Distributed Representations

CMSC 473/673

Frank Ferraro

Some slides adapted from 3SLP

Outline

- Continuous representations
  - Motivation
  - Key idea: represent blobs with vectors
- Evaluation
- Common continuous representation models

How have we represented words?

Each word is a distinct item

- Bijection between the strings and unique integer ids:
- "cat" --> 3, "kitten" --> 792 "dog" --> 17394
- Are "cat" and "kitten" similar?

Equivalently: "One-hot" encoding

- Represent each word type w with a vector the size of the vocabulary
- This vector has V-1 zero entries, and 1 non-zero (one) entry

Recall from Deck 2:
Representing a Linguistic "Blob"

1. An array of sub-blobs
   word → array of characters
   sentence → array of words

2. Integer representation/one-hot encoding

↑This is what we've (implicitly?) been using

3. Dense embedding

Let V = vocab size (# types)

1. Represent each word *type* with a unique integer i, where $0 \leq i < V$
2. Or equivalently, …
   - Assign each word to some index i, where $0 \leq i < V$
   - Represent each word w with a V-dimensional **binary** vector e_w, where $e_{w,i} = 1$ and 0 otherwise

Recall from Deck 2:
One-Hot Encoding Example

- Let our vocab be {a, cat, saw, mouse, happy}
- V = # types = 5
- Assign:

a	4
cat	2
saw	3
mouse	0
happy	1

How do we represent "cat?"

$$e_{\text{cat}} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

How do we represent "happy?"

$$e_{\text{happy}} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

The Fragility of One-Hot Encodings
Case Study: Maxent Plagiarism Detector

Given two documents x_1, x_2, predict y = 1 (plagiarized) or y = 0 (not plagiarized)

What is/are the:

- Method/steps for predicting?
- General formulation?
- Features?

There's no way you'll catch me!

Image: http://3.bp.blogspot.com/_MWJDzvrnu7Y/TTcXpCWZerI/AAAAAAAAAWY/zxu1GwrIBz0/s1600/excellent-mr-burns.gif

Case Study: Maxent Plagiarism Detector (Feature Example)

Given two documents x_1, x_2, predict y = 1 (plagiarized) or y = 0 (not plagiarized)

- Intuition: documents are more likely to be plagiarized if they have words in common

$$f_{\text{any}-\text{common}-\text{word},\text{Plag.}}(x_1, x_2) = ???$$

$$f_{<\text{word v}>,\text{Plag.}}(x_1, x_2) = ???$$

Yes, but surely some words will be in common… these features won’t catch phrases!

Image: http://3.bp.blogspot.com/_MWJDzvrnu7Y/TTcXpCWZerI/AAAAAAAAAWY/zxu1GwrIBz0/s1600/excellent-mr-burns.gif

Case Study: Maxent Plagiarism Detector (Feature Example)

Given two documents x_1, x_2, predict y = 1 (plagiarized) or y = 0 (not plagiarized)

- Intuition: documents are more likely to be plagiarized if they have words in common

$$f_{\text{any}-\text{common}-\text{word},\text{Plag.}}(x_1, x_2) = ???$$

$$f_{<\text{word v}>,\text{Plag.}}(x_1, x_2) = ???$$

$$f_{<\text{ngram Z}>,\text{Plag.}}(x_1, x_2) = ???$$

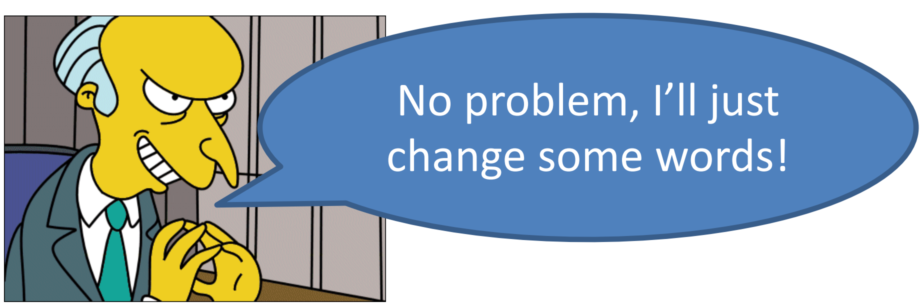


Image: http://3.bp.blogspot.com/_MWJDzvrnu7Y/TTcXpCWZerI/AAAAAAAAAWY/zxu1GwrIBz0/s1600/excellent-mr-burns.gif

Case Study: Maxent Plagiarism Detector (Feature Example)

Given two documents x_1, x_2, predict y = 1 (plagiarized) or y = 0 (not plagiarized)

- Intuition: documents are more likely to be plagiarized if they have words in common

Okay… but there are too many possible synonym n-grams!

$f \ldots{}_{\text{d,Plag.}}(x_1, x_2) = ???$

$\ldots{}_{\text{g.}}(x_1, x_2) = ???$

$f_{\text{<ngram Z>,Plag.}}(x_1, x_2) = ???$

$f_{\text{synonym}-\text{of}-\text{<word v>,Plag.}}(x_1, x_2) = ???$

Image: http://3.bp.blogspot.com/_MWJDzvrnu7Y/TTcXpCWZerI/AAAAAAAAAWY/zxu1GwrIBz0/s1600/excellent-mr-burns.gif

Case Study: Maxent Plagiarism Detector (Feature Example)

Given two documents x_1, x_2, predict y = 1 (plagiarized) or y = 0 (not plagiarized)

- Intuition: documents are more likely to be plagiarized if they have words in common

$$f_{\ldots\text{d,Plag.}}(x_1, x_2) = ???$$

$$\ldots\text{g.}(x_1, x_2) = ???$$

$$f_{\text{<ngram Z>,Plag.}}(x_1, x_2) = ???$$

$$f_{\text{synonym}-\text{of}-\text{<word v>,Plag.}}(x_1, x_2) = ???$$

$$f_{\text{synonym}-\text{of}-\text{<ngram Z>,Plag.}}(x_1, x_2) = ???$$

Image: http://3.bp.blogspot.com/_MWJDzvrnu7Y/TTcXpCWZerI/AAAAAAAAAWY/zxu1GwrIBz0/s1600/excellent-mr-burns.gif

Plagiarism Detection: Word Similarity?

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

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,

Recall from Deck 2:
Representing a Linguistic "Blob"

1. An array of sub-blobs
   word → array of characters
   sentence → array of words

2. Integer representation/one-hot encoding

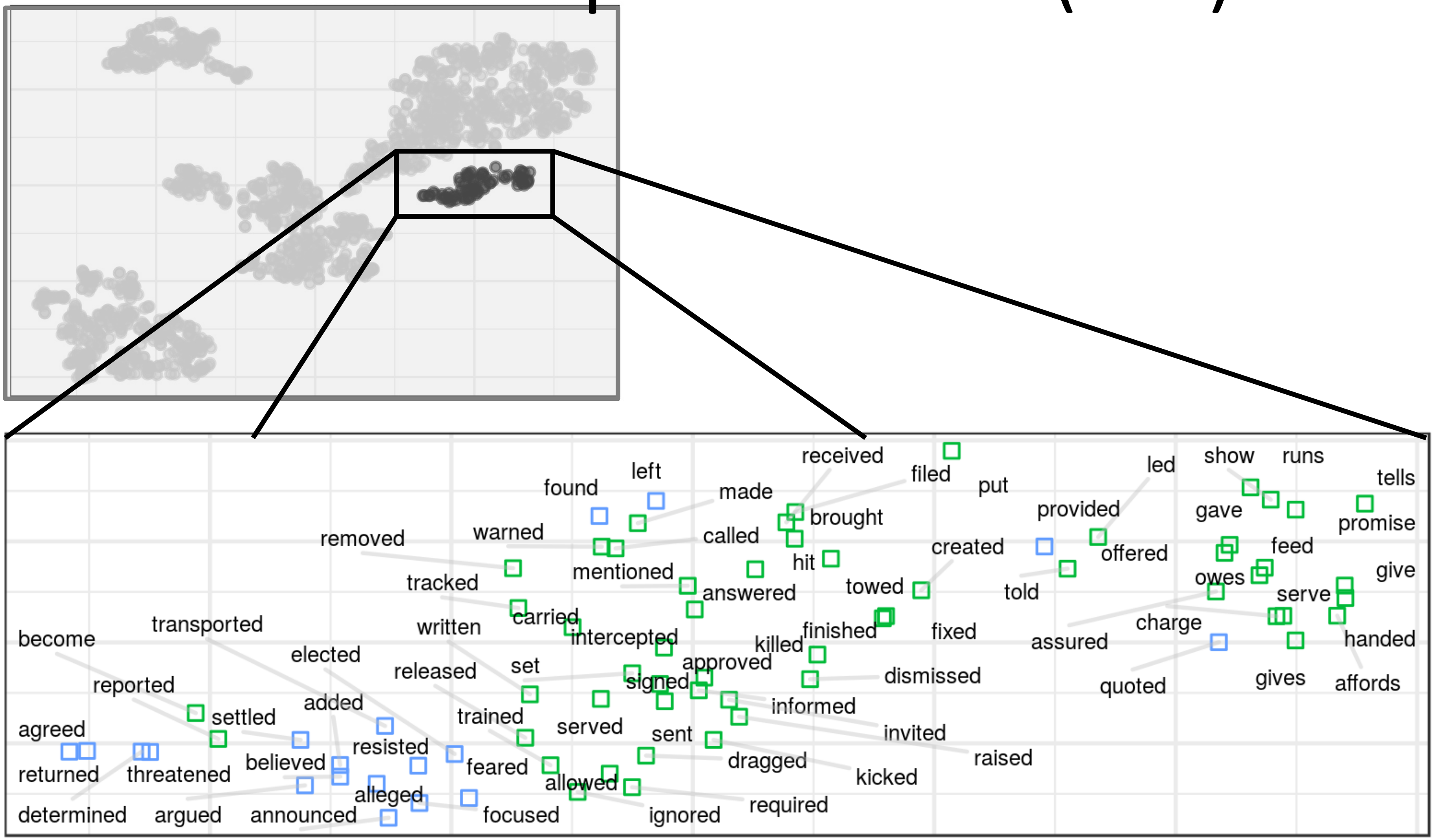
3. Dense embedding

Let E be some *embedding size* (often 100, 200, 300, etc.)

