Information Retrieval and Web Search

Cornelia Caragea

Computer Science
University of Illinois at Chicago

Credits for slides: Mooney

Text Processing

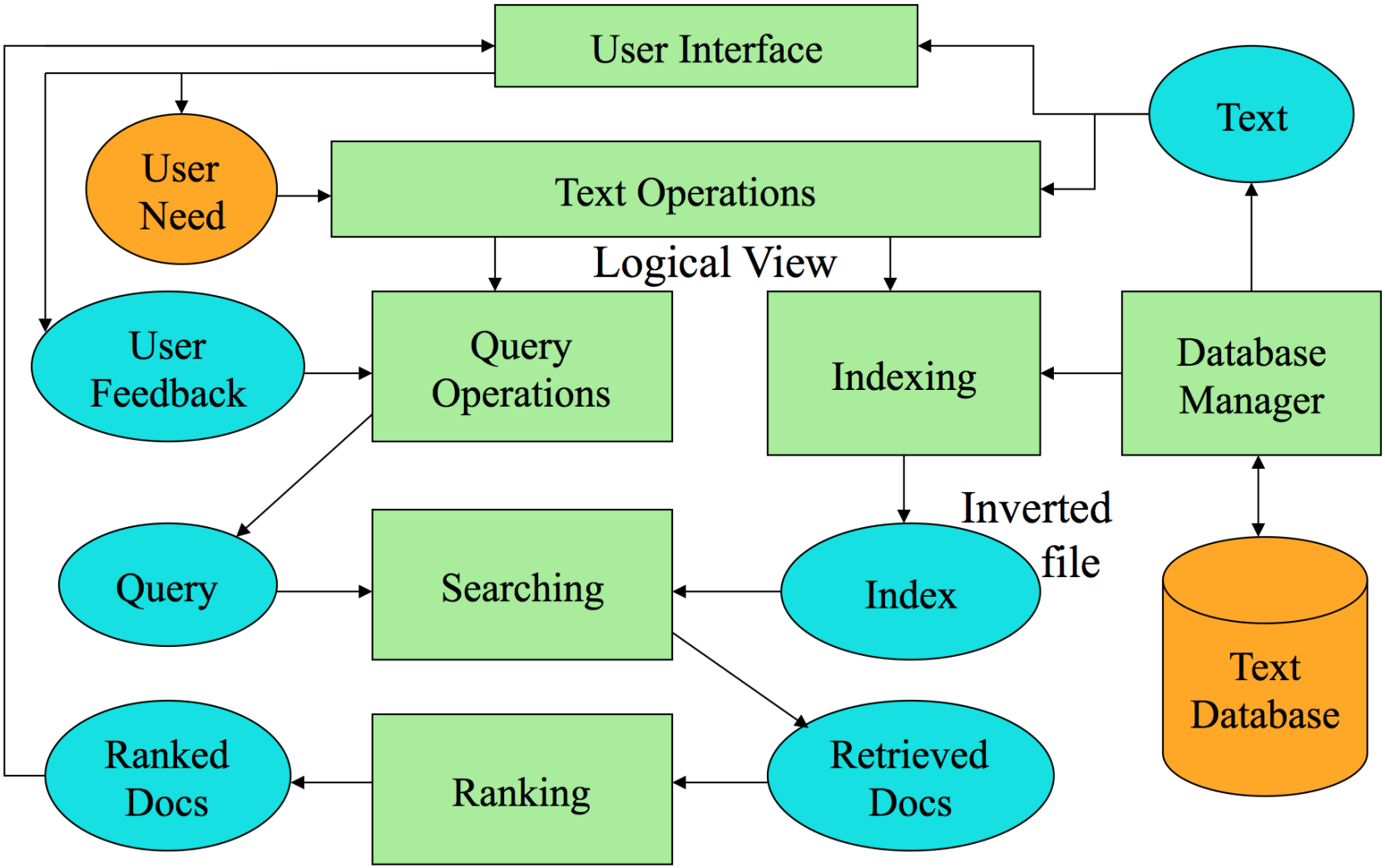
Required Reading

- "Information Retrieval" textbook
  - Chapter 2

Announcements

- TA: Tiberiu Sosea <tsosea2@uic.edu>
- Piazza registration?
- HW 1 will be out soon.
- No class on September 6 (Labor Day).

