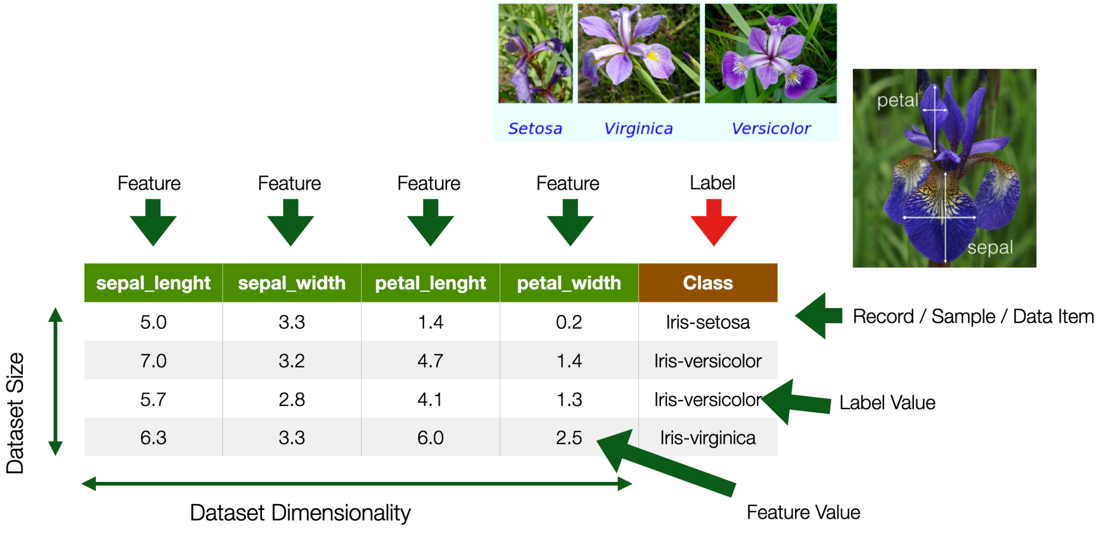
Machine Learning for Design

Lecture 6

Natural Language Processing - Part 2

Previously on ML4D

- **Machine Learning**: Observe *pattern of features* and attempt to imitate it in some way



sepal_lenght	sepal_width	petal_lenght	petal_width	Class
5.0	3.3	1.4	0.2	Iris-setosa
7.0	3.2	4.7	1.4	Iris-versicolor
5.7	2.8	4.1	1.3	Iris-versicolor
6.3	3.3	6.0	2.5	Iris-virginica

- A feature is an individual measurable property or characteristic of a phenomenon
- Choosing informative, discriminating, and independent features is essential for a well-working ML
- Features
  - Images → pixel values (e.g. B/W, RGB)
  - Numbers → OK
- What about **text?**

Textual Documents

- A sequence of alphanumerical characters
  - Short: e.g. tweets
  - Long: e.g Web documents, interview transcripts
- Features are (set of) words
  - Words are also syntactically and semantically organised
- *Feature values* are (sets of) words occurrences
- *Dimensionality* $\longrightarrow$ at least dictionary size

★★★★☆ **I wear this mask to sing lullabies to my children ...**, 24 May 2015

By **Sir Chubs**

Verified Purchase (What is this?)

This review is from: Overhead Rubber Penguin Mask Happy Feet Animal Fancy Dress (Toy)

I wear this mask to sing lullabies to my children. They are terrified of the mask. Whenever they protest about their bed time, or ask for too many sweets, I whip on the mask, and they soon know who is the King Penguin.

Document

I	Wear	Mask	...	W(n)	Class
1	1	1		0	Spam
0	0	1		0	Not Spam
					Spam

Main types of NLP Tasks

- **Label** (*classify*) a region of text
  - e.g. part-of-speech tagging, sentiment classification, or named-entity recognition
- **Link** two or more regions of text
  - e.g. *coreference*
    - are two mentions of a real-world thing (e.g. a person, place) in fact referencing the same real-world thing?
- **Fill in** missing information (missing words) based on context

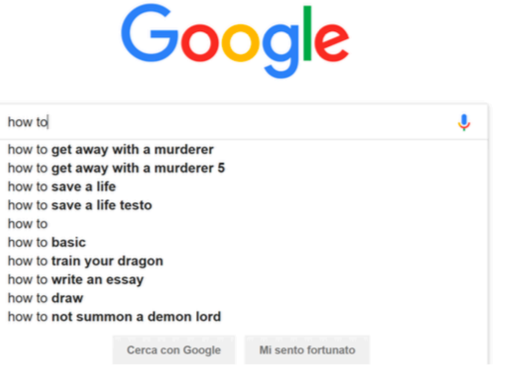
Language Representation

- Language = vocabulary and its usage in a specific context captured by textual data

What is a language model?

- A collection of statistics learned over a particular language
- Almost always empirically derived from a text corpora

What are language models used for?



- **Measure** how *important* (or descriptive) a word is in a given document collection

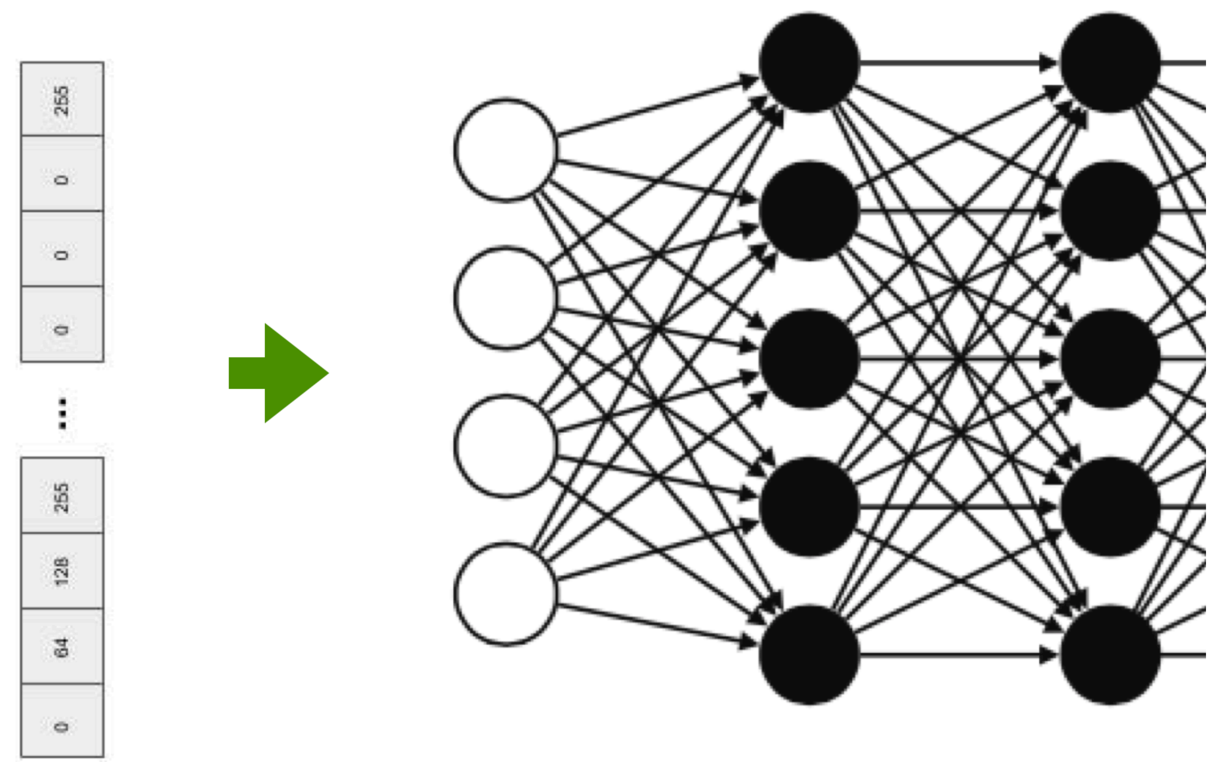
  e.g., find the set of words that best describe multiple clusters (see Assignment 2)

- **Predict** how *likely* a sequence of words is to occur in a given context

  e.g., find the words that are more likely to occur next

What is the issue with word representation?

- Words are **discrete symbols**
  - Machine-learning algorithms cannot process symbolic information as it is
  - We need to transform the text into **numbers**
- But we also need a way to express **relationships** between words!