Represent each word w with an E-dimensional **real-valued** vector e_w

Remember:

A Dense Representation (E=2)



Distributional Representations

A dense, “low” dimensional vector representation

Distributional Representations

A dense, “low” dimensional vector representation

An E-dimensional vector, often (but not always) real-valued

Distributional Representations

A dense, “low” dimensional vector representation

Up till ~2013: E could be any size
2013-present: E << vocab

An E-dimensional vector, often (but not always) real-valued

Distributional Representations

A dense, "low" dimensional vector representation

Many values are not 0 (or at least less sparse than one-hot)

Up till ~2013: E could be any size
2013-present: E << vocab

An E-dimensional vector, often (but not always) real-valued

Distributional Representations

A dense, "low" dimensional vector representation

Many values are not 0 (or at least less sparse than one-hot)

Up till ~2013: E could be any size
2013-present: E << vocab

An E-dimensional vector, often (but not always) real-valued

These are also called

- **embeddings**
- **Continuous representations**
- **(word/sentence/…) vectors**
- **Vector-space models**

Distributional models of meaning
= vector-space models of meaning
= vector semantics

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."

Firth (1957):

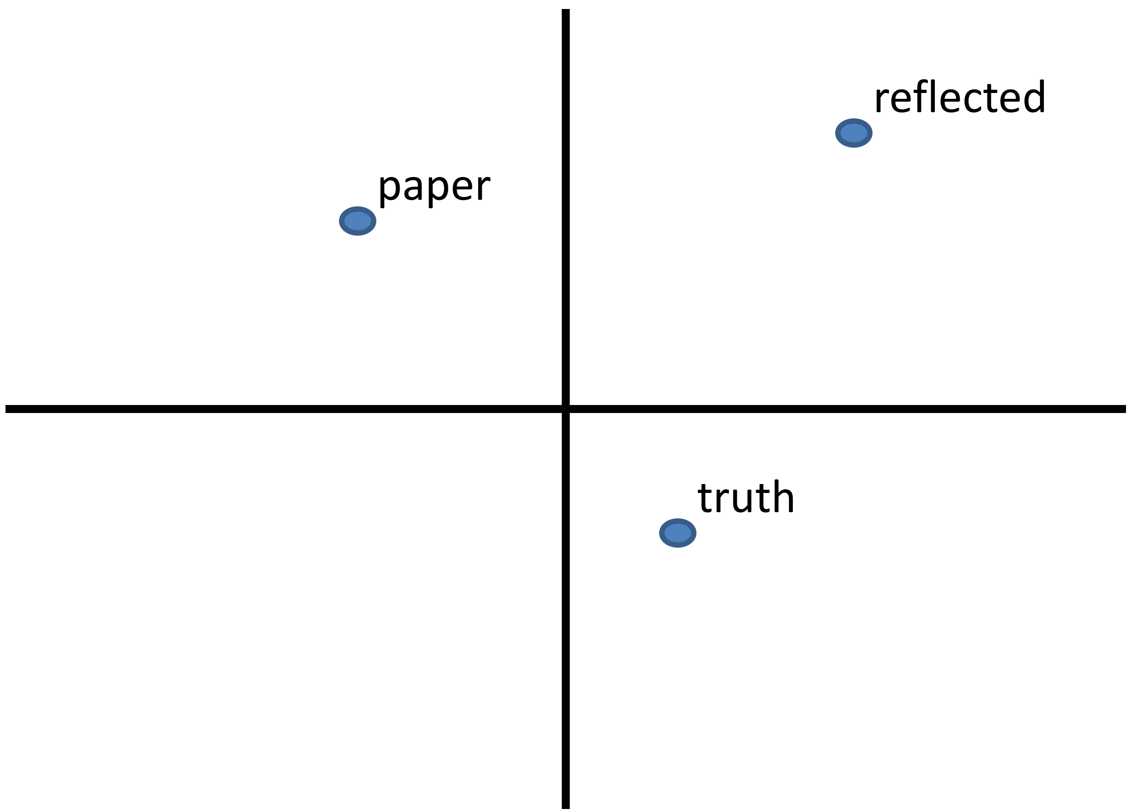
- "You shall know a word by the company it keeps!"

Continuous Meaning

The paper reflected the truth.

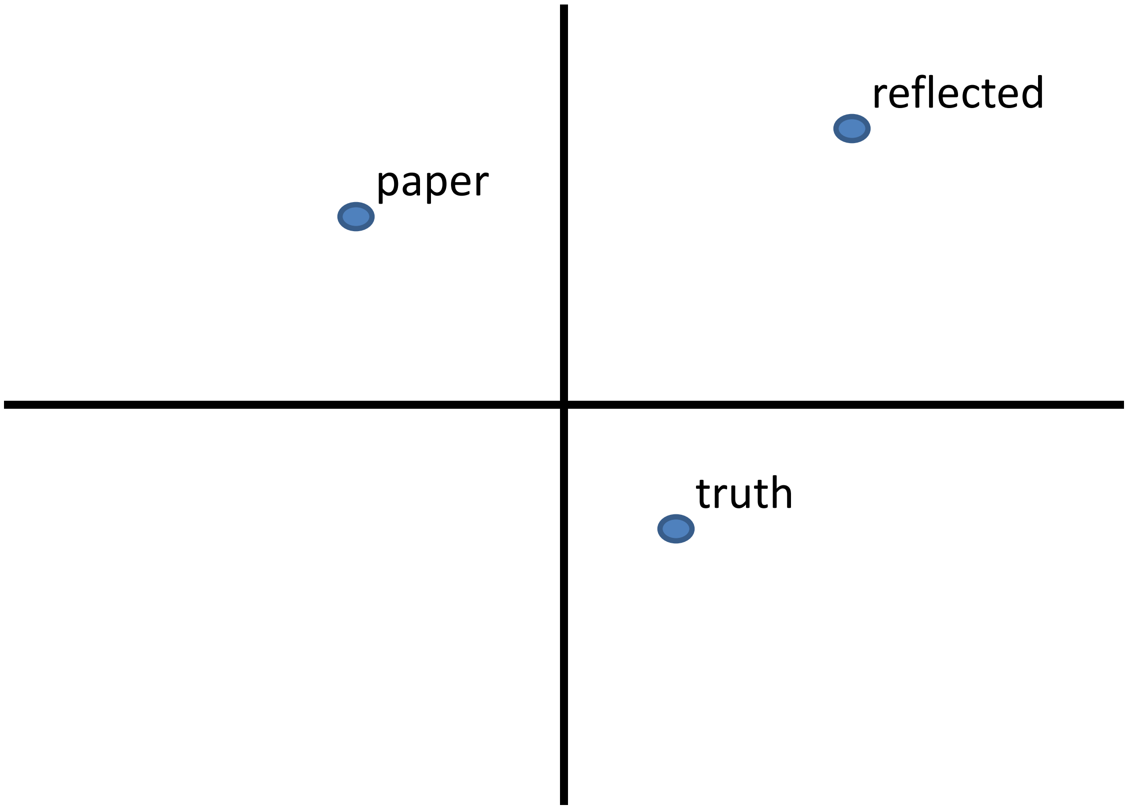
Continuous Meaning

The paper reflected the truth.



Continuous Meaning

The paper reflected the truth.



glean

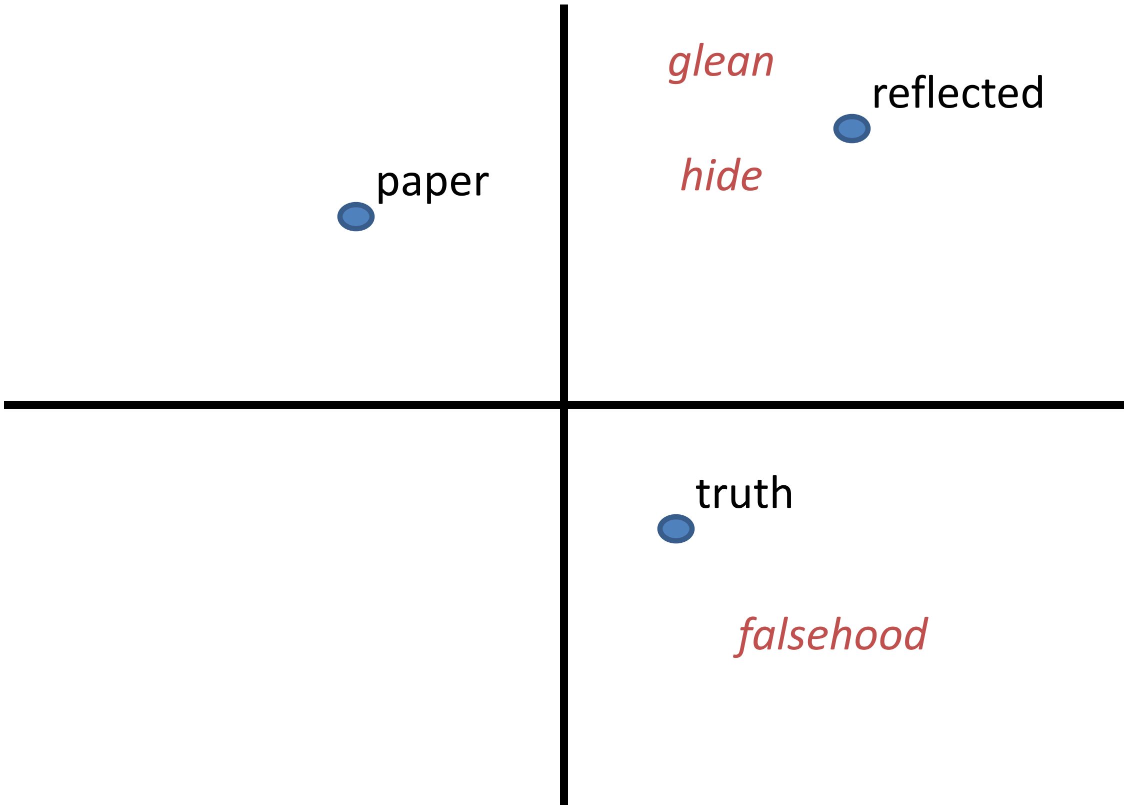
hide

falsehood

where might these go in this space?

