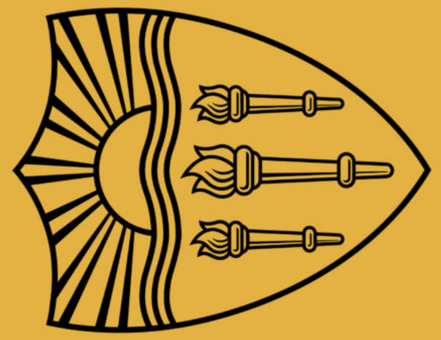


Lecture 6:
Word Embeddings

Instructor: Swabha Swayamdipta
USC CSCI 499 LMs in NLP
Feb 5, 2024 Fall

Slides mostly adapted from Dan Jurafsky, some from Mohit Iyyer

Announcements + Logistics

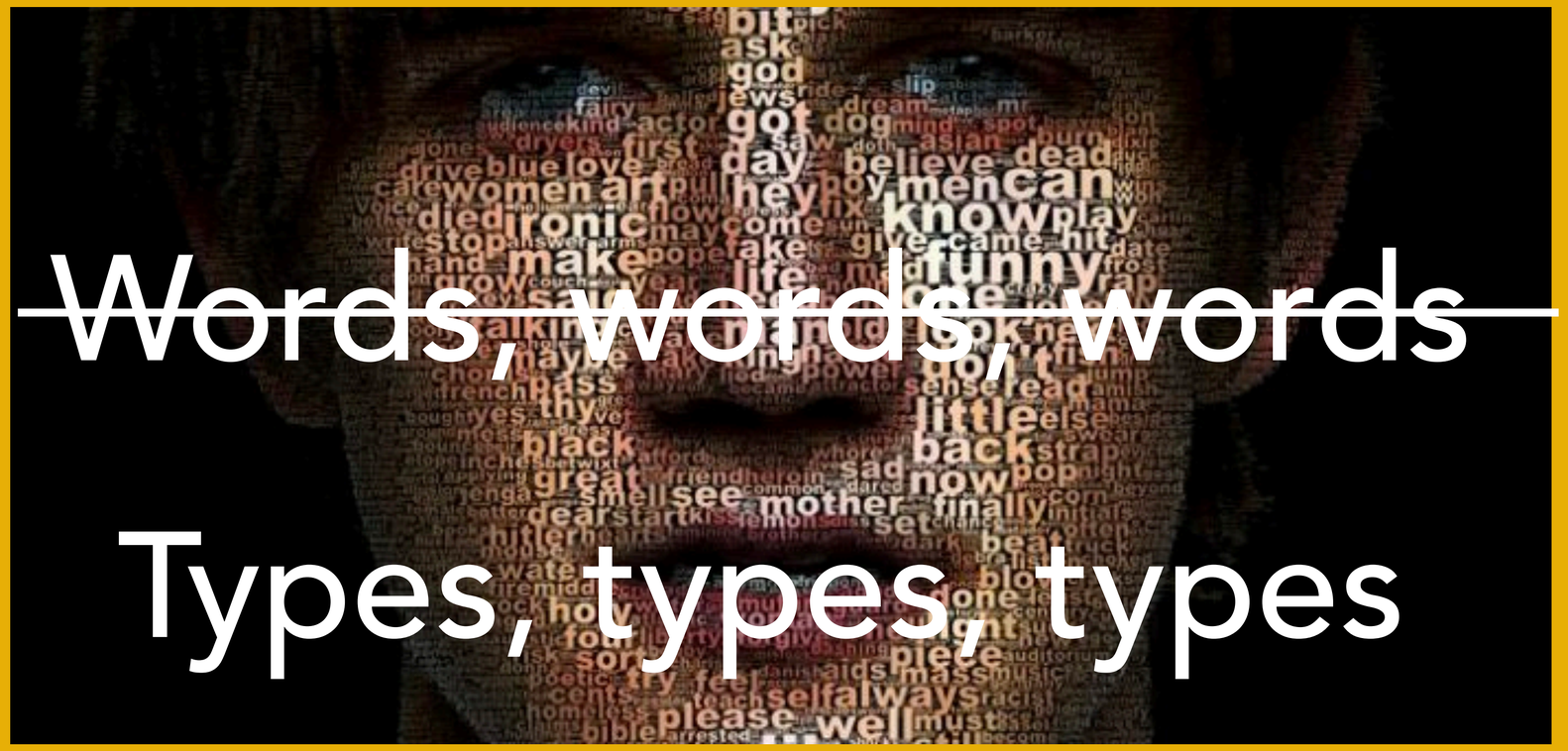
- Project proposal is due by Wed 2/7, 11:59 PM PT
  - Please use Slack / Piazza to find teammates
  - Only three (incomplete) teams announced so far
- We have shared a link for (anonymous) feedback throughout the duration of the course - please use wisely and for constructive feedback
- Graded Quiz 1 distributed today during break
- HW2 Released on Wednesday, 2/7

