Information Retrieval

Natural Language Processing: Jordan Boyd-Graber
University of Maryland
VECTOR SPACE MODELS

Slides adapted from Jimmy Lin

Ranked Retrieval

- Order documents by how likely they are to be relevant to the information need
  - Estimate relevance(q, d_i)
  - Sort documents by relevance
  - Display sorted results
- User model
  - Present hits one screen at a time, best results first
  - At any point, users can decide to stop looking
- How do we estimate relevance?
  - Assume document is relevant if it has a lot of query terms
  - Replace relevance with sim(q, d_i)
  - Compute similarity of vector representations

Ranked Retrieval

- Order documents by how likely they are to be relevant to the information need
  - Estimate relevance(q, d_i)
  - Sort documents by relevance
  - Display sorted results
- User model
  - Present hits one screen at a time, best results first
  - At any point, users can decide to stop looking
- How do we estimate relevance?
  - Assume document is relevant if it has a lot of query terms
  - Replace relevance with sim(q, d_i)
  - Compute similarity of vector representations

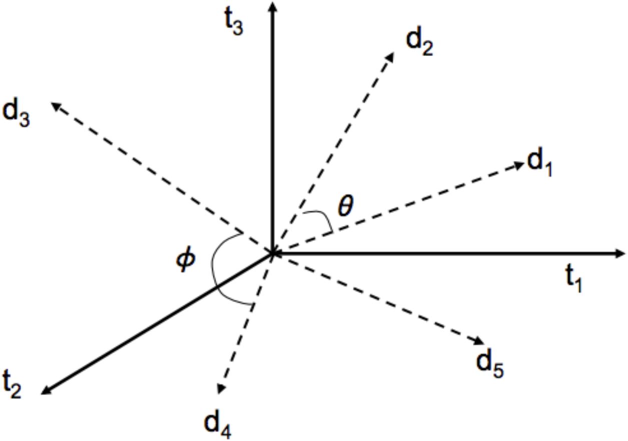
Aside: The Importance of Representation

- Central problem in NLP is how to store / represent / query text
- This is almost certainly wrong model . . . hopefully useful
- Modern NLP typically uses vector representations

Aside: The Importance of Representation

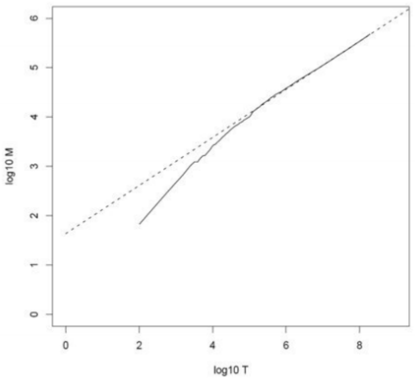
- Central problem in NLP is how to store / represent / query text
- This is almost certainly wrong model . . . hopefully useful
- Modern NLP typically uses vector representations
  - What's the theory
  - How do we build them
  - How do they connect to other tasks?

Representing documents



Each document is vector $d_i = \left\langle w_{i,1}, \ldots w_{i,V} \right\rangle$ (each word is dimension)

High-Dimensional Space (Heaps' Law)

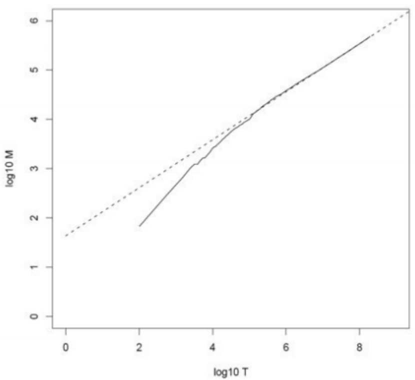


k = 44
b = 0.49

First 1,000,020 terms:
Predicted = 38,323
Actual = 38,365

$$V = kD^b \qquad (1)$$

High-Dimensional Space (Heaps' Law)



