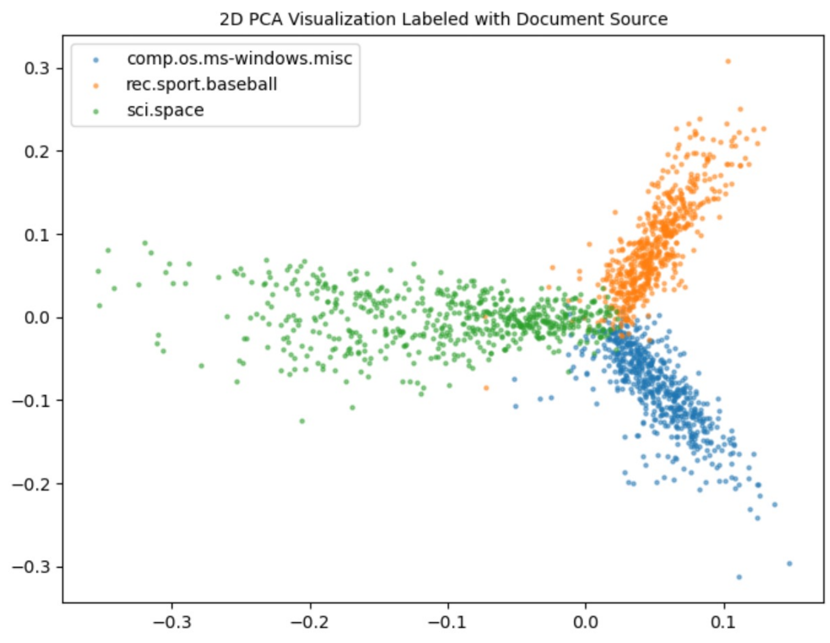
CS 505: Introduction to Natural Language Processing

Wayne Snyder
Boston University

Lecture 7: Vector Models Continued: Principal Component Analysis; Distributional semantics; Word embeddings and word2vec; Sentence and text embeddings



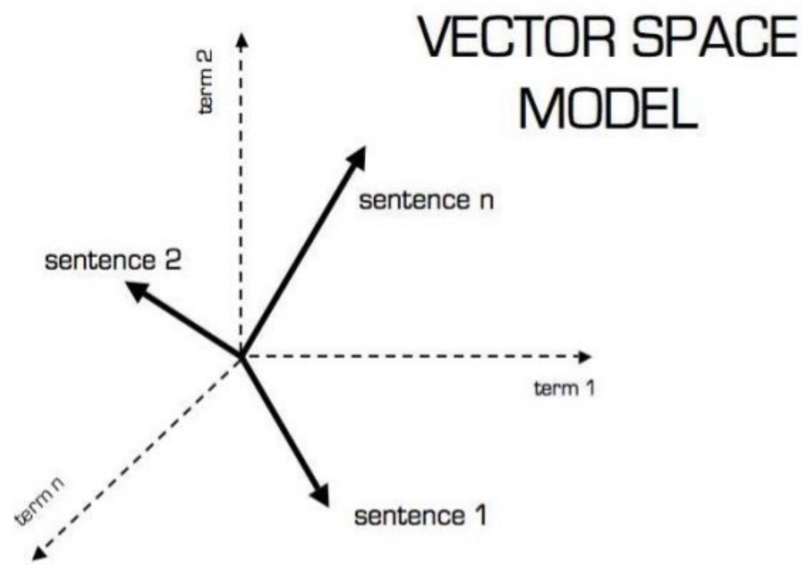
Text as vectors (TF BOW, TF-IDF, etc….)

V = Vocabulary
|V| = Size of V

- So we have a |V|-dimensional vector space
- Words/tokens are the axes of the space
- Sentences, documents etc. are points or vectors in this space
- Very high-dimensional: Number of dimensions = size of vocabulary,

  often 10,000 or more.
- These are very sparse vectors

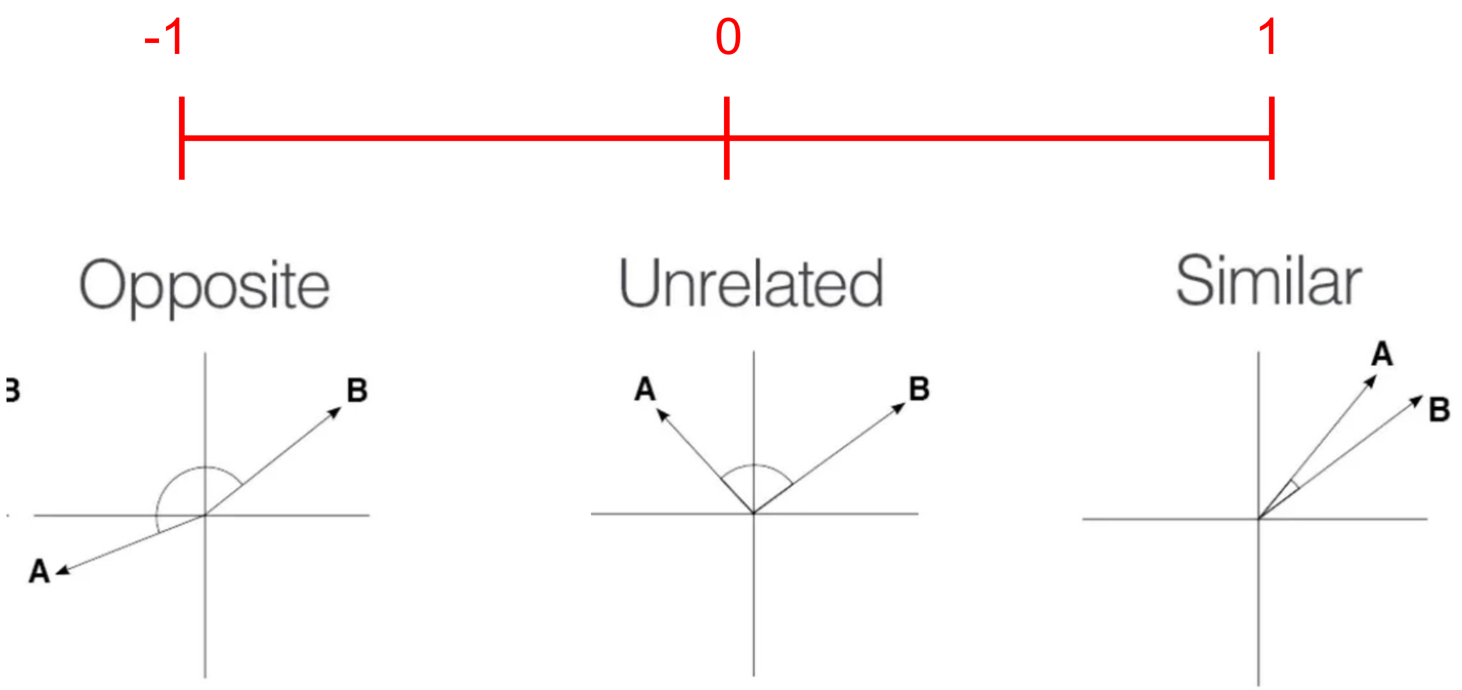
  - most entries are zero.



