CIS 530: Vector Semantics

JURAFSKY AND MARTIN CHAPTER 6

Reminders

Quiz 2 on n-gram LMs is due tonight before 11:59pm.

Homework 3 is due on Wednesday

Read Textbook Chapters 3 and 6

Word Meaning

How should we **represent** the **meaning** of a word?

In N-gram LMs we represented words as a string of letters or as an index in a vocabulary list.

Ideally, we want a meaning representation to encode:

1. **Synonyms** – words that have similar meanings
2. **Antonyms** – words that have opposite meanings
3. **Connotations** – words that are positive or negative
4. **Semantic Roles** – *buy, sell*, and *pay* are different parts of the same underlying *purchasing* event
5. Support for **inference**

Dictionary Definitions

Noun

1. A small insect.
2. A harmful microorganism, as a bacterium or virus.
3. An enthusiastic, almost obsessive, interest in something.
   ‘they caught the sailing bug’
4. A miniature microphone, typically concealed in a room or telephone, used for surveillance.
5. An error in a computer program or system.

Verb

1. Conceal a miniature microphone in (a room or telephone) in order to monitor or record someone's conversations.
2. Annoy or bother (someone)

Polysemy

A lemma that has multiple meanings is called **polysemous**. We call each of these aspects of the meaning of *bug* a **word sense**.

Polysemy can make interpretation difficult.

What if someone types “caught a bug” into Google?

Word sense disambiguation is the task of determining which sense of a word is being used in a context.

Synonymy

When one word has a sense whose meaning is nearly identical to a sense of another word then those two words are **synonyms**.

glitch/error

microbe/bacterium

insect/pest

microphone/wire

Formally, two words are synonymous if they are **substitutable** one for the other in **any sentence** without changing the **truth conditions** of the sentence.

In logic, that means the two words carry the same **propositional meaning**.

Principle of Contrast

Linguists assume that **a difference in form** is always associated with a **difference in meaning**.

While substitutions like *water/H_2O* or *father/dad* are truth preserving, the words are still not identical in meaning.

H_2O is used in scientific contexts, but not general texts like hiking guides

Father is a more formal version of *dad.*

It is possible that no two words have **absolutely identical** meaning.

Word similarity

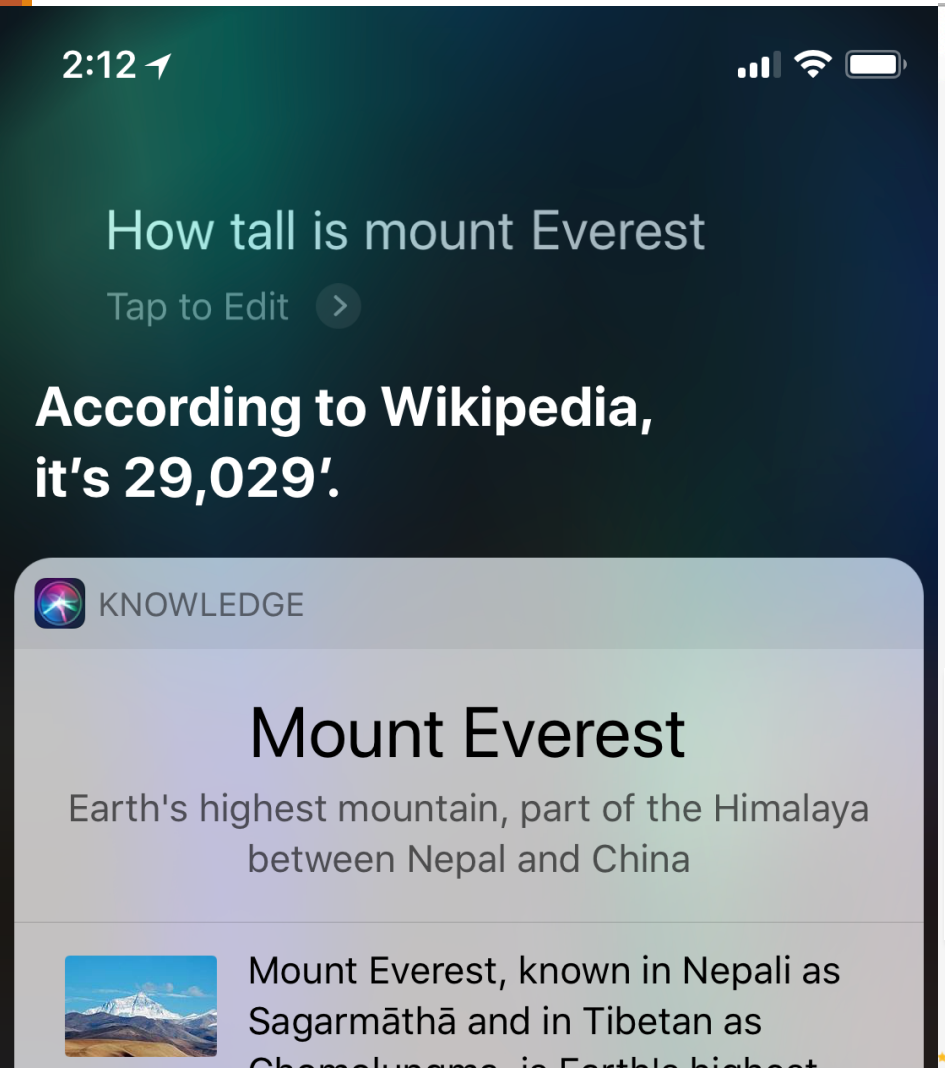
Most words don't have many **synonyms**, but they do have a lot of **similar** words. *Cat* is not a synonym of *dog*, but *cats* and *dogs* are certainly similar words.

"**fast**" is similar to "**rapid**"

"**tall**" is similar to "**height**"

Useful for applications like question answering

Word similarity



Word similarity

Can similar words be substituted in any sentence without changing its truth conditions? No.

How can we measure whether words are similar? One way is to ask humans to judge how similar one word is to another.

Word 1	Word 2	Similarity Score
Vanish	Disappear	9.8
Tiger	Cat	7.4
Love	Sex	6.8
Muscle	Bone	3.6
Cucumber	Professor	0.3

Word Relatedness

Words can still be **related** in ways other than being similar to each other.

Coffee and *Cup* are **not similar** because they don't share any features

1. *coffee* is a plant or a beverage,
2. *cup* is a manufactured object made in a useful shape

But they're **related** by co-participating in the same **event.**

Relatedness is measured with **word association** tests in psychology.

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

Hospitals: *surgeon, scalpel, nurse, anesthetic, hospital*
Restaurants: *waiter, menu, plate, food, chef*
Houses: *family, door, roof, kitchen, bed*

Semantic Roles

An **event** like a commercial transaction described with different **verbs**

1. *buy* (the event from the perspective of the buyer),
2. *sell* (from the perspective of the seller),
3. *pay* (focusing on the monetary aspect),

Or with nouns like *buyer*.

Frames encode **semantic roles** (like *buyer, seller, goods, money*), and the words in a sentence that take on these roles.

Connotation

Words have **affective meanings** or **connotations.** Three important dimensions of affective meaning.

1. *Valence* – the pleasantness of the stimulus
2. *Arousal* – the intensity of emotion provoked by the stimulus
3. *Dominance* – the degree of control exerted by the stimulus

	Valence	Arousal	Dominance
courageous	8.05	5.5	7.38
music	7.67	5.57	6.5
heartbreak	2.45	5.65	3.58
cub	6.71	3.95	4.24
life	6.68	5.59	5.89

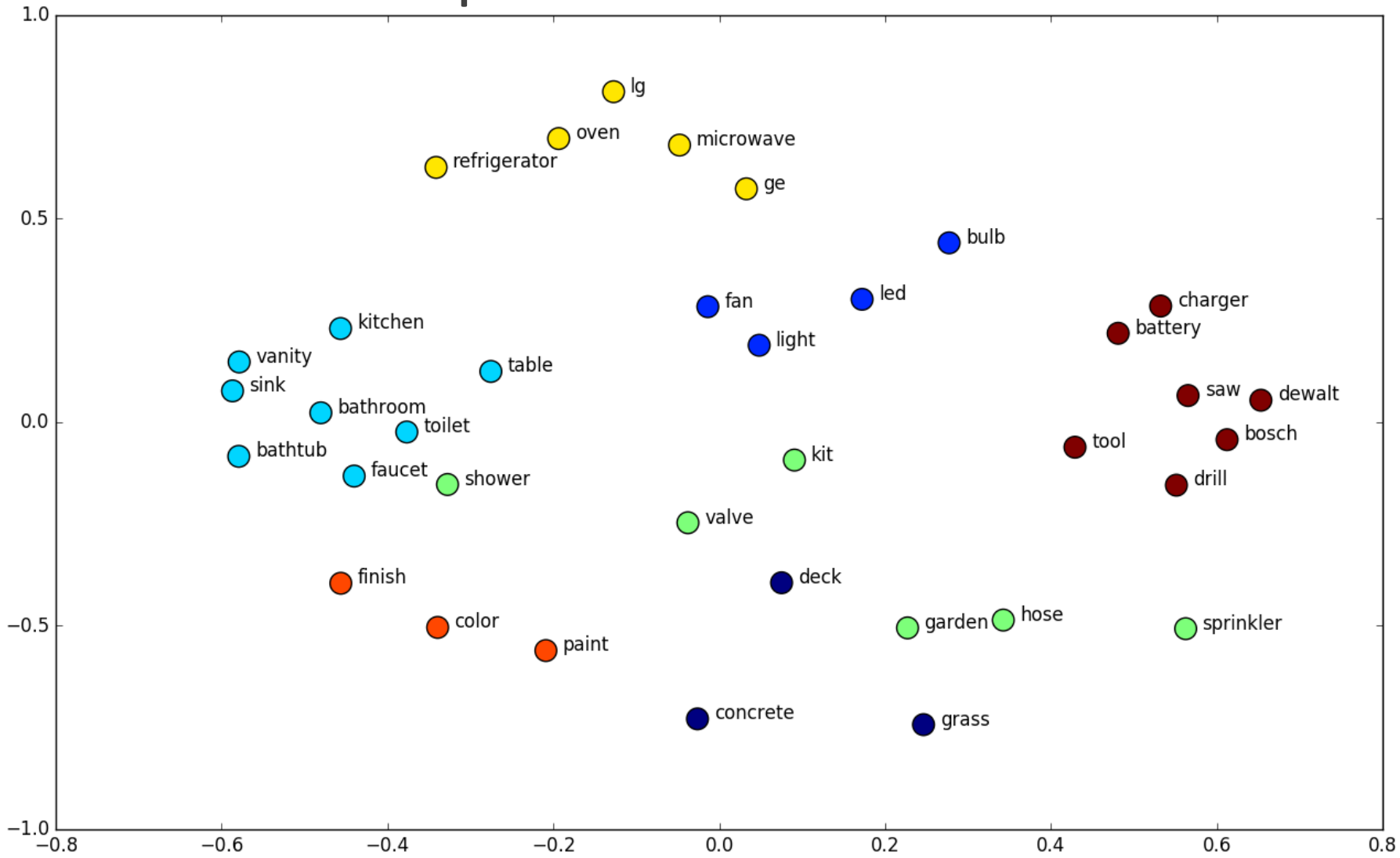
Points in space

Osgood et al. (1957) noticed that in using these 3 numbers to represent the meaning of a word, the model was representing each word as a point in a three-dimensional space

