TF-IDF

Natalie Parde
UIC CS 421

What other ways can we build vector representations for words?

critique

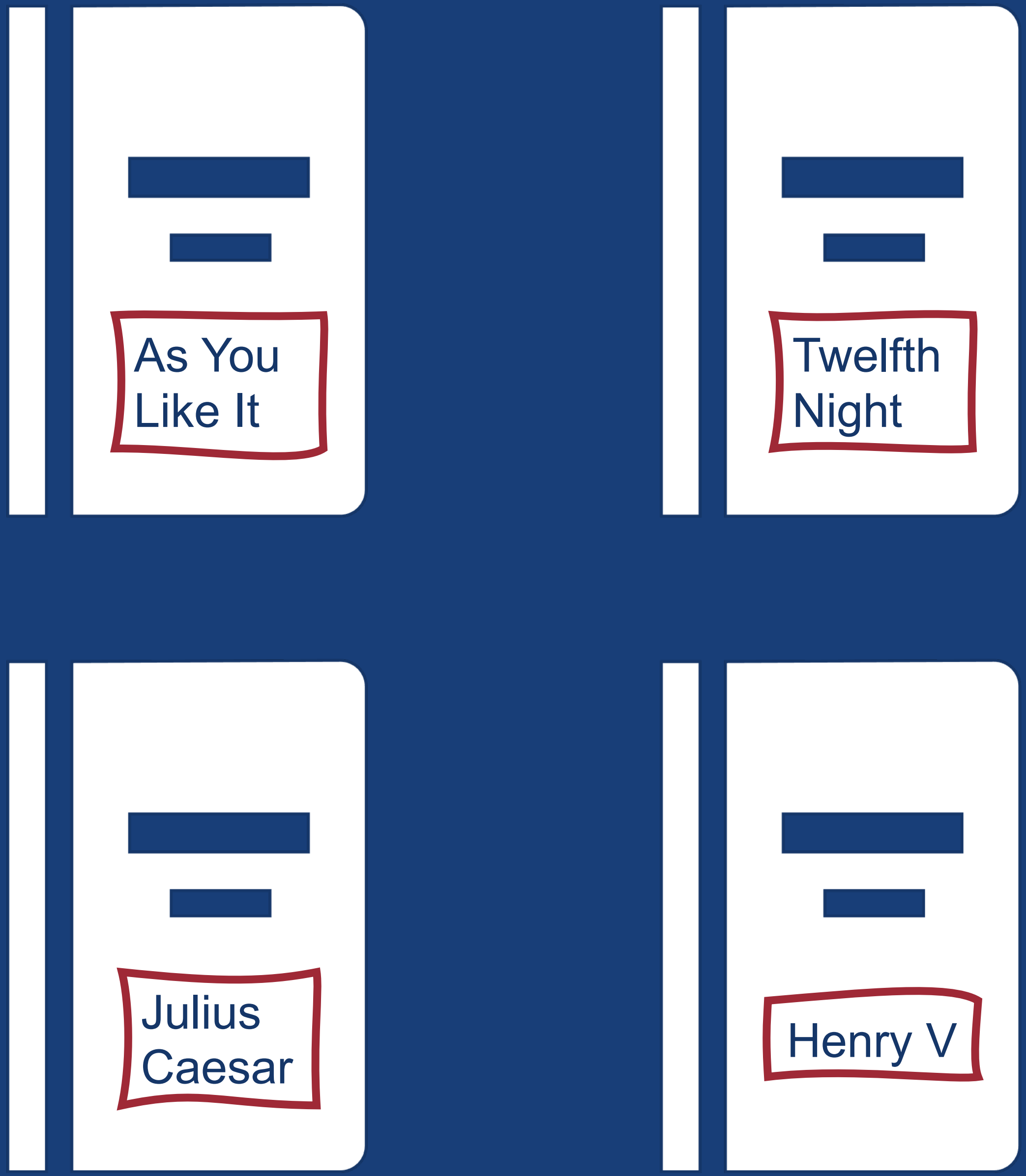
	c_1	…	critique	…	c_n
w_1	…	…	…	…	…
…	…	…	…	…	…
critique	?	?	?	?	?
…	…	…	…	…	…
w_n	…	…	…	…	…

One Approach: TF-IDF

- Term Frequency * Inverse Document Frequency
- Meaning of a word is defined by the counts of words in the same document, as well as overall
- To do this, a **co-occurrence matrix** is needed

TF-IDF originated as a tool for information retrieval.