Quiz 1 - Answers

Redacted

Early Neural Language Models

This class: Word Embeddings — the most important component of a neural LM

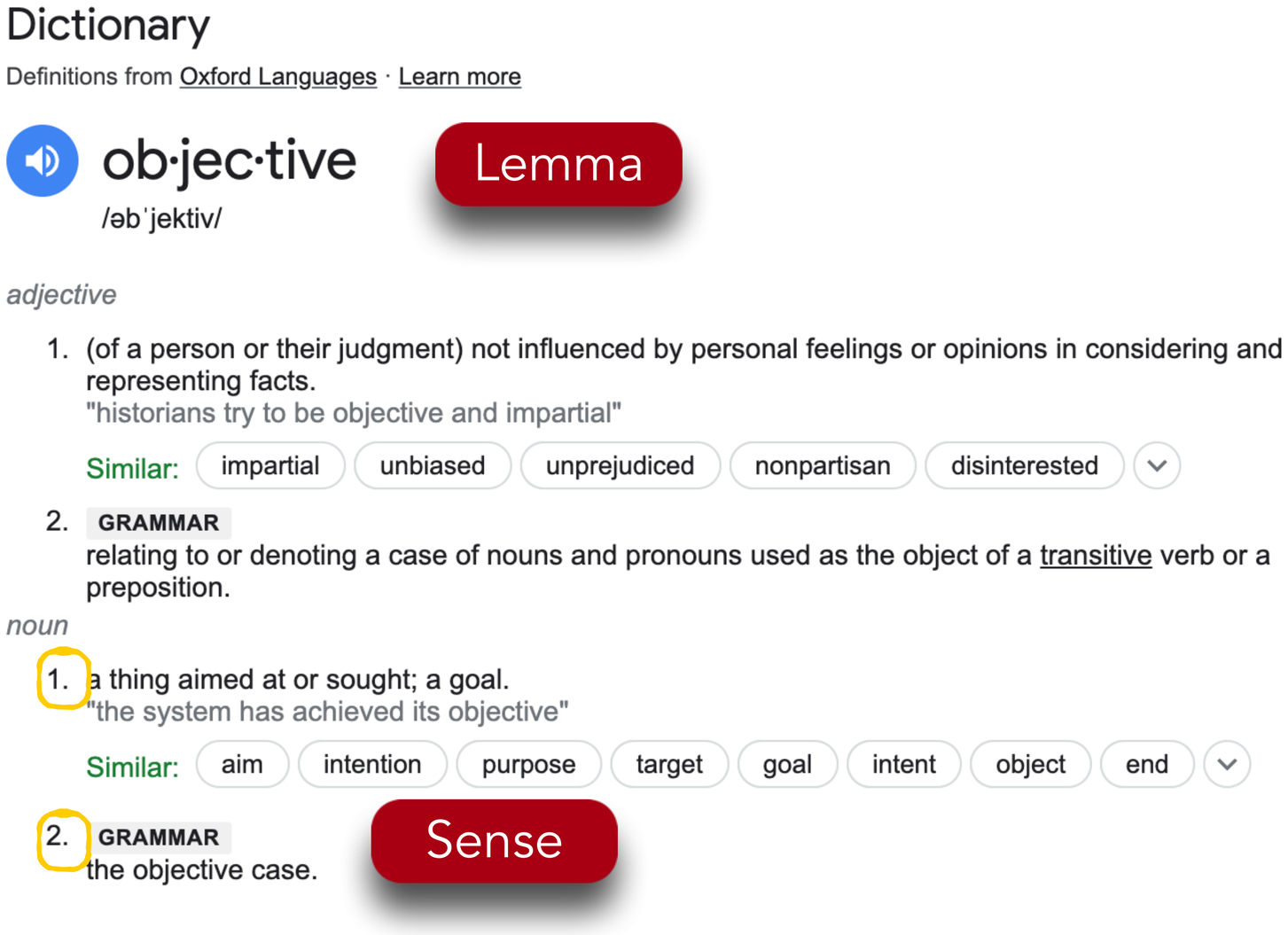

~~Words, words, words~~
Types, types, types

What do words mean?

A sense or "concept" is the meaning component of a word

Lemmas
- Canonical form
- For example,
  break, breaks, broke, broken and breaking all share the lemma "break"

Can be polysemous (have multiple senses)



Synonymy

Synonyms: words that have the same meaning in some or all contexts

- couch / sofa
- big / large
- automobile / car
- vomit / throw up
- water / H_20

Is perfect synonymy possible?

Perfect Synonymy might not be possible…

- Even if many aspects of meaning are identical
- Still may differ based on politeness, slang, register, genre, etc.
  - e.g. cannot use H_20 in a surfing guide!

Similarity

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

word1	word2
vanish	disappear
behave	obey
belief	impression
muscle	bone
modest	flexible
hole	agreement

Human assessment of word similarity

Simlex-999 dataset (Hill et al., 2015)

Not to be confused with word association / relatedness:

- couch / sofa vs. couch / pillow

Antonymy

Senses that are opposites with respect to only one feature of meaning

Otherwise, they are very similar!

- e.g. dark/light, short/long, fast/slow, rise/fall, hot/cold, up/down, in/out

More formally: antonyms can

- define a binary opposition or be at opposite ends of a scale
  - e.g. long/short, fast/slow
- be reversives: denote opposing processes
  - rise/fall, up/down

WordNet

- WordNet® is a large lexical database of English
- Nouns, verbs, adjectives and adverbs are grouped into sets of synonyms (synsets), each expressing a distinct concept
- Relations between synsets:
  - Super-subordinate relations (hyperonymy, hyponymy or ISA relation)
    - an armchair is a kind of chair, chair is a kind of furniture
  - Meronymy (part-of)
    - chair has legs
  - Antonymy

n-grams and Semantics

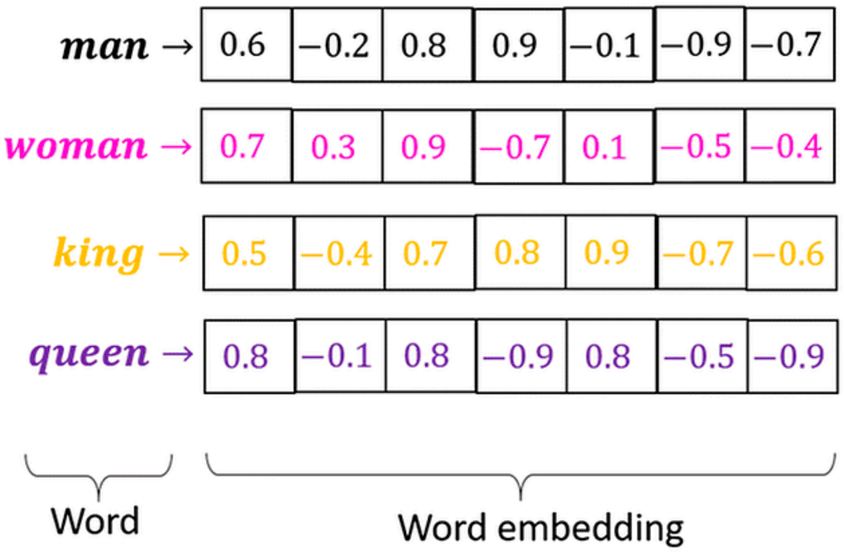
- Were feature representations!
  - $\mathbf{x}$'s in machine learning, which are associated with parameters
- Just strings - atomic symbols!
  - As n increases, we get strings that co-occur
- n-grams do not represent meaning well
  - Do not tell us that the word "rancor" is close in meaning to the word "hatred"
  - Or that "Rise" and "Fall" have opposite meanings
  - Let alone more complex is-a or part-of relations
- **Discrete** representations of meaning!
- Later: feature representations which are continuous

