CS 4650/7650: Natural Language Processing

Vector Semantics

Diyi Yang

Slides from Dan Jurafsky and Michael Collins, and many others

Announcements

- HW1 Regrade Due Jan 29^{th}
- HW2 Due on Feb 3^{rd} , 3pm ET

What are various ways to represent the meaning of a word?

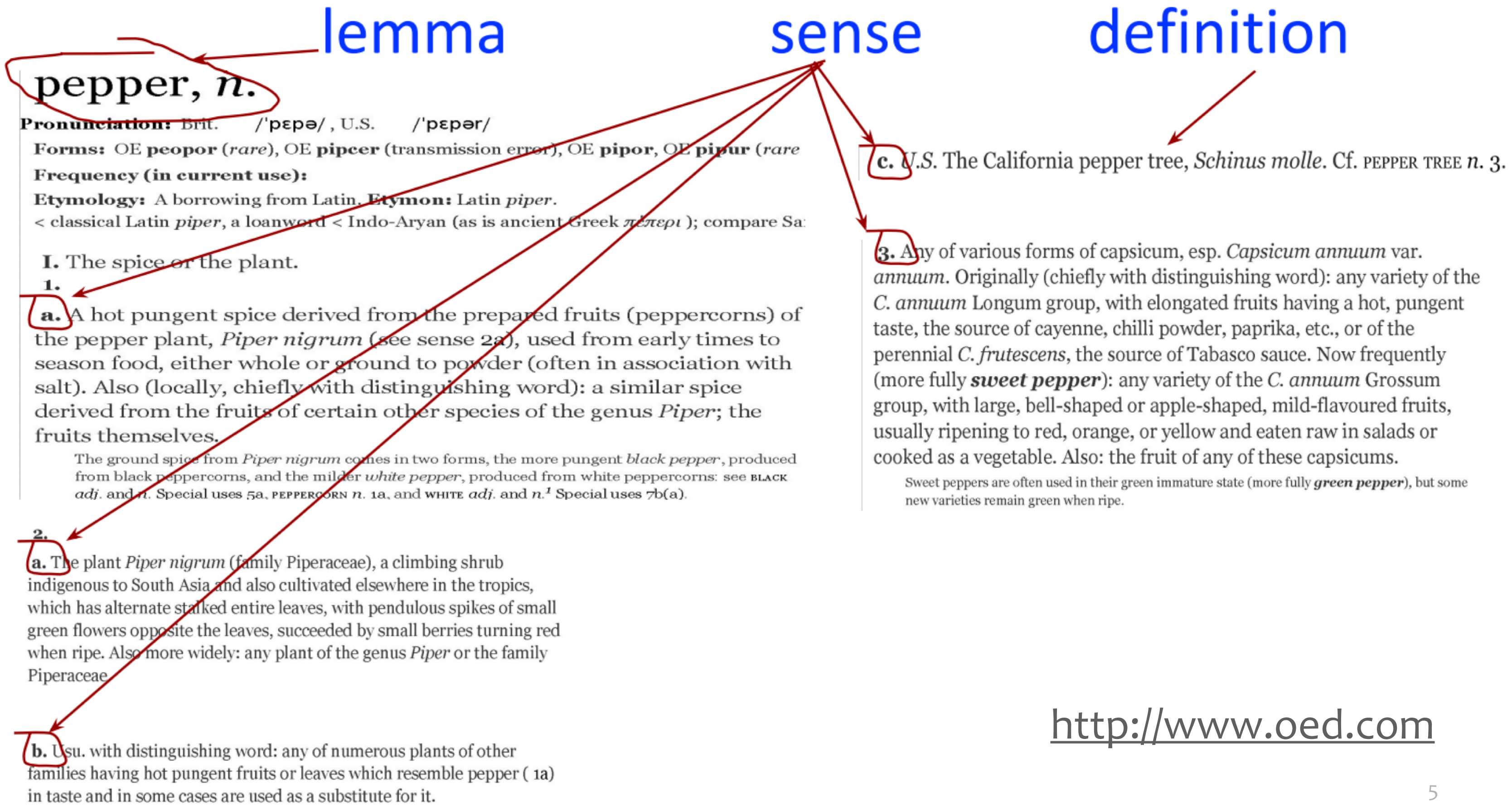
Q: What's the meaning of life?

A: LIFE

Lexical Semantics

How to represent the meaning of a word?

- Words, lemmas, senses, definitions



pepper, ***n.***

Pronunciation: Brit. /ˈpɛpə/ , U.S. /ˈpɛpər/

Forms: OE **peopor** (*rare*), OE **pipcer** (transmission error), OE **pipor**, OE **pipur** (*rare*

Frequency (in current use):

Etymology: A borrowing from Latin. **Etymon:** Latin *piper*.

< classical Latin *piper*, a loanword < Indo-Aryan (as is ancient Greek πέπερι); compare Sa

I. The spice or the plant.

1.

a. A hot pungent spice derived from the prepared fruits (peppercorns) of the pepper plant, *Piper nigrum* (see sense 2a), used from early times to season food, either whole or ground to powder (often in association with salt). Also (locally, chiefly with distinguishing word): a similar spice derived from the fruits of certain other species of the genus *Piper*; the fruits themselves.

The ground spice from *Piper nigrum* comes in two forms, the more pungent *black pepper*, produced from black peppercorns, and the milder *white pepper*, produced from white peppercorns: see BLACK *adj.* and *n.* Special uses 5a, PEPPERCORN *n.* 1a, and WHITE *adj.* and *n.*[1] Special uses 7b(a).

2.

a. The plant *Piper nigrum* (family Piperaceae), a climbing shrub indigenous to South Asia and also cultivated elsewhere in the tropics, which has alternate stalked entire leaves, with pendulous spikes of small green flowers opposite the leaves, succeeded by small berries turning red when ripe. Also more widely: any plant of the genus *Piper* or the family Piperaceae.

b. Usu. with distinguishing word: any of numerous plants of other families having hot pungent fruits or leaves which resemble pepper (1a) in taste and in some cases are used as a substitute for it.

c. *U.S.* The California pepper tree, *Schinus molle*. Cf. PEPPER TREE *n.* 3.

3. Any of various forms of capsicum, esp. *Capsicum annuum* var. *annuum*. Originally (chiefly with distinguishing word): any variety of the *C. annuum* Longum group, with elongated fruits having a hot, pungent taste, the source of cayenne, chilli powder, paprika, etc., or of the perennial *C. frutescens*, the source of Tabasco sauce. Now frequently (more fully ***sweet pepper***): any variety of the *C. annuum* Grossum group, with large, bell-shaped or apple-shaped, mild-flavoured fruits, usually ripening to red, orange, or yellow and eaten raw in salads or cooked as a vegetable. Also: the fruit of any of these capsicums.