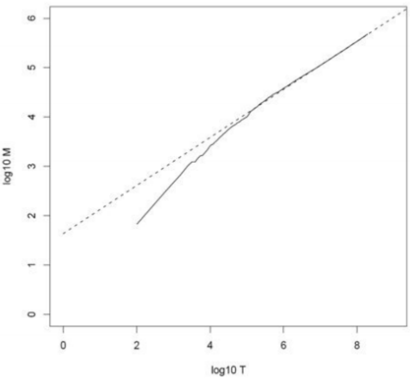
k = 44
b = 0.49

First 1,000,020 terms:
Predicted = 38,323
Actual = 38,365

$$V = kD^b \qquad (1)$$

Vocabulary size

High-Dimensional Space (Heaps' Law)



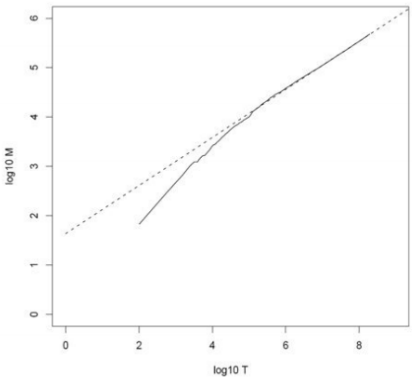
k = 44
b = 0.49

First 1,000,020 terms:
Predicted = 38,323
Actual = 38,365

$$V = k\textcolor{red}{D}^b \qquad (1)$$

Number of documents

High-Dimensional Space (Heaps' Law)



k = 44
b = 0.49

First 1,000,020 terms:
Predicted = 38,323
Actual = 38,365

$$V = kD^b \qquad (1)$$

Constants (per-language, type of document)

Intuitions

- Term weights consist of two components
  - Local: how important is the term in this document?
  - Global: how important is the term in the collection?
- Here's the intuition:
  - Terms that appear often in a document should get high weights
  - Terms that appear in many documents should get low weights
- How do we capture this mathematically?
  - Term frequency (local)
  - Inverse document frequency (global)

tf-idf Term Weighting

$$w_{i,j} = f_{i,j} \log \left(\frac{D}{d_i} \right) \tag{2}$$

- Word i's weight in document j
- Frequency of word i in document j
- Total number of documents
- Number of documents i appears in

tf-idf Term Weighting

$$w_{i,j} = f_{i,j} \log\left(\frac{D}{d_i}\right) \tag{2}$$

- Word *i*'s weight in document *j*
- Frequency of word *i* in document *j*
- Total number of documents
- Number of documents *i* appears in

tf-idf Term Weighting

$$w_{i,j} = f_{i,j} \log\left(\frac{D}{d_i}\right) \tag{2}$$

- Word i's weight in document j
- Frequency of word i in document j
- Total number of documents
- Number of documents i appears in

tf-idf Term Weighting

$$w_{i,j} = f_{i,j} \log\left(\frac{D}{d_i}\right) \quad (2)$$

- Word i's weight in document j
- Frequency of word i in document j
- Total number of documents
- Number of documents i appears in

tf-idf Term Weighting

$$w_{i,j} = f_{i,j} \log\left(\frac{D}{d_i}\right) \quad (2)$$

- Word i's weight in document j
- Frequency of word i in document j
- Total number of documents
- Number of documents i appears in

Frequency of Terms (Zipf's Law)

The most frequent words ("the") are everywhere but useless for queries.
The most useful words are relatively rare ... but there are lots of them.

$$f_i = \frac{c}{R_i} \tag{3}$$

- The frequency of a word *i* is inversly proportional to
- The rank (in frequency) of word
- Scaled by a constant

Can't just throw out useless words

Frequency of Terms (Zipf's Law)

The most frequent words ("the") are everywhere but useless for queries.
The most useful words are relatively rare . . . but there are lots of them.

$$f_i = \frac{c}{R_i} \tag{3}$$

- The frequency of a word *i* is inversly proportional to
- The rank (in frequency) of word
- Scaled by a constant