Words as Vectors

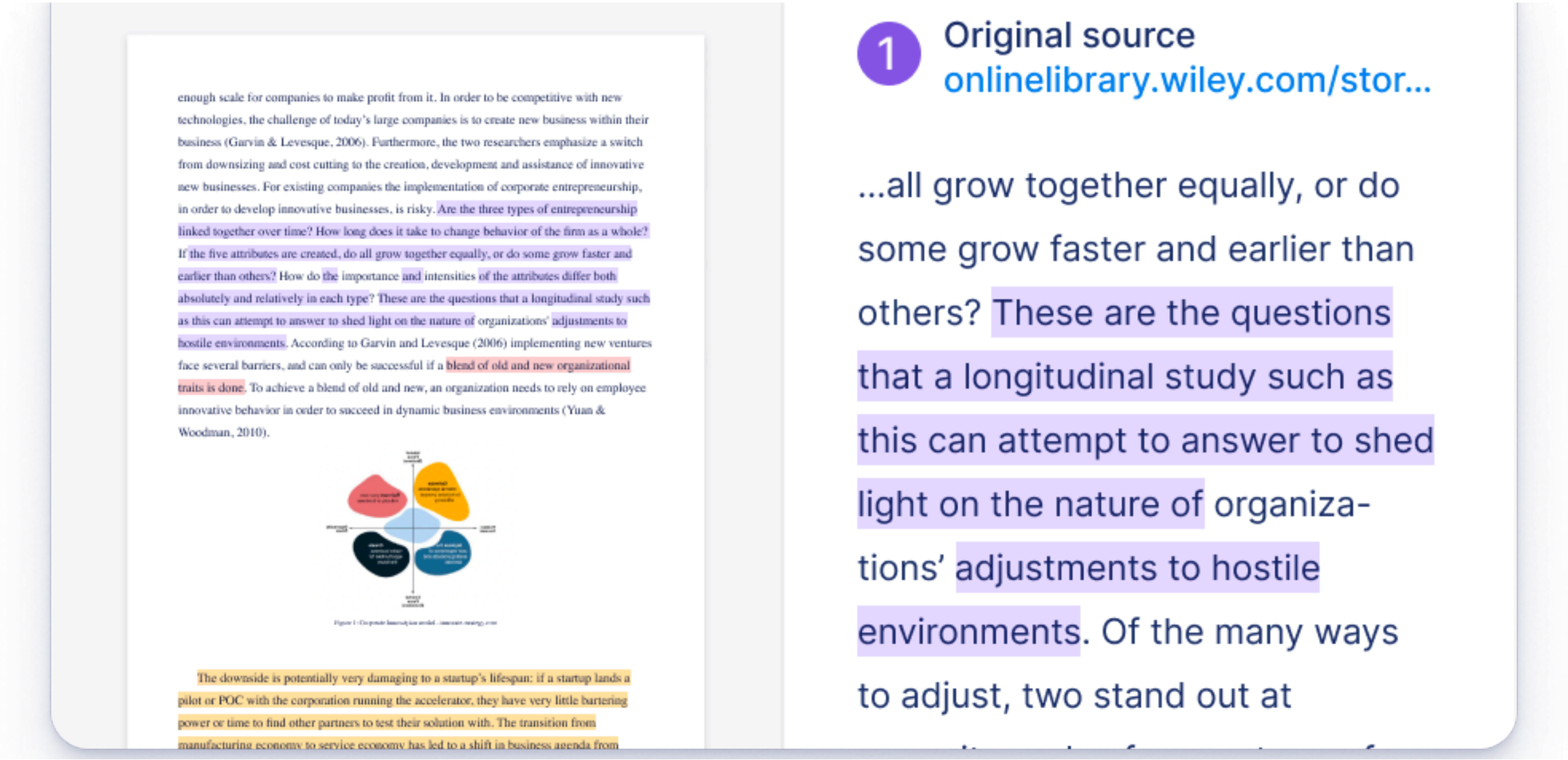
In NLP, we commonly represent word types with vectors!

- Very useful in capturing similarity between words, and other forms of lexical semantics (e.g. synonymy, hypernyms, antonymy)
- Computing the similarity between two words (or phrases, or documents) is extremely useful for many NLP tasks
  - Q: How **tall** is Mount Everest?
  - A: The official **height** of Mount Everest is 29029 ft

Why?



- Similarity for plagiarism detection
- Word similarity can lead to sentence and document similarity



