Chapter 3: Representations of Documents and Information Needs

Overview

- Representations of documents:
  - natural language terms = terms of the text
  - controlled language terms = fixed descriptor terms
  - term weighting
  - detection of collocations

- Representations of information needs:
  - queries
  - relevance feedback
  - personalized and contextualized information needs

Problem definition

- Representations that are easy to process by a computer are made of documents and information needs
- Challenge:
  - to capture as much as possible information and meaning
  - to delimit as much as possible storage overhead and computational complexity at the time of querying

[see Indexing structures and search techniques]

Representation composed of natural language terms

- Basic processes:

  0) preparation of the document: e.g. removal of tags
  1) lexical analysis
  2) removal of stopwords (optional)
  3) stemming (optional)
  4) term weighting (optional)

SWARM INTELLIGENCE

Following a trail of insects as they work together to accomplish a task offers unique possibilities for problem solving.

By Peter Tarasewich & Patrick R. McMullen

Stemming

Even with today's ever-increasing computing power, there are still many types of problems that are very difficult to solve. Particularly combinatorial optimization problems continue to pose challenges. An example of this type of problem can be found in product design. Take as an example the design of an automobile based on the attributes of engine horsepower, passenger seating, body style and wheel size. If we have three different levels for each of these attributes, there are 3^4, or 81, possible configurations to consider. For a slightly larger problem with 5 attributes of 4 levels, there are suddenly 1,024 combinations. Typically, an enormous amount of possible combinations exist, even for relatively small problems. Finding the optimal solution to these problems is usually impractical. Fortunately, search heuristics have been developed to find good solutions to these problems in a reasonable amount of time.

Removal of stopwords

heuristic: weight = 2

~~Over~~ ~~the~~ past decade ~~or~~ ~~so~~, several heuristic techniques have been developed ~~that~~ ~~build~~ upon observations of processes in the physical and biological sciences. Examples of these techniques include Genetic Algorithms (GA) and simulated annealing…

Lexical analysis

- Tokenization = converting input stream of characters into a stream of words or tokens
  - **space-delimited languages** (most European languages):
    - word = string of characters separated by white space
  - **unsegmented languages** (e.g., Chinese, Thai, Japanese):
    - e.g., use of a word list (MRD = machine-readable dictionary)

Lexical analysis

- Difficulties:
  1. use of special characters:
     - e.g., period in abbreviation can be confused with words that end with a full stop at the end of sentence
     - apostrophes
     - hyphens
     - => need for language specific rules
  2. normalization of numbers

Lexical analysis

- Use of a finite state automaton or finite state machine:
  - often integrates:
    - transformations (e.g., case of letters, abbreviations and acronyms)
    - removal of stopwords

[see Indexing structures and search techniques]

Removal of stopwords

- **Stoplist** = a machine-readable list of words (**stopwords**) that are non-content-bearing:
  - e.g. (English), a, about, above, across, after, ...
- Advantages: more efficient indexing, storage, search
- But, what if ?
  - phrase queries: “President of France”, “flight to London”, ..
- Construction based on:
  - **Part-Of-Speech (POS) information**: generic, but language-dependent
  - **threshold frequency of occurrence** of words in document corpus
  - zero ***idf***-values (see below)

Lemmatization

- **Lemmatization** = finding the lemma or lexeme of an inflected word form (the lemma is the canonical dictionary entry form of the word)
- Lookup of terms and their lemma in a **machine-readable dictionary (MRD)**:
  - correct (e.g., ponies -> pony)
  - often large lists that need to be searched efficiently
  - not always available or not all words covered

Stemming

- **Stemming** = reducing the morphological variants of the words to their stem or root
- **Affix removal algorithms**:
  - language dependent rules to remove suffixes and/or prefixes from terms leaving a stem
  - possible language dependent transformation of the resulting stem
  - examples:
    - Lovins stemmer (1968)
    - **Porter algorithm** (1980)