Cosine Similarity

Recall: Cosine Similarity is a measure of how similar two objects in vector space are, as the cosine of the angle between them, irrespective of their magnitude. The scale is [-1 .. 1].

$$\text{cosine similarity} = S_C(A, B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2 \cdot \sum\limits_{i=1}^{n} B_i^2}},$$



An Example: A Vector Space Model for Characters in PotC

BOW for will turner:

Word	Frequency
jack	22
will	12
find	8
get	7
elizabeth	6
know	5
us	5
euh	5
come	5
need	5
key	5
ship	5
hold	5
pearl	4
stop	4
sparrow	4
chest	4
back	4
away	4
leave	3

BOW for jack sparrow:

Word	Frequency
want	15
come	11
know	9
oh	9
will	9
bugger	8
dirt	8
love	8
hey	7
one	7
mate	6
captain	6
key	6
would	6
jones	6
save	6
jar	6
chest	6
much	5
way	5

BOW for gibbs:

Word	Frequency
jack	11
aye	9
us	8
will	6
ho	5
sea	5
got	4
think	4
bit	4
like	4
cast	4
rum	3
captain	3
seems	3
key	3
cap'n	3
back	3
made	3
chief	3
believe	3

BOW for elizabeth swann:

Word	Frequency
will	22
jack	12
find	7
know	7
oh	7
want	6
man	6
good	5
something	5
sparrow	4
would	4
chance	4
us	3
captain	3
compass	3
give	3
going	3
came	3
way	3
yes	3

BOW for lord cutler beckett:

Word	Frequency
jack	7
sparrow	7
one	5
will	5
compass	5
mister	4
turner	4
freedom	3
world	3
must	3
something	3
currency	3
governor	2
mercer	2
arrest	2
william	2
oh	2
believe	2
question	2
perhaps	2

BOW for ragetti:

Word	Frequency
well	4
unh-unh	3
divine	2
providence	2
us	2
immortal	2
souls	2
got	2
save	2
thief/thing	2
give	2
back	2
ha-ha-ha	2
pulling	2
wants	2

BOW for pintel:

Word	Frequency
ai	3
mooring	3
lines	3
us	3
mind	3
aye	3
read	2
must	2
come	2
well	2
back	2
would	2
could	2
heart	2
chest	2
o'	2
think	2

BOW for davy jones:

Word	Frequency
ha	12
will	7
huh	4
captain	4
jack	4
sparrow	4
let	4
fear	3
offer	3
one	3
years	3
three	3
liar	3
chest	3
death	2
depths	2

An Example: A Vector Space Model for Characters in PotC

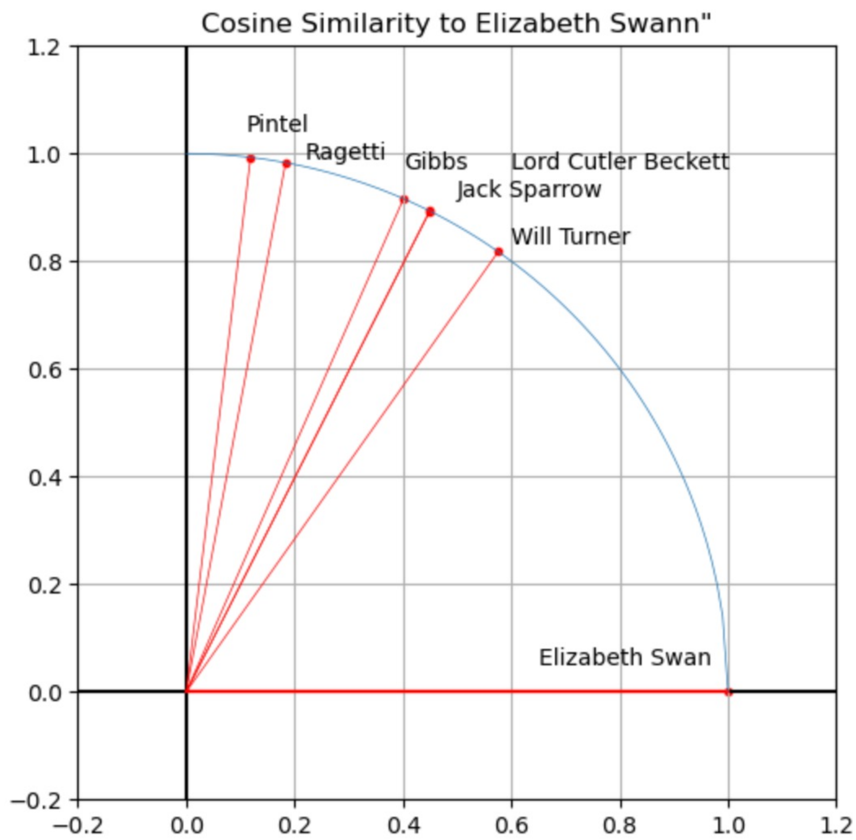
First, let's just calculate the cosine similarity between each character's BOW.

