

Semantic Text Similarity

Salar Mohtaj | DFKI

Semantic textual similarity

- What is semantic similarity?
- Semantic similarity in word level
- Semantic similarity in sentence level
- Semantic textual similarity in Python

Semantic textual similarity

- What is semantic similarity?
- Semantic similarity in word level
- Semantic similarity in sentence level
- Semantic textual similarity in Python

Semantic textual similarity

- Semantic textual similarity (STS) deals with determining how similar two pieces of texts are
- It's about measuring semantic similarity between words/terms, sentences, paragraphs or documents
- Semantic similarity methods usually give a ***ranking*** or ***percentage*** of similarity between texts, rather than a binary decision as similar or not similar
- Related tasks are ***paraphrase identification***, or ***duplicate identification***

Semantic textual similarity

- The techniques like Bag of Words (BoW) and TF-IDF are used to represent text, as real value vectors
- However, these techniques did not attribute to the fact that words have different meanings and different words can be used to represent a similar concept

John and David studied Math and Science.

≠

John studied Math and David studied Science.

Mary is allergic to dairy products.

=

Mary is lactose intolerant.

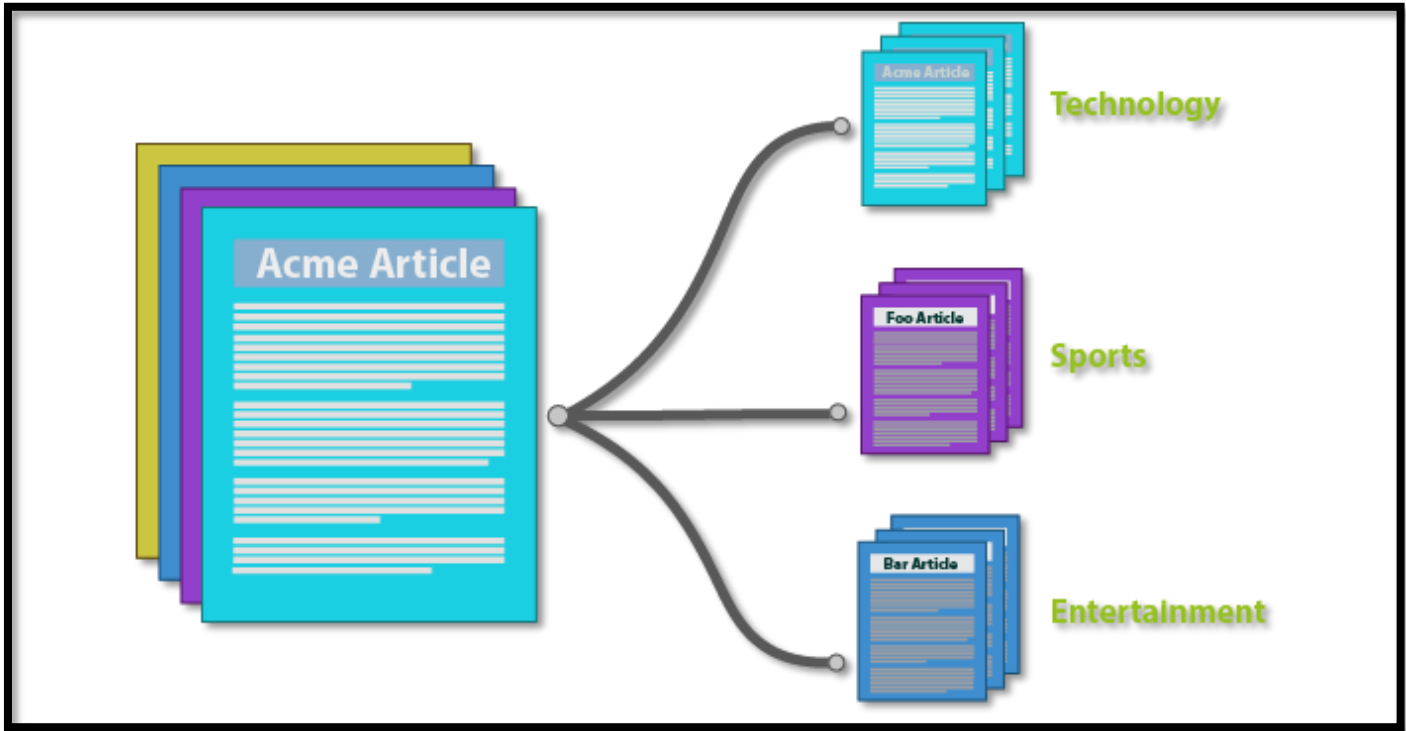
Semantic textual similarity

- Why does it matter?

The screenshot shows a Google search interface. The search bar contains the text "how thin is a dollar bill", with the word "thin" highlighted by a red box. Below the search bar, navigation links for "Alle", "Shopping", "Bilder", "News", "Videos", "Mehr", "Einstellungen", and "Suchfilter" are visible. The search results indicate "Ungefähr 21.600.000 Ergebnisse (0,88 Sekunden)". A featured snippet is displayed, containing the text: "1. U.S. paper currency such as a \$1 bill measures 2.61 inches wide by 6.14 inches long with a thickness of .0043 inches." In this snippet, the word "currency" is crossed out, "2.61" is replaced by "2.6", and the word "inches" is highlighted by a red box. Below the snippet, the source URL "https://www.ehd.org > ... > Technology Articles" and the title "Grasping Large Numbers" are shown. At the bottom of the snippet, there are links for "Informationen zu hervorgehobenen Snippets" and "Feedback geben". Below the snippet, a section titled "Ähnliche Fragen" (Similar Questions) lists four related queries, each with a dropdown arrow: "How thick is a 1 dollar bill?", "How thick is a \$50 bill?", "Can a dollar bill shrink?", and "Is a dollar bill two pieces of paper?". A "Feedback geben" link is located at the bottom right of the page.

Semantic textual similarity

- Applications
 - Plagiarism detection
 - Document clustering
 - Question answering



```
Similarity by term frequency: 97.713468057248      Semantic Similarity: 61.29641283826306
Now a package/folder with the name shape will be created in the current directory and these class files
will be placed in it.

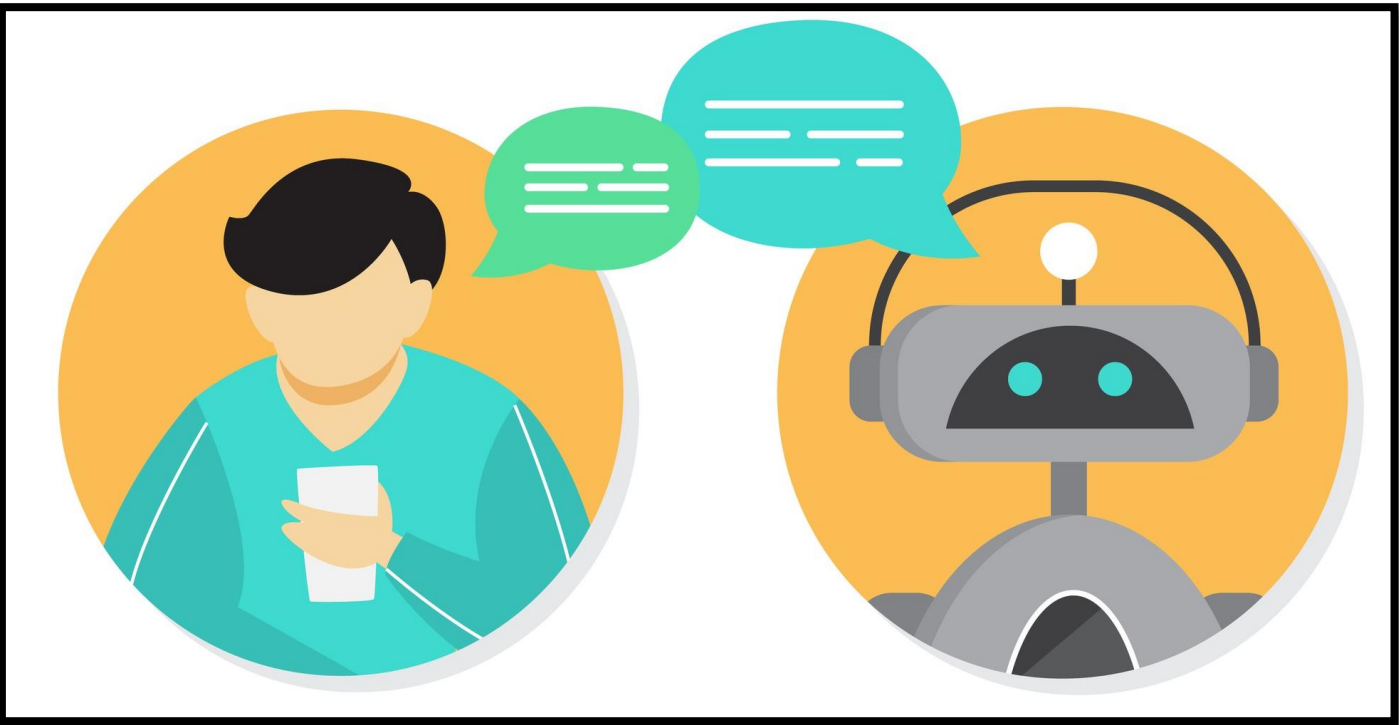
(d) java :
java :
A subclass inherits all the member variables and methods from its super classes. It can use the inherited
methods and variables as they are. It may also override an inherited method by providing its own version
or hide an inherited variable by defining a variable of the same name.
The "@Override" is known as annotation (introduced in JDK 1.5), which asks compiler to check whether
there is such a method in the superclass to be overridden. This helps greatly if you misspell the name of
the method to be overridden. For example, suppose that you wish to override method toString() in a
subclass. If @Override is not used and toString() is misspelled as toString(), it will be treated as a new
method in the subclass, instead of overriding the superclass. If @Override is used, the compiler will
signal an error.
@Override annotation is optional.
Annotations are not programming constructs. They have no effect on the program output. It is only used
by the compiler, discarded after compilation, and not used by the runtime.
C++ :
With the override function called in the subclass, the original function is also called.

(e) The super keyword in Java is a reference variable that is used to refer immediate parent class object.
Whenever you create the instance of subclass, an instance of parent class is created implicitly i.e.
referred by super reference variable. Similarly, the keyword super refers to the superclass, which could
be the immediate parent or its ancestor. The keyword super allows the subclass to access superclass'
methods and variables within the subclass' definition.
If the subclass overrides a method inherited from
its super class, says getArea(), you can use super.getArea() to invoke the superclass' version within the
subclass definition. Similarly, if your subclass hides one of the superclass' variable, you can use super.
variableName to refer to the hidden variable within the subclass definition.

(f) In OOP, we often organize classes in hierarchy to avoid duplication and reduce redundancy. The
classes in the lower hierarchy inherit all the variables (static attributes) and methods (dynamic
behaviors) from the higher hierarchies. A class in the lower hierarchy is called a subclass (or derived
child, extended class). A class in the upper hierarchy is called a superclass (or base, parent class). By
pulling out all the common variables and methods into the super classes, and leave the specialized
variables and methods in the subclasses, redundancy can be greatly reduced or eliminated as these
common variables and methods do not need to be repeated in all the subclasses.

(g) Dynamic dispatch contrasts with static dispatch in which the implementation of a polymorphic
operation is selected at compile time. The purpose of dynamic dispatch is to support cases where the
appropriate implementation of a polymorphic operation can't be determined at compile time because it
depends on the runtime type of one or more actual parameters to the operation.
Dynamic method dispatch is a mechanism by which a call to an overridden method is resolved at
runtime. This is how java implements runtime polymorphism. When an overridden method is called by a
reference, java determines which version of that method to execute based on the type of object it refer
to. In simple words the type of object which it referred determines which version of overridden method
will be called.

(h) The word "polymorphism" means "many forms". Polymorphism just means that, basically, once you've
got a child class, you can use objects of that child class wherever you'd use objects of the parent class.
```