Sweet peppers are often used in their green immature state (more fully ***green pepper***), but some new varieties remain green when ripe.

http://www.oed.com

Lemma "Pepper"

- Sense 1:
  - Spice from pepper plant
- Sense 2:
  - The pepper plant itself
- Sense 3:
  - Another similar plant (Jamaican pepper)
- Sense 4:
  - Another plant with peppercorns (California pepper)
- Sense 5:
  - Capsicum (i.e., bell pepper, etc)

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

Lexical Semantics

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

Relation: Synonymity

- Synonyms have the same meaning in some or all contexts.
  - Filbert/hazelnut
  - Couch/sofa
  - Big/large
  - Automobile/car
  - Vomit/throw up
  - Water/H2o

Relation: Synonymity

- Synonyms have the same meaning in some or all contexts.
- Note that there are probably no examples of perfect synonymy
  - Even if some aspects of meaning are identical
  - Still may not preserve the acceptability based on notions of politeness, slang, register, genre, etc.

Relation: Antonymy

- Senses that are opposites with respect to one feature of meaning
  - Otherwise, they are very similar!
    - Dark/light short/long fast/slow rise/fall
    - Hot/cold up/down in/out
- Many formally: antonyms can
  - Define a binary opposition or be at opposite ends of a scale
    - Long/short, fast/slow
  - Be reverse:
    - Rise/fall, up/down

Relation: Similarity

- Words with similar meanings
- Not synonyms, but sharing some element of meaning
  - Car, bicycle
  - Cow, horse

Ask Humans How Similar 2 Words Are

Word 1	Word 2	similarity
vanish	disappear	9.8
behave	obey	7.3
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

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

Relation: Word Relatedness

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

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

Semantic Field

Hospitals

- Surgeon, scalpel, nurse, anesthetic, hospital

Restaurants

- Waiter, menu, plate, food, menu, chef

Houses

- Door, roof, kitchen, family, bed

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

Relation: Word Relatedness

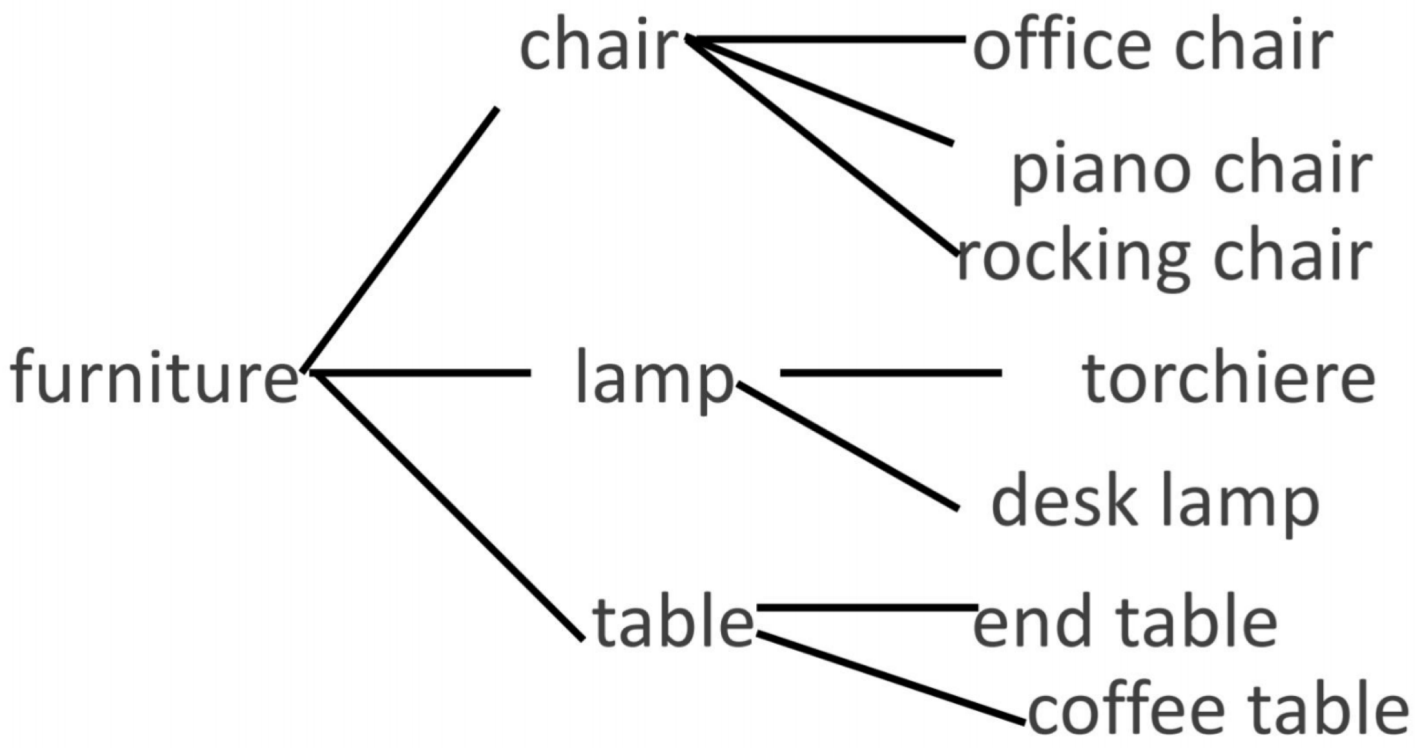
- Also called “word association”
- Words be related in any way, perhaps via a semantic field
  - Car, bicycle: **similar**
  - Car, gas: **related**, not similar
  - Coffee, cup: **related**, not similar

Relation: Superordinate/Subordinate

- One sense is a subordinate of another if the first sense is more specific, denoting a subclass of the other
  - *Car is a subordinate of vehicle*
  - *Mango is a subordinate of fruit*
- Conversely superordinate
  - *Vehicle is a superordinate of car*
  - *Fruit is a superordinate of mango*

Taxonomy

Superordinate *Basic* *Subordinate*



Lexical Semantics

- How should we represent the meaning of the word?
  - Words, lemmas, senses, definitions
  - Relationships between words or senses
  - Taxonomy relationships
  - Word similarity, word relatedness

Lexical Semantics

- How should we represent the meaning of the word?
  - Words, lemmas, senses, definitions
  - Relationships between words or senses
  - Taxonomy relationships
  - Word similarity, word relatedness
  - Semantic frames and roles

Semantic Frame

- A set of words that denote perspectives or participants in a particular type of event
  - “buy” (the event from the perspective of the buyer)
  - “sell” (from the perspective of the seller)
  - “pay” (focusing on the monetary aspect)
  - *John hit Bill*
  - *Bill was hit by John*