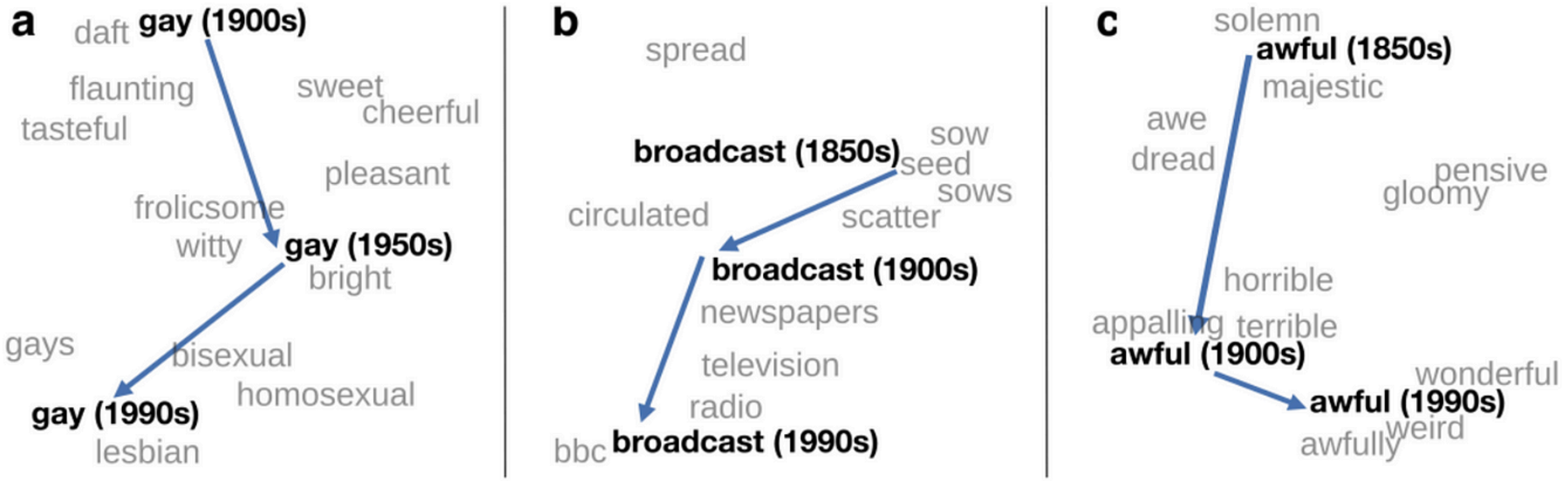
enough scale for companies to make profit from it. In order to be competitive with new technologies, the challenge of today's large companies is to create new business within their business (Garvin & Levesque, 2006). Furthermore, the two researchers emphasize a switch from downsizing and cost cutting to the creation, development and assistance of innovative new businesses. For existing companies the implementation of corporate entrepreneurship, in order to develop innovative businesses, is risky. Are the three types of entrepreneurship linked together over time? How long does it take to change behavior of the firm as a whole? If the five attributes are created, do all grow together equally, or do some grow faster and earlier than others? How do the importance and intensities of the attributes differ both absolutely and relatively in each type? These are the questions that a longitudinal study such as this can attempt to answer to shed light on the nature of organizations' adjustments to hostile environments. According to Garvin and Levesque (2006) implementing new ventures face several barriers, and can only be successful if a blend of old and new organizational traits is done. To achieve a blend of old and new, an organization needs to rely on employee innovative behavior in order to succeed in dynamic business environments (Yuan & Woodman, 2010).

The downside is potentially very damaging to a startup's lifespan: if a startup lands a pilot or POC with the corporation running the accelerator, they have very little bartering power or time to find other partners to test their solution with. The transition from manufacturing economy to service economy has led to a shift in business agenda from

1 Original source
onlinelibrary.wiley.com/stor...

...all grow together equally, or do some grow faster and earlier than others? These are the questions that a longitudinal study such as this can attempt to answer to shed light on the nature of organiza-tions' adjustments to hostile environments. Of the many ways to adjust, two stand out at

- Visualizing semantic change over time
- New words: dank, cheugy, rizz, shook, situationship



~30 million books, 1850-1990, Google Books data

"You shall know a word by the company it keeps."

- Firth (1957)

Word Meaning via Language Use

- The meaning of a word can be given by its distribution in language usage:
  - One way to define "usage": words are defined by their environments
    - Neighboring words or grammatical environments
- Intuitions: Zellig Harris (1954):
  - “oculist and eye-doctor … occur in almost the same environments”
  - “If A and B have almost identical environments we say that they are synonyms.”

A bottle of tesgüino is on the table
Everybody likes tesgüino
Tesgüino makes you drunk
We make tesgüino out of corn.

Two words are similar if they have similar word contexts

Word Meanings via Language Properties

- Meaning of a word can be determined by some properties of the word
- Point in space (Osgood et al., 1957)
- Example Properties: Affective Dimensions

	Word	Score		Word	Score
Valence	love	1.000		toxic	0.008
	happy	1.000		nightmare	0.005
Arousal	elated	0.960		mellow	0.069
	frenzy	0.965		napping	0.046
Dominance	powerful	0.991		weak	0.045
	leadership	0.983		empty	0.081

NRC VAD Lexicon (Mohammad 2018)

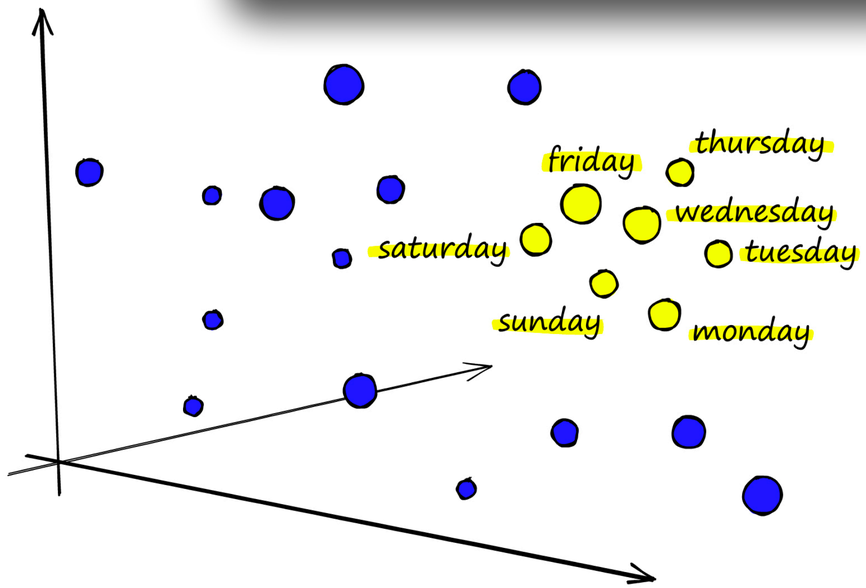
Defining meaning as a point in space based on distribution

- Each word = a vector
  - not just “good” or “word#45”
- Similar words are “nearby in semantic space”
- Can build this space automatically by seeing which words are nearby in text
- 2-D representation

Word Embeddings

Vector Semantics

- Represent a word as a point in a multidimensional semantic space
  - Space itself constructed from distribution of word neighbors
- Called an "embedding" because it's embedded into a space
- Fine-grained model of meaning for **similarity**



Every modern NLP algorithm uses embeddings as the representation of word meaning

Image Credit: Pinecone

荃者所以在鱼，得鱼而忘荃

言者所以在意，得意而忘言

"Nets are for fish; Once you get the fish, you can forget the net.

