Web Content Mining

Generalities, Preprocessing and Word Associations

— Session 1 —

Text Mining and Analytics

- Text mining ~ Text Analytics
- Turn text data into high-quality information or actionable knowledge.
  - Minimizes human effort (on consuming text data)
  - Supplied knowledge for optimal decision making
- Related to text retrieval, which is an essential component in any text mining system
  - Text retrieval can be a preprocessor for text mining
  - Text retrieval is needed for knowledge provenance

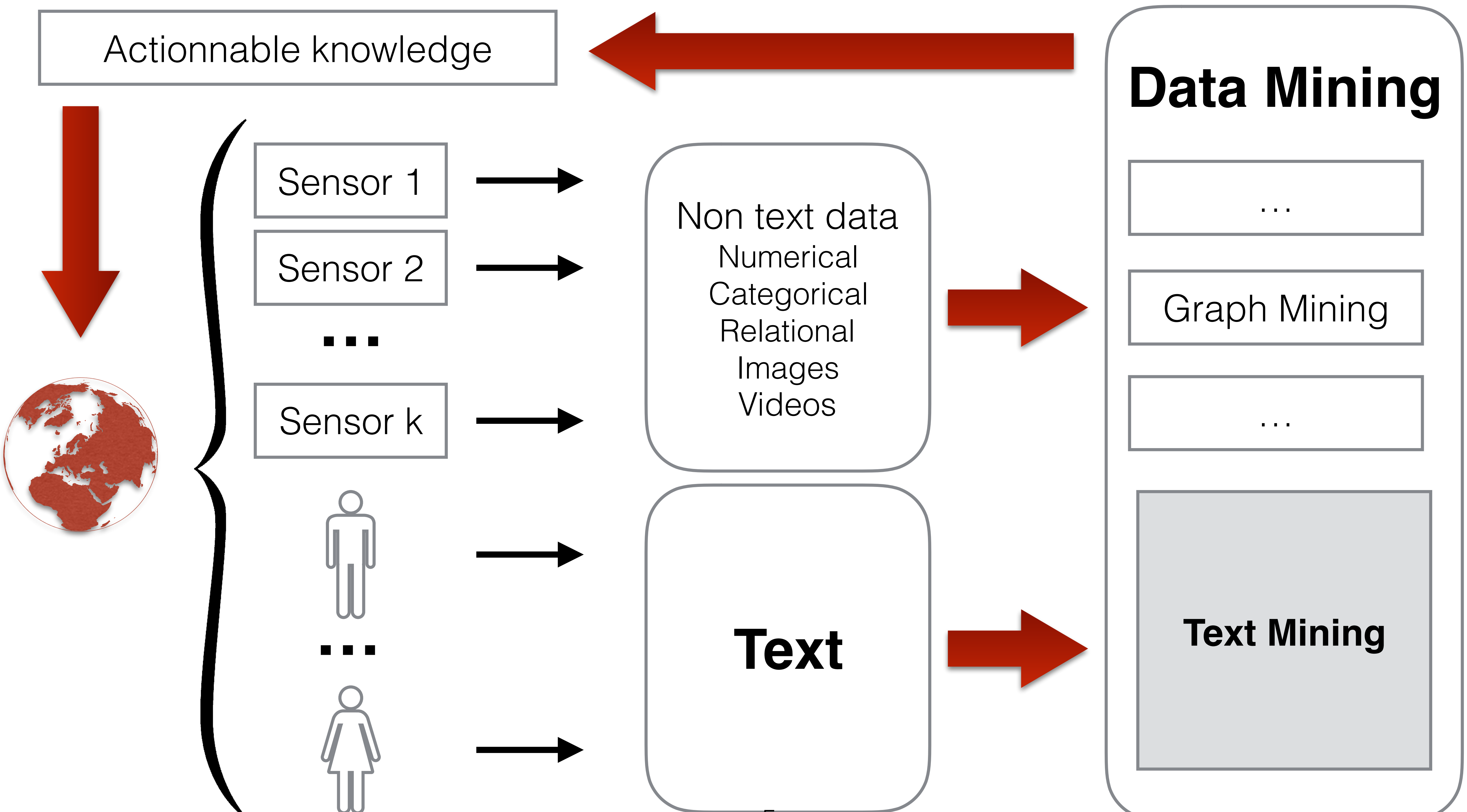
Why Text Mining is hard?

- I ate pizza with friends = I ate pizza with olives?
  - Only one different word
  - Friends = Olives ?
- I ate pizza with friends = friends and I shared some pizza?
  - 3 similar words
  - Different verb
  - 3/5 vs 3/6 words
  - Similar semantic sense

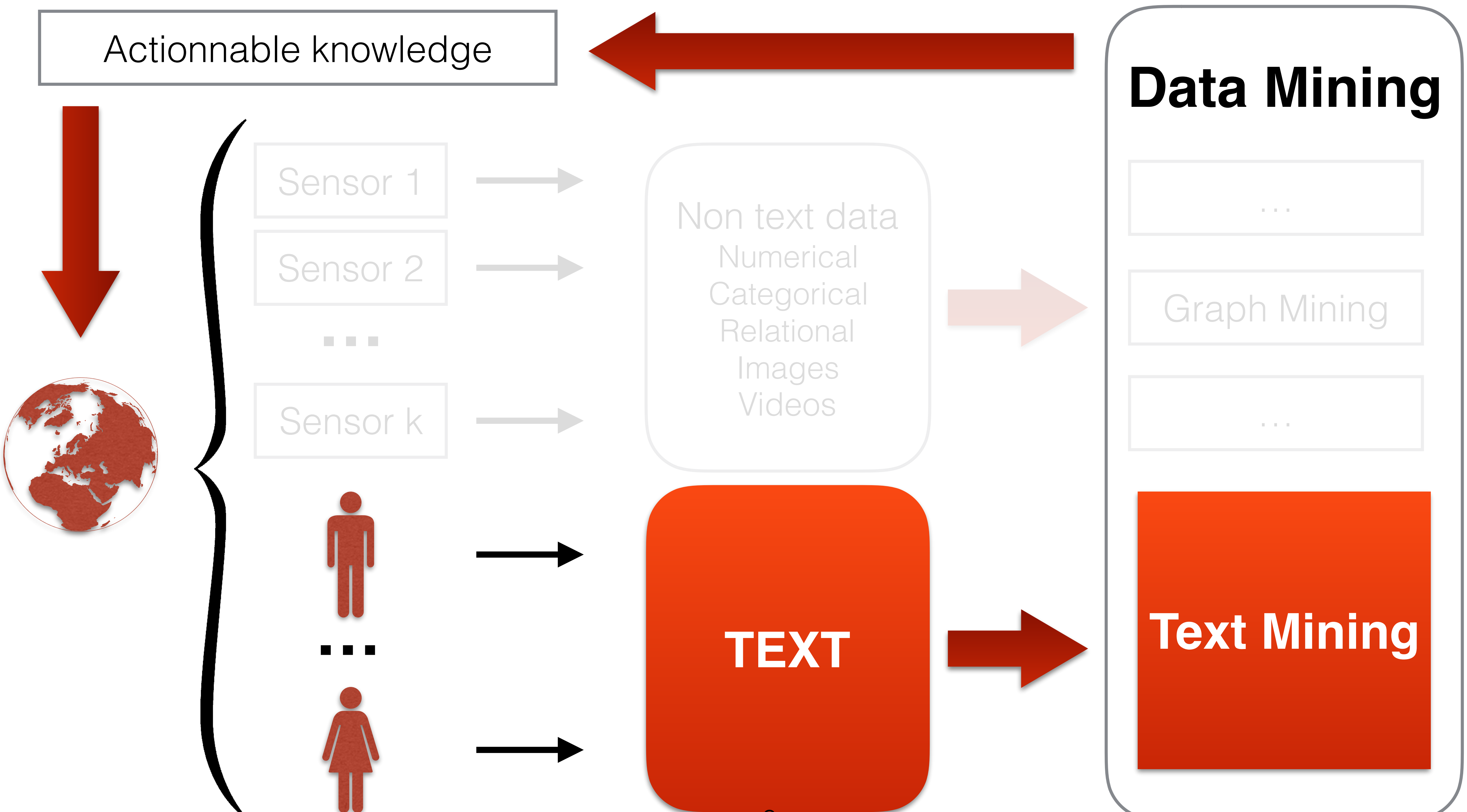
Text vs. Non-Text data

	Sense		Report	
Real world	→	**Sensor**	→	**Data**
Weather	→	Thermometer	→	3C, 15F, etc
Locations	→	Geo Sensor	→	41N and 120W
Networks	→	Network Sensor	→	0100010001100
Real World	→	**Human Sensor**	→	**Text**

