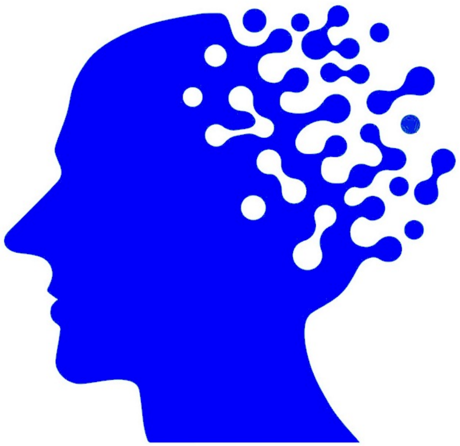

Monsoon 2021

Term Weighting

- Vector Space, 5 Different Term Weighting approaches

Dr. Rajendra Prasath

Indian Institute of Information Technology Sri City, Chittoor

3rd September 2021 (rajendra.2power3.com)

> Topics to be covered

- Recap:
  - Phrase Queries / Proximity Search
  - Spell Correction / Noisy Channel Modelling
  - Index Construction
    - BSBI / SPIMI
    - Distributed Indexing
  - Vector Space Models

- Term Weighting Approaches
  - 5 Different Approaches
    - An Illustration

- More topics to come up … Stay tuned …!!

Overview

Recap: Information Retrieval

- **Information Retrieval (IR)** is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).

- These days we frequently think first of web search, but there are many other cases:
  - E-mail search
  - Searching your laptop
  - Corporate knowledge bases
  - Legal information retrieval
  - and so on . . .

Bag of words model

- We do not consider the order of words in a document.
- John is quicker than Mary and Mary is quicker than John are represented in the same way.
- This is called a bag of words model.
- In a sense, this is a step back: The positional index was able to distinguish these two documents.
- We will look at "recovering" positional information later in this course.
- For now: bag of words model

Term frequency (tf)

- The term frequency $tf_{t,d}$ of term t in document d is defined as the number of times that t occurs in d
- Use tf to compute query-doc. match scores
- Raw term frequency is not what we want
- A document with tf = 10 occurrences of the term is more relevant than a document with tf = 1 occurrence of the term
- But not 10 times more relevant
- Relevance does not increase proportionally with term frequency

Log frequency weighting

- The log frequency weight of term t in d is defined as follows

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d} & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- $\mathrm{tf}_{t,d} \rightarrow w_{t,d}$: $0 \rightarrow 0$, $1 \rightarrow 1$, $2 \rightarrow 1.3$, $10 \rightarrow 2$, $1000 \rightarrow 4$, etc.

- Score for a document-query pair: sum over terms t in both q and d:

- tf-matching-score(q, d) = $\sum_{t \in q \cap d} (1 + \log \mathrm{tf}_{t,d})$

- The score is 0 if none of the query terms is present in the document

Term Weighting

- The Importance of a term increases with the number of occurrences of a term in a text.

- So we can estimate the term weight using some monotonically increasing function of the number of occurrences of a term in a text

- Term Frequency:
- The number of occurrences of a term in a text is called Term Frequency

Term Frequency Factor

- What is Term Frequency Factor?
  - The function of the term frequency used to compute a term's importance

  - Some commonly used factors are:
    - Raw TF factor
    - Logarithmic TF factor
    - Binary TF factor
    - Augmented TF factor
    - Okapi's TF factor

The Raw TF factor

- This is the simplest factor
- This counts simply the number of occurrences of a term in a text
- Simply count the number of terms in each document
- More the number, higher the ranking of the document!!

The Logarithmic TF factor

- This factor is computed as

$$1 + \ln(tf)$$

where tf is the term frequency of a term

- Proposed by Buckley (1993 - 94)

- **Motivation:**
  - If a document has one query term with a very high term frequency then the document is (often) not better than another document that has two query terms with somewhat lower term frequencies
  - More occurrences of a match should not out-contribute fewer occurrences of several matches

Example – log TF factor

- Consider the query: "recycling of tires"
- Two documents:

  D1: with 10 occurrences of the word "recycling"

  D2: with "recycling" and "tires" 3 times each
- Everything else being equal, if we use raw tf, D1(10) gets higher similarity score than D2 (3+3=6)
  - But D2 addresses the needs of the query
  - Log TF: reflects usefulness of D2 in similarities

  D1: $1 + \ln(10) = 3.3$ and

  D2: $2(1+\ln(3)) = 4.1$

The Binary TF factor

- ✧ The TF factor is completely disregards the number of occurrences of a term.
- ✧ It is either one or zero depending upon the presence (one) or the absence (zero) of the term in a text.
- ✧ This factor gives a nice baseline to measure the usefulness of the term frequency factors in document ranking

The Augmented TF factor

- This TF factor reduces the range of the contributions of a term from the freq. of a term
- **How:** Compress the range of the possible TF factor values (say between 0.5 & 1.0)
  - The augmented TF factor is used with a belief that mere presence of a term in a text should have some default weight (say 0.5)
  - Then additional occurrences of a term could increase the weight of the term to some max. value (usually 1.0).

Augmented TF factor - Scoring

- This TF factor is computed as follows:

$$0.5 + 0.5 \times \frac{tf}{\text{maximum tf in text}}$$

- The augmented TF factor emphasizes that more matches are more importance than fewer matches (like log TF factor)

- A single match contributes at least 0.5 and high TFs can only contribute at most another 0.5

- This was motivated by document length considerations and does not work as well as log TF factor in practice.

Okapi's TF factor

- Robertson et. Al (1994) developed Okapi Information Retrieval System and proposed another TF factor

- This TF factor is based on Approximations to the 2-Poisson Model:

- This factor

$$\frac{tf}{2 + tf}$$

is quite close to the log TF factor in its behavior

- In practice, log TF factor is effective for good document ranking

Exercise – Try Yourself

- Consider a collection of n documents
- Let n be sufficiently large (at least 100 docs)
  - You can take our dataset
  - Sports News Dataset (ths-181-dataset.zip)

- **Find two lists:**
  - The most frequency words and
  - The least frequent words
  - Form k (=10) queries each with exactly 3-words taken from above lists (at least one from each)
  - Compute Similarity between each query and and documents

Summary

In this class, we focused on:

(a) Recap: Positional Indexes

i. Wild card Queries

ii. Spelling Correction

iii. Noisy Channel modelling for Spell Correction

(b) Various Indexing Approaches

i. Block Sort based Indexing Approach

ii. Single Pass In Memory Indexing Approach

iii. Distributed Indexing using Map Reduce

iv. Examples

Acknowledgements

Thanks to ALL RESEARCHERS:

1. Introduction to Information Retrieval Manning, Raghavan and Schutze, Cambridge University Press, 2008.
2. Search Engines Information Retrieval in Practice W. Bruce Croft, D. Metzler, T. Strohman, Pearson, 2009.
3. Information Retrieval Implementing and Evaluating Search Engines Stefan Büttcher, Charles L. A. Clarke and Gordon V. Cormack, MIT Press, 2010.
4. Modern Information Retrieval Baeza-Yates and Ribeiro-Neto, Addison Wesley, 1999.
5. Many Authors who contributed to SIGIR / WWW / KDD / ECIR / CIKM / WSDM and other top tier conferences
6. Prof. Mandar Mitra, Indian Statistical Institute, Kolkatata (https://www.isical.ac.in/~mandar/)

Questions
It's Your Time

Contact Information:

Dr. Rajendra Prasath
IIIT Sri City, Chittoor