Words are for meaning; Once you get the meaning, you can forget the words"

– *Zhuangzi*

庄子

Cosine Similarity for Word Similarity

Cosine similarity of two vectors

$$\cos(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}||\vec{w}|}$$

$$= \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

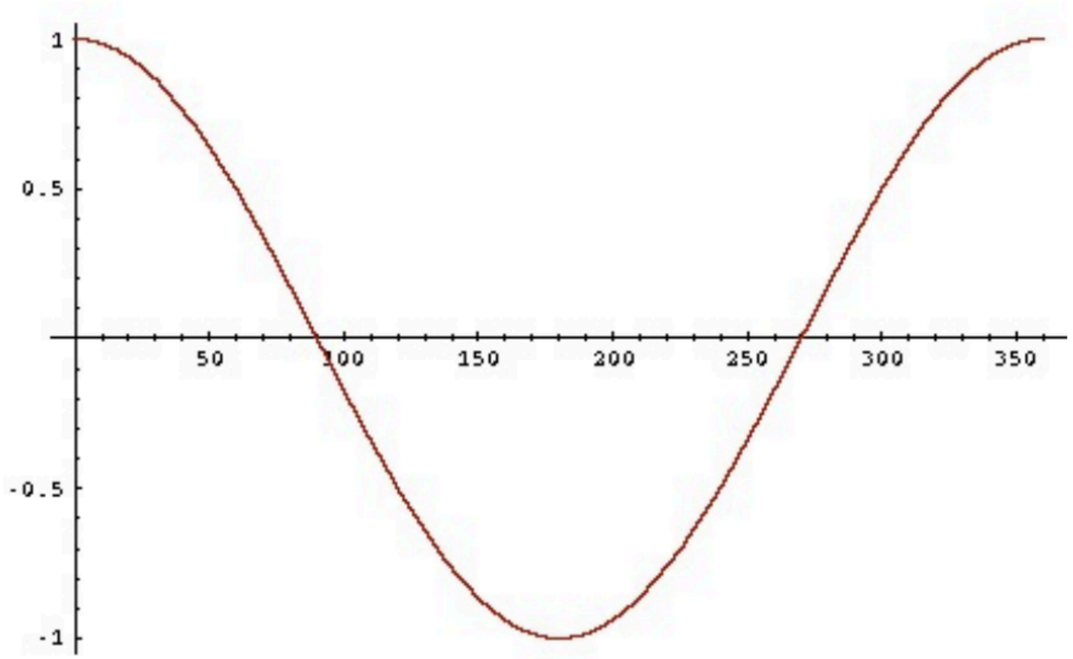
Based on the definition of the dot product between two vectors $\vec{a}$ and $\vec{b}$

$$\vec{v} \cdot \vec{w} = |\vec{v}||\vec{w}|\cos\theta$$

$$\cos\theta = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}||\vec{w}|}$$

Cosine as a similarity metric

-1: vectors point in opposite directions
+1: vectors point in same directions
0: vectors are orthogonal



Greater the cosine, more similar the words

n-grams as One-hot Vectors

vocabulary
i
hate
love
the
movie
film

movie = <0, 0, 0, 0, 1, 0>
film = <0, 0, 0, 0, 0, 1>

One hot vector

Unigram Vectors: Represent each word as a vector of zeros with a single 1 identifying the index of the word

How can we compute a vector representation such that the dot product correlates with word similarity?

Dot product is zero! These vectors are orthogonal

Term-document matrix

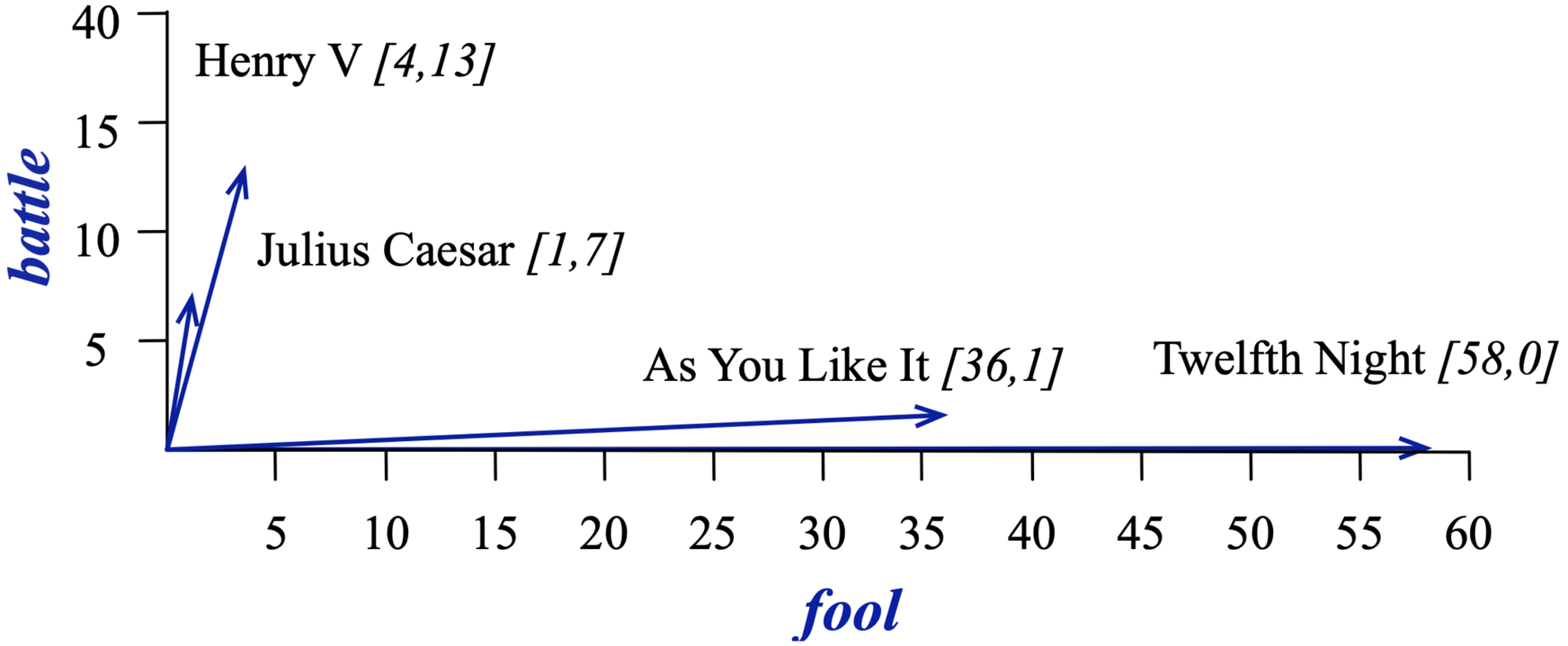
Let us consider a collection of documents and count how frequently a word (**term**) appears in each. A document could be a play or a Wikipedia article. In general, documents can be anything; we often call each paragraph a document!