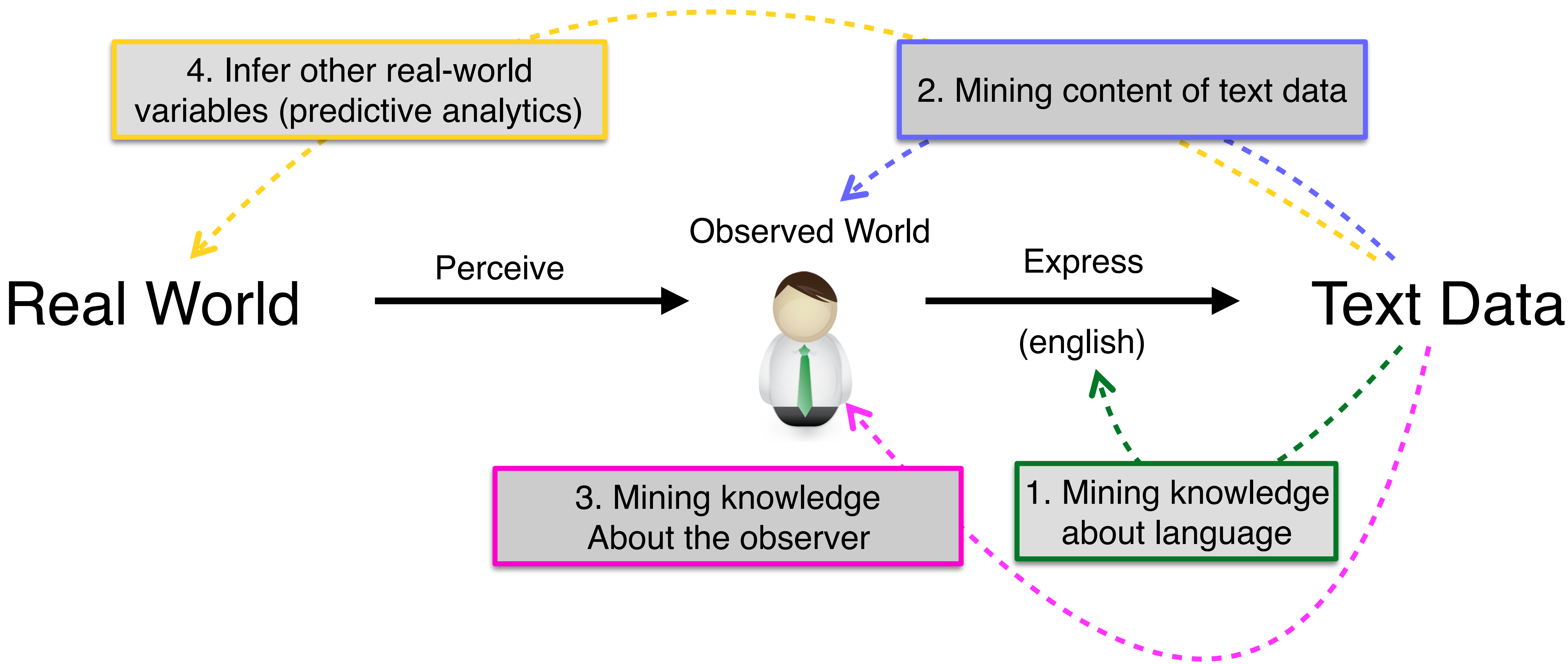
The General Problem of Data Mining



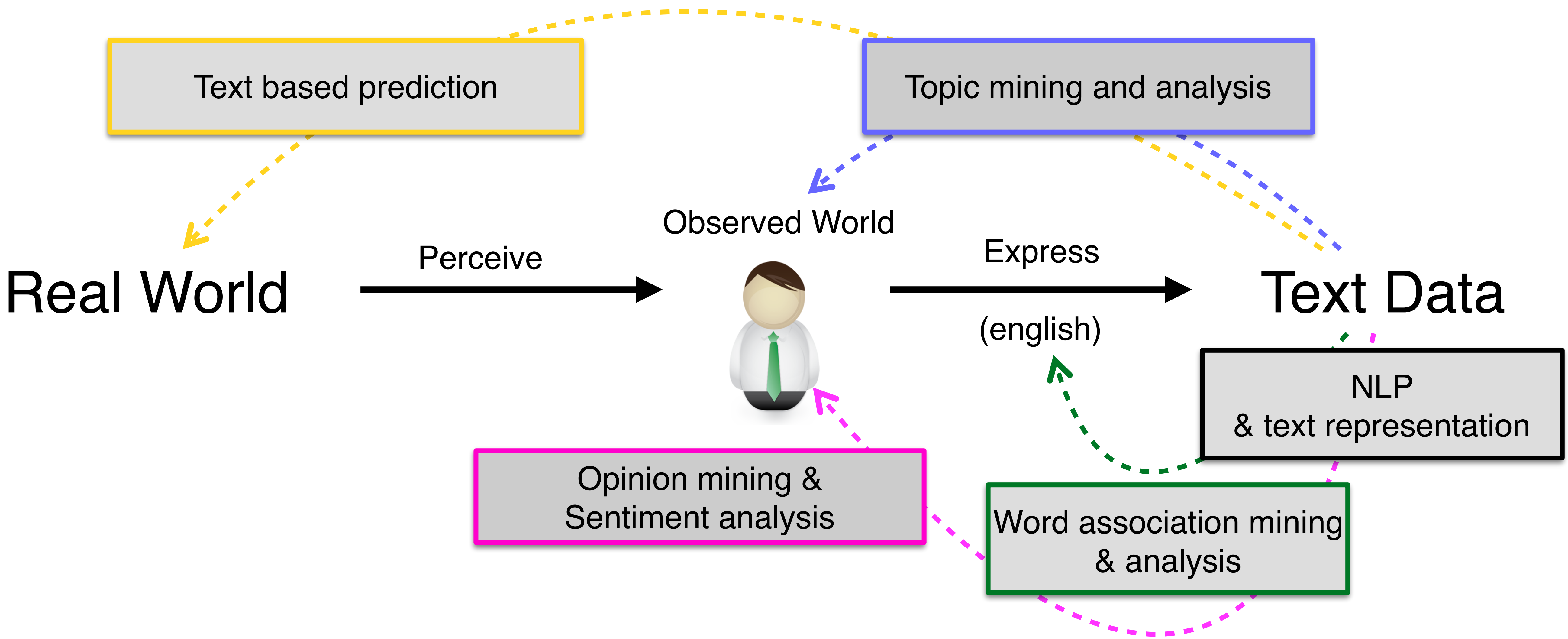
The General Problem of Data Mining



Landscape of Text Mining and Analytics



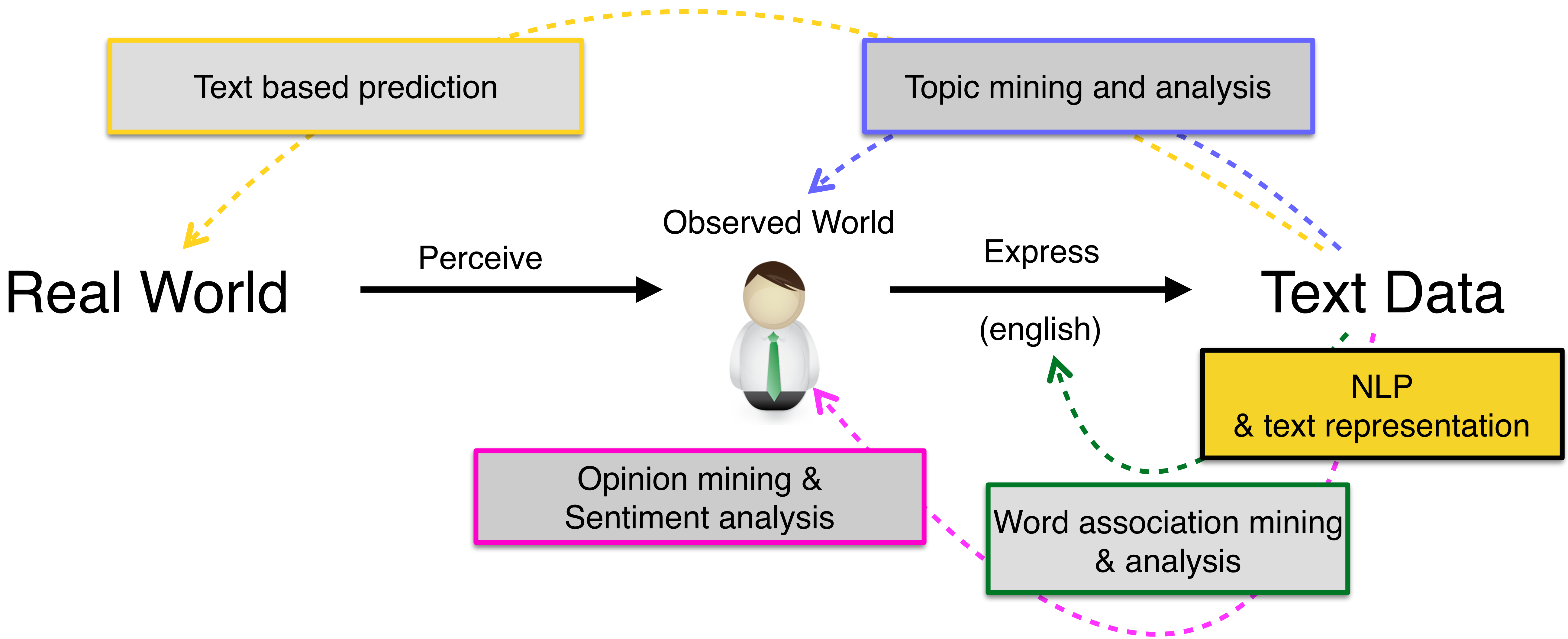
Landscape of Text Mining and Analytics



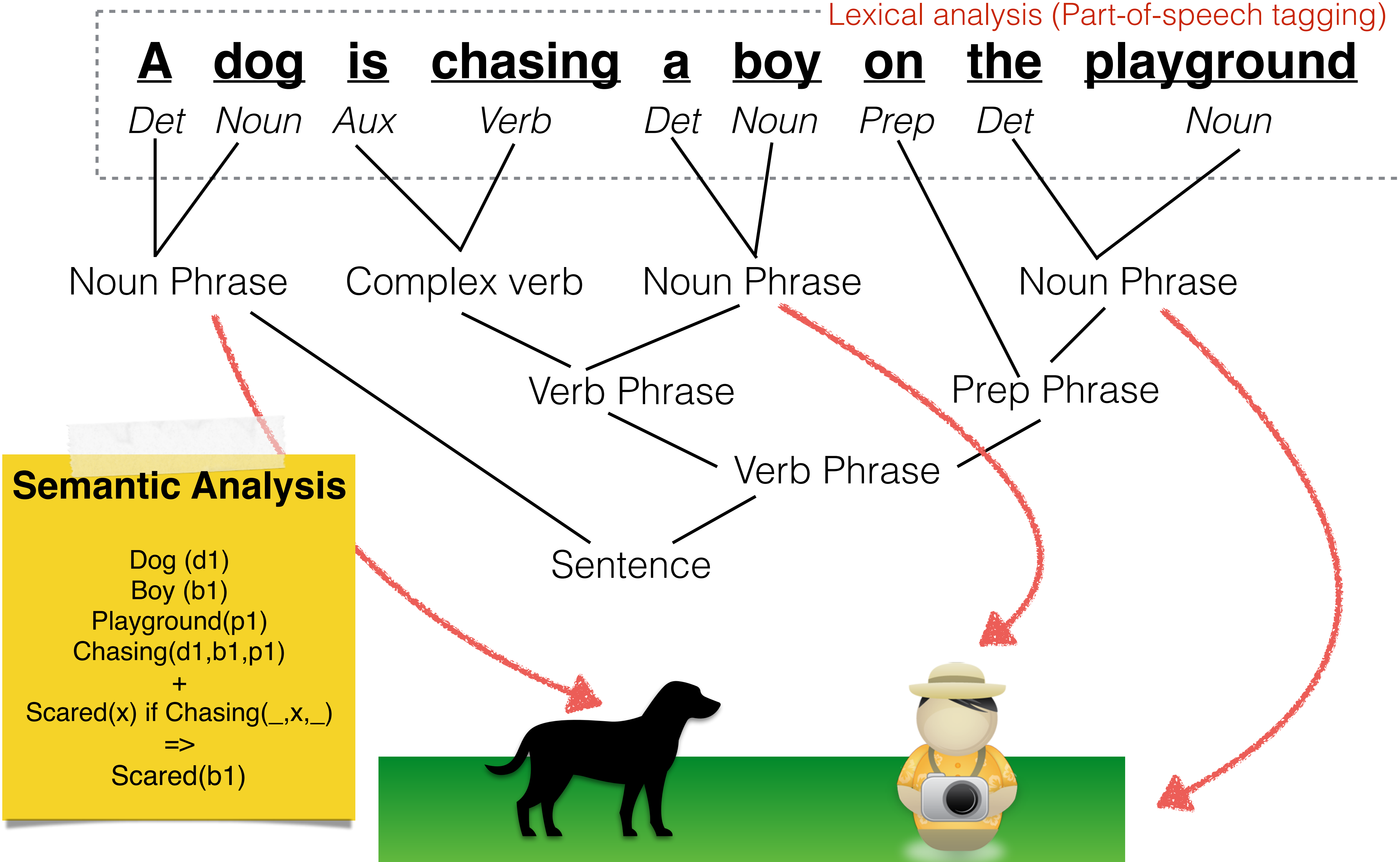
Basics of Text Mining

- **Collections**
  - Documents are compressed
  - Uncommon formats
  - Some times they just don't exist
- **Documents**
  - A lot of preprocessing (encoding, cleaning, splitting, etc.)
  - Granularity level (whole document, paragraph, sentence, etc.)
- **Words**
  - Steaming, upper/lower case, frequent (stopwords) and infrequent words, special characters, dates, prices, names, emails, etc.
- **General**
  - Language: main tasks have been addressed for English (with no 100% performance), but many other languages are far from English and many of them are completely without resources (regional languages)
  - Lot of formulas and concepts
  - Task is the main factor
  - Tools: python (NLKT), java (OpenNLP), c, etc.