- Frames have semantic roles (like buyer, sellers, goods, money) and words in a sentence can take on those roles

Lexical Semantics

- How should we represent the meaning of the word?
  - Words, lemmas, senses, definitions
  - Relationships between words or senses
  - Taxonomy relationships
  - Word similarity, word relatedness
  - Semantic frames and roles
  - Connotation and sentiment

Connotation and Sentiment

- Connotations refer to the aspects of a word's meaning that are related to a writer or reader's emotions, sentiment, opinions, or evaluations.
  - happy vs. sad
  - great, love vs. terrible, hate
- Three dimensions of affective meaning
  - Valence: the pleasantness of the stimulus
  - Arousal: the intensity of emotion
  - 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

Lexical Semantics

- How should we represent the meaning of the word?
  1. Words, lemmas, senses, definitions
  2. Relationships between words or senses
  3. Taxonomy relationships
  4. Word similarity, word relatedness
  5. Semantic frames and roles
  6. Connotation and sentiment

Electronic Dictionaries

WordNet

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

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

(here, for *good*):

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

Problems with Discrete Representation

- Too coarse
  - Expert → skillful
- Sparse
  - Wicked, badass, ninja
- Subjective
- Expensive
- Hard to compute word relationships

expert [0 0 0 **1** 0 0 0 0 0 0 0 0 0 0 0]

skillful [0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0]

Vector Semantics

Distributional Hypothesis

- "The meaning of a word is its use in the language"

  [Wittgenstein PI 43]

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

  [Firth 1957]

- "If A and B have almost identical environments we say that they are synonyms"

  [Harris 1954]

Example: What does OngChoi Mean?

- Suppose you see those sentences:
  - Ongchoi is delicious **sautéed with garlic**
  - Ongchoi is superb **over rice**
  - Ongchoi **leaves** with salty sauces
- And you've also seen these:
  - … spinach **sautéed with garlic over rice**
  - Chard stems and **leave**s are delicious
  - Collard greens and other **salty** leafy greens

Example: What does OngChoi Mean?

- Suppose you see those sentences:
  - Ongchoi is delicious **sautéed with garlic**
  - Ongchoi is superb **over rice**
  - Ongchoi **leaves** with salty sauces
- And you've also seen these:
  - … spinach **sautéed with garlic over rice**
  - Chard stems and **leave**s are delicious
  - Collard greens and other **salty** leafy greens

Word Embedding Representations

- Count-based
  - Tf-idf, PPMI
- Class-based
  - Brown Clusters
- Distributed prediction-based embeddings
  - Word2vec, FastText
- Distributed contextual (token) embeddings from language models
  - Elmo, BERT
- + many more variants
  - Multilingual embeddings, multi-sense embeddings, syntactic embeddings, etc …

Term-Document Matrix

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	17
solider	2	80	62	89
fool	36	58	1	4
clown	20	15	2	3

Context = appearing in the same document.

Term-Document Matrix

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	17
solider	2	80	62	89
fool	36	58	1	4
clown	20	15	2	3

Vector Space Model:
Each document is represented as a column vector of length four

Term-Context Matrix / Word-Word Matrix

	knife	dog	sword	love	like
knife	0	1	6	5	5
dog	1	0	5	5	5
sword	6	5	0	5	5
love	5	5	5	0	5
like	5	5	5	5	2

Two words are "similar" in meaning if their context vectors are similar.

- Similarity == relatedness

Count-Based Representations

	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

Counts: term-frequency

- Remove stop words
- Use $\log_{10}(tf)$
- Normalize by document length

TF-IDF

- What to do with words that are evenly distributed across many documents?

$$tf_{t,d} = \log_{10}(\text{count}(t, d) + 1)$$

Total # of docs in collection

$$idf_i = \log_{10}\left(\frac{N}{df_i}\right)$$

# of docs that have word i

TF-IDF

- What to do with words that are evenly distributed across many documents?

$$tf_{t,d} = \log_{10}(\text{count}(t,d) + 1)$$

$$idf_i = \log_{10}\left(\frac{N}{df_i}\right)$$

Total # of docs in collection

# of docs that have word i

- Words like "the" or "good" have very low idf

$$w_{t,d} = tf_{t,d} \times idf_i$$

Pointwise Mutual Information (PMI)

- Do word w and c co-occur more than if they were independent?

$$\mathrm{PMI}(w, c) = \log_2 \frac{p(w, c)}{p(w)p(c)}$$

Positive Pointwise Mutual Information (PPMI)

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

Positive Pointwise Mutual Information (PPMI)

- PMI is biased toward infrequent events
  - Very rare words have very high PMI values
  - Give rare words slightly higher probabilities α=0.75

$$\text{PPMI}_{\alpha}(w,c) = max(\log_2 \frac{p(w,c)}{p(w)p_{\alpha}(c)}, 0)$$

$$\text{P}_{\alpha}(c) = \frac{count(c)^{\alpha}}{\sum_c count(c)^{\alpha}}$$

Sparse versus Dense Vectors

- PPMI vectors are
  - **Long** (length |V| = 20,000 to 50,000)
  - **Sparse** (most elements are zero)

- Alternative: learn vectors which are
  - **Short** (length 200-1000)
  - **Dense** (most elements are non-zero)

Why Dense Vectors

- Short vectors may be easier to use as features (less weights to tune)
- Dense vectors may generalize better than storing explicit counts
- They may do better at capturing synonymy
  - *Car and automobile are synonyms, but are represented as distinct dimensions; this fails to capture similarity between a word with car as a neighbor and a word with automobile as a neighbor.*

