

COMP 472: Artificial Intelligence Natural Language Processing *part #5* Bag of Word Model *Video #2*

- Russell & Norvig: Section 23.1.1

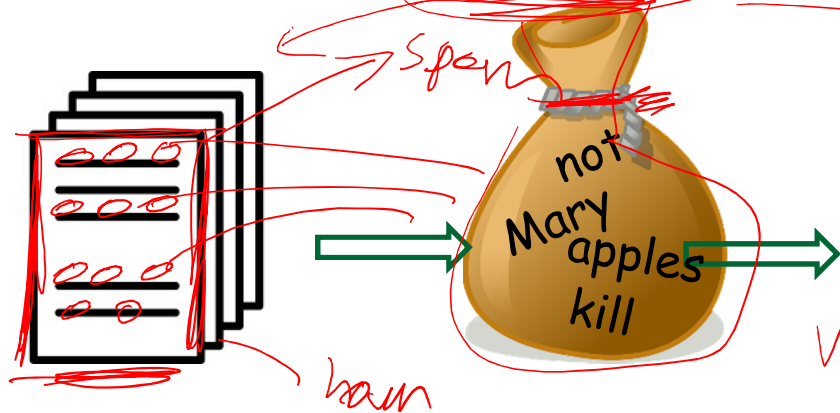
Today

1. Introduction
2. Bag of word model
3. n-gram models
4. Deep Learning for NLP
 1. Word Embeddings
 2. Recurrent Neural Networks



Bag-of-words Model (BOW)

- A simple model where word order is ignored



Word	Value
Mary	1
apples	1
did	1
eat	
John	
kill	
like	
not	
to	

- used in many applications:
 - NB spam filter seen in class a few weeks ago
 - Information Retrieval (eg. google search)
 - ...

- But has severe limits to understand meaning of text...
- Maybe we should take word order into account...

d1 Mary Killed John
d2 John Killed Mary

BOW - Document Representation

■ Representation of a documents = vectors of pairs <word, value>

- word: all word in the vocabulary (aka as a term)
- value: a number associated with the word in the document
 - different possible schemes:

1. binary (0, if term is absent ; 1, if term is present)
2. term frequency
3. some other weighting scheme (e.g. tf.idf)

term frequency
× *inverse document frequency*

doc

Mary did kill John.
Mary did not like to eat apples.



Binary

Word	Value
airplane	0
<u>apples</u>	<u>1</u>
<u>apple</u>	<u>0</u>
banana	0
<u>did</u>	<u>1</u>
dinner	0
element	0
Mary	1
...	...
zoo	0

Term freq

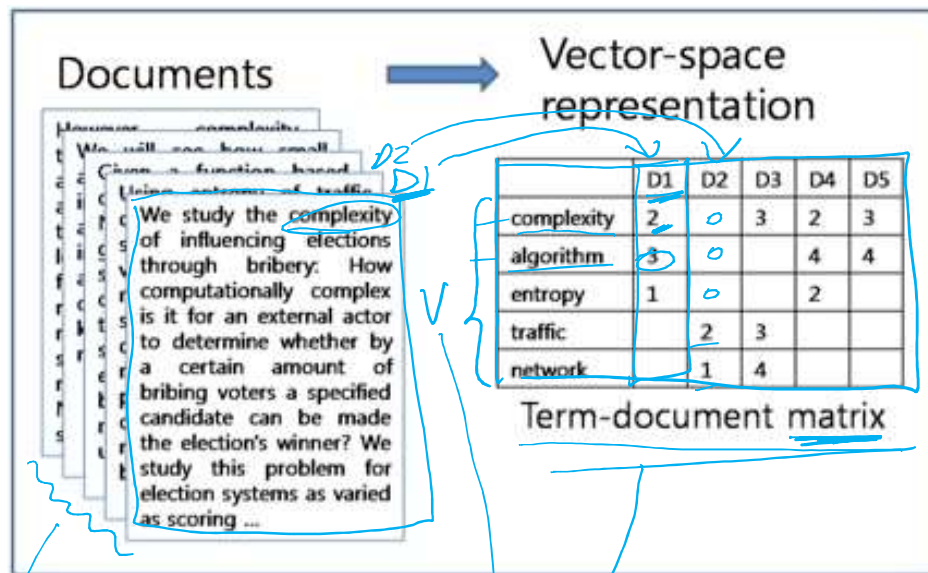
Word	freq
airplane	0
apples	1
banana	0
did	<u>2</u>
dinner	0
element	0
Mary	2
...	...
zoo	0

Word	other
airplane	?
apples	?
banana	?
did	?
dinner	?
element	?
Mary	?
...	...
zoo	?