Continuous Meaning

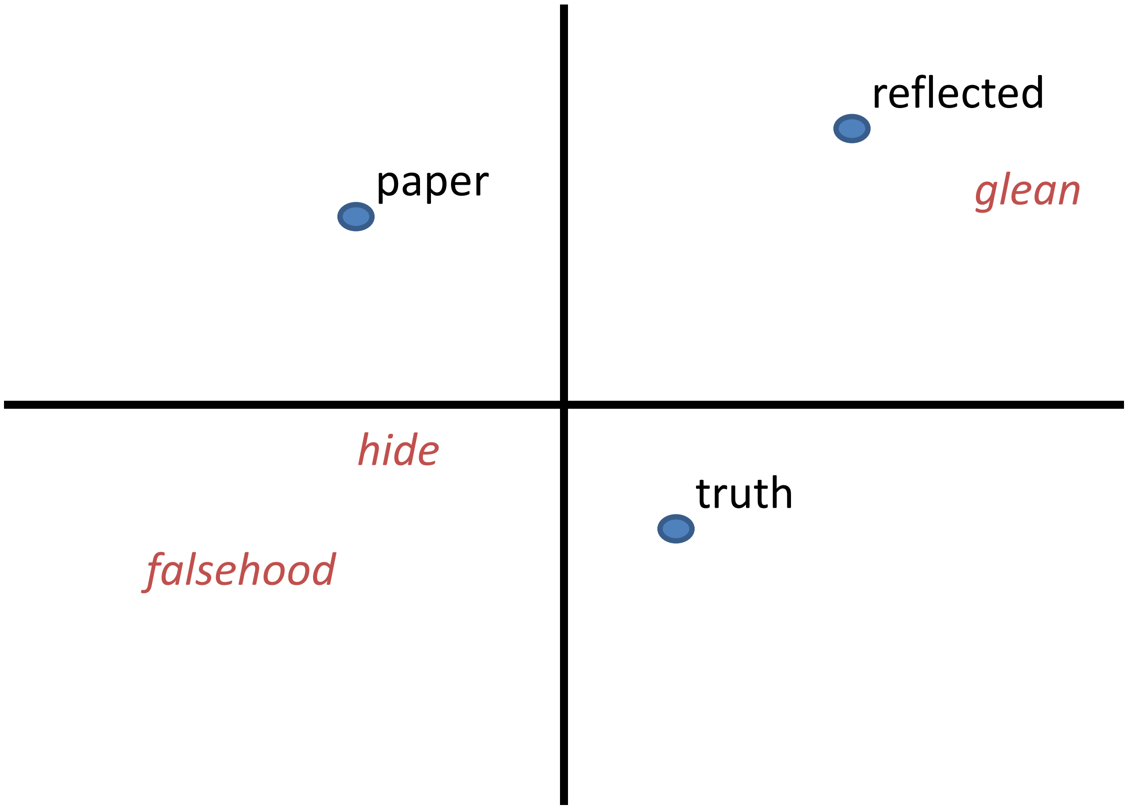
The paper reflected the truth.



One option

Continuous Meaning

The paper reflected the truth.



Another option

(Some) Properties of Embeddings

https://media3.giphy.com/media/3orif0M8U1E7NfpFzq/200_s.gif

Capture "like" (similar) words

target:	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

Mikolov et al. (2013)

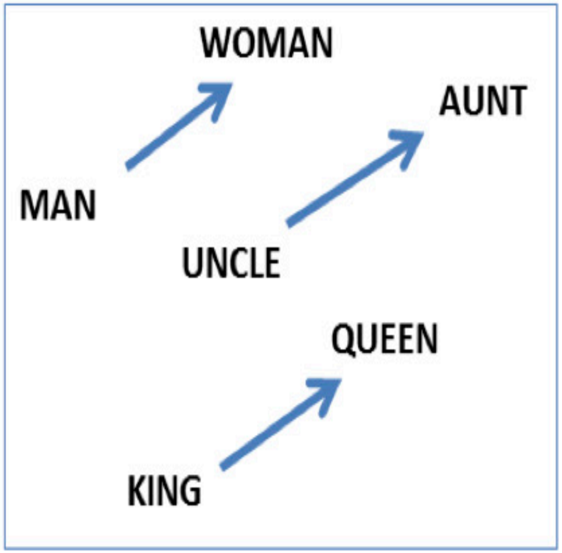
(Some) Properties of Embeddings

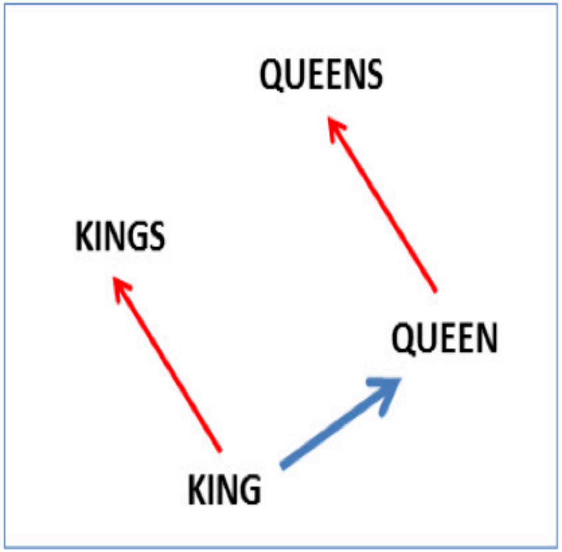
https://media3.giphy.com/media/3orif0M8U1E7NfpFzq/200_s.gif

Capture "like" (similar) words

target:	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

Capture relationships





vector(*'king'*) –
vector(*'man'*) +
vector(*'woman'*) ≈
vector('queen')

vector(*'Paris'*) -
vector(*'France'*) +
vector(*'Italy'*) ≈
vector('Rome')

Mikolov et al. (2013)

Case Study: Maxent Plagiarism Detector (Feature Example)

Given two documents x_1, x_2, predict y = 1 (plagiarized) or y = 0 (not plagiarized)

- Intuition: documents are more likely to be plagiarized if they have words in common

$$f \ldots \text{rd,Plag.}(x_1, x_2) = ???$$

$$\ldots (x_1, x_2) = ???$$

$$f \ldots \text{>,Plag.}(x_1, x_2) = ???$$

$$f_{\text{synonym}-\text{of}-\text{<word v>,Plag.}}(x_1, x_2) = ???$$

$$f_{\text{synonym}-\text{of}-\text{<ngram Z>,Plag.}}(x_1, x_2) =$$

get_similarity_with_embeddings()

Image: http://3.bp.blogspot.com/_MWJDzvrnu7Y/TTcXpCWZerI/AAAAAAAAAWY/zxu1GwrIBz0/s1600/excellent-mr-burns.gif

Outline

- Continuous representations
  - Motivation
  - Key idea: represent blobs with vectors
- Evaluation
- Common continuous representation models

“Embeddings” Did Not Begin In This Century…

Hinton (1986): “Learning Distributed Representations of Concepts”

Deerwester et al. (1990): “Indexing by Latent Semantic Analysis”

Brown et al. (1992): “Class-based n-gram models of natural language”

Key Ideas

1. Acquire basic contextual statistics (often counts) for each word type v

Key Ideas

1. Acquire basic contextual statistics (often counts) for each word type v

2. Extract a real-valued vector e_v for each word v from those statistics

Key Ideas

1. Acquire basic contextual statistics (often counts) for each word type v

2. Extract a real-valued vector e_v for each word v from those statistics

3. Use the vectors to represent each word in later tasks

Key Ideas: Generalizing to “blobs”

1. Acquire basic contextual statistics (often counts) for each blob type v

2. Extract a real-valued vector e_v for each blob v from those statistics

3. Use the vectors to represent each blob in later tasks

Outline

- Continuous representations
  - Motivation
  - Key idea: represent blobs with vectors
- Evaluation
- Common continuous representation models

Evaluating Similarity

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

- Question Answering
- Spell Checking
- Essay grading

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
- Taking TOEFL multiple-choice vocabulary tests

Common Evaluation: Correlation between similarity ratings

- Input: list of N word pairs $\{(x_1, y_1), \dots, (x_N, y_N)\}$
  - Each word pair (x_i, y_i) has a human-provided similarity score h_i
- Use your embeddings to compute an embedding similarity score $s_i = \mathrm{sim}(x_i, y_i)$
- Compute the correlation between human and computed similarities

$$\rho = \mathrm{Corr}((h_1, \dots, h_N), (s_1, \dots, s_N))$$

- Wordsim353: 353 noun pairs rated 0-10

Cosine: Measuring Similarity

Given 2 target words *v* and *w* how similar are their *vectors*?

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 for **orthogonal vectors** with zeros in complementary distribution

Cosine: Measuring Similarity

Given 2 target words *v* and *w* how similar are their *vectors*?