Part of the meaning of *heartbreak* can be represented as a vector with three dimensions corresponded to the word's rating on the three scales.

heartbreak	2.45	5.65	3.58

Vector Space Models



Distributional Hypothesis

If we consider *optometrist* and *eye-doctor* we find that, as our corpus of utterances grows, these two occur in almost the same environments. In contrast, there are many sentence environments in which *optometrist* occurs but *lawyer* does not...

It is a question of the relative frequency of such environments, and of what we will obtain if we ask an informant to substitute any word he wishes for optometrist (not asking what words have the same meaning).

These and similar tests all measure the probability of particular environments occurring with particular elements... If A and B have almost identical environments we say that they are synonyms.
–Zellig Harris (1954)

Intuition of distributional word similarity

Nida (1975) example:

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

From context words humans can guess **tesgüino** means
an alcoholic beverage like beer

Intuition for algorithm:
Two words are similar if they have similar word contexts.

Information Retrieval

- Vector Space Models were initially developed in the SMART information retrieval system (Salton, 1971)
- Each document in a collection is represented as point in a space (a vector in a vector space)
- A user's query is a pseudo-document and is represented as a point in the same space as the documents
- Perform IR by retrieving documents whose vectors are close together in this space to the query vector

Term-Document Matrix

	D1	D2	D3	D4	D5
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					

Term-Document Matrix

	D1	D2	D3	D4	D5
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					

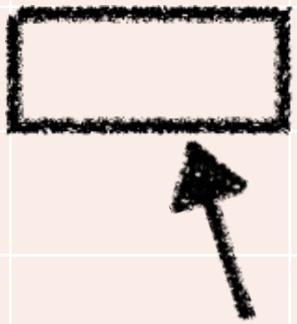
Each column vector represents a Document

Term-Document Matrix

	D1	D2	D3	D4	D5
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					

Term-Document Matrix

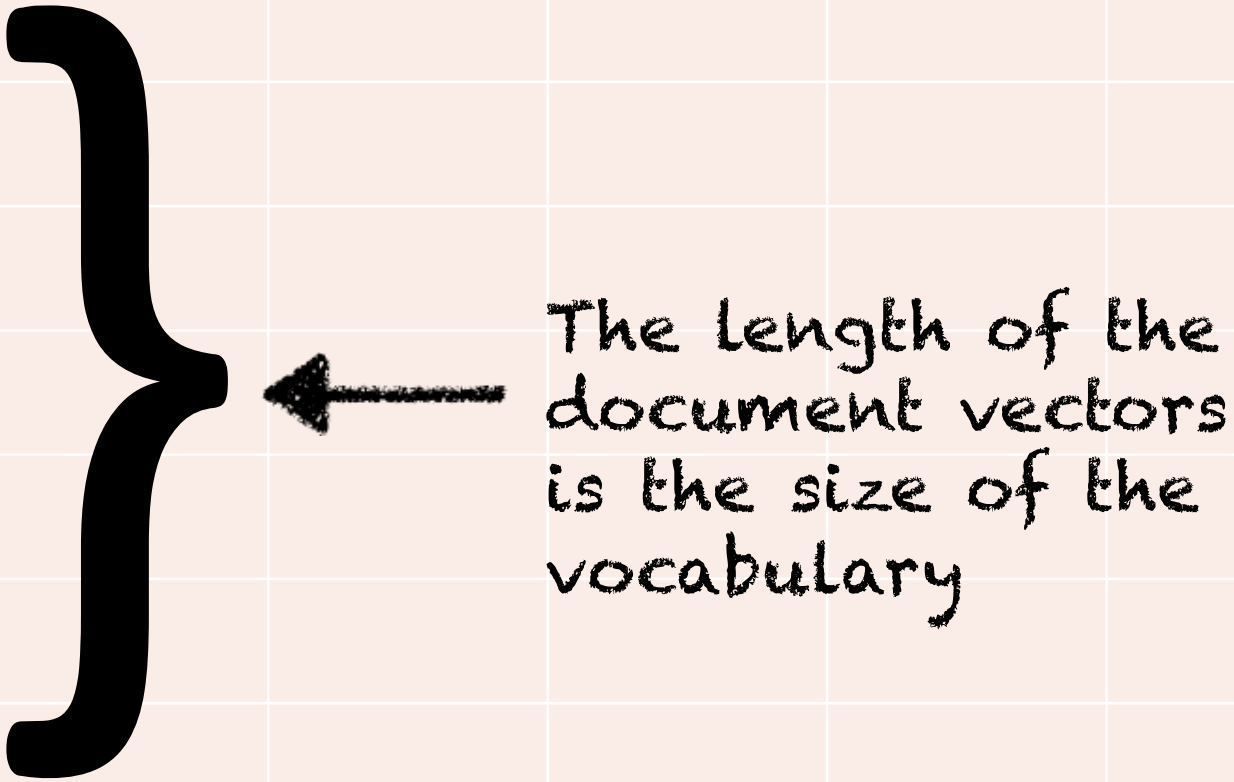
	D1	D2	D3	D4	D5
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					



The value in a cell is based on how often that term occurred in that document

Term-Document Matrix

	D1	D2	D3	D4	D5
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					

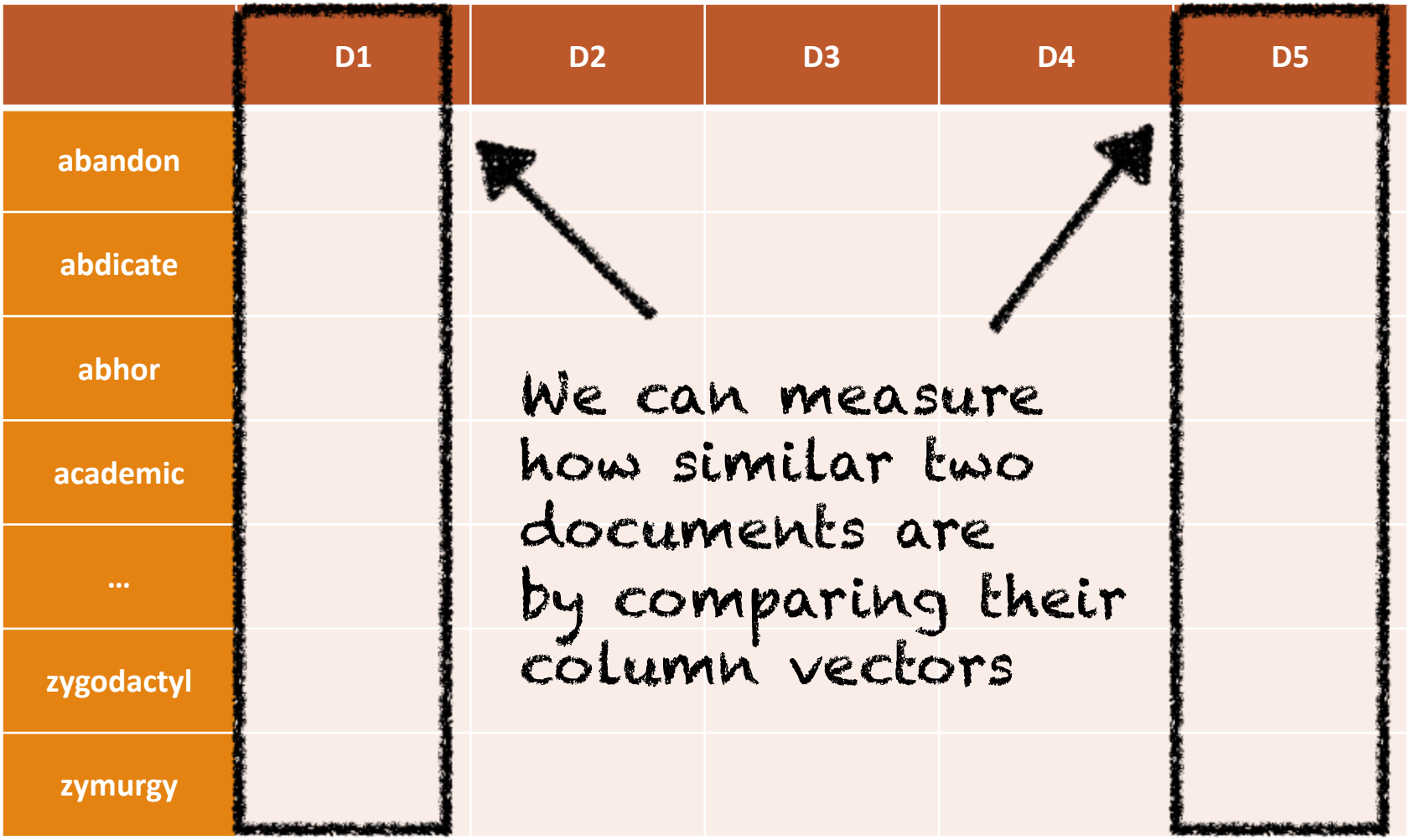


Term-Document Matrix

	D1	D2	D3	D4	D5
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					

Document vectors can be sparse (most values are 0)

Term-Document Matrix



What can document similarity let you do?

