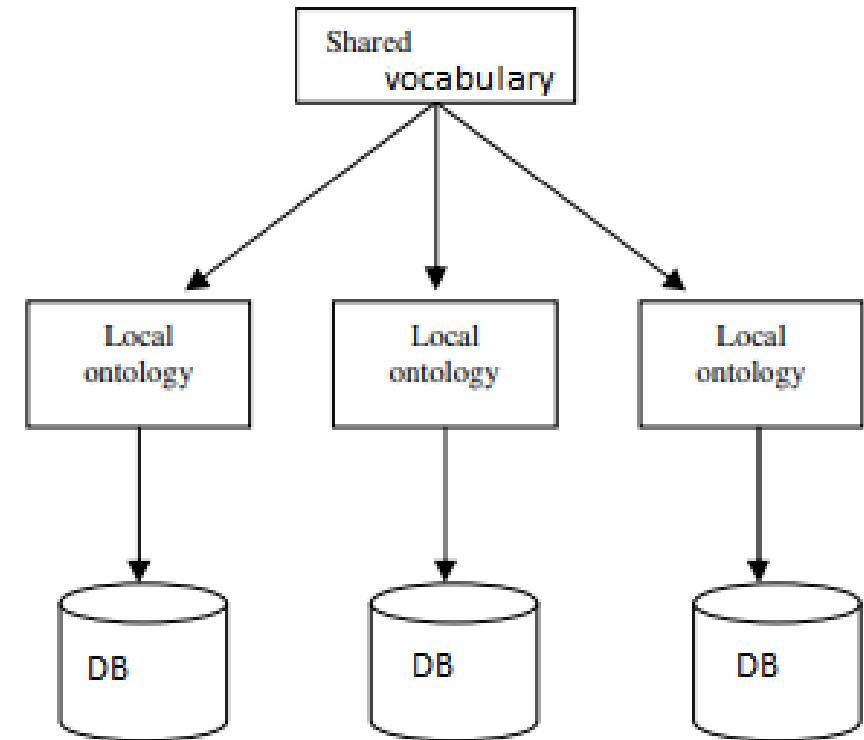


Figure: Hybrid ontology approach

Source: <http://www.ijecbs.com/January2011/N5Jan2011.pdf>

# Ontology construction

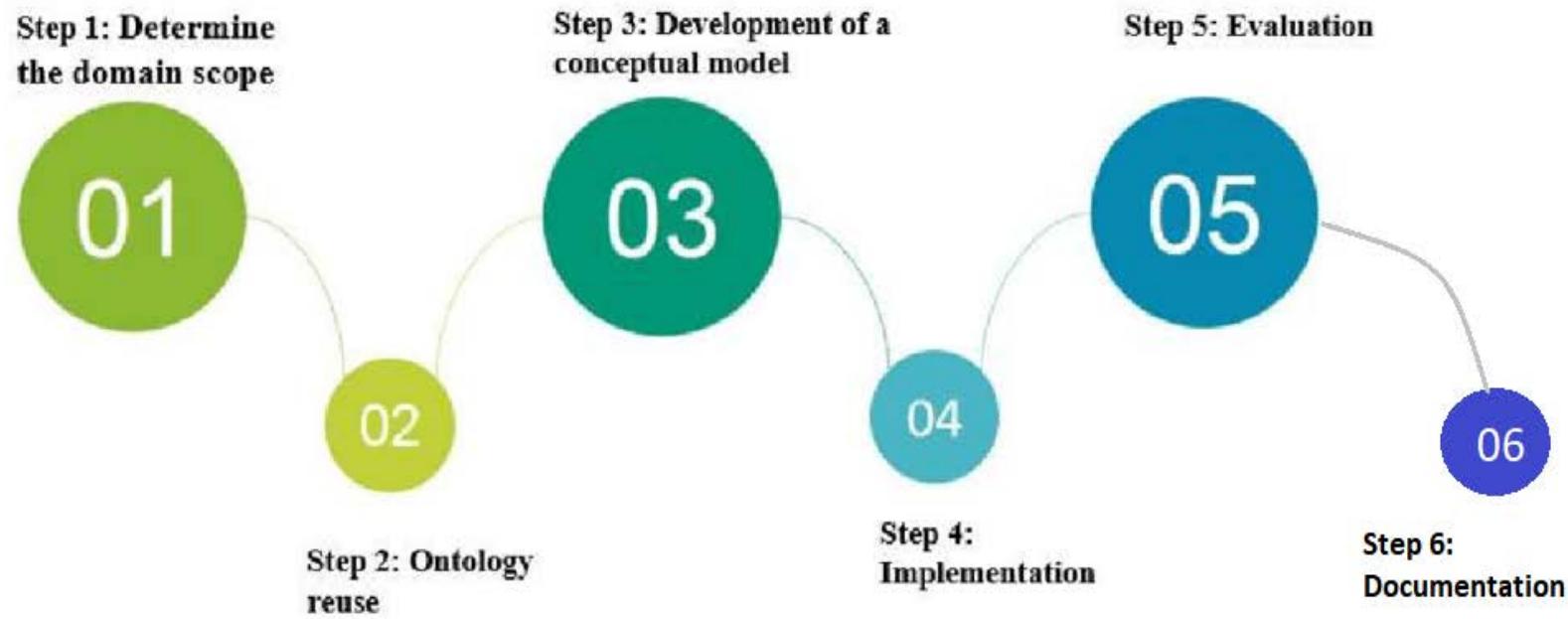


Figure: Ontology Creation Steps

Source: [https://www.researchgate.net/figure/Ontology-Construction-the-basic-steps-proposed-by-Sanchez-58\\_fig3\\_329364160](https://www.researchgate.net/figure/Ontology-Construction-the-basic-steps-proposed-by-Sanchez-58_fig3_329364160)

# Checkpoint (1 of 2)

## Multiple choice questions:

1. A data structure that maps terms back to the parts of a document in which they occur is called an (select the best answer):
  - a) Postings list
  - b) Incidence Matrix
  - c) Dictionary
  - d) Inverted Index
  
2. In information retrieval, extremely common words which would appear to be of little value in helping select documents that are excluded from the index vocabulary are called:
  - a) Stop Words
  - b) Tokens
  - c) Lemmatized Words
  - d) Stemmed Terms
  
3. A group of related documents against which information retrieval is employed is called:
  - a) Corpus
  - b) Text Database
  - c) Index Collection
  - d) Repository

# Checkpoint solutions (1 of 2)

## Multiple choice questions:

1. A data structure that maps terms back to the parts of a document in which they occur is called an (select the best answer):
  - a) Postings list
  - b) Incidence Matrix
  - c) Dictionary
  - d) **Inverted Index**
2. In information retrieval, extremely common words which would appear to be of little value in helping select documents that are excluded from the index vocabulary are called:
  - a) **Stop Words**
  - b) Tokens
  - c) Lemmatized Words
  - d) Stemmed Terms
3. A group of related documents against which information retrieval is employed is called:
  - a) **Corpus**
  - b) Text Database
  - c) Index Collection
  - d) Repository

# Checkpoint (2 of 2)

## Fill in the blanks:

1. A crude heuristic process that chops off the ends of the words to reduce inflectional forms of words and reduce the size of the vocabulary is called \_\_\_\_\_.
2. \_\_\_\_\_ systems deal with queries that are limited to a domain.
3. \_\_\_\_\_ is the specification of shared conceptual terminologies.
4. \_\_\_\_\_ approach relies on Global ontology for the information resources.

## True or False:

1. Stemming increases the size of the vocabulary. True/False
2. In multilingual question answering system the query and the response does not belong to a language but from multiple languages. True/False
3. Chunks created by various processes can be represented either through tags or through trees. True/False

# Checkpoint solutions (2 of 2)

## Fill in the blanks:

1. A crude heuristic process that chops off the ends of the words to reduce inflectional forms of words and reduce the size of the vocabulary is called **Stemming**.
2. **Closed domain** systems deal with queries that are limited to a domain.
3. **Ontology** is the specification of shared conceptual terminologies.
4. **Single ontology** approach relies on Global ontology for the information resources.

## True or False:

1. Stemming increases the size of the vocabulary. **False**
2. In multilingual question answering system the query and the response does not belong to a language but from multiple languages. **True**
3. Chunks created by various processes can be represented either through tags or through trees. **True**

# Question bank

## Two mark questions:

1. How is Information Retrieval in NLP achieved?
2. How does Web Based Question Answering System work?
3. What is the difference between Representing Chunks as Tags vs Trees?
4. What are the advantages of Ontology Representation?

## Four mark questions:

1. Explain the Mathematical basis model in IR.
2. Write about QA system architecture.
3. Explain the Working of information extraction.
4. What are the components of ontology?

## Eight mark questions:

1. What are the design features of information retrieval and its impact.
2. Explain in detail the steps involved in creation of ontologies with examples.

# Unit summary

**Having completed this unit, you should be able to:**

- Understand what is information retrieval and the concepts
- Learn about work with the steps in IR and perform IR
- Gain knowledge on information answering, the various types of QA, how to model a QA
- Understand the concepts of information extraction, basic ideas and operations in IE
- Learn about what is ontology construction, the types, categories and steps involved in OC

# Emerging Applications of Natural Language Generation in Information Visualization, Education, and Health Care



# Unit objectives

**After completing this unit, you should be able to:**

- Gain knowledge on the process of Multimedia Presentation Generation
- Learn the concept of Language Interfaces for Intelligent Tutoring Systems
- Gain an insight into Argumentation for Healthcare Consumers
- Learn the concepts of Clinical Decision Support Systems
- Understand the core concepts of Sentiment Analysis and Subjectivity

# Multimedia Presentation Generation

- Multimedia generation → Combining the various forms of media
- Images, audio and video in the results
- Corpora → Rich Layout, diagrams
- Generators → Enhanced → Multimedia content

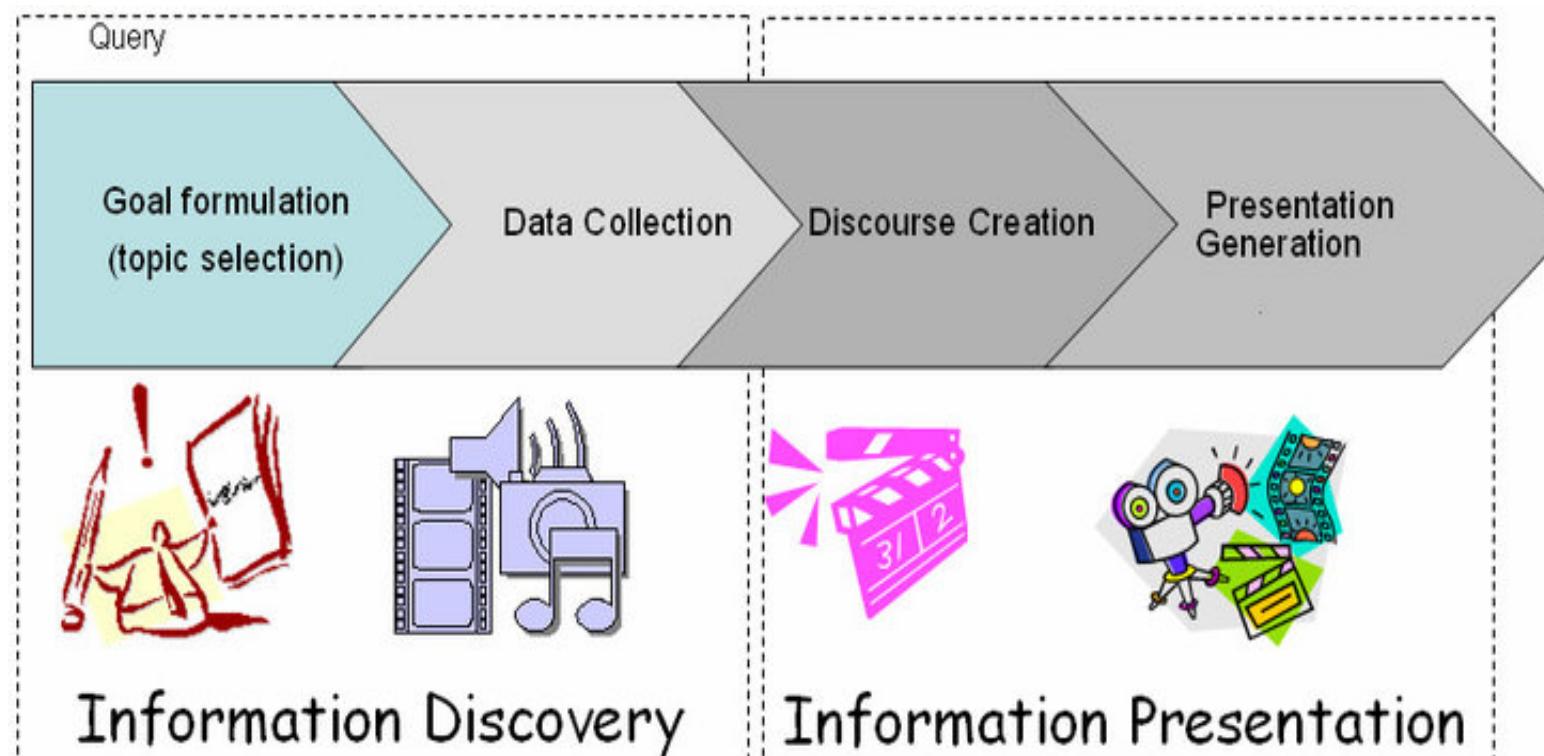


Figure: Multimedia Presentation Generation Outline

Source: [https://www.researchgate.net/figure/Presentation-generation-steps-in-MANA\\_fig2\\_259922670](https://www.researchgate.net/figure/Presentation-generation-steps-in-MANA_fig2_259922670)

# Focus points to add multimedia in NLG

- Meaning of the text
- Style
- Wording of the text

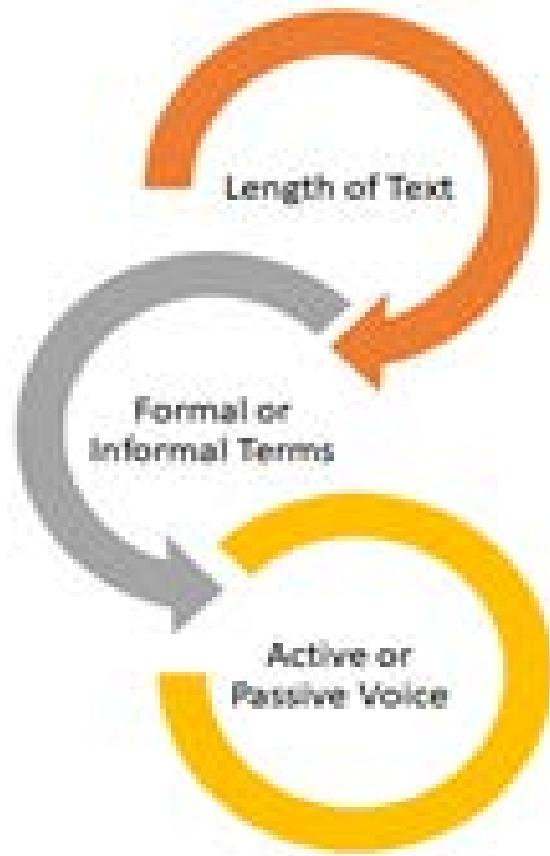


Figure: Text Style

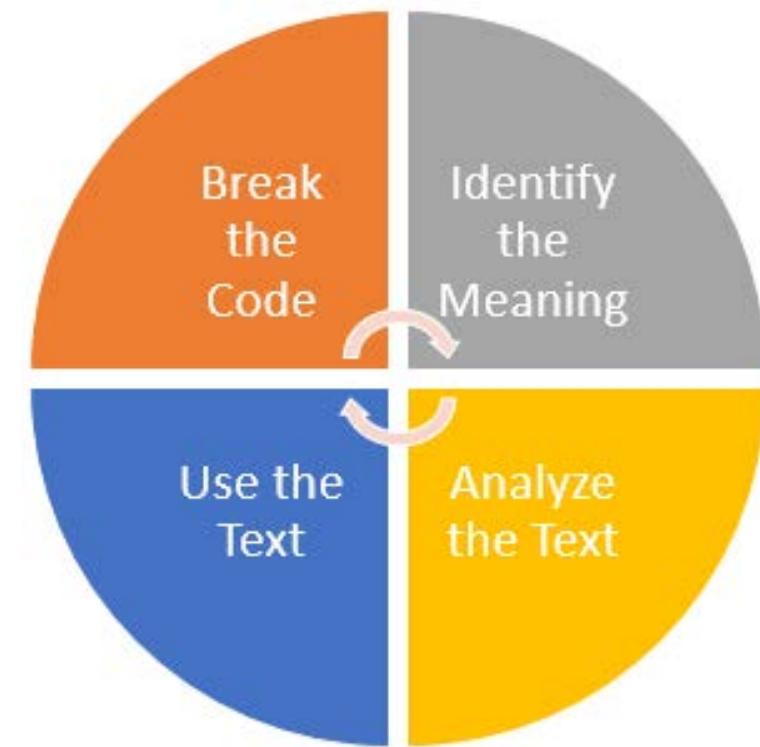


Figure: Text Meaning

**Get cabin crew job, blow up B.A. plane**

A BRITISH Airways worker was given secret orders from a terror chief urging him to get a cabin crew job so he could blow up a US-bound jet, a court heard yesterday

Figure: Usage of Words

# Text generation: Meaning representation

- Sample representation:
  - obligatory(s:suggest(y:doctor(z:patient),d:dose))
  - obligatory(follow(z,s))
  - [unsure-about(z,d) v unsure-about(z,timing(d))] -> obligatory(ask(z,y))
  - procedure(take(z,t:tablet), [remove(z,t,foil,finger,back(t)) & swallow(z,t,water)])
  - [w:take(z,overdose)] -> obligatory([tell(z,y,w) OR visit(z,casualty(hospital(z)))]))
  - store(a:person,m:medicine) -> obligatory(storeawayfrom(a,m,children))

- Nodes are **variables** labelled by **concepts**

- Entities, events, states, properties
- **d** / **dog**: **d** is an instance of **dog**

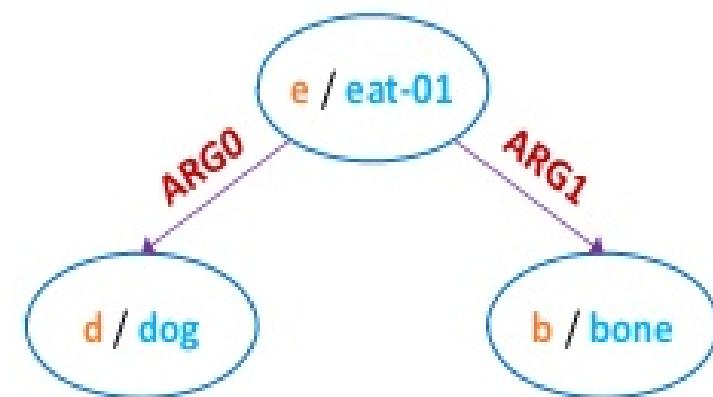
- Edges are semantic **relations**

Figure: Text representation

*The dog is eating bones.*

```
(e / eat-01
  :ARG0 (d / dog)
  :ARG1 (b / bone))
```

Figure: Parameterized representation



**eat.01**: consume (VN-class: **eat-39.1**, FN-frame: **Ingestion**)  
**ARG0-PAG**: consumer, eater (VN-role: **agent**)  
**ARG1-PPT**: meal (VN-role: **patient**)

Figure: Structured representation

# Text generation: Document structure design



IBM ICE (Innovation Centre for Education)

- Natural language generation systems → Semantic rules → Aggregation.
- Left to Right pattern
- Complex sentences → Aggregation.
- Structure → Clauses, Sentences and Paragraphs.

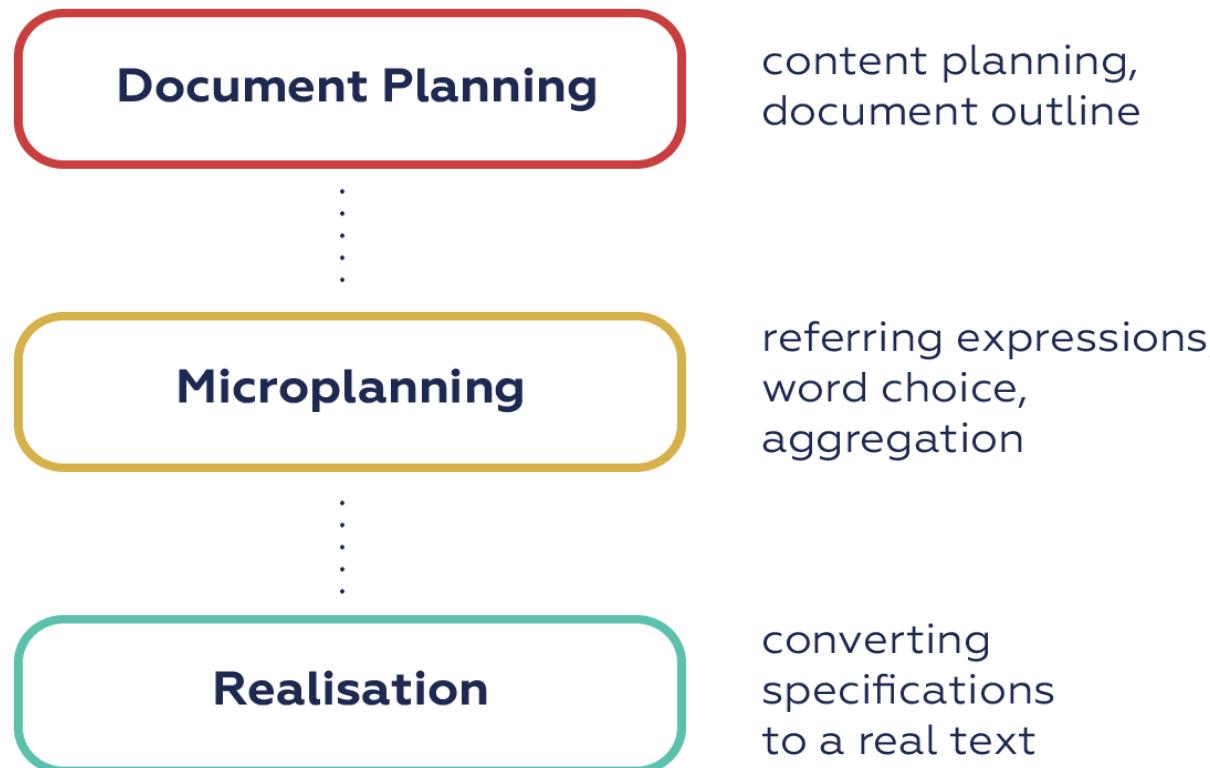


Figure: Text generation: Document structure design

Source: <https://medium.com/sciforce/a-comprehensive-guide-to-natural-language-generation-dd63a4b6e548>

# Text Generation – Document Structure Design

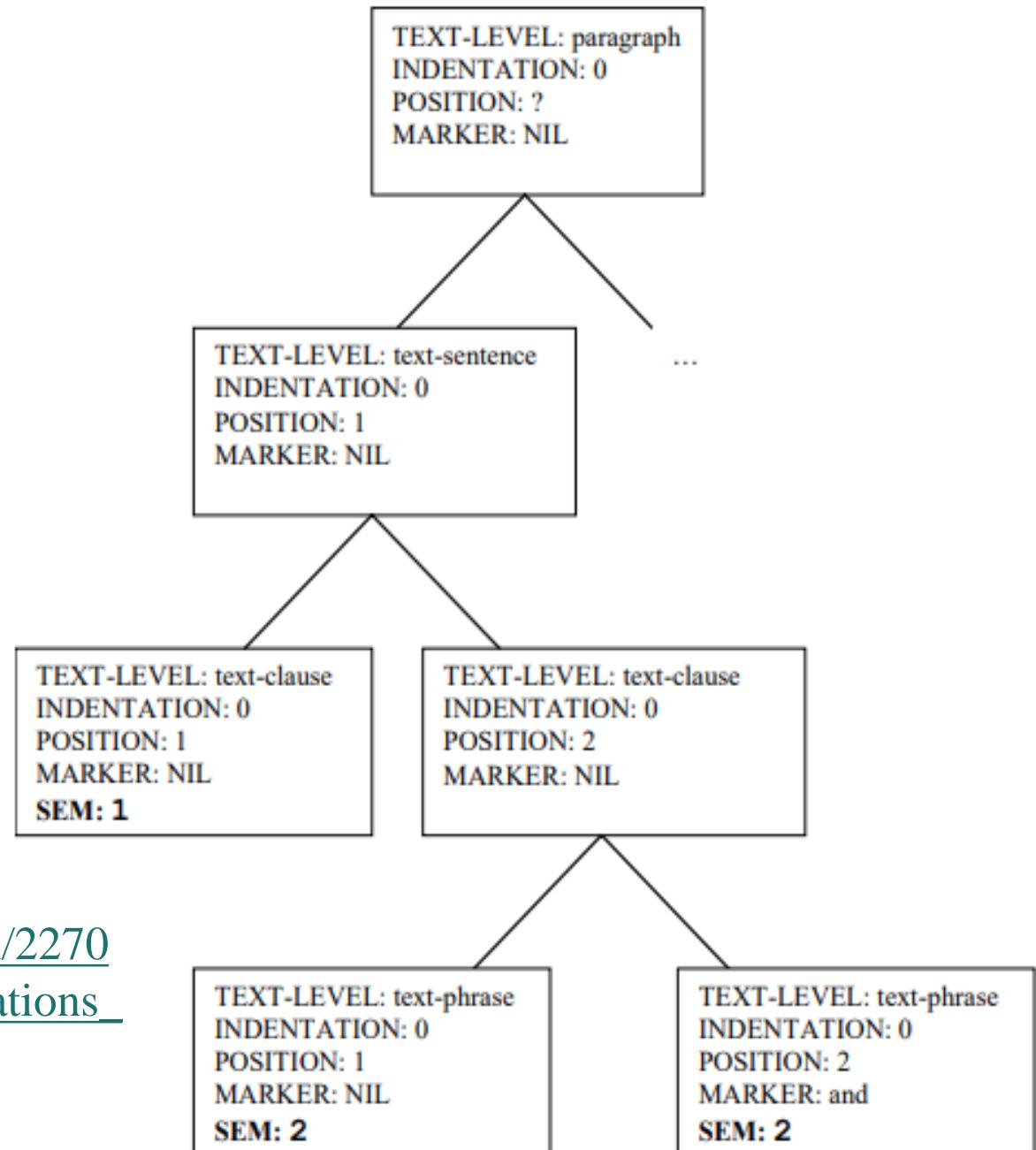
## Sample Formula

1. obligatory(s: suggest(y: doctor (z: patient), d: dose))
2. obligatory (follow (z, s))

## Created Text:

- Your doctor should suggest a dose; follow your doctor's advice and do not change the dose.

Fig: Document Representation



[https://www.researchgate.net/publication/227056814\\_Generating\\_Multimedia\\_Presentations\\_from\\_Plain\\_Text\\_to\\_Screen\\_Play](https://www.researchgate.net/publication/227056814_Generating_Multimedia_Presentations_from_Plain_Text_to_Screen_Play)

# Text generation: Linguistic style control

Example:

Larynx: Throat

Consult: ask

Common usage → Generalized

Individual usage → Personalized

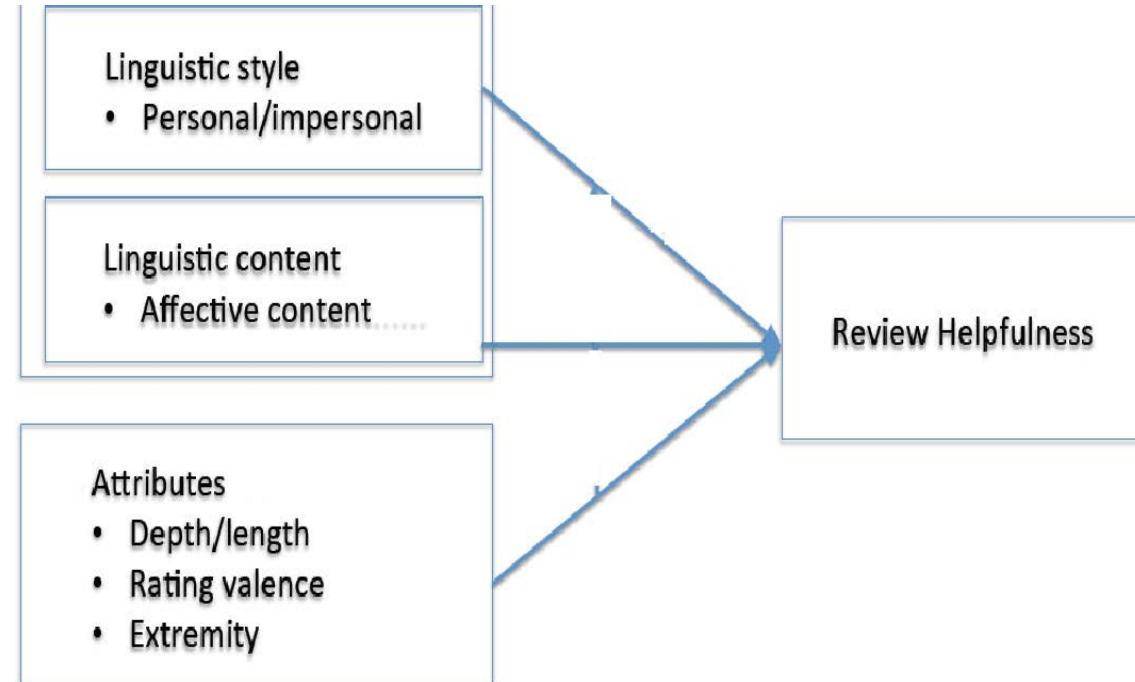


Figure: Linguistic Style Control

Source: <https://www.semanticscholar.org/paper/Linguistic-Style-and-Online-Review-Helpfulness-Wang-Karimi/4bf204a94c6991c56f47f47f0a471c1f0d48d22c>

**Style 1:** Take your tablet by removing it from the foil and swallow it with water.

**Style 2:** The method of delivery for tablets is the removal of the tablet from its foil, followed by its ingestion with water.