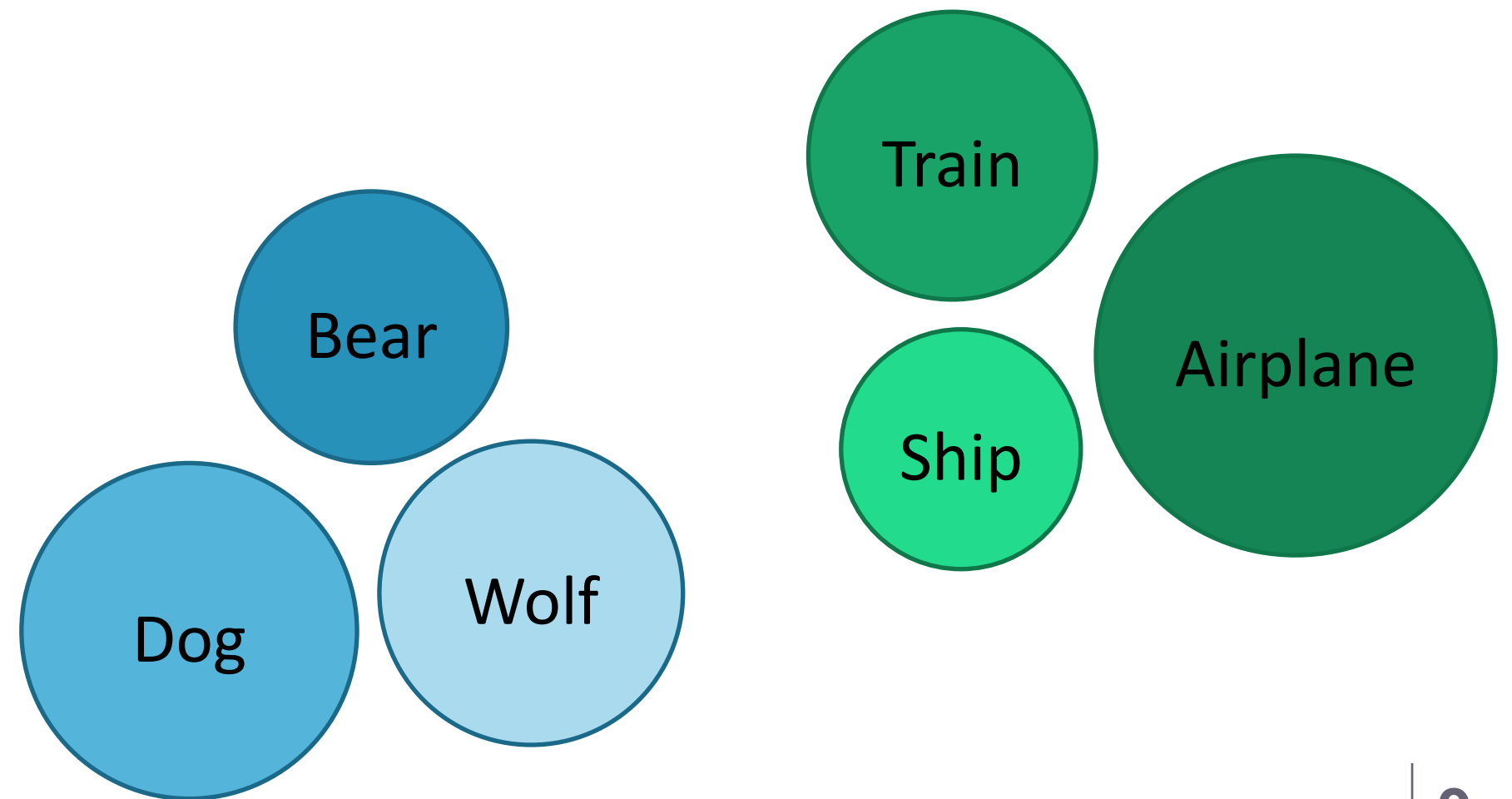
Semantic textual similarity

- What is semantic similarity?
- Semantic similarity in word level
- Semantic similarity in sentence level
- Semantic textual similarity in Python

Semantic similarity in word level

- It tells how close two words/terms are, semantically
- Semantic similarity is often used ***synonymously*** with semantic ***relatedness***

Ship	Airplane	3.8
Ship	Bear	0.2
Dog	Wolf	4.5
Wolf	Bear	3.6



Semantic similarity in word level

- The approaches can be divided into the following categories:
 - Distributional semantics
 - Frequency based
 - Prediction based
 - Knowledge based methods

Distributional semantics

- Frequency based
- PMI

$$PMI(W_1, W_2) = \log_2 \frac{P(W_1, W_2)}{P(W_1)P(W_2)}$$

Distributional semantics

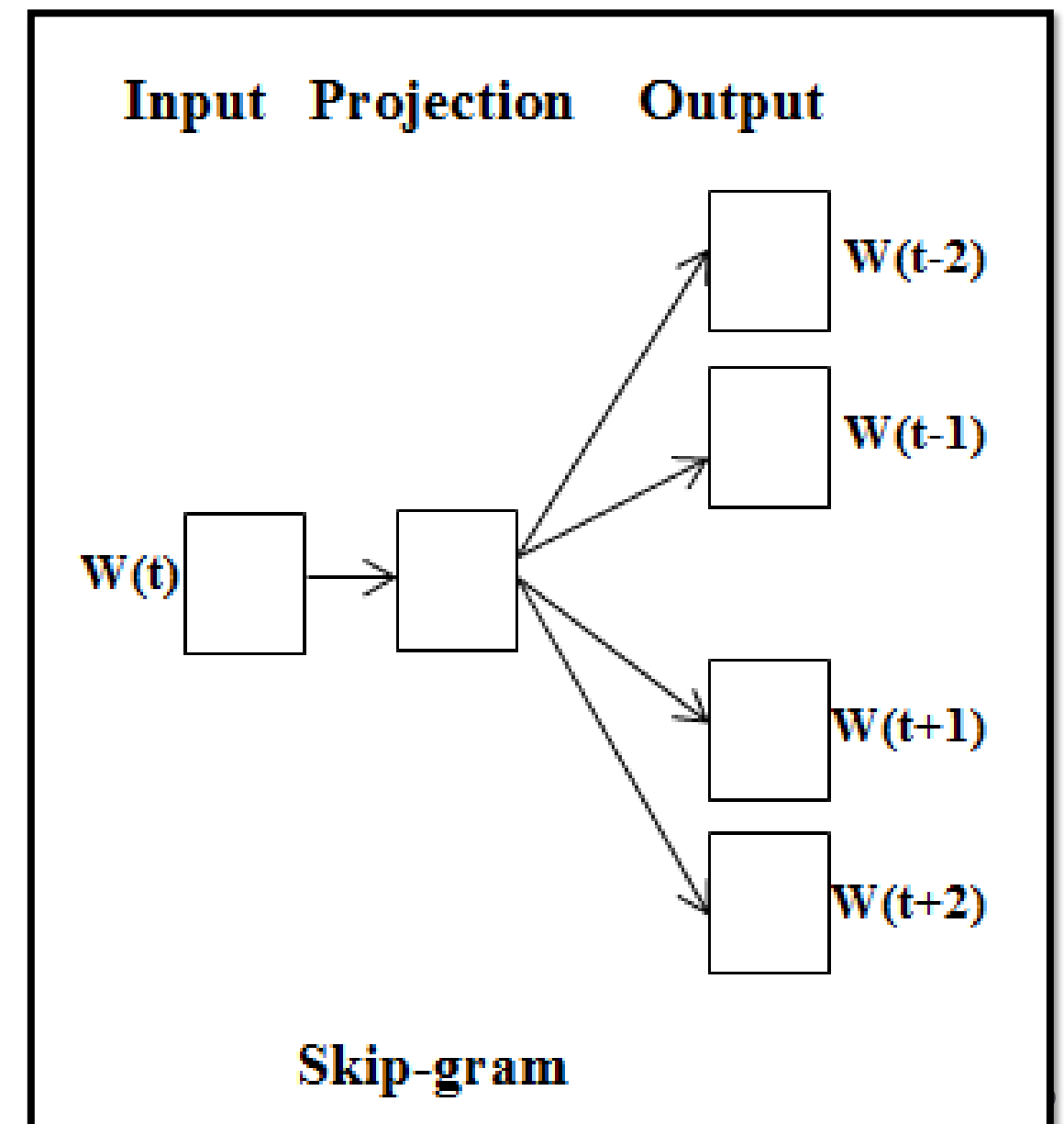
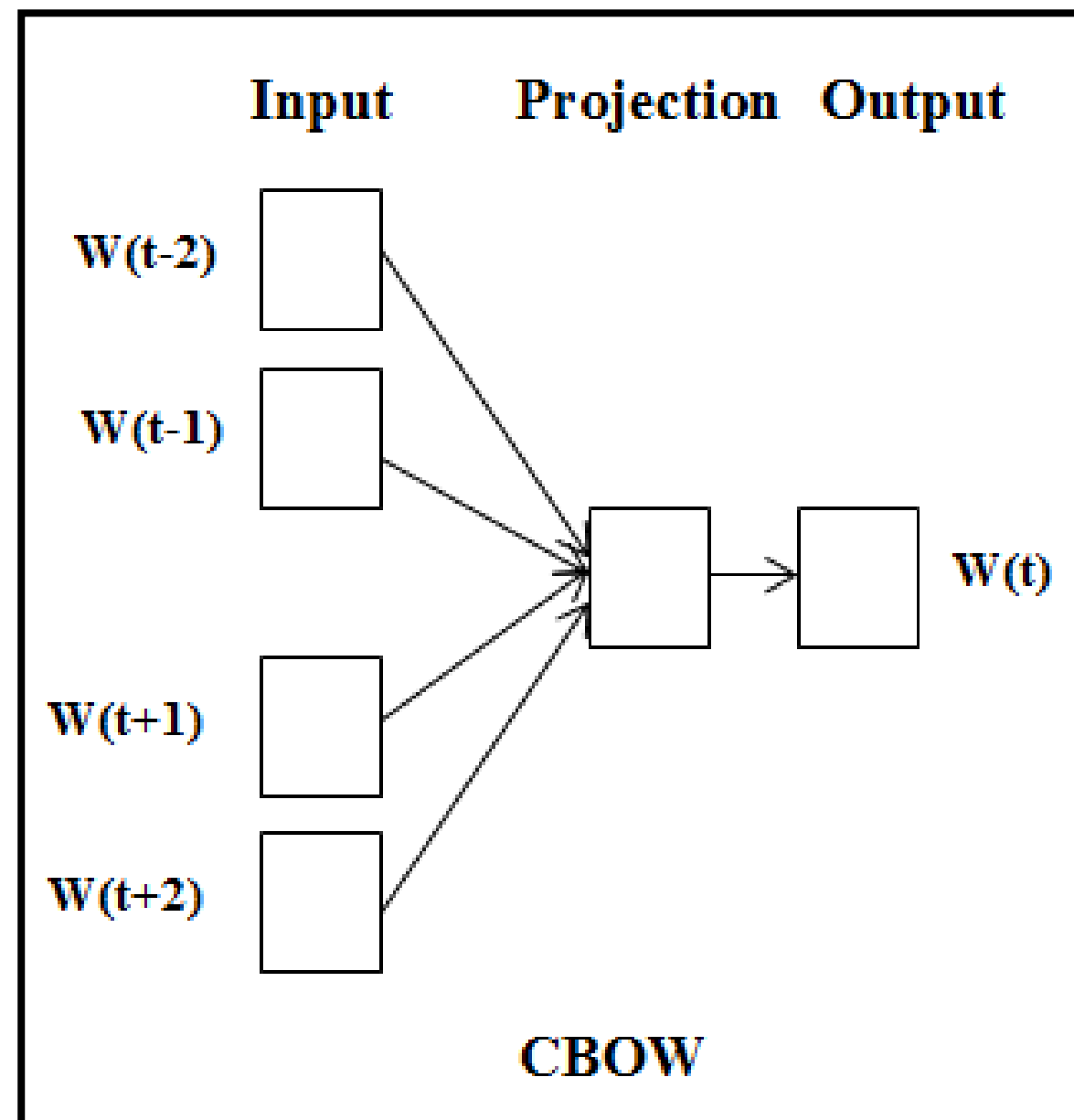
	text	is	a	complex	human	language	representation	natural	and	also	diverse
text	0	1	1	0	0	0	0	0	0	0	0
is	1	0	1	1	1	1	0	0	1	0	0
a	1	1	0	1	1	0	0	0	0	0	0
complex	0	2	1	0	1	2	0	0	1	1	0
human	0	1	1	1	0	2	1	1	0	0	0
language	0	1	0	2	2	0	1	1	0	0	0
representation	0	0	0	0	1	0	0	0	0	0	0
natural	0	0	0	0	1	0	0	0	0	0	0
and	0	2	0	1	0	0	0	0	0	1	0
also	0	1	0	1	0	0	0	0	1	0	1
diverse	0	1	0	0	0	0	0	0	0	1	0

