UNIVERSITÉ DU LUXEMBOURG

Natural Language Processing

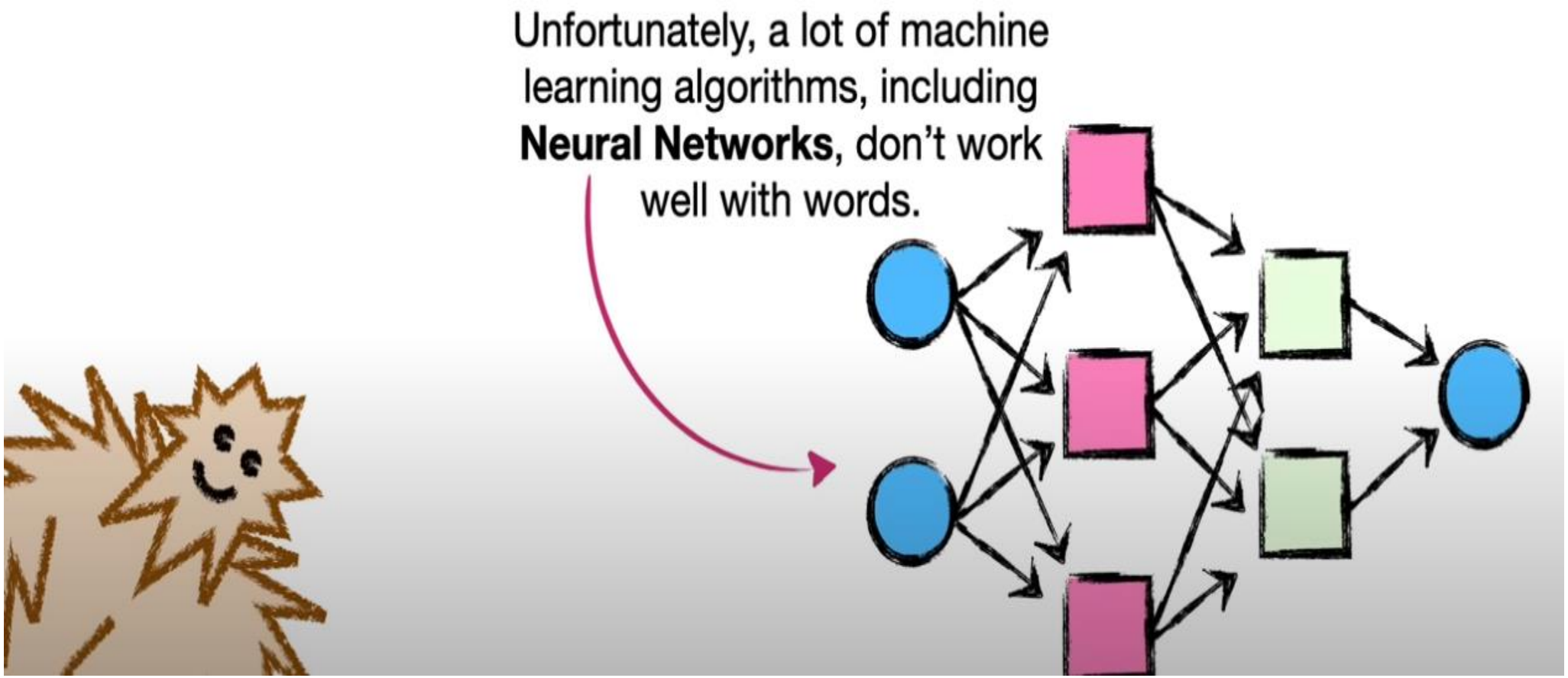
Lecture 3: Text Representation (Part 1)

Salima Lamsiyah
University Luxembourg, FSTM, DCS, MINE Research Group
Salima.lamsiyah@uni.lu

Steps to solve an NLP problem

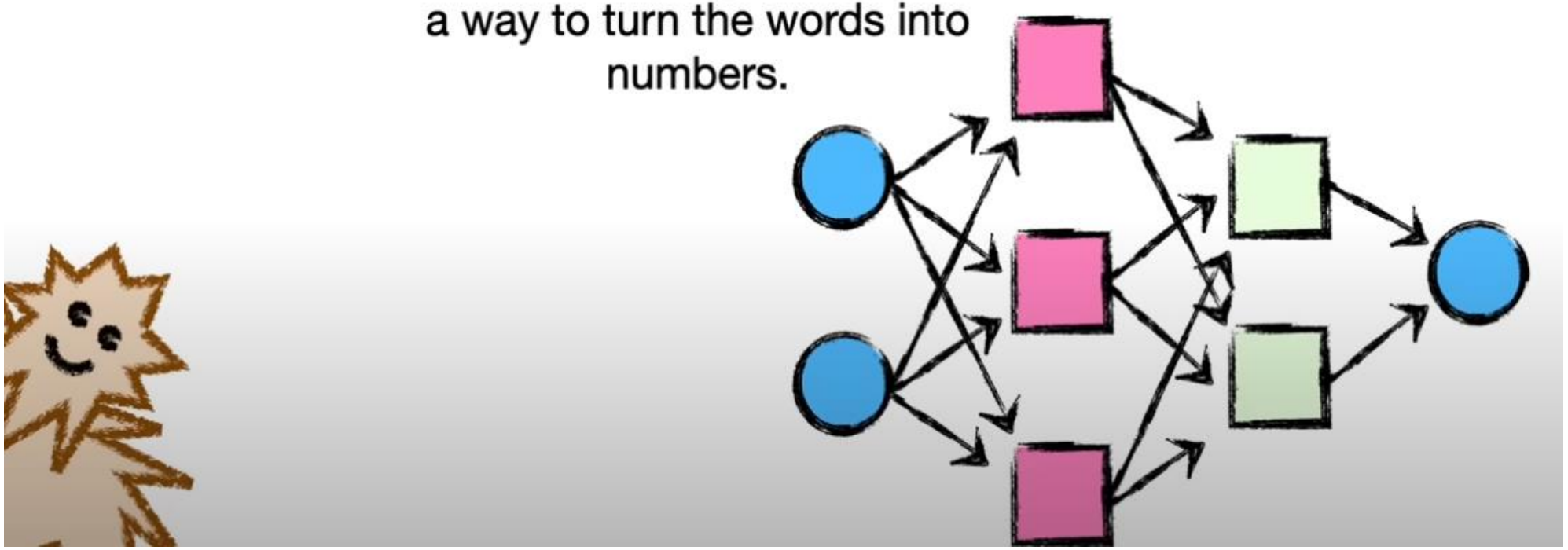
1. Collect Data
2. Text Preprocessing
   - Tokenization
   - Stemming
   - Normalization …
3. **Text Representation**
4. Machine Learning Model Selectin and Training
5. Evaluation of the Selected Model
6. Deploy the Model

