# CS 2731
# Introduction to Natural Language Processing

Session 4: Bag of words and n-grams

Michael Miller Yoder

September 8, 2025

University of Pittsburgh | School of Computing and Information

# Course logistics: homework

- Homework 0 grades were released

    - Come see me if you can't get the custom class conda environment loaded

- Homework 1 will be released this Wed Sep 10

# Course logistics: project

- [Project idea form](#) is **due this Thu Sep 11**

    - You will be able to submit any project ideas that you're interested in: from the [example list](#) or any you have on your own

    - It's fine to incorporate your own research, there just needs to be an NLP component

    - You can submit multiple project ideas

- You will later choose from an anonymized list of project ideas on Project Match Day, next Wed Sep 17

# Course logistics: quiz

- First in-class quiz is **this Wed Sep 10**

  - Covers readings from all the sessions up to that point

  - Looking over the reading is a great way to prepare

  - Session 2: J+M 2-2.4, 2.6-2.7, 2.10

  - Session 4: J+M 5.3-5.4

  - Content from Session 1 and Session 3 will not be on quiz

- Conceptual, not programming

- The lowest quiz scores will be dropped

- Only 6% of your course grade total

# Course logistics: quiz

- On Canvas, 12 minutes to complete it (2:30-2:42pm)

- Allowed resources

  - Textbook

  - Your notes (on a computer or physical)

  - Course slides and website

- Resources not allowed

  - Generative AI

  - Internet searches

# Structure of this course

| MODULE 1 | Introduction and text processing | text normalization, machine learning, NLP tasks | |
|---|---|---|---|

| | Approaches | How text is represented | NLP tasks |
|---|---|---|---|
| MODULE 2 | statistical machine learning | n-grams | language modeling<br>text classification |
| MODULE 3 | neural networks | static word vectors | text classification |
| MODULE 4 | transformers and LLMs | contextual word vectors | language modeling<br>text classification |

| MODULE 5 | Sequence labeling and parsing | named entity recognition, dependency parsing |
|---|---|---|
| MODULE 6 | NLP applications and ethics | machine translation, chatbots, search engines, bias |

# 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!

fairy always love to it
it whimsical it
and seen are I
friend anyone
happy dialogue
adventure recommend
who sweet of satirical
it I but to movie it
romantic I
several yet
again it the humor
the seen would
to scenes I the manages
fun the times
I and about and
whenever while
have
conventions
with

| 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 |
| ... | ... |

*Slide from Jurafsky & Martin*

# Documents Can Be Represented as Bags of Words

A BAG OF WORDS is a BAG or MULTISET 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

- Led $V$ be the sent 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 $\texttt{type}$ and the value at this dimensions corresponds to the number of TOKENS of that type in the document that the vector is representing
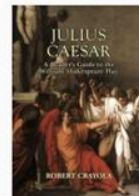
# Term-document and term-term matrices

# 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.



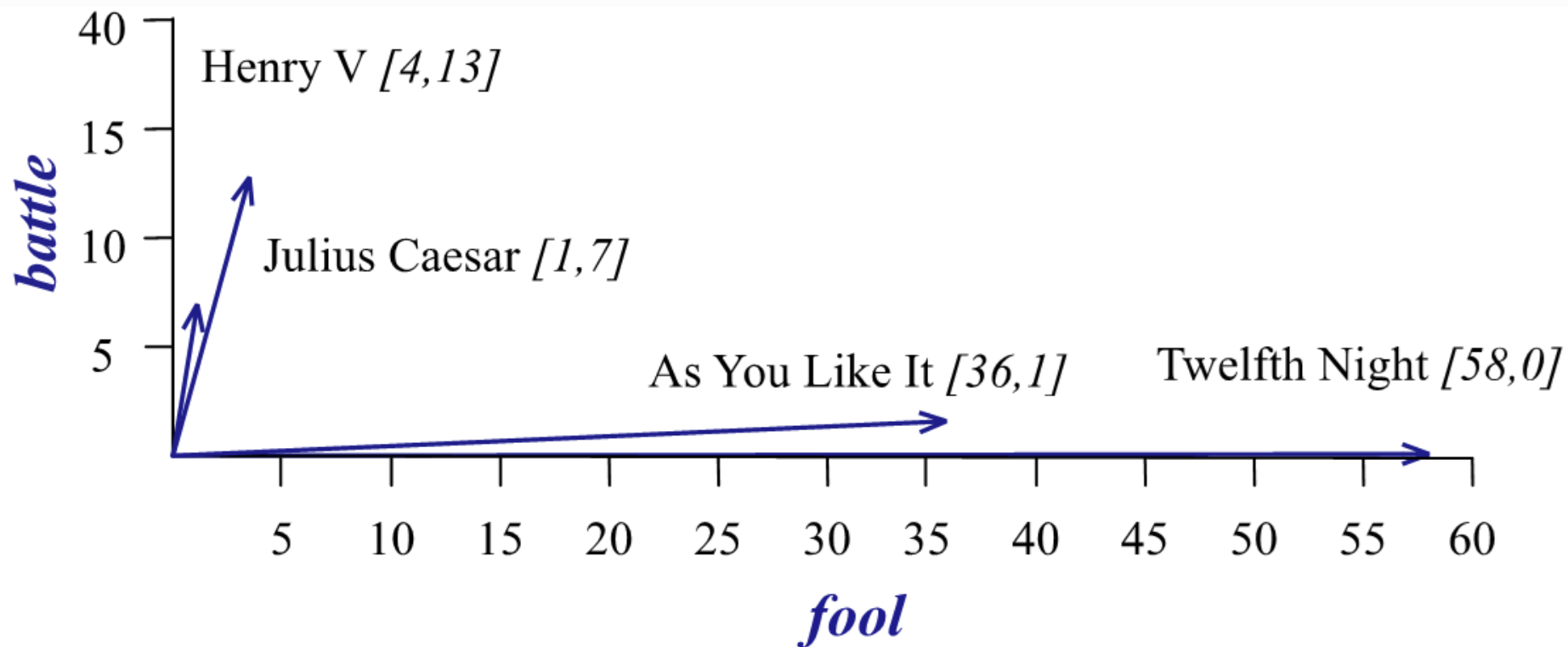|          | As You Like It | Twelfth Night | Julius Caesar | Henry V |
|----------|----------------|---------------|---------------|---------|
| *battle* | 1              | 1             | 8             | 15      |
| *soldier*| 2              | 2             | 12            | 36      |
| *fool*   |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