Word similarity for plagiarism detection

MAINFRAMES

Mainframes are primarily referred to large computers with rapid, advanced processing capabilities that can execute and perform tasks equivalent to many Personal Computers (PCs) machines networked together. It is characterized with high quantity Random Access Memory (RAM), very large secondary storage devices, and high-speed processors to cater for the needs of the computers under its service.

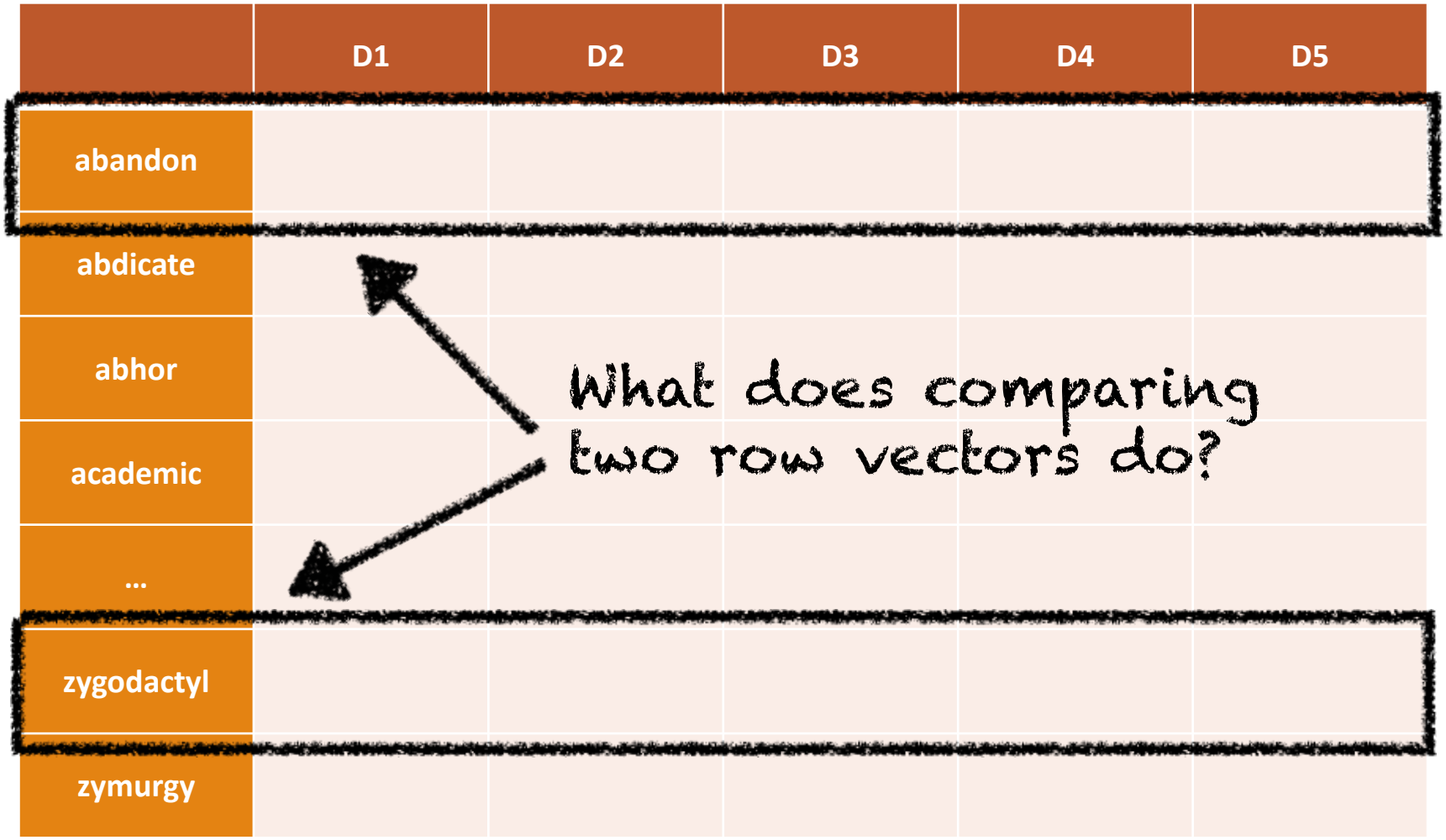
Consisting of advanced components, mainframes have the capability of running multiple large applications required by many and most enterprises and organizations. This is one of its advantages. Mainframes are also suitable to cater for those applications (programs) or files that are of very high demand by its users (clients)

MAINFRAMES

Mainframes usually are referred those computers with fast, advanced processing capabilities that could perform by itself tasks that may require a lot of Personal Computers (PC) Machines. Usually mainframes would have lots of RAMs, very large secondary storage devices, and very fast processors to cater for the needs of those computers under its service.

Due to the advanced components mainframes have, these computers have the capability of running multiple large applications required by most enterprises, which is one of its advantage. Mainframes are also suitable to cater for those applications or files that are of very large demand by its users (clients). Examples of

Term-Document Matrix



Vector comparisons

	doc_X	doc_Y
A	2	4
B	10	15
C	14	10

Vector comparisons

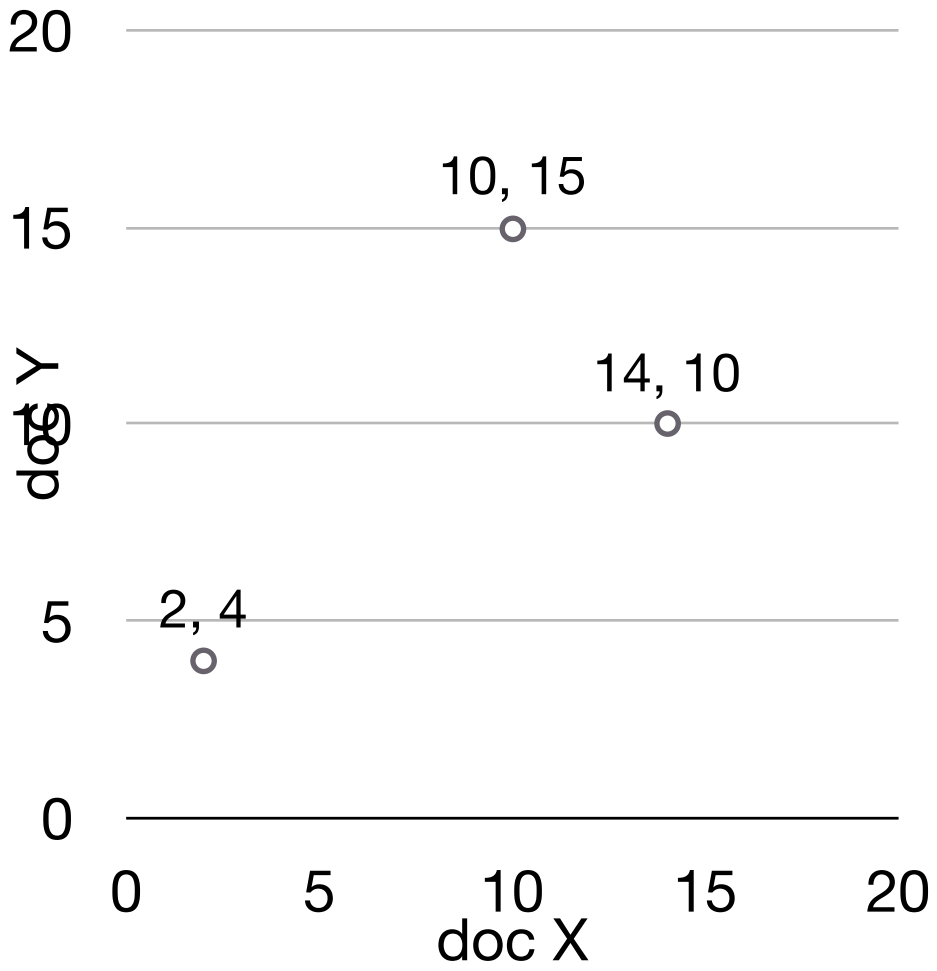
	doc_X	doc_Y
A	2	4
B	10	15
C	14	10

doc_Y is a positive movie review
doc_X is a less positive movie review

A = "superb" positive / low frequency
B = "good" positive / high frequency
C = "disappointing" negative / high frequency