Porter's algorithm

- English: 5 phases of word reductions applied sequentially
- Application of rules with priorities (e.g., removal of longest suffix)
- E.g., in the first phase

Rule	Example
SSES ->SS	caresses -> caress
IES -> I	ponies -> poni
SS -> SS	caress -> caress
S ->	cats -> cat

- Measure (*m*) of a word: is it long enough that the matching portion can be a suffix?
  - E.g., ($m > 1$): EMENT ->
- http://www.tartarus.org/~martin/PorterStemmer/

Stemming

- Advantages of stemming:
  - efficient indexing, storage, search
  - useful when the morphology of a language is rich (e.g., Hungarian, Croatian, Hebrew)
  - assumption: words with the same stem are semantically related or have same meaning:
    - removal of inflectional morphemes (cf. lemmatization): usually SAFE
    - removal of derivational morphemes: CAUTION!

Word splitting

- **Splitters** for compound nouns in languages such as German and Dutch:
  - e.g., Dutch “onroerendgoedmarkt” (market of real estate) to split in “onroerend” “goed” “markt”
  - might combine stemming (e.g., to split Dutch “rechtsleer” in “recht” and “leer”)

Representation composed of controlled language terms

- Problem with natural language terms:
  - semantic ambiguity
  - often not suited for generic searches
- **Controlled language terms**:
  - thesaurus terms, subject and classification terms from ontology
  - traditionally important in information retrieval, manually assigned
  - currently automatically assigned [see Text categorization]

Thesaurus

- Thesaurus: form of machine-readable dictionary (MRD)
- Typical term relationships captured in a thesaurus:
  - **Synonymy** (terms of a synset)
  - **Hypernymy** / **hyponymy**
- Often contains **word meanings**: to disambiguate word senses [see Cross-language information retrieval]

Thesaurus construction and maintenance

- Hand-built:
  - examples: Longman's Dictionary Of Contemporary English (LDOCE), WordNet, CELEX, ...

- Automatically or semi-automatically built: attempt to identify semantic relationships between words from large corpus based on:
  - statistical patterns [see Text clustering]
  - syntactic patterns (e.g., constructing a hierarchical thesaurus from head-modifier relations of noun phrases)
  - combination of the above

Term weighting

- **Weight**
  - = importance indicator of a term regarding content
  - used in certain retrieval models (e.g., vector space model [see Retrieval models]
  - used for term filtering