Landscape of Text Mining and Analytics



Basic Concepts in NLP



NLP is difficult!

- Natural language is designed to make human communication efficient. As a result,
  - We omit a lot of common sense knowledge, which we assume the hearer/reader possesses.
  - We keep a lot of ambiguities, which we assume the hearer/reader knows how to resolve.
- This makes EVERY step in NLP hard
  - Ambiguity is a killer!
  - Common sense reasoning is pre-required.

Examples of Challenges

- Word-level ambiguity:
  - “design” can be a noun or a verb (ambiguous POS)
  - “root” has multiple meanings (ambiguous sense)
- Syntactic ambiguity:
  - “natural language processing” (modification)
  - “A man saw a boy with a telescope” (PP attachment)
  - Anaphora resolution: “John persuaded Bill to buy a TV for himself” (himself = John or Bill?)
  - Presupposition: “He has quite smoking” implies that he smoked before.

What we can't do

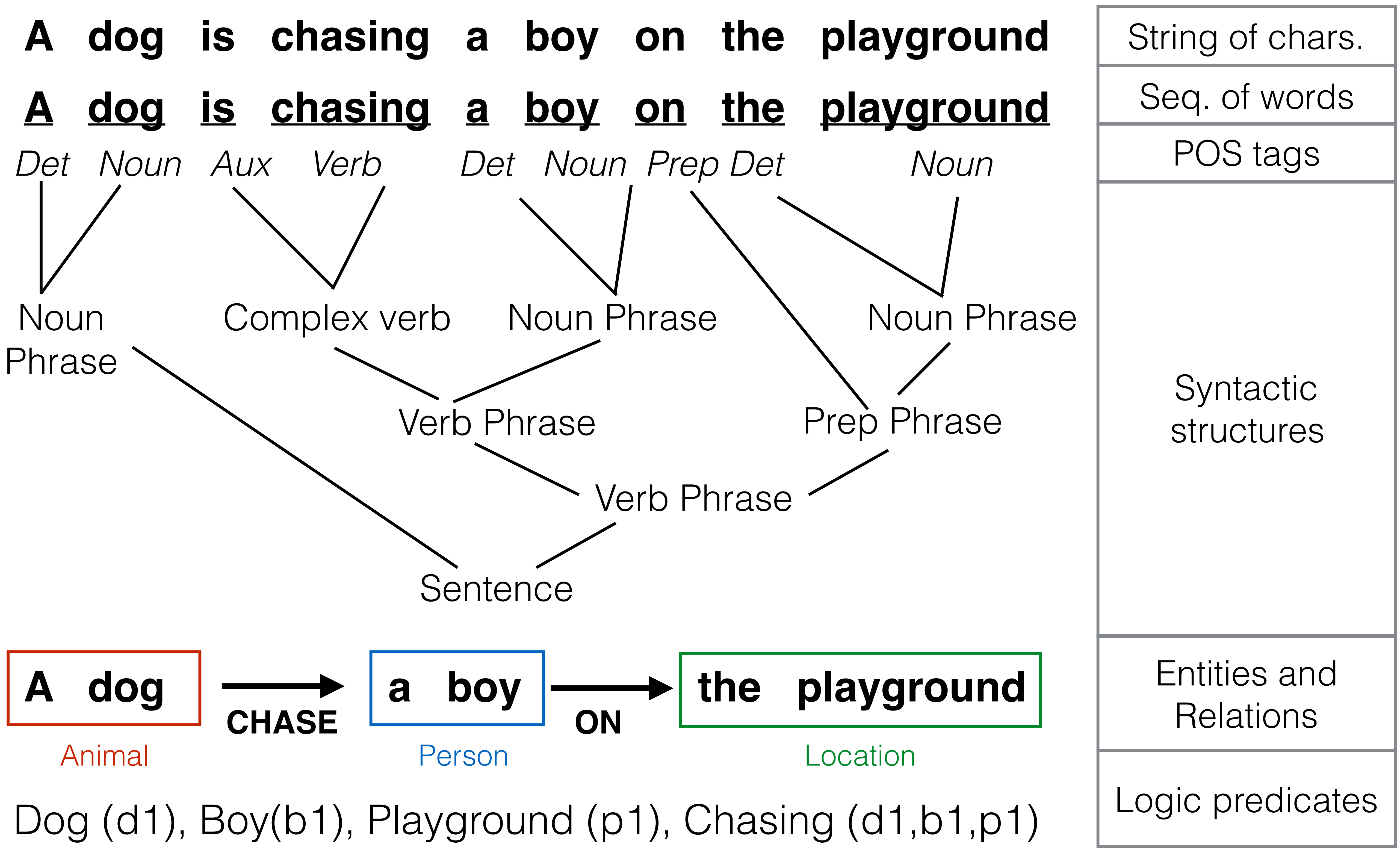
- 100 % POS tagging
  - ”He turned off the highway” vs “He turned off the fan”
- General complete parsing
  - “A man saw a boy with a telescope”
- Precise deep semantic analysis
  - Will we ever be able to precisely define meaning of “own” in “John owns a restaurant”?

Robust and general NLP tends to be shallow while deep understanding doesn’t scale up.

Take away message

- NLP is the foundation for text mining
- Computers are far from being able to understand natural language
  - Deep NLP requires common sense knowledge and inferences, thus only working for very limited domains
  - Shallow NLP based on statistical methods can be done in large scale and is thus more broadly applicable
- In practice: statistical NLP as the basis, while humans provide help as needed

Text representation



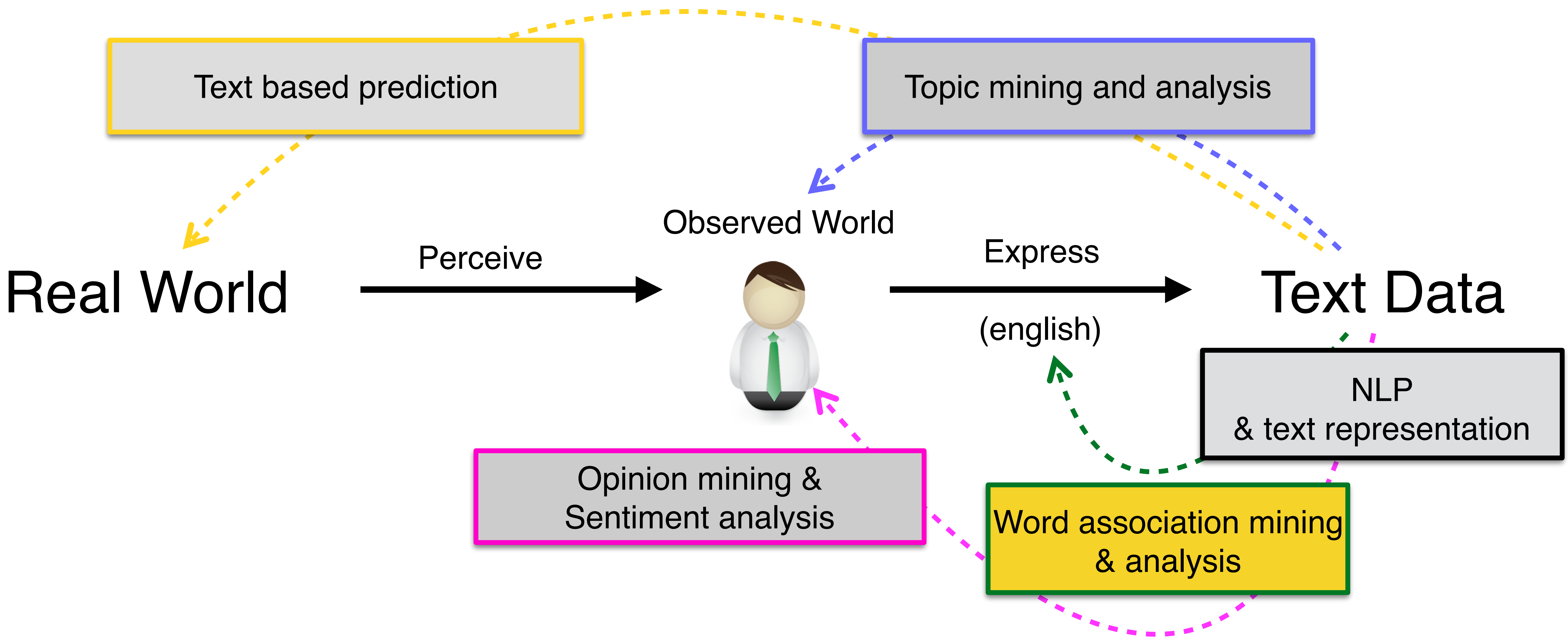
Text Representation and Enabled Analysis

Text Rep	**Generality**	**Enabled Analysis**	**Examples of Application**
String	******	String processing	Compression
Words	****	Word relation analysis; topic analysis; sentiment analysis	Thesaurus discovery; topic and opinion related applications
Syntactic structures	***	Syntactic graph analysis	Stylistic analysis; structure-based feature extraction
Entities & relations	**	Knowledge graph analysis; information network analysis	Discovery of knowledge ans opinions about specific entities
Logic predicates	*	Integrative analysis of scattered knowledge; logic inference	Knowledge assistant for biologist

Take away message

- Text representations determines what kind of mining algorithms can be applied
- Multiple ways of representing text are possible
  - String, words, syntactic structures, entity-relation graphs, predicates, etc.
  - Can/should be combined in real applications
- This course focuses mainly on word-based representation
  - General and robust: applicable to any natural language
  - No/little manual effort
  - “Surprisingly” powerful for many applications (not all!)
  - Can be combined with more sophisticated representations
- Tools
  - Python NLTK

Landscape of Text Mining and Analytics



Basic word relations

- Paradigmatic : A & B have paradigmatic relation if they can be substituted for each other (i.e., A & B are in the same class)
  - e.g;, “cat” and “dog”; “Monday” and “Tuesday”
- Syntagmatic: A & B have syntagmatic relation if they can be combined with each other (i.e., A & B are related semantically)
  - e.g., “cat” and “sit”; “car” and “drive”
- These two basic and complementary relations can be generalized to describe relations of any items in a language

Why mine word associations?