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 for **orthogonal vectors** with zeros in complementary distribution

Correct for high magnitude vectors $\dfrac{\vec{a}\cdot\vec{b}}{|\vec{a}||\vec{b}|}$

Cosine Similarity

Divide the dot product by the length of the two vectors

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

This is the cosine of the angle between them

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

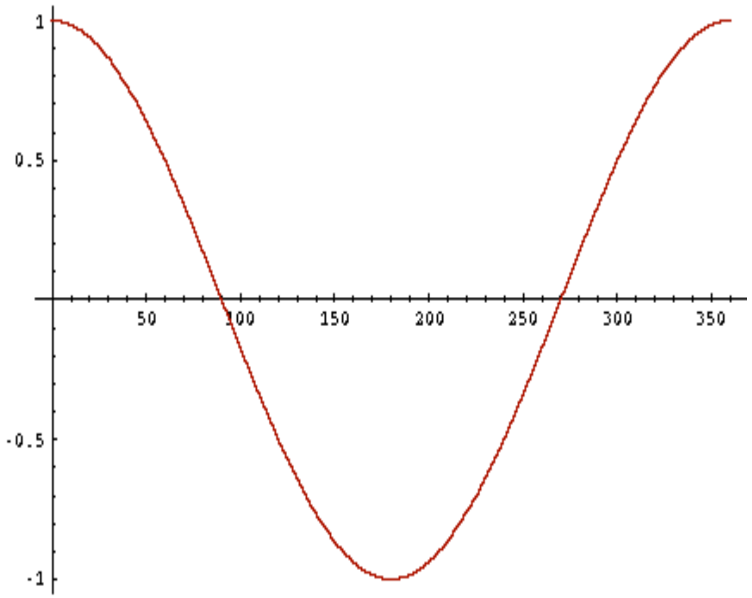
$$\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} = \cos\theta$$

Cosine as a similarity metric

-1: vectors point in opposite directions

+1: vectors point in same directions

0: vectors are orthogonal



Example: Word Similarity

$$\cos(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}}$$

word types ➔

	Dim. 1	Dim. 2	Dim. 3
apricot	2	0	0
digital	0	1	2
information	1	6	1

Example: Word Similarity

$$\cos(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

word types ➔

	Dim. 1	Dim. 2	Dim. 3
apricot	2	0	0
digital	0	1	2
information	1	6	1

cosine(apricot,information) =

cosine(digital,information) =

cosine(apricot,digital) =

Example: Word Similarity

$$\cos(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

word types ➔

	Dim. 1	Dim. 2	Dim. 3
apricot	2	0	0
digital	0	1	2
information	1	6	1

cosine(apricot,information) = $\frac{2+0+0}{\sqrt{4+0+0}\sqrt{1+36+1}} = 0.1622$

cosine(digital,information) =

cosine(apricot,digital) =

Example: Word Similarity

$$\cos(x,y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

	Dim. 1	Dim. 2	Dim. 3
apricot	2	0	0
digital	0	1	2
information	1	6	1

word types ➔

cosine(apricot,information) = $\frac{2+0+0}{\sqrt{4+0+0}\sqrt{1+36+1}} = 0.1622$

cosine(digital,information) = $\frac{0+6+2}{\sqrt{0+1+4}\sqrt{1+36+1}} = 0.5804$

cosine(apricot,digital) = $\frac{0+0+0}{\sqrt{4+0+0}\sqrt{0+1+4}} = 0.0$

Other Similarity Measures

$$\begin{aligned}
\mathrm{sim}_{\mathrm{cosine}}(\vec{v},\vec{w}) &= \frac{\vec{v}\cdot\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\sum_{i=1}^{N} v_i \times w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}} \\
\mathrm{sim}_{\mathrm{Jaccard}}(\vec{v},\vec{w}) &= \frac{\sum_{i=1}^{N} \min(v_i,w_i)}{\sum_{i=1}^{N} \max(v_i,w_i)} \\
\mathrm{sim}_{\mathrm{Dice}}(\vec{v},\vec{w}) &= \frac{2\times\sum_{i=1}^{N} \min(v_i,w_i)}{\sum_{i=1}^{N}(v_i+w_i)} \\
\mathrm{sim}_{\mathrm{JS}}(\vec{v}||\vec{w}) &= D(\vec{v}|\frac{\vec{v}+\vec{w}}{2}) + D(\vec{w}|\frac{\vec{v}+\vec{w}}{2})
\end{aligned}$$

Adding Morphology, Syntax, and Semantics to Embeddings

Lin (1998): “Automatic Retrieval and Clustering of Similar Words”

Padó and Lapata (2007): “Dependency-based Construction of Semantic Space Models”

Levy and Goldberg (2014): “Dependency-Based Word Embeddings”

Cotterell and Schütze (2015): “Morphological Word Embeddings”

Ferraro et al. (2017): “Frame-Based Continuous Lexical Semantics through Exponential Family Tensor Factorization and Semantic Proto-Roles”

and many more…

Outline

Continuous representations

- Motivation
- Key idea: represent blobs with vectors

Evaluation

Common continuous representation models

Shared Intuition

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

The meaning of a word is a vector of numbers

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

Three Common Kinds of Embedding Models

1. Co-occurrence matrices
2. Matrix Factorization: Singular value decomposition/Latent Semantic Analysis, Topic Models
3. Neural-network-inspired models (skip-grams, CBOW)

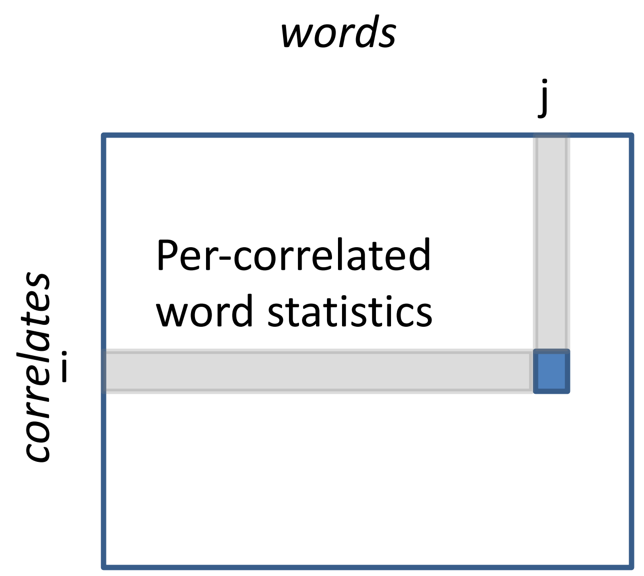
Three Common Kinds of Embedding Models

1. Co-occurrence matrices
2. Matrix Factorization: Singular value decomposition/Latent Semantic Analysis
3. Neural-network-inspired models (skip-grams, CBOW)

Co-occurrence matrices can be used in their own right, but they're most often used as inputs (directly or indirectly) to the matrix factorization or neural approaches

Co-occurrence Matrix

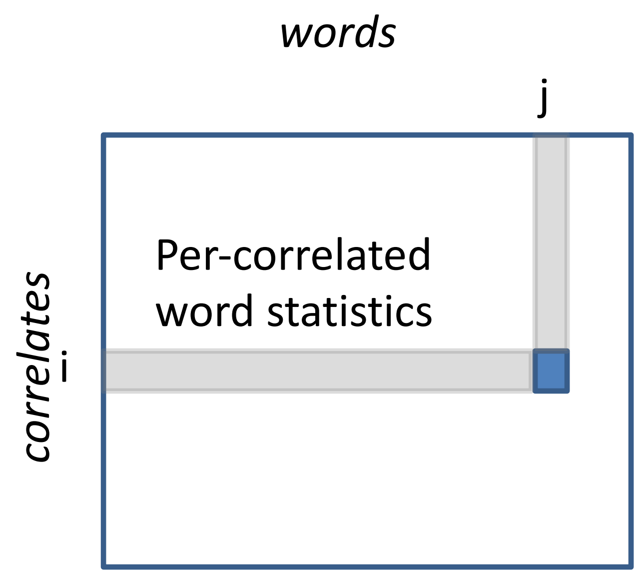
Acquire basic contextual statistics (often counts) for each word type v via *correlate*.



Co-occurrence Matrix

Acquire basic contextual statistics (often counts) for each word type v via *correlate*: For example:

- documents
  - Record how often a word occurs in each document
-

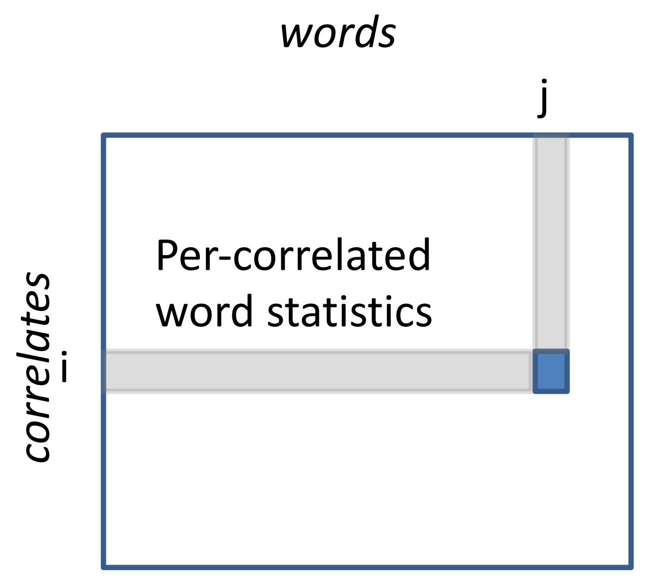


correlates =
documents

Co-occurrence Matrix

Acquire basic contextual statistics (often counts) for each word type v via *correlate*: For example:

- documents
- surrounding context words
  - Record how often v occurs with other word types u
-

