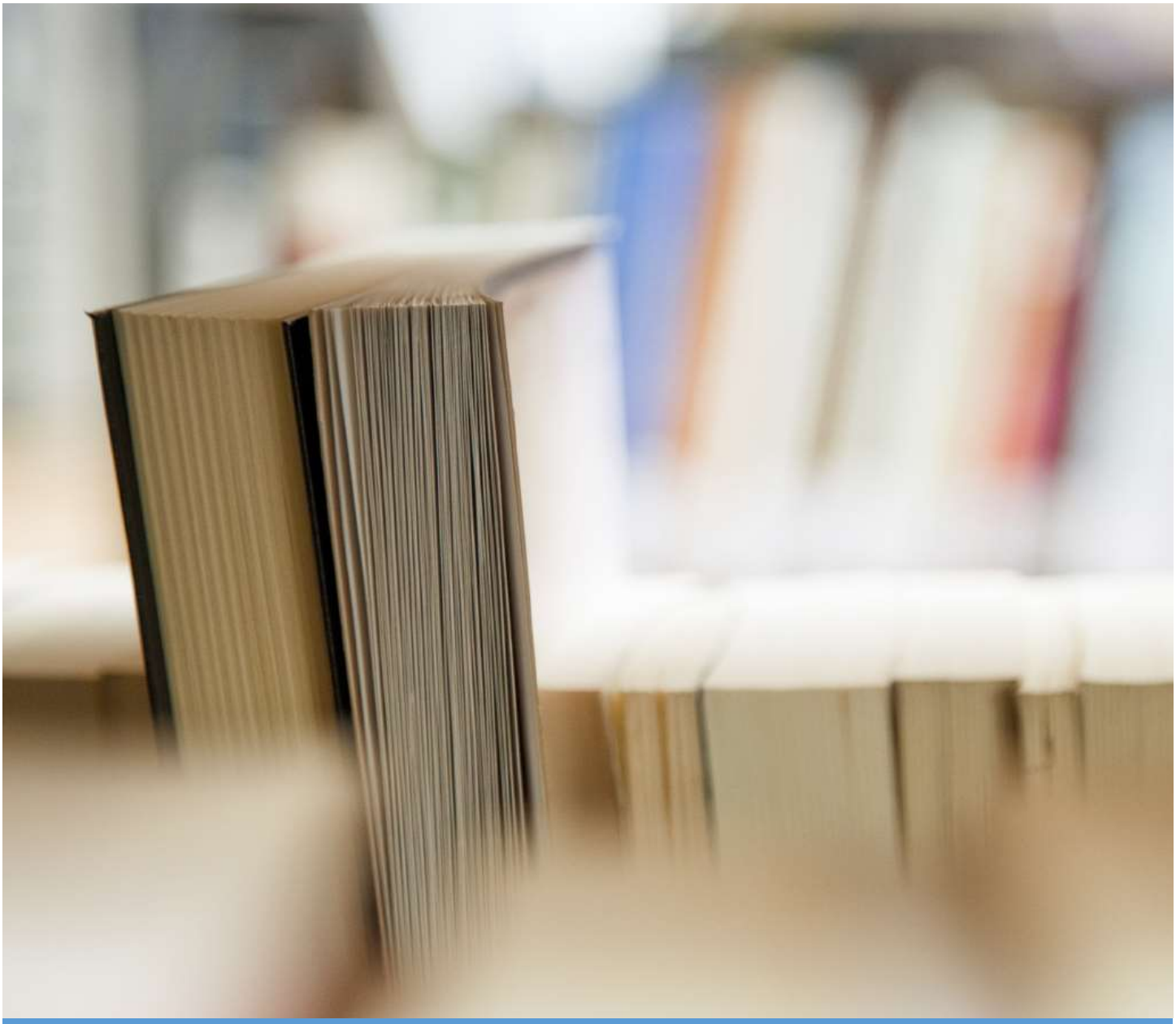

Information Retrieval

Lecture Two

zied othman

2014-2015

[Course title]

LECTURE TWO

THE TERM VOCABULARY AND POSTINGS LISTS

2.1 Document delineation and character sequence decoding

1. Obtaining the character sequence in a document

Digital documents that are the input to an indexing process are typically bytes in a file or on a web server. The first step of processing is to convert this byte sequence into a linear sequence of characters.

The sequence of characters may be encoded by one of various single byte or multibyte encoding schemes, such as Unicode UTF-8, or various national or vendor-specific standards. We need to determine the correct encoding. This can be regarded as a machine learning classification problem, but is often handled by heuristic methods, user selection, or by using provided document metadata.

The idea that text is a linear sequence of characters is also called into question by some writing systems, such as Arabic, where text takes on some two dimensional and mixed order characteristics

2. Choosing a document unit

The next phase is to determine what the *document unit* for indexing is. Thus far we have assumed that documents are fixed units for the purposes of indexing. For example, we take each file in a folder as a document.