Can't just throw out useless words

Frequency of Terms (Zipf's Law)

The most frequent words ("the") are everywhere but useless for queries.
The most useful words are relatively rare ... but there are lots of them.

$$f_i = \frac{c}{R_i} \tag{3}$$

- The frequency of a word *i* is inversly proportional to
- The rank (in frequency) of word
- Scaled by a constant

Can't just throw out useless words

Frequency of Terms (Zipf's Law)

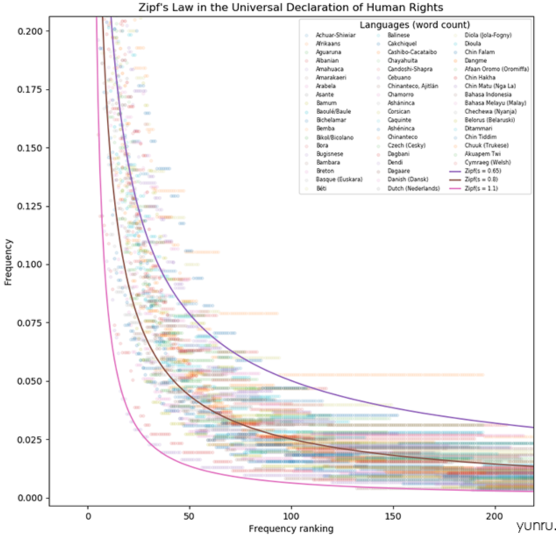
The most frequent words ("the") are everywhere but useless for queries.
The most useful words are relatively rare . . . but there are lots of them.

$$f_i = \frac{c}{R_i} \tag{3}$$

- The frequency of a word *i* is inversly proportional to
- The rank (in frequency) of word
- Scaled by a constant

Can't just throw out useless words

Zipf's Law



Zipf's Law in the Universal Declaration of Human Rights
Languages (word count)
Achuar-Shiwiar
Afrikaans
Aguaruna
Albanian
Amahuaca
Amarakaeri
Arabela
Asante
Bamum
Baoulé/Baule
Bichelamar
Bemba
Bikol/Bicolano
Bora
Bugisnese
Bambara
Breton
Basque (Euskara)
Bèti
Balinese
Cakchiquel
Cashibo-Cacataibo
Chayahuita
Candoshi-Shapra
Cebuano
Chinanteco, Ajitlán
Chamorro
Asháninca
Corsican
Caquinte
Ashéninca
Chinanteco
Czech (Cesky)
Dagbani
Dendi
Dagaare
Danish (Dansk)
Dutch (Nederlands)
Diola (Jola-Fogny)
Dioula
Chin Falam
Dangme
Afaan Oromo (Oromiffa)
Chin Hakha
Chin Matu (Nga La)
Bahasa Indonesia
Bahasa Melayu (Malay)
Chechewa (Nyanja)
Belorus (Belaruski)
Ditammari
Chin Tiddim
Chuuk (Trukese)
Akuapem Twi
Cymraeg (Welsh)
Zipf(s = 0.65)
Zipf(s = 0.8)
Zipf(s = 1.1)
Frequency
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0
50
100
150
200
Frequency ranking
yunru.

Similarity Metric

- "Angle" between vectors

$$\cos(\theta) = \frac{\vec{d}_j \cdot \vec{d}_k}{|\vec{d}_j||\vec{d}_k|} \tag{4}$$

- More generally, dot (inner) product ... normalized vectors

$$\text{sim}(d_j, d_k) = \vec{d}_j \cdot \vec{d}_k = \sum_{i=1}^{n} w_{i,j} w_{i,k} \tag{5}$$

Not just for documents . . .

- Representations central to modern NLP
- Everything's a vector . . .

Not just for documents . . .

- Representations central to modern NLP
- Everything's a vector . . .
- compare everything with cosine / dot products