- In practice, they work better

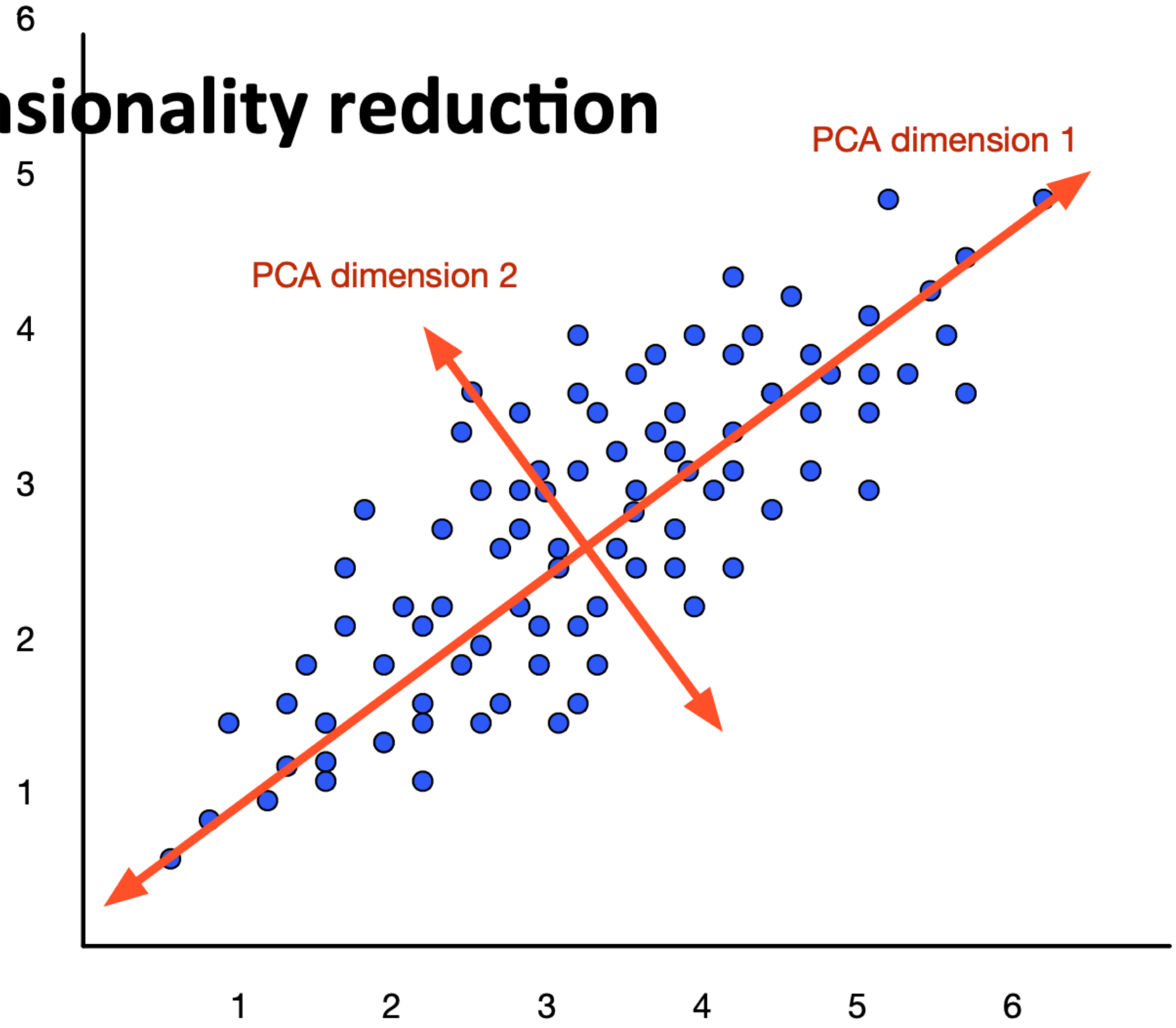
Three Methods for Getting Short Dense Vectors

- Singular Value Decomposition (SVD)
  - A special case of this is called LSA – Latent Semantic Analysis
- Brown Clustering
- “Neural Language Model” – inspired predictive models
  - Skip-grams and CBOW

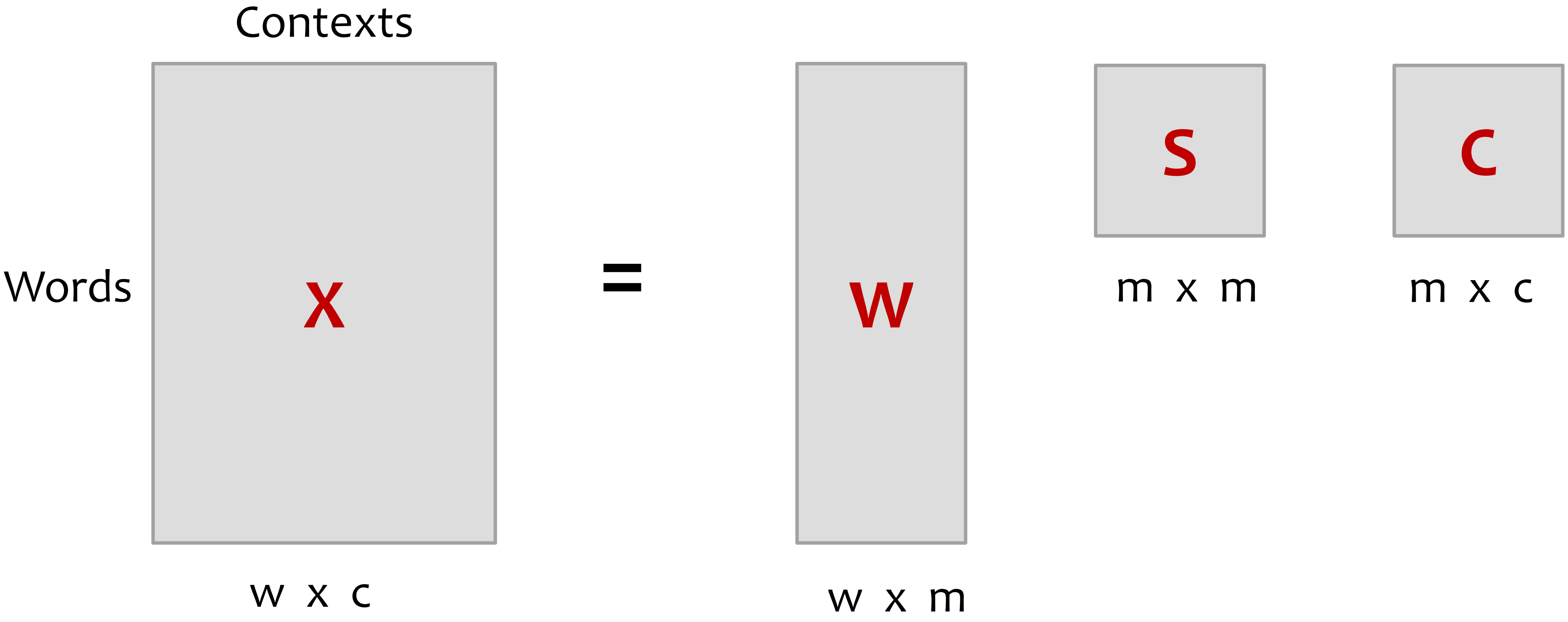
Dense Vectors via SVD

- Intuition
  - Approximate an N-dimensional dataset using fewer dimensions
  - By first rotating the axes into a new space
  - The highest order dimension captures the most variance in the original dataset
  - And the next dimension captures the next most variance, etc
  - Many such (related) methods:
    - PCA - principle components analysis
    - Factor analysis
    - SVD

Dimensionality reduction



Singular Value Decomposition (SVD)



Singular Value Decomposition (SVD)