- Rows: Words in a vocabulary
- Columns: Documents in a selection

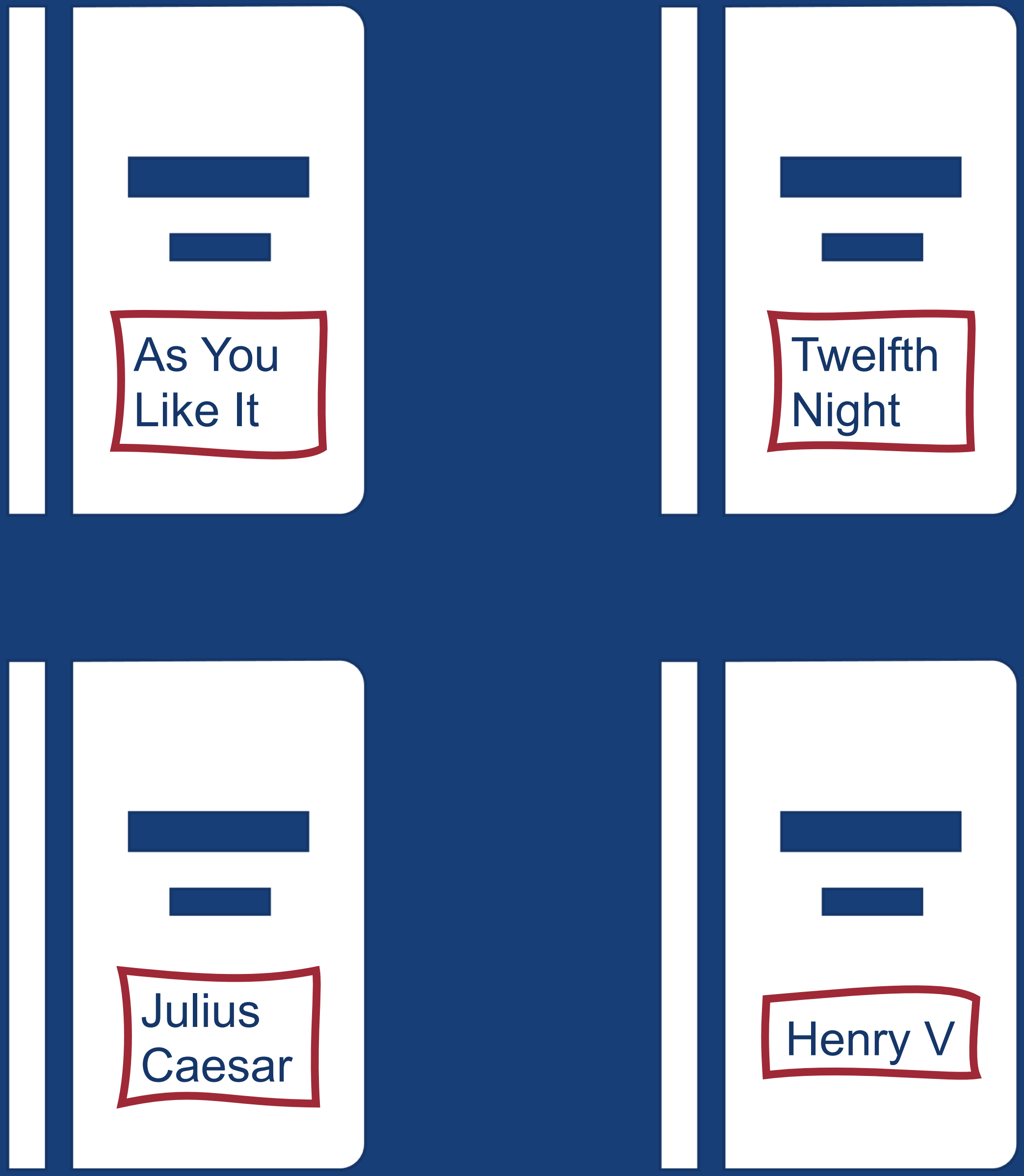


TF-IDF originated as a tool for information retrieval.