# Term-document matrix

- Two documents are similar of 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 |

*Slide credit: David Mortensen*

# Visualizing document vectors

*Slide from Jurafsky & Martin*

# Vectors are the basis of information retrieval

|  | *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 |

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

*Slide adapted from Jurafsky & Martin*

# Term-document matrix: word vectors

Two words are similar if their vectors are similar.
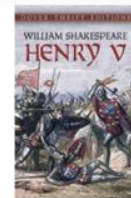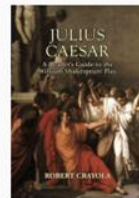
|  | As You Like It | Twelfth Night | Julius Caesar | Henry V |
|---|---|---|---|---|
| *battle* | 1 | 1 | 8 | 15 |
| *soldier* | 2 | 2 | 12 | 36 |
| *fool* | 37 | 58 | 1 | 5 |
| *clown* | 6 | 117 | 0 | 0 |

- *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"

*Slide adapted from David Mortensen, Jurafsky & Martin*

# 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 | soothsayer | bids | you | beware | the | Ides | of | March | · |
|---|---|---|---|---|---|---|---|---|---|
| | | | | $w_{t-2}$ | $w_{t-1}$ | $w_t$ | $w_{t+1}$ | $w_{t+2}$ | |

- 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 $\pm 7$ Words

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 | digital | data | pinch | result | sugar ... |
|---|---|---|---|---|---|---|
| ⋮ | | | | | | |
| *apricot* | 0 | 0 | 0 | 1 | 0 | 1 |
| *pineapple* | 0 | 0 | 0 | 1 | 0 | 1 |
| *computer* | 0 | 2 | 1 | 0 | 1 | 0 |
| *information* | 0 | 1 | 6 | 0 | 4 | 0 |
| ⋮ | | | | | | |

# The Word–Word Matrix

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

- So it is very sparse: Most values are 0.

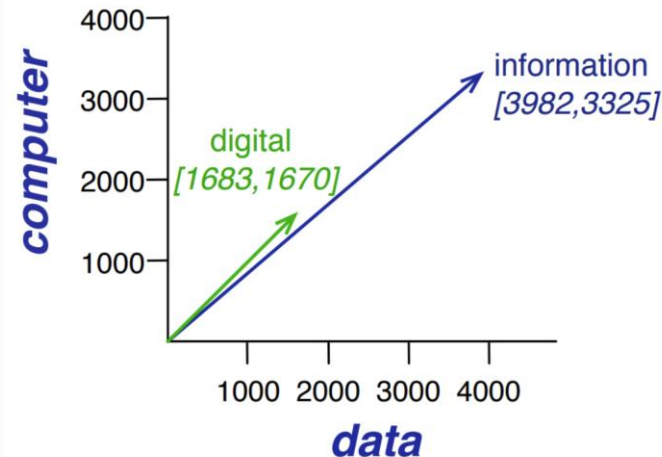- That's OK, since there are lots of efficient algorithms for sparse matrices.
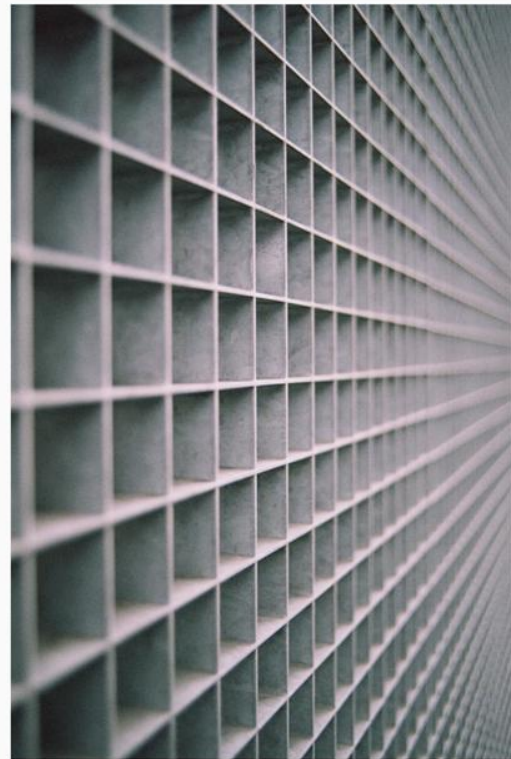
# Cosine similarity