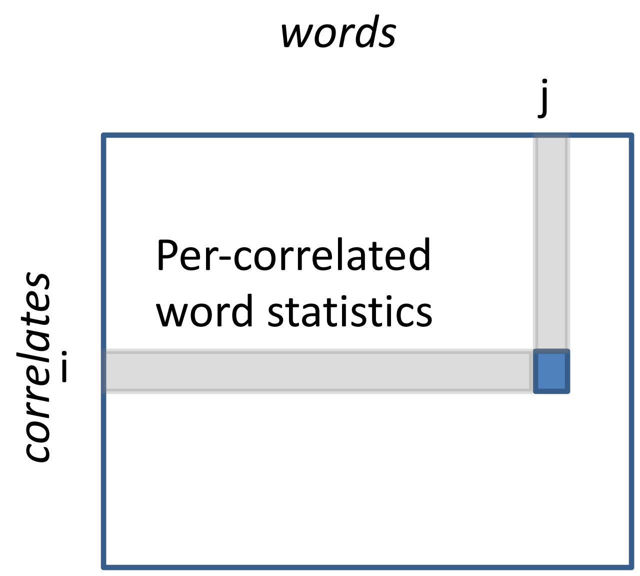


\# correlates =
\# word types

Co-occurrence Matrix

Acquire basic contextual statistics (often counts) for each word type v via *correlate*: For example:

- documents
- surrounding context words
- linguistic annotations (POS tags, syntax)
- …



Assumption: Two words are similar if their vectors are similar

“Acquire basic contextual statistics (often counts) for each word type v”

- Two basic, initial counting approaches
  - Record which words appear in which documents
  - Record which words appear together
- These are good first attempts, but with some large downsides

“You shall know a word by the company it keeps!” Firth (1957)

document (↓)-word (→) count matrix

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

basic bag-of-words counting

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1

"You shall know a word by the company it keeps!" Firth (1957)

document (↓)-word (→) count matrix

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

Assumption: Two documents are similar if their vectors are similar

“You shall know a word by the company it keeps!” Firth (1957)

document (↓)-word (→) count matrix

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

Assumption: Two words are similar if their vectors are similar???

“You shall know a word by the company it keeps!” Firth (1957)

document (↓)-word (→) count matrix

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

Assumption: Two words are similar if their vectors are similar

Issue: Count word vectors are very large, sparse, and skewed!

"You shall know a word by the company it keeps!" Firth (1957)

context (↓)-word (→) count matrix

	apricot	pineapple	digital	information
aardvark	0	0	0	0
computer	0	0	2	1
data	0	10	1	6
pinch	1	1	0	0
result	0	0	1	4
sugar	1	1	0	0

Context: those other words within a small "window" of a target word

"You shall know a word by the company it keeps!" Firth (1957)

context (↓)-word (→) count matrix

	apricot	pineapple	digital	information
aardvark	0	0	0	0
computer	0	0	2	1
data	0	10	1	6
pinch	1	1	0	0
result	0	0	1	4
sugar	1	1	0	0

Context: those other words within a small "window" of a target word

a cloud [computer stores digital data on] a remote computer

"You shall know a word by the company it keeps!" Firth (1957)

context (↓)-word (→) count matrix

	apricot	pineapple	digital	information
aardvark	0	0	0	0
computer	0	0	2	1
data	0	10	1	6
pinch	1	1	0	0
result	0	0	1	4
sugar	1	1	0	0

The size of windows depends on your goals

- The shorter the windows , the more **syntactic** the representation
  - $\pm$ 1-3 more "syntax-y"
- The longer the windows, the more **semantic** the representation
  - $\pm$ 4-10 more "semantic-y"

"You shall know a word by the company it keeps!" Firth (1957)

context (↓)-word (→) count matrix

	apricot	pineapple	digital	information
aardvark	0	0	0	0
computer	0	0	2	1
data	0	10	1	6
pinch	1	1	0	0
result	0	0	1	4
sugar	1	1	0	0

Context: those other words within a small "window" of a target word

Assumption: Two words are similar if their vectors are similar

Issue: Count word vectors are very large, sparse, and skewed!

Pointwise Mutual Information (PMI): Dealing with Problems of 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 ((P)PMI)

Pointwise Mutual Information (PMI): Dealing with Problems of 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 ((P)PMI)

Pointwise mutual information:

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

probability words x and y occur together (in the same context/window)

$$\mathrm{PMI}(x, y) = \log \frac{p(x, y)}{p(x)p(y)}$$

probability that word x occurs

probability that word y occurs

Pointwise Mutual Information (PMI): Dealing with Problems of 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 ((P)PMI)

Pointwise mutual information:

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

$$\mathrm{PMI}(x, y) = \log \frac{p(x, y)}{p(x)p(y)}$$

PMI between two words: (Church & Hanks 1989)

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

Advanced: Equivalent PMI Computations

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

$$\mathrm{PMI}(x, y) = \log \frac{p(x, y)}{p(x)p(y)} = \log \frac{p(y \mid x)}{p(y)} = \log \frac{p(x \mid y)}{p(x)}$$

“Noun Classification from Predicate-Argument Structure,” Hindle (1990)

“drink it” is more common than “drink wine”

“wine” is a better “drinkable” thing than “it”

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

Advanced topic

Brown clustering (Brown et al., 1992)

An agglomerative clustering algorithm that clusters words based on which words precede or follow them

These word clusters can be turned into a kind of vector (binary vector)

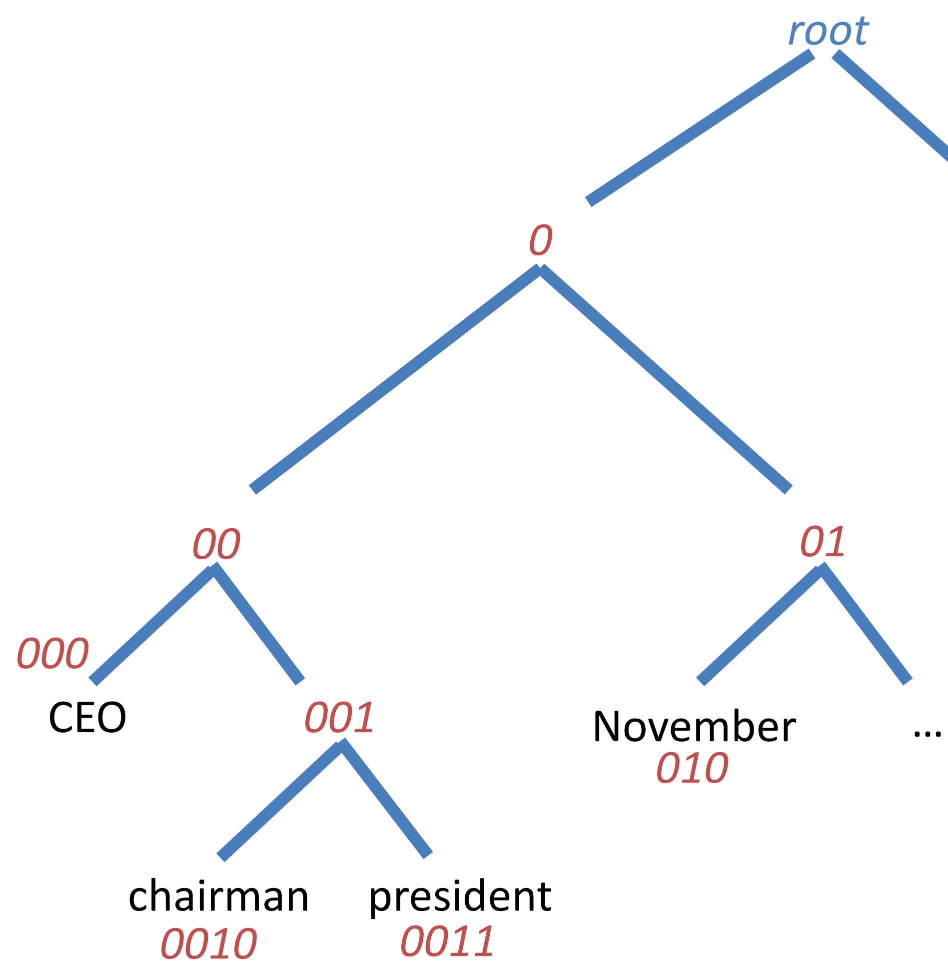
Advanced topic

Brown Clusters as vectors

Build a binary tree from bottom to top based on how clusters are merged

Each word represented by binary string = path from root to leaf

Each intermediate node is a cluster



In practice, use an available implementation: https://github.com/percyliang/brown-cluster

Advanced topic

Brown cluster examples (from 3SLP)

Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays Saturdays

June March July April January December October November September August

pressure temperature permeability density porosity stress velocity viscosity gravity tension

anyone someone anybody somebody

had hadn't hath would've could've should've must've might've

asking telling wondering instructing informing kidding reminding bothering thanking deposing

mother wife father son husband brother daughter sister boss uncle

great big vast sudden mere sheer gigantic lifelong scant colossal

down backwards ashore sideways southward northward overboard aloft downwards adrift

(each of these bars is a different cluster)