Human vs Machine Intelligence

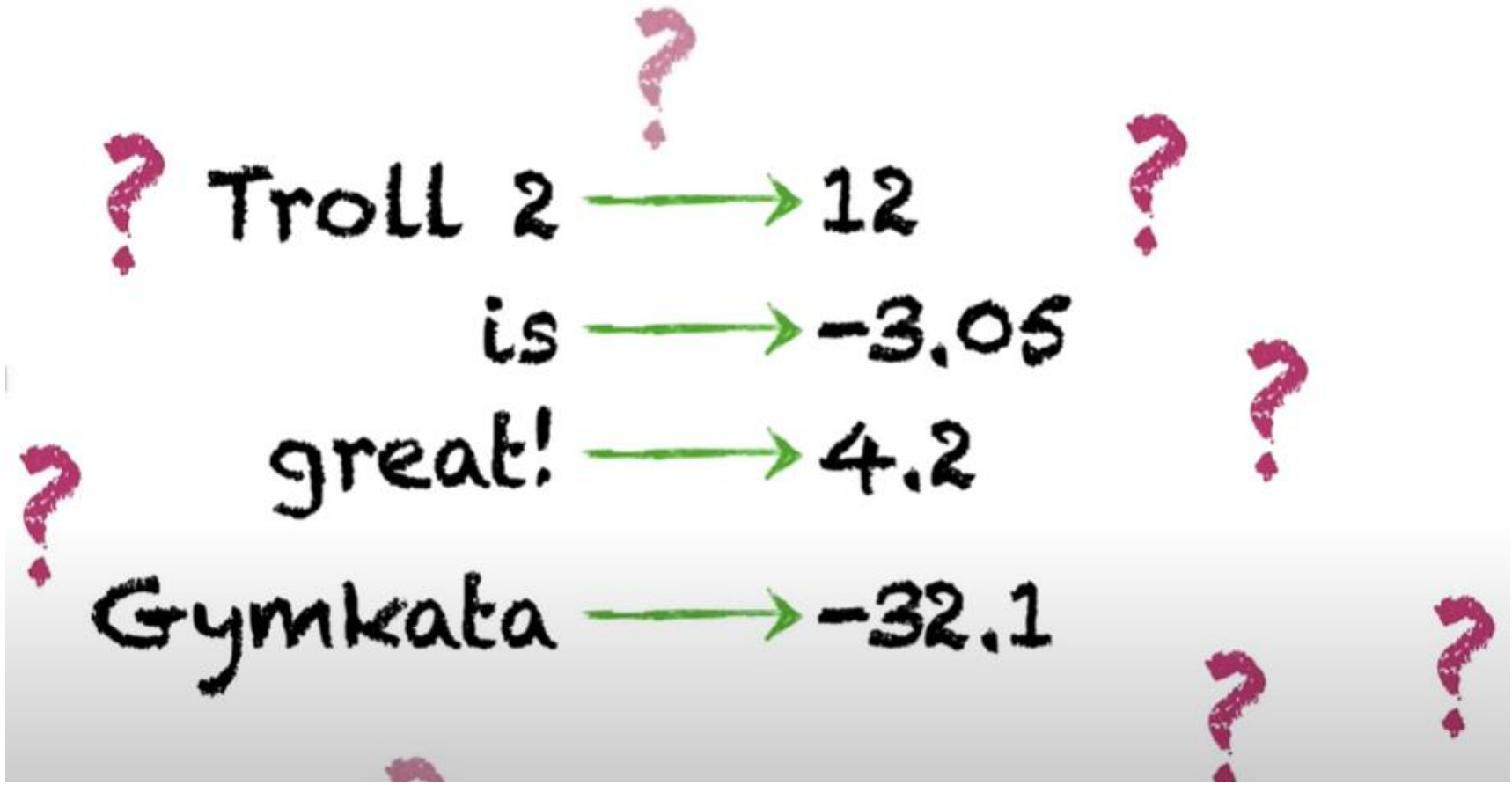


Human vs Machine Intelligence

So, if we want to plug words into a **Neural Network**, or some other machine learning algorithm, we need a way to turn the words into numbers.



Human vs Machine Intelligence



What are the different methods to represent the meaning of a word?

Lecture Plan

- Word Meaning
- Vector Space Models
- Bag-of-words (BOW) Methods
  - One-hot Encoding
  - Term-Document Frequency Matrix
  - Word-word matrix
  - Term Frequency-Inverse Document Frequency
  - Positive Pointwise Mutual Information (PMI)
- Cosine Similarity and Euclidian Distance
- Summary

Word Meaning

What do words mean?

- Classical NLP applications:
  - Words are just strings (or indices w_i in a vocabulary list)
  - That's not very satisfactory!

- Introductory logic classes:
  - The meaning of "dog" is DOG; cat is CAT
  - $\forall x\ DOG(x) \longrightarrow MAMMAL(x)$

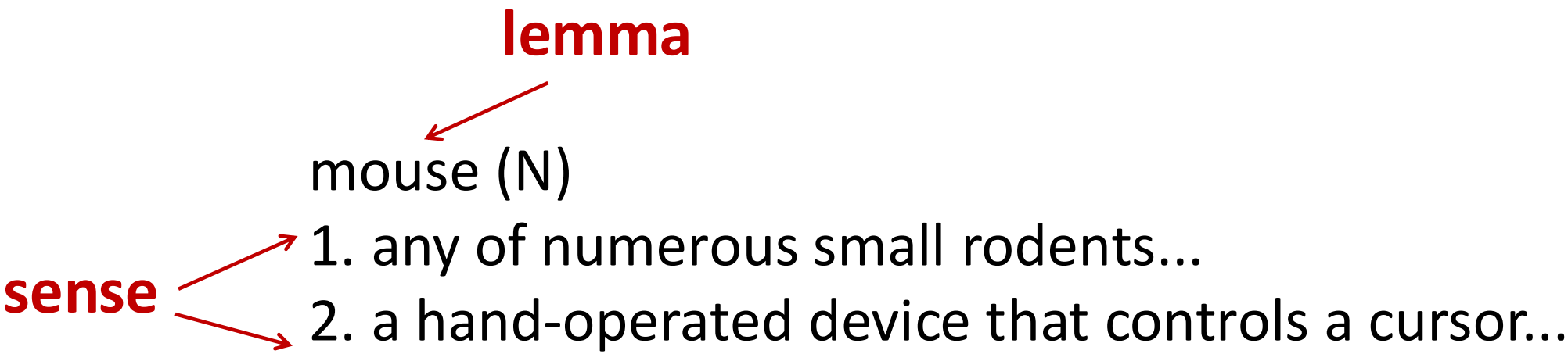
- Old linguistics joke by Barbara Partee in 1967:
  - Q: What's the meaning of life?
  - A: LIFE

That seems hardly better!