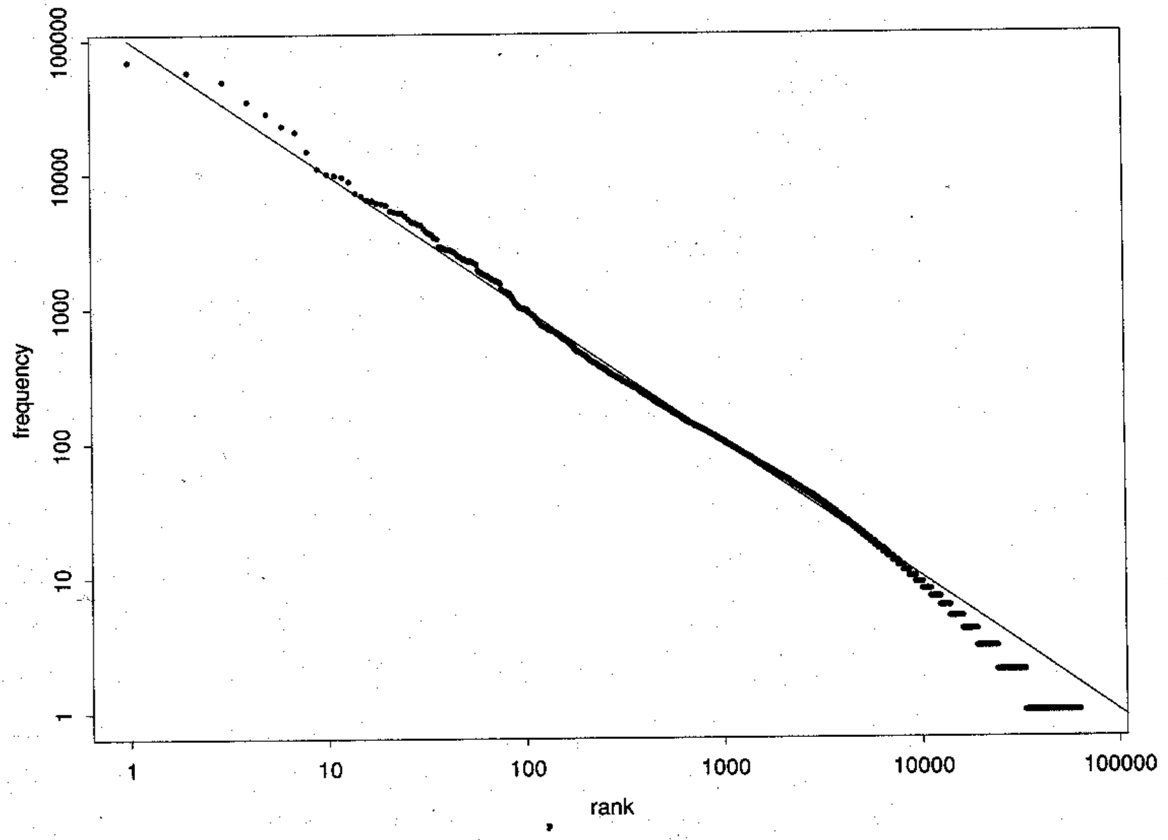
Law of Zipf

- **Zipf** (1949):
  - plotted the logarithm of the frequency of a term in a body of texts against rank (highest frequency term has rank 1, second highest frequency term has rank 2, etc.)
  - for a large body of text of “well-written English”, the resulting curve is nearly a straight line
  - other languages or other writing styles may be expressed by other non-linear functions

- **Constant rank-frequency law of Zipf:**

$$\log(\text{frequency}) \cdot \text{rank} = \text{constant}$$

Law of Zipf



The points correspond to the ranks and frequencies of the words in one corpus (the Brown corpus) using logarithmic scales. The line is the relationship between rank and frequency predicted by Zipf for $k = 100,000$, that is $f \times r = 100,000$

[Manning & Schütze, 1999]

Law of Zipf

- Best indexing terms lay in medium frequency range:
  - has influenced classical weighting functions for determining the weight w_{ij} of index term i in the representation of document d_j

Classical weighting functions

- $w_{ij} = tf_i$ = **term frequency**
  - frequency of occurrence of the index term *i* in the text
- Assumptions:
  - high frequency content-bearing terms = main topics of the text
  - term that occurs with a frequency higher than one would expect in a certain passage = subtopic of the text

Classical weighting functions

- **Inverse document frequency** (*idf*) weight = collection dependent weight often computed as:

$$w_{ij} = \log\left(\frac{N}{n_i}\right)$$

where:

N = number of documents in the reference collection

n_i = number of documents in the reference collection having index term *i*

- Assumption:
  - common words that are distributed over numerous texts = poor indicators of a text’s content

Classical weighting functions

- **Product of the term frequency and inverse document frequency (*tf* x *idf*):**

$$w_{ij} = tf_i \cdot \log\left(\frac{N}{n_i}\right)$$

Classical weighting functions

- **Length normalization**: normalized term frequency

$$\frac{tf_i}{\max tf_j}$$

where

tf_j = term frequency of an index term j in the text

j = 1,..., l (l = number of distinct index terms in the text)

- Variant: **augmented normalized term frequency:**

$$\alpha + (1-\alpha)\left(\frac{tf_i}{\max tf_j}\right)$$

weighted by (1- α) to decrease the difference in weights of frequent and of infrequent terms in the text

smoothing term α (generally set to 0.5)