Vector comparisons

	doc_X	doc_Y
A	2	4
B	10	15
C	14	10

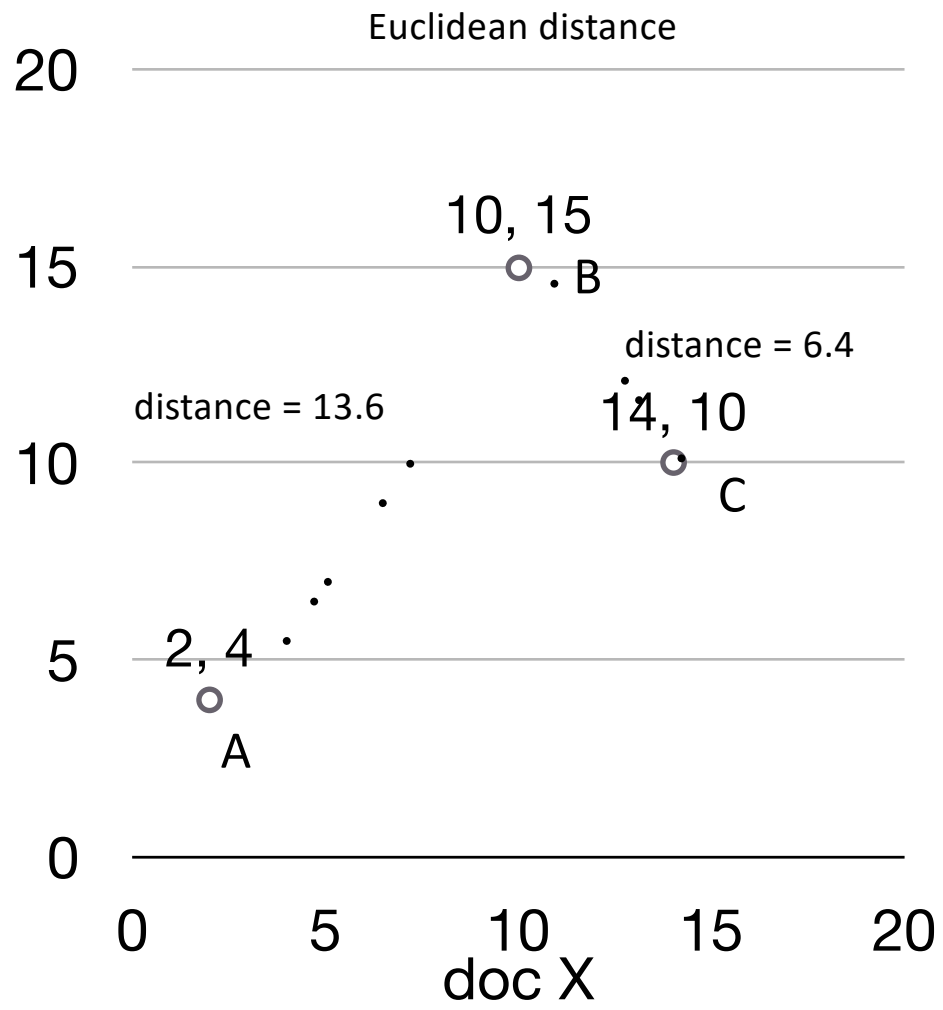


Vector comparisons

	doc_X	doc_Y
A	2	4
B	10	15
C	14	10

Euclidean distance : vectors u, v of dimension N

$$\sqrt{\sum_{i=1}^{N} |u_i - v_i|^2}$$



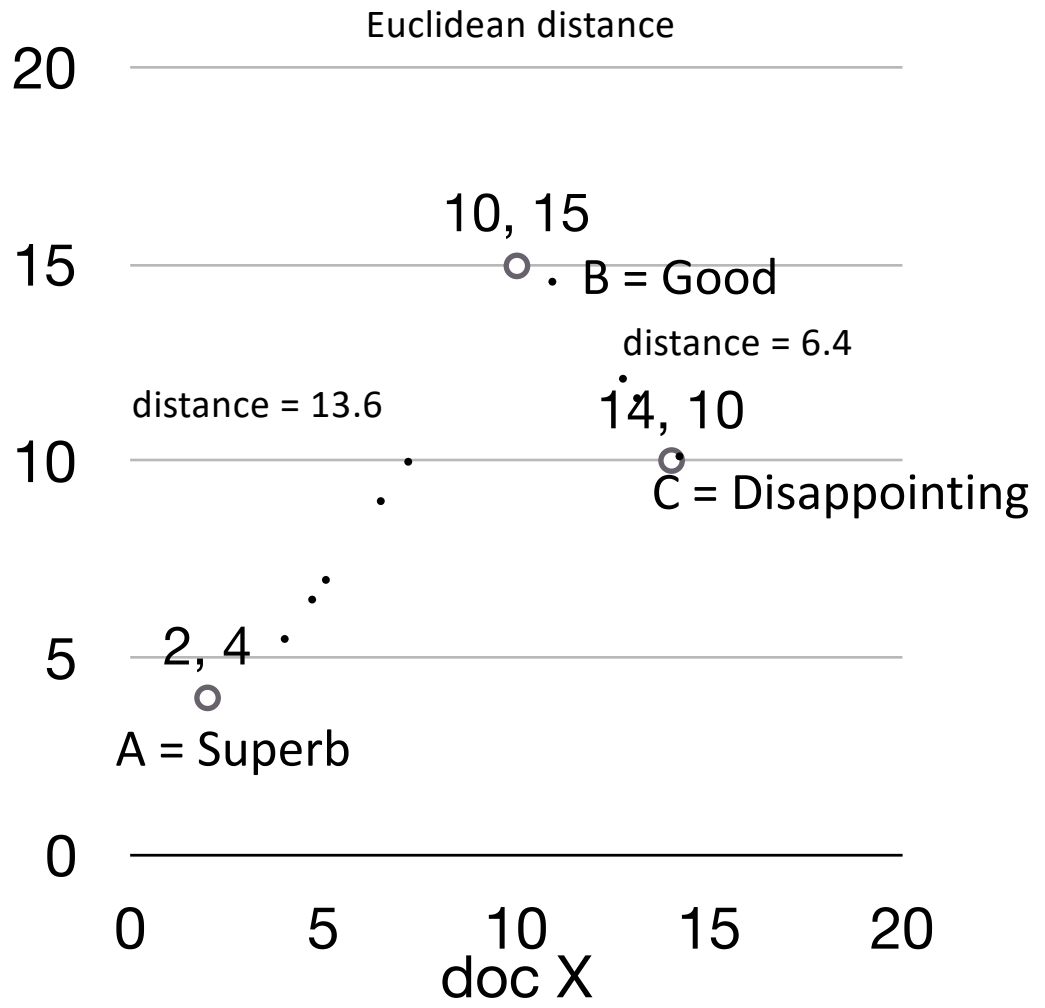
Vector comparisons

Oh no! Good is closer to Disappointing than to Superb.

	docX	docY
A	2	4
B	10	15
C	14	10

Euclidean distance : vectors u, v of dimension N

$$\sqrt{\sum_{i=1}^{N} |u_i - v_i|^2}$$



Vector L2 (length) Normalization

	doc_X	doc_Y	$\|\|u\|\|$
A	2	4	4.47
B	10	15	18.02
C	14	10	17.20

$$\|u\| = \sqrt{\sum_{i=1}^{n} u_i^2}$$

Vector L2 (length) Normalization

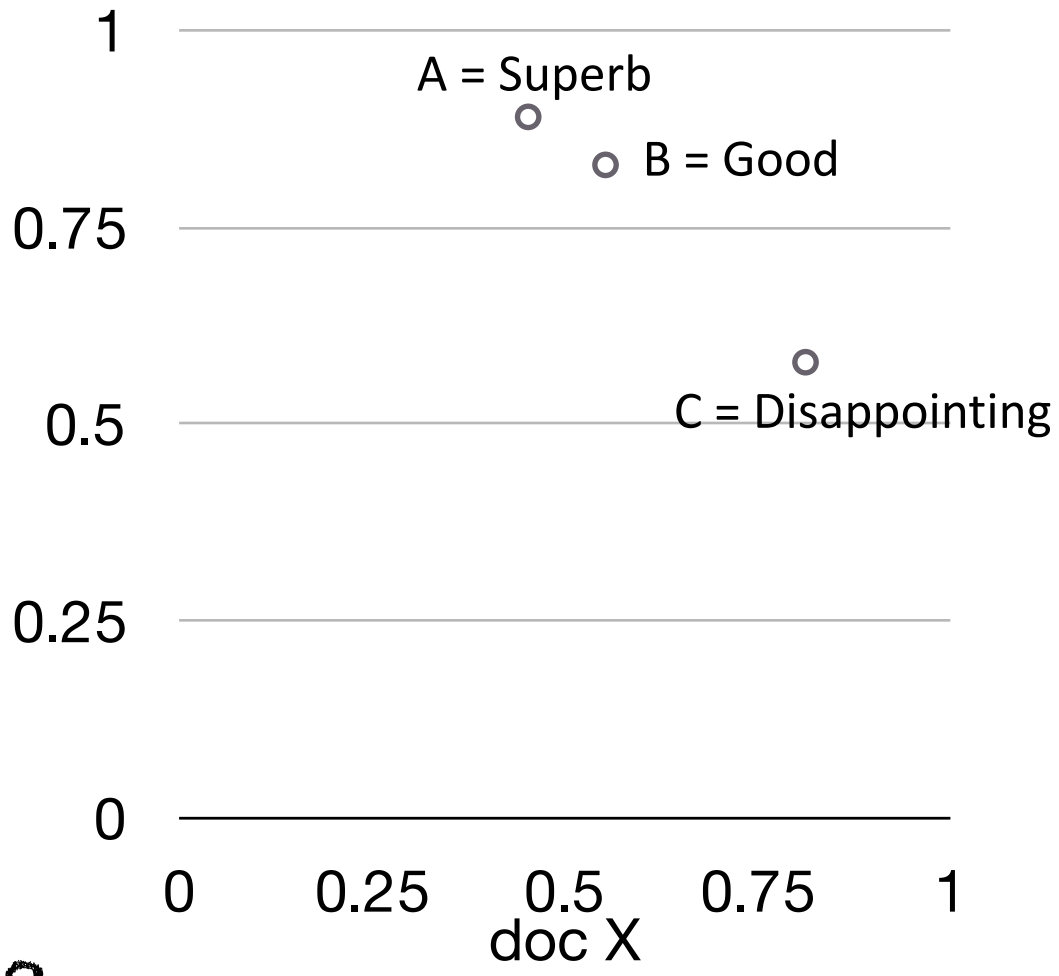
	doc_X	doc_Y	||u||
A	2/4.47	4/4.47	4.47
B	10/18.02	15/18.02	18.02
C	14/17.2	10/17.2	17.20

