

CS 1671/2071

Human Language Technologies

Session 5: Bag of words and n-grams

Michael Miller Yoder

January 27, 2025

Course logistics

- First quiz on Canvas due **this Thu Jan 30**
 - I will release it today or tomorrow
 - Looking over the reading is a great way to prepare
- [Project idea submission form](#) is due **this Thu Jan 30**
 - Check out the example projects on the [project website](#)

Review activity

What do you remember about what machine learning is or what you can do with it?

Structure of this course

MODULE 1

Prerequisite skills for NLP

text normalization, linear alg., prob., machine learning

MODULE 2

Approaches

statistical machine learning

How text is represented

n-grams

NLP tasks

language modeling
text classification

MODULE 3

MODULE 4

MODULE 5

NLP applications and ethics

Overview: Bag of words, n-grams

- Term-document and term-term matrices
- Cosine similarity

N-grams

Coding activity

Bag of words document representation

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
...	...

Documents Can Be Represented as Bags of Words

A BAG OF WORDS is a BAG or MULTISSET of the words in a document.

- Like a set, except that identical elements can appear multiple times
- You could also think of it like a `Counter` object in Python

```
bow = {'and': 23502,  
       'or': 12342',  
       'the': 54939508,  
       ...  
       'hippopotamus': 1}
```

- If you take out sequencing information, any document can be viewed as a bag of words.

Bags of Words Can Be Represented as Sparse Vectors

- So far, we've represented bags of words as dense MAPS, but sometimes it is useful to represent them as sparse vectors
- Let V be the set of all words in the document collection D . Then our BoW vectors for documents in D will have $|V|$ dimensions.
- Each dimension corresponds to a word **type** and the value at this dimension corresponds to the number of **TOKENS** of that type in the document that the vector is representing

Term-document and term-term matrices

Term-document matrix

- Each cell is the count of term t in a document d ($tf_{t,d}$).
- Each document is a **count vector** in \mathbb{N}^V , a column below.



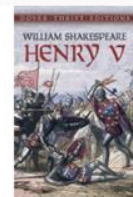
As You Like It



Twelfth Night



Julius Caesar



Henry V

battle
soldier
fool
clown

1
2
37
6

1
2
58
117

8
12
1
0

15
36
5
0

Term-document matrix

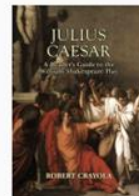
- Two documents are similar if their vectors are similar.