- They are useful for improving accuracy of many NLP tasks
  - POS tagging, parsing, entity recognition, acronym expansion
  - Grammar learning
- They are directly useful for many applications in text retrieval and mining
  - Text retrieval (e.g., use word associations to suggest a variation of a query)
  - Automatic construction of topic map of browsing: words as nodes ans associations as edges
  - Compare and summarize opinions (e.g., what words are most strongly associated with “battery” in positive and negative reviews about iPhone7, respectively?)

Mining Word Associations: Intuitions

My cat eats fish on Satuyday

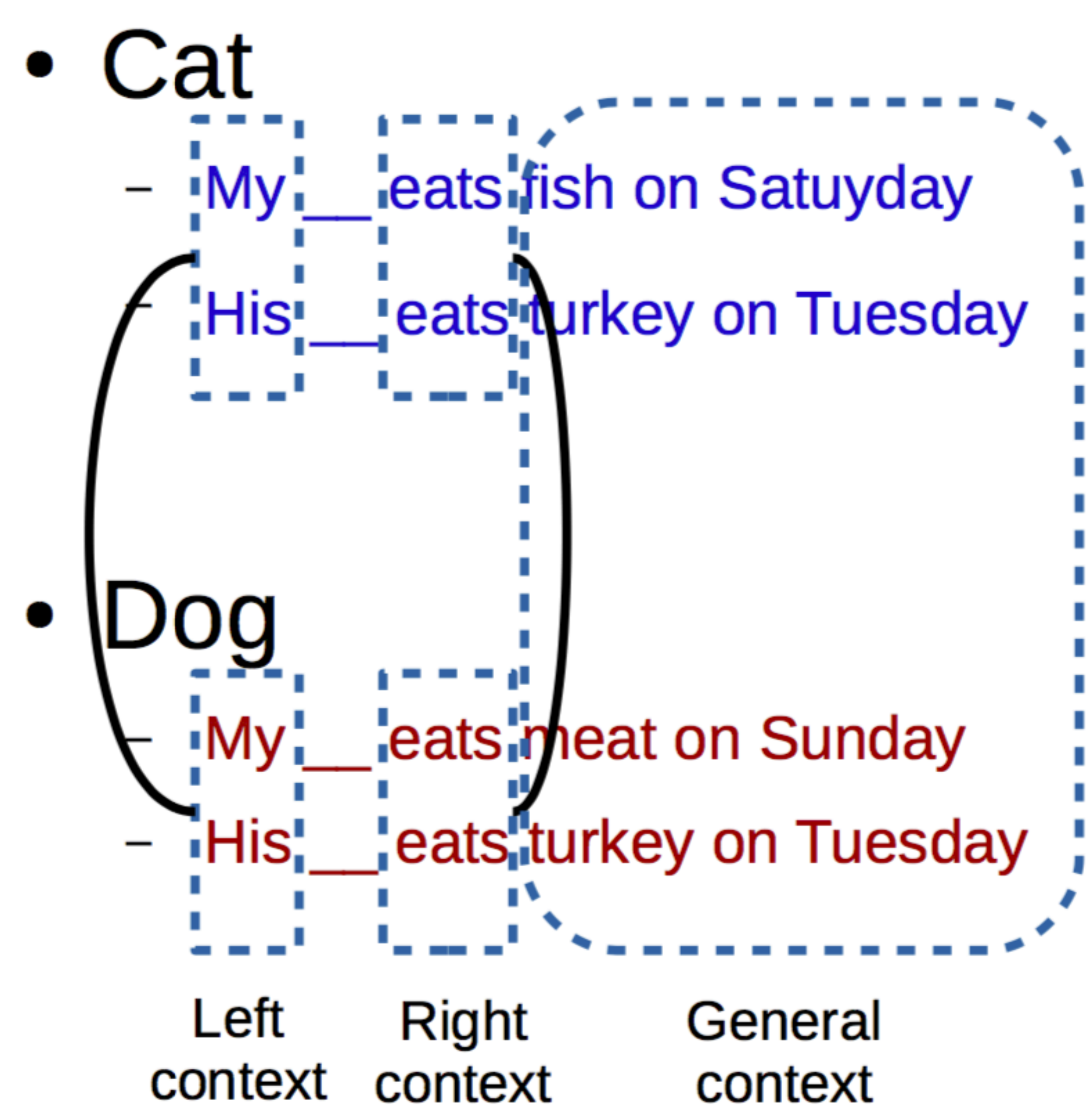
His cat eats turkey on Tuesday

My dog eats meat on Sunday

His dog eats turkey on Tuesday

Paradigmatic:
How similar are the context ("cat") and context ("dog")?
How similar are the context ("cat") and context ("computer")?

Syntagmatic:
How helpful is the occurrence of "eats" for predicting the occurrence of "meat"?
How helpful is the occurrence of "eats" for predicting the occurrence of "text"?



Mining Word Associations: General Ideas

- Paradigmatic
  - Represent each word by its context
  - Compute context similarity
  - Words with high context similarity likely have paradigmatic relation
- Syntagmatic
  - Count how many times two words occur together in a context (e.g., sentence or paragraph)
  - Compare their co-occurrences with their individual occurrences
  - Words with high co-occurrences but relatively low individual occurrences likely have syntagmatic relation

Distributional semantics

- Comparing two words:
  - Look at all context words for word1
  - Look at all context words for word2
  - How similar are those two context collections in their entirety?

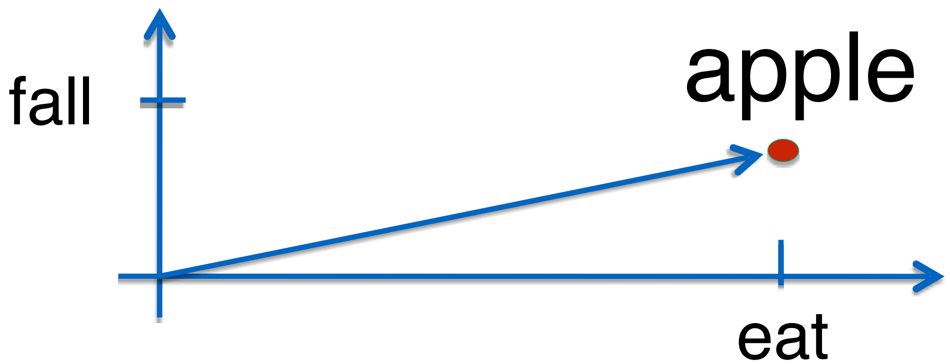
- Compare distributional representations of two words

How can we compare two context collections in their entirety?

Count how often “apple” occurs close to other words in a large text collection (corpus):

eat	fall	ripe	slice	peel	tree	throw	fruit	pie	bite	crab
794	244	47	221	208	160	145	156	109	104	88

Interpret counts as coordinates:



Every context word becomes a dimension.

How can we compare two context collections in their entirety?

Count how often "apple" occurs close to other words in a large text collection (corpus):

eat	fall	ripe	slice	peel	tree	throw	fruit	pie	bite	crab
794	244	47	221	208	160	145	156	109	104	88

Do the same for "orange":

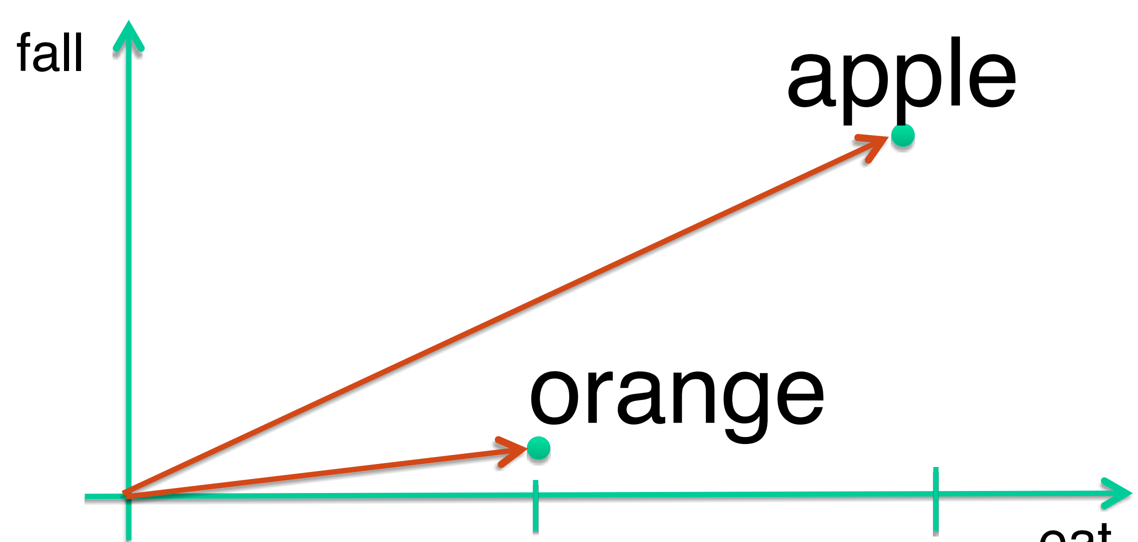
eat	fall	ripe	slice	peel	tree	throw	fruit	pie	bite	crab
265	22	25	62	220	64	74	111	4	4	8

How can we compare two context collections in their entirety?