Classical weighting functions

- Length normalization is used in cases when:
  - term frequency is misleading when texts have different lengths
  - but you might prefer to retrieve long texts about a topic

Detection of compound terms

- Sometimes individual words do not carry a precise meaning
- **Phrase chunks** or **full phrases** as descriptors might be a solution
- However, most interesting to detect are **collocations**

Recognition of phrases

1) Use of a **machine-readable dictionary (MRD)** with phrases:
   - only practical in restricted subject domains

2) **Automated recognition**:
   - detection of base noun or verb phrases:
     - Part-Of-Speech tagging + chunking
   - use of a sentence parser

Part-Of-Speech tagging

- **POS tagging**: often use of a stochastic tagger
  - computes the probability that a word class should be assigned to a word
  - this probability takes into account:
    - lexical probability: probability that class is assigned to word
    - contextual probability: probability that class is appropriate for the particular context
  - lexical and contextual probabilities are learned from manually tagged example texts (e.g., training of a hidden Markov model)

Chunking

- **Chunker**:
  - uses predefined syntactic templates for detecting phrase chunks
- Example of a part-of-speech tagged and chunked text:

  <S>[[The_DT* Turkish_NNP government_NN]],_, [[which_WDT]] ((sent_VBD)) about_IN [[10,000_CD soldiers_NNS]] into_IN [[northern_JJ Iraq_NNP]] ((to_TO attack_VB)) [[Kurdish_JJP rebels_NNS]],_, ((has_VBZ said_VBN)) [[it_PRP]] ((might_MD send_VB)) [[forces_NNS]] into_IN [[Syria_NNP]] ((to_TO eradicate_VB)) [[guerrilla_JJ bases_NNS]] there_RB,_, according_VBG to_TO [[news_NN reports_NNS]]._.</S>

*Penn Treebank tag set

- Good POS-taggers and chunkers are available for many languages that operate with very low error rates: e.g.,
  - English:
    - POS-tagging:
      - LT POS
      - Qtag
      - TnT
      - MXPOST
    - Chunking:
      - BaseNP Chunker
      - LT CHUNK

Sentence parsing

- **Sentence parser**:
  - syntactic parse of each sentence (e.g., captured in dependency tree)
  - usually trained on annotated examples (e.g., maximum entropy modeling)
- Good parsers are available for many languages that operate with low error rates:
  - English: Charniak’s parser, Collin’s parser, Daniel Bikel’s parser
  - Dutch: Alpino parser

Example of syntactic parse captured in dependency tree

```
(S1 (S (NP (NNP Lady) (NNP Hera))
    (VP  (VBD  was)
     (NP (NP (DT a)
         (ADJP   (JJ jealous)
            (,  ,)
           (JJ   ambitious)
           (CC   and)
          (JJ   powerful))
           (NN woman))
      (SBAR (WHNP (WP who))
  (S   (VP   (VBD   was)
          (ADVP   (RB continually))
         (VP   (VB   irated)
              (PP  (IN  over)
               (NP (NP (NP (NNP Zeus) (POS ')) (NN pursuit))
                (PP (IN of)
         (NP     (JJ     mortal) (CC and) (JJ immortal) (NN woman)))))))))))
    (. .)))
```

Problems

- Syntactical constituents of sentences might not be meaningful terms (e.g., “white and blue shiny jacket”)
- Same content can be expressed in many variants (e.g., “prenatal ultrasonic diagnosis” and “in utero sonographic diagnosis of the fetus”)

Collocations

- **Collocation** = expression consisting of two or more words that correspond to some conventional way of saying things
  - usually element of meaning added to the collocation that can not be predicted from the meanings of the composing parts
  - usually phrases
  - e.g., joint venture, make [something] up
  - based on term correlations [see Text clustering]