As You Like It



Twelfth Night



Julius Caesar



Henry V

battle
soldier
fool
clown

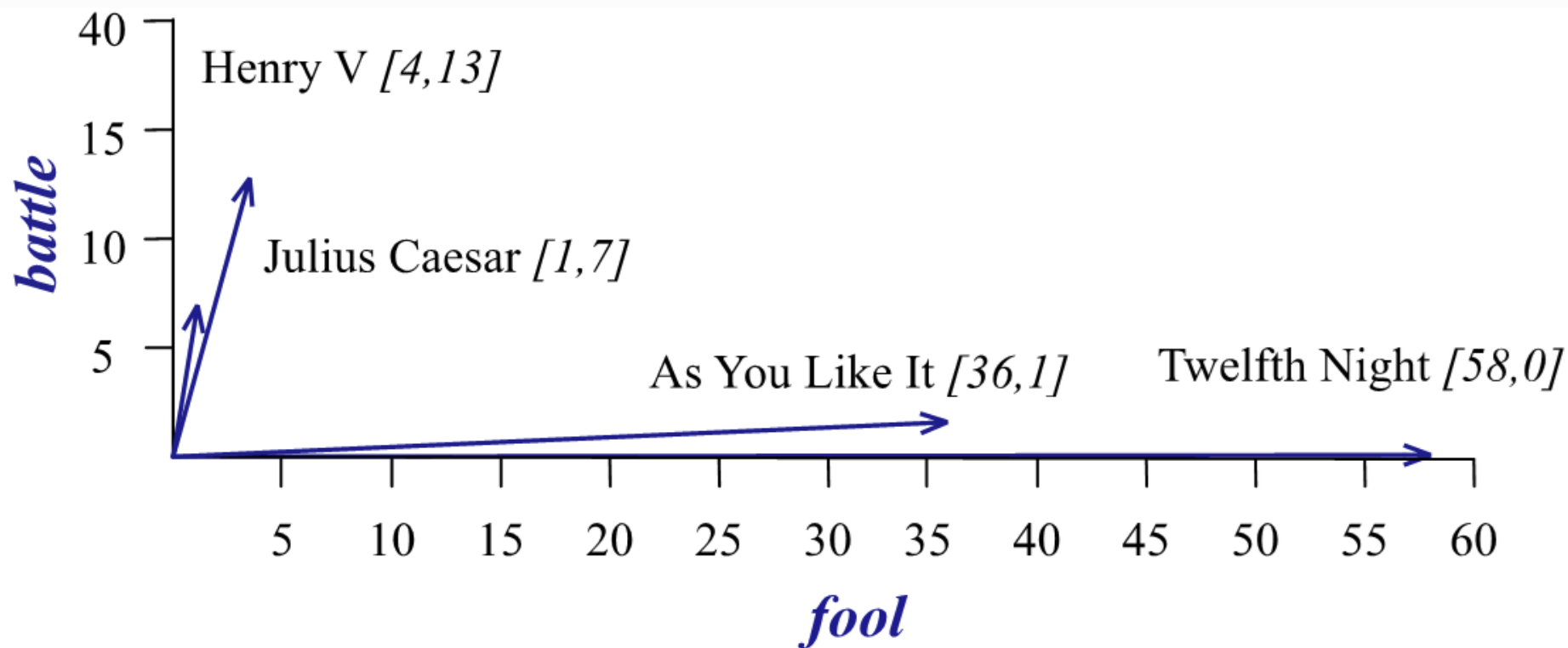
1
2
37
6

1
2
58
117

8
12
1
0

15
36
5
0

Visualizing document vectors



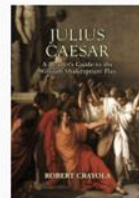
Vectors are the basis of information retrieval

	<i>As You Like It</i>	<i>Twelfth Night</i>	<i>Julius Caesar</i>	<i>Henry V</i>
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- Vectors for comedies are different from tragedies
- Comedies have more *fools* and fewer *battles*

Term-document matrix: word vectors

Two words are similar if their vectors are similar.

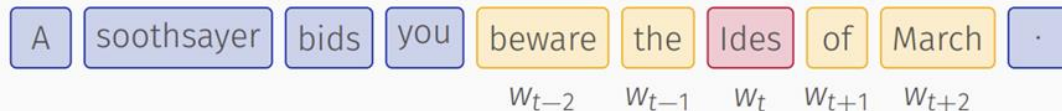


	As You Like It	Twelfth Night	Julius Caesar	Henry V
<i>battle</i>	1	1	8	15
<i>soldier</i>	2	2	12	36
<i>fool</i>	37	58	1	5
<i>clown</i>	6	117	0	0

- *battle* is "the kind of word that occurs in Julius Caesar and Henry V"
- *fool* is "the kind of word that occurs in comedies, especially Twelfth Night"

Term-term matrix (or word-word or word-context matrix)

- Instead of entire documents, use smaller contexts
 - Paragraph
 - Window of a few words (e.g. 3, 5, 7):



- A word is now defined by a vector over counts of words in context.
 - If a word w_j occurs in the context of w_i , increase $count_{ij}$.
- Assuming we have V words,
 - Each vector is now of length V .
 - The word-word matrix is $V \times V$.

Sample Contexts of ± 7 Words

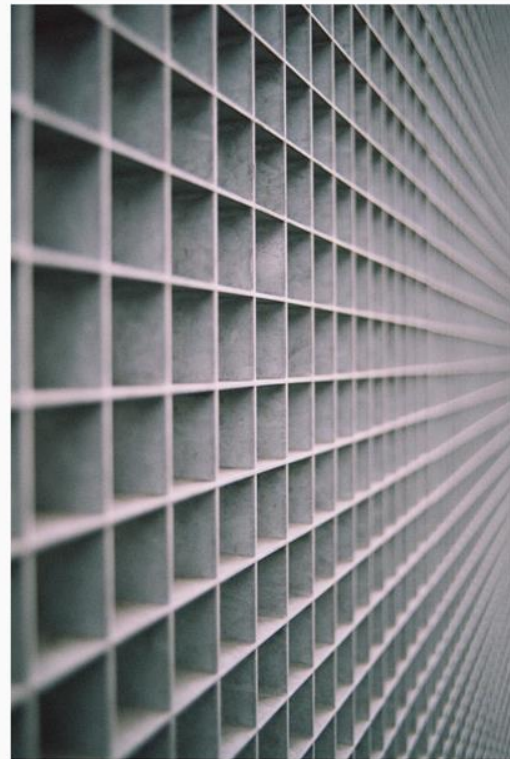
sugar, a sliced lemon, a tablespoonful of their enjoyment. Cautiously she sampled her first well suited to programming on the digital for the purpose of gathering data and **apricot** **pineapple** **computer.** **information** preserve or jam, a pinch each of, and another fruit whose taste she likened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