A simple approach

- Assign an incremental number to each word
  - $cat = 1$
  - $dog = 2$
  - $pizza = 3$

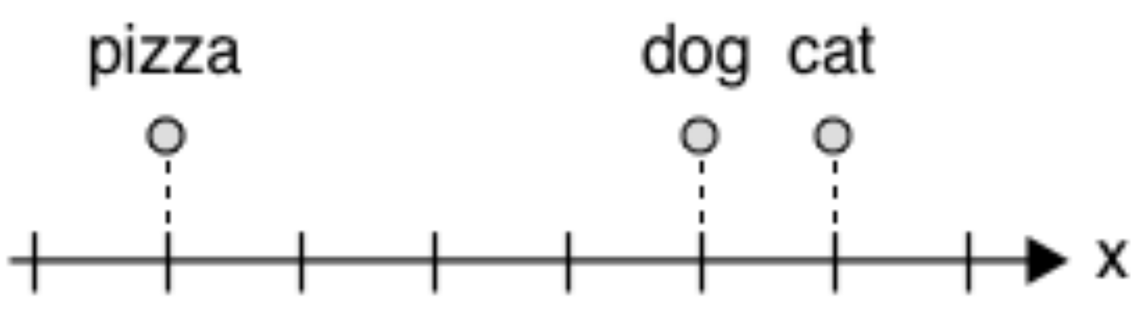
- **Problem**: there is no notion of similarity
  - Is a *cat* as semantically close (similar) to a *dog* as a *dog* is to a *pizza*
- Also, no arithmetic operations
  - Does it make sense to calculate $dog - cat$ to establish similarity?

Word Embeddings

- **Embed** (*represent*) words in a numerical n-dimensional space
- Essential for using machine learning approaches to solve NLP tasks
  - They bridge the symbolic (discrete) world of words with the numerical (continuous) world of machine learning models

Approach 1

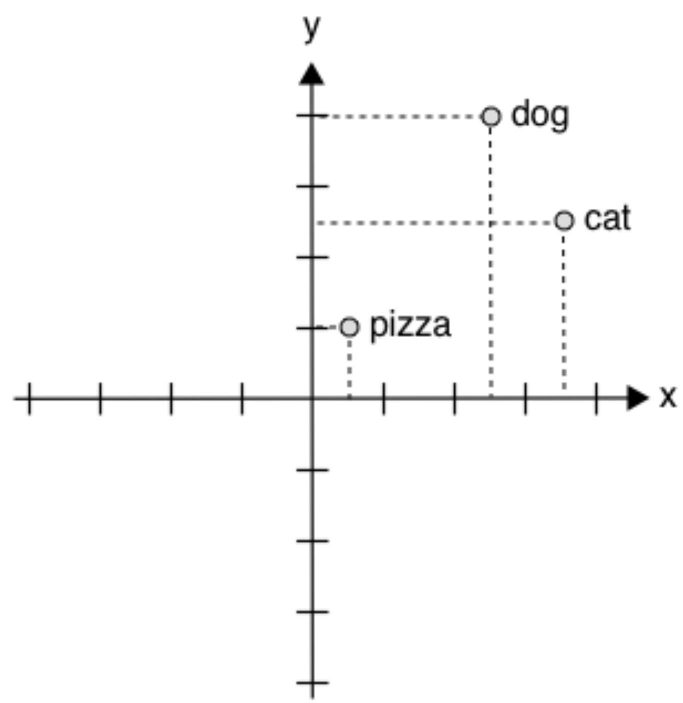
Assign numbers to words, and put semantically related words close to each other



- We can now express that dog is more related to cat than to $pizza$
- But is $pizza$ more related to dog than to cat?

Approach 2

Assign multiple numbers (a vector) to words



$$cat = [4, 2]$$

$$dog = [3, 3]$$

$$pizza = [1, 1]$$

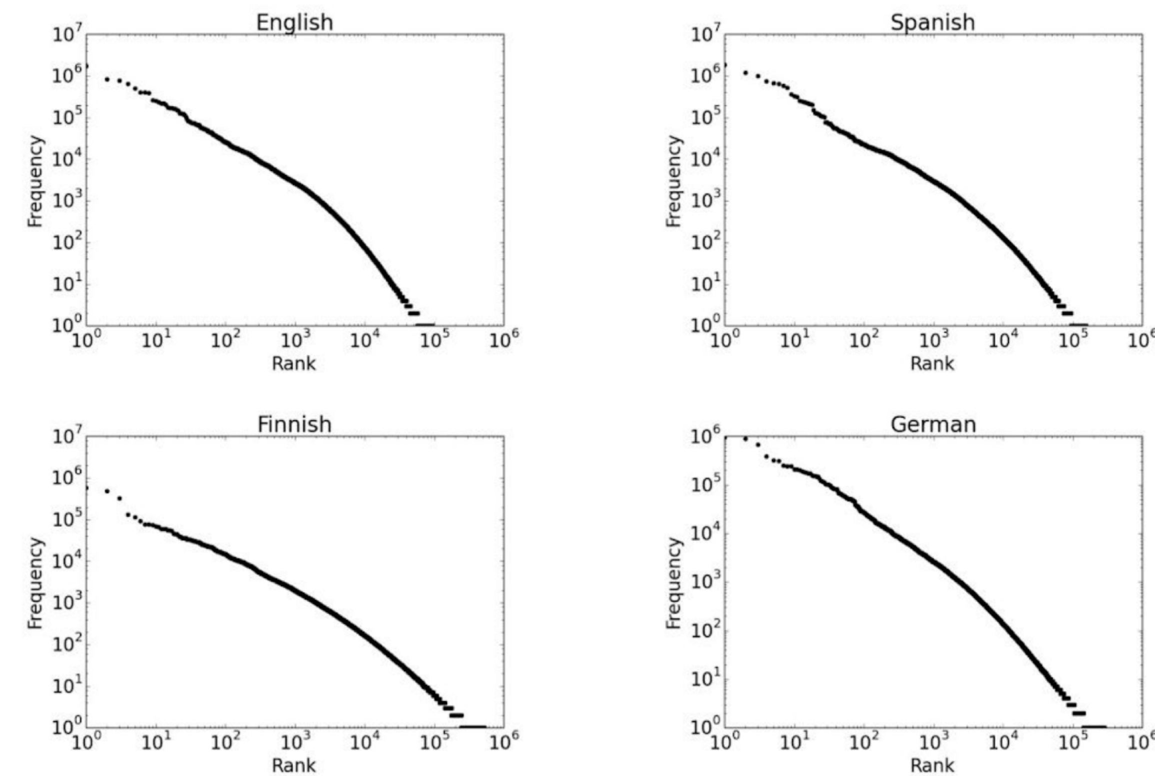
- We can calculate *distance* (and *similarity*)
  - e.g. Euclidean, or Cosine (angles)
- But **what is the meaning of an axis**?

One-Hot Encoding

- Each word in the vocabulary is represented by a one-bit position in a HUGE (**sparse**) vector
  - Vector dimension = size of the dictionary
  - There are an estimated 13 million tokens for the English language

$$
\begin{aligned}
cat &= [0,0,0,0,0,0,0,0,0,0,1,0,0,0,\ldots,0] \\
dog &= [0,0,0,0,0,0,0,1,0,0,0,0,0,0,\ldots,0] \\
pizza &= [1,0,0,0,0,0,0,0,0,0,0,0,0,0,\ldots,0]
\end{aligned}
$$

- Problems with one-hot encoding:
  - The size of the vector can be **huge**
    - Do you Remember Zip's law?
    - Easy to reach 10^6 words
    - But we can use *stemming*, *lemmatisation*, etc
- Still, no notion of similarity or words relationship
  - Each word is an independent, discrete entity



Independent and identically distributed **words assumption**

- The simplest language models assume that each word in a text **appears independently** of the others
  - The text is modeled as generated by a sequence of independent events
- The **probability** of a *word* can be estimated as the *number of times* a word appears in a text corpus
- But high probability **does not mean important** (or descriptive)

Back to the *term-document* matrix

Document

I	Wear	Mask	...	W(n)	Class
1	1	1		0	Spam
0	0	1		0	Not Spam
					Spam

– *How to measure* the importance of words?

Term frequency tf

- Raw frequency

$$\mathrm{tf}(t,d) = f_{t,d}$$

- Log normalisation

$$\mathrm{tf}(t,d) = \log(1 + f_{t,d})$$

- Normalised Frequency

$$\mathrm{tf}(t,d) = 0.5 + \frac{0.5 f_{t,d}}{f_{\max}(d)}$$

- Measuring the importance of a word t to a *document* d
- The more frequent the word, the more important it is to describe the document

Inverse document frequency IDF

$$\mathrm{IDF}(t, D) = \log \frac{N}{|d \in D : t \in d|}$$

- Measuring the importance of a word t to a *document collection* D
- Rare terms are more important than common terms

$$TF - IDF$$

$$\mathrm{tfIDF}(t, d, D) = tf_{t,d} \times IDF_{t,D}$$

- *Scaling* a word's importance (in a document) based on both its frequency and its importance in the collection

N-gram language models

N-gram language models

- Calculate the conditional probabilities among *adjacent* words
- Given the word w, what is the probability of the next word $w+1$
  - e.g., given eat, $eat\ on$ vs. $eat\ British$
- bi-grams $\rightarrow$ 2 words, 3-grams $\rightarrow$ 3 words

$p(w|\text{eat})$

eat on	0.16	eat Thai	0.03
eat some	0.06	eat breakfast	0.03
eat lunch	0.06	eat in	0.02
eat dinner	0.05	eat Chinese	0.02
eat at	0.04	eat Mexican	0.02
eat a	0.04	eat tomorrow	0.01
eat indian	0.04	eat dessert	0.007
eat today	0.03	eat British	0.001