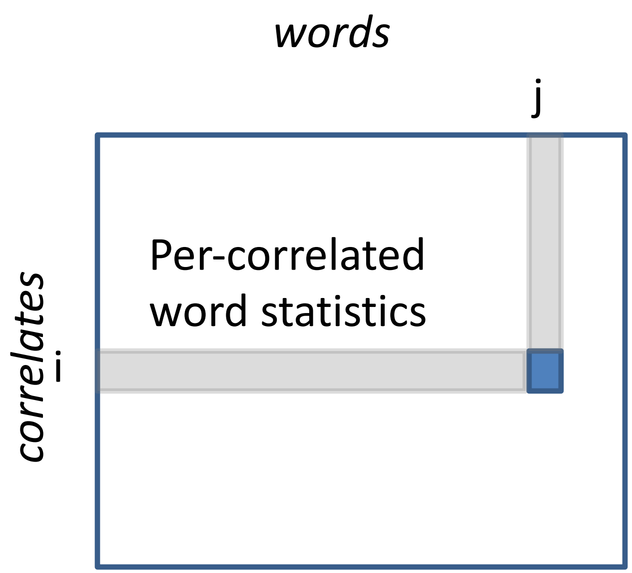
Three Common Kinds of Embedding Models

Learn more in:
- Your project
- Paper (673)
- Other classes (478/678)

1.
2. Matrix Factorization: Singular value decomposition/Latent Semantic Analysis, Topic Models
3. Neural-network-inspired models (skip-grams, CBOW)

Advanced topic

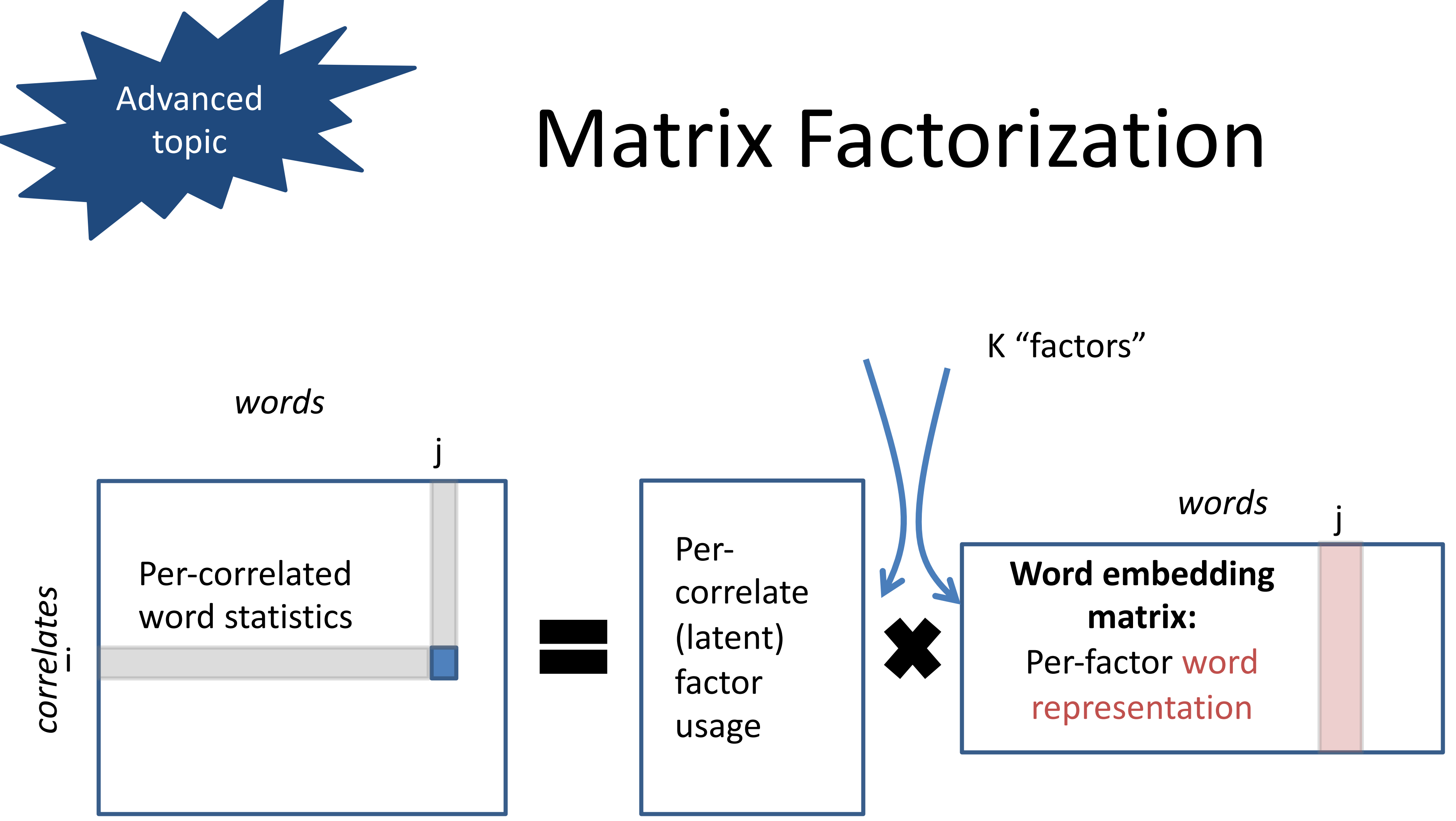
Matrix Factorization



correlate examples:

- documents
- surrounding context words
- linguistic annotations (POS tags, syntax)
- …

Matrix Factorization



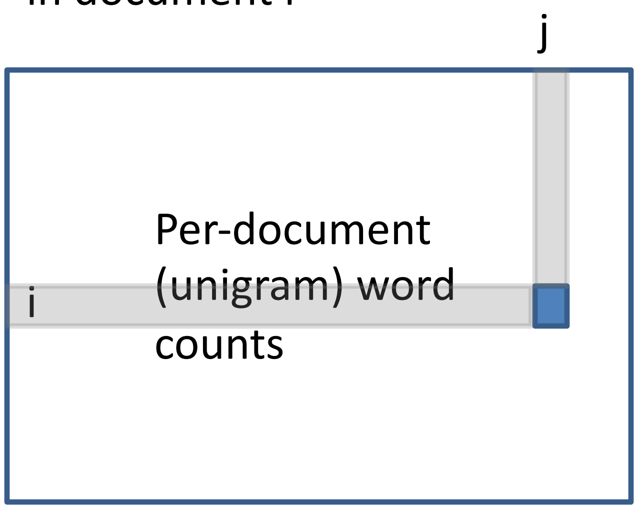
correlate examples:

- documents
- surrounding context words
- linguistic annotations (POS tags, syntax)
- …

Advanced topic

Topic Models: Latent Dirichlet Allocation (Blei et al., 2003)

Count of word j in document i



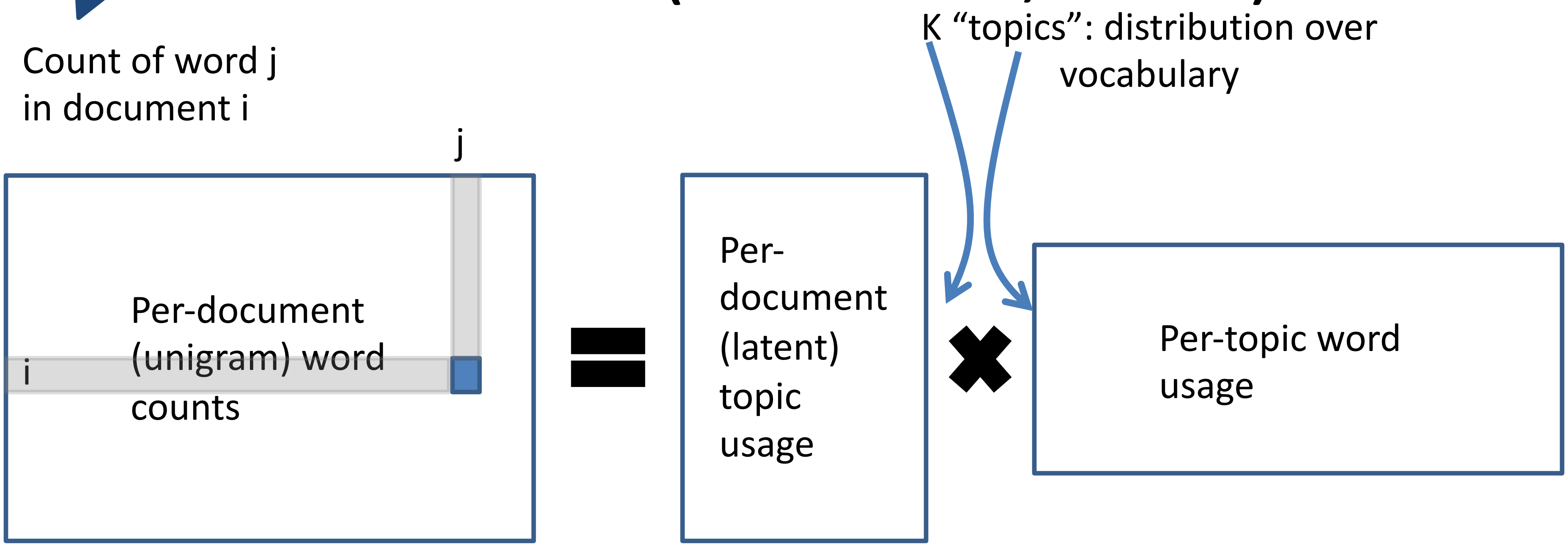
Core assumptions:

1. K “topics”: distributions over possible vocab words
2. Each document i has general “preferences” for which topics to use
3. Each observed word j in a document i can come from a *different* topic

In practice, many people use the `gensim` library

Advanced topic

Latent Dirichlet Allocation (Blei et al., 2003)



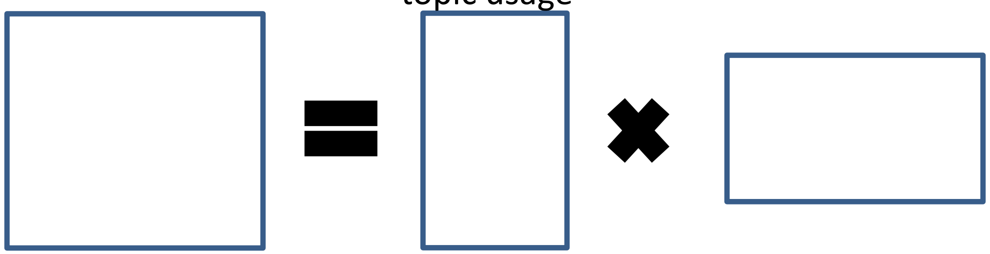
Advanced topic

Latent Dirichlet Allocation (Blei et al., 2003)