Cosine Similarity	Degrees	Characters	
0.5758	54.84	elizabeth swann	will turner
0.4909	60.6	gibbs	will turner
0.4497	63.28	elizabeth swann	lord cutler beckett
0.4493	63.3	elizabeth swann	jack sparrow
0.4203	65.15	jack sparrow	will turner
0.401	66.36	elizabeth swann	gibbs
0.3982	66.53	lord cutler beckett	will turner
0.3618	68.79	davy jones	will turner
0.344	69.88	gibbs	jack sparrow
0.3383	70.23	davy jones	elizabeth swann
0.3236	71.12	jack sparrow	lord cutler beckett
0.3093	71.98	davy jones	lord cutler beckett
0.3021	72.42	gibbs	lord cutler beckett
0.2823	73.6	gibbs	pintel
0.2807	73.7	pintel	ragetti
0.2752	74.03	davy jones	gibbs
0.2717	74.23	davy jones	jack sparrow
0.2709	74.28	gibbs	ragetti
0.2408	76.07	jack sparrow	pintel
0.2357	76.37	jack sparrow	ragetti
0.22	77.29	ragetti	will turner
0.1834	79.43	elizabeth swann	ragetti
0.1826	79.48	pintel	will turner
0.1545	81.11	lord cutler beckett	ragetti
0.1326	82.38	davy jones	ragetti
0.1187	83.18	elizabeth swann	pintel
0.0962	84.48	lord cutler beckett	pintel
0.0936	84.63	davy jones	pintel

An Example: A Vector Space Model for Characters in PotC

Here is each character's cosine similarity to Elizabeth Swann:

Cosine Similarity	Degrees	Characters	
0.5758	54.8443	elizabeth swann	will turner
0.4497	63.2756	elizabeth swann	lord cutler beckett
0.4493	63.3012	elizabeth swann	jack sparrow
0.401	66.3593	elizabeth swann	gibbs
0.1834	79.4321	elizabeth swann	ragetti
0.1187	83.1829	elizabeth swann	pintel



Digression: Dimensionality Reduction using PCA

The total vocabulary size for this task is 1249 words; thus the cosine similarity for character vectors takes place in 1249 dimensions!

It is impossible to get any intuition for so many dimensions!

Principal Components Analysis

- Factor the term-document matrix using singular value decomposition:

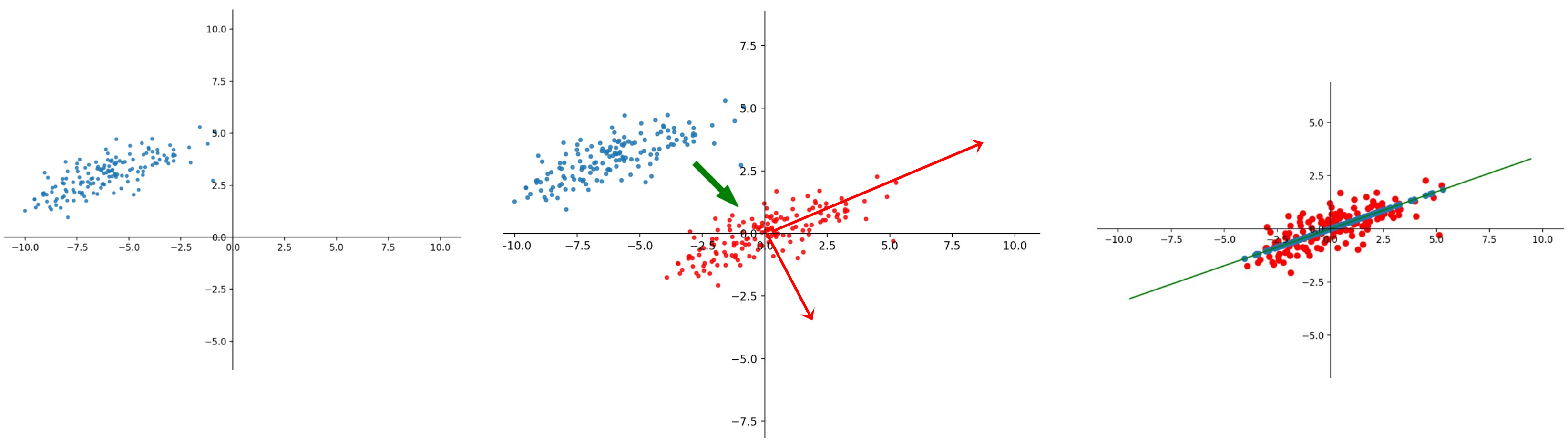
$$m\text{ documents}\left\{ \overbrace{\begin{bmatrix} w_{11} & \cdots & w_{1j} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{i1} & \cdots & w_{ij} & \cdots & w_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{m1} & \cdots & w_{mj} & \cdots & w_{mn} \end{bmatrix}}^{n\text{ words}} \right. = \overbrace{\begin{bmatrix} \vdots & & \vdots \\ \vdots & & \vdots \\ \mathbf{u}_1 & \cdots & \mathbf{u}_r \\ \vdots & & \vdots \\ \vdots & & \vdots \end{bmatrix}}^{r} \times \underbrace{\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \times \begin{bmatrix} \cdots & \cdots & \mathbf{v}_1 & \cdots & \cdots \\ & & \vdots & & \\ \cdots & \cdots & \mathbf{v}_r & \cdots & \cdots \end{bmatrix}}$$

- The eigenvectors are the principal components of the data and the eigenvalues σ_1 ... σ_r give the “importance” of each componenent.