Desiderata

- What should a theory of word meaning do for us?
- Let's look at some desiderata
- From lexical semantics, the linguistic study of word meaning

Lexical Semantics: Lemmas and senses



Modified from the online thesaurus WordNet

A sense or "concept" is the meaning component of a word
Lemmas can be polysemous (have multiple senses)

Lexical Semantics

- How should we represent the meaning of the word?
  - Words, lemmas, senses, definitions
  - Relationships between words or senses

Relations between senses: Synonymy

- Synonyms have the same meaning in some or all contexts.
  - couch / sofa
  - big / large
  - automobile / car
  - vomit / throw up
  - water / H_20
  - filbert / hazelnut

Relations between senses: Synonymy

- Note that there are probably no examples of perfect synonymy.
  - Even if many aspects of meaning are identical
  - Still may not preserve the acceptability based on notions of politeness, register, genre, slang, etc.

- **Examples:**

water/H_20

"H_20" in a surfing guide?

big/large

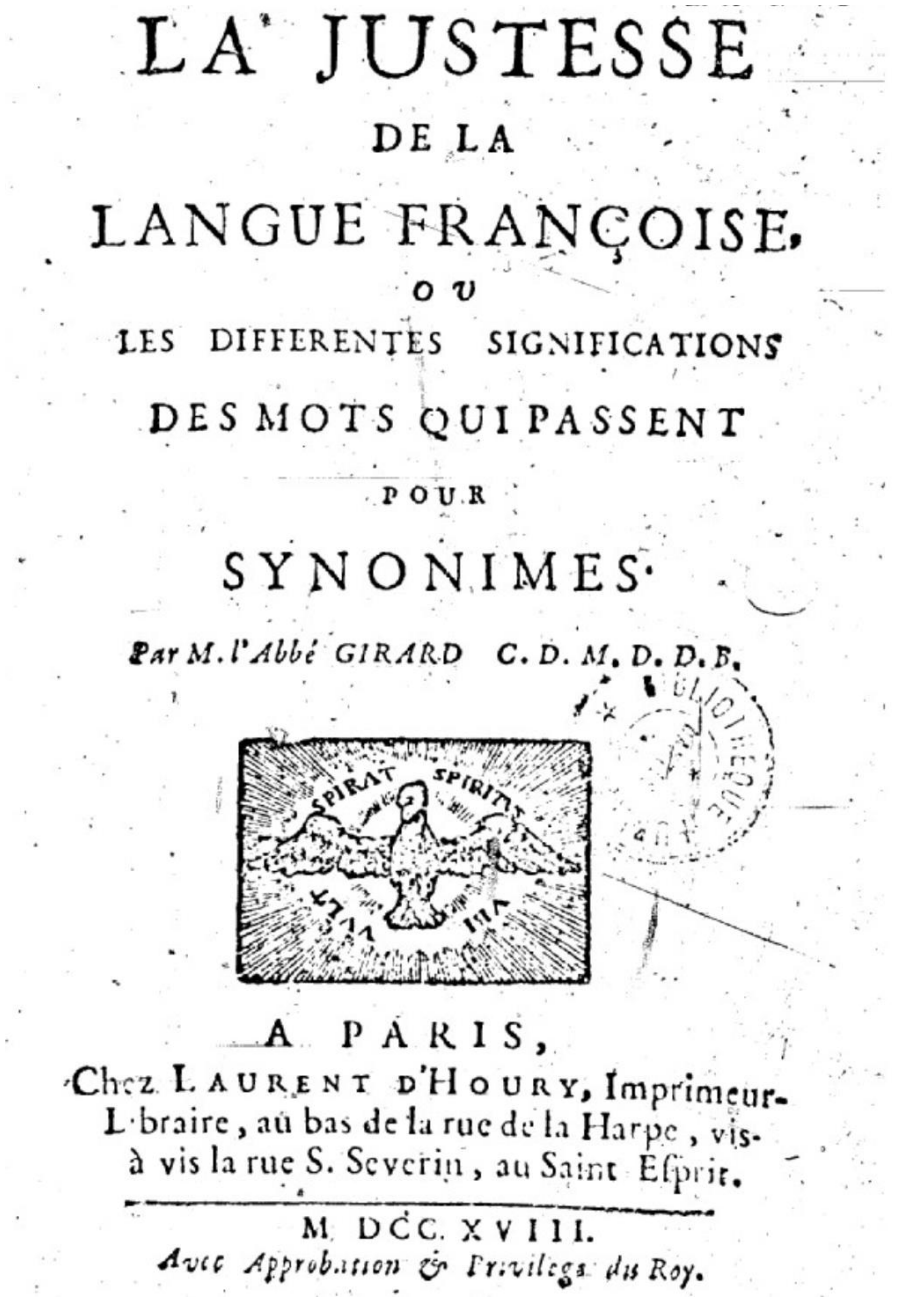
my big sister != my large sister

Abbé Gabriel Girard 1718

Re: "exact" synonyms

"je ne crois pas qu'il y ait de mot ſynonime dans aucune Langue."

[I do not believe that there is a synonymous word in any language]


LA JUSTESSE
DE LA
LANGUE FRANÇOISE,
OU
LES DIFFERENTES SIGNIFICATIONS
DES MOTS QUI PASSENT
POUR
SYNONIMES.
Par M. l'Abbé GIRARD C. D. M. D. D. B.
SPIRAT SPIRITUS
A PARIS,
Chez LAURENT D'HOURY, Imprimeur-Libraire, au bas de la rue de la Harpe, vis-à-vis la rue S. Severin, au Saint Eſprit.
M. DCC. XVIII.
Avec Approbation & Privilege du Roy.

Thanks to Mark Aronoff!

The Linguistic Principle of Contrast

- Difference in form → difference in meaning

- **Semantics:** Different words = different purposes.
- **NLP:** challenge of synonymy & polysemy → models must capture subtle distinctions

Relation: Similarity

- Words with similar meanings.
- Not synonyms, but sharing some element of meaning

  - car, bicycle
  - cow, horse

Ask humans how similar 2 words are

word1	word2	similarity
vanish	disappear	9.8
behave	obey	7.3
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

SimLex-999 dataset (Hill et al., 2015), https://fh295.github.io/simlex.html

Relation: Word relatedness

- Also called "word association"
- Words can be related in any way, perhaps via a **semantic field**

  - `coffee, tea:` **similar**
  - `coffee, cup:` **related**, not similar

A semantic field is a set of words which cover a particular semantic domain and bear structured relations with each other.