V

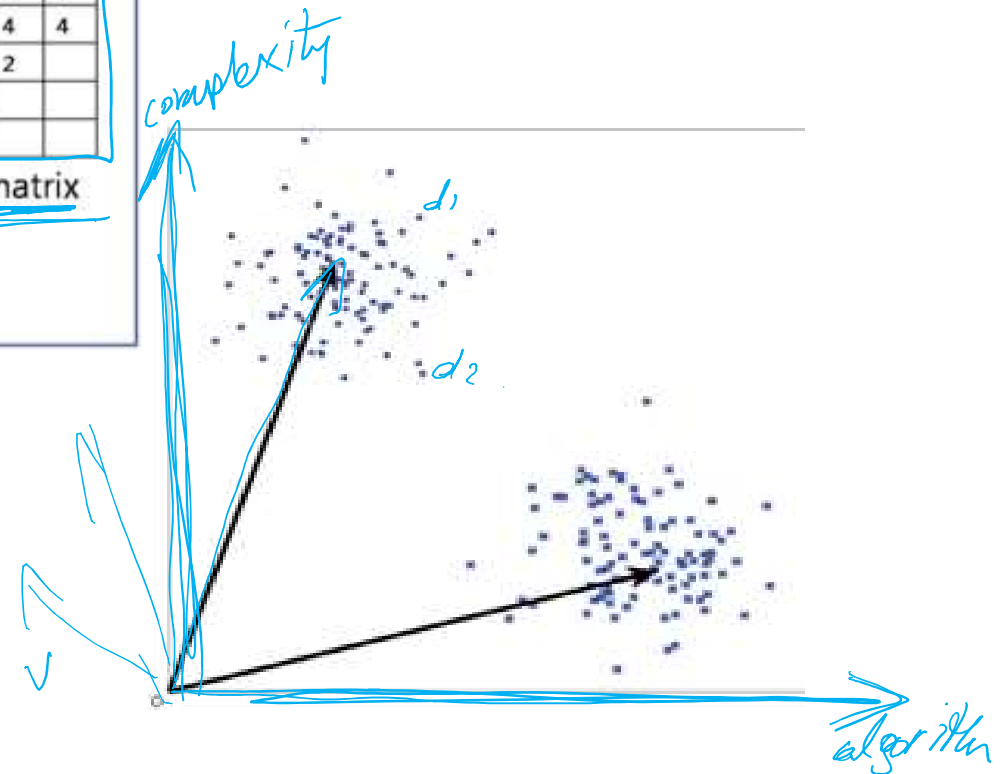
~ 10,000
~ 50,000

BOW - Document Representation



collection of documents

$V = 10,000$
 $50,000$



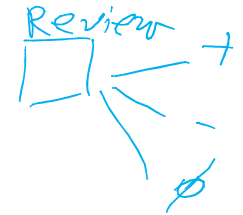
So what?

- once a document is represented as a long vector of numbers, we can:

- Do text categorization/classification

- i.e. Use your favorite supervised machine learning model to classify a document into pre-defined classes

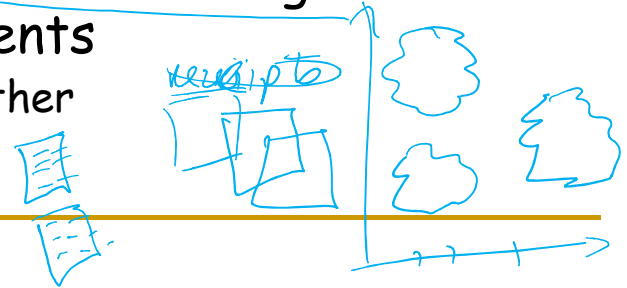
- eg. Spam filtering, News routing, Sentiment Analysis
- using: NB classifier, decision tree, neural networks...



- Do text clustering

- i.e. use your favorite unsupervised machine learning model to compute the similarity between documents

- eg. k-means to group similar documents together



Text Categorization

- Remember this slide?

Example

- Dataset
 - c1: SPAM
 - doc1: "cheap meds for sale"
 - doc2: "click here for the best meds"
 - doc3: "book your trip"
 - c2: HAM
 - doc4: "cheap book sale, not meds"
 - doc5: "here is the book for you"
- Question:
 - doc6: "the cheap book"
 - should it be classified as HAM or SPAM?