N-gram language models

- More accurate
  - The probabilities depend on the considered **context**
- The model accuracy increases with N
  - The syntactic/semantic contexts are better modeled
- *Grammatical rules*
  - e.g., an adjective is likely to be followed by a noun
- *Semantic restrictions*
  - e.g., *Eat a pear* vs. *Eat a crowbar*
- Cultural restrictions
  - e.g., *Eat a cat*

Limits of N-grams-based Language Model

- Conditional probabilities are difficult to estimate
  - For dictionary contains D terms there are D^N N-grams (30K words, 900M bi-grams)
    - the corpus should be billions of documents big for a good estimation
- They **do not generalize to unseen words sequences**

Representing words by their contexts

- **Distributional semantics**: *A word's meaning is given by the words that frequently appear close-by*
- When a word w appears in a text, its **context** is the set of words that appear nearby (within a *fixed-size* window)
- **The contexts in which a word appears tell us much about its meaning**

"You shall know a word by the company it keeps" - The distributional hypothesis, John Firth (1957)

tezgüino

	1	2	3	4
tezgüino	1	1	1	1
loud	0	0	0	0
motor oil	1	0	0	1
tortillas	0	1	0	1
choices	0	1	0	0
wine	1	1	1	0

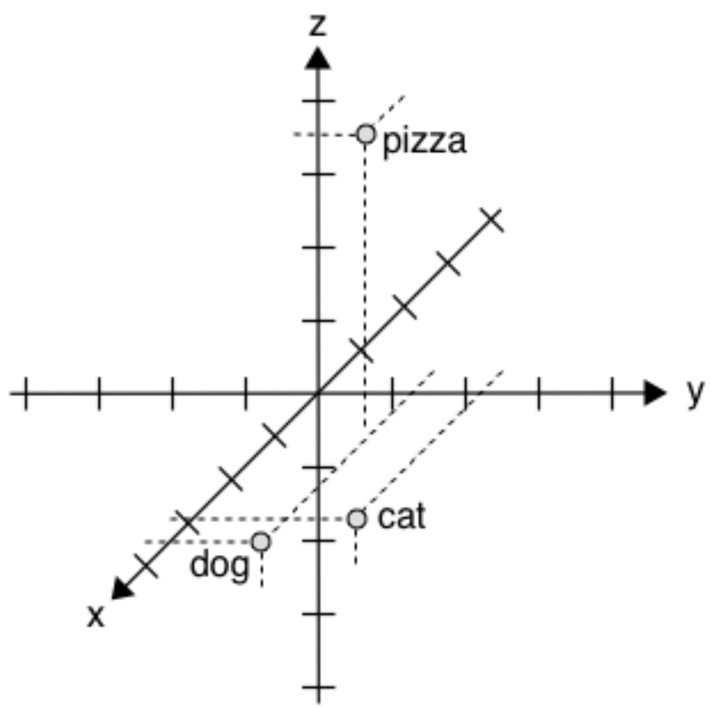
word representation

Contextual similarity

- Contexts
  - **1** A bottle of ___ is on the table
  - **2** Everybody likes ___
  - **3** Don’t have ___ before you drive
  - **4** We make ___ out of corn

- What other words fit into these contexts?

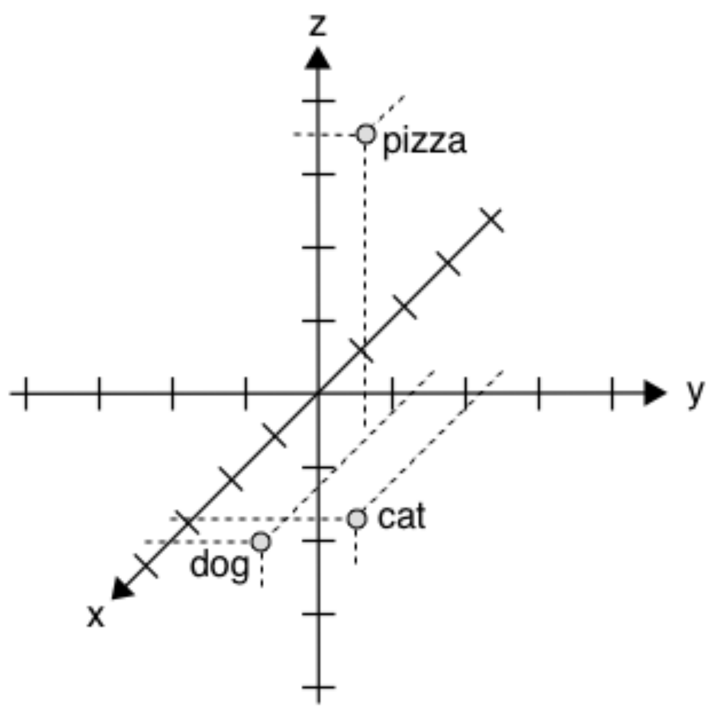
Distributional Word Embeddings



$$
\begin{aligned}
cat &= [0.7, 0.5, 0.1] \\
dog &= [0.8, 0.3, 0.1] \\
pizza &= [0.1, 0.2, 0.8]
\end{aligned}
$$

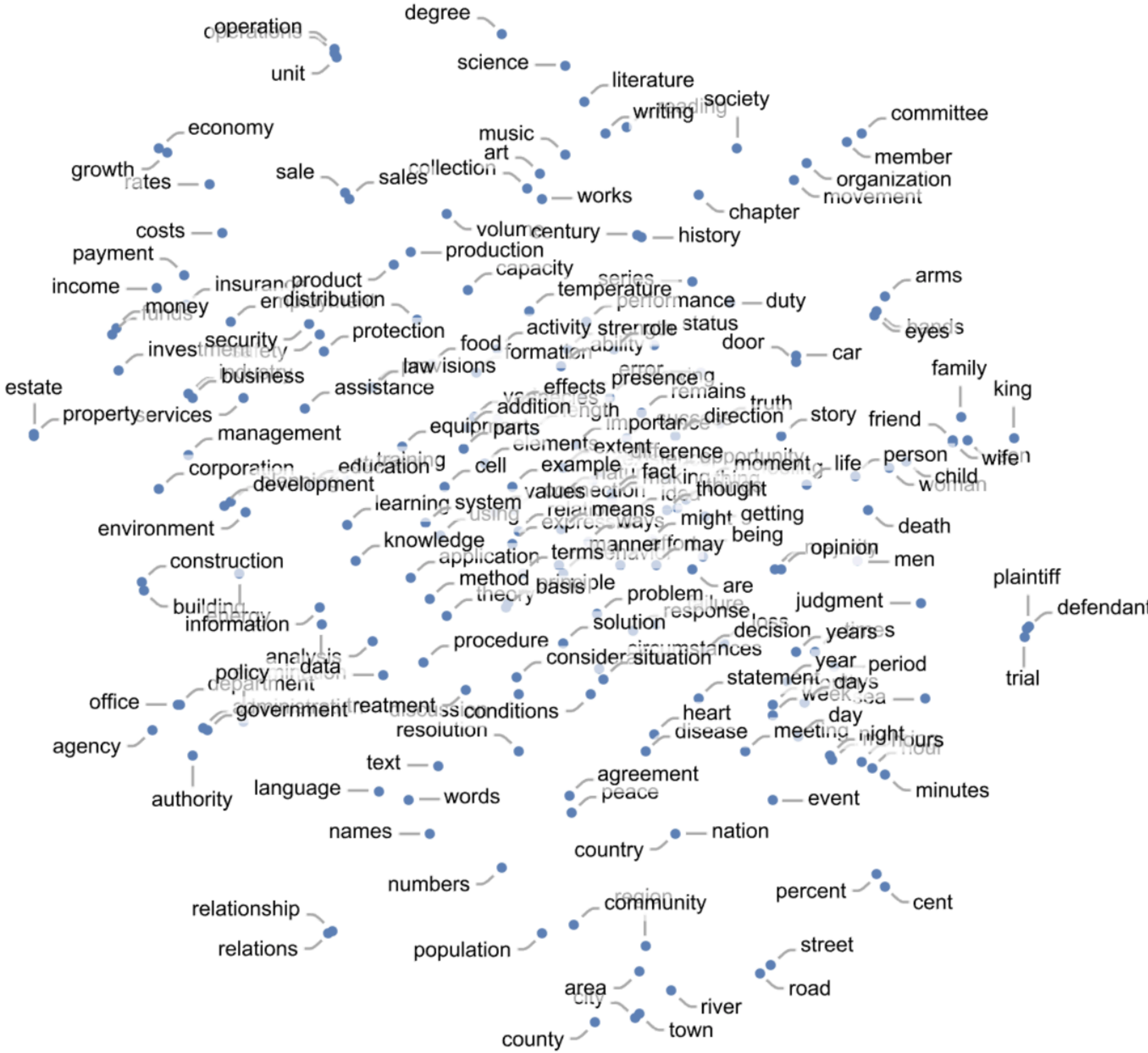
- Define dimensions that allow expressing a *context*
  - The vector for any particular word captures how strongly it is associated with each context
- For instance, in a 3 -dimensional space, the axis could have the semantic meaning
  - x -axis represents some concept of *"animal-ness"*
  - z -axis corresponds to *"food-ness"*