Contexts

Words **X** = **W** **S** **C**

w x c

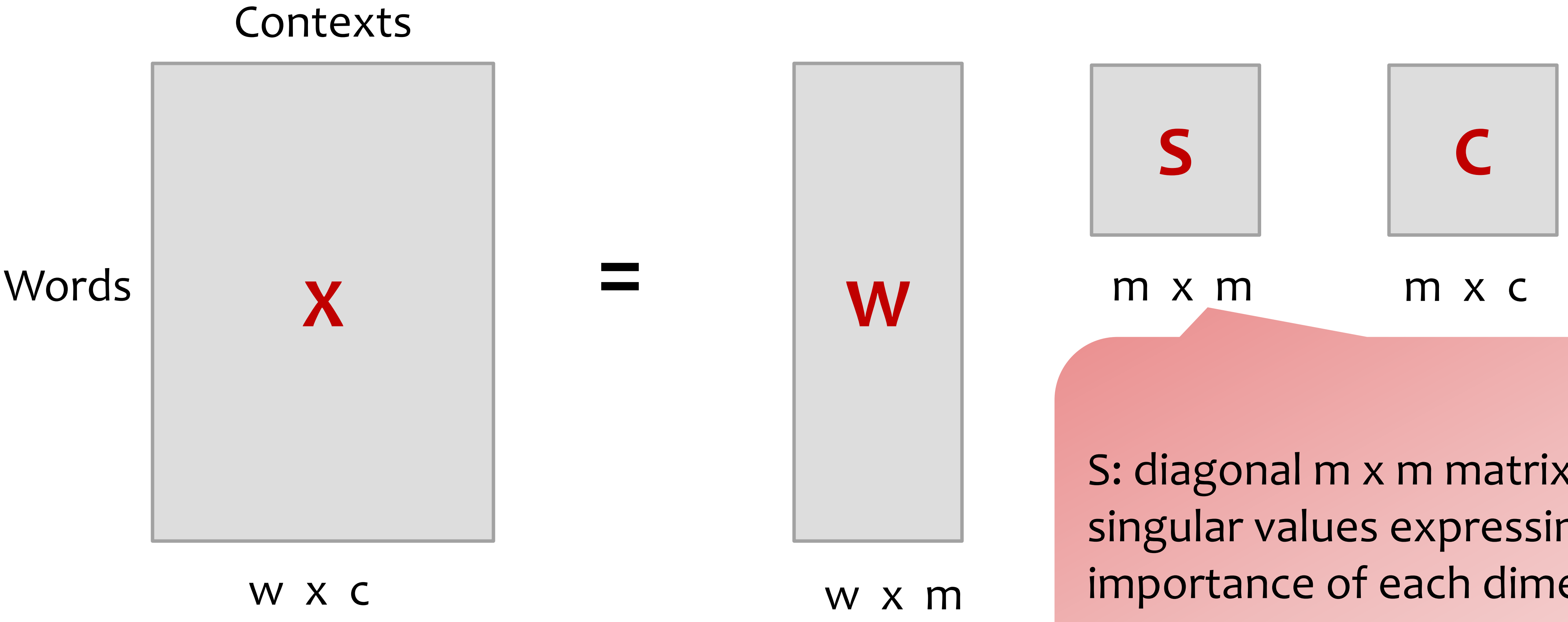
w x m

m x m

m x c

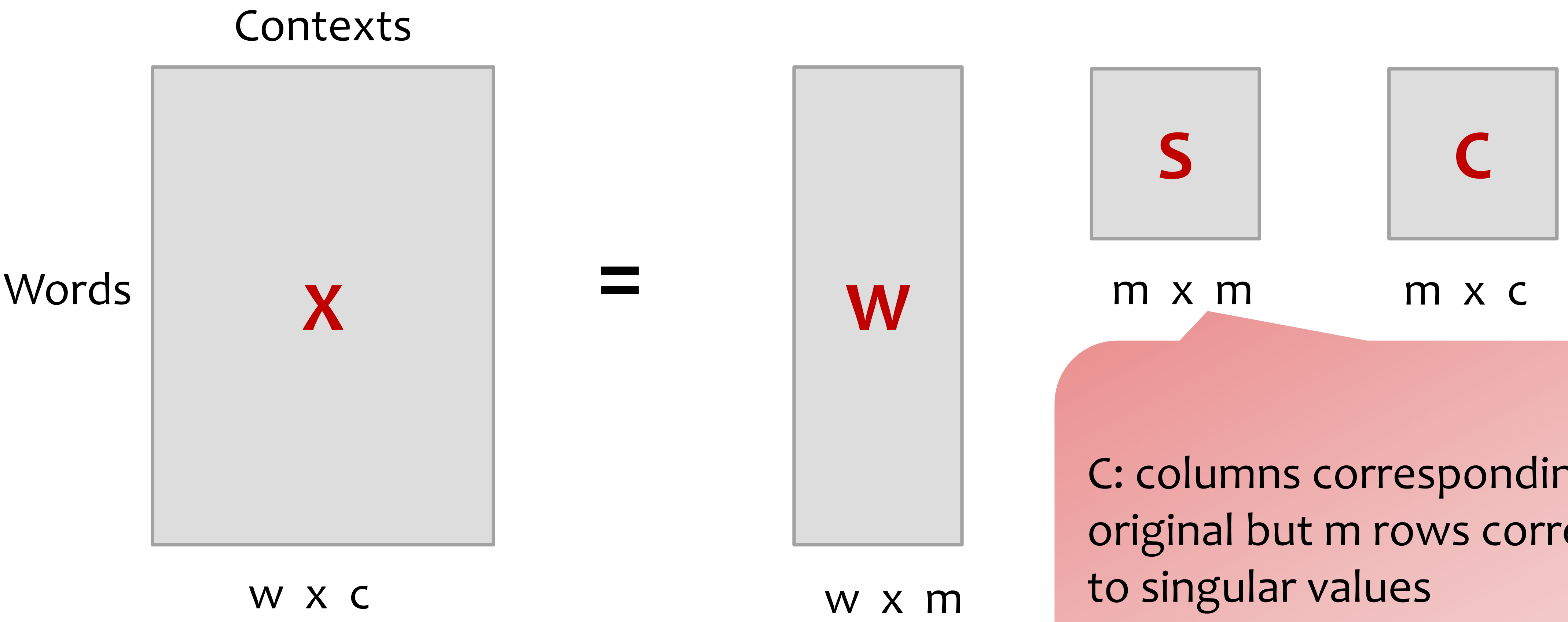
W: rows corresponding to original but m columns represents a dimension in a new latent space, such that (1) m column vectors are orthogonal to each other, and (2) columns are ordered by the amount of variance in the dataset each new dimension accounts for