**Style 3:** To take a tablet, you should first remove it from the foil and then swallow it with water. Your doctor will tell you the dosage.

# Document Layout



Figure: Document layout formats

Source: <https://www.vectorstock.com/royalty-free-vector/document-report-layout-templates-set-vector-9437687>

# Layout and meaning representation

- Mapped to the abstract document structure.
- Simple words put together → Meaning
  - Uniformity
  - Size
  - Length
  - Navigability
  - Spacing

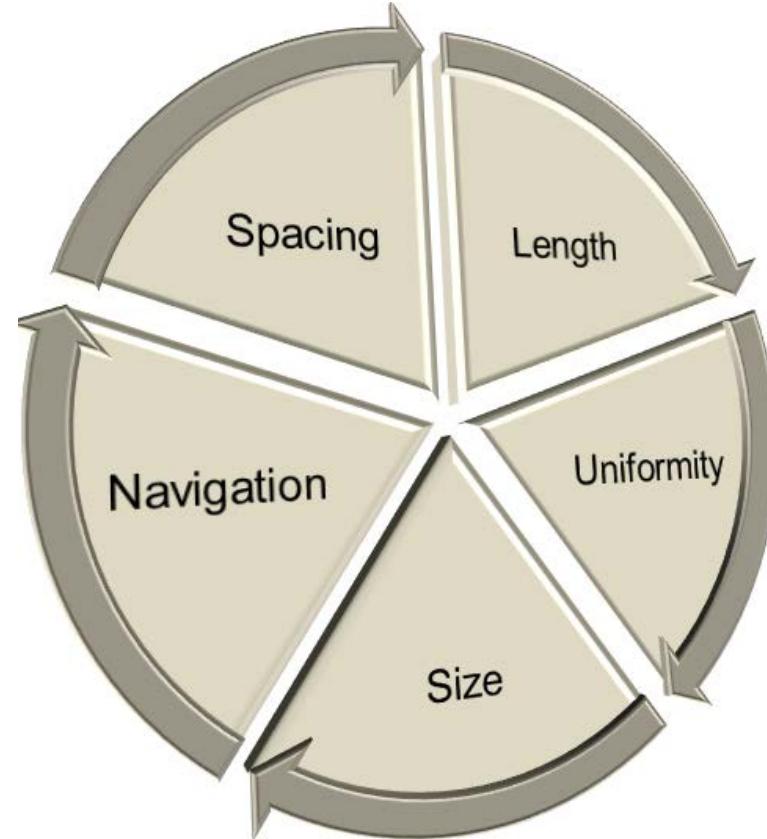


Figure: Layout and meaning factors

Consult your doctor if you have any *problems* concerning your treatment, or any *questions* about your treatment.

Figure: Emphasized elements

# Layout style and wording representation

- Natural language generation systems → Formatting → Syntax
- Layout and Wording interact with one another → Efficient communication.
- Change in Style → Alter the words

Sample Text:

Ask your doctor if:

- you are unsure about the dose
- you are unsure when to take the dose

## Change in style without change in wording

Ask your doctor if: you are unsure about the dose you are unsure when to take the dose.

## Change in style with change in wording

Ask your doctor if you are unsure about the dose or you are unsure when to take the dose.

Figure: Change in Style with Wording

# Image style and meaning representation

Meaning of a picture → Relative and Subjective

Based on conjunction

Library of pictures

Representation of images → Understanding of the context

- Complex piece of information
- Events that should be followed sequentially       $x:\text{person} \ \& \ y:\text{medicine} \ \& \ \text{getmedicine}(x,y)$
- Continuous quantities       $x:\text{person} \ \& \ y:\text{tablet} \ \& \ \text{remove}(x,y,\text{foil},\text{finger},\text{back}(y))$
- Presentation of serial element

Figure: Picture as Representation



Figure: Image representation

# Image and wording usage

- Images become an important part of the document.
- Enhance the power of the document
- References to the illustrative images , Reduce the number of words
- References → Provide connectivity

- **Example:**

'the red vial depicted in Fig.7',  
'the picture of the kit in Fig.5',

Images → Reduction  
of the number of words

- **Sample sentence:**



Take your tablet by removing it from the foil by pressing your against the back of the tablet  
Take your tablet by removing it from the foil

Figure: Sample Representation

# Scripted dialogue

Documents → Objects, Dialogues → Events.

Narrative and Argumentative representation.

Message → "m" Signal → "s".

## Plain text representation

Although the patient asked when he should take the medicine, the pharmacist could only reply to him that his doctor would be able to tell him.

## Dialogue representation

Patient: When should I take the medicine?

Pharmacist: Your doctor will be able to tell you.

Figure: Dialogue based representation

# Scripted dialogue

- Dialogue planner:
- Multimodal generator
- Speech synthesis
- Gesture Assignment
- Media player

Pharmacist: Here is your medicine.  
Store it away from children.  
Your doctor should suggest a dose.

Patient: Can I change the dose?

Pharmacist: No, you should follow your doctor's advice.

Patient: When should I take the medicine?

Pharmacist: Ask your doctor.

Figure: Dialogue instead of plain text

# Language interfaces for intelligent tutoring systems



IBM ICE (Innovation Centre for Education)

Intelligent tutoring systems → Lessons without intervention from human

Goal → Learn and Teach

- User interface
- Pedagogical model
- Domain knowledge model
- Student model

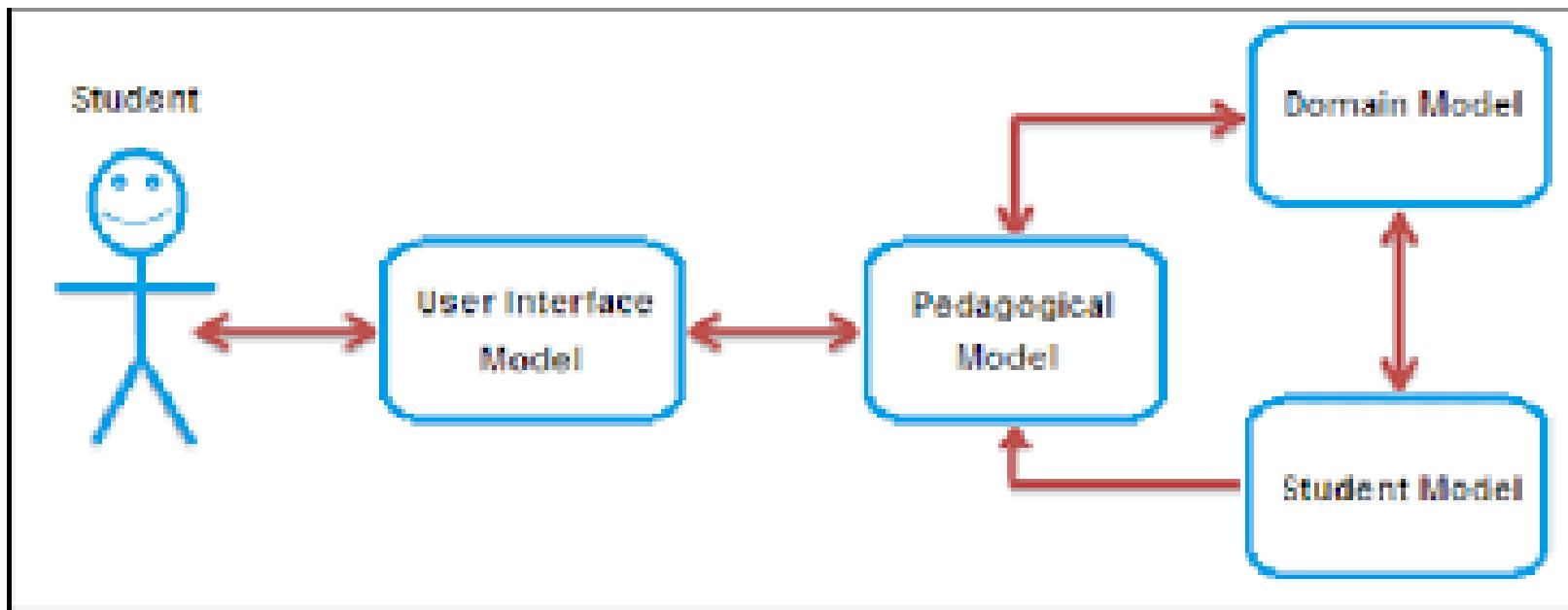


Figure: A simple Tutoring system

Source: [https://www.researchgate.net/figure/Typical-architecture-of-the-Intelligent-Tutoring-System\\_fig1\\_314229691](https://www.researchgate.net/figure/Typical-architecture-of-the-Intelligent-Tutoring-System_fig1_314229691)

# CIRCSIM-Tutor

CIRCSIM-Tutor → Cardiovascular physiology.

Qualitative model → Physiological parameters.

- Central venous pressure
- Stroke volume
- Cardiac Output
- Heart rate
- Inotropic state
- Total peripheral resistance

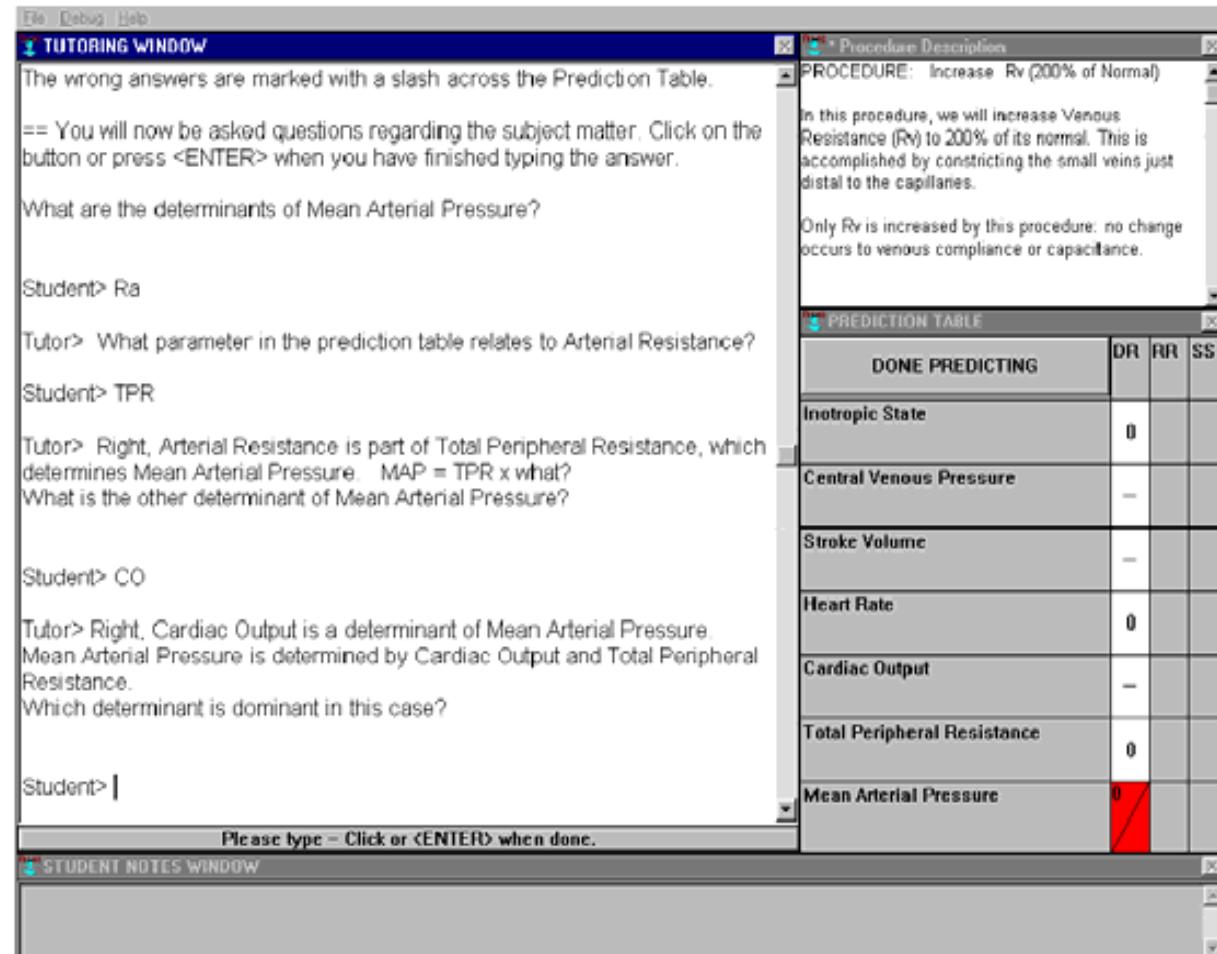


Figure: Screen Interface of CIRCSIM-Tutor

Source: [https://www.researchgate.net/figure/CIRCSIM-Tutor-screen-containing-a-fragment-of-dialogue-from-Session-39-CIRCSIM-Tutor\\_fig1\\_228342817](https://www.researchgate.net/figure/CIRCSIM-Tutor-screen-containing-a-fragment-of-dialogue-from-Session-39-CIRCSIM-Tutor_fig1_228342817)

# CIRCSIM-Tutor architecture, data presentation and process cycle (1 of 3)



IBM ICE (Innovation Centre for Education)

- Student model
- Input understander
- Knowledge base
- Problem solver
- Screen manager
- Text generator

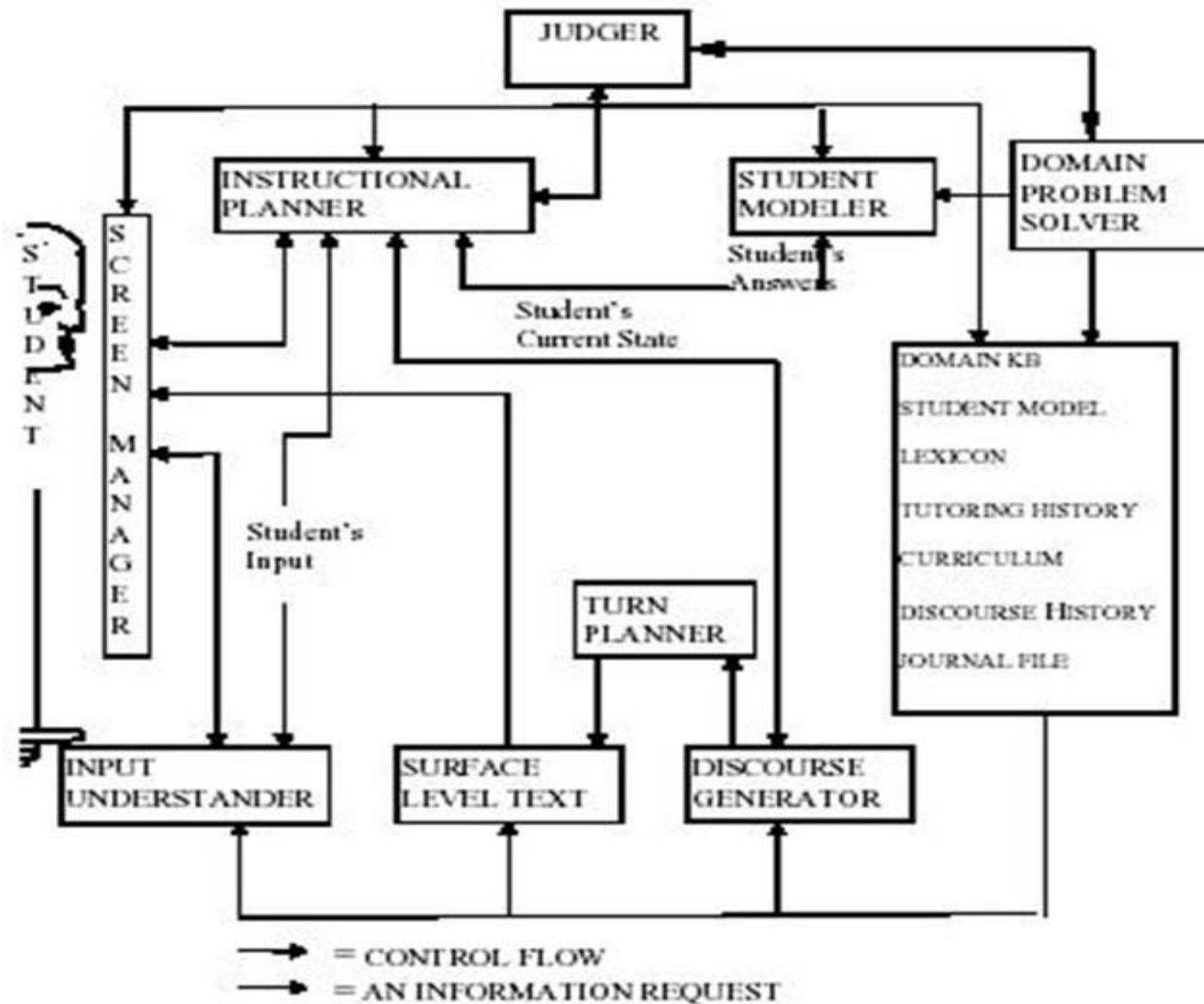


Figure: CIRCSIM-Tutor Architecture

Source: <https://slideplayer.com/slide/3082681/>

# CIRCSIM-Tutor architecture, data presentation and process cycle (2 of 3)



IBM ICE (Innovation Centre for Education)

## CIRCSIM-Tutor Process

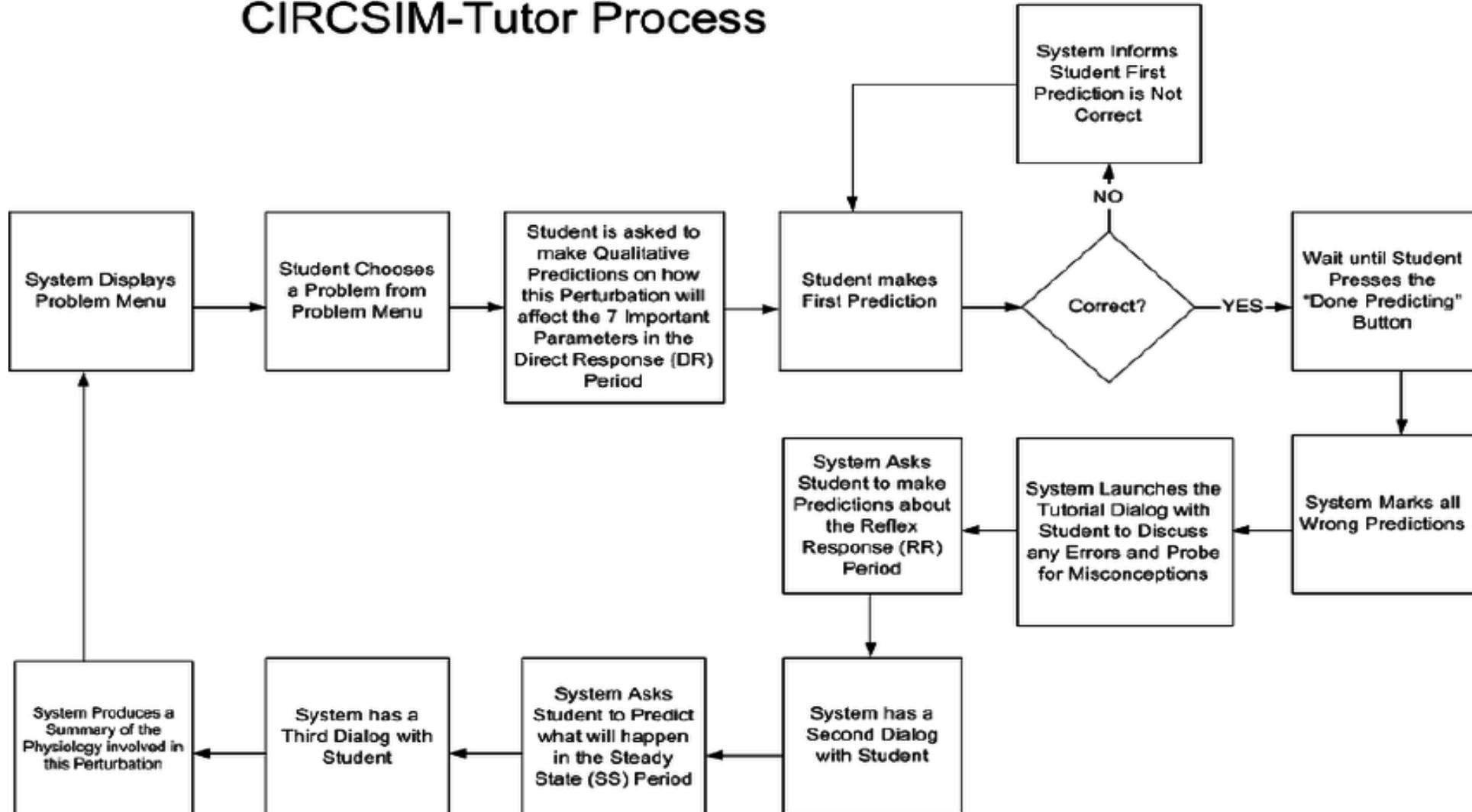


Figure: Process steps

Source: [https://www.researchgate.net/figure/The-Tutoring-Process-in-CIRCSIM-Tutor\\_fig4\\_228797088](https://www.researchgate.net/figure/The-Tutoring-Process-in-CIRCSIM-Tutor_fig4_228797088)

# CIRCSIM-Tutor architecture, data presentation and process cycle (3 of 3)



IBM ICE (Innovation Centre for Education)

T: What is the determinant of Central Venous Pressure?

S: cbv

T: And what parameter in the predictions table determines Central blood Volume?

S: cardiac output

T: Right, CO determines CBV, therefore determines CVP.

T: By what mechanism is Total Peripheral Resistance controlled?

S: Metabolic factors

T: Is the mechanism of control neural or physical?

S: Neura

Figure: Sample Interaction

# AUTOTUTOR

Auto tutor → Science, Technology

Animated conversational agent, dialogue management, electronic documents

Topmost area → Problem

Left area → Conversational agent.

Right area → Auxiliary diagrams, Student interactions.

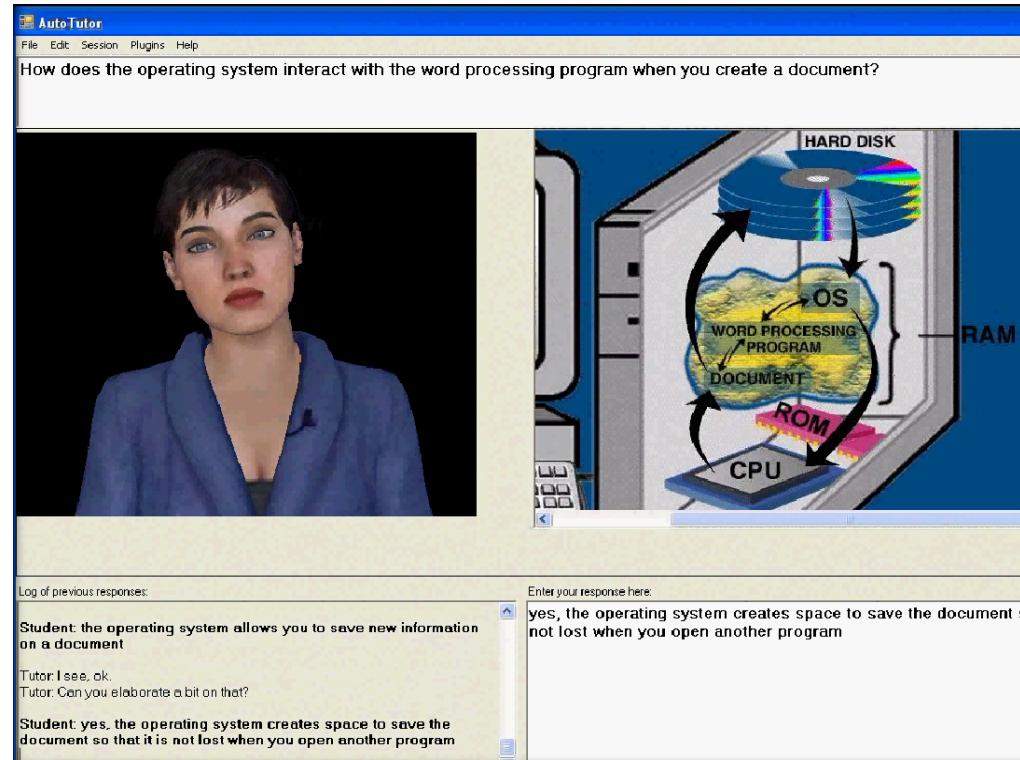


Figure: User interface

Source: <https://www.semanticscholar.org/paper/AutoTutor-and-affective-autotutor%3A-Learning-by-with-D%27Mello-Graesser/a884f00fe18a6abf837b2ccb490165ded90fc29a>

# AUTOTUTOR architecture and process

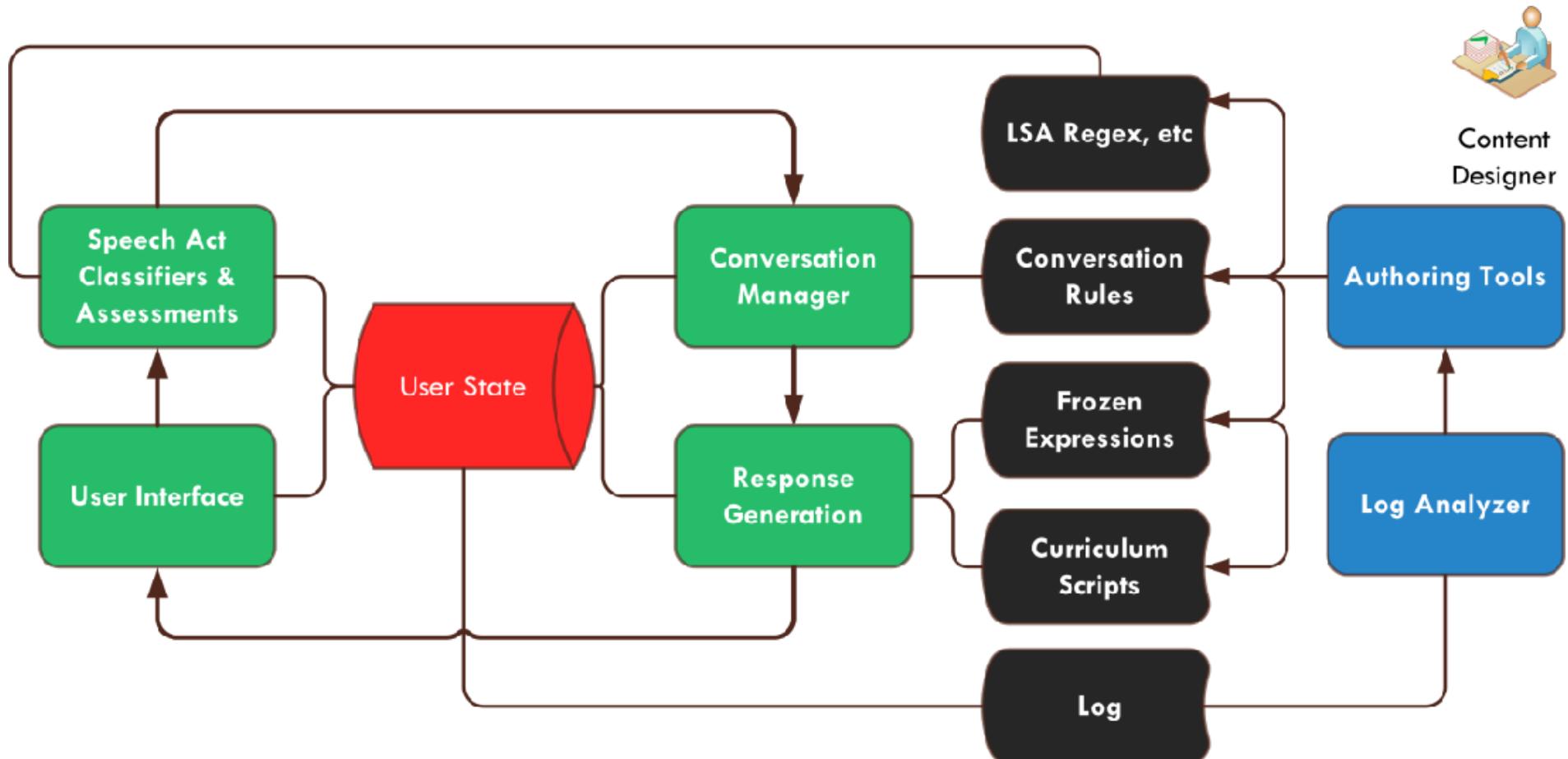


Figure: AUTOTUTOR architecture

Source: <https://www.semanticscholar.org/paper/AutoTutor-and-affective-autotutor%3A-Learning-by-with-D'Mello-Graesser/a884f00fe18a6abf837b2ccb490165ded90fc29a/figure/1>

# WHY2-ATLAS

- Why 2 Atlas → Teaches qualitative physics.
- First window → Problem statements
- Second window → Space for the user
- Third → Chat window