6

- *eg.* Multinomial Naive Bayes Classification is a standard application

Text Clustering

- similar technique as used in information retrieval
- Assume we have 3 documents (Web pages)

$d_1 =$

introduction knowledge in speech and language processing ambiguity models and algorithms language thought and understanding the state of the art and the near-term future some brief history summary

$d_2 =$

hmms and speech recognition speech recognition architecture introduction knowledge in speech and language processing ambiguity models and algorithms language thought and understanding the state of the art and the near-term future some brief history summary the hidden markov models the viterbi algorithm revisited advanced methods in decoding acoustic processing of speech computing acoustic probabilities training a speech recognizer waveform generation for speech synthesis human speech recognition summary}

$d_3 =$

language and complexity the chomsky hierarchy how to tell if a language isn't regular the pumping lemma are English and other language regular language ? is natural language context-free complexity and human processing summary

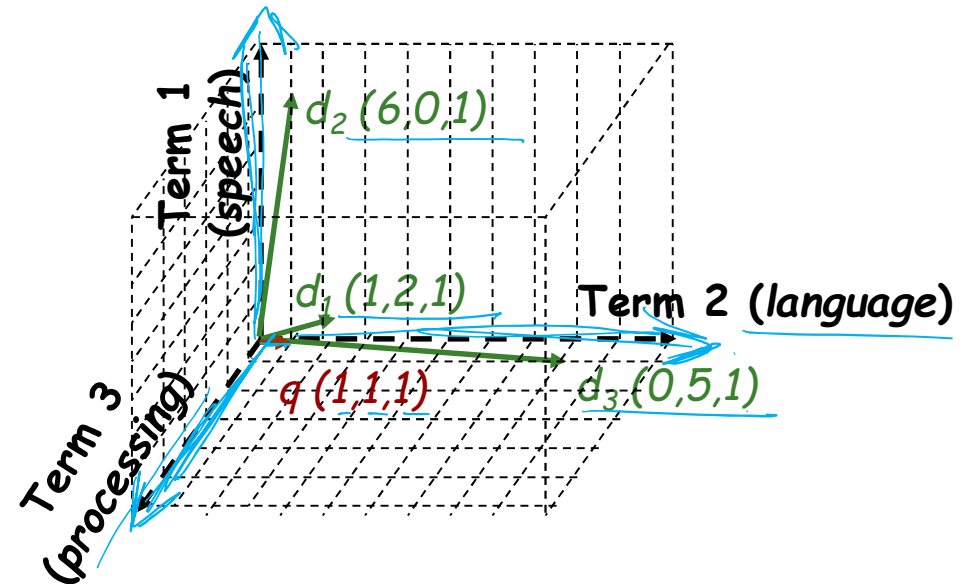
■ a query $Q =$

speech language processing

Example

- using term frequencies

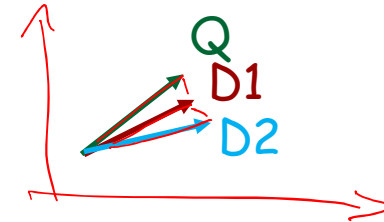
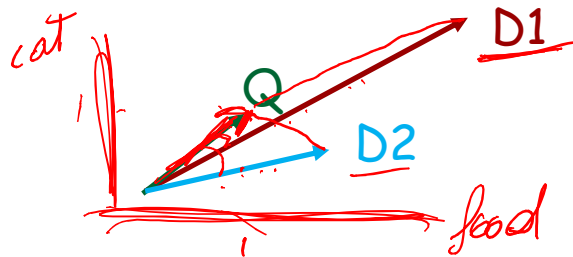
	d_1	d_2	d_3	Q
introduction	...	0	...	0
knowledge	...	0	...	0
...	...	0	...	0
speech	1	6	0	1
language	2	0	5	1
processing	1	1	1	1
...	...	0	...	0



- the documents and the query can be seen as vectors in a multi-dimensional space
- In the case of IR
 - we compare all other documents to a single document (the query Q)
 - so only the terms of the query are relevant
 - so dimensions represent only the terms of the query

Distance Measure

- similarity between two documents (or doc & query) can be measured
 - using the Euclidian distance - as in k-means
 - but longer documents will have larger values and longer lengths
 - what we really care about is the relative distribution of the word values, not the exact values



- so other measures that normalize the length of the vectors are preferable
- simplest/most popular measure is the cosine measure

The Cosine Measure

- The cosine of 2 vectors (in N dimensions)

each dimension

inner product

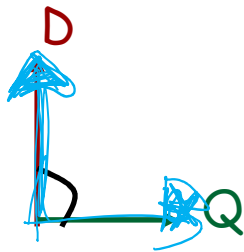
$$\cos(\vec{D}, \vec{Q}) = \frac{\vec{D} \cdot \vec{Q}}{|\vec{D}| |\vec{Q}|} = \frac{\sum_{i=1}^N d_i q_i}{\sqrt{\sum_{i=1}^N d_i^2} \sqrt{\sum_{i=1}^N q_i^2}}$$