Singular Value Decomposition (SVD)



S: diagonal m x m matrix of singular values expressing the importance of each dimension

Singular Value Decomposition (SVD)



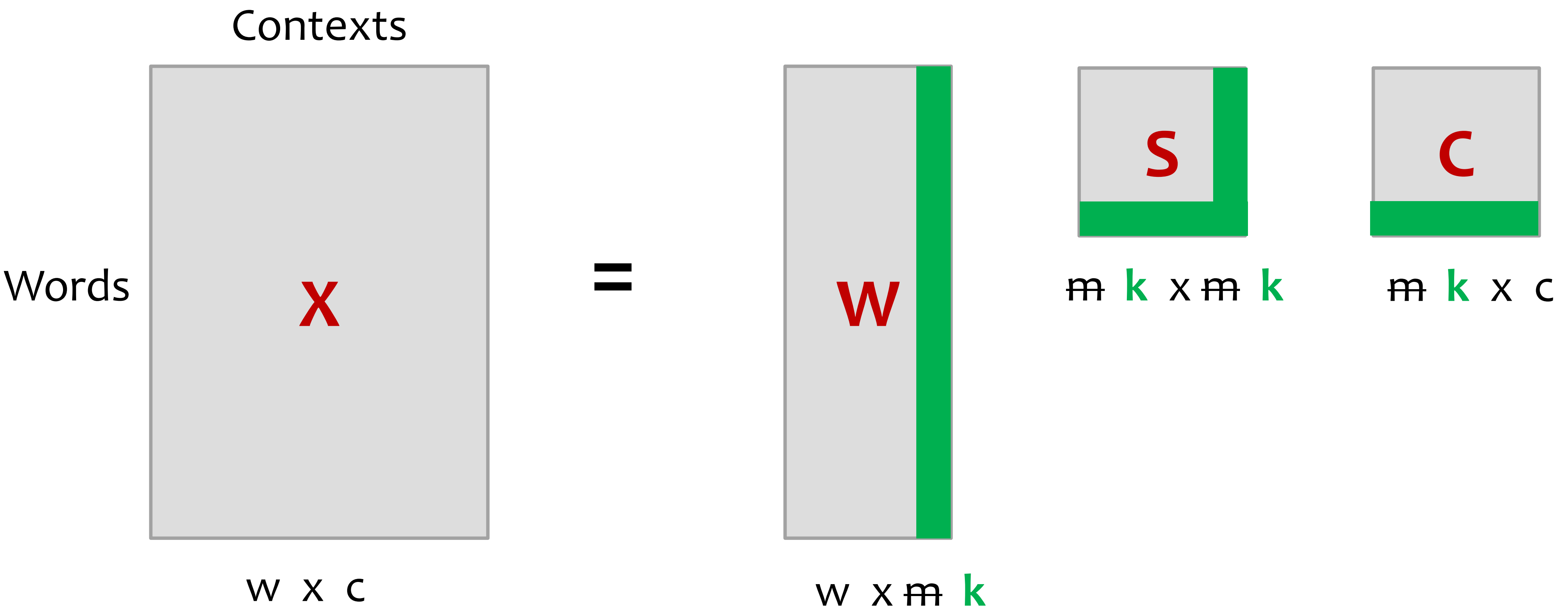
C: columns corresponding to original but m rows corresponding to singular values

SVD Applied to Term-Document Matrix: Latent Semantic Analysis

- If instead of keeping all m dimensions, we just keep the top k singular values. Let's say 300.

- The result is a least-square approximation to the original X
- But instead of multiplying, we'll just make use of W

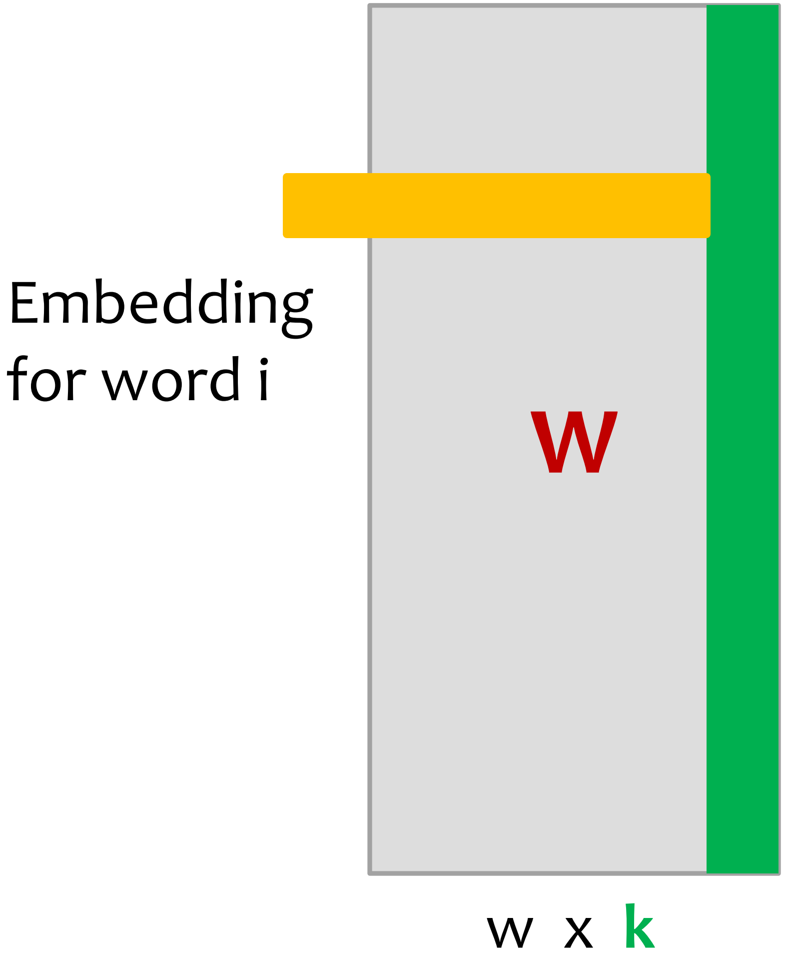
Truncated SVD



Truncated SVD Produces Embeddings

- Each row of W is a k-dimensional representation of each word w
- K might range from 50 to 100

- Generally we keep the top k dimensions, but some experiments suggest that getting rid of the top 1 dimension or even the top 50 dimensions is helpful



Embeddings versus Sparse Vectors

- Dense SVD embeddings sometimes work better than sparse PPMI matrices at tasks like word similarity
  - Denoising: low-order dimensions may represent unimportant information
  - Truncation may help the models generalize better to unseen data
  - Having a smaller number of dimensions may make it easier for classifiers to properly weight the dimensions for the task
  - Dense models may do better at capturing higher order cooccurrence