Digression: Dimensionality Reduction using PCA

Principal Components Analysis

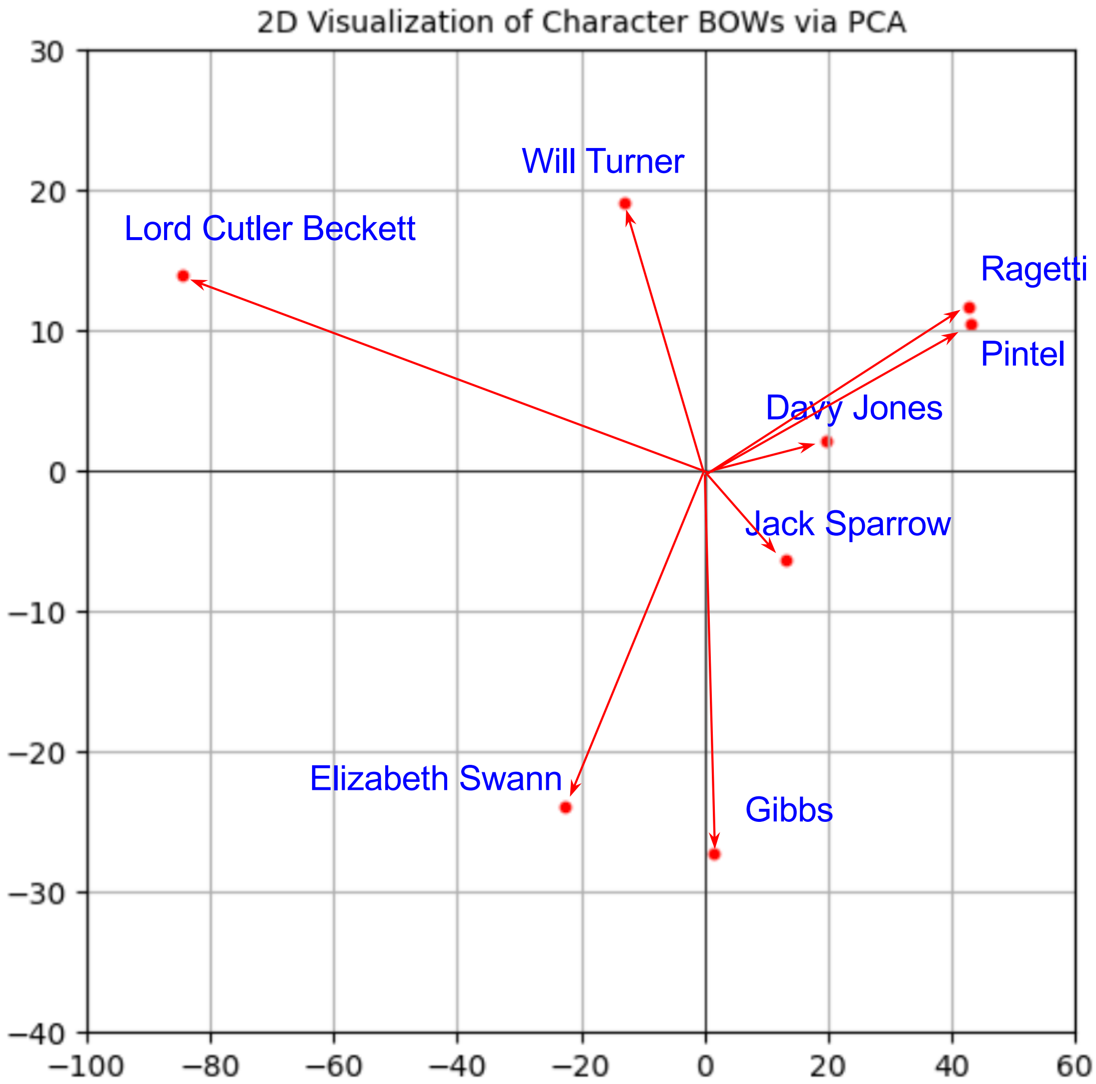
- The principal components tell you which parts of the data are more significance (have more variance)
- The largest two principal components give us a 2D view of the data;
- Here is an illustration of reducing a 2D data set to 1 dimension:



End of digression!

Vector Space Models for Characters in PotC

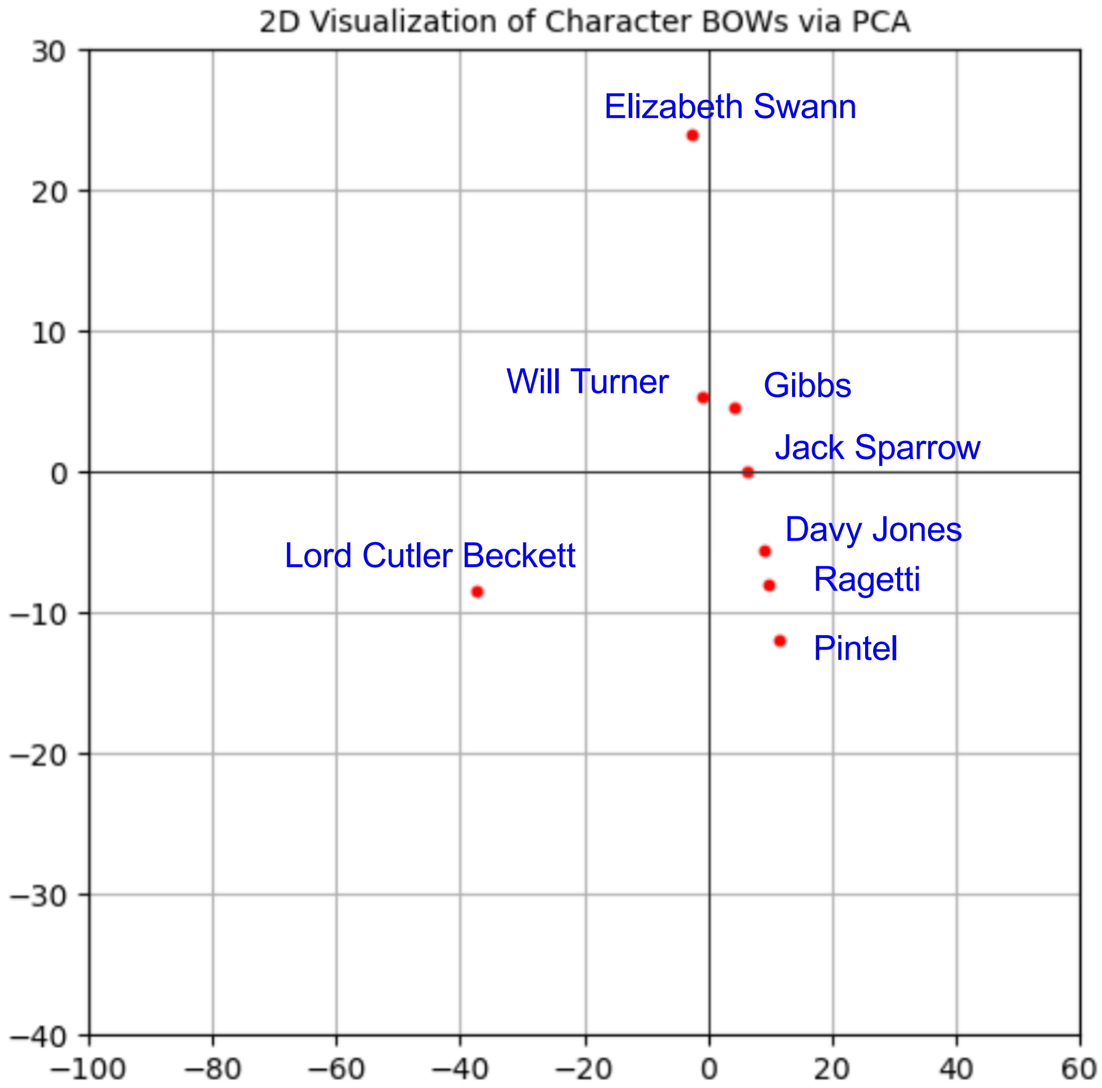
Just using Term Frequency Counts (stop words NOT deleted)



I've reduced a 15 x 1249 matrix to 15 x 2 and displayed the rows in 2D!