$$\|u\| = \sqrt{\sum_{i=1}^{n} u_i^2}$$

Divide each vector by its L2 length

Vector L2 (length) Normalization

	doc_X	doc_Y
Ȧ	0.45	0.89
Ḃ	0.55	0.83
Ċ	0.81	0.58

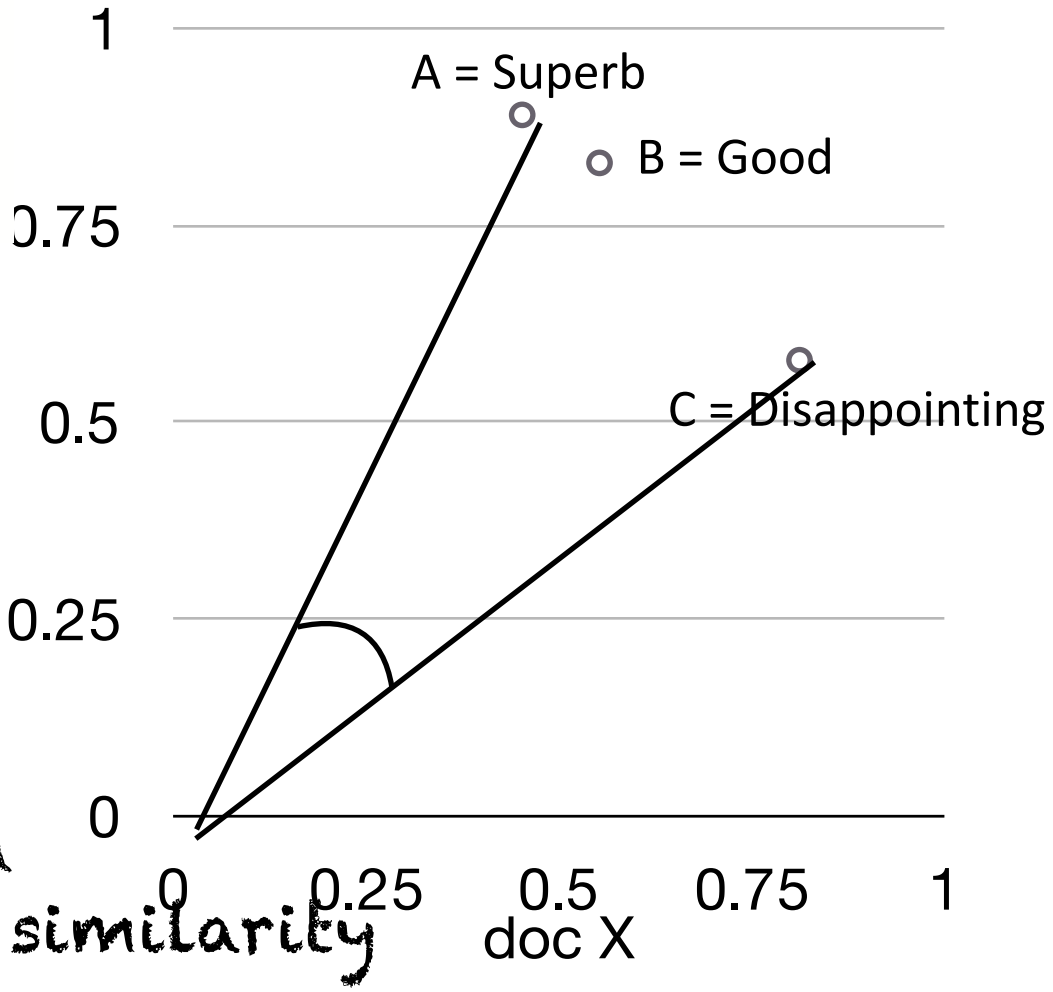


Now Good is closer to Superb than to Disappointing

Cosine Distance

$$1 - \frac{\sum_{i=1}^{n} u_i \times v_i}{\sqrt{\sum_{i=1}^{n} u_i^2} \times \sqrt{\sum_{i=1}^{n} v_i^2}}$$

Cosine does the L2 normalization too



Cosine angle between vectors tells us their similarity

Term-Term Matrix

	abandon	abdicate	abhor	…	zymurgy
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					

Term-Term Matrix

AKA
Term-Context
Matrix

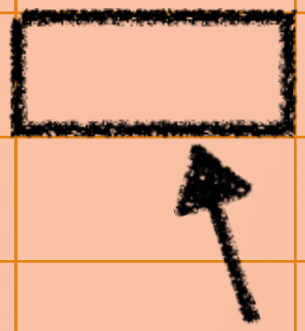
	abandon	abdicate	abhor	…	zymurgy
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					

Length of the vector is now |V|
instead of number of documents

Term-Term Matrix

AKA
Term-Context
Matrix

back	abandon	abdicate	abhor	…	zymurgy
abandon					
abdicate					
abhor					
academic					
…					
zygodactyl					
zymurgy					



The value in a cell indicates how often abandon appears in a context window surrounding abdicate

Context windows

w-2, w-1 **target_word** w+1 w+2

The government most not **abdicate** responsibility to non-elected
it has led men to **abdicate** their family responsibilities
other demands, but declining to **abdicate** his responsibility
leaders **abdicate** their role and present people with no plans

	his	leaders	not	responsibility	to
abandon	1	1	1	2	3

Context windows

Occur in a window of +/- 2 words, in the same sentence, in the same document

Instead of window of words use more complex contexts: dependency patters. Subj-of-verb, adj-mod, obj-of-verb

Languages have long distance dependencies

*The **pictures are** beautiful.*

*The **pictures** of the old man **are** beautiful.*

*The **pictures** of the old man holding his grandchild **are** beautiful.*


nsubj
acomp
The pictures
NOUN
are
VERB
beautiful
ADJ
nsubj
prep
pobj
acomp
The pictures
NOUN
of
ADP
the old man
NOUN
are
VERB
beautiful
ADJ
nsubj
prep
pobj
acl
dobj
acomp
The pictures
NOUN
of
ADP
the old man
NOUN
holding
VERB
his grandchild
NOUN
are
VERB
beautiful
ADJ

Using syntax to define a word's context

Zellig Harris (1968) "The meaning of entities, and the meaning of grammatical relations among them, is related to the restriction of combinations of these entities relative to other entities"

Duty and **Responsibility** have similar syntactic distributions

Modified by adjectives	additional, administrative, assumed, collective, congressional, constitutional …
Object of verbs	assert, assign, assume, attend to, avoid, become, breach..

Alternates to counts

Raw word frequency is not a great measure of association between words. It's very skewed "the" and "of" are very frequent, but maybe not the most discriminative

We'd rather have a measure that asks whether a context word is particularly informative about the target word.

Instead of raw counts, it's common to transform vectors using TF-IDF or PPMI

TF-IDF

*Term frequency * inverse document frequency*

How often a word occurred in a document

1 over the number of documents that it occurred in

Sparse v. Dense Vectors

Co-occurrence matrix (weighted by TF-IDF or mutual information)

- **Long** (length |V| = 50,000+)
- **Sparse** (most elements are zeros)

Alternative: learn vectors that are

- **Short** (length 200-1000)
- **Dense** (most elements are non-zero)

How do we get dense vectors?

One recipe: train a classifier!

1. Treat the target word and a neighboring context word as positive examples.

2. Randomly sample other words in the lexicon to get negative samples.

3. Use logistic regression (similar to Perceptron, but output values range between 0-1) to train a classifier to distinguish those two cases.

4. Use the **weights** as the **embeddings**.

Skip-grams, CBOW Mikolov et al. 2013

Learn embeddings as part of the process of word prediction.

Train a classifier to predict neighboring words

Inspired by neural net language models.

In so doing, learn dense embeddings for the words in the training corpus.

Advantages:

Fast, easy to train (much faster than SVD)

Available online in the word2vec package

Including sets of pretrained embeddings!

Skip-Grams

Predict each neighboring word in a context window of 2C of surrounding words

So for C=2, we are given a word w_t and we try to predict its 4 surrounding words

$$[w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}]$$

Uses "negative sampling" for training

Negative sampling

lemon, a [tablespoon of apricot preserves or] jam

c1 c2 w c3 c4

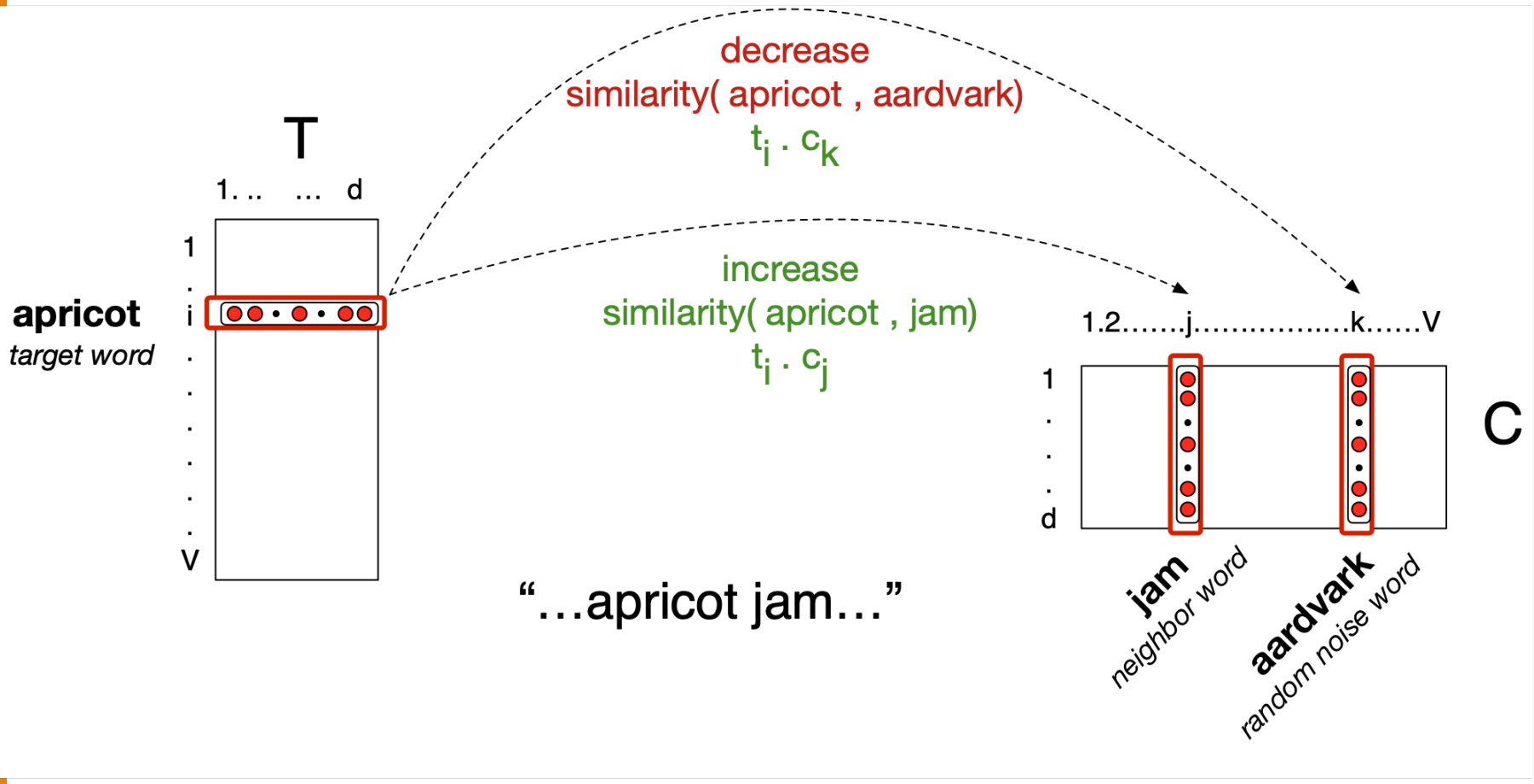
We want predictions
of these words to be high