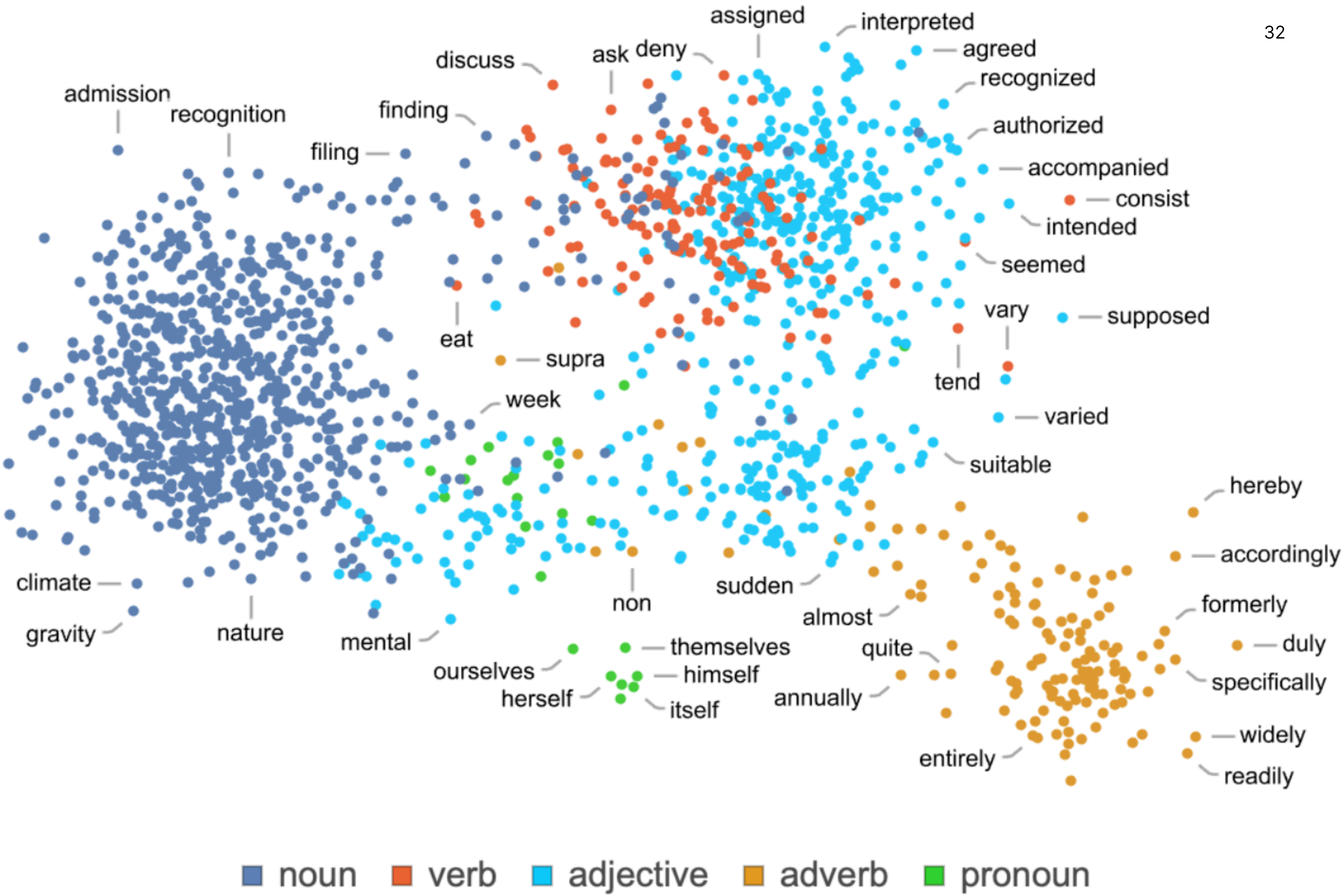
Distributional Word Embeddings



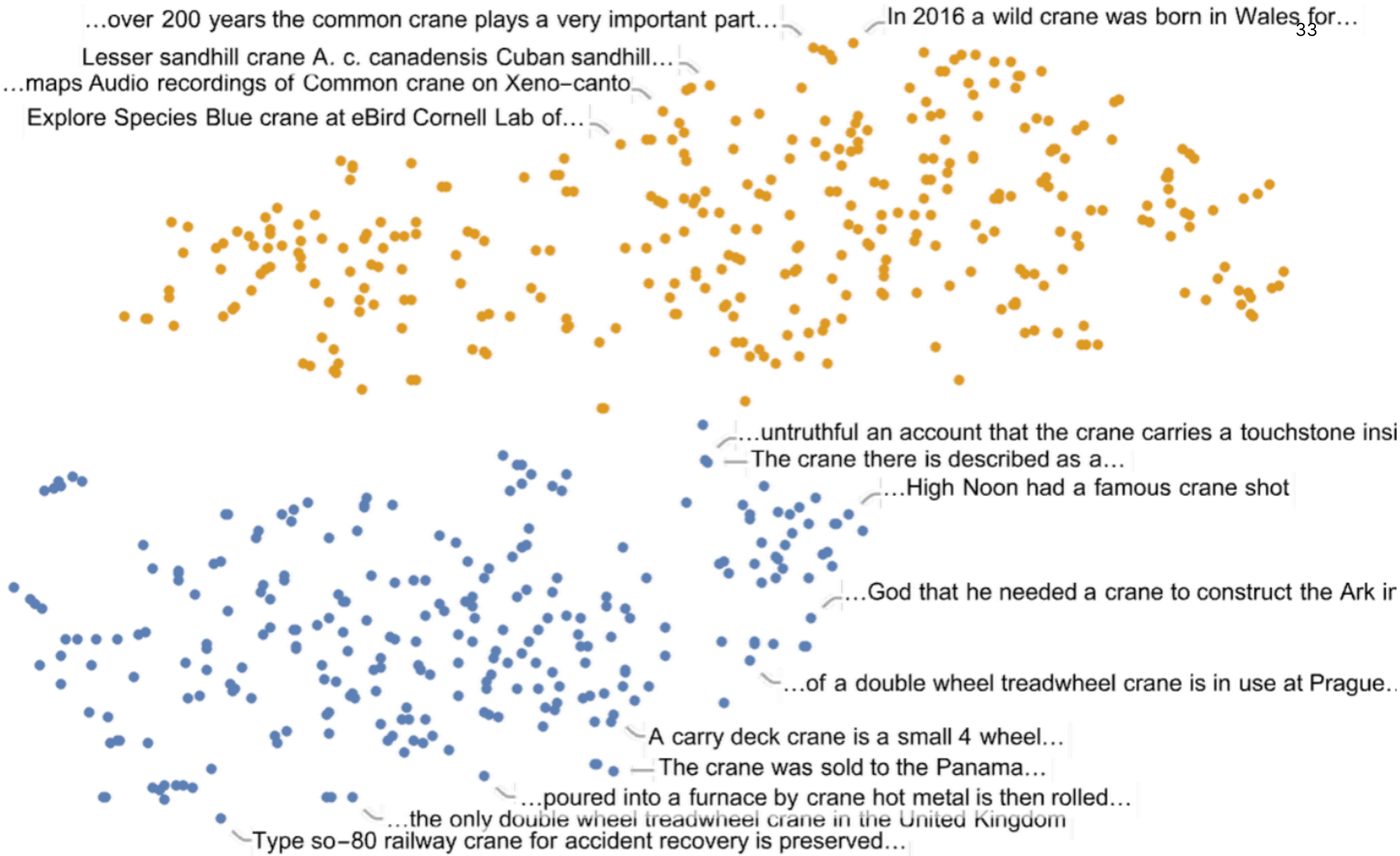
$$
\begin{aligned}
cat &= [0.7, 0.5, 0.1] \\
dog &= [0.8, 0.3, 0.1] \\
pizza &= [0.1, 0.2, 0.8]
\end{aligned}
$$