- **hospitals**
  `surgeon, scalpel, nurse, anaesthetic, hospital`
- **restaurants**
  `waiter, menu, plate, food, menu, chef`
- **houses**
  `door, roof, kitchen, family, bed`

Relation: Antonymy

- Senses that are opposites with respect to only one feature of meaning
- Otherwise, they are very similar!

  `dark/light  short/long  hot/cold  up/down`

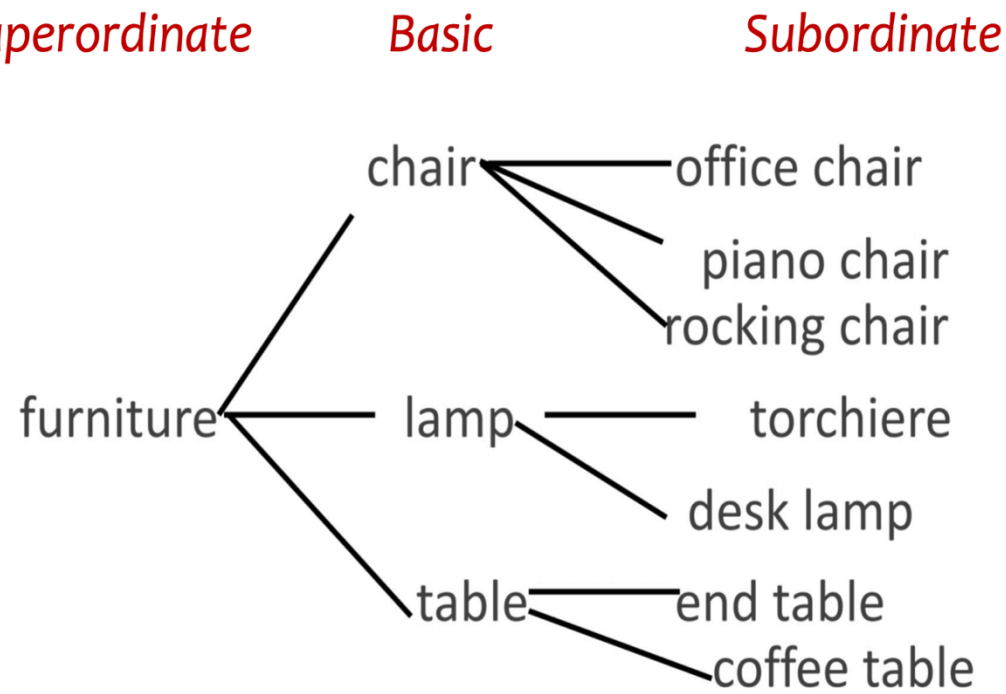
  `in/out`

- More formally: antonyms can
  - define a binary opposition or be at opposite ends of a scale
    - `long/short, fast/slow`
  - Be *reversives*:
    - `rise/fall, up/down`

Relation: Superordinate/Subordinate

- One sense is a subordinate of another if the first sense is more specific, denoting a subclass of the other
  - *Car is a subordinate of vehicle*
  - *Mango is a subordinate of fruit*

- Conversely superordinate
  - *Vehicle is a superordinate of car*
  - *Fruit is a superordinate of mango*



Connotation (sentiment)

- Words have **affective** meanings
  - Positive connotations (*happy*)
  - Negative connotations (*sad*)

- Connotations can be subtle:
  - Positive connotation: *copy, replica, reproduction*
  - Negative connotation: *fake, knockoff, forgery*

- Evaluation (sentiment!)
  - Positive evaluation (*great*, *love*)
  - Negative evaluation (*terrible*, *hate*)

Electronic Dictionaries: WordNet

```
from nltk.corpus import wordnet as wn
panda = wn.synset('panda.n.01')
hyper = lambda s: s.hypernyms()
list(panda.closure(hyper))
```

```
[Synset('procyonid.n.01'),
Synset('carnivore.n.01'),
Synset('placental.n.01'),
Synset('mammal.n.01'),
Synset('vertebrate.n.01'),
Synset('chordate.n.01'),
Synset('animal.n.01'),
Synset('organism.n.01'),
Synset('living_thing.n.01'),
Synset('whole.n.02'),
Synset('object.n.01'),
Synset('physical_entity.n.01'),
Synset('entity.n.01')]
```

(here, for *good*):

```
S: (adj) full, good
S: (adj) estimable, good, honorable, respectable
S: (adj) beneficial, good
S: (adj) good, just, upright
S: (adj) adept, expert, good, practiced,
proficient, skillful
S: (adj) dear, good, near
S: (adj) good, right, ripe
...
S: (adv) well, good
S: (adv) thoroughly, soundly, good
S: (n) good, goodness
S: (n) commodity, trade good, good
```

https://wordnet.princeton.edu/

So far

- **Concepts** or word senses
  - Have a complex many-to-many association with **words** (homonymy, multiple senses)

- Have relations with each other
  - Synonymy
  - Antonymy
  - Similarity
  - Relatedness
  - Connotation

Computational models of word meaning

- Can we build a theory of how to represent word meaning, that accounts for at least some of the desiderata?
  - Words, lemmas, senses, definitions
  - Relationships between words or senses
  - Taxonomy relationships
  - Word similarity, word relatedness

- We'll introduce **vector semantics**
  - The standard model in language processing!
  - Handles many of our goals!

Vector Space Models

Distributional Hypothesis

- “The meaning of a word is its use in the language” [Wittgenstein PI 43]
- “You shall know a word by the company it keeps” [Firth 1957]
- “If A and B have almost identical environments we say that they are synonyms” [Zellig Harris 1954]

Example: What does OngChoi Mean?

- Suppose you see these sentences:
  - Ong choi is delicious **sautéed with garlic**.
  - Ong choi is superb **over rice**
  - Ong choi **leaves** with salty sauces

- And you've also seen these:
  - …spinach **sautéed with garlic over rice**
  - Chard stems and **leaves** are **delicious**
  - Collard greens and other **salty** leafy greens

- ***Conclusion:***
  - Ongchoi is a leafy green like spinach, chard, or collard greens
    - We could conclude this based on words like "leaves" and "delicious" and "sauteed"