Special case: proper names

- Names of persons, companies, institutions, product brands, locations, currencies, proteins, …
- Detection and semantic classification: [see Information extraction]

Special case: term variants

- Term variants of proper names and technical terms:
  - e.g., protein: “NF-Kappa B”, “NF Kappa B”, “NF kappaB”, and “NFkappaB”
- Abbreviations and acronyms:
  - e.g., “International Business Machines” and Intl Business Machines”, “IR” and “information retrieval”
- Solutions
  - e.g., threshold of shared letter sequences (*n*-grams)
  - e.g., learning variation rules from training corpus
  - e.g., edit distance

Latent semantic models

- Latent Semantic Indexing
- Probabilistic Latent Semantic Analysis
- Latent Dirichlet Allocation

[see Advanced representations]

Additional data

- **Metadata:**
  - common metadata such as author, language, date, ...
  - metadata with regard to document structure
  - obtained with information extraction techniques: **semantic labels** [see Information extraction]
  - possibly marked with XML (Extensible Markup Language)
- **Links:**
  - incoming and outgoing hyperlinks [see Web information retrieval]
- **Tags**, **comments** of users

Query terms

- Usually natural language terms (single words or phrases)
- Possibly replaced or expanded by controlled language terms
- Possibly weighted terms, representing the importance of each term in the query (e.g., by user, by *idf*)

Query operators

- Boolean operators (∧,∨,¬)
- Proximity parameter: maximum distance between terms (e.g., in words)
- + and - before term:
    - +: term must appear
    - -: term must not appear
- Phrase indicators: e.g., use of quotes
- ...

Relevance feedback

- **Relevance feedback:**

  = a better query is learned from the retrieved documents judged relevant or non-relevant **by the user**

  (sometimes based on eye movement tracking or clicking behavior of the user)

- **Pseudo relevance feedback:**

  = a better query is learned from the top-ranking documents returned **by the system**

(Pseudo) relevance feedback

- Contributes to:

  1) **query expansion** [see Text clustering]

  2) **query term weight altering** : e.g., increasing the weight of a term that occurs in relevant documents, decreasing of the term weight when it occurs in the non-relevant documents:
     - e.g., Rocchio algorithm [see Text categorization]

(Pseudo) relevance feedback

- Improves the results of a retrieval operation: often feedback iterated until the user is completely satisfied
- Extensively studied in the Text REtrieval Conferences (TREC) (http://trec.nist.gov/)

Personalized and contextualized information needs

- = adding to the query representation extra information

- **Short term model**: relevance feedback, incorporation of contextual information
- **Long term model**: incorporation of contextual information