IR System Architecture



The Term Vocabulary

- Text Operations form index words (tokens)
  - Tokenization
  - Stop-word removal
  - Stemming

Obtaining the Character Sequence in a Document

- Digital documents are typically bytes in a file.
- First step of processing: convert this byte sequence into a sequence of characters
  - Trivial for plain English text in ASCII encoding
  - Need to decode byte to character sequences for various single byte or multibyte encoding schemes, such as Unicode UTF-8
  - The textual part of a document may need to be extracted out of other material, e.g., PS, PDF, or PPT files.

Choosing a Document Unit

- Second step of processing: determine what the document unit for indexing is
  - Take each file in a folder as a document
  - Other cases:
    - A sequence of email messages stored in one file, but you may wish to regard each email message as a separate document
    - Email message and attachment - two separate documents
    - Posts in a thread - separate documents
  - Indexing Granularity may occur for very large documents such as books
    - Not a good idea to index an entire book as a document
    - For example, a search for *Chinese toys* might bring up a book that mentions China in the first chapter and toys in the last chapter, but this does not make it relevant to the query.
    - A better solution: index each chapter or paragraph as a mini-document.

Determining the Vocabulary of Terms

- Tokenization: splitting up a character sequence into tokens
  - Tokens: sequences of characters with a semantic meaning

Input: Friends, Romans, Countrymen, lend me your ears;

Output: | Friends | Romans | Countrymen | lend | me | your | ears |

- Split on whitespace and throw away punctuation
- Problematic cases, e.g., uses of the apostrophe for possession and contractions.

- "Mr. O'Neill thinks that the boys' stories about Chile's capital aren't amusing."

Determining the Vocabulary of Terms

- Tokenization: splitting up a character sequence into tokens
  - Tokens: sequences of characters with a semantic meaning

Input: Friends, Romans, Countrymen, lend me your ears;
Output: | Friends | Romans | Countrymen | lend | me | your | ears |

- Split on whitespace and throw away punctuation
- Problematic cases, e.g., uses of the apostrophe for possession and contractions.

- "Mr. O'Neill thinks that the boys' stories about Chile's capital aren't amusing."
- How many matches for the query neill AND capital, if exact match?

neill
oneill
o'neill
o' neill
o neill ?