normalized by the lengths of the vectors

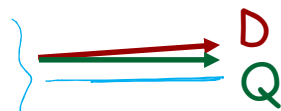
- as if all vectors had a length of 1

The Cosine Measure

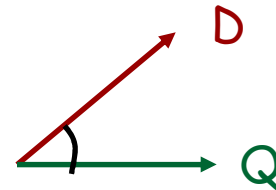
- cosine of the angle between the 2 vectors
 - if 2 document-vectors are identical
 - → they will have a cosine of 1
 - if 2 document-vectors are orthogonal (i.e. share no common term)
 - → they will have a cosine of 0



$$\cos(D, Q) = 0$$



$$\cos(D, Q) \approx 1$$



$$\cos(D, Q) \approx 0.7$$

The example again

	d_1	d_2	d_3	Q
introduction	1	0	0	0
knowledge	1	0	0	0
...				
speech	1	6	0	1
language	2	0	5	1
processing	1	1	1	1
...				

$$\text{sim}(\vec{D}, \vec{Q}) = \cos(\vec{D}, \vec{Q}) = \frac{\vec{D} \cdot \vec{Q}}{|\vec{D}| |\vec{Q}|} = \frac{\sum_{i=1}^N d_i q_i}{\sqrt{\sum_{i=1}^N d_i^2} \sqrt{\sum_{i=1}^N q_i^2}}$$

if we only consider the words of the query {speech, language, processing}, then

query = (1,1,1)

$d_1 = (1,2,1)$ $d_2 = (6,0,1)$ $d_3 = (0,5,1)$

$$\text{sim}(d_1, Q) = \frac{(1 \times 1) + (2 \times 1) + (1 \times 1)}{\sqrt{(1^2 + 2^2 + 1^2)} \times \sqrt{(1^2 + 1^2 + 1^2)}} = \frac{1 + 2 + 1}{\sqrt{6} \times \sqrt{3}} = 0.943$$

$$\text{sim}(d_2, Q) = \frac{(6 \times 1) + (0 \times 1) + (1 \times 1)}{\sqrt{(6^2 + 0^2 + 1^2)} \times \sqrt{(1^2 + 1^2 + 1^2)}} = \frac{6 + 0 + 1}{\sqrt{37} \times \sqrt{3}} = 0.664$$

$$\text{sim}(d_3, Q) = \frac{(0 \times 1) + (5 \times 1) + (1 \times 1)}{\sqrt{(0^2 + 5^2 + 1^2)} \times \sqrt{(1^2 + 1^2 + 1^2)}} = \frac{0 + 5 + 1}{\sqrt{26} \times \sqrt{3}} = 0.680$$

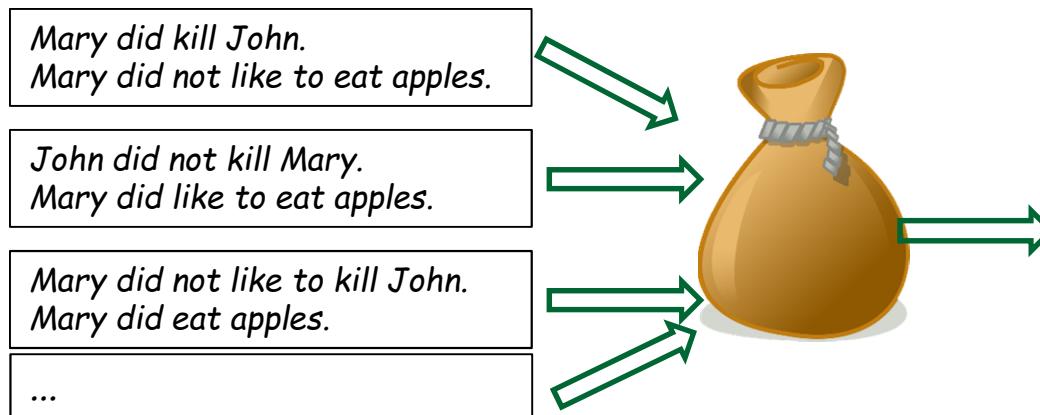
Pro/Cons of BOW Model

■ pros:

- simple model
- efficient for large collections of documents
- basis of many IR, and text categorization systems

■ cons:

- word order is ignored ==> meaning of text is lost.





Word	Freq.
Mary	2
apples	1
did	2
eat	1
John	1
kill	1
like	1
not	1
to	1

■ Solution:

- n-grams take [a bit of] word order into account

Today

1. Introduction 
2. Bag of word model 
3. n-gram models
4. Deep Learning for NLP
 1. Word Embeddings
 2. Recurrent Neural Networks

Up Next

1. Introduction
2. Bag of word model
3. **n-gram models** ✓
4. Deep Learning for NLP
 1. Word Embeddings
 2. Recurrent Neural Networks