$PMI = -0.51$

$PMI = +1.8$

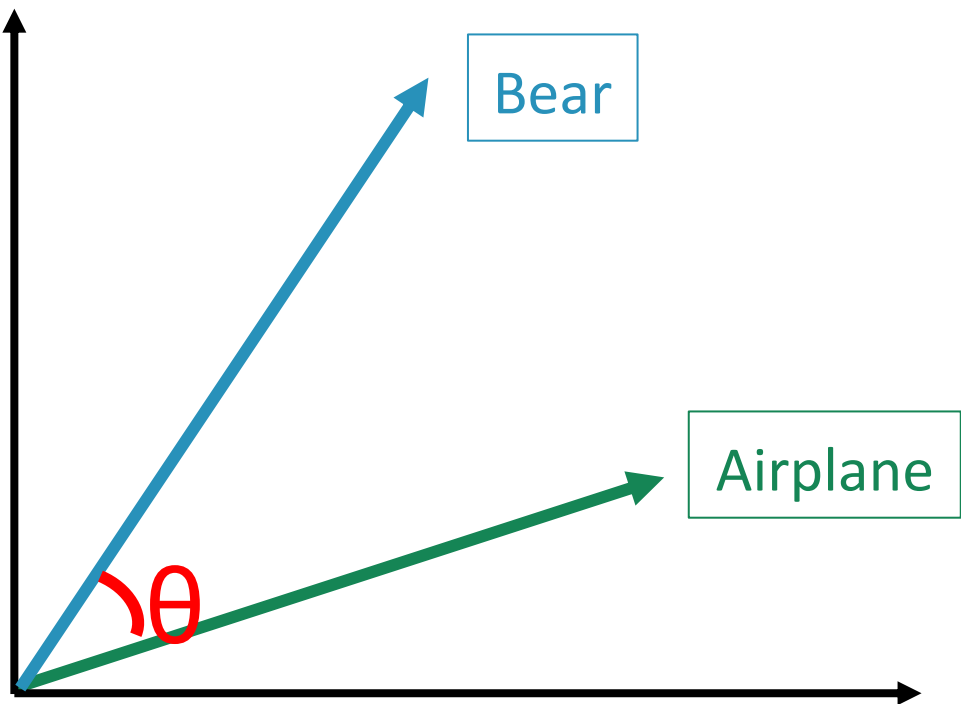
Distributional semantics

- Frequency based
 - PMI
- Prediction based
 - Word2Vec



Distributional semantics

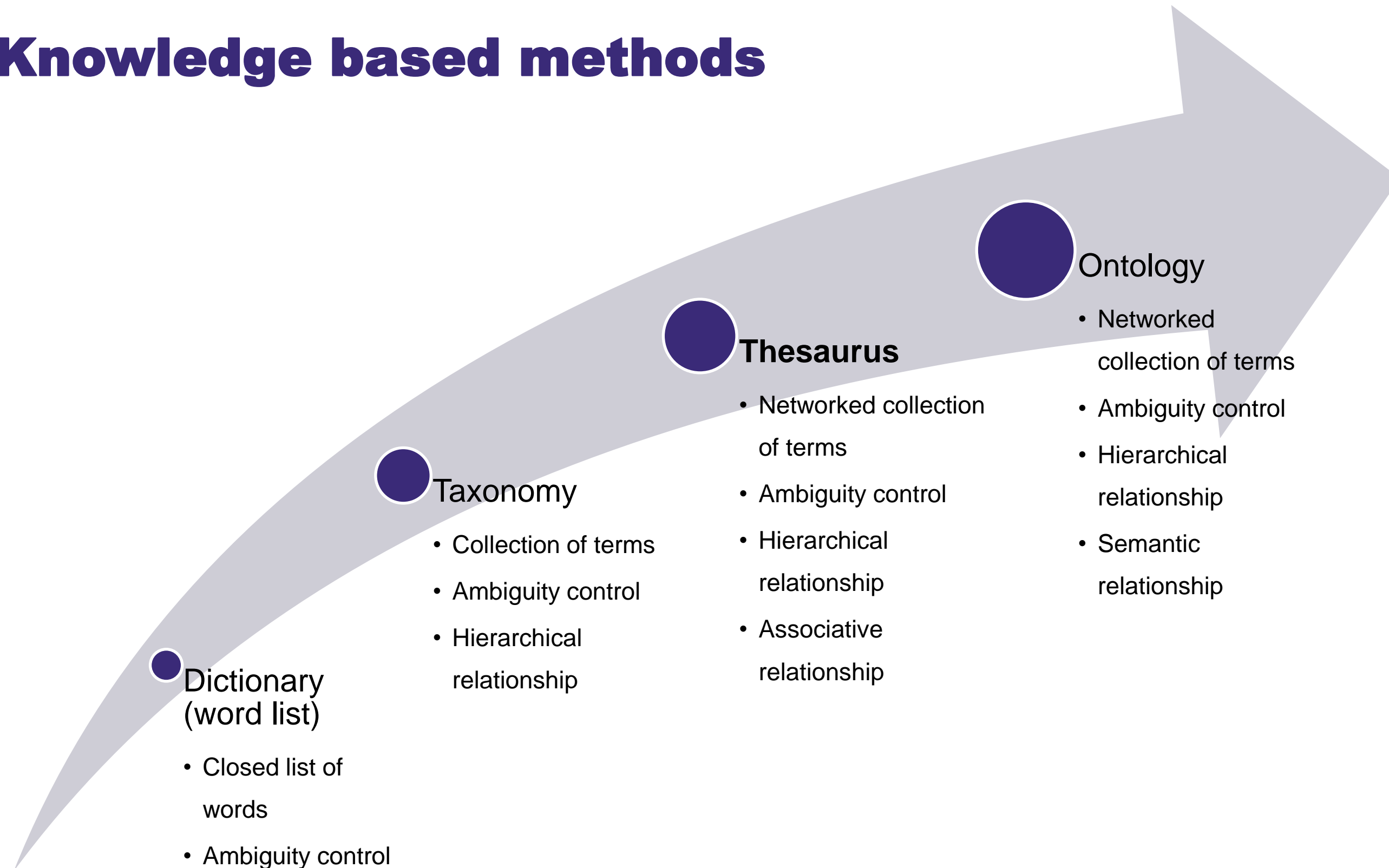
Ship	0.003	- 0.01	0.001	...	0.321	- 0.076	0.014
Airplane	0.002	- 0.009	- 0.001	...	0.337	- 0.054	0.014
Wolf	0.469	0.015	0.373	...	- 0.049	0.533	- 0.148
Dog	0.143	0.445	0.180	...	- 0.683	0.167	- 0.428
Bear	0.397	0.236	- 0.110	...	- 0.256	0.257	- 0.148



Knowledge based methods

- Knowledge-based semantic similarity methods calculate semantic similarity between two terms based on the information derived from underlying knowledge sources like ***ontology***, ***thesaurus***, ***taxonomy***, ***dictionary***
- Ontology, thesaurus, taxonomy, dictionary
- Machine readable knowledge sources that represents how objects are related

Knowledge based methods



Knowledge based methods

Dictionary (term list)

rework

/ri:'wɜ:k/ ⓘ

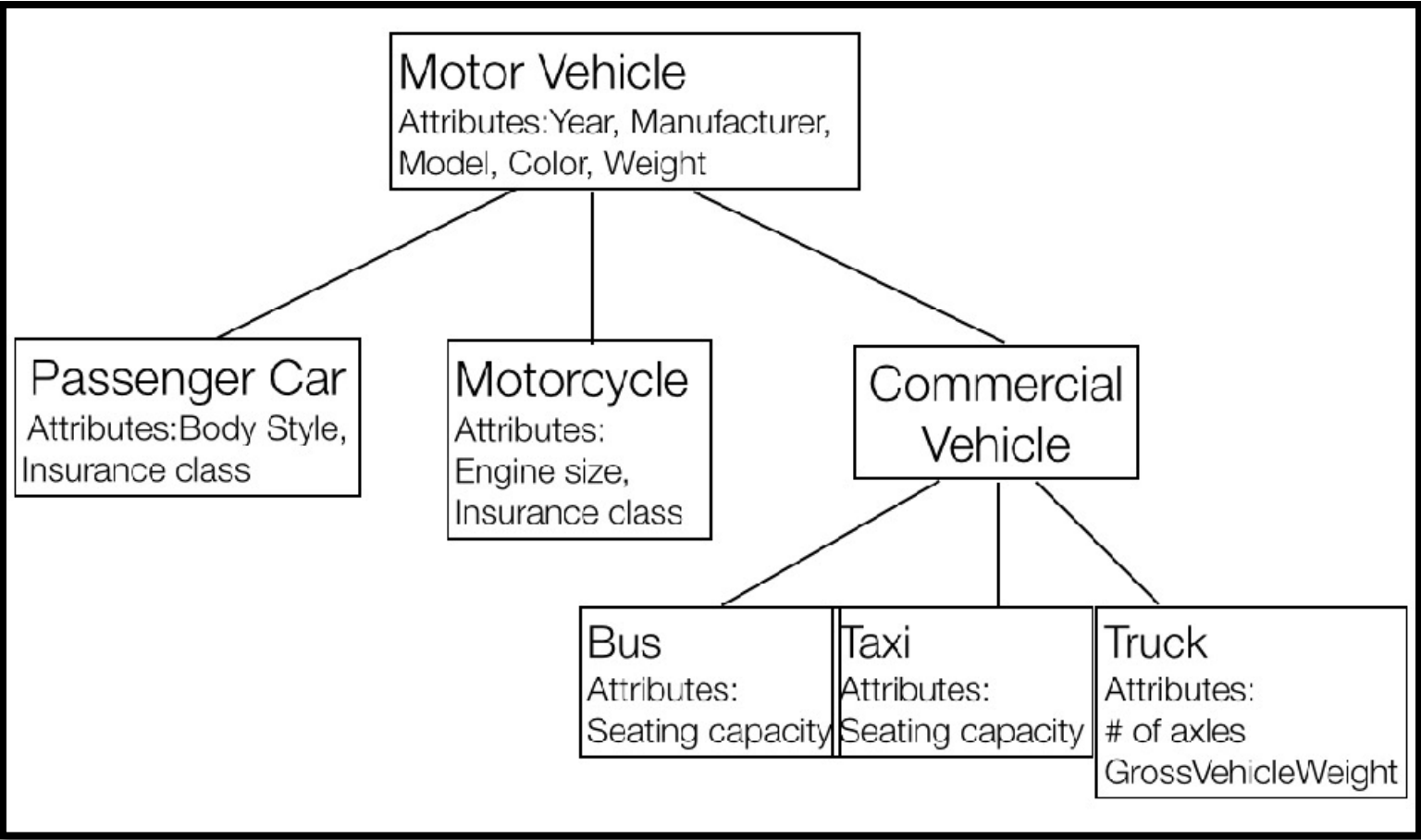
verb

Make changes to the original version of (something)

Sample: Over the course of our trip, the President continually reworked his speech.