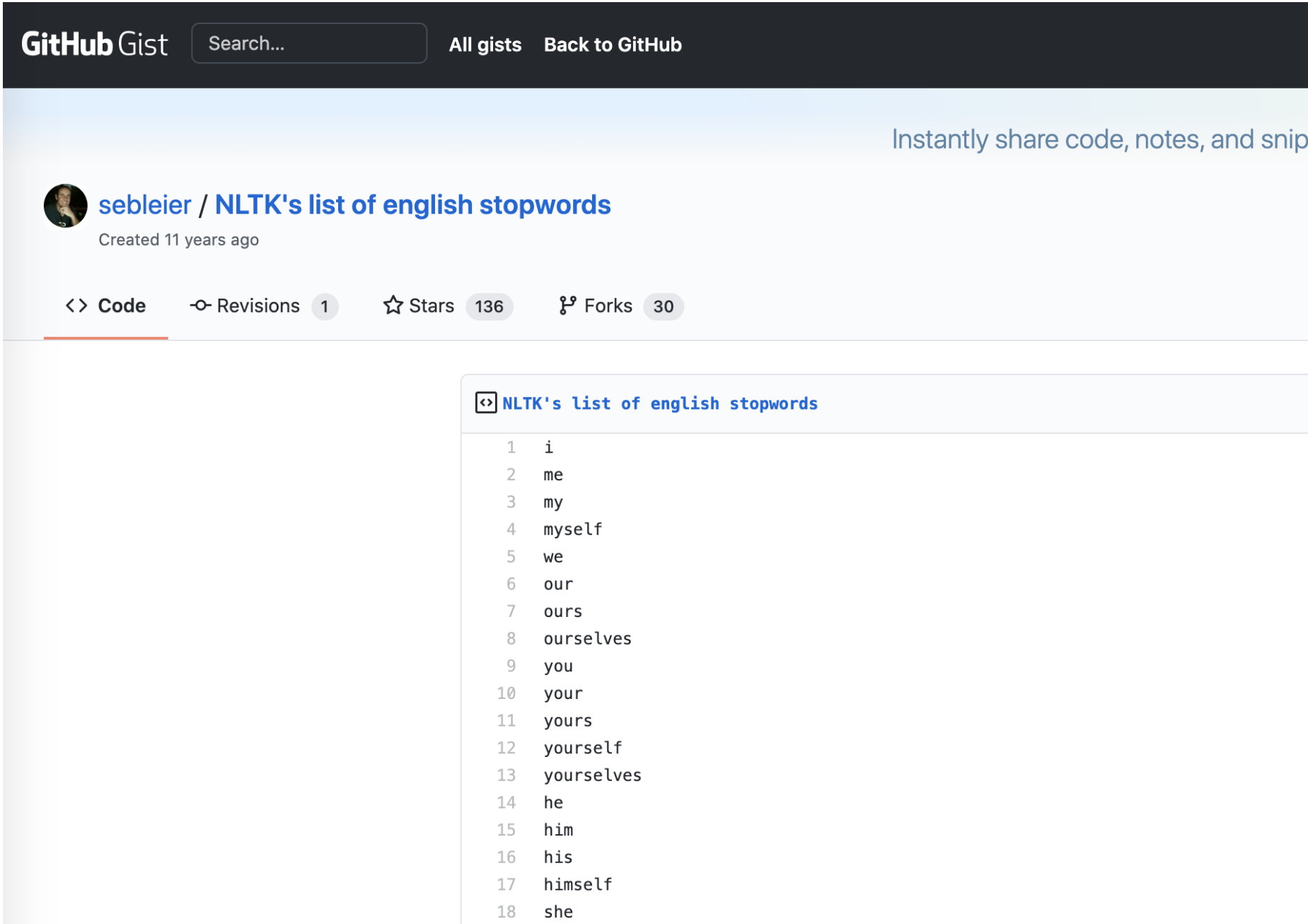
Further Notes on Tokenization

- Generally do the exact same tokenization of document and query words, e.g., by processing queries with the same tokenizer.
- The problems with tokenization are language-specific.
- Sometimes punctuation (e-mail), numbers (1917), and case (Bush vs. bush) can be a meaningful part of a token.
  - Simplest approach is to ignore all numbers and punctuation and use only case-insensitive unbroken strings of alphabetic characters as tokens.
  - Another approach is to preserve special terms:
    - C++, C#
    - M*A*S*H TV show
    - email addresses, web URLs, IP addresses, etc.

Dropping Common Terms: Stop Words

- Extremely common words, or *stop words*, may appear to be of little value in helping select documents matching a user need and hence, are excluded from the vocabulary.
- https://gist.github.com/sebleier/554280
- However, not indexing stop words may be harmful, e.g. "President of the United States" vs. "President AND United States" or "To be or not to be".
- Web search engines generally do not use stopword lists.

A Stop Words List



Normalization (Equivalence Classes of Terms)

- Easy case: match tokens in the query with tokens in the token list of a document.
- More complex cases: two character sequences are not the same but a match between them is desirable.
  - For example, searching for USA hopefully will match documents that contain U.S.A.
  - Token normalization is the process of canonicalizing tokens so that matches occur despite superficial differences.
    - Create *equivalence classes*, e.g., "anti-discriminatory" and "antidiscriminatory" will be mapped onto "antidiscriminatory", in both the document text and queries.
    - Maintain relations between unnormalized tokens, e.g., "car" and "automobile".
    - Equivalence classes or query expansion is a fairly open research question.
    - Doing it seems a good idea, doing it too much may have unexpected outcome, e.g., CAT vs. cat.

Normalization (Equivalence Classes of Terms) (II)

- Accents and diacritics
  - May want cliche and cliché or naive and naïve to match - remove diacritics.
  - However, in some languages, diacritics are a regular part of the writing system and distinguish different sounds.
  - Words may have different meaning depending on the use of diacritics, e.g. in Spanish, penã is "a cliff", whereas pena is "sorrow."
- Capitalization/case-folding: reducing all letters to lower case
  - May want "Automobile" at the beginning of a sentence to match a query "automobile".
  - Use heuristics to convert to lowercase words at the beginning of a sentence and all words occurring in a title.
  - Some terms are distinguished only by case: "General Motors", "Bush". Do not reduce all words to lower case for upper case words that occur in the middle of sentences.