Word Similarity

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

Word Embedding Representations

- Count-based
  - Tf-idf, PPMI
- Class-based
  - Brown Clusters
- Distributed prediction-based embeddings
  - Word2vec, Fasttext
- Distributed contextual (token) embeddings from language models
  - Elmo, BERT
- + many more variants
  - Multilingual embeddings, multi-sense embeddings, syntactic embeddings, etc …

The Brown Clustering Algorithm

- Input: a large collection of words
- Output 1: a partition of words into word clusters
- Output 2 (generalization of 1): a hierarchical word clustering

The Brown Clustering Algorithm

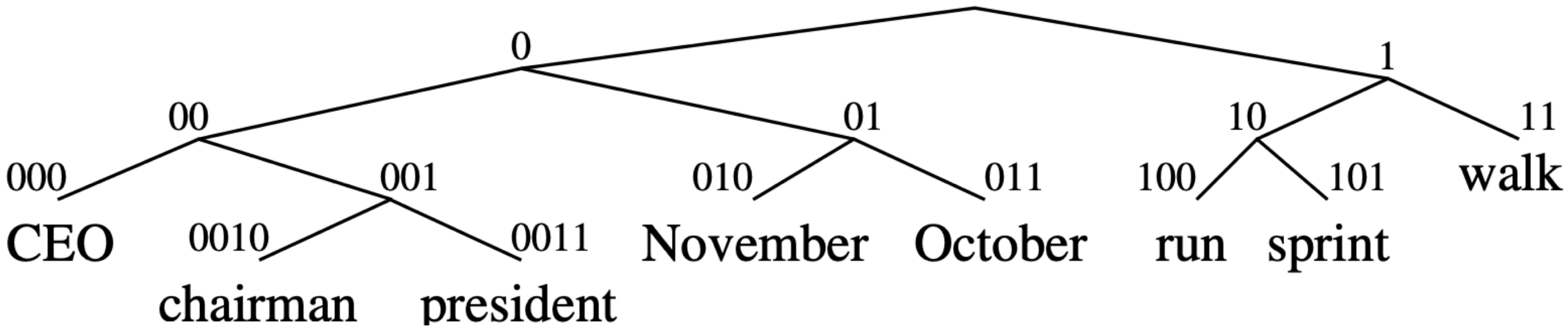
- An agglomerative clustering algorithm that clusters words based on which words precede or follow them
- These word clusters can be turned into a kind of vector
- We'll give a very brief sketch here

Brown Clustering Algorithm

- Each word is initially assigned to its own cluster.
- We now consider merging each pair of clusters. Highest quality merge is chosen.
  - Quality = merges two words that have similar probabilities of preceding and following words
- Clustering proceeds until all words are in one big cluster

Brown Clusters as Vectors

- By tracing the order in which clusters are merged, the model builds a binary tree from bottom to top.
- Each word represented by binary string = path from root to leaf
- Each intermediate node is a cluster
- Chairman = 0010, “months” = 01, and verbs = 1



Brown Clustering Example

A Sample Hierarchy (from Miller et al., NAACL 2004)

lawyer	1000001101000
newspaperman	100000110100100
stewardess	100000110100101
toxicologist	10000011010011
slang	1000001101010
babysitter	100000110101100
conspirator	1000001101011010
womanizer	1000001101011011
mailman	10000011010111
salesman	100000110110000
bookkeeper	1000001101100010
troubleshooter	10000011011000110
bouncer	10000011011000111
technician	1000001101100100
janitor	1000001101100101
saleswoman	1000001101100110
...	
Nike	1011011100100101011100
Maytag	10110111001001010111010
Generali	10110111001001010111011
Gap	1011011100100101011110
Harley-Davidson	10110111001001010111110
Enfield	101101110010010101111110
genus	101101110010010101111111
Microsoft	10110111001001011000
Ventritex	101101110010010110010
Tractebel	1011011100100101100110
Synopsys	1011011100100101100111
WordPerfect	1011011100100101101000
....	
John	101110010000000000
Consuelo	101110010000000001
Jeffrey	101110010000000010
Kenneth	10111001000000001100
Phillip	101110010000000011010
WILLIAM	101110010000000011011
Timothy	10111001000000001110
Terrence	101110010000000011110

Brown Clustering Example

Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays Saturdays
June March July April January December October November September August
people guys folks fellows CEOs chaps doubters commies unfortunates blokes
down backwards ashore sideways southward northward overboard aloft downwards adrift
water gas coal liquid acid sand carbon steam shale iron
great big vast sudden mere sheer gigantic lifelong scant colossal
man woman boy girl lawyer doctor guy farmer teacher citizen
American Indian European Japanese German African Catholic Israeli Italian Arab
pressure temperature permeability density porosity stress velocity viscosity gravity tension
mother wife father son husband brother daughter sister boss uncle
machine device controller processor CPU printer spindle subsystem compiler plotter
John George James Bob Robert Paul William Jim David Mike
anyone someone anybody somebody
feet miles pounds degrees inches barrels tons acres meters bytes
director chief professor commissioner commander treasurer founder superintendent dean custodian

from Brown et al., 1992

Intuition

Similar words appear in similar contexts

Similar words have similar distribution of words to their immediate left and right

Brown Clustering

- $\mathcal{V}$ is a vocabulary
- $C : \mathcal{V} \rightarrow \{1, 2, \ldots k\}$ is a partition of the vocabulary into k clusters
- $q(C(w_i)|C(w_{i-1}))$ is a probability of cluster w_i of to follow the cluster of w_{i-1}
- $\mathrm{e}(w_i | C(w_i)) = \frac{count(w_i)}{\sum_{x \in C(w_i)} count(x)}$

$$p(w_1, w_2, \ldots, w_T) = \prod_{i=1}^{n} e(w_i|C(w_i))q(C(w_i)|C(w_{i-1}))$$

Brown Clustering

- $\mathcal{V}$ is a vocabulary
- $C : \mathcal{V} \rightarrow \{1, 2, \ldots k\}$ is a partition of the vocabulary into k clusters
- $q(C(w_i)|C(w_{i-1}))$ is a probability of cluster w_i of to follow the cluster of w_{i-1}
- $\mathrm{e}(w_i|C(w_i)) = \frac{count(w_i)}{\sum_{x \in C(w_i)} count(x)}$

Quality(C) $= \prod_{i=1}^{n} e(w_i|C(w_i))q(C(w_i)|C(w_{i-1}))$

An Example

$$p(w_1, w_2, \ldots w_n) = \prod_{i=1}^{n} e(w_i | C(w_i)) q(C(w_i) | C(w_{i-1}))$$