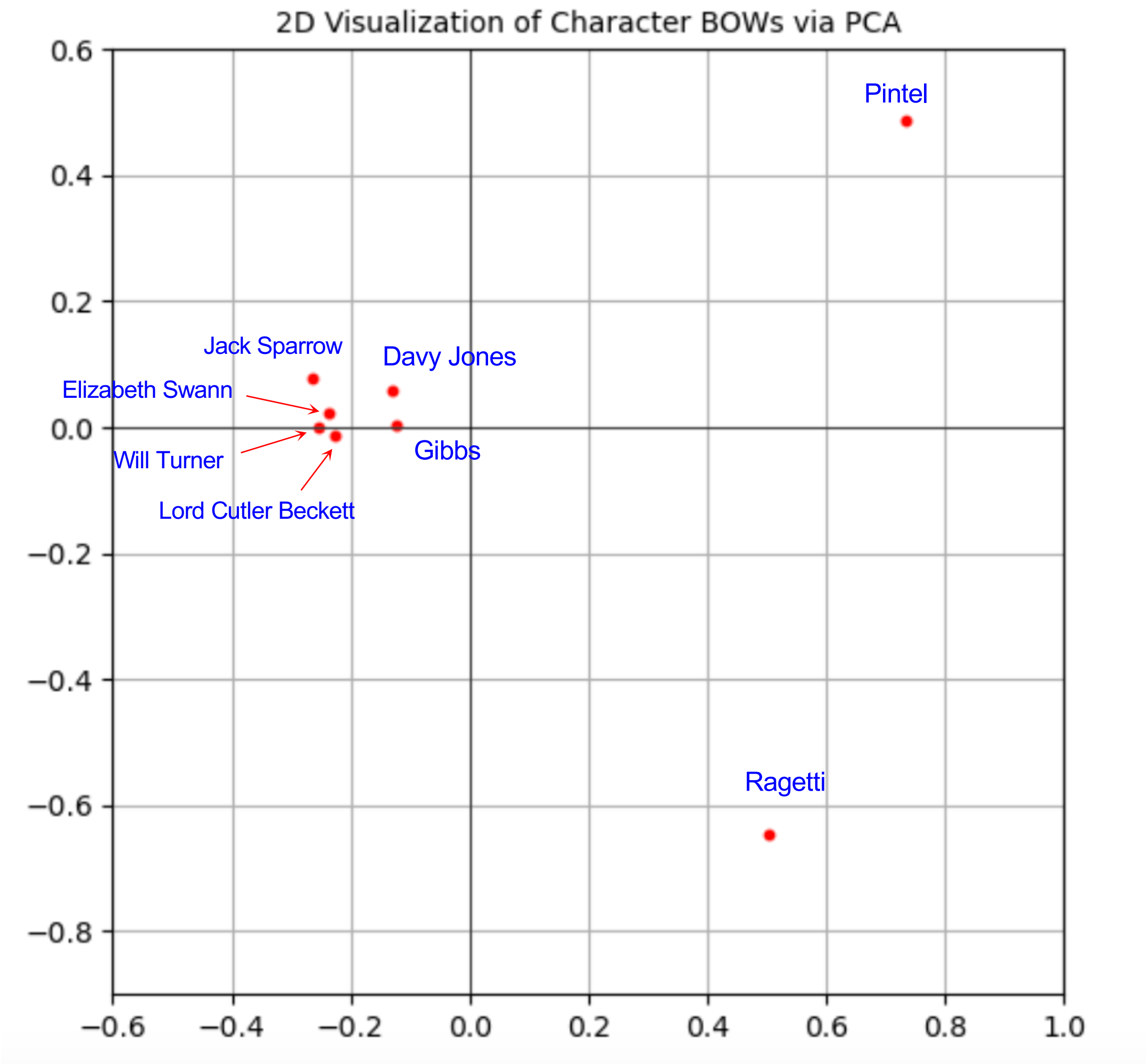
Vector Space Models for Characters in PotC

Just using Term Frequency Counts (with stop words deleted)



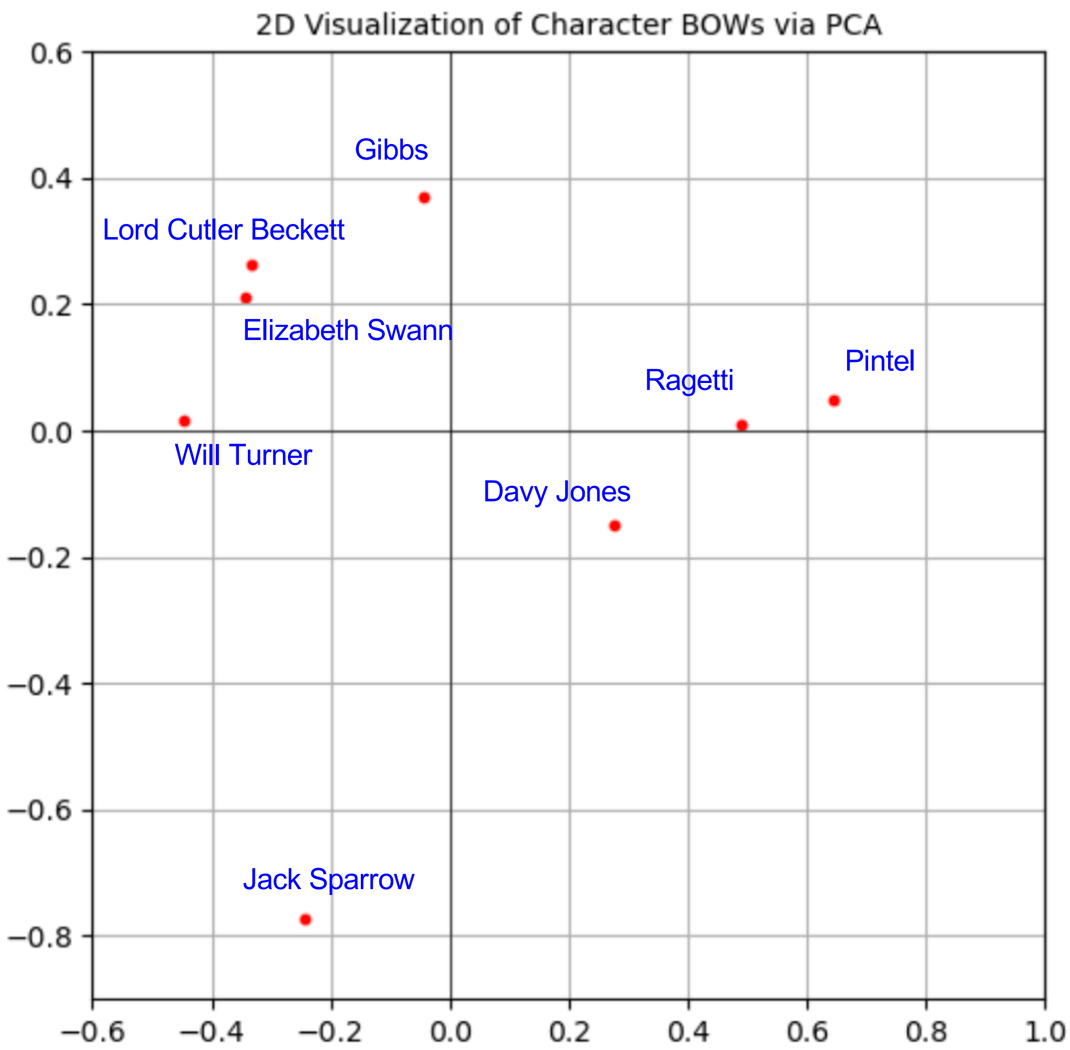
Vector Space Models for Characters in PotC