Lemmatization

- Lemmatization usually refers to returning the base or dictionary form of a word (aka *lemma*).
- Direct impact on the vocabulary size.

am, are, is ⇒ be
car, cars, car's, cars' ⇒ car

the boy's cars are different colors ⇒
the boy car be differ color

- How to do this?
  - Need a list of grammatical rules + a list of irregular words
  - Children → child, spoken → speak ...

Stemming

- Stemming usually refers to a crude heuristic process that chops off the ends of words.
  - "computer", "computational", "computation" all reduced to the same token "comput"
- Correct morphological analysis is language specific and can be complex.

Porter Stemmer: http://tartarus.org/~martin/PorterStemmer/

Typical Rules and Errors in Porter Stemmer

- Typical Rules:
  - *sses* $\rightarrow$ *ss* (as in "princesses" –> "princess")
  - *ies* $\rightarrow$ *i* (as in "butterflies" –> "butterfli")
- Errors of "commission"
  - organization, organ $\rightarrow$ organ
  - police, policy $\rightarrow$ polic
  - arm, army $\rightarrow$ arm
- Errors of "omission"
  - cylinder, cylindrical

Statistical Properties of Text

- Questions:
  - How is the frequency of different words distributed?
  - How fast does vocabulary size grow with the size of a corpus?
- Such factors affect the performance of information retrieval and can be used to select appropriate term weights and other aspects of an IR system.

Word Frequency

- A few words are very common.
  - 2 most frequent words (e.g., “the”, “of”) can account for about 10% of word occurrences.
- Most words are very rare.
  - Half the words in a corpus appear only once
- Called a “heavy tailed” distribution, since most of the probability mass is in the “tail”

Sample Word Frequency Data (from B. Croft, UMass)

Frequent Word	Number of Occurrences	Percentage of Total
the	7,398,934	5.9
of	3,893,790	3.1
to	3,364,653	2.7
and	3,320,687	2.6
in	2,311,785	1.8
is	1,559,147	1.2
for	1,313,561	1.0
The	1,144,860	0.9
that	1,066,503	0.8
said	1,027,713	0.8

Frequencies from 336,310 documents in the 1GB TREC Volume 3 Corpus
125,720,891 total word occurrences; 508,209 unique words

Other More Recent Collections

- Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncased)
- Common Crawl (42B tokens, 1.9M vocab, uncased)
- Common Crawl (840B tokens, 2.2M vocab, cased)
- Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased)

Source: https://nlp.stanford.edu/projects/glove/

Zipf's Law

- Rank (r): The numerical position of a word in a list sorted by decreasing frequency (f).
- Zipf (1949) "discovered" that:

$$f \cdot r \approx k, \text{ for constant k}$$

- If probability of word of rank r is p_r and N is the total number of word occurrences:

$$p_r = \frac{f}{N} \approx \frac{A}{r}, \text{ for corpus independent constant A} \approx 0.1$$

Zipf and Term Weighting

- Luhn (1958) suggested that both extremely common and extremely uncommon words were not very useful for indexing.