- Defining the axes is **difficult**
  - How many?
    - A lot less than the size of the dictionary (**dense vectors**)
    - But at least ~100-dimensional, to be effective
    - GPT-2 has ~1600, GPT-3 12288.
- How to assign values associated with the vectors?
  - Tens of millions of numbers to tweak


operation
degree
unit
science
literature
writing
society
committee
economy
music
art
member
growth
rates
sale
sales
collection
organization
works
movement
chapter
costs
century
history
payment
production
income
product
capacity
arms
money
distribution
temperature
duty
protection
activity
role
status
eyes
security
food
door
car
law
family
business
assistance
effects
presence
king
estate
addition
remains
truth
property
services
direction
story
friend
equipment
parts
management
extent
person
wife
corporation
education
cell
example
fact
moment
life
child
development
values
thought
learning
system
means
getting
death
environment
might
knowledge
terms
manner
may
being
opinion
men
construction
application
method
basis
are
plaintiff
problem
judgment
defendant
information
solution
procedure
decision
years
analysis
situation
trial
policy
data
year
period
statement
days
office
government
conditions
heart
day
resolution
disease
meeting
agency
night
hours
text
agreement
minutes
authority
language
words
event
peace
names
nation
country
numbers
percent
relationship
community
cent
relations
population
street
road
area
river
county
town



assigned
interpreted
agreed
recognized
authorized
accompanied
consist
intended
seemed
supposed
vary
tend
varied
suitable
discuss
ask
deny
admission
recognition
finding
filing
eat
supra
week
climate
gravity
nature
mental
non
sudden
almost
quite
annually
entirely
ourselves
themselves
himself
herself
itself
hereby
accordingly
formerly
duly
specifically
widely
readily
noun
verb
adjective
adverb
pronoun


...over 200 years the common crane plays a very important part...
In 2016 a wild crane was born in Wales for...
Lesser sandhill crane A. c. canadensis Cuban sandhill...
...maps Audio recordings of Common crane on Xeno-canto
Explore Species Blue crane at eBird Cornell Lab of...
...untruthful an account that the crane carries a touchstone insi
The crane there is described as a...
...High Noon had a famous crane shot
...God that he needed a crane to construct the Ark ir
...of a double wheel treadwheel crane is in use at Prague..
A carry deck crane is a small 4 wheel...
The crane was sold to the Panama...
...poured into a furnace by crane hot metal is then rolled...
...the only double wheel treadwheel crane in the United Kingdom
Type so-80 railway crane for accident recovery is preserved...

How to calculate Word Embeddings?

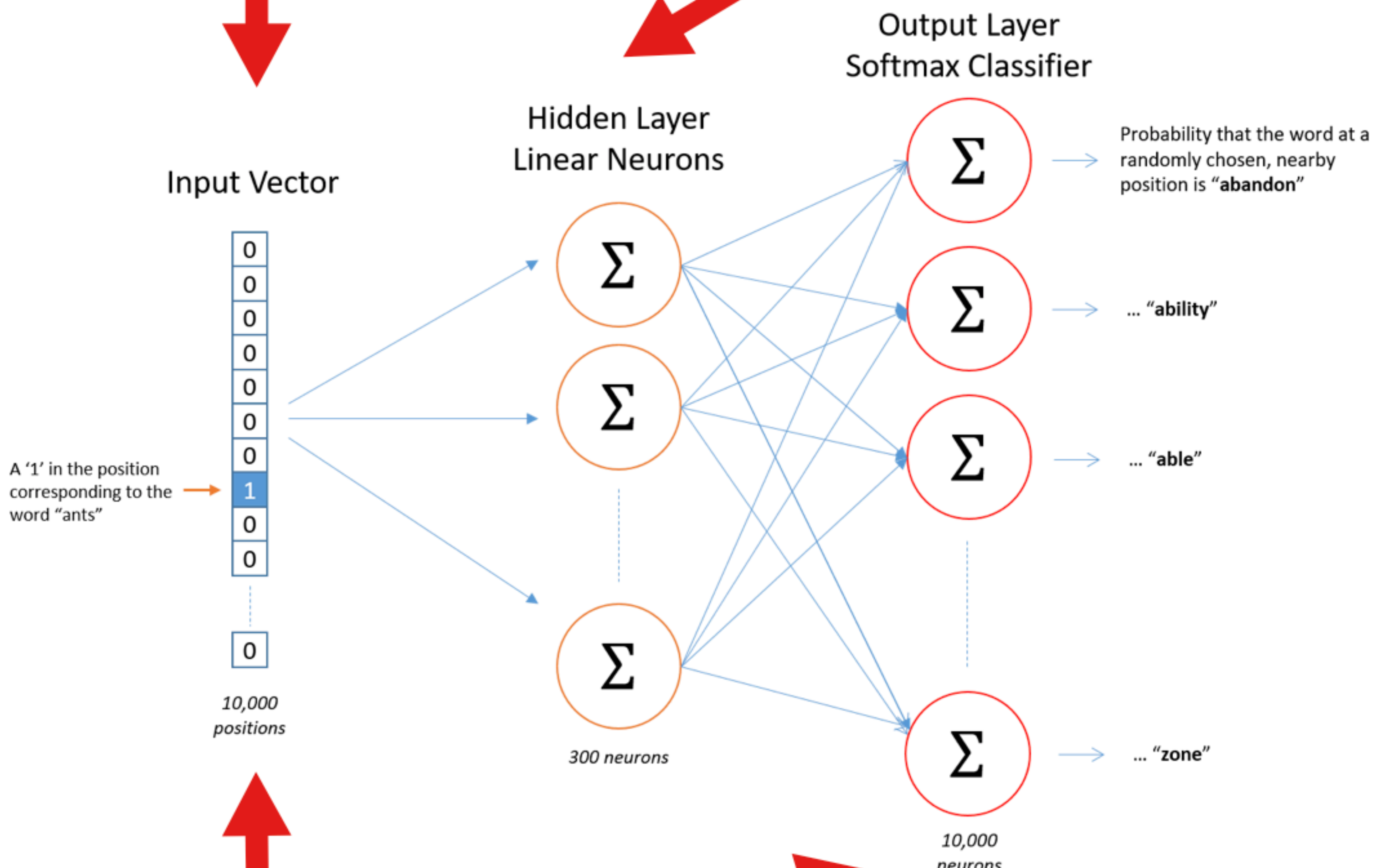
- With **machine learning models**
- Advanced topic
    - Several courses in the IPD master :)

Ok, just a sneak peak

- SKIPGRAM: Predict the probability of context words from a centre word
- **Input**: one-hot vector of the centre word
  - the size of the vocabulary
- **Output**: one-hot vector of the output words
  - the probability that the output word is selected to be in the context window
- *Embeddings*: lower-dimensional representation of *context* of co-occurence

This is the word vector (embedding) that we want to learn

One-hot input vector

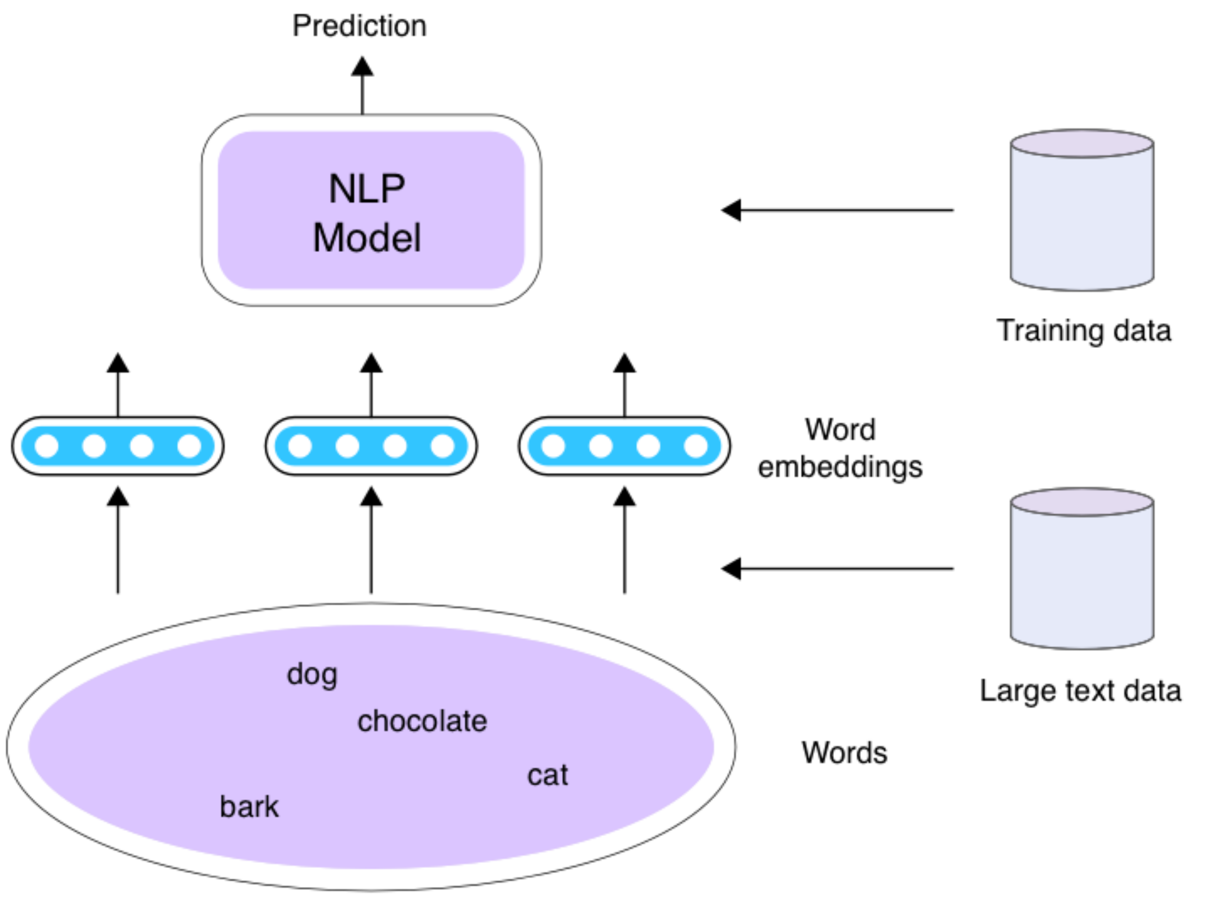


Size of vocabulary

Using Word Embeddings

How can embeddings be used with NLP Models?