Then visualize both count tables as vectors in the same space:

eat	fall	ripe	slice	peel	tree	throw	fruit	pie	bite	crab
794	244	47	221	208	160	145	156	109	104	88

eat	fall	ripe	slice	peel	tree	throw	fruit	pie	bite	crab
265	22	25	62	220	64	74	111	4	4	8



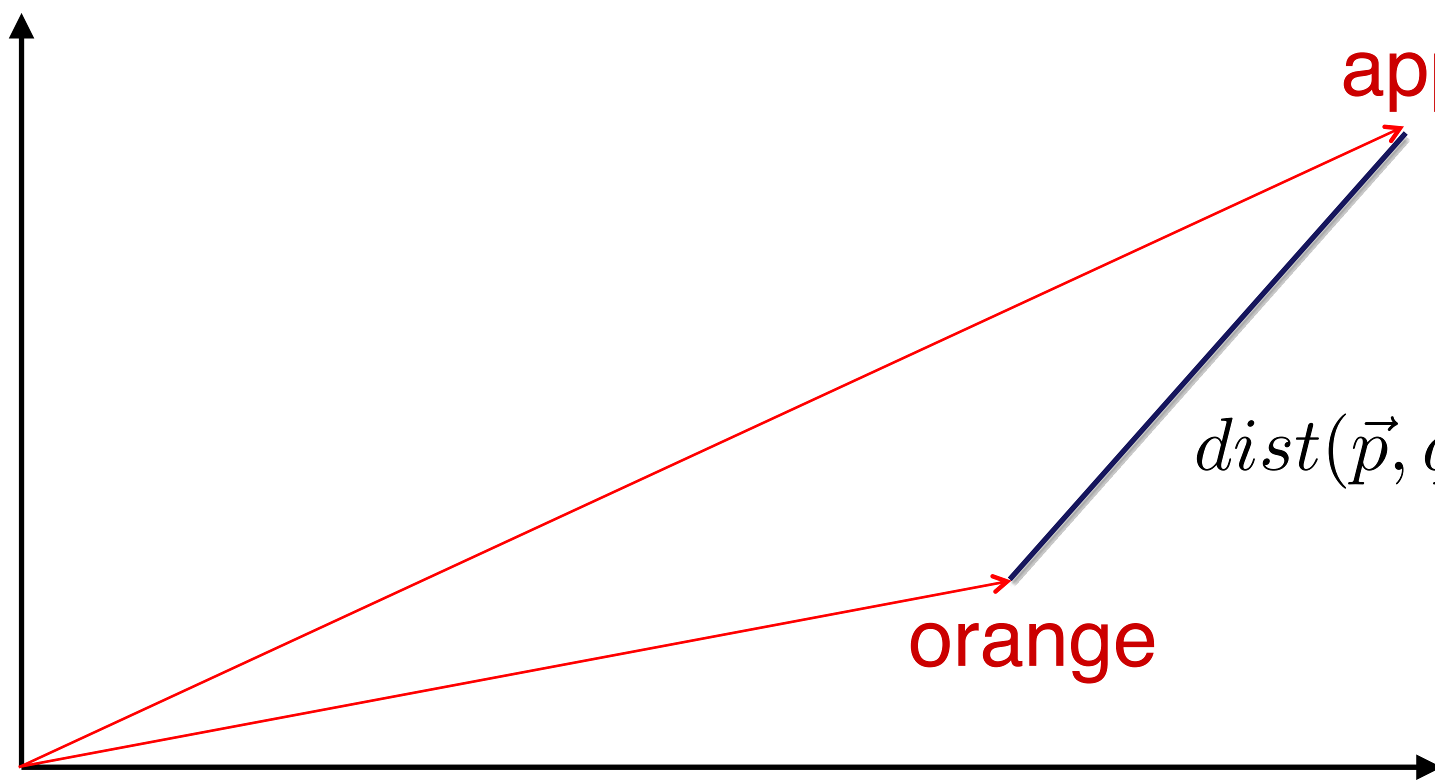
Similarity between two words as proximity in space

Where can we find texts to use for making a distributional model?

- Text in electronic form!
- Newspaper articles
- Project Gutenberg: older books available for free
- Wikipedia
- Text collections prepared for language analysis:
  - Balanced corpora
  - WaC: Scrape all web pages in a particular domain
  - ELRA, LDC hold corpus collections
    - For example, large amounts of newswire reports
  - Google n-grams, Google books

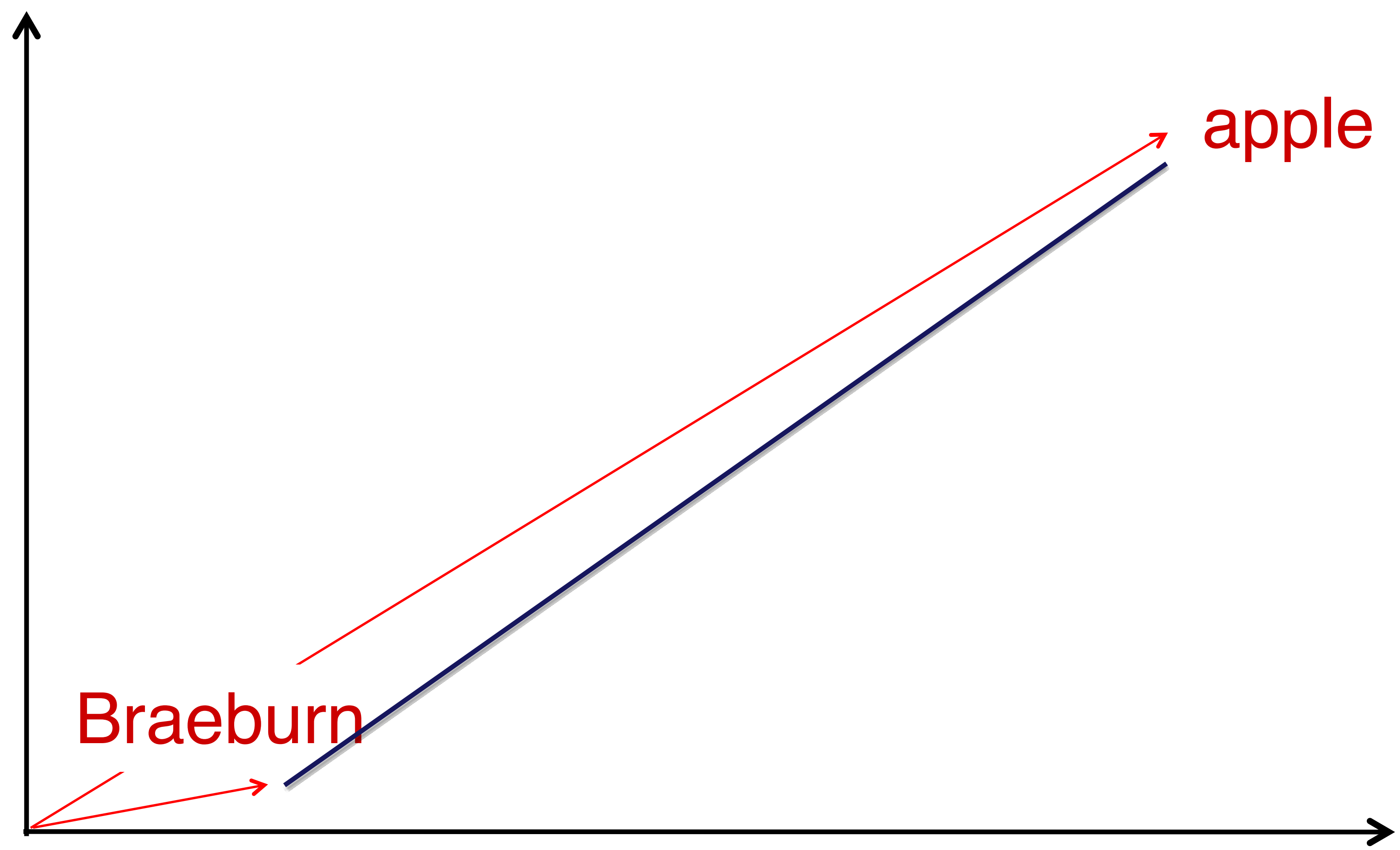
What do we mean by “similarity” of vectors?

Euclidean distance as a dissimilarity measure



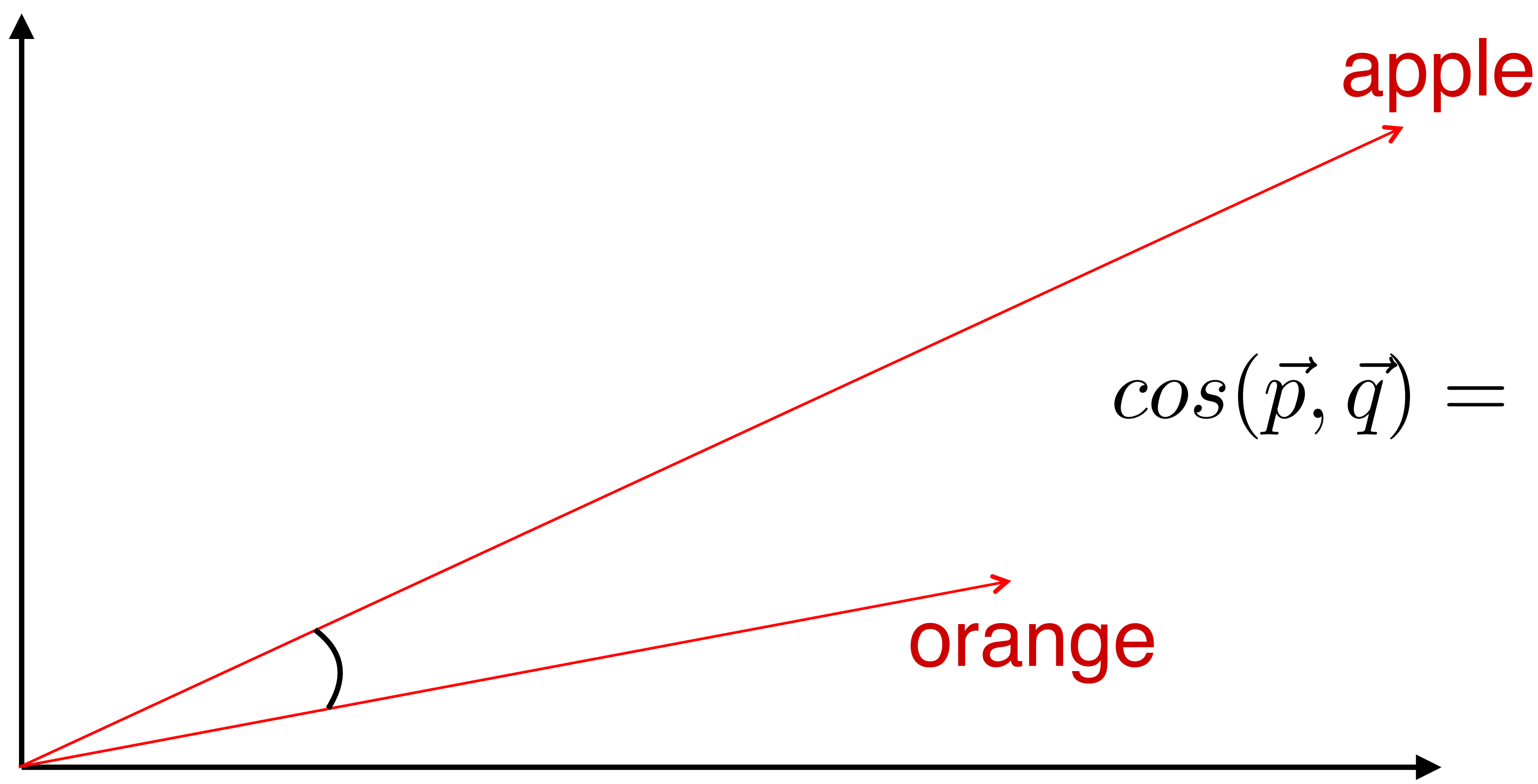
$$dist(\vec{p},\vec{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

Problem with Euclidean distance: very sensitive to word frequency!



What do we mean by "similarity" of vectors?