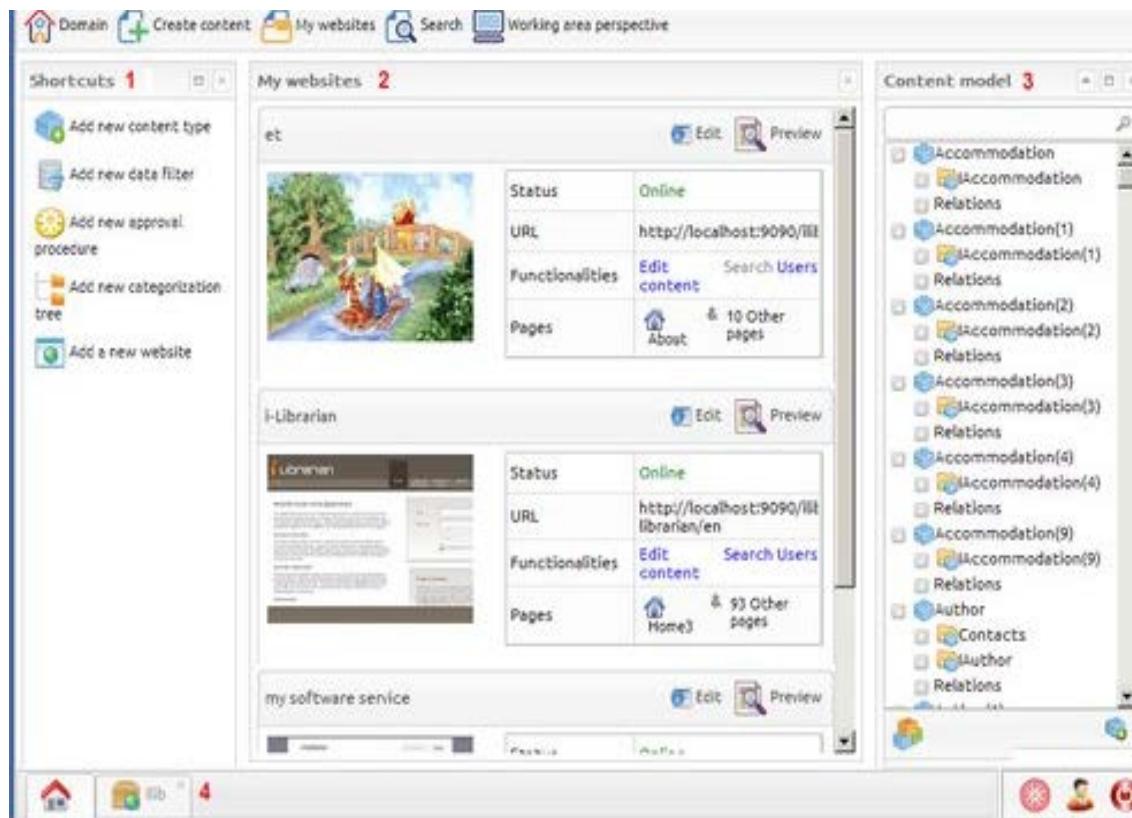


Figure: Atlas User Interface

Source: <http://i-publisher.atlasproject.eu/atlas/documentation/advanced/interface>

# Why 2 Atlas architecture and process

- Sentence level understander → Parsing → Set of prepositions
- Discourse level understander → Assimilates all the prepositions into appropriate explanation.
- Tutorial strategist → Analyses the sentences and identifies the completeness and correctness
- Knowledge construction dialogue → Describing the concept

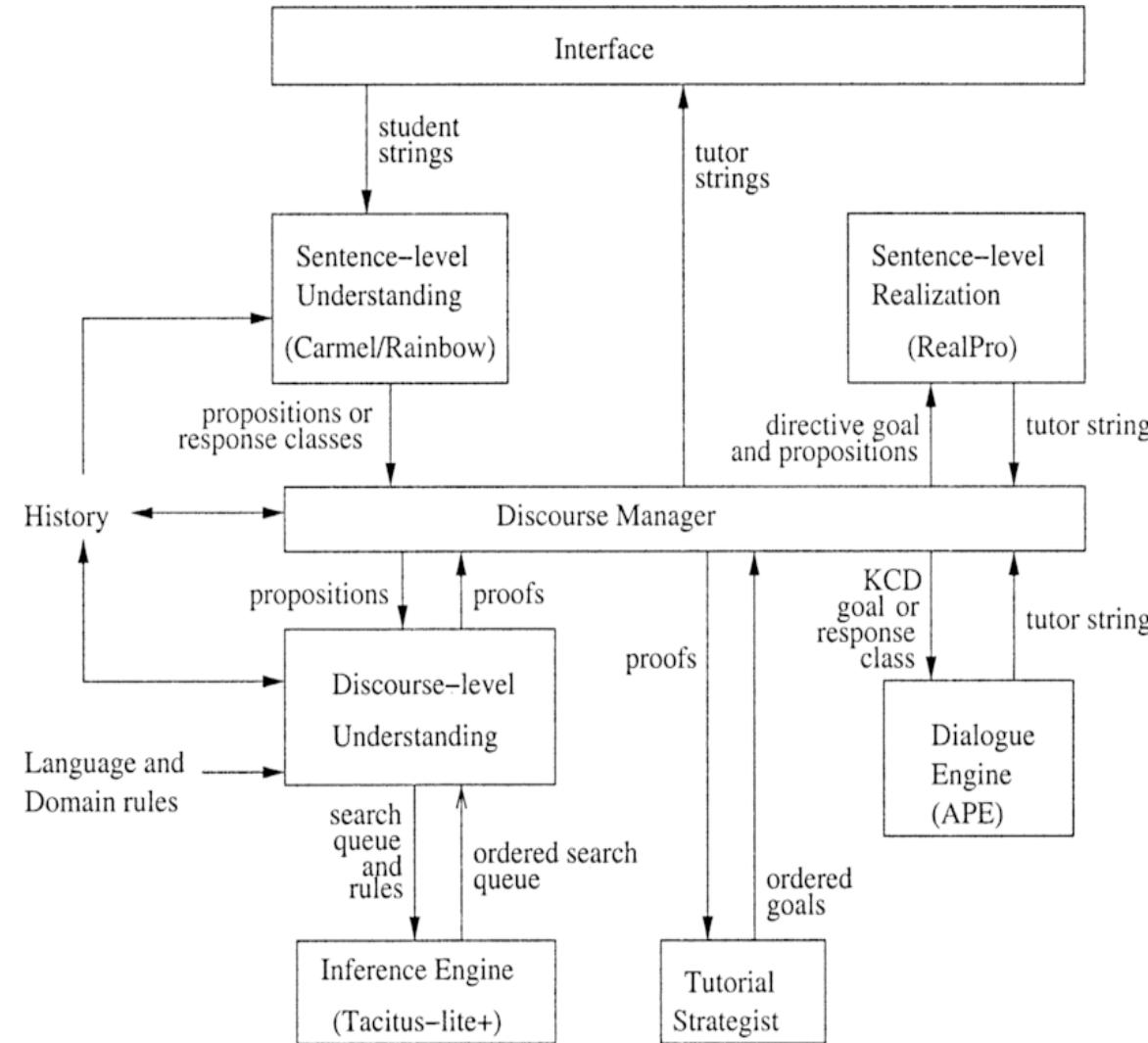


Figure: Architecture

# Argumentation for healthcare consumers (1 of 2)



IBM ICE (Innovation Centre for Education)

Clinical Decision Support systems (CDS) → Automating health related information.

Goal → Clinical decision with knowledge and accuracy.

Patient data Integration Centre

- Decision rules
- Knowledge base
- Assess
- Generate recommendation.

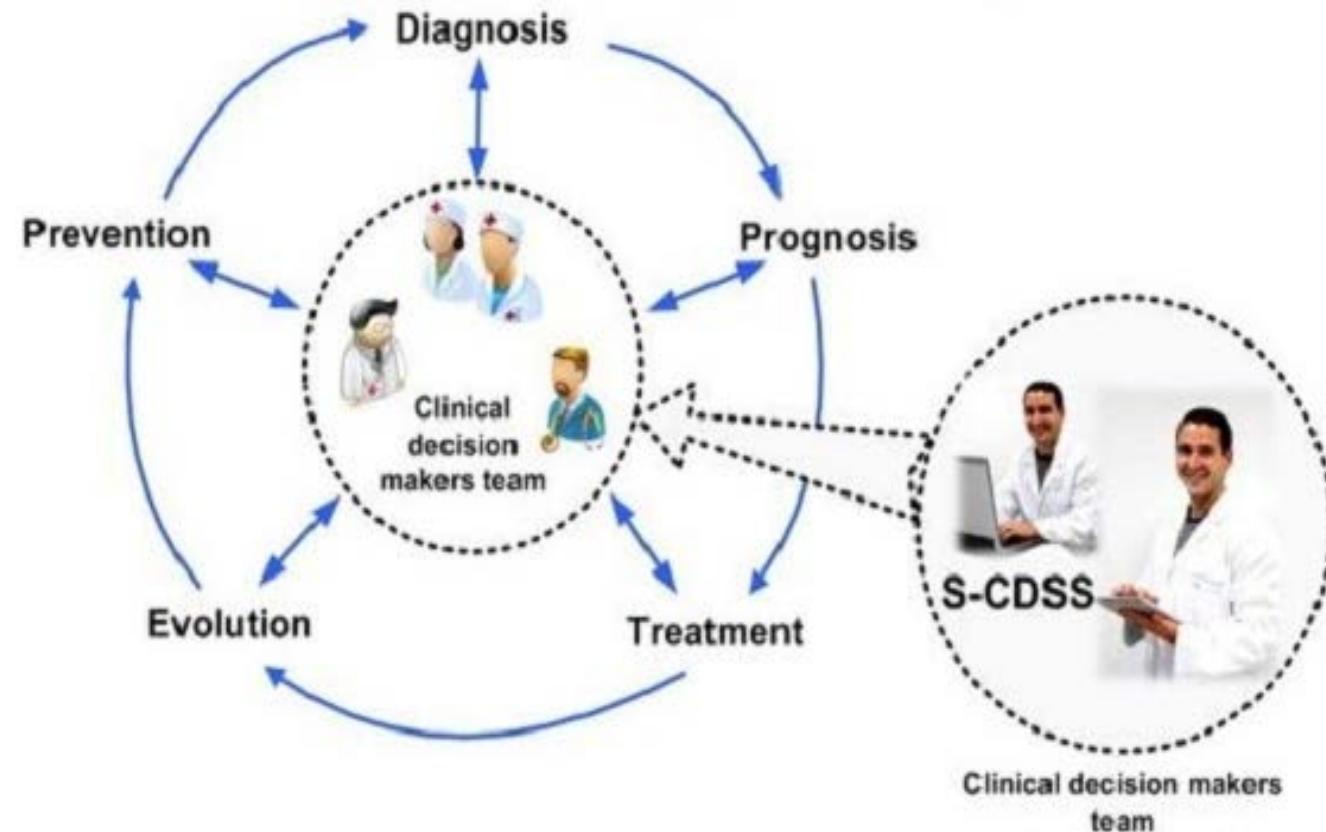


Figure: CDS Outline

Source: <https://www.kenresearch.com/blog/2019/11/global-clinical-decision-support-systems-market/>

# Argumentation for healthcare consumers (2 of 2)

IBM

IBM ICE (Innovation Centre for Education)

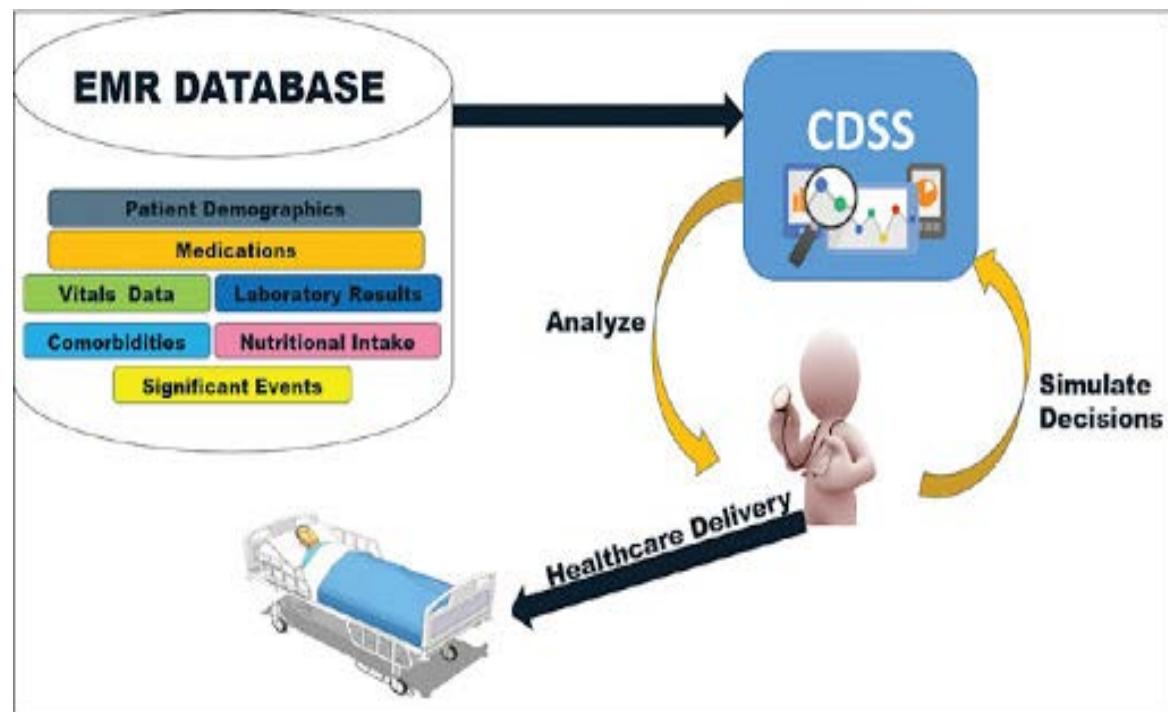
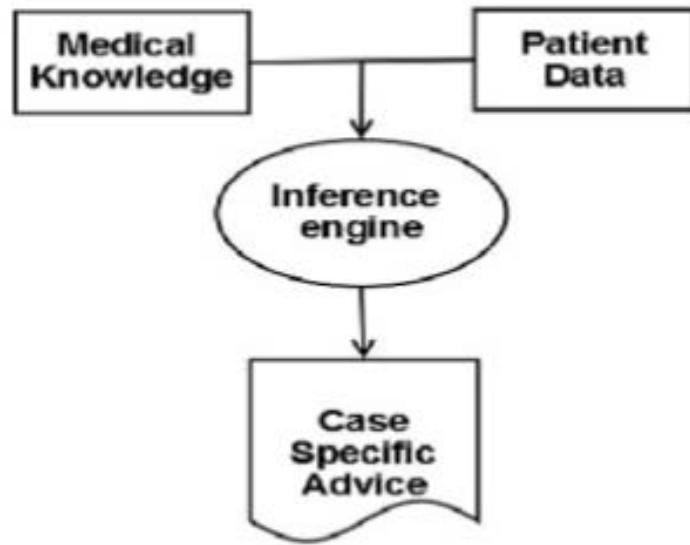


Figure: CDS Components

Source: <https://www.medgadget.com/2018/07/clinical-decision-support-systems-cdss-market-prepare-to-touch-at-a-cagr-of-11-5-by-2023.html>

Figure: CDS – EHR/EMR Interaction

Source: <http://www.ijam-web.org/article.asp?issn=2455-5568;year=2017;volume=3;issue=1;spage=78;epage=83;aulast=Pappada>

# CDS architecture and processing

## Major activities:

- Alerting
- Monitoring
- Coding
- Reminders

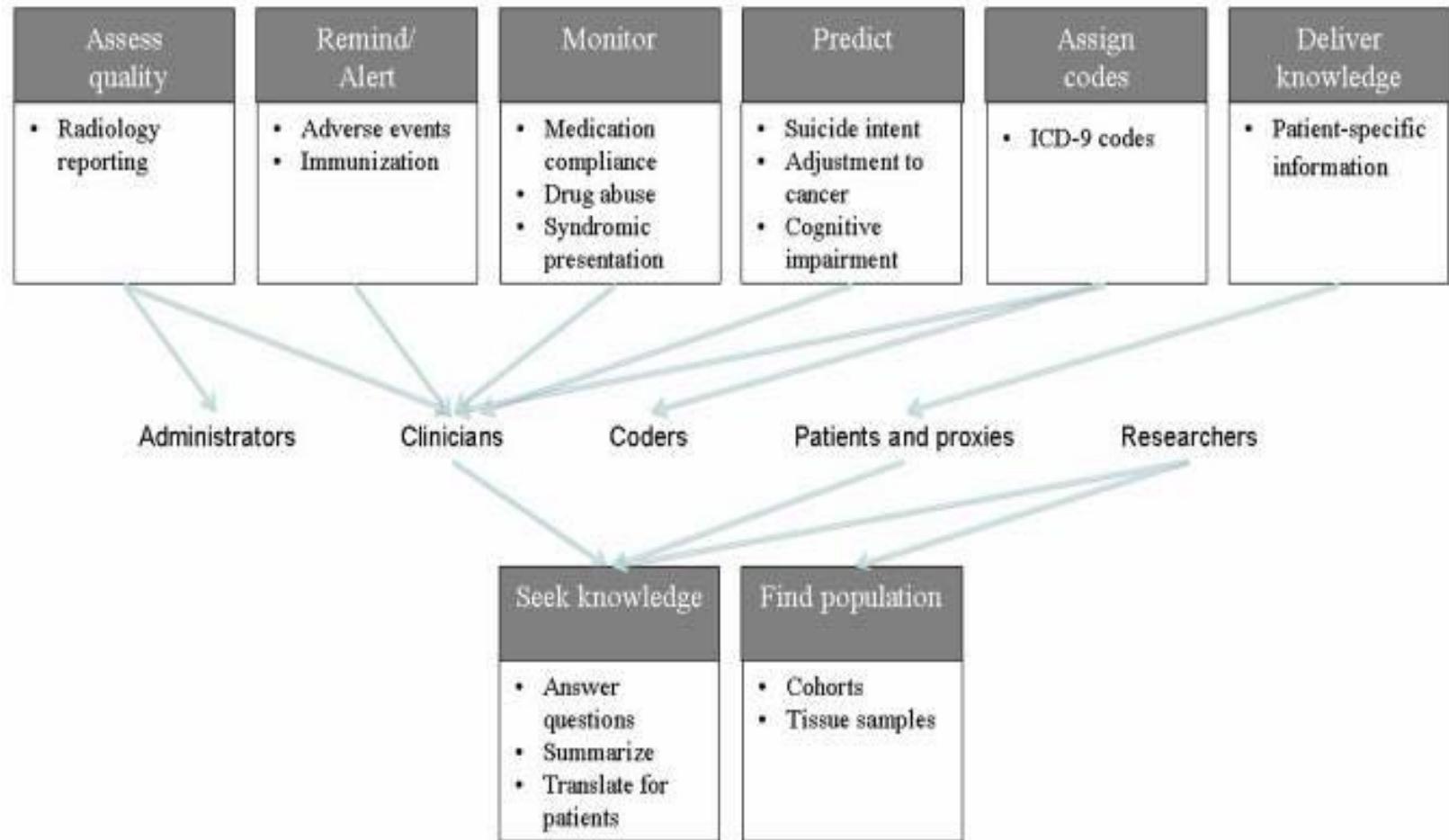


Figure: NLP in CDS

Source:

[https://www.ncbi.nlm.nih.gov/core/lw/2.0/html/tileshop\\_pmc/tileshop\\_pmc\\_inline.html?title=Click%20on%20image%20to%20zoom&p=PMC3&id=2757540\\_nihms145183f1.jpg](https://www.ncbi.nlm.nih.gov/core/lw/2.0/html/tileshop_pmc/tileshop_pmc_inline.html?title=Click%20on%20image%20to%20zoom&p=PMC3&id=2757540_nihms145183f1.jpg)

# NLP for CDS Scope

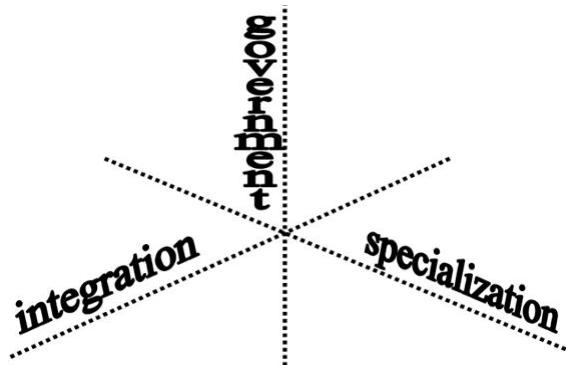


Figure: NLP-CDS Trifecta

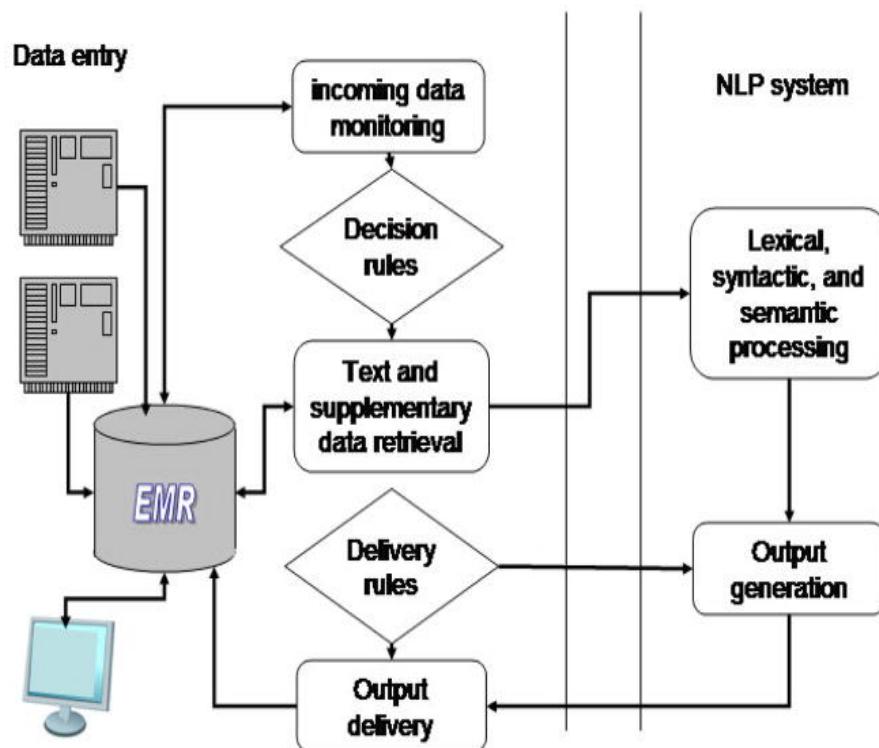


Figure: Specific NLP-CDS

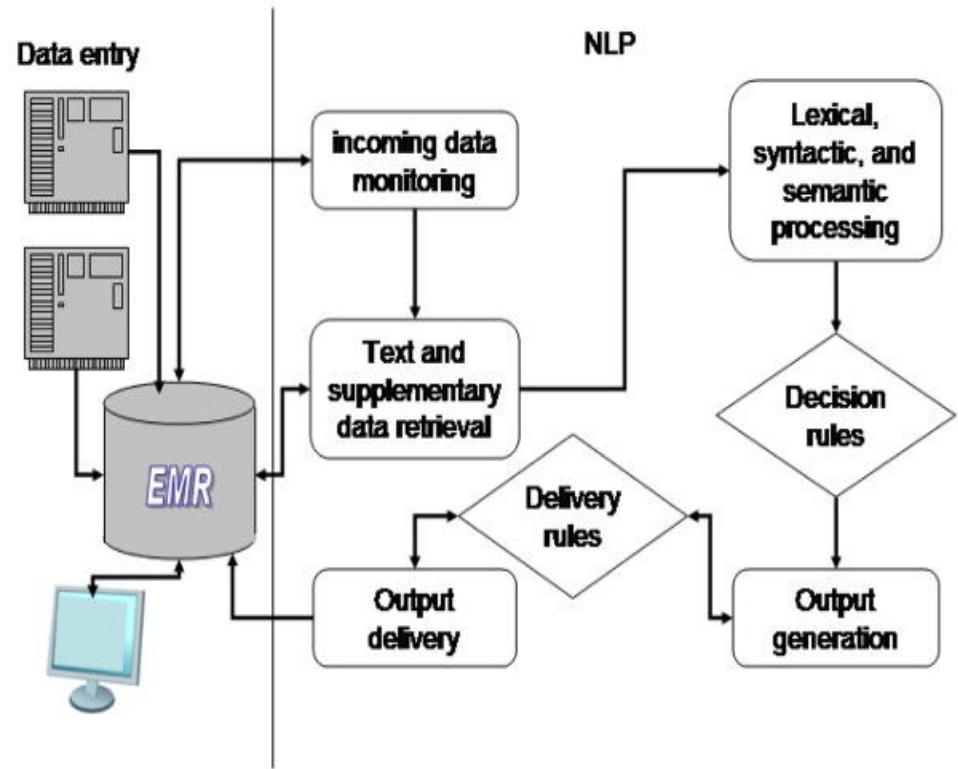


Figure: Generic NLP-CDS

Source: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2757540/>

# NLP models

- **Generic model**

Modules → already customized, Predefined workflow structure.

- **Specialized model**

Specific task → Controls information flow

- **Coupled Model**

Invoked by HER, Identify the presence of a disease, Create corresponding codes.

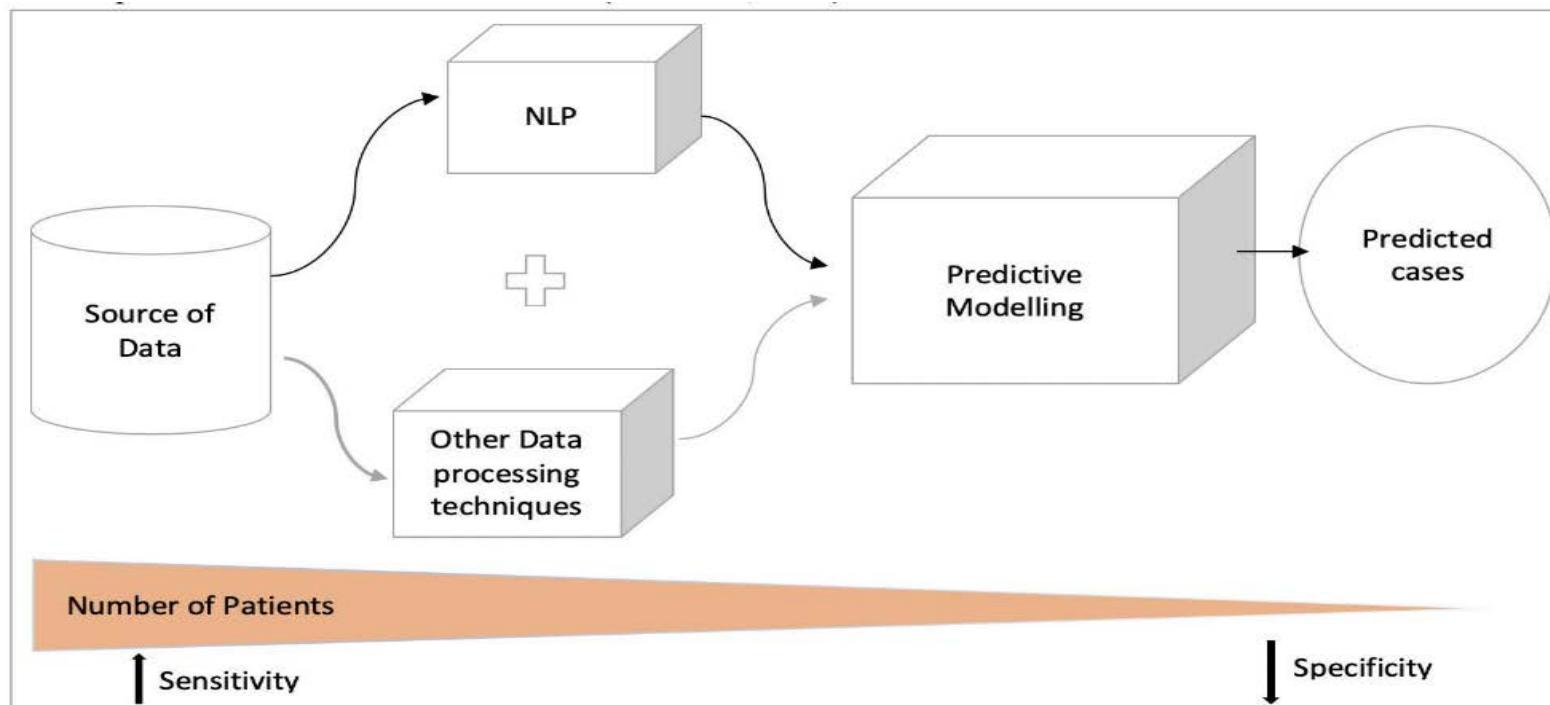


Figure: Generalization vs Specification

Source: <https://ukdiss.com/examples/natural-language-processing.php>

# Building blocks of NLP - CDS

## Text preprocessing:

- Tokenization, part of speech tagging, parsing
- Specific tags → Biomedical taggers
- MedPost taggers.

## Named entity recognition

- Dictionary based named entity recognition
- Statistical named entity recognition

## Context extraction

- Negation of a symptom
- Historicity of symptoms
- Experience levels

## Relations and Associations

- Relations between entities and events
- Propositional representation → Indicator and Evidence.