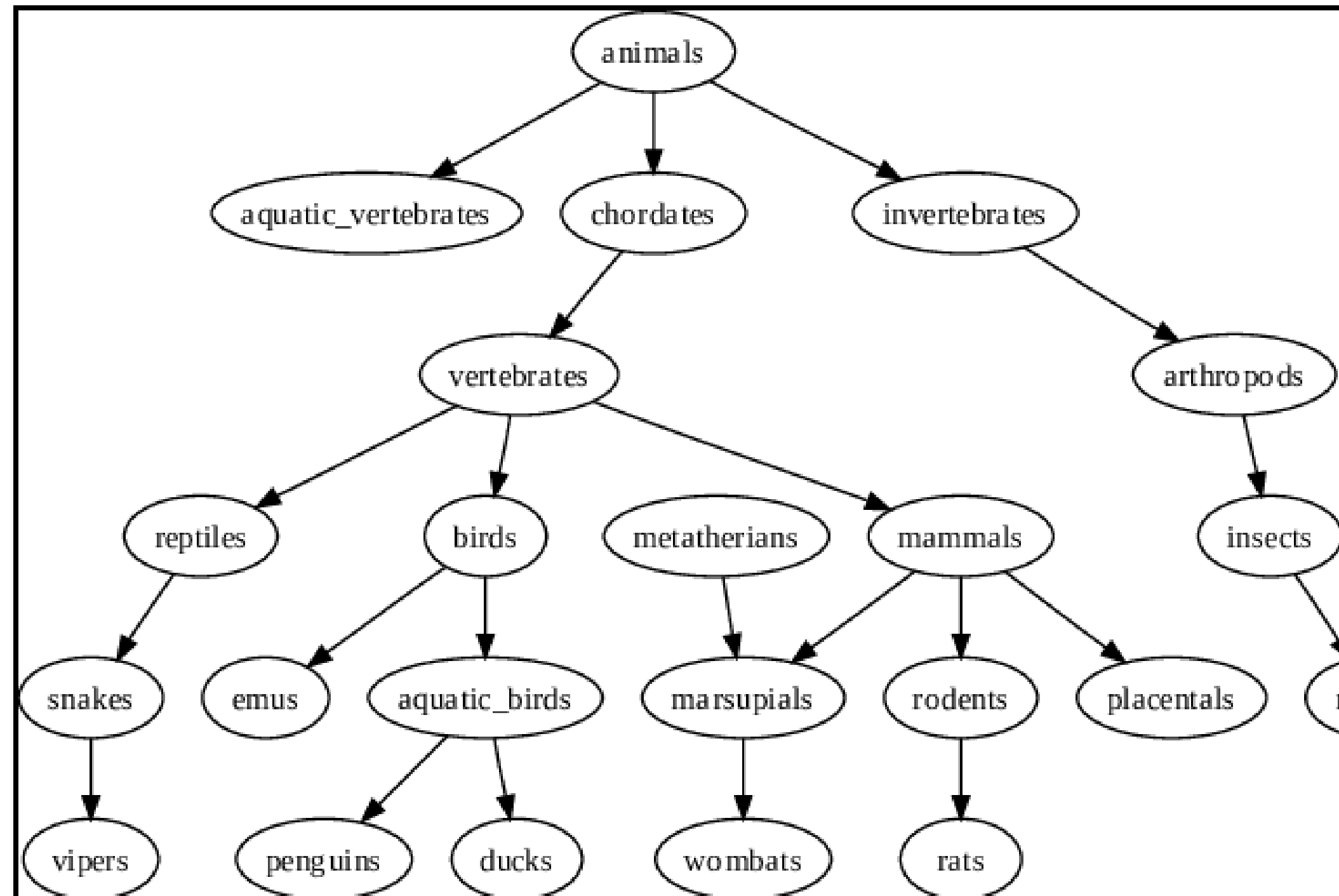
Sample: He reworked the orchestral score for two pianos.

Taxonomy



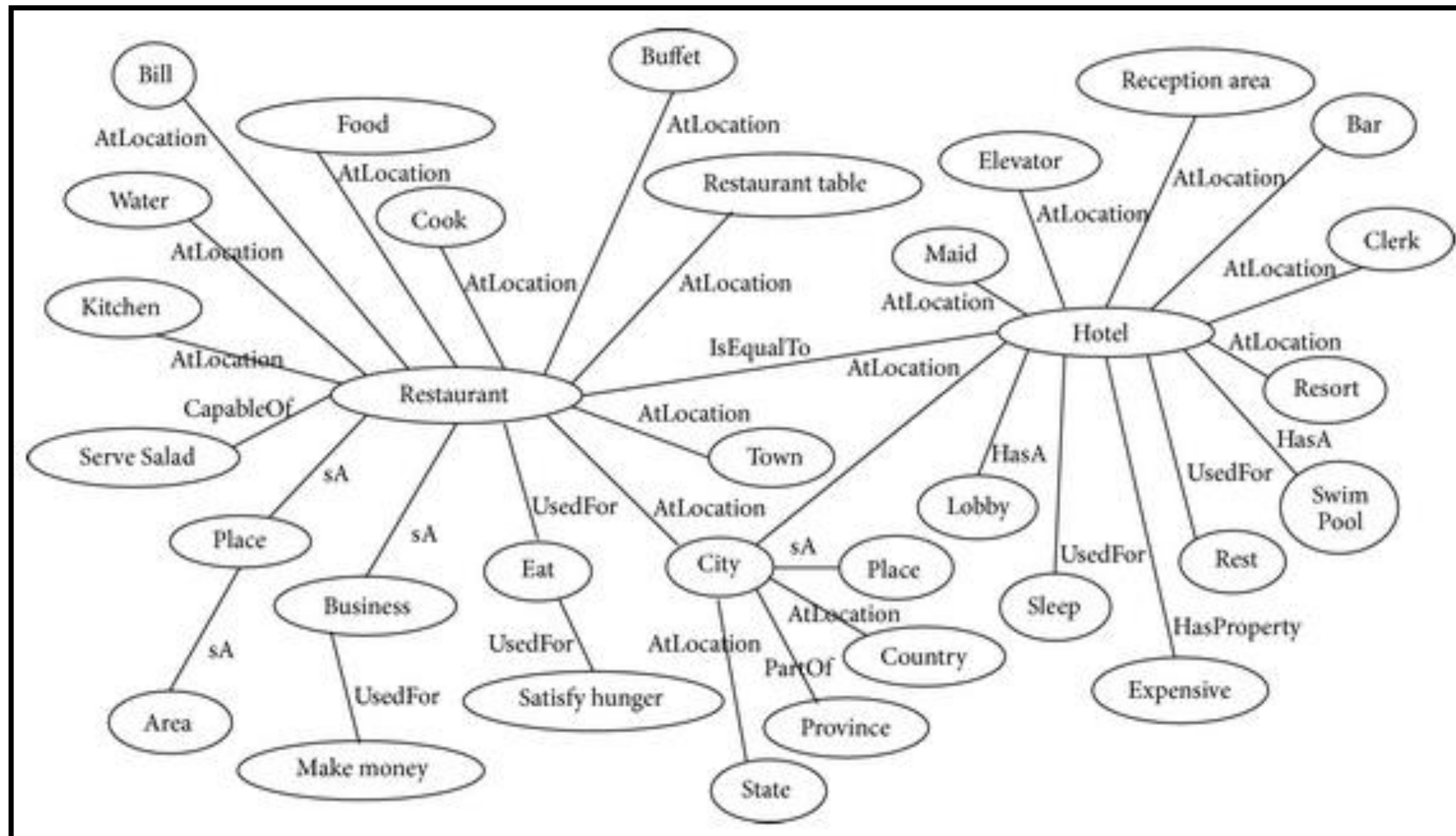
Knowledge based methods

Thesaurus



Knowledge based methods

Ontology



Knowledge based methods

- Thesaurus
- Wordnet
 - WordNet® is a large lexical database of English
 - Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive **synonyms** (**synsets**), each expressing a distinct concept
 - Synsets are interlinked by means of conceptual-semantic and lexical relations

Knowledge based methods

Java

Noun

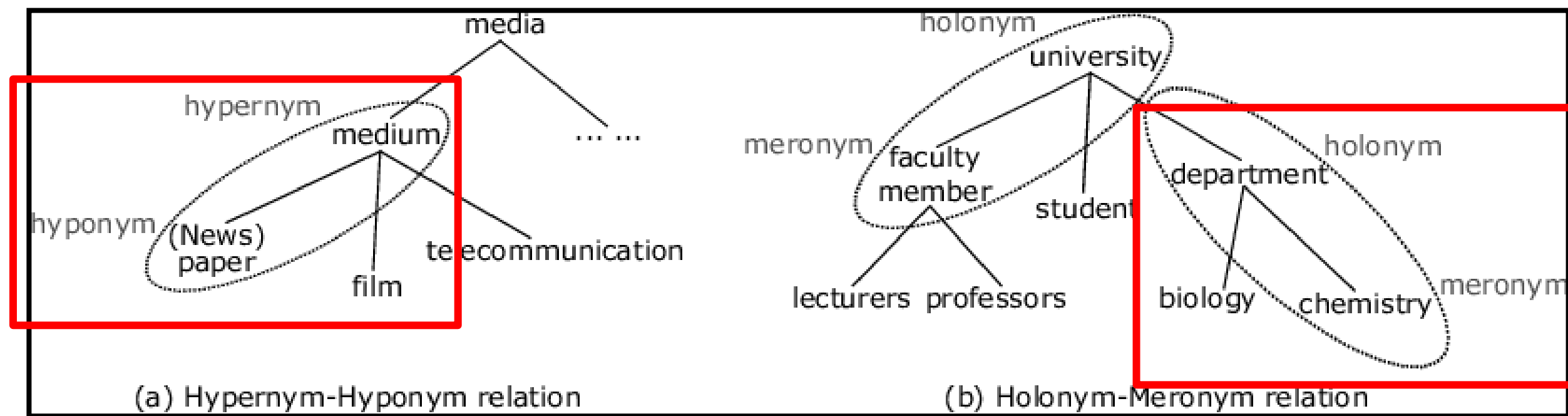
- S: (n) **Java** (an island in Indonesia to the south of Borneo; one of the world's most densely populated regions)
- S: (n) coffee, **java** (a beverage consisting of an infusion of ground coffee beans) "he ordered a cup of coffee"
- S: (n) **Java** (a platform-independent object-oriented programming language)

- S: (n) **Java** (a platform-independent object-oriented programming language)
 - direct hyponym / inherited hyponym / sister term
 - S: (n) object-oriented programming language, object-oriented programming language ((computer science) a programming language that enables the programmer to associate a set of procedures with each type of data structure) "C++ is an object-oriented programming language that is an extension of C"

- S: (n) coffee, **java** (a beverage consisting of an infusion of ground coffee beans) "he ordered a cup of coffee"
 - direct hyponym / full hyponym
 - S: (n) coffee substitute (a drink resembling coffee that is sometimes substituted for it)
 - S: (n) Irish coffee (sweetened coffee with Irish whiskey and whipped cream)
 - S: (n) cafe au lait (equal parts of coffee and hot milk)
 - S: (n) cafe noir, demitasse (small cup of strong black coffee without milk or cream)
 - S: (n) decaffeinated coffee, decaf (coffee with the caffeine removed)
 - S: (n) drip coffee (coffee made by passing boiling water through a perforated container packed with finely ground coffee)
 - S: (n) espresso (strong black coffee brewed by forcing hot water under pressure through finely ground coffee beans)
 - S: (n) cappuccino, cappuccino coffee, coffee cappuccino (equal parts of espresso and hot milk topped with cinnamon and nutmeg and usually whipped cream)
 - S: (n) iced coffee, ice coffee (a strong sweetened coffee served over ice with cream)
 - S: (n) instant coffee (dehydrated coffee that can be made into a drink by adding hot water) "the advantages of instant coffee are speed of preparation and long shelf life"
 - S: (n) mocha, mocha coffee (a superior dark coffee made from beans from Arabia)
 - S: (n) Turkish coffee (a drink made from pulverized coffee beans; usually sweetened)
 - S: (n) cafe royale, coffee royal (black coffee with Cognac and lemon peel and sugar)