An Example

$$p(w_1, w_2, \dots w_n) = \prod_{i=1}^{n} e(w_i | C(w_i)) q(C(w_i) | C(w_{i-1}))$$

$$C(\mathsf{the}) = 1, \quad C(\mathsf{dog}) = C(\mathsf{cat}) = 2, \quad C(\mathsf{saw}) = 3$$

An Example

$$p(w_1, w_2, \ldots w_n) = \prod_{i=1}^{n} e(w_i | C(w_i)) q(C(w_i) | C(w_{i-1}))$$

$$C(\textsf{the}) = 1, \quad C(\textsf{dog}) = C(\textsf{cat}) = 2, \quad C(\textsf{saw}) = 3$$

$$e(\textsf{the}|1) = 1, \quad e(\textsf{cat}|2) = e(\textsf{dog}|2) = 0.5, \quad e(\textsf{saw}|3) = 1$$

An Example

$$p(w_1, w_2, \ldots w_n) = \prod_{i=1}^{n} e(w_i | C(w_i)) q(C(w_i) | C(w_{i-1}))$$

$$C(\textsf{the}) = 1, \quad C(\textsf{dog}) = C(\textsf{cat}) = 2, \quad C(\textsf{saw}) = 3$$

$$e(\textsf{the}|1) = 1, \quad e(\textsf{cat}|2) = e(\textsf{dog}|2) = 0.5, \quad e(\textsf{saw}|3) = 1$$

$$q(1|0) = 0.2, \quad q(2|1) = 0.4, \quad q(3|2) = 0.3, \quad q(1|3) = 0.6$$

An Example

$$p(w_1, w_2, \dots w_n) = \prod_{i=1}^{n} e(w_i | C(w_i)) q(C(w_i) | C(w_{i-1}))$$

$$C(\textsf{the}) = 1, \quad C(\textsf{dog}) = C(\textsf{cat}) = 2, \quad C(\textsf{saw}) = 3$$

$$e(\textsf{the}|1) = 1, \quad e(\textsf{cat}|2) = e(\textsf{dog}|2) = 0.5, \quad e(\textsf{saw}|3) = 1$$

$$q(1|0) = 0.2, \quad q(2|1) = 0.4, \quad q(3|2) = 0.3, \quad q(1|3) = 0.6$$

$$p(\textsf{the dog saw the cat}) =$$

The Brown Clustering Model

- $\mathcal{V}$ is a vocabulary
- $C : \mathcal{V} \rightarrow \{1, 2, \ldots k\}$ is a partition of the vocabulary into k clusters
- $q(C(w_i)|C(w_{i-1}))$ is a probability of cluster w_i of to follow the cluster of w_{i-1}
- $\mathrm{e}(w_i|C(w_i)) = \frac{count(w_i)}{\sum_{x \in C(w_i)} count(x)}$

$$\text{Quality(C)} = \prod_{i=1}^{n} e(w_i|C(w_i))q(C(w_i)|C(w_{i-1}))$$

How to Measure the Quality of C?

- How do we measure the quality of a partition C?

$$\text{Quality}(C) = \sum_{i=1}^{n} \log e(w_i|C(w_i))q(C(w_i)|C(w_{i-1}))$$

$$= \sum_{c=1}^{k}\sum_{c'=1}^{k} p(c,c') \log \frac{p(c,c')}{p(c)p(c')} + G$$

a constant

- Where $p(c,c') = \frac{n(c,c')}{\sum_{c,c'} n(c,c')}$ $p(c) = \frac{n(c)}{\sum_c n(c)}$

Here, n(c) is the number of times class c occurs in the corpus, n(c, c') is the number of times c' is seen following c, under the function C

Notes P45: https://www-cs.stanford.edu/~pliang/papers/meng-thesis.pdf

A First Algorithm

- Start with |V| clusters: each word gets its own cluster
- The goal is to get k clusters
- We run |V|-k merge steps:
  - Pick 2 clusters and merge them
  - Each step picks the merge maximizing Quality(C)
- Cost ?
  - $O(|V|-k) \times O(|V|^2) \times O(|V|^2) = O(|V|^5)$

    # iters   # pairs   compute Quality(C)

A Second Algorithm

- m: a hyper-parameter, sort words by frequency
- Take the top m most frequent words, put each of them in its own cluster $c_1, c_2, c_3, \ldots c_m$
- For $i = (m+1) \ldots |V|$
  - Create a new cluster c_{m+1} *(we have $m+1$ clusters)*
  - Choose two clusters from $m+1$ clusters based on quality(C) and merge *(back to m clusters)*
- Carry out $m-1$ final merges *(full hierarchy)*
- Running time $O(|V|m^2 + n)$, n=#words in corpus

Next Class

- Word2vec, FastText
- Elmo, BERT, XLNet
- Multilingual Embeddings

Additional Notes On Brown Clustering

How to Measure the Quality of C?

- $n(w)$ be the number of times word w appears in the text.
- $n(w, w')$ be the number of times the bigram (w, w') occurs in the text.
- $n(c) = \sum_{w \in c} n(w)$ be the number of times a word in a cluster c appears in the text
- $n(c, c') = \sum_{w \in c, w^i \in c^i} n(w, w')$
- n is simply the length of the text

How to Measure the Quality of C?

$$
\begin{aligned}
\text{Quality}(C) &= \frac{1}{n}\sum_{i=1}^{n} \log P(C(w_i)|C(w_{i-1}))P(w_i|C(w_i)) \\
&= \sum_{w,w'} \frac{n(w,w')}{n} \log P(C(w')|C(w))P(w'|C(w')) \\
&= \sum_{w,w'} \frac{n(w,w')}{n} \log \frac{n(C(w),C(w'))}{n(C(w))}\frac{n(w')}{n(C(w'))} \\
&= \sum_{w,w'} \frac{n(w,w')}{n} \log \frac{n(C(w),C(w'))n}{n(C(w))n(C(w'))} + \sum_{w,w'} \frac{n(w,w')}{n} \log \frac{n(w')}{n} \\
&= \sum_{c,c'} \frac{n(c,c')}{n} \log \frac{n(c,c')n}{n(c)n(c')} + \sum_{w'} \frac{n(w')}{n} \log \frac{n(w')}{n}
\end{aligned}
$$

How to Measure the Quality of C?

Define

$$P(w) = \frac{n(w)}{n} \quad P(c) = \frac{n(c)}{n}; \quad P(c, c') = \frac{n(c,c')}{n}$$

$$\begin{aligned} \text{Quality}(C) &= \sum_{c,c'} P(c, c') \log \frac{P(c, c')}{P(c)P(c')} + \sum_{w} P(w) \log P(w) \\ &= I(C) - H \end{aligned}$$

mutual information
between adjacent clusters

entropy of
the word distribution