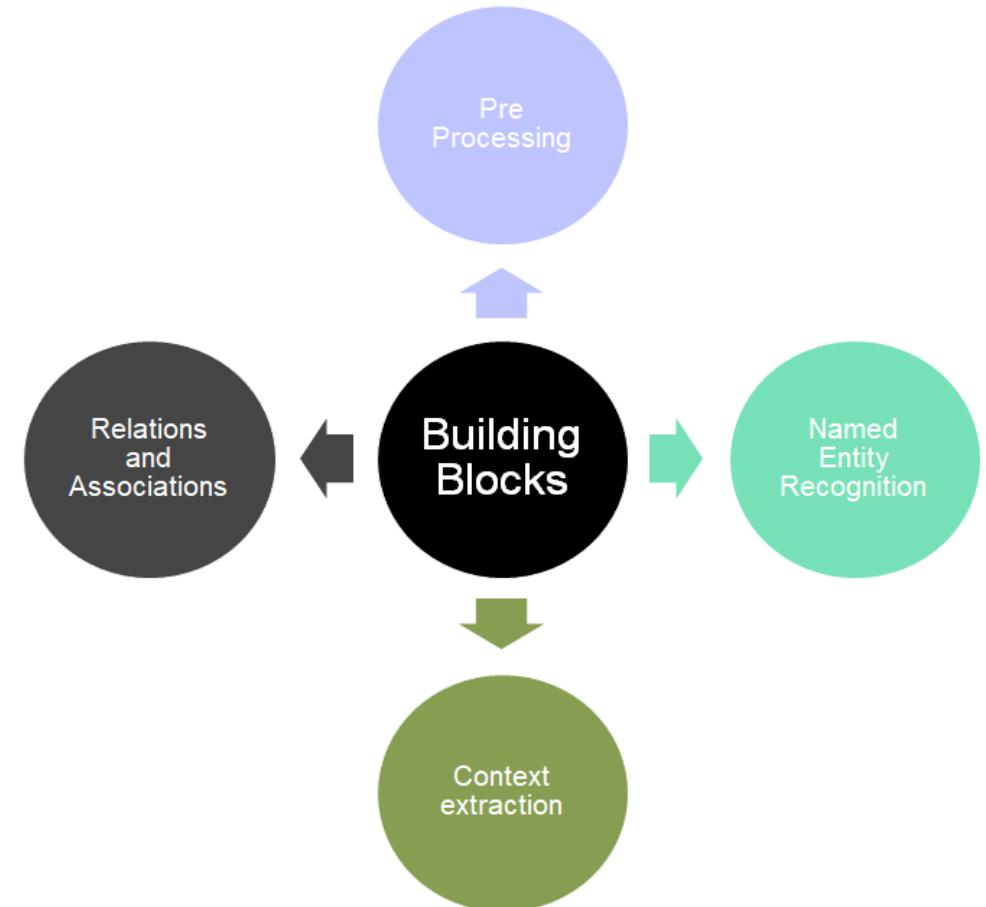


Figure: Building Blocks

# Data based evidence collection: Summarization



IBM ICE (Innovation Centre for Education)

Study	Design	Sample	Cognitive measures	Summary of results
Hajjar et al. (2009) <sup>54</sup> USA	Cross-sectional	N=580 community-dwelling elders (10% in low performance group had HF) Mean age 77.8 HF diagnosis: self-report	HVLT – Revised TMT-A TMT-B <i>Scores adjusted for age, education and race</i>	A group of elderly individuals with a phenotype of slow gait speed, greater depressive symptoms and worse executive function exists Elevated LDL is associated with being a member of this group (OR 1.01, 95% CI 1.00–1.02)
Kerola et al. (2010) <sup>64</sup> Level of evidence: IV Finland	Longitudinal descriptive cohort study	N=303 community-dwelling elders (Highest BNP group – 43% HF) Mean age 78.6–81.3 HF diagnosis: HF diagnosis using medical record	MMSE <i>Effect of age, sex, and education included in analysis</i>	Low HDL associated with lower MMSE at baseline ( $\beta=0.174$ , $p=0.001$ )
Zuccalà et al. (1997) <sup>32</sup> Level of evidence: VI Italy	Cross-sectional descriptive	N=57 HF patients Mean age 77 HF diagnosis: systolic HF using echocardiogram and clinical criteria	MMSE <i>Effect of age and sex included in analysis</i>	Lower serum cholesterol associated with lower MMSE scores ( $r=0.30$ , $p=0.02$ )

BNP: B natriuretic peptide; CI: confidence interval; HF: heart failure; LDL: low-density lipoprotein; MMSE: Mini Mental State Exam; OR: odds ratio; HVLT: Hopkins Verbal Learning Test; TMT: Trail Making Test

Figure: Sample summary

Source: [https://www.researchgate.net/figure/Cognition-and-lipid-levels\\_tb1\\_234099424](https://www.researchgate.net/figure/Cognition-and-lipid-levels_tb1_234099424)

# Applications of NLP in Healthcare



Figure: Application Areas

Source: <https://marutitech.com/nlp-in-healthcare/>

# Sentiment analysis and subjectivity

- Textual information:
  - Facts → Objective expressions about entities
  - Opinions → Subjective expressions



Figure: Sentiment Analysis

# Difficulties in sentiment analysis

Sentiment analysis → Study of opinions, sentiments and emotions

Positive opinion & Negative opinions

Object o → Entity → o: (T, A)

Opinion → Feature f of an object O → positive or negative opinion on f

Feature f → Explicit feature → neither f nor any of its synonyms appear in s

Implicit feature → f is implied

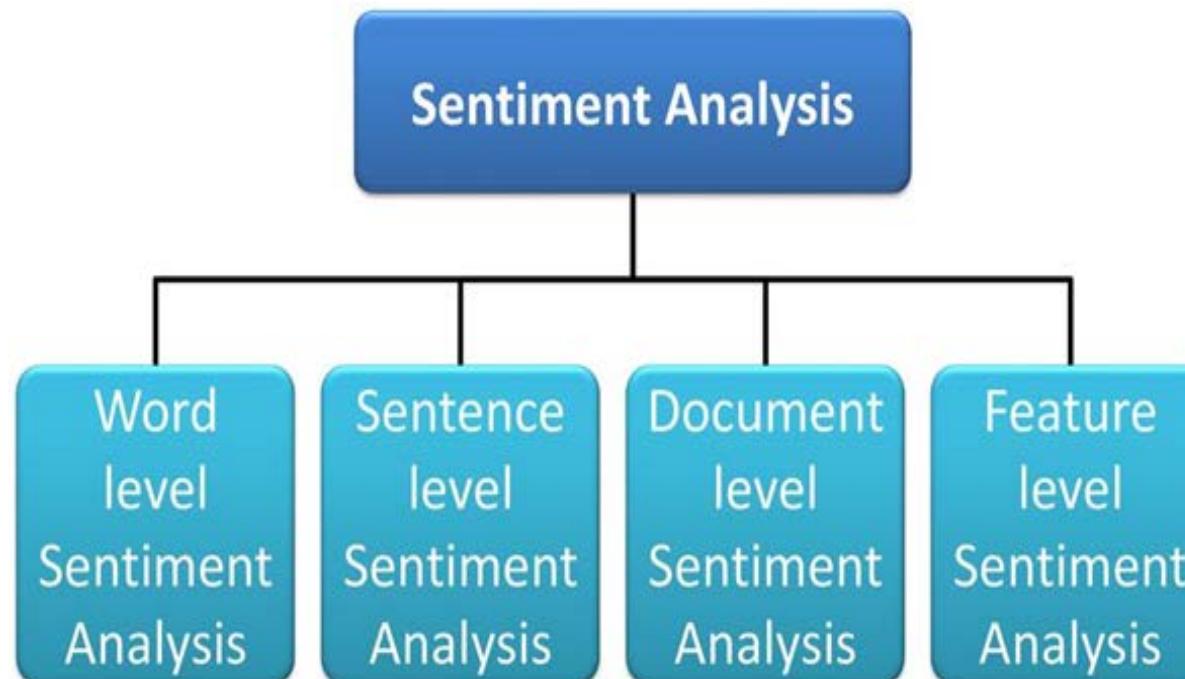


Figure: Levels of Analysis

# Difficulties in Sentiment Analysis

- Direct opinion
- Comparative opinion

Object : PHONE:

Positive: 125

Negative: 7

Feature: voice quality

Positive: 120

Negative: 8

Feature: size

Positive: 80

Negative: 12

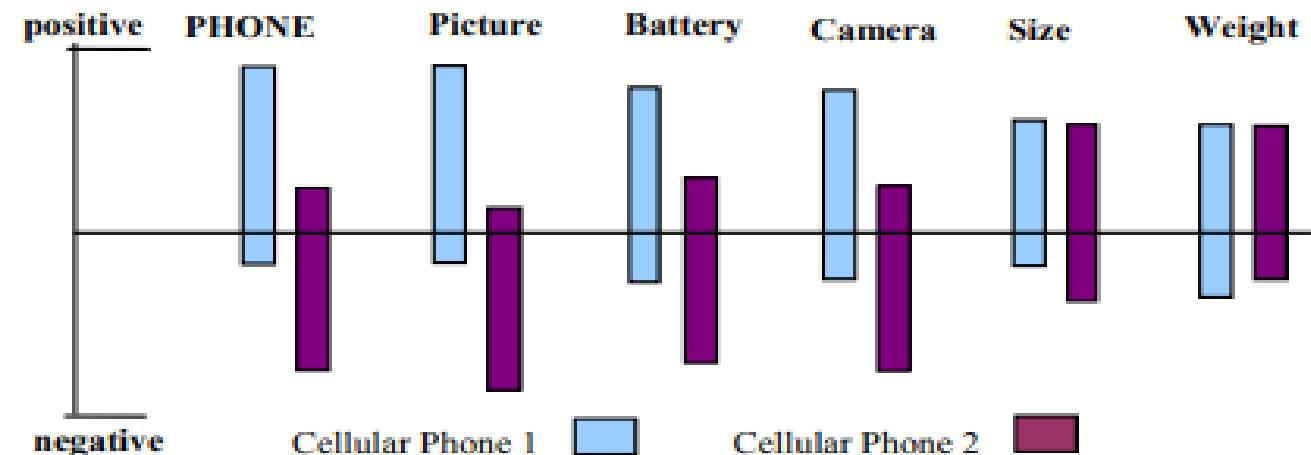
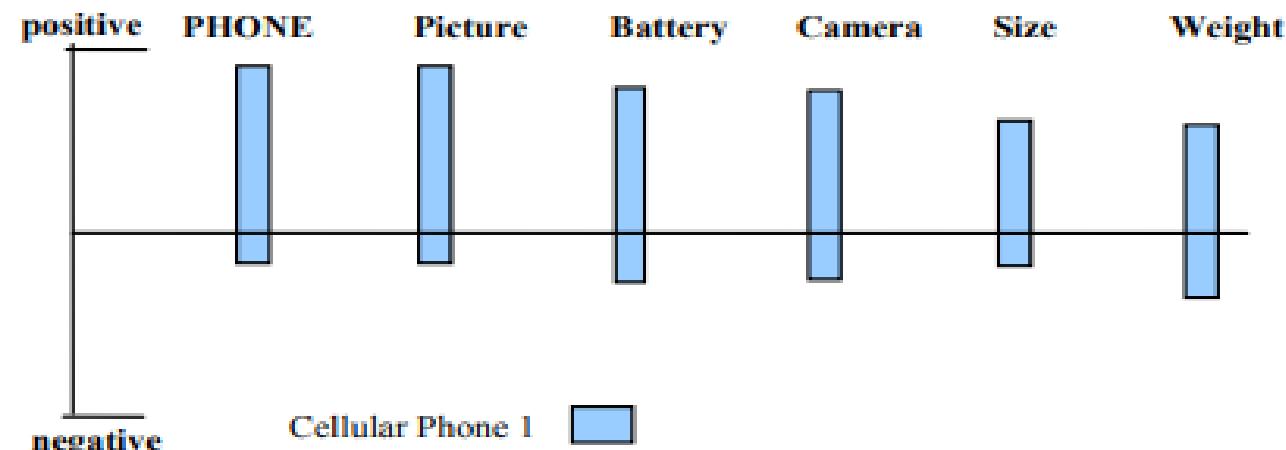


Figure: Opinion based Analysis

# Document level sentiment classification

First word	Second word	Third word (Not Extracted)
1. JJ	NN or NNS	anything
2. RB, RBR, or RBS	JJ	not NN nor NNS
3. JJ	JJ	not NN nor NNS
4. NN or NNS	JJ	not NN nor NNS
5. RB, RBR, or RBS	VB, VBD, VBN, or VBG	anything

Figure: Tagged Expression Sample

$$PMI(term_1, term_2) = \log_2 \left( \frac{\Pr(term_1 \wedge term_2)}{\Pr(term_1) \Pr(term_2)} \right).$$

Figure: Input Term

Source: <https://www.cs.uic.edu/~liub/FBS/NLP-handbook-sentiment-analysis.pdf>

$$oo(\text{phrase}) = PMI(\text{phrase}, "excellent") - PMI(\text{phrase}, "poor").$$

Figure: Output Term

Source: <https://www.semanticscholar.org/paper/Document-Level-Sentiment-Classification-Based-on-Zhang-Miao/3f9042aaefb517b7bf8b8f44b9db92d6d314a252/figure/3>

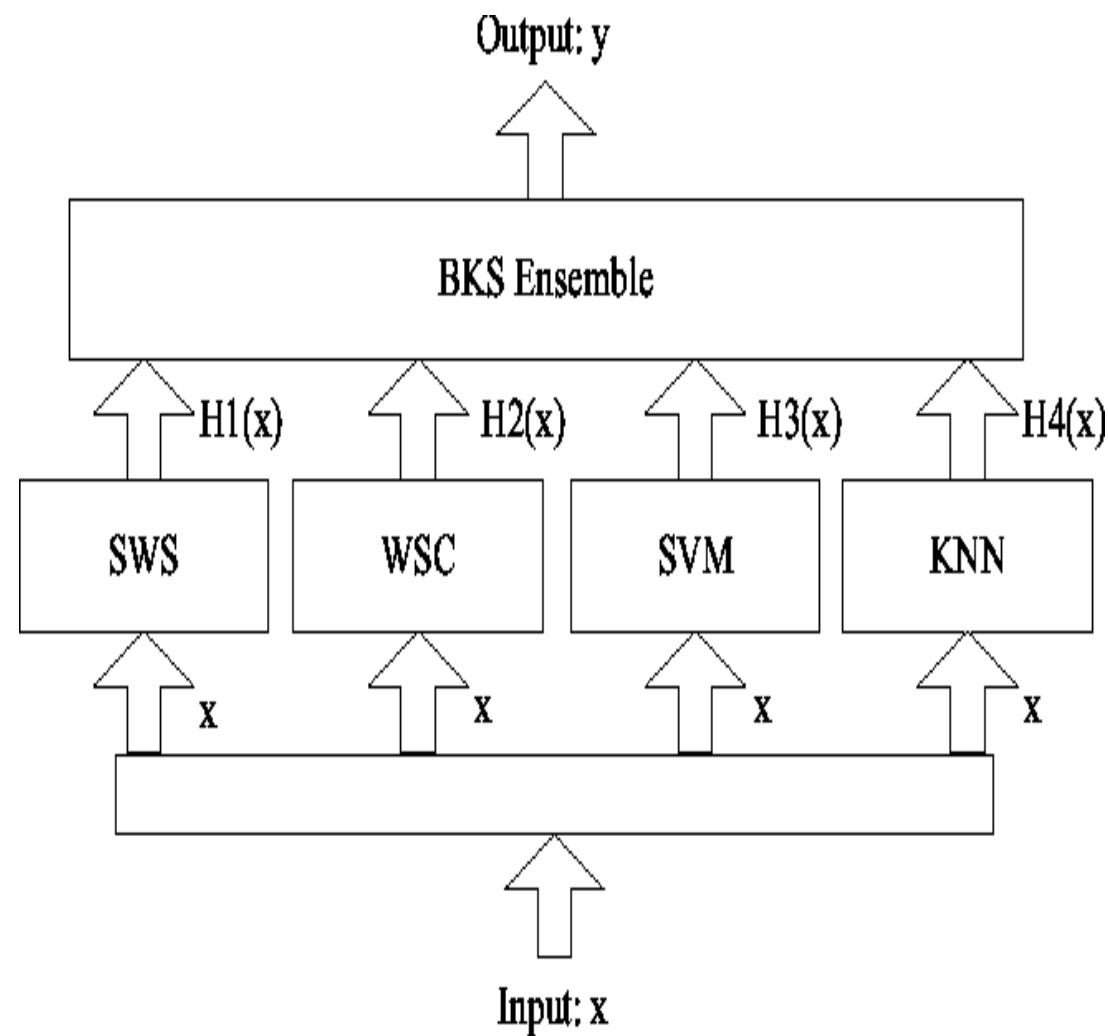
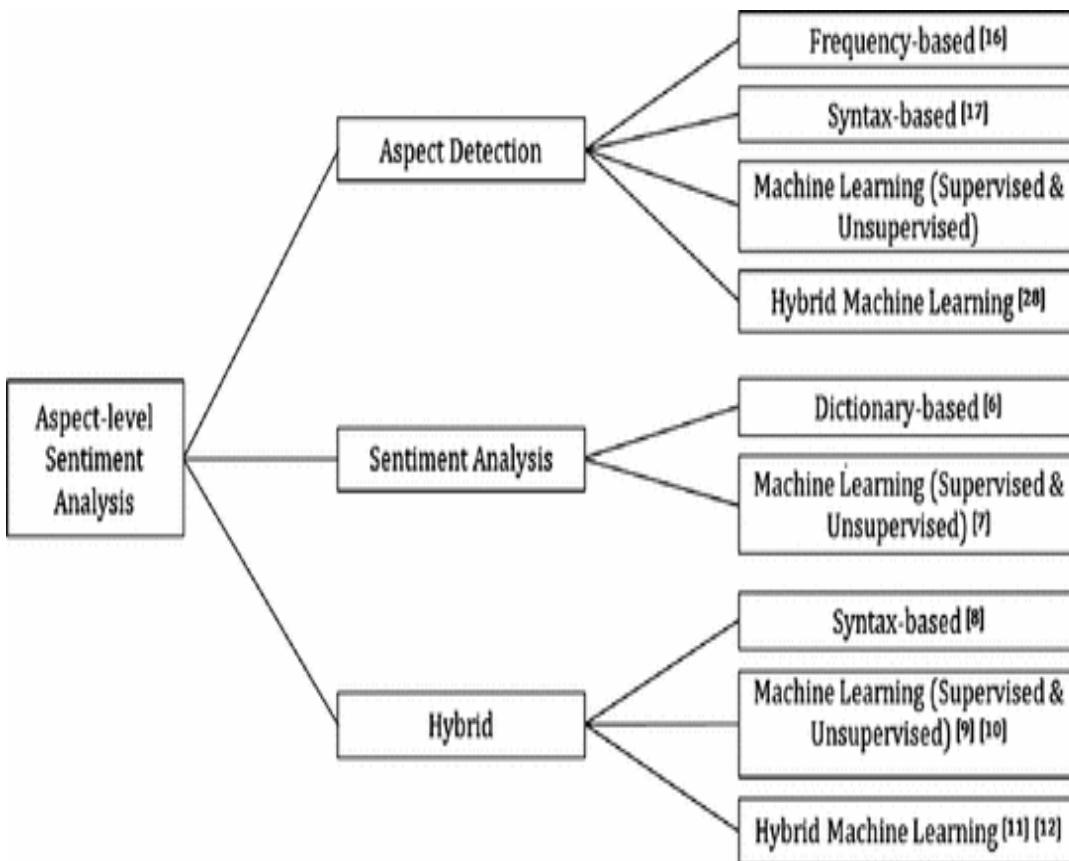


Figure: Doc level Representation

# Sentence level sentiment classification



## Syntactic template

<subj> passive-verb

<subj> active-verb

active-verb <dobj>

noun aux <dobj>

passive-verb prep <np>

## Example pattern

<subj> was satisfied

<subj> complained

endorsed <dobj>

fact is <dobj>

was worried about <np>

Figure: Pattern

Source: [https://link.springer.com/chapter/10.1007/978-981-13-1610-4\\_23](https://link.springer.com/chapter/10.1007/978-981-13-1610-4_23)

Figure: Sentence Level Sentiment Classification

Source: <https://www.cs.uic.edu/~liub/FBS/NLP-handbook-sentiment-analysis.pdf>

# Lexicon (1 of 2)

- Base type:
  - Core words of opinion. Example: Good, beautiful, bad etc.
- Comparative type:
  - Comparison words of opinion Example: Better, worse, more important etc.
- Manual approach
- Dictionary based approach
- Corpus based approach

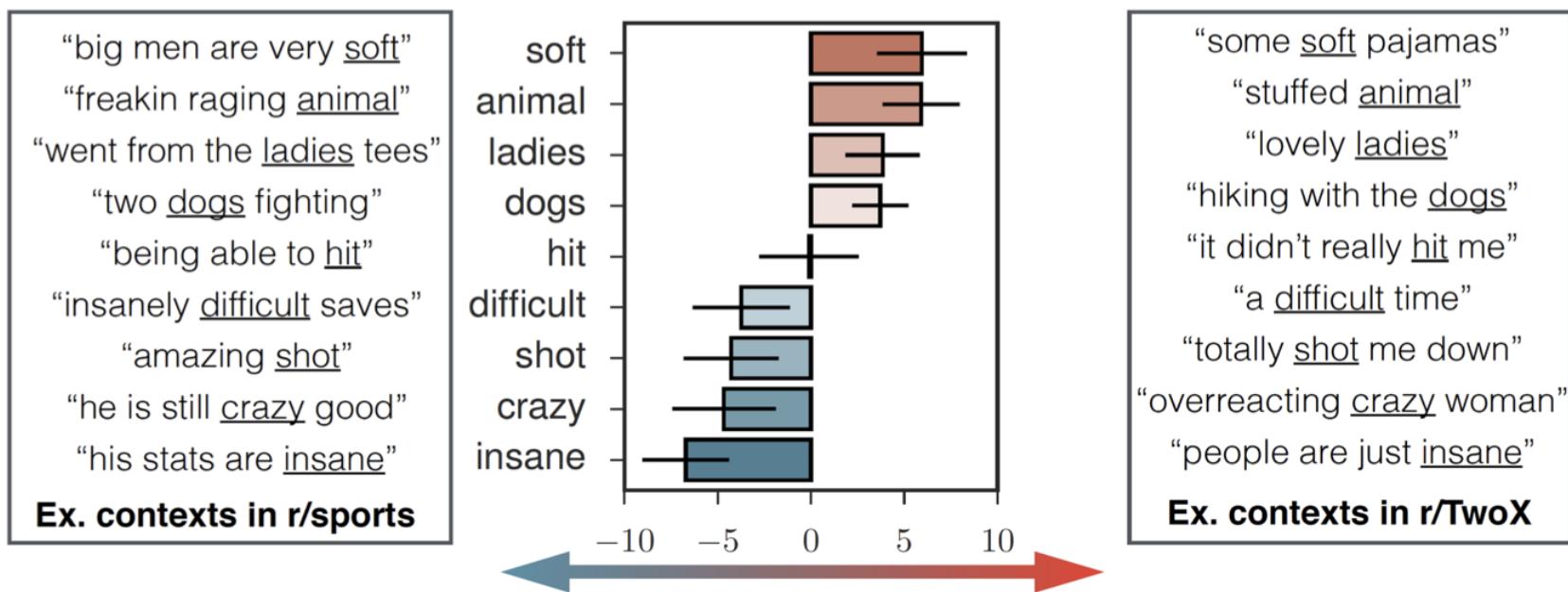


Figure: Domain specific Approach

Source: <https://nlp.stanford.edu/projects/socialsent/>

# Lexicon (2 of 2)

Lexicon	Positive Words	Negative Words
Simplest (SM)	good	bad
Simple List (SL)	good, awesome, great, fantastic, wonderful	bad, terrible, worst, sucks, awful, dumb
Simple List Plus (SL+)	good, awesome, great, fantastic, wonderful, best, love, excellent	bad, terrible, worst, sucks, awful, dumb, waist, boring, worse
Past and Future (PF)	will, has, must, is	was, would, had, were
Past and Future Plus (PF+)	will, has, must, is, good, awesome, great, fantastic, wonderful, best, love, excellent	was, would, had, were, bad, terrible, worst, sucks, awful, dumb, waist, boring, worse
Bing Liu	2006 words	4783 words
AFINN-96	516 words	965 words
AFINN-111	878 words	1599 words
enchantdelearning.com	266 words	225 words
MPAA	2721 words	4915 words
NRC Emotion	2312 words	3324 words

Figure: Corpus collection of opinion words

Source: [https://www.researchgate.net/figure/Examples-of-sentiment-lexicons\\_tbl1\\_288488744](https://www.researchgate.net/figure/Examples-of-sentiment-lexicons_tbl1_288488744)

# Feature-based sentiment analysis (1 of 2)

Identifying the object feature that the opinion is based upon.

Determine whether the opinion is positive negative or neutral.

Step 1: Feature extraction:

Identifying pros and cons elaborately

Example Review:

Pros: Great photos, easy to use, very small

Cons: Battery usage; included memory is stingy.

I had never used a digital camera prior to purchasing this Canon A70. I have always used a SLR

Opinion words:

Figure: Sample Review

great photos <photo>

easy to use <use>

very small <small>? <size>

battery usage <battery>

included memory is stingy <memory>

**Simple Sequence:** <{included, VB}{memory, NN}{is, VB}{stingy, JJ}>

**Labelled Sequence:** <{included, VB}{\$feature, NN}{is, VB}{stingy, JJ}>

**Rule:** <{easy, JJ }{to}{\*, VB}>? <{easy, JJ }{to}{\$feature, VB}>confidence = 90%

Figure: Segregated Opinion Words

# Feature-based sentiment analysis (2 of 2)

## Step 2: Opinion identification

Example:

“The picture quality of this camera is not great, but the battery life is long.”

- Opinion words and phrases count:

“The picture quality of this camera is not great [+1], but the battery life is long [0]”

- Handling negations:

“The picture quality of this camera is not great [-1], but the battery life is long [0]”

- Usage of but clauses:

“The picture quality of this camera is not great [-1], but the battery life is long [+1]”

- Aggregating opinions:

$$\text{score}(f_i, s) = \sum_{op_j \in s} \frac{op_j \cdot so}{d(op_j, f_i)},$$



Figure: Summarization

Source: <https://www.cs.uic.edu/~liub/FBS/NLP-handbook-sentiment-analysis.pdf>

Figure: Sentiment Analysis as a Whole Process

Source: <https://ars.els-cdn.com/content/image/1-s2.0-S2314728817300582-gr1.jpg>

# Self evaluation: Exercise 12

- To continue with the training, after learning the concepts of multimedia presentation generation and language interfaces for intelligent tutoring systems, it is time to write code to work CIRCSIM-Tutor, AUTOTUTOR, WHY2-ATLAS and argumentation for healthcare consumers. It is instructed to utilize the concept of clinical decision support systems and sentiment analysis and subjectivity.
- You are instructed to write the following activities using Python code.
- Exercise 12: Python code to Analyze the Sentiment based on subjects from a movie review Dataset.

# Self evaluation: Exercise 13

- To continue with the training, after learning the concepts of multimedia presentation generation and language interfaces for intelligent tutoring systems, it is time to write code to work CIRCSIM-Tutor, AUTOTUTOR, WHY2-ATLAS and argumentation for healthcare consumers. It is instructed to utilize the concept of clinical decision support systems and sentiment analysis and subjectivity.
- You are instructed to write the following activities using Python code.
- Exercise 13: Python code to Analyze the Sentiment based on sentences from Twitter Samples.

# Checkpoint (1 of 2)

## Multiple choice questions:

1. Select correct statements related to the tasks of Sentiment analysis or opinion mining:
  - a) Classifying the polarity of a given text at the document, sentence, or feature/aspect level
  - b) Check, whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral.
  - c) Some Advanced tasks captures, "beyond polarity" sentiment classification looks
  - d) All the above
2. NLP models are:
  - a) Generic
  - b) Specialized
  - c) Coupled
  - d) All the above
3. The auto tutor includes:
  - a) Animated conversational agent
  - b) Dialogue management
  - c) Electronic documents
  - d) All the above

# Checkpoint solutions (1 of 2)

## Multiple choice questions:

1. Select correct statements related to the tasks of Sentiment analysis or opinion mining:
  - a) Classifying the polarity of a given text at the document, sentence, or feature/aspect level
  - b) Check, whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral.
  - c) Some Advanced tasks captures, "beyond polarity" sentiment classification looks
  - d) **All the above**
2. NLP models are
  - a) Generic
  - b) Specialized
  - c) Coupled
  - d) **All the above**
3. The auto tutor includes
  - a) Animated conversational agent
  - b) Dialogue management
  - c) Electronic documents
  - d) **All the above**

# Checkpoint (2 of 2)

## Fill in the blanks:

1. Two main types of opinions are \_\_\_\_\_ and \_\_\_\_\_.
2. Opinion words are also called as \_\_\_\_\_ words
3. Polysemy is defined as the coexistence of multiple meanings for a word or phrase in a text object. \_\_\_\_\_ model is the best choice to correct Polysemy.
4. While working with text data obtained from news sentences, which are structured in nature, \_\_\_\_\_ grammar-based text parsing techniques can be used for noun phrase detection, verb phrase detection, subject detection and object detection.

## True or False:

1. Sentiment is the subset of emotion. True/False
2. Meaning of the text is important in text analysis for sentiments. True/False
3. Word2Vec model is a machine learning model used to create vector notations of text objects. Word2vec contains multiple deep neural networks. True/False

# Checkpoint solution (2 of 2)

## Fill in the blanks:

1. Two main types of opinions are regular opinions and comparative opinions
2. Opinion words are also called as polar words
3. Polysemy is defined as the coexistence of multiple meanings for a word or phrase in a text object. Convolutional Neural Networks model is the best choice to correct Polysemy.
4. While working with text data obtained from news sentences, which are structured in nature, Dependency Parsing and Constituency Parsing grammar-based text parsing techniques can be used for noun phrase detection, verb phrase detection, subject detection and object detection.