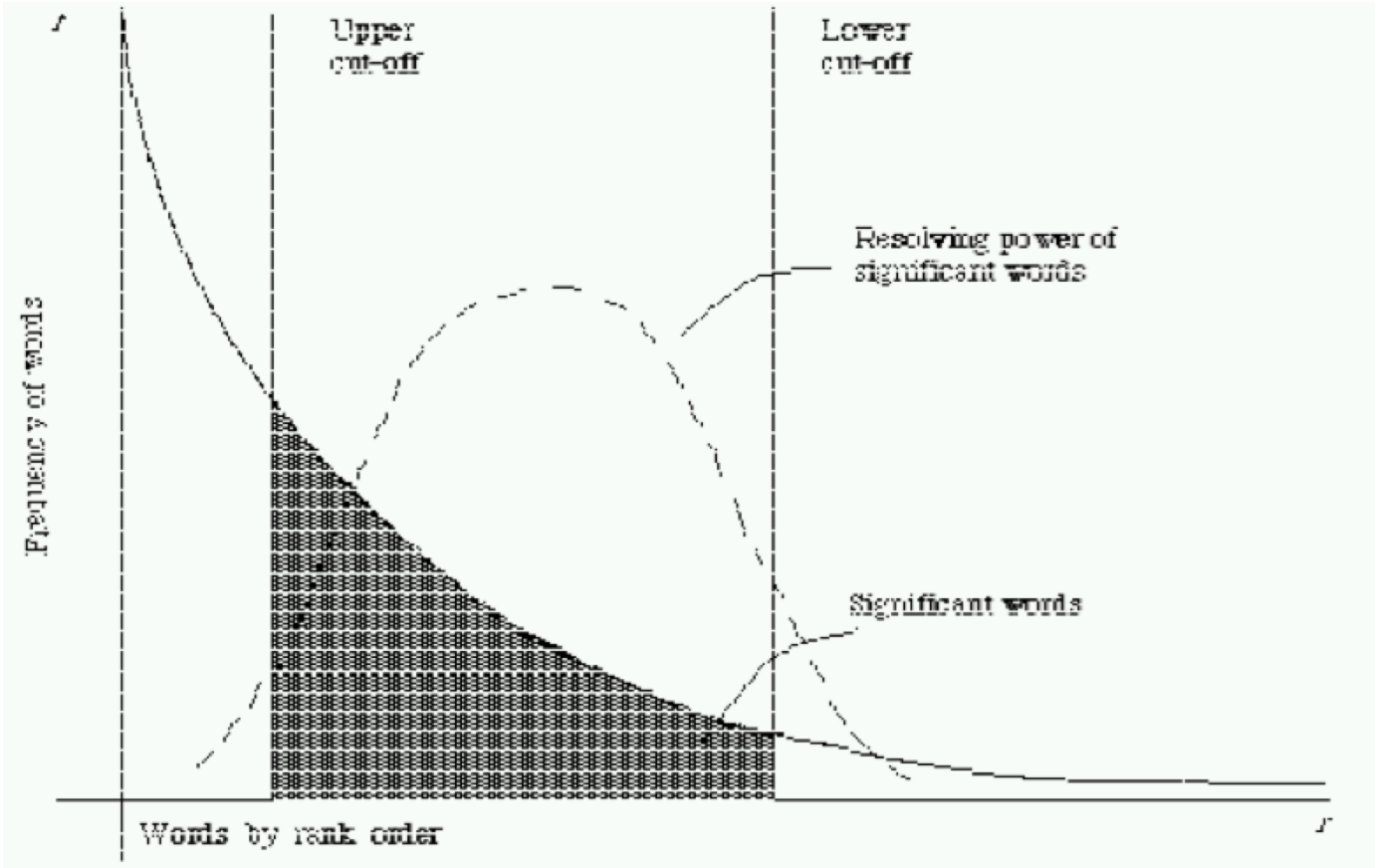


Figure 2.1. A plot of the hyperbolic curve relating f, the frequency of occurrence and r, the rank order (Adapted from Schultz [44] page 120)

Does Real Data Fit Zipf's Law?