- Word embeddings are *trained* from a corpus
  - And then they can be reused!
- 3 scenarios



Scenario 1

- Train word embeddings and your model at the same time using the train set for the task

Scenario 2: Fine-Tuning

- Initialise the model using the pre-trained word embeddings
  - e.g., train on Wikipedia, or large Web corpora
- Keep the embedding fixed while training the model for the task
  - Another example of *transfer learning*

Scenario 3: Adaptation

- The embeddings are adapted while the downstream model is trained, the train set for the task
  - Same as *Scenario 2*, but the embeddings are now more *close* to the words distribution in your training set

Evaluating Word Embeddings

How to evaluate word vectors?

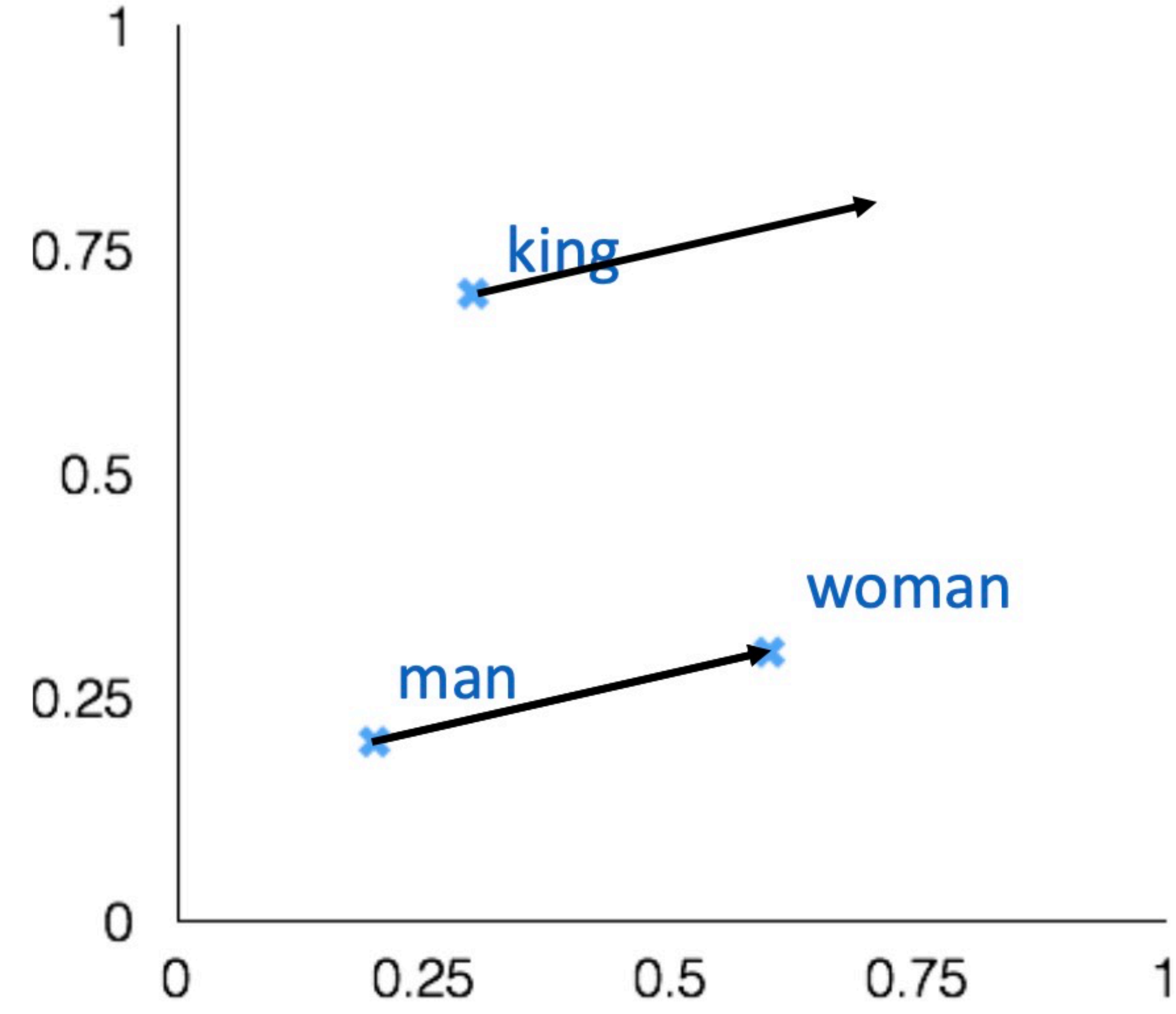
- **Intrinsic**: evaluation on a specific/intermediate subtask (e.g. analogy)
  - Fast to compute
    - It helps to understand that system
    - Not clear if helpful unless correlation to the actual task is established
- **Extrinsic**: evaluation of a real task
  - It can take a long time to compute the accuracy
  - Unclear if the subsystem is the problem or if it is an interaction with other subsystems