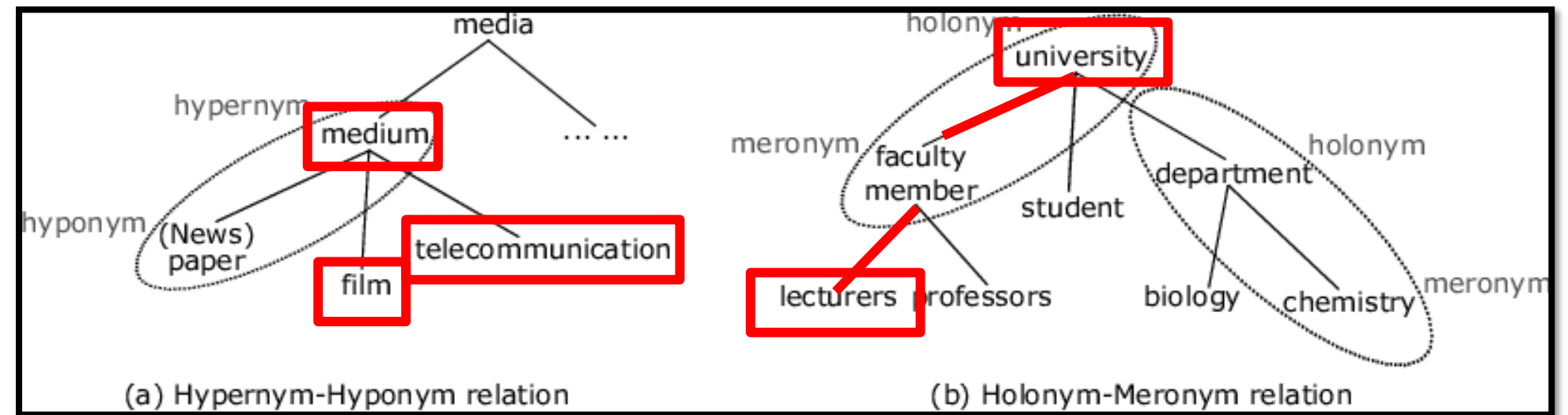
Knowledge based methods

- Semantic relations in Wordnet
 - **Hypernyms:** Y is a hypernym of X if every X is a (kind of) Y (vehicle is a hypernym of bicycle)
 - **Hyponyms:** Y is a hyponym of X if every Y is a (kind of) X (bicycle is a hyponym of vehicle)
 - **Meronym:** Y is a meronym of X if Y is a part of X (window is a meronym of building)
 - **Holonym:** Y is a holonym of X if X is a part of Y (building is a holonym of window)



Knowledge based methods

- How to capture semantic similarity in wordnet?
 - *path* measure
 - *wup* measure



$$sim_{path}(W_1, W_2) = \frac{1}{1 + \min_length(W_1, W_2)}$$

$$sim_{wup}(W_1, W_2) = \frac{2 \times \text{depth}(\text{Least Common Subsumer})}{\text{depth}(W_1) + \text{depth}(W_2)}$$

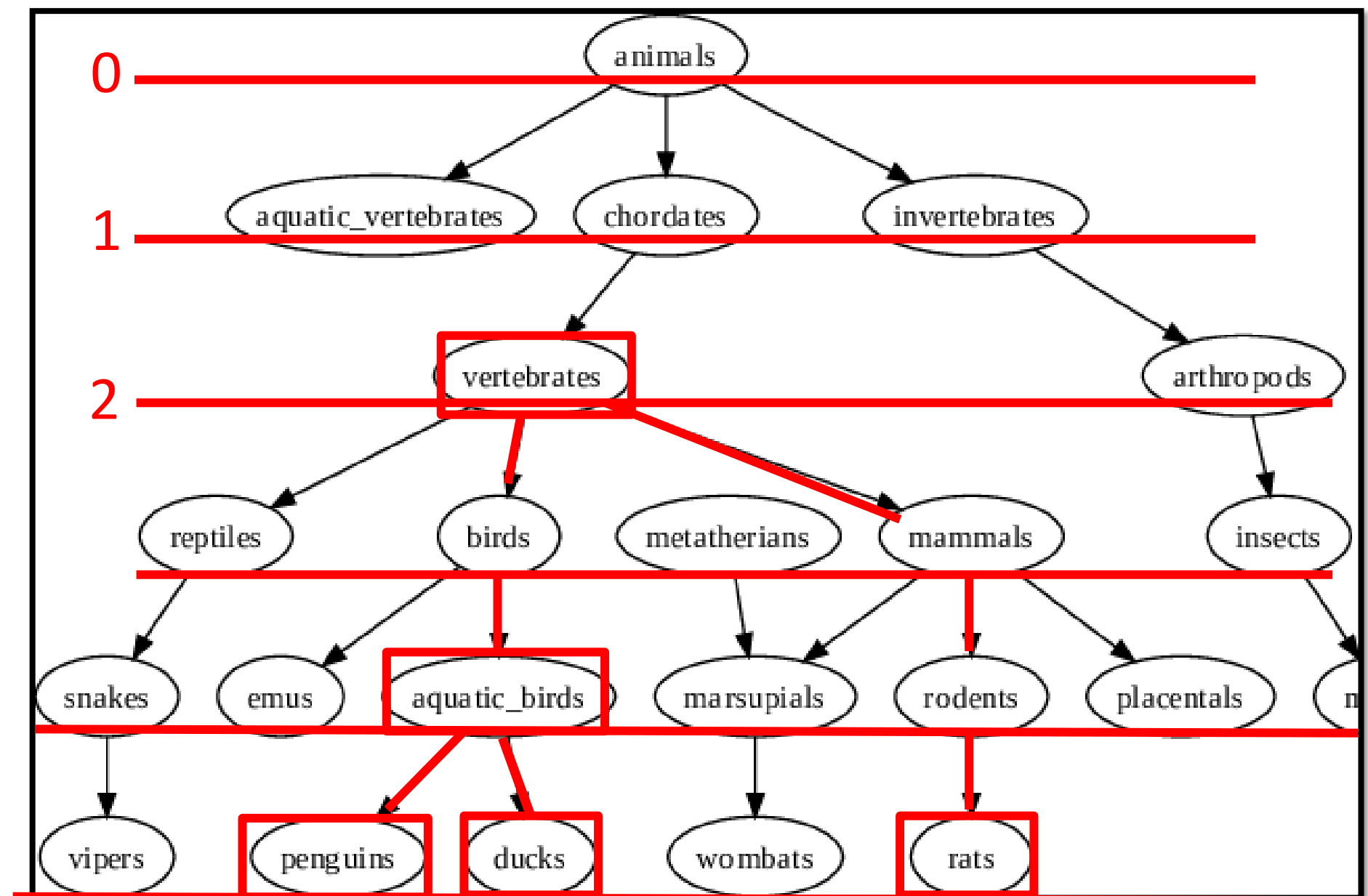
Knowledge based methods

$$sim_{path}(duck, rat) = \frac{1}{7}$$

$$sim_{path}(penguin, duck) = \frac{1}{3}$$

$$sim_{wup}(duck, rat) = \frac{2 \times 2}{5 + 5} = 0.4$$

$$sim_{wup}(penguin, duck) = \frac{2 \times 4}{5 + 5} = 0.8$$

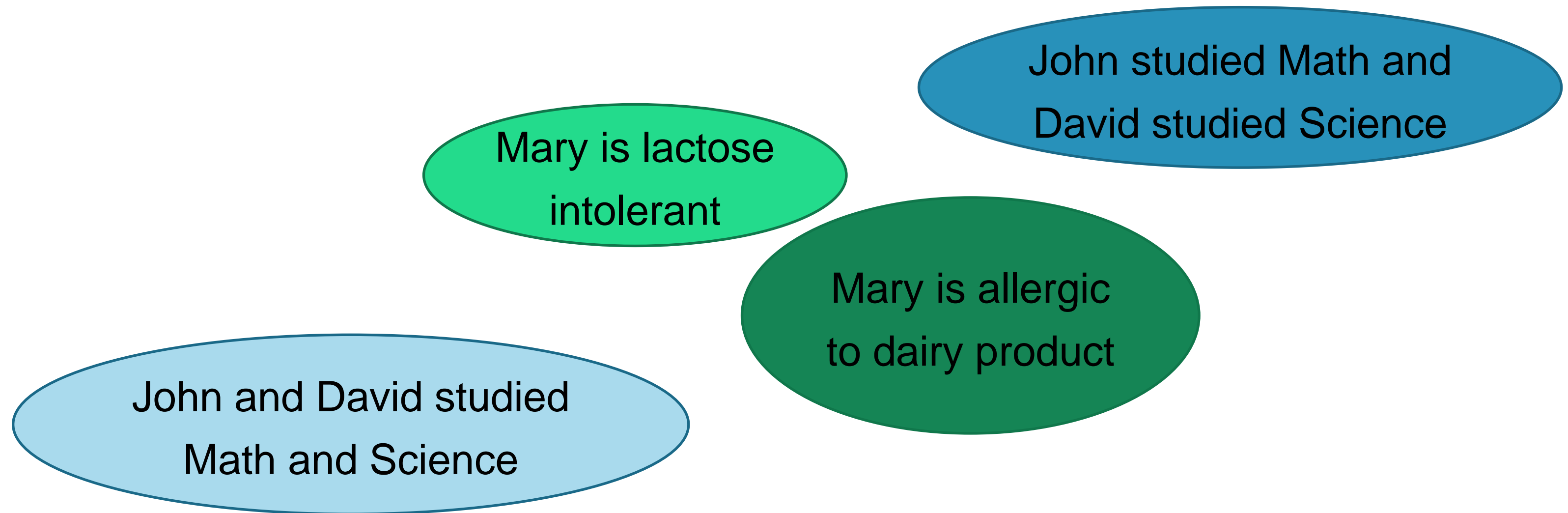


Semantic textual similarity

- What is semantic similarity?
- Semantic similarity in word level
- Semantic similarity in sentence level
- Semantic textual similarity in Python

Semantic similarity in sentence level

- It tells how close two texts (sentences) are, semantically



Semantic similarity in sentence level

- The approaches can be divided into the following categories:
 - Distributional semantics
 - Knowledge based methods

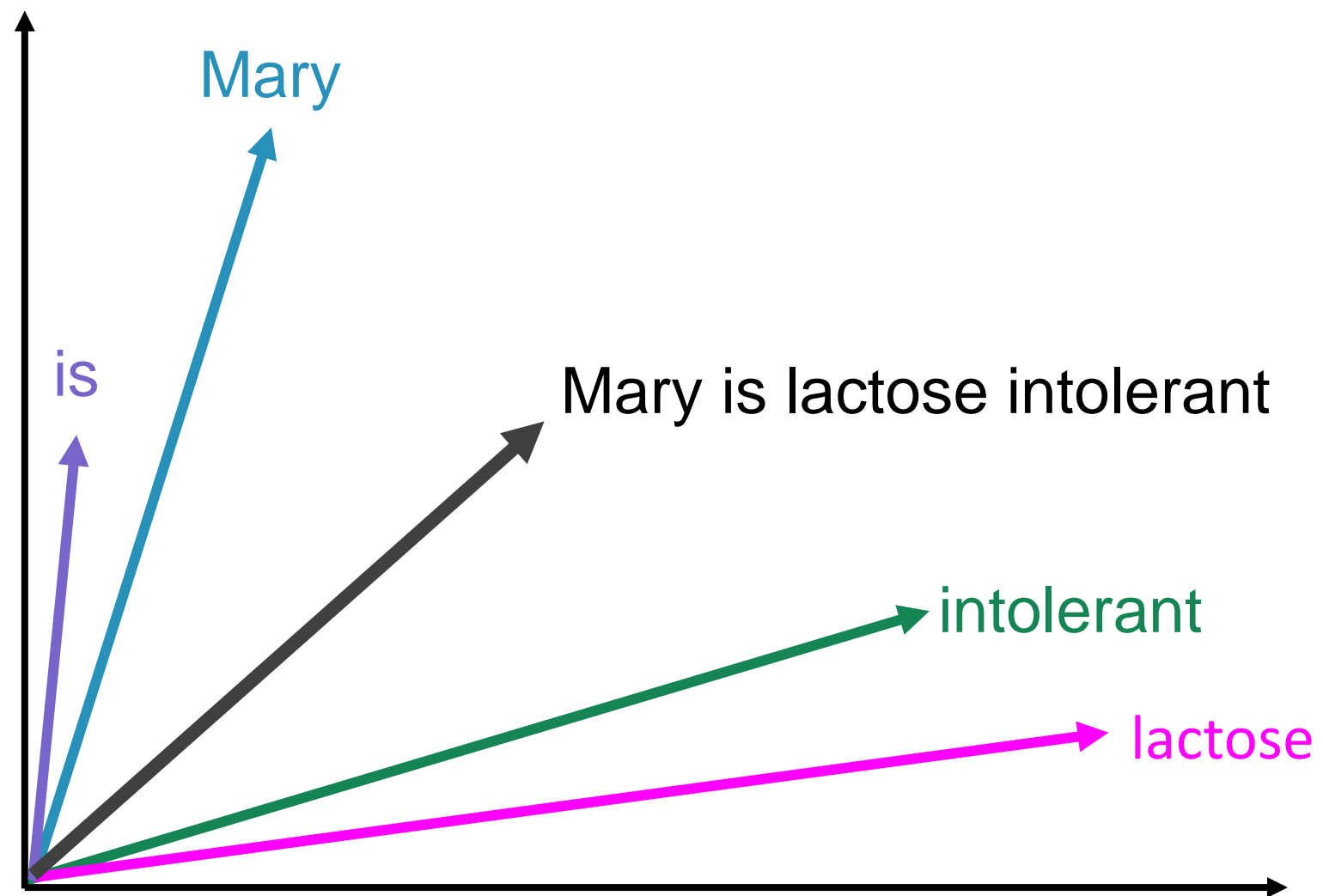
Distributional semantics

- Computing sentence vectors
 - Average of word vectors
 - Average of word vectors with TF-IDF
 - Doc2Vec