## True or False

1. Sentiment is the subset of emotion. **False**
2. Meaning of the text is important in text analysis for sentiments. **False**
3. Word2Vec model is a machine learning model used to create vector notations of text objects. Word2vec contains multiple deep neural networks. **False**

# Question bank

## Two mark questions:

1. How is the Layout of the document relevant in MM?
2. What are the features of why2atlas?
3. What is the core architecture of CDS based on?
4. What is feature based sentiment analysis?

## Four mark questions:

1. How are CDS reports summarized?
2. Describe AUTOTUTOR Architecture and Process
3. What is scripted dialogue and where to use it?
4. How to identify the meaning of text in MM ?

## Eight mark questions:

1. How is Layout and Meaning significant in Multimedia Presentation in NLP?
2. Write in detail the process steps involved in Sentiment Analysis with examples.

# Unit summary

**Having completed this unit, you should be able to:**

- Gain knowledge on the process of Multimedia Presentation Generation
- 
- Learn the concept of Language Interfaces for Intelligent Tutoring Systems
- Gain an insight into Argumentation for Healthcare Consumers
- Learn the concepts of Clinical Decision Support Systems
- Understand the core concepts of Sentiment Analysis and Subjectivity



# Lab Solution

A collage of various IT and business-related terms and acronyms, including Business, IT Infrastructure, Oil & Gas Informatics, Think, Software, Mainframe, LMS, Planet, SME, and various programmatic terms like Smarter Programmes, Open Source, Innovate, eLearning, and Cloud Computing & Virtualization.

# Unit objectives

**After completing this unit, you should be able to:**

- Learn about the solutions for all the lab exercises

# Lab specifications

- Hardware requirement:
  - i5 processor
  - 8GB RAM
  - Stable internet connection
- Software requirement:
  - Python
  - Pycharm or Jupyter IDEs
  - IBM Watson can be implemented along with Python NLTK
  - IBM Cloud - Bluemix

# Exercise 1: Solution

- Exercise 1: Text Retrieval.

# Exercise 2: Solution

- Exercise 2: Processing, Subsetting, Merging and Cleaning Text Data.

# Exercise 3: Solution

- Exercise 3: Simple Language Processing.

# Exercise 4: Solution

- Exercise 4: Accessing Text Corpora.

# Exercise 5: Solution

- Exercise 5: Processing Raw Text.

# Exercise 6: Solution

- Exercise 6: Categorizing and Tagging Words

# Exercise 7: Solution

- Exercise 7: Analysing Text Similarity

# Exercise 8: Solution

- Exercise 8: Analysing Word Sense

# Exercise 9: Solution

- Exercise 9: Analysing Meaning of Words and Sentences.

# Exercise 10: Solution

- Exercise 10: Extracting Information from Text.

# Exercise 11: Solution

- Exercise 11: Information Answering – Rule Based Text Analysis.

# Exercise 12: Solution

- Exercise 12: Analysis of Sentiment and Subjectivity.

# Exercise 13: Solution

- Exercise 13: Analysing Meaning of Sentences.

# Unit summary

**Having completed this unit, you should be able to:**

- Learn about the solutions for all the lab exercises

## 2. Information Retrieval Models

# Growth of textual information

---



*Literature*



*Email*



*www*

*How can we help manage and  
exploit all the information?*



*Desktop*



*News*



*Intranet*



*Blog*

# What is Information Retrieval (IR)?

---

- Narrow-sense:
  - IR= Search Engine Technologies (IR=Google, library info system)
  - IR= Text matching/classification
- Broad-sense: IR = Text Information Management:
  - General problem: how to manage text information?
  - How to find useful information? (retrieval)
    - **Example: Google**
  - How to organize information? (text classification)
    - **Example: Automatically assign emails to different folders**
  - How to discover knowledge from text? (text mining)
    - **Example: Discover correlation of events**

# First, nomenclature...

---

- Information retrieval (IR)
  - Focus on textual information (= text/document retrieval)
  - Other possibilities include image, video, music, ...
- What do we search?
  - Generically, “collections”
  - Less-frequently used, “corpora”
- What do we find?
  - Generically, “documents”
  - Even though we may be referring to web pages, PDFs, PowerPoint slides, paragraphs, etc.

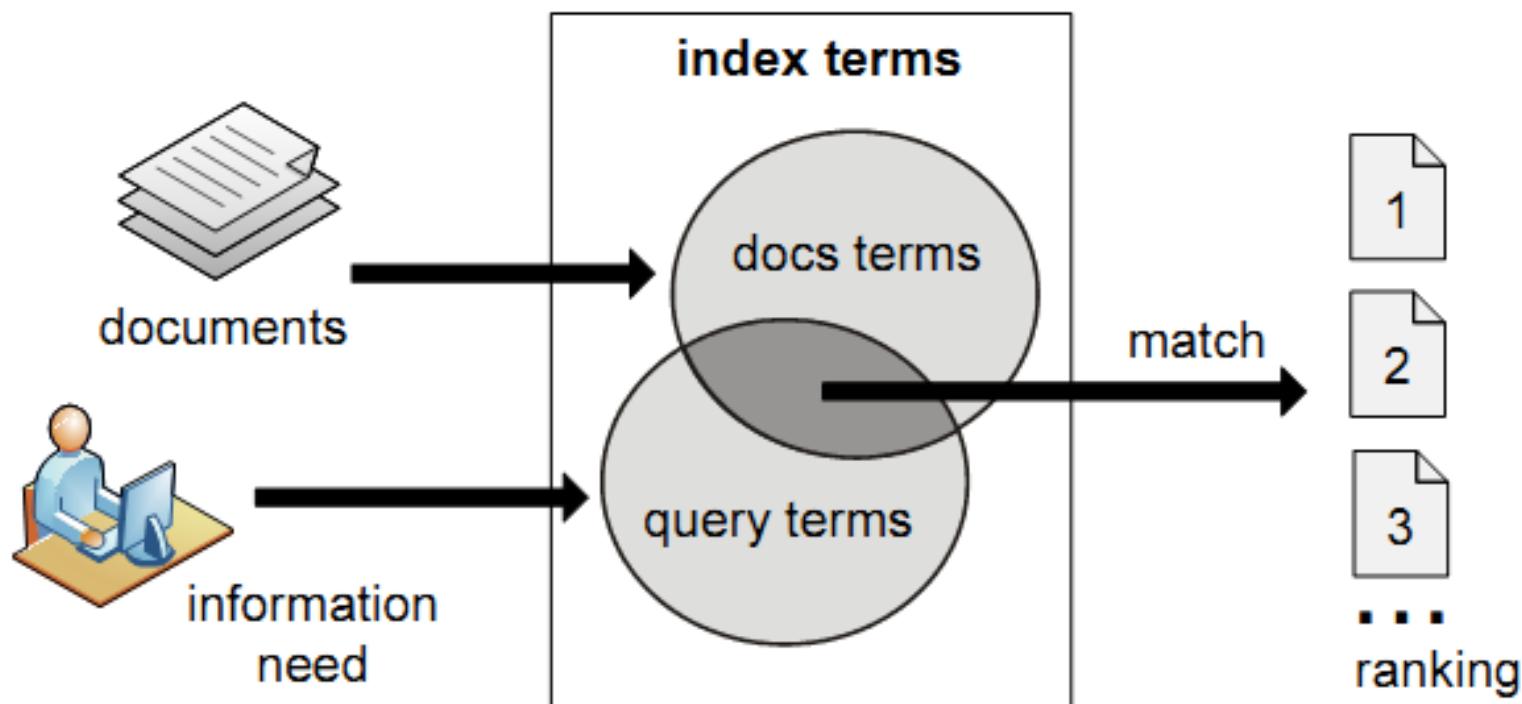
# Documents

---

- Let  $E_1, \dots, E_j, \dots, E_m$  denote entities in general. They can be:
  - Texts (books, journal articles, newspaper articles, papers, lecture notes, abstracts, titles, etc.),
  - Images (photographs, pictures, drawings, etc.),
  - Sounds (pieces of music, songs, speeches, etc.),
  - Multimedia (a collection of texts, images, and sounds),
  - A collection of Web pages,
  - And so on.

# Information Retrieval Process

---



# What's a word?

---

*Tax payers will have to link their PAN with Aadhaar by the stipulated deadline, which is this month-end, as the Supreme Court verdict on privacy has no bearing on the requirement, Unique Identification Authority of India (UIDAI) CEO Ajay Bhushan Pandey said today.*

भारत सरकार ने आर्थिक सर्वेक्षण में वित्तीय वर्ष 2005-06 में सात फ़ीसदी विकास दर हासिल करने का आकलन किया है और कर सुधार पर ज़ोर दिया है

天主教教宗若望保祿二世因感冒再度住進醫院。  
這是他今年第二度因同樣的病因住院。

وقال مارك ريجيف - الناطق باسم  
الخارجية الإسرائيلية - إن شارون قبل  
الدعوة وسيقوم للمرة الأولى بزيارة  
تونس، التي كانت لفترة طويلة المقر  
الرسمي لمنظمة التحرير الفلسطينية بعد خروجهما من لبنان عام 1982

日米連合で台頭中国に対処...アーミテージ前副長官提言

조재영 기자는 서울시는 25일 이명박 시장이 '행정중심복합도시' 건설안에 대해 '군대라도 동원해 막고 싶은 심정'이라고 말했다는 일부 언론의 보도를 부인했다.

# How do we represent text?

---

- Computers don’t “understand” anything!
- “Bag of words”
  - Treat all the words in a document as index terms
  - Assign a “weight” to each term based on “importance” (or, in simplest case, presence/absence of word)
  - Disregard order, structure, meaning, etc. of the words
  - Simple, yet effective!
- Assumptions
  - Term occurrence is independent
  - Document relevance is independent
  - “Words” are well-defined

# Sample Document

---

## McDonald's slims down spuds

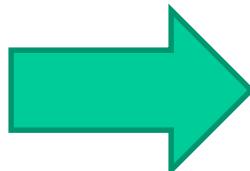
Fast-food chain to reduce certain types of fat in its french fries with new cooking oil.

NEW YORK (CNN/Money) - McDonald's Corp. is cutting the amount of "bad" fat in its french fries nearly in half, the fast-food chain said Tuesday as it moves to make all its fried menu items healthier.

But does that mean the popular shoestring fries won't taste the same? The company says no. "It's a win-win for our customers because they are getting the same great french-fry taste along with an even healthier nutrition profile," said Mike Roberts, president of McDonald's USA.

But others are not so sure. McDonald's will not specifically discuss the kind of oil it plans to use, but at least one nutrition expert says playing with the formula could mean a different taste.

Shares of Oak Brook, Ill.-based McDonald's (MCD: down \$0.54 to \$23.22, Research, Estimates) were lower Tuesday afternoon. It was unclear Tuesday whether competitors Burger King and Wendy's International (WEN: down \$0.80 to \$34.91, Research, Estimates) would follow suit. Neither company could immediately be reached for comment.



## ***"Bag of Words"***

14 × McDonalds

12 × fat

11 × fries

8 × new

7 × french

6 × company, said, nutrition

5 × food, oil, percent, reduce, taste, Tuesday

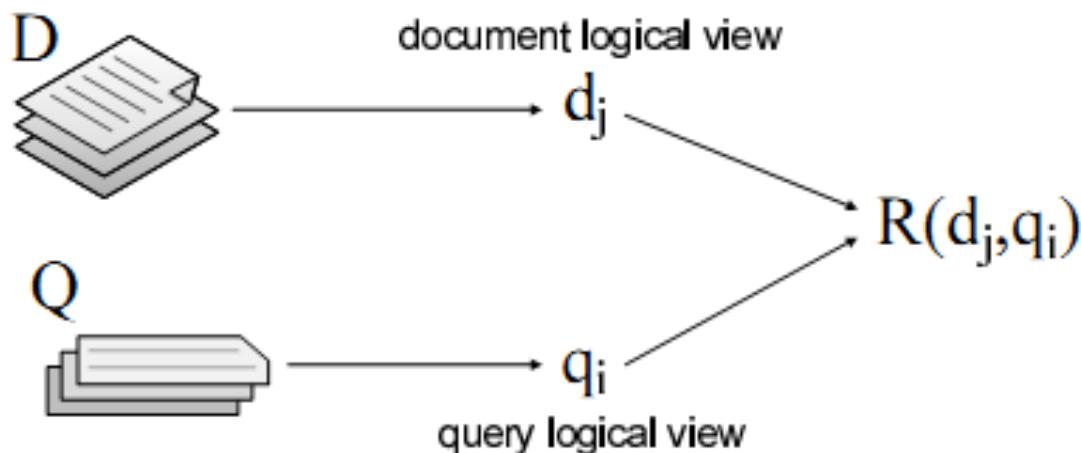
...

...

# IR Models

An **IR model** is a quadruple  $[D, Q, \mathcal{F}, R(q_i, d_j)]$  where

1.  $D$  is a set of logical views for the documents in the collection
2.  $Q$  is a set of logical views for the user queries
3.  $\mathcal{F}$  is a framework for modeling documents and queries
4.  $R(q_i, d_j)$  is a ranking function

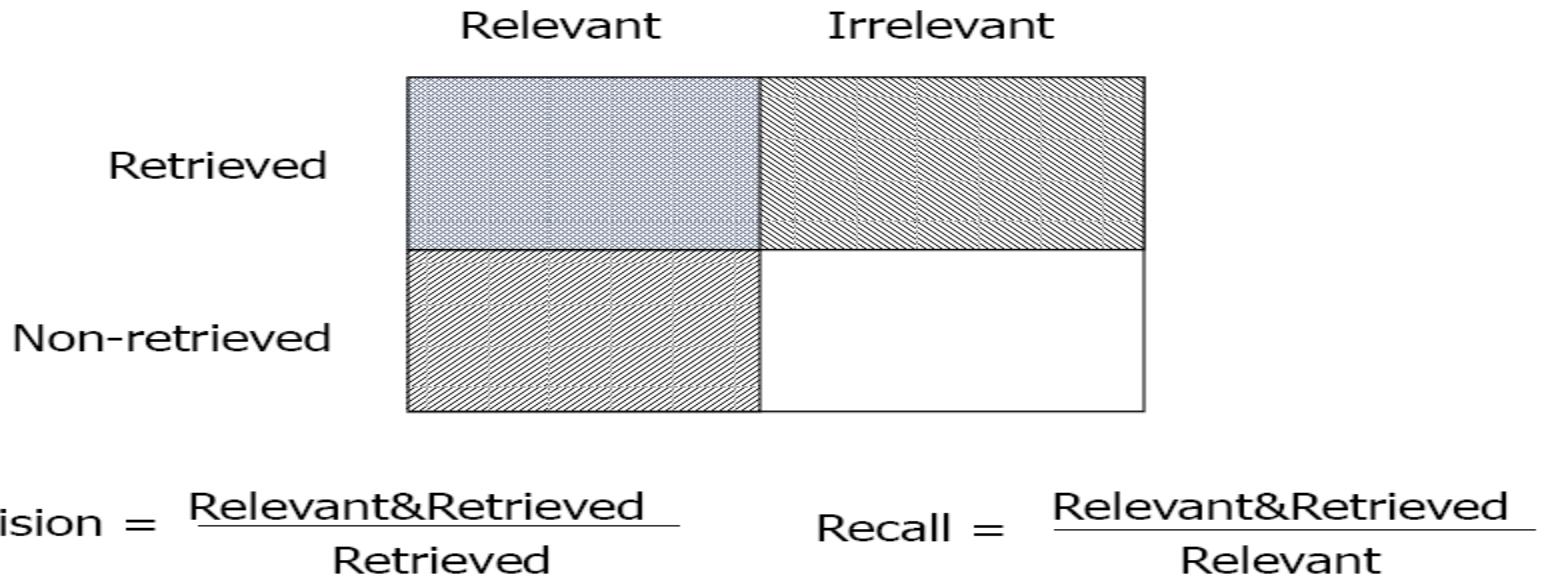


# Formalizing IR Tasks

---

- Vocabulary:  $V = \{w_1, w_2, \dots, w_T\}$  of a language
- Query:  $q = q_1, q_2, \dots, q_m$  where  $q_i \in V$ .
- Document:  $d_i = d_{i1}, d_{i2}, \dots, d_{im}$  where  $d_{ij} \in V$ .
- Collection:  $C = \{d_1, d_2, \dots, d_N\}$
- Relevant document set:  $R(q) \subseteq C$ : Generally unknown and user-dependent
- Query provides a “hint” on which documents should be in  $R(q)$
- IR: find the approximate relevant document set  $R'(q)$

# Precision & Recall



- **Precision** is the percentage of relevant items in the returned set
- **Recall** is the percentage of all relevant documents in the collection that is in the returned set.

# Evaluating Retrieval Performance

---

Total relevant docs = 8

1.	d1	<input checked="" type="checkbox"/> yes
2.	d2	<input checked="" type="checkbox"/> yes
3.	d3	<input type="checkbox"/> no
4.	d4	<input checked="" type="checkbox"/> yes
5.	d5	<input type="checkbox"/> no
6.	d6	<input type="checkbox"/> no
7.	d7	<input type="checkbox"/> no
8.	d8	<input type="checkbox"/> no
9.	d9	<input type="checkbox"/> no
10.	d10	<input checked="" type="checkbox"/> yes

$$\text{Precision} = \frac{\text{Relevant \& Retrieved}}{\text{Retrieved}} = 4/10 = 0.4$$

$$\text{Recall} = \frac{\text{Relevant \& Retrieved}}{\text{Relevant}} = 4/8 = 0.5$$

# Searching

---

- Given a query, score documents efficiently
- The basic question:
  - Given a query, how do we know if document A is more relevant than B?
    - If document A uses more query words than document B
    - Word usage in document A is more similar to that in query
    - ....
- We should find a way to compute relevance
  - Query and documents

# Retrieval Models

---

- A retrieval model specifies the details of:
  - Document representation
  - Query representation
  - Retrieval function
- Determines a notion of relevance.
- Notion of relevance can be binary or continuous (i.e. *ranked retrieval*).

# IR Models

U  
s  
e  
r  
  
T  
a  
s  
k

*Retrieval:*  
*Adhoc*  
*Filtering*

*Browsing*

*Classic Models*

- boolean*
- vector*
- probabilistic*

*Structured Models*

- Non-Overlapping Lists*
- Proximal Nodes*

*Browsing*

- Flat*
- Structure Guided*
- Hypertext*

*Set Theoretic*

*Fuzzy*

*Extended Boolean*

*Algebraic*

**Generalized Vector**

*Lat. Semantic Index*

*Neural Networks*

*Probabilistic*

*Inference Network*

*Belief Network*

# Classes of Retrieval Models

---

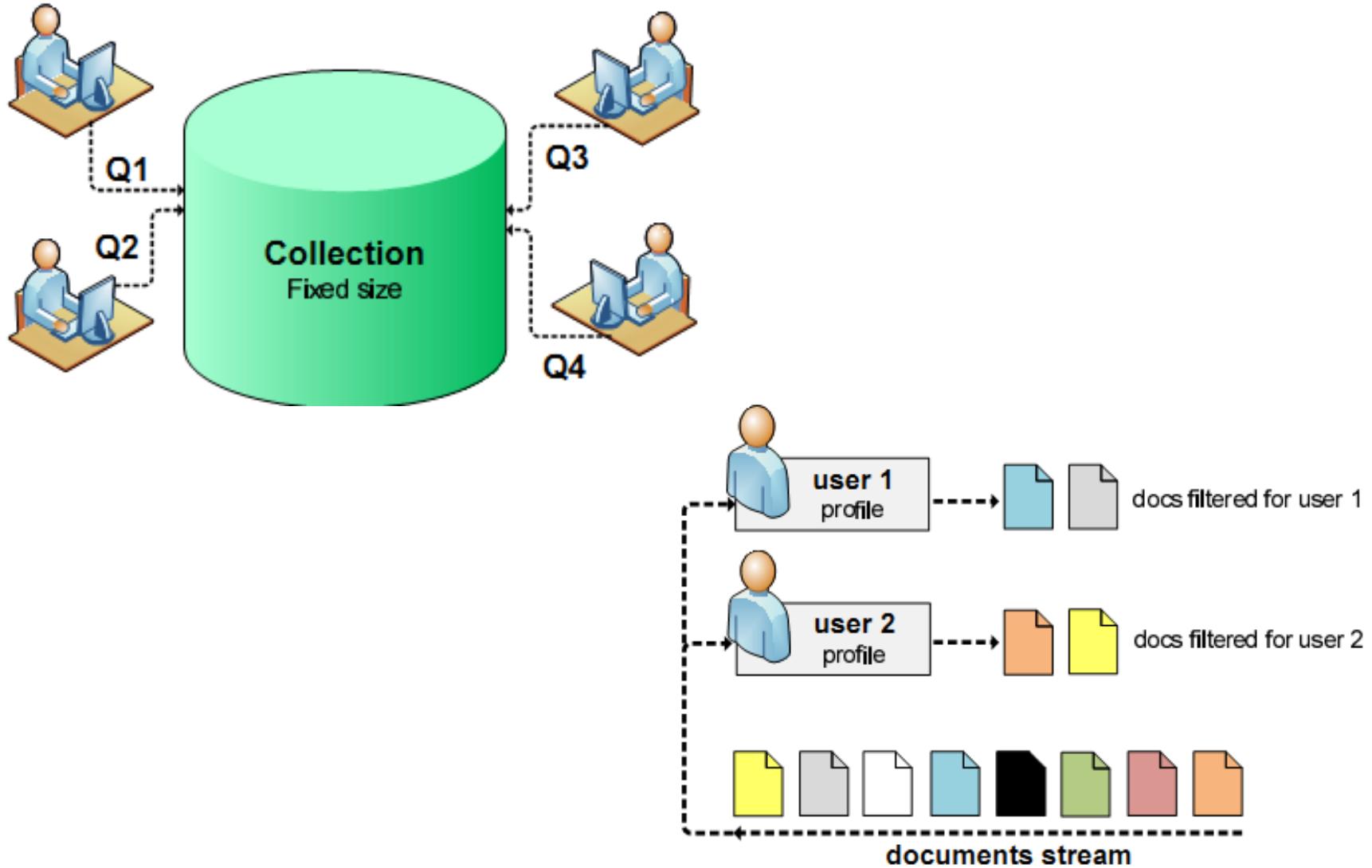
- Boolean models (set theoretic)
  - Extended Boolean
- Vector space models (statistical/algebraic)
  - Generalized VS
  - Latent Semantic Indexing
- Probabilistic models

# Other Model Dimensions

---

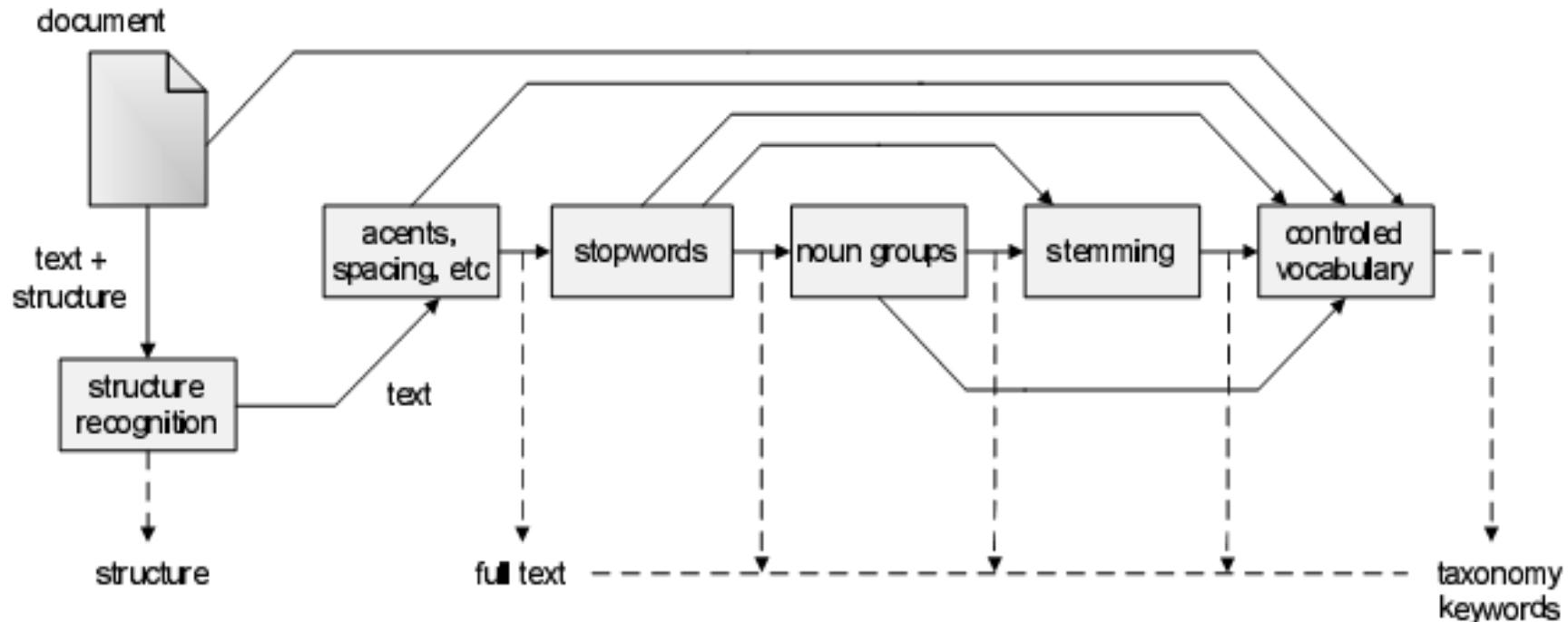
- Logical View of Documents
  - Index terms
  - Full text
  - Full text + Structure (e.g. hypertext)
- User Task
  - Retrieval
  - Browsing

# Retrieval : Ad Hoc x Filtering



# Logical view of a document:

- from full text to a set of index terms



# Stop List

---

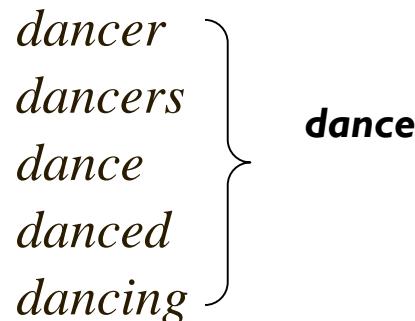
- Function words do not bear useful information for IR  
of, not, to, or, in, about, with, I, be, ...
- Stop list: contain stop words, not to be used as index
  - Prepositions
  - Articles
  - Pronouns
  - Some adverbs and adjectives
  - Some frequent words (e.g. document)
- The removal of stop words usually improves IR effectiveness
- A few “standard” stop lists are commonly used.

# Stemming

---

- *Reason:*
  - *Different word forms may bear similar meaning (e.g. search, searching): create a “standard” representation for them*
- *Stemming:*
  - *Removing some endings of word*

*dancer*  
*dancers*  
*dance*  
*danced*  
*dancing*



The diagram shows five words: "dancer", "dancers", "dance", "danced", and "dancing". A vertical brace on the right side groups all these words together under the bolded stem word "dance".