And these words to be low

[cement metaphysical dear coaxial apricot attendant whence forever puddle]

n1 n2 n3 n4 n5 n6 n7 n8

Neural Network



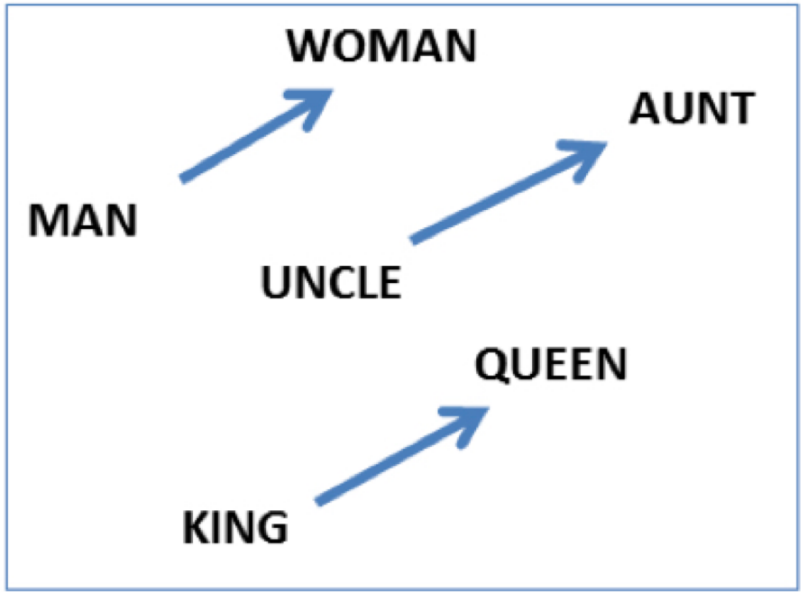
Properties of Embeddings

Nearest Neighbors are surprisingly good

Redmond	Havel	ninjutsu	graffiti	capitulate
Redmond Wash.	Vaclav Havel	ninja	spray paint	capitulation
Redmond Washington	president Vaclav Havel	martial arts	grafitti	capitulated
Microsoft	Velvet Revolution	swordsmanship	taggers	capitulating

Embeddings capture relational meanings

.vector('king') - vector('man') + vector('queen') ≅ vector('woman')



Magnitude: A Fast, Efficient Universal Vector Embedding Utility Package

Ajay Patel
Plasticity Inc.
San Francisco, CA
ajay@plasticity.ai

Alexander Sands
Plasticity Inc.
San Francisco, CA
alex@plasticity.ai

Chris Callison-Burch
Computer and Information
Science Department
University of Pennsylvania
ccb@upenn.edu

Marianna Apidianaki
LIMSI, CNRS
Université Paris-Saclay
91403 Orsay, France
marapi@seas.upenn.edu

Abstract

Vector space embedding models like word2vec, GloVe, and fastText are extremely popular representations in natural language processing (NLP) applications. We present Magnitude, a fast, lightweight tool for utilizing and processing embeddings. Magnitude is an open source Python package with a compact vector storage file format that allows for efficient manipulation of huge numbers of embeddings. Magnitude performs common operations up to 60 to 6,000 times faster than Gensim. Magnitude introduces several

Metric	Cold	Warm
Initial load time	97x	–
Single key query	1x	110x
Multiple key query (n=25)	68x	3x
k-NN search query (k=10)	1x	5,935x

Table 1: Speed comparison of Magnitude versus Gensim for common operations. The 'cold' column represents the first time the operation is called. The 'warm' column indicates a subsequent call with the same keys.

file, a 97x speed-up. Gensim uses 5GB of RAM versus 18KB for Magnitude.

Demo of word vectors

```
# Install Magnitude
pip3 install pymagnitude

# Download Google's word2vec vectors
wget http://magnitude.plasticity.ai/word2
# Warning it's 11GB large

# Start Python, and try the commands
# on the next slide
python3
```

Demo of word vectors

```
from pymagnitude import *
vectors = Magnitude("GoogleNews-vectors-ne

queen = vectors.query('queen')
king = vectors.query("king")
vectors.similarity(king, queen)
# 0.6510958

vectors.most_similar_approx(king, topn=5)
#[('king', 1.0), ('kings', 0.72), ('prince
```

Many possible models

Term-document
Term-context
Pattern-pair

word2vec
GloVe
PCA
LDA
LSA

length norm.
TF-IDF
PPMI
probabilities

cosine
Manhattan
Jaccard
KL divergence
JS distance
DICE

Many possible models

Term-document
Term-context
Pattern-pair

word2vec
GloVe
PCA
LDA
LSA

length norm.
TF-IDF
PPMI
probabilities

cosine
Manhattan
Jaccard
KL divergence
JS distance
DICE

How many dimensions?

What modifications should we make to the input?

How do we pick the right combination?

Evaluating word vectors

2 kinds of evaluation:

1) Extrinsic evaluation = task based

2) Intrinsic

Psycholinguistics Data

…	Love	Sex	6.8
	Tiger	Cat	7.3
	Tiger	Tiger	10
	Fertility	Egg	6.7
	Stock	Egg	1.8
	Professor	Cucumber	0.3

Computing correlation

Human ordering

Love	Sex	>	Professor	Cucumber

System ordering predicts

< … this is a discordant pair

> … this is a concordant pair

Does similarity == semantics?

Word vectors fail to capture logical implications

Dog(x) → Animal(x)
Dog(x) → not (Gorilla(x))

Word vectors for antonyms or logically exclusive things are often very similar
sim(boys, girls)
sim(cats, dogs)
sim(France, Germany)
sim(rise, fall)

Acknowledgements

Thanks to the following people for providing slides

Dan Jurafsky from his textbook Speech and Language Processing version 3

Peter Turney and Patrick Pantel From Frequency to Meaning: Vector Space Models of Semantics

Chris Potts from his course CS224U: Natural Language Understanding

Denis Paperno, Marco Baroni, German Kruszewski from their talk on Deriving Boolean Structures from Distributional Vectors

Vector Semantics

READ CHAPTER 6 OF

SPEECH AND LANGUAGE PROCESSING (3RD ED DRAFT)

Why vector models of meaning? Computing the similarity between words

"**fast**" is similar to "**rapid**"

"**tall**" is similar to "**height**"

Question answering:

*Q: "How **tall** is Mt. Everest?"*
*Candidate A: "The official **height** of Mount Everest is 29029 feet"*

Word similarity for plagiarism detection

MAINFRAMES

Mainframes are primarily referred to large computers with rapid, advanced processing capabilities that can execute and perform tasks equivalent to many Personal Computers (PCs) machines networked together. It is characterized with high quantity Random Access Memory (RAM), very large secondary storage devices, and high-speed processors to cater for the needs of the computers under its service.

Consisting of advanced components, mainframes have the capability of running multiple large applications required by many and most enterprises and organizations. This is one of its advantages. Mainframes are also suitable to cater for those applications (programs) or files that are of very high demand by its users (clients). Examples of such organizations and enterprises using mainframes are online shopping websites such as Ebay, Amazon, and computing-giant

MAINFRAMES

Mainframes usually are referred those computers with fast, advanced processing capabilities that could perform by itself tasks that may require a lot of Personal Computers (PC) Machines. Usually mainframes would have lots of RAMs, very large secondary storage devices, and very fast processors to cater for the needs of those computers under its service.

Due to the advanced components mainframes have, these computers have the capability of running multiple large applications required by most enterprises, which is one of its advantage. Mainframes are also suitable to cater for those applications or files that are of very large demand by its users (clients). Examples of these include the large online shopping websites -i.e. : Ebay, Amazon, Microsoft, etc.

Intuition of distributional word similarity

Nida example:

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

From context words humans can guess ***tesgüino*** means

- an alcoholic beverage like **beer**

Intuition for algorithm:

- Two words are similar if they have similar word contexts.

Intuition

Model the meaning of a word by “embedding” in a vector space.

The meaning of a word is a vector of numbers

- Vector models are also called “**embeddings**”.

Contrast: word meaning is represented in many computational linguistic applications by a vocabulary index (“word number 545”)

Old philosophy joke:

Q: What’s the meaning of life?

A: LIFE’

Term-document matrix

Term-document matrix

Each cell: count of term *t* in a document *d*: $tf_{t,d}$:

- Each document is a count vector in $\mathbb{N}^v$: a column below

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

Comparing the similarity of documents

Comparing the similarity of documents

Author attribution / plagiarism detection / document de-duplication

Clustering documents into categories

Recommendation systems

Term-document matrix

Two documents are similar if their vectors are similar

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

The words in a term-document matrix

Each word is a count vector in $\mathbb{N}^D$: a row below

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

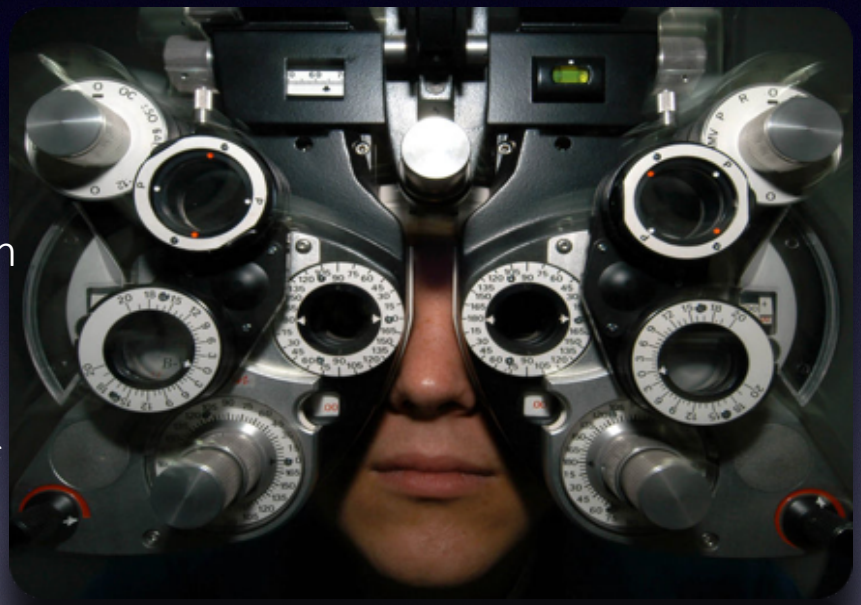
The words in a term-document matrix

Two **words** are similar if their vectors are similar

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

Distributional Hypothesis

If we consider **optometrist** and **eye-doctor** we find that, as our corpus of utterances grows, these two occur in almost the same environments. In contrast, there are many sentence environments in which **optometrist** occurs but **lawyer** does not...