Use **angle** between vectors instead of point distance to get around word frequency issues



$$cos(\vec{p}, \vec{q}) = \frac{\sum_{i=1}^{n} p_i \cdot q_i}{\sqrt{\sum_{i=1}^{n} p_i^2} \cdot \sqrt{\sum_{i=1}^{n} q_i^2}}$$

Some counts for “letter” in “Pride and Prejudice”. What do you notice?

the	to	of	and	a	her	she	his	is	was	in	that
102	75	72	56	52	50	41	36	35	34	34	33

had	i	from	you	as	this	mr	for	not	on	be	he
32	28	28	25	23	23	22	21	21	20	18	17

but	elizabeth	with	him	which	by	when	jane
17	17	16	16	16	15	14	12

Some counts for “letter” in “Pride and Prejudice”. What do you notice?

the	to	of	and	a	her	she	his	is	was	in	that
102	75	72	56	52	50	41	36	35	34	34	33

had	i	from	you	as	this	mr	for	not	on	be	he
32	28	28	25	23	23	22	21	21	20	18	17

but	elizabeth	with	him	which	by	when	jane
17	17	16	16	16	15	14	12

Almost all the most frequent co-occurring words are function words.

Some words are more informative than others

- Function words co-occur frequently with **all** words
  - That makes them less informative
- They have much higher co-occurrence counts than content words
  - They can “drown out” more informative contexts
- **Frequency**
  - Just selecting the most frequently occurring bigrams

$$max_{x,y} p(x,y) \qquad\qquad max_{x,y} \log(p(x,y)+1)$$

  - A simple POS filter drastically improves the results. Filtering couples of words POS Tagged such as “Aux Noun” deals with results such as “Prime Minister”

Some Statistical Measures

- Pointwise Mutual Information (Theory of Information)
  - The PMI tells us the amount of information that is provided by the occurrence of one word about the occurrence of the other word

$$PMI(x,y) = \log_2 \frac{p(x,y)}{p(x)p(y)}$$

- Pearson's Phi-Square Test (Test Hypothesis)
  - The essence of the test is to compare the observed frequencies with the frequencies expected for independence in a contingency table
  - Try to refute the Null Hypothesis. The Higher the score, the more confidently the Hypothesis Ho can be rejected

$$Ho: p(x,y) = p(x)p(y)$$

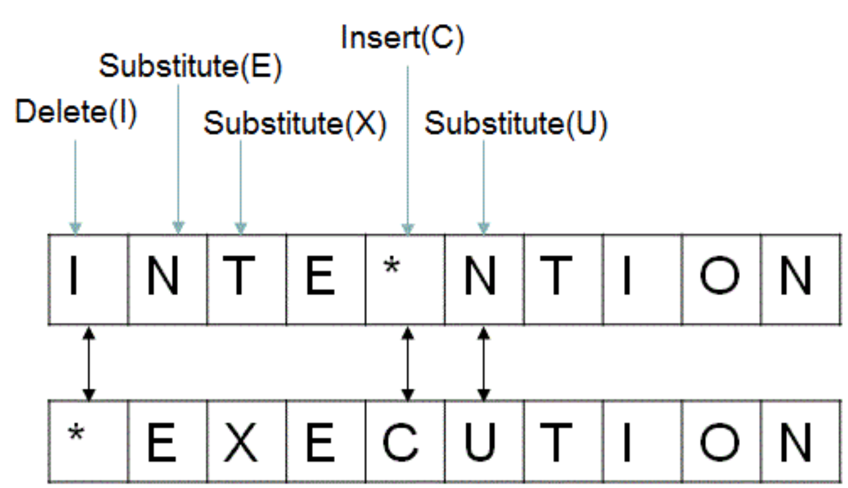
Other Measures

There exists many associative measures…

# Name	Formula
1. Joint probability	$P(xy)$
*2. Conditional probability	$P(y \mid x)$
3. Reverse conditional prob.	$P(x \mid y)$
4. Pointwise mutual inform.	$\log \frac{P(xy)}{P(x*)P(*y)}$
5. Mutual dependency (*MD*)	$\log \frac{P(xy)^2}{P(x*)P(*y)}$
6. Log frequency biased *MD*	$\log \frac{P(xy)^2}{P(x*)P(*y)} + \log P(xy)$
7. Normalized expectation	$\frac{2f(xy)}{f(x*)+f(*y)}$
8. Mutual expectation	$\frac{2f(xy)}{f(x*)+f(*y)} \cdot P(xy)$
*9. Salience	$\log \frac{P(xy)^2}{P(x*)P(*y)} \cdot \log f(xy)$
10. Pearson's χ^2 test	$\sum_{i,j} \frac{(f_{ij}-\hat{f}_{ij})^2}{\hat{f}_{ij}}$
11. Fisher's exact test	$\frac{f(x*)!f(\bar{x}*)!f(*y)!f(*\bar{y})!}{N!f(xy)!f(x\bar{y})!f(\bar{x}y)!f(\bar{x}\bar{y})!}$
12. t test	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{f(xy)(1-(f(xy)/N))}}$
13. z score	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{\hat{f}(xy)(1-(\hat{f}(xy)/N))}}$
14. Poison significance measure	$\frac{\hat{f}(xy)-f(xy)\log \hat{f}(xy)+\log f(xy)!}{\log N}$
15. Log likelihood ratio	$-2\sum_{i,j} f_{ij} \log \frac{f_{ij}}{\hat{f}_{ij}}$
16. Squared log likelihood ratio	$-2\sum_{i,j} \frac{\log f_{ij}^2}{\hat{f}_{ij}}$
Association coefficients:	
17. Russel-Rao	$\frac{a}{a+b+c+d}$
18. Sokal-Michiner	$\frac{a+d}{a+b+c+d}$
19. Rogers-Tanimoto	$\frac{a+d}{a+2b+2c+d}$
20. Hamann	$\frac{(a+d)-(b+c)}{a+b+c+d}$
21. Third Sokal-Sneath	$\frac{b+c}{a+d}$
22. Jaccard	$\frac{a}{a+b+c}$
*23. First Kulczynsky	$\frac{a}{b+c}$
24. Second Sokal-Sneath	$\frac{a}{a+2(b+c)}$
25. Second Kulczynski	$\frac{1}{2}(\frac{a}{a+b}+\frac{a}{a+c})$
*26. Fourth Sokal-Sneath	$\frac{1}{4}(\frac{a}{a+b}+\frac{a}{a+c}+\frac{d}{d+b}+\frac{d}{d+c})$
*27. Odds ratio	$\frac{ad}{bc}$
28. Yulle's ω	$\frac{\sqrt{ad}-\sqrt{bc}}{\sqrt{ad}+\sqrt{bc}}$
29. Yulle's Q	$\frac{ad-bc}{ad+bc}$
30. Driver-Kroeber	$\frac{a}{\sqrt{(a+b)(a+c)}}$
31. Fifth Sokal-Sneath	$\frac{ad}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$
32. Pearson	$\frac{ad-bc}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$
33. Baroni-Urbani	$\frac{a+\sqrt{ad}}{a+b+c+\sqrt{ad}}$
*34. Braun-Blanquet	$\frac{a}{\max(a+b,a+c)}$
*35. Simpson	$\frac{a}{\min(a+b,a+c)}$
36. Michael	$\frac{4(ad-bc)}{(a+d)^2+(b+c)^2}$
37. Mountford	$\frac{2a}{2bc+ab+ac}$
38. Fager	$\frac{a}{\sqrt{(a+b)(a+c)}}-\frac{1}{2}\max(b,c)$
39. Unigram subtuples	$\log\frac{ad}{bc}-3.29\sqrt{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}+\frac{1}{d}}$
40. *U* cost	$\log(1+\frac{\min(b,c)+a}{\max(b,c)+a})$
41. *S* cost	$\log(1+\frac{\min(b,c)}{a+1})^{-\frac{1}{2}}$
42. *R* cost	$\log(1+\frac{a}{a+b})\cdot\log(1+\frac{a}{a+c})$
43. *T* combined cost	$\sqrt{U \times S \times R}$
44. Phi	$\frac{P(xy)-P(x*)P(*y)}{\sqrt{P(x*)P(*y)(1-P(x*))(1-P(*y))}}$
45. Kappa	$\frac{P(xy)+P(\bar{x}\bar{y})-P(x*)P(*y)-P(\bar{x}*)P(*\bar{y})}{1-P(x*)P(*y)-P(\bar{x}*)P(*\bar{y})}$
46. *J* measure	$\max[P(xy)\log\frac{P(y \mid x)}{P(*y)}+P(x\bar{y})\log\frac{P(\bar{y} \mid x)}{P(*\bar{y})},$ $P(xy)\log\frac{P(x \mid y)}{P(x*)}+P(\bar{x}y)\log\frac{P(\bar{x} \mid y)}{P(\bar{x}*)}]$
47. Gini index	$\max[P(x*)(P(y \mid x)^2+P(\bar{y} \mid x)^2)-P(*y)^2$ $+P(\bar{x}*)(P(y \mid \bar{x})^2+P(\bar{y} \mid \bar{x})^2)-P(*\bar{y})^2,$ $P(*y)(P(x \mid y)^2+P(\bar{x} \mid y)^2)-P(x*)^2$ $+P(*\bar{y})(P(x \mid \bar{y})^2+P(\bar{x} \mid \bar{y})^2)-P(\bar{x}*)^2]$
48. Confidence	$\max[P(y \mid x), P(x \mid y)]$
49. Laplace	$\max[\frac{NP(xy)+1}{NP(x*)+2}, \frac{NP(xy)+1}{NP(*y)+2}]$
50. Conviction	$\max[\frac{P(x*)P(*y)}{P(x\bar{y})}, \frac{P(\bar{x}*)P(*y)}{P(\bar{x}y)}]$
51. Piatersky-Shapiro	$P(xy)-P(x*)P(*y)$
52. Certainity factor	$\max[\frac{P(y \mid x)-P(*y)}{1-P(*y)}, \frac{P(x \mid y)-P(x*)}{1-P(x*)}]$
53. Added value (*AV*)	$\max[P(y \mid x)-P(*y), P(x \mid y)-P(x*)]$
54. Collective strength	$\frac{P(xy)+P(\bar{x}\bar{y})}{P(x*)P(y)+P(\bar{x}*)P(*y)} \cdot$ $\frac{1-P(x*)P(*y)-P(\bar{x}*)P(*y)}{1-P(xy)-P(\bar{x}\bar{y})}$
*55. Klosgen	$\sqrt{P(xy)} \cdot AV$
Context measures:	
*56. Context entropy	$-\sum_w P(w \mid C_{xy}) \log P(w \mid C_{xy})$
*57. Left context entropy	$-\sum_w P(w \mid C^l_{xy}) \log P(w \mid C^l_{xy})$
58. Right context entropy	$-\sum_w P(w \mid C^r_{xy}) \log P(w \mid C^r_{xy})$
59. Left context divergence	$P(x*) \log P(x*)$ $-\sum_w P(w \mid C^l_{xy}) \log P(w \mid C^l_{xy})$
60. Right context divergence	$P(*y) \log P(*y)$ $-\sum_w P(w \mid C^r_{xy}) \log P(w \mid C^r_{xy})$
61. Cross entropy	$-\sum_w P(w \mid C_x) \log P(w \mid C_y)$
62. Reverse cross entropy	$-\sum_w P(w \mid C_y) \log P(w \mid C_x)$
63. Intersection measure	$\frac{2\lvert C_x \cap C_y \rvert}{\lvert C_x \rvert + \lvert C_y \rvert}$
*64. Euclidean norm	$\sqrt{\sum_w (P(w \mid C_x)-P(w \mid C_y))^2}$
65. Cosine norm	$\frac{\sum_w P(w \mid C_x)P(w \mid C_y)}{\sum_w P(w \mid C_x)^2 \cdot \sum_w P(w \mid C_y)^2}$
*66. *L1* norm	$\sum_w \lvert P(w \mid C_x)-P(w \mid C_y) \rvert$
67. Confusion probability	$\sum_w \frac{P(x \mid C_w)P(y \mid C_w)P(w)}{P(x*)}$
*68. Reverse confusion prob.	$\sum_w \frac{P(y \mid C_w)P(x \mid C_w)P(w)}{P(*y)}$
*69. Jensen-Shannon diverg.	$\frac{1}{2}[D(p(w \mid C_x) \Vert \frac{1}{2}(p(w \mid C_x)+p(w \mid C_y)))$ $+D(p(w \mid C_y) \Vert \frac{1}{2}(p(w \mid C_x)+p(w \mid C_y)))]$
*70. Cosine of pointwise *MI*	$\frac{\sum_w MI(w,x)MI(w,y)}{\sqrt{\sum_w MI(w,x)^2} \cdot \sqrt{\sum_w MI(w,y)^2}}$
71. KL divergence	$\sum_w P(w \mid C_x) \log \frac{P(w \mid C_x)}{P(w \mid C_y)}$
72. Reverse KL divergence	$\sum_w P(w \mid C_y) \log \frac{P(w \mid C_y)}{P(w \mid C_x)}$
*73. Skew divergence	$D(p(w \mid C_x) \Vert \alpha(w \mid C_y)+(1-\alpha)p(w \mid C_x))$
74. Reverse skew divergence	$D(p(w \mid C_y) \Vert \alpha p(w \mid C_x)+(1-\alpha)p(w \mid C_y))$
75. Phrase word coocurrence	$\frac{1}{2}(\frac{f(x \mid C_{xy})}{f(xy)}+\frac{f(y \mid C_{xy})}{f(xy)})$
76. Word association	$\frac{1}{2}(\frac{f(x \mid C_y)-f(xy)}{f(xy)}+\frac{f(y \mid C_x)-f(xy)}{f(xy)})$
Cosine context similarity:	$\frac{1}{2}(\cos(\mathbf{c}_x,\mathbf{c}_{xy})+\cos(\mathbf{c}_y,\mathbf{c}_{xy}))$ $\mathbf{c}_z=(z_i);\ \cos(\mathbf{c}_x,\mathbf{c}_y)=\frac{\sum x_i y_i}{\sqrt{\sum x_i{}^2}\cdot\sqrt{\sum y_i{}^2}}$
*77. in boolean vector space	$z_i=\delta(f(w_i \mid C_z))$
78. in *tf* vector space	$z_i=f(w_i \mid C_z)$
79. in *tf·idf* vector space	$z_i=f(w_i \mid C_z)\cdot\frac{N}{df(w_i)};\ df(w_i)=\lvert\{x : w_i \in C_x\}\rvert$
Dice context similarity:	$\frac{1}{2}(\mathrm{dice}(\mathbf{c}_x,\mathbf{c}_{xy})+\mathrm{dice}(\mathbf{c}_y,\mathbf{c}_{xy}))$ $\mathbf{c}_z=(z_i);\ \mathrm{dice}(\mathbf{c}_x,\mathbf{c}_y)=\frac{2\sum x_i y_i}{\sum x_i{}^2+\sum y_i{}^2}$
80. in boolean vector space	$z_i=\delta(f(w_i \mid C_z))$
81. in *tf* vector space	$z_i=f(w_i \mid C_z)$
82. in *tf·idf* vector space	$z_i=f(w_i \mid C_z)\cdot\frac{N}{df(w_i)};\ df(w_i)=\lvert\{x : w_i \in C_x\}\rvert$