Ongchoi: *Ipomoea aquatica "Water Spinach"*

空心菜

kangkong

rau muống

…

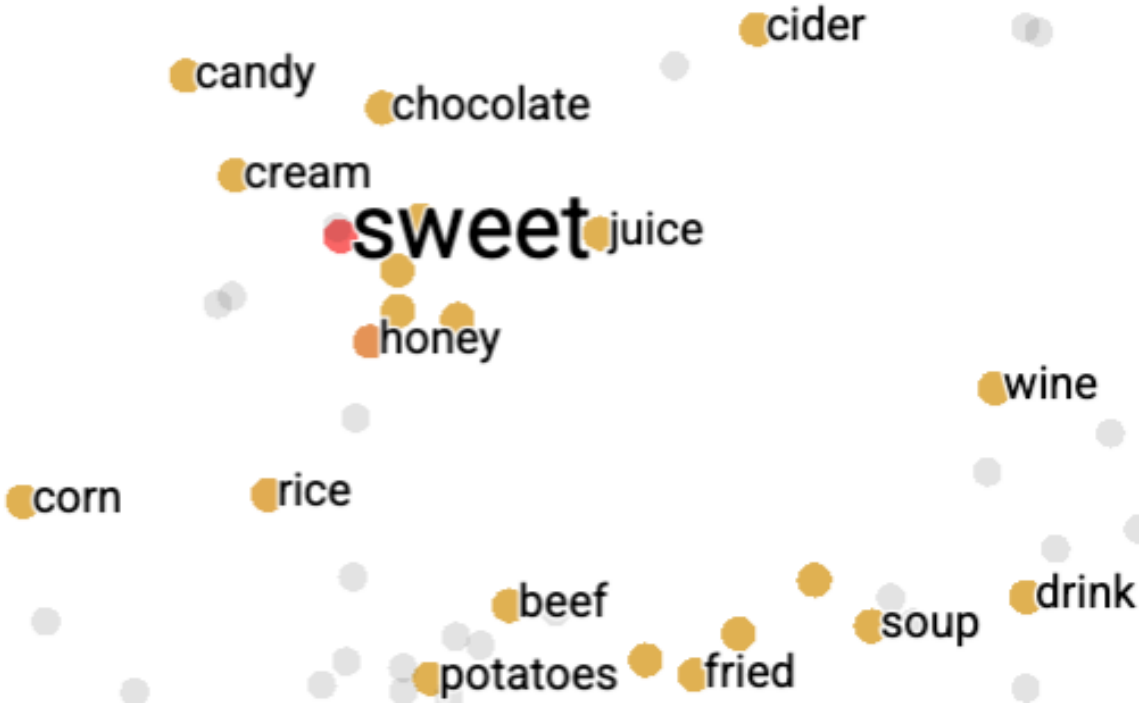
Yamaguchi, Wikimedia Commons, public domain

Idea: Defining meaning by linguistic distribution

- Let's define the meaning of a word by its distribution in language use, meaning its neighboring words or grammatical environments.

Defining meaning as a point in space based on distribution

- Each word = a vector (not just "good" or "w_{45}")
- Similar words are "**nearby in semantic space**"
- We build this space automatically by seeing which words are **nearby in text**



Vector space models

- Vector space models allows us to:
  - Represent words and documents as **vectors** that capture relative **meaning.**
  - The standard way to represent meaning in NLP
- Called an "embedding" because it's embedded into a space
  - **Every modern NLP algorithm uses embeddings as the representation of word meaning**

Why vectors?

- Consider sentiment analysis:
  - With **words**, a feature is a word identity
    - Feature: "The movie was **terrible”**
    - requires **exact same word** to be in training and test

  - With **embeddings**:
    - Feature is a word vector
    - The previous word was vector [35,22,17…]
    - Now in the test set we might see a similar vector [34,21,14]
    - We can generalize to **similar but unseen** words!!!

Text Representation Types

1. Bag-of-words Models
2. Neural Word Embeddings Models (Wor2vec, GloVe)
3. Contextual Embeddings \ Sentence Embeddings Models
4. Topic Modeling Methods
5. …

Bag-of-words (BOW) Representation

- A bag-of-words model, shortened as BOW, is a representation method that counts the number of occurrences, or frequency, of each word in the given corpus of documents.

- BOW involves a vocabulary of known words and a measure of the presence of these words.

- The complexity of BOW representation resides largely in deciding how to create the **vocabulary** of known words and how to score the presence of these words.

Bag-of-words (BOW) Representation

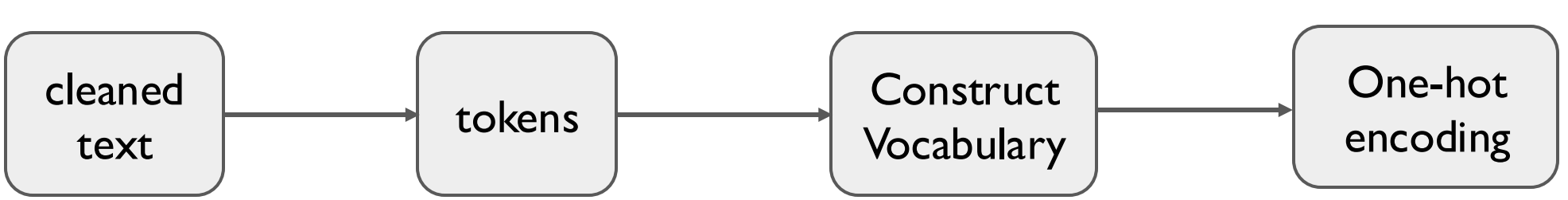
- Usually, the **vocabulary** is built using the words contained in the given corpus after removing the **stop-word list**.

- Several BOW methods exist, including:
  - One-hot encoding
  - Term Count
  - Term Frequency-Inverse Document Frequency (TF-IDF)
  - Positive Pointwise Mutual Information

One-hot Encoding

One-hot encoding

- One-hot encoding represents words as vectors in a **“Vocabulary”.**
- Each word has a one-hot vector of size **V** (total vocabulary size).
  - **Example**: For a vocabulary of 5 words, each vector has size 5 with values “0” or “1” showing **absence** or **presence**.
  - Each word has a **unique** one-hot vector; no two words share the same representation.