Distributional semantics

- Average of word vectors

Mary is lactose intolerant



Distributional semantics

- Average of word vectors

Mary is lactose intolerant

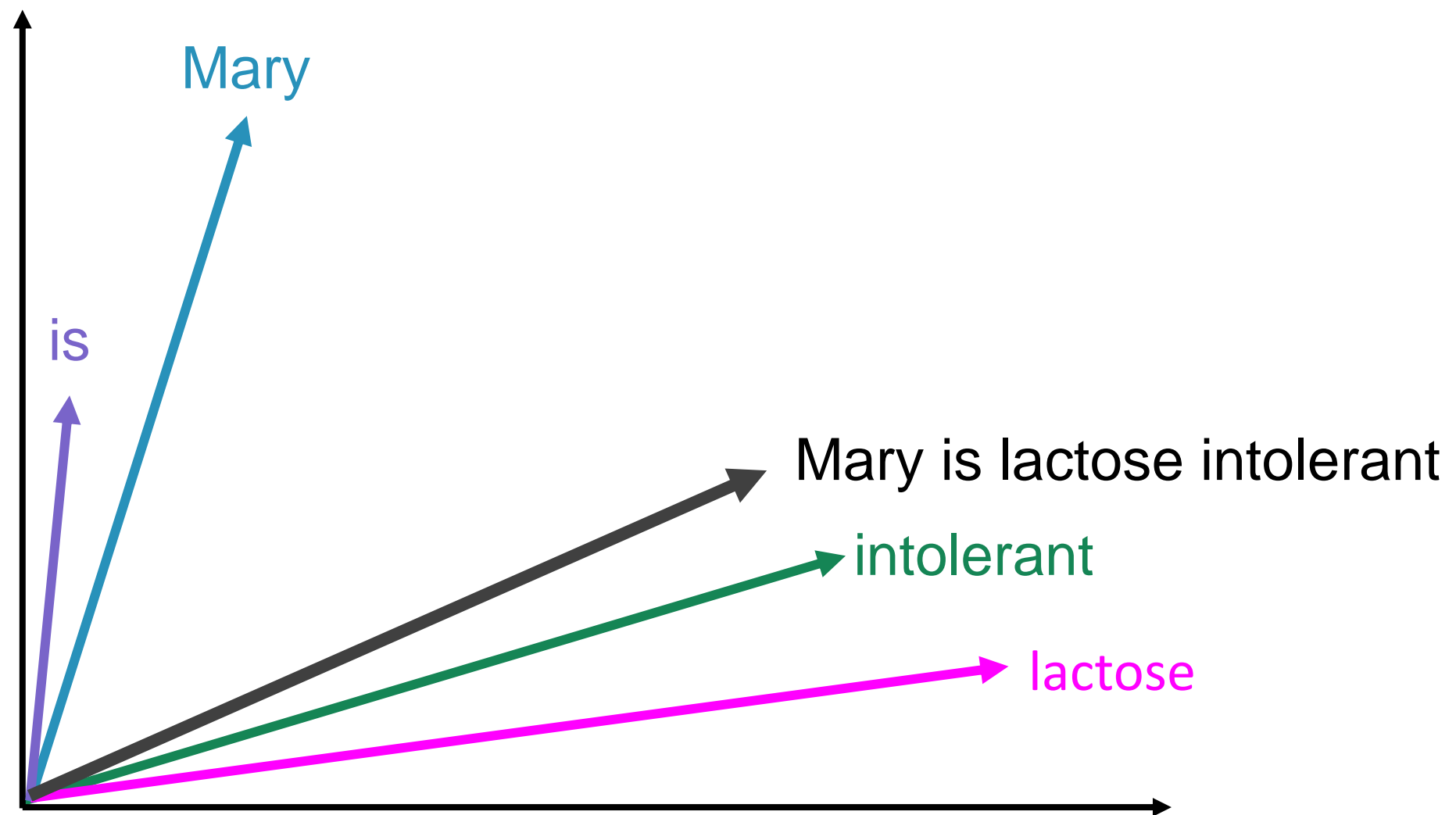
Mary	1	0	1	3	5
is	1	4	2	2	2
lactose	0	3	7	1	1
intolerant	2	1	1	1	2
Sentence	1	2	2.75	1.7	2.5

Distributional semantics

- Average of word vectors with TF-IDF

	TFIDF
Mary	0.5
is	0.1
lactose	5
intolerant	7

Mary is lactose intolerant



Distributional semantics

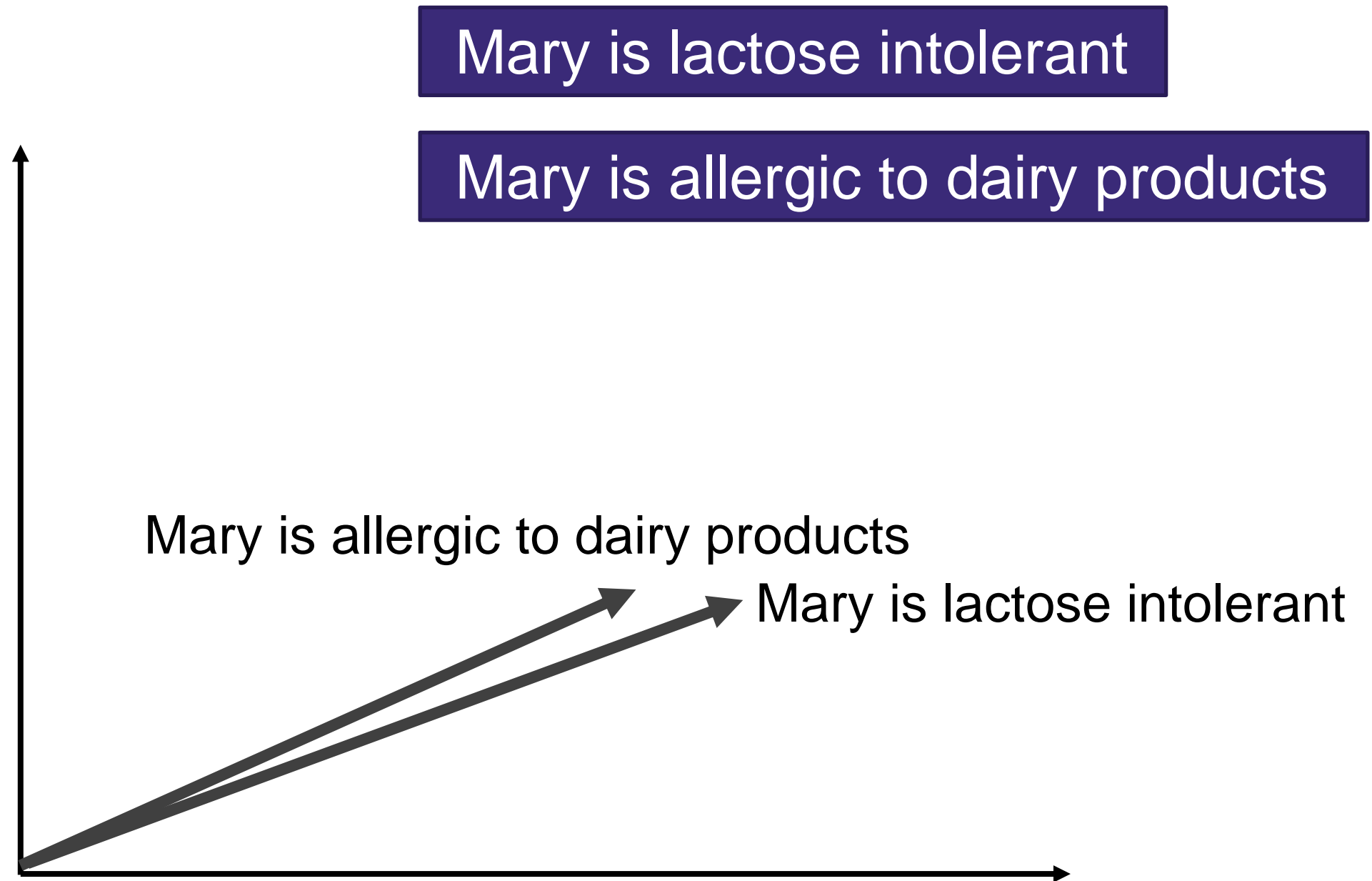
- Average of word vectors with TF-IDF

Mary is lactose intolerant

	TF-IDF					
Mary	0.5	1	0	1	3	5
is	0.1	1	4	2	2	2
lactose	5	0	3	7	1	1
intolerant	7	2	1	1	1	2
Sentence	-	1.15	1.93	3.30	1.15	1.74