Just using TF-IDF (stop words NOT deleted)



Vector Space Models for Characters in PotC

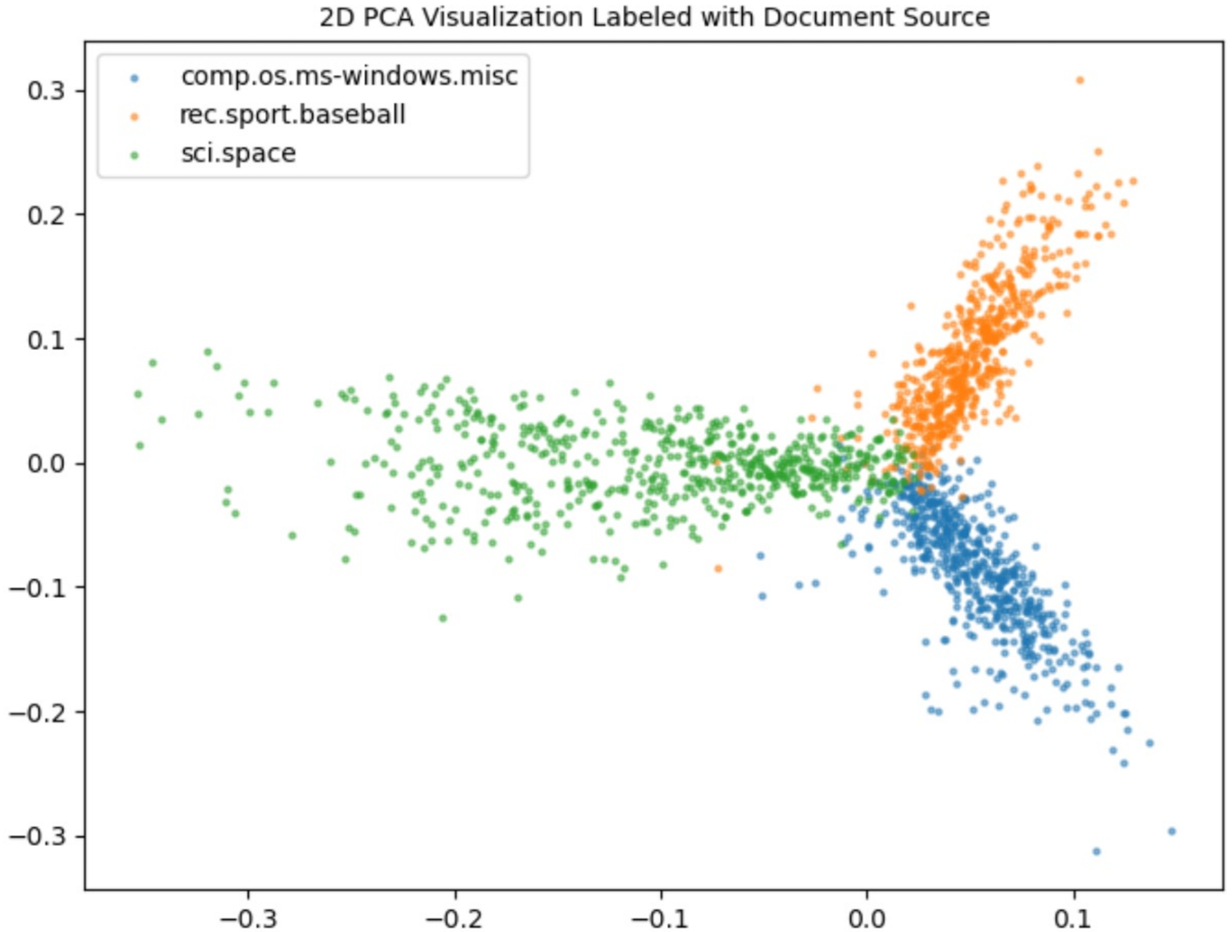
Just using TF-IDF (with stop words deleted)



Why do you think there is not more consistency between these different views of the data?

Vector Space Models for Social Media

```
from sklearn.datasets import fetch_20newsgroups
categories = ['comp.os.ms-windows.misc', 'sci.space', 'rec.sport.baseball']
news_data = fetch_20newsgroups(subset='train', categories=categories)
```



What's wrong with Term-Frequency Vector Models?

1. Inefficient: Huge sparse vectors (could be 100,000 words!), mostly 0's.
   - They don't scale well!
2. They don't generalize well:
   - TF models don't generalize across NLP tasks and domains
   - Can't be used for transfer learning
3. Statistics $\neq$ Semantics:
   - What words occur together?
   - How are they used?
   - What do words mean?
4. Can't handle complex linguistic phenomena:
   - Synonyms: car and automobile have no relationship!
   - Polysemy (multiple meanings): sound = audio signal, healthy, body of water (19 noun meanings, 12 adjective meanings, 12 verb meanings, etc.)