Each **document** is represented by a vector 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

Visualizing document vectors

	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



- Vectors are similar for the two comedies
- Comedies are different from the other two (tragedies)
  - More fools, less battle

Words as vectors in a co-occurrence matrix

	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 appears in Julius Caesar and Henry V

"Fool" is the kind of word that appears in As You Like It and Twelfth Night

Word-word co-occurrence matrix

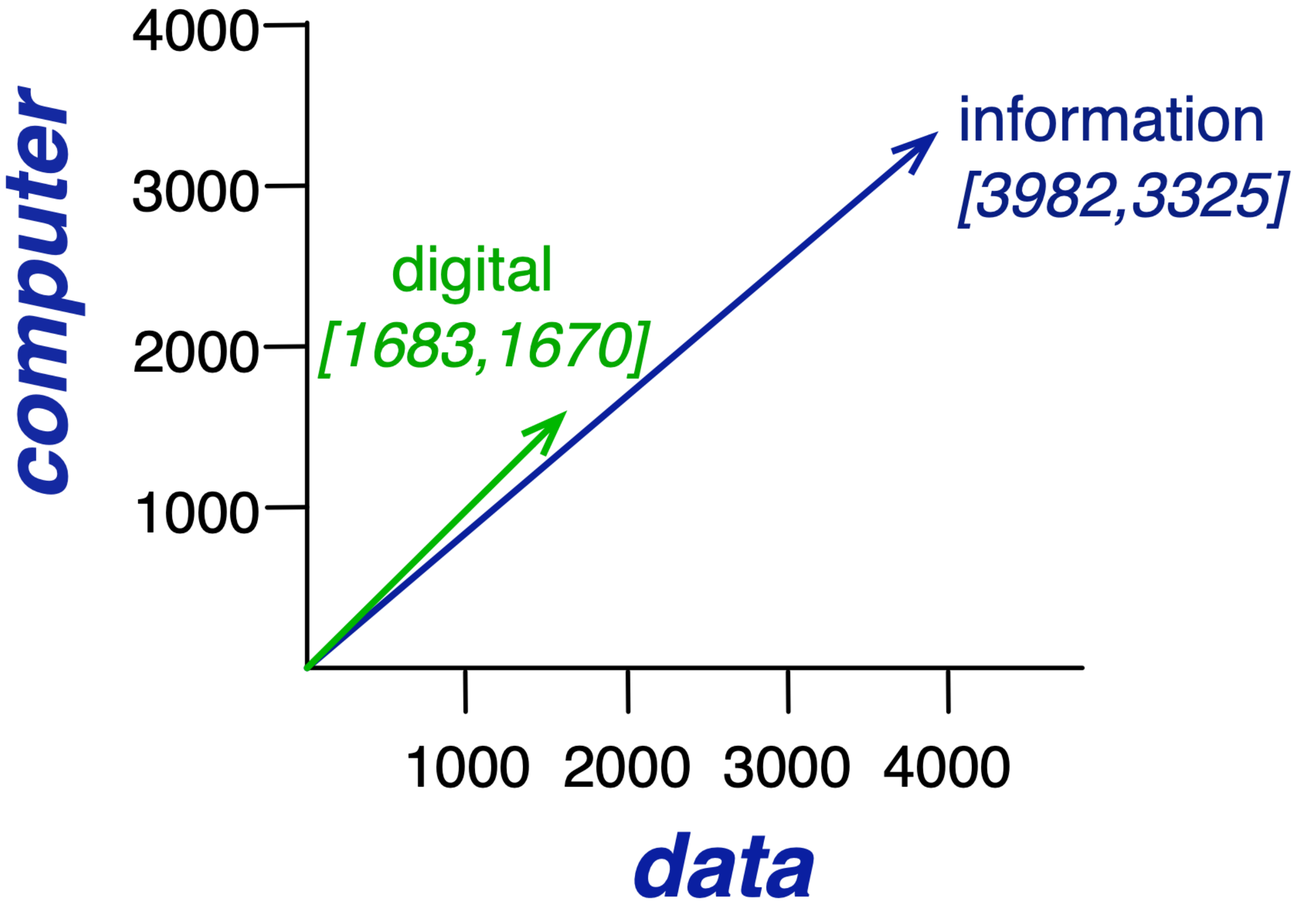
Context Window

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

Words, not 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	...

	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	...