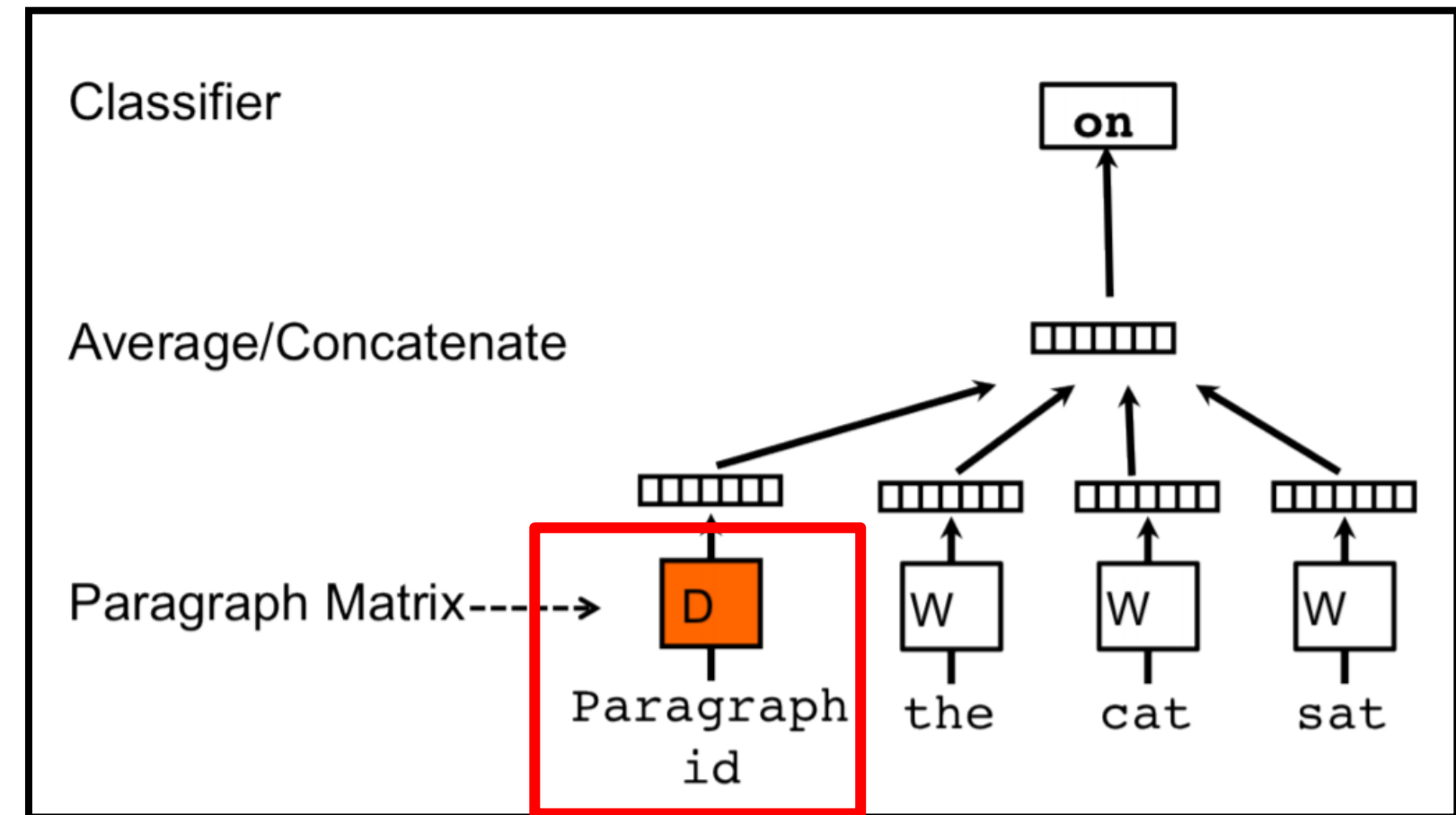
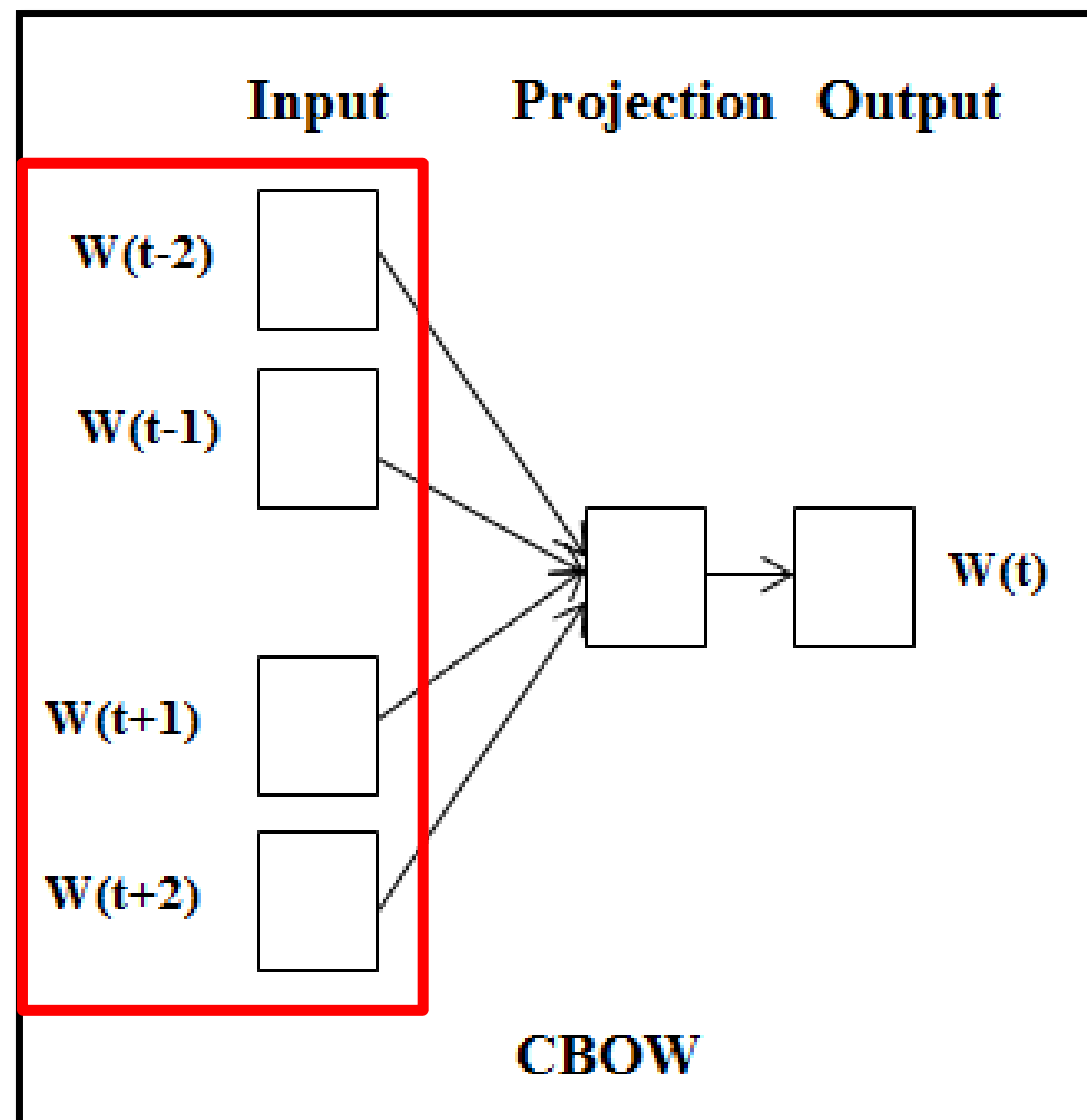
Distributional semantics

- Cosine similarity



Distributional semantics

- Doc2Vec

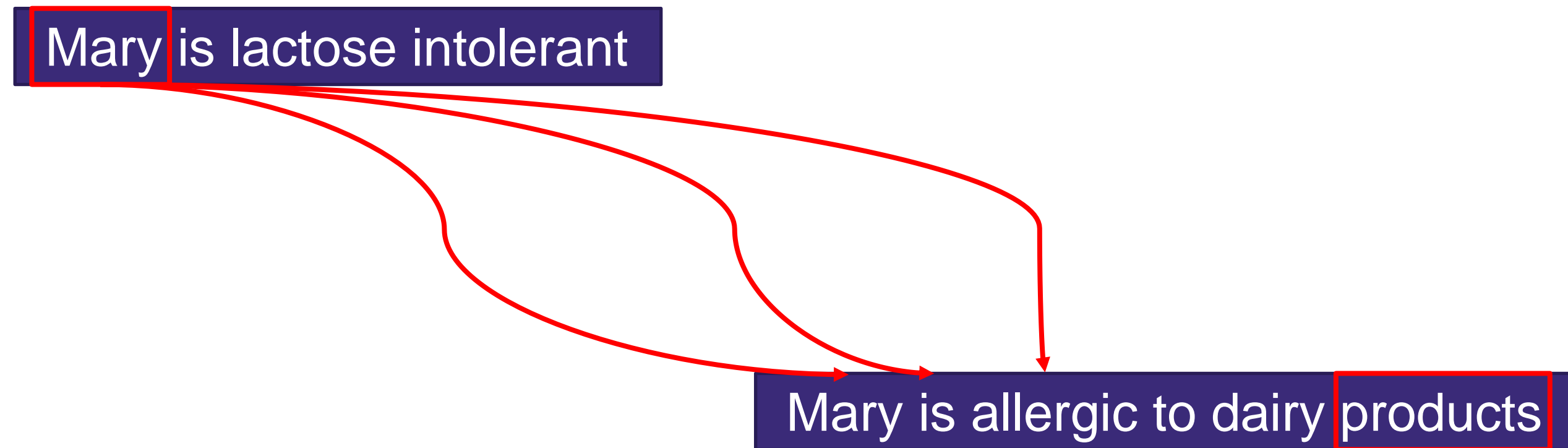


Distributional semantics

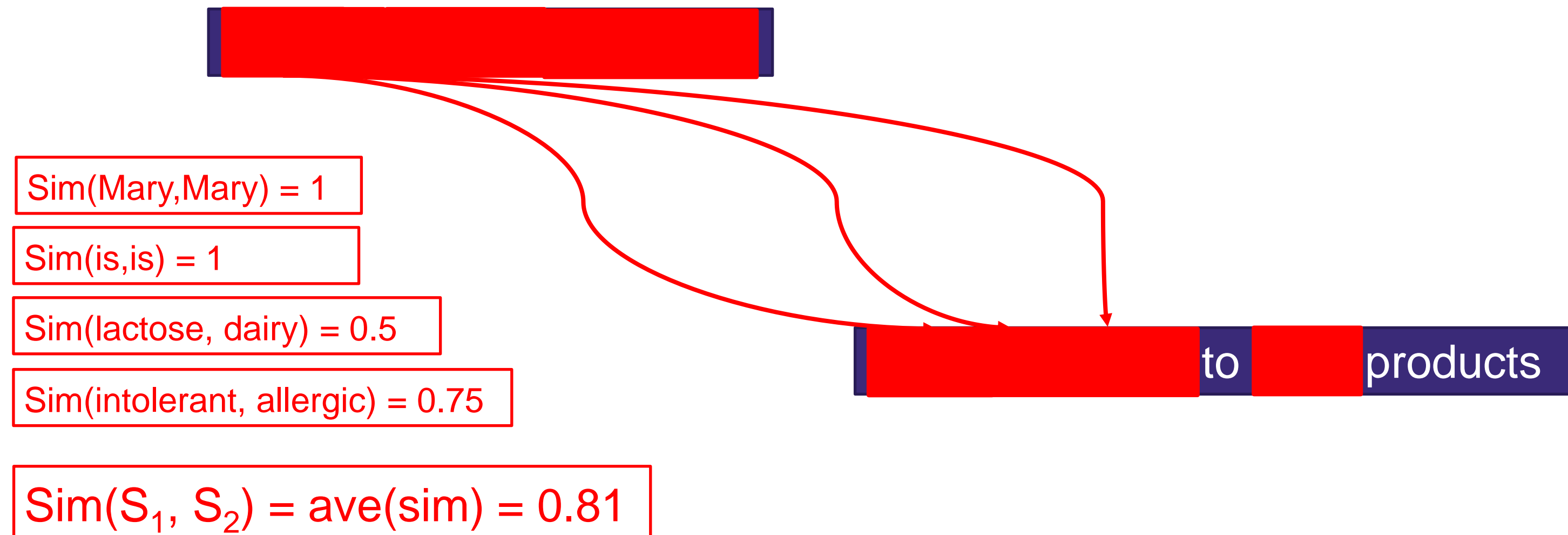
- Doc2Vec

Mary is lactose intolerant	0.112	0.091	0.357	...	- 0.483	0.249	0.747
Mary is allergic to dairy products	0.818	0.343	0.108	...	- 0.777	- 0.310	0.314

Knowledge based methods



Knowledge based methods



Semantic textual similarity

- What is semantic similarity?
- Semantic similarity in word level
- Semantic similarity in sentence level
- Semantic textual similarity in Python

NLTK

- Wordnet synset

```
>>> from nltk.corpus import wordnet
>>> wordnet.synsets('dog')
[Synset('dog.n.01'), Synset('frump.n.01'), Synset('dog.n.03'),
Synset('cad.n.01'), Synset('frank.n.02'), Synset('paw1.n.01'),
Synset('andiron.n.01'), Synset('chase.v.01')]
>>> print(wordnet.synset('dog.n.01').definition())
a member of the genus Canis (probably descended from the common
wolf) ...
```

Noun

- S: (n) **Java** (an island in Indonesia to the south of Borneo; one of the world's most densely populated regions)
- S: (n) **coffee, java** (a beverage consisting of an infusion of ground coffee beans) *"he ordered a cup of coffee"*
- S: (n) **Java** (a platform-independent object-oriented programming language)

NLTK

- Wordnet similarity

```
>>> from nltk.corpus import wordnet
>>> dog = wordnet.synset('dog.n.01')
>>> cat = wordnet.synset('cat.n.01')
>>> print(dog.path_similarity(cat))
0.2
>>> dog.wup_similarity(cat)
0.85
```


Gensim

- Doc2Vec

```
>>> from gensim.test.utils import common_texts
>>> from gensim.models.doc2vec import Doc2Vec, TaggedDocument
>>> documents = [TaggedDocument(doc, [i]) for i, doc in
                  enumerate(common_texts)]
>>> print(documents[0])
TaggedDocument(['human', 'interface', 'computer'], [0])
>>> model = Doc2Vec(documents, vector_size=5)
>>> vector = model.infer_vector(["semantic", "text", "similarity"])
>>> print(vector)
[-0.07941077, -0.0955774, -0.06963827, -0.02995487, 0.09318832]
```

Summary

- Semantic textual similarity (STS) deals with determining how similar two pieces of texts are

John and David studied Math and Science.