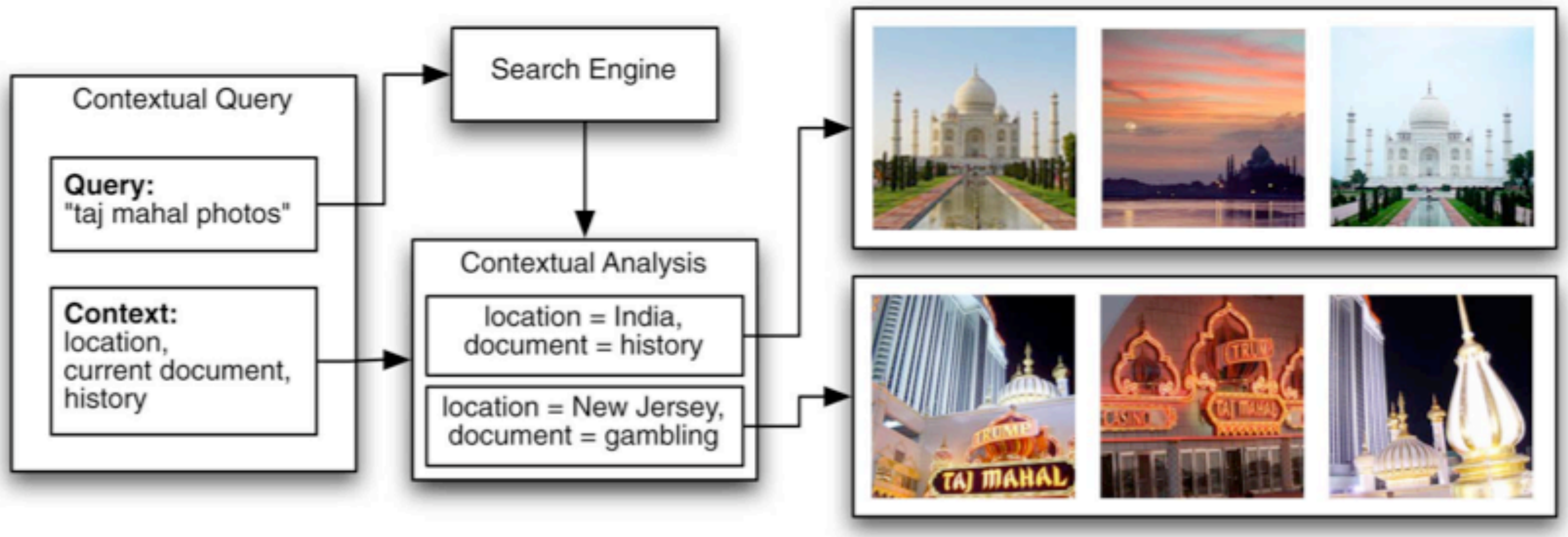


Fig. 5. ***Example of infusing photo queries with contextual awareness from location and task.***

[Kennedy et al. IEEE 2008]

Inferring profiles

- Preprocessing:
  - data cleaning
  - user identification
  - session identification
- Profiling:
  - supervised and unsupervised learning [see Text categorization and Text clustering]

Profile representation

- Category terms:
  - from existing category directory
  - self-defined
- List of URLs
- Natural language terms
- Tensors (queries, clicked pages)

What have we learned?

- Document and query representations:
  - rely upon:
    - assumptions regarding term distributions
    - linguistic knowledge of the language
    - external and contextual knowledge sources
    - machine learning techniques
- Current interest in:
    - **advanced text representations, semantic labeling, information extraction**
    - **personalized and contextualized information needs**

What have we learned?

- Important:
  - *tf* x *idf*
  - relevance feedback and pseudo-relevance feedback

Research questions to be solved

- Generating advanced probabilistic content models from text
- Contextual information can be noisy, imprecise or meaningless:
  - we need mechanisms to detect when contextual cues are important and when they are irrelevant

Further reading

Agichtein, E., Brill, E., Dumais, S. & Ragno, R. (2006). Learning user interaction models for predicting Web search results preferences. In *Proceedings of the Twenty-Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 3- 10). New York: ACM.

Ingwersen, P. & Järvelin, K. (2005). Information retrieval in context. *ACM SIGIR FORUM,* 39 (2), 31-39.

Kelly, D. & Teevan, J. (2003). Implicit feedback for inferring user preference: A bibliography. *SIGIR Forum,* 18-28.

Kennedy, L., Chang, S.-F. & Natsev, A. (2008). Query adaptive fusion for multimodal search. *Proceedings of the IEEE,* 96 (4), 567-588.

Manning, C.D. & Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. Boston, Ma: MIT Press. (p. 23 – 31, p. 152-189)

Moens, M.-F. (2000). *Automatic Indexing and Abstracting of Document Texts* (*The Kluwer International Series on Information Retrieval* 6). Kluwer Academic Publishers: Boston. (p. 77-95; 106-110)

Palmer, D.D. (2000). Tokenization and sentence segmentation. In R. Dale, H. Moisl, & H. Somers, *Handbook of Natural Language Processing* (pp. 11-35). Marcel Dekker.

Salton, G. (1989). *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer.* Reading, MA: Addison-Wesley. (pp. 268-312)