String level metrics

- String metric is a metric that measures distance between two text strings for approximate string matching or comparison and in fuzzy string searching
- For example, the strings "Sam" and "Samuel" can be considered to be close
- A string metric provides a number indicating an algorithm-specific indication of distance



String level metrics

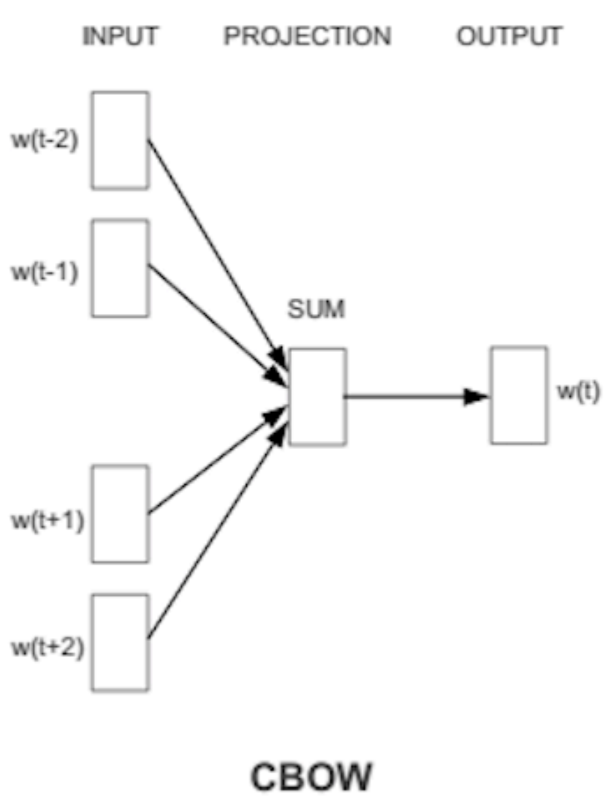
- List of string metrics
  - Sørensen–Dice coefficient
  - Block distance or L1 distance or City block distance
  - Jaro–Winkler distance
  - Simple matching coefficient (SMC)
  - Jaccard similarity or Jaccard coefficient or Tanimoto coefficient
  - Tversky index
  - Overlap coefficient
  - Variational distance
  - Hellinger distance or Bhattacharyya distance
  - Information radius (Jensen–Shannon divergence)
  - Skew divergence
  - Confusion probability
  - Tau metric, an approximation of the Kullback–Leibler divergence
  - Fellegi and Sunters metric (SFS)
  - Maximal matches
  - Grammar-based distance

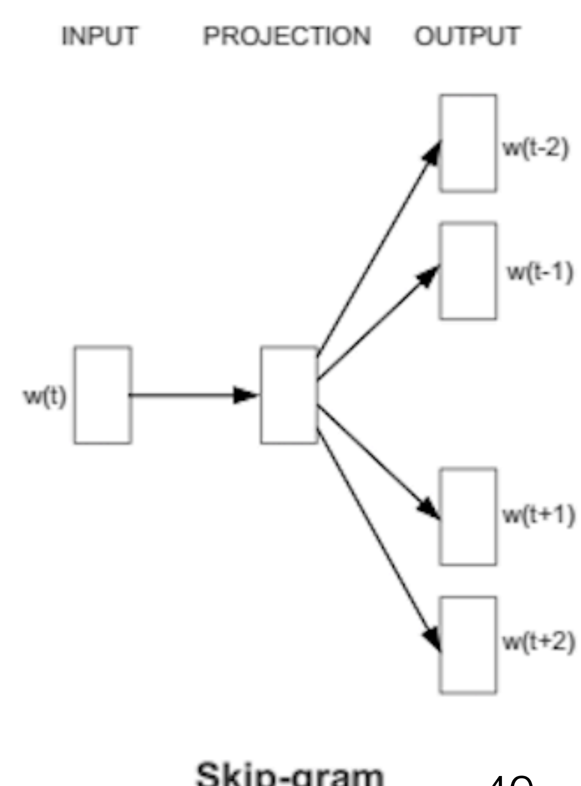
String level metrics

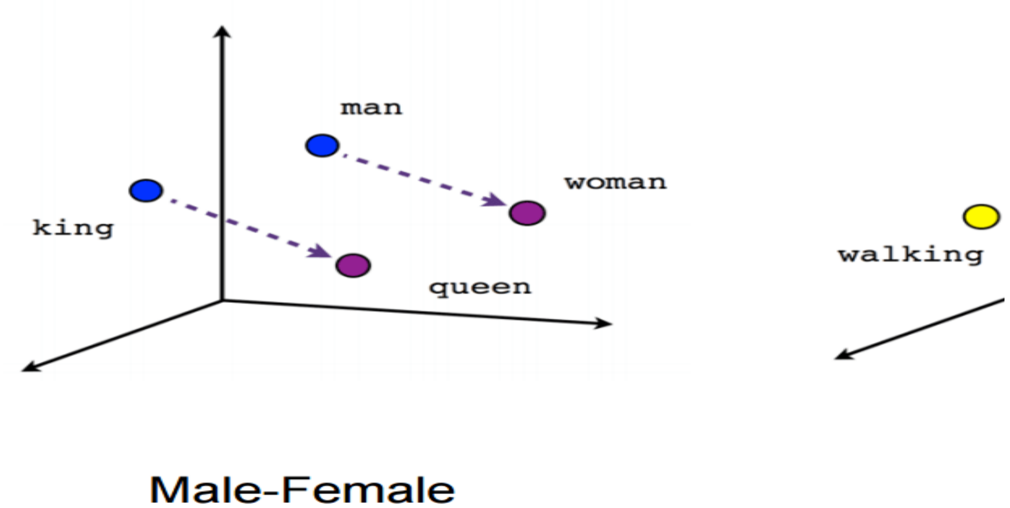
Name	Example
Hamming distance	"**karolin**" and "**kathrin**" is 3.
Levenshtein distance and Damerau–Levenshtein distance	**kitten** and **sitting** have a distance of 3. 1. **k**itten → **s**itten (substitution of "s" for "k") 2. sitt**e**n → sitt**i**n (substitution of "i" for "e") 3. sittin → sittin**g** (insertion of "g" at the end).
Jaro–Winkler distance	JaroWinklerDist("MARTHA","MARHTA") = $d_j = \frac{1}{3}\left(\frac{m}{\|s_1\|} + \frac{m}{\|s_2\|} + \frac{m-t}{m}\right) = \frac{1}{3}\left(\frac{6}{6} + \frac{6}{6} + \frac{6-\frac{2}{2}}{6}\right) = 0.944$ • m is the number of *matching characters*; • t is half the number of *transpositions*(`"MARTHA"[3]!=H, "MARHTA"[3]!=T`).
Most frequent k characters	MostFreqKeySimilarity('research', 'seeking', 2) = 2