Cosine Similarity

$$\cos(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}||\vec{w}|}$$

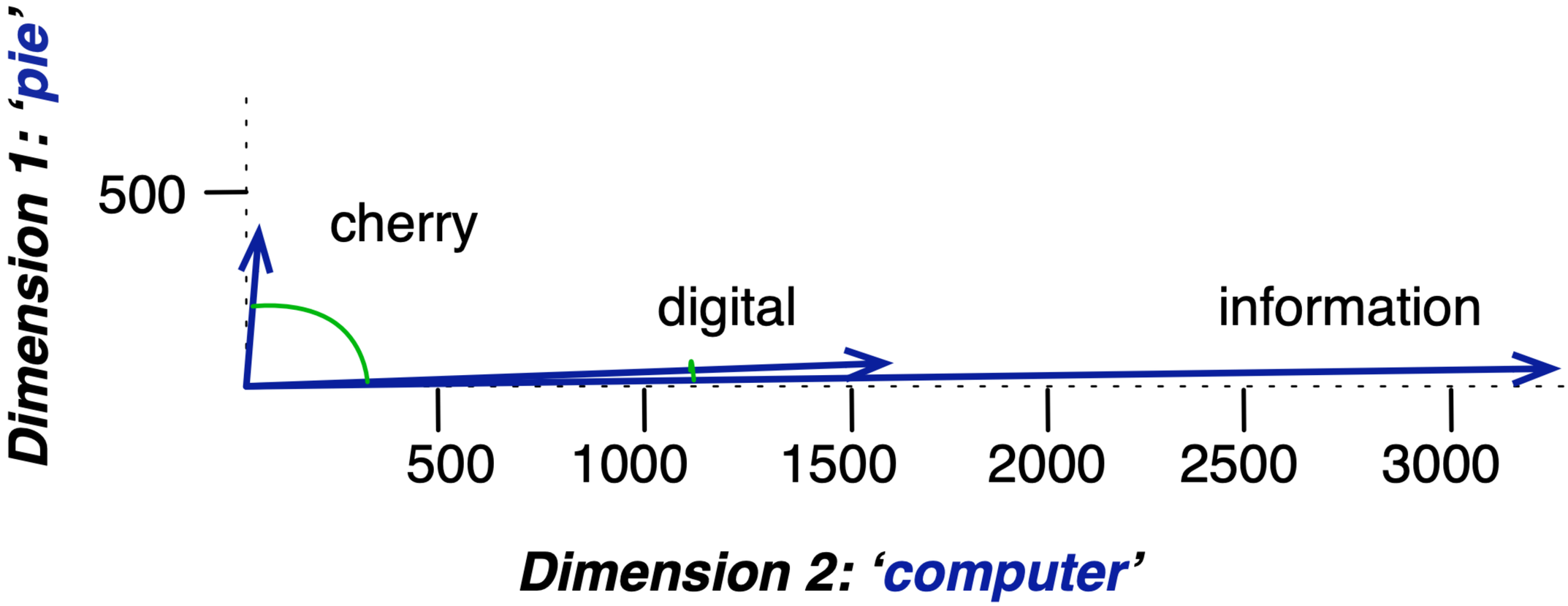
$$= \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

	pie	data	computer
cherry	442	8	2
digital	5	1683	1670
information	5	3982	3325

$$\cos(\text{cherry, information}) = \frac{442*5 + 8*3982 + 2*3325}{\sqrt{442^2 + 8^2 + 2^2}\sqrt{5^2 + 3982^2 + 3325^2}} = .017$$

$$\cos(\text{digital, information}) = \frac{5*5 + 1683*3982 + 1670*3325}{\sqrt{5^2 + 1683^2 + 1670^2}\sqrt{5^2 + 3982^2 + 3325^2}} = .996$$

Visualizing cosines



Raw frequencies though…

- …are a bad representation!
- The co-occurrence matrices we have seen represent each cell by word frequencies
- Frequency is clearly useful; if sugar appears a lot near apricot, that's useful information
- But overly frequent words like the, it, or they are not very informative about the context
- It's a paradox! How can we balance these two conflicting constraints?

	pie	data	computer
cherry	442	8	2
digital	5	1683	1670
information	5	3982	3325

Need some form of weighting!

Two different kinds of weighting

tf-idf: Term Frequency - Inverse Document Frequency

- Downweighting words like “the” or “if”
- Term-document matrices

PMI: Pointwise Mutual Information

- Considers the probability of words like “good” and “great” co-occurring
- Word co-occurrence matrices

Term Frequency

Term Frequency: frequency counting (usually log transformed)

$$\mathbf{tf}_{t,d} = \begin{cases} 1 + \log(\mathbf{count}(t,d)), & \textbf{if } \mathbf{count}(t,d) > 0 \\ 0, & \textbf{otherwise} \end{cases}$$

count(t, d) = # occurrences of word t in document d

Inverse Document Frequency

- Document Frequency: df_t is the number of documents t occurs in.
  - NOT collection frequency: total count across all documents
  - "Romeo" is very distinctive for one Shakespeare play
- Inverse Document Frequency: idf_t

$$\mathbf{idf}_t = \log_{10}\left(\frac{N}{\mathbf{df}_t}\right)$$

N = **total number of documents in the collection**

	Collection Frequency	Document Frequency
Romeo	113	1
action	113	31

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

What does IDF signify?

tf-idf

$$\text{tf}_{t,d} \times \text{idf}_{t,d}$$

	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

Raw Counts

tf-idf Weighted Counts

	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

Pointwise Mutual Information (PMI)

$$PMI(w_1, w_2) = \log \frac{P(w_1, w_2)}{P(w_1)P(w_2)}$$

- **PMI between two words:**
  - Do words w_1 and w_2 co-occur more than if they were independent?

- PMI ranges from $-\infty$ to $+\infty$
  - Negative values are problematic: words are co-occurring less than we expect by chance
  - Only reliable under an enormous corpora
    - Imagine w_1 and w_2 whose probability of occurrence is each 10^{-6}
    - Hard to be sure $P(w_1, w_2)$ is significantly different than 10^{-12}
  - So we just replace negative PMI values by 0

- **Positive PMI**

$$PPMI(w_1, w_2) = \max\left(0, \log \frac{P(w_1, w_2)}{P(w_1)P(w_2)}\right)$$

Church & Hanks 1989

Computing PPMI on a term-context matrix

Context c

Term w

	computer	data	result	pie	sugar	count(w)
cherry	2	8	9	442	25	486
strawberry	0	0	1	60	19	80
digital	1670	1683	85	5	4	3447
information	3325	3982	378	5	13	7703
count(context)	4997	5673	473	512	61	11716