What's wrong with Term-Frequency Vector Models?

Short, dense vectors are a better solution!

- Short: 50-1000 dimensions
- Dense: Most elements are not 0
- Efficient: Easier to use as features in machine learning
- Generalize better than explicit counts
- May capture synonymy better, since fewer dimensions
- In general, to capture semantics better!

Vector Models of Meaning: Short and Dense

Naïve idea: Meaning as a point in space (Osgood et al. 1957)

Define a word by abstract characteristics:

- 3 affective dimensions for a word
  - valence: pleasantness
  - arousal: intensity of emotion
  - dominance: the degree of control exerted
- Hence the connotation of a word is a vector in 3-space

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

But this doesn't generalize well to all English words: what about nouns?

Vector Models of Meaning: Short and Dense

Embeddings are short and dense word/text vectors

- **“Neural Language Model”-inspired embeddings**
  - Word2vec (skipgram, CBOW), GloVe
- **Singular Value Decomposition (SVD)**
  - A special case of this is called LSA – Latent Semantic Analysis
- **Alternative to these "static embeddings":**
  - Contextual Embeddings (ELMo, BERT)
  - Compute distinct embeddings for a word in its context
  - Separate embeddings for each token of a word

Vector Models of Meaning: Embeddings

How to capture the meaning of words? By context!

Ludwig Wittgenstein:

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

Distributional Semantics:

“You shall know a word by the company it keeps” (Firth 1959)

Words are defined by the words around them;

Synonyms can be substituted into the same contexts:

“The fast car was speeding down the road.”

“The fast automobile was speeding down the road.”

Vector Models of Meaning: Embeddings

What does recent English borrowing *ongchoi* mean?

- **Suppose you see these sentences:**
  - Ong choi is delicious **sautéed with garlic**.
  - Ong choi is superb **over rice**
  - Ong choi **leaves** with salty sauces

- **And you've also seen these:**
  - …spinach **sautéed with garlic over rice**
  - Chard stems and **leaves** are **delicious**
  - Collard greens and other **salty** leafy greens
- **Conclusion:**
  - Ongchoi is a leafy green like spinach, chard, or collard greens
    - We could conclude this based on words like "leaves" and "delicious" and "sauteed"

Vector Models of Meaning: Embeddings

Simple way to think about embeddings: A word is placed in vector space by its context:

A skip-gram is a context (+/- N) before and after a word:

(2,2) Skip-Gram

The fast car was speeding

A word context is a set of words nearby (i.e., +/- N words):

Word	Word context:
car	{fast, speeding , the, was }

Words have similar meanings iff they have similar contexts.

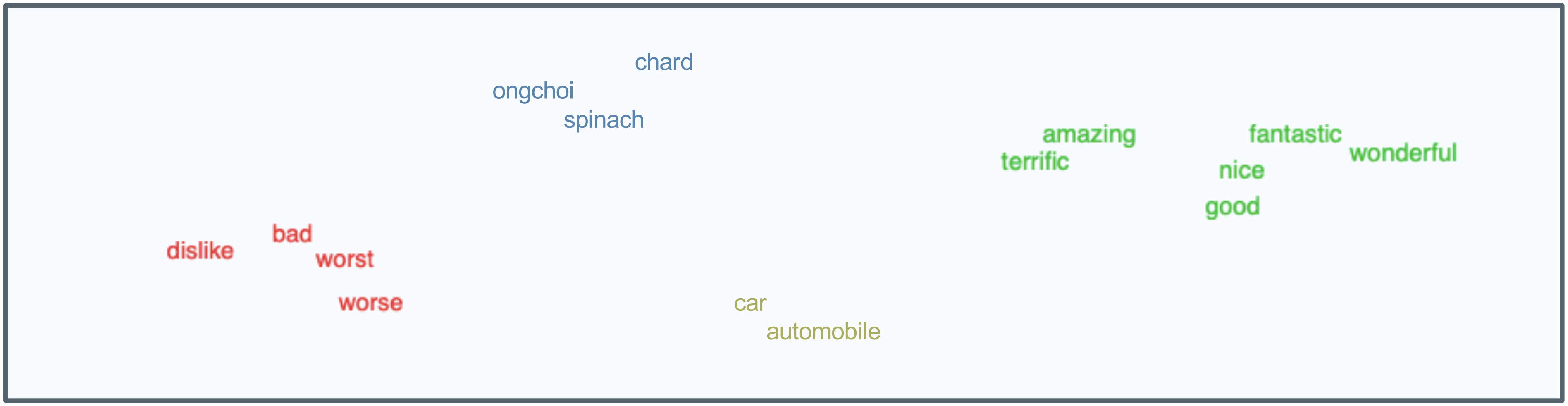
Vector Models of Meaning: Embeddings

This is a machine learning task (we'll return to it).

- Start with a random D-dimensional vectors as a word's initial embedding
- Train a classifier based on embedding similarity
- Take a corpus and take pairs of words that co-occur as positive examples
- Take pairs of words that don't co-occur as negative examples
- Train the classifier to distinguish these by slowly adjusting all the embeddings to improve the classifier performance
- Throw away the classifier code and keep the embeddings.

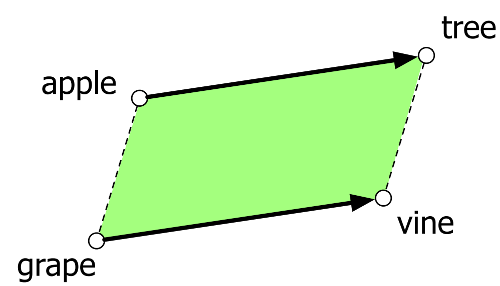
Vector Models of Meaning: Embeddings

Embeddings keep similar words near each other and dissimilar words far apart.



They also allow a simple form of analogical reasoning:

To solve: "Apple is to tree as grape is to _____?"