# 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 $v$ and $w$.

- The **dot product** or **inner product** is usually used as the basis for similarity.

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

- $v \cdot w$ is high when two vectors have large values in the same dimensions.

- $v \cdot w$ 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
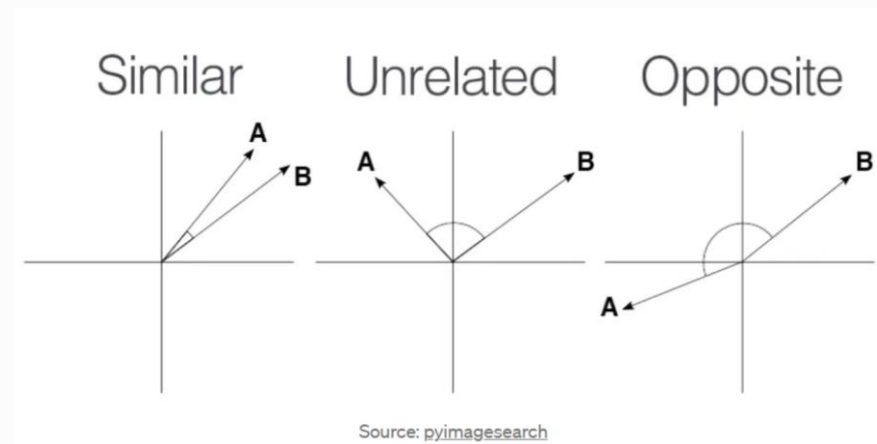
$$|v| = \sqrt{\sum_{(i=1)}^{N} v_i^2}$$

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

# Cosine as a similarity metric for vectors

-1: vectors point in opposite directions

+1:  vectors point in same directions

0: vectors are orthogonal



Source: pyimagesearch

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: clickbait classification
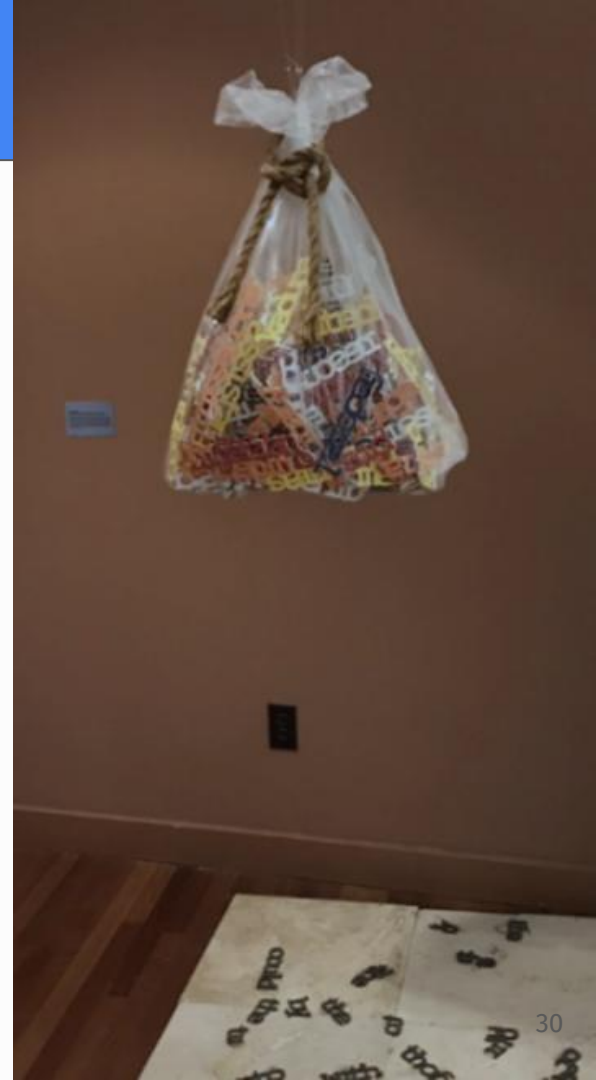
# Clickbait classification on JupyterHub

- [Click on this nbgitpuller link](#)

- Open **session4_clickbait_classification.ipynb**

# 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?*