- Rows: Words in a vocabulary
- Columns: Documents in a selection

	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

"wit" appears 3 times in Henry V

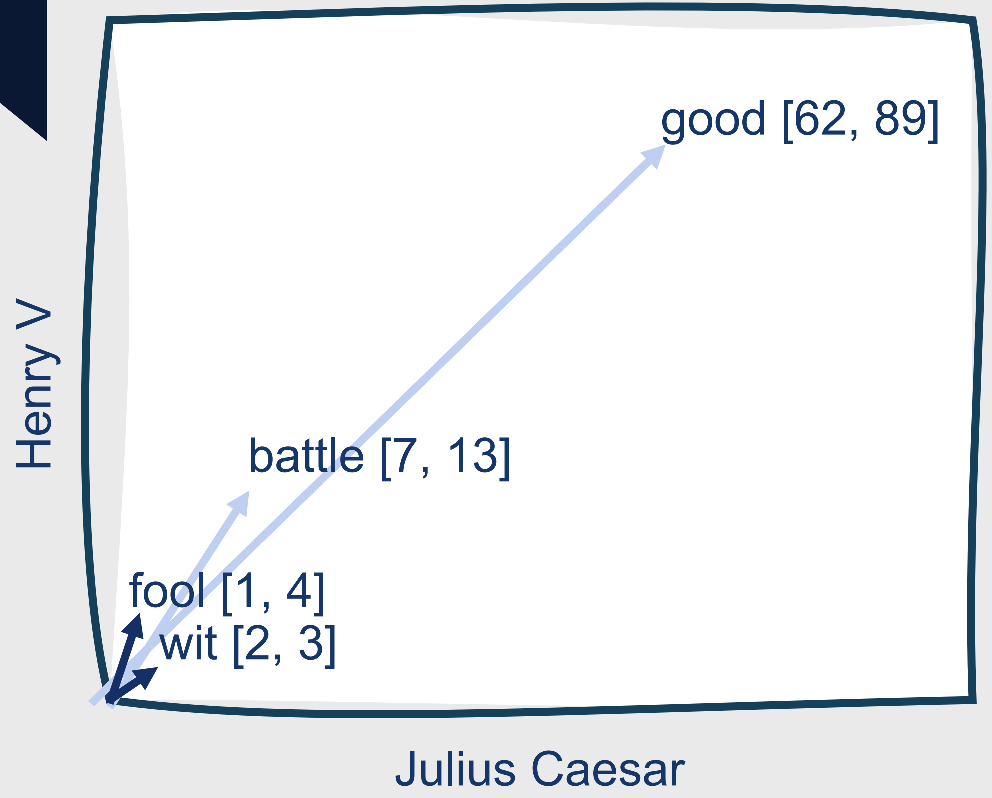


In a term-document matrix, rows can be viewed as word vectors.

- Each dimension corresponds to a document
- Words with **similar vectors** occur in **similar documents**

	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

In a term-document matrix, rows can be viewed as word vectors.



	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

Different Types of Context

- Documents aren't the most common type of context used to represent meaning in word vectors
- More common: **word context**
  - Referred to as a term-term matrix, word-word matrix, or term-context matrix
- In a **word-word matrix**, the columns are also labeled by words
  - Thus, dimensionality is |V| x |V|
  - Each cell records the number of times the row (target) word and the column (context) word co-occur in some context in a training corpus

How can you decide if two words occur in the same context?

- Common **context windows**:
  - Entire document
    - Cell value = # times the words co-occur in the same document
  - Predetermined span surrounding the target
    - Cell value = # times the words co-occur in this span of words

Example Context Window (Size = 4)

- Take each occurrence of a word (e.g., strawberry)
- Count the context words in the four-word spans before and after it to get a word-word co-occurrence matrix

is	traditionally	followed	by	cherry	pie,	a	traditional	dessert
often	mixed,	such	as	strawberry	rhubarb	pie.	Apple	pie
computer	peripherals	and	personal	digital	assistants.	These	devices	usually
a	computer.	This	includes	information	available	on	the	internet

Example Context Window (Size = 4)

- A simplified subset of a word-word co-occurrence matrix could appear as follows, given a sufficient corpus

is	traditionally	followed	by	cherry	pie,	a	traditional	dessert
often	mixed,	such	as	strawberry	rhubarb	pie.	Apple	pie
computer	peripherals	and	personal	digital	assistants.	These	devices	usually
a	computer.	This	includes	information	available	on	the	internet

	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	…

Vector for "strawberry"

So far, our co-occurrence matrices have contained raw frequency counts of word co-occurrences.

- However, this isn't the best measure of association between words
  - Some words co-occur frequently with many words, so won't be very informative
    - *the*, *it*, *they*
- We want to know about **words that co-occur frequently with one another, but less frequently across all texts**

This is where TF-IDF comes in handy!