- *2 Types*
  - *Dictionary-based stemming*
    - *high stemming accuracy*
  - *Porter-style stemming*

---

# Boolean Model

# Boolean Model

---

*The Boolean model of IR (BIR) is*

- *a classical IR model and, at the same time,*
- *the first and most adopted one.*

*It is used by virtually all commercial IR systems today.*

*The BIR is based on:*

- *Boolean Logic and*
- *classical Sets Theory*

*in that both the documents to be searched and the user's query are conceived as sets of terms.*

*Retrieval is based on whether or not the documents contain the query terms.*

# Boolean model

---

- Each document or query is treated as a “**bag**” of **words** or **terms**. Word sequence is not considered.
- Given a collection of documents  $D$ ,
  - let  $V = \{t_1, t_2, \dots, t_{|V|}\}$  be the set of distinctive words/terms in the collection.
  - $V$  is called the **vocabulary**.
- A weight  $w_{ij} > 0$  is associated with each term  $t_i$  of a document  $\mathbf{d}_j \in D$ .
- For a term that does not appear in document  $\mathbf{d}_j$ ,  $w_{ij} = 0$ .

$$\mathbf{d}_j = (w_{1j}, w_{2j}, \dots, w_{|V|j}),$$

# Boolean Logic

---

$$C = A$$

$$C = \overline{A}$$

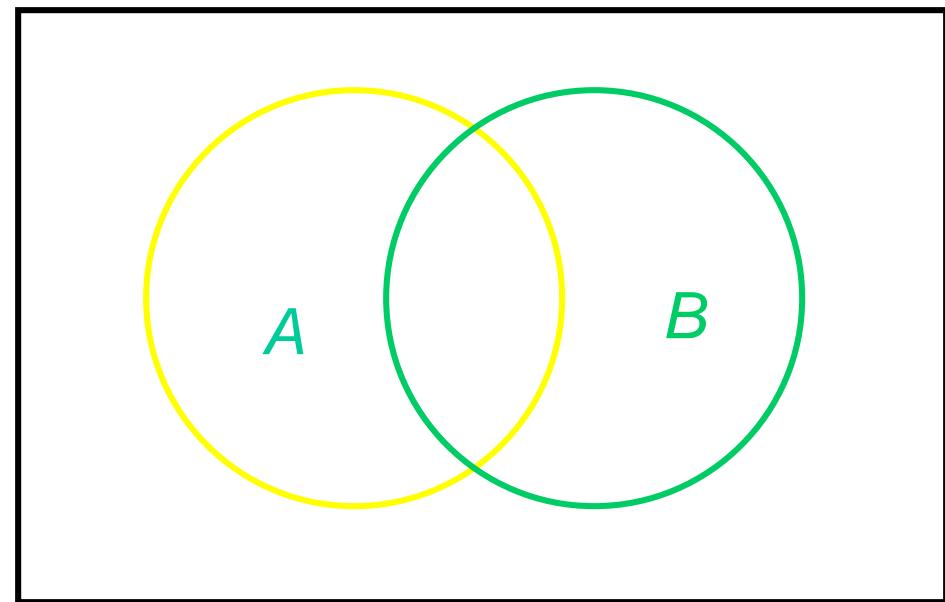
$$C = A \underset{\wedge}{\underset{\circ}{\cap}} B$$

$$C = A \underset{\vee}{\underset{\circ}{\dot{\cup}}} B$$

DeMorgan's Law :

$$\overline{A \underset{\wedge}{\underset{\circ}{\cap}} B} = \overline{A} \underset{\vee}{\underset{\circ}{\dot{\cup}}} \overline{B}$$

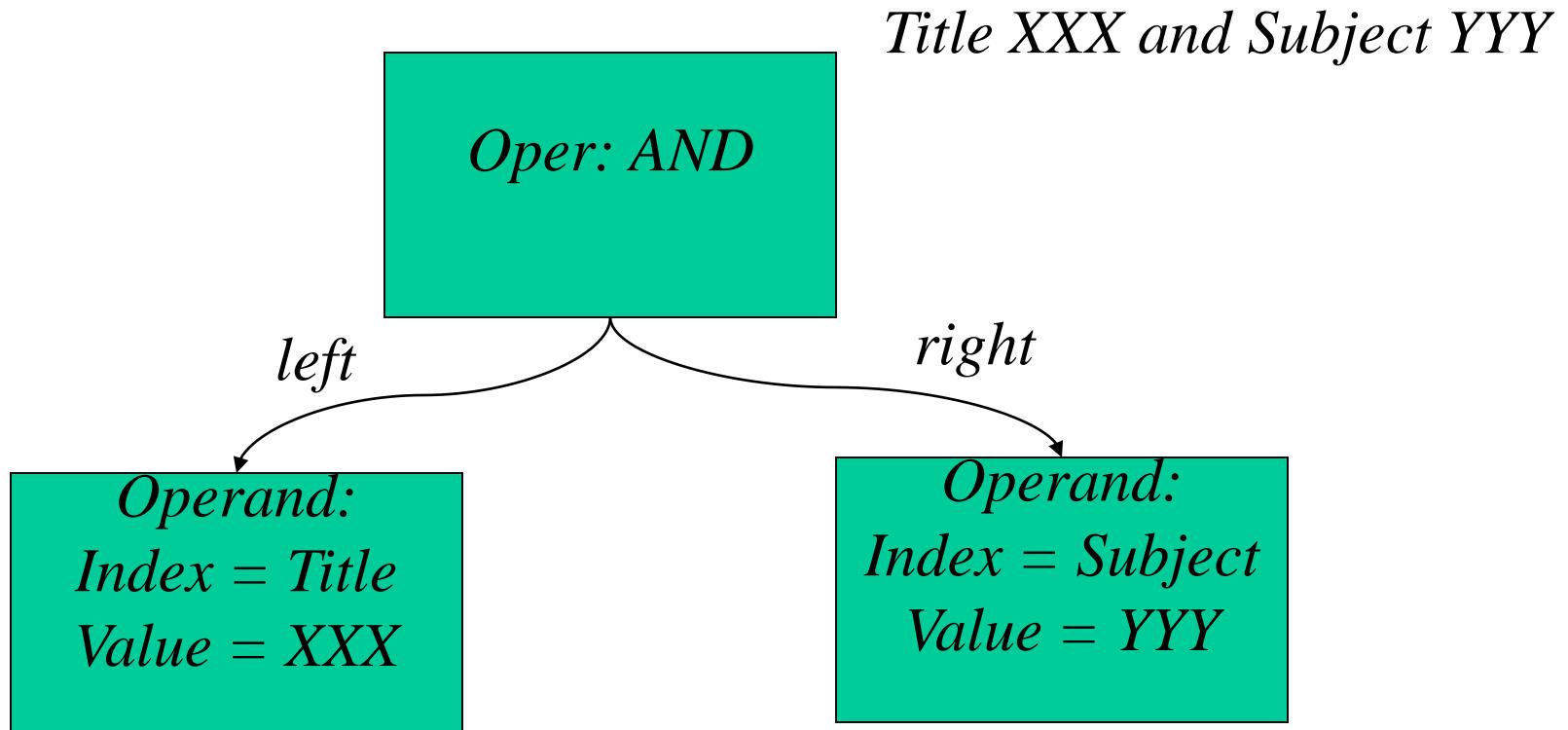
$$\overline{A \underset{\vee}{\underset{\circ}{\dot{\cup}}} B} = \overline{A} \underset{\wedge}{\underset{\circ}{\cap}} \overline{B}$$



# Parse Result (Query Tree)

---

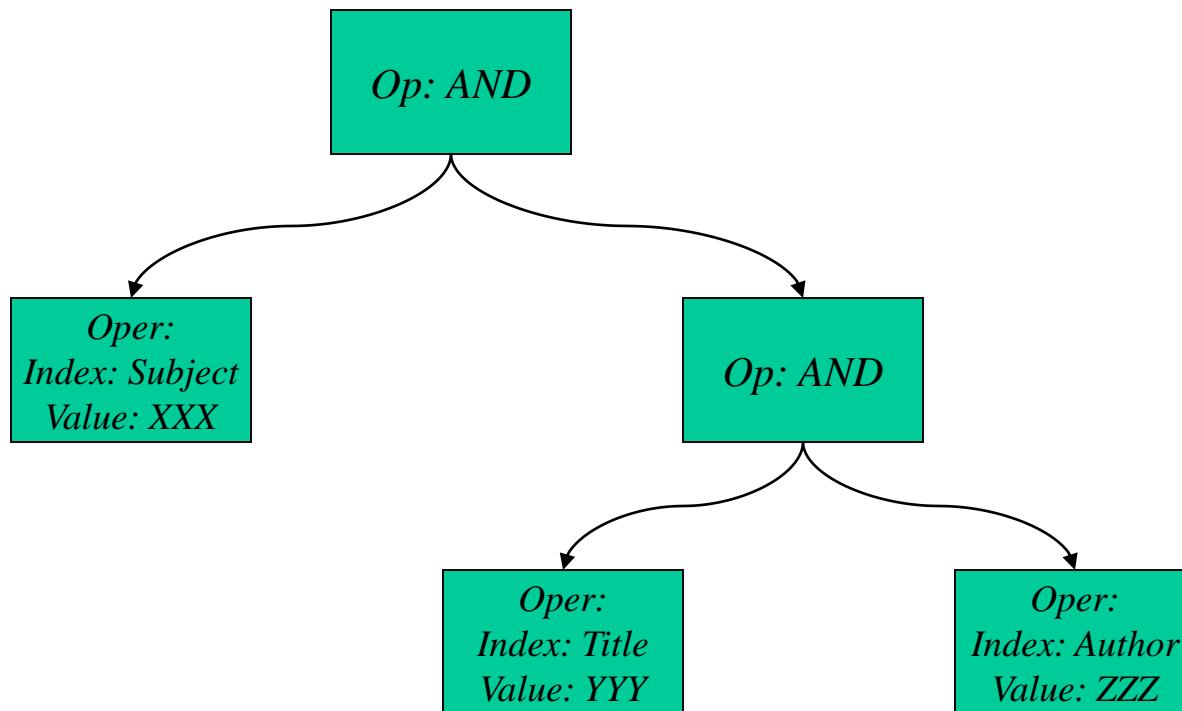
- Z39.50 queries...



# Parse Results

---

- Subject XXX and (title yyy and author zzz)



# Boolean AND Algorithm

---

2
5
7
8
15
29
35
100
135
140
155
189
190
195
198

*AND*

2
8
9
12
15
22
28
50
68
77
84
100
120
128
135
138
141
150
155
188
189
195

=

2
8
15
100
135
155
189
195

# Boolean OR Algorithm

---

2
5
7
8
15
29
35
100
135
140
155
189
190
195
198

*OR*

2
8
9
12
15
22
28
50
68
77
84
100
120
128
135
138
141
150
155
188
189
195

*=*

2
5
7
8
9
12
15
22
28
29
35
50
68
77
84
100
120
128
135
138
141
150
155
188
189
190
195
198

# Boolean AND NOTAlgorithm

---

2
5
7
8
15
29
35
100
135
140
155
189
190
195
198

*AND NOT*

2
8
9
12
15
22
28
50
68
77
84
100
120
128
135
138
141
150
155
188
189
195

=

5
7
15
29
35
140
190
198

# Boolean model (contd)

---

- Query terms are combined logically using the Boolean operators **AND**, **OR**, and **NOT**.
  - E.g.,  $((data \text{ AND } mining) \text{ AND } (\text{NOT } text))$
- Retrieval
  - Given a Boolean query, the system retrieves every document that makes the query logically true.
  - Called **exact match**.
- The retrieval results are usually quite poor because term frequency is not considered.

# Boolean queries: Exact match

---

- The Boolean retrieval model is being able to ask a query that is a Boolean expression:
  - Boolean Queries are queries using *AND*, *OR* and *NOT* to join query terms
    - Views each document as a set of words
    - Is precise: document matches condition or not.
  - Perhaps the simplest model to build an IR system on
- Primary commercial retrieval tool for 3 decades.
- Many search systems you still use are Boolean:
  - Email, library catalog, Mac OS X Spotlight

# ***The Boolean Model***

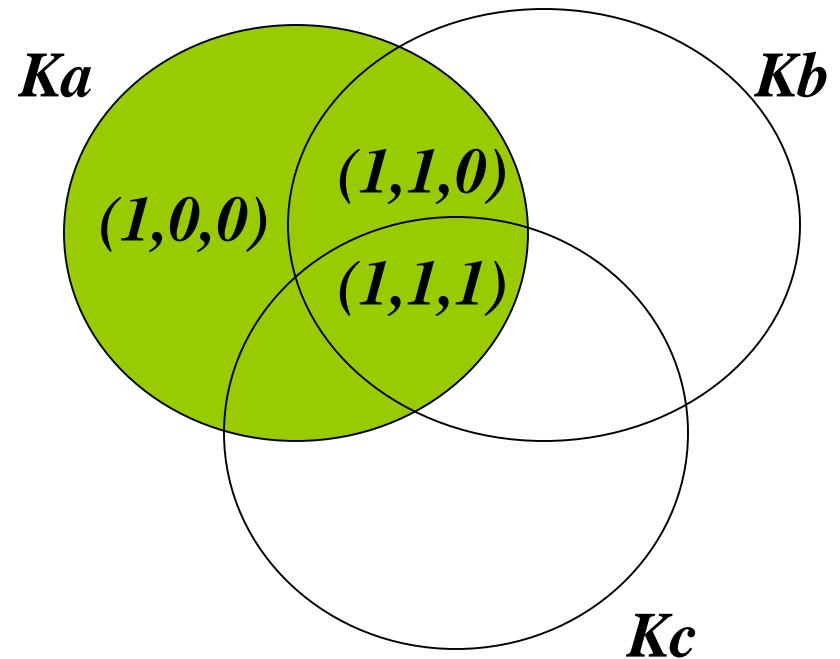
---

- *Simple model based on set theory*
- *Queries specified as boolean expressions*
  - precise semantics
  - neat formalism
  - $q = ka \wedge (kb \vee \neg kc)$
- *Terms are either present or absent. Thus,  $w_{ij} \in \{0,1\}$*
- *Consider*
  - $q = ka \wedge (kb \vee \neg kc)$
  - $\text{vec}(qdnf) = (1,1,1) \vee (1,1,0) \vee (1,0,0)$
  - $\text{vec}(qcc) = (1,1,0)$  is a conjunctive component
- *Each query can be transformed in Disjunctive normal form (DNF) form*

# The Boolean Model

---

- $q = ka \wedge (kb \vee \neg kc)$



- $sim(q,dj) = 1$ , if document satisfies the boolean query
  - o otherwise
- **no in-between, only 0 or 1**

*Retrieval is defined as follows:*

1. *The set  $S_i$  of documents are obtained that contain or not the term under focus:*

*for term A:  $S_i = \{D / A \in D\}$*

*for negated term A, i.e.,  $\neg A$ :  $S_i = \{D / A \notin D\}$*

2. *Those documents are retrieved in response to Q, which belong to the set obtained as a result of the corresponding sets operations:*

*intersection  $\cap$  corresponds to logical AND,  
union  $\cup$  corresponds to logical OR.*

*For example,*

$$Q = A \text{ } OR \text{ } (B \text{ AND } C)$$

*$S_1$  results set for term A,*

*$S_2$  results set for term B,*

*$S_3$  results set for term C,*

*The retrieved set in response to Q:*

$$S_1 \cup (S_2 \cap S_3)$$

## EXAMPLE

Let the set of original (real) documents  $D = \{D1, D2, D3\}$

---

*D1 = Bayes' Principle: The principle that, in estimating a parameter, one should initially assume that each possible value has equal probability (a uniform prior distribution).*

*D2 = Bayesian Decision Theory: A mathematical theory of decision-making which presumes utility and probability functions, and according to which the act to be chosen is the Bayes act, i.e. the one with highest Subjective Expected Utility. If one had unlimited time and calculating power with which to make every decision, this procedure would be the best way to make any decision.*

*D3 = Bayesian Epistemology: A philosophical theory which holds that the epistemic status of a proposition (i.e. how well proven or well established it is) is best measured by a probability and that the proper way to revise this probability is given by Bayesian conditionalisation or similar procedures. A Bayesian epistemologist would use probability to define, and explore the relationship between, concepts such as epistemic status, support or explanatory power.*

Let the set T of terms be:

$T = \{$

- t1 = Bayes' Principle,*
- t2 = probability,*
- t3 = decision-making,*
- t4 = Bayesian Epistemology }*

# Example

---

*The set  $D$  of documents is as follows:*

$D = \{D1, D2, D3\}$ , where

$D1 = \{\text{Bayes' Principle, probability}\}$

$D2 = \{\text{probability, decision-making}\}$

$D3 = \{\text{probability, Bayesian Epistemology}\}$

*Let the query  $Q$  be:*

$Q = \text{probability AND decision-making}$

# Solution

---

$Q = \text{probability AND decision-making}$

1. Firstly, the following sets  $S1$  and  $S2$  of documents  $Dj$  are obtained (retrieved):

$$S1 = \{Dj / \text{probability} \in Dj\} = \{D1, D2, D3\}$$

$$S2 = \{Dj / \text{decision-making} \in Dj\} = \{D2\}$$

2. Finally, the following documents  $Dj$  are retrieved in response to  $Q$ :

$$\begin{aligned} \{Dj / Dj \in S1 \cap S2\} &= \{D1, D2, D3\} \cap \{D2\} \\ &= \{\mathbf{D2}\} \end{aligned}$$

# Boolean Model example

**Doc 1:** "Computers have brought the world to our fingertips. We will try to understand at a basic level the science -- old and new -- underlying this new Computational Universe. Our quest takes us on a broad sweep of scientific knowledge and related technologies... Ultimately, this study makes us look anew at ourselves -- our genome; language; music; "knowledge"; and, above all, the mystery of our intelligence. (cos 116 description)

**Doc 2:** "An introduction to computer science in the context of scientific, engineering, and commercial applications. The goal of the course is to teach basic principles and practical issues, while at the same time preparing students to use computers effectively for applications in computer science ..." (cos 126 description)

**Query:** (principles AND knowledge) OR (science AND engineering)

Doc 1:      0                          1                          1                          0                          FALSE

# Boolean Model example

**Doc 1:** "Computers have brought the world to our fingertips. We will try to understand at a basic level the science -- old and new -- underlying this new Computational Universe. Our quest takes us on a broad sweep of scientific knowledge and related technologies... Ultimately, this study makes us look anew at ourselves -- our genome; language; music; "knowledge"; and, above all, the mystery of our intelligence. (cos 116 description)

**Doc 2:** "An introduction to computer **science** in the context of scientific, **engineering**, and commercial applications. The goal of the course is to teach **basic principles** and practical issues, while at the same time preparing students to use computers effectively for applications in computer **science** ..." (cos 126 description)

**Query:** (principles AND knowledge) OR (science AND engineering)

Doc 2:      1                    0                    1                    1                    TRUE

# Exercise

---

- $D_1 = \text{"computer information retrieval"}$
  - $D_2 = \text{"computer retrieval"}$
  - $D_3 = \text{"information"}$
  - $D_4 = \text{"computer information"}$
- 
- $Q_1 = \text{"information} \wedge \text{retrieval"}$
  - $Q_2 = \text{"information} \wedge \neg \text{computer"}$

# Boolean Retrieval Model

---

- Popular retrieval model because:
  - Easy to understand for simple queries.
  - Clean formalism.
- Strengths
  - Precise, if you know the right strategies
  - Precise, if you have an idea of what you're looking for
  - Implementations are fast and efficient

# Boolean Models – Weaknesses

---

- Very rigid: AND means all; OR means any.
- Difficult to express complex user requests.
- Difficult to control the number of documents retrieved.
  - *All* matched documents will be returned.
- Difficult to rank output.
  - *All* matched documents logically satisfy the query.
- Difficult to perform relevance feedback.
  - If a document is identified by the user as relevant or irrelevant, how should the query be modified?

# Boolean model Weaknesses

---

- Weaknesses
  - Users must learn Boolean logic
  - Boolean logic insufficient to capture the richness of language
  - No control over size of result set: either too many hits or none
  - **When do you stop reading?** All documents in the result set are considered “equally good”
  - **What about partial matches?** Documents that “don’t quite match” the query may be useful also

- 
- The Boolean model imposes a binary criterion for deciding relevance
  - The question of how to extend the Boolean model to accommodate partial matching and a ranking has attracted considerable attention in the past
  - Two extensions of boolean model:
    - Fuzzy Set Model
    - Extended Boolean Model

# Statistical Models

---

- A document is typically represented by a *bag of words* (unordered words with frequencies).
- Bag = set that allows multiple occurrences of the same element.
- User specifies a set of desired terms with optional weights:
  - Weighted query terms:  
 $Q = \langle \text{database} \ 0.5; \text{text} \ 0.8; \text{information} \ 0.2 \rangle$
  - Unweighted query terms:  
 $Q = \langle \text{database}; \text{text}; \text{information} \rangle$
  - No Boolean conditions specified in the query.

# Statistical Retrieval

---

- Retrieval based on *similarity* between query and documents.
- Output documents are ranked according to similarity to query.
- Similarity based on occurrence *frequencies* of keywords in query and document.
- Automatic relevance feedback can be supported:
  - Relevant documents “added” to query.
  - Irrelevant documents “subtracted” from query.

# Issues for Vector Space Model

---

- How to determine important words in a document?
  - Word sense?
  - Word  $n$ -grams (and phrases, idioms,...) → terms
- How to determine the degree of importance of a term within a document and within the entire collection?
- How to determine the degree of similarity between a document and the query?
- In the case of the web, what is a collection and what are the effects of links, formatting information, etc.?

# Vector-Space Model

---

- $t$  distinct terms remain after preprocessing
  - Unique terms that form the VOCABULARY
- These “orthogonal” terms form a vector space.  
Dimension =  $t = |\text{vocabulary}|$ 
  - 2 terms → bi-dimensional; ...;  $n$ -terms →  $n$ -dimensional
- Each term,  $i$ , in a document or query  $j$ , is given a real-valued weight,  $w_{ij}$ .
- Both documents and queries are expressed as  $t$ -dimensional vectors:

$$d_j = (w_{1j}, w_{2j}, \dots, w_{tj})$$

# Vector-Space Model

---

## Query as vector:

- We regard query as short document
- We return the documents ranked by the closeness of their vectors to the query, also represented as a vector.
- Vectorial model was developed in the SMART system (Salton, c. 1970) and standardly used by TREC participants and web IR systems

# Algorithm for Vector Space Retrieval

---

1. Given a set  $D$  of documents.
2. Identify terms.
3. Exclude stopwords.
4. Apply stemming to remaining words.
5. Compute for each document  $D_j$  and term  $t_i$  a weight  $w_{ij}$ .
6. A query  $Q_k$  coming from a user is also conceived as being a document; a weight vector  $\mathbf{v}_k$  can be computed for it as well, in a similar way.
7. Retrieval is defined as follows:

*Document  $D_j$  is retrieved in response to query  $Q_k$  if the document and the query are "similar enough," i.e., a similarity measure  $s_{jk}$  between the document (identified by  $\mathbf{v}_j$ ) and the query (identified by  $\mathbf{v}_k$ ) is over some threshold  $K$ .*

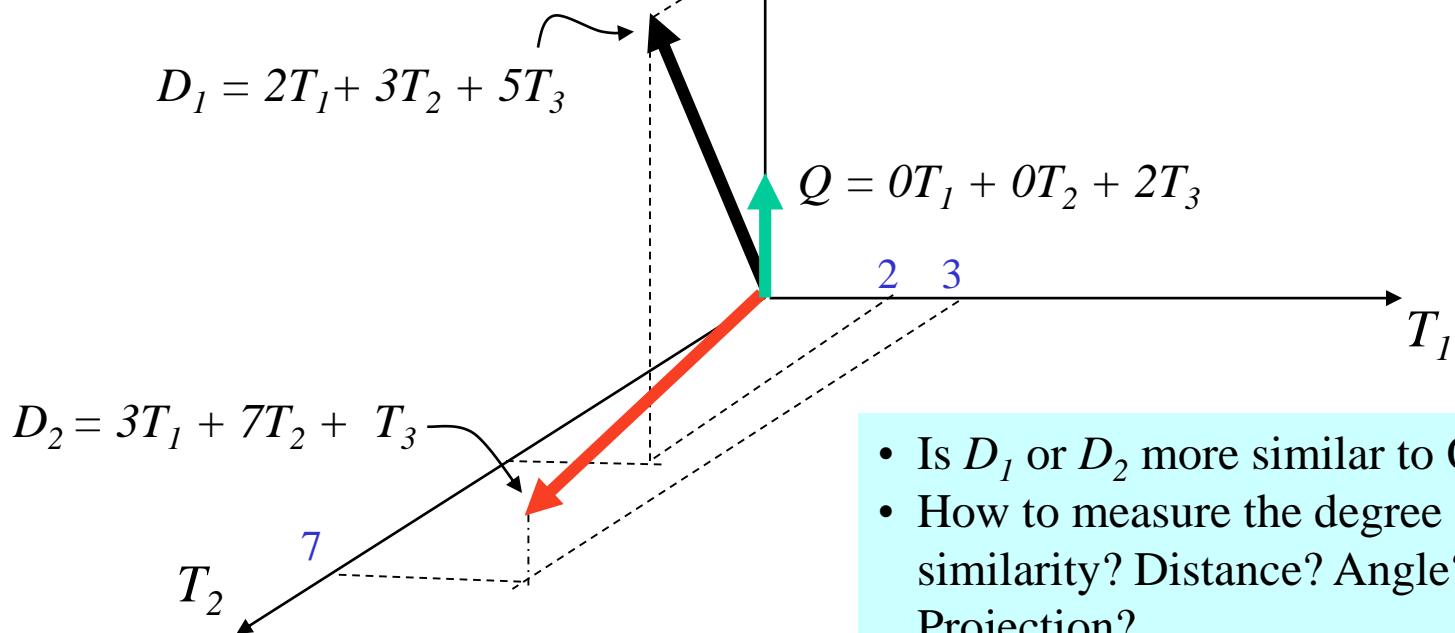
# Graphic Representation

Example:

$$D_1 = 2T_1 + 3T_2 + 5T_3$$

$$D_2 = 3T_1 + 7T_2 + T_3$$

$$Q = 0T_1 + 0T_2 + 2T_3$$



# Document Collection

---

- A collection of  $n$  documents can be represented in the vector space model by a **term-document matrix**.
- An entry in the matrix corresponds to the “**weight**” of a **term in the document**; zero means the term has no significance in the document or it simply doesn’t exist in the document.

$$\left( \begin{array}{ccccc} & T_1 & T_2 & \dots & T_t \\ D_1 & w_{11} & w_{21} & \dots & w_{t1} \\ D_2 & w_{12} & w_{22} & \dots & w_{t2} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ D_n & w_{1n} & w_{2n} & \dots & w_{tn} \end{array} \right)$$

# Term Weights: Term Frequency

---

- More frequent terms in a document are more important, i.e. more indicative of the topic.

$$f_{ij} = \text{frequency of term } i \text{ in document } j$$

- May want to **normalize *term frequency* (*tf*)** by dividing by the frequency of the most common term in the document:

$$tf_{ij} = f_{ij} / \max_i \{f_{ij}\}$$

# Term Weights: Inverse Document Frequency (IDF)

---

- Terms that appear in many *different* documents are *less* indicative of overall topic.

$df_i$  = document frequency of term  $i$

= number of documents containing term  $i$

$idf_i$  = inverse document frequency of term  $i$ ,

=  $\log_2 (N / df_i)$

( $N$ : total number of documents)

- An indication of a term's *discrimination* power.
- Log used to dampen the effect relative to  $tf$ .

# TF-IDF Weighting

---

- A typical combined term importance indicator is *tf-idf weighting*:

$$w_{ij} = tf_{ij} idf_i = tf_{ij} \log_2 (N/ df_i)$$

- *A term occurring frequently in the document but rarely in the rest of the collection is given high weight.*
- Many other ways of determining term weights have been proposed.
- Experimentally, *tf-idf* has been found to work well.

# Computing TF-IDF -- An Example

---

Given a document containing terms with given frequencies:

A(3), B(2), C(1)

Assume collection contains 10,000 documents and  
document frequencies of these terms are:

A(50), B(1300), C(250)

Then:

A:  $tf = 3/3$ ;  $idf = \log_2(10000/50) = 7.6$ ;  $tf\text{-}idf = 7.6$

B:  $tf = 2/3$ ;  $idf = \log_2(10000/1300) = 2.9$ ;  $tf\text{-}idf = 2.0$

C:  $tf = 1/3$ ;  $idf = \log_2(10000/250) = 5.3$ ;  $tf\text{-}idf = 1.8$

# Query Vector

---

- Query vector is typically treated as a document and also tf-idf weighted.
- Alternative is for the user to supply weights for the given query terms.

# Similarity Measure

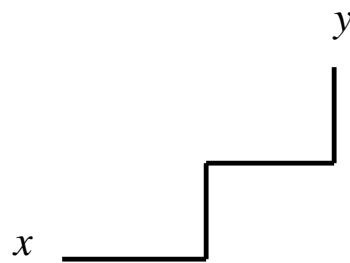
---

- We now have vectors for all documents in the collection, a vector for the query, how to compute similarity?
- A **similarity measure** is a function that computes the *degree of similarity* between two vectors.
- Using a similarity measure between the query and each document:
  - It is possible to rank the retrieved documents in the order of presumed relevance.
  - It is possible to enforce a certain threshold so that the size of the retrieved set can be controlled.

# One cut: Manhattan Distance

---

- Or “city block” measure
  - Based on the idea that generally in American cities you cannot follow a direct line between two points.



- Uses the formula: , also known as hamming distance

$$ManhDist(X, Y) = \sum_{i=1}^n |x_i - y_i|$$

- **Exercise:** Determine the Manhattan distance between the vectors  $(0, 3, 2, 1, 10)$  and  $(2, 7, 1, 0, 0)$

## Second cut: Euclidean distance

---

- Distance between vectors  $d_1$  and  $d_2$  is the length of the vector  $|d_1 - d_2|$ .
  - Euclidean distance

$$\begin{aligned} d(\mathbf{p}, \mathbf{q}) &= d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} \\ &= \sqrt{\sum_{i=1}^n (q_i - p_i)^2}. \end{aligned}$$

- **Exercise:** Determine the Euclidean distance between the vectors  $(0, 3, 2, 1, 10)$  and  $(2, 7, 1, 0, 0)$
- We still haven't dealt with the issue of length normalization
  - Long documents would be more similar to each other by virtue of length, not topic
- However, we can implicitly normalize by looking at *angles* instead

# Third CUT: Similarity Measure - Inner Product

---

- Similarity between vectors for the document  $d_i$  and query  $q$  can be computed as the vector inner product (a.k.a. dot product):

$$\text{sim}(d_j, q) = d_j \cdot q = \sum_{i=1}^t w_{ij} w_{iq}$$

where  $w_{ij}$  is the weight of term  $i$  in document  $j$  and  $w_{iq}$  is the weight of term  $i$  in the query

- For binary vectors, the inner product is the number of matched query terms in the document (size of intersection).
- For weighted term vectors, it is the sum of the products of the weights of the matched terms.

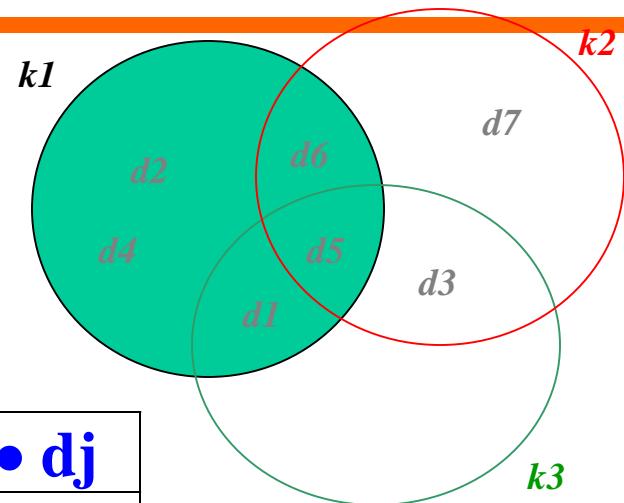
# Properties of Inner Product

---

- The inner product is unbounded.
- Favors long documents with a large number of unique terms.
- Measures how many terms matched but not how many terms are *not* matched.