One-hot encoding: Example

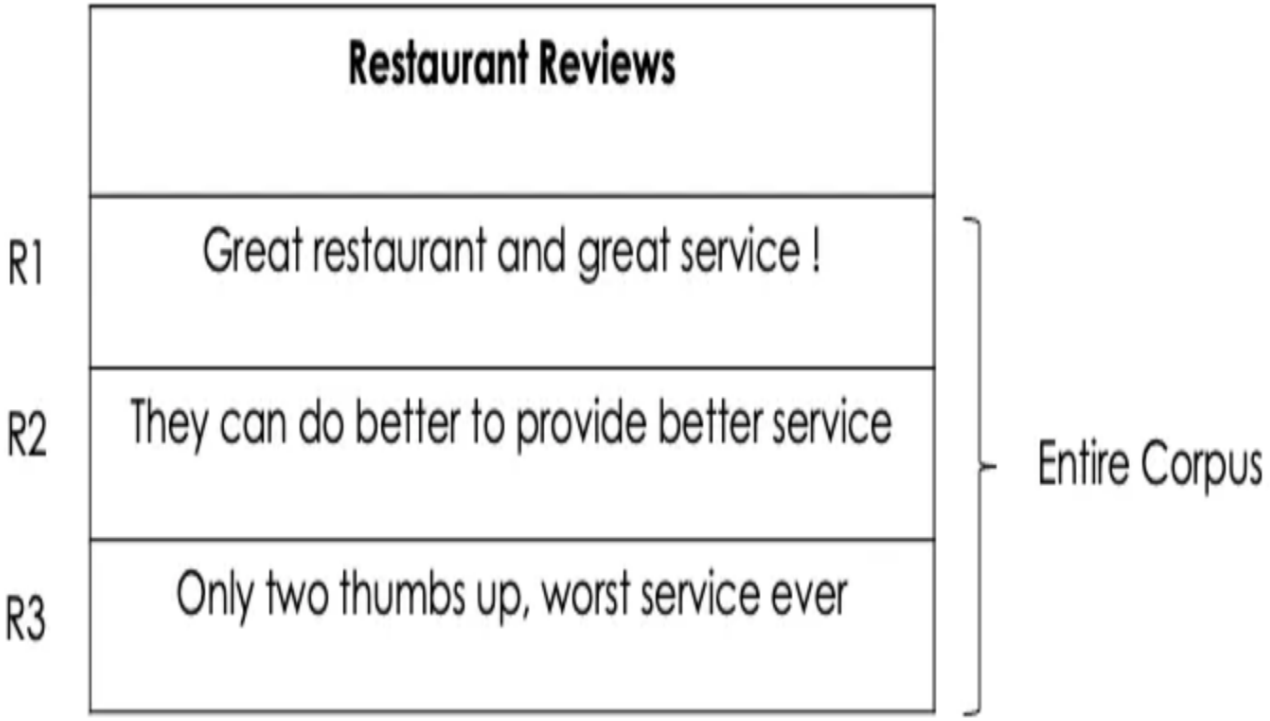


Image by author(Reviews about a restaurant (R1: first review))

- **Step 1**: Create **a set of all the words in the corpus (Create the Vocabulary)**

- **Step 2**: **Determine the presence or absence of a given word** in a particular review.

    - The presence is represented by 1 and the absence represented by 0. Each review will be then represented as a **tuple of 0, 1 elements**.

One-hot encoding: Example

VOCBULARY

Set of all the words in the corpus	R1: Great Restaurant and great service !	R2: They can do better to provide better service	R3: Only two thumbs up, worst service ever
great	1	0	0
restaurant	1	0	0
and	1	0	0
service	1	1	1
they	0	1	0
can	0	1	0
do	0	1	0
better	0	1	0
to	0	1	0
provide	0	1	0
only	0	0	1
Two	0	0	1
thumbs	0	0	1
up	0	0	1
worst	0	0	1
ever	0	0	1