The problems with large document units can be alleviated by use of explicit or implicit proximity search, and the tradeoffs in resulting system performance.

2.2 Determining the vocabulary of terms

1. Tokenization

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens, perhaps at the same time throwing away certain characters, such as punctuation. Here is an example of tokenization:

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

A *token* is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing.

A *type* is the class of all tokens containing the same character sequence.

A *term* is a (perhaps normalized) type that is included in the IR system's dictionary.

The set of index terms could be entirely distinct from the tokens, for instance, they could be semantic identifiers in a taxonomy, but in practice in modern IR systems they are strongly related to the tokens in the document. For example, if the document to be indexed is *to sleep perchance to dream*, then there are 5 tokens, but only 4 types.

Issues in tokenization:

- ***Finland's capital*** → ***Finland*** AND ***s***? ***Finlands***? ***Finland's***?
- ***Hewlett-Packard*** → ***Hewlett*** and ***Packard*** as two tokens?
  - ***state-of-the-art***: break up hyphenated sequence.
  - ***co-education***
  - ***lowercase***, ***lower-case***, ***lower case*** ?
- ***San Francisco***: one token or two?
  - How do you decide it is one token?

Numbers

- ***3/20/91*** ***Mar. 12, 1991*** ***20/3/91***
- ***55 B.C.***
- ***B-52***
- ***My PGP key is 324a3df234cb23e***

- ***(800) 234-2333***

Often have embedded spaces. Older IR systems may not index numbers. But often very useful: think about things like looking up error codes/stacktraces on the web. Will often index "meta-data" separately. Creation date, format, etc.

Tokenization: language issues

- French
  - ***L'ensemble*** → one token or two?
    - ***L*** ? ***L'*** ? ***Le*** ?
    - Want ***l'ensemble*** to match with ***un ensemble***
- German noun compounds are not segmented
  - ***Lebensversicherungsgesellschaftsangestellter***
  - 'life insurance company employee'
- Chinese and Japanese have no spaces between words:

  莎拉波娃现在居住在美国东南部的佛罗里达

- Arabic (or Hebrew) is basically written right to left, but with certain items like numbers written left to right
  - Words are separated, but letter forms within a word form complex ligatures

استقلت الجزائر في سنة 1962 بعد 132 عاما من الاحتلال الفرنسي.

← → ← → ← START

'Algeria achieved its independence in 1962 after 132 years of French occupation.'

2. Dropping common terms: stop words

Sometimes, some extremely common words which would appear to be of little value in helping select documents matching a user need are excluded from the vocabulary entirely. These words are called stop words.

The general strategy for determining a stop list is to sort the terms by collection frequency (*the total number of times each term appears in the document collection*), and then to take the most frequent terms, often hand-filtered for their semantic content relative to the domain of the documents being indexed, as a stop list, the members of which are then discarded during indexing.

3. Normalization (equivalence classing of terms)

Having broken up our documents (and also our query) into tokens, the easy case is if tokens in the query just match tokens in the token list of the document. For instance, if you search for USA, you might hope to also match documents containing U.S.A.

Token normalization is the process of canonicalizing tokens so that matches occur despite superficial differences in the character sequences of the tokens.

Normalization to terms may need to "normalize" words in indexed text as well as query words into the same form. a term is a (normalized) word type, which is an entry in our IR system

dictionary. We most commonly implicitly define equivalence classes of terms by, e.g.,

- deleting periods to form a term
  - ***U.S.A., USA*** ⟍ ***USA***
- deleting hyphens to form a term
  - ***anti-discriminatory, antidiscriminatory*** ⟍ ***antidiscriminatory***

Normalization: other languages:

- Accents: e.g., French ***résumé*** vs. ***resume.***
- Umlauts: e.g., German: ***Tuebingen*** vs. ***Tübingen***
  - Should be equivalent

Capitalization/case-folding. A common strategy is to do *case-folding* by reducing all letters to lower case. Often this is a good idea: it will allow instances of *Automobile* at the beginning of a sentence to match with a query of *automobile*. It will also help on a web search engine when most of your users type in *ferrari* when they are interested in a *Ferrari* car.

4. Stemming and lemmatization

For grammatical reasons, documents are going to use different forms of a word, such as *organize*, *organizes*, and *organizing*. Additionally, there are families of derivationally related words with

similar meanings, such as *democracy*, *democratic*, and *democratization*.

The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance:

- *am, are, is* → *be*
- *car, cars, car's, cars'* → *car*
- *the boy's cars are different colors* → *the boy car be different color*

Stemming is a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes.

Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the *lemma*.

The most common algorithm for stemming English, and one that has repeatedly been shown to be empirically very effective, is *Porter's algorithm*.

The entire algorithm is too long and intricate to present here, but we will indicate its general nature. Porter's algorithm consists of 5 phases of word reductions, applied sequentially. Within each phase there are various conventions to select rules, such as selecting the rule from each rule group that applies to the longest suffix. In the first phase, this convention is used with the following rule group:

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