# Inner Product:

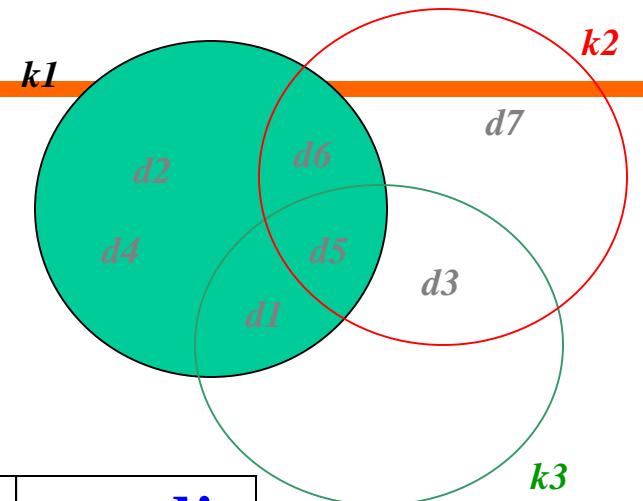
## Example 1



	<b>k1</b>	<b>k2</b>	<b>k3</b>	<b><math>q \bullet dj</math></b>
<b>d1</b>	1	0	1	<b>2</b>
<b>d2</b>	1	0	0	<b>1</b>
<b>d3</b>	0	1	1	<b>2</b>
<b>d4</b>	1	0	0	<b>1</b>
<b>d5</b>	1	1	1	<b>3</b>
<b>d6</b>	1	1	0	<b>2</b>
<b>d7</b>	0	1	0	<b>1</b>
<b>q</b>	1	1	1	

# *Inner Product:*

## *Exercise*



	$k1$	$k2$	$k3$	$q \bullet d_j$
$d1$	1	0	1	?
$d2$	1	0	0	?
$d3$	0	1	1	?
$d4$	1	0	0	?
$d5$	1	1	1	?
$d6$	1	1	0	?
$d7$	0	1	0	?
$q$	1	2	3	

# Inner Product -- Examples

---

Binary: retrieval  
database  
architecture  
computer  
text  
management  
information

- $D = [1, 1, 1, 0, 1, 1, 0]$
- $Q = [1, 0, 1, 0, 0, 1, 1]$

Size of vector = size of vocabulary = 7  
0 means corresponding term not found in document or query

$$\text{sim}(D, Q) = 3$$

Weighted:

$$D_1 = 2T_1 + 3T_2 + 5T_3 \quad D_2 = 3T_1 + 7T_2 + 1T_3 \\ Q = 0T_1 + 0T_2 + 2T_3$$

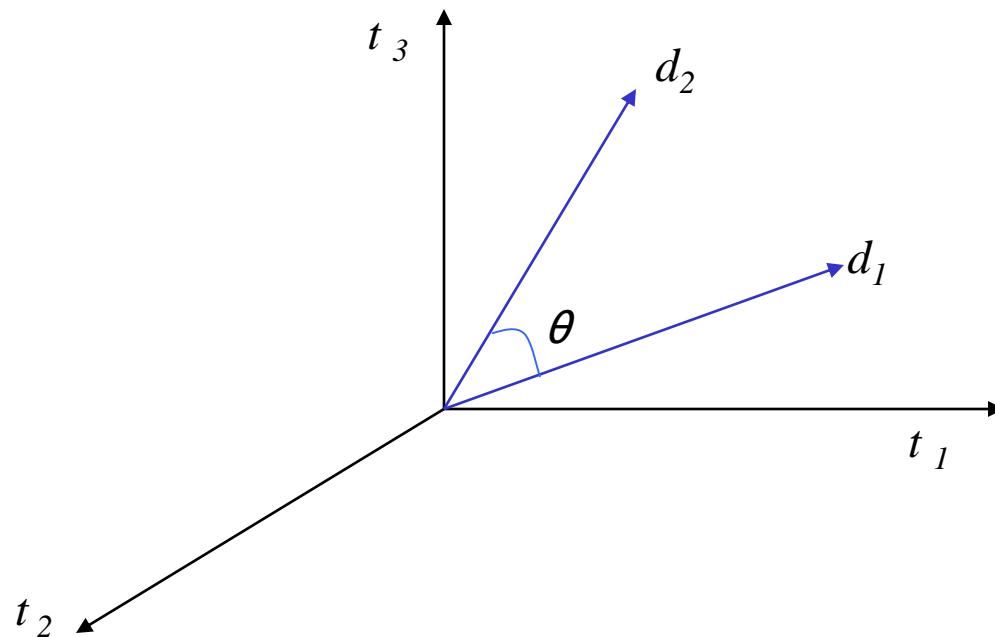
$$\text{sim}(D_1, Q) = 2*0 + 3*0 + 5*2 = 10$$

$$\text{sim}(D_2, Q) = 3*0 + 7*0 + 1*2 = 2$$

# Cosine similarity

---

- Distance between vectors  $d_1$  and  $d_2$  captured by the cosine of the angle  $x$  between them.
- Note – this is *similarity*, not distance



# Cosine similarity

---

$$sim(d_j, d_k) = \frac{\vec{d}_j \cdot \vec{d}_k}{\|\vec{d}_j\| \|\vec{d}_k\|} = \frac{\sum_{i=1}^n w_{i,j} w_{i,k}}{\sqrt{\sum_{i=1}^n w_{i,j}^2} \sqrt{\sum_{i=1}^n w_{i,k}^2}}$$

- Cosine of angle between two vectors
- The denominator involves the lengths of the vectors
- So the cosine measure is also known as the *normalized inner product*

$$\text{Length } |\vec{d}_j| = \sqrt{\sum_{i=1}^n w_{i,j}^2}$$

# Example

---

- Documents: Austen's *Sense and Sensibility*, *Pride and Prejudice*; Bronte's *Wuthering Heights*

	SaS	PaP	WH
<i>affection</i>	115	58	20
<i>jealous</i>	10	7	11
<i>gossip</i>	2	0	6

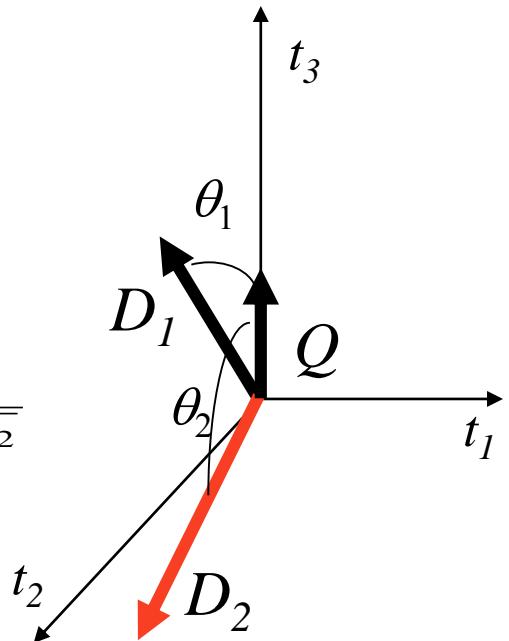
	SaS	PaP	WH
<i>affection</i>	0.996	0.993	0.847
<i>jealous</i>	0.087	0.120	0.466
<i>gossip</i>	0.017	0.000	0.254

- $\cos(\text{SAS}, \text{PAP}) = .996 \times .993 + .087 \times .120 + .017 \times 0.0 = 0.999$
- $\cos(\text{SAS}, \text{WH}) = .996 \times .847 + .087 \times .466 + .017 \times .254 = 0.929$

# Cosine Similarity Measure

- Cosine similarity measures the cosine of the angle between two vectors.
- Inner product normalized by the vector lengths.

$$\text{CosSim}(d_j, q) = \frac{\vec{d}_j \cdot \vec{q}}{|\vec{d}_j| \cdot |\vec{q}|} = \frac{\sum_{i=1}^t (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^t w_{ij}^2} \cdot \sqrt{\sum_{i=1}^t w_{iq}^2}}$$



$$D_1 = 2T_1 + 3T_2 + 5T_3 \quad \text{CosSim}(D_1, Q) = 10 / \sqrt{(4+9+25)(0+0+4)} = 0.81$$
$$D_2 = 3T_1 + 7T_2 + 1T_3 \quad \text{CosSim}(D_2, Q) = 2 / \sqrt{(9+49+1)(0+0+4)} = 0.13$$
$$Q = 0T_1 + 0T_2 + 2T_3$$

$D_1$  is 6 times better than  $D_2$  using cosine similarity but only 5 times better using inner product.

# Comments on Vector Space Models

---

- Simple, mathematically based approach.
- Considers both local ( $tf$ ) and global ( $idf$ ) word occurrence frequencies.
- Provides partial matching and ranked results.
- Tends to work quite well in practice despite obvious weaknesses.
- Allows efficient implementation for large document collections.

# Problems with Vector Space Model

---

- Missing semantic information (e.g. word sense).
- Missing syntactic information (e.g. phrase structure, word order, proximity information).
- Assumption of term independence (e.g. ignores synonymy).
- Lacks the control of a Boolean model (e.g., *requiring* a term to appear in a document).
  - Given a two-term query “A B”, may prefer a document containing A frequently but not B, over a document that contains both A and B, but both less frequently.

# Extended Boolean

---

## Basic Concepts

- Instead of binary values, terms in documents and queries have a weight (importance or some other statistical property)
- *Instead of binary set membership, sets are “fuzzy” and the weights are used to determine degree of membership.*
- Degree of set membership can be used to rank the results of a query

# Alternative Set Theoretic models

## Extended Boolean model

---

- Combination of Boolean and Vector
- In comparison with Boolean model, adds “distance from query”
  - some documents satisfy the query better than others
- In comparison with Vector model, adds the distinction between AND and OR combinations
- There is a parameter (degree of norm) allowing to adjust the behavior between Boolean-like and Vector-like
- This can be even different within one query
- Not investigated well. Why not investigate it?

*In the Extended Boolean model, a document is represented as a vector (similarly to in the vector model). Each dimension corresponds to a separate term associated with the document.*

---

*The weight of term  $K_x$  associated with document  $d_j$  is measured by its normalized Term frequency and can be defined as:*

$$w_{x,j} = f_{x,j} * \frac{Idf_x}{\max_i Idf_x}$$

*where  $Idfx$  is inverse document frequency.*

*The weight vector associated with document  $d_j$  can be represented as:*

$$\mathbf{v}_{d_j} = [w_{1,j}, w_{2,j}, \dots, w_{i,j}]$$

---

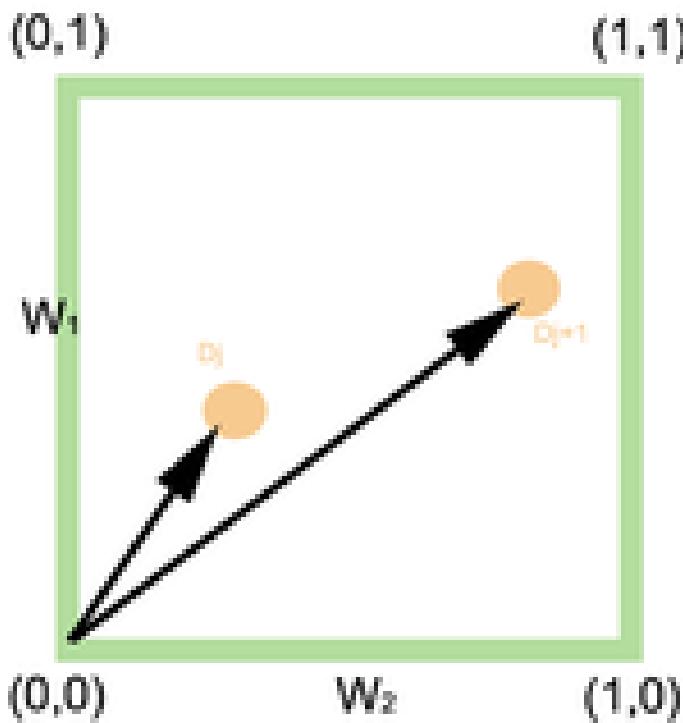
*Considering the space composed of two terms  $Kx$  and  $Ky$  only,  
the corresponding term weights are  $w1$  and  $w2$ .*

*Thus, for query  $q_{or} = (Kx \vee Ky)$ , we can calculate the similarity  
with the following formula:*

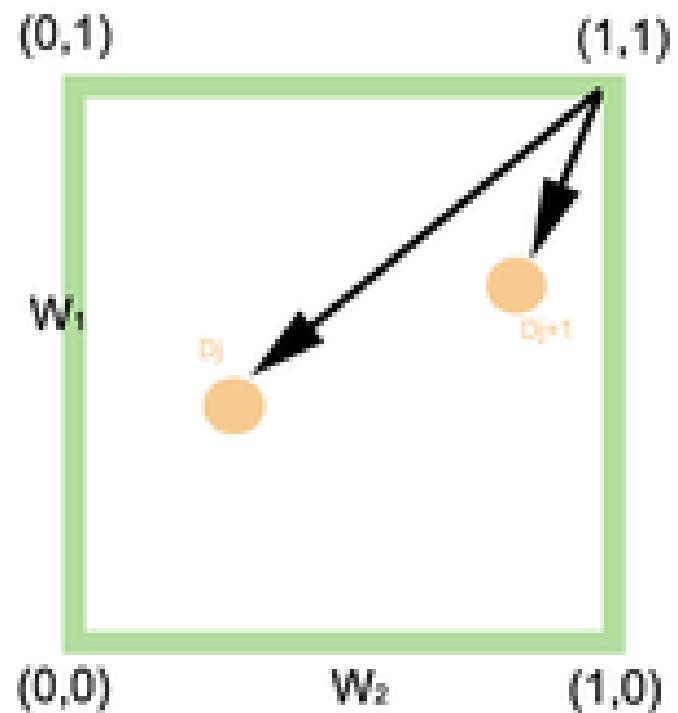
$$sim(q_{or}, d) = \sqrt{\frac{w_1^2 + w_2^2}{2}}$$

*For query  $q_{and} = (K_x \wedge K_y)$ , we can use:*

$$sim(q_{and}, d) = 1 - \sqrt{\frac{(1-w_1)^2 + (1-w_2)^2}{2}}$$



**Figure 1:** The similarities of  $q = (Kx \vee Ky)$  with documents  $dj$  and  $dj+1$ .



**Figure 2:** The similarities of  $q = (Kx \wedge Ky)$  with documents  $dj$  and  $dj+1$ .

---

### *Generalizing the idea and P-norms*

*We can generalize the previous 2D extended Boolean model example to higher t-dimensional space using Euclidean distances.*

*This can be done using P-norms which extends the notion of distance to include p-distances, where  $1 \leq p \leq \infty$  is a new parameter.*

*A generalized conjunctive query is given by:*

$$q_{or} = k_1 \vee^p k_2 \vee^p \dots \vee^p k_t$$

- 
- The similarity of  and  can be defined as:

$$sim(q_{or}, d_j) = \sqrt[p]{\frac{w_1^p + w_2^p + \dots + w_t^p}{t}}$$

A generalized disjunctive query is given by

$$q_{\text{and}} = k_1 \wedge^p k_2 \wedge^p \dots \wedge^p k_t$$

The similarity of



and



can be defined as :

$$\text{sim}(q_{\text{and}}, d_j) = 1 - \sqrt[p]{\frac{(1 - w_1)^p + (1 - w_2)^p + \dots + (1 - w_t)^p}{t}}$$

## Examples

---

Consider the query  $q = (K1 \wedge K2) \vee K3$ . The similarity between query  $q$  and document  $d$  can be computed using the formula:

$$sim(q, d) = \sqrt[p]{\frac{(1 - \sqrt[p]{(\frac{(1-w_1)^p + (1-w_2)^p}{2})})^p + w_3^p}{2}}$$

# Fuzzy Sets

---

- The set membership function can be, for example, the relative term frequency within a document, the IDF or any other function providing weights to terms
- This means that the fuzzy methods use sets of criteria for term weighting that are the same or similar to those used in other ranked retrieval methods (e.g., vector and probabilistic methods)

# Fuzzy Sets

---

Matching of a document to a query terms is approximate or vague

This **vagueness** can be modeled using a fuzzy framework, as follows:

- each query term defines a **fuzzy set**
- each doc has a **degree of membership** in this set

This interpretation provides the foundation for many IR models based on fuzzy theory

In here, we discuss the model proposed by  
Ogawa, Morita, and Kobayashi

# Fuzzy Sets

---

Fuzzy set theory deals with the representation of classes whose boundaries are not well defined

Key idea is to introduce the notion of a **degree of membership** associated with the elements of the class

This degree of membership varies from 0 to 1 and allows modelling the notion of **marginal membership**

Thus, membership is now a **gradual** notion, contrary to the crispy notion enforced by classic Boolean logic

# Fuzzy Set Theory

---

Introduced by Zadeh in 1965.

## Definition

- A fuzzy subset  $A$  of a universe of discourse  $U$  is characterized by a membership function  $\mu_A(u)$  which associate with each element  $u$  of  $U$  a number in the interval  $[0,1]$ .
- Set Theory:  $A=\{a, b, c\}$ . Subset of  $A$ :  $\{a, c\}$ .
- An element is either in a set or not in a set.  $\mu_A(u)$  is either 0 or 1.

$$\mu_A : U \rightarrow [0,1]$$

# Set Theory

---

- Let  $U$  be the set of all elements (universe)
- There are three basic operations:
  - $A \cup B = \{\text{elements in } A \text{ or in } B\}$ .
  - $A \cap B = \{\text{elements in both } A \text{ and } B\}$
  - Not  $A = U - A$ .

# Fuzzy Sets

---

- Definition *Let  $U$  be the universe of discourse,  $A$  and  $B$  be two fuzzy subsets of  $U$ , and  $\bar{A}$  be the complement of  $A$  relative to  $U$ . Also, let  $u$  be an element of  $U$ . Then,*

$$\mu_{\bar{A}} = 1 - \mu_A(u)$$

$$\mu_{A \cup B}(u) = \max\{\mu_A(u), \mu_B(u)\}$$

$$\mu_{A \cap B}(u) = \min\{\mu_A(u), \mu_B(u)\}$$

# Fuzzy Sets

---

- If we have three documents and three terms...
  - $D_1 = (.4, .2, 1)$ ,  $D_2 = (0, 0, .8)$ ,  $D_3 = (.7, .4, 0)$

*For search:*  $t_1 \cup t_2 \cup t_3$

$$v(D_1) = \max(.4, .2, 1) = 1$$

$$v(D_2) = \max(0, 0, .8) = .8$$

$$v(D_3) = \max(.7, .4, 0) = .7$$

*For search:*  $t_1 \cap t_2 \cap t_3$

$$v(D_1) = \min(.4, .2, 1) = .2$$

$$v(D_2) = \min(0, 0, .8) = 0$$

$$v(D_3) = \min(.7, .4, 0) = 0$$

# Fuzzy Sets

---

- Fuzzy set membership of term to document is  $f(A) \rightarrow [0,1]$

$$D_1 = \{(mesons, .8), (scattering, .4)\}$$

$$D_2 = \{(mesons, .5), (scattering, .6)\}$$

Query = MESONS AND SCATTERING

$$RSV(D_1) = \text{MIN}(.8,.4) = .4$$

$$RSV(D_2) = \text{MIN}(.5,.6) = .5$$

$D_2$  is ranked before  $D_1$  in the result set.

- However, consistent with the Boolean model:
  - Query =  $t_1 \cap t_2 \cap t_3 \cap \dots \cap t_{100}$
  - If D not indexed by  $t_1$  then it fails, even if D is indexed by  $t_2, \dots, t_{100}$

# Fuzzy Information Retrieval

---

We first set up *term-term correlation* matrix:

For terms  $k_i$  and  $k_l$ ,

$$C_{i,l} = \frac{n_{i,l}}{n_i + n_l - n_{i,l}}$$

Where  $n_i$  is the number of documents containing  $k_i$  ,

$n_l$  is the number of documents containing  $k_l$

And  $n_{i,l}$  is the number of documents containing both  $k_i$  and  $k_l$ .

**Note  $C_{i,i}=1$ .**

# Fuzzy Information Retrieval

---

We define a fuzzy set for each term  $k_i$ .

In the fuzzy set for  $k_i$ , a document  $d_j$  has a degree of membership

$$\mu_{ij} \text{ computed as } \mu_{i,j} = 1 - \prod_{k_l \in d_j} (1 - c_{i,l})$$

**Example:**

$$c_{1,2}=0.1, c_{1,3}=0.21.$$

$$D1=(0, 1, 1, 0). \mu_{1,1}= 1-0.9*0.79.$$

$$D2=(1, 0, 0, 0). \mu_{1,2}= 1-0. (since c_{1,1}=1.)$$

$$\text{How is } d3=(1, 0, 1, 0)?$$

# Fuzzy Information Retrieval

---

Whenever, the document  $d_j$  contains a term that is strongly related to  $k_i$ , then the document  $d_j$  is belong to the fuzzy set of term  $k_i$ , i.e.,

$\mu_{i,j}$  is very close to 1.

Example,  $c_{1,2}=0.9$ ,  
 $d1=(0, 1, 0, 0)$ .

$$\mu_{1,1} = 1 - (1 - 0.9) = 0.9$$

## Query:

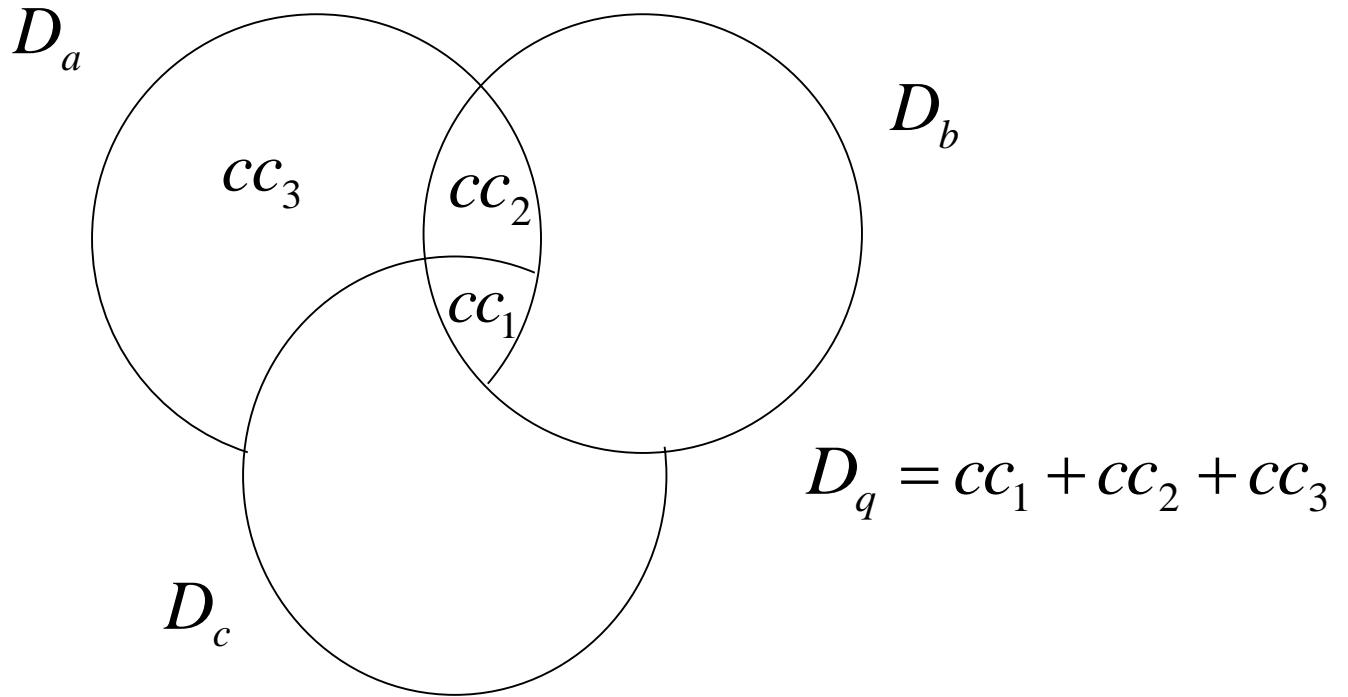
---

- Query is a Boolean formula, e.g.,
- $q = K_a \text{ and } (K_b \text{ or not } K_c)$ .

$$q = k_a \wedge (k_b \vee \neg k_c)$$

- $q = (1, 1, 1)$  or  $(1, 1, 0)$  or  $(1, 0, 0)$ .
- Suppose  $q$  is

$$\xrightarrow{} q_{dnf} = cc_1 \vee cc_2 \vee \dots \vee cc_p$$



*Figure 1. Fuzzy document sets for the query  $[q = k_a \wedge (k_b \vee \neg k_c)]$  Each  $cc_i, i \in \{1,2,3\}$ , is a conjunctive component.  $D_q$  is the query fuzzy set.*

---


$$\begin{aligned}
\mu_{q,j} &= \mu_{cc_1+cc_2+cc_3,j} \\
&= 1 - \prod_{i=1}^3 (1 - \mu_{cc_i,j}) \\
&= 1 - (1 - \mu_{a,j} \mu_{b,j} \mu_{c,j}) \times \\
&\quad (1 - \mu_{a,j} \mu_{b,j} (1 - \mu_{c,j})) \times (1 - \mu_{a,j} (1 - \mu_{b,j}) (1 - \mu_{c,j}))
\end{aligned}$$

Where  $\mu_{i,j}, i \in \{a, b, c\}$ , is the membership of  $d_j$  in the fuzzy set associated with  $k_i$ ,  $\mu_{q,j}$  is the membership of document  $j$  for query  $q$ .