	aardvark	digital	data	pinch	result	sugar ...
⋮						
<i>apricot</i>	0	0	0	1	0	1
<i>pineapple</i>	0	0	0	1	0	1
<i>computer</i>	0	2	1	0	1	0
<i>information</i>	0	1	6	0	4	0
⋮						

The Word-Word Matrix

We showed only a 4×6 matrix, but the real matrix is $50,000 \times 50,000$.

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



Types of Co-occurrence between Two Words

First-order co-occurrence (syntagmatic association):

- They are typically nearby each other.
- *wrote* is a first-order associate of *book* or *poem*.

Second-order co-occurrence (paradigmatic association):

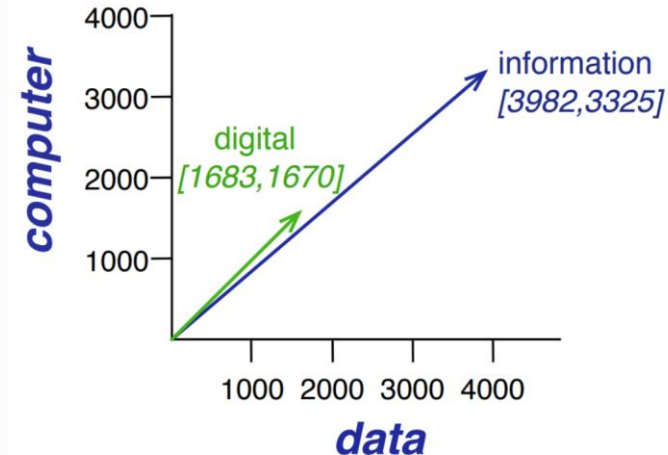
- They have similar neighbors.
- *wrote* is a second-order associate of words like *said* or *remarked*.

Cosine similarity

Measuring similarity between word or document vectors

	aardvark	...	computer	data	result	pie	sugar
cherry	0	...	2	8	9	442	25
strawberry	0	...	0	0	1	60	19
digital	0	...	1670	1683	85	5	4
information	0	...	3325	3982	378	5	13

Do we care about
magnitude/word frequencies?
(No)



Cosine is Used to Measure the Similarity between Word Vectors

- Given two target words represented with vectors \mathbf{v} and \mathbf{w} .
- The **dot product** or **inner product** is usually used as the basis for similarity.

$$\mathbf{v} \cdot \mathbf{w} = \sum_{i=1}^N v_i w_i = v_1 w_1 + v_2 w_2 + \cdots + v_N w_N$$

- $\mathbf{v} \cdot \mathbf{w}$ is high when two vectors have large values in the same dimensions.
- $\mathbf{v} \cdot \mathbf{w}$ is low (in fact 0) with zeros in complementary distribution.
- We also do not want the similarity to be sensitive to word-frequency.
- So normalize by vector length and use the cosine as the similarity