Per-document (unigram) word counts

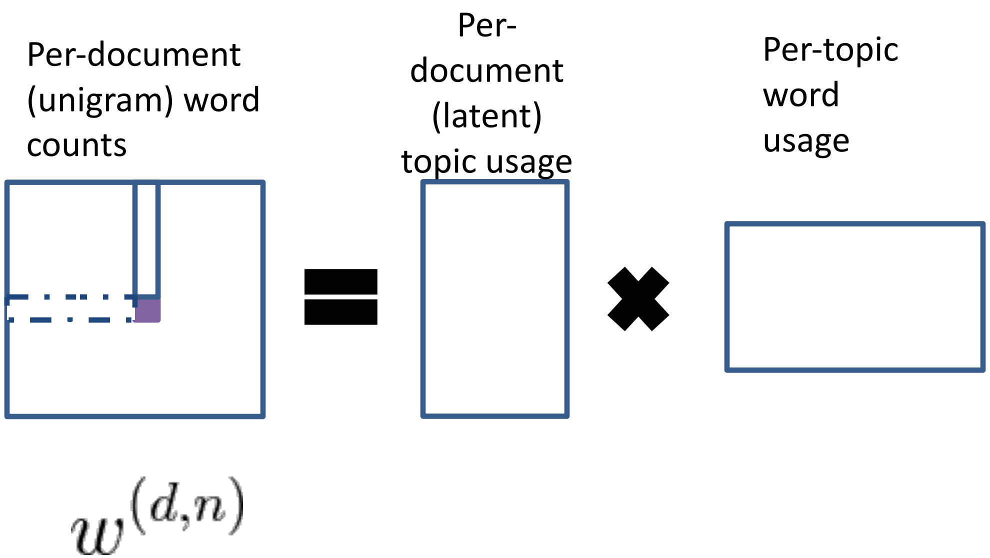
Per-document (latent) topic usage

Per-topic word usage



Advanced topic

Latent Dirichlet Allocation (Blei et al., 2003)



Explicit conditioning left off (for space)

$$w^{(d,n)} \sim \text{Discrete}$$

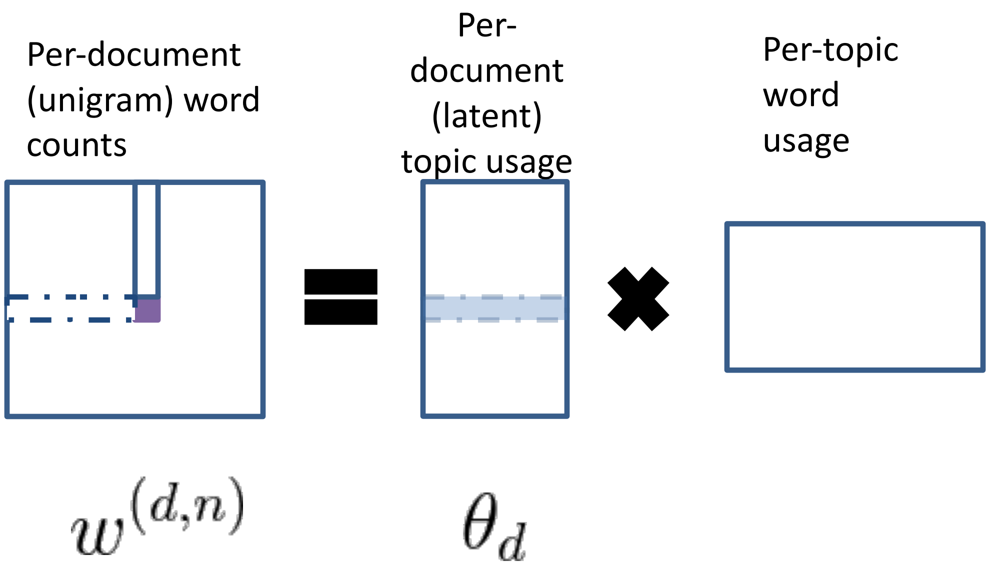
Generative story
(CMSC 691: Graphical & Statistical Models of Learning)

Core assumptions:

1. K "topics": distributions over possible vocab words
2. Each document i has general "preferences" for which topics to use
3. Each observed word j in a document i can come from a *different* topic

Advanced topic

Latent Dirichlet Allocation (Blei et al., 2003)



Explicit conditioning left off (for space)

$$w^{(d,n)} \sim \text{Discrete}$$

$$\boldsymbol{\theta}^{(d)} \sim \text{Dirichlet}\left(\boldsymbol{\alpha}\right)$$

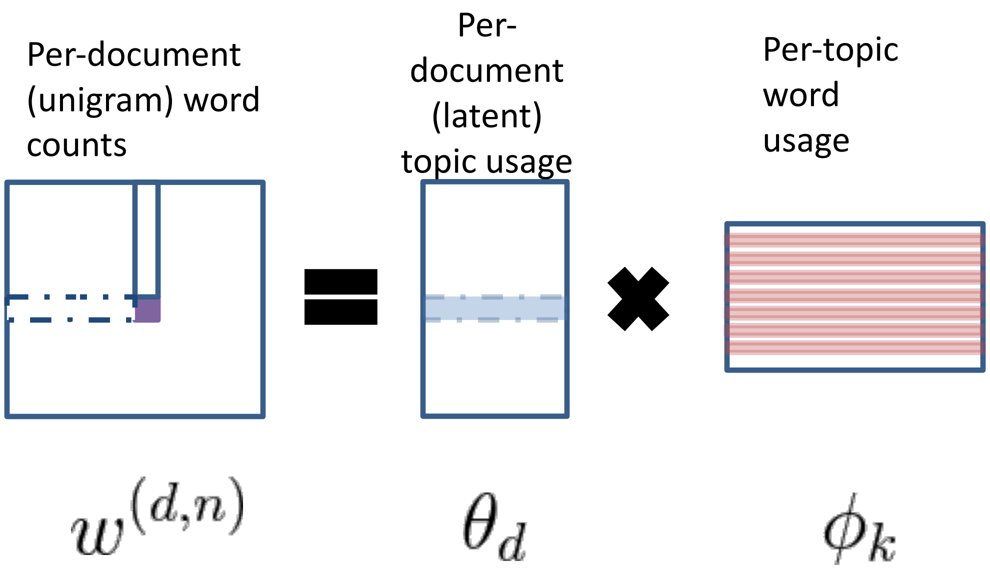
Generative story
(CMSC 691: Graphical & Statistical Models of Learning)

Core assumptions:

1. K “topics”: distributions over possible vocab words
2. Each document i has general “preferences” for which topics to use
3. Each observed word j in a document i can come from a *different* topic

Advanced topic

Latent Dirichlet Allocation (Blei et al., 2003)



Explicit conditioning left off (for space)

$$w^{(d,n)} \sim \text{Discrete}$$
$$\phi_k \sim \text{Dirichlet}\left(\boldsymbol{\beta}\right)$$
$$\boldsymbol{\theta}^{(d)} \sim \text{Dirichlet}\left(\boldsymbol{\alpha}\right)$$

Generative story
(CMSC 691: Graphical & Statistical Models of Learning)

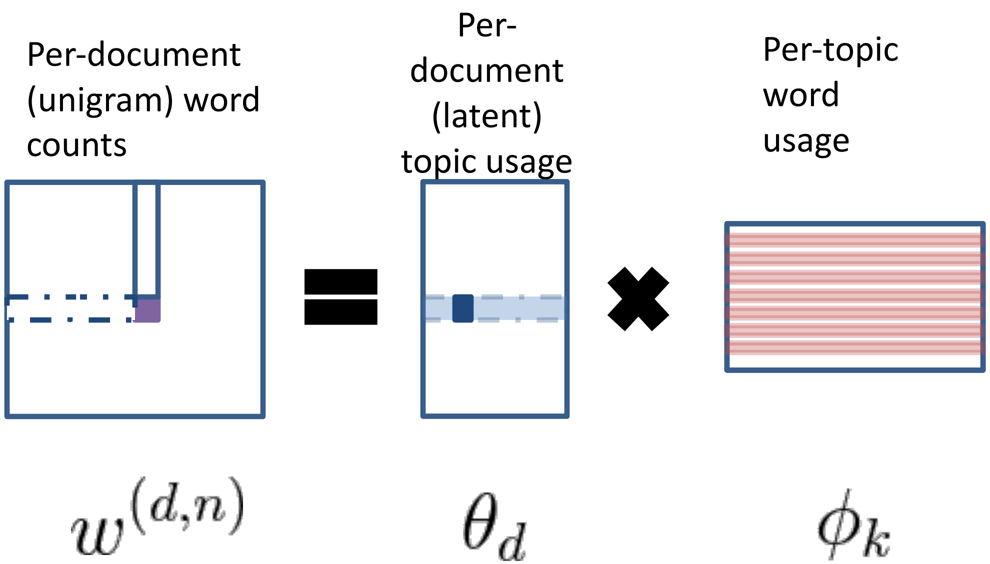
Core assumptions:

1. K “topics”: distributions over possible vocab words
2. Each document i has general “preferences” for which topics to use
3. Each observed word j in a document i can come from a *different* topic

Advanced topic

Latent Dirichlet Allocation (Blei et al., 2003)