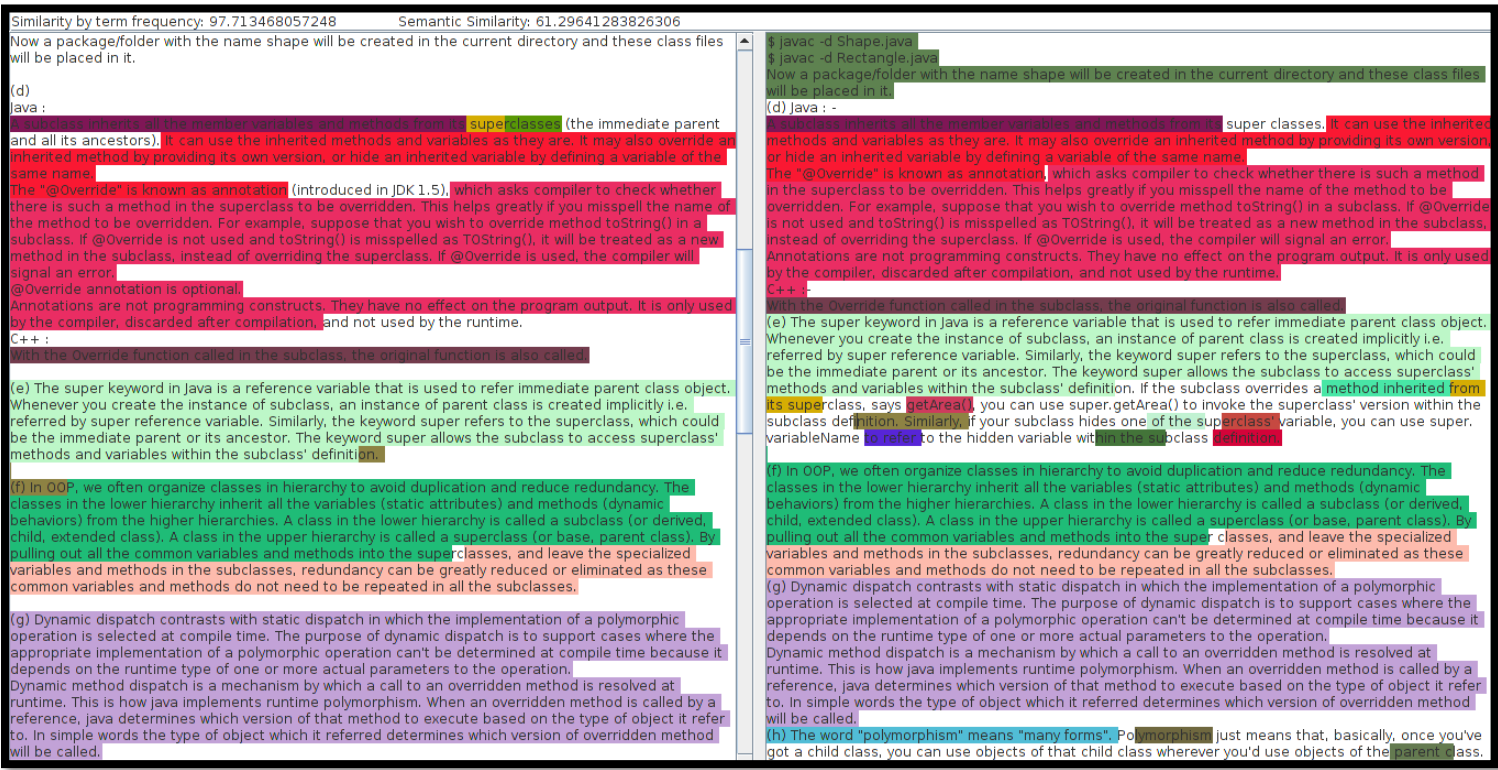
≠

John studied Math and David studied Science.

Mary is allergic to dairy products.

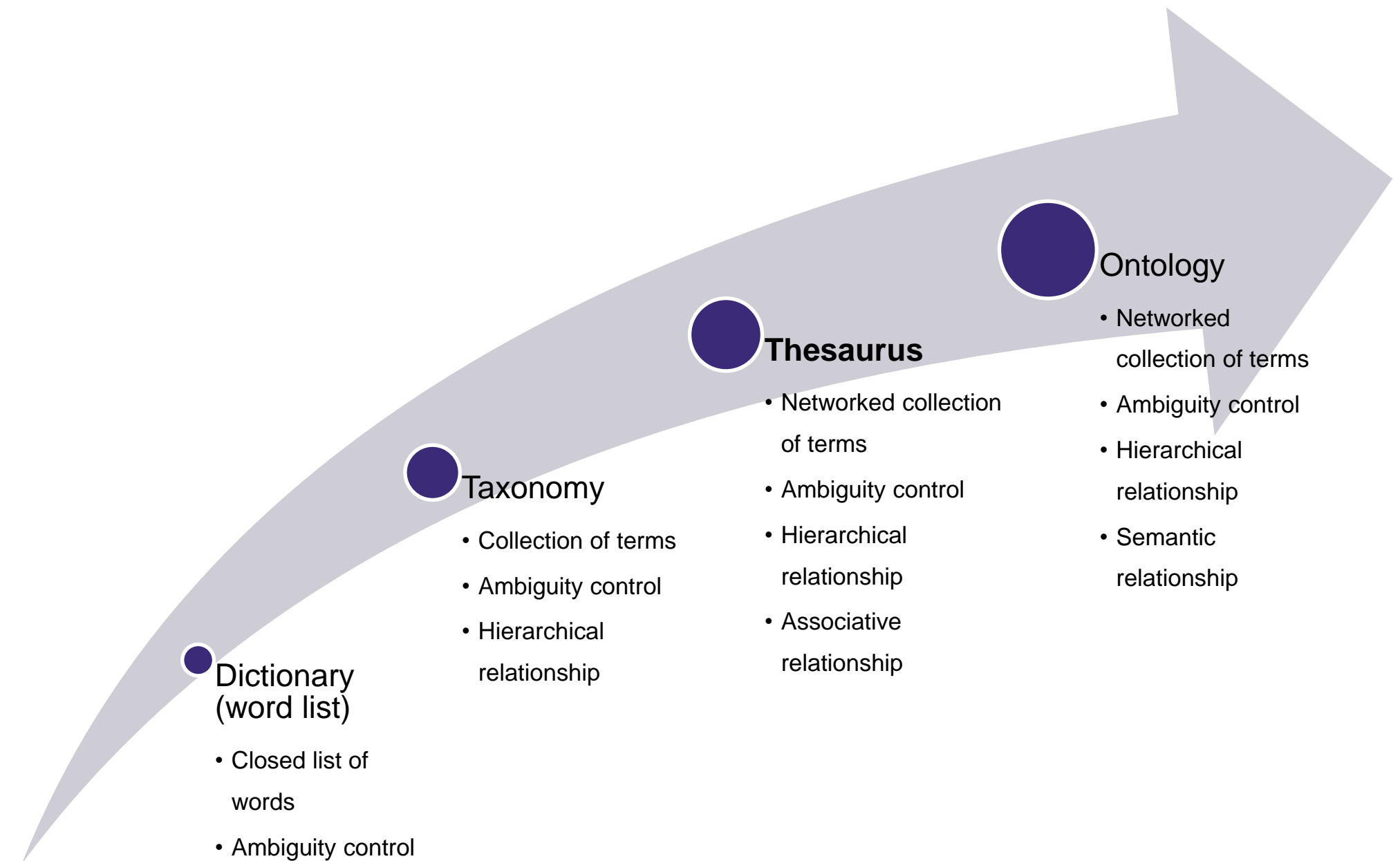
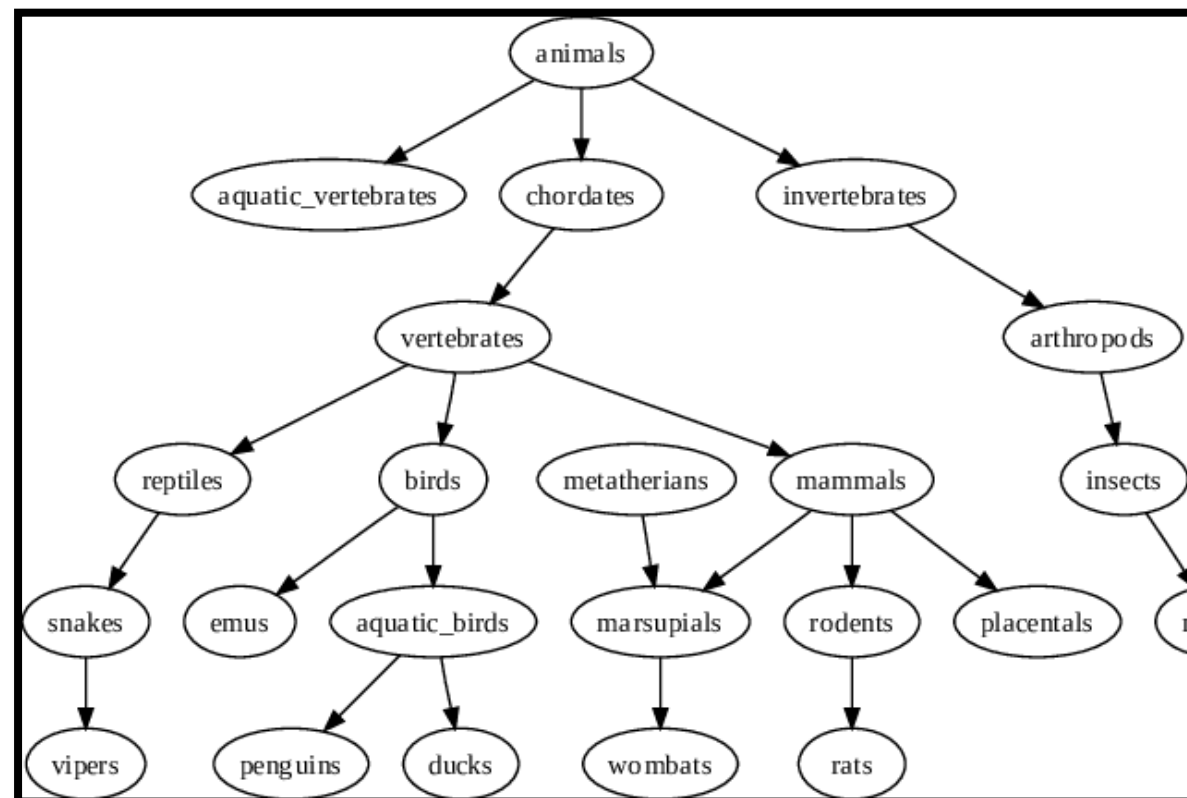
=

Mary is lactose intolerant.



Summary

- Semantic similarity in word level
- Distributional semantics
 - Frequency based
 - Prediction based
- Knowledge based methods



Summary

- Semantic similarity in sentence level
 - Average of word vectors
 - Average of word vectors with TF-IDF
- Doc2Vec
- Knowledge based methods