- Use vector arithmetic:

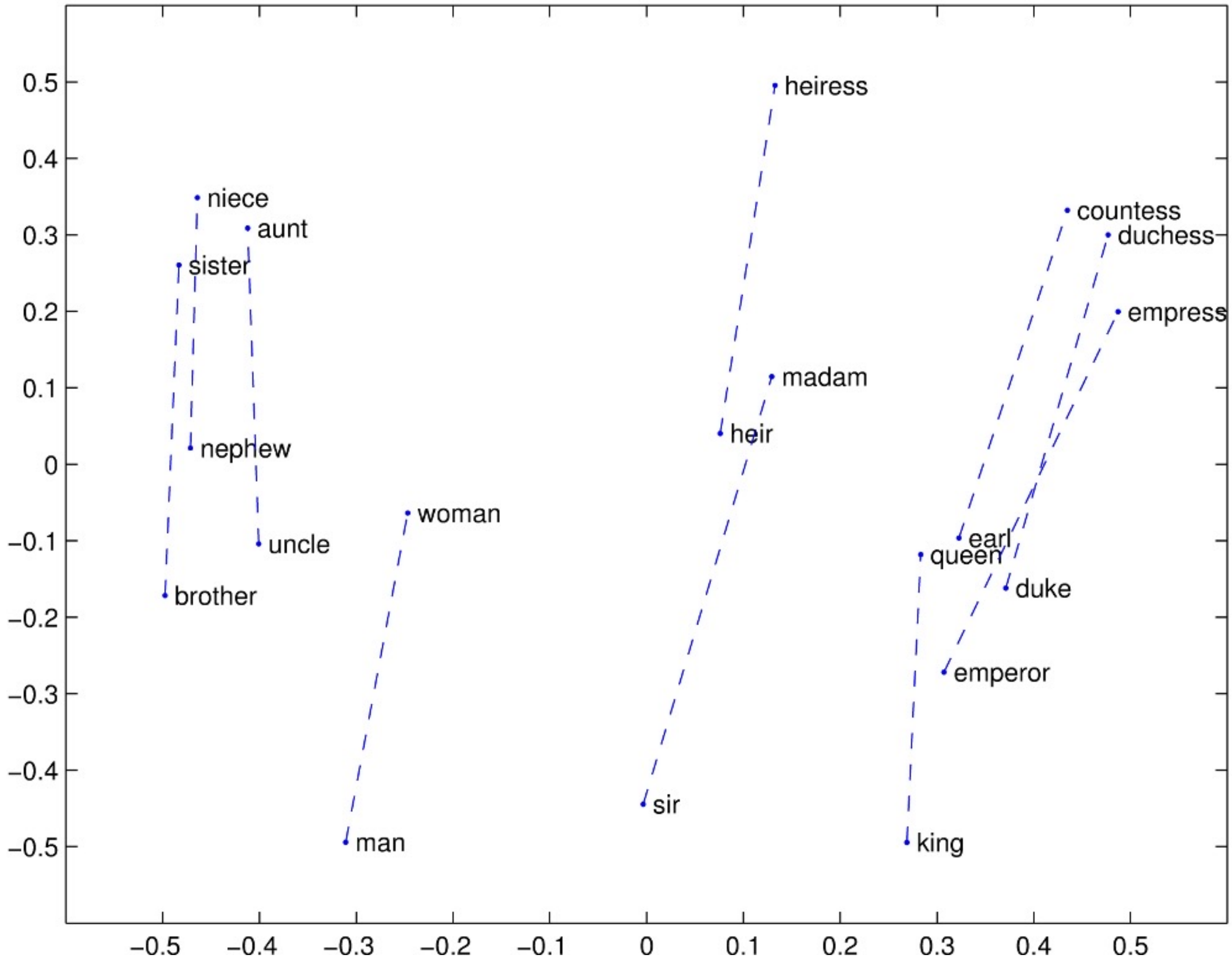
  X = tree – apple + grape

- Search for the closest word (using cosine sim):

  argmin_X(cosine_sim(tree – apple + grape, X))

=> vine

Vector Models of Meaning: Embeddings

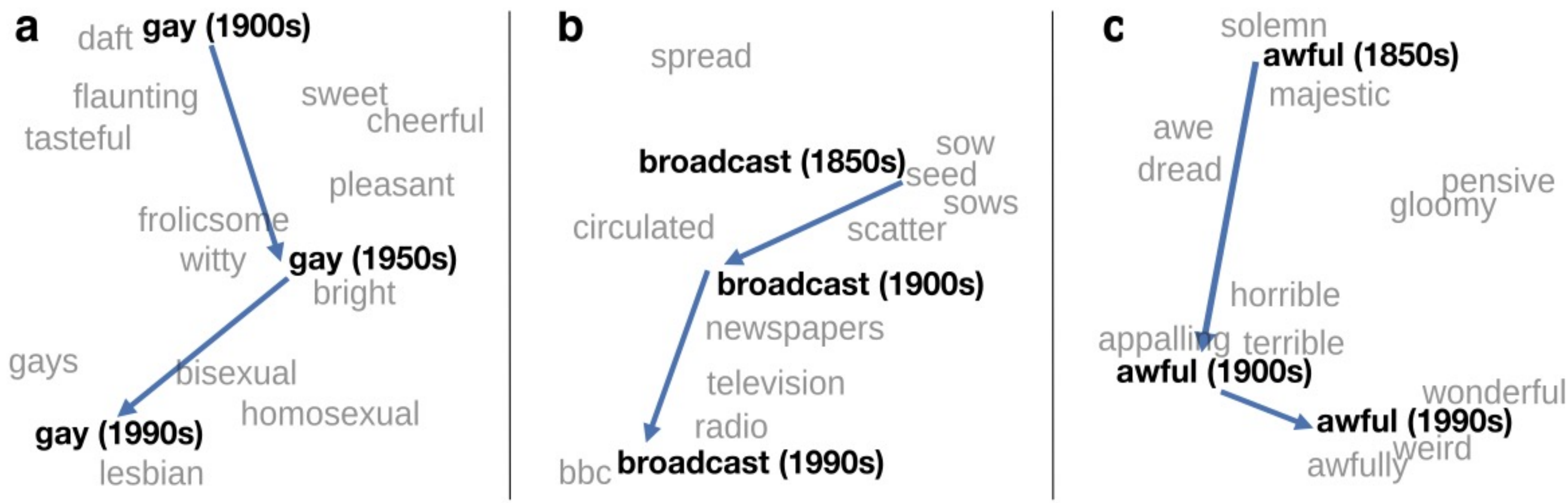


Vector Models of Meaning: Embeddings

Embeddings as a window onto historical semantics

Train embeddings on different decades of historical text to see meanings shift

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



William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. Proceedings of ACL.

Vector Models of Meaning: Embeddings

Word embeddings can be extended to document embeddings:

Generally, a document embedding is simply the sum of its word embeddings.