It is a question of the relative frequency of such environments, and of what we will obtain if we ask an informant to substitute any word he wishes for **optometrist** (not asking what words have the same meaning).

These and similar tests all measure the probability of particular environments occurring with particular elements... If A and B have almost identical environments we say that they are synonyms.

–Zellig Harris (1954)

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

–John Firth (1957)

Word-word matrix

Turney and Pantel (2010) *From Frequency to Meaning:* If units of text have similar vectors in a text-frequency matrix then they tend to have similar meanings.

Defining a co-occurrence matrix

DIRT

Lin and Panel (2001) operationalize the Distributional Hypothesis using dependency relationships to define similar environments.

Duty and **responsibility** share a similar set of dependency contexts in large volumes of text:

modified by adjectives	objects of verbs
additional, administrative, assigned, assumed, collective, congressional, constitutional ...	assert, assign, assume, attend to, avoid, become, breach ...

Define a co-occurrence matrix

sugar, a sliced lemon, a tablespoonful of **apricot** preserve or jam, a pinch each of,
their enjoyment. Cautiously she sampled her first **pineapple** and another fruit whose taste she likened
well suited to programming on the digital **computer**. In finding the optimal R-stage policy from
for the purpose of gathering data and **information** necessary for the study authorized in the

	aardvark	computer	data	pinch	result	sugar	…
apricot	0	0	0	1	0	1	
pineapple	0	0	0	1	0	1	
digital	0	2	1	0	1	0	
information	0	1	6	0	4	0	
…	…						

Term-context matrix for word similarity

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

	aardvark	computer	data	pinch	result	sugar	…
apricot	0	0	0	1	0	1	
pineapple	0	0	0	1	0	1	
digital	0	2	1	0	1	0	
information	0	1	6	0	4	0	

The word-word or word-context matrix

Instead of entire documents, use smaller contexts

- Paragraph
- Window of $\pm$ 4 words

A word is now defined by a vector over counts of context words

Instead of each vector being of length D

Each vector is now of length |V|

The word-word matrix is |V|x|V|

Word-word matrix

We showed only 4x6, but the real matrix is 50,000 x 50,000

- So it's very **sparse**
  - Most values are 0.
- That's OK, since there are lots of efficient algorithms for sparse matrices.

The size of windows depends on your goals

- The shorter the windows , the more **syntactic** the representation

  ± 1-3 very syntacticy
- The longer the windows, the more **semantic** the representation

  ± 4-10 more semanticy

Related versus similar

Related words

- They are typically nearby each other.
- *wrote* is related to words like *book* or *poem*.

Similar words

- They have similar neighbors.
- *wrote* is similar to words like *said* or *remarked*.

Vector Semantics

MEASURING SIMILARITY: THE COSINE

Measuring similarity

Given 2 target words *v* and *w*

We'll need a way to measure their similarity.

Most measure of vectors similarity are based on the:

Dot product or **inner product** from linear algebra

$$\text{dot-product}(\vec{v},\vec{w}) = \vec{v}\cdot\vec{w} = \sum_{i=1}^{N} v_i w_i = v_1 w_1 + v_2 w_2 + \ldots + v_N w_N$$

- High when two vectors have large values in same dimensions.
- Low (in fact 0) for **orthogonal vectors** with zeros in complementary distribution

Problem with dot product

$$\text{dot-product}(\vec{v},\vec{w}) = \vec{v}\cdot\vec{w} = \sum_{i=1}^{N} v_i w_i = v_1 w_1 + v_2 w_2 + \ldots + v_N w_N$$

Dot product is longer if the vector is longer. Vector length:

$$|\vec{v}| = \sqrt{\sum_{i=1}^{N} v_i^2}$$

Vectors are longer if they have higher values in each dimension

That means more frequent words will have higher dot products

That's bad: we don't want a similarity metric to be sensitive to word frequency

Solution: cosine

Just divide the dot product by the length of the two vectors!

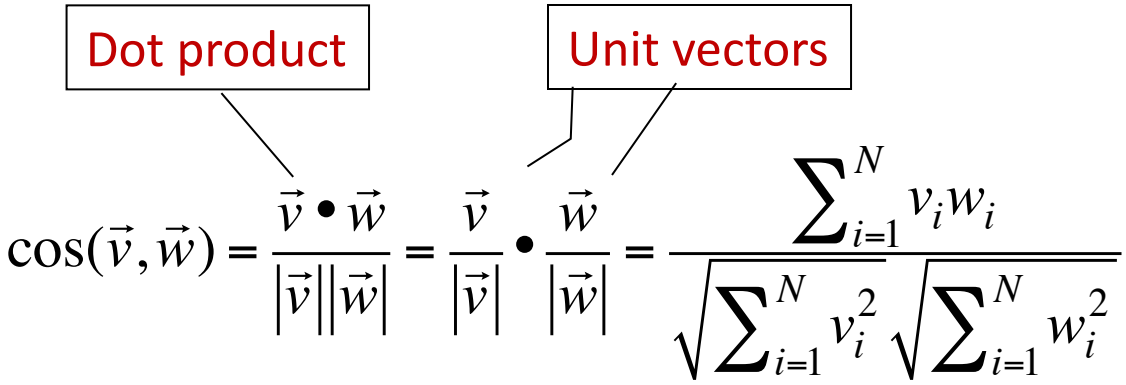
$$\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|}$$

This turns out to be the cosine of the angle between them!

$$\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}| \cos\theta$$

$$\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} = \cos\theta$$

Cosine for computing similarity



v_i is the PPMI value for word *v* in context *i*
w_i is the PPMI value for word *w* in context *i*.

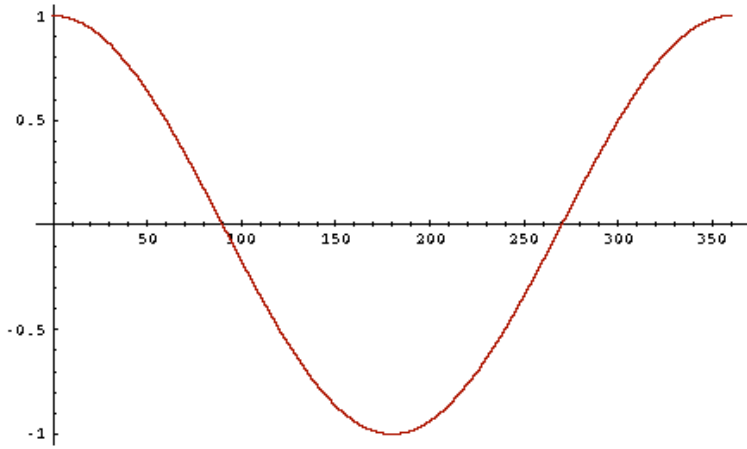
Cos($\vec{v}$,$\vec{w}$) is the cosine similarity of $\vec{v}$ and $\vec{w}$

Cosine as a similarity metric

-1: vectors point in opposite directions

+1: vectors point in same directions

0: vectors are orthogonal



Raw frequency is non-negative, so cosine ranges between 0 and 1

$$\cos(\vec{v},\vec{w}) = \frac{\vec{v}\bullet\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\vec{v}}{|\vec{v}|}\bullet\frac{\vec{w}}{|\vec{w}|} = \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

	large	data	computer
apricot	2	0	0
digital	0	1	2
information	1	6	1

Which pair of words is more similar?

cosine(apricot,information) =

$$\frac{2+0+0}{\sqrt{2+0+0}\ \sqrt{1+36+1}} = \frac{2}{\sqrt{2}\sqrt{38}} = .23$$