Intrinsic evaluation

- Word vector **analogies**

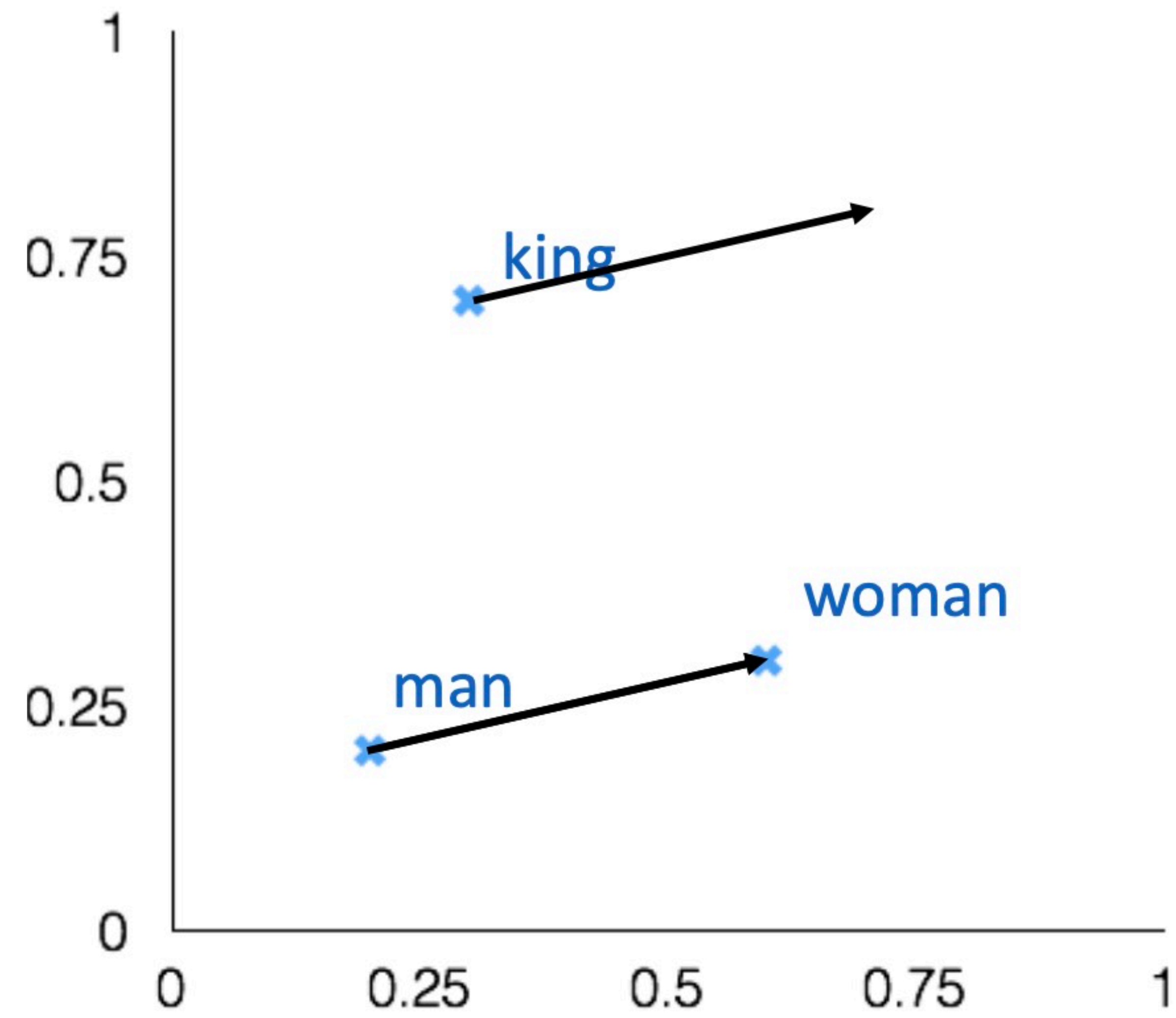
$$a : b = c :?$$

$$man : woman = king :?$$

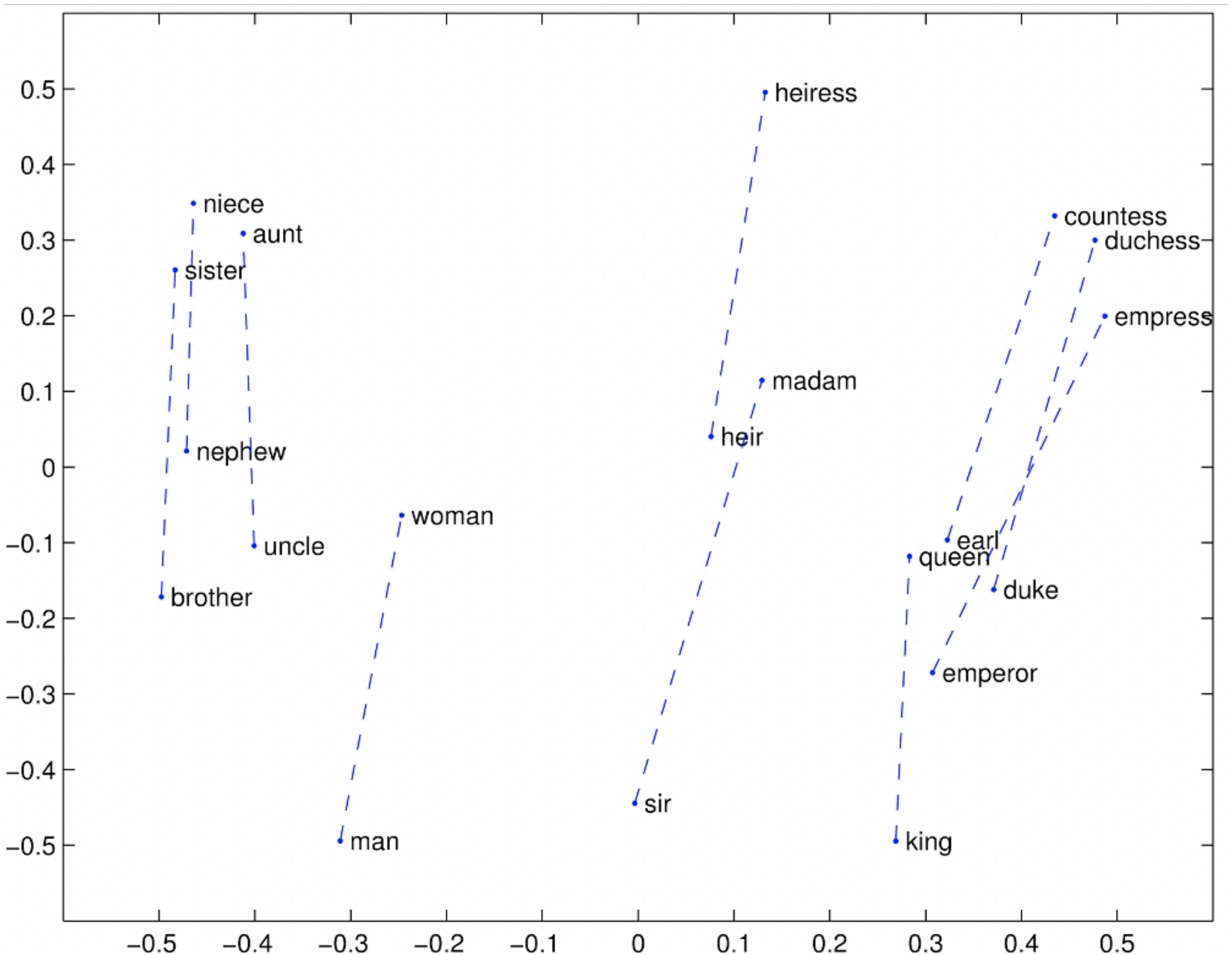


Intrinsic evaluation

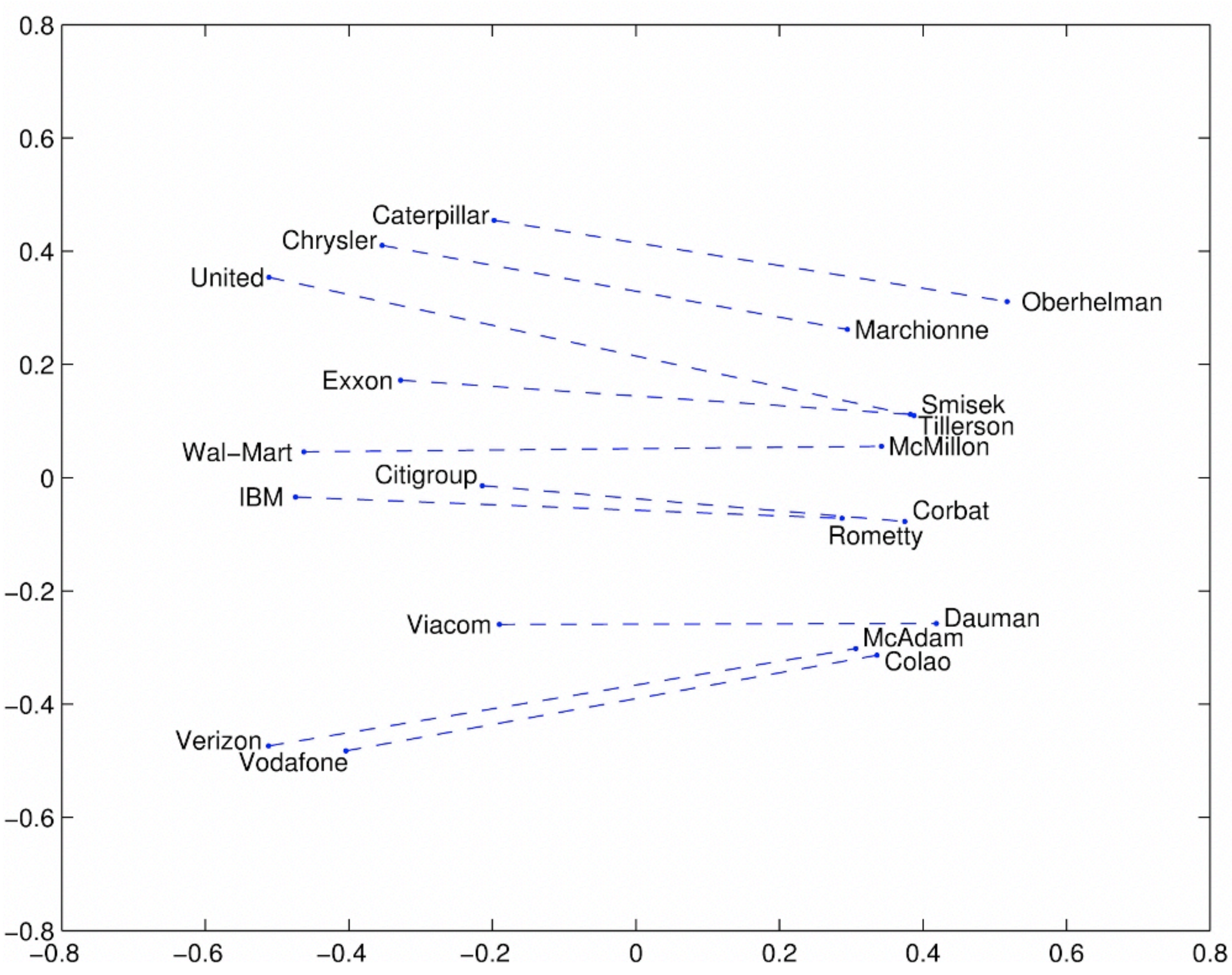
- Find a word such that the vector is closest (cosine similarity) to $vec[man] - vec[woman] + vec[king]$
  - Correct if the word found is $queen$
- Can be applied to test for syntactic analogy as well
  - $Quick : quickly = slow : slowly$