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

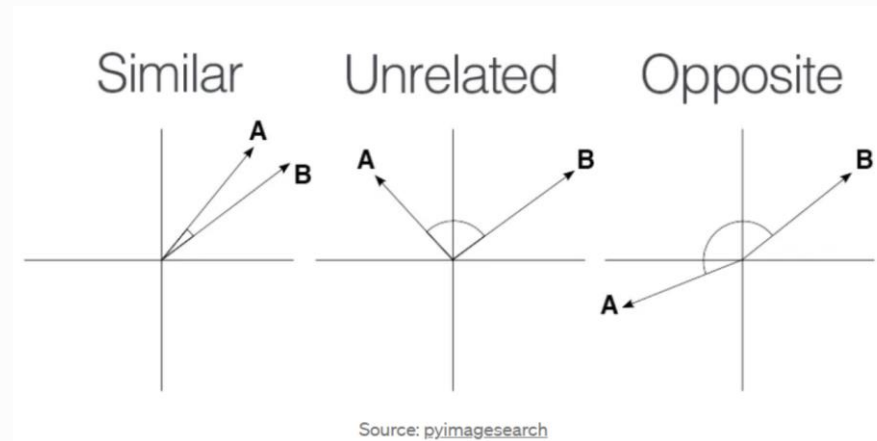
$$\frac{\mathbf{v} \cdot \mathbf{w}}{|\mathbf{v}| |\mathbf{w}|} = \cos(\mathbf{v}, \mathbf{w})$$

Cosine as a similarity metric for vectors

-1: vectors point in opposite directions

+1: vectors point in same directions

0: vectors are orthogonal



But since raw frequency values are non-negative, the cosine for term-term matrix vectors ranges from 0–1

N-grams

N-grams

- A sequence of n words
- “Pittsburgh is cold in the winter.”
- Representations of documents:
 - Unigram: counts of all individual words
 - {Pittsburgh, is, cold, in, the, winter}
 - Bigram: counts of all sequences of 2 words
 - {Pittsburgh is, is cold, cold in, in the, the winter}
 - Trigram: counts of all sequences of 3 words
 - {Pittsburgh is cold, is cold in, cold in the, in the winter}
 - 4gram, etc
- Term-document matrix becomes even sparser!

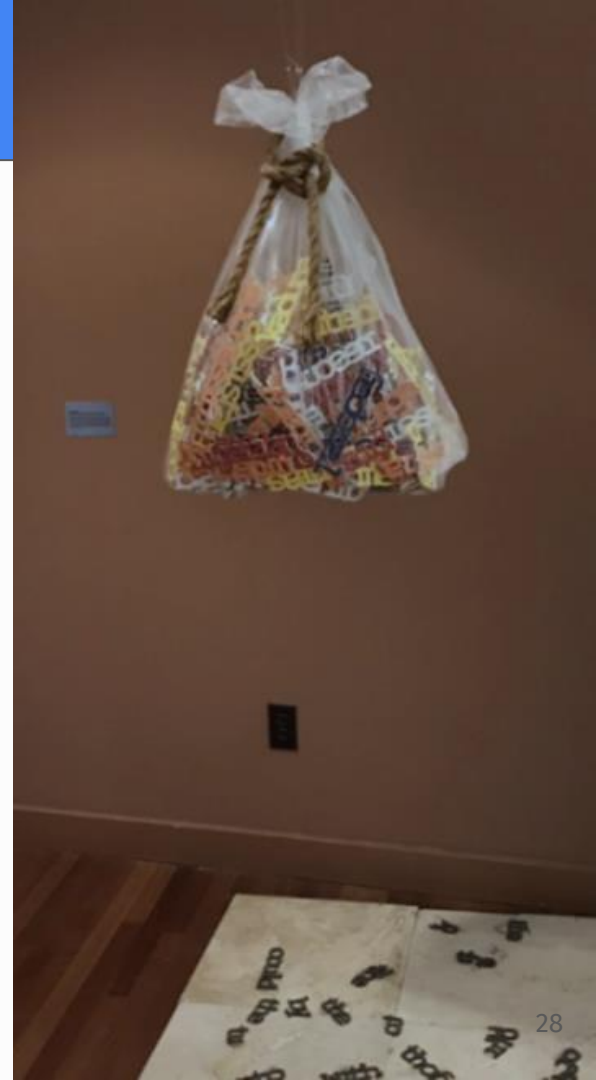
Coding activity:

N-gram document representations on JupyterHub

- [Click on this nbgitpuller link](#)
- Build, examine n-gram document representations for clickbait headlines
- Open `session5_clickbait_ngrams.ipynb`

Conclusion

- **Bag of words** representations of documents
 - Counts of terms in documents (no order info)
 - Can be represented in vector form as...
- **Term-document matrices**
- **Term-term (word-word) matrices**
 - How many times words are used in contexts of other words
- **Cosine** to measure similarity between vectors for documents or words
- **N-grams** are sequences of n words (tokens)



Questions?