cosine(digital,information) =

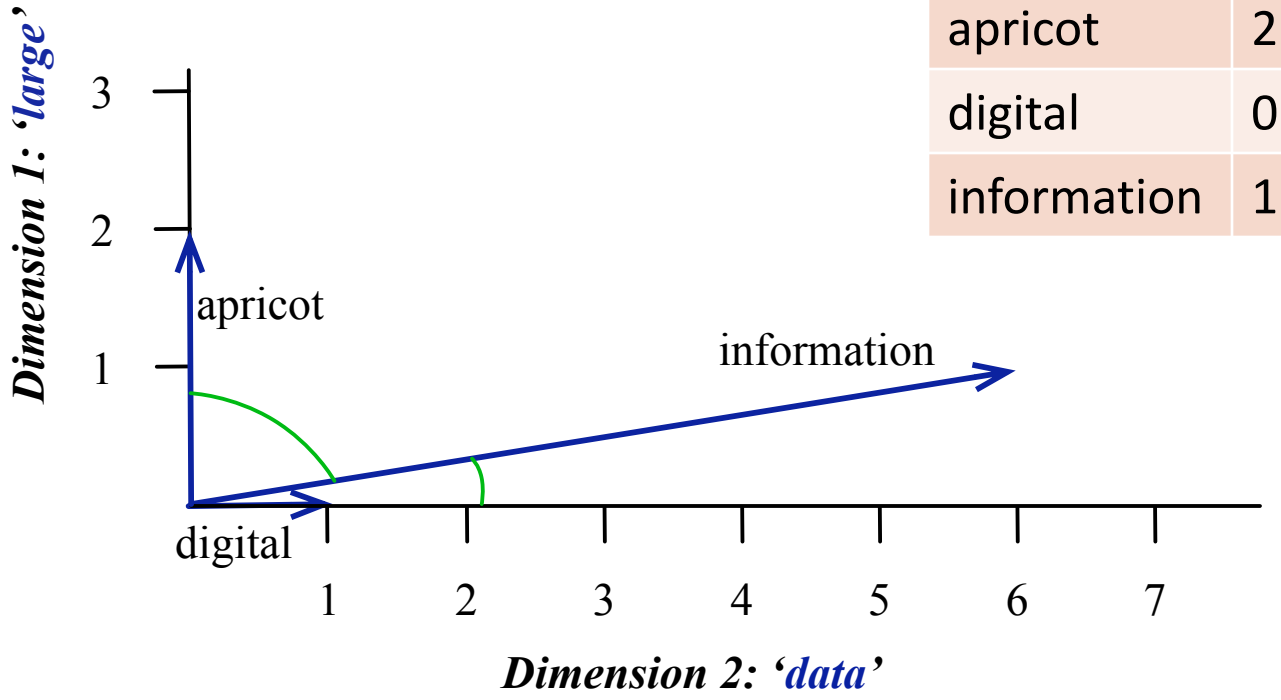
$$\frac{0+6+2}{\sqrt{0+1+4}\ \sqrt{1+36+1}} = \frac{8}{\sqrt{38}\sqrt{5}} = .58$$

cosine(apricot,digital) =

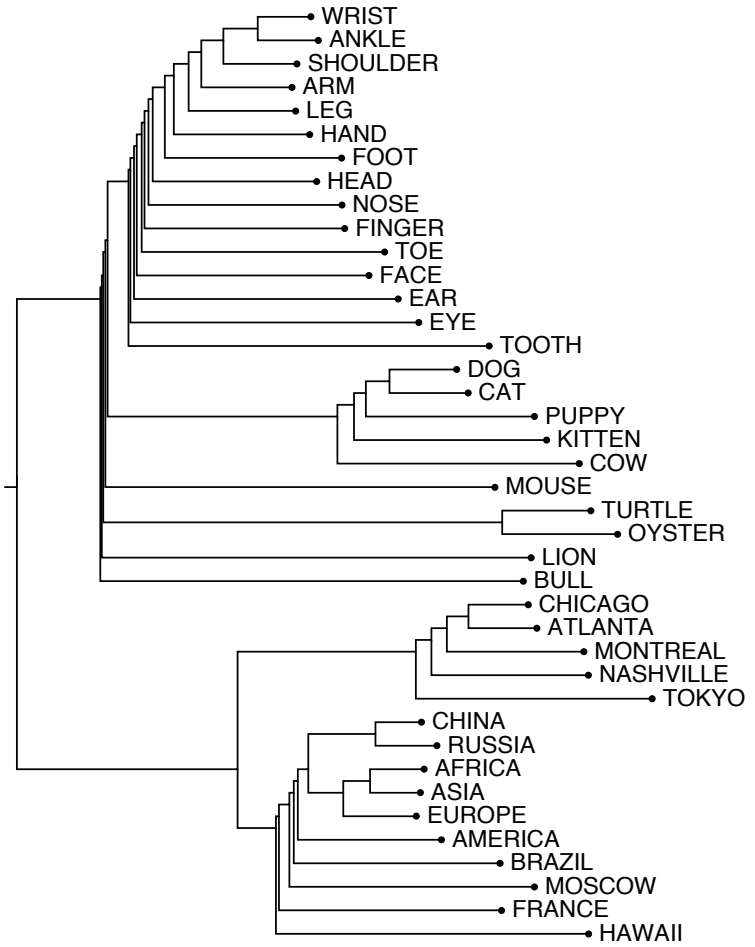
$$\frac{0+0+0}{\sqrt{1+0+0}\ \sqrt{0+1+4}} = 0$$

Visualizing vectors and angles

	large	data
apricot	2	0
digital	0	1
information	1	6



Clustering vectors to visualize similarity in co-occurrence matrices



Rohde et al. (2006)

Other possible similarity measures

$$\text{sim}_{\text{cosine}}(\vec{v},\vec{w}) = \frac{\vec{v}\cdot\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\sum_{i=1}^{N} v_i \times w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

$$\text{sim}_{\text{Jaccard}}(\vec{v},\vec{w}) = \frac{\sum_{i=1}^{N} \min(v_i,w_i)}{\sum_{i=1}^{N} \max(v_i,w_i)}$$

$$\text{sim}_{\text{Dice}}(\vec{v},\vec{w}) = \frac{2\times\sum_{i=1}^{N} \min(v_i,w_i)}{\sum_{i=1}^{N} (v_i+w_i)}$$

$$\text{sim}_{\text{JS}}(\vec{v}||\vec{w}) = D(\vec{v}|\frac{\vec{v}+\vec{w}}{2}) + D(\vec{w}|\frac{\vec{v}+\vec{w}}{2})$$

Vector Semantics

MEASURING SIMILARITY: THE COSINE

Using syntax to define a word's context

Zellig Harris (1968)

> "The meaning of entities, and the meaning of grammatical relations among them, is related to the restriction of combinations of these entities relative to other entities"

Two words are similar if they have similar syntactic contexts

Duty and **responsibility** have similar syntactic distribution:

Modified by adjectives	additional, administrative, assumed, collective, congressional, constitutional …
Objects of verbs	assert, assign, assume, attend to, avoid, become, breach..

Co-occurrence vectors based on syntactic dependencies

Dekang Lin, 1998 “Automatic Retrieval and Clustering of Similar Words”

Each dimension: a context word in one of R grammatical relations

- Subject-of- “absorb”

Instead of a vector of $|V|$ features, a vector of $R|V|$

Example: counts for the word *cell* :

	subj-of, absorb	subj-of, adapt	subj-of, behave	...	pobj-of, inside	pobj-of, into	...	nmod-of, abnormality	nmod-of, anemia	nmod-of, architecture	...	obj-of, attack	obj-of, call	obj-of, come from	obj-of, decorate	...	nmod, bacteria	nmod, body	nmod, bone marrow
cell	1	1	1		16	30		3	8	1		6	11	3	2		3	2	2

Syntactic dependencies for dimensions

Alternative (Padó and Lapata 2007):

- Instead of having a |V| x R|V| matrix
- Have a |V| x |V| matrix
- But the co-occurrence counts aren't just counts of words in a window
- But counts of words that occur in one of R dependencies (subject, object, etc).
- So M("cell","absorb") = count(subj(cell,absorb)) + count(obj(cell,absorb)) + count(pobj(cell,absorb)), etc.

TF-IDF

tf-idf (that's a hyphen not a minus sign)

The combination of two factors

- **Term frequency** (Luhn 1957): frequency of the word (can be logged)
- **Inverse document frequency** (IDF) (Sparck Jones 1972)
  - N is the total number of documents
  - df_i = "document frequency of word *i*"
  - = # of documents with word *I*
- w_{ij} = *word* i *in document* j

$$w_{ij} = tf_{ij}\, idf_i$$

$$\text{idf}_i = \log\left(\frac{N}{df_i}\right)$$

TF-IDF

TF-IDF not generally used for word-word similarity

But is by far the most common weighting when we are considering the relationship of words to documents

Vector Semantics

POSITIVE POINTWISE MUTUAL INFORMATION (PPMI)

Problem with raw counts

Raw word frequency is not a great measure of association between words

- It's very skewed
  - "the" and "of" are very frequent, but maybe not the most discriminative

We'd rather have a measure that asks whether a context word is **particularly informative** about the target word.

- Positive Pointwise Mutual Information (PPMI)

Pointwise Mutual Information

Pointwise mutual information:

Do events x and y co-occur more than if they were independent?

$$\text{PMI}(X,Y) = \log_2 \frac{P(x,y)}{P(x)P(y)}$$

PMI between two words: (Church & Hanks 1989)

Do words x and y co-occur more than if they were independent?

$$\text{PMI}(word_1, word_2) = \log_2 \frac{P(word_1, word_2)}{P(word_1)P(word_2)}$$

PMI applied to dependency relations

Hindle, Don. 1990. Noun Classification from Predicate-Argument Structure. ACL

Object of "drink"	Count	PMI
tea	2	11.8
liquid	2	10.5
wine	2	9.3
anything	3	5.2
it	3	1.3

"Drink it" more common than "drink wine"

But "wine" is a better "drinkable" thing than "it"

Positive Pointwise Mutual Information

- PMI ranges from $-\infty$ to $+\infty$
- But the negative values are problematic
  - Things are co-occurring **less than** we expect by chance
  - Unreliable without enormous corpora
    - Imagine w1 and w2 whose probability is each 10^{-6}
    - Hard to be sure p(w1,w2) is significantly different than 10^{-12}
  - Plus it's not clear people are good at "unrelatedness"
- So we just replace negative PMI values by 0
- Positive PMI (PPMI) between word1 and word2:

$$\text{PPMI}(word_1, word_2) = \max\left(\log_2 \frac{P(word_1, word_2)}{P(word_1)P(word_2)}, 0\right)$$

Weighting PMI

PMI is biased toward infrequent events

- Very rare words have very high PMI values

Two solutions:

- Give rare words slightly higher probabilities
- Use add-one smoothing (which has a similar effect)

Vector Semantics

EVALUATING SIMILARITY

Evaluating similarity

Extrinsic (task-based, end-to-end) Evaluation:

- Question Answering
- Spell Checking
- Essay grading

Intrinsic Evaluation:

- Correlation between algorithm and human word similarity ratings
  - Wordsim353: 353 noun pairs rated 0-10. *sim(plane,car)=5.77*
- Taking TOEFL multiple-choice vocabulary tests
  - `Levied is closest in meaning to:`

    `imposed, believed, requested, correlated`

Summary

Distributional (vector) models of meaning

- **Sparse** (PPMI-weighted word-word co-occurrence matrices)
- **Dense:** Next lecture!