Word2vec: a state-of-the-art technique

- Word2Vec is an interesting technique that could be used to address this problem
  - Proposed in 2013 (just 5 years ago)
  - Tons of available implementations and ready-to-go calculations
  - Each word is mapped to a vector (not really new idea)
  - The main idea is to predict the relationship between a word and a context using a neural network







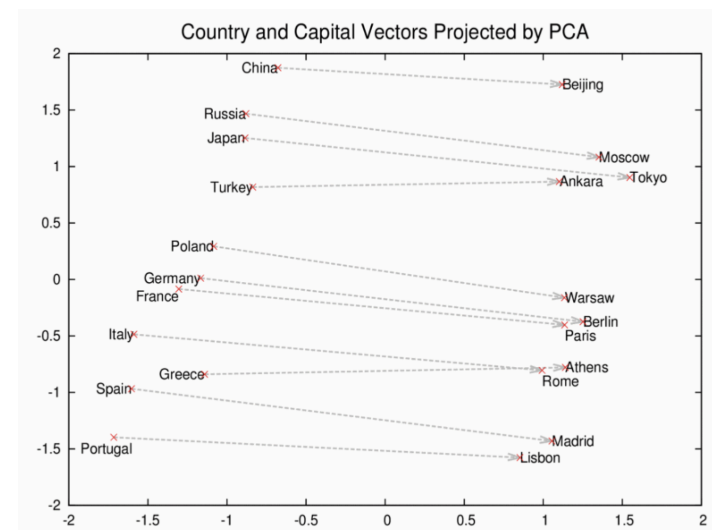
Word2Vec - clarifications

- Deep learning != Word embeddings
- Word2Vec is a technique to calculate Word embeddings, but there are many more
- Several public available implementations and already calculated vectors over the Web
- It is cool and works well, but not so easy to train
- Parameter selection is a big deal (as well as computing processing)

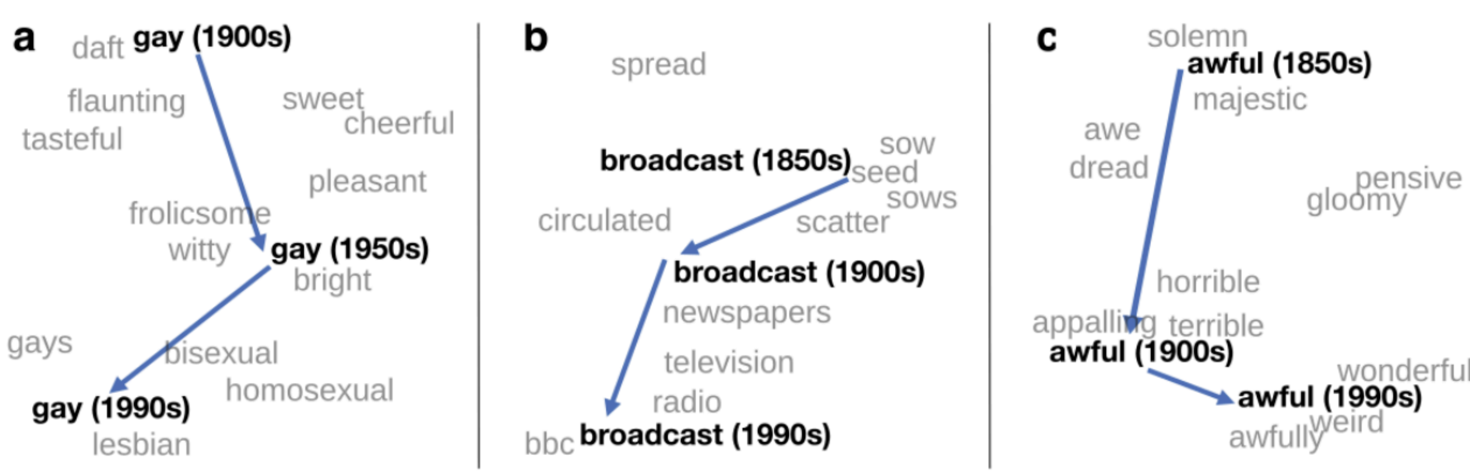
Word embeddings evaluation

- They are just vectors, so use them as it
- Public dataset of analogies

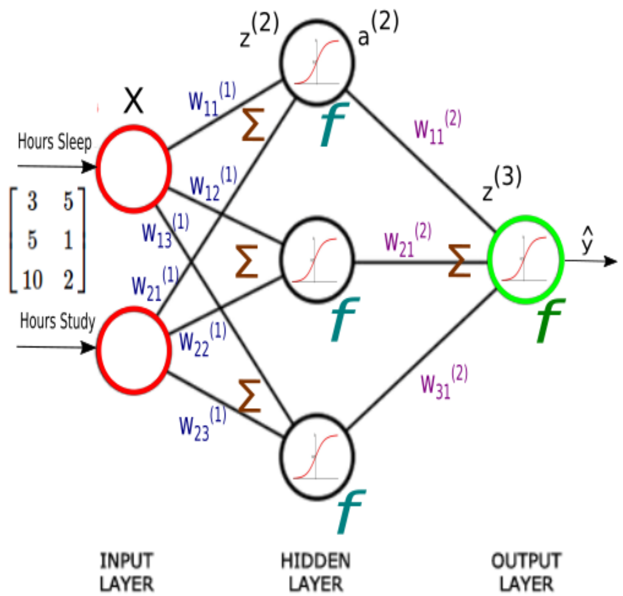
Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza



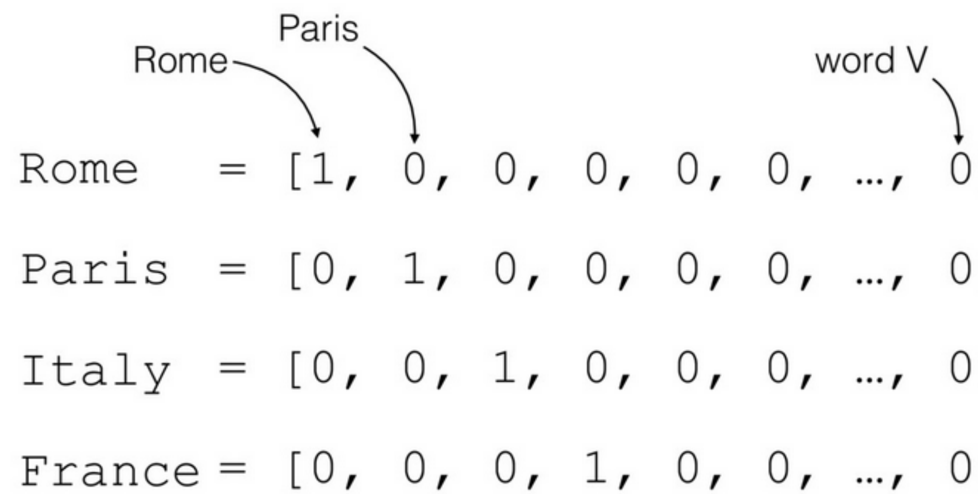
- Higher accuracy the better, but don't forget for what do you want them



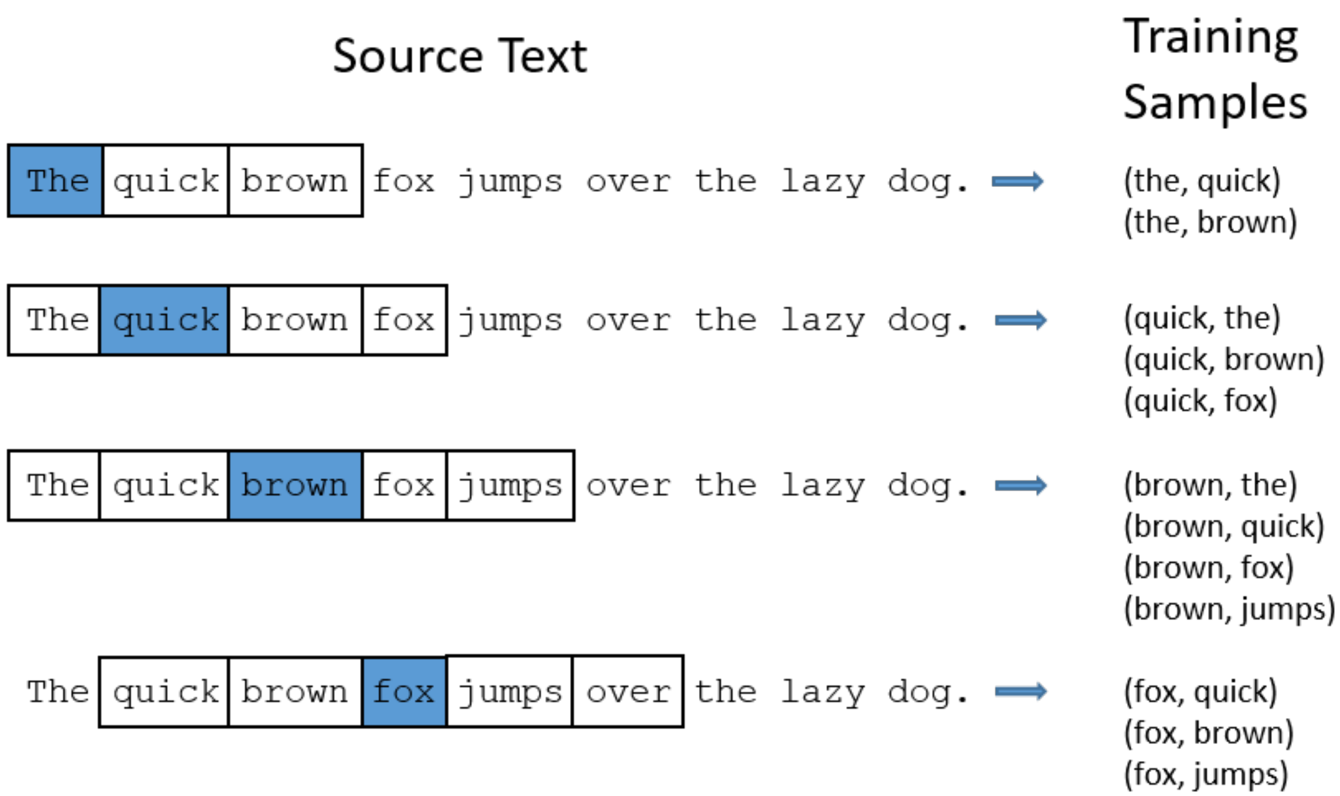
Word2Vec



Artificial Neural Networks



1-hot encoding



Large 'Unsupervised' Training

Take away message

- You know the formulas...so, you can code it by your self!
- Depending of the task, preprocessing could be a critical phase. (e.g. orange and oranges)
- Different granularity levels could be use: document, paragraph, sentence, etc.
- Python
  - nltk.metrics.association module
- How to know what is better?
  - Evaluation!!! Many datasets allows to evaluate the performance of existing algorithms (e.g. SemEval datasets, Semantic and Syntactic similarity dataset)