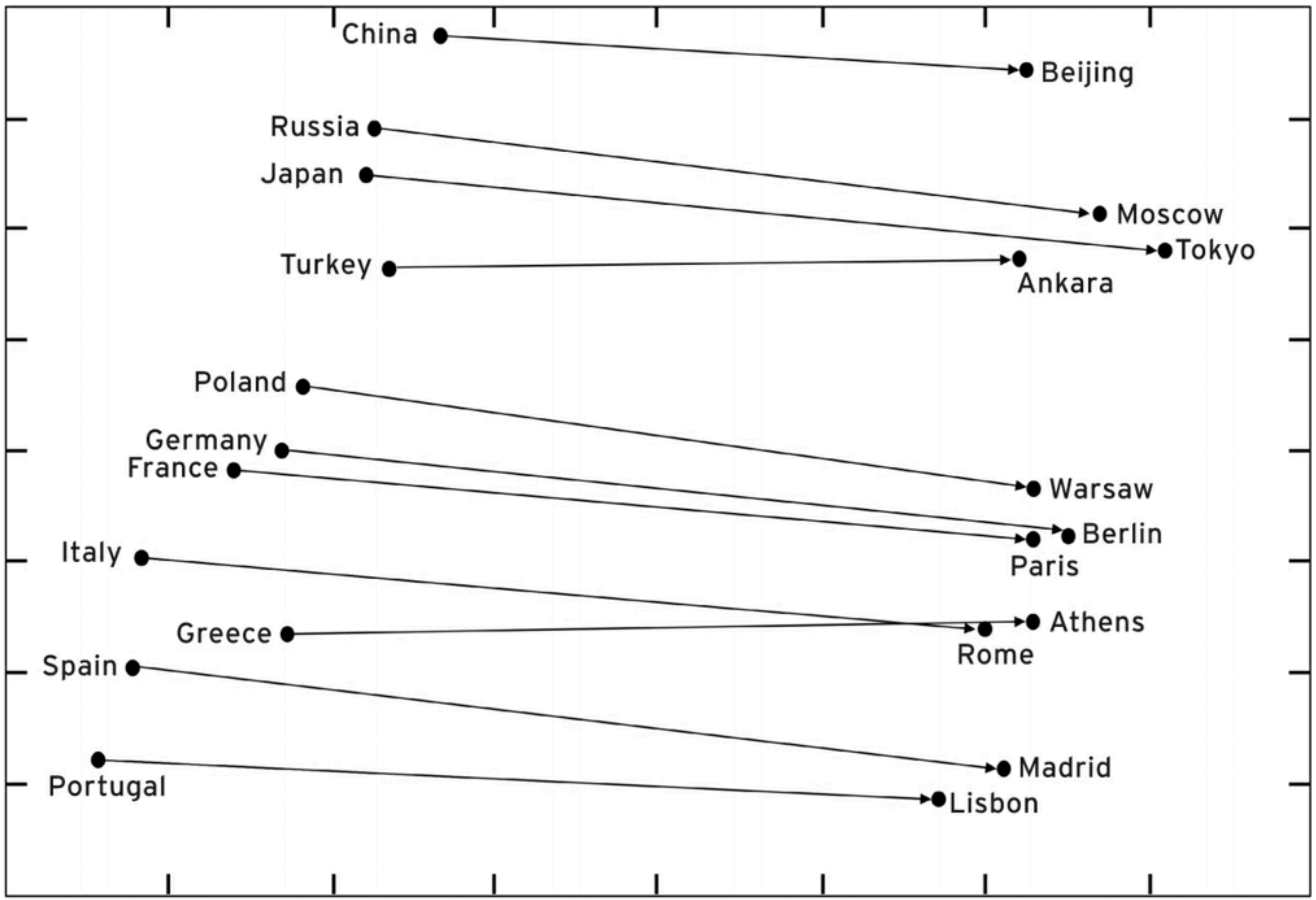
Gender relation



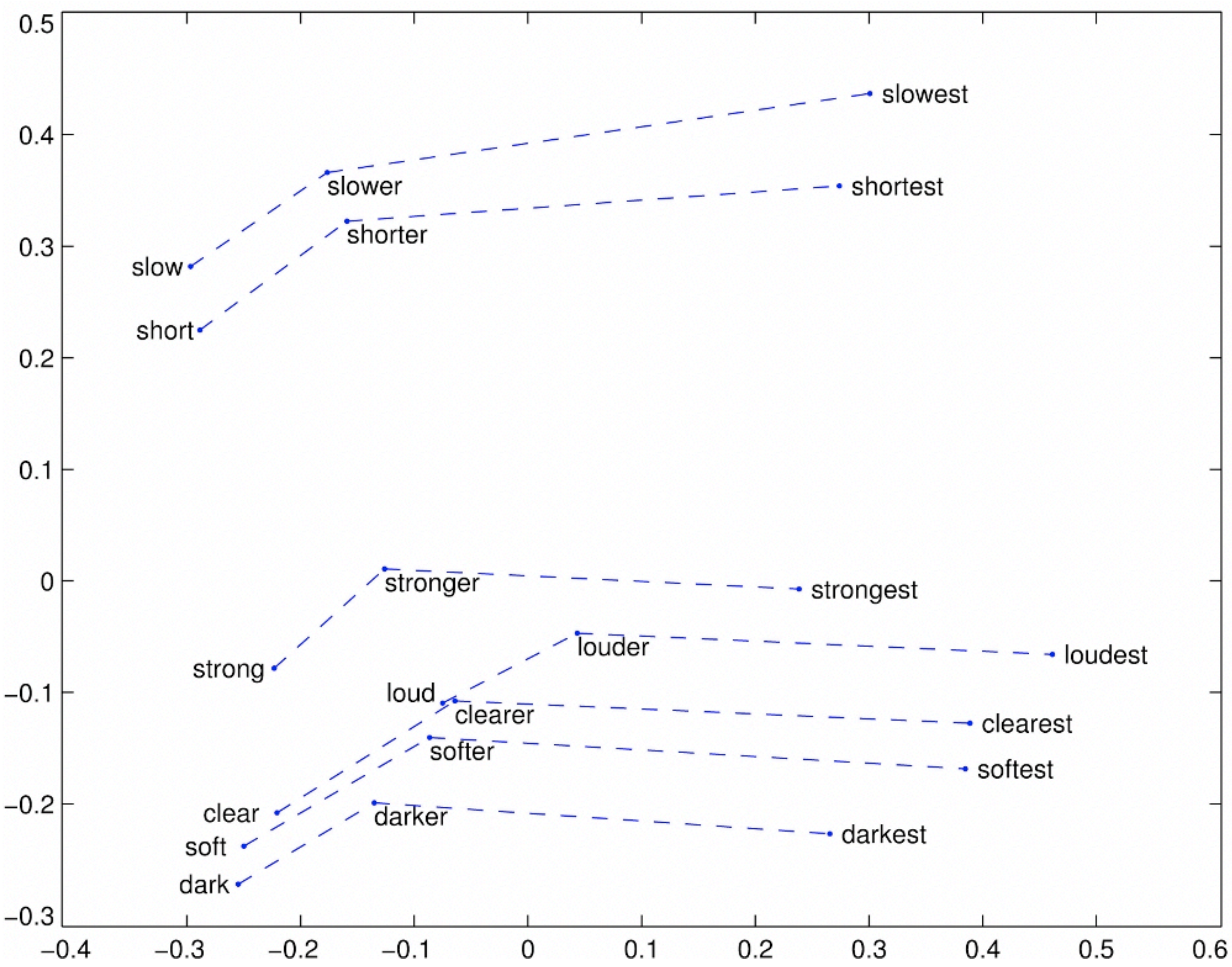
Company - CEO


Caterpillar
Chrysler
United
Exxon
Wal−Mart
Citigroup
IBM
Viacom
Verizon
Vodafone
Oberhelman
Marchionne
Smisek
Tillerson
McMillon
Corbat
Rometty
Dauman
McAdam
Colao
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8

Countries and their capital



Comparatives and Superlatives



There are problems, of course

- By exploring the semantic space, you can also find analogies like
  - $Thirsty$ is to $drink$ as $tired$ is to $drunk$
  - $Fish$ is to $water$ as $bird$ is to $hydrant$

Biases in word vectors might leak through to produce **unexpected**, **hard-to-predict** biases

- man is to $woman$ as $computer\ programmer$ is to ____
- $woman$ is to man as $computer\ programmer$ is to ____
- man is to $genius$ as $woman$ is to ____
- $woman$ is to $genius$ as man is to ____

- man is to $woman$ as $computer\ programmer$ is to homemaker
- $woman$ is to man as $computer\ programmer$ is to mechanical engineer
- man is to $genius$ as $woman$ is to muse
- $woman$ is to $genius$ as man is to geniuses

Machine Learning for Design

Lecture 6

Natural Language Processing - Part 2

Credits

CIS 419/519 Applied Machine Learning. Eric Eaton, Dinesh Jayaraman. https://www.seas.upenn.edu/~cis519/spring2020/

EECS498: Conversational AI. Kevin Leach. https://dijkstra.eecs.umich.edu/eecs498/

CS 4650/7650: Natural Language Processing. Diyi Yang. https://www.cc.gatech.edu/classes/AY2020/cs7650_spring/

Natural Language Processing. Alan W Black and David Mortensen. http://demo.clab.cs.cmu.edu/NLP/

IN4325 Information Retrieval. Jie Yang.

Speech and Language Processing, An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Third Edition. Daniel Jurafsky, James H. Martin.

Natural Language Processing, Jacob Eisenstein, 2018.