Explicit conditioning left off (for space)



$$
\begin{aligned}
w^{(d,n)} &\sim \text{Discrete} \\
\phi_k &\sim \text{Dirichlet}\left(\boldsymbol{\beta}\right) \\
z^{(d,n)} &\sim \text{Discrete}\left(\boldsymbol{\theta}^{(d)}\right) \\
\boldsymbol{\theta}^{(d)} &\sim \text{Dirichlet}\left(\boldsymbol{\alpha}\right)
\end{aligned}
$$

Generative story
(CMSC 691: Graphical & Statistical Models of Learning)

Core assumptions:
1. K “topics”: distributions over possible vocab words
2. Each document i has general “preferences” for which topics to use
3. Each observed word j in a document i can come from a *different* topic

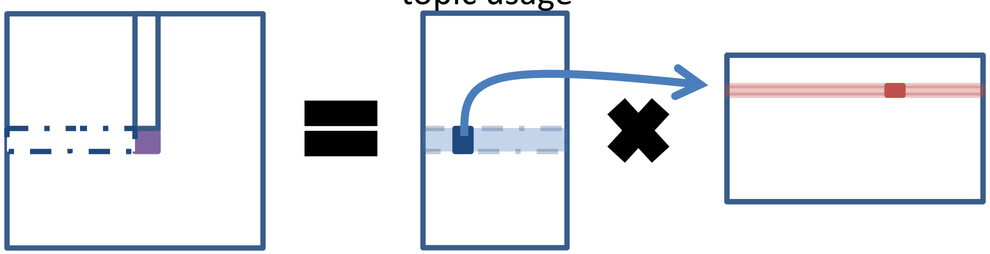
Advanced topic

Latent Dirichlet Allocation (Blei et al., 2003)

Per-document (unigram) word counts

Per-document (latent) topic usage

Per-topic word usage



$w^{(d,n)}$ θ_d $\phi_{z(d,n)}$

Explicit conditioning left off (for space)

$$w^{(d,n)} \sim \text{Discrete}\left(\phi_{z^{(d,n)}}\right)$$
$$\phi_k \sim \text{Dirichlet}\left(\boldsymbol{\beta}\right)$$
$$z^{(d,n)} \sim \text{Discrete}\left(\boldsymbol{\theta}^{(d)}\right)$$
$$\boldsymbol{\theta}^{(d)} \sim \text{Dirichlet}\left(\boldsymbol{\alpha}\right)$$

Generative story
(CMSC 691: Graphical & Statistical Models of Learning)

Core assumptions:

1. K “topics”: distributions over possible vocab words
2. Each document i has general “preferences” for which topics to use
3. Each observed word j in a document i can come from a *different* topic

Three Common Kinds of Embedding Models

1. Co-occurrence matrices
2. Matrix Factorization: Singular value decomposition/Latent Semantic Analysis
3. Neural-network-inspired models (skip-grams, CBOW)

Word2Vec

- Mikolov et al. (2013; NeurIPS): “Distributed Representations of Words and Phrases and their Compositionality”
- Revisits the context-word approach
- Learn a model p(c | w) to predict a context word from a target word

Word2Vec

- Mikolov et al. (2013; NeurIPS): “Distributed Representations of Words and Phrases and their Compositionality”
- Revisits the context-word approach
- Learn a model p(c | w) to predict a context word from a target word
- Learn two types of vector representations
  - $h_c \in \mathbb{R}^E$: vector embeddings for each context word
  - $v_w \in \mathbb{R}^E$: vector embeddings for each target word

$$p(c \mid w) \propto \exp(h_c^T v_w)$$

Word2Vec

context (↓)-word (→) count matrix

	apricot	pineapple	digital	information
aardvark	0	0	0	0
computer	0	0	2	1
data	0	10	1	6
pinch	1	1	0	0
result	0	0	1	4
sugar	1	1	0	0

Context: those other words within a small "window" of a target word

$$\max_{h,v} \sum_{c,w \text{ pairs}} \text{count}(c,w) \log p(c \mid w)$$

Word2Vec

context (↓)-word (→) count matrix

	apricot	pineapple	digital	information
aardvark	0	0	0	0
computer	0	0	2	1
data	0	10	1	6
pinch	1	1	0	0
result	0	0	1	4
sugar	1	1	0	0

Context: those other words within a small "window" of a target word

$$\max_{h,v} \sum_{c,w \text{ pairs}} \text{count}(c, w) \left[h_c^T v_w - \log\left(\sum_u \exp(h_u^T v_w)\right)\right]$$

Word2Vec has Inspired a Lot of Work

Off-the-shelf embeddings

https://code.google.com/archive/p/word2vec/

Off-the-shelf implementations

https://radimrehurek.com/gensim/models/word2vec.html

Follow-on work

"GloVe: Global Vectors for Word Representation" (Pennington, Socher and Manning, 2014)

https://nlp.stanford.edu/projects/glove/

Many others

15000+ citations

FastText

- “Enriching Word Vectors with Subword Information” Bojanowski et al. (2017; TACL)
- Main idea: learn **character n-gram embeddings** for the target word (not context) and modify the word2vec model to use these
- Pre-trained models in 150+ languages
  - https://fasttext.cc

FastText Details

Main idea: learn **character n-gram embeddings** and for the target word (not the context) modify the word2vec model to use these

Original word2vec:

$$p(c \mid w) \propto \exp(h_c^T v_w)$$

FastText:

$$p(c \mid w) \propto \exp\left(h_c^T \left(\sum_{\text{n−gram } g \text{ in } w} z_g \right) \right)$$

FastText Details

Main idea: learn **character n-gram embeddings** and for the target word (not the context) modify the word2vec model to use these

$$p(c \mid w) \propto \exp\left(h_c^T \left(\sum_{\text{n-gram } g \text{ in } w} z_g \right) \right)$$

$$\texttt{fluffy} \xrightarrow{\textit{decompose}} \texttt{fl flu luf uff ffy fy}$$

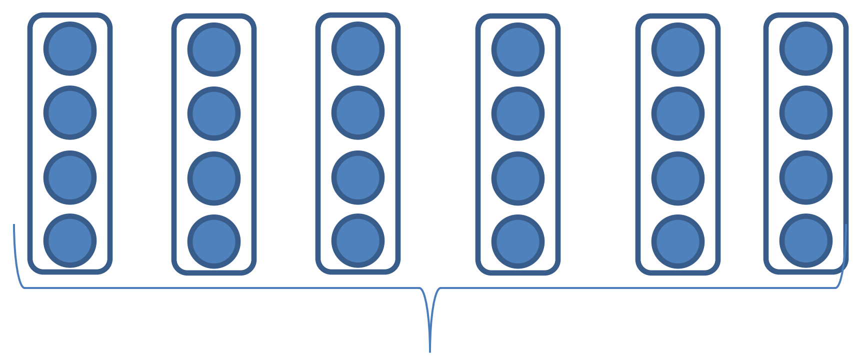
FastText Details

Main idea: learn **character n-gram embeddings** and for the target word (not the context) modify the word2vec model to use these

$$p(c \mid w) \propto \exp\left(h_c^T \left(\sum_{\text{n-gram } g \text{ in } w} z_g\right)\right)$$

decompose

`fluffy` → `fl flu luf uff ffy fy`



Learn n-gram embeddings

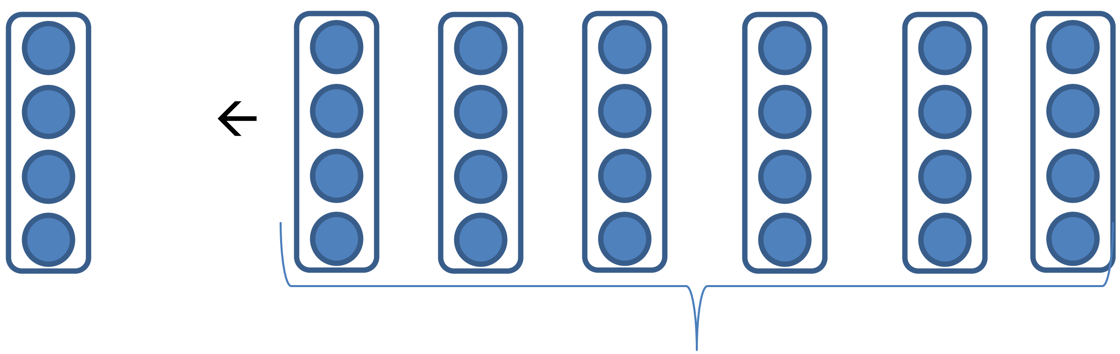
FastText Details

Main idea: learn **character n-gram embeddings** and for the target word (not the context) modify the word2vec model to use these

$$p(c \mid w) \propto \exp\left(h_c^T \left(\sum_{\text{n−gram } g \text{ in } w} z_g \right) \right)$$

decompose

`fluffy` → `fl flu luf uff ffy fy`



To deterministically compute word embeddings

Learn n-gram embeddings


123
SESAME STREET

Contextual Word Embeddings

Word2vec-based models are not context-dependent

- Single word type → single word embedding

If a single word type can have different meanings…

- bank, bass, plant,…

… why should we only have one embedding?

Contextual Word Embeddings

Word2vec-based models are not context-dependent

Single word type → single word embedding

If a single word type can have different meanings…

bank, bass, plant,…

… why should we only have one embedding?

Entire task devoted to classifying these meanings:
Word Sense Disambiguation

Contextual Word Embeddings

Growing interest in this

Off-the-shelf is a bit more difficult

- Download and run a model
- Can't just download a file of embeddings

Two to know about (with code):

- ELMo: "Deep contextualized word representations" Peters et al. (2018; NAACL)
  - https://allennlp.org/elmo

- BERT: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. (2019; NAACL)
  - https://github.com/google-research/bert

Outline

- Continuous representations
  - Motivation
  - Key idea: represent blobs with vectors
- Evaluation
- Common continuous representation models