- **Term Frequency:** The frequency of the word *t* in the document *d*
  - $tf_{t,d} = \text{count}(t, d)$
- **Document Frequency:** The number of documents in which the word *t* occurs
  - Different from collection frequency (the number of times the word occurs in the entire collection of documents)

Computing TF-IDF

- **Inverse Document Frequency:** The inverse of document frequency, where *N* is the total number of documents in the collection
  - $idf_t = \frac{N}{df_t}$
- IDF is higher when the term occurs in fewer documents
- What is a document?
  - Individual instance in your corpus (e.g., book, play, sentence, etc.)
- It is often useful to perform these computations in log space
  - TF: $\log_{10}(tf_{t,d}+1)$
  - IDF: $\log_{10} idf_t$

Computing TF*IDF

- TF-IDF is then simply the combination of TF and IDF
  - $tfidf_{t,d} = tf_{t,d} \times idf_t$

Example: Computing TF-IDF

- TF-IDF(battle, d_1) = ?

	d_1	d_2	d_3	d_4
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Example: Computing TF-IDF

- TF-IDF(battle, d_1) = ?
- **TF(battle, d_1) = 1**

	d_1	d_2	d_3	d_4
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Example: Computing TF-IDF

- TF-IDF(battle, d_1) = ?
- TF(battle, d_1) = 1
- **IDF(battle) = N/DF(battle) = 37/21 = 1.76**

	d_1	d_2	d_3	d_4
battle	1	0	7	13
good	114	80	62	89
fool	36	58		
wit	20	15		

Document frequencies from 37-document corpus

word	df
battle	21
good	37
fool	3[illegible]
wit	34

Example: Computing TF-IDF

- TF-IDF(battle, d_1) = ?
- TF(battle, d_1) = 1
- IDF(battle) = N/DF(battle) = 37/21 = 1.76
- **TF-IDF(battle, d_1) = 1 * 1.76 = 1.76**

	d_1	d_2	d_3	d_4
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Example: Computing TF-IDF

- TF-IDF(battle, d_1) = ?
- TF(battle, d_1) = 1
- IDF(battle) = N/DF(battle) = 37/21 = 1.76
- TF-IDF(battle, d_1) = 1 * 1.76 = 1.76
- **Alternately, TF-IDF(battle, d_1) =** $log_{10}(1+1) * \ log_{10} 1.76 =$ **0.074**

	d_1	d_2	d_3	d_4
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Example: Computing TF-IDF

- TF-IDF(battle, d_1) = ?
- TF(battle, d_1) = 1
- IDF(battle) = N/DF(battle) = 37/21 = 1.76
- TF-IDF(battle, d_1) = 1 * 1.76 = 1.76
- Alternately, TF-IDF(battle, d_1) = $log_{10}(1+1) * \ log_{10} 1.76 = 0.074$

	d_1	d_2	d_3	d_4
battle	0.074	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

To convert our entire word co-occurrence matrix to a TF-IDF matrix, we need to repeat this calculation for each element.

	d_1	d_2	d_3	d_4
battle	0.074	0.000	0.220	0.280
good	0.000	0.000	0.000	0.000
fool	0.019	0.021	0.004	0.008
wit	0.049	0.044	0.018	0.022

How does the TF-IDF matrix compare to the original term frequency matrix?

	d_1	d_2	d_3	d_4
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

	d_1	d_2	d_3	d_4
battle	0.074	0.000	0.220	0.280
good	0.000	0.000	0.000	0.000
fool	0.019	0.021	0.004	0.008
wit	0.049	0.044	0.018	0.022

How does the TF-IDF matrix compare to the original term frequency matrix?

	d_1	d_2	d_3	d_4
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

	d_1	d_2	d_3	d_4
battle	0.074	0.000	0.220	0.280
good	0.000	0.000	0.000	0.000
fool	0.019	0.021	0.004	0.008
wit	0.049	0.044	0.018	0.022

Occurs in every document …not important in the overall scheme of things!

How does the TF-IDF matrix compare to the original term frequency matrix?

	d_1	d_2	d_3	d_4
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

	d_1	d_2	d_3	d_4
battle	0.074	0.000	0.220	0.280
good	0.000	0.000	0.000	0.000
fool	0.019	0.021	0.004	0.008
wit	0.049	0.044	0.018	0.022

Increases the importance of rarer words like “battle”

Note that the TF-IDF model produces a sparse vector.

- **Sparse:** Many (usually most) cells have values of 0

	d_1	d_2	d_3	d_4
battle	0.074	0.000	0.220	0.280
good	0.000	0.000	0.000	0.000
fool	0.019	0.021	0.004	0.008
wit	0.049	0.044	0.018	0.022

Note that the TF-IDF model produces a sparse vector.

- **Sparse:** Many (usually most) cells have values of 0

	d_1	d_2	d_3	d_4	d_5	d_6	d_7
battle	0.1	0.0	0.0	0.0	0.2	0.0	0.3
good	0.0	0.0	0.0	0.0	0.0	0.0	0.0
fool	0.0	0.0	0.0	0.0	0.0	0.0	0.0
wit	0.0	0.0	0.0	0.0	0.0	0.0	0.0

This can be problematic!

- However, TF-IDF remains a useful starting point for vector space models
- Generally combined with standard machine learning algorithms
  - Logistic Regression
  - Naïve Bayes