One-hot Encoding: Implementation

Let's implement One Hot Encoding for the first 10 tokens in the vocabulary:

```
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
integer_label_encoded = label_encoder.fit_transform(tokens[1:10])
label_encoded = integer_label_encoded.reshape(len(integer_label_encoded), 1)
onehot_encoder = OneHotEncoder(sparse=False)
onehot_encoded = onehot_encoder.fit_transform(label_encoded)
print(onehot_encoded)
```

```
[[0. 0. 0. 0. 0. 0. 0. 0. 1.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

One Hot Encoded Vectors

Term-Document matrix

Term-Document Frequency matrix

- Term frequency is the measurement of how frequently a term (word) occurs within a document.
- The easiest calculation is simply counting the number of times a word appears.
- **Example:** lets consider a collection of documents (i.e works of Shakespeare)

		d_1	d_2	d_3	d_4
		As You Like It	Twelfth Night	Julius Caesar	Henry V
w_1	battle	1	0	7	13
w_2	good	114	80	62	89
w_3	fool	36	58	1	4
w_4	wit	20	15	2	3

Term-Document Frequency matrix

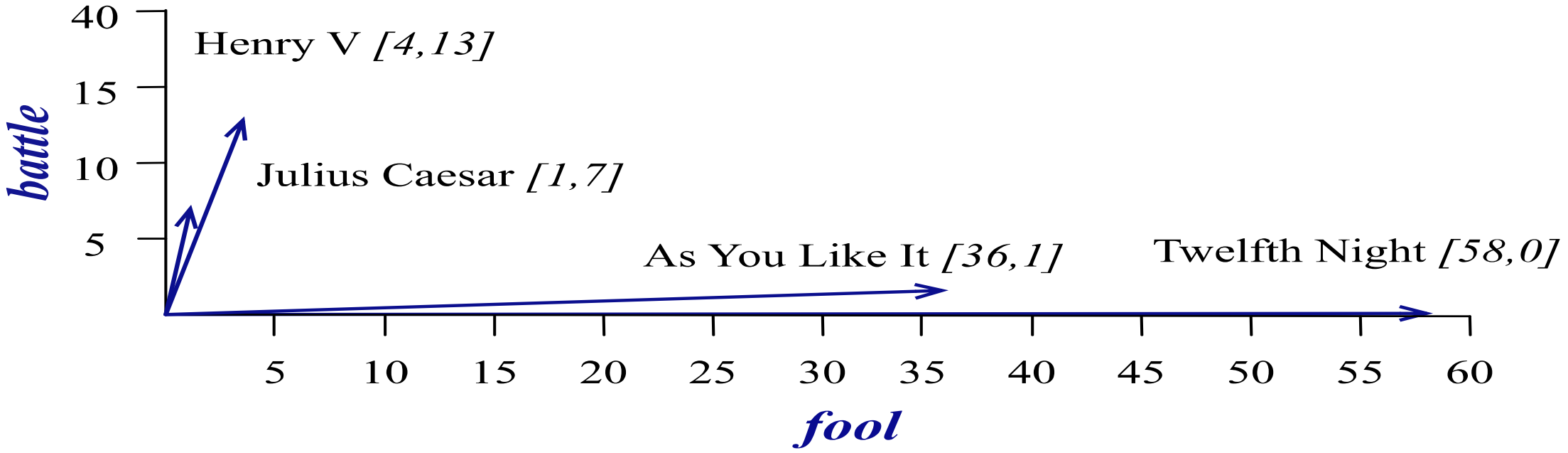
- **Each document d_i is represented by a vector of words.**

		d_1	d_2	d_3	d_4
		As You Like It	Twelfth Night	Julius Caesar	Henry V
w_1	battle	1	0	7	13
w_2	good	114	80	62	89
w_3	fool	36	58	1	4
w_4	wit	20	15	2	3

- Vectors are similar for the two comedies (As you like it, Twelfth Night)
- Different than the history (Julius Caesar, Henry V)
- Comedies pieces have *more fools\wit* and *fewer battles*

Visualizing Document Vectors

- A spatial visualization of the document vectors for the four Shakespeare play documents using just two of the dimensions (Fool and Battle).



- The comedies have high values for the *fool* dimension and low values for the *battle* dimension.

Cosine Similarity

- ***Cosine similarity*** is a measure of similarity between two non-zero vectors that calculates the cosine of the angle between them.

- Its value ranges from -1 (meaning exactly opposite) to 1 (exactly the same), with 0 typically indicating independence and in-between values indicating intermediate similarity or dissimilarity.

- The cosine similarity between "***Henry V***" and "***Julius Caesar***" can be calculated as:

$$\text{Cosine Similarity} = \frac{\vec{A} \cdot \vec{B}}{|\vec{A}||\vec{B}|} = \frac{(4 \times 1) + (13 \times 7)}{\sqrt{4^2 + 13^2} \times \sqrt{1^2 + 7^2}} = \frac{4 + 91}{\sqrt{16 + 169} \times \sqrt{1 + 49}} = \frac{95}{\sqrt{185} \times \sqrt{50}} \approx \frac{95}{96.18} \approx 0.99$$

- This high cosine similarity indicates that "***Henry V***" and "***Julius Caesar***" have a similar distribution of the terms "**fool**" and "**battle**".

Euclidean Distance

- **Euclidean distance** measures the straight-line distance between two points in space.
- In the previous graph, each play is represented as a point based on the number of occurrences of the words "***fool***" (x-axis) and "***battle***" (y-axis).
- For example, "***Henry V***" is positioned at (4,13), and "***Twelfth Night***" at (58,0). The Euclidean distance between them can be calculated using the formula:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- If we calculate it between "***Henry V***" and "***Twelfth Night***," the distance would be:

$$d = \sqrt{(58 - 4)^2 + (0 - 13)^2} = \sqrt{54^2 + (-13)^2} = \sqrt{2916 + 169} = \sqrt{3085} \approx 55.55$$

- This large Euclidean distance indicates that the plays are dissimilar in terms of the frequency of the words "**fool**" and "**battle**.

Words can be vectors too!

- Vector semantics can also be used to represent the meaning of words.

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- ***battle*** is "the kind of word that occurs in Julius Caesar and Henry V"
- ***fool*** is "the kind of word that occurs in comedies, especially Twelfth Night"

Word-word matrix

Word-word matrix or "term-context matrix"

- Two **words** are similar in meaning if their context vectors are similar

context words:
4 words to the left
+ 4 words to the right

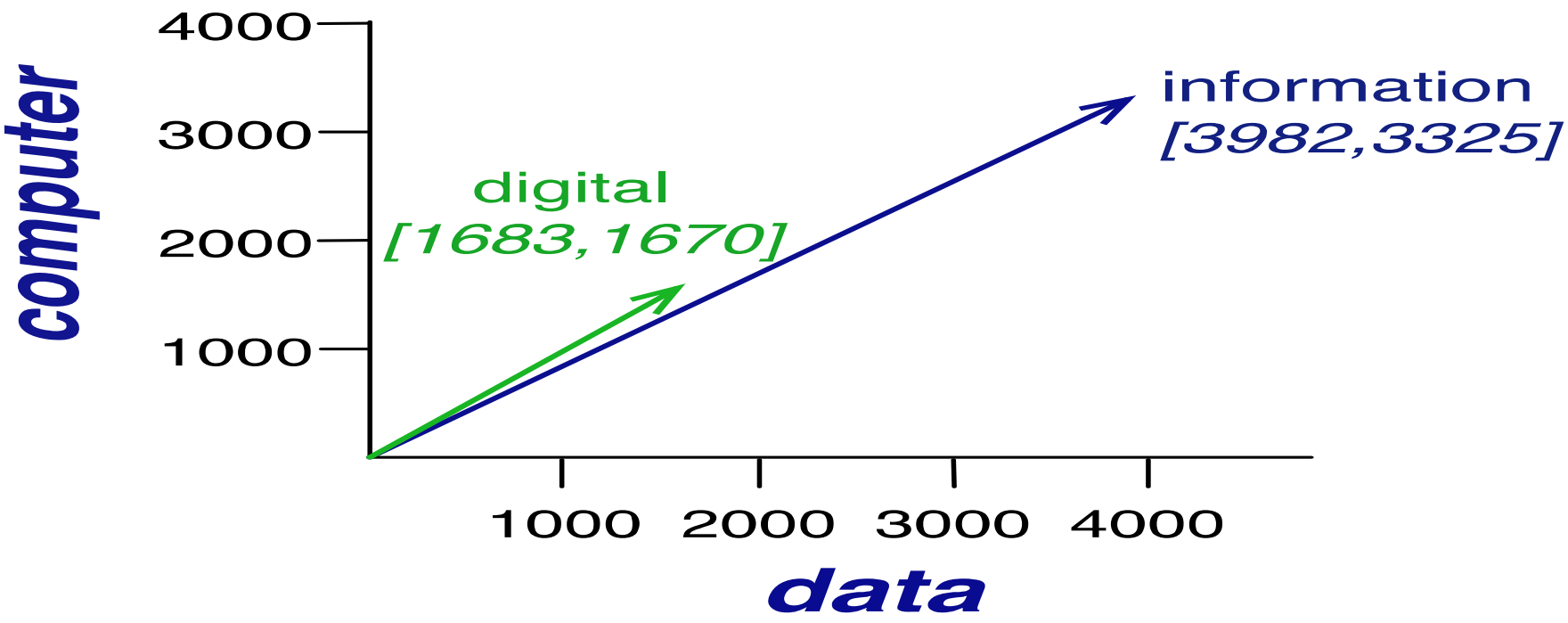
is traditionally followed by **cherry** pie, a traditional dessert
often mixed, such as **strawberry** rhubarb pie. Apple pie
computer peripherals and personal **digital** assistants. These devices usually
a computer. This includes **information** available on the internet

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

The matrix is of dimensionality $|V| \times |V|$

Visualizing word vectors

- A spatial visualization of word vectors for *digital* and *information* *using* just two of the dimensions, corresponding to the words *data* and *computer*.



Why raw frequency is not a good representation?

- The word **"good"** is distributed across many documents, so it is not discriminative between plays.

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- **Question: What to do with words that are distributed across many documents?**

Two common solutions for word weighting
weighted function instead of raw counts!

1. **Term Frequency- Inverse Document Frequency (tf-idf)**
   - **Usually applied when the dimensions are documents**

2. **Pointwise Mutual Information (PMI)**
   - **Usually applied when the dimensions are words**

Term Frequency-Inverse Document Frequency

Term Frequency – Inverse Document Frequency (TF-IDF) (Sparck Jones – 1970s)

- **TF-IDF** model measures the importance of a word in a document by determining its frequency in the document, and its inverse document frequency.

- Given a corpus of N documents, the *TF-IDF* of a word w in a document d, is defined as follows:

$$TF - IDF(w^d) = x_w^d . \log(\frac{N}{N_w}) \quad \textbf{\textit{IDF}}$$

- Where N is the number of documents in the corpus, x_w^d is the number of occurrences (or frequency) of the word w in the document d, and N_w the number of documents that contain the word w.

Term Frequency – Inverse Document Frequency (TF-IDF)

- **The inverse document frequency (idf)** is used to give a higher weight to words that occur only in a few documents
  - Terms that are limited to a few documents are useful for discriminating those documents from the rest of the collection;
  - Terms that occur frequently across the entire collection aren't as helpful.
- A high ***TF-IDF*** value means that the word $\boldsymbol{w}$ important in the document $\boldsymbol{d}$ and appears little in the other documents of the collection.
- ***TF-IDF*** yields a value that shows how important a given word is by not only looking at term frequency, but also analyzing how many times the word appears across all the documents of the corpus.

What is a document?

- Could be a play, Wikipedia article, A sentiment, A movie review, An email…
- But very frequently you might need to break up the corpus into documents yourself for the purposes of computing idf.
- Documents can be **anything**; they don't have to be original documents.

Term Frequency – Inverse Document Frequency (TF-IDF)

- **Raw Counts**

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- **TF-IDF**

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	0.074	0	0.22	0.28
good	0	0	0	0
fool	0.019	0.021	0.0036	0.0083
wit	0.049	0.044	0.018	0.022

Term Frequency-Inverse Document Frequency (tf-idf)

- **TF-IDF Implementation using Sklearn Library**