# Example - Fuzzy Set Model

---

- Q: “**gold silver truck**”  
D1:“Shipment of **gold** damaged in a fire”  
D2:“Delivery of **silver** arrived in a **silver truck**”  
D3:“Shipment of **gold** arrived in a **truck**”

# Eg:::: Fuzzy Set Model

---

- Q: “gold silver truck”
  - D1: “Shipment of gold damaged in a fire”
  - D2: “Delivery of silver arrived in a silver truck”
  - D3: “Shipment of gold arrived in a truck”
- IDF (Select Keywords)
  - a = in = of = 0 =  $\log^{3/3}$
  - arrived = gold = shipment = truck = 0.176 =  $\log^{3/2}$
  - damaged = delivery = fire = silver = 0.477 =  $\log^{3/1}$
- 8 Keywords (Dimensions) are selected
  - arrived(1), damaged(2), delivery(3), fire(4), gold(5), silver(6), shipment(7), truck(8)

# Fuzzy Set Model

---

$$\begin{aligned}\mu_{\text{gold}, d1} &= 1 - \prod_{k_1 \in d_1} (1 - C_{\text{gold}, k_1}) \\&= 1 - (1 - C_{\text{gold, shipment}}) * (1 - C_{\text{gold, gold}}) * (1 - C_{\text{gold, damaged}}) * (1 - C_{\text{gold, fire}}) \\&= 1 - (1 - \frac{2}{2+2-2}) * (1 - \frac{1}{2+1-1}) * (1 - \frac{2}{2+2-2}) * (1 - \frac{2}{2+1-1}) \\&= 1 - 0 * \frac{1}{2} * 0 * \frac{1}{2} \\&= 1\end{aligned}$$

$$\mu_{\text{silver}, d1} = 1 - 1 * 1 * 1 * 1 = 0$$

$$\begin{aligned}\mu_{\text{truck}, d1} &= 1 - \prod_{k_1 \in d_1} (1 - C_{\text{truck}, k_1}) \\&= 1 - (1 - C_{\text{truck, shipment}}) * (1 - C_{\text{truck, gold}}) * (1 - C_{\text{truck, damaged}}) * (1 - C_{\text{truck, fire}}) \\&= 1 - (1 - \frac{1}{2+2-1}) * (1 - \frac{1}{2+2-1}) * (1 - \frac{0}{2+1-0}) * (1 - \frac{0}{2+1-0}) \\&= 1 - \frac{2}{3} * \frac{2}{3} * 1 * 1 \\&= \frac{5}{9}\end{aligned}$$

# Fuzzy Set Model

---

$$\mu_{\text{gold},d2} = 1 - 1 * 1 * \frac{2}{3} * \frac{2}{3} = \frac{5}{9}$$

$$\mu_{\text{silver},d2} = 1$$

$$\mu_{\text{truck},d2} = 1$$

$$\mu_{\text{gold},d3} = 1$$

$$\mu_{\text{silver},d3} = 1 - 1 * 1 * \frac{1}{2} * \frac{1}{2} = \frac{3}{4}$$

$$\mu_{\text{truck},d3} = 1$$

# Fuzzy Set Model

- $\text{Sim}(q,d)$ : Alternative 1

$$\mu_{q,d1} = \mu_{\text{gold} \wedge \text{silver} \wedge \text{truck},d1} = \mu_{\text{gold},d1} * \mu_{\text{silver},d1} * \mu_{\text{truck},d1} = 0$$

$$\mu_{q,d2} = \mu_{\text{gold} \wedge \text{silver} \wedge \text{truck},d1} = \mu_{\text{gold},d2} * \mu_{\text{silver},d2} * \mu_{\text{truck},d2} = \frac{5}{9}$$

$$\mu_{q,d3} = \mu_{\text{gold} \wedge \text{silver} \wedge \text{truck},d1} = \mu_{\text{gold},d3} * \mu_{\text{silver},d3} * \mu_{\text{truck},d3} = \frac{3}{4}$$

$\text{Sim}(q,d_3) > \text{Sim}(q,d_2) > \text{Sim}(q,d_1)$

- $\text{Sim}(q,d)$ : Alternative 2

$$\mu_{q,d1} = \mu_{\text{gold} \wedge \text{silver} \wedge \text{truck},d1} = \min(\mu_{\text{gold},d1}, \mu_{\text{silver},d1}, \mu_{\text{truck},d1}) = 0$$

$$\mu_{q,d2} = \mu_{\text{gold} \wedge \text{silver} \wedge \text{truck},d1} = \min(\mu_{\text{gold},d2}, \mu_{\text{silver},d2}, \mu_{\text{truck},d2}) = \frac{5}{9}$$

$$\mu_{q,d3} = \mu_{\text{gold} \wedge \text{silver} \wedge \text{truck},d1} = \min(\mu_{\text{gold},d3}, \mu_{\text{silver},d3}, \mu_{\text{truck},d3}) = \frac{3}{4}$$

$\text{Sim}(q,d_3) > \text{Sim}(q,d_2) > \text{Sim}(q,d_1)$

# Fuzzy set Model

---

- Advantages
  - The correlations among index terms are considered
  - Degree of relevance between queries and docs can be achieved
- Disadvantages
  - Fuzzy IR models have been discussed mainly in the literature associated with fuzzy theory
  - Experiments with standard test collections are not available
  - Do not consider the frequency (or counts) of a term in a document or a query

# Limitation of Fuzzyset

---

- Fuzzy sets suffer from the same kind of lack of discrimination among the retrieval results almost to the same extent as standard Boolean
- The rank of a document depends entirely on the lowest or highest weighted term in an AND or OR operation

# Example

---

- Q: “**gold silver truck**”  
D1:“Shipment of **gold** damaged in a fire”  
D2:“Delivery of **silver** arrived in a **silver truck**”  
D3:“Shipment of **gold** arrived in a **truck**”

**Solve it with Vector Space model**

# Vector Space Model

---

- Q: “gold silver truck”  
D1: “Shipment of gold damaged in a fire”  
D2: “Delivery of silver arrived in a silver truck”  
D3: “Shipment of gold arrived in a truck”
- 8 Dimensions (arrived, damaged, delivery, fire, gold, silver, shipment, truck)

- Weight = TF \* IDF
- $Q = (0, 0, 0, 0, .176, .477, 0, .176)$

$$D1 = (0, .477, 0, .477, .176, 0, .176, 0)$$

$$D2 = (.176, 0, .477, 0, 0, .954, 0, .176)$$

$$D3 = (.176, 0, 0, 0, .176, 0, .176, .176)$$

*Construction of  
Matrix T*

# Probabilistic Model

---

- **Objective:** to capture the IR problem using a probabilistic framework
- Given a user query, there is an *ideal* answer set
- Querying as specification of the properties of this ideal answer set (clustering)
  - Guess at the beginning what they could be (i.e., guess initial description of ideal answer set)
  - Improve by iteration

# *Probabilistic Model*

---

- An initial set of documents is retrieved somehow
- User inspects these docs looking for the relevant ones (in truth, only top 10-20 need to be inspected)
- IR system uses this information to refine description of ideal answer set
- By repeating this process, it is expected that the description of the ideal answer set will improve
- Have always in mind the need to guess at the very beginning the description of the ideal answer set
- Description of ideal answer set is modeled in probabilistic terms

# Probabilistic model

---

- Asks the question “what is probability that user will see **relevant** information if they read **this document**”
  - $P(\text{rel}|di)$  – **probability** of relevance after reading  $di$
  - How **likely** is the user to get relevance information from reading this document
  - **high** probability means more likely to get relevant info.

# *Probability Ranking Principle*

---

- 1. Collection of documents**
- 2. Representation of documents**
- 3. User uses a query**
- 4. Representation of query**
- 5. A set of documents to return**

**Question:** In what order documents to present to user?

**Logically:** Best document first and then next best and so on

**Requirement:** A formal way to judge the goodness of documents with respect to query

**Possibility:** Probability of relevance of the document with respect to query

# Probability Basics

## **Bayes' Rule**

Let  $a$  and  $b$  are two events

$$p(a | b) p(b) = p(a \cap b) = p(b | a) p(a)$$

$$p(a | b) = \frac{p(b | a) p(a)}{p(b)}$$

$$p(\bar{a} | b) = \frac{p(b | \bar{a}) p(\bar{a})}{p(b)}$$

Odds of an event  $a$  is defined as

$$O(a) = \frac{p(a)}{p(\bar{a})} = \frac{p(a)}{1 - p(a)}$$

# **Probabilistic Ranking Principle**

---

- Given a user query  $q$  and a document  $d_j$ , the probabilistic model tries to estimate the probability that the user will find the document  $d_j$  interesting (i.e., relevant).
- The model assumes that this probability of relevance depends on the query and the document representations only.
- Ideal answer set is referred to as  $R$  and should maximize the probability of relevance. Documents in the set  $R$  are predicted to be relevant.

# Probability Ranking Principle

Let  $x$  be a document in the collection.

Let  $R$  represent **relevance** of a document w.r.t. given (fixed) query and let  $NR$  represent **non-relevance**.

$p(R/x)$  - probability that a retrieved document  $x$  is **relevant**.

$p(NR/x)$  - probability that a retrieved document  $x$  is **non-relevant**.

$$p(R | x) = \frac{p(x | R) p(R)}{p(x)}$$

$$p(NR | x) = \frac{p(x | NR) p(NR)}{p(x)}$$

$p(R), p(NR)$  - prior probability of retrieving a relevant and non-relevant document respectively

$p(x|R), p(x|NR)$  - probability that if a relevant (non-relevant) document is retrieved, it is  $x$ .

# *Probability Ranking Principle*

---

$$p(R | x) = \frac{p(x | R)p(R)}{p(x)}$$

$$p(NR | x) = \frac{p(x | NR)p(NR)}{p(x)}$$

*Ranking Principle (Bayes' Decision Rule):*

*If  $p(R/x) > p(NR/x)$  then  $x$  is relevant,  
otherwise  $x$  is not relevant*

$$sim(d_j, q) = \frac{P(R|\vec{d}_j)}{P(\overline{R}|\vec{d}_j)}$$

# Probabilistic model

---

- “if a reference retrieval system's response to each request is a **ranking** of the documents in the collection in order of decreasing **probability of relevance** ... the overall **effectiveness** of the system to its user will be the **best** that is obtainable...”, **Probability ranking principle**
  - Rank documents based on **decreasing** probability of relevance to user
  - Calculate  $P(\text{rel}/d_i)$  for each document and rank
- Suggests **mathematical** approach to IR and matching
  - Can predict (somewhat) good retrieval **models** based on their **estimates** of  $P(\text{rel}/d_i)$

# Probabilistic Models

---

- Most probabilistic models based on combining probabilities of relevance and non-relevance of individual terms
  - Probability that a term will appear in a relevant document
  - Probability that the term will *not appear in a non-relevant document*
- These probabilities are estimated based on counting term appearances in document descriptions

# Example

---

- Assume we have a collection of 100 documents
  - $N=100$
- 20 of the documents contain the term  $UPES$ 
  - $n_{UPES} = 20$
- Searcher has read and marked 10 documents as relevant
  - $R=10$
- Of these relevant documents 5 contain the term  $UPES$ 
  - $r_{UPES} = 5$
- How important is the word  $UPES$  *to the searcher?*

# Probability of Relevance

- from these four numbers we can *estimate probability of UPES given relevance information*
  - i.e. how **important** term *UPES* is to *relevant documents*

$$\frac{\# \text{ of relevant docs which contain UPES}}{\# \text{ of relevant docs which don't contain UPES}} = \frac{r_{UPES}}{(R - r_{UPES})}$$

- is number of relevant documents containing **UPES** (5)
- is number of relevant documents that do not contain **UPES** (5)
- Eq. (I) is
  - higher if most relevant documents contain UPES
  - lower if most relevant documents do not contain UPES
  - high** value means UPES is **important term** to user in our example (5/5=1)

# Probability of Non-relevance

- Also we can estimate probability of UPES given *non-relevance* information
  - i.e. how important term UPES is to *non-relevant documents*

$$\frac{\# \text{ of non-relevant docs which contain UPES}}{\# \text{ of docs which don't contain UPES}} = \frac{(n_{UPES} - r_{UPES})}{(N - n_{UPES}) - (R - r_{UPES})}$$

- II

- $n_{UPES} - r_{UPES}$  is number of non-relevant documents that contain term sausage (20-5)=15
- $N - n_{UPES} - R + r_{UPES}$  is number of non-relevant documents that do not contain sausage (100-20-10+5)=75

- 
- Eq(II)
    - higher if more documents containing term UPES are non-relevant
    - lower if more documents that do not contain UPES are non-relevant
    - low value means UPES is important term to user in our example ( $15/75=0.2$ )

# F4 reweighting formula

---

- F4 gives new weights to all terms in collection (or just query)
  - high weights to **important** terms
  - Low weights to **unimportant** terms
  - replaces *idf, tf, or any other weights*
  - document score is based on sum of **query** terms in documents

$$\text{Similarity}(d_j, q) = \sum_{i=1}^n F4_{qi}$$

# F4 reweighting formula

---

how important is *UPES being present in relevant documents*

$$\log \left( \frac{\frac{r_{UPES} + 0.5}{0.5 + R - r_{UPES}}}{\frac{0.5 + n_{UPES} - r_{UPES}}{0.5 + N - n_{UPES} - R + r_{UPES}}} \right)$$

how important is *UPES being absent from non-relevant documents*

# Probabilistic model

---

- can also use to **rank** terms for **addition** to query
  - rank terms in *relevant documents by term* reweighting formula
  - i.e. by how **good** the terms are at retrieving relevant documents
    - add all terms
    - add some, e.g. top 5
- in probabilistic formula query expansion and term reweighting done **separately**

# Probabilistic model

---

- Advantages over vector-space
  - Strong theoretical basis
  - Based on probability theory (very well understood)
  - Easy to extend
- Disadvantages
  - Models are often complicated
  - No term frequency weighting
- Which is better vector-space or probabilistic?
  - Both are approximately as good as each other
  - Depends on collection, query, and other factors

# More IR models

---

- Alternative Set and Vector Models
  - Set-Based Model
  - Extended Boolean Model
  - Fuzzy Set Model
  - The Generalized Vector Model
  - Latent Semantic Indexing
  - Neural Network for IR

# More IR models

---

- Alternative Algebraic Models
  - Generalized Vector Model
  - Latent Semantic Indexing
  - Neural Network Model
- Part III: Alternative Probabilistic Models
  - BM25
  - Language Models
  - Divergence from Randomness
  - Belief Network Models
  - Other Models

# More IR models

---

- Other Models
  - Hypertext Model
  - Web-based Models
  - Structured Text Retrieval
  - Multimedia Retrieval
  - Enterprise and Vertical Search

# Structured Text Retrieval

---

- All the IR models discussed here treat the text as a string with no particular structure
- However, information on the structure might be important to the user for particular searches
  - Ex: retrieve a book that contains a figure of the Eiffel tower in a section whose title contains the term “France”
  - Classical information model – Eiffel tower and France
- The solution to this problem is to take advantage of the text structure of the documents to improve retrieval

# Early Text Retrieval Models

---

- We discuss now two early structured text retrieval models
- We use:
  - The term **match point** to refer to the position in the text of a word which matches the user query
  - The term **region** to refer to a contiguous portion of the text
  - The term **node** to refer to a structural component of the document

# Structured Text Retrieval

---

- Keyword-based query answering considers that the documents are flat
  - a word in the title has the same weight as a word in the body of the document
- Document structure is one additional piece of information which can be taken advantage of
  - For instance, words appearing in the title or in sub-titles within the document could receive higher importance

# Structured Text Retrieval

---

- Consider the following information need:
  - Retrieve all documents which contain a page in which the string “atomic holocaust” appears in italic in the text surrounding a Figure whose label contains the word earth
- The corresponding query could be:
  - **same-page( near( italic(“atomic holocaust”),  
Figure( label( “earth” ))))**
- Advanced interfaces that facilitate the specification of the structure are also highly desirable
- *Models which allow combining information on text content with information on document structure are called structured text models*
- Structured text models include no ranking (open research problem)

# Basic Definitions

---

- **Match point:** the position in the text of a sequence of words that match the query
  - Query: “atomic holocaust in Hiroshima”
  - Doc dj: contains 3 lines with this string
  - Then, doc dj contains 3 match points
- **Region:** a contiguous portion of the text
- **Node:** a structural component of the text such as a chapter, a section, etc

# Structured Text Retrieval

---

- Two type
  - Model based on Non-Overlapping Lists
  - Proximal Nodes Model

# Model basd on Non-Overlapping Lists

---

- Due to Burkowski, 1992.
- Idea: **divide the text in non-overlapping regions which are collected in a list**
- Multiple ways to divide the text in non-overlapping parts yield multiple lists:
  - a list for chapters
  - a list for sections
  - a list for subsections
- Text regions from distinct lists might overlap

# Non-Overlapping Lists

---

$L_0$



$L_1$



$L_2$



$L_3$



# Non-Overlapping Lists

---

- Implementation:
  - single inverted file that combines keywords and text regions
  - to each entry in this inverted file is associated a list of text regions
  - lists of text regions can be merged with lists of keywords

# Non-Overlapping Lists

---

- Regions are non-overlapping which limits the queries that can be asked
- Types of queries:
  - select a region that contains a given word
  - select a region A that does not contain a region B (regions A and B belong to distinct lists)
  - select a region not contained within any other region

## Comment : Non-Overlapping Lists

---

- Simple and allows efficient implementation
- Limited types of queries can be asked
- Model does not include any provision for ranking the documents by degree of similarity to the query

# Proximal Nodes

---

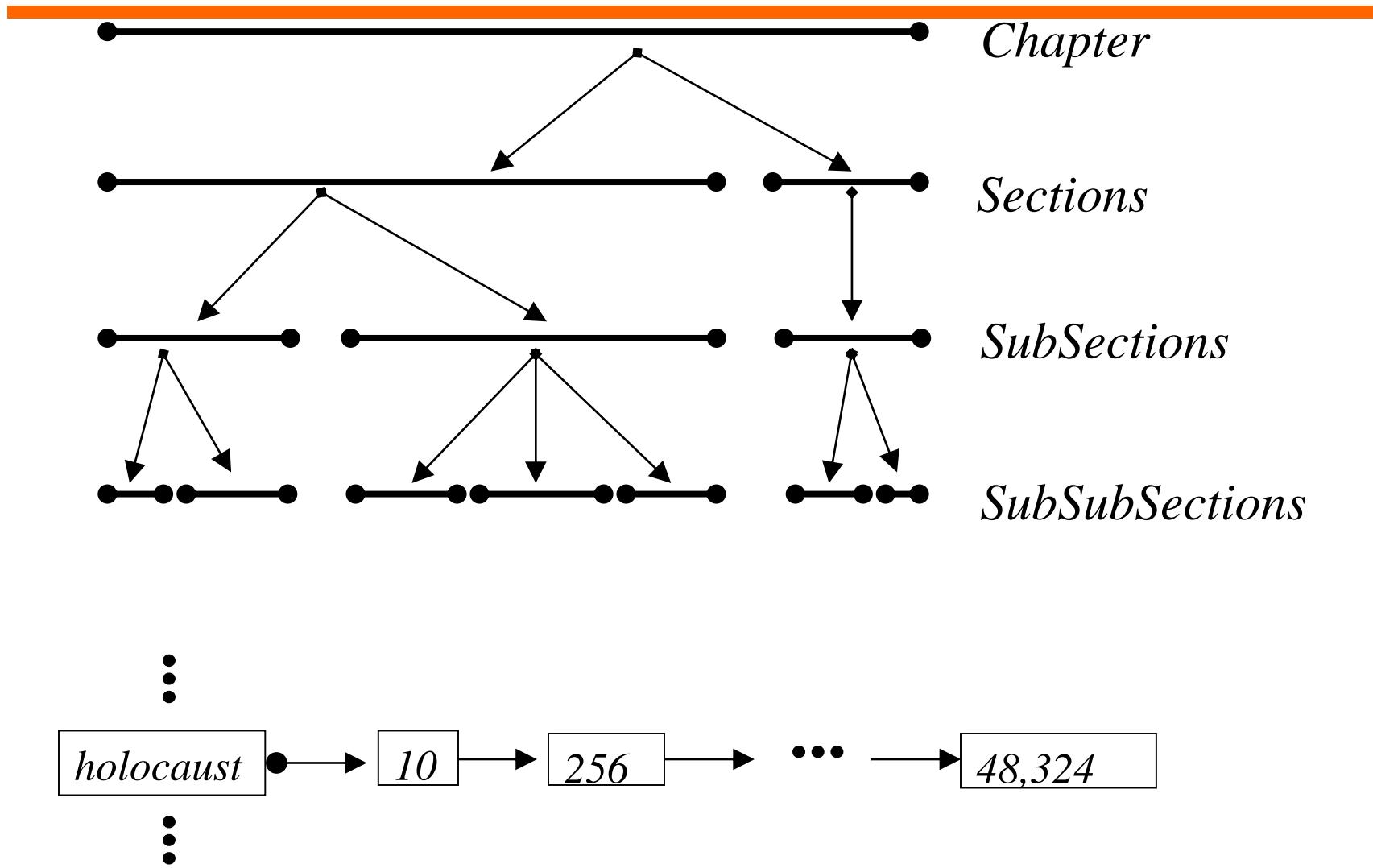
- Navarro and Baeza-Yates, 1997
- Idea: define a strict hierarchical index over the text. This enriches the previous model that used flat lists.
- Multiple index hierarchies might be defined
  - allows defining independent hierarchical indexing structures over the same document text
- Two distinct index hierarchies might refer to text regions that overlap

# Definitions

---

- Each indexing structure is a strict hierarchy composed of
  - chapters
  - sections
  - subsections
  - paragraphs
  - lines
- Each of these components is called a node
- To each node is associated a text region

# Proximal Nodes



# Overlapped Lists

---

- The original idea was to have a lists of disjoint segments, originated by textual searches or by “regions” like chapters.
- It enhances the algebra with overlapping capabilities, some new operators and a framework for an implementation.
- The new operators allow to perform set union, and to combine areas.
- Combination means selecting the minimal text areas including two segments, for any two segments taken from two sets.

# Lists of References

---

- Structure of documents is hierarchical (with only one strict hierarchy),
  - answers to queries are at (only the top-level elements qualify), and all elements must be from the same type (e.g. only sections, or only paragraphs).
- Answers to queries are seen as lists of “references”.
  - A reference is a pointer to a region of the database.
  - Integrates in an elegant way answers to queries and hypertext links, since all are lists of references.

# Lists of References

---

- The model has also navigational features to traverse those lists.
- This model is very powerful, and because of this, has efficiency problems.
- To make the model suitable for our comparisons, we consider only the portion related to querying structures.
- Even this portion is quite powerful, and allows to efficiently solve queries by first locating the text matches and then filtering the candidates with the structural restrictions.

# Proximal Nodes

---

- Key points:
  - In the hierarchical index, one node might be contained within another node
  - But, two nodes of a same hierarchy cannot overlap
  - The inverted list for keywords complements the hierarchical index
  - The implementation here is more complex than that for non-overlapping lists

# Proximal Nodes

---

- Queries are now regular expressions:
  - search for strings
  - references to structural components
  - combination of these
- Model is a compromise between expressiveness and efficiency
- Queries are simple but can be processed efficiently
- Further, model is more expressive than non-overlapping lists

# Proximal Nodes

---

- Query: find the sections, the subsections, and the subsubsections that contain the word “holocaust”
  - [(\*section) with (“holocaust”)]
- Simple query processing:
  - traverse the inverted list for “holocaust” and determine all match points
  - use the match points to search in the hierarchical index for the structural components

# Proximal Nodes

---

- Query: [(\*section) with (“holocaust”)]
- Sophisticated query processing:
  - get the first entry in the inverted list for “holocaust”
  - use this match point to search in the hierarchical index for the structural components
  - Innermost matching component: smaller one
  - Check if innermost matching component includes the second entry in the inverted list for “holocaust”
  - If it does, check the third entry and so on
  - This allows matching efficiently the nearby (or proximal) nodes

# Proximal Nodes

---

- Model allows formulating queries that are more sophisticated than those allowed by non-overlapping lists
- To speed up query processing, nearby nodes are inspected
- Types of queries that can be asked are somewhat limited (all nodes in the answer must come from a same index hierarchy!)
- Model is a compromise between efficiency and expressiveness

# Evaluation measures

---

- The quality of many retrieval systems depends on how well they manage to rank relevant documents.
- How can we evaluate rankings in IR?
  - IR researchers have developed evaluation measures specifically designed to evaluate rankings.
  - Most of these measures combine **precision** and **recall** in a way that takes account of the ranking.

# Conclusion

---

- No model is the best for all applications, especially because the more expressive the model, the less efficient can it be.
- Each application has its own set of requirements, and should select the most efficient model supporting them.
- Another important issue is the perspective of the user. When we incorporate operators and evaluate the cost of implementing them, we are implicitly assuming that they are useful for the user of the system.

# Step 1: Preprocessing

---

- Implement the preprocessing functions:
  - For tokenization
  - For stop word removal
  - For stemming
- Input: Documents that are read one by one from the collection
- Output: Tokens to be added to the index
  - No punctuation, no stop-words, stemmed

## Step 2: Indexing

---

- Build an inverted index, with an entry for each word in the vocabulary
- Input: Tokens obtained from the preprocessing module
- Output: An inverted index for fast access

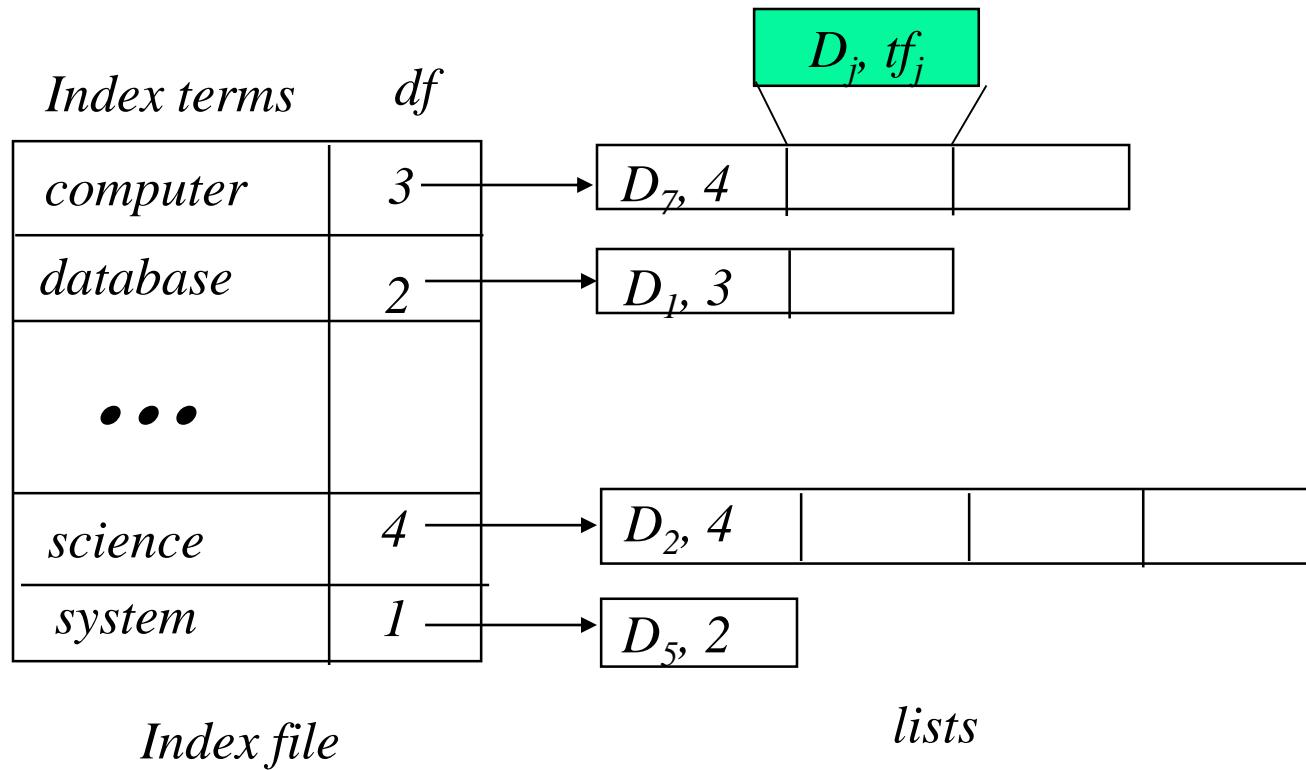
## Step 2 (cont'd)

---

- Many data structures are appropriate for fast access
  - B-trees, skipped lists, hashtables
- We need:
  - One entry for each word in the vocabulary
  - For each such entry:
    - Keep a list of all the documents where it appears together with the corresponding frequency → TF
  - For each such entry, keep the total number of occurrences in all documents:
    - → IDF

## Step 2 (cont'd)

---



## Step 2 (cont' d)

---

- TF and IDF for each token can be computed in one pass
- Cosine similarity also required document lengths
- Need a second pass to compute document vector lengths
  - Remember that the length of a document vector is the square-root of sum of the squares of the weights of its tokens.
  - Remember the weight of a token is:  $TF * IDF$
  - Therefore, must wait until IDF's are known (and therefore until all documents are indexed) before document lengths can be determined.
- Do a second pass over all documents: keep a list or hashtable with all document id-s, and for each document determine its length.

# Time Complexity of Indexing

---

- Complexity of creating vector and indexing a document of  $n$  tokens is  $O(n)$ .
- So indexing  $m$  such documents is  $O(m n)$ .
- Computing token IDFs can be done during the same first pass
- Computing vector lengths is also  $O(m n)$ .
- Complete process is  $O(m n)$ , which is also the complexity of just reading in the corpus.

## Step 3: Retrieval

---

- Use inverted index (from step 2) to find the limited set of documents that contain at least one of the query words.
  - Incrementally compute cosine similarity of each indexed document as query words are processed one by one.
  - To accumulate a total score for each retrieved document, store retrieved documents in a hashtable, where the document id is the key, and the partial accumulated score is the value.
- 
- Input: Query and Inverted Index (from Step 2)
  - Output: Similarity values between query and documents

## Step 4: Ranking

---

- Sort the hashtable including the retrieved documents based on the value of cosine similarity
  - sort  $\{ \$\text{retrieved}\{\$b\} \Leftrightarrow \$\text{retrieved}\{\$a\} \}$  keys %retrieved
- Return the documents in descending order of their relevance
- Input: Similarity values between query and documents
- Output: Ranked list of documents in reversed order of their relevance

Here is a simplified example of the vector space retrieval model. Consider a very small collection C that consists in the following three documents:

- d1: “new york times”
- d2: “new york post”
- d3: “los angeles times”

Some terms appear in two documents, some appear only in one document. The total number of documents is  $N=3$ . Therefore, the  $idf$  values for the terms are:

angles	$\log_2(3/1)=1.584$
los	$\log_2(3/1)=1.584$
new	$\log_2(3/2)=0.584$
post	$\log_2(3/1)=1.584$
times	$\log_2(3/2)=0.584$
york	$\log_2(3/2)=0.584$

For all the documents, we calculate the  $tf$  scores for all the terms in C. We assume the words in the vectors are ordered alphabetically.

	angeles	los	new	post	times	york
d1	0	0	1	0	1	1
d2	0	0	1	1	0	1
d3	1	1	0	0	1	0

Now we multiply the  $tf$  scores by the  $idf$  values of each term, obtaining the following matrix of documents-by-terms: (All the terms appeared only once in each document in our small collection, so the maximum value for normalization is 1.)

	angeles	los	new	post	times	york
d1	0	0	0.584	0	0.584	0.584
d2	0	0	0.584	1.584	0	0.584
d3	1.584	1.584	0	0	0.584	0

Given the following query: “new new times”, we calculate the  $tf-idf$  vector for the query, and compute the score of each document in C relative to this query, using the cosine similarity measure. When computing the  $tf-idf$  values for the query terms we divide the frequency by the maximum frequency (2) and multiply with the  $idf$  values.

q	0	0	$(2/2)*0.584=0.584$	0	$(1/2)*0.584=0.292$	0
---	---	---	---------------------	---	---------------------	---

We calculate the length of each document and of the query:

$$\text{Length of } d_1 = \sqrt{0.584^2 + 0.584^2 + 0.584^2} = 1.011$$

$$\text{Length of } d_2 = \sqrt{0.584^2 + 1.584^2 + 0.584^2} = 1.786$$

$$\text{Length of } d_3 = \sqrt{1.584^2 + 1.584^2 + 0.584^2} = 2.316$$

$$\text{Length of } q = \sqrt{0.584^2 + 0.292^2} = 0.652$$

Then the similarity values are:

$$\text{cosSim}(d_1, q) = (0*0+0*0+0.584*0.584+0*0+0.584*0.292+0.584*0) / (1.011*0.652) = 0.776$$

$$\text{cosSim}(d_2, q) = (0*0+0*0+0.584*0.584+1.584*0+0*0.292+0.584*0) / (1.786*0.652) = 0.292$$

$$\text{cosSim}(d_3, q) = (1.584*0+1.584*0+0*0.584+0*0+0.584*0.292+0*0) / (2.316*0.652) = 0.112$$

According to the similarity values, the final order in which the documents are presented as result to the query will be:  $d_1, d_2, d_3$ .