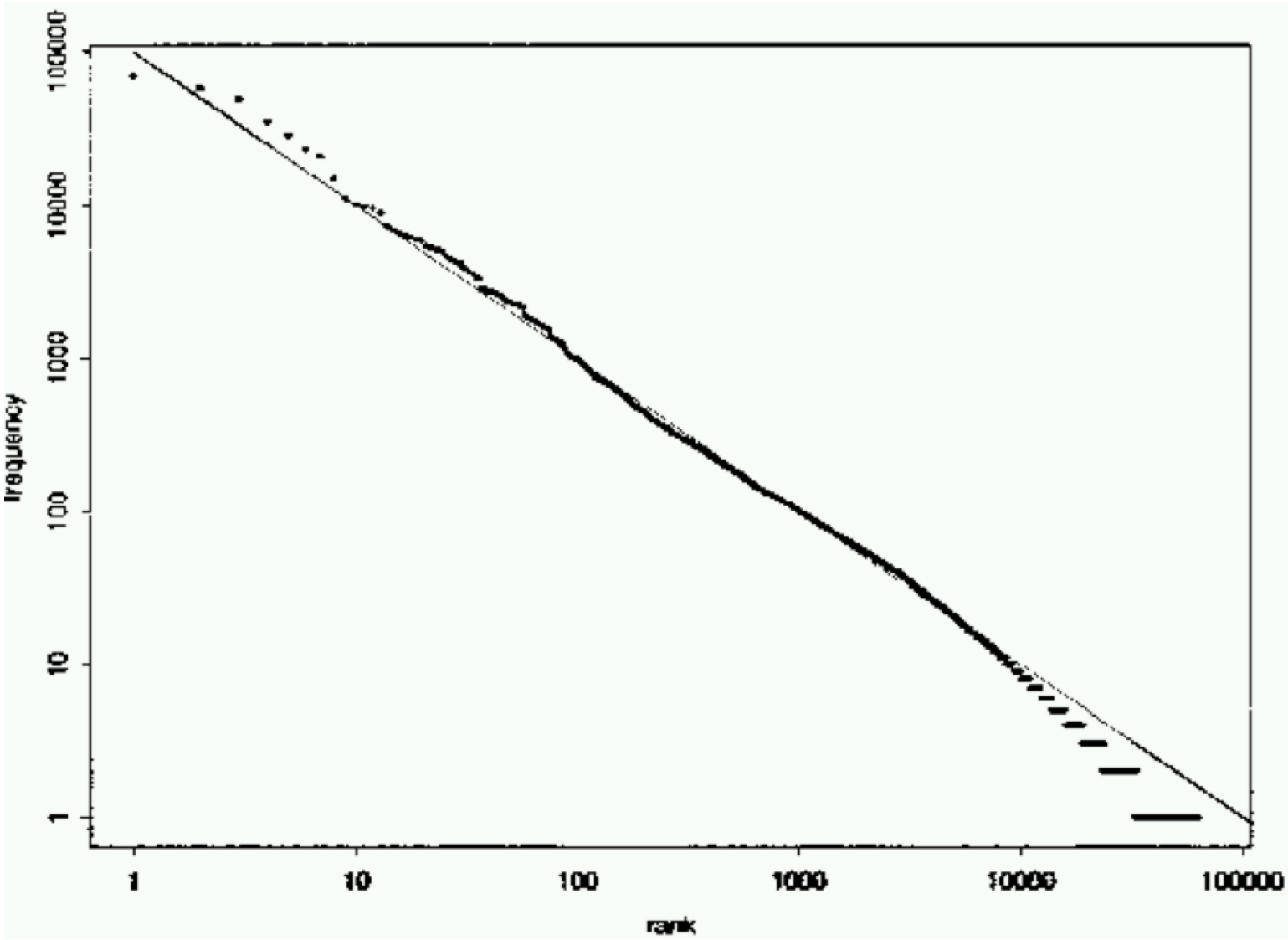
- A law of the form $y = kx^c$ is called a power law.
- Zipf's law is a power law with $c = -1$.
- On a log-log plot, power laws give a straight line with slope c.

$$\log(y) = \log(kx^c) = \log k + c \log(x)$$

- Zipf is quite accurate except for very high and low rank.

Fit to Zipf for Brown Corpus

$k = 100,000$



Zipf's Law Impact on IR

- Good News: Stopwords will account for a large fraction of text so eliminating them greatly reduces inverted-index storage costs.
- Bad News: For most words, gathering sufficient data for meaningful statistical analysis (e.g., for correlation analysis for query expansion) is difficult since they are extremely rare.

Vocabulary Growth

- How does the size of the overall vocabulary (number of unique words) grow with the size of the corpus?
- This determines how the size of the inverted index will scale with the size of the corpus.
- Vocabulary is not really upper-bounded due to proper names, typos, etc.

Heaps' Law

- If V is the size of the vocabulary and n is the length of the corpus in words:

$$V \approx kn^{\beta} \text{ with constants } k,\ 0 < \beta < 1$$

- Typical constants:
  - $k \approx 10 - 100$
  - $\beta \approx 0.4 - 0.6$

Next

- Boolean Retrieval Models
- Vector Space Models