```
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
def tokenize(text):
    tokens = nltk.word_tokenize(text)
    stems = []
    for item in tokens:
        stems.append(PorterStemmer().stem(item))
    return stems
tfidf = TfidfVectorizer(tokenizer=tokenize, stop_words='english')
tfs = tfidf.fit_transform(tweets["Text"])
# to visualize the formed TF-IDF matrix
tfs.toarray()
```

Pointwise Mutual Information

Pointwise Mutual Information (PMI)
(Kenneth Church - 1990)

- **Pointwise Mutual Information (PMI)** is used for *term-context-matrices*, when the vector dimensions correspond to words rather than documents

  - Does the word w co-occur with the context c more frequently than we would expect if they were independent?

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Pointwise Mutual Information (PMI)
(Kenneth Church - 1990)

- **Pointwise Mutual Information (PMI)** is used for *term-context-matrices,* when the vector dimensions correspond to words rather than documents

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

  - *Does a word $\boldsymbol{w}$ and a context $\boldsymbol{c}$ appear together more often than we'd expect just by chance?*
    - If they co-occur **more than chance** → PMI is **high** (strong association).
    - If they co-occur **less than chance** → PMI is **low or negative** (weak or unlikely association).
- **Example:**
  - Word w= "dog", context c= "bark" → high PMI.
  - Word w= "dog", context c= "astronomy" → low PMI.

Pointwise Mutual Information (PMI)

- The pointwise mutual information between a target word ***w*** and a context word **c** is defined as:

$$\text{PMI}(w,c) = \log_2 \frac{P(w,c)}{P(w)P(c)}$$

- **PMI is a useful tool whenever we need to find words that are strongly associated.**

- ***P*(*w*):** The probability of word ***w*** is typically calculated as the number of times ***w*** appears in the corpus divided by the total number of words in the corpus.
- ***P*(*c*):** Similarly, the probability of context ***c*** is calculated as the number of times the context ***c*** appears divided by the total number of contexts.
- ***P*(*w*,*c*):** The joint probability is the number of times word ***w*** appears in context ***c*** divided by the total number of word-context pairs

Positive Pointwise Mutual Information (PPMI)

- PMI ranges from $-\infty$ to $+\infty$
- But the negative values are problematic
  - Things are co-occurring **less than** we expect by chance

- So we just replace negative PMI values by 0
- **Positive PMI (PPMI)** between word w and context c:

$$\text{PPMI}(w,c) = \max(\log_2 \frac{P(w,c)}{P(w)P(c)}, 0)$$

Pointwise Mutual Information

Church, Kenneth**. "From PMI to Bots."** *International Journal of Lexicography* (2025): ecaf007.

Kenneth Church Northeastern University, USA,

Sparse versus **Dense** vectors

- Bag-of-words (Sparse Vectors) models are highly intuitive and straightforward to compute.

- However, Bag-of-words features suffer from several weaknesses:
  - they lose the ordering of the words and they also ignore semantics of the words.

- Moreover, one-hot encoding, tf-idf, or PMI vectors are:
  - **long** (length |V|= 20,000 to 50,000)
  - **sparse** (most elements are zero)

- Alternative: learn vectors which are
  - **short** (length 50-1000)
  - **dense** (most elements are non-zero)

Why Dense vectors?

- Short vectors may be easier to use as **features** in machine learning (fewer weights to tune)
- Dense vectors may **generalize** better than explicit counts
- Dense vectors may do better at capturing synonymy:
  - *car* and *automobile* are synonyms; but are distinct dimensions
    - a word with *car* as a neighbor and a word with *automobile* as a neighbor should be similar, but aren't
- **In practice, they work better**

Exercise: TF-IDF and PPMI Comparison in Word Importance

- You are given a small corpus of 5 documents with the following text:
  1. *The cat sat on the mat.*
  2. *The dog sat on the mat.*
  3. *The dog chased the cat.*
  4. *The cat and the dog are friends.*
  5. *The mat is where the cat sleeps.*

- **Part 1: TF-IDF Calculation**
  1. **Build the vocabulary of this corpus V**.
  2. **Compute the TF-IDF score** for the words "cat" and "mat" in each document.
  3. **Interpret the TF-IDF values:** Which term appears more frequently in specific documents, and which is more spread out across the corpus?

Exercise: TF-IDF and PPMI Comparison in Word Importance

- **Part 2: PPMI Calculation**

1. Consider the words "cat" and "mat" as well as their context word "sat." Create a **term-context co-occurrence matrix** based on word counts for the following pairs: ("cat", "sat") and ("mat", "sat").

2. **Calculate the joint probabilities** *P(cat, sat)* and *P(mat, sat*) using co-occurrence frequencies in the corpus.

3. **Calculate the individual probabilities** *P(cat), P(mat)*, and *P(sat)* based on their frequency in the corpus.

4. **Compute the Positive PMI** for the word pairs ("cat", "sat") and ("mat", "sat").

5. **Interpret the PPMI values:** Which word co-occurs more strongly with "sat"? What does this imply about the contextual relationship between these words?

Reading

- Please refer to this chapter: Chapter 5
- http://d2l.ai/chapter_natural-language-processing-pretraining/index.html

Next Class

- **Lecture 4: Language Modeling**