Frequency $f(w_i, c_j)$ or f_{ij} is the # times w_i occurs in context c_j

$$P_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{i,j}} \qquad P_i = \sum_{j=1}^{C} P_{ij} \qquad P_j = \sum_{i=1}^{W} P_{ij}$$

$$PPMI(w_i, c_j) = PPMI_{i,j} = \max\left(0, \log \frac{P_{ij}}{P_i P_j}\right)$$

	computer	data	result	pie	sugar	count(w)
cherry	2	8	9	442	25	486
strawberry	0	0	1	60	19	80
digital	1670	1683	85	5	4	3447
information	3325	3982	378	5	13	7703
count(context)	4997	5673	473	512	61	11716

$$P_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{i,j}}$$

$$P_i = \sum_{j=1}^{C} P_{ij}$$

$$P_j = \sum_{i=1}^{W} P_{ij}$$

	p(w,context)					p(w)
	computer	data	result	pie	sugar	p(w)
cherry	0.0002	0.0007	0.0008	0.0377	0.0021	0.0415
strawberry	0.0000	0.0000	0.0001	0.0051	0.0016	0.0068
digital	0.1425	0.1436	0.0073	0.0004	0.0003	0.2942
information	0.2838	0.3399	0.0323	0.0004	0.0011	0.6575
p(context)	0.4265	0.4842	0.0404	0.0437	0.0052	

p(w=information,c=data) = 3982/111716 = .3399

p(w=information) = 7703/11716 = .6575

p(c=data) = 5673/11716 = .4842

	p(w,context)					p(w)
	computer	**data**	**result**	**pie**	**sugar**	**p(w)**
cherry	0.0002	0.0007	0.0008	0.0377	0.0021	0.0415
strawberry	0.0000	0.0000	0.0001	0.0051	0.0016	0.0068
digital	0.1425	0.1436	0.0073	0.0004	0.0003	0.2942
information	0.2838	0.3399	0.0323	0.0004	0.0011	0.6575
p(context)	0.4265	0.4842	0.0404	0.0437	0.0052	

	computer	**data**	**result**	**pie**	**sugar**
cherry	0	0	0	4.38	3.30
strawberry	0	0	0	4.10	5.51
digital	0.18	0.01	0	0	0
information	0.02	0.09	0.28	0	0

$$PPMI_{i,j} = \max\left(0, \log \frac{P_{ij}}{P_i P_j}\right)$$

pmi(information,data) = $\log_2$ (.3399 / (.6575*.4842)) = .0944

The problem…

- 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)

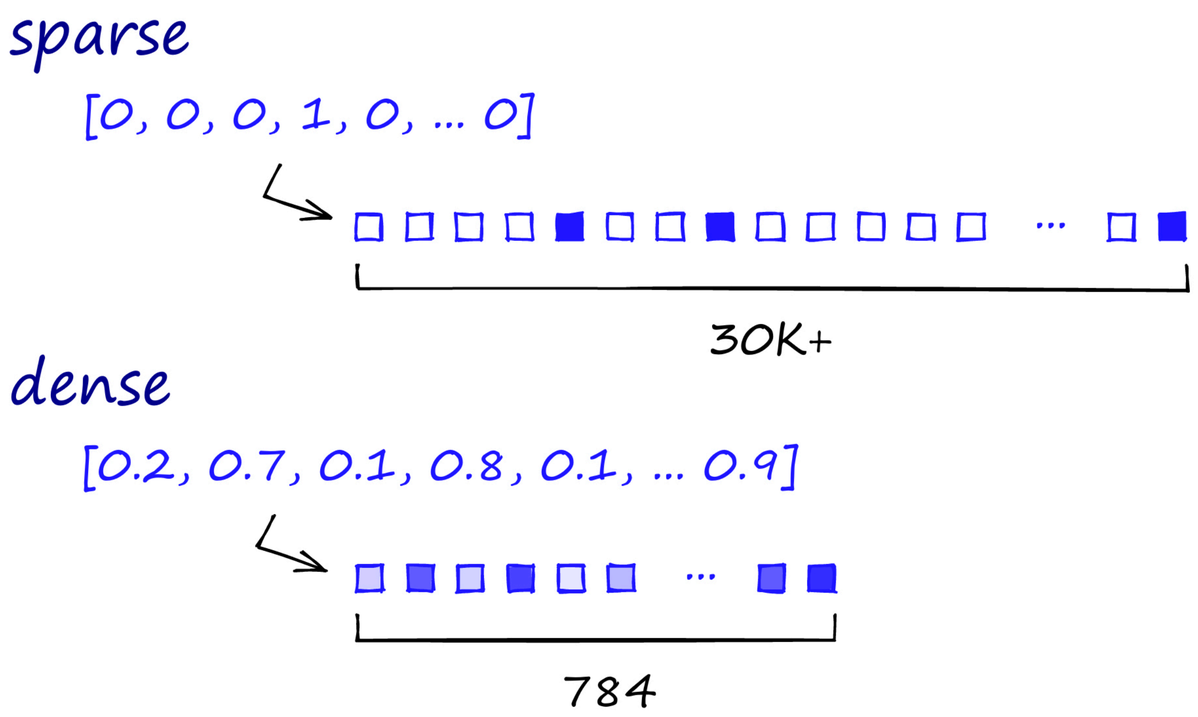


Image Credit: Pinecone

Sparse vs. Dense Vectors

- Why dense vectors?
  - Memory efficiency is not a problem for sparse 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

How to obtain dense vectors?

"Neural Language Model"-inspired models

- Word2vec (skipgram, CBOW), GloVe

Singular Value Decomposition (SVD)

- Special case: Latent Semantic Analysis (LSA)

Alternative to these "static embeddings":

- Contextual Embeddings (ELMo, BERT)
- Compute distinct embeddings for a word in its context
- Separate embeddings for each token of a word

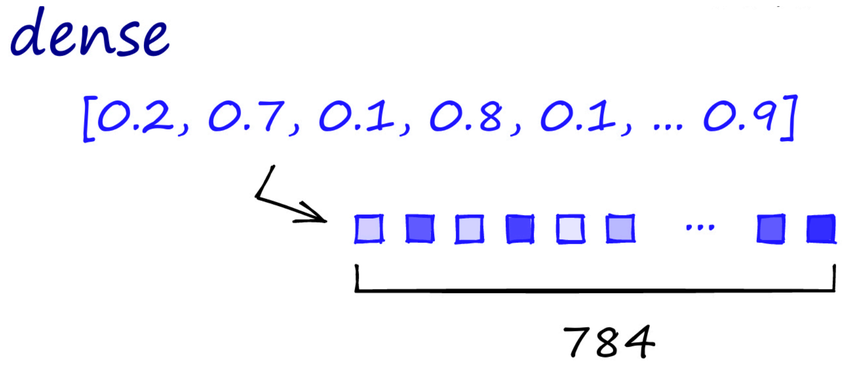


Image Credit: Pinecone

Next Class

More complex word embeddings, inspired by neural language models

- Word2vec (skip-gram, CBOW)
- ~~GloVe~~